Lecture Notes in Computer Science 4814

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

Ahmed Elgammal Bodo Rosenhahn
Reinhard Klette (Eds.)

Human Motion – Understanding, Modeling, Capture and Animation

Second Workshop, Human Motion 2007
Rio de Janeiro, Brazil, October 20, 2007
Proceedings

Springer

Volume Editors

Ahmed Elgammal
Rutgers State University
Department of Computer Science
110 Frelinghuysen Road, Piscataway, NJ 08854-8019, USA
E-mail: elgammal@cs.rutgers.edu

Bodo Rosenhahn
Max Planck Center for Visual Computing and Communication
Stuhlsatzhausenweg 85, 66123 Saarbrücken, Germany
E-mail: rosenhahn@mpi-sb.mpg.de

Reinhard Klette
The University of Auckland
Computer Science Department
Private Bag 92019, Auckland Mail Center, Auckland 1142, New Zealand
E-mail: r.klette@auckland.ac.nz

Library of Congress Control Number: Applied for

CR Subject Classification (1998): I.2.9, I.2.10, I.4.8, I.7.5, I.3.7

LNCS Sublibrary: SL 6 – Image Processing, Computer Vision, Pattern Recognition, and Graphics

ISSN 0302-9743
ISBN-10 3-540-75702-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-75702-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12175420 06/3180 5 4 3 2 1 0

Preface

This LNCS volume contains the papers presented at the second Workshop on Human Motion Understanding, Modeling, Capture and Animation, which took place on October 20th, 2007, accompanying the 11th IEEE International Conference on Computer Vision in Rio de Janeiro, Brazil.

In total, 38 papers were submitted to this workshop, of which 22 papers were accepted. We were careful to ensure a high standard of quality when selecting the papers. All submissions were double-blind reviewed by at least two experts. Out of the 22 accepted papers, 10 were selected for oral presentation and 12 for posters. We thank the authors of the accepted papers for taking the reviewers' comments into account in the final published versions of their papers. We thank all of the authors who submitted their work, and we trust that the reviewers' comments have been of value for their research activities.

The accepted papers reflect the state of the art in the field and cover various topics related to human motion tracking and analysis. The papers in this volume have been classified into three categories based on the topics they cover: human motion capture and pose estimation, body and limb tracking and segmentation, and activity recognition.

It was a special honor to have Prof. Demetri Terzopulos (University of California, Los Angeles) as the invited speaker at the workshop. We are especially grateful to the members of the Program Committee for their remarkable efforts and the quality of their timely reviews. The organization of this event would not have been possible without the effort and the enthusiasm of several people, and we thank all who contributed.

October 2007

Ahmed Elgammal
Bodo Rosenhahn
Reinhard Klette

Organization

Organizing Committee

Program Chairs
Ahmed Elgammal (Rutgers University, USA)
Bodo Rosenhahn (Max-Planck Institute for Computer Science, Germany)
Reinhard Klette (The University of Auckland, New Zealand)

Program Committee

Michael Black (Brown University, USA)
Richard Bowden (University of Surrey, UK)
Thomas Brox (University of Bonn, Germany)
Stefan Carlsson (KTH Royal Institute of Technology, Sweden)
Larry Davis (The University of Maryland, College Park, USA)
Leo Dorst (University of Amsterdam, The Netherlands)
Pascal Fua (Ecole Polytechnique Fédérale de Lausanne, Switzerland)
David Fleet (University of Toronto, Canada)
Vaclav Hlavac (Czech Technical University, Czech Republic)
Jessica K. Hodgins (Carnegie Mellon University, USA)
Atsushi Imiya (Chiba University, Japan)
Reinhard Koch (Christian-Albrechts-University of Kiel, Germany)
Volker Krüger (Aalborg University, Denmark)
Marcus Magnor (TU Braunschweig, Germany)
Dimitris Metaxas (Rutgers University, USA)
Meinard Müller (University of Bonn, Germany)
Lars Muendermann (Stanford University, USA)
Michael Neff (University of California, Davis, USA)
Ramakant Nevatia (University of Southern California, USA)
Vladimir Pavlovic (Rutgers University, USA)
Fatih Porikli (Mitsubishi Electric Research Laboratories, USA)
Stan Sclaroff (Boston University, USA)
Hans-Peter Seidel (Max-Planck Institute for Computer Science, Germany)
Cristian Sminchisescu (Bonn University, Germany)
Gerald Sommer (Christian-Albrechts-Universität zu Kiel)
Demetri Terzopoulos (University of California, Los Angeles, USA)
Matthias Teschner (University of Freiburg, Germany)
Christian Theobalt (Stanford University, USA)
Matthew Turk (University of California, Santa Barbara, USA)
Jian J. Zhang (Bournemouth University, UK)

Additional Reviewers	Nils Hasler (Max-Planck Institute for Computer Science, Germany) Chan Su Lee (Rutgers University, USA) Christian Schmaltz (Saarland University, Germany) Martin Sunkel (Max-Planck Institute for Computer Science, Germany)
Editorial Assistant	Chan Su Lee (Rutgers University, USA)

Table of Contents

Motion Capture and Pose Estimation

Body and Limb Tracking and Segmentation

Activity Recognition

Marker-Less 3D Feature Tracking for Mesh-Based Human Motion Capture

Edilson de Aguiar[1], Christian Theobalt[2], Carsten Stoll[1], and Hans-Peter Seidel[1]

[1] MPI Informatik, Germany
[2] Stanford University, USA
{edeaguia, stoll, hpseidel}@mpi-inf.mpg.de, theobalt@cs.stanford.edu

Abstract. We present a novel algorithm that robustly tracks 3D trajectories of features on a moving human who has been recorded with multiple video cameras. Our method does so without special markers in the scene and can be used to track subjects wearing everyday apparel. By using the paths of the 3D points as constraints in a fast mesh deformation approach, we can directly animate a static human body scan such that it performs the same motion as the captured subject. Our method can therefore be used to directly animate high quality geometry models from unaltered video data which opens the door to new applications in motion capture, 3D Video and computer animation. Since our method does not require a kinematic skeleton and only employs a handful of feature trajectories to generate lifelike animations with realistic surface deformations, it can also be used to track subjects wearing wide apparel, and even animals. We demonstrate the performance of our approach using several captured real-world sequences, and also validate its accuracy.

1 Introduction

Nowadays, generating realistic and lifelike animated characters from captured real-world motion sequences is still a hard and time-consuming task. Traditionally, marker-based optical motion capture systems [1] reconstruct the motion of a moving subject by measuring the 3D trajectories of optical beacons attached to her body. The optical markers are then mapped to a kinematic skeleton structure [2]. Marker-free methods also exist that are able to measure human motion in terms of a kinematic skeleton without any intrusion into the scene. Thereafter, the model geometry and the skeleton need to be connected such that the surface deforms realistically with the body motion by specifying the influence of each bone on both rigid and non-rigid surface deformation [3].

Stepping directly from a captured real-world sequence to the corresponding realistic moving character is still challenging. Several methods in the literature are able to partly solve this problem. Since marker-based and marker-free motion capture systems measure the motion in terms of a kinematic skeleton, they have to be combined with other scanning technologies to capture the time-varying shape of the human body surface [4,5,6]. However, dealing with people

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 1–15, 2007.

wearing arbitrary clothing from only video streams is still not possible. Time-varying scene representations can also be reconstructed by means of shape-from-silhouette approaches [7], or with combined silhouette- and stereo-based methods [8]. Unfortunately, the measured models often lack detail if only a small number of input camera views is available and it is hard to preserve topological correspondences over time. Researchers have also used physics-based methods to track simple human motions if a kinematic skeleton is available [9]. However, the methods can not be directly applied to objects made of a variety of different materials, and they are not able to track arbitrarily dressed humans completely passively.

Instead, we present a robust skeleton-less approach to automatically capture the motion of a moving human subject and generate plausible and realistic surface deformations from multiple video streams without optical markers. Our algorithm is simple and versatile and enables us to directly animate a high quality static human scan from unaltered video footage which enables potential new applications in motion capture, computer animation and 3D Video.

The main contribution of this paper is a simple and robust method to automatically identify features on a moving human wearing everyday apparel, and track their 3D trajectories. It does not employ any a priori information about the subject, e.g. a kinematic skeleton, and can therefore be straightforwardly applied to other subjects, e.g. animals or mechanical objects. We also present a fast mesh deformation approach that uses only a handful of feature trajectories to directly and realistically animate a static human body scan making it performs the same motion as the captured subject. Our algorithm handles humans wearing arbitrary and sparsely textured clothing. As an additional benefit, it also preserves the mesh's connectivity over time.

The remainder of this paper is structured as follows: Sect. 2 reviews the most relevant related work and Sect. 3 briefly describes our overall framework. Thereafter, Sect. 4 details our automatic approach to identify features and track their 3D trajectories without optical markers. Sect. 5 describes our fast deformation scheme that is used to animate the static human model over the whole sequence according to the constraints derived from the estimated 3D point trajectories. Experiments and results with several captured real-world sequences are shown in Sect. 6, and the paper concludes in Sect.7.

2 Related Work

In our research we capitalize on ideas that have been published in the fields of object tracking, motion capture and scene reconstruction. For the sake of brevity, we refer the interested reader to overview articles on object tracking [10,11]. The following, is by no means a complete list of references from the other two research topics, but merely a summary of the most related categories of approaches.

Human motion is normally measured by marker-based or marker-less optical motion capture systems [1] that parameterize the data in terms of kinematic skeletons. Unfortunately, these approaches can not directly measure time-varying

body shape and they even fail to track people wearing loose apparel. To overcome this limitation, some methods use hundreds of optical markings [5] for deformation capture, combine a motion capture system with a range scanner [4,12] or a shape-from-silhouette approach [6], or jointly use a body and a cloth model to track the person [13]. Although achieving good results, most of these methods require active interference with the scene or require hand-crafted models for each individual.

Alternatively, shape-from-silhouette algorithms [7], multi-view stereo approaches [14], or methods combining silhouette and stereo constraints [8] can be used to reconstruct dynamic scene geometry. To obtain good quality results, however, several cameras are required and it is hard to generate connectivity-preserving dynamic mesh models.

Some passive methods extract 3D correspondences from images to track simple deformable objects [15] or cloth [16]. They can also be employed to jointly capture kinematic motion parameters and surface deformations of tightly dressed humans [17,18]. Researchers have also used physics-based shape models to track textiles [19,20] or simple articulated humans [9]. Unfortunately, none of these methods is able to track people dressed in arbitrary everyday apparel completely passively.

In contrast, we propose a skeleton-less method to directly capture the poses of a moving human subject and generate plausible surface deformations from only a handful of input video streams. This is achieved by first robustly identifying and tracking image features in 3D space. Thereafter, using the 3D trajectories of the features as constraints in a Laplacian mesh editing setting [21], the human model is realistically animated over time. By relying on differential coordinates, plausible shape deformations for the human scan are computed without having to specify explicit material parameters. Our algorithm is simple, robust, easy to implement and works even for moving subjects wearing wide and loose apparel.

3 Overview

An overview of our approach is shown in Fig. 1. Our system expects as input a multi-view video (MVV) sequence that shows the person moving arbitrarily. After acquiring the sequence, silhouette images are calculated via color-based background subtraction and we use the synchronized video streams to extract and track features in 3D space over time.

Our hybrid 3D point tracking framework jointly uses two techniques to estimate the 3D trajectories of the features from unmodified multi-view video streams. First, features in the images are identified using the Scale Invariant Feature Transform (SIFT). Furthermore, SIFT is able to match a feature to its corresponding one from a different camera viewpoint. This allows us to generate a set of pairwise pixel correspondences between different camera views for each time step of input video. Unfortunately, tracking the features over time using only local descriptors is not robust if the human subject is wearing sparsely textured clothing. Therefore, we use a robust dense optical flow method as an additional step to track the features for each camera view separately to fill the

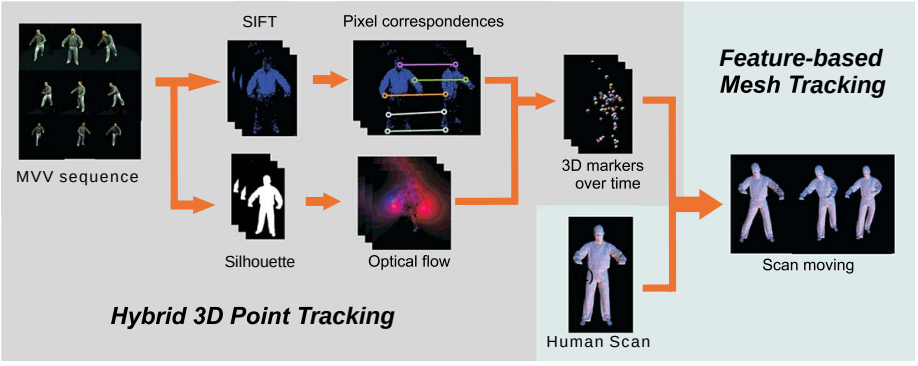


Fig. 1. Overview of our framework: given a multi-view video sequence showing a human performing, our method automatically identifies features and tracks their 3D trajectories. By applying the captured trajectories to a static laser-scan of the subject we are able to realistically animate a human model making it move the same as its real-world counterpart in the video streams.

gaps in the SIFT tracking. By merging both source of information we are able to reconstruct the 3D trajectories for many features over the whole sequence.

Our hybrid technique is able to correctly identify and track many 3D points. In addition to the estimation of 3D point positions, our approach also calculates a confidence value for each estimation. Using confidence-weighted feature trajectories as deformation constraints, our system robustly brings a static laser-scanned triangle mesh M of the subject into life by making it follow the motion of the actor recorded in the video frames.

4 Hybrid 3D Point Tracking

Our hybrid framework jointly employs local descriptors and dense optical flow to identify features and estimate their 3D positions over time from multiple calibrated camera views. In contrast to many other approaches [22,23,24], we developed an automatic tracking algorithm that works directly on the images without any a priori knowledge about the moving subject. It is our goal to create a simple and generic algorithm that can be used to track features on rigid bodies, articulated objects and non-rigidly deforming subjects in the same way.

The input to our algorithm comprises of synchronized video streams recorded from K cameras, each containing N video frames (Fig. 2a). In the first step, we automatically identify L important features, also called keypoints, for each camera view k and time step t and generate a set of local descriptors $F_{k,t} = \{f^0_{k,t}, \ldots, f^L_{k,t}\}$ using SIFT [25], Fig. 2b. We extract these features using the interest point detector proposed by Lowe [26] that is based on local 3D extrema in the scale-space pyramid built with difference-of-Gaussian filters. The local descriptors are built as a distinctive representation of the feature in an image from a patch of pixels in its neighborhood.

Since the SIFT descriptors are invariant to image scale, rotation, change in viewpoints, and change in illumination, they can be used to find corresponding features across different camera views. Given an image $I_{k,t}$, from camera view k and time step t, and the respective set of SIFT descriptors $F_{k,t}$, we try to match each element of $F_{k,t}$ with the set of keypoints from all other camera views. We use a matching function similar to [25], which assigns a match between $f^i_{k,t}$ and a keypoint in $F_{j,t}$ if the Euclidean distance between their invariant descriptor vectors is minimum. In order to discard false correspondences, nearest neighbor distance ratio matching is used with a threshold T_{MATCH} [27] .

After matching the keypoints across all K camera views at individual time steps, we gather all R correct pairwise matches into a list of pixel correspondences $C_t = \{c^0_t, \ldots, c^R_t\}$ by using all reliable matches found for each time step t (Fig. 2c). Each element $c^r_t = ((cam_u, P^i_t), (cam_v, P^j_t))$ stores the information about a correspondence between two different camera views, i.e. that pixel P^i_t in camera cam_u corresponds to pixel P^j_t in camera view cam_v at time t.

Unfortunately, tracking the features over time using only the list of correspondences C and connecting their elements at different time steps is not robust, because it is very unlikely that the same feature will be found at all time instants. This is specially true if the captured images show subjects performing fast movements, where features can be occluded for a long period of time, or when the subject wears everyday apparel with sparse texture. In the latter case, SIFT only detects a small number of keypoints per time step, which is usually not enough for tracking articulated objects. Therefore, in order to robustly reconstruct the 3D trajectories for the features we decided to use optical flow to track both elements of all c^r_t for each camera view separately, i.e. the pixel P^i_t is tracked using camera view cam_u and the pixel P^j_t using camera view cam_v.

The 2D flow-based tracking method works as follows: for each camera view k, we track all pixels over time using the warping-based method for dense optical flow proposed by Brox et al. [28]. After calculating the optical flow $\boldsymbol{o^t_k}(I_{k,t}, I_{k,t+1})$ between time step t and $t+1$ for camera k, we use $\boldsymbol{o^t_k}$ to warp the image $I_{k,t}$ and we verify for each pixel in the warped image if it matches the corresponding pixel in $I_{k,t+1}$. We eliminate the pixels that do not have a partner in $t+1$ and the pixels that belong to the background by comparing the warped pixels with the pre-computed silhouette $SIL_{k,t+1}$. This process is repeated for all consecutive time steps and for all camera views. As a result, we construct a tracking list $D_k = \{E^0, \ldots, E^g\}$ with G pixel trajectories for each camera view k (Fig. 2d). Each element $E^i = \{P^i_0, \ldots, P^i_N\}$ contains the positions of the pixel P^i_t for all time steps t.

The last step of our hybrid tracking scheme merges the optical flow tracking information with the list of correspondences to reconstruct the 3D trajectories for all features. We take pixel correspondences from all time steps into account. For instance, if a matching c^r_t is detected by SIFT only at the end of the sequence we are still able to recover the anterior positions of the feature by using the optical flow information.

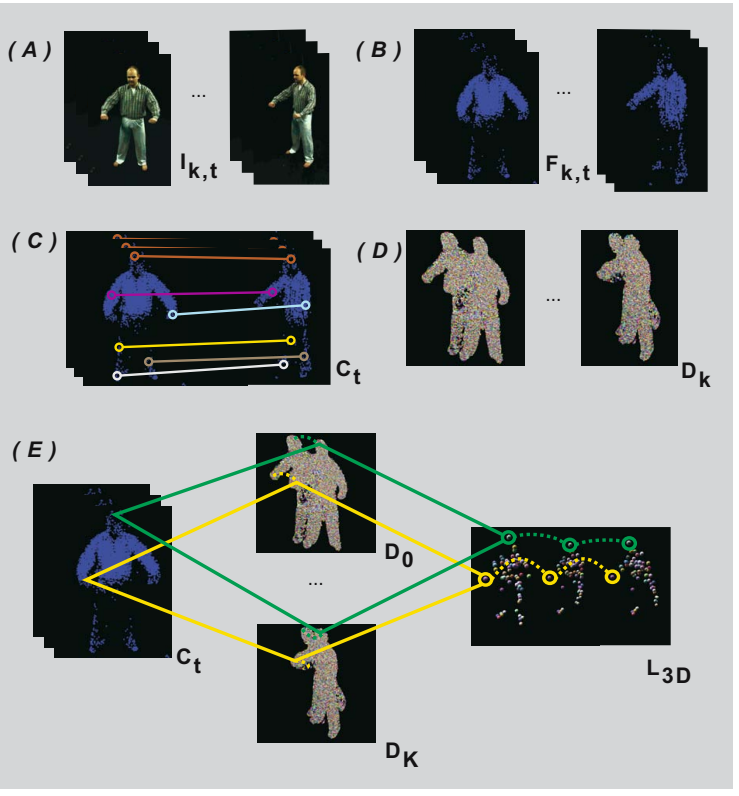


Fig. 2. Using the synchronized video streams as input (A), our hybrid approach first identifies features in the images using SIFT (B) and then matches these features between different pairs of camera views based on their descriptors (C). In addition, we track these features for each camera view separately using optical flow (D). At the end, reliable 3D trajectories for the features are reconstructed by merging both information (E).

For each entry $c_t^r = ((cam_u, P_t^i), (cam_v, P_t^j))$, we verify if the pixel P_t^i is found in D_{cam_u} and if the pixel P_t^j is found in D_{cam_v}. In case both elements are found, we estimate the position of the respective 3D point, $mm_r(t)$, for the whole sequence (Fig. 2e), otherwise c_t^r is discarded. The 3D positions are estimated by triangulating the viewing rays that start at the camera views cam_u and cam_v and pass through the respective image plane pixel at P_t^i and P_t^j. However, due to inaccuracies, these rays will not intersect exactly at a single point. However, we can compute a pseudo-intersection point $pos_t^r = \{x, y, z\}$ that minimizes the sum of squared distance to each pointing ray. We also use the inverse of this distance, cv_r, as a confidence measure indicating how reliable a particular feature has been located. If cv_r is below a threshold T_{CONF} we discard it, since it indicates that c_t^r assigns a wrong pixel correspondence between two different camera views.

We also discard a trajectory mm_r if it does not project into the silhouettes in all camera views and at all time steps. This way, we can prevent the use of 3D points whose trajectories degenerate over time as deformation constraints. We assess silhouette-consistency using the following measure:

$$TSIL(mm_r) = \sum_{t=0}^{N} \sum_{k=0}^{K} PROJ_{sil}^{k}(pos_t^r, t) \quad (1)$$

where $PROJ_{sil}^{k}(pos_t^r, t)$ is a function that evaluates to 1 if $mm_r(t)$ projects inside the silhouette image of camera view k at time step t, and it is 0 otherwise. We only consider mm_r a reliable 3D trajectory if $TSIL(mm_r) > TR_{SIL}$. Appropriate values for the thresholds are found through experiments.

After processing all elements of C for all time steps, we generate a list with reliable 3D trajectories for the features. The list $L_{3D} = \{mm_0, \ldots, mm_h\} = \{(LP_0, LE_0), \ldots, (LP_H, LE_H)\}$ assigns to each trajectory mm_i, a tuple (LP_i, LE_i) containing the 3D point positions, $LP_i = \{pos_0^i, \ldots, pos_N^i\}$, and the respective list of confidence values for each estimated 3D position, $LE_i = \{cv_0^i, \ldots, cv_N^i\}$. As shown in Sect. 6, our hybrid approach is able to identify and accurately track many 3D points for sequences where the human subject is performing fast motion, even when he is dressed in everyday apparel.

5 Feature-Based Laplacian Mesh Tracking

It is our goal to animate the human scan making it move the same way as its real-world counterpart in the video streams by using the reconstructed 3D point trajectories as motion constraints. For this purpose, we first roughly align the human model with the 3D point positions at the first time step of video (our reference), Fig. 3(a). This is automatically done by applying a PCA-based alignment scheme to a reconstructed volumetric shape-from-silhouette model of the moving subject. Thereafter, we select H target vertices $V_T = \{v_{Th} | h \in \{0 \cdots H\}\}$ in the human model M by choosing vertices that are closest to the 3D point positions at the reference time step, Fig. 3(b). These target vertices V_T are used to guide the mesh deformation method.

We deform the human scan by employing a Laplacian mesh deformation scheme that jointly uses rotational and positional constraints on the target vertices V_T in a similar way as [29]. The details of the human model M are encoded in its differential coordinates. The differential coordinates d of M are computed once at the beginning of the sequence by solving a matrix multiplication of the form $d = Lv$, where L is the discrete Laplace operator based on the cotangent-weights, and v is the vector of M's vertex coordinates [30]. Thereafter we perform the following three processing steps for each time step t:

Since the differential coordinates d are rotation-dependent [31], we need to first calculate the local rotations that should be applied to d. We derive these rotational constraints from the 3D trajectories. The local rotation for each target vertex v_{Th} of M is calculated from the rotation of the corresponding 3D point

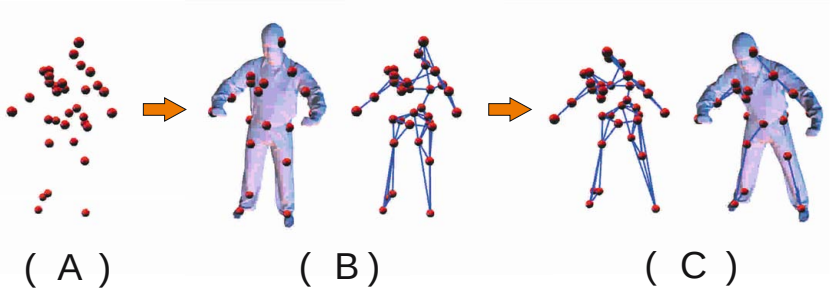


Fig. 3. After aligning the 3D point positions (A) with the human model at the reference time step (B), our method reconstructs a novel pose jointly using rotational and positional constraints on the target vertices which we derived from the 3D feature trajectories (C)

$mm_h(t)$ between reference time and time t by means of a graph-based method, Fig. 3. To this end, 3D points are considered as nodes in a graph, and edges between them are determined by constructing the minimal spanning tree [32] using the approximated geodesic distance as edge weights. For each 3D point $mm_h(t)$, we find the minimal rotation that makes its outgoing edges at the reference time match its outgoing edges at time t (i.e. using the Jacobian). This local rotation is then converted into a quaternion $q_{mm_h(t)}$. Since we want the target vertices V_T to perform the same rotations as the 3D points, we set $q_{v_{Th}} = q_{mm_h(t)}$ for all H 3D points.

Using the estimated rotations for the target vertices, we interpolate them over M using the idea proposed in [33] in order to estimate rotations for each vertex of the model. Each component of a quaternion $q = [w, q_1, q_2, q_3]$ is regarded as a scalar field defined over the entire mesh. A smooth interpolation is guaranteed by regarding these scalar fields as harmonic fields. The interpolation is performed by solving the Laplacian equation $Lq = 0$ over the whole mesh using the constrained target vertices as Dirichlet boundary conditions and normalizing the resulting quaternions.

At the end, we reconstruct the vertex positions v of M such that the mesh best approximates the rotated differential coordinates, as well as the positional constraints. This can be formulated as a least-squares problem of the form

$$\underset{v}{argmin}\{\|Lx - (q \cdot d \cdot \overline{q})\|^2 + \|Av - p\|^2\}. \tag{2}$$

which can be transformed into a linear system

$$(L^T L + A^T A)v = L^T(q \cdot d \cdot \overline{q}) + A^T p. \tag{3}$$

In Eq. 3, p is the vector of positional constraints of the form $v_j = pos_t^j$, $j \in \{1, \ldots, H\}$ specified for the H target vertices and derived from the position of the 3D points at time t. The matrix A is a diagonal matrix containing non-zero weights $A_{ij} = c * cv_t^j$, c being a constant, only for constrained vertices j. We

weight the target vertex position pos_t^j for v_{Tj} at time t proportional w.r.t its corresponding confidence value since small values for cv_t^j indicate inaccuracies in the estimated 3D position. As demonstrated in Sect. 6, this weighting scheme leads to a better visual animation quality for the animated human scans.

After applying this algorithm to the whole sequence, our mesh deformation approach is able to animate the human scan making it correctly follow the motion of the actor recorded in all video frames. As shown in Sect. 6, our approach preserves the details and features of the mesh and is able to generate plausible and realistic surface deformation for subjects wearing even loose everyday apparel.

6 Results and Discussion

We tested our method on several real-world sequences with different male and female test subjects recorded in our studio. Our acquisition procedure works as follows: we first acquire the scanned model with a Vitus Smart$^{\text{TM}}$ full body laser scanner. After scanning, the subject immediately moves to the nearby area where she is recorded with eight synchronized video cameras that run at 25 fps and provide 1004x1004 pixels frame resolution. The calibrated cameras are placed in an approximately circular arrangement around the center of the scene and color-consistency across cameras is ensured by applying a color-space transformation to each camera stream. The captured video sequences are between 150 and 400 frames long and show a variety of different clothing styles. We captured different motions ranging from simple walking to yoga and capoeira moves.

As shown in the second and third columns of Table 1, our hybrid 3D point tracking approach is able to identify and track many features in 3D space accurately. The average confidence value (CV) for the 3D point positions are large, which corresponds to position errors of around $1.0-2.2cm$. Three different frames for the yoga (YOGA) and walking (WALK) sequences with selected features shown as dots can be seen in the upper row of Fig. 4. Looking at the temporal evolution one can see that the features are reliably tracked over time. The left images in the middle row of Fig. 4 also show two closeups on the legs in the walking sequence. Features were accurately tracked despite the appearance ambiguities caused by the trousers with homogeneous color. If we had used only SIFT descriptors to track these features, it would have been impossible to track them in these homogeneous areas.

High tracking accuracy and reliability even in such difficult situations is upheld by additionally taking into account optical flow information. Even if a correspondence was only found for one time step, we can reconstruct the complete trajectory for this feature by looking at the optical flow information. This second source of information also enables us to apply a very high threshold T_{MATCH} which eliminates unreliable 3D feature matches already at an early stage.

Before using the 3D point trajectories to guide the deformation of the human scan we first choose a subset of N_M points from the initial set of 3D trajectories L_{3D} at the reference time step. This subset of points should be distributed

Fig. 4. (Upper row) Selected features tracked in three different frames for the yoga and walking sequences; (middle and lower row) two frames of the walking sequence in detail and side-by-side comparisons between input video frames and reconstructed poses for the human scan. Our framework correctly tracks 3D trajectories of features even in the presence of occlusions or appearance ambiguities. By combining the 3D point trajectories with our mesh deformation method, our algorithm is able to directly animate a human body scan.

evenly on the model surface. This is done by ramdonly choosing a element in L_{3D} and all adjacent points next to it at the reference time step by using a distance threshold T_{DIST}. We compare the confidence values for this group of elements and choose the point with the maximum confidence value. We continue the same procedure choosing another point in L_{3D} until all selected points are separated by a distance T_{DIST}, and consequently distributed over the model's surface. We conducted several experiments with different values for T_{DIST} and found out that in general, values between $10cm$ and $20cm$ produce best results. For a typical sequence, this leads to around $20 - 50$ selected points. Note that although our hybrid 3D tracking approach is able to correctly track many more points over time, even a subset of points is sufficient to track body poses reliably.

Fig. 5. Overlap between the reprojected model (red) and the input image for the female and male subjects. Our framework is able to correctly reconstruct their pose even when they are wearing wide and loose apparel.

Our selection criteria also enable us to eliminate multiple trajectories of the same feature (stemming for different camera pairs) which bears no useful information.

The middle (right) and lower row of Fig. 4 shows several side-by-side comparisons between input video frames and tracked poses of the human scan. Our algorithm reliably recovers the poses and creates plausible and realistic surface deformations for the male actor performing a capoeira move and even for the female subject wearing a long dress. Due to the occlusion of the limbs or the wide and loose apparel, tracking the motion of these subjects over time would have been hard with a normal motion capture system. More captured real-world results are shown in the accompanying video.

Due to the lack of ground truth for our experiments, we evaluate our results by overlapping the reprojected model with the input images as shown in Fig. 5. We also calculate a multi-view overlap measure by counting the average number of pixels that do not match between the reprojected model and the input image silhouettes for all camera views and all time steps. As shown in the plot in Fig. 6, our system automatically animates the human scan making it follow the motion of the real-world actor with a consistent silhouette-accuracy of more than 94%.

We also performed experiments to evaluate the performance of our framework in animating the human scan. Table 1 summarizes the results we have obtained employing quality and accuracy measures for several sequences. The column shows the average volume change in the animated scan over the whole sequence. This measure is a numerical indicator for implausible deformations. The preservation of mesh quality is analyzed by looking at the average distortion of the triangles, . It is computed by averaging the per-triangle Frobenius norm over the mesh and over time [34]. This norm is 0 for an equilateral triangle and approaches infinity with increasing degeneracy. Finally, the column labeled contains the average multi-view overlap between the reprojected model and the input image silhouettes over time.

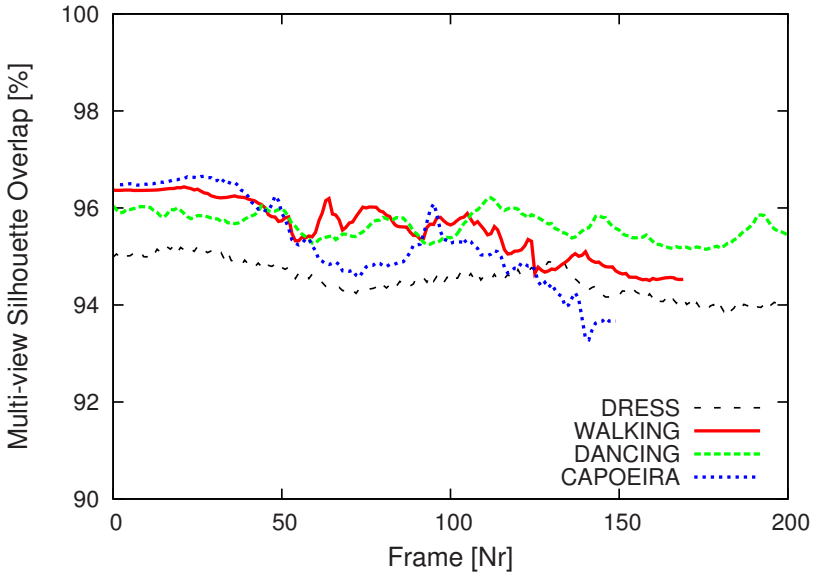


Fig. 6. Multi-view silhouette overlap for several captured sequences. Our system automatically makes the static human scan follow the motion of the captured real-world actor with high precision.

Table 1 shows that the volume change in the animated human scan is in the range of normal non-rigid body deformations, and that triangles remain in nice shape. It also shows that our mesh deformation approach reconstructs the poses of the scan with high accuracy, even if the subjects wear wide and loose everyday apparel.

We performed experiments to demonstrate the importance of the confidence value as a weight in Eq. 3 (Sect. 5) as well. Our experiments show that when using the confidence value in our mesh deformation approach, surface deformations are

Table 1. For each captured real-world sequence, the number of identified features (FEAT) and the average confidence value (CV) are shown. We also employ accuracy and quality measures for the animated scan, i.e. changes in volume (VOL), distortion of triangles (QLT) and multi-view silhouette overlap (OVL), to demonstrate the performance of our framework.

SEQ	FEAT	CV $[m^{-1}]$	OVLP	VOL	QLT
CAPO	1207	65.18	95.4%	3.2%	0.03
DANC	1232	58.30	95.6%	1.8%	0.01
YOGA	1457	112.23	93.7%	3.6%	0.10
WALK	2920	71.78	95.5%	1.5%	0.01
DRSS	3132	45.72	94.4%	2.0%	0.01

generated in a more reasonable and lifelike way, which leads to a better visual animation quality.

Our results show that our purely passive tracking method can automatically identify and track the 3D trajectories of features on a moving subject without the need of any a priori information or optical markers. By combining it with our fast deformation technique it also enables us to directly and realistically animate a static human scan making it follow the same motion as its real-world counterpart even if he wears casual everyday apparel.

Nonetheless, our algorithm is subject to a few limitations. Currently, if the subject moves very quickly, the optical flow method may fail to track the 2D features. However, in such situations one might use one of the many high-speed camera models available today for capturing fast scenes. Another limitation is the run time of our tracking system. Currently, we need around 3-5 minutes per multi-view frame on a Pentium IV with 3GHz, with more than 90% of the time spent for the SIFT and optical flow calculations. We are planning to investigate the use of lower image resolutions for tracking without compromising the overall tracking accuracy. Our fast mesh deformation approach, on the other hand, can generate animations at 5 fps for models comprising of $20k$ to $50k$ triangles.

Another problem is that our mesh deformation method does not handle volume constraints [35]. In starkly under-constrained settings such a constraint would prevent inaccurate deformations, however at the cost of slower runtimes. Also, in some situations, e.g. very wide apparel, a volume constraint might even prevent correct deformations. Finally, although our algorithm correctly captures the body deformations at a coarse scale, the deformations of subtle details, such as small wrinkles, are not captured. We are planning to extend our method in the future to also capture these details by means of a multi-view stereo algorithm.

Despite these limitations our automatic method is a simple, flexible, easy to implement and reliable purely passive method to robustly track 3D trajectories of features on a moving human and even other subjects. These features can then be used to animate a static human body scan making it perform the same motion as the captured subject recorded from only a handful of cameras.

7 Conclusion

We have presented a new skeleton-less approach to automatically identify features and track them on a moving subject who has been recorded with only eight video cameras. Our algorithm does not require optical markings, does not need a priori information about the tracked subject, and behaves robustly even for humans wearing sparsely textured and wide apparel. By applying the captured feature trajectories as constraints in a fast mesh deformation approach, we can make a high-quality human scan move and deform in the same way as its real-world counterpart in the input video footage. We expect that our new mesh-based paradigm will pave the trail for many new applications in motion capture in general, 3D Video and character animation.

Acknowledgments. This work is supported by EC within FP6 under Grant 511568 with the acronym 3DTV and AIM@SHAPE, a Network of Excelence project (506766) within EU's Sixth Framework Programme. We would like to thank Thomas Brox and David Lowe for letting us use their code.

References

1. Moeslund, T.B., Granum, E.: A survey of computer vision-based human motion capture. CVIU 81(3), 231–268 (2001)
2. Silaghi, M.C., Plänkers, R., Boulic, R., Fua, P., Thalmann, D.: Local and global skeleton fitting techniques for optical motion capture. In: Magnenat-Thalmann, N., Thalmann, D. (eds.) CAPTECH 1998. LNCS (LNAI), vol. 1537, pp. 26–40. Springer, Heidelberg (1998)
3. Lewis, J.P., Cordner, M., Fong, N.: Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. ACM Trans. Graph, 165–172 (2000)
4. Allen, B., Curless, B., Popović, Z.: Articulated body deformation from range scan data. ACM Trans. Graph (SIGGRAPH 2002), 612–619 (2002)
5. Park, S.I., Hodgins, J.K.: Capturing and animating skin deformation in human motion. ACM Trans. Graph (SIGGRAPH 2006) 25(3) (2006)
6. Sand, P., McMillan, L., Popovic, J.: Continuous capture of skin deformation. ACM Trans. Graph. 22(3), 578–586 (2003)
7. Goldluecke, B., Magnor, M.: Space-time isosurface evolution for temporally coherent 3d reconstruction. In: CVPR 2004, vol. 1, pp. 350–355 (2004)
8. Furukawa, Y., Ponce, J.: Carved visual hulls for image-based modeling. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 564–577. Springer, Heidelberg (2006)
9. Metaxas, D., Terzopoulos, D.: Shape and nonrigid motion estimation through physics-based synthesis. IEEE Trans. Pattern Anal. Mach. Intell. 15(6), 580–591 (1993)
10. Lepetit, V., Fua, P.: Monocular model-based 3d tracking of rigid objects. Found. Trends. Comput. Graph. Vis. 1(1), 1–89 (2006)
11. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Comput. Surv. 38(4), 13 (2006)
12. Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J., Davis, J.: Scape: shape completion and animation of people. ACM Trans. Graph. 24(3), 408–416 (2005)
13. Rosenhahn, B., Kersting, U., Powel, K., Seidel, H.P.: Cloth x-ray: Mocap of people wearing textiles. In: DAGM, pp. 495–504 (2006)
14. Zitnick, C.L., Kang, S.B., Uyttendaele, M., Winder, S., Szeliski, R.: High-quality video view interpolation using a layered representation. ACM Trans. Graph. 23(3), 600–608 (2004)
15. Decarlo, D., Metaxas, D.: Optical flow constraints on deformable models with applications to face tracking. Int. J. Comput. Vision 38(2), 99–127 (2000)
16. Pritchard, D., Heidrich, W.: Cloth motion capture. Eurographics, 263–271 (September 2003)
17. de Aguiar, E., Theobalt, C., Magnor, M., Seidel, H.P.: Reconstructing human shape and motion from multi-view video. In: CVMP 2005, pp. 42–49 (2005)

18. Plänkers, R., Fua, P.: Articulated soft objects for multiview shape and motion capture. IEEE Trans. Pattern Anal. Mach. Intell. 25(9), 1182–1187 (2003)
19. Hasler, N., Asbach, M., Rosenhahn, B., Ohm, J.R., Seidel, H.P.: Physically based tracking of cloth. In: Proc of VMV 2006, Aachen, Germany, pp. 49–56 (2006)
20. Salzmann, M., Ilic, S., Fua, P.: Physically valid shape parameterization for monocular 3-d deformable surface tracking. In: British Machine Vision Conference (2005)
21. Sorkine, O.: Differential representations for mesh processing. Computer Graphics Forum 25(4) (2006)
22. Balan, A.O., Black, M.J.: An adaptive appearance model approach for model-based articulated object tracking. In: Proc. of CVPR 2006, pp. 758–765. IEEE Computer Society Press, Los Alamitos (2006)
23. Brox, T., Rosenhahn, B., Cremers, D., Seidel, H.P.: High accuracy optical flow serves 3-d pose tracking: Exploiting contour and flow based constraints. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 98–111. Springer, Heidelberg (2006)
24. Kehl, R., Gool, L.V.: Markerless tracking of complex human motions from multiple views. Comput. Vis. Image Underst. 104(2), 190–209 (2006)
25. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 20, 91–110 (2004)
26. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proc. of ICCV, pp. 1150–1157 (1999)
27. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors (2003)
28. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: Pajdla, T., Matas, J. (eds.) ECCV 2004. LNCS, vol. 3024, pp. 25–36. Springer, Heidelberg (2004)
29. de Aguiar, E., Theobalt, C., Stoll, C., Seidel, H.P.: Rapid animation of laser-scanned humans. In: IEEE Virtual Reality 2007, pp. 223–226. IEEE Computer Society Press, Los Alamitos (2007)
30. Lipman, Y., Sorkine, O., Cohen-Or, D., Levin, D., Rössl, C., Seidel, H.P.: Differential coordinates for interactive mesh editing. In: SMI 2004, pp. 181–190 (2004)
31. Stoll, C., Karni, Z., Rössl, C., Yamauchi, H., Seidel, H.P.: Template deformation for point cloud fitting. In: Symposium on Point-Based Graphics, pp. 27–35 (2006)
32. Kruskal, J.B.: On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. of the American Mathematical Society 7, 48–50 (1956)
33. Zayer, R., Rössl, C., Karni, Z., Seidel, H.P.: Harmonic guidance for surface deformation. In: Proc. of Eurographics 2005, vol. 24, pp. 601–609 (2005)
34. Pebay, P.P., Baker, T.J.: A comparison of triangle quality measures. In: Proc. of the 10th International Meshing Roundtable, pp. 327–340 (2001)
35. Huang, J., Shi, X., Liu, X., Zhou, K., Wei, L.Y., Teng, S.H., Bao, H., Guo, B., Shum, H.Y.: Subspace gradient domain mesh deformation. ACM Trans. Graph. 25(3), 1126–1134 (2006)

Boosted Multiple Deformable Trees for Parsing Human Poses

Yang Wang and Greg Mori

School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
{ywang12,mori}@cs.sfu.ca

Abstract. Tree-structured models have been widely used for human pose estimation, in either 2D or 3D. While such models allow efficient learning and inference, they fail to capture additional dependencies between body parts, other than kinematic constraints. In this paper, we consider the use of multiple tree models, rather than a single tree model for human pose estimation. Our model can alleviate the limitations of a single tree-structured model by combining information provided across different tree models. The parameters of each individual tree model are trained via standard learning algorithms in a single tree-structured model. Different tree models are combined in a discriminative fashion by a boosting procedure. We present experimental results showing the improvement of our model over previous approaches on a very challenging dataset.

1 Introduction

Estimating human body poses from still images is arguably one of the most difficult object recognition problems in computer vision. The difficulties of this problem are manifold – humans are articulated objects, and can bend and contort their bodies into a wide variety of poses; the parts which make up a human figure are varied in appearance (due to clothing), which makes them difficult to reliably detect; and parts often have small support in the image or are occluded. In order to reliably interpret still images of human figures, it is likely that multiple cues relating different parts of the figure will need to be exploited.

Many existing approaches to this problem model the human body as a combination of rigid parts, connected together in some fashion. The typical configuration constraints used are kinematic constraints between adjacent parts, such as torso-upper half-limb connection, or upper-lower half-limb connection (e.g. Fig. 1). This set of constraints has a distinct computational advantage – since the constraints form a tree-structured model, inferring the optimal pose of the person using this model is tractable.

However, this computational advantage comes at a cost. Simply put, the single tree model does not adequately model the full set of relationships between parts

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 16–27, 2007.

of the body. Relationships between parts not connected in the kinematic tree cannot be directly captured by this model.

In this paper, we develop a framework for modeling human figures as a collection of trees. We argue that this framework has the advantage of being able to locally capture constraints between the parts which constitute the model. With a collection of trees, a global set of constraints can be modeled. In our work, these constraints are spatial constraints, but this framework could be extended to other cues (e.g. color consistency, occlusion relationships). We demonstrate that the computational advantages of tree-structured models can be kept, and provide tractable algorithms for learning and inference in these multiple tree models.

The rest of this paper is organized as follows. Section 2 reviews previous work. Section 3 gives the details of our approach. Section 4 shows some experimental results. Section 5 concludes this paper and points to some future work.

2 Related Work

One of the earliest lines of research related to finding people from images is in the setting of detecting and tracking pedestrians. Starting with the work of Hogg [4], there have been a lot work done in tracking with kinematic models in both 2D and 3D. Forsyth et al. [3] provide a survey of this work.

Some of the these approaches are exemplar-based. For example, Toyama & Blake [26] use 2D exemplars for people tracking. Mori & Malik [13] and Sullivan & Carlsson [24] address the pose estimation problems as 2D template matching using pre-stored exemplars upon which joint locations have been marked. In order to deal with the complexity due to variations of pose and clothing, Shakhnarovich et al. [19] adopt a brute-force search, using a variant of locality sensitive hashing for speed. Exemplar-based models are effective when dealing with regular human poses. However, they cannot handle those poses that rarely occur. See Fig. 5 for some examples.

There are many approaches which explicitly model the human body as an assembly of parts. Ju et al. [7] introduce a "cardboard people" model, where body parts are represented by a set of connected planar patches. Felzenszwalb & Huttenlocher [2] develop a tree-structured model called pictorial structure (PS) and applied it to 2D human pose estimation. Lee & Cohen [10] present results on 3D pose estimation from a single image based on proposal maps, using skin and face detection as extra cues to guide the MCMC sampling of 3D models. Ramaman & Forsyth [16] describe a self-starting tracker that tracks people by building an appearance model from a stylized pose detected by a top-down PS method. Sudderth et al. [23] introduce a non-parametric belief propagation method with occlusion reasoning for hand tracking. Sigal & Black [20] use a similar idea for pose estimation. Ren et al. [18] use bottom-up detections of parallel lines as part hypotheses, and combine these hypotheses with various pairwise part constraints via an integer quadratic programming. Hua et al. [5] use bottom-up cues such as skin/face detection to guide a belief propagation

inference algorithm. There is also some work on using segmentation as a preprocessing step [12,14,22].

Our work is closely related to some recent work on learning discriminative models for localization. Ramanan & Sminchisescu [17] use a variant of conditional random fields (CRF) [8] for training localization models for articulated objects, such as human figures, horses, etc. Ramanan [15] extends their work by iteratively building a region model based on color cues.

Our work is also related to boosting on structured outputs. Boosting was originally proposed for classification problems. Recently people have adopted it for various tasks where the outputs have certain structures (e.g., chains, trees, graphs). For example, Torralba et al. [25] use boosted random fields for object detection with contextual information. Truyen et al. [27] use a boosting algorithm on Markov Random Fields for multilevel activity recognition.

Another line of research related to our work is on various extensions of tree models in both the computer vision and the machine learning literature. Song et al. [21] detect corner features in video sequences and model them using a decomposable triangulated graph, where the graph structure is found by a greedy search. Ioffe & Forsyth [6] propose a sampling method based on body part candidates found by a rectangle detector. Meila & Jordan [11] propose "mixtures-of-trees" that combine multiple tree models. The parameters of such models are learned by an EM algorithm in either maximum likelihood or Bayesian framework. We would like to point out that although our work seems similar to "mixtures-of-trees", there are some important differences. Instead of using the maximum likelihood criterion, our method optimizes a loss function that is directly tied to inference. And our model is learned by an efficient boosting procedure.

3 Our Approach

Our method is a combination of tree-structured deformable models for human pose estimation [15,17] and boosting on MRFs [27]. The basic idea is to model a human figure as a weighted combination of several tree-structured deformable models. The parameters of each tree model, and the weights of different trees are learned from training data in a discriminative fashion using boosting. In this section, we first review deformable models for human pose estimation (Sect. 3.1), followed by the learning and inference algorithms in such models (Sect. 3.2). Then we introduce the boosted multiple trees (Sect. 3.3).

3.1 Deformable Model

Consider a human body model with K parts, where each part is represented by an oriented rectangle with fixed size. We can construct an undirected graph $G = (V, E)$ to represent the K parts. Each part is represented by a vertex $v_i \in V$ in G, and there exists an undirected edge $e_{ij} = (v_i, v_j) \in E$ between vertices v_i and v_j if v_i and v_j has a spatial dependency. Let $l_i = (x_i, y_i, \theta_i)$ be a random

variable encoding the image position and orientation of the i-th part. We denote the configuration of the K part model as $L = (l_1, l_2, ..., l_K)$. Given the model parameters Θ, the conditional probability of L in an image I can be written as:

$$Pr(L|I,\Theta) \propto \exp\left(\sum_{(i,j)\in E} \psi(l_i - l_j) + \sum_{i=1}^{K} \phi(l_i)\right) \tag{1}$$

$\psi(l_i - l_j)$ corresponds to a spatial prior on the part geometry, and $\phi(l_i)$ models the local image evidence at each part located at l_i. Most previous approaches use Gaussian shape priors $\psi(l_i - l_j) \propto \mathcal{N}(l_i - l_j; \mu_i, \Sigma_i)$ [2,17]. However, since we are dealing with images with a wide range of poses and aspects, Gaussian shape priors seem too rigid. Instead we choose a spatial prior using discrete binning (Fig. 2) similar to the one used in Ramanan [15]:

$$\psi(l_i - l_j) = \alpha_i^T \text{bin}(l_i - l_j) \tag{2}$$

α_i is a parameter that favors certain relative spatial and angular bins for part i with respect to its parent j. This spatial prior captures more intricate distributions than a Gaussian prior.

For the appearance model $\phi(l_i)$, we follow the one used in Ramanan [15]. $\phi(l_i)$ corresponds to the local image evidence for a part and is defined as:

$$\phi(l_i) = \beta_i^T f_i(I(l_i)) \tag{3}$$

$f_i(I(l_i))$ is the part-specific feature vector extracted from the oriented image patch at location l_i. We use a binary vector of edges for all parts. β_i is a part-specific parameter that favors certain edge patterns for an oriented rectangle patch $I(l_i)$ in image I, where l_i defines the location and orientation of the patch.

To facilitate tractable learning and inference, G is usually assumed to form a tree $T = (V, E_T)$ [2,15,17]. In particular, most work uses the kinematic tree (see Fig. 1) as the underlying tree model.

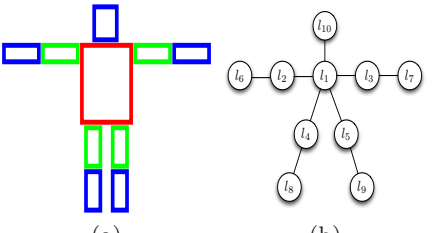


Fig. 1. Representation of a human body. (a) human body represented as a 10-part model; (b) corresponding kinematic tree structured model.

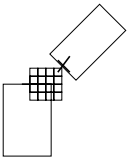


Fig. 2. Discrete binning for spatial prior

3.2 Learning and Inference in a Single Tree Model

Inference: Given the model parameters $\Theta = \{\alpha_i, \beta_i\}$, parsing an image I of a human body involves computing the posterior distribution over part locations L, i.e., $P(L|I, \Theta)$. Then the optimal part locations can be found by the *maximum a posterior* estimation $L_{MAP} = \arg\max_L Pr(L|I, \Theta)$. We use message-passing to carry out this computation (see [15,17] for details).

We first pick a node (e.g., the torso) in the tree model as the root and make a directed graph from the tree structure. Then we pass messages "upstream" starting from leaf nodes to their parents. The message from part i to part j is:

$$m_i(l_j) \propto \sum_{l_j} \psi(l_i - l_j) a_i(l_i) \tag{4}$$

$$a_i(l_i) \propto \phi(l_i) \prod_{k \in kids_i} m_k(l_i) \tag{5}$$

$\phi(l_i)$ is obtained by convolving the edge image with the filter β_i. $m_i(l_j)$ can be computed by convolving $a_i(l_i)$ with a 3D spatial filter (with coefficient α_i) extending the bins from Fig. 2. $a_i(l_i)$ is obtained by multiplying the response image $\phi(l_i)$ together with messages from its child nodes $m_k(l_i)$. At the root, a_i is the true conditional marginal distribution $Pr(l_i|I)$. Then starting from the root, we pass messages "downstream" from part j to part i to compute the true conditional marginal of each node:

$$Pr(l_i|I) \propto a_i(l_i) \sum_{l_j} \psi(l_i - l_j) Pr(l_j|I) \tag{6}$$

It can been shown that in a tree structure, the inference is exact, and converges to the true conditional marginal distributions after this message-passing scheme. Similar to previous work [15,17], we normalize each a_i to 1 for numerical stability, and keep track of the normalizing constants, which are needed for computing the partition function of the posterior $Pr(L|I, \Theta)$.

Learning Θ_{ML}: If we are given a set of training images I^t where part locations L^t have been labeled, one way of learning the model parameters $\Theta = \{\alpha_i, \beta_i\}$ is to maximize the joint likelihood of the labeled data:

$$\Theta_{ML} = \max_{\Theta} \prod_t Pr(I^t, L^t|\Theta) \tag{7}$$

$$= \max_{\Theta} \prod_t Pr(L^t|\Theta) \prod_t Pr(I^t|L^t, \Theta) \tag{8}$$

Θ_{ML} is also known as the maximum likelihood (ML) estimate of the model parameter Θ. Θ_{ML} can be found by independently fitting the ML estimate of each factor [17].

Learning Θ_{CL}: It has been noticed [17] that the ML estimate is not directly tied to the inference, and a better criterion is to optimize the posterior distribution:

$$\Theta_{CL} = \max_{\Theta} \prod_{t} Pr(L^t | I^t, \Theta) \tag{9}$$

Finding Θ_{CL} is equivalent to learning a Conditional Random Field (CRF) [8]. There are standard algorithms to learn Θ_{CL} using gradient ascent methods.

3.3 Boosted Multiple Trees

There is a trade-off between representational power and computational complexity amongst different forms of spatial priors. A complete graph captures all the possible spatial dependencies between all the parts, but the learning and inference of such models are intractable. On the other hand, tree-structured models are appealing due to their tractability. However, previous work [9,21] has shown that tree models fail to capture some additional dependencies between body parts.

In order to alleviate the limitation of tree models, various classes of graph structures that allow tractable learning and inference have been studied, e.g., mixture of trees [11], triangulated graph [21], k-fan [1], common-factor model [9]. In this section, we present our algorithm on boosting multiple trees for human pose estimation. Our algorithm is based on AdaBoost.MRF proposed in Truyen et al. [27] with some modifications. The basic idea of this method is to combine multiple tree-structured models. Since each component of the combined model is still a tree, learning and inference will be tractable. At the same time, since we are using several trees, we can capture additional spatial dependencies that are missing from a single tree model. Although our model is similar to "mixtures of trees" at a first glance, there are some importance differences. "Mixtures of trees" is trained by the EM algorithm to maximize the likelihood of the training data, while our model is trained by boosting to minimize a loss function directly tied to inference.

Given an image I, the problem of pose estimation is to find the best part labeling L^* that maximize some function $F(L, I)$, i.e. $L^* = \arg\max_L F(L, I)$. $F(L, I)$ is known as the "strong learner" in the boosting literature. Given a set of training examples $(I^i, L^i), i = 1, 2, ..., N$. $F(L, I)$ is found by minimizing the following loss function:

$$L_O = \sum_i \sum_L \exp\left(F(I^i, L) - F(I^i, L^i)\right) \tag{10}$$

We assume $F(L, I)$ is a linear combination of a set of so-called "weak learners", i.e., $F(I, L) = \sum_t \alpha_t f_t(L, I)$. The t-th weak learner $f_t(L, I)$ and its corresponding weight α_t are found by minimizing the loss function defined in Equation 10, i.e. $(f_t, \alpha_t) = \arg\max_{f,\alpha} L_O$.

Since we are interested in finding the distribution $p(L|I)$, we can choose the weak learner as $f(L, I) = \log p(L|I)$. To achieve computational tractability, we assume each weak learner is defined on a tree model.

If we can successfully learn a set of tree-based weak learners $f_t(L, I)$ and their weights α_t, the combination of these weak learners captures more spatial dependencies than a single tree model. At the same, the inference in this model is still tractable, since each component is a tree.

Optimizing L_O is difficult, Truyen et al. [27] suggest optimizing the following alternative loss function:

$$L_H = \sum_i \exp\left(-F(L^i, I^i)\right) \tag{11}$$

It can be shown that L_H is an upper bound of the original loss function L_O, provided that we can make sure $\sum_j \alpha_j = 1$. In Truyen et al. [27], the requirement $\sum_j \alpha_j = 1$ is met by scaling down each previous weak learner's weight by a factor of $1 - \alpha_t$ as $\alpha_j^{'} \leftarrow \alpha_j(1 - \alpha_t)$, for $j = 1, 2, ..., t-1$, so that $\sum_{j=1}^{t-1} \alpha_j^{'} + \alpha_t = \sum_{j=1}^{t-1} \alpha_j(1 - \alpha_t) + \alpha_t = 1$, since $\sum_{j=1}^{t-1} \alpha_j = 1$.

In practice, we find this trick sometimes has the undesirable effect of scaling down previous weak learners to have zero weights. So we use another method by scaling down each weak learner's weight up to t by a factor of $1/(1 + \alpha_t)$, i.e., $\alpha_j^{'} \leftarrow \frac{\alpha_j}{1+\alpha_t}$ for $j = 1, 2, ..., t$. It can be easily shown that we still have $\sum_{j=1}^{t} \alpha_j^{'} = \sum_{j=1}^{t-1} \frac{\alpha_j}{1+\alpha_t} + \frac{\alpha_t}{1+\alpha_t} = 1$, since $\sum_{j=1}^{t-1} \alpha_j = 1$.

In practice, the algorithm could be very slow, since learning CRF parameters requires gradient ascent on a high dimensional space. To speed up the learning process, we employ several simple tricks. Firstly, we learn $\Theta_{CL} = \{\alpha_i, \beta_i\}$ using the kinematic tree structure, and fix the appearance parameters $\{\beta_i\}$ during the boosting process. The rational behind this is that multiple tree structures should only affect the spatial prior, not the appearance model. Secondly, during each boosting iteration, we learn Θ_{ML} instead of Θ_{CL}. Thirdly, instead of selecting the best tree structure in each iteration, we simply sequentially select a tree from a set of pre-specified tree structures. We also allow re-selecting a tree.

4 Experiments

We test our algorithm on the people dataset used in previous work [15,17]. This dataset contains 305 images of people in various interesting poses. First 100 images are used for training, and the remaining 205 images for testing. We manually select three tree structures shown in Fig. 4, although it will be an interesting future work on how to automatically learn the tree structure at each iteration in an efficient way. The results are obtained by running 15 boosting iterations. We visualize the posterior distribution $Pr(L|I)$ on a 2D image using the same technique in Ramanan [15], where the torso is represented as red, upper-limbs as green, and lower-limbs and the head as blue. Some of the parsing

Input: $i = 1, 2, ..., D$ data pairs, graphs $\{G_i = (V_i, E_i)\}$
Output: set of trees with learned parameters and weights
Select a set of spanning trees $\{\tau\}$
Choose the number of boosting iterations T
Initialize $\{w_{i,0} = \frac{1}{D}\}$, and $\alpha_1 = 1$
for each boosting round $t = 1, 2, ..., T$
  Select a spanning tree τ_t
  /* Add a weak learner */
  $\Theta_t = \arg\max_{\Theta} \sum_i w_{i,t-1} \log Pr_{\tau_t}(L_i, I_i|\Theta)$
  $f_t = \log Pr_{\tau_t}(L|I, \Theta_t)$
  if $t > 1$ **then**
    select the step size $0 < \alpha_t < 1$ using line searches
  end if
  /* Update the strong learner */
  $F_t = \frac{1}{1+\alpha_t} F_{t-1} + \frac{\alpha_t}{1+\alpha_t} f_t$
  /* Scale down the previous learners' weights*/
  $\alpha_j \leftarrow \frac{\alpha_j}{1+\alpha_t}$, for $j = 1, 2, ..., t$
  /* Re-weight training data*/
  $w_{i,t} \propto w_{i,t-1} \exp(-\alpha_t f_{i,t})$
end for
Output $\{\tau_t\}$,$\{\Theta_t\}$ and $\{\alpha_t\}$, $t = 1, 2, ..., T$

Fig. 3. Algorithm of boosted multiple trees

results are shown in Fig. 5. We can see that our parsing results are much clearer than the one using the kinematic tree. In many images, the body parts are almost clearly visible from our parsing results. In the parsing results of using the kinematic tree, there are many white pixels, indicating high uncertainty about body parts at those locations. But with multiple trees, most of the white pixels are cleaned up. We can imagine if we sample the part candidates l_i according to $Pr(l_i|I; \Theta)$ and use them as the inputs to other pose estimation algorithms (e.g., Ren et al. [18]), the samples generated from our parsing results are more likely to be the true part locations.

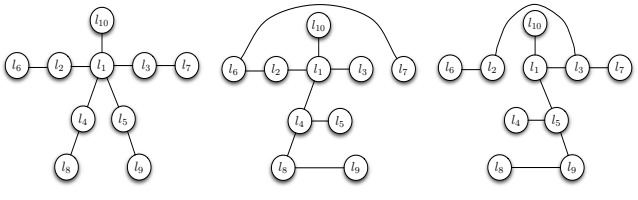


Fig. 4. Three tree structures used for boosting

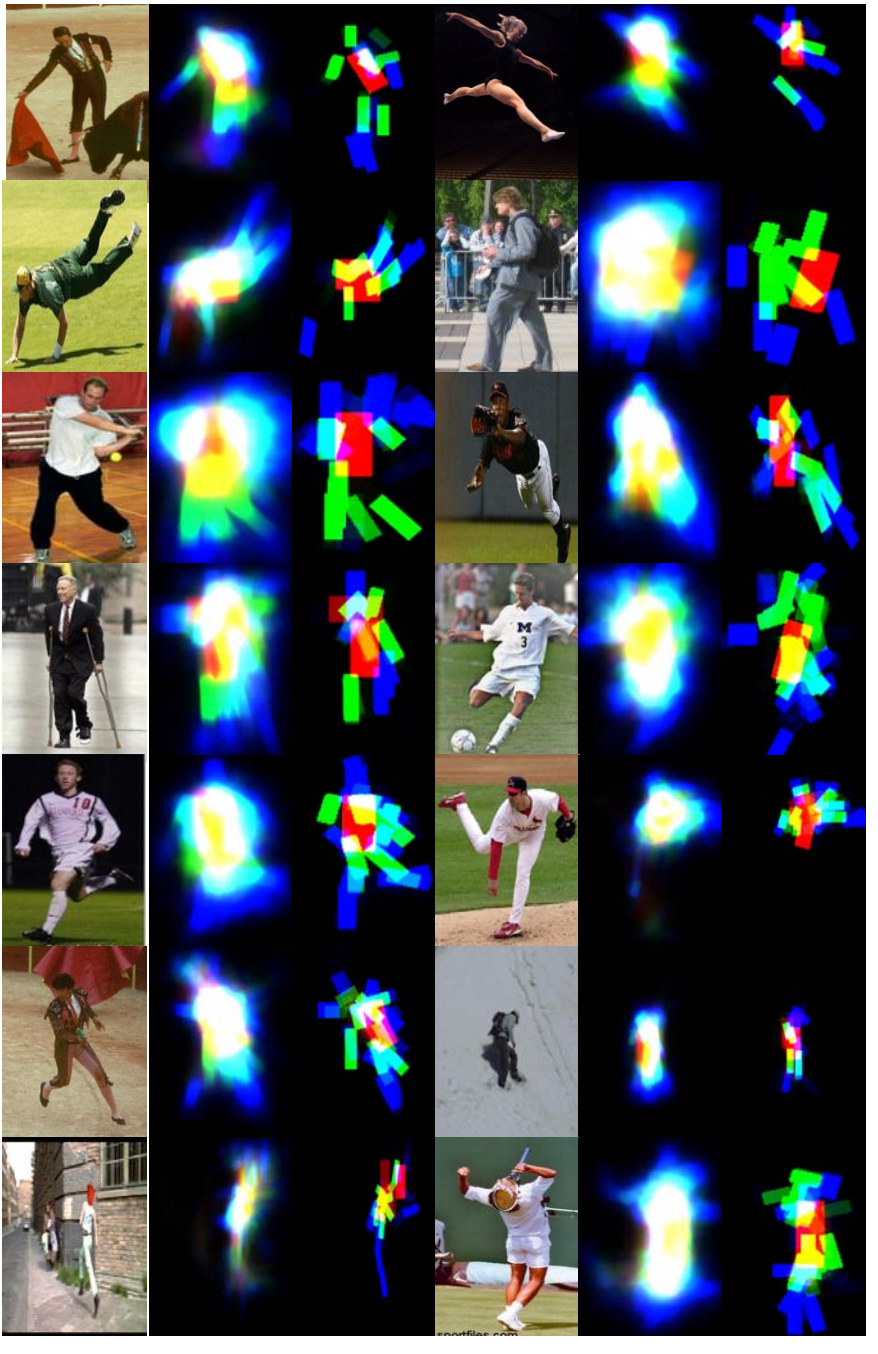
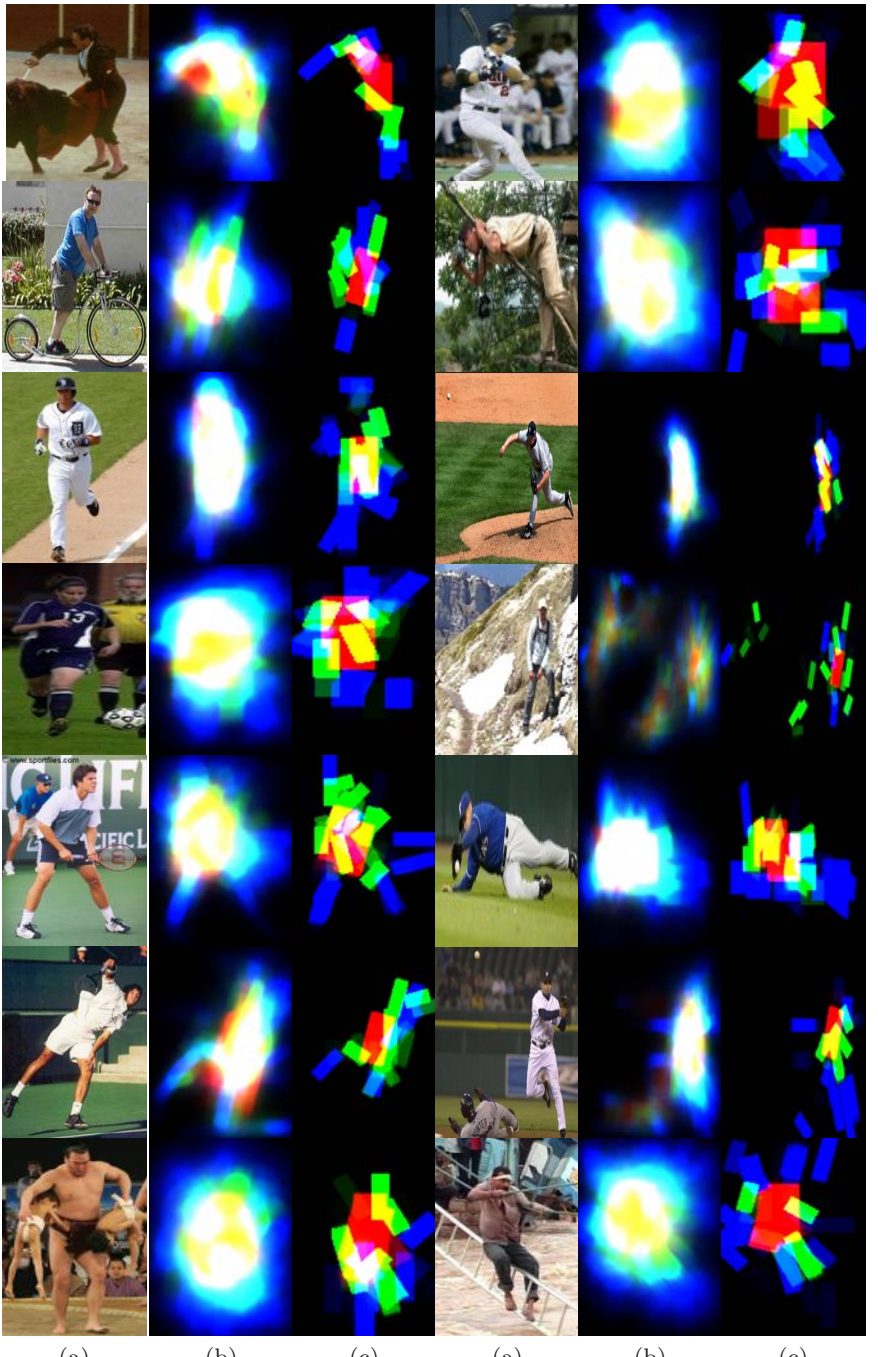


Fig. 5. Some results of our algorithm: (a) original images; (b) results of using one kinematic tree; (c) results of using multiple trees



(a) (b) (c) (a) (b) (c)

Fig. 5. (*Continued*)

5 Conclusion and Future Work

We have presented a framework for modeling human figures as a collection of tree-structured models. This framework has the computational advantages of previous tree-structured models used for human pose estimation. At the same time, it models a richer set of spatial constraints between body parts. We demonstrate our results on a challenging dataset with substantial pose variations.

Human pose estimation is an extremely difficult computer vision problem. The solution of this problem probably requires the symbiosis of various kinds of visual cues. This paper represents our first step in that direction. Our framework nicely solves the problem of modeling spatial dependencies between non-connected body parts. In the future, we would like to extend our framework to other cues (e.g., color consistency, occlusion relationships). We would also like to combine our framework with the iterative color parsing [15].

Acknowledgement

We would like to thank Deva Ramanan for providing source code, the dataset, and many helpful discussions.

References

1. Crandell, D., Felzenszwalb, P.F., Huttenlocher, D.P.: Spatial priors for part-based recognition using statistical models. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 10–17 (2005)
2. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. International Journal of Computer Vision 61(1), 55–79 (2003)
3. Forsyth, D.A., Arikan, O., Ikemoto, L., O'Brien, J., Ramanan, D.: Computational studies of human motion: Part 1, tracking and motion synthesis. Foundations and Trends in Computer Graphics and Vision 1(2/3), 77–254 (2006)
4. Hogg, D.: Model-based vision: a program to see a walking person. Image and Vision Computing 1(1), 5–20 (1983)
5. Hua, G., Yang, M.H., Wu, Y.: Learning to estimate human pose with data driven belief propagation. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 747–754 (2005)
6. Ioffe, S., Forsyth, D.: Finding people by sampling. In: IEEE International Conference on Computer Vision, vol. 2, pp. 1092–1097 (1999)
7. Ju, S.X., Black, M.J., Yaccob, Y.: Cardboard people: A parameterized model of articulated image motion. In: International Conference on Automatic Face and Gesture Recognition, pp. 38–44 (1996)
8. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML. International Conference on Machine Learning, pp. 282–289 (2001)
9. Lan, X., Huttenlocher, D.P.: Beyond trees: Common-factor models for 2d human pose recovery. In: IEEE International Conference on Computer Vision, vol. 1, pp. 470–477 (2005)

10. Lee, M.W., Cohen, I.: Proposal maps driven mcmc for estimating human body pose in static images. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 334–341 (2004)
11. Meila, M., Jordan, M.I.: Learing with mixtures of trees. Journal of Machine Learning Research 1, 1–48 (2000)
12. Mori, G.: Guiding model search using segmentation. In: IEEE International Conference on Computer Vision, vol. 2, pp. 1417–1423 (2005)
13. Mori, G., Malik, J.: Estimating human body configurations using shape context matching. In: European Conference on Computer Vision, vol. 3, pp. 666–680 (2002)
14. Mori, G., Ren, X., Efros, A., Malik, J.: Recovering human body configuration: Combining segmentation and recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 326–333 (2004)
15. Ramanan, D.: Learning to parse images of articulated bodies. In: Advances in Neural Information Processing Systems, vol. 19, pp. 1129–1136 (2007)
16. Ramanan, D., Forsyth, D.A., Zisserman, A.: Strike a pose: Tracking people by finding stylized poses. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 271–278 (2005)
17. Ramanan, D., Sminchisescu, C.: Training deformable models for localization. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 206–213 (2006)
18. Ren, X., Berg, A., Malik, J.: Recovering human body configurations using pairwise constraints between parts. In: IEEE International Conference on Computer Vision, vol. 1, pp. 824–831 (2005)
19. Shakhnarovich, G., Viola, P., Darrell, T.: Fast pose estimation with parameter sensitive hashing. In: IEEE International Conference on Computer Vision, vol. 2, pp. 750–757 (2003)
20. Sigal, L., Black, M.J.: Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2041–2048 (2006)
21. Song, Y., Goncalves, L., Perona, P.: Unsupervised learning of human motion. IEEE Transaction on Pattern Analysis and Machine Intelligence 25(7), 814–827 (2003)
22. Srinivasan, P., Shi, J.: Bottom-up recognition and parsing of the human body. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2007)
23. Sudderth, E.B., Mandel, M.I., Freeman, W.T., Willsky, A.S.: Distributed occlusion reasoning for tracking with nonparametric belief propagation. In: Advances in Neural Information Processing Systems, pp. 1369–1376. MIT Press, Cambridge (2004)
24. Sullivan, J., Carlsson, S.: Recognizing and tracking human action. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2352, pp. 629–644. Springer, Heidelberg (2002)
25. Torralba, A., Murphy, K.P., Freeman, W.T.: Contextual models for object detection using boosted random fields. In: Advances in Neural Information Processing Systems, vol. 17, pp. 1401–1408. MIT Press, Cambridge (2005)
26. Toyama, K., Blake, A.: Probabilistic exemplar-based tracking in a metric space. In: IEEE International Conference on Computer Vision, vol. 2, pp. 50–57 (2001)
27. Truyen, T.T., Phung, D.Q., Bui, H.H., Venkatesh, S.: AdaBoost.MRF: Boosted markov random forests and application to multilevel activity recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1686–1693 (2006)

Gradient-Enhanced Particle Filter for Vision-Based Motion Capture

Daniel Grest and Volker Krüger

Aalborg University Copenhagen, Denmark
Computer Vision and Machine Intelligence Lab
{dag,vok}@cvmi.aau.dk

Abstract. Tracking of rigid and articulated objects is usually addressed within a particle filter framework or by correspondence based gradient descent methods. We combine both methods, such that (a) the correspondence based estimation gains the advantage of the particle filter and becomes able to follow multiple hypotheses while (b) the particle filter becomes able to propagate the particles in a better manner and thus gets by with a smaller number of particles. Results on noisy synthetic depth data show that the new method is able to track motion correctly where the correspondence based method fails. Further experiments with real-world stereo data underline the advantages of our coupled method.

1 Introduction

Motion tracking and human pose estimation are important applications in motion analysis for sports and medical purposes. Motion capture products used in the film industry or for computer games are usually marker based to achieve high quality and fast processing.

Marker-less motion capture approaches often rely on gradient based methods [13, 3, 19, 7, 10, 18]. These methods estimate the parameters of a human body model by minimizing differences between model and some kind of observations, e.g. depth data from stereo, visual hulls or silhouettes. Necessary for minimization are correspondences between model and observed data. The main problem of these correspondence based optimization methods is, that they often get stuck in wrong local minima. From this wrong estimated pose they can usually not recover.

Other approaches like particle filters [5, 11] try to approximate the probability distribution in the state space by a large number of particles (poses) and are therefore unlikely to get stuck in local minima, because they can follow and test a large number of hypotheses. However, to be sure that the "interesting" region (*typical set*) of a high-dimensional state space is properly sampled, a large amount of particles is usually necessary [15, 2].

To take the advantages of both approaches, we combine a particle filter based approach [15, 11, 6] with correspondence based gradient estimation.

The effect can be interpreted in two ways: (1) Following the local gradient within the particle filter allows to find minima (including the global minimum) with less particles and (2) enhancing the gradient descent method with multiple hypothesis helps to avoid to get stuck in local minima.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 28–41, 2007.

The key to our approach and the main difference to a particle filter like CONDENSATION [11] is the gradient descent for each particle and the merging of particles. Particles, which are close to each other after the gradient descent, are merged into a single particle. Then, the idea is to re-distribute (propagate) the particles in a way that takes into account the local shape of the likelihood function. That way the number of particles can be greatly reduced while maintaining the ability to follow multiple hypotheses.

We will discuss our new approach in the context of marker less motion tracking from a single stereo view with two cameras. There is a wide variety of stereo algorithms available differing in quality and processing time. Commercial stereo cameras calculate depth data on chip using a simple algorithm [20] in real-time and keep the CPU free. Other more sophisticated algorithms need up to multiple minutes per image pair [8].

We will at first discuss relevant work within the field of motion tracking. Then, introduce our body model and the motion parameterization, which are used in the correspondence based optimization. The next section briefly explains important aspects of particle filter tracking methods, which are necessary for the combination. Then, results on synthetic data and real motion sequences are given, that show the advantages of the combined motion tracking. The last section concludes the paper with a short discussion of the presented achievements.

2 Related Work

Motion tracking of the human body is addressed in the literature with different methods. A recent and extensive survey of vision based human motion tracking can be found in [16]. Approaches relevant to this work arise from different directions depending on the kind of input data, e.g. depth data or images, and the number of cameras. Visual hull approaches have shown to give very accurate results [13, 3]. They build the visual hull of the person from segmented images of multiple cameras and then fit a template model to the 3D hull. Usually some kind of gradient based optimization is utilized to estimate the motion parameters of the model. Fitting the template model directly to segmented images is another possibility as done in [19], where it was shown, that the marker-less approach has an accuracy similar to marker based tracking.

Similar to the visual hull approach is [12], where multiple stereo cameras observe the motion of a person. The resulting 3D points are then used with an Extended Kalman Filter to estimate the upper body motion.

When only two or less cameras are used for motion tracking, particle filters [11] or particle filtering methods are utilized [5]. Their advantage is, that multiple tracking hypotheses can be followed, because a large number of particles approximates the posterior probability of the system's state. Therefore, the tracking is not prone to get stuck in local minima and multiple hypotheses can be followed. This ability to track multiple hypotheses is very useful, because body parts can become occluded, if only a single viewpoint is used. Then, the occluded motion has to be 'guessed' in order to track successfully, when the occluded body part becomes visible again. However, if full body motion with 30 DOF is to be estimated the number of particles can approximate the posterior distribution only within a small region in the state space. Therefore, once again it is likely to face the problem of local minima. Otherwise the number of particles has to be increased, which increases computation time. For 30 DOF the necessary

amount of particles can result in a computation time of up to multiple hours per image frame of a video sequence. However a parallel processing of particles is possible. Our approach decreases the amount of necessary particles significantly and allows such processing in reasonable time.

Motion tracking from a single stereo view as in this work has been addressed before in [4], where the human is modeled with 6 cylinders and motion can be roughly tracked with 10Hz. The authors use projective methods, which are inferior to direct methods as they state themselves in [4], but are easier to implement and require less computation time (no comparison is made). In [18] a direct approach is taken, where a human model consisting of spheres (meta-balls) is fitted to stereo data silhouettes from a single view. Due to the high number of estimated parameters, the method is not real-time capable. Both methods use a correspondence based gradient descent method, which requires manual initialization and can get stuck in local minima.

In [17], 3D body tracking is done using particle propagation. The Zakai Equation is applied to model the propagation of probability density function (pdf) over time in order to reduce the number of particles.

In a previous work [10] we showed that our direct approach is able to track upper arm motion with 5Hz and can therefore compete with the projective method of [4] according to processing time. Here we present a combination with a particle filter that allows us to track very noisy arm motion and complex full body motion of the whole body from stereo data alone, even though body parts are temporarily occluded.

A nice review on Monte Carlo-based techniques can be found in [15]. Classical papers about particle filtering are Condensation [5, 11], Sequential Importance Sampling [6] and sequential Monte Carlo [14].

3 Body Model and Motion Parameterization

The motion capabilities of the human model is based on the MPEG4 standard, with up to 180 DOF. An example model is shown in Fig. (1) left. The MPEG4 description allows to exchange body models easily and to re-animate other models with the captured motion data. The model for a specific person is obtained by silhouette fitting of a template model as described in [9].

The MPEG4 body model is a combination of kinematic chains. The motion of a point, e.g. on the hand, may therefore be expressed as a concatenation of rotations [10]. As the rotation axes are known, e.g. the flexion of the elbow, the rotation has only one degree of freedom (DOF), i.e. the angle around that axis. In addition to the joint angles, there are 6 DOF for the position and orientation of the object within the global world coordinate frame. For an articulated object with p joints we describe the transformation of the point $\boldsymbol{p}$ within the chain [10] as

$$\begin{aligned}\boldsymbol{m}(\boldsymbol{\theta}, \boldsymbol{p}) = &(\theta_x, \theta_y, \theta_z)^T + \\ &(R_x(\theta_\alpha) \circ R_y(\theta_\beta) \circ R_z(\theta_\gamma) \circ R_{\omega_1,q_1}(\theta_1) \circ \cdots \\ &\cdots \circ R_{\omega_p,q_p}(\theta_p))\boldsymbol{p}\end{aligned}$$

where $(\theta_x, \theta_y, \theta_z)^T$ is the global translation, R_x, R_y, R_z are the rotations around the global x, y, z-axes with Euler angles α, β, γ and $R_{\boldsymbol{\omega},\boldsymbol{q}}(\theta_i), i \in \{1..p\}$ denotes the

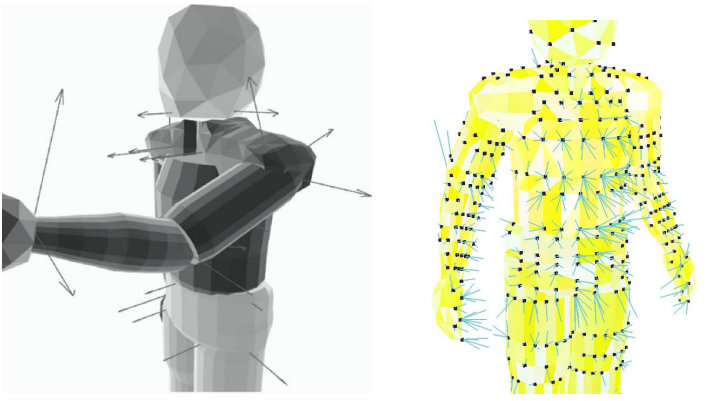


Fig. 1. Left: The body model with rotation axes shown as arrows. Right: The difference (small blue lines) between observed depth point and nearest model point (black boxes) is minimized.

rotation around the known axis with angle θ_i. The axis is described by the normal vector $\boldsymbol{\omega_i}$ and a point $\boldsymbol{q_i}$ on the axis.

Eq. 1 gives the position of a point $\boldsymbol{p}$ on a specific segment of the body (e.g. the hand) with respect to joint angles $\boldsymbol{\theta}$ and an initial body pose.

If the current pose is $\boldsymbol{\theta}_t$ and only relative motion is estimated the resulting Jacobian is:

$$J = \begin{bmatrix} 1\,0\,0 & \frac{\partial m_x}{\partial \theta_\alpha} & \frac{\partial m_x}{\partial \theta_\beta} & \frac{\partial m_x}{\partial \theta_\gamma} & \frac{\partial m_x}{\partial \theta_1} & .. & \frac{\partial m_x}{\partial \theta_p} \\ 0\,1\,0 & \frac{\partial m_y}{\partial \theta_\alpha} & \frac{\partial m_y}{\partial \theta_\beta} & \frac{\partial m_y}{\partial \theta_\gamma} & \frac{\partial m_y}{\partial \theta_1} & .. & \frac{\partial m_y}{\partial \theta_p} \\ 0\,0\,1 & \frac{\partial m_z}{\partial \theta_\alpha} & \frac{\partial m_z}{\partial \theta_\beta} & \frac{\partial m_z}{\partial \theta_\gamma} & \frac{\partial m_z}{\partial \theta_1} & .. & \frac{\partial m_z}{\partial \theta_p} \end{bmatrix} \tag{1}$$

The derivatives at zero are:

$$\left.\frac{\partial \boldsymbol{m}(\boldsymbol{\theta}, \boldsymbol{p})}{\partial \theta_j}\right|_{\boldsymbol{\theta}=\boldsymbol{0}} = \boldsymbol{\omega_j} \times (\boldsymbol{p} - \boldsymbol{q_j}) \tag{2}$$

where $j \in \{1, .., p\}$ and $\boldsymbol{q}_j$ is an arbitrary point on the rotation axis. The simplified derivative at zero is valid, if relative transforms in each iteration step of the *Nonlinear Least Squares* are calculated and if all axes and corresponding point pairs are given in world coordinates.

4 Gradient Enhanced Particle Filtering

Because our method combines the advantages of gradient based optimization methods with particle filtering, we give now a brief overview of important aspects of both. Then, we present the combined algorithm and discuss the main differences to particle filters.

4.1 Correspondence Based Pose Estimation

Correspondence based methods for pose estimation of articulated objects minimize an error function with respect to motion parameters. The human body can be modeled as an articulated object, consisting of multiple kinematic chains.

It is common to assume that the shape and size of the body model is known for the observed person, such that the minimization is only with respect to joint angles and the global transform. In that case, the kinematic chain simplifies to a chain of rotations around arbitrary axes in space Given here is a short description of the estimation algorithm, for more details see [10].

Estimating the motion of the human body from given 3D-3D correspondences $(\boldsymbol{p_i}, \tilde{\boldsymbol{p}}_{\boldsymbol{i}})$ is done here by solving a Nonlinear Least Squares Problem. The minimization yields the joint angles and the global orientation and position. For n correspondences the minimization problem is given as:

$$\min_{\boldsymbol{\theta}} \sum_{i}^{n} |\boldsymbol{m}(\boldsymbol{\theta}, \boldsymbol{p_i}) - \tilde{\boldsymbol{p}}_{\boldsymbol{i}}|^2 \quad . \tag{3}$$

To find the minimizer with the iterative *Gauss-Newton* method the Jacobian of the residual functions, Eq. (1), is necessary.

The points $\boldsymbol{p_i}$ and $\tilde{\boldsymbol{p}}_{\boldsymbol{i}}$ form a correspondence. For each observed point $\tilde{\boldsymbol{p}}_{\boldsymbol{i}}$ the closest point $\boldsymbol{p_i}$ on the model is sought. Therefore, different observed points can have the same corresponding model point. This is shown in Fig. 1 right, where each correspondence is shown as a small blue line and the model points are drawn as black boxes.

The minimization problem is solved with the dampened Gauss-Newton method [1], which is similar to the Levenberg-Marquardt [1] method. Dampening ensures that the parameter change does not increase infinitely, if the determinant of the Gram matrix $J^T J$ is close to zero, which can happen when a body part is largely occluded. The solution is found by solving iteratively:

$$\boldsymbol{\theta_{t+1}} = \boldsymbol{\theta_t} - (J^T J + \lambda I)^{-1} J^T \boldsymbol{r}(\boldsymbol{\theta}_t) \tag{4}$$

where the Jacobian J is given in equation (1), I is the identity matrix, λ is the dampening value (set to 0.1) and $\boldsymbol{r}(\boldsymbol{\theta}_t)$ is the vector with current residuals. For each point there are three residuals, one for each component (x, y, z):

$$\boldsymbol{r_i}(\boldsymbol{\theta}_t) = \boldsymbol{m}(\boldsymbol{\theta}, \boldsymbol{p_i}) - \tilde{\boldsymbol{p}}_{\boldsymbol{i}} \tag{5}$$

and $\boldsymbol{r_i} = (r_{ix}, r_{iy}, r_{iz})$.

The optimization consists of two loops: The Gauss-Newton method (GN) loops until it converges on the given set of point correspondences. Because the correspondences are not always correct, new correspondences are calculated again after convergence of the GN. Then, GN starts anew with the improved set of correspondences. This Iterative Closest Point (ICP) method [1] can be repeated until convergence. However, the gain in more then 3 ICP iterations is very small, therefore we usually apply only 2 or 3 ICP optimizations. We will use in the following the term "gradient tracking" in order to refer to this technique.

It is important to note, that the Gauss-Newton optimization is more efficient than a standard Gradient Descent and requires less control parameters. However we will refer to it as "gradient tracking", because both methods rely on the gradient.

4.2 Particle Filter

The new method borrows some ideas from particle filter methods like CONDENSATION [11]. Particle filter approaches aim at estimating the posterior probability

distribution of a system state $\boldsymbol{z}_t$ at time t, the person's current pose in our case, from observations $I_1, \ldots, I_t$:

$$\begin{aligned} & p(\boldsymbol{z}_t | I_1, I_2, \ldots, I_t) \equiv p_t(\boldsymbol{z}_t) \\ & = \int_{\boldsymbol{z}_{t-1}} p(I_t|\boldsymbol{z}_t) p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}) p_{t-1}(\boldsymbol{z}_{t-1}) \ . \end{aligned} \tag{6}$$

In this equation the state space is randomly sampled according to $p_{t-1}(\boldsymbol{z}_{t-1})$ and propagated according to a motion/diffusion model $p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})$. A likelihood probability $p(I_t|\boldsymbol{z}_t)$ for each particle is calculated, which reflects how good the observations fit to the hypothesis (the position of the particle in the state space). The posterior probability is then approximated by the weight and density of particles within the state space.

The major problem with these approaches is, that the number of necessary particles usually needs to reflect the dimensionality of the state space [15, 2]. If full body motion with 30 DOF is to be estimated the particles can usually approximate the posterior distribution only within a small region in the state space.

If depth data is the only input, calculation of the likelihood requires computation of differences between observed points and model surface. One way to compute these differences is a nearest neighbor search as described in the previous section. This search is, however, expensive in terms of computation time. Therefore, methods are desirable, which reduce the number of particles and allow to distribute the reduced set of particles in the most important regions of the state space.

4.3 Combination

In order to get by with a smaller number of particles, we apply the gradient tracking (section 4.1) to each particle. Since similar particles move to the *same* optimum, they can be merged with an appropriate adaption of the particle weight. The state space posterior probability is approximated in a particle filter by the particle weights and their spatial density. A possible weight adaption is the average of merged particle weights. However, all merged particles are nearly at the same position in the state space and therefore have nearly the same weight (likelihood). As a result it is sufficient to assign them the same weight. Another possible weight adaption is the addition of weights. However, these would favor large flat valleys in the posterior probability surface, because all particles within in this valley will descent towards the same minimum. This will also lead to a clustering of particles at specific positions, which is not desired, because we want to track as many hypotheses as possible. If the merged particles are redistributed in the next time step only according to a fixed motion model and diffusion model, it is likely that they end up in the same locally convex region (valley) and again merge at the same position in state space. Each valley can be understood as one likely pose hypothesis. It is desirable to track as many hypotheses as possible with a fixed amount of particles. To increase the number of tracked hypotheses and decrease the amount of particles per valley, the particles need to be redistributed at the next time step of Eq. (6) according tho the size and shape of their valley. This can be understood as enhancing our rather simple motion model (fixed velocity) to include the shape of the local probability surface.

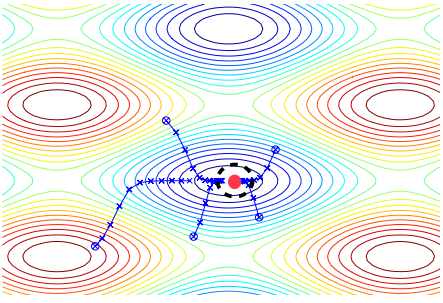


Fig. 2. Principle of the combined method. Particles whose position is within a specific area after optimization (black dashed circle) are merged into one particle (red circle).

In order to achieve this, we merge all particles after optimization, which are close to each other according to some distance d. In detail, if we use N particles $\boldsymbol{z}_1^{t-}, \dots \boldsymbol{z}_N^{t-}$ in our particle filter, then we have after the merging $M < N$ meaningful particles $\boldsymbol{z}_1^t, \dots \boldsymbol{z}_M^t$ left, and $N - M$ particles were merged into the remaining M particles.

Then, in order to distribute the particles in the next time step optimally, we estimate the size of the locally convex region by computing the covariance Σ_i of all those particles. Let $\boldsymbol{z}_{i_1}^{t-}, \dots \boldsymbol{z}_{i_k}^{t-}$ be the particles that merged together into the particle $\boldsymbol{z}_i^t$. The $^-$ at the top denotes the particles *before* their gradient descent and merging, t denotes the time step. The final particle $\boldsymbol{z}_i^t$ (without the $^-$) after gradient descent and merging (red circle in Fig.2) is the one with highest likelihood and is used as the mean for the covariance:

$$\Sigma_i = \frac{1}{K} \sum_j^K (\boldsymbol{z}_j^{t-} - \boldsymbol{z}_i^t)(\boldsymbol{z}_j^{t-} - \boldsymbol{z}_i^t)^T \tag{7}$$

where K is the amount of particles, which merged into $\boldsymbol{z}_i^t$. It is important to note that the covariance is calculated from the particles *before* the gradient descent and merging. Fig. 2 illustrates the merging. The blue lines show a few steps of the gradient descent for five particles. They are within a certain region (the black dashed circle) after optimization and therefore merged. The idea is to distribute particles in the next frame outside that locally convex region (valley), because otherwise they would end up again at the same position and give no additional information about the state space. Thus, at the next time step $t + 1$ N new particles are drawn from the remaining M particles of time step t according to the prior p_{t+1}. Then, each particle is propagated according to some motion model $f(\boldsymbol{z}^{t+1-})$ with added Gaussian noise. Let the particle $\boldsymbol{z}^{t+1-}$ be spawned off from the particle $\boldsymbol{z}_i^t$. The covariance of the above Gaussian is given by the covariance Σ_i of the original particle $\boldsymbol{z}_i^t$.

Our gradient enhanced particle filter can be summarized as follows: Input to our algorithm are the depth points for the current image frame and an initial pose in the beginning. The steps of the algorithm are for each frame similar to a particle filter method except for the gradient descent and the merging of particles.

1. Only at the first frame: Distribute particles according to an initial distribution in the vicinity of the given initial pose.
2. Draw new particles $\boldsymbol{z}_1^{t-}, \ldots \boldsymbol{z}_N^{t-}$ according to $p_t(\boldsymbol{z}_t)$.
3. Distribute and propagate each particle according to the covariance Σ_i of the original particle and propagate according to a diffusion/propagation model: $p(\boldsymbol{z}^{t+1-}|\boldsymbol{z}^{t-}, \Sigma)$ $= Gauss(f(\boldsymbol{z}^{t-}), \Sigma)$. Here, the motion model consists of a deterministic motion model f plus Gaussian noise, and Σ_i defines the Gaussian covariance matrix.
4. Gradient Descent for each particle $\boldsymbol{z}_i^{t+1-}$ with 2 or 3 ICP optimizations:
   (a) Render model in current pose
   (b) Calculate visible model points
   (c) For each observed point find the closest model point, which makes a correspondence
   (d) Calculate a new pose by minimizing the differences of the correspondences
5. Assign the likelihood, calculated from the residual error.
6. Merge particles $\boldsymbol{z}_{i_1}^{t+1-}, \ldots \boldsymbol{z}_{i_k}^{t+1-}$, which are within a certain distance d to each other into one particle $\boldsymbol{z}_i^{t+1}$.
7. For each particle $\boldsymbol{z}_i^{t+1}$: calculate the covariance matrix Σ_i from all the particles $\boldsymbol{z}_{i_1}^{t+1-}, \ldots \boldsymbol{z}_{i_k}^{t+1-}$ which merged into $\boldsymbol{z}_i^{t+1}$.

The motion model $f(\boldsymbol{z}^{t-1})$ predicts the new pose of the human. In our experiments, we assume constant velocity. The distance d was chosen to be 4 degrees, such that particles are merged only if each single joint angle differs less than 4 degrees to a neighboring particle. The distance check is only applied on joint angles, not on the position of the body model in the world, because it is highly unlikely, that the same pose is estimated at different positions. This way additional control parameters are avoided.

The initial covariance for each particle is chosen to equal the distance d, such that particles are definitely distributed outside the merge area. The covariance is also reduced by 10% each frame. Without this reduction the covariance can increase indefinitely. Especially if only one particle is left in a valley, it is desirable to reduce the covariance, such that it is ensured, that nearby poses are tested.

5 Synthetic Data with Noise

In order to show the robustness of the new method to noisy measurements we conduct an experiment on depth data generated with openGL on a synthetic sequence. The motion involves four DOF, the elbow flexion and the shoulder flexion abduct and twisting. Three example images out of the 176 frame sequence are shown in Fig. 3. The depth data is calculated from the z-buffer values after rendering the model.

For testing, Gaussian noise with different standard deviations is added to the original depth data. Fig. 4 shows two views on the depth data and the model in starting pose. The z-buffer image was sub-sampled in order to generate approximately 10000 depth points. Approximately 750 points are visible on the right arm each frame.

At a deviation rate of 15cm in depth and 1cm in the other directions the normal tracking methods loses track after a few frames and gets stuck in an arbitrary pose as shown in the right of Fig. 4.

Fig. 3. First frame (left). Frame 60 and frame 176 of the synthetic sequence.

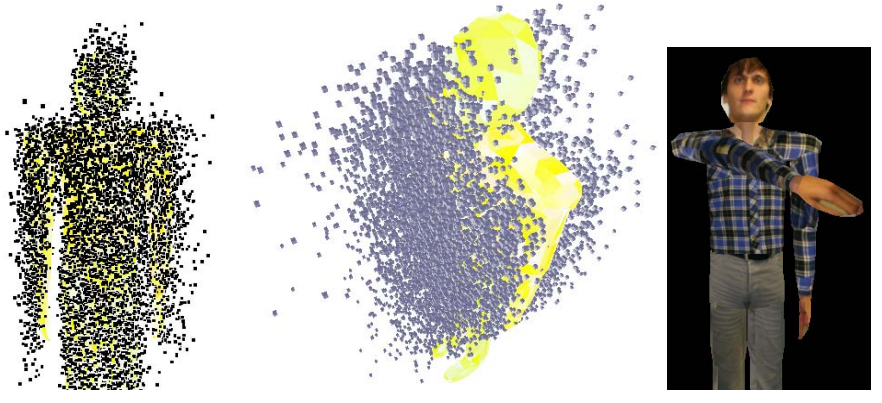


Fig. 4. Two views on the depth point cloud with Gaussian noise (deviation 15cm). Right: The standard tracking method loses track after a few frames and gets stuck in the pose shown.

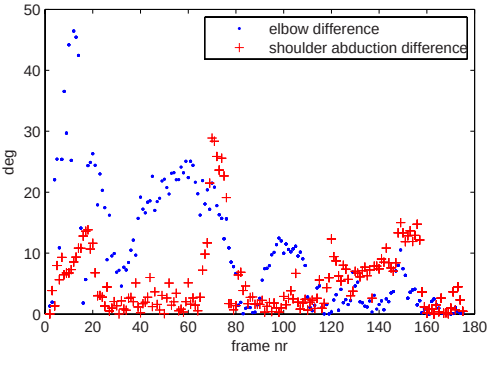


Fig. 5. Difference to ground truth with noise (deviation 15cm). The multi hypotheses tracking is able to track the whole sequence.

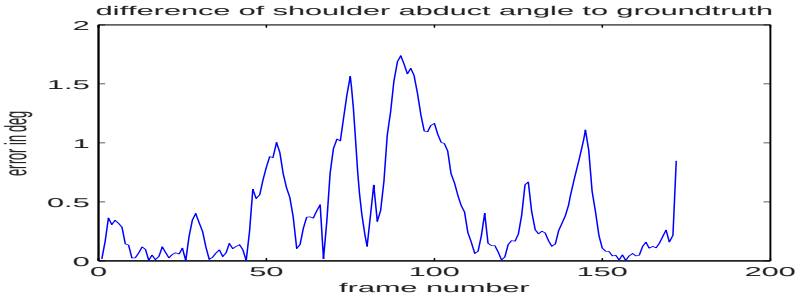


Fig. 6. Difference of the shoulder abduct angle to ground truth without noise

The multi-hypotheses tracking with 20 particles is able to track the motion correctly in spite of the heavy noise. The difference to the ground truth is shown in Fig. 5. In the beginning the estimate is far off with 50 degrees however the plot shows, that the tracking recovers from the wrong local minimum. For all hypotheses the minimum difference to the ground truth is taken, because the particle with the largest likelihood does not always give the correct pose. Plotted is the absolute error in degree over the whole sequence. The elbow difference is high around frames where the arm is close to the body as at frame 60 and 105, because the correspondences established with nearest neighbor are then most ambiguous. Where the arm is far away from the body the noise on the data does not result in so many wrong correspondences. Therefore, the error is decreased. Though the error is still large for most frames, because of the heavy noise, the results show, that the new method can track the motion over the whole sequence.

Without noise the gradient tracking estimates joint angles, whose difference to the ground truth is close to zero error as shown in Fig. 6 for the abduction angle of the

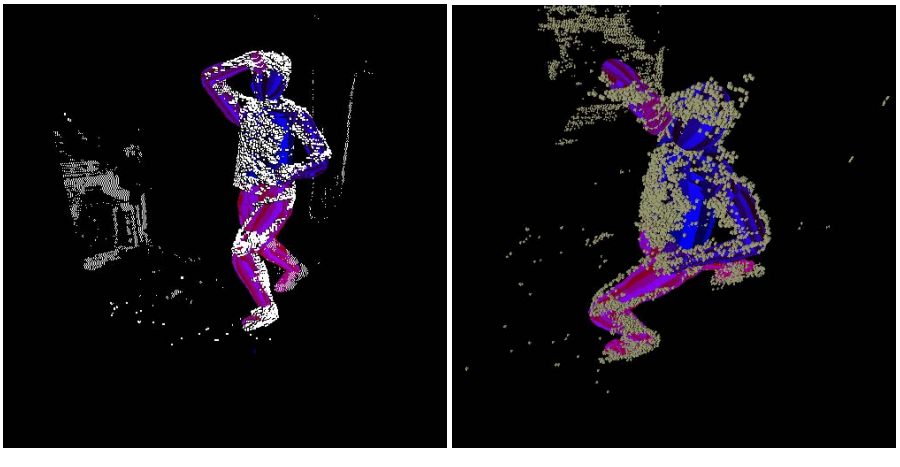


Fig. 7. Two views on the input data. Approx 10000 points are shown as white boxes.

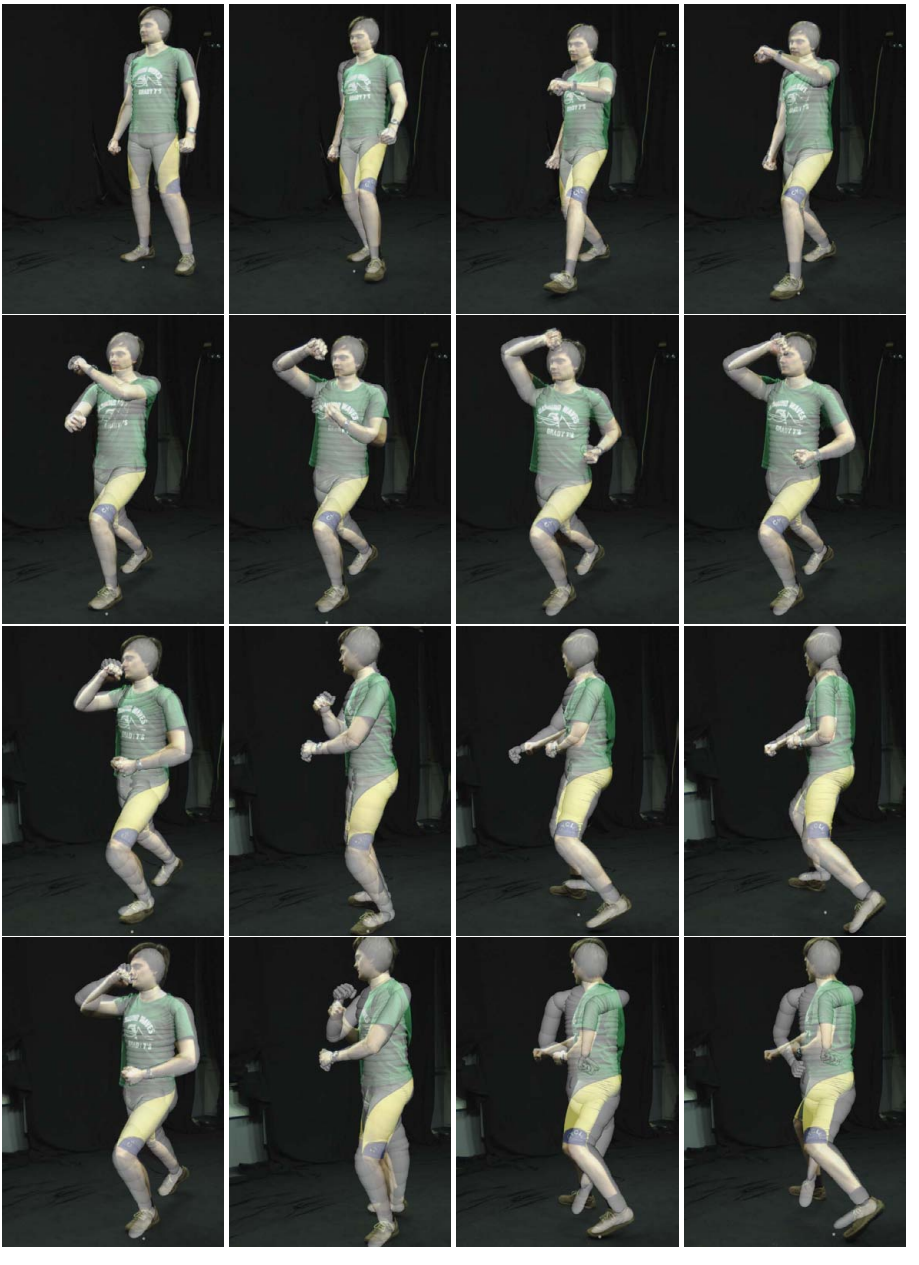


Fig. 8. Estimation results of the new combined method with 24 DOF (first 3 rows). The figure shows the original images together with the projected model in the estimated pose (overlayed in white). The last row shows the estimation results for the ”gradient tracking” for the same images as in the third row. The ”gradient tracking” loses track, because the right arm gets largely occluded, and is unable to recover.

shoulder. The error is not zero, because the model surface is approximated by the vertices of the model's triangles. Therefore, the nearest neighbor correspondences are not perfect.

6 Real Data

In order to show the possibilities with our new method, we give further results for a video sequence, which was recorded in a motion capture lab with 8 cameras at 25fps. Two of the cameras were arranged approx. 25 cm next to each other, such that stereo depth estimation can be performed. The stereo algorithm [8] produces dense accurate results in non-homogenous regions within approx. one minute computation time per frame. The used effective image size is 512x512. Fig. 7 shows the depth data that is used as input. The only assumption made here is, that no scene objects are within 80cm range of the person. Also knowledge about the floor position was incorporated from camera calibration, which was conducted for the internal parameters with a small checkerboard pattern according to [21]. This calibration also yields the orientation and distance of the stereo cameras. The external parameters (orientation of the floor) were then estimated with a large checkerboard pattern lying on the floor.

The initial pose of the person is provided manually. Estimated are 24 DOF, these are in detail 3 at each shoulder, one elbow angle, 3 at each hip, one angle for each knee, one angle at the ankle and 6 parameters for the global orientation and position.

The estimation time on a 2Ghz intel Core2 Duo (1 CPU) was about 2 seconds per particle and approximately 10000 data points. For 1000 data points and 14 DOF the computation time is about 200ms per particle.

Fig. 8 shows a few frames from the resulting estimation. The multi-hypotheses tracking with 100 particles is able to track the whole sequence of 180 frames (first 3 rows in the Fig.), even though one arm and one leg are almost completely occluded temporarily. Approximately 10000 data points from the complete set of 40000 reliable data points are used per frame, resulting in a Jacobian of size 30000×24.

The gradient tracking method (row 4) is able to track correctly up to 130 frames, but loses track when the person turns and the body parts become occluded (second image last row). The gradient tracking method is lost in that case and is unable to recover.

7 Conclusions

We presented a new method for motion tracking, that combines gradient based optimization from correspondences and motion tracking with particle filters. The combined approach allows to track arm motion in spite of heavy noise, where a normal gradient descent method fails. Further results on stereo video sequences showed that motion with 24 DOF can be tracked from a single viewpoint. At frames where the normal tracking gets stuck in local minima and thus loses track, the new method recovers and estimates correct poses.

The main contribution of the new method is the enhanced motion model, which includes the shape of the locally convex regions of the probability surface by estimation

of the covariance matrix from merged particles. In that way the number of particles is used more efficiently and allows to track a higher number of hypotheses.

We will exploit in the future further aspects of the new method: (1) the likelihood probability of the particle filter allows to easily include image information into the tracking, as for example motion areas from temporal image differences, (2) increase speed by parallel processing of particles and (3) include motion recognition probabilities into the motion model.

References

1. Chong, E.K.P.: Introduction to Optimization, 2nd edn. Wiley, Chichester (2001)
2. Cover, T., Thomas, J.: Elements of Information Theory. Wiley, Chichester (1991)
3. de Aguiar, E., Theobalt, C., Magnor, M., Seidel, H.-P.: Reconstructing Human Shape and Motion from Multi-View Video. In: CVMP. 2nd European Conference on Visual Media Production, London, UK (2005)
4. Demirdjian, D., Ko, T., Darrell, T.: Constraining Human Body Tracking. In: Proceedings of ICCV, Nice, France (October 2003)
5. Deutscher, J., Blake, A., Reid, I.: Articulated Body Motion Capture by Annealed Particle Filtering. In: CVPR, vol. 2 (2000)
6. Doucet, A., Godsill, S., Andrieu, C.: On sequential monte carlo sampling methods for bayesian filtering, vol. 10, pp. 197–209 (2000)
7. Mündermann, L., et al.: Validation of a markerless motion capture system for the calculation of lower extremity kinematics. In: Proc. American Society of Biomechanics, Cleveland, USA (2005)
8. Falkenhagen, L.: Hierarchical block-based disparity estimation considering neighbourhood constraints. In: International workshop on SNHC and 3D Imaging (1997)
9. Grest, D., Herzog, D., Koch, R.: Human Model Fitting from Monocular Posture Images. In: VMV (November 2005)
10. Grest, D., Woetzel, J., Koch, R.: Nonlinear Body Pose Estimation from Depth Images. In: Kropatsch, W.G., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, Springer, Heidelberg (2005)
11. Isard, M., Blake, A.: CONDENSATION – Conditional Density Propagation for Visual Tracking. International Journal of Compputer Vision 29, 5–28 (1998)
12. Stiefelhagen, R., Ziegler, J., Nickel, K.: Tracking of the Articulated Upper Body on Multi-View Stereo Image Sequences. In: CVPR, vol. 1, pp. 774–781. IEEE Computer Society, New York (2006)
13. Kehl, R., Bray, M., Van Gool, L.J.: Full Body Tracking from Multiple Views Using Stochastic Sampling. In: Proc. CVPR, pp. 129–136 (2005)
14. Liu, J.S., Chen, R.: Sequential monte carlo for dynamic systems, vol. 93, pp. 1031–1041 (1998)
15. MacKay, D.: Learning in Graphical Models. In: Jordan, M. (ed.) chapter Introduction to Monte Carlo Methods, pp. 175–204. MIT Press, Cambridge (1999)
16. Moeslund, T., Hilton, A., Krueger, V.: A Survey of Advances in Vision-based Human Motion Capture and Analysis. Computer Vision and Image Understanding 104(2-3), 90–127 (2006)
17. Moon, H., Chellappa, R., Rosenfeld, A.: 3d object tracking using shape-encoded particle propagation (2001)
18. Plänkers, R., Fua, P.: Model-Based Silhouette Extraction for Accurate People Tracking. In: Tistarelli, M., Bigun, J., Jain, A.K. (eds.) ECCV 2002. LNCS, vol. 2359, pp. 325–339. Springer, Heidelberg (2002)

19. Rosenhahn, B., Kersting, U., Smith, D., Gurney, J., Brox, T., Klette, R.: A System for Marker-Less Human Motion Estimation. In: Kropatsch, W.G., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, Springer, Heidelberg (2005)
20. Videre. Videre Design: Stereo On A Chip (2007), www.videredesign.com
21. Zhang, Z.: Flexible Camera Calibration By Viewing a Plane From Unknown Orientations. In: ICCV, Corfu, Greece, pp. 666–673 (1999)

Multi-activity Tracking in LLE Body Pose Space

Tobias Jaeggli[1], Esther Koller-Meier[1], and Luc Van Gool[1,2]

[1] ETH Zurich, D-ITET/BIWI, CH-8092 Zurich
[2] Katholieke Universiteit Leuven, ESAT/VISICS, B-3001 Leuven
jaeggli@vision.ee.ethz.ch

Abstract. We present a method to simultaneously estimate 3d body pose and action categories from monocular video sequences. Our approach learns a low-dimensional embedding of the pose manifolds using Locally Linear Embedding (LLE), as well as the statistical relationship between body poses and their image appearance. In addition, the dynamics in these pose manifolds are modelled. Sparse kernel regressors capture the nonlinearities of these mappings efficiently. Body poses are inferred by a recursive Bayesian sampling algorithm with an activity-switching mechanism based on learned transfer functions. Using a rough foreground segmentation, we compare Binary PCA and distance transforms to encode the appearance. As a postprocessing step, the globally optimal trajectory through the entire sequence is estimated, yielding a single pose estimate per frame that is consistent throughout the sequence. We evaluate the algorithm on challenging sequences with subjects that are alternating between running and walking movements. Our experiments show how the dynamical model helps to track through poorly segmented low-resolution image sequences where tracking otherwise fails, while at the same time reliably classifying the activity type.

1 Introduction

We consider the problem of estimating human body pose and action categories from image sequences. This is a difficult problem, especially when dealing with low quality low resolution imagery. Often the individual images do not provide enough information to resolve ambiguous situations, and strong prior models have to be adopted in order to compensate for that lack of information.

To address these problems we propose a method to estimate 3d body pose and action categories simultaneously. We learn strong dimensionality-reduced models of feasible body poses that belong to a certain activity or motion pattern, as well as the temporal evolution of the body poses over time. Furthermore, the transition functions between different activities are learned from training data as well. All the mappings are modelled using sparse kernel regressors, leading to efficient evaluation during tracking.

The observations are taken into account in the form of roughly segmented images that are obtained by a pre-processing step such as motion segmentation, background subtraction or other. The underlying relationship between image appearance and body poses is multivalued and ambiguous, thus non functional. Other learning based approaches have explicitly modelled the one-to-many discriminative mapping from appearances to poses (or the joint probability density function between appearance and

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 42–57, 2007.

pose) with mixtures of regressors or Gaussians (e.g. [1,2,3,4,18]). The number of required regressors is however a delicate parameter of these systems, as is the regularisation during the learning stage, which is needed to avoid overfitting. We therefore follow the opposite strategy, and model the generative mapping from body poses to appearance descriptors, which can be assumed to be functional and thus be approximated with a nonlinear kernel regressor. Although single-valued, the appearance prediction will be subject to uncertainty, because other factors than just the body configuration (pose) may affect appearance (clothing, physical constitution, lighting conditions etc). This is taken into account by learning a prediction variance matrix of the mapping.

A main focus of the proposed approach lies on the ambiguities and uncertainties that are inherent in monocular body tracking. Recursive Bayesian Sampling [5,6] offers a framework for dealing with non-Gaussian and multimodal body pose posteriors and allows us to integrate the nonlinear learned dynamical model. However, sampling-based algorithms are generally not applicable for inference in high-dimensional state spaces like the space of body poses. We therefore use Locally Linear Embedding (LLE, [7]) to find a low-dimensional embedding of our 60-dimensional pose parametrisation. With 4 LLE dimensions, the considered motions can be captured reasonably well.

In this paper we investigate typical human motion patterns such as walking and running. Rather than learning a unified representation that contains both walking and running motions, we learn separate activity specific models that allow us to explicitly recognize the performed activity along with the pose estimation, using a switching mechanism of the inference algorithm.

The main novelties of this paper are the generative appearance modelling, the tracking in a LLE-reduced pose representation with a nonlinear dynamical model, simultaneous recognition of multiple action categories, and the extraction of a globally optimal trajectory through the entire sequence.

1.1 Related Work

There is a wide variety of literature about body pose estimation and tracking (see [8] for an overview). Here we will have a look at the application of statistical methods that infer poses from one or multiple camera streams. Many authors adopt a discriminative strategy to infer poses directly from image descriptors [1,2,3,4,9,10,11].

Synchronous image sequences from multiple cameras typically provide enough information to resolve ambiguities. The discriminative mapping from descriptors to body poses can thus be modelled using a *single* regressor. In [11], a new image descriptor is introduced based on a voxel representation that is derived from segmented images of multiple cameras. This descriptor can then be directly mapped into pose space. In [10] multiple silhouette image descriptors and corresponding pose descriptors are concatenated and modelled with a mixture of Probabilistic PCA; poses can then be inferred given multiple views of the subject.

Monocular approaches have to deal with the one-to-many discriminative mapping from appearance to pose. This issue is explicitly addressed in [1,2,3,4] by learning *multiple* mappings in parallel as a mixture of regressors. In order to choose between the different hypotheses that the different regressors deliver, [1,2] use a geometric model

that is projected into the image to verify the hypotheses. Inference is performed for each frame independently in [1]. In [2] a temporal model is included using a bank of Kalman filters, and a Viterbi algorithm finds a path through the peaks of the posterior distribution. In [3,4] gating functions are learned along with the regressors in order to pick the right regressor(s) for a given appearance descriptor. The distribution is propagated analytically in [3], and temporal aspects are included in the learned discriminative mapping, whereas [4] adopts a generative sampling-based tracking algorithm with a first-order autoregressive dynamic model.

The mentioned discriminative approaches work in a bottom-up fashion, starting with the computation of the image descriptor, which requires the location of the figure in the images to be known beforehand. When including 2d bounding box estimation in the tracking problem, a learned dynamical model of the appearance might help the bounding box tracking, and avoid loosing the subject when it is temporarily occluded. To this end, [12] learns a subject-specific dynamic appearance model from a small set of initial frames, consisting of a low-dimensional embedding of the appearances and a motion model. This model is used to predict the location and appearance of the figure in future frames, within a *CONDENSATION* tracking framework. Similarly, low-dimensional embeddings of appearance (silhouette) manifolds are found using LLE in [13], where additionally the mapping from the appearance manifold to 3d pose in body joint space is learned using RBF interpolants, allowing for pose inference from sequences of silhouettes.

Instead of modelling manifolds in *appearance* space, [14,15,16] work with low dimensional embeddings of body *poses*. In [14,17], the low-dimensional pose representation, its dynamics, and the mapping back to the original pose space are learned in a unified framework. This approach does not include a learned statistical model of image appearance. Our method also models *pose* manifolds rather than *appearance* manifolds, because the pose manifold has fewer self-intersections than the appearance manifold, making the dynamics and tracking less ambiguous.

Regarding activity switching, [19] has proposed a state switching mechanism, where different dynamical models are chosen, depending on a discrete state variable. In our approach, the different states (activities) involve separate models for pose, dynamics and appearance.

Our approach fundamentally differs from the above-mentioned papers in that it simultaneously tracks in a state space that includes body pose, 2d bounding box location and a discrete activity label. Furthermore, we present a full-fledged pipeline with *generative* rather than *discriminative* modelling of the appearance, entirely based on learned models. The framework is built-up in a module based manner. Some choices of precise statistical methods that are applied for the individual modules are mainly based on practical considerations (e.g. efficiency, sparsity). They could be substituted by equivalent methods, like e.g. Isomap [20] instead of LLE, or regularised kernel regressors instead of RVMs.

The remainder of this paper is organised as follows. Section 2 introduces our learned models. In Section 3 the sample-based inference is presented and Section 4 shows experimental results on different video sequences.

2 Statistical Modelling

Figure 1 a) shows an overview of the tracking framework, reduced to a single activity category for clarity. Central element is the low-dimensional body pose parameterisation, with learned mappings back to the original pose space and into the appearance space. In this section all elements of the framework will be described in detail.

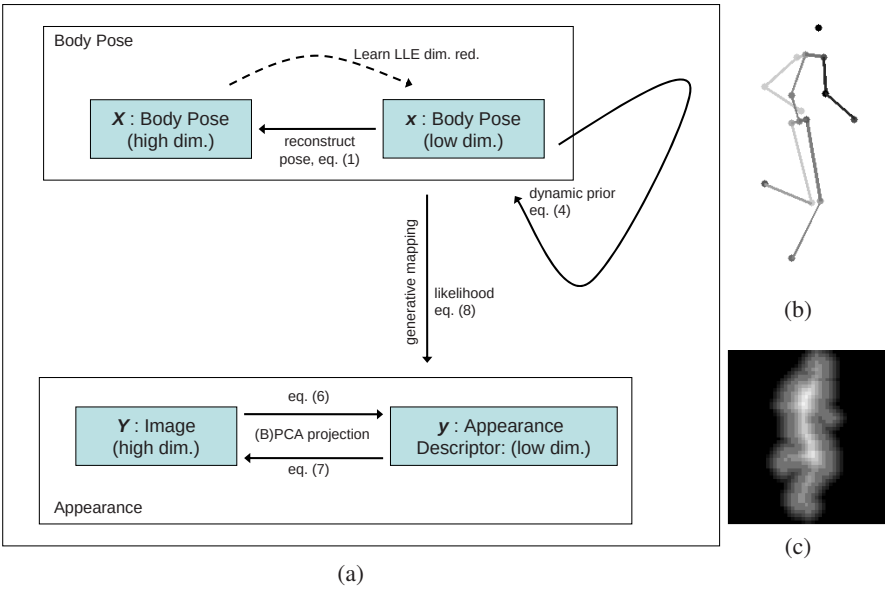


Fig. 1. a) An overview of the tracking framework. Solid arrows represent signal flow during inference, the dashed arrow stands for the nonlinear dimensionality reduction during training. The figure refers to equations in Section 2. b) Body pose representation as a number of 3d joint locations. c) Distance transformed image descriptor $dt(Y)$. Each pixel value is proportional to the distance to the silhouette, and its sign indicates whether the pixel lies inside the silhouette.

Our models were trained on real motion capture data sets of different subjects, running and walking at different speeds. Walking and running training examples were separately processed to train activity specific models.

2.1 Pose and Motion Model

Representations for the full body pose configuration are high dimensional by nature; our current representation is based on 3d joint locations of 20 body locations such as hips, knees and ankles, but any other representation (e.g. based on relative orientations between neighbouring limbs) can easily be plugged into the framework. To alleviate the difficulties of high dimensionality in both the learning and inference stages, a dimensionality reduction step identifies a low dimensional embedding of the body pose

representations. We use Locally Linear Embedding (LLE) [7], which approximately maintains the local neighbourhood relationships of each data point and allows for global deformations (e.g. unrolling) of the dataset/manifold.

LLE dimensionality reduction is performed on all poses in the data set that belong to a certain activity, and expresses each data point in a space of desired low dimensionality. However, LLE does not provide explicit mappings between the two spaces, that would allow to project new data points (that were not contained in the original data set) between them. Therefore, we model the reconstruction projection from the low-dimensional LLE space to the original pose space with a kernel regressor.

$$X = f_p(x) = W_p \Phi_p(x) \tag{1}$$

Here, X and x are the body pose representations in original resp. LLE-reduced spaces, Φ_p is a vector of kernel functions, and W_p is a sparse matrix of weights, which are learned with a Relevance Vector Machine (RVM). We use Gaussian kernel functions, computed at the training data locations. Separate models are learned for the two distinct activities, $f_p^w(x^w)$ and $f_p^r(x^r)$. In the following we will use superscripts (e.g. w for $walk$ and r for run) to indicate activity categories in the notation if necessary and omit them if the same formulation holds for all actions.

The training examples form a periodic twisted 'ring' in LLE space, with a curvature that varies with the phase within the periodic movement. A linear dynamical model, as often used in tracking applications, is not suitable to predict future poses on this curved manifold. We view the nonlinear dynamics as a regression problem, and model it using another RVM regressor, yielding the following *dynamic* prior,

$$p_d(x_t|x_{t-1}) = \mathcal{N}(x_t; x_{t-1} + f_d(x_{t-1})\Delta_T, \Sigma_d), \tag{2}$$

where $f_d(x_{t-1}) = W_d \Phi_d(x_{t-1})$ is the nonlinear mapping from poses to local velocities in LLE pose space, Δ_T is the time interval between the subsequent discrete timesteps $t-1$ and t, and Σ_d is the variance of the prediction errors of the mapping, computed on a hold-out data set that was not used for the estimation of the mapping itself. Again, the dynamics are learned separately for the different action categories.

Not all body poses that can be expressed using the LLE pose parametrisation do correspond to valid body configurations that can be reached with a human body. The motion model described so far does only include information about the temporal evolution of the pose, but no information about how likely a certain body pose is to occur in general. In other words, it does not yet provide any means to restrict our tracking to feasible body poses. The additional prior knowledge about feasible body poses, or likely poses for a given activity, is introduced as a *static* prior that is modelled with a Gaussian Mixture Model (GMM),

$$p_s(x) = \sum_c^C p_c \mathcal{N}(x; \mu_c, \Sigma_c), \tag{3}$$

with C the number of mixture components and p_c, μ_c and Σ_c the mixture proportions and parameters of the Gaussian components. The influence of this pose prior can be kept low, avoiding a distortion of the tracking results towards typical average motion.

We introduce a weighting factor $\lambda > 1$ and obtain the following formulation for the temporal prior by combination with the *dynamic* prior $p_d(x_t|x_{t-1})$.

$$p(x_t|x_{t-1}) \propto p_d(x_t|x_{t-1})\, p_s(x_t)^{\frac{1}{\lambda}} \tag{4}$$

We also want to model the transition between the considered action categories, that each have their own low dimensional pose parametrisation expressed in distinct LLE spaces. Informally, we want to find walking poses that are very similar to a given running pose and vice versa, since we know that the transition is performed smoothly, without any sudden or jerky 'jump' of the body configuration.

Given our distinct training sets of walking and running poses, two sets of training pairs are generated by looking for the most similar running pose for every walking pose and vice versa, and the nonlinear mapping between these pairs is modelled using two sparse kernel regressors $f^{r\rightarrow w}_{switch}(x^r)$ and $f^{w\rightarrow r}_{switch}(x^w)$. This can be generalised to more action categories[1] and leads to the following motion model, where the state space from eq. (4) is augmented by a discrete state variable a_t.

$$p(x_t, a_t|x_{t-1}, a_{t-1}) \propto \begin{cases} p_{noswitch} \quad p^{a_t}(x_t|x_{t-1}) & \text{if } a_t = a_{t-1} \\ p_{switch} \quad p^{a_{t-1}\rightarrow a_t}(x_t|x_{t-1}) & \text{else} \end{cases} \tag{5}$$

Here, the motion model for the case of activity switching $p^{a_{t-1}\rightarrow a_t}(x_t|x_{t-1})$ is modelled as a normal distribution around the pose predicted by the regressor $f^{a_{t-1}\rightarrow a_t}_{switch}$. The probabilities that an activity transition does or does not occur are denoted p_{switch} and $p_{noswitch}$. In the case of more than two activity categories, these transition probabilities could be represented as a transition matrix with the $p^a_{noswitch}$ of the different categories a on the diagonal.

2.2 Appearance Model

The representation of the subject's image appearance is based on a rough figure-ground segmentation. Under realistic imaging conditions, it is not possible to get a clean silhouette, therefore the image descriptor has to be robust to noisy segmentations to a certain degree. We consider two types of image descriptors, *distance transforms* $dt(Y)$ [21] of segmented figures with a subsequent linear PCA dimensionality reduction step (see Figure 1c), and a representation obtained by applying *Binary PCA (BPCA)* [22] to binary foreground images. Both image descriptors are computed from the content of a bounding box around the centroid of the figure, and 10 to 20 PCA resp. BPCA components have been found to yield good reconstructions. We introduce the following notation for the computation of these descriptors and the projection on the respective subspaces given the raw pixel image Y:

$$\begin{aligned} y_{DT} &= V(dt(Y) - \mu) \\ y_{BPCA} &= BPCA(Y) \end{aligned} \tag{6}$$

In this equation, μ and V are the mean and basis vectors obtained by PCA. $BPCA(Y)$ and $dt(Y)$ are nonlinear operations, in the BPCA case the projection on the subspace

[1] The number of transitions grows quadratically with the number of categories, which should therefore be kept low.

is done iteratively (see [22]). As we will see later, it is useful in some situation to consider the inverse operation that projects the image descriptors y_{DT} and y_{BPCA} back into high dimensional pixel space and transforms it into binary images or foreground (fg) probability maps. From the descriptors we compute probability maps via the sigmoid function $\sigma(.)$. In the case of the distance transformed descriptor this is based on the intuition that the foreground/background probabilities are higher far away from the silhouette, and lower very close to the silhouette. BPCA reconstruction is also based on the sigmoid function [22].

$$\begin{aligned} p(Y = fg|y_{DT}) &\propto \sigma(V^T y_{DT} + \mu) \\ p(Y = fg|y_{BPCA}) &\propto \sigma(V^T y_{BPCA} + \mu) \end{aligned} \tag{7}$$

Again, μ and V are the mean and basis vectors from linear resp. binary PCA.

Now that we have seen how to compute image descriptors from segmented images and back, we will look how the image appearance is linked to the LLE body pose representation x. We will model the *generative* mapping from pose x to image descriptors y that allows to predict image appearance given pose hypotheses and fits well into generative inference algorithms such as recursive Bayesian sampling. In addition to the local body pose x, the appearance depends on the global body orientation ω (rotation around vertical axis).

$$\begin{aligned} p(y|x,\omega) &= \mathcal{N}(y; f_a(x,\omega), \Sigma_a) \\ f_a(x,\omega) &= W_a \Phi_a(x,\omega) \end{aligned} \tag{8}$$

Here, the functional mapping $f_a(x,\omega)$ is approximated by a sparse kernel regressor (RVM) with weight matrix W_a and kernel functions $\Phi_a(x)$. Σ_a is the prediction variance matrix, it indicates which dimensions of the descriptor y can be well predicted and which cannot, and thus accounts for the fact that the prediction of y will always be subject to uncertainty. Σ_a is estimated from a hold-out set of the original training data and restricted to a diagonal matrix for simplicity.

3 Inferring Image Position, Orientation, Activity and Pose

In this section we will show how the 2d image position, body orientation, activity category, and body pose of the subject are simultaneously estimated given a video sequence, by using the learned models from the previous section within the framework of recursive Bayesian sampling. Both pose estimation and image localisation can benefit from the coupling of pose and image location. For example, the known current pose and motion pattern can help to distinguish subjects from each other and track them through occlusions. We therefore believe that tracking should happen jointly in the entire state space Θ,

$$\Theta_t = [a_t, \omega_t, u_t, v_t, w_t, h_t, x_t], \tag{9}$$

consisting of the discrete activity a, orientation ω, the 2d bounding box parameters (position, width and height) u, v, w, h, and the body pose x.

Despite the reduced number of pose dimensions, we face an inference problem in 10-dimensional space. Having a good sample proposal mechanism like our dynamical

model is crucial for the Bayesian recursive sampling to run efficiently with a moderate number of samples. For the monocular sequences we consider, the posteriors can be highly multimodal. Our experiments have shown that a strong dynamical model is necessary to avoid confusion between these posterior modes and reduce ambiguities. The remaining posterior multimodalities correspond to a small number of different interpretations of the images, which are all valid and feasible motion patterns.

The precise inference algorithm is very similar to classical *CONDENSATION* [5], with normalisation of the weights and resampling at each time step. If we neglect the activity switching mechanism for a moment, the prior and likelihood for our inference problem are obtained by extending (4) and (8) to the full state space Θ. In our implementation, the *dynamical* prior $p_d(\Theta_t^i|\Theta_{t-1}^i)$ serves as the sample proposal function. It consists of the learned dynamical pose prior from eq. (2), and a simple motion model for the remaining state variables $\theta = [\omega_t, u_t, v_t, w_t, h_t]$.

$$p_d(\Theta_t^i|\Theta_{t-1}^i) = p_d(x_t^i|x_{t-1}^i)\mathcal{N}(\theta_t^i; \theta_{t-1}^i, \Sigma_\theta) \tag{10}$$

In practice, one may want to use a standard autoregressive model for propagating θ, omitted here for notational simplicity. We assume statistical independence between the body pose x and the state variables θ in (10), since modelling these dependencies would imply restricted camera motions (e.g. static camera). The *static* prior over likely body poses (3) and the likelihood (8) are then used for assigning weights w^i to the samples.

$$w_t^i \propto p(y_t^i|\Theta_t^i)p_s(\Theta_t^i)^{\frac{1}{\lambda}} = p(y_t^i|x_t^i, \omega_t^i)p_s(x_t^i)^{\frac{1}{\lambda}} \tag{11}$$

Here, i is the sample index, and y_t^i is the image descriptor computed from the sampled bounding box $(u_t^i, v_t^i, w_t^i, h_t^i)$. Note that our choice for sample proposal and weighting functions differs from *CONDENSATION* in that we only use one component (p_d) of the prior (4) as a proposal function, whereas the other component (p_s) is incorporated in the weighting function.

Likelihood computation in image space or on a PCA subspace. Our framework has a generative flavour, since we model the pdf of the appearance given the body pose in a top-down manner. The computation of the image descriptor and projection on the subspace and back can be issued in both directions, as seen in eq. (6) and (7). One possibility is to compute the image descriptors in a bottom-up manner and project them onto the PCA or BPCA subspace (6), where the likelihood is then directly obtained using (8).

Alternatively, in a purely generative top-down manner, we can predict whether we expect a certain pixel to be foreground or background given a pose hypothesis. This is done by concatenating the mapping $f_a(x, \omega)$ from eq. (8) and the projection of the appearance descriptor into full appearance space (image space) (7). This yields a discrete 2d probability distribution of foreground probabilities Seg over the pixels $\mathbf{p}$ in the bounding box. From this pdf, a likelihood measure can then be derived by comparing it to the actually observed segmented image Obs, also viewed as a discrete pdf, using the Bhattacharyya similarity measure [23] which measures the affinity between distributions.

$$\begin{aligned} Seg_t^i(\mathbf{p}) =& p(\mathbf{p} = fg | f_a(x_t^i, \omega_t^i)) \\ Obs_t^i(\mathbf{p}) =& p(\mathbf{p} = fg | image_t, u_t^i, v_t^i, w_t^i, h_t^i) \\ BC_t^i =& \sum_{\mathbf{p}} \sqrt{Seg_t^i(\mathbf{p}) Obs_t^i(\mathbf{p})} \end{aligned} \tag{12}$$

Both alternative ways of likelihood computation have advantages and drawbacks. The bottom-up variant requires binary images to compute the image descriptors, whereas the top-down variant can handle continuous foreground probabilities. Often the foreground segmentation is available in the form of probability maps, and thresholding it may cause an unnecessary loss of information and yield unsatisfying results. On the other hand, evaluation of likelihood on the (B)PCA subspace can benefit from the learned variance matrix Σ_a. Also, the bottom-up computation of descriptors can be disturbed by noisy segmentations. This holds particularly for the *distance transformed* image descriptor y_{DT}. In the case of the descriptor based on BPCA, the projection on the subspace is iterative and therefore slow, which in this case reduces the attractivity of the bottom-up variant from a practical perspective. Experimentally, the combination of *distance transformed* descriptors and bottom-up descriptor computation fails when the input image segmentation is very noisy, the other three combinations perform similarly well.

3.1 Activity Switching

When turning to the multi activity tracking case, the sample proposal function is adapted according to eq. (5). A sample i undergoes an activity switch with probability p_{switch}. In our experiments, the scheme is demonstrated for two activity categories, *walking* and *running*, therefore we set $p_{switch}^{w \to r} = p_{switch}^{r \to w} = 1 - p_{noswitch}$. In case of an activity switch, the sample i is initialised with a value in LLE pose space of the new activity a_t by sampling from the activity transition function $p^{a_{t-1} \to a_t}(x_t | x_{t-1})$. In such a manner, at each time step a number of samples are generated that allow for a smooth transition into the other activity. If these hypotheses are supported by the image information, they will be selected in the subsequent resampling step and take overhand. The percentage of samples of a certain activity category is a measure for the algorithm's belief about the currently observed action. The image support for the hypotheses is given by the observation likelihood, which is always based on the action specific appearance model (f_a^w resp. f_a^r in eq. (8)).

3.2 Globally Optimal Trajectory

The described sample-based tracking algorithm provides a set of N samples with corresponding weights for each frame of the sequence. This representation of the posterior is not suitable for many purposes, even visualisation is difficult. Furthermore, the posteriors are computed on a per-frame level, i.e. at time step t we compute $p(X_t | Y_{1:t})$. Often we are interested in a consistent trajectory through the entire image sequence, i.e. in the maximum of the posterior $p(X_{1:T} | Y_{1:T})$ over the poses of *all* time steps, given

all observations. In other words, we are interested in the value for $X_{1:T}$ with maximal probability rather than marginals for each X_t.

In our framework this is achieved by a postprocessing algorithm that finds optimal paths through the set of samples. We use the Max-Product algorithm resp. its numerically more stable counterpart, the Min-Sum algorithm that operates with negative logarithms instead of probabilities (see [24] or [25] for belief propagation algorithms). These algorithms are discrete by nature, i.e. each node of the Markov chain (each time step) has a number of discrete states that in our case is equal to the number of samples N of the tracking algorithm. The algorithm will thus choose one sample per node to form a trajectory through time and state space that best satisfies both observation likelihood and temporal prior. Instead of finding the optimal trajectory for the entire sequence, the algorithm can also be applied to sub-sequences, in a sliding-window fashion.

More formally, the goal is to find a sequence of state variables $\Theta_{1:T}$ that maximises the global function $p(\Theta_{1:T})$, which is factorised into the product of *evidence* functions υ that take into account the image information, and *compatibility* functions ψ of temporally adjacent nodes.

$$p(\Theta_{1:T}) = \frac{1}{Z} \prod_{t=2}^{T} \psi(\Theta_t, \Theta_{t-1}) \prod_{t=1}^{T} \upsilon(\Theta_t), \tag{13}$$

where Z is a normalisation constant. The equations from the recursive tracking can be reused as the global function uses the same terms. The *evidence* functions $\upsilon(\Theta_t)$ are computed according to eq. (11). In fact we can directly reuse the sample weights computed during tracking. The *compatibility* between neighbouring nodes is given by eq. (10). The Max-Product resp. Min-Sum algorithm performs inference in this chain graph by propagating local messages (*beliefs*) between neighbouring nodes.

4 Experiments

Training. The described models were trained on a database of motion sequences from 6 different subjects, walking and running at 3 speeds per activity (2.5, 4.2, 6 resp. 8,10,12 km/h). The data was recorded using an optical motion capture system at a framerate of 60 Hz and subsampled to 30 Hz. The resulting sequences of body poses were normalised for limb lengths and used to animate a realistic computer graphics figure in order to create matching silhouettes for all training poses. The figure was rendered from different view points, located every 10 degrees in a circle around the figure. Due to this choice of training data, our system currently assumes that the camera is in an approximately horizontal position. The training set consists of 2000 body poses of each activity. All the kernel regressors were trained using the Relevance Vector Machine algorithm [27], with Gaussian Kernels. Different kernel widths were tested and compared using a crossvalidation set consisting of 50% of the training data, in order to avoid overfitting.

Tracking. We evaluated our tracking algorithm on a number of different sequences. The main goals were to show its ability to deal with noisy sequences with poor foreground segmentation, image sequences of very low resolution, varying viewpoints through the sequence, and switching between activities. The figures in this section show

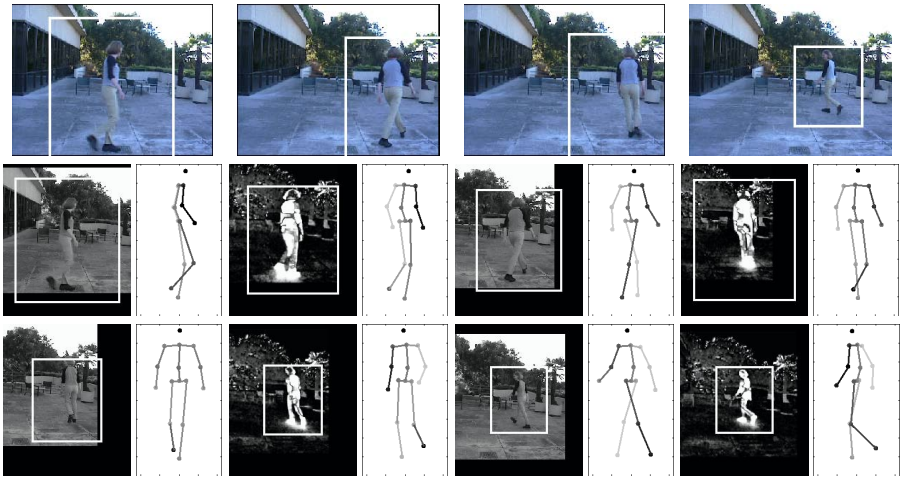


Fig. 2. Circular walking sequence from [26]. The figure shows full frames (top), and cutouts with bounding boxes in original or segmented input images and estimated poses. Darker limbs are closer in depth.

the body poses of the *optimal trajectory* that was computed according to Section 3.2, based on the samples from the recursive Bayesian sampling algorithm.

The particle filtering was performed using a set of 500 samples, leading to a computation time of approx. 2-3 seconds per image frame in unoptimised Matlab code. The sample set is initialised in the first frame as follows. Hypotheses for the 2d bounding box locations are either derived from the output of a pedestrian detector that is run on the first image, or from a simple procedure to find connected components in the segmented image. Pose hypotheses x_1^i are difficult to initialise, even manually, since the LLE parametrisation is not easily interpretable. Therefore, we randomly sample from the entire space of feasible poses in the reduced LLE space to generate the initial hypotheses. Thanks to the low-dimensional representation, this works well, and the sample set converges to a low number of clusters after a few time steps, as desired.

The first experiment (Fig. 2) shows tracking on a standard test sequence[2] from [26], where a person walks in a circle. We segmented the images using background subtraction, yielding noisy foreground probability maps. The main challenge here is the varying viewing angle that is difficult to estimate from the noisy silhouettes. Figure 3 shows another publicly available sequence[3]. Here we used only one camera, while this sequence has been mainly used for multi-camera tracking (e.g. [28,11]).

Figure 4 shows an extract from a treadmill sequence that was 1660 frames long in total. In this sequence, the subject initially walks and switches to running and back to walking several times. The figure shows a few frames from the transition from running to walking; the first two frames clearly contain running poses, then the arms are lowered

[2] http://www.nada.kth.se/ hedvig/data.html
[3] http://www.cs.brown.edu/ ls/

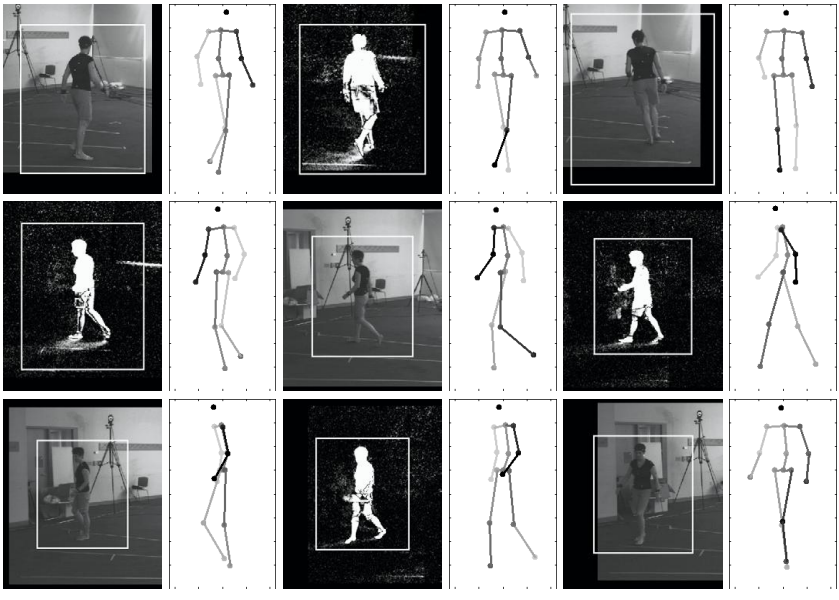


Fig. 3. Circular walking sequence from [28], original resp. segmented input images with estimated bounding boxes, and estimated poses.

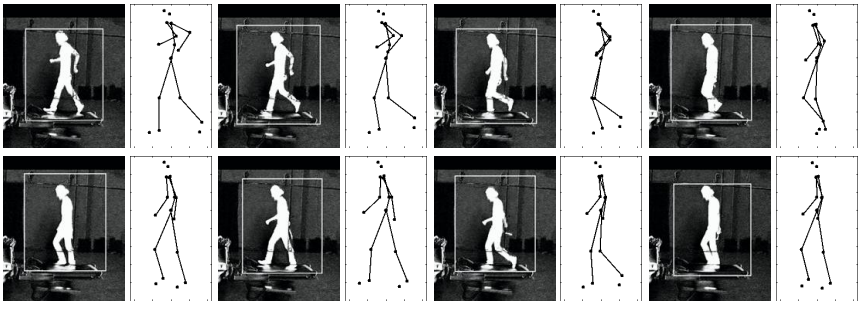


Fig. 4. Transition from running to walking. The original sequence is 1660 frames long, here we show selected frames from the transition phase between frame 921 and frame 936. See also Fig. 5a) for a plot of the estimated activity categories.

and the last 3 frames show walking. The plot in Figure 5a) shows the estimated running probabilities throughout the sequence. Even for humans, it is not obvious to identify the exact moment of activity change, there is typically a transition phase of about 0.5 seconds. In our experiments, the activity switch was always detected within this transition phase, as desired. Note that we do not take into account the typical periodic motion

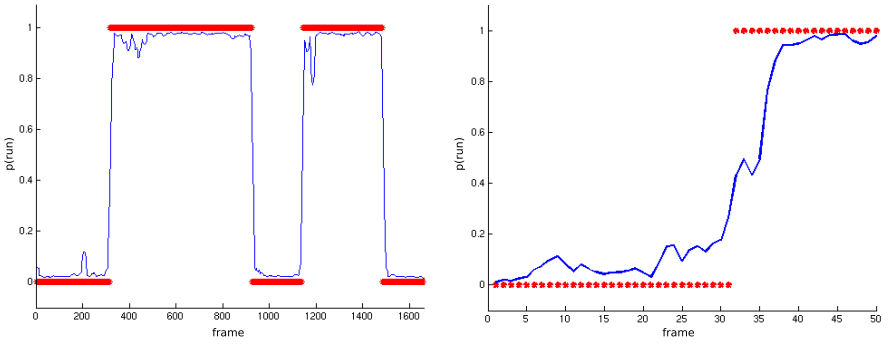


Fig. 5. Activity plots of the sequences of figures 4 (a) and 7 (b). The figures show the estimated activities; the blue curve shows the continuous probability that we observe running rather than walking over the entire sequence, the red bars/dots indicate the activity label that has been inferred by the global optimisation.

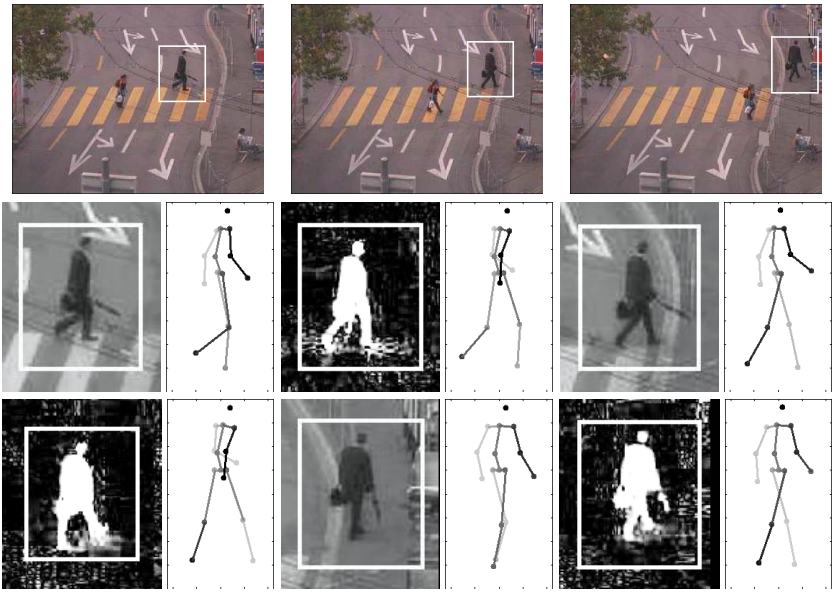


Fig. 6. Real traffic scene with low resolution input images, noisy segmentation, disturbing objects (umbrella, bag), and varying viewangle. Original frames (top) and cutouts.

in vertical direction that distinguishes running from walking, the activity is correctly estimated from the local shape and its deformation over time alone.

The sequences of Figures 6 and 7 were recorded in a real traffic environment with a webcam. The image resolution is 320 $\times$ 200 pixels, with subjects as small as 40-50

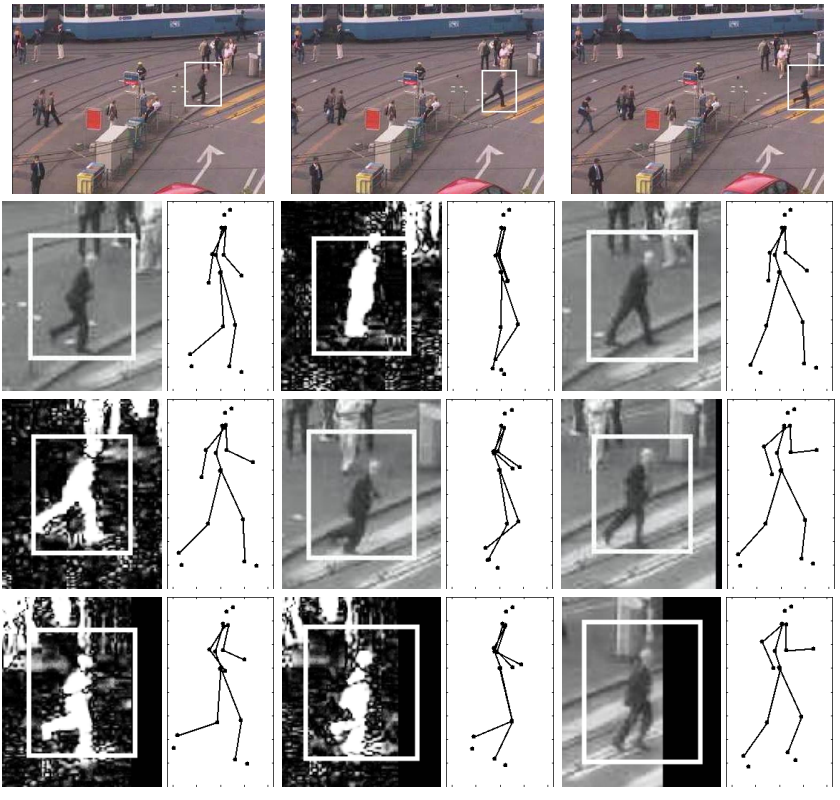


Fig. 7. Real traffic scene with a transition from walking to running. Full Frames (top) and cutouts with estimated poses. Figure 5b) shows the inferred activity categories of this sequence.

pixels in height. Furthermore, the image quality is unfavourable due to severe MPEG compression artefacts and noisy foreground segmentation. In Figure 6 the person carries an umbrella that could be misinterpreted as a leg, and a bag that distorts the overall shape of the pedestrian. The subject also turns away from the camera over the duration of the sequence. Our experiments showed that such a challenging sequence, combining different kinds of difficulties, can only be tracked thanks to the dynamical model, since the information from individual images is unreliable and therefore has to be accumulated over time. The pedestrian in Figure 7 suddenly starts to run when crossing a street. The activity switch is reliably detected, as can be seen in the activity plot in Figure 5b).

5 Summary

We presented a monocular tracking approach that simultaneously estimates the 2d bounding box coordinates, the performed activity, and the 3d body pose of a moving person. To this end, we learn statistical models of pose, dynamics, activity transition, and appearance using efficient sparse kernel regressors. The relationship of pose and

appearance is learned in a generative manner. Using LLE, we find an embedding of the pose manifolds of low dimensionality, which allows us to use a Monte-Carlo sampling algorithm for tracking. A Max-Product algorithm finally extracts the optimal sequence through the entire image sequence. We demonstrated the method on different challenging video sequences of low resolution with noisy segmentation.

Acknowledgements

This work is supported, in parts, by the EU Integrated Project DIRAC (IST-027787), the SNF project PICSEL and the SNF NCCR IM2.

References

1. Rosales, R., Sclaroff, S.: Learning body pose via specialized maps. In: NIPS (2001)
2. Thayananthan, A., Navaratnam, R., Stenger, B., Torr, P., Cipolla, R.: Multivariate relevance vector machines for tracking. In: Ninth European Conference on Computer Vision (2006)
3. Sminchisescu, C., Kanaujia, A., Li, Z., Metaxas, D.: Discriminative density propagation for 3d human motion estimation. In: CVPR (2005)
4. Agarwal, A., Triggs, B.: Monocular human motion capture with a mixture of regressors. In: IEEE Workshop on Vision for Human-Computer Interaction at CVPR, IEEE Computer Society Press, Los Alamitos (2005)
5. Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. Int. J. Computer Vision (1998)
6. Doucet, A., Godsill, S., Andrieu, C.: On sequentional monte carlo sampling methods for bayesian filtering. Statistics and Computing (2000)
7. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
8. Forsyth, D.A., Arikan, O., Ikemoto, L., O' Brien, D.R.J.: Computational studies of human motion: Part 1. Computer Graphics and Vision 1(2/3) (2006)
9. Agarwal, A., Triggs, B.: 3d human pose from silhouettes by relevance vector regression. In: CVPR (2004)
10. Grauman, K., Shakhnarovich, G., Darrel, T.: Inferring 3d structure with a statistical image-based shape model. In: ICCV (2003)
11. Sun, Y., Bray, M., Thayananthan, A., Yuanand, B., Torr, P.: Regression-based human motion capture from voxel data. In: Proceedings British Machine Vision Conference (2006)
12. Lim, H., Camps, O.I., Sznaier, M., Morariu, V.I.: Dynamic appearance modeling for human tracking. In: Conference on Computer Vision and Pattern Recognition, pp. 751–757 (2006)
13. Elgammal, A., Lee, C.S.: Inferring 3d body pose from silhouettes using activity manifold learning. In: CVPR (2004)
14. Wang, J.M., Fleet, D.J., Hertzmann, A.: Gaussian process dynamical models. Advances in Neural Information Processing Systems 18, 1441–1448 (2006)
15. Sminchisescu, C., Jepson, A.: Generative modeling for continuous non-linearly embedded visual inference. In: ICML (2004)
16. Li, R., Yang, M.H., Sclaroff, S., Tian, T.P.: Monocular tracking of 3d human motion with a coordinated mixture of factor analyzers. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 137–150. Springer, Heidelberg (2006)
17. Urtasun, R., Fleet, D.J., Fua, P.: 3d people tracking with gaussian process dynamical models. In: CVPR 2006. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 238–245. IEEE Computer Society Press, Los Alamitos (2006)

18. Jaeggli, T., Koller-Meier, E., Van Gool, L.: Monocular Tracking with a Mixture of View-Dependent Learned Models. In: Perales, F.J., Fisher, R.B. (eds.) AMDO 2006. LNCS, vol. 4069, pp. 494–503. Springer, Heidelberg (2006)
19. Isard, M., Blake, A.: A mixed-state CONDENSATION tracker with automatic model-switching. In: ICCV, pp. 107–112 (1998)
20. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science 290(5500), 2319–2323 (2000)
21. Bailey, D.G.: An efficient euclidean distance transform. In: Klette, R., Žunić, J. (eds.) IWCIA 2004. LNCS, vol. 3322, pp. 394–408. Springer, Heidelberg (2004)
22. Zivkovic, Z., Verbeek, J.: Transformation invariant component analysis for binary images. In: CVPR (1), pp. 254–259 (2006)
23. Bhattacharyya, A.: On a measure of divergence between two statistical populations defined by their probability distributions. Bull. Calcutta Math Soc. (1943)
24. Kschischang, F., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Trans. Info. Theory 47, 498–519 (2001)
25. Yedidia, J., Freeman, W., Weiss, Y.: Understanding belief propagation and its generalizations. Technical Report TR-2001-22, MERL (2002)
26. Sidenbladh, H., Black, M., Fleet, D.: Stochastic tracking of 3d human figures using 2d image motion. In: Vernon, D. (ed.) ECCV 2000. LNCS, pp. 702–718. Springer, Heidelberg (2000)
27. Tipping, M.: The relevance vector machine. In: NIPS (2000)
28. Sigal, L., Bhatia, S., Roth, S., Black, M., Isard, M.: Tracking loose-limbed people. In: CVPR (2004)

Exploiting Spatio-temporal Constraints for Robust 2D Pose Tracking*

Grégory Rogez[1,**], Ignasi Rius[2], Jesús Martínez-del-Rincón[1], and Carlos Orrite[1]

[1] Computer Vision Lab, I3A, University of Zaragoza, Spain
grogez@unizar.es
[2] Computer Vision Center, UAB, Bellaterra, Spain
irius@cvc.uab.es

Abstract. We present a Spatio-temporal 2D Models Framework (STMF) for 2D-Pose tracking. Space and time are discretized and a mixture of probabilistic "local models" is learnt associating 2D Shapes and 2D Stick Figures. Those spatio-temporal models generalize well for a particular viewpoint and state of the tracked action but some spatio-temporal discontinuities can appear along a sequence, as a direct consequence of the discretization. To overcome the problem, we propose to apply a Rao-Blackwellized Particle Filter (RBPF) in the 2D-Pose eigenspace, thus interpolating unseen data between view-based clusters. The fitness to the images of the predicted 2D-Poses is evaluated combining our STMF with spatio-temporal constraints. A robust, fast and smooth human motion tracker is obtained by tracking only the few most important dimensions of the state space and by refining deterministically with our STMF.

1 Introduction

Full-body human tracking from monocular image sequences is one of the most challenging problems of human motion analysis. Therefore, the number of difficulties related to the problem are very large. Among others, the shape and appearance of a human body in 2D images may change drastically over time due to changing lighting conditions, loose fitting clothes and background clutter. Although self-occlusion of body parts can be easily predicted in a 3D model using dynamic information from joints (or limbs) [19], they are difficult to handle if a 2D model is used. In contrast, 2D models can better handle clutter, partial occlusions, multiple people tracking and present fewer parameters: an interesting advantage for tracking purpose. They also have the great advantage over 3D models of being directly observable in the image. Johansson [7] shows

* Work supported by spanish grant TIN2006-11044 (MEyC) and FEDER.
** Funded by FPU grant AP2003-2257 from Spanish Ministry of Education.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 58–73, 2007.

that the trajectories of the 2D joints provide sufficient information to interpret the performed action. Moreover Taylor's method [17] can be used, as in [11], to estimate the 3D configuration of a body given the position of the 2D key points from a single image taken with an uncalibrated camera.

On the other hand, the main disadvantage of 2D-models is their dependance on the viewpoint. Most of the previous works are based on the fundamental assumption of "in-plane" motion [20,12]. Recently many authors have proposed a common approach consisting in discretizing the space considering a series of view-based 2D models [21,10,9,13]. This method gives some preliminary good results, but there are two main problems that need to be addressed: spatial discontinuities due to the viewpoint discretization and temporal discontinuities due to the difficulties of maintaining the dynamics of the motion when the view is tuned. It appears as a challenging problem, first to interpolate data "between views", and second to establish motion correspondences between viewpoints without considering a mapping to a complex 3D model.

In this work, we consider the walking action, but the methodology can be extended to any motion. 2D pose and 2D appearance parameters are extracted from training images of the same gait sequences seen from different viewpoints. This database is clustered following both spatial and temporal criteria: the spatial clustering is directly provided by the training views and the temporal one is obtained from the gait cycles (gait states). The resulting clusters correspond in terms of dynamics or viewpoint. A Spatio-Temporal 2D Models Framework (STMF) is then learnt by fitting a mixture of Principal Component Analysers (PPCA) to the clustered feature space.

Our approach is similar to [9] in that we integrate spatial and temporal models into a common framework, but differs in that we consider a combined transition that takes into account simultaneous state and viewpoint changes. A Probabilistic Transition Matrix (PTM) is evaluated frame to frame and provides the transition probability combining both spatial and temporal constraints. A feature space "reduction" can then be performed by weighting all the local models by their respective probability.

To overcome the problem of discontinuity, inherent when using a discretized space, we track the parameters of our 2D Pose model at each time step by means of a Rao-Blackwellized Particle Filter [3]. We thus marginalize the viewpoint parameter from the state and compute it deterministically, imposing a strong restriction of continuity between viewpoints of consecutive frames. On the first hand, the STMF is used within this model-based tracking framework to evaluate the fitness of the predicted postures given the input images. On the other hand, in order to deal with the computational cost inherent to high-dimensional Particle Filters, we track a lower dimensional version of the 2D pose space, and subsequently refine the estimations from the particle filtering stage by performing a local deterministic optimization within the STMF. As a result, this work addresses the problem of using this constraint-based evolution through the 2D-Models framework jointly with a particle filter in order to obtain a more robust and smoother 2D pose tracker.

2 Probabilistic Modelling

2.1 Spatio-temporal 2D-Models Framework

The methodology we propose is generalizable to any 2D representation [20,12] and any kind of motion: 2D appearance parameters of the same sequences seen from different views are associated to the corresponding 2D joints.

As in [21], we extracted precise training human Shapes from the CMU MoBo dataset considering 15 subjects, two walking speeds (fast and slow) and 8 different viewpoints uniformly distributed between 0 and 2π. The first 50 frames of each sequence were considered in order to capture at least one gait cycle per subject. For each sequence, we hand-labelled 5 views and, using the periodicity and symmetry of human walking, we interpolated the last 3 ones. Simultaneously, we labelled 13 fundamental 2D joints that parameterize a Stick model. By this process we generated a training database encompassing 12000 Shape vectors and the corresponding Skeleton vectors (1500 pair of vectors for each different viewpoint). Following the methodology proposed in [1], a pedestrian model is then constructed encapsulating within a Point Distribution Model (PDM) 2D Shape landmarks and 2D Skeleton joints. Shapes, normalized to 100 landmarks, $\mathbf{s}_i = [x_{s1}, ..., x_{s100}, y_{s1}, ..., y_{s100}]$ and Skeletons $\mathbf{k}_i = [x_{k1}, ..., x_{k13}, y_{k1}, ..., y_{k13}]$ are then concatenated into a Shape-Skeleton vector $\mathbf{v}_i = [\mathbf{s}_i, \mathbf{k}_i]$. Both 2D Poses and 2D Shapes belong to the same coordinate space so that no scaling issue is encountered.

The complete database is now concatenated (8 views together) and PCA is applied to offer a more parsimonious description of the data, obtaining the Eigenspace $\mathcal{A}$:

$$\mathbf{v} \simeq \bar{\mathbf{v}} + \Phi_{\mathrm{v}}\mathbf{a} \tag{1}$$

where $\bar{\mathbf{v}}$ is mean Shape-Skeleton vector, Φ_{v} is the matrix of Eigenvectors and $\mathbf{a}$ is the projection in $\mathcal{A}$.

A series of local dynamic motion models is learnt by clustering the structural parameters subspace. The gait cycle is then divided into 6 basic steps (see [14] for details), providing the temporal clusters C_j, while the 8 training views directly provide the spatial clustering (clusters R_r). The different clusters correspond in terms of dynamics or viewpoint. Using this structure-based partitioning and the correspondences between training viewpoints, 48 spatio-temporal clusters $\{\{T_{j,r} = C_j \cap R_r\}_{j=1}^{6}\}_{r=1}^{8}$ are obtained.

Thus, following [18,4], a local linear model is learnt for each spatio-temporal cluster $T_{j,r}$ and a mixture of PPCA is fitted to the clustered $\mathcal{A}$ space, obtaining $\mathbb{T}$, i.e. our Spatio-Temporal 2D Models Framework (STMF). Parameters for the 48 Gaussian mixture models components are determined using EM algorithm. The prior Shape-Skeleton model probability is then expressed as:

$$p(\mathbf{a}) = \sum_{j,r} p(\mathbf{a}|\, T_{j,r}) \;\; p(\, T_{j,r}), \tag{2}$$

where $\mathbf{a}$ is the projection in $\mathcal{A}$ of the Shape-Skeleton vector, $p(\mathbf{a}|\, T_{j,r})$ is a single PPCA, and $p(\, T_{j,r})$ is the mixing parameter corresponding to $T_{j,r}$.

This prior probability provides an indication on how well a Shape **s** and Skeleton **k** fit together; however an efficient search method is required. In that way, temporal and spatial constraints will be considered to constrain the evolution through the STMF along a sequence and define a stronger prior.

2.2 Constraint-Based Search Through the STMF

At each time step, only part of the STMF is useful and a smart utilization requires a feature space reduction. The purpose is to obtain a transition probability between clusters (models) combining both spatial and temporal constraints.

Markov Chain for Modelling Temporal Constraint. Following the standard formulation of probabilistic motion model [16], if we partition the state space into N clusters $\mathcal{C} = \{C_1, ..., C_N\}$, the conditional probability $p(C_j^t | C_k^{t-1})$ corresponds to the probability of being in C_j at time t conditional on being in C_k at time *t-1* (using Markov Chain).

Modelling Spatial Constraint. In [13], a spatial prior $p(\theta_t|\theta_{t-1})$ was introduced for modelling spatial constraint. It expresses the statement that θ_t, the current orientation of the subject (viewpoint), can be predicted given the previous one. We follow the same procedure and consider a reasonable approach making this probability a normal distribution with expected value equal to the previous viewpoint $\theta_{t\text{-}1}$ and, variance σ_θ calculated as a function of the sampling rate: $p(\theta_t|\theta_{t-1}) \sim \mathcal{N}(\theta_{t\text{-}1}, \sigma_\theta)$. In this approach, the continuous values of all possible viewpoints θ are discretized into M training views $\{\theta_1, ..., \theta_M\}$ corresponding to M clusters $\mathcal{R} = \{R_1, ..., R_M\}$. In other words, the probability of going from cluster s to cluster r is given by: $p(R_r^t | R_s^{t\text{-}1}) = \mathcal{N}(\theta_r, \theta_{t\text{-}1}, \sigma)$ where R_r^t is the spatial cluster at time t, $R_s^{t\text{-}1}$ at time *t-1*. $\theta_{t\text{-}1}$ is the previous viewpoint and θ_r viewpoint of cluster R_r.

Combining Temporal and Spatial Constraints. Let T be the NxM matrix whose columns and rows represent respectively the N temporal and the M spatial clusters. Thus the conditional spatio-temporal transition probability is defined as $p(T_{j,r}^t | T_{k,s}^{t-1}) = p(T_{j,r}^t | C_k^{t-1}, R_s^{t-1})$, probability of being in C_j and R_r at time t conditional on the preceding spatio-temporal cluster. In this paper, we assume that the two considered events, state and direction changes, are independent. This leads to the following equation:

$$p(T_{j,r}^t | T_{k,s}^{t-1}) \propto p(C_j^t | C_k^{t-1}).p(R_r^t | R_s^{t\text{-}1}). \tag{3}$$

The resulting NxM matrix is the Probabilistic Transition Matrix (PTM) that gives, at each time step, the probability density function that limits the region of interest in the STMF. Considering the cyclic nature of the walking action and the circular distribution of the training viewpoints, we can observe that the resulting PTM is a toroidal matrix (Fig. 1) whose lines correspond to the M view-based gait manifold. To compute this PTM, we need both previous pose and orientation. This viewpoint could be evaluated by estimating the direction of motion in the image. In this work, it will be inferred from the 2D pose.

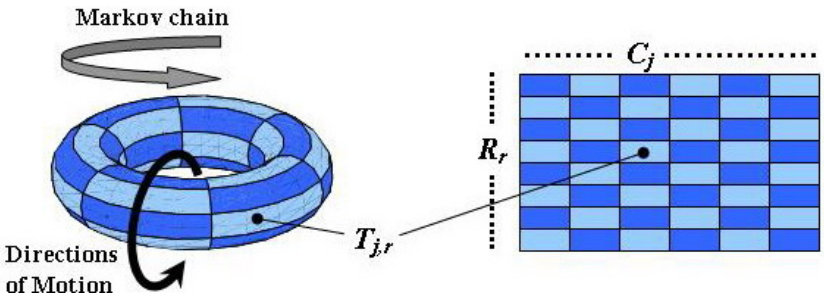


Fig. 1. 3D and 2D representations of the toroidal PTM

Applying the Markov State assumption ($p(\mathbf{a}^t|\, T^t_{j,r}, T^{t-1}_{k,s}) \sim p(\mathbf{a}^t|\, T^t_{j,r})$), the posterior $p(\mathbf{a}|\, T^{t-1}_{k,s})$ becomes:

$$\begin{aligned} p(\mathbf{a}^t|\, T^{t-1}_{k,s}) &= \textstyle\sum_{j,r} p(\mathbf{a}^t|\, T^t_{j,r}, T^{t-1}_{k,s})\ \ p(T^t_{j,r}|\, T^{t-1}_{k,s}) \\ &= \textstyle\sum_{j,r} p(\mathbf{a}^t|\, T^t_{j,r})\ \ p(T^t_{j,r}|\, T^{t-1}_{k,s}). \end{aligned} \tag{4}$$

In this way, we re-weigh the mixture at each time step t, by introducing the spatio-temporal constraints and consider the PTM elements as new mixing parameters. We thus redefine a new STMF $\mathbb{T}_t$ at each time step t and transcribe it in this new notation for describing the probability:

$$p_{\mathbb{T}_t}(\mathbf{a}^t) = \sum_{j,r} p(\mathbf{a}^t|\, T^t_{j,r})\ \ p(T^t_{j,r}|\, T^{t-1}_{k,s}). \tag{5}$$

In [4], Grauman inferred 3D structure from multi-view contour. In the same way, when presented a new Shape, we treat the unknown 2D structure as missing variables and find the maximum *a posteriori* (MAP) estimate of both contour and structure parameters, based on our prior Shape-Skeleton model $\mathbb{T}_t$. We propose to use this constraint-based search through the STMF jointly with a particle filter in order to obtain a more robust and smoother 2D pose tracker.

3 2D Pose Tracking

The discretization of the space causes spatial discontinuities that affect the smoothness of the results. Hence, MAP estimates of 2D Poses along a sequence are logically orientated in one of the 8 views of the discretized set of training viewpoints, while the 2D poses taken from ground truth are orientated in all the possible directions (See Fig.3).

3.1 2D Pose Eigenspace

Given the high dimensionality and the non-linearity of the $\mathcal{A}$ space, a second Eigenspace $\mathcal{B}$ is considered for 2D Pose tracking. All the training 2D Poses are

thus projected in a common space, the 8 views together, and PCA is applied. As a result, we obtain for each training pose $\mathbf{k}$ a related $\mathbf{b}$ such that: $\mathbf{k} \simeq \bar{\mathbf{k}} + \Phi_k \mathbf{b}$. The pose eigenspace $\mathcal{B}$ is highly non-linear in the first modes that capture the information of viewpoint and state variations: we can observe in Fig.3(b), a view-based grouping of the clusters that are globally organized as a Torus.

Although the 2D pose Eigenspace has been constructed with a discrete dataset, we can check that it is continuous: the virtual poses generated linearly between synchronized training data, at arbitrary angles, are valid (Fig.2). The linear interpolation between view-based clusters also provides valid Shape-Skeleton associations. This makes possible a smooth tracking through $\mathcal{B}$ space.

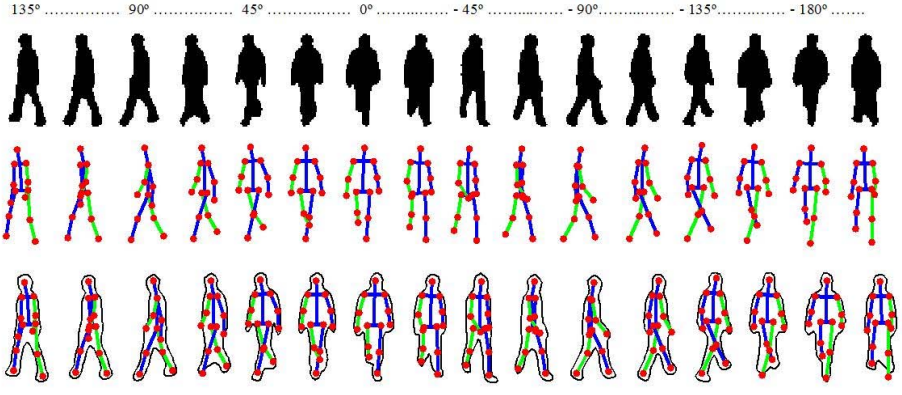


Fig. 2. Data interpolation: (up) Silhouettes $\mathbf{s}_i$, (center) Skeletons $\mathbf{k}_i$ and (down) Shape-Skeleton combination $\mathbf{v}_i$. Data at 0, π, $\pm\frac{\pi}{4}$, $\pm\frac{\pi}{2}$ and $\pm\frac{3\pi}{4}$ are taken from training set while the rest are linearly interpolated.

Viewpoint Inference. Considering that people are able to accurately estimate the orientation of a 2D pose, and that ambiguities can be solved from dynamics, we propose to learn a mapping between pose parameters and the viewpoint parameter. Following the idea of a toroidal manifold for joint view and state representation, we learn a non-linear mapping between our 2D Poses training set and a circle used as a representation of the viewpoint θ as in [10]. The mapping function f_θ relates θ and $\mathbf{k}$ such that: $\theta = f_\theta(\mathbf{k})$.

We have shown that valid 2D poses exist between training viewpoints. Thus we propose to solve the spatial discontinuities of the Framework and avoid the jumps between view-based clusters (Fig.3 b-c) by applying a tracking algorithm in the $\mathcal{B}$ space.

3.2 Particle Filtering

As mentioned before we consider a Particle Filter[2,12,16,19] to track the parameters of our 2D Pose Model. We formulate the tracking problem as a Bayesian

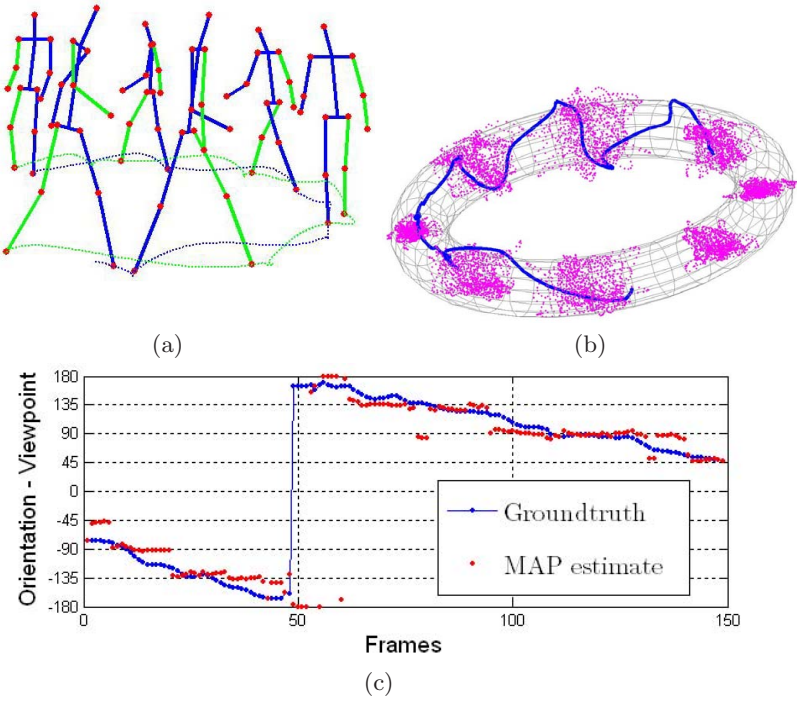


Fig. 3. Viewpoint Inference for the "WalkCircle" sequence (from www.nada.kth.se/∼hedvig): (a) Groundtruth 2D Poses, (b) torus embedding training data and groundtruth trajectory. (c) Viewpoint evaluation by mapping for MAP estimate and groundtruth 2D Poses.

inference task, where the state of the tracked object is recursively estimated at each time step given the evidences (image data) up to that moment.

Formally, within the Bayesian filtering framework, we formulate the computation of the *posterior* distribution $p(\phi_t|\mathbf{I}_t)$ of our model parameters ϕ_t over time as follows:

$$p(\phi_t|\mathbf{I}_t) \propto p(I_t|\phi_t)\, p(\phi_t|\mathbf{I}_{t\text{-}1}), \tag{6}$$

where $\mathbf{I}_t$ is the image sequence up to time t and $p(I_t|\phi_t)$ is the *likelihood* of observing the image I_t given the parametrization ϕ_t of our model at time t, in other words the *observation density*. Finally $p(\phi_t|\mathbf{I}_{t\text{-}1})$ is the *a priori density*, result of applying the *dynamic model* $p(\phi_t|\phi_{t-1})$ to the *a posteriori density* $p(\phi_{t\text{-}1}|\mathbf{I}_{t\text{-}1})$ of the previous time step:

$$p(\phi_t|\mathbf{I}_{t\text{-}1}) = \int p(\phi_t|\phi_{t\text{-}1})\, p(\phi_{t\text{-}1}|\mathbf{I}_{t\text{-}1})\, d\phi_{t\text{-}1}. \tag{7}$$

Unfortunately, when the involved distributions are non-Gaussian, Eq. (6) cannot be solved analytically. Instead, we use a Particle Filter (PF) in order to

approximate the true *posterior* pdf $p(\phi_t|\mathbf{I}_t)$ by means of a discrete weighted set of samples.

Hence, whilst the likelihood function decides which particles are worth to propagate, the dynamic model is responsible for guiding the exploration of the space of solutions. The *posterior* $p(\phi_t|\mathbf{I}_t)$ represents all the current knowledge about the model state we have extracted from image measurements. The state ϕ_t at a particular time step will be estimated by computing the expectation of the posterior pdf, i.e. $\phi_t = \mathcal{E}[\phi_t] = \sum_{n=1}^{N} \overline{\pi}_t^n \phi_t^n$, where $\overline{\pi}_t^n$ stands for the normalized weight assigned to each particle n.

Rao-Blackwellization. A classical PF would produce temporal consistency compared with an approach without tracking. But when applied in $\mathcal{B}$ space it would not avoid the jumps between training viewpoints because of the clustered nature of the space. We propose to marginalize the viewpoint parameter from the state and compute it deterministically, imposing a strong restriction of continuity between consecutive viewpoints. Rao-Blackwellization technique allows marginalizing out some of the variables and refers to integrate out part of the state analytically when using Particle Filter [3]. Rao-Blackwellized Particle Filter (RBPF) has been applied to reduce dimensionality. In our case, we propose to use it for imposing continuity to our state estimation in a discrete space.

The state of the subject at time t is defined to be:

$$\mathbf{x}_t = [\mathbf{b}_t \; \dot{\mathbf{b}}_t], \quad \mathbf{b}_t, \dot{\mathbf{b}}_t \in \mathbb{R}^D, \tag{8}$$

where $\mathbf{b}_t$ is the projection in $\mathcal{B}$ of the 2D Pose $\mathbf{k}_t$ at time t, and $\dot{\mathbf{b}}_t$ is the velocity. We thus define the extended state as:

$$\phi_t = [\mathbf{x}_t \; \theta_t], \quad \theta_t \in [0 \; 2\pi]. \tag{9}$$

in which we include θ_t, the orientation of the subject at time t (or viewpoint), that will be marginalized out and determined analytically from $\mathbf{k}_t$ by mapping.

The Bayes filter now becomes:

$$p(\mathbf{x}_t, \theta_t|\mathbf{I}_t) \propto p(I_t|\mathbf{x}_t, \theta_t) \times \int_{\mathbf{x}_{t\text{-}1}} \int_{\theta_{t\text{-}1}} p(\mathbf{x}_t, \theta_t|\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1})\, p(\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}|\mathbf{I}_{t\text{-}1}). \tag{10}$$

Integrating out the viewpoint part θ_t of the state, we obtain the marginal filter for the non-extended state $\mathbf{x}_t$:

$$p(\mathbf{x}_t|\mathbf{I}_t) \propto \int_{\theta_t} p(I_t|\mathbf{x}_t, \theta_t) \times \int_{\mathbf{x}_{t\text{-}1}} \int_{\theta_{t\text{-}1}} p(\mathbf{x}_t, \theta_t|\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1})\, p(\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}|\mathbf{I}_{t\text{-}1}). \tag{11}$$

Posterior Density. As in [8], the posterior $p(\phi_{t\text{-}1}|\mathbf{I}_{t\text{-}1})$ over the previous extended state is now approximated by a set of particles $\{\mathbf{x}_{t\text{-}1}^{(n)}, \pi_{t\text{-}1}^{(n)}, \alpha_{t\text{-}1}^{(n)}(\theta_{t\text{-}1})\}_{n=1}^{N}$:

$$\begin{aligned} p(\phi_{t\text{-}1}|\mathbf{I}_{t\text{-}1}) &= p(\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}|\mathbf{I}_{t\text{-}1}) \\ &= p(\mathbf{x}_{t\text{-}1}|\mathbf{I}_{t\text{-}1}) p(\theta_{t\text{-}1}|\mathbf{x}_{t\text{-}1}, \mathbf{I}_{t\text{-}1}) \\ &\approx \textstyle\sum_{n=1}^{N} \pi_{t\text{-}1}^{(n)} \delta(\mathbf{x}_{t\text{-}1}^{(n)}) \alpha_{t\text{-}1}^{(n)}(\theta_{t\text{-}1}), \end{aligned} \tag{12}$$

where for each particle, δ denotes the Dirac delta, $\pi_{t\text{-}1}^{(n)}$ the importance weight and $\alpha_{t\text{-}1}^{(n)}(\theta_{t\text{-}1})$ is the conditional distribution over the viewpoint, defined as the density on $\theta_{t\text{-}1}$:

$$\alpha_{t\text{-}1}^{(n)}(\theta_{t\text{-}1}) = p(\theta_{t\text{-}1}|\mathbf{x}_{t\text{-}1}^{(n)}, \mathbf{I}_{t\text{-}1}) \tag{13}$$

Dynamic Model. By the chain rule of probability, the dynamic model $p(\phi_t^n|\phi_{t\text{-}1}^n)$ becomes:

$$\begin{aligned} p(\phi_t^n|\phi_{t\text{-}1}^n) &= p(\mathbf{x}_t, \theta_t|\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}) \\ &= p(\theta_t|\mathbf{x}_t, \mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}) p(\mathbf{x}_t|\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}) \\ &= p(\theta_t|\mathbf{x}_t, \mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}) p(\mathbf{x}_t|\mathbf{x}_{t\text{-}1}), \end{aligned} \tag{14}$$

making the assumption that motion model does not depend on the previous viewpoint $\theta_{t\text{-}1}$: $p(\mathbf{x}_t|\mathbf{x}_{t\text{-}1}, \theta_{t\text{-}1}) = p(\mathbf{x}_t|\mathbf{x}_{t\text{-}1})$.

Rao-Blackwellized PF. Substituting both the dynamic model and the approximated posterior density into the expression for the marginal filter (11), we obtain after some manipulations the following approximation:

$$p(\mathbf{x}_t|\mathbf{I}_t) \propto \sum_{n=1}^{N} \pi_{t\text{-}1}^{(n)} p(\mathbf{x}_t|\mathbf{x}_{t\text{-}1}^{(n)}) \times \int_{\theta_t} p(I_t|\mathbf{x}_t, \theta_t) \int_{\theta_{t\text{-}1}} p(\theta_t|\mathbf{x}_t, \mathbf{x}_{t\text{-}1}^{(n)}, \theta_{t\text{-}1}) \alpha_{t\text{-}1}^{(n)}(\theta_{t\text{-}1}). \tag{15}$$

Image Measurements - Observation Density. The *likelihood* function $p(I_t|\phi_t^n)$ computes how likely is to observe the image I_t given a particle ϕ_t^n. Postures which are not likely to appear during the performance of a particular action should return a poor *likelihood* and disappear through time. For each frame I_t, we compute the Shape $\mathbf{s}_t^*$ from the image and for each particle n we reconstruct the 2D Pose $\mathbf{k}_t^n \simeq \bar{\mathbf{k}} + \Phi_k \mathbf{b}_t^n$.The *likelihood* function thus becomes:

$$p(I_t|\phi_t^n) = p(I_t|\mathbf{x}_t^n, \theta_t^n) = p(\mathbf{s}_t^*|\mathbf{k}_t^n, \theta_t^n). \tag{16}$$

Given this 2D Pose reconstruction and the Shape extracted from the image I_t, we then define for each particle n the Shape-Skeleton association $\mathbf{a}_t^n = {\Phi_v}^{-1}([\mathbf{s}_t^*, \mathbf{k}_t^n] - \bar{\mathbf{v}})$, and propose to use our STMF $\mathbb{T}_t$ for computing the *likelihood* function:

$$p(I_t|\mathbf{x}_t^n, \theta_t^n) \approx p_{\mathbb{T}_t}(\mathbf{a}_t^n), \tag{17}$$

where $p_{\mathbb{T}_t}(\mathbf{a}_t^n)$ is the mixture of PPCA at time t defined in Eq.5. Basically, a vector $\mathbf{a}$ that correctly combines Shape and Skeleton would fall in a cluster or between two consecutive clusters (interpolation), getting one or two highly valued $p(\mathbf{a}_t^n|\, T_{j,r}^t)$ terms and consequently a high likelihood. On the contrary, a bad Shape-Skeleton association would produce a vector $\mathbf{a}$ far from all the clusters and all $p(\mathbf{a}_t^n|\, T_{j,r}^t)$ terms would be low, obtaining a bad likelihood.

Analytical Update. During this "Extra Step", as called in [8], the posterior density $\alpha_t(\theta_t)$ on the current viewpoint conditioned on the chosen $\widehat{\mathbf{x}}_t^{(m)}$ is calculated for each new sampled particle m:

$$\alpha_t^{(m)}(\theta_t) = \omega_t^{(m)} p(I_t|\mathbf{x}_t^{(m)}, \theta_t^{(m)}) \times \int_{\theta_{t\text{-}1}} p(\theta_t^{(m)}|\widehat{\mathbf{x}}_t^{(m)}, \mathbf{x}_{t\text{-}1}^{(n)}, \theta_{t\text{-}1}) \alpha_{t\text{-}1}^{(n)}(\theta_{t\text{-}1}), \tag{18}$$

where $\omega_t^{(m)}$ is the normalization constant. Assuming that θ_t changes smoothly over time following a Gaussian process, the result of the integral in 18 is a normal centered on the previous viewpoint $\mathcal{N}(\theta_{t\text{-}1}, \sigma_\theta)$. In that way $\alpha_t^{(m)}(\theta_t)$ becomes:

$$\alpha_t^{(m)}(\theta_t) = \omega_t^{(m)} p(I_t|\mathbf{x}_t^{(m)}, \theta_t^{(m)}) \times exp(-\frac{1}{2}\|\theta_t^{(m)} - \widehat{\theta}_{t\text{-}1}\|_{\sigma_\theta}). \tag{19}$$

Importance Weights. The importance weight is finally derived combining measurement likelihood and analytical update:

$$\pi_t^{(m)} = \int_{\theta_t} p_{\mathbb{T}_t}(\mathbf{a}_t^n)\ exp(-\frac{1}{2}\|\theta_t^{(m)} - \widehat{\theta}_{t\text{-}1}\|_{\sigma_\theta}). \tag{20}$$

Since $p_{\mathbb{T}_t}(\mathbf{a}_t^n) = p(\mathbf{a}_t^n | T_{k,s}^{t-1})$ does not depend on θ_t and $\int_x q(x) = q(\mu)\sqrt{|2\pi\sigma|}$ for any function $q(x) = kexp(-\frac{1}{2})\|\mu - x\|_\sigma$, we obtain the final expression:

$$\pi_t^{(m)} = p_{\mathbb{T}_t}(\mathbf{a}_t^n)\ exp(-\frac{1}{2}\|\widehat{\theta}_t^{(m)} - \widehat{\theta}_{t\text{-}1}\|_{\sigma_\theta}). \tag{21}$$

Because of the viewpoint discretization inherent to our model, the first term gives a higher importance to the particles that fall inside the clusters compared with the ones that fall in the space in between. But this is balanced by the continuous spatial constraint imposed by the second term that gives more importance to a "correctly-orientated" Pose whose particle fell in the space in between two view-based clusters.

3.3 Deterministic Sample Optimization

A well known disadvantage of particle filtering methods for object tracking is that they are typically much slower than deterministic local optimization techniques since the number of required particles grows up exponentially as the number of dimensions of parameters spaces does. Unfortunately, fewer samples decrease the performance of the filters. To cope with this, we consider fewer dimensions of the state space to be explored by the Particle Filter (PF), and then refine the result by searching deterministically using the STMF. As demonstrated before only the first few modes of the $\mathcal{B}$ space contain most of the spatio-temporal information required for tracking, while the rest of the dimensions provide details such as morphological information or little variations in joints localization. Therefore, a "coarse to fine" search is proposed, combining a stochastic search in the first modes of $\mathcal{B}$ space with a deterministic optimization technique used for refinement in last dimensions, both integrated into a Particle Filter.

In that way, if we search stochastically in the first d modes and deterministically in the other ones, for each particle n we can reconstruct Ψ_t^n, a Pose in eigen space $\mathcal{B}$, whose d first components are provided by ϕ_t^n and (D-d) last components are taken from the deterministic evaluation $\mathbf{k}^{\text{map}}$ (cf Sec. 3.1):

$$\Psi_t^n = [\phi_t^n(1), ...\phi_t^n(d), \mathbf{b}_t^{\text{map}}(d{+}1), ..., \mathbf{b}_t^{\text{map}}(D)]. \tag{22}$$

In that way, the final state would be estimated by computing the expectation of the posterior pdf:

$$\hat{\Psi} = \mathcal{E}[\Psi_t] = \sum_{n=1}^{N} \overline{\pi}_t^n \Psi_t^n. \tag{23}$$

The estimated 2D Pose $\hat{\mathbf{k}}_t$ can be reconstructed by back-projecting $\hat{\Psi}$ from $\mathcal{B}$ space to 2D Pose space. We therefore pretend tracking the 26 parameters of our 2D model by estimating only the few first dimensions of $\mathcal{B}$ space.

4 Experiments

Testing Data. In this section, the total tracking framework is evaluated with two testing sequences that present extensive viewpoint changes. Sequences from Sidenbladh & Black [16] have appeared to become some references for testing and evaluation of tracking algorithms [21,9,19]. The "Walkcircle" sequence, maybe the most challenging one in terms of self-occlusions and viewpoint changes, has been selected. A second sequence from HumanEVA dataset [5] that presents the same conditions in an indoor environment has also been selected for testing.

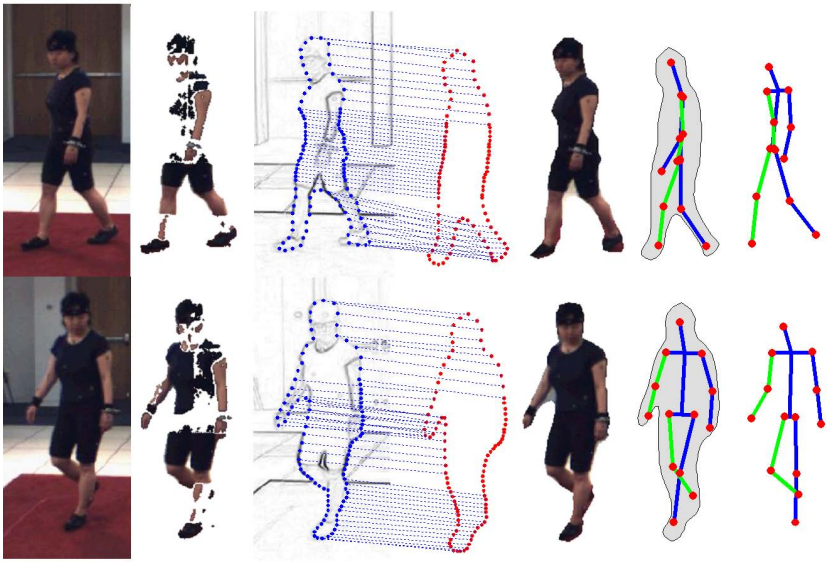


Fig. 4. Silhouette Extraction: (a) Original image. (b) Segmentation obtained by background subtraction. (c) Corresponding points between edge map of (a) and landmarks of the estimated Shape (d). (e) Segmentation Result. (f) MAP estimates of Shape and Pose. (g) Groundtruth Pose.

Segmentation - Silhouette Extraction. We use our 2D-models framework to extract reliable silhouettes required for our tracking process. We first calculate

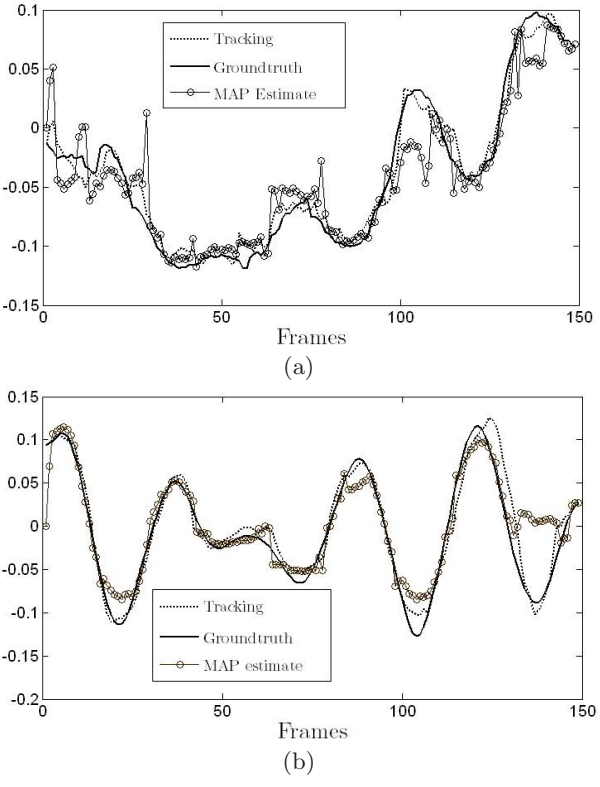


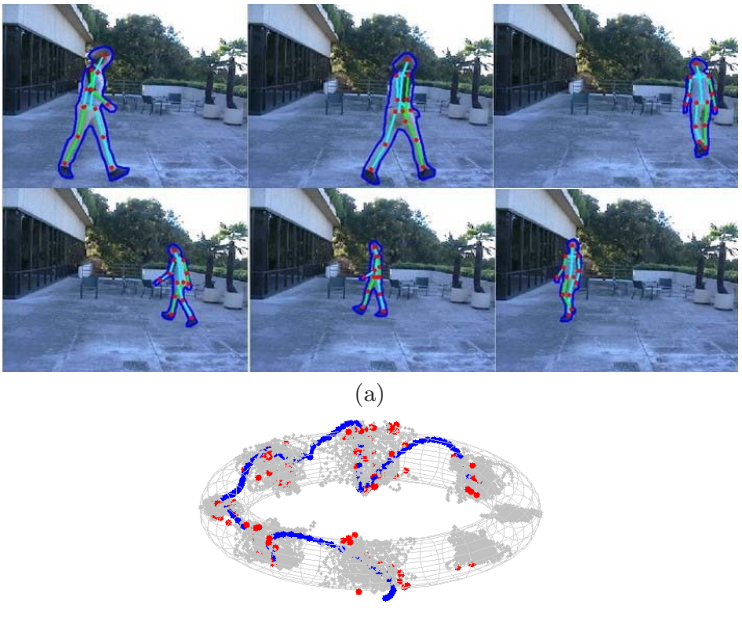
Fig. 5. Tracked positions: 1^{st} (a) and 2^{nd} (b) dimensions of the 2D Pose Eigenspace. Tracking results for "Walkcircle" sequences are compared to MAP estimates and groundtruth at each time step.

the edge image and combine it with the dilated foreground region (obtained by background subtraction) to select only valuable features. Given an initial estimate for shape parameters (i.e. mean Shape or previous Shape), the goal is to move each landmark and find the position that best matches it with the image features. We constrain the search along the normals to the shape at each boundary point and calculate a global optimal solution by using Dynamic Programming. The solution results in a smooth extracted contour that is corrected using the selected view-based Shape-Skeleton models, as explained in [13]. This corrected Shape is then warped to the extracted one resulting in a first estimation of the human silhouette. This complete process is repeated until convergence updating at each iteration the foreground blob by decreasing/increasing the detection threshold inside/outside the warped shape. More details about Segmentation stage are given in [15]. We provide the tracking framework with the normalized human Shapes that has been extracted.

2D Pose Tracking. The process begins with a manual initialization: indicating the adequate model and the viewpoint in the first frame. We project forward the

particle set $\{\phi_{t-1}^n\}$ by propagating the 2D Pose through the $\mathcal{B}$ space following a 1st order motion model plus some Gaussian noise (cf [6]). We deliberately select a "basic"/generic motion model since it does not apply any restriction on possible solutions and allows tracking more kinds of motion even if at this moment we only consider the gait action.

The best compromise between particles number and accuracy has been obtained when applying the tracking in the first three modes with 500 particles (and optimizing deterministically in the rest of the space). The results presented in this paper have been obtained with this configuration but note that others promising trials have been made by tracking only the two first modes with only 200 particles. In figure 5 we demonstrate the robustness of our algorithm to viewpoint changes: the tracking performs better than the MAP estimate in terms of robustness and precision. Fig.6 and Fig.7 shows some frames, silhouettes and estimated 2D Poses for the 2 testing sequences.



(a)

(b)

Fig. 6. "Walkcircle" Sequence. (a) Captured images (frames 3, 22, 59, 88, 117 and 145), extracted Shapes and inferred 2D-joints location (Skeletons). (b) Torus Embedding: the trajectory obtained by our tracking algorithm is represented by a solid blue line while the MAP estimates are represented by circular red markers.

However the most important improvement is definitely the smoothness achieved when visualizing the sequence of estimated Poses. In Fig.6 (torus embedding), we can observe how the viewpoint inferred from the tracked 2D Pose

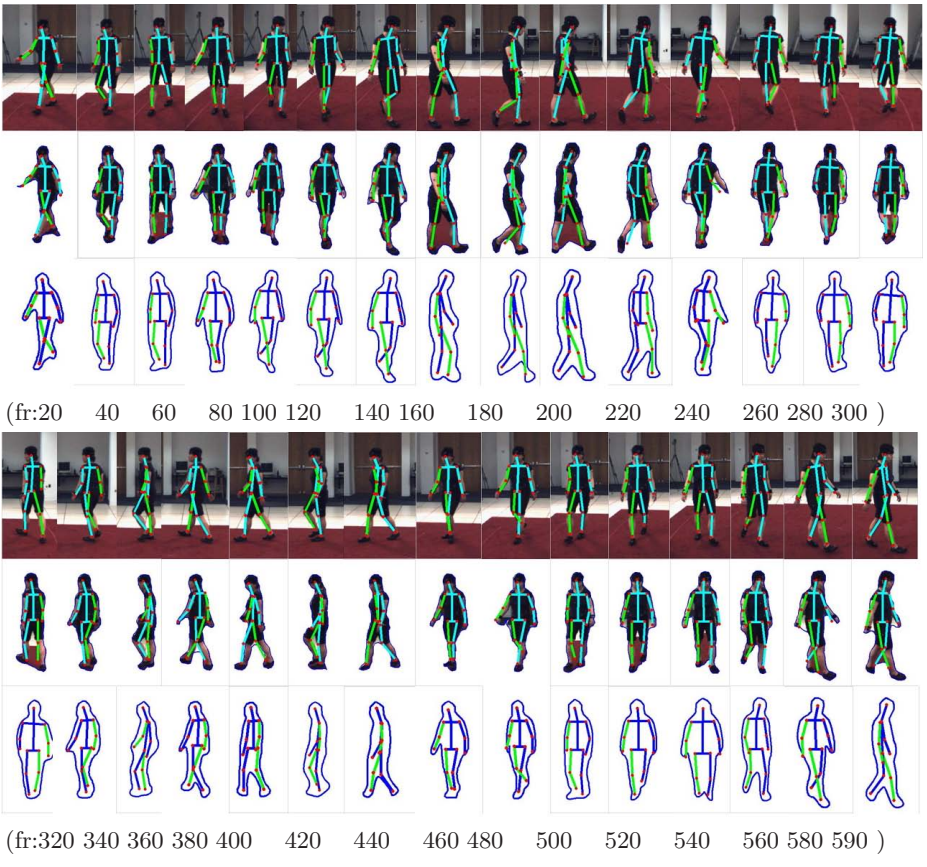


Fig. 7. Indoor Sequences from HumanEVA dataset. For each frames: (*up*) original images and 2D Poses, (*centre*) Segmented Silhouette and 2D Poses obtained by applying our tracking process and (*down*) MAP estimates for Pose and Shape.

smoothly changes and takes some "unseen values" (absent from the training views) while the MAP estimates are all orientated in the few training directions.

5 Conclusions

We have presented a Spatio-temporal 2D Models Framework (STMF) for 2D-Pose inference. Space and time are discretized and a mixture of probabilistic "local models" is learnt associating 2D Shapes and 2D Stick Figures. The discretization of the space provokes spatial discontinuities that affect the smoothness and precision of the results. To overcome the problem, we have applied a Rao-Blackwellized Particle Filter in the 2D-Pose Eigenspace, marginalizing the viewpoint parameter out from the state and computing it deterministically. We

thus imposed a strong restriction of continuity between consecutive viewpoints, thus forcing the interpolation of unseen data between view-based clusters.

We have demonstrated that a robust, fast and smooth human motion tracker is obtained by tracking only the few most important dimensions of our state space and refining deterministically with our STMF. This makes the process viable for real-time application.

References

1. Bowden, R., Mitchell, T.A., Sarhadi, M.: Non-linear statistical models for the 3d reconstruction of human pose and motion from monocular image sequences. Image Vision Comput. 18(9), 729–737 (2000)
2. Deutscher, J., Reid, I.: Articulated body motion capture by stochastic search. IJCV 61(2), 185–205 (2005)
3. Doucet, A., de Freitas, N., Murphy, K., Russell, S.: Rao-blackwellised particle filtering for dynamic bayesian networks. In: Conf. on Uncertainty in Artif. Int. (2000)
4. Grauman, K., Shakhnarovich, G., Darrell, T.: Inferring 3D Structure with a Statistical Image-Based Shape Model. In: ICCV, pp. 641–648 (2003)
5. Sigal, L., Black, M.J.: Humaneva: Synchronized video and motion capture dataset for evaluation of articulated human motion (2006)
6. Isard, M., Blake, A.: Condensation – conditional density propagation for visual tracking. IJCV 29(1), 5–28 (1998)
7. Johansson, G.: Visual interpretation of biological motion and a model for its analysis. Percept. Psychophys. 73(2), 201–211 (1973)
8. Khan, Z., Balch, T., Dellaert, F.: A rao-blackwellized particle filter for eigentracking. In: CVPR (2004)
9. Lan, X., Huttenlocher, D.P.: A unified spatio-temporal articulated model for tracking. In: CVPR (1), pp. 722–729 (2004)
10. Lee, C.-S., Elgammal, A.M.: Simultaneous Inference of View and Body Pose using Torus Manifolds. In: ICPR (3), pp. 489–494 (2006)
11. Mori, G., Malik, J.: Recovering 3d human body configurations using shape contexts. IEEE Trans. on PAMI 28(7), 1052–1062 (2006)
12. Ning, H., Tan, T., Wang, L., Hu, W.: Kinematics-based tracking of human walking in monocular video sequences. Image Vision Comp. 22, 429–441 (2004)
13. Rogez, G., Orrite, C., Martínez, J., Herrero, J.: Probabilistic Spatio-Temporal 2D-Model for Pedestrian Motion Analysis in Monocular Sequences. In: Conf. on Articulated Motion and Deformable Objects, pp. 175–184 (2006)
14. Rogez, G., Martínez-del-Rincón, J., Orrite, C.: Dealing with Non-linearity in Shape Modelling of Articulated Objects. In: Ib. Conf. on Pattern Recognition and Image Analysis, Part I. LNCS, vol. 4477, pp. 63–71 (2007)
15. Rogez, G.: Estimating Human Body Pose using Shape + Structure Models Warping. Tech. Report CVLab (2007)
16. Sidenbladh, H., Black, M.J., Sigal, L.: Implicit Probabilistic Models of Human Motion for Synthesis and Tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 784–800. Springer, Heidelberg (2002)
17. Taylor, C.J.: Reconstruction of articulated objects from point correspondences in a single uncalibrated image. CVIU 80(3), 349–363 (2000)

18. Tipping, M.E., Bishop, C.M.: Mixtures of probabilistic principal component analysers. Neural Computation 11(2), 443–482 (1999)
19. Urtasun, R., Fleet, D., Hertzmann, A., Fua, P.: Priors for people tracking from small training sets. In: ICCV (1), pp. 403–410 (2005)
20. Zhang, J., Collins, R., Liu, Y.: Representation and matching of articulated shapes. In: CVPR, pp. 342–349 (2004)
21. Zhang, J., Collins, R., Liu, Y.: Bayesian Body Localization Using Mixture of Nonlinear Shape Models. In: IEEE ICCV, pp. 725–732 (2005)

Efficient Upper Body Pose Estimation from a Single Image or a Sequence

Matheen Siddiqui and Gérard Medioni

Univ. of Southern Calif
1010 Watt Way, Los Angeles, CA
{mmsiddiq,medioni}@usc.edu

Abstract. We propose a method to find candidate 2D articulated model configurations by searching for locally optimal configurations under a weak but computationally manageable fitness function. This is accomplished by first parameterizing a tree structure by its joints. Candidate configurations can then efficiently and exhaustively be assembled in a bottom-up manner. Working from the leaves of the tree to its root, we maintain a list of locally optimal, yet sufficiently distinct candidate configurations for the body pose.

We then adapt this algorithm for use on a sequence of images by considering configurations that are either near their position in the previous frame or overlap areas of interest in subsequent frames. This way, the number of partial configurations generated and evaluated significantly reduces while both smooth and abrupt motions can be accommodated. This approach is validated on test and standard datasets.

1 Introduction

Articulated pose estimation and tracking in images and video has been studied extensively in computer vision. The estimation of an articulated pose, however, continues to remain a difficult problem as this process is complicated by high dimensional search-spaces riddled with local minima and ambiguous configurations.

In general, pose-estimation can be viewed as optimizing a multi-dimensional quality of fit function. This function encodes fidelity of a model to observables and a prior distribution. The success of aligning a model in this way depends on the amount of information that can be encoded into this function as well as the ability to optimize it. The more relevant observable and prior information one can fuse into a fitness function, the more likely the error surface becomes peaked on the *right* solution. However, highly detailed models often become more computationally expensive and difficult to optimize.

Instead of focusing on the single best solution under a complex and computationally costly fitness function, one can instead focus on efficiently generating candidates under a weaker but more computationally manageable function. Higher level information can then be used to select the real answer. This is the basis of many sampling based techniques[9][5].

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 74–87, 2007.

In many cases, the form of a fitness function can be restricted so that the global optimal or good approximations to the global optimum can be obtained efficiently. This can limit what one can actually model, which may result in configurations that minimize the fitness function but do not necessarily correspond to the correct answer. Nevertheless, it is likely that the true solution has at least a local optimum under such a function.

In this work we seek to find 2D articulated poses. This is done by first parameterizing these structures as a trees of 2D joints in which the relative position between parent child pairs are constrained. We then define a quality of fit function on this tree structure by attaching simple part based detectors between parent child joint pairs.

We then propose a method that explores this space of joint configurations and identifies locally optimal yet sufficiently distinct configurations, using a bottom up technique that maintains such configurations as it advances from the leaves of the tree to the root. A solution can then be selected from this list via continuity of motion or a detailed top down model based metric.

We also adapt this algorithm for use on a sequence of images to make it even more efficient by considering configurations that are either near their positions in the previous frame or overlap areas of interest in the subsequent frame. This allows the number of partial configurations generated and evaluated to be significantly reduced while both smooth and abrupt motions are accommodated. These algorithms are then validated on several sets of data including the HumanEva set.

The rest of this paper is organized as follows: In section 2 we discuss related work. Section 3 describes the parameterization of our model and our framework to align this model in a single image, together with results of this single frame alignment. In section 5, we discuss how to apply our framework to an image sequence efficiently, and we summarize our contributions in section 6.

2 Related Work

There exist many approaches to address pose estimation [9][5][12][15] [1][11]. These approaches differ in how the bodies are encoded, the visual saliency metrics used to align these models, and the machinery used to perform this alignment with the underlying image. The essence of these methods is to try to optimize their respective parameterization of a human pose over a function that depends both on observable image data and prior statistics on the pose.

In [9][4][2][18], fully articulated and detailed 3D models are employed. These methods search a solution space defined by complex top down metrics. In [2] this is accomplished using a gradient search. In [4] this is done via randomized search. In [9] a data driven MCMC approach is used to explore a high-dimensional solution space and bottom up limb detectors are used to enhance this search. In [18], both gradient and randomized search techniques are employed.

An alternative to searching directly for a 3D parameterization of the human body, is to search for its projection. In [12] this idea is formalized using the Scaled Prismatic Model (SPM), which is used for tracking poses via registering frames.

In [10] a 2D articulated model is aligned with an image by matching a shape context descriptor. In [6] articulated poses are estimated directly using a large database of exemplars and an appropriate hashing function. In [14], multiple 2D limb models are used to track the 3D motion of a forearm.

Other relevant approaches model a human as a collection of separate but elastically connected limb sections, each associated with its own detector[5][13][8]. In [5] this collection of limbs is arranged in a tree structure. It is aligned with an image by first searching for each individual part over translations, rotations, and scales. Following this, the optimal pose is found by combining the detection results of each individual part efficiently. In [16] limbs are arranged in a graph structure and belief propagation is used to find the most likely configuration.

Sampling methods in general will attempt to represent a distribution over the space of pose, but explicit exploration of multiple hypothesis has been investigated in [9] and [5]. In [9], a number of candidates are reported for a given instance and the solution with the least error with respect to the ground truth is often not the solution with the best score. In [5], this idea of multiple candidates is explored further by sampling a part based, bottom up metric and selected a final solution based on a top down metric.

In both works, the idea was to explore using sampling based methods, which allow one to explore multiple modes in a solution space. This, however, requires a randomize exploration of the search space. Here, we propose a method that finds these local minima both efficiently and exhaustively, such that they are sufficiently different.

3 Formulation

Here, we seek to estimate a 2D articulated model. This is a accomplished by parameterizing this structure by its 2D joint locations. A fitness function is constructed by attaching individual part detectors between pairs of joints. We then proceed to find potential joint configurations that locally optimize the response of the detectors.

3.1 Model

We model the projection of a 3D articulated model. These are positions in the image plane as shown in Fig. 1(a). This is a natural representation for human image alignment in a single image. As shown in [12], modeling the projection of a articulated 3D object eliminates depth related degeneracies. Furthermore it may be possible to estimate the 3D joint positions in a post processing step using either multiple views or geometry [19][10].

We further encode this collection of joints in a tree structure (shown in Fig. 1(a)) and constrain the locations where a child joint can be relative to its parent joint as shown in Fig. 1(b). We will refer to $\mathbf{X}$ as a tree, or configuration, of joints. Individual joints are specified as $x^i \in \mathbf{X}$. A sub-tree is specified with the

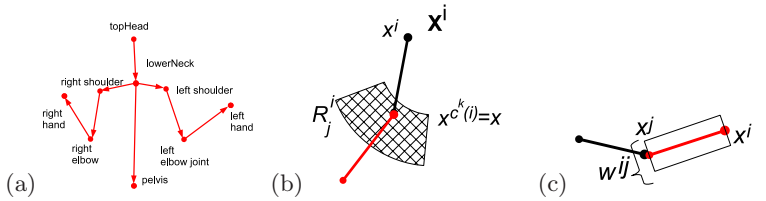


Fig. 1. In (a) a configuration of joints (labeled) assembled in a tree structure is shown. In (b) the notation used is illustrated along with the permitted locations of child joint relative to its parent. In (c) a fixed width rectangle associated with a pair of joints is shown.

super-script of its root joint, $\mathbf{X}^i$. Also note that $root(\mathbf{X}^i) = x^i$. The k^{th} child of joint of x^i is specified by $x^{c^k(i)}$. The locations a child joint can have relative to its parent are specified by R^i_j.

Similar to a collection of elastically connected rigid parts[5], this representation can be used to find configurations in a bottom up manner. A collection of joints, however, has fewer explicit degrees of freedom. In a pictorial structure, the additional degrees of freedom incurred by parameterizing each rigid limb section separately are constrained by using elastic constraints between parts. A collection of joints enforces these constraints implicitly.

For example, by modeling the upper body as a collection of fixed width limbs, we end up with 15 joints. With two degrees of freedom for each joint, this gives us 30 parameters. A similar pictorial structure would give us 10 limbs, each of which has a translation and rotation and length or a total of 40 parameters.

3.2 Quality of Fitness Function

To align this model, we construct a cost function:

$$\Psi(\mathbf{X}) = \alpha P_{image}(\mathbf{X}) + (1-\alpha)P_{model}(\mathbf{X}) \tag{1}$$

where $\mathbf{X}$ denotes a tree of joint locations defined in section 3.1. The terms P_{image} and P_{model} evaluate $\mathbf{X}$'s image likelihood and prior respectively. The parameter, α, controls the relative weight of the two terms.

The term P_{image} is a part-based metric computed by evaluating part detectors within the fixed width rectangles between pairs of joints in $\mathbf{X}$. This is illustrated in Fig. 1(c). In particular

$$P_{image}(\mathbf{X}) = \prod_{(i,j)\in edges(\mathbf{X})} P_{part_{ij}}(x^i, x^j, w^{ij})\mathbf{M}^i(x^i)\mathbf{M}^j(x^j) \tag{2}$$

where x^i and x^j are parent-child pairs of joints in $\mathbf{X}$. This pair of joints correspond to a limb with fixed width w^{ij} . The term, $P_{part_{ij}}$ is a part based detector defined on rectangle of width w^{ij} extending from joint x^i to x^j. The term $\mathbf{M}^i(x^i)$ is a mask that can be used to explicitly penalize a joint from certain locations.

The P_{model} term biases a solution toward a prior distribution. In this work we do not model this term explicitly. Instead, we have constrained the locations a child joint can have relative to its parent, R_j^i to be points sampled on a rectangular or polar grid. We thus assume all poses that satisfy the parent-child constraints are equally likely.

4 Peak Localization

To find optimal configurations, one could use the algorithm in Fig. 2 as a baseline design. There each configuration is graded according to, Ψ. The least cost configuration, $\mathbf{X}*$, is repeatedly identified, and all configurations that are sufficiently similar (i.e. $diff(\mathbf{X}, \mathbf{X}*) < \sigma$) are removed. In this work $diff(\mathbf{X},\mathbf{X}^*)$ is the maximum difference between corresponding joint locations.

This procedure would produce an optimal sequence of solutions that are sufficiently different. The complexity of this procedure is

$$\begin{aligned} & O(|C_{in}|MN + F(N)|C_{in}|) \\ = \; & O(|C_{in}|(MN + F(N))) \end{aligned} \tag{3}$$

where the first term arises from applying $diff(\mathbf{X}, \mathbf{X}*)$, which is $O(N)$, to elements of C_{in} in order to get M configurations. The second term arises from applying Ψ, whose complexity we denote for now as a function of N, $F(N)$, to elements of C_{in}.

Such an approach is computationally intractable given the size of C_{in}. We note that there are 15 joints, 14 of which have a parent. Thus, if we define R to be the maximum number of distinct locations a child joint can have relative to its parent (i.e. $\forall_{ij}|R_j^i| < R$), and denote $|I|$ to be the number of locations the root joint can have in the image, the number of candidate solutions is on the order of $R^{14}|I|$.

We can approximate this procedure, however, by assembling partial joint configuration trees in a bottom-up manner. Working from the leaves of the tree to its root, we maintain a list of locally optimal, yet sufficiently distinct configurations for each sub-tree. These lists are pruned using the algorithm shown in Fig. 2 to avoid exponential growth. As the configurations for sub-trees are assembled, they are reweighted with likelihood functions, $\Psi(\mathbf{X}^i)$, that depend only on the sub-tree.

This process continues until the root of the tree, and a list of optimally distinct configurations of joints is returned. The complexity of this procedure is $O(M^3N^3)$ and is described in detail below.

4.1 Candidate Search

To generate these partial configurations, we maintain a list of candidates for each sub-tree in $\mathbf{X}$, and at each possible location this tree can exist in the image. This is denoted by: $\{{}^k\mathbf{X}_l^i\}_{k=1}^M$. Here i refers to the node id of the root

Function C_{out} = wprune(C_{in}, , σ, M)
/* Finds the best M configuration that are different by at least σ.
C_{in} ${}^{k}\{\mathbf{X}\}_{i=1}^{N_k}$ input candidates
C_{out} output candidates
/

grade each configuration in C_{in} according to Ψ
do
remove $\mathbf{X}*$ with lowest score from C_{in}
insert $\mathbf{X}*$ into C_{out}
remove any $\mathbf{X}$ from C_{in} s.t. $diff(\mathbf{X}, \mathbf{X}*) < \sigma$
while $|C_{out}| <= M$ and $|C_{in}| > 0$

Fig. 2. Pseudo-code for baseline algorithm

node in this configuration (for example, $i = shoulder$). These configurations are located at the l^{th} pixel $p_l = (x, y)$ and each candidate configuration in this list has a common root joint referred to as x_l^i. The index, k, specifies one such configuration.

This list can be constructed from the candidate configurations associated with the children of joint x_l^i, denoted by

$$\begin{gathered}\{\mathbf{X}_l^i\} = wprune(\{x_l^i \otimes {}^{k}\mathbf{X}_{l'}^{c^1(i)} \otimes \ldots \otimes {}^{k'^{(nc_i)}}\mathbf{X}_{l^{(nc_i)}}^{c^{nc_i}(i)}\} \\ l' \in R^i_{c^1(i)}, \ldots, l^{(nc_i)} \in R^i_{c^{nc_i}(i)} \\ k', \ldots, k^{(nc_i)} \in [1, M], M)\end{gathered} \tag{4}$$

The operator $\otimes$ denotes the joining of branches into trees, and $wprune()$ is shown in the algorithm of Figure 2. As before, the variable R_j^i is the list of locations the child joint j can have relative to its parent i, and nc_i is the number of children of node i.

Here we combine the M candidates from each sub-tree located at each point in R_j^i. If R is a bound on the size of $|R_j^i|$, the number of candidates passed to $wprune$ is bounded by $(MR)^{nc_i}$. This can be reduced if we prune candidates as we fuse branches in pairs:

$$\begin{gathered}\{\mathbf{X}_l^i\} = wprune(wprune(\{x_l^i \otimes {}^{k'}\mathbf{X}_{l'}^{c^1(i)}\}\forall k'l', M) \otimes \\ \ldots) \otimes \{{}^{k^{nc_i}}\mathbf{X}_{l^{(nc_i)}}^{c^{nc_i}(i)}\}\forall k^{nc_i} l^{nc_i}, M)\end{gathered} \tag{5}$$

By processing pairs, we limit the number of candidates sent to $wprune()$ to be $M(RM)$. If we denote N^i as the number of joints in the sub-tree $\mathbf{X}^i$, the complexity for $wprune()$ is $(MN^i + F(N^i))$ times the size of the list to operate on. It will also be called nc_i times. Thus the overall complexity for an individual joint is $nc_i(MRM)(MN^i + F(N^i))$. This processing must be done for all N joints and at every pixel, p_l that a sub-tree's root can be located. Since the number of joints in each sub tree is bounded by N, and the number of locations p_l is bounded by the size of the image $|I|$, the overall complexity is bounded by:

$$O(NR|I| \max_i(nc_i)(M^3N^2 + M^2F(N)) \tag{6}$$

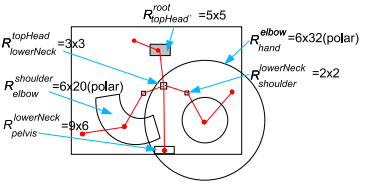


Fig. 3. The relative positions of each child joint relative to its parent. Sizes shown are the number of discrete point locations in each region.

Rank	Image Error(pixels)	std
0	17.52	21.83
2	15.21	19.66
4	13.09	17.23
6	11.79	15.48
8	10.97	14.19

(a)

(b) (c) (d)

Fig. 4. In (a) the average positional joint error for the Rank N solution taken over a 70 frame sequence along with its standard deviation. The top rows in (b)-(d) shows the optimal results with respect to Ψ, returned from the algorithm in section 3. The second row shows the *Rank 5* solution.

We also note that the Ψ defined in section 3.2 is computed as a sum of response to parts of a configuration. In this framework it can be computed in constant time, β, as a sum of the scores of the partial configurations already computed and the computation of a constant number of terms. Thus the overall complexity is

$$O(R|I|\max_i(nc_i)(M^3N^3 + \beta M^2 N)) \tag{7}$$

Here we preserve the second term, because the constant is very large.

4.2 Results and Analysis

Examples of running the method described in section 4.1 are shown in Figure 4. Here the model used is shown in Figure 3, with each R^i_j superimposed. The images used here are part of a 70 frame annotated sequence. In this sequence, we assumed the *topHead* joint to be within the gray rectangle shown. We further constrain the relative positions of the elbow and hand joints to be at polar grid locations within the regions shown. In particular, we considered 6 different lengths and 20 angular positions within the 90 degree angular range for the elbow joints relative to the shoulder and 6 different lengths with 32 angular positions

in a 360 degree angular range for the hand joint relative to the elbow. The other joints are quantized at modulo 4 pixel locations within their corresponding rectangles.

In selecting these ranges, there is a trade-off between accuracy and speed. The overall runtimes depend on the number of samples within each $|R_j^i|$, but accuracy depends on the resolution at which we sample. Nevertheless, the ranges selected yield good results and reasonable speed.

The term, $P_{part_{ij}}$ is the image likelihood of an individual part and is computed as:

$$P_{part_{ij}}(x^i, x^j, w^{ij}) = \prod_{cp \in Rect(x^i, x^j, w^{ij})} \hat{P}_{ij}(cp) \tag{8}$$

This likelihood is based on how well each underlying color pixel, cp, in the rectangle of width w^{ij} extending from joint x^i to x^j. (i.e. $Rect(x^i, x^j, w^{ij})$) belongs to a color distribution. These distributions are modeled as simple color RGB and HS histograms and trained from example images. The widths of the limbs, w^{ij}, are known.

We found a set to 10 configuration under Ψ, such that no two joints were within 40 pixels (i.e $\sigma = 40$). Results with respect to the ground truth joint locations are summarized in Figure 4(a). Here we show the average joint error for the *Rank N* solution. Since our algorithm produces an order set of $M = 10$ configurations, the *Rank* $N < M$ solution is the configuration among the first N of M with the smallest average joint error with respect to the ground truth. From this we see that as the number of candidates returned increases, the average distance to the correct solution decreases. This shows that while the solution that minimizes Ψ may not correspond to the actual joint configuration, its likely a local minium will.

This is consistent with the results shown in Figure 4(b)-(d). In the top row the optimal solution with respect to Ψ is shown, while the Rank 5 solution is shown in the second row. In these images, the *Rank 5* solution is closer to the ground truth. On average it takes 982ms seconds to process each image. Of this time, 232ms is not dependent on the size of this problem (i.e. does not depend on N,M, R and nc) and can be thought of as a pre-processing step necessary for evaluating Ψ. Of the remaining 750ms that depend on the size of this problem, 200ms was devoted to evaluating Ψ.

5 Image Sequence Analysis

While the algorithm in section 4.1 is polynomial, it may still be too slow for practical applications. Significant speed improvements can be gained if we exploit the smoothness of motion available in video, and limit the number of times Ψ is evaluated.

5.1 Motion Continuity

The complexity of our algorithm is directly proportional to the number of pixel locations, p_l, where each joint can be located. In computing the complexity in

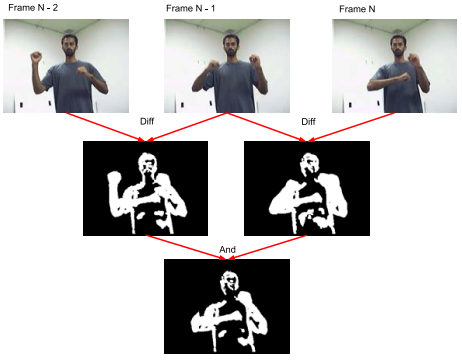


Fig. 5. Computation of a mask that coarsely identifies motion regions

equation 7, this was bounded by the size of the image, $|I|$. If the motion of the joints in an image sequence is smooth, we only need to look for a joint in a subsequent frame around its position in a previous frame. In this work we seek to maintain a list of M configurations. We can avoid having to commit to any one of these solutions by considering joint locations about any of the M candidate positions in the previous frame. In particular we constrain each joint to be in a small a rectangle, W, about the corresponding joints in one of its M previous positions. This translates to a complexity of:

$$O(R|MW|\max_i(nc_i)(M^3N^2 + \beta M^2 N)) \tag{9}$$

Constraining the joints position in this way works well when the motion is smooth. However, there may be motion between frames that violates this assumption. This will likely occur on the hands and arms especially when the frame rate is 10-15fps. We now describe an efficient way to handle the presence of such discontinuities while enforcing smoothness.

5.2 Motion Discontinuities

To contend with fast motion, we first estimate moving foreground pixels by frame differencing. In particular we compute:

$$\begin{aligned} F_n(i,j) = D(I_n, I_{n-1}, \sigma_{TH})(i,j) \bigcap \\ D(I_n, I_{n-L}, > \sigma_{TH})(i,j) \end{aligned} \tag{10}$$

Here $D(I_i, I_j, \sigma_{TH})$ computes a difference mask between frames I_n and I_{n-1} and then between I_n and I_{n-L}. The resulting difference masks are fused with a Boolean *and* operation. The result of this procedure is a mask that identifies those pixels in frame n that are different from the previous frames $n-1$ and $n-L$. As shown in Figure 5, this coarsely identifies regions of the image that have changed.

This mask can be used when generating candidates in equation 5. We assign to each candidate a number P_{limb} based on the fixed with rectangle associated with the joint position x_l^i and the root location of its child configuration, $\mathbf{X}_l^{c^k(i)}$. In particular P_{limb} is the percent occupancy of this rectangle with fore-ground pixels identified from F_n.

Instead of evaluating each candidate sent to $wprune()$ with Ψ, we instead only consider those candidates that are either in the windows, W, about their previous location (for smooth motion) or have $P_{limb} > thresh$ (for discontinuous motion). Computation of P_{limb} is still $O(R|I|M^2N)$, however the computation can be computed with integral images [20] and is extremely efficient. It also significantly reduces the number of candidates generated and the number of calls to Ψ.

5.3 Partial Update

We also reduce the run time by first updating only the head and torso, while fixing the arms and then updating the arms and fixing the head and torso. This is a reasonable updating scheme as the head and torso are likely to move smoothly, while the arms may move more abruptly.

This is done by updating the joint locations, *topHead*, *lowerNeck*, and *pelvis* in equation 5 while ignoring the sets, $\{{}^k\mathbf{X}_l^{shoulderL}\}$ and $\{{}^k\mathbf{X}_l^{shoulderR}\}$. When updating the head and torso we assume continuity and only consider the region defined in section 5.1.

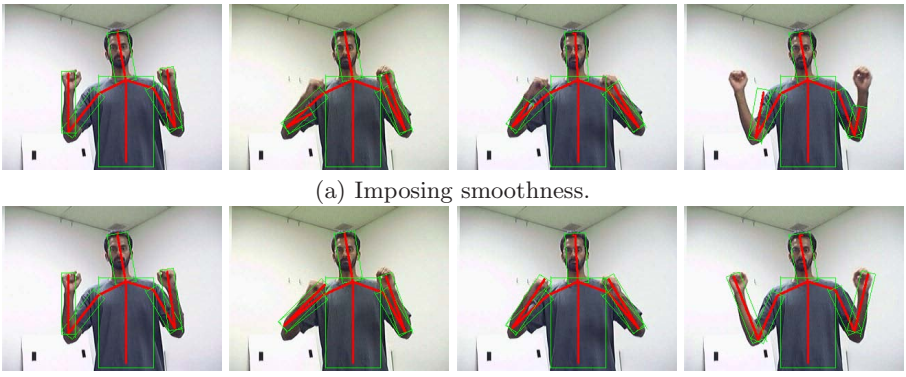
Once the joints *topHead*, *lowerNeck*, corresponding to *pelvis* have been computed, we can lock the topHead and lower neck positions and recompute $\{{}^k\mathbf{X}_j^{lowerNeck}\}$. Updating in this way reduces the number of candidates generated significantly and allows the *topHead* to move about the image as needed.

5.4 Results

Examples of output of the methods described in section 5 are shown in Figure 6. Here, the same model shown in Figure 3 and the sequence in section 4.2 are used. This sequence was acquired at 30 frames per second and then down sampled to 6 frames per second. In the first row, continuous motion is assumed, and the modification described in section 5.1 and in section 5.3 are used. Window sizes of 60×60 are used. In the first frame, a full search with the topHead joint positioned on the head is completed. The processing time devoted to finding joint configurations was 781ms. In subsequent frames, this time is reduced to 70ms.

In the second row, we also use the method described in section 5.2. Here we reduce the window size to $W = 30 \times 30$ and use candidate configurations when $P_{limb} > 1/2$. In these frames, it took on average 84ms to compute the foreground masks (shown in the 3rd row), and the time associated with configuration construction increased to 114ms.

From these sequences, we see that assuming continuous motion allows for significant improvements in speed. As shown in Figure 6(a), smoothness may not



(a) Imposing smoothness.

Frame 1 Frame 44 Frame 53 Frame 70

(b) Preserving fast motion.

Fig. 6. The top row shows the Rank 5 results with respect to Ψ, when only continuous motion is assumed using the method in section 5.1. The second row shows the Rank 5 solution when discontinuous motion is allowed using the method in section 5.2.

always be assumed. Adding the information from the motion mask can correct this (Figure 6(b)) while maintaining efficiency.

5.5 HumanEva Data Set

We also evaluated this algorithm on a sequence from the HumaEva [17] data set. In particular we used frames 615 to 775 in increments of 5 from the S2/Gesture_1_(C1) sequence. The model use here is essential the same as that shown in Figure 3. The main difference is that $R^{root}_{topHead}$, $R^{topHead}_{lowerNeck}$, $R^{lowerNeck}_{pelvis}$, are enlarged and elongated to better accommodate changes in scale.

The limb detectors used on this sequence consist of several non-overlapping areas representing foreground, background, and skin colored regions. The

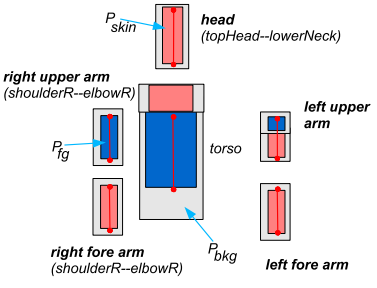


Fig. 7. The limb detectors used on the HumanEva data set

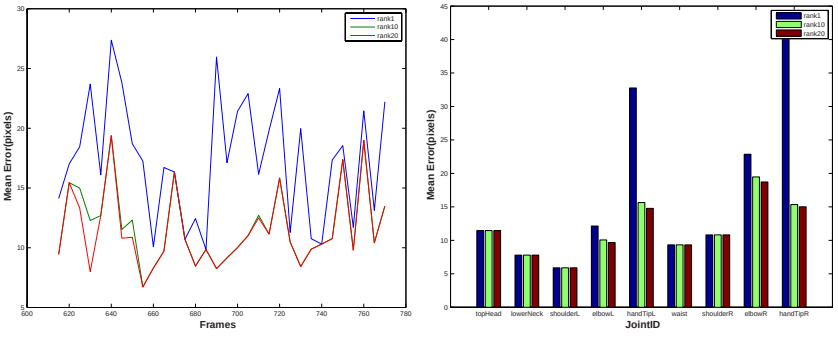


Fig. 8. In (a) the average joint error at each frame in sequence. In (b) the average joint error for each joint over the sequence.

likelihood of each patch is the product of the underlying color pixel's probability of membership in each region.

$$P_{part_{ij}}(x^i, x^j, w^{ij}) = \prod_{p \in fg} P_{fg}(p) \prod_{p \in bkg} P_{bkg}(p) \prod_{p \in skin} P_{skin}(p) \tag{11}$$

where P_{bkg}, is obtained from the background model provide with the HumanEva data set. The terms P_{fg}, the foreground likelihood and and P_{skin}, the skin likelihood are modeled as histograms extracted from the sequence itself. The shape of each part detector is shown in Figure 7.

For the range of images we worked with, we established the ground truth by annotating the joints. This was necessary because, in several of these frames, the projected ground truth provided was misaligned. Also, we are looking for the hand tip, not the wrist, which is what is marked in this data set.

The average error with respect to the corrected projected joints is shown in Figure 8. Example poses are shown in Figure 9. In this sequence, we identified a point near the top of the head in the first frame and use the method described in section 3 to align the pose. The pose was then tracked using the methods described in section 5.

Throughout the sequence, we maintain 20 candidates. In the first frame, we detect 10 using the method in section 3 and then another 10 constraining the hand to be away from a the detected face using $\mathbf{M}^{handR}$ and $\mathbf{M}^{handL}$. Time devoted to assembling candidates during the initial detectoin was 1.531s (i.e. not including the image pre-proccessing and the like) while the time associated only with constructing candidates while tracking was on average 188ms.

In Figures 8 and 9 the ranked results are shown. Here we see the rank1 solutions, which minimize Ψ are not correct and performance is poor. However the rank 10 (and rank 20) coincide with pose that appears more correct. The joints for which this has the greatest effect are the hand tips. Though we are focusing on the upper body, the performance on this sequence is comparable to that of [7] on the S3/Walking_1_(C2) sequence.

(a) Frame 0 (b)Frame 10 (c) Frame 20 (d) Frame 31

Fig. 9. The top row shows the Rank 1 results with respect to Ψ, when only continuous motion is assumed using the method in section 5.1. The second row shows the Rank 10 solution when discontinous motion is allowed using the method in section 5.2. The third row shows the moving forground pixel as computed using three consecutive frames (not shown).

6 Summary and Future Work

We have developed a method to find candidate 2D articulated model configurations by searching for local optima under a weak fitness function. This is accomplished by first parameterizing poses by their joints organized in a tree structure. Candidate configurations can then efficiently and exhaustively be assembled in a bottom-up manner. Our results suggest that while the configurations that minimizes the fitness function may not correspond to the correct pose, a local optima will. One can then make a selection based on top down metrics and spatial continuity.

Our next steps include feeding these results to a gesture recognition module [3]. We also are interested in making use of multiple cameras and extending this formulation to 3D.

References

1. Agarwal, A., Triggs, B.: 3d human pose from silhouettes by relevance vector regression. In: CVPR, vol. II, pp. 882–888 (2004)
2. Bregler, C., Malik, J.: Tracking people with twists and exponential maps. In: CVPR, Santa Barbara, CA, pp. 8–15 (June 1998)

3. Cohen, I., Li, H.: Inference of human postures by classification of 3d human body shape. In: AMFG, Nice, France, pp. 74–81 (2003)
4. Deutscher, J., Reid, I.: Articulated body motion capture by stochastic search. IJCV 61(2), 185–205 (2005)
5. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV 61(1), 55–79 (2005)
6. Shakhnarovich, T.D.G., Viola, P.: Fast pose estimation with parameter sensitive hashing. In: CVPR, Madison, WI (June 2003)
7. Howe, N.R.: Evaluating lookup-based monocular human pose tracking on the humaneva test data. In: EHuM. Evaluation of Articulated Human Motion and Pose Estimation (2006)
8. Ju, S.X., Black, M.J., Yacoob, Y.: Cardboard people: A parameterized model of articulated image motion. In: Proc. of the 2nd Int. Conf. on Automatic Face and Gesture Recognition, pp. 38–44 (1996)
9. Lee, M.W., Cohen, I.: A model-based approach for estimating human 3d poses in static images. PAMI 29(6), 905–916 (2006)
10. Mori, G., Malik, J.: Recovering 3d human body configurations using shape contexts. PAMI 28(7), 1052–1062 (2006)
11. Mori, G., Ren, X., Efros, A.A., Malik, J.: Recovering human body configurations: combining segmentation and recognition. In: CVPR, Washington, DC, vol. II, pp. 326–333 (2004)
12. Morris, D.D., Rehg, J.M.: Singularity analysis for articulated object tracking. In: CVPR, Santa Barbara, CA (June 1998)
13. Ramanan, D., Forsyth, D.A.: Finding and tracking people from the bottom up. In: CVPR, Madison, WI, vol. II, pp. 467–474 (2003)
14. Siddiqui, M., Medioni, G.: Real time limb tracking with adaptive model selection. In: ICPR, pp. 770–773 (2006)
15. Sigal, L., Bhatia, S., Roth, S., Black, M.J., Isard, M.: Tracking loose-limbed people. In: CVPR, Washington, DC, vol. I, pp. 421–428 (2004)
16. Sigal, L., Isard, M., Sigelman, B.H., Black, M.J.: Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In: NIPS (2004)
17. Sigal, B.M.J., Humaneva, L.: Synchronized video and motion capture dataset for evaluation of articulated human motion. Technical report, Brown University, CS-06-08 (2006)
18. Sminchisescu, C., Triggs, B.: Covariance scaled sampling for monocular 3d body tracking. In: CVPR, Kauai, Hawaii (December 2001)
19. Taylor, C.J.: Reconstruction of articulated objects from point correspondences in a single uncalibrated image. CVIU 80(3), 349–363 (2000)
20. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR, Kauai, Hawaii (2001)

Real-Time and Markerless 3D Human Motion Capture Using Multiple Views

Brice Michoud, Erwan Guillou, and Saïda Bouakaz

Laboratory LIRIS - CNRS, UMR 5205
University of Lyon, France
{brice.michoud, erwan.guillou, saida.bouakaz}@liris.cnrs.fr

Abstract. We present a fully automated system for real-time markerless 3D human motion capture. Our approach, based on fast algorithms, uses simple techniques and requires low-cost devices. Using input from multiple calibrated webcams, an extended Shape-From-Silhouette algorithm reconstructs the person in real-time. Fast 3D shape and 3D skin parts analysis provide a robust and real-time system for human full-body tracking. Animation skeleton and simple morphological constraints make easier the motion capture process. Thanks to fast and simple algorithms and low-cost cameras, our system is perfectly apt for home entertainment device.

Results on real video sequences with complicated motions demonstrate the robustness of the approach.

1 Introduction

In this paper we propose a fully automated system for real-time and markerless motion capture dedicated to home entertainment (see Fig. 2(a)).

Marker-free motion capture has long been studied in computer vision as classic and fundamental problems. While commercial real-time products using markers are already available, sound online marker-free systems remain an open issue because many real-time algorithms still lack robustness, or require expensive devices and time-consuming algorithms. While most popular techniques run on PC cluster, our system requires a small set of low-cost cameras (three or more) and a single computer. Our system works in real-time (30 fps), without markers (active or passive) or special devices.

1.1 Related Work

Several techniques have been proposed to tackle the marker-free motion capture problem. They vary according to the features used for analysis and the number of cameras. We now review techniques related to our work. For more information the reader is refereed to [1].

Motion capture systems vary in the number of cameras used. Single camera systems [2,3,4], although simple, encounter several limitations. They suffer from

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 88–103, 2007.

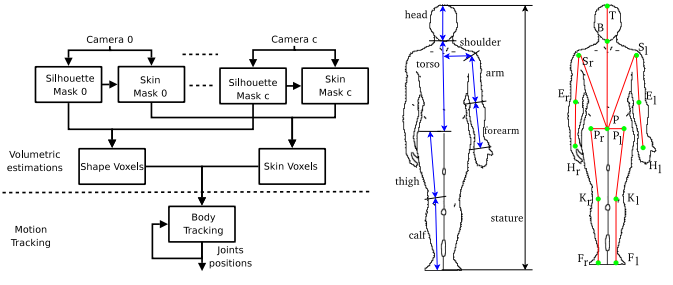
the occlusion problem. In some cases they suffer from ambiguous response as different positions can yield the same image. Our system uses multiple cameras.

Concerning multi-view approaches, most of the techniques are on Silhouette analysis [5,6,7,8]. These techniques provide good results if the topology of the reconstructed 3D shape complies with human topology *i.e.* each body parties is unambiguously mapped to the 3D shape estimation.

With self-occlusion cases or large contacts between limbs and body these techniques frequently fail. Caillette *et al.* [9,10] method involves shape and color clues. They link colored blobs to a kinematic model to track individual body parts. This technique requires contrasted clothing between each body part for tracking, thus adding a usability constraint.

Few methods provide real-time motion capture from multiple views. Most of them run at interactive frame rates (10 fps for [9] an 15 fps for [10]). The initialization step is an inherent limitation of real-time motion capture. Most of the proposed techniques impose conventional body pose [11], or manual intervention for anthropometric measurements and initial pose estimation [12]. These protocols are not user-friendly. They are also or disturbing or even impossible.

We propose a fully automated system for practical real-time motion capture thanks to a few number of cameras. It includes initialization step and motion tracking. Our process is based on simple heuristics, driven by shape and skin-parts topology analysis and temporal coherence. It runs at 30 fps on a single and standard computer. No parallel intensive computing nor batch processing is required.



(a) (b) (c)

Fig. 1. (a) System overview: Reconstruction algorithm and Pose estimation algorithms. Body parts labeling (b) and joint naming (c).

Fig. 1(a) outlines the two main stages of our method: the 3D volumetric estimation step and the motion tracking step based on the analysis of the 3D reconstruction.

This article is organized as fallows: Section 2 presents our work for real-time 3D reconstruction. Section 3 describes an overview of the motion tracking. Section 4 details the full-body motion tracking and Section 5 presents the fully-automated initialization step. Experimental results, in which the algorithm is

applied to real and complex examples are presented Section 6. They show the validity and the robustness of our process. In Section 7 we conclude about our contributions and develop some perspectives for this work.

2 3D Shape and Skin-Parts Estimation

We present an extension of Shape-From-Silhouette (SFS) algorithms. It reconstructs in real-time 3D shape and 3D skin-colored parts of a person from calibrated cameras.

Usually, only SFS methods compute in real-time 3D shape estimation of an object, from its silhouette images. Silhouette images are binary masks corresponding to captured images where 0 corresponds to background, and 1 stands for the (interesting) feature of the object. The formalism of SFS was introduced by A. Laurentini [13]. By definition, an object lies inside the volume generated by back-projecting its silhouette through the camera center (called silhouette's cone). With multiple views of the same object at the same time, the intersection of all the silhouette's cones build a volume called "Visual Hull", which is guaranteed to contain the real object. There are mainly two ways to compute an object's Visual Hull.

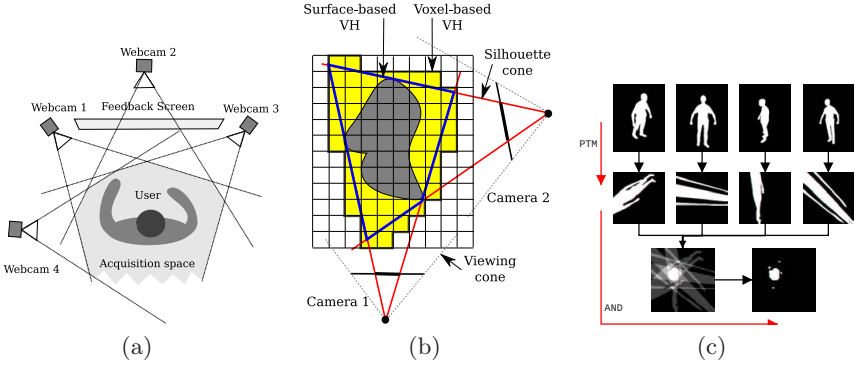


Fig. 2. (a) Interaction setup. (b) Object reconstruction by surface and volumetric approaches represented in 2D. (c) SFS computation using "Projective Texture Mapping" method: First all silhouette masks are projected on a stack, logical AND is used to compute the projection intersection.

Surface-Based Approaches. They compute the intersection of silhouette's cone surfaces (see Fig. 2(b)). First silhouettes are converted into polygons. Each edge is back-projected to form a 3D polygon. Then each 3D polygon is projected onto each other's images, and is intersected with each silhouette in 2D. The resulting polygons are assembled to form an estimation of the polyhedral shape (see [12,14]). Resulting Surface-based shape from silhouette is underlined Fig. 2(b). These approaches are not well suited to our application because of the

complexity of the underlying geometric calculations. Furthermore incomplete or corrupted surface models could be created, directly depending upon polyhedron sharpness and silhouette noise.

Volumetric-Based Approaches. They usually estimate shape by processing a set of voxels [6,15,16,17]. The object's acquisition area is split up into a 3D grid of voxels (volume elements). Each voxel remains part of the estimated shape if it projection in all images lies in all silhouettes (see Fig. 2(b)). This volumetric approach is adapted for real-time pose estimation, due to its fast computation and robustness to noisy silhouettes.

We propose a new framework which computes a 3D volumetric shape and skin parts estimation on a single computer. The system consists of two tasks: (1) Input data : Camera calibration data, silhouette and skin parts segmentation, (2) 3D Shape and skin parts estimation: shape voxels are computed by a GPU SFS implementation and skin parts are determined using voxel visibility. Each task is described in their respective section.

2.1 Input Data

First, webcams are calibrated using a popular algorithm proposed by Zhang *et al.* [18]. To enforce coherence between the cameras, color calibration is done using the method proposed by N.Joshi [19].

The second step consists in silhouette segmentation (see [20] for silhouette segmentation algorithm comparative study). We use the method proposed by [15]. First we acquire images of the background. The foreground (the human) is then detected in the pixels whose value has changed. We assume that only one person is in the field of view of the cameras, thus he or she is represented by only one connex component. Due to webcam noise, we can have several connex parts, but the smallest ones are removed : they correspond to noise.

The last step before shape estimation is skin part extraction from silhouette and color images. Normalized Look-up Table method [21] provides fast skin-colored segmentation. This segmentation is applied on each image restricted to its silhouette mask (skin-colored pixels outside the silhouette correspond to background pixels).

2.2 3D Shape and Skin Parts Estimation

First we estimate a 3D shape of the person filmed using a GPU implementation of SFS. Volumetric SFS is usually based on voxel projection : a voxel remains part of the estimated shape if it projects itself into each silhouette. In order to find the best way to fit GPU implementation, we propose to use reciprocal property. We project each silhouette into the 3D voxel grid as proposed in [15]. If a voxel is the intersection of all the silhouette projections, then it represents the original object.

The classical N^3 voxel cube can be considered as a stack of N images of resolution $N \times N$. We stack the N image in screen parallel planes. For each

camera view, silhouette masks are projected on each slice using the "projective texture mapping" technique [22]. Intersection of silhouettes projections on all slices provides voxel-based 3D shape. Intersection of silhouette mask projections on a single slice is underlined Fig 2(c). To save video bus bandwidth, computations for a cube of voxels are made in the same framebuffer, which is tiled by all the N slices of resolution $N \times N$.

To estimate skin voxels, we compute each voxel's visibility from each camera. The voxel visibility test is based on the "Item Buffer" method used in some voxels coloring algorithms [23]. A unique identifier is associated to each voxel (*e.g.* color) and voxels are rendered on raster-based framebuffers, corresponding to each camera views. For each framebuffer, colors describe visible voxels and this enables bidirectional pixel to voxel mapping. If a voxel is skin consistent (*i.e.* it is mapped to skin mask pixels in all of its viewing camera) then it is classified as skin voxel. To improve visibility computation time, only surface voxels are tested (*i.e.* voxels which have less than 26 neighbors).

To reduce computation time for pose estimation we propose to keep the visible voxels. Let $\mathcal{V}_{\overline{\text{skin}}}$ be the selected voxels form the shape voxel set, $\mathcal{V}_{\text{skin}}$ be the skin consistent voxel set, and $\mathcal{V}_{\text{all}}$ be their union.

Our implementation provides up to 100 reconstructions per second. As webcam acquisition is done at 30 fps, it allows us to save time for motion capture computation, hence achieving our real-time goal.

3 Motion Capture

The goal of motion capture is to determine the pose of the body throughout time. We can determine the pose of the body if we can associate each voxel to a body part. Joints labeling is presented in Fig. 1(c).

We propose a system based on simple and fast heuristics. This approach, less accurate than registration based methods, but nonetheless runs in real-time. Robustness is increased by using a multi-modal scheme composed of both shape and skin-parts analysis, temporal coherence, and human anthropometric constraints.

Our system runs on two steps: initialization and tracking; both use the same algorithm with different initial conditions. The initialization step (see Section 5) estimates anthropometric values, and the initial pose. Then using this information, the latter step tracks joint positions (see section 4). Our premises are that both hands and person's face are partially uncovered, that the torso is dressed, and that the clothing have a non-skin color. We present here some common notations the reader could refer to:

L_{x} denotes the length of body part x (see Fig. 1(b)),
D_{x} its orientation
R_{x} its radius (of sphere or cylinder).
J^n denotes the value of a quantity J (joint position, voxel set...) at frame n.
l and r indices denote respectively left and right side

$\mathcal{V}_x$ denotes a set of voxel,
$\mathcal{E}_{\mathcal{V}_x}$ its inertia ellipsoid
$Cog(\mathcal{V}_x)$ its gravity center.
$Q(i)$ denotes the Q quantity value at step i when dealing with iterative algorithms.

4 Body Parts Tracking

To track body parts, we assume that the previous body pose and anthropometric estimations are known. Using 3D shape estimation and 3D skin parts we track the human body parts in real-time. The tracking algorithm works by using active voxels $\mathcal{V}_{\text{act}}$. This set of voxels is initialized to all voxels $\mathcal{V}_{\text{all}}$ and updated at each step by removing voxels used to estimate body parts.

First, we estimate head joints. Then, since the torso is connected to the head, we track the torso. In the end we compute limb joints that are connected to torso.

4.1 Head Tracking

This step aims at finding T^n and B^n, respectively the positions of the top of the head and the connection point between head and neck at frame n. Head articulations are tracked using a sphere fitting algorithm inspired from the method proposed by [6]. To speed up the process we initialize the sphere center to the current face center, extracted from the set of skin voxels.

Let $\mathcal{V}^n_{\text{face}}$ be the face's voxels at the current frame. $\mathcal{V}^n_{\text{skin}}$ contains face and hands voxels. Using Temporal coherence criteria, $\mathcal{V}^n_{\text{face}}$ is the nearest[1] connex component of $\mathcal{V}^n_{\text{skin}}$ from the previous set of face voxels $\mathcal{V}^{n-1}_{\text{face}}$.

The center of the head C^n is computed by fitting a sphere $S(i)$ in $\mathcal{V}^n_{\text{act}}$ (see Fig. 3). The sphere $S(i)$ is defined by its center $\text{C}^n(i)$ and radius R_{head}.

Head Fitting Algorithm. $\text{C}^n(0)$ is initialized as the centroid of $\mathcal{V}^n_{\text{face}}$.

At step i of the algorithm, $\text{C}^n(i)$ is the centroid of the set $\mathcal{V}^n_{\text{head}}(i)$ of active voxels that lie into a sphere $\mathcal{S}(i-1)$ defined by its center $\text{C}^n(i-1)$ and its radius R_{head} (see Fig. 3(a)).

The algorithm iterates until step k when the position of C^n stabilizes, *i.e.* the distance between $\text{C}^n(k-1)$ and $\text{C}^n(k)$ falls below a threshold ϵ_{head}.

Head Joints Estimation. Knowing C^n position, B^n (respectively T^n) is computed as the lower (resp. upper) intersection between $\mathcal{S}(k)$ and the principal axis of $\mathcal{E}_{\mathcal{V}^n_{\text{head}}}$ (see Fig. 3(b)).

The back-to-front direction D^n_{b2f} is defined as the direction from C^n towards the centroid of $\mathcal{V}^n_{\text{face}}$ (note that voxels from the back of the head are not in $\mathcal{V}_{\text{skin}}$). At this point, we remove from $\mathcal{V}^n_{\text{act}}$ the set of elements that belongs to $\mathcal{V}^n_{\text{head}}$.

[1] Using point-ellipsoid euclidean distance.

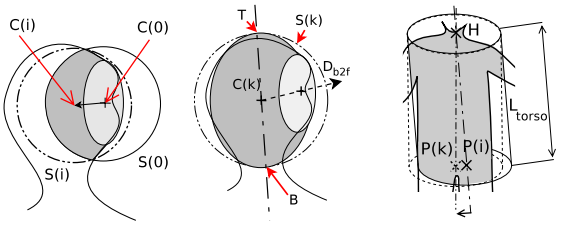


Fig. 3. (a) Sphere fitting (light gray denotes $\mathcal{V}^n_{\text{face}}$, dark gray denotes $\mathcal{V}^n_{\text{head}}(i)$), (b) joints estimation and (c) torso segmentation by cylinder fitting

4.2 Torso Tracking

This step aims at finding P^n the pelvis position by fitting a cylinder in $\mathcal{V}^n_{\text{act}}$. Estimating the torso shape by a cylinder provides a simple and fast method to localize the pelvis. Let $\mathcal{V}^n_{\text{torso}}$ be the set of voxels that describe the torso, they are initialized using voxels $\mathcal{V}^n_{\text{act}}$. At step i, the algorithm estimates D^n_{torso} by fitting a cylinder $\mathcal{CYL}(i-1)$ in $\mathcal{V}^n_{\text{torso}}$(i) (see Fig. 3(c)). $\mathcal{CYL}(i)$ has a cap anchored at B^n bottom of the head, as radius R_{torso}, its length is L_{torso} and its axis is D^n_{torso}(i).

Torso Fitting Algorithm. $\mathcal{V}^n_{\text{torso}}(0)$ is initialized with $\mathcal{V}^n_{\text{act}}$ and the vector from B^n to P^{n-1} defines $D^n_{\text{torso}}(0)$ initial value.

At step i, $\mathcal{V}^n_{\text{torso}}$(i) is computed as the set of elements from $\mathcal{V}^n_{\text{torso}}$(i-1) that lie in $\mathcal{CYL}(i-1)$. D^n_{torso}(i) is then the principal axis of $\mathcal{E}_{\mathcal{V}^n_{\text{torso}}(i)}$ (see Fig. 3(c)).

The algorithm iterates until step k when the distance between the axis of $\mathcal{CYL}(k)$ and the centroid of $\mathcal{V}^n_{\text{torso}}$(k) falls below a threshold ϵ_{torso}. P^n position is defined as the center of the lower cap of $\mathcal{CYL}(k)$.

Global Body Orientation. The top-down orientation D^n_{t2d} of the acquired subject is given by $\text{P}^n - \text{B}^n$. D_{b2f} was computed in 4.1. The left-to-right orientation D^n_{l2r} of the acquired subject is given by $D^n_{\text{l2r}} = D^n_{\text{t2d}} \times D^n_{\text{b2f}}$.

$\mathcal{V}^n_{\text{act}}$ is then updated by removing the elements that belong to $\mathcal{V}^n_{\text{torso}}$.

4.3 Hands and Forearms Tracking

We propose a simple and robust algorithm to compute the forearm joint positions. First, we compute hand positions from skin voxels. Helped by given anthropometric measurement of forearm length, we determine the elbows positions. Temporal coherence is used to compute their sides.

Let $\mathcal{V}^n_{\text{hand}}$ be the set of potential voxels of hands. $L_{\text{stat}}/2$ is an upper bound of arm length. $\mathcal{V}^n_{\text{hand}}$ is defined by the voxels of $\mathcal{V}^n_{\text{skin}} - \mathcal{V}^n_{\text{face}}$ that lie within a sphere defined by its center B^n and its radius $L_{\text{stat}}/2$. $\mathcal{V}^n_{\text{skin}}$ contains hands and face voxels. The different forearms configurations underlined Fig. 4 are:

Two Distinct Hands. $\mathcal{V}^n_{\text{hand}}$ contains several connex components. Let $\mathcal{V}^n_{\text{hand0}}$ and $\mathcal{V}^n_{\text{hand1}}$ be the two biggest, corresponding to the two hands with $\text{H}^n_x = Cog(\mathcal{V}^n_{\text{hand}x})$ with $x \in [0, 1]$.

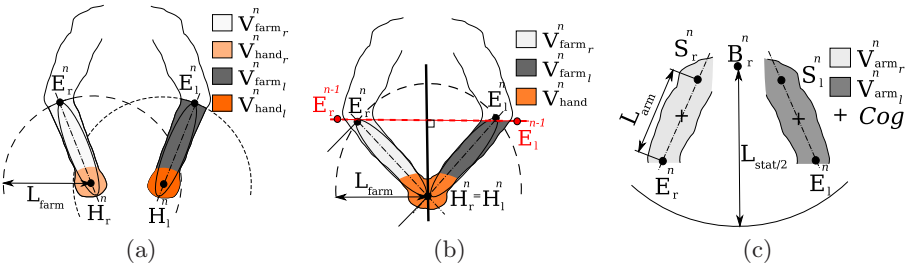


Fig. 4. Forearms tracking in different arms configuration: (a) represents two distinct forearms configuration and (b) represents jointed hands configuration. (c) Illustration of the shoulders tracking algorithm.

Forearms have constant length L_{farm} across time. The potential voxels for forearm$_x$ are the voxels from $\mathcal{V}^n_{\text{act}}$ which lies within a sphere of radius L_{farm}, centered in H^n_x. The connex component of these voxels which contains H^n_x represents the forearm$_x$. Let $\mathcal{V}^n_{\text{farm}\,x}$ be this connex component; there are two possible cases to identify elbow.

If forearms did not collide *i.e.* $\mathcal{V}^n_{\text{farm0}} \cap \mathcal{V}^n_{\text{farm1}} = \emptyset$, then we use the principal axis of $\mathcal{E}_{\mathcal{V}^n_{\text{farm}\,x}}$ and L_{farm} to compute the elbow position E^n_x. The sides are computed using temporal coherence criteria: the side of the forearm$_x$ is the same than the closest forearm computed at the previous frame. This configuration is underlined Fig. 4(a).

Otherwise forearms collide and $\mathcal{V}^n_{\text{farm0}} \cap \mathcal{V}^n_{\text{farm1}} \neq \emptyset$. In that case, we first identify the hand sides by the property of constant forearms length. H^n_x is right sided if

$$||d(\text{H}^n_x, \text{E}^{n-1}_r) - L_{\text{farm}}|| < ||d(\text{H}^n_x, \text{E}^{n-1}_l) - L_{\text{farm}}||, \tag{1}$$

else H^n_x is left sided. The voxels v_i of $\mathcal{V}^n_{\text{farm0}} \cup \mathcal{V}^n_{\text{farm1}}$ are segmented in two parts $\mathcal{V}^n_{\text{farm}r}$ and $\mathcal{V}^n_{\text{farm}l}$ using "point to line mapping" algorithm (see 4.5). If v_i is more close to $[\text{H}^n_r \text{E}^{n-1}_r]$ than to $[\text{H}^n_l \text{E}^{n-1}_l]$, v_i is added on $\mathcal{V}^n_{\text{farm}r}$. Else v_i is added on $\mathcal{V}^n_{\text{farm}l}$. Principal axis of $\mathcal{E}_{\mathcal{V}^n_{\text{farm}r}}$,$\mathcal{E}_{\mathcal{V}^n_{\text{farm}l}}$ and L_{farm} are used to compute E^n_r and E^n_l.

One Hand or Jointed Hands. $\mathcal{V}^n_{\text{hand}}$ contains only one connex component and it corresponds to jointed hands or to only one hand (the other is not visible). We use the temporal coherence to disambiguate these two cases.

If $\text{H}^{n-1}{}_r$ and $\text{H}^{n-1}{}_l$ are close to $\mathcal{V}^n_{\text{hand}}$, then the hands are jointed (Fig. 4(b)) and $\text{H}^n{}_r = \text{H}^n{}_l = Cog(\mathcal{V}^n_{\text{hand}})$ and we compute $\mathcal{V}^n_{\text{farm}}$ as proposed previously. We segment $\mathcal{V}^n_{\text{farm}}$ in two parts $\mathcal{V}^n_{\text{farm}r}$ and $\mathcal{V}^n_{\text{farm}l}$ by the orthogonal plane to $[\text{E}^{n-1}_r \text{E}^{n-1}_l]$ containing $\text{H}^n{}_l$. Principal axis of $\mathcal{E}_{\mathcal{V}^n_{\text{farm}r}}$, $\mathcal{E}_{\mathcal{V}^n_{\text{farm}l}}$ and L_{farm} are used to compute E^n_r and E^n_l.

Otherwise the closest hand $\text{H}^{n-1}{}_x$ to $\mathcal{V}^n_{\text{hand}}$ is used to compute the side of $\text{H}^n{}_x$ and $\text{H}^n{}_x = Cog(\mathcal{V}^n_{\text{hand}})$. We compute $\mathcal{V}^n_{\text{farm}}$ as proposed previously and its principal axis of inertia is used to compute $\text{E}^n{}_x$.

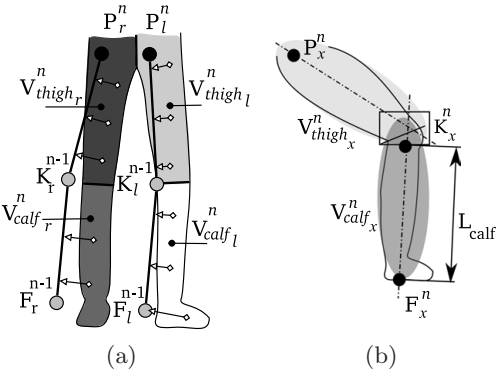


Fig. 5. (a) the "binding" step of legs tracking and (b) legs articulations estimation

No Visible Hand. $\mathcal{V}^n_{\text{hand}}$ is empty, then no hand is visible. We take back the positions computed at the $n-1$ frame to the current frame.

In all cases $\mathcal{V}^n_{\text{act}}$ is updated by removing the elements that belong to forearms or hands.

4.4 Shoulders Tracking

We have estimated articulations positions of the head, the torso, the hands and the elbows. To finalize upper body tracking, we compute shoulder positions. As we argue that arms are in a sphere centered on bottom head, with a radius of $L_{\text{stat}}/2$, then voxels of $\mathcal{V}^n_{\text{act}}$ which are in this sphere, contain arms voxels and noise voxels. Let $\mathcal{V}^n_{\text{arms}}$ be the set of these voxels.

The elbows are on one extremity of the arms, thus the second extremity of the arms corresponds to the shoulders. We know the current position of elbow, then we determine arm voxels. Let $\mathcal{V}^n_{\text{arm}\,x}$ (where x corresponds to the side) be the closest[2] connex component of $\mathcal{V}^n_{\text{arms}}$ to $\mathrm{E}^n{}_x$. Furthermore arm length L_{arm} is constant, then current shoulder position $\mathrm{S}^n{}_x$ for the x side is given by:

$$\mathrm{S}^n{}_x = \mathrm{E}^n{}_x + \frac{Cog(\mathcal{V}^n_{\text{arm}\,x}) - \mathrm{E}^n{}_x}{|Cog(\mathcal{V}^n_{\text{arm}\,x}) - \mathrm{E}^n{}_x|} L_{\text{arm}}. \tag{2}$$

Shoulders tracking algorithm is underlined Fig.4(c).

$\mathcal{V}^n_{\text{act}}$ is updated by removing the elements that belong to each arm.

4.5 Legs Tracking

Until now all body parts but the legs have been estimated, hence $\mathcal{V}^n_{\text{act}}$ contains the legs voxels. Our leg joints extraction is inspired from "point to line mapping" process used to bind an animation skeleton on a 3D mesh [24]. The elements of

[2] In term of euclidean distance.

$\mathcal{V}^n_{\text{act}}$ are split up into four sets $\mathcal{V}^n_{\text{thigh}_l}$, $\mathcal{V}^n_{\text{calf}_l}$, $\mathcal{V}^n_{\text{thigh}_r}$ and $\mathcal{V}^n_{\text{calf}_r}$ depending of their euclidean distance to segments $[\mathrm{P}_l^{n-1}, \mathrm{K}_l^{n-1}]$, $[\mathrm{K}_l^{n-1}, \mathrm{F}_l^{n-1}]$, $[\mathrm{P}_r^{n-1}, \mathrm{K}_r^{n-1}]$, and $[\mathrm{K}_r^{n-1}, \mathrm{F}_r^{n-1}]$ (see Fig. 5(a)). For the left/right side x, we compute the inertia ellipsoid $\mathcal{E}_{\mathcal{V}^n_{\text{calf}_x}}$ (let Ex_0 and Ex_1 be its extrema points) and the inertia ellipsoid $\mathcal{E}_{\mathcal{V}^n_{\text{thigh}_x}}$.

The knee is the intersection point between thigh and calf (Fig. 5(b)), hence the foot position ${\mathrm{F}^n}_x$ is given by $\mathcal{E}_{\mathcal{V}^n_{\text{calf}_x}}$ point farthest from the inertia ellipsoid of $\mathcal{V}^n_{\text{thigh}_x}$ (let say it's Ex_1). Then knee is aligned on $[Ex_0 Ex_1]$, Ex_0 sided, at a L_{calf} distance of ${\mathrm{F}^n}_x$. Hip position ${\mathrm{P}^n}_x$ is given by the farthest extrema point of $\mathcal{E}_{\mathcal{V}^n_{\text{thigh}_x}}$ from the inertia ellipsoid of $\mathcal{V}^n_{\text{calf}_x}$, corrected to be at a L_{thigh} distance of ${\mathrm{K}^n}_x$.

5 Body Parts Initialization

In this section we present our techniques to estimate the anthropometric measures and the initial pose of the body. We can classify in three the methods presented in the literature regarding the initial pose estimation. The first kind [12], the anthropometric measurements and initial pose are entered manually. Another class of methods need a fixed pose, like T-pose [11], these methods work in real-time. The last class of methods are fully automatic [6] and do not need a specific pose, but are not real-time. Our approach is a real-time and fully automated one for any kind of movement as long as the person filmed is standing up, hands below the level of the head, and feet not joined. After anthropometric estimations, our method computes each body parts parameters sequentially according to the tracking steps.

Anthropometric Measurements. Several studies that include the anthropometric data [25,26,27] are used to develop ratio estimations. Statistical analysis is performed, including fitting to normal distribution. We propose simplified anthropometric ratios, whose accuracy is sufficient for human-machine interactions. Let L_{stat} be the acquired human body length, estimated as the maximum distance from foreground voxels to floor plane. Hence, knowing L_{stat}, guesses for anthropometric measures are given by these ratios:

$$R_{\text{head}} \approx L_{\text{stat}}/16,\ L_{\text{torso}} \approx 3\ L_{\text{stat}}/8,\ L_{\text{calf}} \approx L_{\text{stat}}/4,$$
$$L_{\text{farm}} \approx L_{\text{stat}}/6,\ \ L_{\text{arm}} \approx L_{\text{stat}}/6,\ L_{\text{thigh}} \approx L_{\text{stat}}/4.$$

As in the tracking step, active set of voxels $\mathcal{V}_{\text{act}}$ is initialized by all voxels $\mathcal{V}_{\text{all}}$.

Head Initialization. This step aims at finding T^0 and B^0. From our initialization hypothesis, the face's voxels $\mathcal{V}^0_{\text{face}}$ of acquired subject are defined by the topmost connex component among $\mathcal{V}^0_{\text{skin}}$. Then head tracking algorithm (section 4.1) is applied to compute T^0 and B^0, without estimation of the face position step. $\mathcal{V}^0_{\text{act}}$ is updated by removing elements belonging to $\mathcal{V}^0_{\text{head}}$.

Torso Initialization. The torso fitting algorithm (section 4.2) is applied using $\mathcal{V}^0_{\text{act}}$ as initial value for $\mathcal{V}^0_{\text{torso}}(0)$. $D^0_{\text{torso}}(0)$ is initialized as the vector from N^0 toward the centroid of $\mathcal{E}_{\mathcal{V}^0_{\text{act}}(0)}$. Pelvis position P^0, D^0_{t2d} and D^n_{l2r} are then computed. $\mathcal{V}^0_{\text{act}}$ is updated by removing the elements that belong to $\mathcal{V}^0_{\text{torso}}$.

Arms Initialization. We initialize the hand and the forearm positions using the tracking algorithm presented Section 4.3. Since we have no previous arms positions, we can only compute forearm positions when there is two distinct forearms. Having verified this criteria, we can compute ${\text{H}^0}_r$, ${\text{H}^0}_l$,${\text{E}^0}_r$ and ${\text{E}^0}_l$. $\mathcal{V}^0_{\text{act}}$ is updated by removing the elements that belong to the forearms. Shoulders positions ${\text{S}^0}_r$ and ${\text{S}^0}_l$ are initialized directly using the shoulder tracking algorithm presented Section 4.4.

Legs Initialization. Tracking algorithm outlined in Section 4.5 need the legs' previous position. We simulate them by a coarse estimation of knees, feet and hips articulations, then we compute more precise position of the legs articulations using the legs tracking algorithm.

$\mathcal{V}^0_{\text{act}}$ contains the voxels that haven't been used for any other parts of the body. First, we compute the set of connex components from elements of $\mathcal{V}^0_{\text{act}}$ having their height below $L_{\text{stat}}/8$. If there is less than 2 connex components, we assume that feet are joined and can't be distinguished. Otherwise, we use the two major connex components $\mathcal{V}^0_{\text{foot}l}$ and $\mathcal{V}^0_{\text{foot}r}$. Left and right assignation of voxel's set is done using the left-to-right vector D_{l2r}. For the left/right side x, let v_x be the vector from P^0 to the centroid of $\mathcal{V}^0_{\text{foot}x}$. Knee and foot joints are determined using the following equations:

$$\text{K}^{-1}{}_x = \text{P}^0 + v_x \frac{L_{\text{thigh}}}{|v_x|}, \tag{3}$$

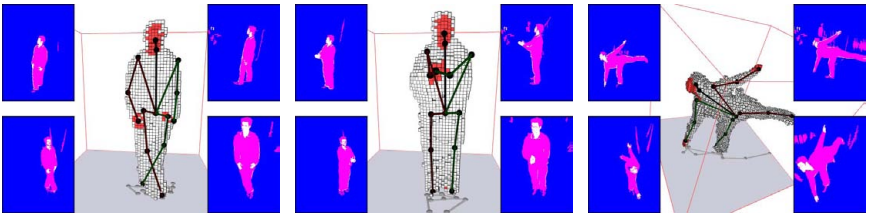
$$\text{F}^{-1}{}_x = \text{P}^0 + v_x \frac{L_{\text{thigh}} + L_{\text{calf}}}{|v_x|}. \tag{4}$$

We estimate hips previous positions P^{-1}_l and P^{-1}_r as P^0. Finally we compute ${\text{F}^0}_r$, ${\text{K}^0}_r$, ${\text{F}^0}_l$ and ${\text{K}^0}_l$ using the legs tracking algorithm.

6 Results

We now present results from our system. Fig. 2(a) shows the system configuration. The infrastructure of acquisition is composed of four Phillips webcams (SPC900NC) connected to a single PC (CPU: P4 3.2ghz,GPU: NVIDIA Quadro 3450). Webcams produce images of resolution 320×240 at 30fps.

Our method has been applied on different persons doing fast and challenging motions. By using shape *and* skin color analysis, our algorithm can handle challenging poses like the one shown in Fig. 6(a). This pose is difficult because the topology of the 3D reconstructed shape is not coherent with the topology of the human 3D shape. The temporal coherence is the key to success for the pose presented Fig. 6(b). This underlines the case of jointed hands (4.3) which is



(a) (b) (c)

Fig. 6. (a) (b) and (c) underline results for challenging poses. The user recovered pose is presented as an animation skeleton having right sided parts in red and left sided parts in green. Shape voxels are presented in white and skin voxels in red.

successfully recognized. A very difficult pose is shown in Fig. 6(c) and is successfully recovered by our system. Images from Fig. 7 demonstrate that our system works for large range of motions.

Additional results are included in the supplementary video. It presents long sequences with rapid and complex motions and shows the robustness of our approach.

Our current experimental implementation can track more than 30 poses per second on a single computer, which is faster than the webcams acquisition frame rate. An optimized implementation can be usable for current generation of home entertainment computers. As our algorithm is based on 3D reconstruction, it is independent of the number of cameras used, but it depends on the voxel grid resolution. We reconstruct a voxel grid composed by 64^3 voxels in a 6 m^3 box which gives an approximate resolution of $2.7 \times 2.7 \times 2.7$ cm per voxel. This resolution is enough for human-machine interfaces in the field of entertainment.

Table 1 shows some speed comparison with other current real-time tracking techniques. Our approach furthermore offers the best frame rate with only one commodity PC and an original fully automated and real-time initialization.

Our motion capture system is based on a Shape-From-Silhouette algorithm. This algorithm computes an object 3D shape estimation from its silhouettes. The result directly depends on the silhouette segmentation quality, which is always an opened problem of the computer vision science. If the silhouette mask contains

Table 1. Speed comparison with current techniques. It is made with results published by peers.

Reference	Nbr. of Joints	Real-Time Init.	Tracking Frame Rate	Nbr. of CPUs
Our	15	Yes	$\approx$ 30 fps	1
[8]	21	No	$\approx$ 25 fps	> 8
[10]	15	No	$\approx$ 15 fps	1
[7]	19	No	$\approx$ 10 fps	1
[9]	15	No	$\approx$ 10 fps	1
[12]	15	No	$\approx$ 1 fps	1

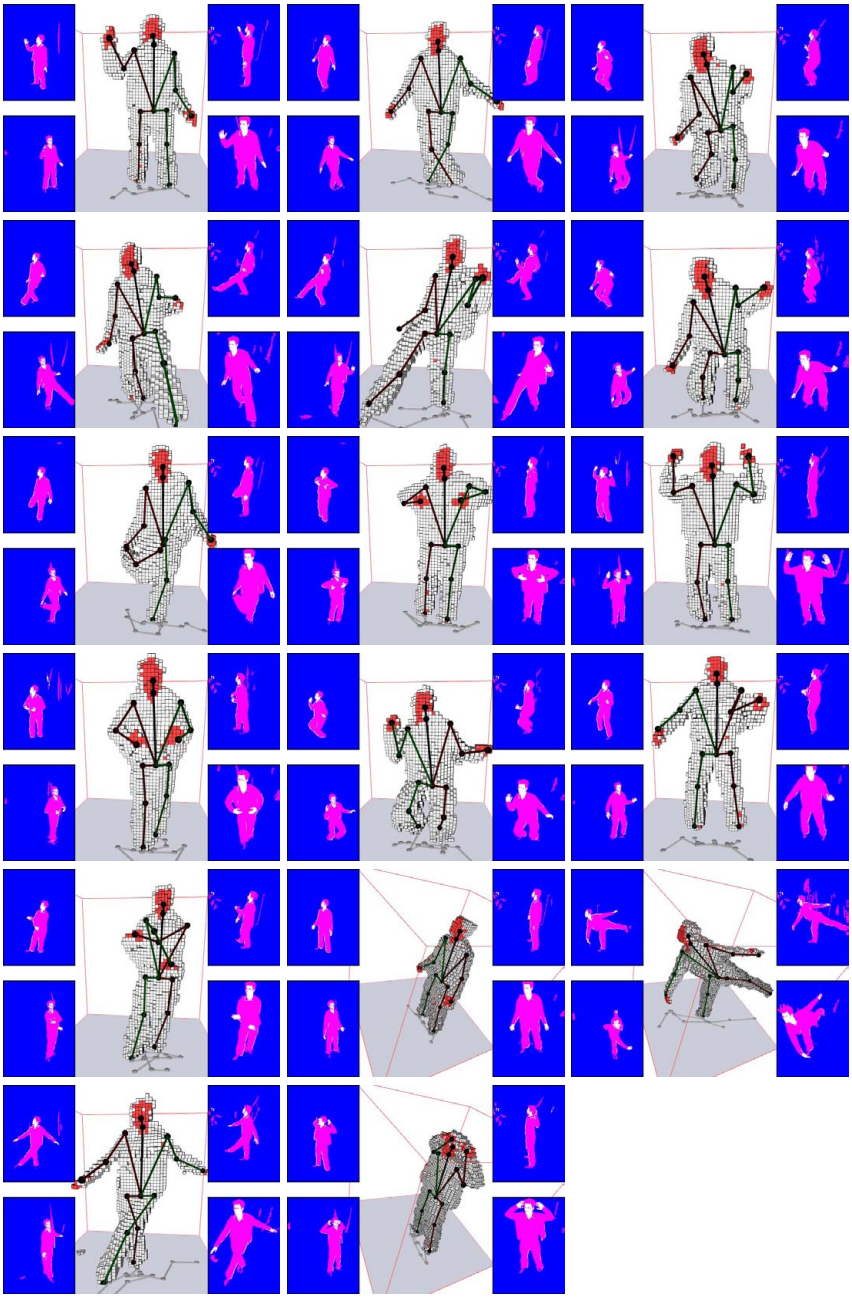


Fig. 7. Results for a wide range of movements

some noises like camera noise or object shadows, the volume reconstruction will be very noised. Thus the results of the motion capture will be worse. But our method is also based on a skin segmentation which is a more robust faced to camera noise. Then the hand and head articulations are more noise-resistant, than others articulations.

The segmentation we have selected is based on a skin-colored stochastic learning from colored image set. It is important to make the leaning process on a big data set, with different kind of skin sample. If the skin sampling is biased then the system will provide worse results, especially when the skin color of the person filmed is not learned.

7 Conclusion

In this paper, we describe a new marker-free system of human motion capture that works with few cameras (three or more) and a single computer. The system is based on both a 3D shape analysis, human morphology constraints, and a 3D shape skin segmentation. It is fully automated and runs in real-time. Combining different 3D information, the approach is robust to self-occlusion. It estimates the fifteen main human body joints at about more than 30 frames per second. This frame-rate is well suited for a new generation of human machine interactions. Our system is based on simple heuristics and the results obtained show that the method is robust and promising.

The actual system provides real-time motion capture for one person. Future work aims at providing motion capture for serveral persons filmed together in the same area, even if they are in contact.

References

1. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Comput. Vis. Image Underst. 104(2), 90–126 (2006)
2. Agarwal, A., Triggs, B.: Monocular human motion capture with a mixture of regressors. In: CVPR 2005, vol. 72, IEEE Computer Society, Los Alamitos (2005)
3. Chen, Y., Lee, J., Parent, R., Machiraju, R.: Markerless monocular motion capture using image features and physical constraints. In: CGI 2005. Proceedings of the Computer Graphics International 2005, pp. 36–43. IEEE Computer Society Press, Los Alamitos (2005)
4. Micilotta, A., Ong, E., Bowden, R.: Real-time upper body detection and 3D pose estimation in monoscopic images. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3953, pp. 139–150. Springer, Heidelberg (2006)
5. de Aguiar, E., Theobalt, C., Magnor, M., Theisel, H., Seidel, H.P.: M3: Marker-free model reconstruction and motion tracking from 3d voxel data. In: Cohen-Or, D., Ko, H.S., Terzopoulos, D., Warren, J. (eds.) PG 2004. 12th Pacific Conference on Computer Graphics and Applications, pp. 101–110. IEEE Computer Society Press, Los Alamitos (2004)
6. Mikic, I., Trivedi, M., Hunter, E., Cosman, P.: Human body model acquisition and tracking using voxel data. Int. J. Comput. Vision 53(3), 199–223 (2003)

7. Tangkuampien, T., Suter, D.: Real-Time Human Pose Inference using Kernel Principal Component Pre-image Approximations. In: Proceedings BMVC 2006, pp. 599–608 (2006)
8. Okada, R., Stenger, B., Ike, T., Kondoh, N.: Virtual Fashion Show Using Real-Time Markerless Motion Capture. In: Narayanan, P.J., Nayar, S.K., Shum, H.-Y. (eds.) ACCV 2006. LNCS, pp. 801–810. Springer, Heidelberg (2006)
9. Caillette, F., Galata, A., Howard, T.: Real-Time 3-D Human Body Tracking using Variable Length Markov Models. In: Proceedings BMVC 2005, vol. 1 (September 2005)
10. Caillette, F., Howard, T.: Real-Time Markerless Human Body Tracking Using Colored Voxels and 3-D Blobs. In: ISMAR. Proceedings of the 3rd IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 266–267. IEEE Computer Society Press, Los Alamitos (2004)
11. Fua, P., Gruen, A., D'Apuzzo, N., Plankers, R.: Markerless Full Body Shape and Motion Capture from Video Sequences. In: Symposium on Close Range Imaging, International Society for Photogrammetry and Remote Sensing, Corfu, Greece (2002)
12. Ménier, C., Boyer, E., Raffin, B.: 3d skeleton-based body pose recovery. In: Proceedings of the 3rd International Symposium on 3D Data Processing, Visualization and Transmission, Chapel Hill (USA) (June 2006)
13. Laurentini, A.: The visual hull concept for silhouette-based image understanding. IEEE Trans. Pattern Anal. Mach. Intell. 16(2), 150–162 (1994)
14. Franco, J.S., Boyer, E.: Exact polyhedral visual hulls. In: Proceedings BMVC 2003, Norwich, UK, pp. 329–338 (September 2003)
15. Hasenfratz, J.M., Lapierre, M., Sillion, F.: A real-time system for full body interaction with virtual worlds. In: Eurographics Symposium on Virtual Environments, pp. 147–156 (2004)
16. Cheung, K.M., Kanade, T., Bouguet, J.Y., Holler, M.: A real time system for robust 3d voxel reconstruction of human motions. In: CVPR 2000. Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 714–720. IEEE Computer Society Press, Los Alamitos (2000)
17. Michoud, B., Guillou, E., Bouakaz, S.: Shape from silhouette: Towards a solution for partial visibility problem. In: Eurographics 2006 Short Papers Preceedings, pp. 13–16 (September 2006)
18. Zhang, Z.: Flexible camera calibration by viewing a plane from unknown orientations. In: ICCV, pp. 666–673 (1999)
19. Joshi, N.: Color calibration for arrays of inexpensive image sensors. Technical report, Stanford University (2004)
20. Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: Principles and practice of background maintenance. In: ICCV (1), pp.255–261 (1999)
21. Vezhnevets, V., Sazonov, V., Andreeva, A.: A survey on pixel-based skin color detection techniques. In: Proceedings of Graphicon-2003 (2003)
22. Segal, M., Korobkin, C., van Widenfelt, R., Foran, J., Haeberli, P.: Fast shadows and lighting effects using texture mapping. In: Proceedings of SIGGRAPH (1992)
23. Culbertson, W.B., Malzbender, T., Slabaugh, G.G.: Generalized voxel coloring. In: Workshop on Vision Algorithms, pp. 100–115 (1999)

24. Sun, W., Hilton, A., Smith, R., Illingworth, J.: Layered animation of captured data. The Visual Computer 17(8), 457–474 (2001)
25. Dreyfuss, H., Tilley, A.R.: The Measure of Man and Woman: Human Factors in Design. John Wiley & Sons, Chichester (2001)
26. Motmans, R., Ceriez, E.: Anthropometry table, Ergonomie RC, Leuven, Belgium (2005)
27. Anthropometric source book. Vol. 1, Anthropometry for designers, NASA Report Number: NASA-RP-1024, S-479-VOL-1 (1978)

Modeling Human Locomotion with Topologically Constrained Latent Variable Models

Raquel Urtasun[1], David J. Fleet[2], and Neil D. Lawrence[3]

[1] Massachusetts Institute of Technology, Cambridge, MA 02139 USA
[2] University of Toronto, Canada M5S 3H5
[3] School of Computer Science, University of Manchester, M13 9PL, U.K.

Abstract. Learned, activity-specific motion models are useful for human pose and motion estimation. Nevertheless, while the use of activity-specific models simplifies monocular tracking, it leaves open the larger issues of how one learns models for multiple activities or stylistic variations, and how such models can be combined with natural transitions between activities. This paper extends the Gaussian process latent variable model (GP-LVM) to address some of these issues. We introduce a new approach to constraining the latent space that we refer to as the locally-linear Gaussian process latent variable model (LL-GPLVM). The LL-GPLVM allows for an explicit prior over the latent configurations that aims to preserve local topological structure in the training data. We reduce the computational complexity of the GPLVM by adapting sparse Gaussian process regression methods to the GP-LVM. By incorporating sparsification, dynamics and back-constraints within the LL-GPLVM we develop a general framework for learning smooth latent models of different activities within a shared latent space, allowing the learning of specific topologies and transitions between different activities.

1 Introduction

Modeling human motion is important for computer vision, computer graphics, orthopedics, sports and the entertainment industry (e.g., movies, computer games). In computer vision, for example, pose and motion models help resolve ambiguities in monocular people tracking. Nevertheless, learning human motion models is challenging since we must learn models from relatively sparse training data while avoiding overfitting, and the models must generalize to previously unobserved styles. One common approach has been to focus on *activity-specific* models [4,12,19,26]. This greatly simplifies learning, but leaves open the larger issues of how one learns models for a wide range of activities and stylistic variations, and how such models are combined with natural transitions from one activity to another [4,26]. To address these problems, this paper introduces a new form of Gaussian process latent variable model (GPLVM) [10] for learning models of different activities within a shared latent space, along with transitions between activities.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 104–118, 2007.

The GPLVM and the Gaussian process dynamical model (GPDM) have been used to learn generative models for human motion, including walking, running, and baseball pitching, proving useful for people tracking and computer animation [5,14,24,27,26,28]. Nonetheless, when there are large stylistic variations or different activities, the Gaussian process framework can produce models that are not smooth, fail to generalize well to nearby motions, and are therefore unsuitable for tracking and simulation [25].

One major problem lies with the formulation of the GPLVM. In its standard form the GPLVM ensures that the mapping from the latent space to the pose space is smooth, but not the map from pose space to latent space. That is, nearby latent positions will generate similar poses, but similar poses do not necessarily map to nearby latent positions. It is interesting to contrast this behaviour with other methods for nonlinear dimensionality reduction, such LLE [18], which aim to preserve the local topological structure of the training data. Unfortunately such methods for embedding do not provide generative models.

To preserve topological structure, the back-constrained GPLVM [8], introduces smoothness by constraining the latent positions to be a smooth function of the data space. The first goal of this paper is to generalize this approach with the formulation of a generative latent variable model whose prior, like the objective function used for LLE, aim to preserve local topological structure of the training data. We refer to this model as the locally-linear Gaussian process latent variable model (LL-GPLVM).

The second goal of this paper is to explore the ways in which one can constrain learning to produce topologically meaningful latent models. Most existing methods for learning motion models ignore useful prior information about human motion, such as the cyclic nature of locomotion, the physical laws at play, and the limited ways people transition between activities. Wang et al. [29] proposed a multifactor model for learning distributions of styles of human motion, parametrizing the space of human motion styles by a small number of low-dimensional factors, such as identity and gait. Their multifactor model can be viewed as a special case of the GPLVM, where the covariance function of the GP is a product of covariance functions for the individual factors. Here we take a complementary approach, incorporating prior knowledge about different activities and transitions between them within the LL-GPLVM and the back-constrained GPLVM. Importantly, transitions between different activities do *not* have to be present in the training data to be learned.

Finally, the application of the GPLVM has also been limited to small training sets because of its computational complexity, $O\left(N^3\right)$, where N is the number of data points. One way to reduce this computational burden is to use the informative vector machine [9] to obtain a sparse representation; however, by using just a subset of the data, this approach ignores valuable training data, and is not guarrenteed to converge. We review the use of sparse Gaussian process regression in the GPLVM [11] in the context of human motion data. This results in a more effective algorithm, which converges to a generative model that depends on the entire data set.

2 Related Work

It has long been accepted that useful representations for visual analysis should capture the intrinsic structure of the data. This is the motivation to separate style and content with multilinear models for example [23,6,29]. Such models embody the tacit assumptions that there are a number of somewhat independent causes generating the data, and that these should be represented with separate dimensions. More generally, to learn such models, it is also often important to exploit all available prior knowledge about the domain.

Elgammal et al. [4] showed that for walking, nearby poses on a single person's gait are more similar than poses from other peoples'. As a consecuence, even for a single activity like walking, it can be challenging to learn a coherent model for multiple people, in which the motions are aligned and can be interpolated smoothly to generate new plausible styles. This becomes more difficult with multiple activities, where we wish to interpolate over different people's poses if their style is sufficiently similar, or to transition from one activity to another.

Perhaps the simplest approach to the general modeling problem is to use non-parametric models (e.g., [1,7,19]). This approach was used successfully for multi-activity character animation and tracking, but it does not easily generalize to nearby activities and styles, so one must have an enormous training corpus.

Classical multilinear models [23,6] produce style-content separation but are not suitable for many types of motions such as those with cyclic behaviour or nonlinearities [4]. Moreover, they have not been extended to handle transitions between activities. Elgammal et al. [4] learn a nonlinear model with stylistic variation (multiple people) for walking by first building individual models, and then using nonlinear regression to align the manifolds to build a unified model. While evocative, it is not clear whether this piecemeal approach will scale to more complex motions, multiple motions, or to many people. Wang et al. [29] propose a multifactor model for learning distributions of styles of human locomotion (walking, running and striding) but they do not explicitly model transitions. Somewhat smooth transitions can be achieved by linearly inerpolating the style factors.

Switching LDS [13,15] and HMMs [3] can represent multiple motions and diverse styles. Switching models have attractive properties that are somewhat complementary to the GP-LVM, but at present they require large amount of training data. Smoothness of the global model is another issue, as is the intractability of learning. In practice, it is sometimes argued that current switching models do not generalize well beyond the training data.

GP models [26,14,28] have proven very effective when dealing with a single motion type. Nevertheless, as discussed above, they have problems when dealing with multiple motions and different styles [25]. The aim of this paper is to discuss recently developed formulations that help address these issues.

3 Gaussian Process Latent Variable Models (GP-LVM)

We begin with a brief review of the GP-LVM. The GP-LVM represents a high-dimensional data set, $\mathbf{Y}$, through a low dimensional latent space, $\mathbf{X}$, and a Gaussian process mapping from the latent space to the data space. Let $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_N]^T$ be a matrix in which each row is a single training datum, $\mathbf{y}_i \in \Re^D$. Let $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^T$ denote the matrix whose rows represent the corresponding positions in latent space, $\mathbf{x}_i \in \Re^d$. The GPLVM is a generative model of the data

$$\mathbf{y}_t = \sum_j \mathbf{b}_j f_j(\mathbf{x}_t) + \mathbf{n}_{y,t} \tag{1}$$

for weights $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, ...]$, basis functions f_j and additive zero-mean white Gaussian noise $\mathbf{n}_{y,t}$. Given a covariance function for the Gaussian process, $k_Y(\mathbf{x}, \mathbf{x}')$, the likelihood of the data given the latent positions is,

$$p(\mathbf{Y} \mid \mathbf{X}, \bar{\beta}) \;=\; \frac{1}{\sqrt{(2\pi)^{ND}|\mathbf{K}_Y|^D}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{Y}^T\right)\right), \tag{2}$$

where $\mathbf{K}_Y$ is known as the kernel matrix, and $\bar{\beta}$ is a vector of kernel hyperparameters. The elements of the kernel matrix are defined by the covariance function, $(\mathbf{K}_Y)_{i,j} = k_Y(\mathbf{x}_i, \mathbf{x}_j)$. A common choice is the radial basis function (RBF), $k_Y(\mathbf{x}, \mathbf{x}') = \beta_1 \exp(-\frac{\beta_2}{2}||\mathbf{x} - \mathbf{x}'||^2) + \frac{\delta_{\mathbf{x},\mathbf{x}'}}{\beta_3}$, where $\bar{\beta} = \{\beta_1, \beta_2, ...\}$ are the kernel hyperparameters that determine the output variance, the RBF support width, and the variance of the additive noise. Learning in the GP-LVM consists of maximizing (2) with respect to the latent space configuration, $\mathbf{X}$, and the hyperparameters, $\bar{\beta}$.

When one has time-series data, $\mathbf{Y}$ represents a sequence of observations, and it is natural to augment the GP-LVM with an explicit dynamical model. For example, the Gaussian Process Dynamical Model (GPDM) models the latent sequence as a latent stochastic process with a Gaussian process prior [28], i.e.,

$$p(\mathbf{X} \mid \bar{\alpha}) \;=\; \frac{p(\mathbf{x}_1)}{\sqrt{(2\pi)^{(N-1)d}|\mathbf{K}_X|^d}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_X^{-1}\mathbf{X}_{out}\mathbf{X}_{out}^T\right)\right) \tag{3}$$

where $\mathbf{X}_{out} = [\mathbf{x}_2, ..., \mathbf{x}_N]^T$, $\mathbf{K}_X \in \Re^{(N-1)\times(N-1)}$ is the kernel matrix constructed from $\mathbf{X}_{in} = [\mathbf{x}_1, ..., \mathbf{x}_{N-1}]$, $\mathbf{x}_1$ is given an isotropic Gaussian prior and $\bar{\alpha}$ are the kernel hyperparameters. Below we use an RBF kernel for $\mathbf{K}_X$. Like the GP-LVM the GPDM provides a generative model for the data, but additionally it provides one for the dynamics. One can therefore predict future observation sequences given past observations, and simulate new sequences.

4 Sparse Approximations

By exploiting a sparse approximation to the full Gaussian process it is usually possible to reduce the computational complexity from an often prohibitive

$O\left(N^3\right)$ to a more manageable $O\left(k^2 N\right)$, where k is the number of points retained in the sparse representation [11]. A large body of recent work has been focussed on approximating the covariance function with a low rank approximation [9,16,21]. All approximations involve augmenting the function values at the training points, $\mathbf{F} \in \Re^{N \times d}$, with $\mathbf{F} = [\mathbf{f}_1, ..., \mathbf{f}_N]^T$ and the function values at the test points, $\mathbf{F}_* \in \Re^{\infty \times d}$, by an additional set of variables, $\mathbf{U} \in \Re^{k \times d}$, called inducing variables [16]. The number of these variables, k, can be specified by the user.

The factorisation of the likelihood across the columns of $\mathbf{Y}$ allows us to focus on one column of $\mathbf{F}$ without loss of generality. We therefore consider function values at $\mathbf{f} \in \Re^{N \times 1}$, $\mathbf{f}_* \in \Re^{\infty \times 1}$ and $\mathbf{u} \in \Re^{k \times 1}$. These variables are considered to be jointly Gaussian distributed with $\mathbf{f}$ and $\mathbf{f}_*$ such that

$$p\left(\mathbf{f}, \mathbf{f}_*\right) = \int p\left(\mathbf{f}, \mathbf{f}_* | \mathbf{u}\right) p\left(\mathbf{u}\right) d\mathbf{u},$$

where the prior distribution over the inducing variables is given by a Gaussian process,

$$p\left(\mathbf{u}\right) = N\left(\mathbf{u} | \mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\right),$$

with a covariance function given by $\mathbf{K}_{\mathbf{u},\mathbf{u}}$. This covariance is constructed on a set of inputs $\mathbf{X}_\mathbf{u}$ which may or may not be a subset of $\mathbf{X}$. Assume that the variables associated with the training data, $\mathbf{f}$, are conditionally independent of those associated with the test data, $\mathbf{f}_*$, given the inducing variables, $\mathbf{u}$[16]

$$p\left(\mathbf{f}, \mathbf{f}_*, \mathbf{u}\right) = p\left(\mathbf{f} | \mathbf{u}\right) p\left(\mathbf{f}_* | \mathbf{u}\right) p\left(\mathbf{u}\right),$$

where

$$p\left(\mathbf{f} | \mathbf{u}\right) = N\left(\mathbf{f} | \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}}\right)$$

is the training conditional and

$$p\left(\mathbf{f}_* | \mathbf{u}\right) = N\left(\mathbf{f}_* | \mathbf{K}_{*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},*}\right)$$

is the test conditional. $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ is the covariance function computed between the training inputs, $\mathbf{X}$, and the inducing variables, $\mathbf{X}_\mathbf{u}$, $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ is the symmetric covariance between the training inputs, $\mathbf{K}_{*,\mathbf{u}}$ is the covariance function between the test inputs and the inducing variables and $\mathbf{K}_{*,*}$ is the symmetric covariance function the test inputs. This decomposition does not in itself entail any approximations: the approximations are introduced through assumptions about the form of these distributions. Here we consider the Fully Independence approximation (FITC) of Snelson and Ghahramani [21], where the training conditional is assumed to be

$$q\left(\mathbf{f} | \mathbf{u}\right) = N\left(\mathbf{f}_{(j)} | \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, diag\left(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}}\right)\right).$$

For more details and other approximations we refer the reader to [11].

The smaller the number of inducing variables, the greater the computational and memory savings. Furthermore, estimation of a large number of inducing

variables may come with a risk of overfitting. However, a small number of inducing variables can result in poor reconstruction. In general the optimal amount of inducing variables is a function of the variation of the training data. If there is small variability in the training data (e.g. few persons with similar styles), a small amount of inducing variables is sufficient for accurate reconstruction. On the other hand, if the test data is very different from the training data, one has to choose a small number of inducing variables to avoid overfitting and generalize well to unseen styles. Unfortunately, there is no optimal way to automatically choose the number of inducing variables, and one has, for example, to use a validation set.

5 Top Down Imposition of Topology

The smooth mapping in the GP-LVM ensures that distant points in data space remain distant in latent space. However, as discussed in [8], the mapping in the opposite direction is not required to be smooth. While the GPDM may mitigate this effect, it often produces models that are neither smooth nor generalize well [26,28].

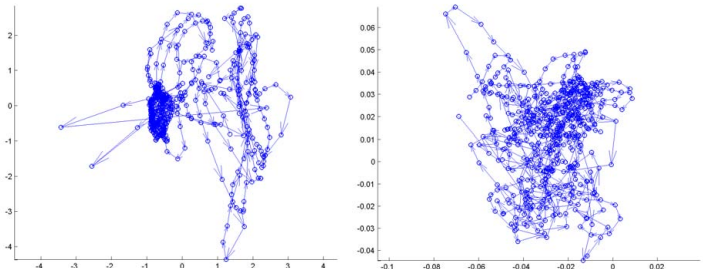
To help ensure smoother, well-behaved models, [8] suggest the use of *back-constraints*, wherein each point in the latent space is a smooth function of its corresponding point in data space, $x_{ij} = g_j(\mathbf{y}_i; \mathbf{a}_j)$, where $\{\mathbf{a}_j\}$ is the set of parameters of the mapping. Optimisation proceeds by substituting each x_{ij} in (2) with $g_j(\mathbf{y}_i; \mathbf{a}_j)$ and maximizing the likelihood with respect to $\{\mathbf{a}_j\}$. This approach is known as the back-constrained GP-LVM.

Nevertheless, when learning human motion data with large stylistic variations or different motions, neither GPDM nor back-constrainted GP-LVM produce smooth models that generalize well. For example, Fig. 1(a) shows a GPDM learned from a training set comprising 9 walks and 10 runs. Compared to the models in Fig. 3 learned from the same training data, the GPDM (Fig. 1(a)) and the back-constrained GPDM (Fig. 1(a)) do not generalize to new runs and walks well, nor do they produce realistic looking simulations.

In this paper we first consider an alternative approach to the hard constraints on the latent space arising from $g_j(\mathbf{y}_i; \mathbf{a}_j)$. We introduce topological constraints through a prior distribution in the latent space, based on a neighborhood structure learn through a generalized local linear embedding (LLE) [18]. We then show how to incorporate domain-specific prior knowledge. This allows us to develop motion models with specific topologies that incorporate different activities within a single latent space and transitions between them.

5.1 Locally Linear GP-LVM

The locally linear embedding (LLE) [18] preserves topological constraints by finding a representation based on reconstruction in a low dimensional space with an optimized set of local weightings. Here we show how the LLE objective can be combined with the GP-LVM, yielding a *locally linear GP-LVM* (LL-GPLVM).



(a) (b)

Fig. 1. When training data contain large stylistic variations and multiple motions, the generic GPDM (a) and the back-constrained GPDM (b) do not produce useful models. The latent models here were learned with the same training data as used to learn those Fig. 3. Simulations of both models here do not look realistic.

The locally linear embedding assumes that each data point and its neighbors lie on, or close to, a locally linear patch on the data manifold. The local geometry of these patches can then be characterized by linear coefficients that reconstruct each data point from its neighbors. This is done in a three step procedure: (1) the K nearest neighbors, $\{\mathbf{y}_j\}_{j\in\eta_i}$, of each point, $\mathbf{y}_i$, are computed using Euclidean distance in the input space, $d_{ij} = ||\mathbf{y}_i - \mathbf{y}_j||^2$; (2) the weights $\mathbf{w} = \{w_{ij}\}$ that best reconstruct each data point from its neighbors are obtained by minimizing $\Phi(\mathbf{w}) = \sum_{i=1}^{N} ||\mathbf{y}_i - \sum_{j\in\eta_i} w_{ij}\mathbf{y}_j||^2$; and (3) the latent positions $\mathbf{x}_i$ best reconstructed by the weights w_{ij} are computed by minimizing $\Phi(\mathbf{X}) = \sum_{i=1}^{N} ||\mathbf{x}_i - \sum_{j\in\eta_i} w_{ij}\mathbf{x}_j||^2$. In the LLE, the weight matrix $\mathbf{w} = [w_{ij}]$, is sparse (only a small number of neighbors is used), and the two minimizations can be computed in close form.

Locally Linear GP-LVM. The LLE energy function is a function of the latent positions and can be interpreted, for a given set of weights $\mathbf{w}$, as a prior over latent configurations that forces each latent point to be locally reconstruct by its neighbors[1]. $p(\mathbf{X}|\mathbf{w}) = \frac{1}{Z}\exp\left\{-\frac{1}{\sigma^2}\Phi(\mathbf{X})\right\}$, where Z is a normalization constant, and σ^2 represents a global scaling of the prior. Following [18], we first compute the neighbors based on Euclidean distance. For each training point $\mathbf{y}_i$, we then compute the weights in closed form by solving, $\forall j \in \eta_i$, the following system,

$$\sum_k C_{kj}^{sim} w_{ij}^{sim} = 1, \quad \text{where} \quad C_{kj}^{sim} = \begin{cases} (\mathbf{y}_i - \mathbf{y}_k)^T(\mathbf{y}_i - \mathbf{y}_j), & \text{if } j,k \in \eta_i \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Once the weights are computed, they are rescaled so that $\sum_j w_{ij} = 1$.

[1] Strictly speaking this is not a proper prior as it is conditioned on the weights, and the weights depend on the training data.

Learning the locally linear GP-LVM is then equivalent to minimizing the negative log posterior of the model, that is up to an additive constant equal to [2]

$$\begin{aligned}\mathcal{L}_{\mathcal{S}} &= \log\, p(\mathbf{Y}|\mathbf{X},\hat{\beta})\, p(\hat{\beta})\, p(\mathbf{X}|\mathbf{w}) \\ &= \frac{D}{2}\ln|\mathbf{K}_Y| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{Y}^T\right) + \sum_i \ln\beta_i + \frac{1}{\sigma^2}\sum_{i=1}^{N}\|\mathbf{x}_i - \sum_{j=1}^{N} w_{ij}\mathbf{x}_j\|^2 \quad (5)\end{aligned}$$

Fig. 2 (a) shows a model compose of 2 walks and 2 runs learned with the Locally linear GPDM. Note how smooth the latent trajectories are.

In what follows we generalize the top-down imposition of topology strategies (i.e. back-constraints and locally linear GP-LVM) to incorporate domain specific prior knowledge.

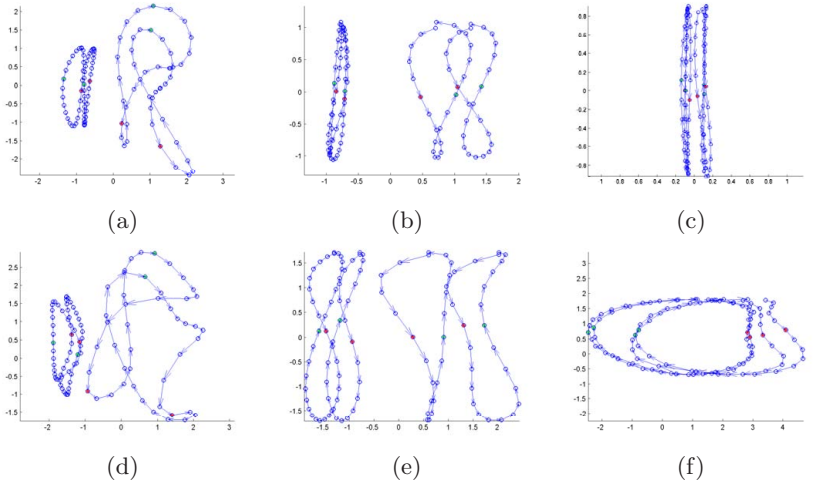


Fig. 2. First two dimensions of models learned using (a) LL-GPDM (b) LL-GPDM with topology (c) LL-GPDM with topology and transitions. (d) Back-constrained GPDM with and RBF mapping. (e) GPDM with topology through backconstraints. (f) GPDM with backconstraints for the topology and transitions. For the models using topology, the cyclic structure is imposed in the last 2 dimensions. The two types of transition points (left and right leg contact points) are shown in red and green, and are used as prior knowledge in (c,f).

6 Reflecting Knowledge in Latent Space Structure

A problem for modelling human motion data is the sparsity of the data relative to the diversity of naturally plausible motions. For example, while we might have a

[2] When learning a locally linear GPDM, the dynamics and the locally linear prior are combined as a product of potentials. The objective function becomes $\mathcal{L}_{\mathcal{S}} + \frac{d}{2}\ln|\mathbf{K}_X| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_X^{-1}\mathbf{X}_{out}\mathbf{X}_{out}^T\right) + \sum_i \ln\alpha_i$, with $\mathcal{L}_{\mathcal{S}}$ defined as in (5).

data set comprising different motions, such as runs, walks *etc.*, the data may not contain transitions between motions. In practice however, we know that these motions will be broadly cyclic and that transitions can only physically occur at specific points in the cycle. How can we encourage our model to respect such topological constraints which arise from prior knowledge?

We consider this problem both in the context of the functionally constrained GP-LVM and the locally linear GP-LVM introduced above. We show how one can adjust the distance metric used in the locally linear embedding to better reflect different types of prior knowledge (such as the location of possible transition points). We also show how one can define similarity measures for use with the functionally constrained GP-LVM. Both these approaches force the latent space to construct a representation that reflects our prior knowledge.

6.1 Prior Knowledge with Back Constraints

As we mentioned in Section 3, back constraints involve a mapping from data space to latent space. If the mapping is smooth the constraints forces points that are close in data space to be close in latent space. The back-constraint functions are used to explicitly constrain the latent positions associated with training points. One possible mapping is a kernel-based regression model, where a regression, on a kernel induced feature space, provides the mapping, $x_{ij} = \sum_{m=1}^{N} a_{jm} k(\mathbf{y}_n, \mathbf{y}_m)$. Many kernels have interpretations as similarity measures [17]. In particular, any similarity measure that leads to a positive semi-definite matrix can be interpreted as a kernel. When learning the back-constrained GP-LVM, one needs to determine the hyperparameters of the kernel matrices (for the back-constraints and the covariance of the GP), as well as the weights of the mapping, $\{a_{jm}\}$. Following [8], we fixed the hyperparameters of the back-constraint's kernel matrix, and optimized over the remaining parameters. We extend the original formulation of back constraints by constructing similarity measures (i.e. kernels) to reflect prior knowledge. We consider two examples: the first involves transitions between activities; with the second we show how topological constraints can be placed on the form of the latent space.

Similarity for Transitions. To capture transitions between two motions, we wish to design a kernel that expresses strong similarity between points in the respective motions where transitions may occur. To express this similarity we first define an index on the frames of the motion sequence, $\{t_i\}_{i=1}^{N}$. We then define subsets of this set, $\{\hat{t}_i\}_{i=1}^{M}$, which represents frames where transitions are possible. For walking and running motions one might use two transition sets, one in which the left foot makes ground contact, and one for the right foot; ground contact can be automatically extracted as it coincides with non linearity in the dynamics [2]. For other motion types a similarity measure can be used to define transitions [7]. We can encourage transition points of different sequences to be proximal with the following kernel matrix for the back-constraint mapping:

$$k^{trans}(t_i, t_j) = \sum_m \sum_l \delta_{ml} k(t_i, \hat{t}_m) k(t_j, \hat{t}_l) \tag{6}$$

where $k(t_i, \hat{t}_l)$ is an RBF centered at $\hat{t}_l$, and $\delta_{ml} = 1$ if $\hat{t}_m$ and $\hat{t}_l$ are in the same set. The 'strength' of the encouragement is controlled by the support width of the RBF kernel.

Combining Similarities. One advantage of our framework is that kernel matrices can be combined in a principled manner to form new kernel matrices. Kernels can be multiplied (on an element by element basis) or added together. Multiplication of kernel-based similarity measures has, loosely speaking, an 'AND gate effect', i.e. both similarity measures must agree that an object is similar for their product to express similarity. Adding them produces more of an 'OR gate effect', i.e. if either representation expresses similarity the resulting measure will also express similarity.

Topologically Constrained Latent Spaces. We now consider kernels that encourage the latent space to have a particular topology. Specifically we are interested in suitable topologies for walking and running data. Because the data are approximately periodic, it seems appropriate to have a non-Cartesian topology. To this end one could extract the phase of the motion[3], ϕ, and use it with a suitable similarity measure to encourage the latent points to response a non-Cartesian topology within a Cartesian space. As an illustrative example we consider a cylindrical topology within a three dimensional latent space by constraining two of the latent dimensions with phase. In particular to represent the cyclic motion we construct a distance function on the unit circle, where a latent point corresponding to phase ϕ is represented with the point $(\cos(\phi), \sin(\phi))$. A periodic mapping can be constructed from a kernel matrix as follows,

$$\begin{aligned} x_{n,1} = g_1^{cos}(\mathbf{y}_n, \phi_n) &= \sum_{m=1}^{N} a_m^{cos} k(\cos(\phi_n), \cos(\phi_m)) + a_0^{cos} \delta_{n,m}, \\ x_{n,2} = g_2^{sin}(\mathbf{y}_n, \phi_n) &= \sum_{m=1}^{N} a_m^{sin} k(\sin(\phi_n), \sin(\phi_m)) + a_0^{sin} \delta_{n,m}, \end{aligned}$$

where k is an RBF kernel function, and $x_{n,i}$ is the i^{th} coordinate of the n^{th} latent point. These two mappings project onto two dimensions of the latent space, forcing them to have a periodic structure (which comes about through the dependence of the kernel with respect to cosine and sine of the phase). Fig. 2 (e) shows a model learned using GPDM with the last two dimensions constrained in this way (the third dimension is out of plane). The first dimension is constrained by an RBF mapping on the input space. Each dimension's kernel matrix can then be augmented by adding the transition similarity given in (6), resulting in the model depicted by Fig. 2 (f).

[3] The phase can be easily extracted from the data by Fourier analysis or by detecting key postures and interpolating the phases between them. Another idea, not further explored here, would be to optimize the GP-LVM with respect to the phase.

6.2 Prior Knowledge Through Local Linearities

We now turn to consider how one might incorporate prior knowledge in the locally linear GP-LVM framework. This is accomplished by replacing the local Euclidean distance measures presented in Section 5.1 with other similarity measures. The standard Euclidean distance measure leads to the covariance matrice given in (4). However, just as we developed similarity matrices above, we can also modify this covariance function to reflect our prior knowledge in the latent space.

Covariance for Transitions. To capture transitions in the latent model we define the elements for the covariance matrix as follows,

$$C_{ij}^{trans} = 1 - \left[\delta_{ij} \exp(-\zeta(t_i - t_j)^2)\right] \tag{7}$$

with ζ a constant, and $\delta_{ij} = 1$ if t_i and t_j are in the same set $\{\hat{t}_k\}_{k=1}^{M}$, and otherwise $\delta_{ij} = 0$. This covariance encourages the points at which transitions are physically possible to be close together in the latent space.

Combining Covariances. As above the covariance matrices can also be combined in a principled way. Covariances can be added or multiplied (on an element by element basis) to loosely speaking produce and 'OR' or 'AND' gate effect.

Covariance for Topologies. To force a cylindrical topology on the latent space, we can introduce covariances based on the phase, specifying different covariances for each latent dimensions. As before we construct a distance function in the unit circle, that takes into account the phase.

$$C_{j,k}^{cos} = \big(\cos(\phi_i) - \cos(\phi_{\eta_j})\big)\ \big(\cos(\phi_i) - \cos(\phi_{\eta_k})\big) \tag{8}$$

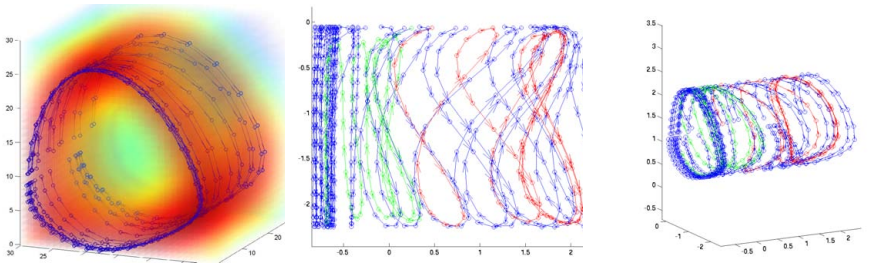
$$C_{j,k}^{sin} = \big(\sin(\phi_i) - \sin(\phi_{\eta_j})\big)\ \big(\sin(\phi_i) - \sin(\phi_{\eta_k})\big)\,, \tag{9}$$

The covariance for the remaining dimension is constructed as usual, based on Euclidean distance in the data space. In Fig. 2 (b) a GPDM is constrained in this way, and in Fig. 2 (c) the covariance is augmented with transitions.

Note that the use of different distance measures for each dimension of the latent space implies that the neighborhood and the weights in the locally linear prior will also be different for each dimension. One way of looking at it is that three different locally linear embeddings are developed for constructing the prior distribution. Each gives a prior distribution for a different dimension of the latent space.

6.3 Model Combination

The two sections above have shown how to incorporate prior knowledge in the GP-LVM by means of 1) local linearities and 2) backconstraints. In general, the latter converges faster but it is more intuitive to constraint the latent space with soft constraints. Both techniques are complementary and can be combined straightforwardly by including priors over some dimensions, and constraining the others through back-constraint mappings. Fig. 3 shows a model learned with the locally linear GPDM to impose smoothness and backconstraints for topology.



(a) (b) (c)

Fig. 3. Hybrid model learned using local linearities for the style and backconstraints for the content. The training data is composed of 9 walks and 10 runs performed by different subjects and speeds. (a) Likelihood for the reconstruction of the latent points (b) First two components and (c) 3D view of the latent trajectories for the training data in blue, and the automatically generated motions of Figs. 4 and 5 in green and red respectively.

6.4 Modeling Multiple Activities and Transitions Between Them

Once we know how to ensure that transition points are close together and that the latent structure has the topology we want, we still need to address two issues. How do we learn models that have very different dynamics? How will the model interpolate between the different dynamics? Our goal in this section is to show how latent models for different motions can be learned independently, but in a shared latent space that facilitates transitions between activities with different dynamics.

Dynamics for Multiple Activities. Let $\mathbf{Y} = [\mathbf{Y}_1^T, ..., \mathbf{Y}_M^T]^T$ denote training data for M different activities. Each $\mathbf{Y}_m$ comprises several different motions. Let $\mathbf{X} = [\mathbf{X}_1^T, ..., \mathbf{X}_M^T]^T$ denote the corresponding latent positions. When dealing with multiple activities, a single dynamical model cannot cope with the complexity of the different dynamics. Instead, we consider a model where the dynamics of each activity are modeled independently, $p(\mathbf{X}) = \prod_{m=1}^{M} p(\mathbf{X}_m)$. This approach has the advantage that a different kernel can be used for each acitivity. Another interpretation is that we have a block diagonal kernel matrix for the Gaussian process that governs the dynamics.

To enable interpolation between motions with different dynamics, we combined these independent dynamical models learned in the form of a mixture model. By interpolating between the components of the mixture distribution we can produce motions that gracefully transition between different styles and motion types (Figs. 4 and 5).

7 Results

We first evaluate the sparse GPLVM in the CMU Mocap dataset comprising 1613 poses from walking and running motions of subject 35. For learning we used a

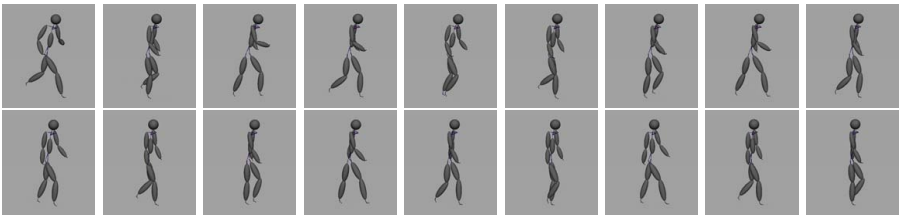


Fig. 4. Transition from running to walking: The system transitions from running to walking in a smooth and realistic way. The transition is encouraged by incorporating prior knowledge in the model. The latent trajectories are shown in green in Fig. 2 (b,c).

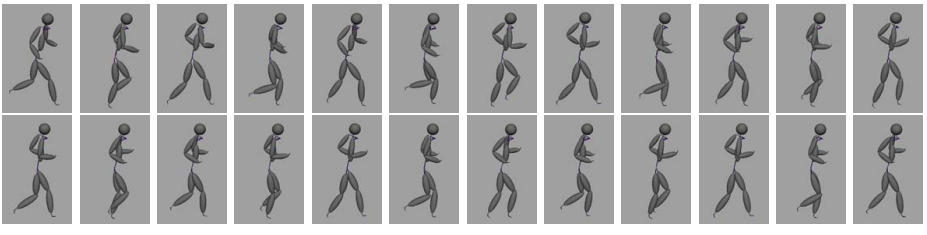


Fig. 5. Different running styles and speeds: The system is able to simulate a motion with considerably changes in speed and style. The latent trajectories are shown in red in Fig. 2 (b,c).

FITC approximation with 100 inducing points. However, rather than allowing these points to be moved freely, they were fixed to latent positions that were uniformily sub-sampled from the data. The models were also backconstrained to encourage smoothness. Following [22], we apply the model to two missing data problems, the first in which the right leg data was removed from a test sequence and the second in which the upper body was removed. A summary of the results is shown in Table 1, showing that that the GPLVM outperforms nearest neighbour, and that a latent space of dimension 3 is sufficient to model the data.

To illustrate how prior knowledge can impact the learned models, we consider a training set comprising 4 gait cycles (2 walks and 2 runs) performed by one

Table 1. Results from the missing data problem. Headings: L and B are the leg and body data sets. The error measure is the mean square error in the angle space. Methods are: NN: nearest neighbour, GPLVM (latent dimension): the GPLVM with different latent dimensions, q. Bold results are the best reported for a given column.

Data	L	B
GPLVM ($q = 3$)	3.40	**2.49**
GPLVM ($q = 4$)	**3.38**	2.72
GPLVM ($q = 5$)	4.25	2.78
NN	4.11	3.20

subject at different speeds. Figure 2 shows the latent spaces learned under different priors. All models are learned using two independent dynamical models, one for walking and one for running. Note how the phases are aligned when promoting a cylindrical topology, and how the LL-GPDM is smooth. Notice the difference between the LL-GPDM (Fig. 2 (c)) and the backconstrained GPDM (Fig. 2 (f)) when transition constraints are included. Both models ensure that the transition points (shown in red and green) are proximal.

Figure 3 (a) shows a hybrid model learned using LL-GPDM for smoothness, and back-constraints for topology. The larger training set comprises approximately one gait cycle from each of 9 walking and 10 running motions performed by different subjects at different speeds (3 km/h for walking, 6–12 km/h for running). Colors in Fig. 3 (a) represent the log variance of the GP as a function of latent position. Only points close to the surface of the cylinder generate poses with high certainty.

We now illustrate the model's ability to simulate different motions and transitions. Given an initial latent position $\mathbf{x}_0$, we generate new motions by sampling the mixture model, and using mean prediction for pose reconstruction. Choosing different initial conditions results in different simulations (see Fig. 3 (b-c), where the training data are shown in blue). For the first simulation (in green), the motion is initialized to a running pose, with a latent position not far from walking poses. The system transitions to walking quickly and quite naturally. The resulting animation is depicted in Fig. 4. For the second example (in red), we initialize the simulation to a latent position far from walking data. The system evolves to different running styles and speeds (Fig. 5). Note how the dynamics, and the strike length, change considerably during simulation.

8 Conclusions

In this paper we have proposed a general framework of probabilistic models that learn smooth latent variable models of different activities within a shared latent space. We have also introduced a principled way to include prior knowledge, that allow us to learn specific topologies and transitions between the different motions. Although we have learned models composed of walking and running, our framework is general, being applicable in any data sets where there is a large degree of prior knowledge for the problem domain, but the data availability is relatively sparse compared to its complexity.

References

1. Arikan, O., Forsyth, D.: Interactive motion generation from examples. In: SIGGRAPH, pp. 483–490 (2002)
2. Bissacco, A.: Modeling and Learning Contact Dynamics in Human Motion. In: CVPR, pp. 421–428 (2005)
3. Brand, M., Hertzmann, A.: Style Machines. In: SIGGRAPH, pp. 183–192 (2000)
4. Elgammal, A., Lee, C.: Separating Style and Content on a Nonlinear Manifold. In: CVPR, vol. 1, pp. 478–485 (2004)

5. Grochow, K., Martin, S., Hertzmann, A., Popovic, Z.: Style-based inverse kinematics. In: SIGGRAPH, pp. 522–531 (2004)
6. Vasilescu, M.A.: Human Motion Signatures: Analysis, Synthesis. In: ICPR, pp. 456–460 (2002)
7. Kovar, L., Gleicher, M., Pighin, F.: Motion Graphs. In: SIGGRAPH, pp. 473–482 (2002)
8. Lawrence, N., Quinonero-Candela, J.: Local distance preservation in the GP-LVM through back constraints. In: ICML, pp. 513–520 (2006)
9. Lawrence, N.D., Seeger, M., Herbrich, R.: Fast sparse Gaussian process methods: The informative vector machine. In: NIPS, pp. 609–616 (2003)
10. Lawrence, N.D.: Gaussian Process Models for Visualisation of High Dimensional Data. In: NIPS (2004)
11. Lawrence, N.: Learning for larger datasets with the Gaussian process latent variable model. In: AISTATS (2007)
12. Li, R., Yang, M.H., Sclaroff, S., Tian, T.: Monocular Tracking of 3D Human Motion with a Coordinated Mixture of Factor Analyzers. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 137–150. Springer, Heidelberg (2006)
13. Li, Y., Wang, T., Shum, H.: Motion Texture: A Two-Level Statistical Model for Character Motion Synthesis. In: SIGGRAPH, pp. 465–472 (2002)
14. Moon, K., Pavlovic, V.: Impact of Dynamics on Subspace Embedding and Tracking of Sequences. In: CVPR, pp. 198–205 (2006)
15. Pavlovic, J.M., Rehg, J., MacCormick, J.: Learning switching linear models of human motion. In: NIPS, pp. 981–987 (2000)
16. Quiñonero-Candela, J., Rasmussen, C.E.: A Unifying View of Sparse Approximate Gaussian Process Regression. In: JMLR, vol. 6, pp. 1939–1959 (2006)
17. Rasmussen, C.E., Williams, C.K.: Gaussian Process for Machine Learning. MIT Press, Cambridge (2006)
18. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
19. Sidenbladh, H., Black, M.J., Sigal, L.: Implicit Probabilistic Models of Human Motion for Synthesis and Tracking. In: Tistarelli, M., Bigun, J., Jain, A.K. (eds.) ECCV 2002. LNCS, pp. 784–800. Springer, Heidelberg (2002)
20. Sminchisescu, C., Jepson, A.: Generative Modeling for Continuous Non-Linearly Embedded Visual Inference. In: ICML, pp. 96–103 (2004)
21. Snelson, E., Ghahramani, Z.: Sparse Gaussian processes using pseudo-inputs. In: NIPS, pp. 1257–1264 (2006)
22. Taylor, G.W., Hinton, G.E., Roweis, S.: Modeling human motion using binary latent variables. In: NIPS, pp. 1345–1352 (2007)
23. Tenenbaum, J., de Silva, V., Langford, J.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)
24. Tian, T., Li, R., Sclaroff, S.: Articulated Pose Estimation in a Learned Smooth Space of Feasible Solutions. In: CVPR Learning Workshop (2005)
25. Urtasun, R.: Motion Models for Robust 3D Human Body Tracking. PhD thesis, EPFL (2006)
26. Urtasun, R., Fleet, D.J., Fua, P.: 3d people tracking with gaussian process dynamical models. In: CVPR, vol. 1, pp. 238–245 (2006)
27. Urtasun, R., Fleet, D.J., Hertzman, A., Fua, P.: Priors for people tracking from small training sets. In: ICCV, pp. 403–410 (2005)
28. Wang, J., Fleet, D.J., Hertzman, A.: Gaussian Process dynamical models. In: NIPS, pp. 1441–1448 (2005)
29. Wang, J., Fleet, D.J., Hertzman, A.: Multifactor Gaussian Process Models for Style-Content Separation. In: ICML, pp. 975–982 (2007)

Silhouette Based Generic Model Adaptation for Marker-Less Motion Capturing*

Martin Sunkel, Bodo Rosenhahn, and Hans-Peter Seidel

Max-Planck-Institut für Informatik,
Stuhlsatzenhausweg 85,
66123 Saarbrücken, Germany

Abstract. This work presents a marker-less motion capture system that incorporates an approach to smoothly adapt a generic model mesh to the individual shape of a tracked person. This is done relying on extracted silhouettes only. Thus, during the capture process the 3D model of a tracked person is learned.

Depending on a sparse number of 2D-3D correspondences, that are computed along normal directions from image sequences of different cameras, a Laplacian mesh editing tool generates the final adapted model. With the increasing number of frames an approach for temporal coherence reduces the effects of insufficient correspondence data to a minimum and guarantees smooth adaptation results. Further, we present experiments on non-optimal data that show the robustness of our algorithm.

1 Introduction

We address the problem of human shape and motion capture (MoCap) from multi-view video sequences. Surveys on these topics can be found in [15,14]. Approaching marker-less methods, researchers working in the area of computer vision typically prefer simplified human body models [4,13,10,11]. There are also methods in the field of Computer Graphics [12,6,24]. However, techniques for image processing or pose estimation are often oversimplified.

Cheung et al. [8] propose a shape-from-silhouettes approach that is applied to track human beings and incorporates surface point clouds with skeleton models. A work of Rosenhahn et al. [18] combines silhouette based pose estimation with more realistic human template models. These are represented by free-form surface patches and gain more accurate tracking results.

In order to allow for individual shapes during MoCap, Mündermann et al. [16] propose a MoCap system that combines the tracking algorithm with a database of articulated 3D models. The models are generated from a template mesh given by a deformable human model that is learned from a database of full body laser scans. During MoCap they select the best fitting 3D model from their database.

Bălan et al. [5] follow an approach of immediately incorporating a low-dimensional deformable human body model (SCAPE) into MoCap. Their idea

* We gratefully acknowledge funding by the Max-Planck Center for Visual Computing and Communication.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 119–135, 2007.

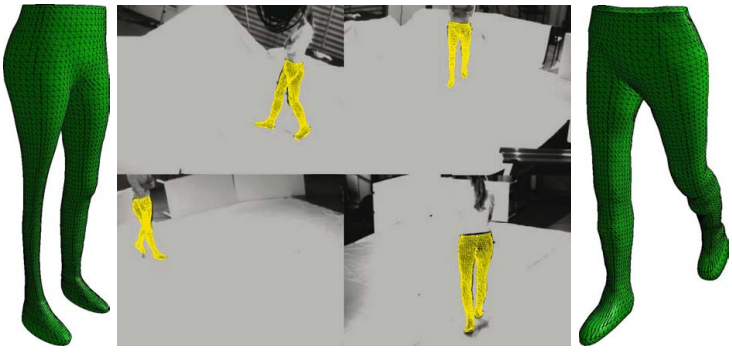


Fig. 1. A multi-view image sequence: *Left:* Generic model, *Middle:* Tracked adapted model within image sequence, *Right:* Adapted model in tracked pose

is to learn pose and detailed shape by a stochastic optimization function that estimates the model parameters directly from 2D image data.

Another approach to jointly capture human motion and shape is presented by De Aguiar et al. [9]. They make use of a high-quality laser scan of the person to track and combine an image-based 3D correspondence estimation algorithm with a fast Laplacian mesh deformation scheme.

The proposals of [9,5,16] have in common, that there is the need for 3D laser scans. The latter ones even require special databases. Thus, the actual MoCap process comes along with extra costs. In contrast to these approaches, we present an algorithm that extends a marker-less MoCap system by an iterative adaptation algorithm. By the use of silhouette information, it smoothly adapts a generic template model to the true shape of the tracked person. Depending on a sparse number of 2D-3D correspondences, that are computed along normal directions from image sequences of 4 cameras, a Laplacian mesh editing tool generates the final adapted model. With the increasing number of frames an approach for temporal coherence reduces the effects of insufficient correspondence information to a minimum. As a result our algorithm increases tracking accuracy.

This work is built upon the basic tracking system and foundations in Section 2. This includes techniques for image segmentation with level sets, pose estimation of kinematic chains and shape registration based on ICP. Focus of this work is embedding the silhouette-based model adaptation algorithm. It is described in Section 3. Section 4 presents experiments as well as their results, and Section 5 concludes this work.

1.1 Contributions

This work contributes a method to compute accurate and smooth 3D models from a generic template. That is done during a silhouette based, marker-less MoCap process. In a temporal coherent approach we apply sophisticated mesh processing techniques, and thus incorporate mesh modelling techniques into a

problem of computer vision. Finally, we perform experiments to test the robustness of our algorithm and present a quantitative error analysis for knee joints.

2 Twists, Kinematic Chains and Pose Estimation

This work is based on a marker-less MoCap system [18,19]. The human being is represented in terms of free-form surface patches. Joint indices are added to each surface node and the joint positions are assumed. This allows us to generate arbitrary body configurations, steered by joint angles. The corresponding counterparts in the images are 2D silhouettes: These are used to reconstruct 3D ray bundles. A spatial distance constraint is minimized to determine the position and orientation of the surface mesh as well as the joint angles. In this section we give a brief summary of the MoCap system.

2.1 Twists

A rigid body motion of a 3D point $\mathbf{x}$ can be expressed in homogeneous coordinates as

$$X' = (\mathbf{x}', 1)^T = \mathbf{M}\mathbf{X} = \mathbf{M}(\mathbf{x}, 1)^T = \begin{pmatrix} \boldsymbol{R}\ \mathbf{t} \\ \mathbf{0}\ 1 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix}. \tag{1}$$

The matrix $\boldsymbol{R}$ is a 3×3 rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ a translation vector. The set of all matrices of type $\boldsymbol{M}$ is called the *Lie Group* $SE(3)$. To every Lie group there exists an associated Lie algebra, whose underlying vector space is the tangent space of the Lie group, evaluated at its origin. The Lie algebra associated with $SE(3)$ is $se(3) := \{(\mathbf{v}, \boldsymbol{\omega}) | \mathbf{v} \in \mathbb{R}^3, \boldsymbol{\omega} \in so(3)\}$, with $so(3) := \{\boldsymbol{A} \in \mathbb{R}^{3\times3} | \boldsymbol{A} = -\boldsymbol{A}^T\}$. Elements of $so(3)$ and $se(3)$ can be written as vectors $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)^T$, $\boldsymbol{\xi} = (\omega_1, \omega_2, \omega_3, v_1, v_2, v_3)^T$ or matrices

$$\hat{\boldsymbol{\omega}} = \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}, \hat{\boldsymbol{\xi}} = \begin{pmatrix} \hat{\boldsymbol{\omega}}\ \boldsymbol{v} \\ \mathbf{0}\ 0 \end{pmatrix}.$$

A twist $\boldsymbol{\xi}$ can be converted into an element of the Lie group $\boldsymbol{M} \in SE(3)$ by computation of its exponential form. That can be done efficiently by using the Rodriguez formula [17].

Note: For varying θ the one-parametric Lie-subgroup $M_\theta = \exp(\theta\hat{\boldsymbol{\xi}})$ yields a screw motion around an axis in space. We will use a degenerate screw (without pitch component) for the model joints.

2.2 Kinematic Chains

A kinematic chain is modeled as the consecutive evaluation of exponential functions of twists $\boldsymbol{\xi}_i$ as done in [4]. A point at an end effector that is additionally transformed by a rigid body motion is given by

$$\boldsymbol{X}'_i = \exp(\theta\ \hat{\boldsymbol{\xi}}) \cdot (\exp(\theta_1\ \hat{\boldsymbol{\xi}}_1) \cdots \exp(\theta_n\ \hat{\boldsymbol{\xi}}_n)) \cdot \boldsymbol{X}_i. \tag{2}$$

In the remainder of this paper we will note a pose configuration by the vector $\boldsymbol{\chi} = (\boldsymbol{\xi}, \theta_1, \ldots, \theta_n) = (\boldsymbol{\xi}, \boldsymbol{\Theta})$ of dimension $(6+n)$ consisting of the 6 degrees of freedom for the rigid body motion $\boldsymbol{\xi}$ and the joint angle vector $\boldsymbol{\Theta}$. In our setup, the vector $\boldsymbol{\chi}$ is unknown and has to be determined from the image data.

2.3 Silhouette Extraction

In order to find the silhouette of an object in the image, a level set function $\Phi \in \Omega \mapsto \mathbb{R}$ is employed. It splits the image domain Ω into two regions Ω_1 and Ω_2 with $\Phi(\boldsymbol{x}) > 0$ if $\boldsymbol{x} \in \Omega_1$ and $\Phi(\boldsymbol{x}) < 0$ if $\boldsymbol{x} \in \Omega_2$. The zero-level line thus marks the boundary between both regions.

For an optimum partitioning, the following energy functional is minimized, which is an extended version of the Chan-Vese model [7]:

$$E(\Phi, p_1, p_2) = -\int_{\Omega} \big(H(\Phi(\boldsymbol{x})) \cdot \log p_1(I(\boldsymbol{x})) + (1 - H(\Phi(\boldsymbol{x}))) \cdot \log p_2(I(\boldsymbol{x})) + \nu \cdot |\nabla H(\Phi(\boldsymbol{x}))| \big)\, d\boldsymbol{x} \qquad (3)$$

with a weighting parameter $\nu > 0$ and $H(s)$ being a regularized version of the Heaviside (step) function, e.g. the error function. The probability densities p_1 and p_2 measure the fit of an intensity value $I(\boldsymbol{x})$ to the corresponding region. We model these densities by local Gaussian distributions. The partitioning and the probability densities p_i are estimated according to the expectation-maximization principle.

2.4 Registration, Pose Estimation

Assuming an extracted image contour and the silhouette of the projected surface mesh, the closest point correspondences between both contours are used to define a set of corresponding 3D lines and 3D points. Then a 3D point-line based pose estimation algorithm for kinematic chains is applied to minimize the spatial distance between both contours: For point based pose estimation each line is modeled as a 3D Plücker line $L_i = (\boldsymbol{n}_i, \boldsymbol{m}_i)$ [1]. For pose estimation the reconstructed Plücker lines are combined with the twist representation for rigid motions: Incidence of the transformed 3D point $\boldsymbol{X}_i$ with the 3D ray $L_i = (\boldsymbol{n}_i, \boldsymbol{m}_i)$ can be expressed as

$$(\exp(\theta\hat{\boldsymbol{\xi}})\boldsymbol{X}_i)_\pi \times \boldsymbol{n}_i - \boldsymbol{m}_i = 0. \qquad (4)$$

Since $\exp(\theta\hat{\boldsymbol{\xi}})\boldsymbol{X}_i$ is a 4D vector, the function π denotes the projection of the homogeneous 4D vector to a 3D vector by neglecting the homogeneous component.

For the case of kinematic chains, we exploit the property, that joints are expressed as special twists with no pitch of the form $\theta_j\hat{\boldsymbol{\xi}}_j$ with known $\hat{\boldsymbol{\xi}}_j$ (the location of the rotation axes is part of the model) and unknown joint angle θ_j. The constraint equation of an ith point on a jth joint has the form

$$(\exp(\theta\hat{\boldsymbol{\xi}})\exp(\theta_1\hat{\boldsymbol{\xi}}_1)\dots\exp(\theta_j\hat{\boldsymbol{\xi}}_j)\boldsymbol{X}_i)\times\boldsymbol{n}_i-\boldsymbol{m}_i=\boldsymbol{0}. \tag{5}$$

To minimize for all correspondences in a least squares sense, we optimize

$$\underset{\boldsymbol{\chi}}{\operatorname{argmin}}\sum_i\left\|\pi\left(\exp(\hat{\boldsymbol{\xi}})\prod_{j\in\mathcal{J}(\boldsymbol{x}_i)}\exp(\theta_j\hat{\boldsymbol{\xi}}_j)\begin{pmatrix}\boldsymbol{x}_i\\1\end{pmatrix}\right)\times\boldsymbol{n}_i-\boldsymbol{m}_i\right\|^2. \tag{6}$$

The function $\mathcal{J}(\boldsymbol{x}_i)$ denotes the set of joints that affect the point $\boldsymbol{x}_i$. Linearization of this equation leads to three linear equations with $6+j$ unknowns, the six pose parameters and j joint angles. Collecting enough correspondences yields an over-determined linear system of equations and allows to solve for these unknowns in the least squares sense. Then the Rodriguez formula is applied to reconstruct the group action and the process is iterated for the transformed points until convergence.

2.5 The Tracking System

Since segmentation and pose estimation can both benefit from each other, it is convenient to couple both problems in a joint optimization problem. To this end, the energy functional for image segmentation in (3) is extended by an additional term that integrates the surface model. Thus, by means of the contour Φ, the tracking system in [19] can be described by the following energy functional, which is sought to be minimized:

$$\begin{aligned}E(\Phi,p_1,p_2,\boldsymbol{\chi}) = &-\int_\Omega\left(H(\Phi)\log p_1+(1-H(\Phi))\log p_2+\nu|\nabla H(\Phi)|\right)d\boldsymbol{x}\\ &+\underbrace{\lambda\cdot\int_\Omega(\Phi-\Phi_0(\boldsymbol{\chi}))^2d\boldsymbol{x}}_{\text{shape error}}.\end{aligned} \tag{7}$$

The quadratic error measure in the shape term has been proposed in the context of 2D shape priors, e.g. in [20]. The prior $\Phi_0\in\Omega\to\mathbb{R}$ is assumed to be represented by the signed distance function. This means in our case, $\Phi_0(\boldsymbol{x})$ yields the distance of x to the silhouette of the projected object surface. The influence of the shape prior on the segmentation is steered by the parameter λ (we chose 0.05). Due to the nonlinearity of the optimization problem, we propose an iterative minimization scheme: first the pose parameters $\boldsymbol{\chi}$ are kept constant while the functional is minimized with respect to the partitioning. Then the contour is kept constant while the pose parameters are estimated to fit the surface mesh to the silhouettes (Section 2.4). A comparable approach for combined segmentation and pose estimation using graph cuts has been presented in [3].

Given the contour Φ, the pose estimation method from Section 2.4 minimizes the shape term in (7). Minimizing (7) with respect to the contour Φ leads to the gradient descent equation

$$\partial_t\Phi=H'(\Phi)\left(\log\frac{p_1}{p_2}+\nu\,\operatorname{div}\left(\frac{\nabla\Phi}{|\nabla\Phi|}\right)\right)+2\lambda\,(\Phi_0(\theta\boldsymbol{\xi})-\Phi). \tag{8}$$

The total energy is minimized by iterating both minimization procedures. Both iteration steps minimize the distance between Φ and Φ_0. While the pose estimation method draws Φ_0 towards Φ, thereby respecting the constraint of a rigid motion, in return (8) draws the curve Φ towards Φ_0, thereby respecting the data in the image.

3 Generic Model Adaptation

Our adaptation approach extends the tracking loop as sketched in the overview of Figure 2. Given a generic 3D model, the motion tracking algorithm estimates 3D pose and 2D image contours in an iterative process. Each iteration refines the quality for pose- and silhouette estimation.

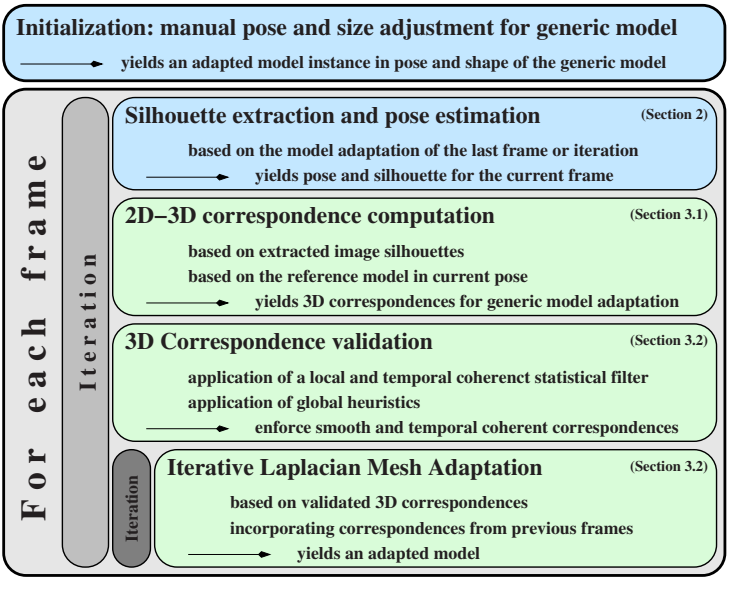


Fig. 2. Model adaptation algorithm incorporated into the basic MoCap system. Green marks the contribution of this paper.

For each new pose estimation the adapted model of the previous iteration is used. Opposite to that, in each frame and iteration the starting point for our adaptation algorithm is the generic 3D mesh model in pose of the current frame. In the further context of this work it is referred to as *reference model* or *reference mesh*. On basis of that reference mesh 3D correspondences are computed for a sparse set of vertices. The correspondences are generated from silhouettes in the image and the reference model along the vertex normals projected to the image planes of the cameras (Figure 3). The reason for projecting to image planes of the cameras is, that, if relying on image contours as seen by a camera, in general it is not possible to state changes parallel to the optical axis (viewing direction).

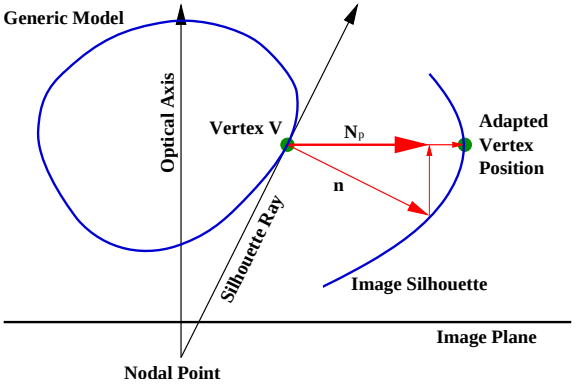


Fig. 3. Sketch for our ICP-algorithm. The algorithm generates a correspondence along normal direction n. Since, in general silhouettes contain no information about depth, the search is restricted to N_p, the normal direction projected into the camera plane.

In order to gain smooth and robust adaptation results, the correspondences have to be validated beforehand. That is done by filter heuristics. One filter operates locally and exploits temporal coherence, incorporating all correspondences of previous frames and iterations up to a maximum number (Section 3.2). Relying on statistical means all correspondences, that appear to be noise or outliers, are removed. Furthermore, there are two global heuristics. One removes all correspondences with lengths (i.e. distance between vertex and its displacement position) greater than a given threshold. The other heuristic ensures that only those vertices are involved into mesh adaptation, whose vertex normal is almost perpendicular to the viewing direction. That filter is an important instrument, in order to avoid ill placed correspondences.

Afterwards the correspondences are applied as constraints in a Laplacian mesh adaptation [21,22] process. The final Laplacian adaptation is designed as an extension of the algorithm as proposed by Stoll et al. [23]. The main extension is to combine all hitherto available correspondences (up to a given number) into one final adaptation routine. Appending the correspondences as constraints to the Laplacian matrix, the solution of the iterative deformation algorithm yields the final adapted mesh model. In all our experiments a number of 15 iterations was sufficient.

Note: The reference mesh is updated in terms of pose only, but not in terms of shape. If artifacts should occur in the final adaptation result (which may happen in the first few frames), they cannot propagate into the reference mesh. That would lead to non-smooth results or to even more artifacts.

3.1 2D-3D Correspondences

The computation of 2D-3D correspondences starts from the generic reference model in pose of the current frame. In order to identify silhouette vertices, the

Fig. 4. Intermediate tracking results for one camera view. *From Left to Right:* Input picture, Reference model in current pose, Extracted Silhouette, 2D Correspondences (blue: ICP starting points, green: search path, white: 2D correspondence point), Final model adaptation.

mesh is projected into the camera planes. The silhouette vertices and their projections are stored. Then, for each silhouette vertex an ICP algorithm [26] computes the 2D correspondence to the segmented image contours. The direction of the ICP search is restricted to the projection of the vertex normal to the camera plane. Given the optical axis $\mathbf{d}$ and normal direction $\mathbf{N}$ (Figure 3) the projected normal $\mathbf{N_p}$ is given by

$$\mathbf{N_p} = \mathbf{N} - \langle \mathbf{N}, \mathbf{d} \rangle \cdot \mathbf{N} \tag{9}$$

Thus, the 3D correspondence for a point $\mathbf{p}$ is found along the Plücker line

$$l : (\mathbf{N_p}, \mathbf{M}), \qquad \mathbf{M} = \mathbf{p} \times \mathbf{N_p} \tag{10}$$

If a 2D correspondence point (x, y) is found, its projection ray is reconstructed. Given the transformation matrix $[\mathbf{R}, \mathbf{t}] \in \mathbb{R}^{3\times 4}$ for a camera, the projection ray is represented by the Plücker line $r : (\mathbf{d}, \mathbf{m})$ with

$$\mathbf{d} = \frac{\mathbf{b} - \mathbf{a}}{\|\mathbf{b} - \mathbf{a}\|}, \qquad \mathbf{m} = \mathbf{a} \times \mathbf{d} \tag{11}$$

$$\mathbf{a} = -\mathbf{R^{-1}} \cdot \mathbf{t}, \qquad \mathbf{b} = \mathbf{R^{-1}} \cdot \left(\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} - \mathbf{t} \right) \tag{12}$$

Then, intersecting l with r, the final 3D correspondence is computed, i.e. that point on l that is closest to r. The intersection $\mathbf{s}$ is calculated by [1]

$$\mathbf{s} = \frac{\langle \mathbf{N_p},\ \mathbf{d} \times \mathbf{m} \rangle - \langle \mathbf{N_p},\ \mathbf{d} \rangle \cdot \langle \mathbf{N_p},\ \mathbf{d} \times \mathbf{M} \rangle}{(\mathbf{N_p} \times \mathbf{d})^2} \cdot \mathbf{N_p} + (\mathbf{N_p} \times \mathbf{M}) \tag{13}$$

Since correspondences of the ongoing frame are reused in those to come, they are stored. Thus, they have to be transformed into a representation that is invariant to rigid body motion. The global coordinates are transformed to local coordinate systems on vertex basis. Given the vertex normal $\mathbf{n}$ and the edge to

a definite neighbour vertex, we compute coordinate axes **u** and **v**, such that **u**, **v** and **n** form a local coordinate system (Figure 5).

$$\mathbf{u} = \mathbf{n} \times \mathbf{e}, \qquad \mathbf{v} = \mathbf{n} \times \mathbf{u} \tag{14}$$

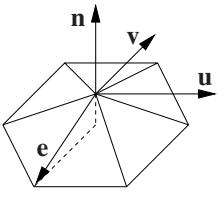


Fig. 5. Coordinate system on a vertex basis: vertex normal **n** and edge **e** to a distinct neighbour vertex are given by the mesh. **u** and **v** are computed such that **u**, **v**, **n** are pairwise perpendicular (see Equation (14)).

Given a vertex point **p**, the coordinate axes **u**, **v**, **n** and a correspondence point **c**, its new representation is computed by the projections of $\mathbf{d} := \overrightarrow{\mathbf{pc}}$ onto **u**, **v**, **n**.

$$\tilde{u} = \langle \mathbf{u}, \mathbf{d} \rangle, \quad \tilde{v} = \langle \mathbf{v}, \mathbf{d} \rangle, \quad \tilde{n} = \langle \mathbf{n}, \mathbf{d} \rangle \tag{15}$$

Vice versa **c** is restored by

$$\mathbf{c} = \mathbf{p} + \tilde{u} \cdot \mathbf{u} + \tilde{v} \cdot \mathbf{v} + \tilde{n} \cdot \mathbf{n} \tag{16}$$

3.2 Correspondence Analysis

All 3D correspondences are analyzed and filtered by three heuristics:

(a) Temporal coherent filter over the local variance of correspondences: For all 3D correspondences **c** with distance d to mesh vertex **p** ($d = \|\overrightarrow{\mathbf{pc}}\|$) the directional distance value $\tilde{d} = (\tilde{n}/|\tilde{n}|) \cdot d$ ($\tilde{n}$ from Equation (15)) is computed. Then, within a predefined radius r (we found 3 cm most suitable) around **p**, the variance σ^2 for all correspondences c_i over the values $\widetilde{d_i}$ is calculated.

$$\sigma^2 = \frac{1}{n-1} \cdot \sum_{i=1}^{n} (\widetilde{d_i} - \overline{d})^2, \qquad \overline{d} = \frac{1}{n} \sum_{i=1}^{n} \widetilde{d_i} \tag{17}$$

We also compute

$$\widetilde{\sigma^2} = (\widetilde{d_i} - \overline{d})^2 \tag{18}$$

and all correspondences with

$$\widetilde{\sigma^2} > k_1, \quad k_1 \in \mathbb{R} \geq 0 \tag{19}$$

or

$$\widetilde{\sigma^2} > k_2 \cdot \sigma^2, \quad k_2 \in \mathbb{R} \geq 0 \tag{20}$$

are filtered out. k_1 and k_2 are threshold parameters and can be interpreted as follows: If the distance between $\mathbf{c}$ and $\mathbf{p}$ differs from the surrounding mean of distances more than $\sqrt{k_1}$, or if the ratio of the squared distance $\widetilde{\sigma^2}$ to its approximate *mean* σ^2 exceeds k_2, then $\mathbf{c}$ is not smooth or reliable enough.

In order to ensure temporal coherent filter results, the list of correspondences used for the filter includes the correspondences of previous frames and iterations up to a maximum number. In our experiments we used a maximum count of 400 sets of correspondence. Older correspondences are deleted.

(b) Distance heuristic: The distance heuristic is a simple global heuristic, removing all correspondences $\mathbf{c}$ with a distance greater than a given value. This value highly depends on the conformity level of the generic model with the tracked person. In our experiments a maximum distance of 6 cm was sufficient for leg tracking, even for poorly designed generic leg models.

(c) Directional heuristic: Since the correspondences are computed along the direction $\mathbf{N_p}$ (Figure 3), all vertices with normals parallel to the optical axis would yield no contribution. The more parallel normal and camera direction are, the more unreliable a correspondence is. The angle between vertex normal and optical axis gives a criterion to deal with this issue. If the angle is below a given threshold, the attached correspondence is filtered out. For our needs, angle thresholds of about 75 degrees yield admissible results.

3.3 Model Adaptation

Foundation for the final adaptation is a linear variational surface deformation method [2]. We choose a Laplacian Mesh Processing [21,22] implementation that is based on cotangent weights. Given a reference mesh $M = (P, E)$ (P a set of vertices ($\mathbf{p_i}$) and E a set of edges ($\mathbf{p_i}, \mathbf{p_j}$)), the Laplacian scheme encodes the knowledge about structural details of the model M in terms of differential coordinates that are stored in a vector $\mathbf{d_p}$. They are computed by solving $\mathbf{d} = \mathbf{L} \cdot \mathbf{p}$ (component-wise for x, y and z). $\mathbf{p}$ is a column vector consisting of either x, y or z coordinates of all points $\mathbf{p_i}$ and $\mathbf{L}$ denotes the Laplacian matrix that is constructed from the model M. The reconstruction $\overline{\mathbf{p}}$ (again a column vector) is subject to a number of constraints (3D point correspondences) $\mathbf{c_j}$ which leads to minimizing a linear least-squares problem of the form

$$\underset{\overline{\mathbf{p}}}{\operatorname{argmin}}\{\|\mathbf{L}\,\overline{\mathbf{p}} - \mathbf{d_p}\|^2 + \|\mathbf{C}\,\overline{\mathbf{p}} - \mathbf{q}\|^2\} \tag{21}$$

which can be transformed into a system of linear equations

$$(\mathbf{L}^T\mathbf{L} + \mathbf{C}^T\mathbf{C}) \cdot \overline{\mathbf{p}} = \mathbf{L}^T\mathbf{d_p} + \mathbf{C}^T\mathbf{q} \tag{22}$$

Here $\mathbf{C}$ is a diagonal matrix with non-negative weights $C_{j,j} = w_j$ for all correspondence constraints.

Since for each frame the tracking- and adaptation solution is computed as refinement result of multiple iterations, the weights are tuned accordingly. Starting with 0.5, they linearly increase up to 1 in the last iteration. At this we obtained good results with 4 iterations for the first frame and 2 for the rest of the sequence.

In order to produce temporal coherent model adaptations, the constraints of previous iterations (as given by the stored 3D point correspondences) are added to the ongoing Laplacian deformation. However, in order to provide the possibility to change the adapted shape over time, the constraint weights are adjusted according to their age, such that they diminish. That is done by multiplying the weights by $(1 + k)^{-t}$, where t is the age and k a small positive value. Storing constraints for a maximum of 400 iterations, we used $k = 1/400$.

A consequence of that approach is, that there may be more than one correspondence that is attached to a vertex. But there are also vertices, without any correspondence attached. Following the idea of Stoll et al. [23] that issue is addressed by exploiting the Laplacian framework for a least-squares approximation for harmonic interpolation [25] over correspondence weights w and vertex displacements $\tilde{u}$, $\tilde{v}$ and $\tilde{n}$. It is computed by solving the Laplace systems $\mathbf{Lw} = \mathbf{0}$, $\mathbf{L\tilde{u}} = \mathbf{0}$, $\mathbf{L\tilde{v}} = \mathbf{0}$ and $\mathbf{L\tilde{n}} = \mathbf{0}$. Finally, the interpolation result contains as many correspondences as there are vertices in the reference mesh.

Note: With the exception of vertices without correspondence, the interpolation over all correspondences of past frames is also used for filter (a) in Section 3.2. Independent of the number of previously stored correspondences that smoothes the set of correspondences and limits computational filter costs to a distinct maximum.

In the end, an iterative mesh deformation algorithm yields the adaptation for the ongoing tracking iteration. In iteration t this algorithm first performs a Laplacian deformation according to tentative constraints. Then, it measures the distance l between correspondence point and mesh vertex of the tentative adaptation. If exceeding a given maximum distance l_{max}, the correspondence is excluded from iteration $t + 1$. For all remaining correspondences the weights are adjusted by multiplication with $(1 - l/l_{max})^2$. In experiments we yield good results performing 15 iterations and using values of 5 to 6 cm for l_{max} .

4 Experiments

We tested our algorithm in several experiments. Playing with the parameter values we found, that in terms of smoothness and robustness the parameters r, k_1 and k_2 in Section 3.2 influence the adaptation result most. Suitable values are $r = 30\ mm$, $k_1 = 100\ mm^2$ and $k_2 = 1$.

We restrict our experiments to tracking the legs of a person in different motion sequences. The actor wears a tight suit. At distinctive positions there are additional markers attached, sticking out of the legs (see Figure 4). The markers are used in order to compute ground-truth for our motion sequences with a

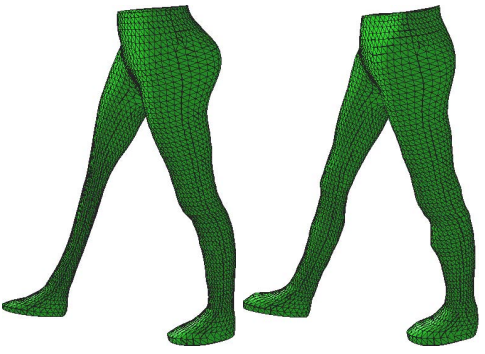


Fig. 6. Leg model in pose of the first frame of a walking sequence. *Left:* Degenerated reference model. The right leg is thinned out, and the left half of the backside is blown up. *Right:* Model adaptation after the first iteration.

marker-based MoCap system. Thus, secondary objective in our experiments is trying to adapt a shape model that also includes these markers.

We test our algorithm in walking and jumping scenes with a degenerated generic leg model. The right leg is thinned out, and the left half of the backside is visibly blown up. Figure 6 shows that model as well as its adaptation after the first iteration of the first frame of a walking sequence.

In order to reliably extract silhouettes, the shape prior λ of equation (7) is adjusted to low values. However, similar to the adaptation result in Figure 4, in both scenarios most markers are too tiny to be reflected within the silhouette. Often, the model adapts the markers at the shinbones only - and in form of smooth bumps.

Figure 7 visualizes typical 3D correspondences that are computed. The left image shows the correspondences after applying filters (b) and (c) from Section 3.2. The center image reflects the effects of additionally applying filter (a). The field of correspondences is smoothed. Some visible outliers at the left foot are removed. The right image shows the set of correspondences after harmonic interpolation. They are used for the final adaptation. The thin leg grows to its true size, and the thick backside is thinned out. The marker on the left shinbone results in a smooth bump. Note: Since heuristic (a) implements a filter for temporal coherence, in the first frames it has fewer effects.

Analyzing the effects of adaptation on the generic model, Figure 8 presents model overlays for frames 0 and 10 of the jumping scene. The template model is rendered in red, its adaptation in green. Both are overlaid by addition and 50% alpha channel. Pure green regions show where the template has grown, red regions reflect shrinking.

Figure 9 shows a quantitative error analysis of the walking sequence (frames 0 to 140) for the knee angles. The black lines give the comparison with ground truth as obtained by a marker-based MoCap system. Red represents the angles

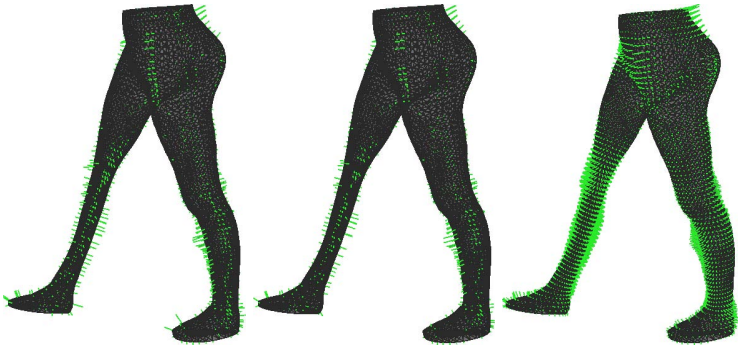


Fig. 7. 3D correspondences, first frame, first iteration. *Left:* Correspondences after application of filters (b) and (c) from Section 3.2. *Center:* Additional application of filter (a). *Right:* Correspondences after application of filter (a) and harmonic interpolation.

Fig. 8. Adaptation analysis: Reference model and its adaptation are overlaid by addition and 50% alpha channel. Red: Reference model, Green: Adapted model. Pure green indicates where the reference model has increased in size, pure red indicates a shrinking.

for the marker-less tracking system without model adaptation. Blue shows the results of our approach. The marker-less MoCap system with model adaptation yields results that are closer to ground truth. The relative error for the red curve is 4.2°. The blue curve deviates by 3.4°.

In more challenging experiments we adjusted the setup by adding noise to the image sequences. Visualizations for tests with uncorrelated noise show adaptation results independent of the noise level that are like those of previous experiments. We also tested with box noise, that is adding boxes of different colors and sizes to the uncorrelated noise. Figure 10 shows silhouette extractions and the adaptation result for the first frame of the walking sequence. We added 25% uncorrelated noise as well as 50 boxes with sizes between 5×5 and 15×15 pixels. At positions of the boxes the silhouettes are seriously distorted. Though dented, we obtain a relatively smooth model adaptation.

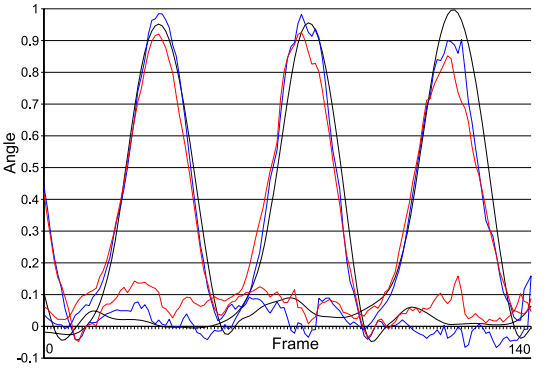


Fig. 9. Quantitative error analysis of the knee angles for a walking sequence (frames 0 to 140). *Black:* Ground truth as obtained by a marker based tracking system. *Red:* Marker-less tracking without model adaptation. *Blue:* Marker-less tracking with model adaptation. *Relative errors:* Blue 3.4°, Red 4.2°.

Concentrating on the originally disfigured right leg, Figure 11 shows its adaptations for the frames 0, 5, 10, 15 and 20. The bumps and dents smooth out and finally vanish.

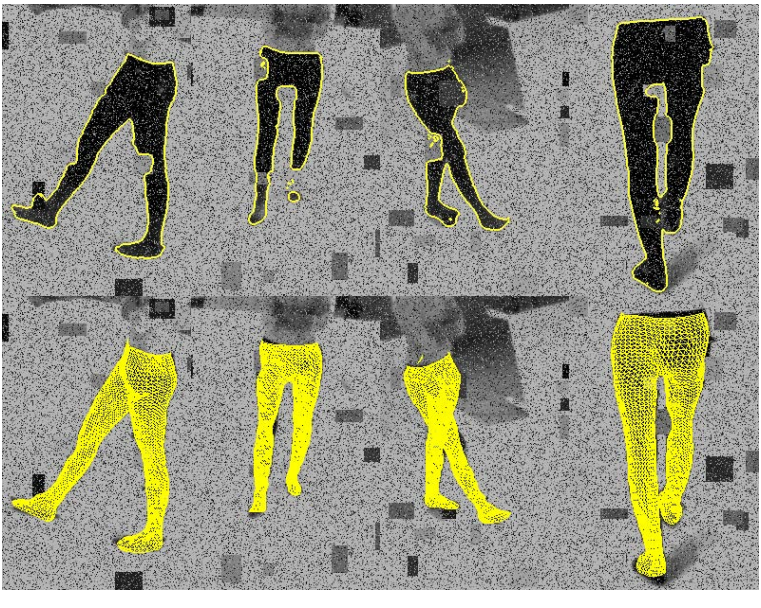


Fig. 10. Experiment with uncorrelated noise (25%) and box noise (50 boxes with sizes between 5×5 and 15×15 pixels). Both rows show a cropped image for each camera view. *Top row:* Silhouette extraction. *Bottom row:* Model adaptation.

Fig. 11. Adaptation result of the right leg for the box noise scenario. Results for the frames 0, 5, 10, 15 and 20.

5 Summary

Our approach extends the motion capture process incorporating sophisticated methods for correspondence analysis as well as for mesh processing. It provides a robust method to smoothly adapt any given generic model to the observed shape. That is done using silhouette information only. Since the main goal is to provide smooth and robust adaptation solutions, the algorithm concentrates on low frequency details. High frequency shape details, such as markers on legs, are most likely to be adapted if they are visible in the first frames. Otherwise, they violate our concept for temporal coherence.

We tested the performance of our algorithm on worst case scenarios, limited to tracking legs of a person in different sequences. We applied our algorithm to a clearly sub-optimal generic model and performed a quantitative error analysis for the accuracy of tracking the knee joints. Compared to the non-adaptive approach, our method provides an improvement. We also performed tests with input images that are distorted with intense noise. Our approach yields smooth results.

A weak point, that sometimes occurs in our algorithm, is adapting regions like the feet. The reason for that is, that on one hand shadows on the floor cause faulty silhouette extractions. On the other hand, all cameras recorded the feet mostly from above, such that it is hard to obtain information about the instep of the feet (see Figures 7 and 8). However, our validation heuristics combined with the smoothing effect of the temporal coherent adaptation approach reduce that issue to a minimum.

References

1. Blaschke, W.: Kinematik und Quaternionen, Mathematische Monographien. 4. Deutscher Verlag der Wissenschaften (1960)
2. Botsch, M., Sorkine, O.: On linear variational surface deformation methods. IEEE Transactions on Visualization and Computer Graphics (TVCG) (2007)
3. Bray, M., Kohli, P., Torr, P.: Posecut: Simultaneous segmentation and 3d pose estimation of humans using dynamic graph-cuts. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 642–655. Springer, Heidelberg (2006)

4. Bregler, C., Malik, J., Pullen, K.: Twist based acquisition and tracking of animal and human kinetics. International Journal of Computer Vision 56(3), 179–194 (2004)
5. Bălan, A., Sigal, L., Black, M., Davis, J., Haussecker, H.: Detailed human shape and pose from images. In: CVPR. Proc. Computer Vision and Pattern Recognition (June 2007)
6. Carranza, J., Theobalt, C., Magnor, M.A., Seidel, H.-P.: Free-viewpoint video of human actors. In: Proc. SIGGRAPH 2003, pp. 569–577 (2003)
7. Chan, T., Vese, L.: Active contours without edges. IEEE Transactions on Image Processing 10(2), 266–277 (2001)
8. Cheung, K.M., Baker, S., Kanade, T.: Shape-from-silhouette across time: Part ii: Applications to human modeling and markerless motion tracking. International Journal of Computer Vision 63(3), 225–245 (2005)
9. de Aguiar, E., Theobalt, C., Stoll, C., Seidel, H.-P.: Marker-less deformable mesh tracking for human shape and motion capture. In: CVPR. IEEE International Conference on Computer Vision and Pattern Recognition, Minneapolis, USA (2007)
10. Fua, P., Plänkers, R., Thalmann, D.: Tracking and modeling people in video sequences. Computer Vision and Image Understanding 81(3), 285–302 (2001)
11. Herda, L., Urtasun, R., Fua, P.: Implicit surface joint limits to constrain video-based motion capture. In: Pajdla, T., Matas, J. (eds.) ECCV 2004. LNCS, vol. 3022, pp. 405–418. Springer, Heidelberg (2004)
12. Magnenat-Thalmann, N., Seo, H., Cordier, F.: Automatic modeling of virtual humans and body clothing. Computer Science and Technology 19(5), 575–584 (2004)
13. Mikic, I., Trivedi, M., Hunter, E., Cosman, P.: Human body model acquisition and tracking using voxel data. International Journal of Computer Vision 53(3), 199–223 (2003)
14. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Computer Vision and Image Understanding 104(2), 90–126 (2006)
15. Moeslund, T.B., Granum, E.: A survey of computer vision based human motion capture. Computer Vision and Image Understanding 81(3), 231–268 (2001)
16. Mündermann, L., Corazza, S., Andriacchi, T.: Accurately measuring human movement using articulated icp with soft-joint constraints and a repository of articulated models. In: CVPR. Proc. Computer Vision and Pattern Recognition (June 2007)
17. Murray, R.M., Li, Z., Sastry, S.S.: Mathematical Introduction to Robotic Manipulation. CRC Press, Baton Rouge (1994)
18. Rosenhahn, B., Brox, T., Kersting, U., Smith, A., Gurney, J., Klette, R.: A system for marker-less motion capture. Künstliche Intelligenz (1), 45–51 (2006)
19. Rosenhahn, B., Brox, T., Weickert, J.: Three-dimensional shape knowledge for joint image segmentation and pose tracking. International Journal of Computer Vision 73(3), 243–262 (2006)
20. Rousson, M., Paragios, N.: Shape priors for level set representations. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2351, pp. 78–92. Springer, Heidelberg (2002)
21. Sorkine, O.: Laplacian mesh processing. In: Eurographics. Proc. of Eurographics - STAR Volume, pp. 53–70 (2005)
22. Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., Seidel, H.-P.: Laplacian surface editing. In: SGP 2004. Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing, pp. 175–184. ACM Press, New York (2004)

23. Stoll, C., Karni, Z., Rössl, C., Yamauchi, H., Seidel, H.-P.: Template deformation for point cloud fitting. In: Botsch, M., Chen, B. (eds.) Eurographics. Symposium on Point-Based Graphics, Boston, USA, pp. 27–35 (2006)
24. You, L., Zhang, J.J.: Fast generation of 3d deformable moving surfaces. IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics 33(4), 615–616 (2003)
25. Zayer, R., Rössl, C., Karni, Z., Seidel, H.-P.: Harmonic guidance for surface deformation. In: Alexa, M., Marks, J. (eds.) Eurographics. Proc. of 26th Annual Eurographics Conference, Computer Graphics Forum, Dublin, Ireland, vol. 24, pp. 601–609. Blackwell (2005)
26. Zhang, Z.: Iterative points matching for registration of free form curves and surfaces. International Journal of Computer Vision 13(2), 119–152 (1994)

3D Hand Tracking in a Stochastic Approximation Setting

Desmond Chik[1,2], Jochen Trumpf[1,2], and Nicol N. Schraudolph[1,2]

[1] Research School of Information Sciences and Engineering,
Australian National University, Canberra ACT 0200, Australia
[2] Statistical Machine Learning, NICTA, Locked Bag 8001,
Canberra ACT 2601, Australia
desmond.chik@rsise.anu.edu.au, jochen.trumpf@anu.edu.au,
nic.schraudolph@nicta.com.au

Abstract. This paper introduces a hand tracking system with a theoretical proof of convergence. The tracking system follows a model-based approach and uses image-based cues, namely silhouettes and colour constancy. We show that, with the exception of a small set of parameter configurations, the cost function of our tracker has a well-behaved unique minimum. The convergence proof for the tracker relies on the convergence theory in stochastic approximation. We demonstrate that our tracker meets the sufficient conditions for stochastic approximation to hold locally. Experimental results on synthetic images generated from real hand motions show the feasibility of this approach.

1 Introduction

Pose estimation for articulated structures such as the human body and hand is a growing field that has real-world applications. Conceivable uses for such technology range from surveillance, over HCI, to motion capture. There are many image-based 3D tracking approaches to date, often tailored for specific applications, see the surveys [1,2].

There have been notable works on tracking articulated bodies using monocular images, including [3,4]. However, depth ambiguities from single images do mean that pose recovery is limited.

Multiple camera views are needed for applications that require a more precise estimation of the body pose. A stereo pair of cameras is enough for depth recovery, but having more cameras reduces ambiguities arising from self-occlusion. Some approaches in multi-view tracking explicitly extract 3D information. For example [5] uses volume reconstruction for a voxel-based fitting process. Silhouette fitting is a popular technique employed by many, *e.g.* [6]. Additional cues are often used to complement silhouette fitting to make the tracking more robust. For example in [7], the reconstruction of a 3D motion field is used to help with the tracking. In [8], motion and spatial cues from images are used to recover displacement in parameter space. In [9], edges, optical flow and shading information from the hand model are used for tracking.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 136–151, 2007.

The evaluation of tracking performance is typically based on ground truth sequences. These might be pre-rendered sequences or data retrieved from an alternative source, such as a commercial motion capture system. Whether a tracker can inherently converge to the ideal pose has largely been empirically verified, *e.g.* [10]. A tracker is said to converge to the optimal parameters if the predicted parameters are close enough to the real ground truth values for a particular set of test sequences. Works on the theoretical convergence of a tracker have been lacking in this area. This is understandable as most tracking systems are complex enough to make this task difficult.

The contribution of this paper is to provide a theoretical framework for proving tracker convergence. We present a 3D hand tracking system that has a theoretical proof for convergence. The tracking system is built to be parallelizable and employs stochastic approximation techniques. We will show that the tracking system locally meets the conditions required for stochastic approximation to work, and by that virtue, the results on stochastic convergence from stochastic approximation theory follow.

The paper is organised as follows; Section 2 describes the tracker. Section 3 shows the existence of a unique global minimum for most cases. Section 4 examines the relevance of stochastic approximation for our tracker and introduces the sufficient conditions for stochastic approximation. Proof that these conditions are met locally is also examined. Tracking results are presented in section 5.

2 Tracking System

A stereo pair of images of the hand is acquired by a pair of calibrated cameras. Using these images, our model-based tracking system estimates the 3D pose of the hand in a stochastic approximation framework. Points are sampled from the surface of a fully articulated hand model and projected onto model image planes. By looking at corresponding pixel coordinates in the real images, a cost function based on the hand silhouette and the colour constancy assumption is evaluated. Errors from the cost evaluation are backpropagated as gradients to the parameter space of the hand model. The gradients are then used to minimise the cost function. Figure 1 shows the tracking process.

This paper concentrates on showing that this tracker setup is compatible with the stochastic approximation approach.

2.1 Hand Model

The tracker uses a fully articulated hand model (see figure 2) having 16 joints, totalling 26 degrees of freedom (DOF). There are 6 DOFs at the palm joint, defining the global rotation and translation of the hand. Each digit has 4 DOFs to encapsulate its articulated movement. Rotations at the joints are parameterised with Euler angles. The skin is modelled by a dense mesh (*e.g.* acquired from the 3D scanning of a real hand) and is bound to the underlying skeleton via linear skin blending. Linear skin blending allows sample points taken near the joint regions to deform in a more realistic manner when the joint is bent [11].

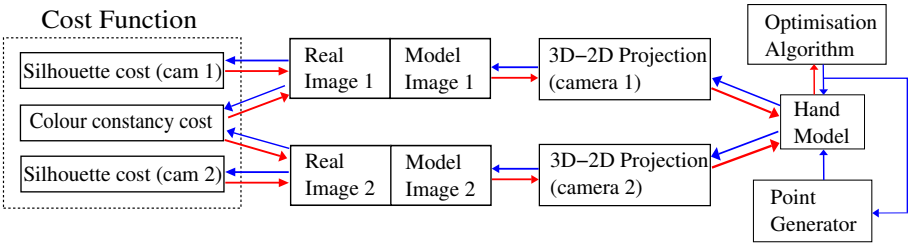


Fig. 1. Flow diagram showing the components making the tracking system

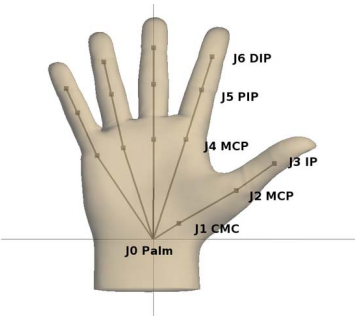


Fig. 2. Deformable hand model used by our tracker

2.2 3D to 2D Projection Pipeline

This part projects the ith sample point p_i on the hand to a model image plane. Let A_j be the rigid transformation that takes a point in the world coordinates and transforms it to the jth camera coordinates. Let K_j be the calibration matrix of the jth camera. Then $s_{i,j}$, the projection of the ith sample point on the jth image plane is given as

$$s_{i,j} = K_j A_j p_i. \tag{1}$$

2.3 Cost Function

Assigned to each pixel coordinate $s_{i,j}$ is a YUV value $I(s_{i,j}) \in \mathbb{R}^3$ and a silhouette cost value $V(s_{i,j}) \in \mathbb{R}^+$. Note that $s_{i,j}$ depends on x, where x is the vector of the 26 hand parameters. Both I and V are dependent on the set of optimal hand parameters x^* in the sense that x^* determines the real image seen by the cameras. I and V are used in the construction of our cost function $C_{x^*}(x)$.

$C_{x^*}(x)$ comprises of two parts, a silhouette cost function $C_s(i,x)$ and a cost function using the colour constancy assumption $C_c(i,x)$ per sample point i. Let α be a scaling factor for $C_s(i,x)$. Then the overall cost function $C_{x^*}(x)$ we wish to minimise is

$$C_{x^*}(x) = \frac{1}{N}\sum_{i=1}^{N}(\alpha C_s(i,x) + C_c(i,x)), \tag{2}$$

where N is the cardinality of a set of points on the surface of the hand model that is chosen to be sufficiently dense.

Silhouette Cost Function. Silhouette information is used as a global constraint on the region which the projected hand model can occupy. Silhouette images are obtained from the real images via background subtraction.

The chamfer 5-7-11 distance transform [12] is then applied over the silhouette image, assigning a distance value V to each pixel based on the pixel's proximity to the closest pixel that belongs to the hand silhouette. The silhouette cost function over j camera views is given by

$$C_s(i,x) = \sum_j V(s_{i,j}), \tag{3}$$

Colour Constancy Cost Function. The colour constancy assumption is used for local fine tuning by resolving pose ambiguities in silhouette information. For two camera views, it is given by

$$C_c(i,x) = \frac{1}{2}||I(s_{i,1}) - I(s_{i,2})||^2. \tag{4}$$

Using a 2-norm for the colour constancy cost function and a 1-norm for the silhouette cost function is an attempt to make the overall cost function more robust. When a parameter is far from the optimal value, the silhouette cost function dominates, causing $C_{x^*}(x)$ to behave linearly. When a parameter is close to the optimal value, the colour constancy cost function dominates, and $C_{x^*}(x)$ becomes quadratic. Section 3 will show that C_{x^*} has a unique global minimum for almost all possible values of x^*.

3 A Unique Minimum Exists

A unique global minimum for C_{x^*} does not exist for all x^*, *i.e.* each possible pair of real camera images. As a trivial example, one cannot determine the parameter values of a joint if the joint is occluded in both camera views. However, we will show that for a substantial subset of the possible x^*, there is always a unique global minimum at x^*. We use the term 'substantial' to mean that the exceptions can be described by a finite set of (not necessarily polynomial) equations. Such exception cases will be highlighted. The following assumptions are used to ease the analysis:

1. The y-axes of both camera image planes are parallel to each other, but the x-axes are not.
2. The palm is modelled by a rectangular cuboid and the digits of the hand are modelled by chains of cylinders. Our proof of proposition 1 will rely on a suitable choice of sample points. This choice becomes only easier for a more structured hand model. Hence the proof applies *a fortiori* to our hand model (figure 2).

3. The hand model has a Lambertian surface and has a uniform texture. Hence the YUV value of a point on the hand is completely determined by the surface normal and the light direction.
4. Only one light source illuminating from the front.

We also exclude the aforementioned trivial example in the analysis by assuming that all hand segments are at least partially visible in both camera views.

Proposition 1. *The cost at the optimal position* $C_{x^*}(x^*) = 0$. *Perturbing* x^* *to* $x \neq x^*$ *strictly increases* $C_{x^*}(x)$.[1]

The first part of the proposition is obvious, because at the optimal position, all the sample points lie in the silhouette and each point on the Lambertian surface of the hand model will have the same YUV value seen from different camera views. In the latter part, perturbing x^* will cause certain parts of the hand to move. We denote the points on the hand affected by this perturbation as 'active points'. Figure 3 is the tree of possibilities that can occur for the active points. We now examine each of these cases in detail.

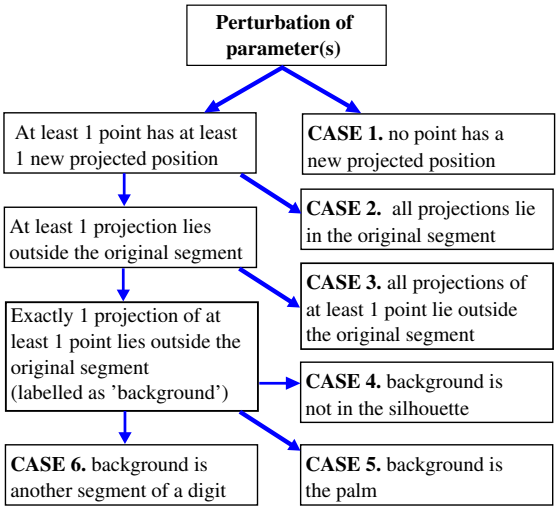


Fig. 3. Possible scenarios under a perturbation

3.1 Case 1

Eight points that lie in a non-degenerate configuration in Euclidean space uniquely define the epipolar geometry of the camera pair [13]. Conversely, a known epipolar geometry uniquely defines the projections of eight points that

[1] Note that proposition 1 does not preclude the existence of other stationary points in the cost function.

belong to a non-degenerate configuration. Let γ be the set of eight points in a non-degenerate configuration, chosen from the set of active points on a rigid segment of the hand. Then, a perturbation will move at least one of the eight points in γ. Thus, case 1 cannot occur.

3.2 Case 2

We ignore the trivial example of a cylindrical segment rotating around the main cylindrical axis as this type of movement is not possible for the digits of the hand without making the palm rotate, which in turn causes other digits to move outside their original positions.

For a cylindrical segment to lie inside the original region of a given camera view after perturbation, it can only move in a conic region of the plane spanned by the end points of the cylinder to the camera's optical centre. Given that there are two cameras (see figure 4), the intersection of the two conic planes is the only region where movement is allowed.

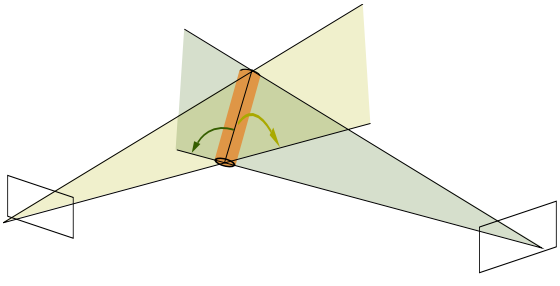


Fig. 4. Foreshortening of a cylindrical segment when the main axis does not lie on the epipolar plane

This intersection specifies the position of the cylinder uniquely unless the conic regions lie on the same plane, namely the epipolar plane spanned by one end of the cylinder (see figure 5, right). If the cylinder's main axis lies on this plane, then movements on the plane can cause the resulting projection to lie within the original segment for both cameras. For convenience, we shall denote this set of movements as κ.

Pure translational movements on the plane belong to κ (see figure 5, left) only if the projection of the cylindrical segment is longer than the baseline of the camera pair. In our setup, the baseline is much longer than all the segments of the hand, so pure translational movements can be ignored.

A combination of rotational and translational movements on the plane belong to κ if the projection of the cylinder's main axis to the camera image plane is shorter in both cameras after the rotational movement, and the translation movement only moves the perturbed segment within the original region.

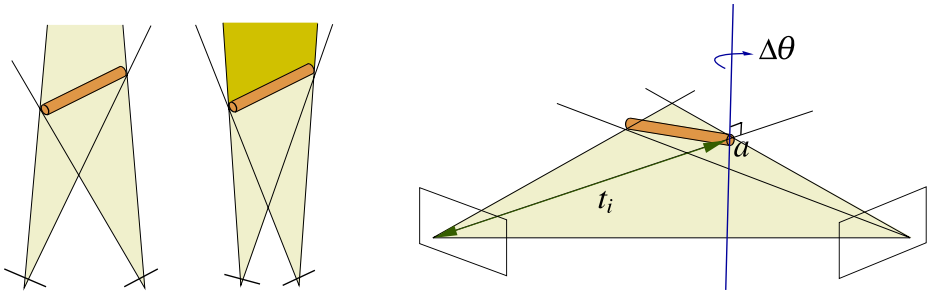


Fig. 5. Left: A cylinder undergoing a pure translation violates the condition for case 2, unless the camera baseline is shorter than the length of the cylinder (*e.g.* 2nd diagram from the left). The dark yellow area indicates the region where the cylinder can translate to. Right: Foreshortening in both cameras when the cylindrical segment rotates on the epipolar plane spanned by a, the centre of rotation.

Without loss of generality, we take the epipolar plane to be the plane spanned by the x and z axis in the world coordinates. It can be shown that for C_{x^*} not to increase after a rotation, $\triangle\theta$, and a restricted translation, the following equality must hold:

$$\frac{T_1}{T_1 + l \sin \triangle\theta} = \frac{T_2}{T_2 + l \sin \triangle\theta}, \tag{5}$$

where l is the coordinate of a surface point along the major axis of the cylinder. Note that for the ith camera,

$$T_i = t_{i,x} \sin \alpha_i + t_{i,z} \cos \alpha_i, \tag{6}$$

where α_i is the rotation angle and t_i is the translation vector that transforms a point from the local coordinates of cylinder to the coordinates of the ith camera.

For the equality to hold (and therefore C_{x^*} not to increase), either the perturbation is zero (*i.e.* $\triangle\theta = 0$) or $T_1 = T_2$.

Substituting the geometry of camera placement implies

$$\alpha_1 = tan^{-1}(-\frac{D_z}{D_x}) - \theta_r, \tag{7}$$

where θ_r, the rotation angle, and D, the translation vector, are the transformation parameters that convert points from the local coordinates of camera 1 to camera 2. Hence (7) is the only choice for x^* that might not cause C_{x^*} to increase.

The same argument can be applied to the palm, as the palm is attached to the digits, which are cylindrical chains. However, this ambiguity for the palm can only occur if a) the palm and the digits all lie on the epipolar plane or b) all digits of the hand are touching their adjacent digits to form a convex shape. Condition b) ensures that there no gaps between the fingers that would otherwise lead to case 4 when movement occurs on the epipolar plane.

3.3 Case 3

By the continuity argument, one can show that it is not possible to move a hand segment completely off the original segment in both camera views without causing other segments of the hand to partially move from their original position or to leave the silhouette. Therefore one can use the arguments in cases 4, 5 or 6 for the other hand segments to show that C_{x^*} increases.

3.4 Case 4

If one of the active point projections lies outside the original segment and falls outside the silhouette region, then C_{x^*} increases due to the silhouette cost function C_s.

3.5 Case 5

Let p_1, p_2 be the projections of the active point p in the two camera views. p_1 is projected to a surface point s_c on the original cylindrical segment while p_2 is projected onto a surface point s_p on the palm. Suppose the YUV value at p_1 is the same as in p_2, which implies that the surface normal at s_p and s_c are equidistant to the light direction. Then this point will not increase C_{x^*}.

However note that p is chosen from a closed set, and the projection of closed sets remains closed. Therefore the neighbourhood of p will also be projected onto the palm. We can always choose p' from this neighbourhood such that the surface normal at s'_c and s_c are not equidistant to the light direction. Since the palm is modelled as a plane on a cuboid, it has a constant surface normal. Therefore p'_1 will be different to p'_2, and so p' increases C_{x^*}.

The only situation where C_{x^*} does not increase is when the light direction l comes from behind the palm or is orthogonal to the surface normal n_{palm} of the palm, *i.e.* $l \cdot n_{palm} \geq 0$. In this situation, the palm is completely black. This can lead to ambiguity as there always exists a closed set of points with different surface normals on the cylinder that is always black. If p belongs to this set, C_{x^*} will not increase as the neighbourhood of points will also be black. One should note that this situation has been excluded previously in the list of assumptions.

3.6 Case 6

Firstly we assume that none of the active points fall into case 4 or case 5, otherwise we can use those points instead to show that C_{x^*} increases after the perturbation.

Let p_1, p_2 be the projections of the active point p in the two camera views. p_1 lands on a surface point s_c of the original segment c, while p_2 lands on a surface point s_d of another segment d. We first take the simple situation where the main axes of c and d are parallel to the y-axes of both cameras (see figure 6).

Assume that p_1 and p_2 (and thus s_c and s_d) have the same YUV value u, which is not black and is not the brightest YUV value on both cylinders. Also

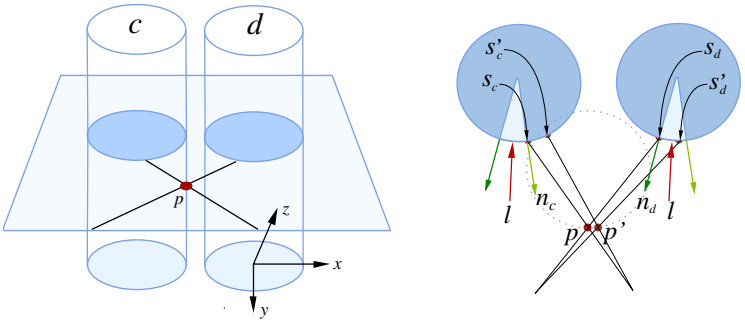


Fig. 6. Left: The setup of the simple situation. Right: The cross-section of the setup. l indicates the projected light direction. The dotted circle indicates the perturbed cylinder. The lighter regions are parts where the YUV value is greater than u.

assume that the surface normals n_c and n_d at s_c and s_d respectively are different. We know that the light direction is equidistant to n_c and n_d. This produces the lighting pattern as seen in figure 6. Points to the right (anticlockwise) of s_c on the cylinder are darker than u while points to the left are lighter than u. This pattern is reversed on d, where points to the right of s_d are lighter than u.

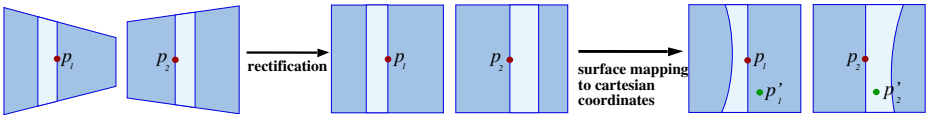


Fig. 7. The projections of the neighbourhood of p after image rectification. The light regions are parts where the YUV value is greater than u.

Suppose we examine a square neighbourhood of p in 3D space, visible in both cameras (see figure 7). Its projection is an affine transformation of the square, and is different in both cameras. Suppose the two views are rectified. Note that the bounding lines marking the lighter region do not cross over each other after rectification. If we account for the fact that one can only sample on the neighbourhood of p in 3D space that belongs to the surface of the perturbed cylinder, then these bounding lines become curves after rectification of the surface to cartesian coordinates. As seen in figure 7, one can always choose a point p' in the final rectified neighbourhood such that p'_1 lands on the lighter region while p'_2 lands on the darker region or vice versa. Hence C_{x^*} will increase.

We now generalise to other cylinder configurations. For convenience, we use c as an example, and denote the band of lighter region as R, the light direction as l, the bounding line of R that contains s_c (or p_1 in the projection) as b_1 and the other bounding line in the same image as b_2. Rotation of the cylinder around the z-axis changes the slope of the bounding lines and narrows R as the n_c becomes

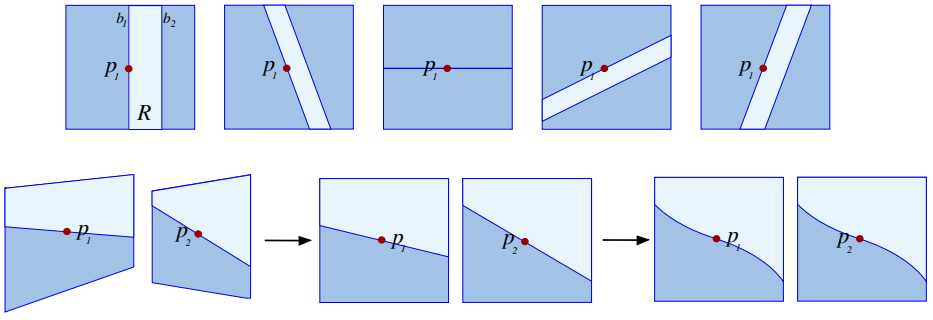


Fig. 8. Top: R varying upon different z-rotations. Note the degenerate (3rd) case. Bottom: Ambiguity is possible if the bounding line matches exactly after rectification.

more aligned to the light direction. Degeneracy D occurs when n_c is the surface normal on c that is closest aligned to the light direction. Then R does not exist. Note that the light direction determines the particular z-rotation that results in this degeneracy. As n_c rotates away from the light source, R reappears but b_2 now flips to the other side of b_1 (see figure 8).

A rotation on the x-axis determines the brightest value on the cylinder and changes the slope of the bounding lines seen in the projection. However the surface area of R in 3D space remains the same. Degeneracy occurs when an x-rotation causes $n_c \cdot l \geq 0$. Then s_c and its neighbourhood will be black.

Additionally, given that the projection of the neighbourhood of p is an affine transformation, it may be possible to construct cases where b_1 of one image aligns with b_1 of the other image after rectification (see figure 8), creating ambiguity. Ambiguity also arises when $n_c = n_d$ or when D occurs in both cylinders. Then whether one can choose a p' that causes C_{x^*} to increase will depend on the rate of curvature change on c, d and the perturbed cylinder. It also depends on the camera placement. Other than these degenerate cases, one can always choose p' such that p'_1 lands on R and p'_2 lands outside of R or vice versa. Note that these degenerate cases are rare.

4 Stochastic Approximation

Noise is a highly noticeable process and occurs at various parts of the tracking system. For example there is measurement noise from the camera and discretisation noise in the evaluation of image gradients that are computed using Sobel masks. Also, we are only sampling a small subset of $n \ll N$ points from the hand model to evaluate an approximation of the true cost function. This introduces sampling noise. For a tracking system to perform adequately, the effect of noise must be addressed and minimised.

Stochastic approximation [14,15] is a technique for finding the root of a function $f(x)$ where only noise-corrupted measurements of function values are available. This can be applied to an optimisation setting like the one in our tracker

if we set $f(x)$ to be the gradient of our cost function $C_{x^*}(x)$. Then finding the root of f equates to finding the critical point (the minimum) of C_{x^*}.

Let the random variable $Y_t(x)$ be the noisy observation of $f(x_t)$, *i.e.* the gradient of $C_{x^*}(x_t)$ in parameter space, and a_t be the step size in our optimisation procedure. Then a possible iterative scheme for stochastic approximation is

$$X_{t+1} = X_t - a_t Y_t. \tag{8}$$

This is the so-called Robbins-Monro method [14,15]. Note that X_t is a random variable and x_t is the actual event of X_t at time t. X_t of (8) converges to x^* in mean square and with probability 1, *i.e.*:

$$\lim_{t\to\infty} E[(X_t - x^*)^2] = 0 \quad \text{and} \quad P(\lim_{t\to\infty} X_t = x^*) = 1, \tag{9}$$

if the following conditions are met:

1. A bound on the step size a_t,

$$\sum_{t=1}^{\infty} a_t = \infty, \sum_{t=1}^{\infty} a_t^2 < \infty. \tag{10}$$

2. Y_t is unbiased,

$$E(Y_t) = f(x_t). \tag{11}$$

3. Y_t has uniformly bounded variance in the sense,

$$\sup\{Var(Y(x)) : x \in \mathbb{R}^K\} < \infty. \tag{12}$$

4. f is well-behaved around x^* in the sense

$$\inf\{(x - x^*)^T f(x) : \epsilon < ||x - x^*|| < \epsilon^{-1}\} > 0, \tag{13}$$

for all $\epsilon \in \mathbb{R}, 0 < \epsilon < 1$.

The following subsections will demonstrate how our hand tracking system meets these requirements.

In the classical Robbins-Monro scheme [14], the step size a_t of the iterate update (8) is set to $\frac{q}{t}$ for some constant q. In practice, this method may not be desirable due to its slow convergence rate.

In practice, we use the SMD algorithm [16,17] which tends to converge much faster. Although there is no convergence proof available yet for SMD, it has been shown empirically to work well under noisy conditions in many practical situations. Prior applications of SMD to body/hand tracking work include [18,5]. The following arguments are independent of the chosen optimisation algorithm.

4.1 Y_i Is an Unbiased Estimator

Noise from image gradients is unbiased since the Sobel mask used for calculating image gradients is centred and symmetric. Bias due to camera noise is minimised as the cameras are calibrated prior to use.

In addition, bias in Y_i heavily depends on the sampling scheme used to evaluate the gradient estimates. We are aware that our way of proving condition (13) in section 4.3 potentially introduces certain requirements/constraints on the sampling scheme, thereby producing bias. However, this is easily mitigated by taking more sample points.

4.2 Y_i Has Uniformly Bounded Variance

The observed gradients Y_i generated by the cost function have a bounded variance. For the silhouette cost function, the variance in distance estimation is uniformly bounded by the size of the image. Therefore the gradient estimates generated by it via finite differences will also be bounded. The variance in Y_i due to the colour constancy cost function is also uniformly bounded since the range of YUV values at each pixel is uniformly bounded.

4.3 $f = \bigtriangledown C_{x^*}$ Is Well-Behaved

Condition (13) does not hold globally for all x^*, but we can show it holds locally for most x^*. To satisfy (13), it is sufficient that f has a zero root at x^* and that the Jacobian J_f of f (*i.e.* the Hessian of C_{x^*}) is positive definite (*i.e.* our cost function C_{x^*} has a strict local minimum at x^*). The former part was shown in section 3. We now show that J_f is positive definite.

The tracker can be viewed as a composite function $C \circ M$, where C is the cost function and M the remaining parts of the tracker. Because $C_{x^*}(x^*) = 0$ (proposition 1), the Hessian H of $C \circ M$ (or the J_f of f) at the minimum can be rewritten [17] as

$$J_f = H = \frac{1}{N} \sum_{i=1}^{N} J_{M,i}^T H_c J_{M,i}, \tag{14}$$

where H_c is the Hessian of the cost function and $J_{M,i}$ is the Jacobian of M at the ith sample point.

H_c is the sum of the Hessian H_{c_s} of the silhouette cost function and the Hessian H_{c_c} of the colour constancy cost function. H_{c_c} is given as,

$$H_{C_c} = \begin{pmatrix} I_{3\times3} & -I_{3\times3} \\ -I_{3\times3} & I_{3\times3} \end{pmatrix}. \tag{15}$$

H_{c_s} can either be positive definite or zero at the minimum; the latter due to ambiguity for certain poses of the silhouette. We can ignore the H_{c_s} terms as they cannot decrease the rank of H. It is sufficient to show that J_f attains full rank due to H_{c_c} alone.

J_f is at least positive semi-definite since the summands in (14) are positive semi-definite. Also the rank of J_f is non-decreasing when adding samples since the summands are added.

As an empirical verification, each frame in the test video sequence was tested to see if taking an adequate amount of sample points led to the J_f estimate achieving full rank at the minimum. On average, approximately 100 points were required for the J_f estimate to achieve full rank.

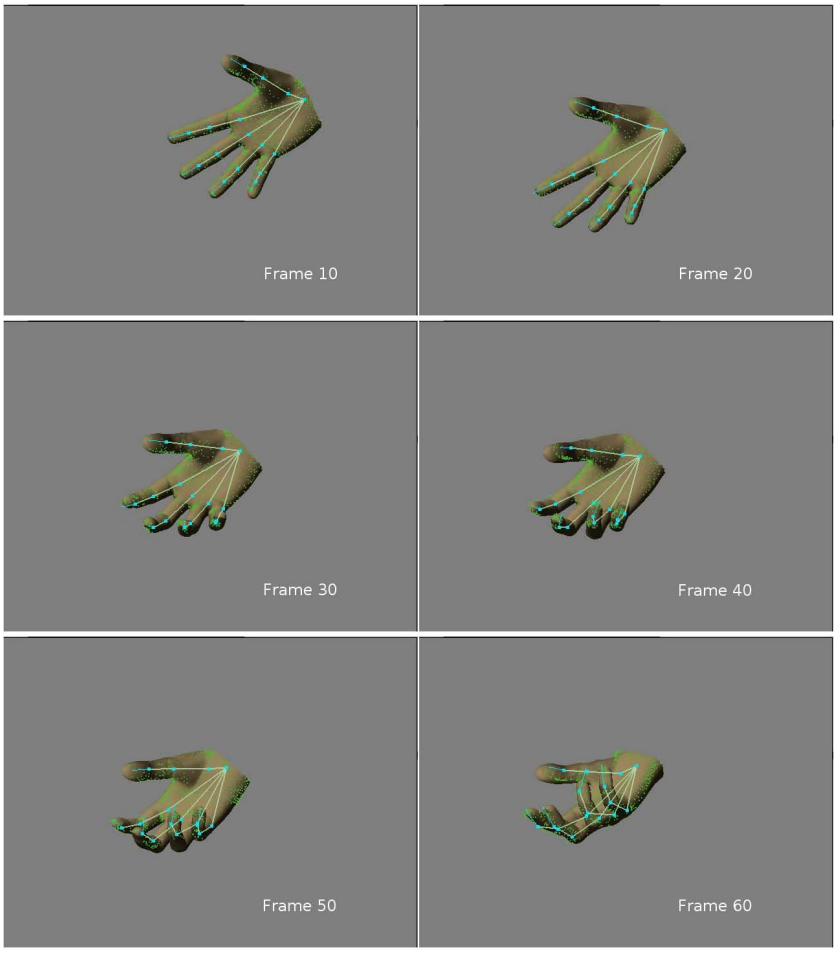


Fig. 9. Tracking results at every 10th video frame

5 Tracking Results

The tracker has been tested over a short video sequence of 60 frames (640×480 pixels) that shows the hand extending to grip an imaginary object (figure 9). The sequence contains elements of lateral translation, wrist rotation, and articulated motion of the digits in the form of gripping. To obtain a ground truth assessment of the tracker accuracy, the hand model is initially fitted frame by frame, by eye over the real captured sequence. The parameter values obtained from this procedure are then taken to be the ground truth. Using these parameter values, a synthetic sequence is rendered using OpenGL. Tracking performance is evaluated by running the tracker on the synthetic sequence over $G = 50$ trials. The experiment was conducted on a P4 3.4GHz machine.

Approximately $n = 280$ active sample points were used to track the moving hand. The optimisation algorithm was allowed to perform a maximum of 50 iterations per frame. In terms of computational speed, an average of 1.4s were required to track one frame, of which 0.8s were required for image preprocessing such as extracting silhouette and calculating image gradients. The remaining 0.6s were spent by the iterations of the optimisation procedure. The code for the tracker has not been optimised and the parallelizable structure of the system has not been exploited.

The error measures used to evaluate performance are based on the difference in distance between actual and predicted joint positions in Euclidean space. The first error measure used is the overall mean error. Let $p_{k,g}$ and a_k be the predicted and actual 3D positions of the kth joint for the gth trial. Then the overall mean error is given as

$$\frac{1}{GK}\sum_{g}^{G}\sum_{k}^{K}||a_k - p_{k,g}||, \tag{16}$$

where K is the total number of joints in the hand model. Figure 10 (left) shows the tracking accuracy over the test sequence. The overall mean error is given in Euclidean and image space.

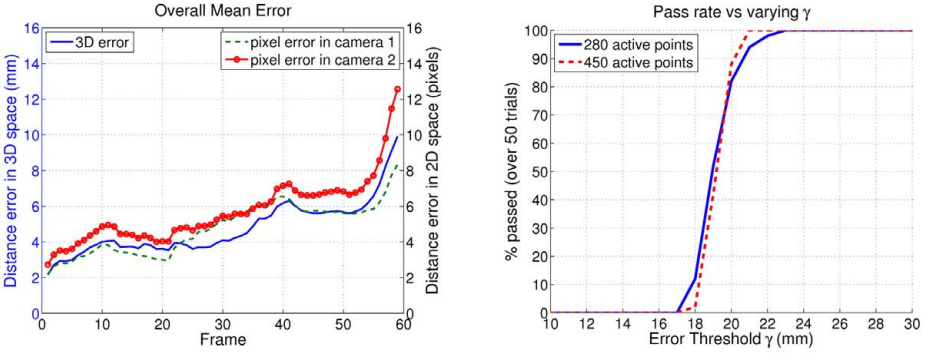


Fig. 10. Left: Overall mean error of the video frames over time. Right: Pass rates for varying γ values.

To put the error measurements into perspective, the hand model in a relaxed open palm position can be roughly bounded by a $180 \times 100 \times 30mm$ cuboid and is located approximately $1m$ from the cameras (roughly bounded by a rectangle of 220×150 pixels in image space). The cameras are pointed towards the hand in a convergent setup, at an angle of 30° from each other. The baseline between the two cameras is $0.85m$.

To classify whether the tracker has irrecoverably lost track of the hand, we introduce another measure. A tracker is classified as having passed the tracking

sequence if during the entire sequence, the error distance between predicted and actual position of each joint is below an error threshold γ. Figure 10 (right) plots the results for varying γ. The figure also shows that increasing the number of active sample points improves the pass rate.

6 Conclusion

A 3D hand tracking system has been presented that uses silhouette and the colour constancy assumption. A theoretical proof of local stochastic convergence has been provided for the tracker. It shows that except for certain degenerate hand pose configurations, local stochastic convergence holds. It is possible that such a system can be generalised to multiple cameras or to track other articulated structures such as the human body. Experimental results on synthetic images are promising, although we believe that a more sophisticated sampling scheme will improve tracking accuracy.

Acknowledgements

We wish to thank the BIWI laboratory, ETH Zürich for permitting the use of their hand model.

NICTA is funded by the Australian Government's Department of Communications, Information Technology and the Arts (DCITA) and the Australian Research Council (ARC) through Backing Australia's Ability and the ICT Centre of Excellence program.

References

1. Erol, A., Bebis, G.N., Nicolescu, M., Boyle, R.D., Twombly, X.: A review on vision-based full DOF hand motion estimation. In: Vision for Human-Computer Interaction, vol. III, p. 75 (2005)
2. Wang, L., Hu, W.M., Tan, T.N.: Recent developments in human motion analysis. Pattern Recognition 36(3), 585–601 (2003)
3. Sminchisescu, C., Triggs, B.: Covariance scaled sampling for monocular 3D body tracking. In: CVPR, vol. I, pp. 447–454 (2001)
4. Sudderth, E.B., Mandel, M.I., Freeman, W.T., Willsky, A.S.: Visual hand tracking using nonparametric belief propagation. In: Workshop on Generative Model Based Vision, 189 (2004)
5. Kehl, R., Gool, L.J.V.: Markerless tracking of complex human motions from multiple views. Computer Vision and Image Understanding 103(2-3), 190–209 (2006)
6. Carranza, J., Theobalt, C., Magnor, M., Seidel, H.: Freeviewpoint video of human actors. In: ACM Transactions on Graphics. ACM SIGGRAPH, pp. 569–577. ACM Press, New York (2003)
7. Theobalt, C., Carranza, J., Magnor, M.A., Seidel, H.P.: Combining 3d flow fields with silhouette-based human motion capture for immersive video. Graph. Models 66(6), 333–351 (2004)

8. Sundaresan, A., Chellappa, R.: Multi-camera tracking of articulated human motion using motion and shape cues. In: Narayanan, P.J., Nayar, S.K., Shum, H-Y. (eds.) ACCV 2006. LNCS, vol. 3852, pp. 131–140. Springer, Heidelberg (2006)
9. Lu, S., Metaxas, D., Samaras, D., Oliensis, J.: Using multiple cues for hand tracking and model refinement. In: IEEE Computer Vision and Pattern Recognition or CVPR, vol. II, pp. 443–450. IEEE Computer Society Press, Los Alamitos (2003)
10. Balan, A.O., Sigal, L., Black, M.J.: A quantitative evaluation of video-based 3D person tracking. In: International Workshop on Performance Evaluation of Tracking and Surveillance, pp. 349–356 (2005)
11. Lewis, J.P., Cordner, M., Fong, N.: Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In: SIGGRAPH 2000, New York, USA, pp. 165–172 (2000)
12. Borgefors, G.: Distance transformations in digital images. Comput. Vision Graph. Image Process. 34(3), 344–371 (1986)
13. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge (2004)
14. Robbins, H., Monro, S.: A stochastic approximation method. Annals of Mathematical Statistics 22, 400–407 (1951)
15. Blum, J.R.: Multidimensional stochastic approximation methods. Annals of Mathematical Statistics 25, 737–744 (1954)
16. Schraudolph, N.N.: Local gain adaptation in stochastic gradient descent. In: ICANN, Edinburgh, Scotland, pp. 569–574. IEE, London (1999)
17. Schraudolph, N.N.: Fast curvature matrix-vector products for second-order gradient descent. Neural Computation 14(7), 1723–1738 (2002)
18. Bray, M., Koller-Meier, E., Müller, P., Schraudolph, N.N., Gool, L.V.: Stochastic Optimization for High-Dimensional Tracking in Dense Range Maps. IEE Proceedings Vision, Image & Signal Processing 152(4), 501–512 (2005)

Nonparametric Density Estimation with Adaptive, Anisotropic Kernels for Human Motion Tracking*

Thomas Brox[1], Bodo Rosenhahn[2], Daniel Cremers[1], and Hans-Peter Seidel[2]

[1] Computer Vision Group, University of Bonn
Römerstr. 164, 53117 Bonn, Germany
{brox,dcremers}@cs.uni-bonn.de
[2] Max Planck Center for Visual Computing and Communication
Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
{rosenhahn,seidel}@mpi-sb.mpg.de

Abstract. In this paper, we suggest to model priors on human motion by means of nonparametric kernel densities. Kernel densities avoid assumptions on the shape of the underlying distribution and let the data speak for themselves. In general, kernel density estimators suffer from the problem known as the curse of dimensionality, i.e., the amount of data required to cover the whole input space grows exponentially with the dimension of this space. In many applications, such as human motion tracking, though, this problem turns out to be less severe, since the relevant data concentrate in a much smaller subspace than the original high-dimensional space. As we demonstrate in this paper, the concentration of human motion data on lower-dimensional manifolds, approves kernel density estimation as a transparent tool that is able to model priors on arbitrary mixtures of human motions. Further, we propose to support the ability of kernel estimators to capture distributions on low-dimensional manifolds by replacing the standard isotropic kernel by an adaptive, anisotropic one.

1 Introduction

In recent years, human tracking has emerged as a vivid research area. In particular 3D human tracking, where one seeks to estimate the pose and joint angles of a 3D human model from 2D images, has attracted a lot of attention [9]. Having applications in surveillance and biomechanics, human tracking also serves as a playground for new machine learning techniques. Due to self-occlusions, inaccurate, corrupted, or missing data, it requires the use of prior knowledge on typical human poses and movements in order to avoid ambiguous solutions. Moreover, the solution space generally comprises multiple locally optimal solutions. This is a great challenge for optimization algorithms if not being supported by predictions generated from strong priors.

* We thank U. Kersting for providing data from a marker-based system, as well as the German Research Foundation for their financial support.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 152–165, 2007.

Consequently, the literature provides numerous works on different learning techniques that can be used to exploit prior knowledge for human tracking. These works range from rather simple explicit joint angle limits [19,6], over static pose priors [17,2], to priors on motion dynamics [16]. Some recent dynamic models are based on sophisticated nonlinear regression methods including nonlinear dimensionality reduction [5,20].

Most of these works stick to a maximum a-posteriori (MAP) formulation of the tracking problem. Given the input image I in the current frame and pose configurations in previous frames $\chi_{t-1}, ..., \chi_{t-k}$, one looks for the new configuration χ_t that maximizes

$$p(\chi_t|I, \chi_{t-1}, ..., \chi_{t-k}) \propto p(I|\chi_t)p(\chi_t|\chi_{t-1}, ..., \chi_{t-k}). \quad (1)$$

While the first factor considers how well a solution χ_t explains the image data, the second factor represents the conditional prior probability density of some pose given the poses of previous frames. One can directly model this prior, which leads to regression methods. Usually, such methods comprise a parametric component, which means that they cannot accurately model a prior consisting, for instance, of running and jumping motions, since the parametric model would mix up both motion patterns to yield an (unprecise) mean prediction. In order to handle such cases consisting of multiple motions, one has to employ a mixture of regressors [8,11,18], which includes many critical hyperparameters and is quite demanding with regard to optimization.

In this paper, we pursue an alternative strategy. Since $p(a|b) = \frac{p(a,b)}{p(b)}$, and we maximize with respect to a, $p(b)$ can be neglected as a constant factor and we may consider

$$p(\chi_t|I, \chi_{t-1}, ..., \chi_{t-k}) \propto p(I|\chi_t)p(\chi_t, ..., \chi_{t-k}). \quad (2)$$

Here the second factor is the joint prior density of poses in previous frames and the current one. Such an unconditional probability density can be estimated from training samples using a Parzen estimator. Since the density is fully nonparametric, it can easily model arbitrary mixtures of motion patterns. The Parzen estimator only implies the assumption of a locally smooth density. Consequently, it can capture *all* smooth densities provided there are enough training samples.

The input space of human motion is rather high-dimensional. For a reasonable human body model, at least 20 degrees of freedom are needed. Looking only 4 frames into the past, already implies a 100-dimensional space. It is well known that estimating wide-spread densities in such spaces with a typical kernel estimator, would need huge amounts of training data [15]. However, in practice, this problem is often less severe. This is because high-dimensional spaces are often only sparsely populated, i.e., the density to be estimated concentrates on a small subspace, a low-dimensional manifold in the high-dimensional space. In this paper, we demonstrate that in case of human motion tracking, already the standard Parzen estimator can deal with a 121-dimensional space.

Nevertheless, this estimator is not optimal for such high-dimensional spaces. This is due to the fixed isotropic kernel, which does not adapt to the local structure of the subspace. Hence, the Parzen estimator looses predictive power in

normal direction to the manifold. This drawback can be circumvented by introducing anisotropy in the estimation process. Therefore, we propose to replace the isotropic kernel of the standard Parzen estimator by adaptive anisotropic kernels. The same concept has been proposed in the context of general density estimation in [14,21]. Also the work in [4] based on kernel PCA can be interpreted as sort of an anisotropic kernel density estimator. However, the latter has quadratic complexity in the test phase, which is problematic when the number of training samples becomes large.

2 Anisotropic Kernel Density Estimation

Consider some prior knowledge given by a set of training samples $\{x_i | i = 1, ..., N\}$. In order to integrate such knowledge into a Bayesian model, one must estimate a probability density from the samples. In contrast to typical parametric densities, such as a Gaussian density, which are very restricted in the priors they can model, this paper is concerned with nonparametric kernel densities. The classic Parzen-Rosenblatt density estimator employs an isotropic kernel $K(x, x')$ with a fixed width h. Given such a kernel, the estimated density reads [1,12,10]:

$$p(x) = \frac{1}{N} \sum_{i=1}^{N} K_h(x, x_i). \tag{3}$$

A very common kernel is the Gaussian kernel

$$K_h(x, x') = \frac{1}{(2\pi h^2)^{\frac{D}{2}}} \exp\left(-\frac{\|x - x'\|^2}{2h^2}\right), \tag{4}$$

where D denotes the dimensionality of the data. This density estimator, though simple, reveals many advantages. Firstly, it can model arbitrary densities and one can show that in the limit, for $N \to \infty$ and $h \to 0$ adequately, the estimator converges to the true density [15]. Secondly, the estimator is very transparent. In contrast to many learning techniques that rely on modeling in an abstract feature space, the Parzen estimator is easily interpretable. Moreover, it contains only a single hyperparameter, the kernel width h, which can be estimated efficiently from the training data via cross-validation or, depending on the application, by even simpler criteria like average nearest neighbor distance. In contrast to Gaussian mixture models or related techniques, there is no need to determine the number of mixture components, which is a difficult non-convex optimization problem.

As mentioned in the introduction, the main weakness of the Parzen estimator appears when it is employed in high-dimensional spaces where the support of the density is located on a low-dimensional manifold. Then it looses predictive power in normal direction to this manifold due to the fixed isotropic kernel. This is the motivation for using adaptive, anisotropic kernels leading to an anisotropic version of the Parzen estimator. Again, the density is a sum of kernels centered at the training samples

$$p(x) = \frac{1}{N} \sum_{i=1}^{N} K_i(x, x_i), \tag{5}$$

where now $K_i(x, x_i)$ is the locally adaptive anisotropic Gaussian kernel

$$K_i(x, x_i) = \frac{1}{|2\pi\Sigma_i|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x - x_i)^\top \Sigma_i^{-1}(x - x_i)\right). \tag{6}$$

Its window width and preferred direction is defined by the covariance matrix Σ_i. This covariance matrix is computed locally by means of

$$\Sigma_i = \alpha\mathbf{1} + \sum_{j=1}^{N} K_h(x_i, x_j)(x_i - x_j)(x_i - x_j)^\top, \tag{7}$$

where $\alpha\mathbf{1}$ denotes the identity matrix scaled by a regularization parameter α and $K_h(x, x')$ is the isotropic Gaussian kernel stated in (4).

This anisotropic kernel density estimator has several nice properties. Firstly, the absolute width of the kernel is locally adaptive. This allows for smaller windows in areas with many training samples, whereas sparsely populated areas can still be approximated by larger windows. Secondly, the windows have a preferred orientation in which the kernel size is increased. Since the kernel integrates to 1, this effect automatically decreases the kernel size in orthogonal directions. Such an anisotropy is particularly useful to model data on low-dimensional manifolds, as most of the kernel's power is focused on the tangential space of the manifold. In contrast to Gaussian mixture models, there is still no need to determine the number of mixture components. The estimator can be regarded as a degenerate version of a Gaussian mixture, where the number of components equals the number of training samples. Obviously, this also provides an increased accuracy in respect to the density's local structure compared to a Gaussian mixture with only a small number of components.

The density estimator still imposes only two hyperparameters h and α. These hyperparameters can be estimated from the training data via leave-one-out (LOO) cross validation, i.e., one minimizes the following loss function based on Kullback-Leibler divergence

$$E(h, \alpha) = -\log\left(\sum_{i=1}^{N} \hat{p}_{i,h,\alpha}(x_i)\right), \tag{8}$$

where $\hat{p}_{i,h}(x)$ denotes the estimated probability density with parameters h and α when sample i has been removed from the training set. In the application case of human tracking, we found that one can simplify the parameter optimization by setting h to the average nearest neighbor distance of all training samples and $\alpha = \frac{h}{5}$. This is reasonable since training data is obtained via motion capture systems with a fixed frame rate, i.e., samples always come in larger groups.

Figure 1 demonstrates the qualitative difference between the isotropic kernel density estimator and the anisotropic one. Having some data points sampled

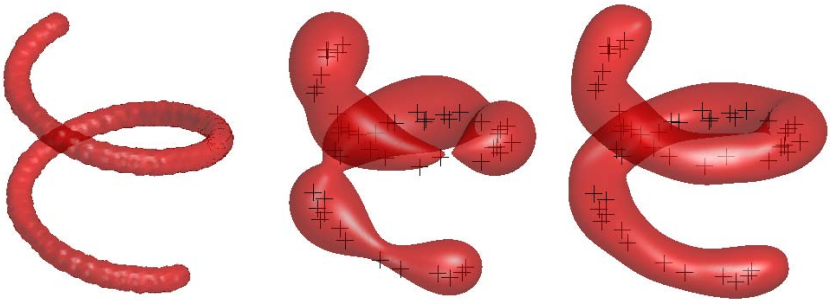


Fig. 1. From left to right: (a) Intrinsically one-dimensional **density** in three-dimensional space. **(b)** Density estimate with the conventional **Parzen** method (ISE $= 12.2 \cdot 10^{-4}$). **(c)** Density estimate with the **anisotropic Parzen** method (ISE $= 9.3 \cdot 10^{-4}$).

from the true density (left), the isotropic estimator yields a density estimate that approximates the true density quite well but lacks the ability to interpolate in some of the gaps (middle). In contrast, the anisotropic estimator focuses better on the structure of the density. This is also reflected by the lower integrated mean square error (ISE) between the true and the estimated density.

3 Kernel Densities in Human Tracking

Density estimators can be a valuable component in an application like human motion tracking. In this task, we expect a given surface model consisting of several limbs that are interconnected by predefined joints. The sought pose configuration χ at each frame consists of a global rigid body motion, represented by the six parameters of a twist ξ, as well as a number of joint angles $\Theta = (\theta_1, ..., \theta_M)^\top$. Estimation of these parameters at frame t from image data and poses from previous frames can be regarded in a MAP setting

$$p(\chi_t | I, \chi_{t-1}, ..., \chi_{t-k}) \propto p(I|\chi_t) p(\chi_t, ..., \chi_{t-k}), \tag{9}$$

where the conditional prior density $p(\chi_t|\chi_{t-1}, ..., \chi_{t-k})$ has already been replaced by the joint density $p(\chi_t, ..., \chi_{t-k})$, as explained in the first section of this paper. The right hand side consists of a data fidelity factor and the prior density of certain sequences of pose configurations.

3.1 Modeling the Data Fidelity

There are several ways to model the data fidelity, such as keypoint tracking or silhouette constraints. Since this issue is not the focus of this paper, we stick to an existing silhouette based method [13], where (9) is expanded to

$$p(\chi_t, \Phi | I, \chi_{t-1}, ..., \chi_{t-k}) \propto p(I|\Phi) p(\Phi|\chi_t) p(\chi_t, ..., \chi_{t-k}) \tag{10}$$

by introducing the silhouette represented as the zero level of a function $\Phi : \Omega \rightarrow \mathbb{R}$. Maximizing the probability in (10) is equivalent to minimizing its negative logarithm. With certain model assumptions on the appearance of the object and background region [13], this yields the energy

$$\begin{aligned} E(\chi_t) = &- \int_\Omega H(\Phi) \log p_1 + (1 - H(\Phi)) \log p_2 \, dx \\ &+ \lambda \int_\Omega (\Phi - \Phi_0(\chi_t))^2 \, dx - \log p(\chi_t, ..., \chi_{t-k}), \end{aligned} \tag{11}$$

where $H(s)$ is the step function that distinguishes the object and background region, p_1 and p_2 are densities of the intensity in these regions, $\Phi_0(\chi_t)$ is the level set function representing the silhouette of the projected model given the pose χ_t, and $\lambda = 0.05$ is a weighting parameter that steers how much the contour Φ may deviate from the model silhouette Φ_0.

In contrast to many tracking works that use a sampling strategy to minimize similar energies as the one in (11), we use a gradient descent in Φ and χ, which yields the next local minimum starting from some initialization χ^0. In the tracking context, finding the next local minimum can be sufficient, especially if the model is supported by a prior density that allows for reasonable predictions of poses in successive frames. For this reason, we now concentrate on the last term in (11), which comprises this prior density.

3.2 Modeling the Prior Density

For building a prior density, a set of training samples with certain motion patterns is required. A database with a rather large variety of motions is available at Carnegie Mellon University [3]. We used this database to assemble training samples for estimating a prior density.

In order to ensure certain invariance properties, and to keep the required number of samples as well as the dimensionality as small as possible, we arrange the sample vectors in the following way. Firstly, we restrict the degrees of freedom of our model to the 29 most important ones. , i.e., 3 dof at each shoulder, 1 dof at each elbow, 1 dof at each hand, 3 dof at each upper leg, 1 dof at each knee, 2 dof at each foot, and 1 dof at the neck. Together with the global rigid body motion, this yields a total of 29 dof. Further, since we are interested in invariance with regard to the location and orientation of the person, we only consider the joint angles at previous frames, not the global twist. Finally, for keeping the dimensionality small and nonetheless considering configurations that are several frames in the past, the time axis is non-uniformly sampled. In detail, we assemble vectors x_i, where the first six components are the twist parameters representing the rigid body motion between $t-1$ and t. The next $M = 23$ components are the absolute joint angles in t. There follow successively the M joint angles in $t-1$, in $t-2$, $t-5$, and $t-10$. Similar to the so-called snippets in [7], this yields training vectors x_i of dimension $D = 121$, from which a density can be estimated according to Section 2. In case the frame rate of the input sequence does not match the training sequences, the prior is scaled accordingly.

3.3 Density Gradient and Pose Prediction

For the minimization of (11) we are not interested in the absolute density, but in the local gradient of its logarithm. This gradient corresponding to the anisotropic density estimator in (5) reads:

$$\begin{aligned} K_i(x, x_i) &:= \exp\left(-\frac{1}{2}(x - x_i)^\top \sigma_i^{-1}(x - x_i)\right) \\ \frac{\partial \log p(x)}{\partial x} &= -\frac{1}{2}\frac{\sum_{i=1}^N K(x, x_i)\Sigma_i^{-1}(x - x_i)}{\sum_{i=1}^N K(x, x_i)}. \end{aligned} \tag{12}$$

Note that only the first $6 + M$ components of $\frac{\partial \log p(x)}{\partial x}$ are needed, since the pose at previous frames is fixed. Starting from some point x^0 and ignoring the data fidelity term, gradient ascent will converge to the local mode of the density in the $(6 + M)$D subspace, i.e., the most likely pose configuration in a local neighborhood given the poses at previous frames. In combination with the data fidelity term, the result is the local maximum a-posteriori solution given the image data *and* the prior density.

In some cases, one is indeed interested in the local mode of the density alone, starting from some motion vector x^0. In human tracking, this situation arises when predicting the pose in a successive frame, irrespective the image data, which may be unreliable without a good hypothesis of the pose in the new frame. For prediction, it is beneficial to estimate a density where the absolute joint angles at t in x_i are replaced by relative angles between $t - 1$ and t. This prevents predictions far from the tracked motion in case the training data are sparse and rather dissimilar from the tracked motion.

4 Experiments

Our experiments demonstrate that kernel density estimators in general, but in particular the one based on anisotropic kernels, are well suited to model dynamic motion priors in human tracking. Firstly, Figure 2 and a video in the supplementary material show that one can generate an enduring cyclic motion from the anisotropic density. For this motion generation, we simply provide a short sequence of poses. Starting the gradient descent (12) from the last such pose, and ignoring the image-driven part, yields an enduring running motion. This means, from previous poses alone, the density can predict a reasonable succession of poses like a regressor would do.

An important challenge in human tracking are monocular sequences. Since only few limbs are visible in a single view, the problem is generally underconstrained and prior assumptions, such as the suggested prior density, are needed for a unique solution. Figure 3 shows the tracking result for a standard test sequence[1]. The synthesized views confirm that the estimated 3D pose is very

[1] The sequence is available at `www.nada.kth.se/∼hedvig/data.html`

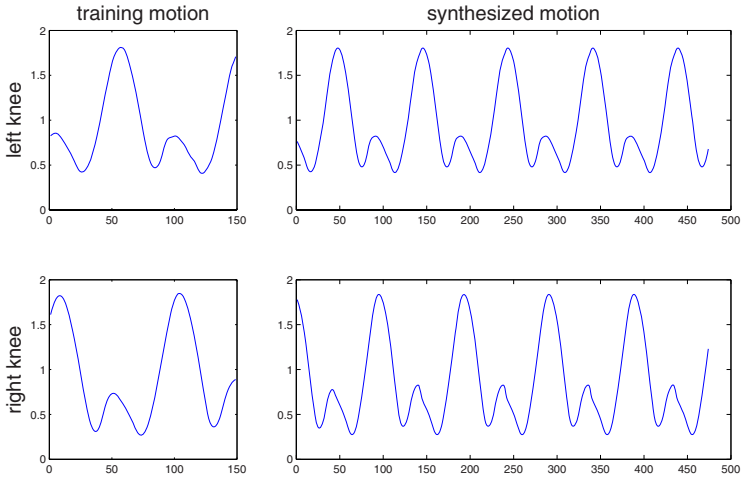


Fig. 2. Synthesis of a cyclic motion from the density estimated from one (or multiple) training motions. Only the left and right knee angles are depicted.

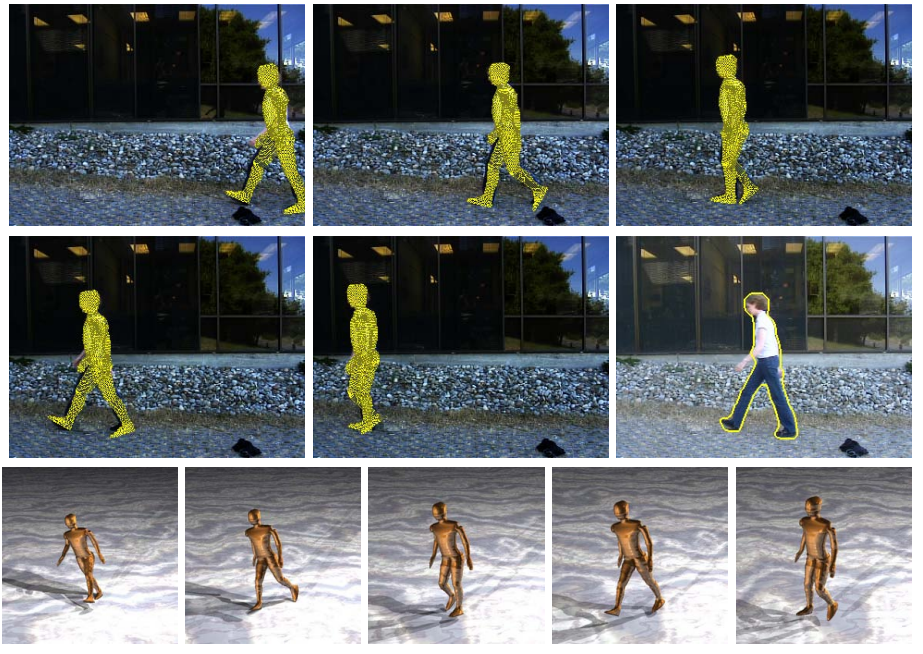


Fig. 3. Tracking result for a monocular sequence, where the density has been estimated with anisotropic kernels. **Center right:** Input image with extracted contour. **Bottom row:** Synthesized view generated from the tracked pose.

Fig. 4. Top row, from left to right: Tracking result for the sequence in Figure 3, but using the isotropic kernel density estimator. Results are not as good as in the anisotropic case. **Rightmost:** Without any motion prior, tracking fails already after a few frames. **Bottom row:** Synthesized view generated from the tracked pose.

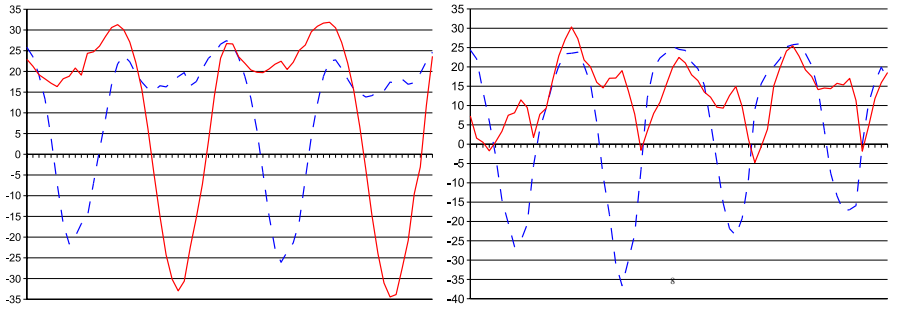


Fig. 5. Tracking curves of the left and right knee. **Left:** Anisotropic kernel density. **Right:** Classic kernel density. The isotropic kernel, due to its weaker predictive power, leads to a hopping motion, as the left and right leg are partially interchanged.

accurate thanks to the prior density estimated from two standard and one exaggerated walking sequences. In contrast, the isotropic kernel estimator in Figure 4 yields results that explain the 2D image data very well, but since the density is less distinct, the estimated pose is not as good as with the anisotropic kernel. As Figure 5 and the video in the supplementary material show, the isotropic kernel density estimator partially mixes up the left and the right leg. The rightmost image in Figure 4 shows that replacing the prior density by some static pose prediction fails completely. Due to the the weak prediction, the gradient descent runs into a suboptimal local minimum and not even consistency with the 2D image data can be ensured.

Partial occlusions are another challenge in human tracking. Figure 6 demonstrates the robustness of the proposed technique in the presence of severely

Table 1. Mean error and standard deviation of knee and elbow joints between tracking results of the jogging sequence in Figure 6 and the outcome of a marker-based tracking system (ground truth)

Setting	mean error	std. dev.
0 boxes	4.01°	±3.3°
20 boxes	5.47°	±4.7°
40 boxes	5.71°	±4.5°
additional samples, 40 boxes	6.13°	±4.9°
walking samples only, 20 boxes	39.58°	±35.3°
isotropic kernel density, 0 boxes	3.64°	±2.4°

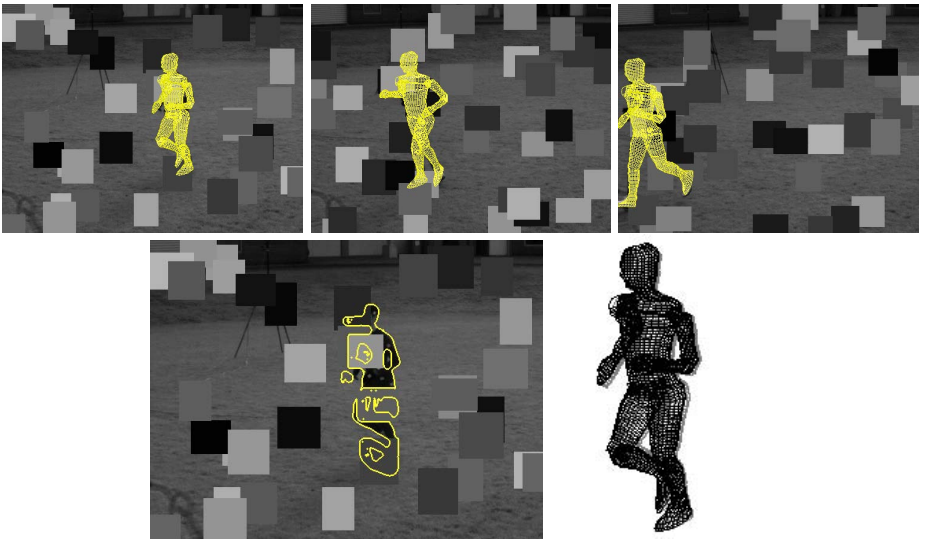


Fig. 6. Tracking of a jogging sequence with 40 occlusions randomly placed in each image. Only one out of four camera views is shown. For the whole sequence, see the supplementary online material. **Top row:** Tracking result. Despite the substantially disturbed image cues (see contour), reasonable poses are computed. **Bottom left:** Contour used for estimating the pose. **Bottom right:** Prediction of the pose in a new frame (black) relative to the previous frame (gray) by means of the prior density.

corrupted image data. 40 occluding boxes have been randomly added to the sequence. In contrast to pixel noise, image data is not only missing, but even misleading, since the occluding boxes create false object boundaries like real occlusions. The contour shown in Figure 6 demonstrates this negative effect on the contour extraction. The prior density estimated from 9 different running and jogging motions, which were subsampled to yield a total of 606 points, keeps the solution close to a jogging motion and, hence, allows for successful tracking.

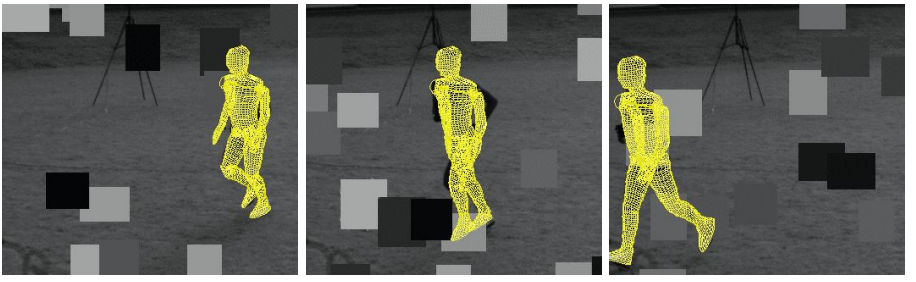


Fig. 7. Tracking of the jogging sequence with 20 occluding boxes and only samples from a walking motion being available for density estimation. The image contains few information on the arms. Hence, the arm pose is hallucinated from the walking prior. The legs, however, are tracked well, despite the unfitting prior.

We also investigated whether the image/prior tandem is able to generalize to sequences where the motion seen in the image does not perfectly fit to any of the prior motions. Figure 7 shows a result where the density has been estimated only from samples of a single walking sequence. Due to poorly constrained image data, the arms reflect the walking motion of the prior. The legs, though, fit well to the jogging motion seen in the image. This shows that the prior can be voted down by clear image data.

For getting a better insight in what happens in the high-dimensional space, Figure 8 depicts on the left the training data consisting of 9 running and jogging motions projected into 2D space via multidimensional scaling. In blue one can see the trajectory of the tracked jogging sequence in this space. Clearly, the pure running prior has a very simple structure. In contrast, the bottom figure shows the situation when additional motions are added to the prior. Learning such priors is a problem for many techniques, especially for typical regressors, which can only model functions. Kernel density estimation, however, handle such situations in a very natural way without any need to adapt the methodology. Hence, tracking the jogging motion in Figure 6 with such more general training data is not a problem.

Table 1 compares several experimental settings of the jogging sequence quantitatively by showing the mean error of the results. Ground truth has been provided by parallel tracking with a marker-based system. Tracking with the more general prior is almost as good as with the special running prior. Interestingly, the isotropic kernel density estimator yields a higher accuracy than the anisotropic density estimator in this sequence. The arm pose in all training patterns does not fit well to the tracked arm motion. Since the anisotropic kernel leads to more concise density estimates, it also tends to a stronger prior. This explains the better result of the isotropic estimator in this case, as we kept the weighting between image and prior data fixed. The large error of the result with the walking prior emerges from the large impact of the wrong elbow angles. For the knee joints alone, one obtains an average error of only 5.29°.

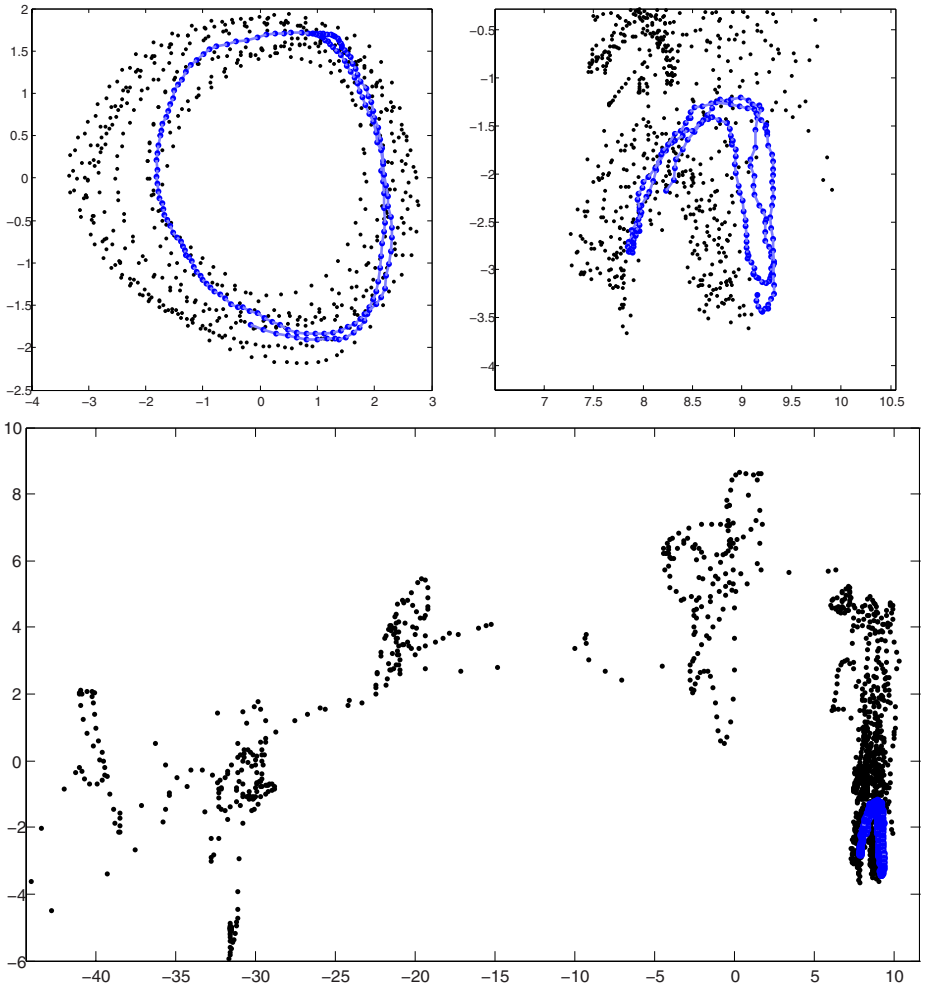


Fig. 8. Training samples (black dots) and tracked pose (blue dots) for the jogging sequence. Points have been projected to 2D via multidimensional scaling (MDS). **Top Left:** Training set consists only of running and jogging motions. **Bottom:** Other motions, such as walking, jumps, leaps, cardwheels, flips, and break dance, have been added to the training set. The sample distribution becomes very irregular in the 2D projection. **Top Right:** Zoom into the part of the more general prior that is relevant for tracking the jogging motion.

Finally, Figure 9 shows a highly dynamic handspring sequence. Without the ability to predict the rough pose at a successive frame, such a motion is hard to track. With the anisotropic kernel density estimate, however, we obtain a rather accurate result.

Fig. 9. Tracking of a handspring with four camera views. See the video in the supplementary material for the whole sequence.

5 Conclusions

We have introduced the use of kernel density estimation as a transparent way to model motion priors in human tracking. In order to cope with the high-dimensional nature of the input space, we proposed density estimation with anisotropic kernels. They are especially appropriate when the density concentrates on a low-dimensional subspace. We suggested a Bayesian tracking framework that makes use of such an anisotropic density estimator by combining the prior density with observation probabilities derived from the image data. A broad experimental evaluation showed the main properties of such a tracking technique. In particular, the prior density is able to support the tracking in case of missing or corrupted image data, even when there are only few, mildly fitting training samples. Moreover, as it is a nonparametric technique, it can easily model multiple motions. Future work will concentrate on appropriate data structures that allow for an efficient sublinear computation of densities from many thousand training samples.

References

1. Akaike, H.: An approximation to the density function. Annals of the Institute of Statistical Mathematics 6, 127–132 (1954)
2. Brox, T., Rosenhahn, B., Kersting, U., Cremers, D.: Nonparametric density estimation for human pose tracking. In: Franke, K., Müller, K.R., Nickolay, B., Schäfer, R. (eds.) Pattern Recognition. LNCS, vol. 4174, pp. 546–555. Springer, Heidelberg (2006)

3. CMU. Carnegie-Mellon Motion Capture Database. http://mocap.cs.cmu.edu
4. Cremers, D., Kohlberger, T., Schnörr, C.: Shape statistics in kernel space for variational image segmentation. Pattern Recognition 36(9), 1929–1943 (2003)
5. Grochow, K., Martin, S.L., Hertzmann, A., Popović, Z.: Style-based inverse kinematics. ACM Transactions on Graphics (Proc. SIGGRAPH) 23, 522–531 (2004)
6. Herda, L., Urtasun, R., Fua, P.: Implicit surface joint limits to constrain video-based motion capture. In: Pajdla, T., Matas, J. (eds.) ECCV 2004. LNCS, vol. 3022, pp. 405–418. Springer, Heidelberg (2004)
7. Howe, N., Leventon, M., Freeman, W.: Bayesian reconstruction of 3D human motion from single-camera video. In: Proc. Neural Information Processing Systems, pp. 820–826 (2000)
8. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Computation 3, 79–87 (1991)
9. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Computer Vision and Image Understanding 104(2), 90–126 (2006)
10. Parzen, E.: On the estimation of a probability density function and the mode. Annals of Mathematical Statistics 33, 1065–1076 (1962)
11. Rosales, R., Sclaroff, S.: Learning body pose via specialized maps. In: Proc. Neural Information Processing Systems (2001)
12. Rosenblatt, F.: Remarks on some nonparametric estimates of a density function. Annals of Mathematical Statistics 27, 832–837 (1956)
13. Rosenhahn, B., Brox, T., Weickert, J.: Three-dimensional shape knowledge for joint image segmentation and pose tracking. International Journal of Computer Vision 73(3), 243–262 (2007)
14. Sain, S.R.: Multivariate locally adaptive density estimation. Computational Statistics & Data Analysis 39(2), 165–186 (2002)
15. Scott, D.: Multivariate Density Estimation. Wiley, Chichester (1992)
16. Sidenbladh, H., Black, M.J., Sigal, L.: Implicit probabilistic models of human motion for synthesis and tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 784–800. Springer, Heidelberg (2002)
17. Sminchisescu, C., Jepson, A.: Generative modeling for continuous non-linearly embedded visual inference. In: Proc.International Conference on Machine Learning (2004)
18. Sminchisescu, C., Kanaujia, A., Metaxas, D.: Learning joint top-down and bottom-up processes for 3D visual inference. In: Proc. International Conference on Computer Vision and Pattern Recognition, pp. 1743–1752 (2006)
19. Sminchisescu, C., Triggs, B.: Estimating articulated human motion with covariance scaled sampling. International Journal of Robotics Research 22(6), 371–391 (2003)
20. Urtasun, R., Fleet, D.J., Fua, P.: 3D people tracking with Gaussian process dynamical models. In: Proc. International Conference on Computer Vision and Pattern Recognition, pp. 238–245. IEEE Computer Society Press, Los Alamitos (2006)
21. Vincent, P., Bengio, Y.: Manifold parzen windows. In: Proc. Neural Information Processing Systems, vol. 15, pp. 825–832 (2003)

Multi Person Tracking Within Crowded Scenes

Andrew Gilbert and Richard Bowden

University of Surrey, Guildford, Surrey, GU2 7XH, UK
{a.gilbert,r.bowden}@Surrey.ac.uk

Abstract. This paper presents a solution to the problem of tracking people within crowded scenes. The aim is to maintain individual object identity through a crowded scene which contains complex interactions and heavy occlusions of people. Our approach uses the strengths of two separate methods; a global object detector and a localised frame by frame tracker. A temporal relationship model of torso detections built during low activity period, is used to further disambiguate during periods of high activity. A single camera with no calibration and no environmental information is used. Results are compared to a standard tracking method and groundtruth. Two video sequences containing interactions, overlaps and occlusions between people are used to demonstrate our approach. The results show that our technique performs better that a standard tracking method and can cope with challenging occlusions and crowd interactions.

1 Introduction

Visual surveillance systems are commonly placed in large areas of high traffic such as in airports, rail stations and shopping centres. Tracking individuals within crowds remains a difficult problem due to the complex interactions and occlusions that occur. This paper presents an approach to tracking individuals and retaining object identity through occlusions and object interactions, within a single camera. Most existing methods of tracking individuals, in the area of visual surveillance, involves the segmentation of foreground objects from a modelled background. These methods often fail when tracking an individual in a crowded scene since individuals cannot be easily segmented in isolation from the background. There are two categories of techniques to aid this problem; frame-by-frame trackers are highly accurate for scenes with little or no occlusion. While object detectors work well at recognizing specific objects in individual frames. Therefore the method proposed within this paper is designed to use both and take advantage of the strengths of each.

There have been many possible solutions presented with regard to the problem of tracking multiple objects, which can categorised as either single or multiple camera approaches.

1.1 Multiple Camera Tracking

The use of multiple wide baseline cameras allows simpler occlusion reasoning and can allow for a 3D environment to be built of the scene through camera

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 166–179, 2007.

calibration. This area has seen significant research and some success due to the easier conditions. However this paper uses a single camera, therefore a couple of significant papers are presented here. Fleuret *et al* [1] uses the information from multiple cameras to produce a probabilistic occupancy map based on the dynamics and appearance models of the objects. Dynamic programming is then used to track the multiple objects through significant occlusion and lighting changes. Khan *et al* [2] use overlapping cameras to learn when objects should be visible, to build up relationships between cameras. This allows the system to predict the new location of an object that leaves a camera's field of view due to occlusion.

1.2 Single Camera Tracking

Although very promising results have been presented through the use of multiple cameras, there are large practical restrictions on having multiple cameras covering large scale installations due to cost. In addition, a single camera allows for simple and easy deployment. Therefore, a number of algorithms have been proposed to track objects on a single camera. A blob-based method like that proposed by Isard and MacCormick [3] is one solution. This uses a background segmentation and appearance model within a Particle Filter framework to track an unknown number of people. Linking blobs together and learning their relationships is used by Bose *et al* [4] to track multiple interacting objects. Particle Filters are also used by Okuma *et al* [5] in conjunction with a boosted detector to help remove false particles in the filter to track fast moving ice hockey players. The dynamics of objects can be used with a Kalman Filter [6] to track objects that have near constant velocity, as proposed by Xu *et al* [7] within a football context or to track vehicles as presented by Boykov and Huttenlocher [8].

Through the use of appearance, the accuracy of tracking can be significantly increased. Giebel *et al* [9] uses shape, texture, and depth information from the image within a Particle Filter Bayesian framework, to track using learnt spatio-temporal object shapes. Comaniciu *et al* [10] uses a metric derived from the Bhattacharyya coefficient as an appearance similarly measure and optimises the tracking with a Mean Shift procedure. This copes with low resolution and partial occlusion, and can be extended with a Kalman Filter to use the velocity of the objects to improve tracking. Our approach fuses segmentation, object detection and object appearance together using dynamic programming to overcome periods of uncertainly during occlusion. Furthermore we demonstrate how priors on the detector can be built during periods of low activity and used to further improve tracking and occlusion handling. Nillius *et al* [11] built a graph of all interactions between football players for a 10 minute sequence. Then they use Bayesian Inference based on the expected locations of players on the pitch to join up and separate areas of interactions into separate paths through the graph.

This paper is organized as follows. Section 2 gives an overview of the system, section 3 explains the object detector and how it was trained and how priors on detection formed. How the object tracks are formed is in section 4 and how they are combined together is presented in section 5. With results and a conclusion presented in sections 6 and 7 respectively.

2 Experiment System Overview

Following the recommendations by Perez *et al* [12] the colour space used to represent object appearance is Hue-Saturation-Value (HSV). Since HSV introduces some degree of illumination invariance, it is well suited to cope with the lighting changes that occur in non uniformly lit areas.

First a prior of the reappearance locations over time of torso detections is learnt. The torso detector is then applied to every frame to provide "seed" positions of visible individuals. Each seed is represented as an appearance model that is tracked over consecutive frames using a Mean Shift [10] tracker. Tracking stops when there is a marked difference in the current object's appearance to that of the appearance of the original seed. This is repeated for all objects detected in each frame, resulting in sequences of short tracklets. The optimum path through these tracklets is found through a Viterbi style dynamic programming algorithm to find the best trajectory through the scene individuals. Where a model based on the spatial reappearance of detections over time is used to constrain the trajectory.

3 Torso Detector

In a crowded scene with overlapping people, traditional techniques such as a blob-based background segmentation cannot be used. In addition, while people are overlapping and interacting, much of the body outline is occluded. However, with the camera positioned above head height as is often the case, the head and shoulders are often visible, even in a crowd. Therefore, a torso detector is used to produce the seed locations of objects to be tracked. The detector is based on the one presented by Mikolajczyk *et al* [13]. The recognition technique is based on a codebook representation where appearance clusters built from edge based features are shared among several object classes. The features of the image, are arranged in an efficient tree type hierarchical design. Where the basic edge based features can then be shared by several object classes. By clustering the features they are used to represent an object class, which in this case is a human torso. The use of a hierarchical clustering detection allows for minor occlusion of edge features of the torso. A detector trained to detect faces would be unsuitable as often faces are not looking directly at the camera, and thus would cause a failure. Likewise a full body detector would be unsuitable as often parts of the body are heavily occluded. An example frame from the sequence of where a face or full body detector would fail is shown in figure 2 where people are facing away from the camera and have part of their body shape occluded.

In order to train the detector, 1200 torso examples were used, The positive training set contained examples of people looking in arbitrary directions. The negative was made up of non people images sauch as landscapes and buildings. Figure 1 shows the response by the detector to the video sequence used. A threshold was chosen at 400 false positives. This is far higher than is usual in the object recognition and classification but allows for a greater number of detections for the *harder* objects such as people walking away from the camera

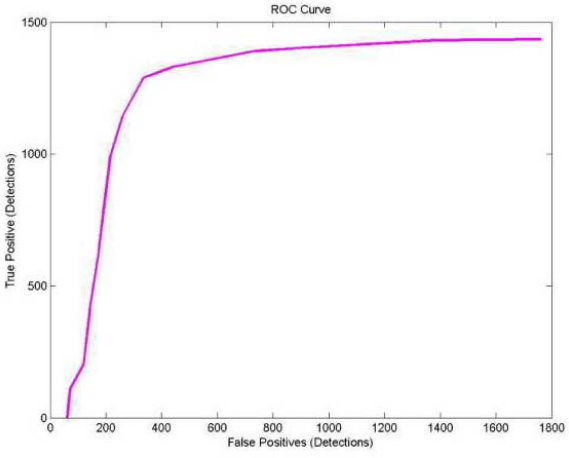


Fig. 1. This shows an ROC curve for the Torso detector used with the sequence of a crowded scene

Fig. 2. This shows a frame in a sequence showing the main Torso detections. Note there are both true positive and false positive detections.

or who have part of their outline occluded. This is acceptable as the false positive detections are discarded as outliers to the motion and appearance models and are ignored. Each detection has a rectangular kernel that is centred on the torso of a person with a bounding box. The height of the kernel is then used as a weak heuristic to estimate the total size of the person to cover their whole body, and the kernel's height size is adjusted accordingly.

3.1 Learning Torso Detection Relationships

Over time, people follow similar routes and speeds on specific cameras, this can be used to build a predictive motion model. Using the torso detections, a

relationship between the location and time of the reappearance of future torso detections can be formulated. This can be used to predict future torso positions and therefore reduce false positive detections in a video sequence and is used with the predicted Mean Shift tracker locations in section 4 to reinforce strong possible tracks. The spatial and temporal relationship between torso detections is learnt during periods of low activity when detection is at its most reliable. The x and y distance difference between detections over time is modelled, together with the frame difference between the detections. An eight hour sequence of video is used to learn the model, each detection is compared to previous detections with respect to their reappearance period and x, y difference to the original detection. A unit increment with a in-Parzen window blur is centred on the x and y difference between the two detections at the specific period time. For a set of J torso detections $D_i \epsilon \{D_1, ...D_J\}$ Equation 1 shows the frequency of bins in the histogram representing the reappearance of torso detections.

$$f = \sum_{j=1}^{J} \sum_{i=1}^{J} G((x_j - x_i, y_j - y_i, t_j - t_i), I\sigma^2) \quad where \quad 0 \leq (t_j - t_i) < T \quad (1)$$

Where i is the current torso detection and j is all the previous detections. $x_j - x_i$ and $y_j - y_i$ are the difference in position in both x and y respectively between detections j and i. t is the time of the detection, with T set as a maximum reappearance period to limit the temporal length of the prior, this is commonly 100 frames. $G(\overline{V} \epsilon \Re^3, I\sigma^2)$ represents a 3D gaussian kernel positioned at V with spherical co-variance σ^2. This is repeated for all detections in the sequence, to allow a single torso detection to be modelled over T frames without any new information. Given a torso detection D_i its prior reappearance probability over time can be modelled by equation 2

$$p(D_{tj}|D_{ti}) = \frac{f_t^{i|j}}{J^2} \quad (2)$$

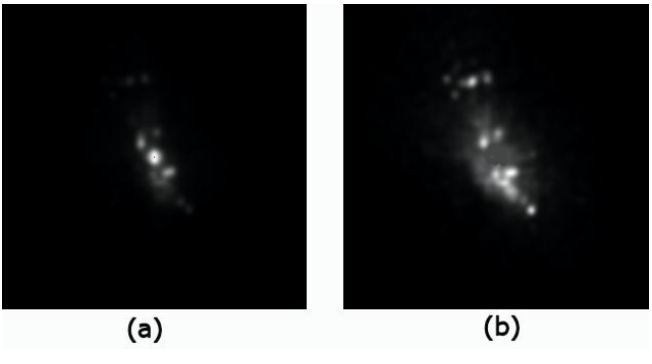


Fig. 3. Figure (a) shows the conditional probability of a torso detection given a detection that occurred 5 frames ago in the centre. Figure (b) shows the conational probability of a new torso given a detection occurred 100 frames ago in the centre.

Figure 3 shows the reappearance probability of a detection after (a) $T = 5$ frames and (b) $T = 100$ frames. Notice how this encodes domain knowledge for the camera, as the slight diagonal trend corresponds to the off centre placement of the camera and therefore the trend for people to move diagonally. By 100 frames there is a more dispersed intensity area, this is due to the increased uncertainty of predicting further into the future.

4 Forming Object Tracks

For each torso detection a tracker is initialised to form a short tracklet. Short tracklets are used, as within crowded scenes frame-by-frame tracker will often fail due to heavy occlusions. The tracker is terminated when it is determined that it is no longer tracking the original object. The tracking system takes all the torso detection locations, with the aim of extending their trajectory further through the sequence. The area within the object's Kernel is used to form an appearance model(a colour histogram), $K*$. The objects change in movement over the sequence is tracked over time using a Mean Shift tracker with Kalman Filter prediction. A Kalman Filter is used in conjunction with the Mean Shift with the assumption that generally people walk with a constant velocity. This allows dynamics to be introduced to the Mean Shift which otherwise only optimises tracking based upon a local moment of a colour distribution. A constant-velocity model with white noise drift is assumed for the Kalman Filter. The Kalman Filter predicts for each new frame the initial Mean Shift search region. This is optimised through iterations of the Mean Shift procedure until convergence and the Kalman Filter is then updated with this converged measurement, and the process repeated on the following frame. This is continued until some termination criterion is met.

4.1 Track Termination Criteria

Although the Mean Shift with velocity prediction will cope with minor occlusions or lighting changes, large scale appearance changes to the kernel of the tracked object can cause Mean Shift to fail. Therefore a termination criteria is applied, to determine when the Mean Shift kernel has failed and is no longer tracking the original object. If multiple short but significant tracks of the same object are found, these can then be combined together to produce the full trajectory. The termination criteria is based on a learnt likelihood ratio. Every 25 frames the reference appearance model for the Mean Shift track $K*$ is updated, subsequent frames are then compared to this appearance model, with a measure of similarity. The similarity measure is the likelihood ratio of the image within the track's kernel K_t at frame t being *more* similar to the appearance model of the track, $K*$, than *not* similar to the appearance model, this is shown in equation 3.

$$L(K_t|K*) = \frac{P(K_t|K*)}{P(K_t|\overline{K*})} \tag{3}$$

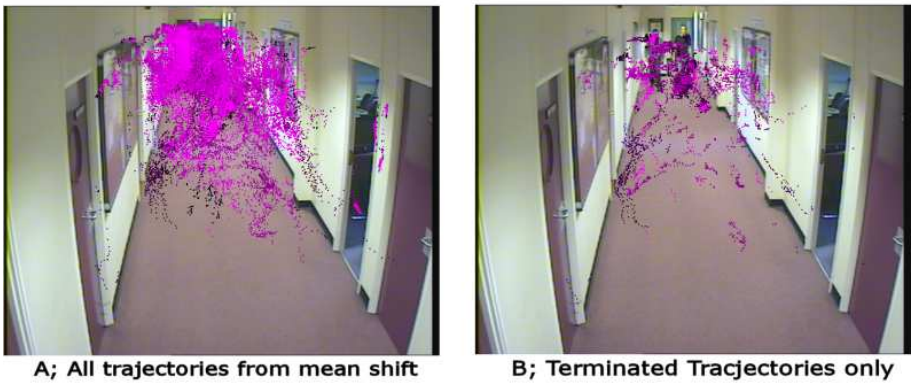


Fig. 4. A, shows all the trajectories from the seeded mean shift tracker of a crowded video sequence, B shows the trajectories, where the tracks have been terminated using the termination criteria in section 4

where $P(K_t|K*)$ is the probability of the kernel K_t being the appearance model for the track. This is found through a cumulative probability, shown in equation 4

$$P(K_t|K*) = \sum_{i=0}^{\delta(K_t,K*)} F_i \tag{4}$$

Where $\delta(K_t, K*)$ is the Bhattacharyya similarity coefficient between K_t and $K*$ and is shown in equation 5 and F_i^K is the normalised frequency histogram of the Bhattacharyya coefficient over a number of ground-truthed object sequences.

$$\delta(K_t, K*) = \sum_{u=1}^{m} \sqrt{K_u * K*} \tag{5}$$

The threshold of 0.5 is applied to the likelihood ratio as this allows tracks to continue until the track is more likely to *not* be the reference model. At this point the trajectory track is terminated, forming a short tracklet, $\{S_{t\ start}, ..., S_{t\ end}\}$ of that person. Where S_t is the state at time t represented by the colour appearance model (colour histogram) and motion model (Kalman filter state). $P(K_t|\overline{K*})$ is compute is a similar method, though negative groundtruth examples are used to form the frequency histogram F_i in equation 4. Figure 4 shows how the tracks are significantly reduced once the termination criteria is applied.

4.2 Kernel Scale Adaption

Within the Mean Shift kernel tracking proposed by Comaniciu [10], there is a simple scale adaption technique that adjusts the kernel size based maximising the Bhattacharyya coefficient for different kernel sizes. This adapts the scale of the kernel well for sequences with little or no occlusions, although as the

occlusion level increases the overall tracking performance drops. This is because the optimised scale doesn't take into account that part of the person could be occluded, and will therefore only optimise on the visible part of the person. This can cause misleading and incorrect scale adaption. Within this work there is no need for an adaptive scale within the Mean Shift tracker as the torso detector is used over many scales and the short tracks, therefore as the person increases in size the scale in the torso detection will increase this ensures the trajectories bounding box will adapt in size as required. This means that the overall trajectory of an object has a linear approximation to scale changes when the short tracklets are recombined in section 5.

5 Combining Tracklets into Trajectories

The Mean Shift kernel tracker produces short tracklets for each person visible within the video sequence. Each person will have multiple tracklets at any one time, and the aim of this section is to find the optimum trajectory through the complete sequence. Each tracklet $S_\tau = \{S_{t\ start}, ..., S_{t\ end}\}$ form a partial row in the state matrix S between the time indexes t *start* and t *end*. The objective is to find the optimal path through this state matrix that maximises the likelihood of the trajectory for the object S_{REF}. This is found using three similarity measures and the learnt detection reappearance model from section 3.1. The appearance similarity between the State appearance and the Reference appearance model is found using the Bhattacharyya similarity coefficient 5.

$$L_{Ref}(S(t)|S(REF)) = \delta(S_t, S_{REF}) \tag{6}$$

This provides a constraint ensuring the trajectory will stay visually similar to the original person's reference image. Between frames t and $t+1$ the appearance similarity between the states is computed using the Bhattacharyya similarity coefficient 5.

$$L_{app}(S(t)|S(t-1)) = \delta(S_t, S_{t-1}) \tag{7}$$

A motion model is computed for each state trajectory adding a non appearance based constraint.

$$L_{KF}(S(t)|S(t-1)) = MH(S_t, S_{t-1}) \tag{8}$$

The Kalman Filter [6] in the Mean Shift kernel tracking in section 4 computes predicted positions $S_\tau, t\ end + n$ allowing each tracklet to be artificially extended. n is typically set to 4 seconds, i.e. 100 frames. The Mahalanobis similarity measure MH is used to find the difference in position between the predicted state positions and current state positions. The detection reappearance model $P(D_{tj}|D_{ti})$ from equation 2 is used as a prior to constrain the other similarity measures. The likelihood of the state at frame t is computed from the path in frames t to $t-T$.

$$L_{loc}(S(t)|S(t-1), ..., S(t-T)) = \prod_{i=1}^{T} P(D_t|D_{t-i}) \tag{9}$$

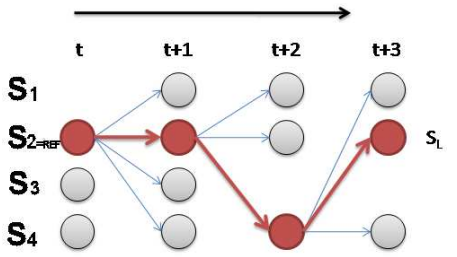


Fig. 5. A visualisation of the Trellis diagram for find the lowest cost path through the video sequence

Dynamic programming is then used to select the optimal path that maximises the objective function.

$$\Phi(l) = \max_{\tau}\{\underbrace{L_{Ref}(S_\tau(t)|S(REF))}_{Ref.\ Appear.}\underbrace{L_{app}(S_\tau(t)|S_\tau(t-1)}_{State\ Appear}$$
$$\underbrace{L_{KF}(S_\tau(t)|S_\tau(t-1))}_{Motion}\underbrace{L_{loc}(S(t)|S(t-1),...,S(t-T))}_{Learnt\ prior}\}\Phi_{t-1}(\tau) \tag{10}$$

Where there are τ possible states for a frame. This can be visualised as a trellis diagram, as shown in figure 5. There are 4 state tracks shown in figure 5, with the S_{REF} selected as the starting image of the person to track. Then at each time interval t, all possible paths from the current state are examined and the cost is maximised to find the next state based on appearance and motion models. The recursive algorithm in equation 10 maximises a single best path. Therefore should two or more trajectory states maximise to the same destination state, the state transition with the highest livelihood will use that destination state. The remaining trajectories will reevaluate the the remaining states, and repeat the process of the maximising the state transitions. This allows multiple trajectories within the state transition trellis design while achieving multiple near maximum best paths through it.

6 Experiments

This section shows the tracking results on people walking and interacting in an enclosed wide indoor corridor. A single surveillance style PAL resolution camera is used to capture two video sequences at 25fps. A conventional low cost camera is typically used, with a poor colour response and higher pixel noise levels. A number of benchmarked data sets including the CAVIAR [14] and PETS [15] video sequences were examined to test this technique on. However it was found these were not challenging enough as they often don't contain crowded scenes. Therefore two new sequences were groundtruthed and used, these are available on the internet [16]. The first sequence consists of a five of

people walking towards and away from the camera. There is overlap between the individuals and complete occlusions occur within the sequence. The second video is similar but many of the seven individuals in the sequence stop mid way through the sequence and there are a greater number of interactions and occlusions. This is designed to show the limitations of motion based trackers such as the Kalman Filter, and appearance based trackers such as Mean Shift. Both sequences are very challenging due to a number of reasons. The lighting used is non uniform, this provides a challenge for the mean shift tracker and torso detector. There is a large field of view causing a large scale variation as people move along the corridor, and in addition the camera is mounted only just above head height causing many occlusions.

Our approach is compared with the Mean Shift algorithm [10]. The Results are presented both qualitatively, with bounding boxes indicating the people tracked and quantitatively. To assess performance the Euclidean distance error between the centre of the groundtruth's bounding box and that of the computed trajectory is calculated. The percentage of overlap between the groundtruth's and computed trajectories bounding box gives an indication of correct scaling and accuracy of tracking individuals.

Figure 6a shows the euclidian error for Mean Shift and our tracklet tracker with and without the torso prior. All techniques have similar initial tracking errors. Then the main area of occlusion occurs as indicated by the shaded area on Figure 6. After this the euclidian error for the Mean Shift tracker and tracklet tracker without prior increase. While the tracklet tracker with prior is consistent with error around an average of 40 pixels. This is a relatively low mean error considering the sequences are on full PAL 720x560 resolution. Figure 6b shows the mean percentage overlap, where again the mean shift and the tracklet tracker

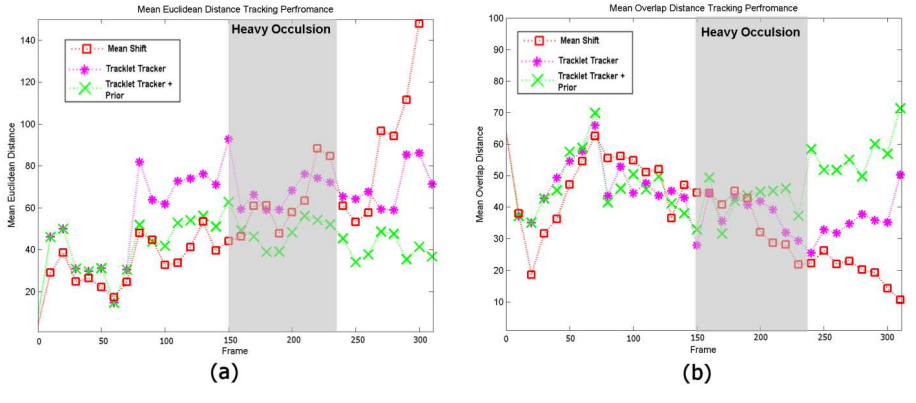


Fig. 6. Video 1: Figure (a) shows the mean Euclidean distance difference per frame for our approach with the learnt prior and without and also Mean Shift (less is better). Figure (b) shows the mean overlap per frame for our proposed approaches with and without the prior and Mean Shift (more is better).

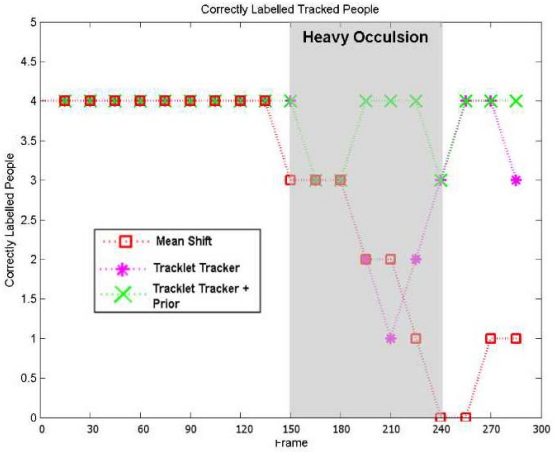


Fig. 7. Video 1: A comparison between how well three techniques correctly label tracked people through the sequence

without the prior decrease in value after the heavy occlusion has occurred. While our tracklet tracker with prior has the highest overlap percentage at the end of the seqeunce.A graph of the number of people correctly labelled through the sequence is shown in figure 7. It can been seen that initially all methods track well, however after the heavy occlusion the mean shift fails dramatically and only tracks one person correctly by the end of the sequence. This is confirmed by figure 8, here a visual comparison of 4 frames in the same video sequence is presented. Looking at frame 298 the Mean Shift has correctly labelled only one person, while our tracklet tracker with a torso prior labelled all 4 correctly. The long 200 frame occlusion of the person who is in the bottom left of frame 2485 in figure 8, makes it impossible to track them through the whole sequence. The results for video 2 are shown in figure 9 shows the mean per frame Euclidean and percentage overlap. The graphs are similar to those for video 1; our tracklet tracker and Mean Shift work well before the occlusion, however after the occlusion the Mean Shift errors increase and starts to fail. While our approach of the tracklet tracker with prior keeps tracking correctly with controlled pixel error. However at the start of the sequence our tracklet tracker error is higher than Mean Shift. This is due to the limitation of the current torso tracker. For challenging people that are far from the camera facing away from the camera, the torso detector has a lower true positive rate. This means there are less tracklets to form the trajectory of the chosen people and the best path trajectory doesn't exist in the states. In addition our approach works by linking short but significant tracklets to form a single trajectory through the sequence. However if some of the people are continually occluded for very long periods through the sequence, there will be no significant tracklet to link together and this will cause the tracking to fail.

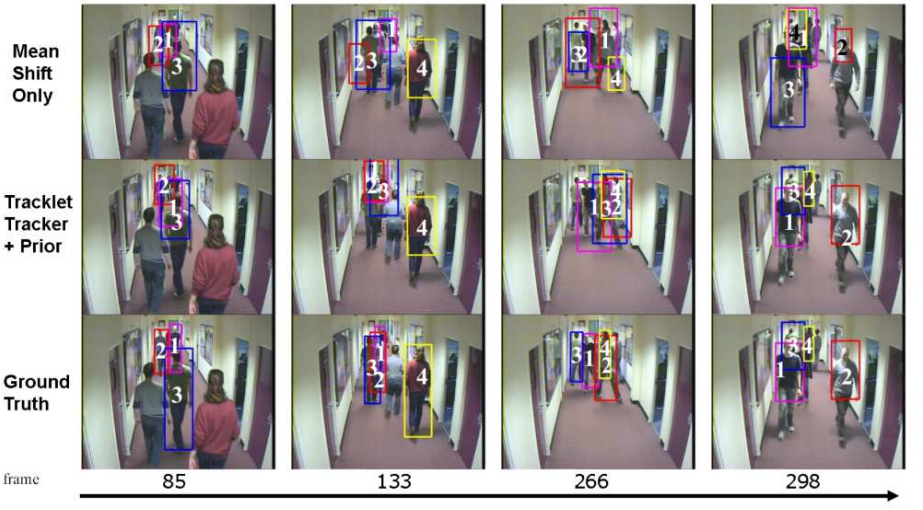


Fig. 8. Video 1: A comparison between our technique and a Mean Shift Tracker and the groundtruth over 4 frames

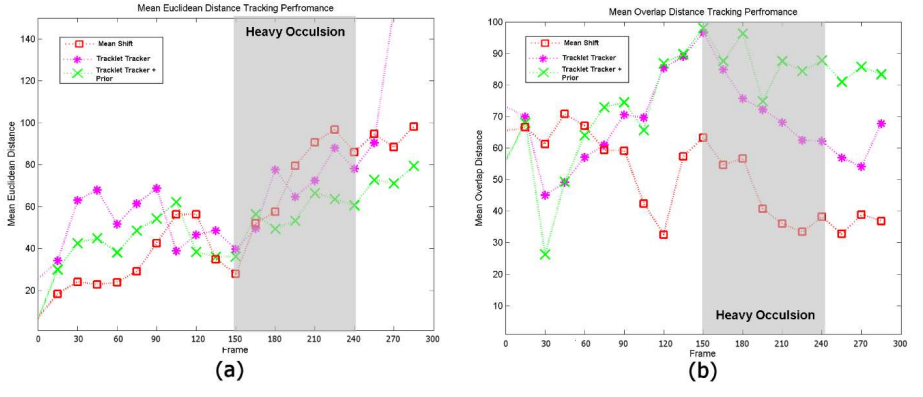


Fig. 9. Video 2: A comparison of the mean Euclidean distance difference (figure (a) , less is better) and percentage overlap (figure (b) more is better) between our approaches and a Mean shift tracker

7 Conclusion

We have described an approach to combine the strengths of a torso detector to locate people, with those of a Mean Shift tracker for frame to frame correspondence. A model of the spatial reappearance of the torso detector over time has been used to model the motion of people. The Mean Shift tracker produces short tracklets which are recombined using a Viterbi style approach to produce a full

trajectory through the video. This approach leads to an increased robustness to occlusions and interactions between people in a crowded scene.

We have tested this on two difficult groundtruthed sequences of people interacting and occluding within a large corridor. Our results are consistently better than that of a Mean shift tracker which fails to cope with heavy occlusion. These promising results are possible using a single uncalibrated low cost surveillance type camera. While in the future results could be further improved through enhancing the detection of people facing away from the camera. Also by addressing the issue of people who are heavily occluded for long periods.

Acknowledgements

This work is funded under the EU FP6 Project "URUS", IST-045062.

References

1. Fleuret, F., Berclaz, J., Lengagne, R., Fua, P.: Multi-Camera People Tracking with a Probabilistic Occupancy Map. IEEE Transactions on Pattern Analysis and Machine Intelligence (to be published, 2007)
2. Khan, S., Javed, O., Shah, M.: Tracking in Uncalibrated Cameras with Overlapping Fields of View. In: 2nd IEEE Workshop on Performance Evaluation of Tracking and Surveillance, IEEE Computer Society Press, Los Alamitos (2001)
3. Isard, M., MacCormick, J.: Brammble: A Bayesian Multiple Blob Tracker. In: Proc. IEEE International Conference on Computer Vision and Pattern Reconition, vol. 2, pp. 34–41 (2001)
4. Bose, B., Wang, X., Grimson, E.: Detecting and Tracking Multiple Interacting Objects Without Class-Specifc Models. Technical report MIT-CSAIL-TR-2006-027 Massachusetts Institute of Technology (2006)
5. Okuma, K., Taleghani, A., DeFreitas, N., Little, J., Lowe, D.: A Boosted Particle Filter. In: Proc. European Conference on Computer Vision (2004)
6. Welch, G., Bishop, G.: An Introduction to the Kalman Kilter. Technical Report 95-041, University of North Carolina at Chapel Hill (1995)
7. Xu, M., Orwell, J., Jones, G.: Tracking Football Players with Multiple Cameras. In: IEEE International Conference on Image Processing, IEEE Computer Society Press, Los Alamitos (2004)
8. Boykov, Y., Huttenlocher, D.: Adaptive Bayesian Recognition in Tracking Rigid Objects. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, pp. 697–704. IEEE Computer Society Press, Los Alamitos (2000)
9. Giebel, J., Gavrilla, D., Schnorr, C.: A Bayesian Framework for Multi-cue 3d Object Tracking. In: Proc. European Conference on Computer Vision (2004)
10. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-Based Object Tracking. IEEE Transactions Pattern Analysis and Machine Intelligence 25, 564–577 (2003)
11. Nillius, P., Sullivan, J., Carlsson, S.: Multi-Target Tracking - Linking Identities using Bayesian Netweork Inference. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2006)

12. Perez, P., Hue, C., Vermaak, J., Gannet, M.: Color-Based Probabilistic Tracking. In: Proc. European Conference on Computer Vision (2002)
13. Mikolajczyk, K., Leibe, B., Schiele, B.: Multiple Object Class Detection with a Generative Model. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2006)
14. CAVIAR: Context Aware Vision using Image-based Active Recognition, EC project/ist 2001 37540, http://homepages.inf.ed.ac.uk/rbf/CAVIAR/
15. PETS: Performance Evaluation of Tracking and Surveillance, http://www.cvgcs.rdg.ac.uk/slides/pets.html
16. Crowded groundtruthed video sequnces, http://personal.ee.surrey.ac.uk/Personal/a.gilbert

Joint Appearance and Deformable Shape for Nonparametric Segmentation

S. Boltz, É. Debreuve, and M. Barlaud

Laboratoire I3S, Université de Nice-Sophia Antipolis, CNRS
2000 route des Lucioles, 06903 Sophia Antipolis, France

Abstract. This paper deals with region-of-interest (ROI) segmentation in video sequences. The goal is to determine in one frame the region which best matches, in terms of a similarity measure, a ROI defined in a reference frame. A similarity measure can combine color histograms and geometry information into a joint PDF. Geometric information are basically interior region coordinates. We propose a system of shape coordinates constant under shape deformations. High-dimensional color-geometry PDF estimation being a difficult problem, measures based on these PDF distances may lead to an incorrect match. Instead, we use an estimator for Kullback-Leibler divergence efficient for high dimensional PDFs. The distance is expressed from the samples using the kth-nearest neighbor framework (kNN). We plugged this distance into active contour framework using shape derivative. Segmentation results on both rigid and articulated objects showed promising results.

1 Introduction

Region-of-interest (ROI) segmentation is to determine in a frame the region which best matches, in terms of a similarity measure, an ROI (user-) defined in a reference frame. A similarity measure is a distance between two data sets, the reference data set and a candidate, or target, data set. In the discrete framework, each data set is composed of one data vector per pixel of the region. The similarity measure does not necessarily make use of a one-to-one match between the pixels of the regions.

Two aspects of similarity measures between the reference region and a target region can be distinguished: radiometry, which indicates if the regions have similar color distributions, and geometry, which correlates where these colors are present in each region. Similarity measures based solely on radiometry include distances between color histograms or probability density functions (PDF), for instance, mutual information [1], Kullback distance [2]. Not accounting for the information of where a given color was present in the region allows to be more flexible regarding the geometric transformation between the reference region and the target region. However, it increases the number of potential matches and then the risk for the tracking to fail after a few frames. This can be avoided by using a geometry-aware similarity measure. The absence of geometric information implies that several candidate regions can appear as good matches.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 180–195, 2007.

As an alternative, geometry can be added by means of a motion model used to compute the point-wise residual between reference and candidate regions. A function of the residual can serve as a similarity measure, classically, the sum of squared differences (SSD), functions used in robust estimation [3] such as the sum of absolute differences (SAD), or statistical measures. An example where the energy is defined on a first order approximation of the point-wise residual, the optical flow constraint, in segmentation is [4]. However, in presence of complex motions or homogeneous zones, the residual term loose its efficiency. The geometric constraint can be soften, *e.g.*, by combining an energy based on a color distribution and an other based on optical flow [5]. An alternative is to add a shape prior to the energy [6,7].

Another approach defines a joint geometric/radiometric PDF, for example in bounding box tracking [8]. Geometric data add a spatial location information to the radiometric PDF. In bounding box tracking [8], the choice of good spatial coordinates is straightforward, for instance Cartesian coordinates of the box. However, defining spatial coordinates for deformable regions is an issue. One can think on defining a coordinate system embedded in the region, shape coordinates. Shape coordinates can be canonical Cartesian coordinates or polar coordinates. The latter would better handle rotations of some parts of the shape. The polar coordinates mapping of regions is a simplified version of the recent results for shape recognition in [9]. However, the last two shapes coordinate change when the region goes under deformations. A third shape coordinate based on the distance map, already applied in medical imaging [10], can be considered. This shape coordinates suit very well for medical imaging as the intensity is constant over the level sets of the regions. To handle non medical videos, one can extend it adding a nearest contour coordinate, the arc length of the nearest point (NAL). This approach is motivated by the fact most of the shape coordinates and shape correspondences algorithms in the literature [11] are based on correspondence of the contour. By combining both distance map and contour correspondence with NAL, we extend this model to region correspondence. These region coordinates are constant under many object deformations.

Shape coordinates are added to radiometric data in a single joint high-dimensional PDFs. Although there are efficient methods [12] to estimate multivariate PDFs using Parzen windowing, limitations appear as the dimension of the domain of definition of the PDFs increases. This is described in [12] as the curse of dimensionality: as the dimension of the data space increases, the space sampling gets sparser. Dilating the Parzen window is not a satisfying solution since it implies an over-smoothing of the PDFs. Consequently, PDF-based similarity measures might not be estimated accurately enough for segmentation. To overcome this high-dimension problem, a PDF estimator based on a k-th nearest neighbor (kNN) search was proposed [13,14] and used to define a consistent entropy estimator [15]. This PDF estimator is not differentiable, and it is an issue in numerous segmentations algorithms. However the shape derivative tool circumvents this issue [16].

The first contribution of this paper is to apply in segmentation a joint radiometric/geometric, color-spatial [8]. We propose as geometric data, shape coordinates, adapted for deformable regions. Recent segmentation methods also tried to combine multiple features (spatial data, gradient, wavelets coefficients, motion) to perform accurate segmentation [5]. The PDFs are then high-dimensionals and some assumptions have to be made (*e.g.*: independence between components, gaussian assumptions). The second contribution of this paper is to compute the Kullback-Leibler divergence, between high-dimensional PDFs using the kNN framework. This new estimator is efficient for high-dimensional PDFs [15,14] with very weak assumptions on the PDFs. The third contribution of this paper is to plug these methods in an active contour framework using the shape derivative tool [16]. This high dimensional Kullback distance is not differentiable, however using the shape derivative, no direct differentiation of the Kullback distance is needed and we can bypass this difficulty.

In this paper we use the high dimensional statistical measure estimation based on the kNN framework proposed in [17]. This study in tracking did not require PDF estimation. Here we build the kNN framework using both kNN PDF estimation and statistical measure estimation. Indeed, in this paper we compute the shape derivative of the criterion proposed in [17] and this derivative requires the underlying PDF estimation of the statistical measure. Moreover the work [17] was presented on rigid shapes (rectangles) with Cartesian coordinates. This paper takes into account deformable shapes and a new system of coordinates had then to be defined.

This paper is organized as follows. Section 2 defines Kullback distance on geometric/radiometric data. Section 3 defines geometric data for deformable shapes. In Section 4 we plug this distance in a segmentation method through active contours. Section 5 presents the kNN approach and the kNN-based expression of the Kullback-Leibler distance. Section 6 provide some results of segmentation performed on two standard sequences. Finally Section 7 concludes and gives some perspectives.

2 Similarity Measures with a Soft Geometric Constraint

Let I_{ref} and I_{target} be, respectively, the reference frame in which the ROI Ω_R is (user-) defined and the candidate, or target, frame in which the region Ω_T best matches the ROI, in terms of a given similarity measure, is to be searched for. This search amounts to finding the region Ω_T which minimizes

$$J(\Omega_T) = D(I_{\text{ref}}(\Omega_R), I_{\text{target}}(\Omega_T)) \tag{1}$$

where D is a similarity measure, or distance, between the two sets of data. Ω_T and Ω_R are subsets of $\mathbb{R}^2$ (or subsets of $\mathbb{N}N^2$ in the discrete framework).

For clarity, the reference data set $I_{\text{ref}}(\Omega_R)$ will be denoted by R and the target data set $I_{\text{target}}(\Omega_T)$ will be denoted by T. Thus, $R(x)$ resp. $T(x)$, $x \in \Omega_R$ resp. $x \in \Omega_T$, represent corresponding samples from their respective region.

Traditionally, $R(x)$ and $T(x)$ are a triplet of color components in a given color space, *e.g.*, RGB or YUV.

Two aspects of similarity measures can be distinguished: radiometry which indicates if the regions have similar colors and geometry which correlates where these colors are present in the regions. Measures based solely on geometry, do the point-wise difference between the reference region and the target region. An example in segmentation is based on the Taylor expansion of this point-wise difference, the optical flow constraint [4].

Measures based solely on radiometry include distances between the probability density functions (PDF) of the color information in the regions, for example mutual information [1], Hellinger distance [16].

A widely used distance, in segmentation in [2], is the Kullback divergence[1]

$$\begin{aligned} D_{\mathrm{KL}}(T,R) &= \int_{\mathbb{R}^d} f_T(\alpha)\ \log \frac{f_T(\alpha)}{f_R(\alpha)}\ \mathrm{d}\alpha \\ &= -H(f_T) + H_\times(f_T, f_R) \end{aligned} \tag{2}$$

where f_T is the PDF of data set T, f_R is the PDF of data set R, H is the Shannon entropy and $H_\times$ is the cross entropy, also called relative entropy or likelihood.

The geometric constraint can be soften by expressing it in the PDF-based approach, *i.e.*, by adding geometry to the original radiometric data [8]. Formally, the PDF $f_R(\alpha)$ resp. $f_T(\alpha)$ is built on $\alpha = T(x) = \{I_{\mathrm{target}}(x), x\}$ for $x \in \Omega_T$ resp. on $\alpha = R(x) = \{I_{\mathrm{ref}}(x), x\}$ for $x \in \Omega_{\mathrm{ref}}$. x are spatial features, based on a coordinate system. In the next section, we will discuss which coordinate system could be used and which best fit for our method.

3 Spatial Features

First spatial features choice are the canonical Cartesian coordinates of the image. It needs though to be compensated by the motion of the region. A complex motion model is hard to define (for example for articulated objects) and computationally expensive to estimate. Instead we propose to define coordinates embedded in the region, named shape coordinates. In this way, estimation motion is skipped as shape coordinates change when the region deforms. The segmentation and motion deformation (hidden in the shape coordinates) are then jointly solved.

Most shape coordinates for shape correspondence are based on contour [11]. Here we aim at defining interior region coordinates. In practice, shape coordinates should have three properties. First they should map efficiently the region, *i.e.* each point should have a unique representation in the shape coordinates. Second, the shape coordinates should remain constant when the region goes

[1] Kullback-Leibler divergence is a not a distance, as it is not symmetric. The symmetrised Kullback divergence $D_{\mathrm{KL}}(T,R) + D_{\mathrm{KL}}(R,T)$ is a distance. For clarity, we presented all the calculus with the classic Kullback-Leibler divergence.

under deformation. Third property, computational speed must remain reasonable, as the spatial features are computed at each iteration of the active contour evolution.

3.1 Region Cartesian Coordinates

One can define cartesian coordinates local in the region, for example, we choose the bounding box of the region. Inside the bounding box, we define canonical cartesian region coordinates $\{x_{\text{region}}, y_{\text{region}}\}$.

This method, Kullback with Cartesian Geometric data (KL-CG) has 5-dimensional features: $\{Y, U, V, x_{\text{region}}, y_{\text{region}}\}$. The last two features are plotted on a example on Fig. 1.

This model should perform well for rigid objects, and is the one used for bounding box tracking in [8].

3.2 Region Polar Coordinates

We now consider polar coordinates. We define as the origin of the polar coordinates the barycenter of the region. When the object is articulated, the radius coordinate should remain constant, while the angle coordinate should measure the deformation. On the opposite, with the previous region Cartesian coordinates, both coordinates changes, in particular articulated members under rotation far from the barycenter (for example feet of a human body).

This approach could be extended to conformal mapping of a shape to a circle [9]. But for computational considerations, as the spatial features will have to be computed at each iteration of the active contour framework, we preferred basic polar coordinates.

This method, Kullback with Polar Geometric data (KL-PG) has 5-dimensional features: $\{Y, U, V, r_{\text{region}}, \theta_{\text{region}}\}$. The last two features are plotted on a example on Fig. 1. The discontinuity visible on this figure is due to the transition between the angles $-\pi$ to $+\pi$.

3.3 Distance Map and Contour Correspondence

As mentioned in the previous section, both Cartesian and polar coordinates on the region change when the region goes under deformations. In this section, we define shape coordinates constant under shape deformations. First, we propose to use the distance map d

$$d(x) = \min_{t \in [0,1]} ||x - C(t)||_2 \tag{3}$$

where $C : [0, 1] \rightarrow \mathbb{R}^2$ is a parametric curve representation of the contour.

This model has been proposed in medical images [10] where region of interest have uniform intensity on the level sets of the distance map. This model is in general not true on non medical videos. We propose to complete the distance map with another spatial feature: a contour correspondence. We chose the simplest

Fig. 1. Spatial features components $\{x, y\}$, first component red color, second component green color: $\{0, 0\}$ black, $\{1, 0\}$ red, $\{0, 1\}$ green, $\{1, 1\}$ yellow. *From left to right*: region cartesian coordinates (CG), region polar coordinates (PG), distance Map and NAL (DG).

contour correspondence coordinate, arc length of the nearest point on the contour l (NAL).

$$t_x = \arg \min_{t \in [0,1]} ||x - C(t)||_2 \tag{4}$$

$$l(x) = \int_0^{t_x} ||C'(t)|| \, \mathrm{d}t \tag{5}$$

Combining distance map coordinate and a nearest contour coordinate, we have a unique representation of each point in the shape. We must define an origin $C(0)$ on the parametric contour to define the NAL. We used as reference point the highest point of the curve in the image. This reference point can move with rotating or articulated objects but it is not the case in the videos used in our experiments. There are many works in the literature to define more efficient contour correspondences between two deformable shapes [11]. However, we did not yet implement these techniques, and even with this basic contour correspondence, our shape coordinates can handle many types of deformations.

This method Kullback with Distance Map and NAL Geometric data (KL-DG) has 5-dimensional features: $\{Y, U, V, d, l\}$. The last two features are plotted on a example on Fig. 1. The two discontinuities visible on this figure are on the skeleton of the region as the NAL changes. The other discontinuity is on the top of the head as it is the origin of the arc length.

Finally, this method can be seen as a joint distribution of radiometric data with a shape prior. While shape priors energy are often internal as they do not depend on the image (they contains no radiometric information). Shape priors geometrically match two regions [6] and the idea is to add radiometric information in this geometric match.

3.4 Weighting Between Components

One could claim our method is parameter-free as the use of joint probability allows no difficult weightings considerations (in comparison with quadratical error of different physical quantities). However, one may still want to define weightings between color and spatial components. On one hand, if there is knowledge on the rigidity on the object, one can increase the weightings on the spatial components, allowing more variability in color changes. On the other hand, if the object is articulated, one can lower the weighting of spatial components, relying more on color distributions.

To tune this parameter, we rescale spatial features, shape coordinates, into the interval $[0, 1]$, and we rescale the color features on the interval $[0, \alpha]$, α being the weighting parameter. Moreover, for the DG coordinates system, if we assume the region is a circle of radius r, the maximum of the distance map would be r while the maximum of the NAL would be $2\pi r$. To circumvent this, we divide the NAL coordinates by 2π.

4 Segmentation Using Active Contours

4.1 Estimation of Entropy and Probability Distributions

The energy used is the KL divergence. As developped in Eq. (2), it is a sum of two entropies. In this section we present a method to estimate entropy.

Ω_U defining dataset $U = \{I(x), x\}$ for x in Ω_U (U being either T or R, Ω_U being either Ω_T or Ω_R) has the following Shannon entropy:

$$H(U) = -\int_{\mathbb{R}^d} f_U(\alpha)\ \log f_U(\alpha)\ \mathrm{d}\alpha \tag{6}$$

Segmentation energies are usually defined by an integral on the region, see for examples [4,5,1,6,16].

To be coherent with these methods, we use the Ahmad-Lin estimate [18], named resubstitution estimate of entropy

$$\hat{H}_{\mathrm{AL}}(f_U) = -\frac{1}{|U|}\int_{\Omega_U} \log f_U(U(x))\, dx \tag{7}$$

where $|U|$ is the area of Ω_U. Since the actual PDF f_U is unknown, it must be estimated. A common practice is to use the non-parametric, Parzen windowing method.

$$\hat{f}_U(s) = \frac{1}{|U|}\int_{\Omega_U} K_\sigma(s - w)\mathrm{d}w \tag{8}$$

where K_σ is a multivariate, Gaussian kernel with standard variation, or bandwidth, σ.

4.2 Shape Derivative

The energy to be minimized through active contours is the Kullback divergence (2). In addition, as the distribution of the object can be characterized by a subregion inside the object, we propose to add a maximum area constraint with a weighting λ.

$$J(T) = D_{\mathrm{KL}}(T, R) - \lambda.|T| \tag{9}$$

To differentiate this energy we write its expression (7).

$$\begin{aligned} J(T) = &-\frac{1}{|T|}\int_{\Omega_T} \log f_T(T(x))\, dx \\ &- \frac{1}{|T|}\int_{\Omega_T} \log f_R(R(x))\, dx - \lambda.|T| \end{aligned} \tag{10}$$

We have an energy which has many dependencies with the region Ω_T. We propose to compute the shape derivative [16] on Ω_T in the direction of a vector field V. We have then:

$$\mathrm{d}J_r(T, V) = \int_{\partial\Omega_T} \mathcal{A}(s) \quad V(s).N(s)\mathrm{d}s \tag{11}$$

with

$$\begin{aligned} \mathcal{A}(s) = &\frac{1}{|T|}(D_{\mathrm{KL}}(T, R) - \log f_T(s) + \log f_R(s)\,) + \lambda \\ &+ \frac{1}{|T|^2}\int_{\Omega_T} 1 - \frac{K_\sigma(T(x) - T(s))}{f_T(T(x))}\mathrm{d}x \end{aligned} \tag{12}$$

and where N is the inward unit normal of $\partial\Omega_T$.

The minimization of the energy is then performed using the active contour technique [4,16] using B-splines. An initial contour is iteratively deformed according to V chosen such that derivative is negative or equal to zero at each iteration. The minimum is reached when the derivative is equal to zero. The corresponding shape of the active contour represents the segmentation. The shape derivative leads to the following velocity:

$$V(s) = -\mathcal{A}(s)N(s) \tag{13}$$

One can note that Eq. (12) requires the PDF estimation of a 5-dimensional joint geometric/radiometric data set (three color components plus two spatial components) of the reference region and the target region: R(x) and T(x). The sparsity of this high-dimension data space makes the PDF estimation, and therefore the similarity measure estimation, even more problematic. Let us present a new framework for computing high dimensional PDFs and Kullback divergence.

5 The k-th Nearest Neighbor (kNN) Framework

5.1 Parzen Windowing Limitations

The Parzen method for PDF estimation makes no assumption about the actual PDF. Consequently, the estimated PDF cannot be described in terms of a small number of parameters, as opposed to, say, a Gaussian distribution defined by its mean and variance. This method is therefore qualified as non-parametric. It approximates the density at sample s with the relative number of samples $k(s)/|U|$ falling into the open ball of volume v centered on s

$$\hat{f}_U(s) = \frac{k(s)}{v\ |U|}. \tag{14}$$

Let us remind U is either R or T, and $|U| = |\Omega_U|$ is the number of samples of U. The first difficulty in using the Parzen method is the critical choice of the window size [12]. Another difficulty is due to what is informally called the curse of dimensionality. As the dimension of the data space increases, the space sampling gets sparser. Therefore, less samples fall into the Parzen windows centered on each sample, making the PDF estimation less reliable. Dilating the Parzen window does not solve this problem since it leads to over-smoothing the PDF. In a way, the limitations of the Parzen method come from the fixed size of the window: the method cannot adapt to the local sample density. The k-th nearest neighbor (kNN) framework provides an advantageous alternative.

5.2 kNN Estimation of PDFs

In the Parzen method, the density of U at sample s is related to the number of samples falling into a window of fixed size σ centered on the sample (see Eq. (14)). The kNN method is the *dual* approach: the density is related to the size of the window σ necessary to include its k nearest neighbors. Let us note $\sigma(s) = \rho_k(U, s)$ the distance to the k-th nearest neighbor of s among the data set U.

This variable size estimate is called locally adaptive, it can be performed it two different ways. The first way is called balloon estimate [14]:

$$\hat{f}_U(s) = \frac{1}{|U|} \sum_{x \in \Omega_U} K_{\sigma(s)}(s - U(x)) = \frac{k}{v_d\ |U|\ \rho_k^d(U, s)} \tag{15}$$

where v_d is the volume of the unit ball in $\mathbb{R}^d$. Size of kernels $\sigma(s) = \rho_k(U, s)$ depends on the sample point s where $\hat{f}$ is evaluated. It can be reduced to a simple expression when $K_{\sigma(s)}$ is a uniform kernel. The second way is called sample point estimate [14]:

$$\hat{f}_U(s) = \frac{1}{|U|} \sum_{x \in \Omega_U} K_{\sigma(U(x))}(s - U(x)). \tag{16}$$

Size of kernels $\sigma(U(x)) = \rho_k(U, U(x))$ depends on each sample points $U(x)$.

We will consider the balloon estimate as it is the underlying PDF estimate in the kNN expression of entropy. However, we will discuss later why the sample point estimate can be useful. Although the distance is usually computed in the Euclidean sense, other distances can be used. Let us remind that the data are a subset of $\mathbb{R}^d$ with $d = 5$ The choice of k appears to be a much less critical than the choice of the window size in the Parzen method. Actually, when the kNN approach is used for parameter estimation, k must be greater than the number of parameters and such that $k/|U|$ tends toward zero when both k and $|U|$ tends toward infinity. A typical choice is $k = \sqrt{|U|}$.

5.3 Kullback Distance Estimation

Based on the Ahmad-Lin entropy estimator (7) and the kNN PDF estimation (15), a consistent and unbiased entropy estimator was proposed with a proof of consistency under weak conditions on the underlying PDF [15] The kNN estimate of the Shannon entropy is equal to

$$\begin{aligned}\hat{H}_{\text{kNN}}(f_T) = \log(v_d\ (|T|-1)) - \psi(k) \\ +d\ \mu_T(\log \rho_k(T\backslash\{.\}))\end{aligned} \tag{17}$$

where $\mu_T(g)$ is the mean of g for all the values taken over data set T where ψ is the Polygamma function Γ'/Γ and where $T\backslash\{.\}$ indicates that the k-th nearest neighbor is search for each samples in T excluding the sample itself. Note that estimator (17) does not depend on the PDF $\hat{f}_T$. Informally, the main term in estimate (17) is the mean of the log-distances to the k-th nearest neighbor of each sample.

In the same framework, the cross entropy of two data sets R and T can be approximated by

$$\begin{aligned}\hat{H}_{\times,\text{kNN}}(f_T, f_R) = \log(v_d\ |R| - \psi(k) \\ +d\ \mu_T(\log \rho_k(R)).\end{aligned} \tag{18}$$

Note again that estimator (18) does not depend on any PDF and that its main term is the mean of the log-distances to the k-th nearest neighbor among data set R of each sample of T. Finally, since the Kullback-Leibler distance is a difference between a cross entropy and a Shannon entropy (see Eq. (2)), the kNN estimate of this distance is equal to

$$\begin{aligned}D_{\text{KL}}(T, R) = \log \frac{|R|}{|T|-1} + d\ \mu_T(\log \rho_k(R)) \\ -d\ \mu_T(\log \rho_k(T\backslash\{.\}))\end{aligned} \tag{19}$$

where $T\backslash\{.\}$ indicates that the k-th nearest neighbor is search for in T excluding the sample itself.

5.4 Simplification in Active Contours Using kNN

Using for f_R and f_T the kNN expression given in (15) and for D_{KL} the expression given in (19), expression (12) reduces to:

$$\begin{aligned}\mathcal{A}(s) = &- \frac{d}{|T|}[\mu_T(\log \rho_k(R)) - \mu_T(\log \rho_k(T)) \\ &- \log \rho_k(R,s) + \log \rho_k(T,s)] + \lambda \\ &+ \frac{1}{|T|}(1 - \frac{1}{k} \sum_{x \in \nu(s,T)} \left(\frac{\rho_k(T,x)}{\rho_k(T,s)} \right)^d).\end{aligned} \tag{20}$$

where $\nu(s,T)$ is the support of $K_{\sigma(s)}$, which in the kNN framework, is a uniform kernel centered in s of size $\rho_k(T,s)$ (15). Moreover one can note that choosing the sample point estimate expression (16) instead of the balloon estimate expression (15), the last row in (20) would be equal to zero. However both simplified expression of Kullback distance (19) and simplified expression of PDF estimate (15) are only valid for balloon estimate of PDFs.

Finally, kNN version of $\mathcal{A}(s)$ (20) is plugged in the Eulerian derivative (11) and evolution equation (13). Let us remind active contour energy (9) and active contour evolution equation (13) required a high dimensional joint PDF over the data. Here, we presented a new framework to estimate both active contour energy (9) and active contour evolution equation (13) without explicit estimation of the PDF but with a reduced expression using distances to nearest neighbors.

6 Experimental Results

In this section we will compare two methods, the Kullback distance computed through kNN but with no geometry kNN-KL-NG (no spatial features, R and T are 3-Dimensionals) and the Kullback distance computed through kNN with geometry in a general sense kNN-KL-G (spatial features, R and T are 5-Dimensionals), geometry being either cartesian kNN-KL-CG, polar kNN-KL-PG or distance map with NAL kNN-KL-DG. k of the kNN framework is set to $\sqrt{|T|}$.

The reference histograms for kNN-KL-NG and kNN-KL-G are built over a region Ω_R on frame 1 for "Erik" Fig. 2, frame 74 for "Football" Fig. 3 using a manual segmentation. The goal is to find the corresponding region Ω_T in frame 6 for "Erik", frame 75 for "Football". We initialize the segmentation with a circle far from the solution to show the stability of the method.

First we present results on sequence "Erik" Fig. 2, size 288×352. This sequence shows a translating man over a static background. This sequence was chosen because its motion is very simple, while it is composed of many colors which will lead to complex color histograms. This sequence is considered as rigid, we tuned the weighting α presented in Section 3.4 to 1. This means, an error of 1 unit in geometry is similar to an error of 1 color intensity. Some parts of the background have similar colors than Erik. Therefore kNN-KL-NG includes it as object while kNN-KL-G detects their spatial features are not correct so it does not include

Fig. 2. Segmentation on sequence "Erik" on frame 6: *(from left to right and top to bottom)* region of interest Ω_R manually segmented on frame 1, initialization of the segmentation, results Ω_T of segmentation with method kNN-KL-NG, results Ω_T of segmentation with method kNN-KL-CG,results Ω_T of segmentation with method kNN-KL-PG,results Ω_T of segmentation with method kNN-KL-DG

it as object. These results did not use maximum area constraint, $\lambda = 0$. As expected for rigid objects, all spatial features kNN-KL-CG, kNN-KL-PG and kNN-KL-DG led to exactly the same results, which is the correct segmentation.

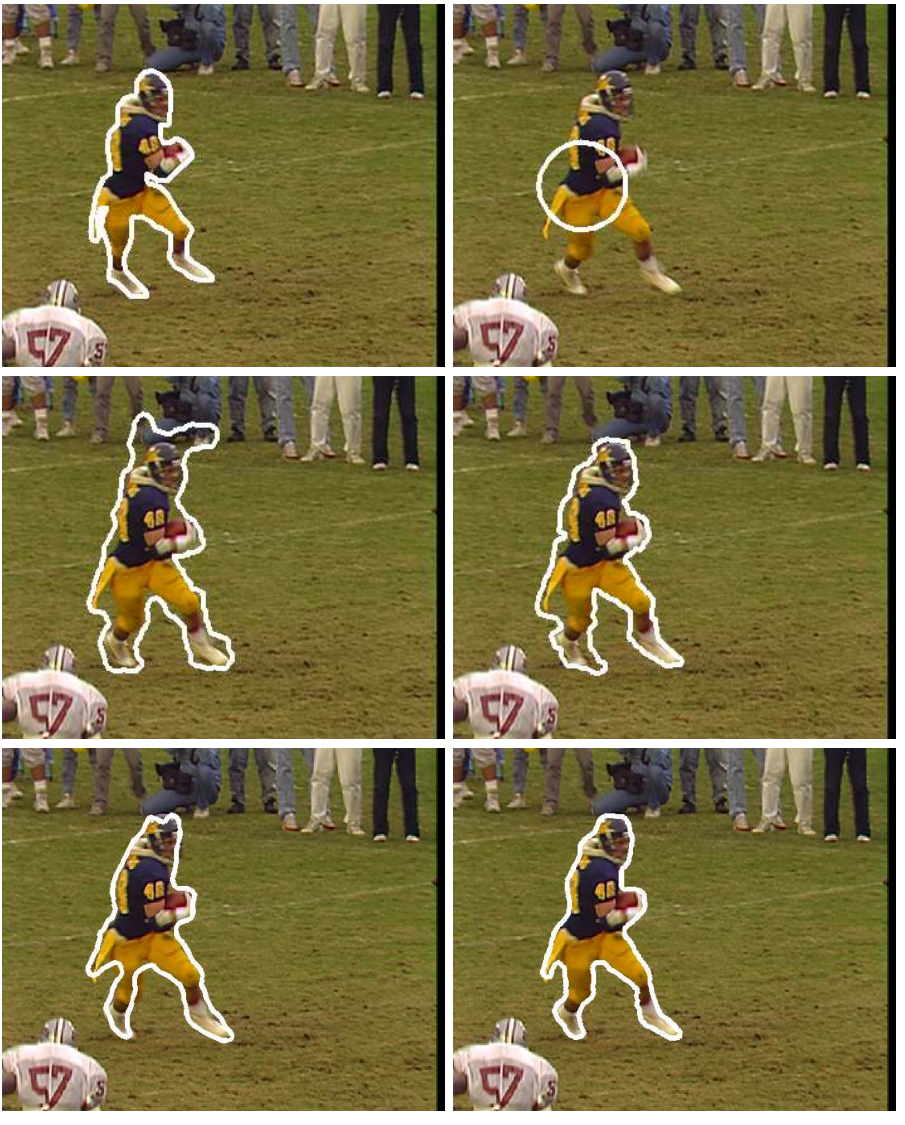


Fig. 3. Segmentation on sequence "Football" on frame 75: *(from left to right and top to bottom)* region of interest Ω_R manually segmented on frame 74, initialization of the segmentation, results Ω_T of segmentation with method kNN-KL-NG, results Ω_T of segmentation with method kNN-KL-CG,results Ω_T of segmentation with method kNN-KL-PG,results Ω_T of segmentation with method kNN-KL-DG

Results are presented on sequence "Football" Fig. 3, size 288×352. This sequence shows fast and articulate motions. This sequence is considered as non-rigid, we tuned the weighting α presented in Section 3.4 to 10. This means, an

error of 10 units in geometry is similar to an error of 1 in color intensity, a good motion variability is allowed in this sequence. Some parts of the public on the upper part of the video have the same colors as the player. kNN-KL-G excludes again them as their spatial features are not correct while kNN-KL-NG includes them in the segmentation. The Kullback distance kNN-KL-G slightly increase when taking the legs of the player as their are articulated (error of registration in the spatial features). However, as the geometric constraint is soft, it increases less than with segmenting the public, the player is then correctly segmented with the help of maximum area constraint. Here the results with different spatial features are not the same. kNN-KL-CG has difficulties to properly segment the legs of the play as it is nonrigid (spatial features have changed). kNN-KL-PG is a little better as the spatial features change only on one spatial feature components (the angle). Finally kNN-KL-DG gives the best results as the spatial features on the reference frame and on the target frame represent the same pixels.

The maximum area constraint was tuned to segment the whole object in all cases. kNN-KL-NG and kNN-KL-CG required a parameter $\lambda = 2.10^{-4}$ to ignore color variability for kNN-KL-NG, and to ignore spatial features variability for kNN-KL-CG. It segmented all the football player but it led to an over-segmentation in some parts of the image. kNN-KL-PG only needed a parameter $\lambda = 1.10^{-4}$ to segment all the football player, leading to less over-segmentation. Finally kNN-KL-DG needed a parameter $\lambda = 5.10^{-5}$ to segment all the football player, leading to accurate segmentation.

Finally we discuss about the robustness to the bandwidth parameter k of kNN. $k = \sqrt{|T|}$, setting k to $2^n\sqrt{|T|}$, from $n = -2$ to $n = 2$, the absolute difference between the segmentation masks and the one generated is less than $3.7\% \times |T|$. The segmentation algorithm is robust to the bandwidth k, changing from 1 time to 16 times its initial value, while segmentation methods based on Parzen techniques observed an high sensitivity to the bandwidth parameter.

7 Conclusion and Future Works

The results presented for this method show the applicability of kNN framework in active contour segmentation. It shows that it can be particularly efficient in high dimensional cases such as joint feature-spatial segmentation. We proposed new shape coordinates for deformable regions. The results tend to show that for rigid object, shape coordinates are not an issue. On the opposite, for deformable regions, our region shape coordinates compares favourably to other region coordinates systems.

We only compared soft geometric to no geometric, and different shape coordinates. One could expect comparisons with classical Parzen techniques for segmentation. First, a comparison would not be fair as the model presented in [10] is only valid for medical images as discussed in Section 3.3. Second, this model uses only grayscale and distance map feature, namely 2-dimensional features. This model is acceptable for Parzen, but when using more high dimensional features (5-dimensional features in our model) Parzen techniques suffer the curse

of dimensionality while kNN can handle it. We refer to [14] for full details on this subject.

Finally our method compares favorably to state of the art, on sequence "Football". In this paper we compared to a radiometric method. Geometric methods often require a motion estimation step. An affine motion model cannot model the high level of deformations of the articulated player. In addition the sequence has motion blur and there are fast motions. As a consequence, optical flow estimation is also very difficult on this sequence.

Future works will improve our shape coordinates by using a more efficient contour correspondence, shape contexts [11], to combine it with a distance map coordinate, and explore other shape coordinates such as Free Form deformations [19]. We will also apply this high dimensional framework to combine other multiple features [5] (motion, texture, wavelet coefficients).

References

1. Kim, J., Fisher, J.W.F., Yezzi, A., Çetin, M., Willsky, A.S.: A nonparametric statistical method for image segmentation using information theory and curve evolution. IEEE Transactions on Image Processing 14(10), 1486–1502 (2005)
2. Freedman, D., Zhang, T.: Active contours for tracking distributions. IEEE Transactions on Image Processing 13(4), 518–526 (2004)
3. Black, M.J., Anandan, P.: The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. Computer Vision and Image Understanding 63(1), 75–104 (1996)
4. Cremers, D., Soatto, S.: Motion competition: A variational framework for piecewise parametric motion segmentation. International Journal of Computer Vision 62(3), 249–265 (2005)
5. Brox, T., Rousson, M., Deriche, R., Weickert, J.: Unsupervised segmentation incorporating colour, texture, and motion. In: Petkov, N., Westenberg, M.A. (eds.) CAIP 2003. LNCS, vol. 2756, pp. 353–360. Springer, Heidelberg (2003)
6. Rousson, M., Paragios, N.: Shape priors for level set representations. In: European Conference on Computer Vision, pp. 78–92 (2002)
7. Cremers, D., Osher, S.J., Soatto, S.: Kernel density estimation and intrinsic alignment for shape priors in level set segmentation. International Journal of Computer Vision 69(3), 335–351 (2006)
8. Elgammal, A., Duraiswami, R., Davis, L.S.: Probabilistic tracking in joint feature-spatial spaces. In: IEEE International Conference on Computer Vision and Pattern Recognition, Madison, WI, pp. 781–788. IEEE Computer Society Press, Los Alamitos (2003)
9. Sharon, E., Mumford, D.: 2d-shape analysis using conformal mapping. International Journal of Computer Vision 70(1), 55–75 (2006)
10. Leventon, M.E., Faugeras, O., Grimson, W.E.L., Wells III, W.M.: Level set based segmentation with intensity and curvature priors. In: IEEE Computer Society Workshop on Mathematical Methods in Biomedical Image Analysis (2000)
11. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Transactions Pattern Analysis Machine Intelligence 24(4), 509–522 (2002)

12. Scott, D.: Multivariate Density Estimation: Theory, Practice, and Visualization. Wiley, Chichester (1992)
13. Fukunaga, K.: Introduction to statistical pattern recognition, 2nd edn. Academic Press Professional, Inc, London (1990)
14. Terrell, G.R., Scott, D.W.: Variable kernel density estimation. The Annals of Statistics 20, 1236–1265 (1992)
15. Goria, M.N., Leonenko, N.N., Mergel, V.V., Inverardi, P.L.N.: A new class of random vector entropy estimators and its applications in testing statistical hypotheses. J. Nonparametr. Stat. 17(3), 277–297 (2005)
16. Aubert, G., Barlaud, M., Faugeras, O., Jehan-Besson, S.: Image segmentation using active contours: Calculus of variations or shape gradients? SIAM Applied Mathematics 1(2), 2128–2145 (2003)
17. Boltz, S., Debreuve, E., Barlaud, M.: High-dimensional statistical distance for region-of-interest tracking: Application to combining a soft geometric constraint with radiometry. In: CVPR 2007. IEEE International Conference on Computer Vision and Pattern Recognition, Minneapolis, USA (2007)
18. Ahmad, I., Lin, P.: A nonparametric estimation of the entropy for absolutely continuous distributions. IEEE Transactions on Information Theory 22, 372–375 (1976)
19. Huang, X., Paragios, N., Metaxas, D.: Shape registration in implicit spaces using information theory and free form deformations. IEEE Transactions Pattern Analysis Machine Intelligence 28(8), 1303–1318 (2006)

Robust Spectral 3D-Bodypart Segmentation Along Time

Fabio Cuzzolin, Diana Mateus, Edmond Boyer, and Radu Horaud

INRIA Rhone-Alpes, 655 avenue de l'Europe, 38334 Montbonnot, France
Fabio.Cuzzolin@inrialpes.fr, Diana.Mateus@inrialpes.fr,
Edmond.Boyer@inrialpes.fr, Radu.Horaud@inrialpes.fr

Abstract. In this paper we present a novel tool for body-part segmentation and tracking in the context of multiple camera systems. Our goal is to produce robust motion cues over time sequences, as required by human motion analysis applications. Given time sequences of 3D body shapes, body-parts are consistently identified over time without any supervision or *a priori* knowledge. The approach first maps shape representations of a moving body to an embedding space using locally linear embedding. While this map is updated at each time step, the shape of the embedded body remains stable. Robust clustering of body parts can then be performed in the embedding space by k-wise clustering, and temporal consistency is achieved by propagation of cluster centroids. The contribution with respect to methods proposed in the literature is a totally unsupervised spectral approach that takes advantage of temporal correlation to consistently segment body-parts over time. Comparisons on real data are run with direct segmentation in 3D by EM clustering and ISOMAP-based clustering: the way different approaches cope with topology transitions is discussed.

1 Introduction

Human motion analysis is an important topic in computer vision with many applications in surveillance, human machine interface and animation, among others. Such analysis, when based on image observations, relies on the ability to extract body motion information from images. This problem has received a considerable attention from the community over the last decades [1]. The existing approaches mainly differ in the amount and type of information that is known in advance. *Model based* approaches, e.g. [2,3,4], assume a known, often kinematic, model for the human body and recover parameters of this model in a joint space using image evidence. However, the joint space is generally high dimensional, making the search for model parameters difficult without adequate initializations, an issue sometimes solved by stochastic sampling. *Learning based* approaches, e.g. [5,6,7,8] directly relate visual information to learned body configurations, without the need for an intermediate model. While solving the initialization issue, these approaches are anyway limited by the classes of examples used for training.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 196–211, 2007.

In opposition, approaches have been proposed that directly infer body poses for markerless motion capture from multiple image cues, in particular volume sequences [9,10,11]. This includes *skeletonization* methods that recover the intrinsic articulated structure of 3D shapes, either directly in 3D, e.g. [12], or in an embedded space, e.g. [13,14]. A nice feature with embeddings is the ability to map 3D shapes onto low-dimensional manifolds in higher dimensional spaces, thus naturally revealing the intrinsic structure of an articulated shape. A critical issue, though, is the presence of topological ambiguities raised by self contacts (joint hands, for instance). This was noticed by Sundaresan and Chellappa [14] who used an *a priori* graphical body model to resolve these ambiguities.

In this work, we propose an approach that segment body-parts in 3D body shape sequences. We consider this an intermediate step towards a robust human motion analysis framework without any *a priori* information nor learned examples, as demonstrated in this paper. Our approach uses spectral embedding to map shapes onto low-dimensional manifolds which are then clustered into body-parts. One crucial innovation is that we take advantage of the correlation between information along time sequences to segment in a consistent way. Our goal is to guarantee robustness, in particular to topological ambiguities that occur over time. Recent attempts to extend nonlinear reduction to spatio-temporal data [15,16] provide indeed elegant solutions to enforce temporal relationships when embedding time sequences. Unfortunately, such relationships are not easily identified with the dense shape representations of moving bodies that we consider. Instead, we propose to enforce temporal consistency through clustering in the embedded space, where clusters are remarkably stable under articulated motions and there propagated over time. Although several spectral embedding methods could be, in principle, considered for that purpose [17,18], we chose *Local Linear Embedding* [19] (LLE) which exhibits better performances in our specific scenario. LLE is fast, maps a shape to a low-dimensional manifold in the embedded space and is already partially robust to topology changes as it depends on the local structure of the data.

The rest of the paper is organized as follows: after motivating the choice of clustering in time in an embedding space (Section 2), we present each step of the algorithm in Section 3: the use of *k-wise* clustering to segment the embedded cloud (3.1), starting from detected branch terminations (3.2); how to propagate cluster seeds over time to ensure consistency (3.3) and merge/split them according to the current topology of the body (3.4). In an extensive experimental section (5) we present results on unsupervised segmentation over a large number of dense voxelset sequences, we compare them with segmentation in 3D by EM clustering, show how to learn the parameters of the algorithm from the data, and discuss the way different approaches cope with topology transitions.

2 Motivation: Clustering After Locally Linear Embedding

Embedding techniques are interesting for clustering purposes because of their characteristic of "amplifying" the separation between different parts of the same

3D shape. This is true in particular for Locally linear Embedding (LLE) [19], but also holds for other spectral methods like Laplacian Eigenmaps.

LLE is an unsupervised learning algorithm which computes d-dimensional embeddings Y of sets of input points $X = \{x_i, i = 1, .., N\}$ living in a nonlinear manifold, while preserving their local structure (i.e. the distances between each point and its k neighbors). Groups of local neighborhoods belonging to a same part of the shape are "redistributed" by the algorithm along distinct chains. Figure 1-middle shows how legs and arms of a human body are much well

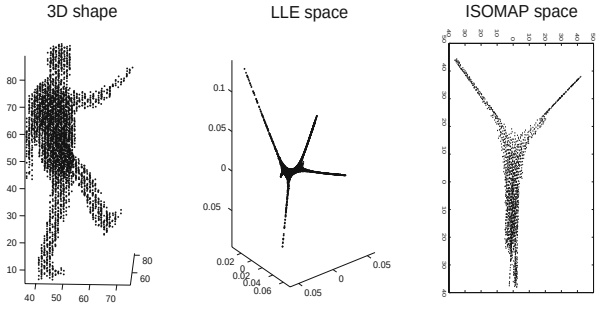


Fig. 1. How LLE (middle) and ISOMAP (right) map the same 3D cloud (left) for the same number of neighbors $k = 13$. Arms (lower appendices) are indistinguishable in the ISOMAP space.

separated in the LLE embedding space than in 3D (left). This is much less true, however, for methods (like ISOMAP [18], right) based on geodesic distances. Clustering in the embedding space is then typically easier than in 3D.

LLE, in particular, has some edges over other embedding schemes: for obvious reasons it is less sensitive (with respect to geodesic-based embeddings) to *changes in the topology* of the moving body (as we show in Section 5.4). It is computationally less expensive than ISOMAP. Finally, as we will show in Section 3.1, the lower dimension of the embedded clouds it generates makes them suitable to be clustered in a more robust way using *k-wise clustering* [23].

2.1 Clustering Along Time and Pose-Invariance

When clustering sequences, though, we need the segmentation obtained at different time instants to be *consistent*. A desirable cue, in this sense, is the fact that some embedding schemes (like ISOMAP) are inherently *pose-invariant* under articulated motion (as while the articulated body evolves geodesic distances between pairs of points do not change). This is not true, in a strict sense, for LLE. However, as its embedding depends only on the local structure of the input dataset, it is reasonable to conjecture that under articulated motion the shape of the LLE embedded cloud would exhibit remarkable stability. An articulated object is formed by a number of rigid bodies linked by a small number of joints:

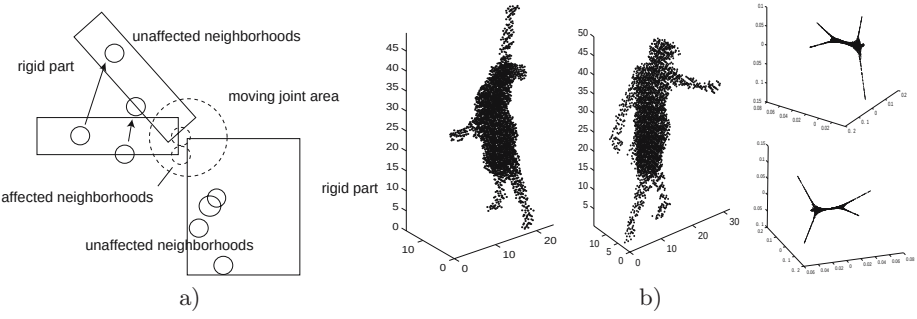


Fig. 2. a) The number of neighborhoods affected by articulated motion is relatively small. b) Some anecdotal evidence on the stability of locally linear embedding under articulated motion. Different poses (left, middle) of the same articulated body are mapped to the same embedded cloud (right), for a large interval of parameter values.

Clearly all local neighborhoods incident on a rigid part are preserved along the motion, while only the few neighborhoods interested by the evolving joint(s) are affected (Figure 2-a).

The validity of this claim depends of course on the number of points N in the dataset, the neighborhood size k (as smaller ks reduce the number of neighborhoods with non-empty intersection with moving joints), and last but not least the number of evolving joints.

As an example let us consider two different poses of the same articulated body, for instance a dancer performing a ballet, represented as voxelsets (Figure 2-b, middle and left). Figure 2-b right shows the related embedded clouds obtained through LLE for $d = 3$ and $k = 10$. Their similarity is apparent. Analogous results can be obtained for a wide range of values of the critical parameter k.

Stability and other desirable properties are actually shared by other embedding schemes like, for instance, Laplacian Eigenmaps [17]. In the following we will make reference in particular to LLE.

3 Approach

3.1 K-Wise Clustering in the Embedded Shape

Let us first focus on the problem of segmenting an embedded cloud at a given time instant. As we mentioned above (and as Figures 1 and 2 confirm) for $d = 3$ the embedded cloud is (for a wide interval of values of k) a tree-like one-dimensional curve. It is then natural to look for clusters formed by sets of roughly collinear embedded points. With a few exceptions, clustering algorithms (like k-means [20]) are based on the assumption that a pairwise measure of distance between data-points is available. As every pair of data-points trivially defines a line, however, there does not exist a useful measure of similarity between such pairs. It is instead possible to define measures of similarity over *triplets* of points to indicate how close they are to being collinear (Figure 3-a) [21,22]).

In general the problem of clustering points based on similarity between k-tuples of points is called *k-wise clustering*. An interesting approach to k-wise clustering has been proposed in [23]. Consider a set of datapoints $V = \{v_i, i = 1, ..., N\}$.

K-Wise Clustering Algorithm

1. In the first step an *affinity hypergraph* is built. A weighted undirected hypergraph H is a pair (V, h), where V is the set of vertices of H, and subsets z of V of size k are called hyperedges. The function h associates nonnegative weights $h(z)$ with each hyperedge (k-tuple) $z = \{v_{j_1}, ..., v_{j_k}\}$, and measures the affinity of each hyperedge.
2. Then, a weighted graph $G = (V, g)$ that approximates the hypergraph H is constructed by constrained least square optimization, based on the assumption that $h(z) = \sum_{v_i, v_j \in z; i<j} g(v_i, v_j)$, i.e. the weight of each hyperedge is the arithmetic mean of the weights of the edges of G incident on it (*clique averaging*).
3. Finally, to partition the approximating graph G into k parts a spectral clustering algorithm is adopted that uses the first k eigenvectors of the normalized Laplacian of the graph and performs k-means clustering on the resulting k-dimensional embedding [24,25].

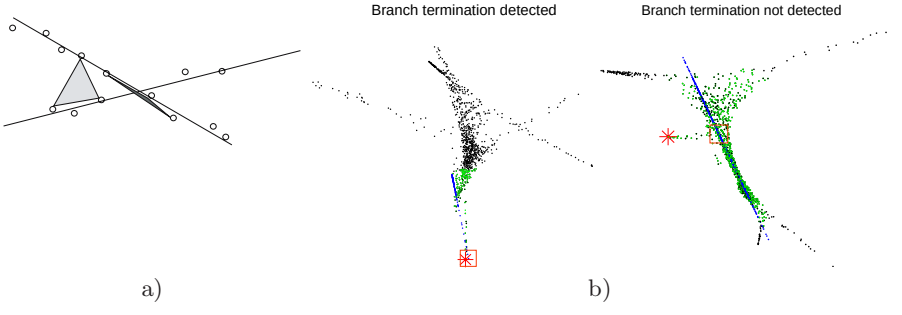


Fig. 3. a) 3-wise (k-lines) clustering. Areas of triangles defined by triads of points measure their collinearity. The smaller the area (dark triangle) the greater the collinearity. b) Termination (left) and internal (right) points of the embedded cloud are characterized by the fact that their projection (red square) on the line (in blue) interpolating their neighborhoods (in green) is an extremum of the interval of all projections.

In our case, the hypergraph to approximate has as set of vertices the embedded cloud $V = Y$, and hyperedges formed by d elements (for a d-dimensional embedding space). Specifically, these hyperedges are triads of points for $d = 3$. A natural choice for the affinity of these triads is then the area of the triangle they form (the volume of the $d-1$-dimensional hyperedge in the general case).

The application of the k-wise clustering algorithm to the embedded cloud yields a segmentation in the embedding space that can be trivially remapped

back to the original 3D space, using the ordering of the data-points. Notice that, as step 3. of the k-wise clustering algorithm involves standard k-means, and the latter requires a set of initial seeds to start clustering from, seeds are required to initialize the overall algorithm as well. We will address this issue in Section 3.3. It remains to decide the number of clusters for a given embedded shape.

3.2 Branch Detection and Number of Clusters

The fact that embedded clouds typically appear as one-dimensional strings formed by a number of branches (corresponding to the extremities of the moving body) provides us with a simple method to estimate, at each given time instant, the "correct" number of clusters (Figure 3-b).

Each point of the embedded cloud is tested to decide whether or not it is a branch termination. This test is performed by finding its nearest neighbors (for a certain threshold distance which can be empirically learned from the data), plotted in green. The best interpolating line for all neighbors is then found (in blue), and all neighbors projected on it. A point of the embedded cloud (red star in Figure 3-b) is a branch termination if the projection of all its neighbors on the interpolating line lay on one side of its own projection (red square) like in Figure 3-b-left, it is not when the projection has neighbors on both sides (Figure 3-b-right).

This algorithm proves to work extremely well on embedded clouds generated through LLE. It becomes then possible to detect transitions in the topology of the moving body when they happen, and modify the number of clusters accordingly. We will return on this in Section 3.4.

3.3 Temporal Consistency and Seed Propagation

When considering entire sequences of 3D clouds we need to ensure the *temporal consistency* of the segmentation: in normal situations (no topology changes due to contact of different body-parts) the cloud has to be decomposed into the "same" groups in all instants of the sequence. We propose a propagation scheme in which centroid clusters at time t are used to generate initial seeds for clustering at time $t+1$ (Figure 4). Let n be the number of clusters.

Seed Propagation Algorithm

1. The embedded cloud $Y(t)$ at time t is clustered using d-wise clustering (Section 3.1, Figure 4-bottom-left) using the current seeds $c'_j(t)$ (the branch terminations of $Y(0)$ if $t = 0$, computed as in step 4 for $t > 0$);
2. For each centroid $c_j(t)$ $j = 1, ..., n$, of these clusters, the original datapoint $x_{i_j}(t)$ (*3D cluster centroid*) whose embedding $y_{i_j}(t)$ is the closest neighbor of $c_j(t)$ is found (Figure 4-top-left):

$$i_j(t) = \arg \min_{i=1,...,N} \|y_i(t) - c_j(t)\|^2. \quad (1)$$

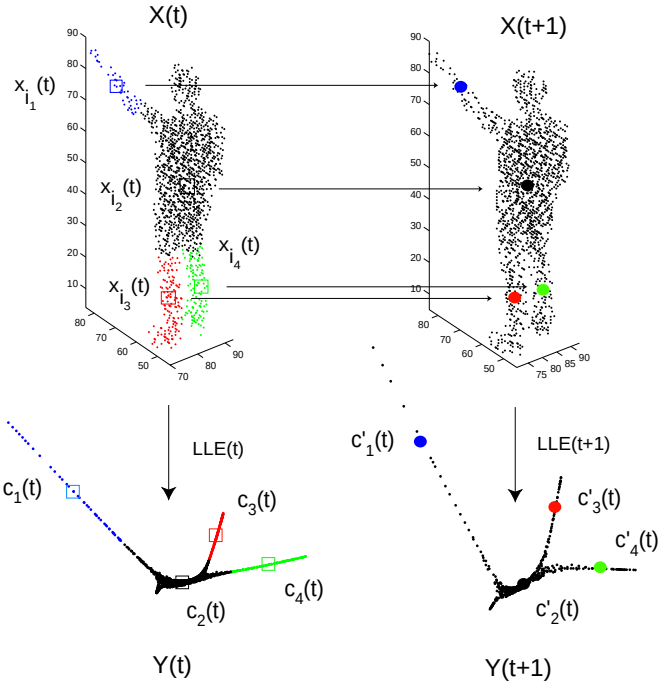


Fig. 4. Seed propagation for consistent clustering along time in the embedding space. The anti-images of centroids at time t are added to the 3D cloud at time $t+1$. Their embeddings $c'_j(t+1)$ are the seeds from which to start clustering the new embedded cloud $Y(t+1)$.

3. At time $t+1$, the dataset of 3D input points $X(t+1) = \{x_i(t+1), i = 1, ..., N(t+1)\}$ at time $t+1$ is augmented with the positions of the old 3D centroids at time t, yielding a new dataset (Figure 4-top-right)

$$X'(t+1) = X(t+1) \cup \{x_{i_j}(t), j = 1, ..., n\}. \tag{2}$$

4. LLE is applied to the extended dataset $X'(t+1)$, obtaining (Figure 4-bottom-right) [1]

$$Y(t+1) \cup \{c'_j(t+1), j = 1, ..., n\}. \tag{3}$$

5. The images $c'_j(t+1)$ of the *old* 3D centroids $x_{i_j}(t)$ in the *new* embedded space will then be used as seeds to start the k-wise clustering of the new embedded cloud $Y(t+1)$.

3.4 Topology Changes and Dynamic Clustering

The question of how to initialize the seeds for $t = 0$ naturally arises. Besides (even though working in an embedding space helps to dramatically reduce the

[1] Embeddings $c'_j(t+1)$ of the old 3D centroids $x_{i_j}(t)$ in the new embedded cloud can also be computed by *out of sample extension* [26].

problem of segmenting body-parts which get close to each other) moments in which different parts of the articulated body come to contact still have important effects on the shape of the embedded cloud. In fact, in an unsupervised context in which we do not possess prior knowledge about the number of rigid parts which form the body or the way they are arranged, there is no reason to distinguish adjacent body-parts. It is instead more sensible to adapt the number of clusters to the number of actually distinguishable parts.

The branch detection algorithm of Section 3.2 provides a tool to initialize the clustering machinery and implement the necessary change in the number and location of clusters when a topology change occurs.

Clusters' Merging/ Splitting Algorithm

1. At each time instant t all branch terminations of the embedded cloud $Y(t)$ are detected; if $t = 0$ they are used as seeds for k-wise clustering.
2. Otherwise ($t > 0$) standard k-means is performed on $Y(t)$ using branch terminations as seeds, yielding a rough partition of the embedded cloud into distinct branches.
3. Propagated seeds $c'_j(t)$ in the same partition are merged.
4. For each partition of $Y(t)$ not containing any old seed a new seed is defined as the related branch termination.

The third situation takes place when previously separated body-parts get too close to be distinguished: it makes then sense to merge the corresponding clusters. 4. embodies the opposite event in which a body-part which was previously impossible to distinguish becomes well separated, requiring then the introduction of a new cluster. This way clusters merge and/or split according to topological changes in the moving articulated body.

4 Algorithm

It is time to summarize our approach for unsupervised robust segmentation of parts of moving articulated bodies in a consistent way along a sequence (by assembling the separate algorithms we described in Sections 3.1, 3.3, 3.4). For each time instant t:

1. The current dataset ($X(t) = \{x_i(t), i = 1, ..., N(t)\}$ for $t = 0$, $X'(t) = X(t) \cup \{x_{i_j}(t-1)\}$ for $t > 0$, Figure 5-a) is mapped to an embedding space of dimension d yielding $Y(t) = \{y_i(t), i = 1, ..., N(t)\} = LLE(X(t))$.
2. All branch terminations of the embedded cloud $Y(t)$ are detected (Section 3.2): the natural number of clusters $n(t)$ for time t is then set to the number of branches (plus one for the torso), Figure 5-b.
3. The embedded cloud $Y(t)$ is clustered into $n(t)$ groups by d-wise clustering (Section 3.1) starting from $n(t)$ seeds (Figure 5-c):

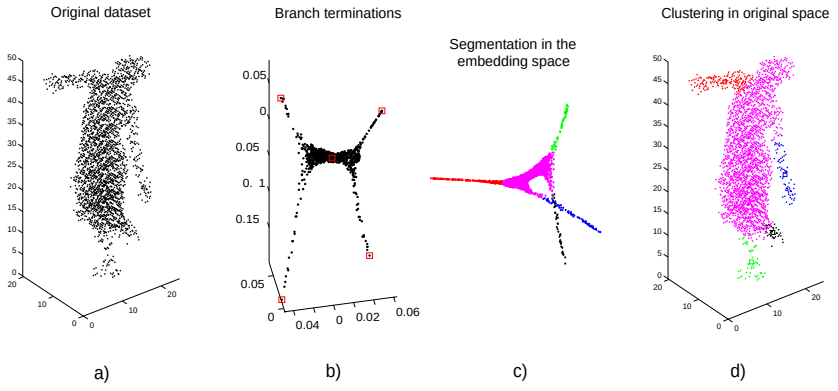


Fig. 5. Graphical illustration of the segmentation algorithm

- if $t = 0$, we use all branch terminations as seeds;
- if $t > 0$, the seeds are derived from the old centroids $\{c'_j(t), j = 1, ..., n(t-1)\}$ after the splitting/merging procedure exposed in Section 3.4.

4. This yields a new set of centroids $\{c_j(t), j = 1, .., n(t)\}$.
5. The labeling of the embedded points induces a segmentation in the original 3D shape (Figure 5-d).
6. All cluster centroids $c_j(t)$ are remapped to 3D (3.3), the corresponding 3D centroids $x_{i_j}(t)$ are added to the new dataset $X(t+1)$ at time $t+1$ (Figure 4).

5 Experiments

To test our spectral approach to body-part segmentation, we analyzed its performance on a large number of sequences acquired through our acquisition system composed by 8 synchronized cameras. Silhouettes were processed in order to compute first their visual hull, and the moving 3D articulated body was finally rendered as a uniformly sampled voxelset (Figure 6). We applied the algorithm to several different high-resolution sequences, one of which a 200-frame-long sequence capturing a dancer who moves and swirls all around the scene.

Figure 6 illustrates some typical results of the dynamic segmentation algorithm of Section 4, for a number of sequences. It can be appreciated that segmentation along time turns out to be pretty consistent, yielding very smooth cluster trajectories, non only in situations where body-parts are well separated (Figure 6-a,-b) but also during walking gaits (c) or even extremely complicated motions like the dance performed in (Figure 6-d,-e). In particular, in the "danceuse" case several topology transitions take place, due to the presence of a moving scarf around the waist of the woman, and numerous contacts between legs and arms during the dancer's performance. The algorithm segments the sequence in subsequences with constant topology (e.g. 6-d,-e), within which clusters are smoothly tracked. Another example of such a temporal segmentation in a different sequence is illustrated in 6-f).

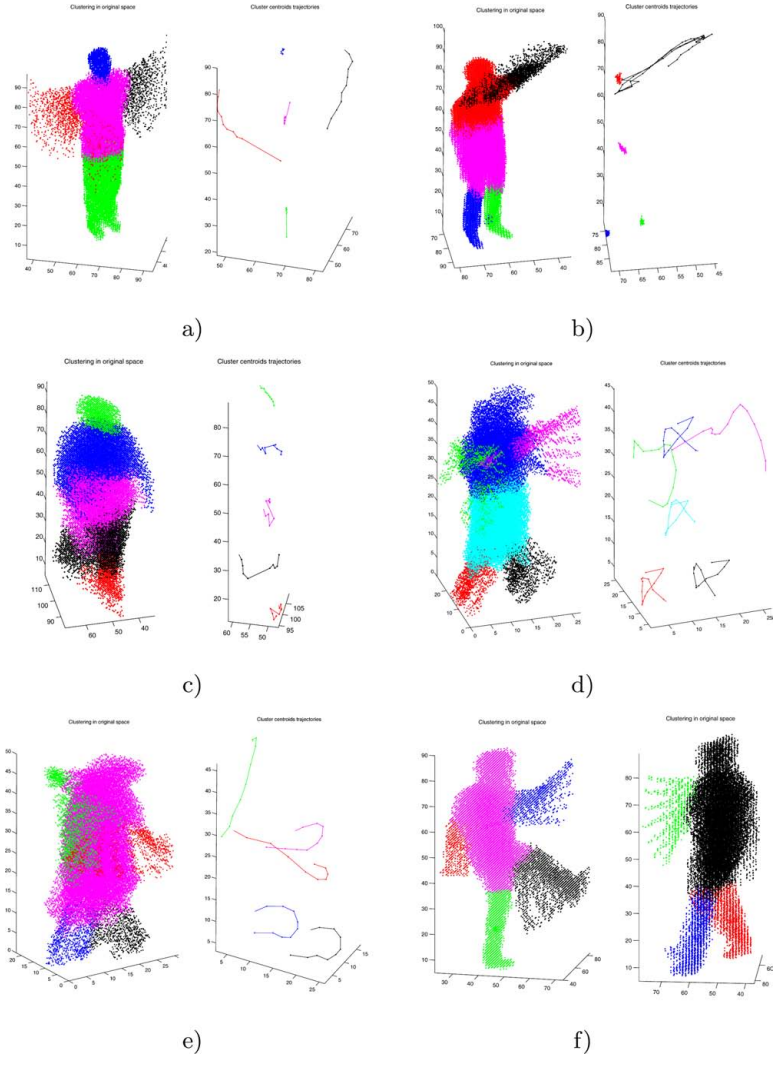


Fig. 6. Examples of the results produced by the dynamic segmentation algorithm of Section 4 on a number of sequences. a) a "fly" sequence of 11 frames. b) "arm-waving" sequence of length 50. c) "walking" motion, length 10. d) a subsequence of "danceuse", 16 frames. e) another "danceuse", length 10. Both whole clusters' evolution and their centroids' trajectories are shown. f) Topology transition management: after 11 frames in which arms are spread out (left) the left arm gets in contact with the torso: the algorithm adapts the number of clusters accordingly and proceeds to segment in a smooth way for other 7 frames (right, from a different viewpoint).

5.1 A Comparison with Direct EM Clustering in 3D

In order to show the advantages of our methodology over direct clustering in 3D we compared these results with those of a scheme similar to that of Section 4 in which old seeds and weights are passed to the next frame in order to make the segmentation coherent along time, but k-means or k-wise clustering in the embedding space are replaced by straightforward EM clustering of the original 3D shape. The idea is to model the probability density of the data as a convex combination of (typically Gaussian) components (which can be seen as clusters) $f(y) = \sum_j w(j) f_j(y)$, $\sum_j w(j) = 1$, $f_j(y) \sim N(\mu_j, \Sigma_j)$ whose parameters are estimated through the EM algorithm [27]. Figure 7 shows the resulting segmentation

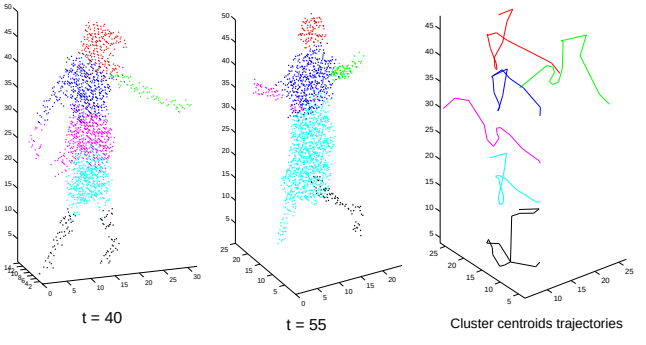


Fig. 7. Behavior of dynamic segmentation based on EM clustering. Even in presence of seed propagation, clusterings in different instants of a sequence (here $t = 40$ and $t = 55$ of "danceuse") are not consistent. This is confirmed by apparent irregularities in all centroids' trajectories (right).

on the same sequence of Figure 6-d, for some sample frames $t = 40, 45, 50, 55$, as an example of its typical performance. Besides spanning different body-parts at the same time (since clustering is based on the original Euclidean distance), clusters evolve inconsistently along time. A comparison of the related cluster trajectories with Figure 6-d highlights this irregular behavior.

5.2 Dimension of the Embedding Space

The quality of the segmentation depends on the stability of the embedded shape, which is in turn affected by the parameters of the embedding, like the dimension of the embedding space (i.e. the number of eigenvectors we selected after SVD of the affinity matrix M [19]). A better segmentation performance can be in general observed when we set as dimension of the embedding space a number similar to the expected number of clusters. Figure 8 shows, for a given frame ($t = 59$ of the "dancer" sequence) and an equal number of neighbors $k = 20$, the different segmentations we obtain for $d = 3$ (top) and $d = 4$ (bottom). Even though the scarf is clearly visible in the 3D cloud, a three-dimensional embedding space

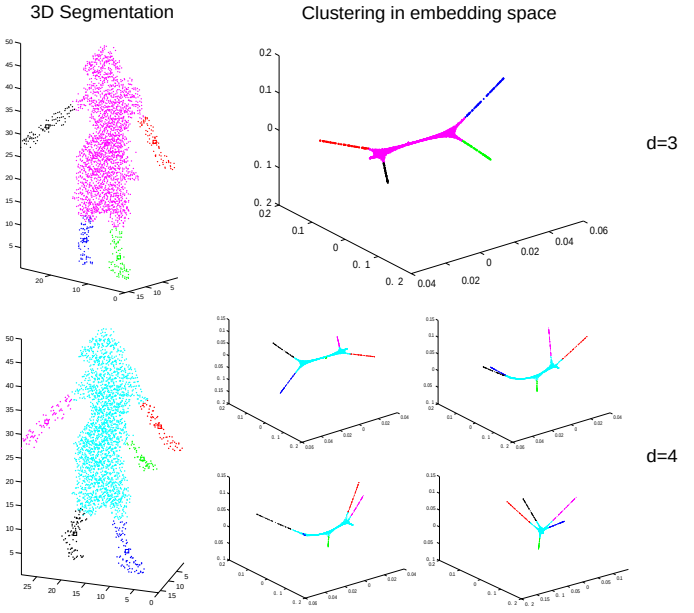


Fig. 8. Different segmentations of the same pose obtained for different dimensions of the embedding space. Top: for $d = 3$ the scarf merges with the torso into the same branch of the embedded cloud (top-right). Bottom: for $d = 4$ a separate branch associated with the scarf (in green) is present in the embedding shape.

fails to reveal its presence as separated branch of the embedded cloud (Figure 8-top-right). Such a new (green) branch appears instead in the four-dimensional embedding space: Figure 8-bottom-right displays the 4 orthogonal projections of the embedded cloud for $d = 4$ (with axes, x, y, z, w) onto its three-dimensional subspaces (x, y, z), (x, y, w), (x, z, w), (y, z, w).

5.3 Estimating the Optimal Number of Neighbors

The most critical parameter for the entire segmentation algorithm is however the number of neighbors k of the LLE step, as it affects the stability of the embedded shape from which the estimation of the number of clusters depends. It can be empirically noticed that, while the embedded shape shows a remarkable stability for some values of k this is not in general the case for arbitrary such values. It is then desirable to *estimate a variable number of neighbors in time*, in order to guarantee the stability of $Y(t)$ and ensure a consistent segmentation along time (Figure 9). For too large values of k some neighborhoods of points in a given body-part can comprise regions of a different body-part (middle). These "anomalous" neighborhoods are characterized by the fact that their farthest elements (as they belong to another, distinct link) are relatively distant from all others elements (which instead lie all on the same rigid part). If we then plot

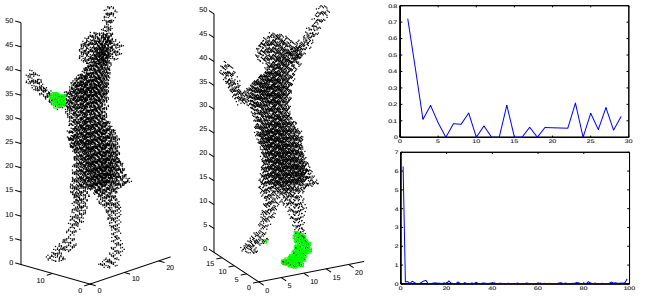


Fig. 9. How to estimate the correct number of neighbors k in the LLE algorithm. Non-admissible values of k are characterized by "anomalous" neighborhoods which span distinct body-parts (middle), in opposition to admissible values (left). The corresponding distance plots are visible to the right.

the distance between the farthest point of the neighborhood and all its fellows we notice a large jump (right-bottom).

This is not the case for neighborhoods which span a single rigid part (left). It is natural to choose as correct k any of those values which yield only "regular" neighborhoods (for instance the average of the interval of admissible values).

5.4 Robustness to Topology Changes

In any case, moments in which different parts of the articulated body come to contact still have important effects on the shape of $\{Y_i\}$. Figure 10 illustrates how the dynamic clustering technique copes with such changes. These events have dramatic consequences on embeddings based on measuring geodesic distances along the body, since new paths appear affecting in general the distance between

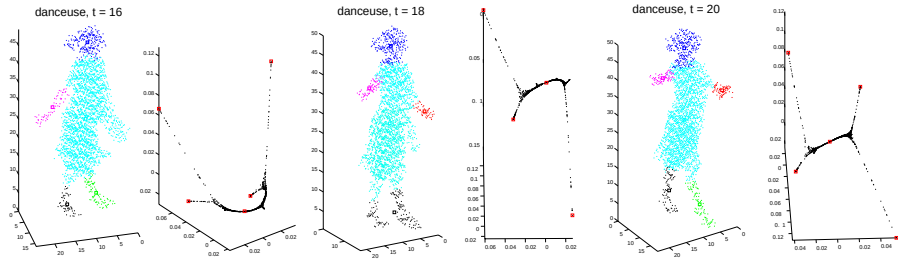


Fig. 10. How the splitting/merging algorithm deals with topology changes in the embedded space. At $t = 16$ the left arm of the dancer touches her scarf (left), and a single cluster covers body, arm, and scarf. Then ($t = 18$) the left arm becomes visible and a new cluster is assigned to it, while the dancer's feet get too close to be distinguished (middle). Finally ($t = 20$) her legs widen again, inducing a separate cluster for each one of them (right).

all pairs of points in the original cloud. Figure 11 shows how ISOMAP (as a representative of all geodesic-based spectral methods) copes with the transitions of Figure 10. Clustering is performed in the ISOMAP space by k-means.

Besides not having the desirable behavior in terms of branch separation of LLE or Laplacian Eigenmaps, geodesic spectral methods prove incapable of handling those situations, which are extremely common in any natural articulated motion. Even though we left the algorithm free to estimate the "best" number of clusters along the sequence, EM proved also incapable of adapting itself to the mutated topology of the body. As a result, clusters would "shift" around the body in a rather unstable way, failing to segment in a consistent way moving body-parts.

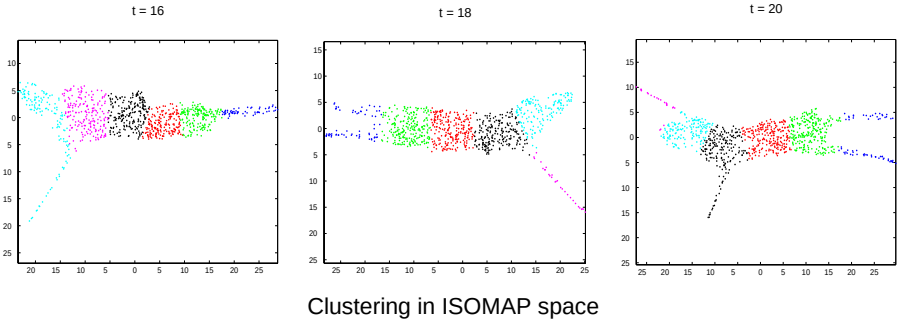


Fig. 11. Behavior of ISOMAP along the sequence of topology transitions of Figure 10. The shape of the embedded cloud (we chose $d = 2$ for sake of readability) changes dramatically, gravely affecting the segmentation in the original 3D space.

6 Conclusions

In this paper we presented a novel dynamic segmentation scheme in which moving articulated bodies are clustered in an embedding space, and clusters propagated in time to ensure temporal consistency. Exploiting some desirable features of LLE, in particular, we proposed a systematic way of estimating the optimal number of clusters in order to merge/split clusters in correspondence of topology transitions. To ensure stability and improve segmentation performance we proposed a method to learn the critical parameter k of the embedding algorithm. We compared the performance of the algorithm with similar propagation schemes based on direct EM clustering in 3D, and k-means clustering in ISOMAP space.

It is natural to imagine this unsupervised segmentation procedure as a building block of more detailed motion analysis, in which kinematic or stick models are fitted to the data based on the obtained segmentation. Even though the reconstructed segments are not in general associated with "natural" body-parts (as we assume no model of the moving body is available) temporal consistency as assured by our propagation scheme guarantees the coherence of the rough stick model which corresponds to those segments. The latter can be seen as a first guess of the underlying kinematic model, which could be later improved by

exploting 3D point matching schemes based on shape alignment in the embedding space [28]. Once obtained trajectories for each point of the cloud we could indeed easily cluster them according to the similarity of the associated motions, distinguish this way distinct rigid links belonging to the same initial segment, eventually achieving a finer bottom-up model of the body.

In a different context, as they are inherently invariant with respect to the direction of the motion, cluster centroids may provide good features to use for action recognition, for instance by feeding them to a classical hidden Markov model. We plan to explore both those opportunities in the near future.

References

1. Moeslund, T., Hilton, A., Krüger, V.: A survey of advances in vision based human motion capture and analysis. Computer Vision and Image Understanding 103(2-3), 90–126 (2006)
2. Hogg, D.: Model based vision: a program to see a walking person. Image and Vision Computing 1(1), 5–20 (1983)
3. Gavrila, D., Davis, L.: 3-D model-based tracking of humans in action: A multi-view approach. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, USA, pp. 73–80. IEEE Computer Society Press, Los Alamitos (1996)
4. Deutscher, J., Blake, A., Reid, I.: Articulated body motion capture by annealed particle filtering. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head Island, USA, vol. 2, pp. 126–133 (2000)
5. Brand, M.: Shadow puppetry. In: Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece, vol. 2, pp. 1237–1244 (1999)
6. Grauman, K., Shakhnarovich, G., Darrell, T.: Inferring 3D structure with a statistical image-based shape model. In: Proceedings of the 9th International Conference on Computer Vision, Nice, France, pp. 641–648 (2003)
7. Elgammal, A., Lee, C.: Inferring 3D body pose from silhouettes using activity manifold learning. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Washington, USA, pp. 681–688. IEEE Computer Society Press, Los Alamitos (2004)
8. Peursum, P., Venkatesh, S., West, G.: Tracking-as-recognition for articulated full-body human motion analysis. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, USA, pp. 1–8. IEEE Computer Society Press, Los Alamitos (2007)
9. Cheung, G., Kanade, T., Bouguet, J.Y., Holler, M.: A real time system for robust 3D voxel reconstruction of human motions. In: Proceedings of CVPR 2000, pp. 2714–2720 (2000)
10. Mukasa, T., Nobuhara, S., Maki, A., Matsuyama, T.: Finding articulated body in time-series volume data. In: Perales, F.J., Fisher, R.B. (eds.) AMDO 2006. LNCS, vol. 4069, pp. 395–404. Springer, Heidelberg (2006)
11. de Aguiar, E., Theobalt, C., Magnor, M., Theisel, H., Seidel, H.P.: M3: Marker-free model reconstruction and motion tracking from 3D voxel data. In: Cohen-Or, D., Ko, H.S., Terzopoulos, D., Warren, J. (eds.) PG 2004. 12th Pacific Conference on Computer Graphics and Applications, Seoul, Korea, pp. 101–110. IEEE Computer Society Press, Los Alamitos (2004)

12. Brostow, G.J., Essa, I., Steedly, D., Kwatra, V.: Novel skeletal representation for articulated creatures. In: Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic, vol. 3, pp. 66–78 (2004)
13. Chu, C.W., Jenkins, O.C., Mataric, M.J.: Markerless kinematic model and motion capture from volume sequences. In: CVPR 2003. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 475–482. IEEE Computer Society Press, Los Alamitos (2003)
14. Sundaresan, A., Chellappa, R.: Segmentation and probalistic registration of articulated body models. In: Proceedings of the 18th International Conference on Pattern Recognition, Hong Kong, vol. 2, pp. 92–96 (2006)
15. Jenkins, O., Mataric, M.: A spatio-temporal extension to isomap nonlinear dimension reduction. In: Proceedings of the 31th International Conference on Machine Learning, Alberta, Canada (2004)
16. Lin, R., Liu, C.B., Yang, M.H., Ahuja, N., Levinson, S.: Learning nonlinear manifolds from time series. In: Proceedings of the 9th European Conference on Computer Vision, Graz, Austria, vol. 2, pp. 245–256 (2006)
17. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems 14, MIT Press, Cambridge (2002)
18. Tenenbaum, J.B., Silva, V.d., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290(5500), 2319–2323 (2000)
19. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
20. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
21. Heiser, W.J., Bennani, M.: Triadic distance models: Axiomatization and least squares representation. J. Math. Psy. 41, 189–206 (1997)
22. Hayashi, C.: Two dimensional quantification based on the measure of dissimilarity among three elements. Ann. I. Stat. Math. 24, 251–257 (1972)
23. Agarwal, S., Lim, J., Zelnik-Manor, L., Perona, P., Kriegman, D., Belongie, S.: Beyond pairwise clustering. In: Proceedings of CVPR, vol. 2, pp. 838–845 (2005)
24. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. PAMI 22(8), 888–905 (2000)
25. Ng, M.J.A., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems 14: Proceedings of the 2001 (2001)
26. Bengio, Y., Paiement, J.F., Vincent, P.: Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. Technical report, Universite' de Montreal (2003)
27. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Royal Stat. Soc. 39, 1–38 (1977)
28. Mateus, D., Cuzzolin, F., Boyer, E., Horaud, R.: Articulated shape matching by locally linear embedding and orthogonal alignment. In: Proceedings of the ICCV'07-NTRL Workshop, Rio de Janeiro, Brasil (2007)

Articulated Object Registration Using Simulated Physical Force/Moment for 3D Human Motion Tracking

Bingbing Ni, Stefan Winkler, and Ashraf Kassim

Department of Electrical and Computer Engineering
National University of Singapore
Singapore 117576
{g0501096, elews, eleashra}@nus.edu.sg

Abstract. In this paper, we present a 3D registration algorithm based on simulated physical force/moment for articulated human motion tracking. Provided with sparsely reconstructed 3D human surface points from multiple synchronized cameras, the tracking problem is equivalent to fitting the 3D model to the scene points. The simulated physical force/moment generated by the displacement between the model and the scene points is used to align the model with the scene points in an Iterative Closest Points (ICP) [1] approach. We further introduce a hierarchical scheme for model state updating, which automatically incorporates human kinematic constraints. Experimental results on both synthetic and real data from several unconstrained motion sequences demonstrate the efficiency and robustness of our proposed method.

Keywords: Iterative Closest Points, 3D Registration, Articulated Human Motion Tracking, Simulated Physical Force/Moment, Kinematic Constraints.

1 Introduction

Multiple view based, marker-less articulated human body tracking has attracted a growing research interest in the computer vision community through the last decade. This is because of the large number of potential applications such as motion capture, human computer interaction, virtual reality, smart surveillance systems etc. Due to the high dimensionality of human body motion, the 3D tracking problem is inherently a difficult problem.

Image based methods were the first to be introduced in literature. Gavrilla and Davis [2] project a kinematic model onto each image plane and search for the best fit between the projected model and the image contours. Delamarre and Faugeras [3] create forces between the 3D human model and the detected image contours of the moving person to achieve their alignment.

Instead of deterministic search algorithms, stochastic search algorithms are more likely to guarantee a global optimum. Deutscher [4] propose an annealed

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 212–224, 2007.

particle filter approach for full body motion tracking. With as little as 100 particles, they can track full body motion in a stochastically optimal way. These particles are weighted by the similarity score between the extracted human contours and the synthetic silhouettes of the human model. Other methods based on belief propagation [5] were introduced to reduce the computational complexity.

To address the self-occlusion problem, various 3D reconstruction-based algorithms have been proposed. Cheung [6] introduce a shape-from-silhouette method for full body tracking from both silhouette and color information. Based on the results of visual hull reconstruction, they segment the surface points of the human body into rigid moving body parts using color information. Thus the 3D articulated human body tracking problem is reduced to first estimating the rigid motion of each body part independently and then imposing constraints between the connected body parts.

Cheung [7] also introduce a method based on volumetric reconstruction. Using an efficient voxel reconstruction technique, they fit ellipsoid models of human body parts into the reconstructed voxels in real-time. However this ellipsoid fitting approach fails when two body parts are too close to distinguish.

In this paper we present a registration-based 3D articulated human body tracking method. The inputs to our system are sparsely reconstructed human surface points. We have developed a simulated physical force/moment based 3D registration method that performs in an Iterative Closest Points (ICP) flavor. Tracking is achieved by registering our human model to the reconstructed 3D points in each frame. Our algorithm incorporates different constraints of human motion kinematics in an automatic way, which makes the registration procedure more flexible. Our experiments show both the efficiency and robustness of our proposed method.

The most related work to ours is performed by Delamarre and Faugeras [8], in which they apply similar ICP approach to track the 3D hand motion. However their work differs from ours in several ways. First, their inputs are dense reconstruction of the hand. Second, they use recursive dynamics algorithms to update the model state while we introduce a simple hierarchical model state updating scheme. Further more, no quantitative comparisons between their results and the ground truth data are given.

The paper is organized as follows: Section 2 gives a brief introduction of our human model. The 3D registration method based on simulated physical force/moment is presented in detail in Section 3. Section 4 shows our tracking results for several human motion sequences with discussions. Section 5 concludes the paper.

2 3D Human Model

As shown in Fig. 1, the human body is represented by a combination of 10 cylinders. The torso can be regarded as a degenerate cylinder since it has an elliptical cross-section. Although more sophisticated tapered cylinders or super quadrics could be employed as models, our experiments show that the simple

cylinder model is adequate for our tracking algorithm in terms of accuracy, and it also has the advantage of lower computational cost. For each part except the torso, a local coordinate frame is defined with the origin at the base of the cylinder. These origins also correspond to the center of rotation of each body part. The global coordinate system originates at the center of the torso. All body parts and corresponding parameters are indexed from 0 to 9 (cf. Fig. 1).

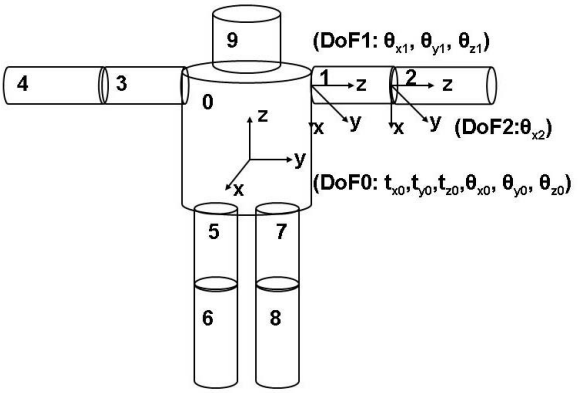


Fig. 1. 3D human model

Human kinematic knowledge is employed as a prior to define the degrees of freedom (DoF) for our human model. We incorporate 25 DoF: 3 DoF for upper arms, legs and head (rotating about their X, Y and Z axes), 1 DoF for lower arms and legs (they are only allowed to rotate about their X axes), and 6 DoF for the torso (global translation and rotation). With these definitions, the entire 3D pose of the body is determined by a 25D model state vector $\mathbf{s} = (t_{0x}, t_{0y}, t_{0z}, \theta_{0x}, \theta_{0y}, \theta_{0z}, \theta_{1x}, \theta_{1y}, \theta_{1z}, \theta_{2x}, ...)^T$, which are the joint angles of shoulders, elbows, hips, and knees, plus the global position and orientation of the torso. The objective of our registration task is to find such a 25D state vector for the model that best fits the scene points (i.e., minimize some distance function between the model and the scene).

To further constrain this high dimensional solution space as well as to eliminate the ambiguity during tracking, 2 types of motion constraints are imposed:

1. **Kinematic constraints:** The connectivity between adjacent body parts as well as the length constancy of the body parts are enforced through kinematic constraints. Each body part is only allowed to move according to its DoF (e.g., the lower arms are only allowed to rotate about their X axes).
2. **Joint angle limits:** For real human motion, the joint angles between adjacent body parts are within certain ranges (e.g., the elbow can only rotate around its X axis about 135 degrees). Therefore it would be necessary to incorporate this constraint to further reduce the solution space.

One property of our simulated physical force/moment based registration algorithm is that we can incorporate the above constraints automatically during iterations, which will be described in detail in the following section.

3 Registration Using Simulated Physical Force/Moment

Given a set of 3D surface points for each frame and a 3D human model composed of a set of connected body parts, the tracking problem is equivalent to a registration problem: find a suitable configuration of the state vector **s** to minimize some distance function between the human model and the 3D scene points. The distance metric we choose here is the average Euclidean distance between 3D scene points and their assigned parts of the human model (the point-cylinder distance):

$$D(M,S) = \frac{1}{N}\sum_{i} \|P_i - P'_{i,m(\pi(i))}\|^2 \tag{1}$$

$$\pi(i) = argmin_j \|P_i - P'_{i,m(j)}\|^2, j = 0,1,...,9 \tag{2}$$

where M and S denote the model and the set of scene points respectively; N is the number of scene points; $m(j)$ denotes the j^{th} model part; P_i is a 3D scene point and $P'_{i,m(j)}$ is its corresponding (closest) point on the j^{th} model part; $\pi(i)$ is the index of the model part which P_i is assigned to (i.e., with the closest distance).

3.1 Iterative Closest Points (ICP)

The well-known iterative closest points (ICP) algorithm is commonly used to coarsely align a rigid model with 3D scene points in an iterative manner:

1. For each 3D scene point, find its closest point on the model; calculate the local displacement between the model and the scene point.
2. Estimate the transform by integrating the local displacement over the entire object.
3. Apply the transform to the model.
4. Repeat above steps until convergence (i.e., the displacement between the scene and model is small or the consecutive estimated updating transforms are negligible).

The original version of the ICP algorithm is only designed for rigid models. Recently some variations of the ICP algorithm were proposed to deal with articulated objects. Demirdjian [9] presents a method that first applies ICP to each rigid body part, then linearly projects each independent transform into the solution space (i.e., the solution space that constrains the connectivity of the articulated body parts). Knoop [10] model several different types of joint constraints to enforce the articulated property after estimating the transform for each part independently. These independent transform estimation/solution

space projection based methods can easily become stuck in local minima. We therefore present a novel ICP-flavored, simulated physical force/moment based registration method, which iteratively aligns the model with the 3D scene points in a global and hierarchical style.

3.2 Registration

Our basic idea comes from the observation of the physical world. Suppose a displacement between a scene point and its closest point on the model creates a simulated physical force. This force generates two effects on the model: translation velocity and angular moment to pull/rotate the model into the alignment with the 3D scene point. Fig. 2 illustrates the force/moment created by the displacement between a scene point and its closest point on the model.

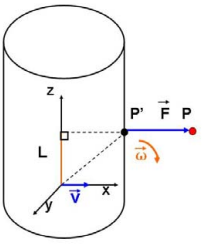


Fig. 2. Translation and rotation created by the simulated physical force/moment

The force can be expressed as $\overrightarrow{F} = \overrightarrow{P'P}$, where P is the scene point and P' is its closest point on the model. The moment is denoted as $\overrightarrow{M} = \overrightarrow{F}L$, where L is the vertical distance from force $\overrightarrow{F}$ to the rotation center. The magnitude of the translation and rotation vectors generated is proportional to the magnitude of the physical force/moment, namely: $\overrightarrow{v} = \rho\overrightarrow{F}$ and $\overrightarrow{\omega} = \lambda\overrightarrow{M}$, where ρ and λ are some small coefficients.

As in the ICP, we iteratively compute the closest points and then update the model state according to the estimated transform. During each iteration step, the displacements between all 3D scene points and the model are calculated, and all forces and moments are summed up, resulting in a translation and a rotation vector to align the model with the 3D scene points:

$$(\delta t_x, \delta t_x, \delta t_z)^T = \sum_i \overrightarrow{v_i} = \sum_i \rho\overrightarrow{F_i} \tag{3}$$

and

$$(\delta\theta_x, \delta\theta_x, \delta\theta_z)^T = \sum_i \overrightarrow{\omega_i} = \sum_i \lambda\overrightarrow{M_i} \tag{4}$$

Here $\overrightarrow{F_i}$ and $\overrightarrow{M_i}$ are the simulated physical force and moment created by the scene point P_i. With enough iterations, the misalignment between the model and the

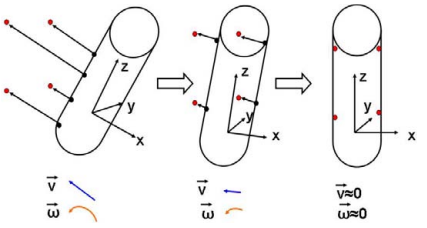


Fig. 3. Alignment procedure between the model and the scene points

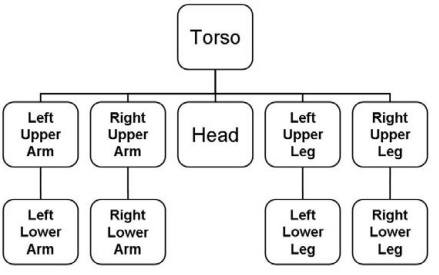


Fig. 4. Human model hierarchy tree

3D scene points will be minimized, and the overall physical force/moment will be balanced, indicating convergence. Fig. 3 illustrates the alignment procedure between the model and the scene points.

Given a set of articulated cylinder model parts, we start with assigning each scene point to its closest model part. However, instead of applying the above method to each body part independently, we adopt a hierarchical approach for applying the transform to the human model, which has the following intuition: Suppose a physical force is applied to the right lower arm of the model; this force will not only create the angular moment for the right lower arm to rotate around the elbow, but will also contribute to the right upper arm's rotation about the shoulder, as well as the global rotation and translation of the torso. Our hierarchical updating approach is consistent with this key observation. The human model will be treated as a hierarchy tree with its root at the torso, and it has sub-trees rooted at the right and left upper arms, the right and left upper legs, and the head. Fig. 4 illustrates the hierarchy of the human model.

When estimating the transform associated with a certain body part, the physical forces applied to all body parts in its group will be integrated. For example, when calculating the global translation and rotation $(\delta t_{0x}, \delta t_{0y}, \delta t_{0z}, \delta\theta_{0x}, \delta\theta_{0y}, \delta\theta_{0z})^T$, the forces applied to all body parts will be counted as follows:

$$\begin{pmatrix} \delta t_{0x} \\ \delta t_{0y} \\ \delta t_{0z} \end{pmatrix} = \sum_{j} \sum_{\overrightarrow{F_i} \in m(j)}^{i} \lambda_{j0} \overrightarrow{F_i} \tag{5}$$

and

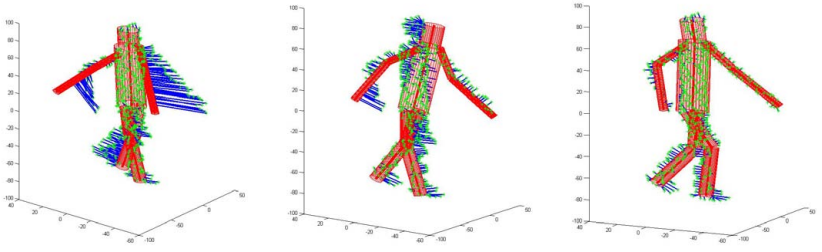
$$\begin{pmatrix} \delta\theta_{0x} \\ \delta\theta_{0y} \\ \delta\theta_{0z} \end{pmatrix} = \sum_{j} \sum_{\overrightarrow{M_i} \in m(j)}^{} \rho_{j0} \overrightarrow{M_i} \tag{6}$$

where λ_{j0} and ρ_{j0} $(j = 0, 1, 2...9)$ are some weighting factors.

Similarly, when estimating the rotation $(\delta\theta_{1x}, \delta\theta_{1y}, \delta\theta_{1z})^T$ of the right upper arm about the right shoulder (there would be no translation for the right upper arm as defined by its DoF), the physical forces applied to the right upper arm and right lower arm will be counted:

$$\begin{pmatrix} \delta\theta_{1x} \\ \delta\theta_{1y} \\ \delta\theta_{1z} \end{pmatrix} = \sum_{i}^{\overrightarrow{M_i} \in m(1)} \rho_{11} \overrightarrow{M_i} + \sum_{i}^{\overrightarrow{M_i} \in m(2)} \rho_{21} \overrightarrow{M_i} \tag{7}$$

We further concatenate the transform vectors estimated for each body part to obtain the 25D updating vector $\delta\mathbf{s} = (\delta t_{0x}, \delta t_{0y}, \delta t_{0z}, \delta\theta_{0x}, \delta\theta_{0y}, \delta\theta_{0z}, \delta\theta_{1x}, \delta\theta_{1y}, \delta\theta_{1z}, \delta\theta_{2x}...)^T$. Obviously the DoF of each body part is preserved and the articulated structure of the human model is maintained implicitly in this updating scheme.



(a) Iteration 4 (b) Iteration 6 (c) Iteration 10

Fig. 5. Example of the iterative registration procedure for our human model (The scene points are plotted by green stars. The simulated forces between the model and the scene points are represented by blue lines. The human model is represented by red cylinders.)

Furthermore, the kinematic constraints and joint angle limits can be incorporated in this framework automatically. Given the original state vector $\mathbf{s}$ and the updating vector $\delta\mathbf{s}$ generated by our registration algorithm, we can clamp the new state vector to avoid the violation of any constraint by the following inequality:

$$\mathbf{s}_{lb} \leq \mathbf{s} + \delta\mathbf{s} \leq \mathbf{s}_{ub} \tag{8}$$

where $\mathbf{s}_{lb}$ and $\mathbf{s}_{ub}$ are the lower and upper bounds of the joint angles.

Fig. 5 illustrates a few iterations of a typical registration procedure for our human model by this hierarchical updating scheme.

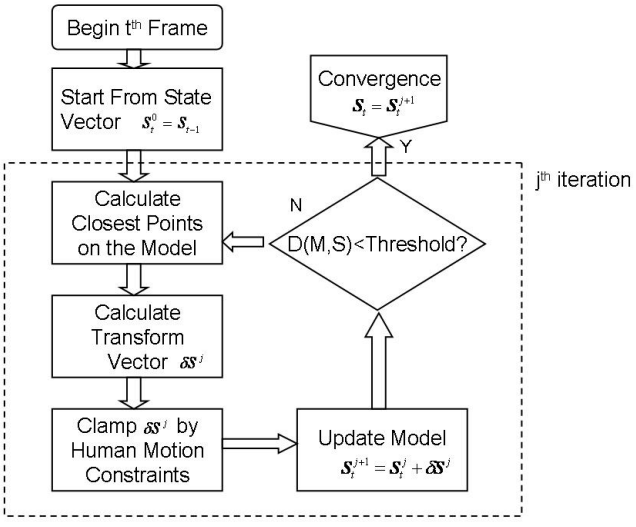


Fig. 6. A diagram for the tracking procedure

3.3 Tracking

The tracking procedure is straightforward: given a new frame of reconstructed scene data, we start from the registered state of last frame $\mathbf{s}_{t-1}$ and iteratively apply the simulated physical force/moment based registration algorithm until convergence. We choose the convergence criteria as follows: The distance function between the model and the scene is smaller than some threshold, and the estimated state transforms for consecutive iteration steps become negligible. Fig. 6 gives a detailed diagram of the tracking procedure.

4 Results and Discussion

We validate our proposed method using both synthetic and real data. In total about 1000 frames from both synthetic and real sequences are tested. For the synthetic data, the 3D surface points are generated by the 3D modeling software package Maya, combined with the motion sequences provided by the CMU motion capture database [11]. Each of the sequences spans hundreds of frames during which the subject performs unconstrained, fully articulated motions such as walking and dancing. Gaussian noise is added to the 3D coordinates of the original data points to simulate reconstruction errors. We then compare the joint angles estimated by our method with the ground truth from the database.

Several examples of the tracking results of the synthetic sequences are shown in Figs. 7 and 8. It can be observed that our registration/tracking algorithm performs well for most poses. In Fig 8(a), the two upper arms are tightly coupled with the torso, which makes them difficult to distinguish, however our algorithm

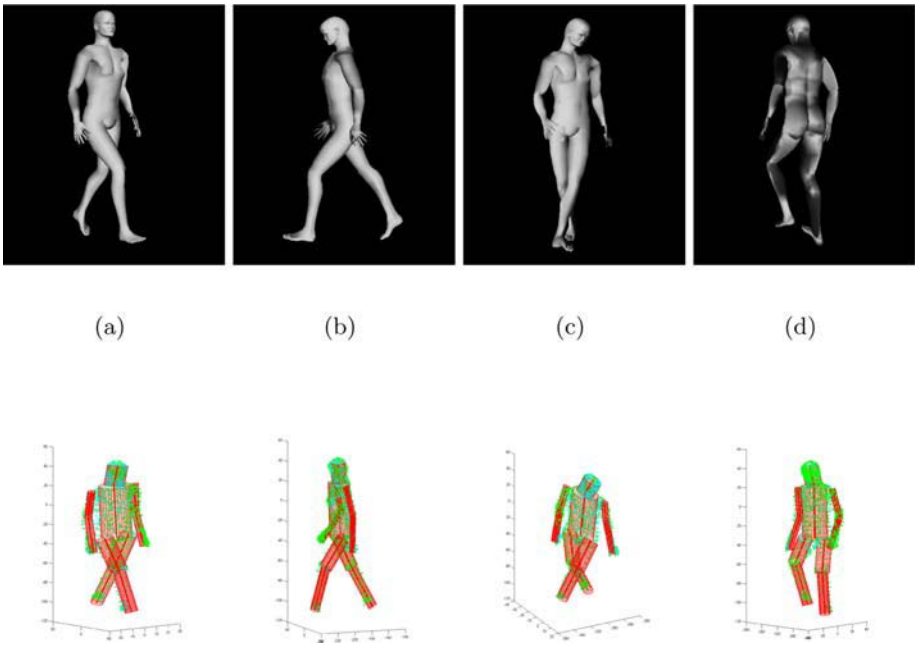


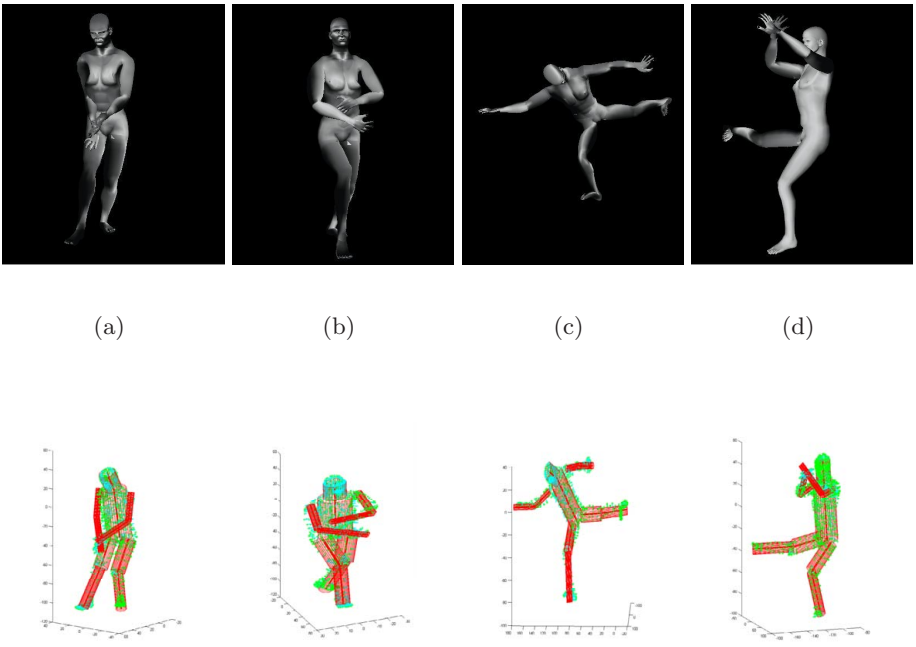
Fig. 7. Examples of the tracking results of the walking sequence. The top row shows the synthetic scenes created with Maya, from which the 3D surface points are sampled. The bottom row shows the corresponding registration results. The scene points are plotted as green stars, and the human model is represented by red cylinders.

performs properly under this scenario. Another case of robustness is demonstrated in Fig 8(b), where the arms are folded very close to each other, yet the pose is correctly tracked. Besides its reliable performance under such conditions, our registration/tracking algorithm is also flexible when it comes to more unusual human poses; Figs. 8(c) and 8(d) show such examples.

Examples of the tracking results of a real motion sequence are shown in Fig. 9. This sequence shows a female dancer performing various movements. Using 7 synchronized cameras surrounding the scene, the human surface points are obtained by visual hull reconstruction [12]. Although ground truth data is not provided, we can still evaluate the results qualitatively. While the tracking results are good overall, the loose dress of the dancer causes problems in the reconstruction accuracy in some cases.

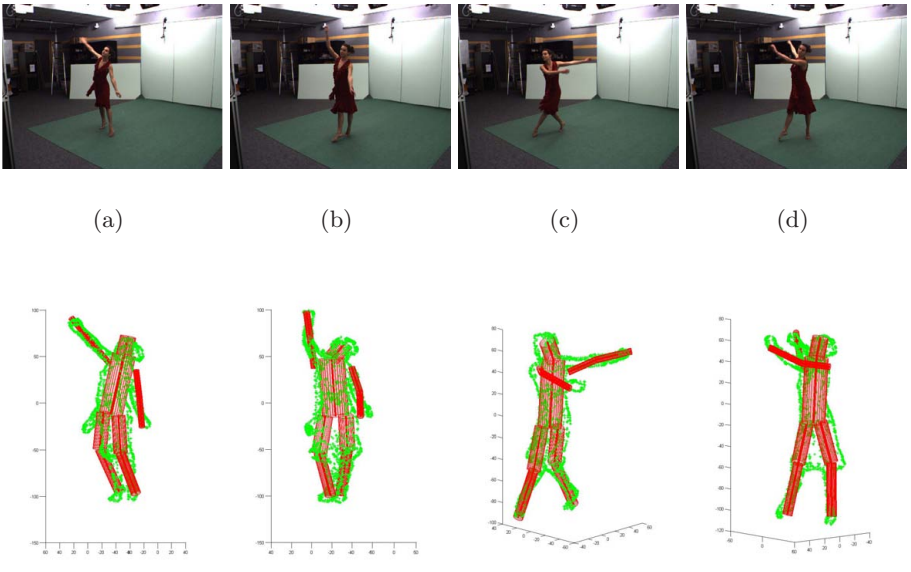
Fig. 10 compares the ground truth with the estimated joint angles in the walking and dancing sequences. It can be observed that in the walking sequence our estimated joint angles of the left knee follow the periodical motion of the left leg very accurately.

Fig. 11 shows the histograms of the root mean squared error (RMSE) for the different sequences. The average RMSE of the estimated joint angles for both sequences are about 8 and 16 degrees, respectively. The larger errors in



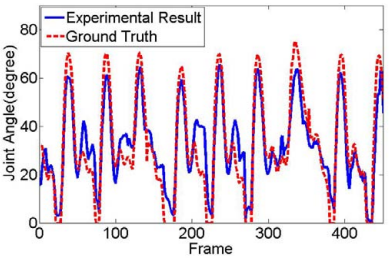
(a) (b) (c) (d)

Fig. 8. Examples of the tracking results of the dancing sequences

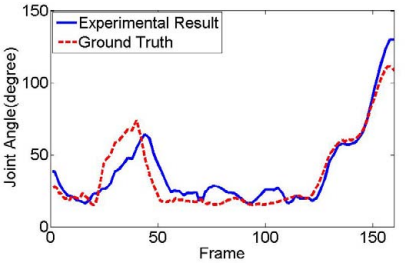


(a) (b) (c) (d)

Fig. 9. Examples of the tracking results of the real sequence

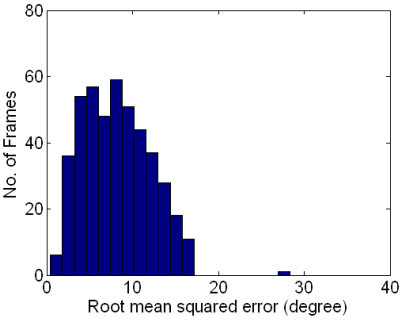


(a) Joint angle of the left knee in the walking sequence.

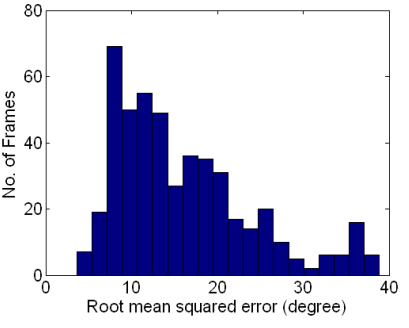


(b) Joint angle of the right elbow in the dancing (1) sequence.

Fig. 10. Comparison between the ground truth and experimental results for two joints



(a) Walking sequence (450 frames, average RMSE=8.2 degrees).

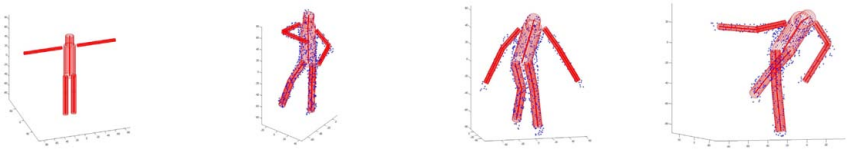


(b) Dancing sequences (480 frames, average RMSE=16.1 degrees).

Fig. 11. RMSE for synthetic sequences

the dancing sequences are mainly due to the coarseness of our human model; for example, the subject's upper torso bends forward, but our model fits the upper/lower torso as a single rigid body part, resulting in estimation errors. Lack of minor extremities (e.g., hands and feet) also induces further estimation errors. One can notice in Fig. 5(c) that the points of the feet pull the lower legs slightly away from their actual positions. In our future work, we will refine our human model to avoid this kind of errors, i.e., split the torso into two parts, add parts for feet and hands etc.

In normal cases when the body parts are not close to each other (i.e., arms and legs are stretched out), a roughly aligned pose is adequate for accurate initialization even if the displacements between the scene points and human model are large. Fig. 12 shows 3 initialization results starting from the initial model pose on the left.However, for some poses, especially when two body parts



(a) (b) (c) (d)

Fig. 12. Examples of the initialization results. The left figure shows the initial pose of the model. The other 3 figures show the converged results of the initialization. The scene points are plotted as blue stars, and the human model is represented by red cylinders.

are too close to distinguish or have sharp joint angles, the initialization can become trapped in a local minimum. To address this problem, various methods can be used. We are investigating 3D shape context [13,14] to coarsely detect the human posture in the initial frame. The 3D shape descriptor calculated from the reconstructed points of the first frame will be compared with all candidates in a small pre-generated shape descriptor database. The configurations with the best matching scores can then be used to initialize the model state vector.

5 Conclusions

We have introduced a simulated physical force/moment based 3D registration method, which we have applied to articulated human motion tracking. We also presented a hierarchical model state updating scheme that incorporates different human kinematic constraints in an automatic way. Our experiments on sequences of unconstrained human motion show the robustness and effectiveness of our proposed method.

Acknowledgments. The synthetic motion key data used in this project was obtained from mocap.cs.cmu.edu. The multiple-video data used here are from INRIA Rhone-Alpes multiple-camera platform Grimage and PERCEPTION research group. The database is available at https://charibdis.inrialpes.fr.

References

1. Besl, P., McKay, H.: A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14(2), 239–256 (1992)
2. Gavrila, D., Davis, L.: 3-D model-based tracking of humans in action: A multi-view approach. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, pp. 73–80. IEEE Computer Society Press, Los Alamitos (1996)

3. Delamarre, Q., Faugeras, O.: 3D articulated models and multi-view tracking with silhouettes. In: Proc. IEEE International Conference on Computer Vision, Corfu, Greece, vol. 2, pp. 716–721. IEEE Computer Society Press, Los Alamitos (1999)
4. Deutscher, J., Blake, A., Reid, I.: Articulated body motion capture by annealed particle filtering. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, SC, USA, vol. 2, pp. 126–133. IEEE Computer Society Press, Los Alamitos (2000)
5. Han, T., Huang, T.: Articulated body tracking using dynamic belief propagation. In: Sebe, N., Lew, M.S., Huang, T.S. (eds.) Proc. IEEE International Workshop on Human-computer Interaction. LNCS, vol. 3766, pp. 26–35. Springer, Heidelberg (2005)
6. Cheung, K., Baker, S., Kanade, T.: Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin, USA, vol. 1, pp. I–77–I–84 (2003)
7. Cheung, K., Kanade, T., Bouguet, J., Holler, M.: A real time system for robust 3D voxel reconstruction of human motions. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, SC, USA, vol. 2, pp. 714–720. IEEE Computer Society Press, Los Alamitos (2000)
8. Delamarre, Q., Faugeras, O.: 3D articulated models and multi-view tracking with physical forces. Computer Vision and Image Understanding 81(2), 328–357 (2001)
9. Demirdjian, D.: Enforcing constraints for human body tracking. In: Proc. Workshop on Multi-Object Tracking (2003)
10. Knoop, S., Vacek, S., Dillmann, R.: Modeling joint constraints for an articulated 3D human body model with artificial correspondences in ICP. In: Proc. IEEE-RAS International Conference on Humanoid Robots, Tsukuba, Japan, pp. 74–79 (December 2005)
11. Lab, C.G.
12. Franco, J., Boyer, E.: Exact polyhedral visual hulls. In: Proc. British Machine Vision Conference (2003)
13. Belongie, S., Malik, J.: Matching with shape context. In: Proc. IEEE Workshop on Content-based Access of Image and Video Libraries, Hilton Head, SC, USA, pp. 20–26. IEEE Computer Society Press, Los Alamitos (2000)
14. Kortgen, M., Park, G., Novotni, M., Klein, R.: 3D shape matching with 3D shape context. In: Proc. 7th Central European Seminar on Computer Graphics (April 2003)

An Ease-of-Use Stereo-Based Particle Filter for Tracking Under Occlusion

Ser-Nam Lim* and Larry Davis

[1] Cognex Corp., Natick, MA, USA
[2] CS Dept., University of Maryland, College Park, Maryland, USA

Abstract. We describe a tracker that handles occlusion by clustering foreground pixels based on their disparity values. Stereo-matched foreground pixels, mapped from multiple views to a reference view, allow the tracker to recover occluded foreground regions in the reference view. The stereo algorithm utilizes a common plan view into which foreground regions from multiple views are projected and intersected to construct polygons that contain the ground plane locations of objects, followed by constraining the epipolar search to only those pixels with ground plane locations lying within these polygons. Consequently, a stereo-matched foreground pixel is easily mapped between views by first mapping its ground plane location using pre-computed homography, followed by intersecting the vertical axis passing through the mapped location with the epipolar line. Finally, tracking in a reference view allows a particle filter to be "seamlessly" integrated, so that uncertainties can be effectively dealt with. Experimental results illustrate the effectiveness of our algorithm.

Keywords: Surveillance, Detection, Tracking, Stereo, Multi-camera fusion.

1 Introduction

Many surveillance tasks, such as detection and tracking, become particularly challenging in the presence of occlusion. Specifically, in the presence of occlusion, images of multiple objects are frequently segmented as a single foreground region, causing surveillance tasks to fail. To effectively deal with problems caused by occlusion, both single camera and multiple camera algorithms have been proposed.

Single-camera approaches include [2], which assumed that targets are sufficiently isolated from one another so that their appearances can be modeled and then utilized to segment them subsequently under occlusion condition. Another single-camera approach was described in [22]. The algorithm uses 3D human shape models together with an efficient Markov Chain Monte Carlo method to segment humans in crowded scenes. [20] described an algorithm that utilizes three object states: before, during and after occlusion. They assumed that the trajectory of each individual object is similar to the entire group during occlusion. So, by tracking and labeling each individual object before and after occlusion, and tracking the entire group during the occlusion, the complete trajectory of each object can be recovered. [23] described using motion layer estimation to determine depth ordering of foreground layers. The

* This research was funded in part by the U.S. Government VACE program.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 225–239, 2007.

algorithm uses a Maximum A Posteriori framework to simultaneously update the motion layer parameters, the ordering parameters, and the background occluding layers.

In contrast to a single-camera approach, a multi-camera approach can be more effective since occluded regions in the view of one camera may be visible in the view of another suitably placed camera. [13] described a region-based stereo algorithm to find 3D points inside an object, together with a Bayesian classification scheme that assigns priors for different objects to each pixel. It uses a top view of the objects to combine results obtained from each camera. [10] also described an algorithm that projects the vertical axes of segmented foreground regions in different views onto a top view. Intersections between these projections are possible ground points and are utilized to refine foreground segmentations in a particle filter framework. The algorithm, however, relies on getting good initial segmentations using a color-based human appearance model. Another algorithm that utilizes such ground plane constraints is described in [9], where foreground regions from different views are mapped to a reference view using precomputed homographies. Then, given a sufficient number of views, the feet regions are expected to accumulate the most "votes", allowing one to separate a crowded scene into individual object based on these segmented feet regions. Once the feet regions are found, the height is calculated by moving up the connected foreground region before background is encountered. [8] introduced a unified framework that deals with long periods of occlusion, tracking across non-overlapping views and updating appearance models for tracked objects over time. The method suspends tracking in areas where objects are likely to be occluded, or in between non-overlapping views. Tracking is then resumed by matching "suspended objects" using full kinematic motion models and Gibbsian distributions for object appearance.

Most of these algorithms employ intensity-based change detection for foreground extraction. [15] avoided doing so by utilizing a Kanade-Lucas tracker [12] to construct trajectories of features, which are then integrated with a learned object descriptor to achieve motion segmentations of individual objects. Alternatively, disparity-based methods can be used to overcome the problems of using intensity-based change detection, whereby foreground detections are augmented with disparity computation to eliminate noise in the foreground regions. One such disparity-based tracker was described in [21]. The algorithm uses a mixture of single and stereo cameras, each of which performs object tracking independently. These tracks are then combined to maintain identity of each individual object across different views.

Short periods of occlusion can also be effectively overcome with a particle filter [1,7] if tracks can be accurately initialized for objects in the scene. Given observed tracks of an object, a particle filter predicts the future location of each tracked object and combines the predictions with subsequent measurements.

1.1 Overview of Our Approach

Our algorithm consists of three steps for detecting and tracking people. The first step employs an algorithm described in [19]. It projects foreground silhouettes detected in different views into a common plan view. Such a plan view is "artificially" constructed as a top-view image and point correspondences are drawn to compute ground plane

homographies between itself and different camera views. Projected regions belonging to different views are then intersected, generating a number of polygons. These polygons represent possible ground plane locations of objects in the scene. However, some of these polygons are invalid (phantom polygons) while others may have resulted from the projections of foreground silhouettes comprising multiple objects. Polygons are mapped between different views using pre-computed homographies.

To estimate the true locations of objects in the scene, we first try to compute disparities of foreground pixels in every view. Each foreground pixel's disparity is computed based on a plane+parallax measure described in [4,5,6,16], with the epipolar search in a second view constrained to those pixels whose ground plane locations fall within the polygons. In another word, the stereo algorithm selects pixels within the polygons that are close to the vertical axis[1] of the foreground pixel in the first view, and maps each of these pixels to a second view using pre-computed homography. The vertical axis passing through the mapped position intersects the epipolar line to generate a candidate for the conjugate pixel. The selected ground plane location is then the one that generates a conjugate pixel with the best color match. We will refer to these ground plane locations as ground plane pixels.

The computed disparities allow foreground pixels to be easily clustered into individual objects. Pixels lying in the polygons that are not valid ground plane locations are removed, in effect eliminating phantom polygons. While the resulting clusters are good estimates of the image positions of the objects, the extent of each cluster does not accurately represent the true image extent of the corresponding object since only visible foreground pixels have been processed. We overcome this by sensor fusion, combining pairs of foreground pixels and their ground plane pixels that have been constructed by different stereo pairs in a reference view[2]. This generates, for each object, a set of candidate bounding boxes in the reference view. We determine the best weighted combination of these bounding boxes during tracking by employing a particle filter [7].

1.2 Comparing Other Ground Plane Based Trackers

The utilization of a common plan view is not uncommon; several trackers [9,10,13,19] also make use of similar constraints for tracking under occlusion. It is important, therefore, to comparatively identify our contributions, several of which are due to the stereo algorithm we employ. Firstly, in contrast to [13], we require only minimal calibration in the form of ground plane homography and epipolar geometry (both of which can be pre-computed, in practice, accurately using point correspondences between a given pair of images obtain from two views) to map a stereo-matched foreground pixel from different views to a reference view due to the ease of mapping its ground plane pixel. This is an important feature of our stereo algorithm; it avoids error-prone and cumbersome camera calibration that would otherwise be necessary. Tracking in a "centralized" reference view in turn allows the full image extent of the objects to be tracked as opposed

[1] The vertical axis connects the foreground pixel to the camera's vanishing point computed with respect to the ground plane.

[2] The reference view is the view in which we segment foreground regions into indvidual objects.

to tracking just the top view profile in the plan view, and, most importantly, facilitates the recovering of occluded regions.

Furthermore, by recovering disparity values of occluded regions in a reference view (in which a pixel can thus be assigned multiple disparity values belonging to different objects in the same line of sight) and performing disparity-based clustering, it is unnecessary to initialize the appearance of objects in order to segment them. Such an approach was seen in [10]. It assumes that each object is initially well-isolated so that its appearance can be modeled based on color and eventually used for segmentation in each view. The disadvantages of doing so, however, include (1) the difficulties with which the tracker can be fully automated in terms of acquiring these color models, (2) the problems faced by the tracker in discriminating similar-colored objects, and (3) the applicability of the acquired color models during tracking as the objects vary in pose and scale. A third tracker, described in [9], also utilizes a similar procedure to intersect foreground regions in a common plan view so as to identify foot regions. However, it attempts to grow the full extent of each object by moving up connected foreground region beginning from the foot regions - a procedure that potentially yields the wrong height whenever there is another object behind the object of interest. Finally, [19], which first suggested the intersection of foreground regions on a common plan view, is essentially different from the rest of the ground plane based trackers in its use of an iterative approach to eliminate phantom polygons.

2 Polygon Construction

We begin by constructing polygons that represent possible ground plane locations of objects in the scene by performing background subtraction [18] in each view. The basic idea is to project foreground regions in each view into a common plan view, as shown in Figure 1, and construct polygons formed from the intersections of these view-specific projections.

In each view, foreground pixels are first clustered using connected component analysis. The leftmost, ℓ, and rightmost, r, pixel of each foreground region are connected by straight lines to the vertical vanishing point, v. The lines are then projected into the plan view using pre-computed homography, H_{plan}. The resulting region is bounded by lines L_ℓ and L_r in the plan view defined by[3]:

$$\begin{aligned} L_\ell &= H_{plan} * \ell \times H_{plan} * v, \\ L_r &= H_{plan} * r \times H_{plan} * v, \end{aligned} \tag{1}$$

and the scene boundaries.

Finally, the projected regions for different views are intersected. These intersections include some "phantom" polygons (colored in red) and valid polygons (colored in green), as illustrated in Figure 1. Phantom polygons do not contain valid objects and are merely geometric artifacts of the region-intersection procedure. In the next section, we will describe a disparity computation step that typically eliminates phantom polygons.

[3] We will use the notation $*$ to represent multiplication and $\times$ to represent cross product.

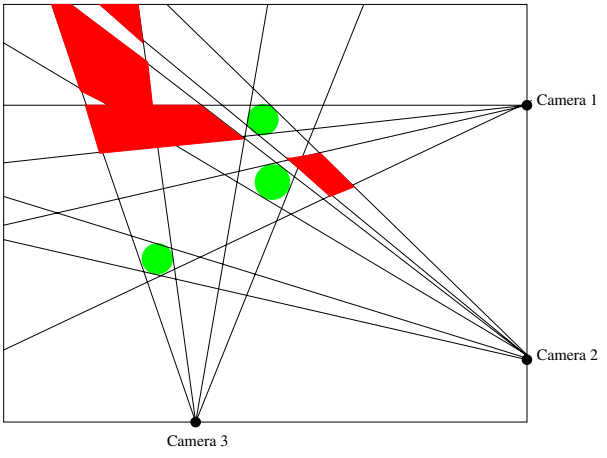


Fig. 1. Projections of the foreground silhouettes into a common plan view. The green circles represent projections in the plan view that contain people while the red regions are phantom polygons.

3 Disparity Computation

The constructed polygons are mapped from the plan view to each camera view, resulting in regions of candidate ground plane pixels. Figure 2 illustrates the mapping of the polygons in Figure 1 to a camera's view.

The subsequent steps of the algorithm are illustrated in Figure 3. Given a foreground pixel (top of the person in the example, but in general any foreground pixel), ρ, in the first view, we seek to determine its conjugate pixel, ρ', in a second view. ρ' lies on the corresponding epipolar line [3], L_e, of ρ. Instead of searching "everywhere" for ρ' along L_e, we constrain the search to those pixels along L_e belonging to the set S:

$$S = \{(p \times v') \times L_e | p \in S_p\}, \tag{2}$$

where v' is the vertical vanishing point of the second camera and S_p is:

$$S_p = \{p | p \in S_{poly} \bigcap \|H_{first}^{-1} * p - L_\rho\| < T\}. \tag{3}$$

Here, S_{poly} is the set of pixels lying in the constructed polygons, L_ρ is the vertical axis of ρ in the first view, H_{first} is the ground plane homography from the first to second view, and T is a threshold value for the perpendicular distance of $H_{first}^{-1} * p$ to L_p.

Intuitively, we are constraining the stereo search to pixels on L_e, whose ground plane pixels are close to ρ's vertical axis. Moreover, by construction, these ground plane pixels must lie in the constructed polygons. Phantom polygons are generally removed since pixels in them are typically not selected in the stereo matching step. In addition, due to occlusion, conjugate pixels with color similarity below a pre-specified threshold are not considered.

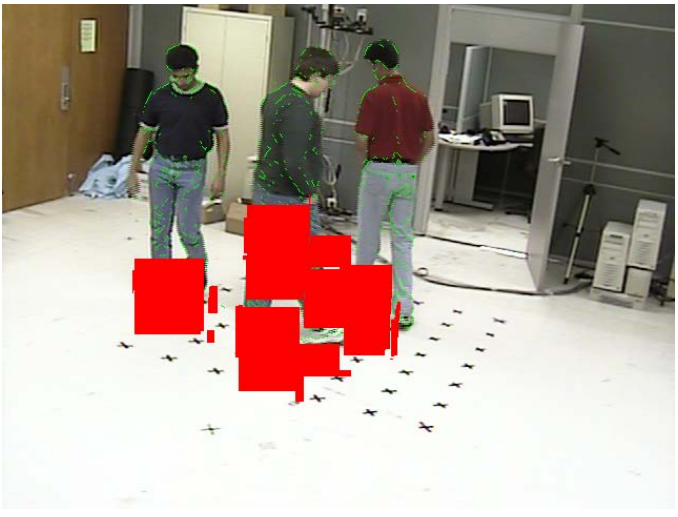


Fig. 2. The polygons in Figure 1 mapped to one of the views, shown as red rectangular image regions. The green pixels are the silhouette pixels.

We then assign a disparity value to each stereo-matched foreground pixel. For this purpose, given a pair of conjugate pixels (ρ, ρ'), and homography, H_ρ, that maps ρ to ρ', we use a plane+parallax measure given as:

$$\|\rho - H_\rho^{-1} * \rho'\|. \tag{4}$$

Such a measure is associated with a scale factor that depends on the stereo pair being used. To maintain a consistent disparity measure, the reference view is paired with a fix second view, to which stereo-matched foreground pixels from other stereo pairs are always mapped before Equation 4 is applied. Following this, foreground pixels are clustered based on their disparities, which is usually effective for separating merged foreground regions into their constituent individuals. To further improve accuracy, the clustering also utilizes the Euclidean distance between a ground plane pixel and the reference camera's vertical vanishing point when projected onto the plan view, which provides a projected depth, as shown in Figure 4. Note, however, that this is not the "true" depth.

It is also important to point out that since the stereo algorithm computes intersections between vertical axes and epipolar lines, degeneracies can occur when they have similar gradients. This happens when both cameras of a stereo pair are positioned vertically with respect to the ground plane, and should be avoided. If the cameras' positions can be chosen offline, strategies that maximize "horizontal" inter-camera distance can be employed.

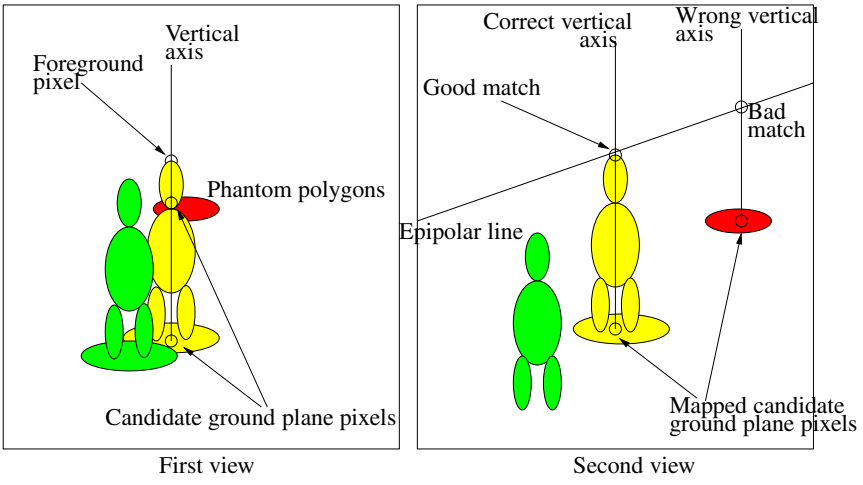


Fig. 3. Determining the conjugate pixel of the foreground pixel in the first view. Candidates for the foreground pixel's ground plane pixel lie in the constructed polygons and close to its vertical axis in the first view. Referring to the second view, a wrong candidate's vertical axis in the second view will intersect the epipolar line at a pixel that will likely be a poor match to the foreground pixel. On the other hand, the intersection of the correct candidate's vertical axis and the epipolar line in the second view will generate a good match.

4 Sensor Fusion

Occluded regions of an object will cause its image size to be underestimated, which we overcome with sensor fusion. We construct for each stereo pair a set of stereo-matched foreground pixels and their ground plane pixels. We then map every stereo-matched foreground pixel, ρ', and its ground plane pixel, g', to (ρ, g) in the reference view:

$$\begin{aligned} g &= H_{ref}^{-1} * g', \\ \rho &= L_g \times L_{\rho'}. \end{aligned} \tag{5}$$

H_{ref} is the ground plane homography mapping g to g', L_g is the vertical axis of g and $L_{\rho'}$ is the epipolar line of ρ' in the reference view. Consequently, as long as the 3D point corresponding to ρ' is visible in one or more stereo pairs, then even if it is occluded in the reference view, it can still be recovered.

After sensor fusion, each pixel in the occluded regions is associated with multiple disparity values, corresponding to different foreground layers. Clustering based on these disparity values helps to recover the true extents of occluded objects. We illustrate the result of sensor fusion in Figure 5, where the disparity map in (a) could only separate the visible foreground regions. After performing sensor fusion, we show in (b) the bounding boxes that correctly recovered the occluded regions.

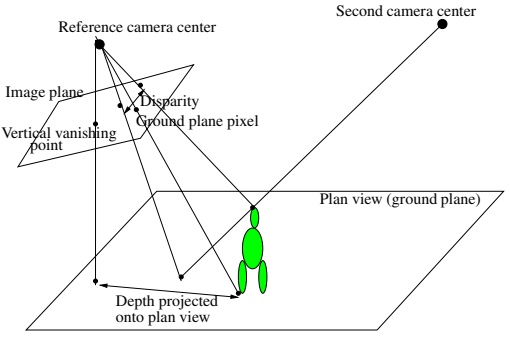


Fig. 4. We map the camera's vanishing point and ground plane pixel to the plan view, compute the distance between the projected points, and use it for clustering foreground pixels. This is in addition to using Equation 4, which is computed by mapping the conjugate pixel in the second view to the reference view with the homography, followed by measuring the image distance between the mapped location and the foreground pixel.

(a) Disparity map constructed with a pair of cameras.

(b) Result after sensor fusion.

Fig. 5. In (a), the disparity map only separates out the visible foreground pixels. In (b), the occluded regions are recovered from other stereo pairs, giving bounding boxes that correctly localize the three persons in the scene.

5 Stereo Pair Selection and Tracking

While the recovery of occluded object regions by sensor fusion is often effective, incorrect stereo matches can adversely affect performance. Consequently, during sensor fusion, blindly combining results from all stereo pairs could degrade performance. Such uncertainties can be alleviated by conducting sensor fusion in a particle filter [7], the

application of which is, again, made possible by the ease with which stereo-matched foreground pixels can be mapped to the reference view.

5.1 State Space

The state space of the particle filter at time step $t-1$ is $\{s_{t-1}^{(n)}, n = 1...N\}$, N being the number of particles and $s_{t-1}^{(n)}$ the set of foreground objects determined initially by the n^{th} stereo pair, but would eventually converge to the set determined by the n^{th} element of the N "strongest" particles. Each object is represented by the image coordinates of the upper-left and lower-right corners of its bounding box. Here, stereo-matched foreground pixels from each particle are first mapped, using Equation 5, to the reference view before the corresponding bounding box is drawn. Particles are assigned weights, $\pi_t^{(n)}$, that are used to combine them at the end of every iteration to arrive at an estimate of the tracked positions of the bounding boxes. A cumulative weight, $c_{t-1}^{(n)}$, is also used for importance sampling, so that we are more likely to select at time t the set of foreground objects, $s'^{(n)}_t$, determined by a stereo pair that has a larger cumulative weight (importance). Importance sampling is performed using the same approach given in [7].

5.2 Prediction

The tracker keeps track of the velocity of each bounding box in the reference view. These velocities are estimated at the end of each iteration and are modeled with M Gaussian distributions:

$$\{\langle \mu_1, \sigma_1^2 \rangle, ..., \langle \mu_M, \sigma_M^2 \rangle\}, \tag{6}$$

where M is the number of objects, and μ and σ^2 are the means and variances. We generate a prediction, $x_t = \{x_{t,1}, ..., x_{t,M}\}$, from $s'^{(n)}_t = \{s'^{(n)}_{t,1}, ..., s'^{(n)}_{t,M}\}$ as:

$$x_t = \{s'^{(n)}_{t,1} + \mu_1 * \Delta t, ..., s'^{(n)}_{t,M} + \mu_M * \Delta t\}, \tag{7}$$

with corresponding probability:

$$P(x_t | x_{t-1} = s'^{(n)}_t) = \prod_{i=1}^{M} P(\mu_i), \tag{8}$$

where $P(\mu_i)$ is given by the i^{th} Gaussian and Δt is a discrete time step.

5.3 Measurement

The measurement stage considers two issues - data association and changing number of objects as objects exit and enter the scene. Data association is performed for every selected particle, using Maximum Weight Matching on bipartite graphs [17]. The

matching algorithm employed is the Kuhn-Munkres algorithm [11], and the bipartite graph is constructed from the set of tracked objects and the set of objects that the particle newly detected, $z_t^{(n)} = \{z_{t,1}^{(n)}, ..., z_{t,M}^{(n)}\}$. Edges between the two sets are weighted based on grayscale histogram similarity [14] and overlap between the bounding boxes.

The weight for a particle, $\pi_t^{(n)} = \{\pi_{t,1}^{(n)}, ..., \pi_{t,M}^{(n)}\}$, is then updated based on the matching:

$$\begin{aligned} \pi_{t,i}^{(n)} &= P(z_{t,i}^{(n)} | x_{t,i} = s_{t,i}^{(n)}), \\ &= P(z_{t,i}^{(n)} - s_{t,i}^{(n)}). \end{aligned} \tag{9}$$

$P(z_{t,i}^{(n)} - s_{t,i}^{(n)})$ represents the probability of the deviation of the measured velocity, $(z_{t,i}^{(n)} - s'^{(n)}_{t,i})$, from the predicted velocity, $(s_{t,i}^{(n)} - s'^{(n)}_{t,i})$, which can be computed by a Normal difference distribution with zero mean and variance, $2\sigma_i^2$. Thus, Equation 9 becomes:

$$\pi_{t,i}^{(n)} = \frac{1}{2\sigma_i\sqrt{\pi}} \exp^{-\frac{((z_{t,i}^{(n)} - s_{t,i}^{(n)}))^2}{4\sigma_i^2}}. \tag{10}$$

μ_i and σ_i are obtained from Equation 6. We then perform normalization so that $\forall i$, $\sum_n \pi_{t,i}^{(n)} = 1$. To update the cumulative weights, we add $\prod_{i=1}^{M} \pi_{t,i}^{(n)}$ to $c_t^{(n)}$ at the end of every iteration.

5.4 Update

After constructing the N particles, sensor fusion is performed by combining the values of $\pi_{t,i}^{(n)}$ to compute the tracked position, $p_i(t)$, of the bounding box $i(= 1...M)$ at time t as:

$$p_i(t) = \sum_{n=1}^{N} \pi_{t,i}^{(n)} * s_{t,i}^{(n)}, \tag{11}$$

followed by updating its Gaussian with velocity $p_i(t) - p_i(t-1)$.

5.5 Bootstrapping

The tracker is fully automatic, and is bootstrapped with an initial set of objects detected using section 4. Without immediately invoking the particle filter, a few iterations of detection, followed by data association using the same graph matching in Section 5.3, is first performed. The particle filter is started once enough data has been gathered about the velocities of the objects, after which a newly arrived object is only added to it after been bootstrapped in the same manner. We summarize the algorithm in Algorithm 1.

Algorithm 1. Particle Filter($\{s_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1...N\}$, $\{\langle \mu_1, \sigma_1^2 \rangle, ..., \langle \mu_M, \sigma_M^2 \rangle\}$)

1: Let $\alpha = 0$ and increment time.
2: Select a particle, $s'^{(n)}_t$, from $s_{t-1}^{(n)}$ based on cumulative probability $c_{t-1}^{(n)}$.
3: Predict the new locations, $s_t^{(n)}$, of the tracked bounding boxes, using the locations of the bounding boxes in $s'^{(n)}_t$ as starting points and $\{\langle \mu_1, \sigma_1^2 \rangle, ..., \langle \mu_M, \sigma_M^2 \rangle\}$ as the velocities (Equation 7). Assign probabilities to these new locations based on $\{\langle \mu_1, \sigma_1^2 \rangle, ..., \langle \mu_M, \sigma_M^2 \rangle\}$ (Equation 8).
4: Perform disparity-based segmentation using the selected particle, obtaining a set of foreground objects and their bounding boxes, $z_t^{(n)}$. After performing data association, tracked objects that are not matched are removed. Each new object is assigned a newly initialized Gaussian distribution to model its velocity after boostrapping.
5: The probabilities of the deviations of the measured velocities from the predictions are determined from $\{\langle \mu_1, \sigma_1^2 \rangle, ..., \langle \mu_M, \sigma_M^2 \rangle\}$ (Equations 10). Assign these probabilities to $\pi_t^{(n)}$ and perform normalization.
6: Update $c_t^{(n)}$ with $\pi_t^{(n)}$.
7: Increment α.
8: **if** $\alpha < N$ **then**
9: Go to step 2.
10: **end if**
11: Determine the new tracked position of each bounding box as given by the sampled particles, weighted using $\pi_t^{(n)}$. Update $\{\langle \mu_1, \sigma_1^2 \rangle, ..., \langle \mu_M, \sigma_M^2 \rangle\}$ with the velocities in moving on to these new tracked positions. Go to step 1.

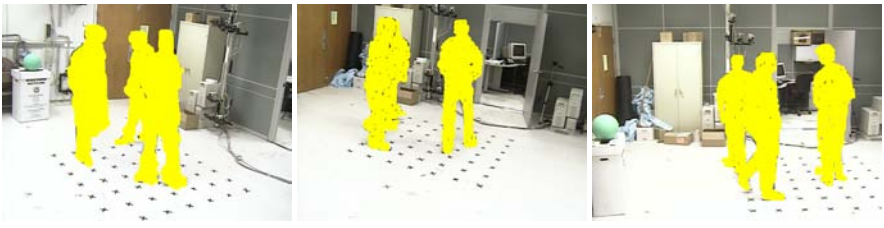
6 Implementation and Results

We tested our algorithm on a four-camera video sequence in which three persons were initially walking in circles in a small space, resulting in substantial occlusion. We illustrate the algorithm in Figure 6. We performed background subtraction in the views of four different cameras, shown in the second row. Foreground regions, projected into a common plan view, were interesected to obtain a set of polygons containing candidate ground plane pixels, shown as red regions in the third row. Using these candidate ground plane pixels, we obtain the set of disparity maps shown in the fourth row. Finally, tracking results based on these disparity maps are combined in the particle filter. We show the final result in the last row.

We also show in Figure 7 the performance of the tracker when an additional person walked into the scene, midway through the video sequence. The data association step correctly detects the new object, and the particle filter was able to maintain tracks of the objects throughout the video sequence. Finally, we compared the performance of our tracker to that in [10][4]. While its performance was reasonably good, we face two significant problems: (1) It was hard to properly initialize the color model of each object (which is manually performed with the source code provided), due to pose and scale issues, and (2) as shown in Figure 8, objects with similar appearance will cause confusion in the tracker when they come near each other.

[4] Thanks to the authors for providing the source code.

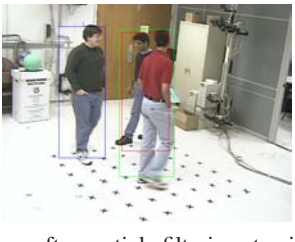
Camera 1, 2, 3, 4.



Detected foreground in camera 1, 2, 3.

Candidate ground plane pixels (red) in camera 1, 2, 3.

Disparity maps for stereo pair (1, 2), (2, 3) and (3, 4).



Final bounding boxes after particle filtering step in camera 1's view.

Fig. 6. Notice that the disparity maps could be incomplete due to occlusion and limited field of view, but are still useful for recovering occluded regions in other views

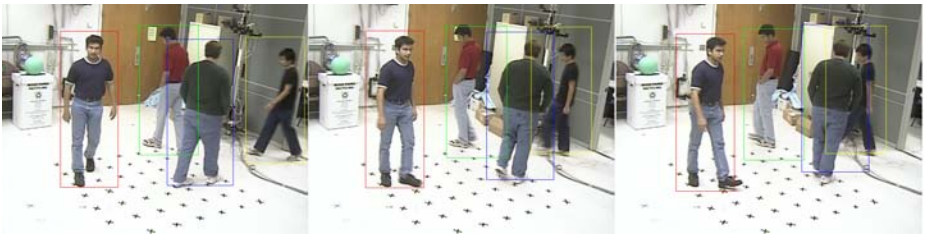


Fig. 7. We show here the tracking results (each tracked object was bounded by the same color box throughout the sequence) when an additional person walked into the scene in Figure 6. The particle filter was able to maintain the correct tracks throughout the video sequence.

Camera 1, 2, 3 and 4.

Left image was the result of our tracker while the right was obtained from the tracker in [10].

Fig. 8. In this four views, the tracker in [10] had difficulties segmenting the two individuals (in the red circles) with similar color profile, which is needed to invoke a particle filter for refining the locations of the vertical axes of objects. A large portion of the pixels lying in one of the individuals were incorrectly classify as belonging to the other. In contrast, stereo pair (2, 4) was used by our tracker to segment the two individuals with lesser errors. Both trackers were given the same set of foreground regions.

7 Concluding Remarks

We have presented a tracker that utilizes disparity-based sensor fusion to detect and track people under occlusion in a particle filter framework, of which several major contributions can be identified as follows:

Ease of Calibration. Due to the way information is transferred between different views, the tracker requires only homographies and vertical vanishing points that can be easily computed once offline. Specifically, ground plane homography can be computed with four point correspondences between two images and vertical vanishing point can be computed as the intersection of a given pair of lines in the image that corresponds to parallel lines in the world [3]. In contrast to a 3D tracker such as [13], which involves projecting 3D points onto the image plane, the calibration step requires 3D to 2D point correspondences to compute both intrinsic and extrinsic camera parameters. Such full-scale calibration is often more susceptible to errors and requires cumbersome setup.

Improved Stereo Matching. During tracking, the tracker does not search everywhere along the epipolar line, which is a major pitfall for generating wrong stereo matches. Instead, ground plane pixels lying in constructed polygons and near the vertical axis of a foreground pixel are mapped to the second view, after which the vertical axes of the mapped ground plane pixels are intersected with the epipolar line to generate better candidates for the corresponding pixel.

Probabilistic Sensor Fusion. The ease of information transferrence between different views simplifies sensor fusion, so that the tracker can be run in a particle filter framework. Doing so is instrumental for handling uncertainties (Equation 11), since the strength of each "particle" is considered in determining the tracked position.

Fully Automatic. The tracker is fully automatic, while still demonstrating similar (or better, in certain situations) performance when compared to a recent state-of-the-art tracker that also utilizes a common plan view and a particle filter but requires initializing an appearance model for every object.

References

1. Arulampalam, S., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking (2002)
2. Elgammal, A., Davis, L.S.: Probabilistic framework for segmenting people under occlusion. In: Proc. of IEEE 8th International Conference on Computer Vision, IEEE Computer Society Press, Los Alamitos (2001)
3. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000)
4. Irani, M., Anandan, P.: Parallax geometry of pairs of points for 3d scene analysis. In: European Conference on Computer Vision, pp. 17–30 (1996)
5. Irani, M., Anandan, P., Cohen, M.: Direct recovery of planar-parallax from multiple frames. In: Workshop on Vision Algorithms, pp. 85–99 (1999)
6. Irani, M., Anandan, P., Weinshall, D.: From reference frames to reference planes: Multi-view parallax geometry and applications. In: European Conference on Computer Vision, vol. 2, pp. 828–845 (1998)
7. Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. International journal of computer vision 29(1), 5–28 (1998)
8. Kaucic, R., Perera, A.G.A., Brooksby, G., Kaufhold, J., Hoogs, A.: A unified framework for tracking through occlusions and across sensor gaps. In: CVPR (2005)

9. Khan, S.M., Shah, M.: A multiview approach to tracking people in crowded scenes using a planar homography constraint. In: European Conference on Computer Vision, Graz, Austria (2006)
10. Kim, K., Davis, L.S.: Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3953, pp. 98–109. Springer, Heidelberg (2006)
11. Kuhn, H.W.: The hungarian method for the assignment problem. Naval Research Logistic Quarterly 2, 83–97 (1955)
12. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: International Joint Conference on Artificial Intelligence (1981)
13. Mittal, A., Davis, L.S.: M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene using region-based stereo. In: 7th European Conference on Computer Vision, Copenhagen, Denmark (June 2002)
14. Niblack, W., Barber, R., Equitz, W., Flickner, M., Glasman, E., Petkovic, D., Yanker, P., Faloutsos, C.: The qbic project: Querying images by content using color, texture, and shape. Storage and Retrieval for Image and Video Databases, SPIE 1908 (1993)
15. Rabaud, V., Belongie, S.: Counting crowded moving objects. In: CVPR (2006)
16. Sawhney, H.S.: 3d geometry from planar parallax. Computer Vision and Pattern Recognition, 929–934 (1994)
17. Tarabanis, K., Allen, P., Tsai, R.: A survey of sensor planning in computer vision. IEEE Transactions on Robotics and Automation 11(1), 86–104 (1995)
18. Grimson, W.E.L., Stauffer, C.: Adaptive background mixture models for real-time tracking. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (1999)
19. Yang, D.B., Gonzalez-Banos, Guibas, L.J.: Counting people in crowds with a real-time network of simple image sensors. In: Ninth IEEE International Conference on Computer Vision, IEEE Computer Society Press, Los Alamitos (2003)
20. Yang, T., Pan, Q., Li, J., Li, S.: Real-time multiple objects tracking with occlusion handling in dynamic scenes. In: CVPR (2005)
21. Zhao, T., Aggarwal, M., Kumar, R., Sawhney, H.: Real-time wide area multi-camera stereo tracking. In: CVPR (2005)
22. Zhao, T., Nevatia, R.: Bayesian human segmentation in crowded situations. In: CVPR (2003)
23. Zhou, Y., Tao, H.: A background layer model for object tracking through occlusion. In: ICCV (2003)

Semi-Latent Dirichlet Allocation: A Hierarchical Model for Human Action Recognition

Yang Wang, Payam Sabzmeydani, and Greg Mori

School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
{ywang12,psabzmey,mori}@cs.sfu.ca

Abstract. We propose a new method for human action recognition from video sequences using latent topic models. Video sequences are represented by a novel "bag-of-words" representation, where each frame corresponds to a "word". The major difference between our model and previous latent topic models for recognition problems in computer vision is that, our model is trained in a "semi-supervised" way. Our model has several advantages over other similar models. First of all, the training is much easier due to the decoupling of the model parameters. Secondly, it naturally solves the problem of how to choose the appropriate number of latent topics. Thirdly, it achieves much better performance by utilizing the information provided by the class labels in the training set. We present action classification and irregularity detection results, and show improvement over previous methods.

1 Introduction

Recognizing human actions from image sequences is a challenging problem in computer vision. It has applications in many areas, e.g., motion capture, medical bio-mechanical analysis, ergonomic analysis, human-computer interaction, surveillance and security, environmental control and monitoring, sport and entertainment analysis, etc. Various visual cues (e.g., motion [6,8,16,20] and shape [26]) can be used for recognizing actions. In this paper, we focus on recognizing the action of a person in an image sequence based on motion cues. We develop a novel model of human actions based on the "bag-of-words" paradigm.

Our model is motivated by the recent success of "bag-of-words" representation for object recognition problems in computer vision. The common paradigm of these approaches consists of extracting local features from a collection of images, constructing a codebook of visual words by vector quantization, and building a probabilistic model to represent the collection of visual words. While these models of an object as a collection of local parts are certainly not "correct" ones, for example they only model a few parts of objects and often ignore much structure, they have been demonstrated to be quite effective in object recognition tasks [9,12,15].

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 240–254, 2007.

In this paper we explore the use of a similar model, for recognizing human actions. Figure 1 shows an overview of our "bag-of-words" representation. In our model, each frame in an image sequence is assigned to a visual word by analyzing the motion of the person it contains. The unordered set of these words over the image sequence becomes our bag of words. As with the object recognition approaches, some structure has been lost by moving to this representation. However, this model is much simpler than one which explicitly models temporal structure. Instead we capture "temporal smoothing" via co-occurrence statistics amongst these visual words, i.e., which actions tend to appear together in a single track. For example, in a single track of a person, the combination of "walk left" and "walk right" actions is much more common than the combination of "run left" "run right" "run up" "run down". In this paper we provide evidence that this simple model can be quite effective in recognizing actions.

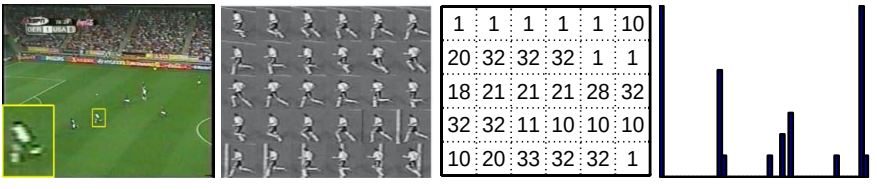


Fig. 1. The processing pipeline of getting the "bag-of-words" representation: (a) given a video sequence, (b) track and stabilize each human figure, (c) represent each frame by a "motion word", (d) ignore the ordering of words and represent the image sequences of a tracked person as a histogram over "motion words"

In particular, our model is based on the latent Dirichlet allocation (LDA) [2] model. LDA, the probabilistic Latent Semantic Indexing (pLSI) [13] model, and their variants have been applied to various computer vision applications, such as scene recognition [5,10], object recognition [11,22,25], action recognition [19], human detection [1], etc.

Despite the great success achieved, there are some unsolved, important issues remaining in this line of research. First of all, it is not clear how to choose the right number of latent topics in one of these models. Previous methods usually take a rather ad-hoc approach, e.g., by trying several different numbers. But this is often not possible in realistic settings. Secondly, most of the previous approaches use their models for some specific recognition problem, say object class recognition. However, there is no guarantee that the latent topics found by their algorithms will necessarily correspond to object classes. Thirdly, the features used in these approaches are usually SIFT-like local features computed at locations found by interest-point detectors. The only exceptions are histogram of oriented gradients in Bissacco et al. [1] and multiple segmentations in Russell et al. [22]. Features based on local patches may be appropriate for certain recognition problems, such as scene recognition or object recognition. But for human

action recognition, it is not clear that they can by sufficiently informative about the action being performed. Instead, we use descriptors that can capture the large-scale properties of human figures, and compare these results to approaches using local patches.

In this paper, we attempt to address the above mentioned issues in two aspects. First of all, we introduce a new "bag-of-words" representation for image sequences. Our representation is dramatically different from previous ones (e.g., Niebles et al. [19]) in that we represent a frame in an image sequence as a "single word", rather than a "collection of words" computed at some spatial-temporal interest points. Our main motivation for this new representation is that human actions may be characterized by large-scale features, rather than local patches. Secondly, we propose a new topic model called *Semi-Latent Dirichlet Allocation (S-LDA)*. The major difference between our model and the *Latent Dirichlet Allocation (LDA)* model is that some of the latent variables in LDA are observed during the training stage in S-LDA. We show that our model naturally solves the problem of choosing the right number of latent topics. Also by pushing the information provided by class labels of training data directly into our model, we can guide the latent topics to be our class labels, and consequently achieve much better performance.

The rest of this paper is organized as follows. In Sect. 2 we review previous work. Section 3 gives the details of our approach. We present experimental results in Sect. 4 and conclude in Sect. 5.

2 Previous Work

A lot of work has been done in recognizing actions from both still images and video sequences. Much of this work is focused on analyzing patterns of motion. For example, Cutler & Davis [6], and Polana & Nelson [20] detect and classify periodic motions. Little & Boyd [16] analyze the periodic structure of optical flow patterns for gait recognition. Rao et al. [21] describe a view-invariant representation for 2D trajectories of tracked skin blobs. Others consider the shape of human figure. For example, Sullivan & Carlsson [26] use "order structure" to compare the shape of extracted edges for the purpose of action recognition. There is also work using both motion and shape cues. For example, Bobick & Davis [3] use a representation known as "temporal templates" to capture both motion and shape, represented as evolving silhouettes. Zhong et al. [27] cluster segments of long video sequences by looking at co-occurrences of patterns of motion and appearance.

Our approach is closely related to a body of work on recognition using "bag-of-words". The "bag-of-words" model was originally proposed for analyzing text documents [2,13]. Recently, researchers in the computer vision community have used "bag-of-words" models for various recognition problems. Fei-Fei & Perona [10] use a variant of LDA for natural scene categorization. Sivic et al. [25], Fergus et al. [11] and Russell et al. [22] use pLSI for unsupervised object class recognition and segmentation. Niebles et al. [19] use pLSI for action recognition

using spatial-temporal visual words. Bissacco et al. [1] use LDA for human pose classification from vector-quantized words from histograms of oriented gradients.

3 Our Approach

Similar to Niebles et al. [19], we represent a video sequence as a "bag of words". But our representation is different from Niebles et al. [19] in two aspects. First of all, our method represents a frame as a single word, rather than a collection of words from vector quantization of space-time interest points. In other words, a "word" corresponds to a "frame", and a "document" corresponds to a "video sequence" in our representation. Secondly, our model is trained in a semi-supervised fashion. We will show that by utilizing the class labels, we can greatly simplify the training algorithm, and achieve much better recognition accuracy.

3.1 Motion Features and Codebook

We use the motion descriptor in Efros et al. [8] to represent the video sequences. This motion descriptor has been shown to perform reliably with noisy image sequences, and has been applied in various tasks, such as action classification, motion synthesis, etc.

To calculate the motion descriptor, we first need to track and stabilize the persons in a video sequence. We use the human detection method in Sabzmeydani & Mori [23] in some of our experiments. But any tracking or human detection methods can be used, since the motion descriptor we use is very robust to jitters introduced by the tracking.

Given a stabilized video sequence in which the person of interest appears in the center of the field of view, we compute the optical flow at each frame using the Lucas-Kanade [17] algorithm. The optical flow vector field F is then split into two scalar fields F_x and F_y, corresponding to the x and y components of F. F_x and F_y are further half-wave rectified into four non-negative channels F_x^+, F_x^-, F_y^+, F_y^-, so that $F_x = F_x^+ - F_x^-$ and $F_y = F_y^+ - F_y^-$. These four non-negative channels are then blurred with a Gaussian kernel and normalized to obtain the final four channels Fb_x^+,Fb_x^-,Fb_y^+,Fb_y^- (see Fig. 2).

The motion descriptors of two different frames are compared using a version of the normalized correlation. Suppose the fours channels for frame A are a_1, a_2, a_3 and a_4, similarly, the four channels for frame B are b_1, b_2, b_3 and b_4, then the similarity between frame A and frame B is:

$$S(A,B) = \sum_{c=1}^{4} \sum_{x,y \in I} a_c(x,y) b_c(x,y) \tag{1}$$

where I is the spatial extent of the motion descriptors. In Efros et al. [8], a temporal smoothing is also used, but we found the simplified version without temporal smoothing works good enough for our application.

To construct the codebook, we randomly select a subset from all the frames, compute the affinity matrix A on this subset of frames, where each entry in the

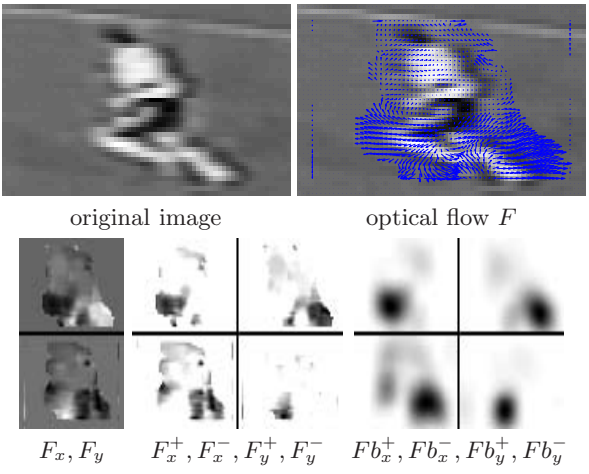


Fig. 2. Construction of the motion descriptor

affinity matrix is the similarity between frame i and frame j calculated using the normalized correlation described above. Then we run k-medoid clustering on this affinity matrix to obtain V clusters. Codewords are then defined as the centers of the obtained clusters. In the end, all the video sequences are converted to the "bag-of-words" representation by replacing each frame by its corresponding codeword.

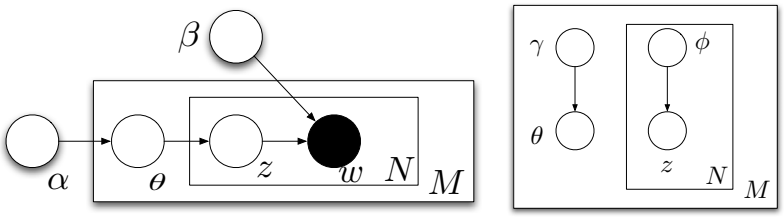
3.2 Latent Dirichlet Allocation

Our model is based the Latent Dirichlet Allocation (LDA) [2]. In the following, we briefly introduce LDA model using the terminology in our context.

Suppose we are given a collection D of video sequences $\{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M\}$. Each video sequence $\mathbf{w}$ is a collection of frames $\mathbf{w} = (w_1, w_2, ..., w_N)$, where w_i is the motion word representing the i-th frame. A motion word is the basic item from a codebook (see Sect.3.1) indexed by $\{1, 2, ..., V\}$.

The LDA model assumes there are K underlying latent topics (i.e., action class labels) according to which video sequences are generated. Each topic is represented by a multinomial distribution over the $|V|$ motion words. A video sequence is generated by sampling a mixture of these topics, then sampling motion words conditioning on a particular topic. The generative process of LDA for a video sequence $\mathbf{w}$ in the collection can be formalized as follows (see Fig. 3(a)):

1. Choose $\theta \sim \mathrm{Dir}(\alpha)$
2. For each of the N motion words w_n:
   (a) Choose an action label (i.e., topic) $z_n \sim \mathrm{Mult}(\theta)$;
   (b) Choose a motion word w_n from $w_n \sim p(w_n|z_n, \beta)$, a multinomial probability conditioned on z_n.



(a) (b)

Fig. 3. (a) Graphical representation of LDA model, adopted from Blei et al. [2]; (b) Graphical representation of the variational distribution

The parameter θ indicates the mixing proportion of different actions labels in a particular video sequence. α is the parameter of a Dirichlet distribution that controls how the mixing proportions θ vary among different video sequences. β is the parameter of a set of multinomial distributions, each of them indicates the distribution of motion words within a particular action label. Learning a LDA model from a collection of video sequences $D = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M\}$ involves finding α and β that maximize the log likelihood of the data $l(\alpha, \beta) = \sum_{d=1}^{M} \log P(\mathbf{w}_d|\alpha, \beta)$. This parameter estimation problem can be solved by the variational EM algorithm developed in Blei et al. [2].

3.3 Semi-Latent Dirichlet Allocation

In the original LDA, we are only given the word $(w_1, w_2, ..., w_N)$ in each video sequence, but we do not know the topic z_i for the word w_i, nor the mixing proportion θ of topics in the sequence. In order to use LDA for classification problems, people have applied various tricks. For example, Blei et al. [2] use LDA to project a document onto the topic simplex, then train an SVM model based on this new representation, rather than the original vector representation of a document based on words. Although this simplex is a much compact representation for the documents, the final SVM classifier based on this new representation actually performs worse than the SVM classifier trained on the original vector representation based on words. Sivic et al. [25] use a simpler method by classifying an image to a topic in which the latent topics of this document is most likely to be drawn from. There are two problems with this approach. First of all, there is no guarantee that a "topic" found by LDA corresponds to a particular "object class". Secondly, it is not clear how many "topics" to choose.

In this paper, we are interested in the action classification problem, where all the frames in the training video sequences have action class labels associated with them. In this case, there is no reason to ignore this important information. In this section, we introduce a semi-supervised version of the LDA model called *Semi-Latent Dirichlet Allocation (S-LDA)*. S-LDA utilizes class labels by enforcing a one-to-one correspondence between topics and class labels. Since we use a word w_i to represent a frame in a video sequence $\mathbf{w} = (w_1, w_2, ..., w_N)$, the topic z_i

for the word w_i is simply the class label of w_i. The graphical representation of S-LDA model is shown in Fig. 4. We should emphasize that the model in Fig. 4 is only for training (i.e., estimating α and β). In testing, we will use the same model shown in Fig. 3(a), together with estimated model parameters α and β.

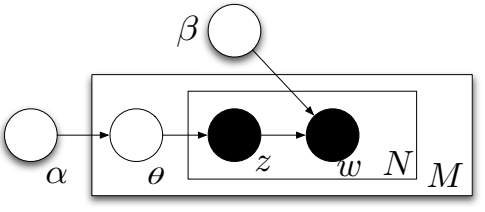


Fig. 4. Graphical representation of the Semi-Latent Dirichlet Allocation (S-LDA) for training. Note the difference from Fig. 3(a) is that z is observed in this case.

Our model has three major advantages over previous approaches of using a topic model for classification problems. First of all, choosing the right number of topics is trivial, since it is simply the number of class labels in the training sequences. Secondly, the training process of the S-LDA model is much easier than the original LDA. Thirdly, we can achieve much better recognition accuracy by taking advantage of the class labels (see Sect. 4).

In LDA (see Fig. 3(a)), the parameters α and β are coupled, conditioning on the observed words $\mathbf{w}$. In that case, the model parameters (α and β) have to be estimated jointly, which is difficult. Various approximation approaches (e.g., sampling, variational EM, etc) have to be used. However, in S-LDA (Fig. 4), the parameters α and β become independent, conditioning on observed words $\mathbf{w}$ and their corresponding topics (i.e., class labels) $\mathbf{z}$. So we can estimate α and β separately, which makes the training procedure much easier. In the following, we describe the details of estimating these parameters.

The parameter β can be represented by a matrix of size $K \times V$, where K is the number of possible topics (i.e., class labels) and V is the number of possible words. The i-th row of this matrix (β_i) is a V-dimensional vector that sums to 1. β_i is the parameter for a multinomial distribution, which defines the probability of drawing each word in the i-th topic. The maximum-likelihood estimate of β_i can be calculated by simply counting the frequency of each word appearing together with topic z_i, i.e., $\beta_{ij} = n_{ij}/n_{i\cdot}$, where $n_{i\cdot}$ is the count of the i-th topic in the corpus, and n_{ij} is the count of i-th topic with j-th word in the corpus.

The estimation of the Dirichlet parameter α is a bit more involved. The Dirichlet distribution is a model of how topic mixing proportions θ vary among documents. This distribution has the form $p(\theta|\alpha) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k \theta_k^{\alpha_k - 1}$. In order to estimate α, we first need to compute θ for each document $\mathbf{w} = (w_1, w_2, ..., w_N)$. Suppose the topics (i.e., the class labels of the words) of the document are $\mathbf{z} = (z_1, z_2, ..., z_N)$, then the i-th coordinate θ_i of θ can be calculated as $\theta_i =$

$|\{i : z_j = i, j = 1, 2, ..., N\}|/N$. After we collect all the θ^t $(t = 1, 2, ..., M)$ values (as a notation convention, we use subscripts to denote coordinates of θ and superscripts to denote document numbers), the parameter α can be estimated from $\Theta = \{\theta^1, \theta^2, ..., \theta^M\}$ using generalized Newton Raphson iterations [18].

3.4 Classification of New Video Sequences

Given a new video sequence for testing, we would like to classify each frame in the sequence. Suppose the test video sequence is represented as $\mathbf{w} = (w_1, w_2, ..., w_N)$, i.e., there are N frames in the sequence, and the i-th frame is represented by the motion word w_i. Then, we need to calculate $p(z_i|\mathbf{w}, \alpha, \beta)$ $(i = 1, 2, ..., N)$. The frame w_i is classified to be action class k if $k = \text{argmax}_j p(z_i = j|\mathbf{w}, \alpha, \beta)$. Notice that we use $p(z_i|\mathbf{w})$ instead of $p(z_i|w_i)$ for classification. This reflects our assumption that the class label z_i not only depends on its corresponding word w_i, but also depends on the video sequence $\mathbf{w} = (w_1, w_2, ..., w_N)$ as a whole.

To calculate $p(z_i|\mathbf{w}, \alpha, \beta)$, we use the variational inference algorithm proposed in Blei et al. [2]. The basic idea of the variational inference is to approximate the distribution $p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta)$ by a simplified family of variational probability distributions $q(\theta, \mathbf{z})$ with the form $q(\theta, \mathbf{z}) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n)$. The graphical representation of $q(\theta, \mathbf{z}|\gamma, \phi)$ is shown in Fig. 3(b). In order to make the approximation as close to the original distribution as possible, we need to find (γ^*, ϕ^*) that minimize the Kullback-Leibler (KL) divergence between the variational distribution $q(\theta, \mathbf{z}|\gamma, \phi)$ and the true distribution $p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta)$, i.e., $(\gamma^*, \phi^*) = \arg\min_{(\gamma,\phi)} D(q(\theta, \mathbf{z}|\gamma, \phi) \| p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta))$, where $D(\cdot|\cdot)$ is the KL divergence. Finding (γ^*, ϕ^*) can be achieved by iteratively updating (γ, ϕ) using the following update rules (see Blei et al. [2] for detailed derivation):

$$\phi_{ni} \propto \beta_{iv} \exp(\Psi(\gamma_i) - \Psi(\sum_{j=1}^{K} \gamma_j)) \tag{2}$$

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni} \tag{3}$$

Several insights can be drawn from examining the variational parameters $(\gamma^*(\mathbf{w}), \phi^*(\mathbf{w}))$. First of all, $(\gamma^*(\mathbf{w}), \phi^*(\mathbf{w}))$ are document-specific. For a particular document $\mathbf{z}$, $\gamma^*(\mathbf{w})$ provides a representation of a document in the topic simplex. Also notice that $\text{Dir}(\gamma^*(\mathbf{w}))$ is the distribution from which the mixing proportion θ for the document $\mathbf{w}$ is drawn. We can imagine that if we draw a sample $\theta \sim \text{Dir}(\gamma^*(\mathbf{w}))$, θ will tend to peak towards the true mixing proportion θ^* of topics for the document $\mathbf{w}$. So the true mixing proportion θ^* can be approximated by the empirical mean of a set of samples θ_i drawn from $\text{Dir}(\gamma^*(\mathbf{w}))$. The second insight comes from examining the ϕ_n parameters. These distributions approximate $p(z_n|w_n)$. The third insight is that, since the topic z_n is drawn from $\text{Mult}(\theta^*)$, θ^* is an approximation of $p(z_n)$. Then we can

get $p(z_n|\mathbf{w}) \propto p(z_n|\theta^*)p(z_n|w_n) \approx \theta^*_{z_n}\phi_{z_n w_n}$.This equation has a very appealing intuition. It basically says the class label z_n is determined by two factors, the first factor $\theta^*_{z_n}$ tells us the probability of generating topic z_n in a document with mixing proportion θ^*, the second factor $\phi_{z_n w_n}$ tells us the probability of generating topic z_n conditioning on a particular word w_n.

3.5 Irregularity Detection in Video Sequences

An interesting application of our method is detecting irregularities (i.e., novelty) in video sequences. This has a lot of potential applications in surveillance and monitoring. Previous approaches to irregularity detection can be broadly classified into two classes: rule-based method and statistical methods [4]. Our method falls into the statistical methods, which try to learn a model of regularity from data, and infer about irregularity using the model.

There are various notions of "irregularity". For example, one possibility is to define all the actions that never appears in the training set to be "irregular". But in this paper, we focus on another case, where "irregularity" is defined by the composition of different actions, rather than the actions themselves. For example, loitering in a parking lot is composed of actions which by themselves are regular. But taken together, those regular actions form an unusual and suspicious behavior. This irregularity is characterized by the unique combination of regular actions. Other irregularity detection algorithms using only low-level cues (e.g. Boiman & Irani [4]) would not be able to identify it.

The application of our method to irregularity detection is quite straightforward. We first build our S-LDA model from a collection of training video sequences that are considered to be "regular", i.e., estimating the model parameters α and β using the method in Sect. 3.3. Given a new testing video sequence $\mathbf{w}$, we calculate the likelihood $l(\mathbf{w}; \alpha, \beta) = p(\mathbf{w}|\alpha, \beta)$ using the method in Sect. 3.4. If $\mathbf{w}$ is very different from those in the training set, i.e., it is not generated by the LDA model defined by α and β, it will probably have a very low likelihood under the model. So the likelihood of this new testing video sequence is an indicator of "irregularity". Lower likelihood means being more "irregular".

4 Experiments

We test our algorithm on two datasets: KTH human motion dataset [24] and soccer dataset [8].

4.1 Action Classification on KTH Dataset

The KTH human motion dataset is one of the largest video datasets of human actions. It contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed several times by 25 subjects in

Fig. 5. Representative frames in KTH dataset

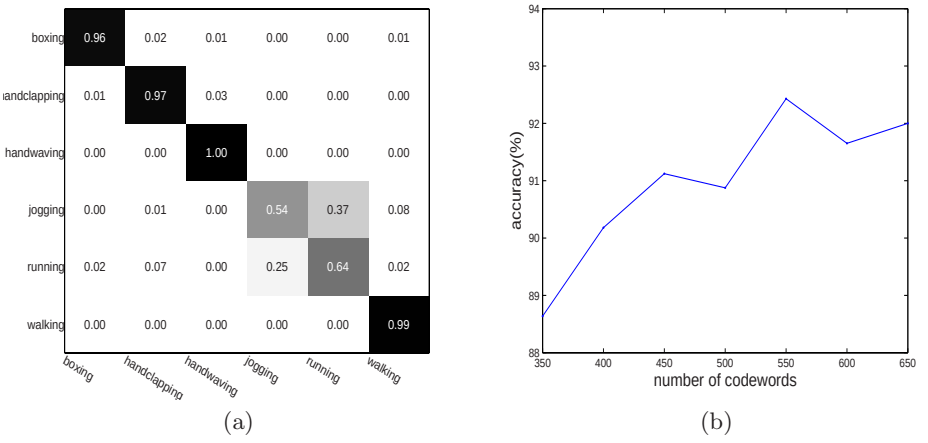
four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. Representative frames of this dataset are shown in Fig. 5.

We first run an automatic preprocessing step to track and stabilize the video sequences using the algorithm in Sabzmeydani &Mori [23], so that all the figures appear in the center of the field of view. We perform leave-one-out cross-validation on this dataset. For each run, we choose the video sequences of one subject as the test set, and build our model on the rest of the video sequences. We run the same process on each of the video sequence. For each run, we take the features obtained from Sect. 3.1 as the feature vectors. Then these feature vectors are quantized by k-medoid clustering to form the motion words. Since the number of feature vectors is huge, we randomly select a small number (about 30 frames) from each training video sequence for the k-mediod clustering.

The confusion matrix for the KTH dataset using 550 codewords is shown in Fig. 6(a). We can see that the algorithm correctly classifies most of actions. Most of the mistakes the algorithm makes are confusion between "running" and "jogging" actions. This is intuitively reasonable, since "running" and "jogging" are similar actions.

We also test the effect of the codebook size on the overall accuracy. The result is shown in Fig. 6(b). The best accuracy is achieved with 550 codewords, but is relatively stable.

We compare our results with previous approaches on the same dataset, as shown in Table 1. We would like to point out that the numbers in Table 1 are not directly comparable, since different approaches use different split of training and test data. In particular, the first three approaches use "leave-one-out" cross validation, while the remaining two approaches equally split the dataset into training, validation, and test sets. Nevertheless, Our method achieves better performance by a large margin. As a sanity check, we also run our experiment by equally splitting the dataset into training and test sets (our algorithm does not need a validation set), the recognition accuracy is similar.



(a) (b)

Fig. 6. (a) Confusion matrix for KTH dataset using 550 codewords (overall accuracy=92.43%). Horizontal rows are ground truth, and vertical columns are predictions. The action labels are "boxing","handclapping","handwaving","jogging","running","walking"; (b) classification accuracy vs. codebook size for KTH dataset.

Table 1. Comparison of different methods in terms of recognition accuracy on the KTH dataset

methods	**recognition accuracy(%)**
Our method	**92.43**
Niebles et al. [19]	81.50
Dollár et al. [7]	81.17
Schuldt et al. [24]	71.72
Ke et al. [14]	62.96

4.2 Action Classification on Soccer Dataset

The soccer dataset we use is from Efros et al. [8][1]. This dataset contains several minutes of digitized World Cup football game from an NTSC video tape. A preprocessing step is taken to track and stabilize each human figure. In the end, we obtain 35 video sequences, each corresponding to a person moving in the center of the field of view. All the frames in these video sequences are hand-labeled with one of 8 action labels: "run left 45 °", "run left", "walk left", "walk in/out", "run in/out", "walk right", "run right", "run right 45 °". Representative frames of a single tracked person are shown in Fig. 1(b).

Again, we perform leave-one-out cross-validation on the dataset. The confusion matrix using 350 codewords is shown in Fig. 7(a). The overall accuracy is 79.19%. Table 2 shows the main diagonal, compared with the main diagonal

[1] Unfortunately, other datasets (tennis, ballet) used in this paper were not available.

from Efros et al. [8], which used a k-nearest neighbor classifier based on the temporally smoothed motion feature vectors. We can see that our method performs better by a large margin for most of the class labels. Also, we can see that a lot of the mistakes made by our algorithm makes intuitive sense. For example, "run left 45 °" is confused with "run left" and "run in/out", "walk right" is confused with "walk in/out" and "run right", etc. In addition to achieving a higher accuracy, our algorithm has the added advantage that it is faster to classify a new video sequence, since we do not have to search over all the video sequences in the training set, as required by k-nearest neighbor classifiers.

We test the effect the codebook size on the overall accuracy. The result is shown in Fig. 7(b). The best accuracy peaks at around 350.

Table 2. Comparison of the main diagonal of the confusion matrix of our method and the method in Efros et al. [8] on the soccer dataset

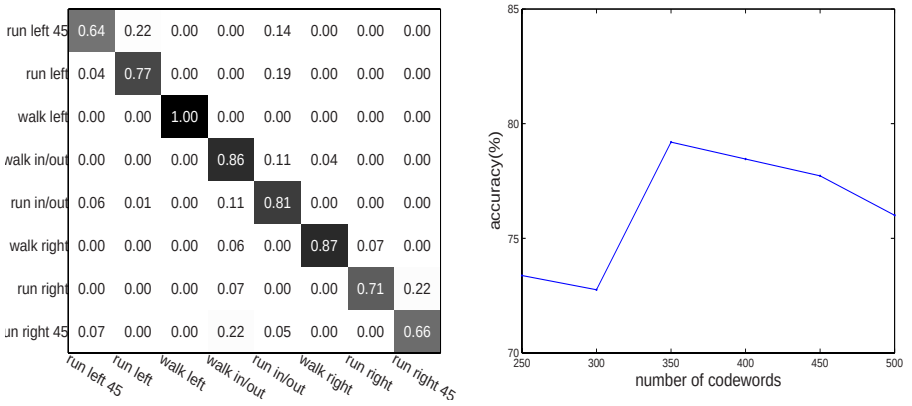
	Our method	Efros et al. [8]
run left 45 °	0.64	0.67
run left	**0.77**	0.58
walk left	**1.00**	0.68
walk in/out	**0.86**	0.79
run in/out	**0.81**	0.59
walk right	**0.86**	0.68
run right	**0.71**	0.58
run right 45 °	0.66	0.66

4.3 Irregularity Detection

Since all the video sequences in the KTH dataset only contain a single action, we only test the irregularity detection of our algorithm on the soccer dataset. Our experimental setting is similar to that in Sect. 4.2. For each run, we choose one video sequence as the test set, and build our model from the remaining video sequences. Then we calculate the likelihood $p(\mathbf{w}|\alpha, \beta)$ of the testing video sequence under the built model. The likelihood value gives us some indication on how "irregular" this testing video sequence is, compared with the remaining video sequences.

We repeat the above process for all the video sequences, then rank them according to the increasing order of their likelihood values $p(\mathbf{w}|\alpha, \beta)$. Under our assumption, the top few videos in the list should be considered to be "irregular".

Since there is no ground truth in this experiment, we can only report our results empirically. Table 3(a) shows the frame labels in the top five video sequences (i.e., the most "irregular" ones). We can see that the videos in Table 3(a) are in general "irregular". For example, they all involve a human figure runs/walks out of the scene. In fact, the combinations of the frame labels only appear once or twice in our training set. Table 3(b) shows the frame labels in the bottom five video sequences (i.e., the most "regular" or boring ones). They are obviously "boring" video sequences, since they only contain people running. It is



(a) (b)

Fig. 7. (a)Confusion matrix for soccer dataset using 350 codewords (overall accuracy=79.19%). Horizontal rows are ground truth, and vertical columns are predictions. The action labels are "run left 45°","run left","walk left", "walk in/out", "run in/out", "walk right", "run right", "run right 45°"; (b) classification accuracy vs. codebook size for the soccer dataset.

Table 3. Results of irregularity detection: (a) top five "irregular" video sequences; (b) top five "regular" video sequences

Sequence No.	frame labels
sequence 6	"run right" "run right 45°" "walk in/out" "walk right"
sequence 5	"walk right" "walk in/out"
sequence 9	"run left" "run left 45°" "run in/out" "run right 45°"
sequence 32	"run in/out" "walk in/out"
sequence 33	"walk in/out"

(a)

Sequence No.	frame labels
sequence 1	"run left"
sequence 2	"run left 45°" "run left"
sequence 29	"run left"
sequence 4	"run left"
sequence 3	"run left"

(b)

interesting to see that the "boring" video sequences picked out by our algorithms are not necessarily the ones with a single action (see sequence 33 in Table 3(a) and sequence 2 in Table 3(b)).

5 Conclusion

We have presented a hierarchical probabilistic model (semi-latent Dirichlet allocation) for action recognition based on motion words, where each word

corresponds to a frame in the video sequence. By naturally exploiting class labels of training data in our model, we are able to achieve much better results, compared with previous "bag-of-words" methods.

Of course, our method has its own limitations. For example, it requires a preprocessing stage of tracking and stabilizing human figures. However, we believe this is a reasonable assumption in many scenarios. In fact, all the video sequences in our experiments are pre-processed by off-the-shelf tracking/detection algorithms without much efforts.

References

1. Bissacco, A., Yang, M.H., Soatto, S.: Detecting humans via their pose. In: NIPS. Advances in Neural Information Processing Systems, vol. 19, pp. 169–176. MIT Press, Cambridge (2007)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
3. Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(3), 257–267 (2001)
4. Boiman, O., Irani, M.: Detecting irregularities in images and in video. In: IEEE International Conference on Computer Vision, vol. 1, pp. 462–469 (2005)
5. Bosch, A., Zisserman, A., Munoz, X.: Scene classification via pLSA. In: European Conference on Computer Vision, vol. 4, pp. 517–530 (2006)
6. Cutler, R., Davis, L.S.: Robust real-time periodic motion detection, analysis, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 781–796 (2000)
7. Dollár, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: ICCV 2005. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (2005)
8. Efros, A.A., Berg, A.C., Mori, G., Malik, J.: Recognizing action at a distance. In: IEEE International Conference on Computer Vision, vol. 2, pp. 726–733 (2003)
9. Fei-Fei, L., Fergus, R., Perona, P.: One-shot learning of object categories. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(4), 594–611 (2006)
10. Fei-Fei, L., Perona, P.: A bayesian hierarchical model for learning natural scene categories. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 524–531 (2005)
11. Fergus, R., Fei-Fei, L., Perona, P., Zisserman, A.: Learning object categories from google's image search. In: IEEE International Conference on Computer Vision, vol. 2, pp. 1816–1823 (2005)
12. Grauman, K., Darrell, T.: The pyramid match kernel: Discriminative classification with sets of image features. In: IEEE International Conference on Computer Vision, vol. 2, pp. 1458–1465 (2005)
13. Hofmann, T.: Probabilistic latent semantic indexing. In: SIGIR. Proceedings of Twenty-Second Annual International Conference on Research and Development in Information Retrieval, pp. 50–57 (1999)
14. Ke, Y., Sukthankar, R., Hebert, M.: Efficient visual event detection using volumetric features. In: IEEE International Conference on Computer Vision, vol. 1, pp. 166–173 (2005)

15. Lazebnik, S., Schmid, C., Ponce, J.: A maximum entropy framework for part-based texture and object recognition. In: IEEE International Conference on Computer Vision, vol. 1, pp. 832–838 (2005)
16. Little, J.L., Boyd, J.E.: Recognizing people by their gait: The shape of motion. Videre 1(2), 1–32 (1998)
17. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the DARPA Image Understanding Workshop, pp. 121–130 (April 1981)
18. Minka, T.P.: Estimating a Dirichlet distribution. Technical report, Massachusetts Institute of Technology (2000)
19. Niebles, J.C., Wang, H., Fei-Fei, L.: Unsupervised learning of human action categories using spatial-temporal words. In: British Machine Vision Conference, vol. 3, pp. 1249–1258 (2006)
20. Polana, R., Nelson, R.C.: Detection and recognition of periodic, non-rigid motion. International Journal of Computer Vision 23(3), 261–282 (1997)
21. Rao, C., Yilmaz, A., Shah, M.: View-invariant representation and recognition of actions. International Journal of Computer Vision 50(2), 203–226 (2002)
22. Russell, B.C., Efros, A.A., Sivic, J., Freeman, W.T., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1605–1614 (2006)
23. Sabzmeydani, P., Mori, G.: Detecting pedestrians by learning shapelet features. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2007)
24. Schuldt, C., Laptev, L., Caputo, B.: Recognizing human actions: a local SVM approach. In: IEEE International Conference on Pattern Recognition, vol. 3, pp. 32–36 (2004)
25. Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A., Freeman, W.T.: Discovering objects and their location in images. In: IEEE International Conference on Computer Vision, vol. 1, pp. 370–377 (2005)
26. Sullivan, J., Carlsson, S.: Recognizing and tracking human action. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2352, pp. 629–644. Springer, Heidelberg (2002)
27. Zhong, H., Shi, J., Visontai, M.: Detecting unusual activity in video. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 819–826 (2004)

Recognizing Activities with Multiple Cues

Rahul Biswas[1], Sebastian Thrun[1], and Kikuo Fujimura[2]

[1] Stanford University
{rahul,thrun}@cs.stanford.edu
[2] Honda Research Institute
kfujimura@hra.com

Abstract. In this paper, we introduce a first-order probabilistic model that combines multiple cues to classify human activities from video data accurately and robustly. Our system works in a realistic office setting with background clutter, natural illumination, different people, and partial occlusion. The model we present is compact, requires only fifteen sentences of first-order logic grouped as a Dynamic Markov Logic Network (DMLNs) to implement the probabilistic model and leverages existing state-of-the-art work in pose detection and object recognition.

1 Introduction

In recent years, there has been considerable success in bolstering the performance of object recognition by considering not just the object itself but also the context in which it occurs. For example, in [18] and [35], the recognition of objects is boosted by analyzing contextual information within a camera image, such as the presence and absence of other objects, the relative location of the object in question, and global features characterizing the scene. That work expresses the concept of context through probabilistic relationships of multiple recognizers. A probabilistic model posits the relationship between context and objects.

In this paper, we seek a plausible extension of this work to image sequences. By tying together information about peoples' poses, objects seen, and relative locations, we seek to identify peoples' activities using probabilistic models pertaining to these cues. Using all three cues in tandem offers far greater accuracy than any of the cues on its own.

To achieve these results, we develop a novel framework for expressing the spatio-temporal relation of cues and activities. Our framework is based on Dynamic Markov Logic Networks (DMLNs) [31], a well-established first-order probabilistic representation. We demonstrate in this paper that the DMLN framework provides a powerful language to express the probabilistic relationship of cues and activities in the video analysis domain. In fact, we show that useful DMLN theories establish the notion of context significantly more compactly than the propositional representations used, for example, in [18] and [35]. Our approach introduces new inference techniques for DMLN inference that accommodates the specific nature of the inference problem in the computer vision domain.

Our experiments apply state-of-the-art techniques for pose detection and object recognition. We show empirically that DMLNs effectively leverage context

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 255–270, 2007.

and achieve improved recognition rates. These results are well in tune with earlier work on this topic. Hence, we conjecture that the DMLN framework provides an elegant and effective way to extend that work into the temporal domain involving human activities – a research topic that has found considerable attention in the computer vision field in past years [25], [16], [13], [7].

2 Overview of DMLNs

In this section, we present the concept of Dynamic Markov Logic Networks (DMLNs). While a formal treatment is far beyond the scope of this work, we refer the interested reader to [29] and [31], the definitive works regarding DMLNs.

A DMLN uses the language of First-Order Logic [10] to express a First-Order Probabilistic Model [23]. It explicitly posits the notion of objects and boolean predicates regarding them. Let us illustrate these notions by way of an example.

First, let p, q, and r be objects. These can be many things, depending on the problem of interest. Examples include an image, a single pixel, a feature being tracked, a robot, or a database entry. Probabilistic models such as Markov Random Fields or Dynamic Bayesian Networks consider only propositions, possibly about objects, but cannot consider objects explicitly.

Second, let $A(x, y, t)$ and $B(z, t)$ be fluents. Examples of fluents include whether one tracked object is occluding another at time step t or whether a specific pixel position in a video stream is part of an edge. The fluents relate to variables which will be substituted with actual objects at runtime.

Each possible assignment of objects to fluents forms what is known as a ground fluent. In this example, the ground fluents are $A(p, p)$, $A(p, q)$, $A(p, r)$, $A(q, p)$, $A(q, q)$, $A(q, r)$, $A(r, p)$, $A(r, q)$, $A(r, r)$, $B(p)$, $B(q)$, and $B(r)$. Each ground fluent is true or false. Let us consider a concrete example. Let p, q, and r be three people. Let $A(x, y, t)$ mean that x and y are friends at time t and let $B(z, t)$ mean that z is cheerful at time t. In this example, at time 0, let p and q be friends with one another and r is unknown to the other two. Moreover, let p be the only cheerful one. Then ground fluents $A(p, q, 0)$, $A(q, p, 0)$, $A(p, p, 0)$, $A(q, q, 0)$, $A(r, r, 0)$, and $B(p, 0)$ are all true and all other fluents at time 0 are false. We follow the arbitrary convention that one is always friends with oneself.

In addition to objects and fluents, DMLNs also have weighted sentences that give rise to a probability distribution over models, i.e. truth assignments to all ground fluents. Such a sentence might be

$$B(x, t) \rightarrow B(x, succ(t)) \tag{1}$$

with weight 2.0. It means that one typically but not always remains cheery in the future when one has been cheery in the past.

Another sentence might be

$$B(x, t) \wedge A(x, y, t) \rightarrow B(y, t) \tag{2}$$

with weight 1.0. It means that cheerful people pass on their cheerfulness to their friends. This effect is weaker than retaining one's own cheerfulness.

While DMLNs properly give rise to a joint probability distribution over ground fluents, let it suffice to state for this exposition that models that satisfy more sentences are more likely than those that do not. Moreover, models that satisfy higher weight sentences are more likely in a probabilistic sense than those that satisfy lower weight sentences.

3 Activity Recognition System

3.1 Probabilistic Model

The key contribution of this paper is to express a graphical model for combining multiple cues – pose, object presence, and movement location – through a compact DMLN theory. While graphical models have produced excellent results in computer vision applications, they have been too complicated – difficult to construct, modify, and communicate. We present our DMLN model in Figure 1 as an alternative.

1	$\forall t \forall a$ Activity(a,t) $\rightarrow$ Activity(a,succ(t))	8.3
2	$\forall t \forall a_1 \forall a_2$ Activity(a_1,t) $\rightarrow$ =(a_1,a_2) $\vee \neg$ Activity(a_2,t)	∞
3	$\forall t \exists a$ Activity(a,t)	∞
4	$\forall t \forall a$ Activity(a,t) $\rightarrow$ Pose(a,t)	0.7
5	$\forall t$ Activity(a,t) $\wedge$ Useful(o,a) $\rightarrow$ Present(o,t)	0.7
6	Useful(TYPING,KEYBOARD)	∞
7	Useful(MOUSING,MOUSE)	∞
8	Useful(EATING,CANDY)	∞
9	Useful(EATING,APPLE)	∞
10	Useful(DRINKING,SODA)	∞
11	Useful(READING,BOOK)	∞
12	Useful(TALKING,PHONE)	∞
13	Useful(WRITING,PEN)	∞
14	Useful(WRITING,PAPER)	∞
15	$\forall t \forall a$ Activity(a,t) $\rightarrow$ Movement(a,t)	0.2

Fig. 1. Probabilistic Model as Dynamic Markov Logic Network (sentence weights appear in the right column)

3.2 Fluents and Sentences

Sentence 1 of Figure 1 refers to the predicates Activity(a,t) and Activity (a,succ(t)). Activity(a,t) is a binary variable that means that the person is engaged in activity a at time t. For example, Activity($WRITING$,1244) means that the observed person is writing in frame 1244 of the video sequence. We will assume throughout that there is exactly one person in each image. One could relax this assumption along the lines of [12] if need be.

This sentence states that activities tend to persist over time. In other words, if you are drinking from a can of soda, you will most likely continue to be

still drinking from that can in the next frame, tens of milliseconds later. In the sentence, succ(t) means that successor to t, i.e. the next frame. The sentence literally reads – for all time steps t and for all activities a, if the user is engaged in activity a at time step t, then the user will be engaged in activity a in the time step following t.

Sentence 2 establishes a mutual exclusion constraint between simultaneous activities. Simply put, we are providing the machine a generalization of the notion that a person cannot walk and chew gum at the same time. This sentence reads – for all time steps t and for all activities a_1 and for all activities a_2, if the user is engaged in activity a_1 at time t, then either a_1 and a_2 refer to the same activity or the user is not engaged in activity a_2 at time t. This sentence has the special property that it has infinite weight, i.e. this is a hard constraint.

Sentence 3 states that the user must be doing something. It is possible to add a special activity, $NOTHING$, to indicate times where the user is not engaged in any activity.

3.3 Incorporating Cues

The output of the pose recognition system described in section 4 is represented by asserting the sentence Pose(a,t), which means that the person's observed pose appears to be in keeping with being engaged in activity a at time t. The pose detection posits a sentence weight reflecting its confidence, as do the other components. Sentence 4 states that a person's pose will reflect the activity they are currently engaged in.

The objects a person is using also provide valuables cues as to what they are doing. In this component, we are content to consider what objects are in view. If we see both a can of soda and a phone, we cannot definitely say which one is being used but if we do not see a keyboard, we are less likely to think that the person is typing.

The object recognition component uses the fluents Present(o,t) (an object of type o is present at time t) and Useful(a,o) (objects of type o are useful for activity a). Sentence 5 states that objects useful to an activity will be present when that activity is engaged in. Sentences 6 through 14 specify which objects are useful for which activities. Here, mousing refers to using a computer mouse and talking refers to conversing on a telephone.

Sentence 15 states that the activity undertaken influences the location of the movement in the camera image. The fluent $Movement(a, t)$ states which activity seems likely by analyzing movement alone.

Before we move on, let us ensure that we understand the sentences by understanding the effect of removing any one sentence in isolation. Removing the first sentence would eliminate the continuity of our temporal model, forcing the inference procedure to consider each image in isolation without the benefit of the entire video sequence. Removing the second and/or third sentence would allow for the classifier to choose no activity or to choose more than one. Removing the fourth sentence would prevent the pose detector from providing useful information and removing the fifth or fourteenth sentence would do the same for

the object and movement detection, respectively. Removing any of the sentences from six through fourteen would make the system blind to the corresponding object type.

3.4 Ease of Expression in DMLNs

This paper is as much about building an activity recognition system using multiple cues as it is about the efficacy of DMLNs in computer vision. We use multiple cues and do activity recognition in this manner because it is robust against background clutter, lighting differences, intra- and inter-personal variance, and difficulties in pose and object recognition.

Our motivation for leveraging DMLNs is very different. We use the DMLN because it is possible to posit a temporal probabilistic model that is simple to express and easy to understand without compromising the sophistication necessary for high performance. Table 1 is our entire probabilistic model. We wrote a set of sentences that we thought represented activities and cues well and revised it until we were pleased with it. Contrast this with comparable models expressed with Markov Random Fields (MRFs) and Dynamic Bayesian Networks (DBNs). Those models are equally effective but they are hard to understand and challenging to implement.

In practice, DMLNs are inferentially equivalent to MRFs and DBNs. Indeed, to perform inference on the DMLN, we convert it twice, first to a ground MLN which is an MRF and then to a DBN. This process is straightforward except for the challenge that is addressed in section 5.1. Thus, one can think of DMLNs much as one would view a high-level computer programming language like C as opposed to a low-level assembly language. The study and design of systems in assembly language paved the way for the new languages. While initially the new language only brought programmer convenience, it eventually allowed for greater abstraction and better programs.

4 Obtaining the Ground Predicates

In section 3, we explored a theory for understanding how various activities give rise to cues that are useful in identifying those activities. In this section, we discuss how we use computer vision algorithms to translate our video stream into assertions regarding the cue predicates.

4.1 Pose Detection

Comparing Poses. Traditionally, pose detection is the problem of recovering the position of important body parts from a single image. For example, for a person playing baseball, a pose detection algorithm may identify the x,y positions in the image of her feet, hands, and head [17]. Or, for a person walking down the street, an algorithm may fit a skeleton, thus identifying the feet, knees, and hip [34].

Fig. 2. Examples of difference images

We take an alternate approach to pose detection. The process of inferring body part positions and then linking configurations to activities is problematic on two levels. First, traditional pose detection is a difficult problem and even the best algorithms are subject to both inaccuracy and catastrophic failure. Second, the simplified 2D representation of pose is a poor indicator of human activity. Studying the evolution of pose over time is even less effective as the errors in pose detection create more phantom movements than actual ones. Earlier versions of our pose detection system suffered from each of the aforementioned difficulties.

We eliminate the intermediate step of recovering body part positions and instead map changes in pose to activities directly. To identify the change in pose, we start with difference image generated by subtracting two consecutive frames and thresholding it:

$$d_{x,y} = abs(i^t_{x,y} - i^{t+1}_{x,y}) \tag{3}$$

Examples of such difference images appear in Figure 2. Note that this difference image captures the outline of the body part that is moving, even if it does so imperfectly.

Second, to allow comparison between difference images, we construct a list of all pairwise point to point distances of the illuminated pixels in the difference image. This list retains the basic shape of the image while removing exact position and orientation information.

Third, we form a histogram over the list of distances just constructed. This is similar to how shape contexts [4] compact distance information except that angles are considered there but not here.

To compute the similarity of two histograms, we use the same scoring function as shape contexts:

$$\sum_{i=1}^{N} \frac{(h^1_i - h^2_i)^2}{h^1_i + h^2_i} \tag{4}$$

where N is the number of bins in the histogram.

Realizing the Pose Cue Predicates. We have established a metric for determining if two frames from different video streams represent the same or different

changes in pose. To make this useful, we capture a several minute training video with different people performing each of the activities we are working to recognize. The new activity being undertaken is marked at activity transitions (e.g. when a person stops typing and starts drinking) in this video are labeled by hand. When the machine is later observing novel video, it compares the histogram of the incoming frame t^* with each of these histograms of the frames in the training video, as in [32]. All histograms are cached to ensure that this is a fast operation. For each activity, the number of comparisons exceeding a threshold is counted. The activity with the greatest count is selected (let it be a^*) and if the count exceeds a predetermined threshold, then the ground predicate $Pose(a^*, t^*)$ is observed to be true.

4.2 Object Detection

Object detection offers a far more out of the box output for activity recognition than does pose detection. To compare two images, we find all of the SIFT features from each of the images. Then we compute the number of features the two images share in common.

To realize object-related cues, we first crop by hand images of the objects used form the training video. Second, we label each image with the object inside that image (e.g. a pen).

When identifying objects in novel frames, the program compares the large novel frame with each of the cropped training images. For each object category o^* for which the novel frame t^* has over a fixed threshold of matches, the predicate $Present(o^*, t^*)$ is asserted. Note that the matches may come from different cropped images. For example, if the threshold is fifth SIFT matches and there are three cropped images of a candy bar with twenty-four, fifteen, and twelve SIFT matches respectively, then the program believes that a candy bar is present even though that could not have been ascertained from a single image.

4.3 Movement

We found that for some activities, movement tends to occur in some areas more than others. This pattern was previously noted in [35]. For example, mouse movement tends to occur in the bottom half of the screen. While this is a weak source of information on its own, it provides a valuable third stream of information to the probabilistic model. It can differentiate between drinking and using the mouse, for example, but not between drinking and eating. On its own, it fares rather poorly because the relative movement depends on camera placement relative to the subject and that was not rigidly fixed in our experiments.

We learn a mixture of Gaussians from training data about typical movement locations as expressed in the training data. Each activity is thus represented by the mean and covariance of observed movement locations of that activity in the training data. The posterior weights of novel frames are asserted as Movement(a,t).

5 Inference

As DMLNs are inferentially equivalent to DBNs [29], we can use efficient approximate inference algorithms designed for the latter. We use a variant of the standard Rao-Blackwellized Particle Filter (RBPF) [9] that runs the particles both forward and backward, as in [26].

A difficulty arises however. Sentences 2, 3, and 6 through 14 all have infinite weight and while this does not break inference, it does slow it down considerably. In our proposed DMLN, nearly all of proposed next states will be inconsistent and receive weights of 0.

What we have here is a mixed network – a probabilistic model with both probabilistic and deterministic components. Inference in mixed networks is well understood in atemporal settings [2], [3], [8], [15]. In handling DBNs generated from DMLNs though, we believe that generating a more compact, inferentially equivalent non-mixed DBN works best. For DMLNs to have broader applicability in computer vision past our specific application, we need a general algorithm to compact mixed DMLNs. We include such an algorithm here.

5.1 Compacting Mixed DMLNs

The process of converting a DMLN into a DBN is slightly tortuous and involves generating a Markov Random Field (MRF) as an intermediate step. It is at this stage we will compact the model. Let L be a single time slice of the MRF with $\bar{L}$ referring to deterministic potentials and $\hat{L}$ referring to probabilistic ones. Moreover, let $\bar{P}$ refer to all nodes referred to by deterministic potentials and let $\hat{P}$ refer to all nodes not referred to by deterministic potentials. Thus, if a node is referred to in both $\bar{L}$ and $\hat{L}$, it will be in the set $\bar{P}$ but not in the set $\hat{P}$.

First, we divide the nodes in $\bar{P}$ into independent subproblems, i.e. into maximally disjoint sets such that there exists no potential in $\bar{P}$ that refers to nodes in different subproblems. This is done by postulating a graph with MRF nodes as graph nodes and adding edges if and only if there exists a potential in $\bar{P}$ that refers to both nodes. The connected components of this graph are the independent subproblems of our MRF.

Second, for each independent subproblem, we use WalkSAT, a fast Constraint Satisfaction Problem sampling technique [39], to identify all solutions to the subproblem. This can take up to $O(2^N)$ time where N is the number of nodes that are linked together by deterministic potentials. This is clearly better than the non-compacted network, which will require at least that much time at every single time step whereas compacting requires it just once. In practice, WalkSAT takes only linear time in the number of solutions, barring pathological DMLNs.

Third, we replace each independent subproblem with a single multinomial variable, in the same manner as the variable elimination algorithm [24].

While the entire process of converting DMLNs into a form suitable for RBPF inference does have such intricacies, it is important to note that this pipeline is independent of the specific DMLN and application. Indeed, toolkits for DMLN inference are already publicly available [1].

5.2 Weight Learning

The algorithm in [31] learns weights directly from training data, freeing us from such considerations as how long people actually drink from a can of soda. It essentially follows a frequentist strategy, assigning a weight that corresponds to the observed probability of that sentence in the training data. The sentences appearing in the right-hand column of Figure 1 are from this algorithm. Higher weights indicate greater certainty with infinite weights indicating deterministic sentences.

6 Experimental Results

To evaluate our algorithm, we solicited volunteers for what they believed was a psychology experiment. When each volunteer arrived, they were seated at a desk in our laboratory and provided with a list of the following activities in a random order and with repetition. First, they were to write a paragraph with a pen and notebook. These and all other objects were placed on the desk at which the subjects were seated. Second, they skimmed several pages of a textbook. Third, they ate part of a candy bar and drank from a can of soda. Fourth, they answered a phone call. Fifth, they typed on a laptop and used an external mouse.

The subjects were seated at a wooden desk illuminated by sunlight. A Canon HV10 consumer grade video camera recorded the scene from atop a fixed tripod looking down on the scene. This viewpoint is common on many laptop cameras with built-in webcams as well as several off the shelf webcam mounting kits. Objects often remain in view when not being used and other objects clutter the desk where activities take place. The camera was set to focus and adjust light balance automatically. Audio input was not used.

Training data for the experiments was comprised of the same activities performed in the same manner. Each transition from one activity to another was marked by hand and this was used to generate activity labels for each frame. Also, four images of each object were cropped by hand. The training data includes twenty minutes of video. All frames had to be labeled as one of the seven activities.

The algorithm received no additional information about the test sequence except for the raw video stream. The metrics that follow are calculated on a frame by frame basis. The algorithm was forced to label all frames. The complete test sequence was eight minutes long.

6.1 Information Gain

Figure 3 presents the information gain of each component, that is, the reduction in entropy of the confusion matrices as different components are added to the DMLN. Here, an asterisk denotes that Sentence 1 from Table 1 was included in the DMLN. For reference, ground truth represents an information gain of 2.8.

Components	Information Gain
POSE	0.4
PRESENT	1.3
MOVEMENT	0.5
POSE *	1.7
PRESENT *	2.0
MOVEMENT *	0
POSE + PRESENT *	2.5
POSE + MOVEMENT *	1.8
PRESENT + MOVEMENT *	2.3
POSE + PRESENT + MOVEMENT *	2.6
Ground Truth	2.8

Fig. 3. Information Gain from Different Components

6.2 Confusion Matrices

Figure 4 shows a confusion matrix for pose detection. Confusing drinking and eating is the most common mistake and for good reason – we lift food to our mouths without regard to its phase. In mousing and talking, we hold roughly similar sized objects in our hands, casting a unique signature to the pose detection system. The difference in location only comes in when movement location is considered.

Figure 5 shows a confusion matrix for movement location detection. Movement fares poorly but the confusion matrix holds interesting clues. It works well for mousing, which is predominantly in the lower right of the image as all of our subjects were right handed. Activities such as writing, reading, and typing were found to be predominantly in the bottom half of the screen while drinking and talking were in the top half.

	WR	RE	EA	DR	TA	TY	MO
Writing	55	5	7	2	14	7	10
Reading	7	64	9	0	15	2	1
Eating	5	14	46	7	17	8	3
Drinking	15	12	32	9	13	5	13
Talking	21	8	3	2	40	17	9
Typing	11	2	24	3	27	26	8
Mousing	23	3	9	4	31	17	13

Fig. 4. Confusion Matrix for Pose Detection

6.3 Accuracy

Since many of the computer vision components are computationally expensive, the processing time is linearly dependent on both the resolution and frame rate. While lowering the resolution can make object recognition impossible, the frame

	WR	RE	EA	DR	TA	TY	MO
Writing	3	0	0	7	1	89	0
Reading	0	0	0	26	7	67	0
Eating	0	0	0	33	16	44	7
Drinking	0	0	0	81	1	18	0
Talking	0	0	0	69	22	9	0
Typing	1	0	0	0	0	99	0
Mousing	0	0	0	11	0	24	65

Fig. 5. Confusion Matrix for Movement Location Detection

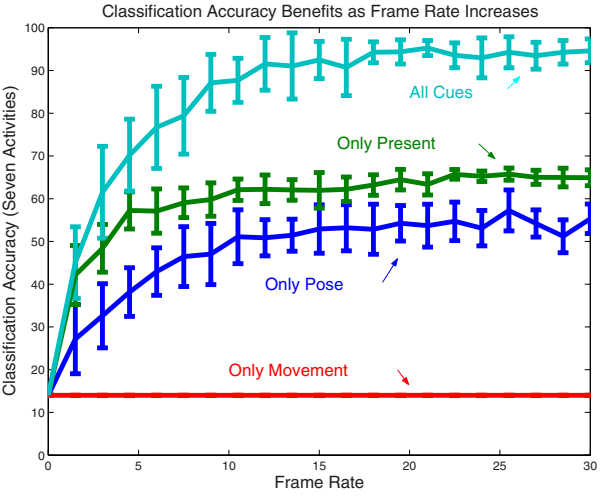


Fig. 6. Classification Accuracy as Frame Rate Increases

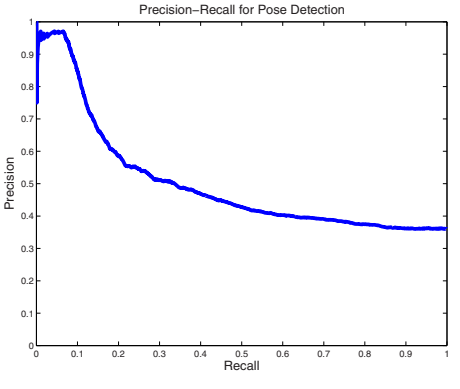


Fig. 7. Precision-Recall for Pose Detection

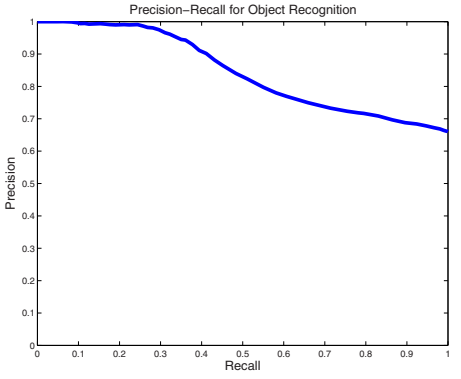


Fig. 8. Precision-Recall for Object Recognition

Fig. 9. An example where the pose detector correctly classifies the image but the object recognizer finds neither the pen nor the paper (left) and an example where the object detector finds the book but the pose detector misclassifies the activity (right)

rate provides a more balanced trade-off between computation time and performance. The effects of this balance is explored in Figure 6.

This figure also shows an alternate view of the value of each component after being processed by the DMLN as well as the benefit of bringing all the cues together. By itself, pose recognition could only classify 55% of the frames cor-

rectly; object recognition could only classify 65% of the frames correctly; and movement location could classify only 14% of the frames correctly. In tandem however, they are able to classify 95% of the frames correctly. This is the benefit of using multiple cues.

Figures 7 and 8 show the precision-recall graphs for pose detection and object recognition, respectively. These graphs highlight how far the field has progressed and the potency of [32] and [38]. They also show the benefit of using a DMLN as the latter incorporates information in proportion to the pose detection and object recognition algorithms' confidences.

Figure 9 show examples of where the pose detector works but the object recognizer fails and vice versa. This speaks to the need for multiple cues as no single cue will suffice in all situations.

7 Related Work

Pose detection seeks to identify the positions of a person's body parts from images without the use of motion capture devices or other artificial markers. It is a difficult problem in general because of differences in people's appearances, self-occlusion, self-shadows, and non-Lambertian clothing. Despite these challenges, several promising approaches have emerged [27], [22], [33], [28], [34], [32]. Three basic approaches typify the field. The first (e.g. [28]) takes a bottom-up approach, first identifying body parts and building them up into complete poses. The second (e.g. [34]) takes a top-down approach, performing joint optimization on the entire image at once. The third, introduced by [32], opts for a memory-based approach, recognizing pose by comparing images to existing training examples.

Object recognition and localization are equally difficult problems but we see substantial progress here as well. The challenges here include occlusion, illumination inconsistency, differences in viewpoint, intra-class variance, and clutter. Most techniques focus on either features [37], [38], [19], [30] or shapes [6], [21]. The advent of the SIFT descriptor [14] marked a great step forward for same-object recognition and most feature based approaches use SIFT or similar descriptors in a bag of words (e.g. [38]) or constellation (e.g. [11]) setting. Shape based approaches work well for different objects of the same category. The geometric blur feature descriptor [5] has proven especially effective here.

The use of multiple cues for object detection has primarily been explored by Torralba and colleagues in [20], [18], [35], [36]. They explore different probabilistic models in these works but the central theme throughout is to leverage hypothesized location information to place a prior over possible objects. They do this both to increase classification accuracy as well as improve running time. [16] follows a similar approach but focuses on integrating speech and pose.

Activity recognition [25], [16], [13], [7] has seen growing interest. [25] is similar to this work in their notion of activities but focuses on more complicated tasks (e.g. infant care) and uses RFID-tagged objects instead of cameras. [7] offers an elegant algorithm for detecting irregularities without requiring any labeling. [13] is also similar to this work in their notion of activities but focuses on substantial

movement (e.g. going to the supermarket) and uses GPS data. None of these approaches however combine multiple cues, relying instead on a single source of information.

8 Future Work

Our results are promising and our framework makes it straightforward to extend our system with additional capabilities. This includes additional cues such as object location, perceived sound, and scene classification. We also want to evaluate our system on a non-stationary camera and in a wider variety of environments. Lastly, we want to explore integrating our system with a mobile robot.

9 Conclusion

In this paper, we introduced a first-order probabilistic model that combines multiple cues to classify human activities from video data accurately and robustly. The model we presented is compact, requiring only fifteen sentences of first-order logic grouped as a Dynamic Markov Logic Network (DMLNs) to implement the probabilistic model and leveraging existing state-of-the-art work in pose detection and object recognition.

Our results show that the algorithm performs well in a realistic office setting with background clutter, natural illumination, different people, and partial occlusion. It is robust against intra- and inter-person variance. We have shown promising results on classification accuracy, information gain, precision-recall, and confusion matrices.

References

1. `http://alchemy.cs.washington.edu/`
2. Allen, D., Darwiche, A.: New advances in inference by recursive conditioning. In: UAI 2003 (2003)
3. Bacchus, F., Dalmao, S., Pitassi, T.: Value elimination: Bayesian inference via backtracking search. In: UAI 2003 (2003)
4. Belongie, S., Malik, J., Puzicha, J.: Shape context: A new descriptor for shape matching and object recognition. In: NIPS 2000 (2000)
5. Berg, A.: Shape Matching and Object Recognition. PhD thesis, University of California, Berkeley, (Adviser-Jitendra Malik) (2005)
6. Berg, A., Berg, T., Malik, J.: Shape matching and object recognition using low distortion correspondence. In: CVPR 2005 (2005)
7. Boiman, O., Irani, M.: Detecting irregularities in images and in video. In: ICCV 2005 (2005)
8. Dechter, R., Mateescu, R.: Mixtures of deterministic-probabilistic networks and their and/or search space. In: UAI 2004 (2004)
9. Doucet, A., Freitas, N., Murphy, K., Russell, S.: Rao-blackwellised particle filtering for dynamic bayesian networks. In: UAI 2000 (2000)

10. Enderton, H.: A Mathematical Introduction to Logic. Academic Press, Inc, Florida (1972)
11. Fei-Fei, L., Fergus, R., Perona, P.: One-shot learning of object categories. In: PAMI 2006 (2006)
12. Huang, C., Ai, H., Li, Y., Lao, S.: Vector boosting for rotation invariant multi-view face detection. In: ICCV 2005 (2005)
13. Liao, L., Fox, D., Kautz, H.: Extracting places and activities from gps traces using hierarchical conditional random fields. In: IJRR 2007 (2007)
14. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV 60(2) (2004)
15. McAllester, D., Collins, M., Pereira, F.: Case-factor diagrams for structured probabilistic modeling. In: UAI 2004 (2004)
16. Morency, L., Sidner, C., Lee, C., Darrell, T.: The role of context in head gesture recognition. In: AAAI 2006 (2006)
17. Mori, G., Ren, X., Efros, A., Malik, J.: Recovering human body configurations: combining segmentation and recognition. In: CVPR 2004 (2004)
18. Murphy, K., Torralba, A., Freeman, W.: Using the forest to see the trees: A graphical model relating features, objects, and scenes. In: NIPS 2003 (2003)
19. Mutch, J., Lowe, D.: Multiclass object recognition with sparse, localized features. In: CVPR 2006 (2006)
20. Oliva, A., Torralba, A.: Building the gist of a scene: The role of global image features in recognition. Progress in Brain Research: Visual Perception 155 (2006)
21. Opelt, A., Pinz, A., Zisserman, A.: A boundary fragment model for object detection. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 575–588. Springer, Heidelberg (2006)
22. Ormoneit, D., Black, M., Hastie, T., Kjellstrom, H.: Representing cyclic human motion using function analysis. In: IVC 2005 (2005)
23. Pasula, H., Russell, S.: Approximate inference for first-order probabilistic languages. In: IJCAI 2001 (2001)
24. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco (1988)
25. Philipose, M., Fishkin, K., Perkowitz, M., Patterson, D., Fox, D., Kautz, H., Haehnel, D.: Inferring activities from interactions with objects. In: IEEE-PC 2004 (2004)
26. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition, pp. 267–296 (1990)
27. Ramanan, D., Forsyth, D., Zisserman, A.: Strike a pose: Tracking people by finding stylized poses. In: CVPR 2005 (2005)
28. Ren, X., Berg, A., Malik, J.: Recovering human body configurations using pairwise constraints between parts. In: ICCV 2005 (2005)
29. Richardson, M., Domingos, P.: Markov logic networks. Mach. Learn. 62(1-2) (2006)
30. Russell, B.C., Efros, A.A., Sivic, J., Freeman, W., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: CVPR 2006 (2006)
31. Sanghai, S., Domingos, P., Weld, D.: Learning models of relational stochastic processes. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 715–723. Springer, Heidelberg (2005)
32. Shakhnarovich, G., Viola, P., Darrell, T.: Fast pose estimation with parameter-sensitive hashing. In: ICCV 2003 (2003)

33. Sidenbladh, H., Black, M.: Learning the statistics of people in images and video. IJCV 54(1-3) (2003)
34. Sigal, L., Black, M.: Predicting 3d people from 2d pictures. In: Perales, F.J., Fisher, R.B. (eds.) AMDO 2006. LNCS, vol. 4069, pp. 185–195. Springer, Heidelberg (2006)
35. Torralba. A.: Contextual priming for object detection. International Journal of Computer Vision 53(2) (2003)
36. Torralba, A., Murphy, K.: Context-based vision system for place and object recognition. In: ICCV 2003 (2003)
37. Viola, P., Jones, M.: Robust real time object detection. In: SCTV 2001 (2001)
38. Wang, S., Quattoni, A., Morency, L., Demirdjian, D., Darrell, T.: Hidden conditional random fields for gesture recognition. In: CVPR 2006 (2006)
39. Wei, W., Erenrich, J., Selman, B.: Towards efficient sampling: Exploiting random walk strategies. In: AAAI 2004 (2004)

Human Action Recognition Using Distribution of Oriented Rectangular Patches

Nazlı İkizler and Pınar Duygulu

Dept. of Computer Engineering, Bilkent University, Ankara, Turkey
{inazli,duygulu}@cs.bilkent.edu.tr

Abstract. We describe a "bag-of-rectangles" method for representing and recognizing human actions in videos. In this method, each human pose in an action sequence is represented by oriented rectangular patches extracted over the whole body. Then, spatial oriented histograms are formed to represent the distribution of these rectangular patches. In order to carry the information from the spatial domain described by the bag-of-rectangles descriptor to temporal domain for recognition of the actions, four different methods are proposed. These are namely, (i) frame by frame voting, which recognizes the actions by matching the descriptors of each frame, (ii) global histogramming, which extends the idea of Motion Energy Image proposed by Bobick and Davis by rectangular patches, (iii) a classifier based approach using SVMs, and (iv) adaptation of Dynamic Time Warping on the temporal representation of the descriptor. The detailed experiments are carried out on the action dataset of Blank et. al. High success rates (100%) prove that with a very simple and compact representation, we can achieve robust recognition of human actions, compared to complex representations.

1 Introduction

Understanding human motion is one of the appealing, yet challenging problems of computer vision. Reliable and effective solutions to this problem can serve many areas, ranging from human-computer interaction to security surveillance. However, although tracking is now a usable technology, understanding what people are doing is still at its infancy.

Human action recognition has been a widely studied topic (for extensive reviews see [12,8]). Yet, the solutions to the problem are very premature and very specific to dataset at hand.

For human motion understanding in videos, there are three major approaches: First, one can use temporal logics to represent crucial order relations between states that constrain activities. Examples to such approaches include Pinhanez and Bobick [19,20], describing a method based on interval algebra, and Siskind [24] describing methods to infer activities related to objects using a form of logical inference.

Second, one can use spatio-temporal templates to identify instances of activities. Spatio-temporal patterns date at least to Polana and Nelson [21]. Thinking

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 271–284, 2007.

actions as such spatio-temporal templates is made famous by Bobick and Davis [2]. They introduce Motion-Energy-Image and Motion-History-Image templates for recognizing different motions. Efros *et al.* [5] use a motion descriptor based on optical flow of a spatio-temporal volume. Blank *et al.* [1] also define actions as space-time shapes, making use of Poisson distributions to define the details of such shapes.

Third general approach to recognize human motion is to use models of dynamics, such as hidden markov models ([3], [27], [18]), conditional random fields [26], finite state models [11,10]. These (mostly generative) models rely on modeling the details of the action dynamics and need lots of training data to build effective models. Ikizler and Forsyth [13] show how to make use of motion capture data in such a case.

We argue that, human pose encapsulates many useful clues for recognizing the ongoing activity. Actions can mostly be represented by the configuration of the body parts, before building complex models for understanding the dynamics. Using this idea, we focus on building a pose descriptor which can be used to discriminate actions. Unlike most of the methods that use complex modeling of the body configurations, we follow the analogy of Forsyth *et al.* [7], which represents the body as a set of rectangles, and explore the layout of these rectangles.

Our pose descriptor is based on a basic intuition: human body can be represented by a collection of oriented rectangles in the spatial domain and the orientations of these rectangles form a signature for each action. Rather than detecting and learning the exact configuration of body parts, we are only interested in the distribution of the rectangular regions which may be the candidates for the body parts.

This idea follows from the bag-of-words approach, where the images are represented by collection of regions, ignoring their spatial relationships. Bag-of-words approach – which is adapted from text retrieval literature – has shown to be successful for object and scene recognition [6,25] and for annotation and retrieval of large image and video collections [16,28]. In such approaches, the images are represented by the distribution of words from a fixed visual vocabulary (i.e. image patches) which is usually obtained by vector quantization of visual features.

Histogramming is an old trick that has been frequently used in computer vision research. For action recognition, Freeman and Roth [9] used orientation histograms for hand gesture recognition. Recently, Dalal and Triggs used histograms of oriented gradients (HOGs) for human detection in images [4], which is shown to be quite successful.

Our main contribution is to adapt the bag-of-words approach for action recognition, by considering the distribution of higher level rectangular patches which are candidates for body parts.

In the following, we first describe our "bag-of-rectangles" pose descriptor, which represents the human figures as a distribution of oriented rectangular patches. Then, we utilize four different methods to recognize the actions. These are namely; frame by frame voting, global histogramming, SVM classification

and Dynamic Time Warping. The detailed experiments are carried out on the data set of Blank *et al.* [1].

2 Bag-of-Rectangles Method

Following the body plan analogy of Forsyth *et al.* [7], we represent human body as a collection of rectangular patches and we base our motion understanding approach on the fact that orientations and positions of these rectangles change over time w.r.t. actions carried out. With this intuition, our algorithm first extracts rectangular patches over the human figure available in each frame, and then forms a spatial histogram of these rectangles by grouping over orientations. We then evaluate the changes of these histograms over time.

More specifically, given the video, first, the tracker identifies the location of the subject. Then, the bounding box around its silhouette is extracted. This bounding box is then divided into a $N \times N$ equal-sized spatial bins. While forming these spatial bins, the ratio between the body parts, i.e. head torso and legs, is taken into account. At each time t, a pose is represented with a histogram H_t formed based on the orientations of the rectangles in each spatial bin. This process is depicted in Fig. 1.

Having formed the spatio-temporal rectangle histograms for each video, we match any newly seen sequence to the examples at hand and label the videos accordingly. We now describe the steps of our method in greater detail.

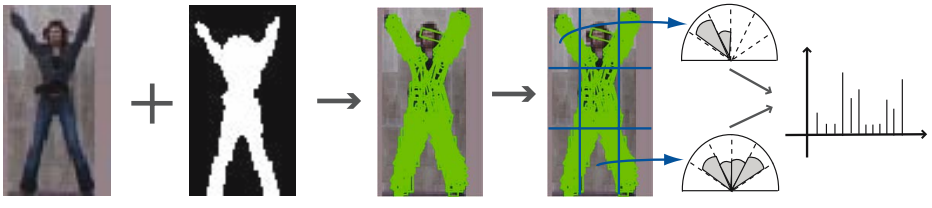


Fig. 1. Here, feature extraction stage of our approach is shown (this figure is best viewed in color). First, the human figure in each frame is extracted using background subtraction or an appropriate tracker. Using these silhouettes, we search for the rectangular patches that can be candidates of limbs. We do not discriminate between legs and arms here. Then, we divide the bounding box around the silhouette into an equal-sized grid and compute the histograms of the oriented rectangles inside each region. We form our feature vector by combining the histograms coming from each subregion.

2.1 Rectangle Extraction

For describing the human pose, we make use of rectangular patches. These patches are extracted in the following way:

1) The tracker fires a response for the human figure. This is usually done using a foreground-background discrimination method. The simplest approach is to apply background subtraction, after forming a dependable model of the

background. The reader is referred to [8] for a detailed overview of the subject. In our experiments, we use the results of a background subtraction scheme (the extracted masks by Blank *et al.*) to localize the subject in motion. Note that any other method that extracts the silhouette of the subject will work just fine.

2) We then search for rectangular regions over the human silhouette using convolution of a rectangular filter on different orientations and scales. We make use of undirected rectangular filters, following Ramanan *et al.* [22]. The search is performed using 12 tilting angles, which are 15° apart, covering a search space of 180°. Note that since we don't have the directional information of these rectangle patches, orientations do not cover 360° but its half. To tolerate the difference in the limb sizes and varying camera distances to the subject, we perform the rectangle convolution over multiple scales.

More formally, we form a zero-padded rectangular Gaussian filter G_{rect} and produce the rectangular regions $R(x, y)$ by means of the convolution of the binary silhouette image $I(x, y)$ with this rectangle filter G_{rect}.

$$R(x, y) = G_{rect}(x, y) \circ I(x, y) \tag{1}$$

where G_{rect} is zero-padded rectangular patch of a 2-D Gaussian $G(x, y)$

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{2}$$

Higher response areas to this filter are more likely to include patches of particular kind. The filters used are shown in Fig. 2.

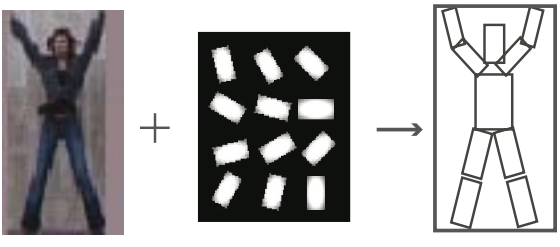
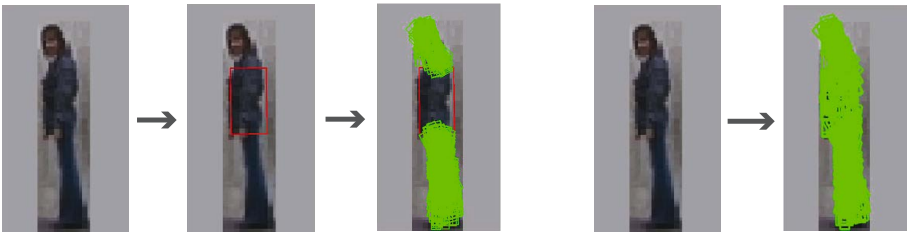


Fig. 2. The rectangular filtering process is shown. We use zero-padded Gaussian filters with 15° tilted orientations over the human silhouette. We search over various scales, without discriminating between different body parts. The perfect rectangular search for the given human subject would result in the tree structure to the right.

To tolerate noise and imperfect silhouette extraction, this rectangle search allows a portion of the candidate regions to remain non-responsive to the filters. Regions that have low overall responses are eliminated this way. Then, we select k of the remaining candidate regions of each scale by random sampling (we used $k = 300$).

One can also perform a special search for the torso rectangle, which is considerably larger than limb rectangles and omit this torso region while searching

for the remaining body parts and then form rectangular histograms. Example images of this rectangular search is given in Fig. 3. In (a) torso is excluded, in (b) the rectangles are searched over the whole silhouette. We evaluate the effects of such a torso exclusion step in the experiments section.



(a) (b)

Fig. 3. Rectangle detection with and without torso detection (best viewed in color). In (a) first, the torso region is detected. This is done by applying a larger rectangular filter and taking the mean of the responses. After finding the torso, the remaining silhouette is examined for remaining limb rectangles. In (b) the whole silhouette is used in the rectangle extraction phase.

Rectangle extraction phase results in $\sim$ 1000 rectangles per frame. While forming the histogram, one can use all of these rectangles or select representative rectangles for each limb. The rectangles which cover the silhouette as much as possible and have high responses to rectangular filters are considered as the representative ones. To achieve these constraints, the higher response candidates that are more than a specified distance apart from each other are selected. By this way, rectangle count is reduced to $\sim$ 10 rectangles per frame. Figure 4 shows this process. Although reducing rectangles gives a more compact representation, it supresses valuable information about the distribution density of the rectangles, making the approach more prone to noise. The experiments (Sect. 4) show the outcomes of such an elimination.

2.2 Pose Descriptor - Histograms of Oriented Rectangles

After finding the rectangular regions of the human body, in order to define the pose, we propose a simple pose descriptor, which is the Histogram of Oriented Rectangles (HOR). We calculate the histogram of extracted rectangular patches based on their orientations. The rectangles are histogrammed over 15° orientations resulting in 12 circular bins. In order to incorporate spatial information of the human body, we evaluate these circular histograms within a $N \times N$ grid placed over the whole body. Our experiments show that $N = 3$ gives the best results. We form this grid by splitting the silhouette over the y-dimension based on the length of the legs. The area covering the silhouette is divided into equal-sized bins from bottom to up and left to right (see Fig. 5 for details). Note that,

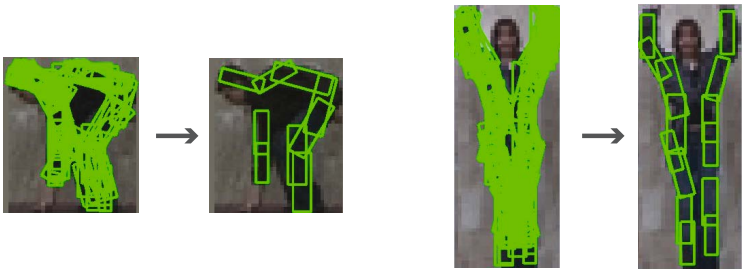


Fig. 4. Rectangle elimination process is shown. Left is a frame from the `bend` action and right is a frame from the `two-hands-wave` action. Rectangles maximizing the coverage area of the silhouette and the rectangular filter response are selected as representatives, eliminating the rest. By this way, rectangle count can be reduced to acquire a more compact representation of the body.

by this way, we let some space to the top part of the head, to allow action space for the arms (for actions like reaching, waving, etc.).

We have also evaluated effect of using 30° orientation bins and 2×2 grid, which have more concise feature representations, but coarser detail of the human pose. We show the corresponding results in Sect. 4.

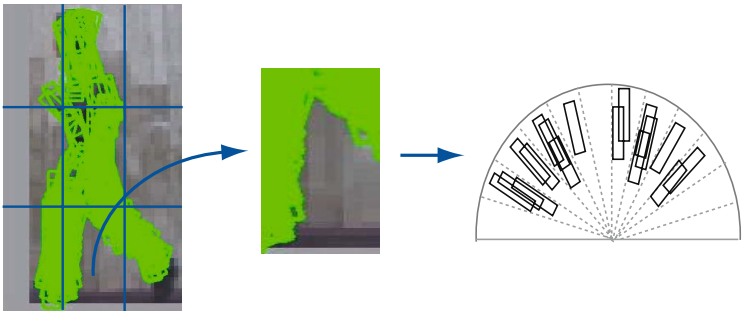


Fig. 5. Details of histogram of oriented rectangles (HORs). The bounding box around the human figure is divided into an $N \times N$ grid (in this case, 3×3) and the HOR from a single spatial bin is magnified. The resulting feature vector is a concatenation of the HORs from each spatial bin.

3 Recognizing Actions with Bag-of-Rectangles

After calculating the pose descriptors for each frame, we perform action classification in a supervised manner. There are four methods we tried in order to evaluate the performance of our pose descriptor.

3.1 Frame by Frame Voting

The simplest scheme we utilize is to perform matching based on single frames, ignoring dynamics of the sequence. That is, for each test instance frame, we

find the closest frame in the training set and employ a voting throughout the sequence. The distance between frames is calculated using Chi-square distance between the histograms (as in [14]). Each frame with the histogram H_i is labelled with the class of the frame having histogram H_j that has the smallest distance χ^2 such that

$$\chi^2(H_i, H_j) = \frac{1}{2} \sum_n \frac{(H_i(n) - H_j(n))^2}{H_i(n) + H_j(n)} \tag{3}$$

We should note that both χ^2 and L_2 distance functions are very prone to noise, because a slight shift of the human center may cause in the different binning of the rectangles, and therefore, large fluctuations in distance. One can utilize Earth Mover's Distance [23] or Diffusion Distance [15] which are shown to be more efficient for histogram comparison in the presence of such shifts by taking the distances between bins into account.

3.2 Global Histogramming

Global histogramming is similar to the Motion Energy Image (MEI) proposed by Bobick and Davis [2]. In this method, we sum up all spatial histograms of oriented rectangles through the whole sequence and form a single compact representation for the entire video. This is simply done by collapsing all time information into single dimension by summing the histograms and forming a global histogram H_{global} such that

$$H_{global}(d) = \sum_t H(d, t) \tag{4}$$

for each dimension d of the histogram. Each test instance's H_{global} is compared to that of the training instances using χ^2 distance and the closest match's label is reported. The corresponding global images are shown in Fig. 6.

3.3 SVM Classification

We also evaluate the performance of SVM-based classification with our pose-descriptor. We trained separate SVM classifiers for each action. These SVM classifiers are formed using *RBF* kernels over snippets of frames using a windowing approach. A grid search over the parameter space of the SVM classifiers is done and the best classifiers are selected using 10-fold cross validation. In our windowing approach, the sequence is segmented into k-length chunks with some overlapping ratio o, then these chunks are classified separately (we achieved the best results with $k = 15$, and $o = 3$). The whole sequence is then labelled with the most frequent action class among its chunks.

3.4 Dynamic Time Warping

Since the periods of the actions are not uniform, comparing sequences is not straightforward. In the case of human actions, the same action can be performed

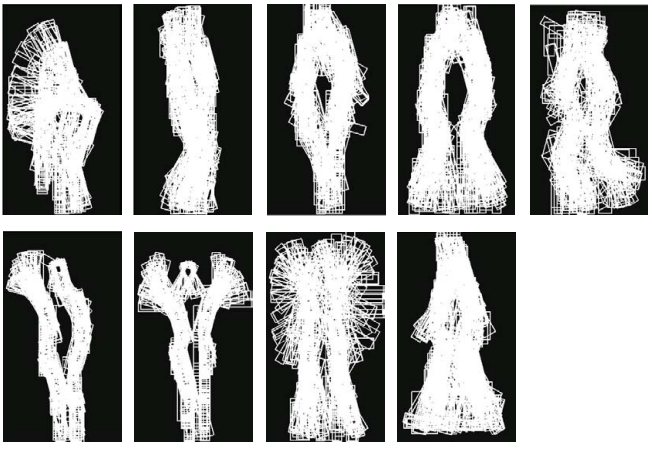


Fig. 6. Global histograms are generated by summing up all the sequence and forming the spatial histograms of oriented rectangles from these global images. In this figure, global images after the extraction of the rectangular patches are shown for 9 separate action classes.**Top row:** `bend, jump, jump in place, gallop sideways` and `run` actions. **Bottom row:** `one-hand wave, two-hands wave, jumpjack` and `walk` actions. These images resemble to Motion Energy Images introduced by [2], however we do not use these shapes. Instead, we form the global spatial histogram of the oriented rectangles as our feature vector.

in different speeds, resulting the sequence to be expanded or shrinked in time. In order to eliminate such effects of different speeds and to perform robust comparison, the sequences need to be aligned.

Dynamic time warping (DTW) is a method to compare two time series which may be different in length. DTW operates by trying to find the optimal alignment between two time series by means of dynamic programming. Figure 7 shows an example of time warping process. The time axes are warped in such a way that samples of the corresponding points are aligned.

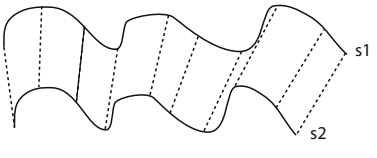


Fig. 7. Dynamic Time Warping (DTW) process in 1-d time series: The distance between corresponding sample points of two sequences $s1$ and $s2$ are calculated using dynamic programming and the time axes are warped in such a way that the corresponding sample points are aligned

More specifically, given two time series $x_1 \dots x_n$ and $y_1 \dots y_m$, the distance $D(i, j)$ is calculated with

$$D(i,j) = \left\{ \begin{array}{l} D(i,j-1) \\ D(i-1,j) \\ D(i-1,j-1) \end{array} \right\} + d(x_i, y_j) \tag{5}$$

where $d(.,.)$ is the local distance function specific to application. In our implementation, we have chosen $d(.,.)$ as the χ^2 distance function as in Equation 3.

We use dynamic time warping along each dimension of the histograms separately. As shown in Fig. 8, we take each 1-d series of the histogram bins of the test video X and compute the DTW distance $D(X(d), Y(d))$ to the corresponding 1-d series of the training instance Y. We then sum up the distances of all dimensions to compute the global DTW distance (D_{global}) between the videos. We label the test video with the label of the training instance that has the smallest D_{global} such that,

$$D_{global}(X,Y) = \sum_{d=1}^{M} D(X(d), Y(d)) \tag{6}$$

where M is the total number of bins in the histograms. While doing this, we exclude the top k of the distances to reduce the effect of noise introduced by shifted bins and inaccurate rectangle regions. We choose k based on the size of the feature vector such that $k = \lfloor \#num\_bins/2 \rfloor$ where $\#num\_bins$ is the total number of bins of the spatial grid.

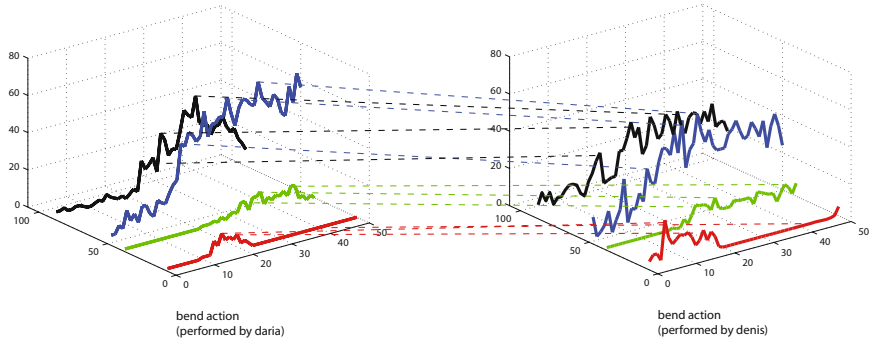


Fig. 8. Dynamic Time Warping (DTW) over 2D histograms: We calculate DTW distances between the histograms by evaluating DTW cost over single dimensions separately and summing up all costs to get a global distance between sequences. Here, histograms of two bend actions performed by different actors are shown. We try to align these sequences along each histogram dimension by DTW and report sum of the smallest distances. Note that, separate alignment of each histogram bin also allows us to handle the fluctuations in distinct body part speeds.

4 Experimental Results

4.1 Dataset

For experimental evaluation we use dataset that Blank *et al.* introduced in [1]. We used the same set of actions as [1], which is a set of 9 actions: walk, run, jump, gallop sideways, bend, one-hand wave, two-hands wave, jump in place and jumping jack. We used extracted masks provided to localize the human figures in each image. These masks have been obtained using background subtraction. We test the effectiveness of our method using leave-one-out cross validation.

4.2 Rectangle Elimination

In Table 1, the effect of of rectangle elimination is shown. We observe that using the extracted rectangles as is – without selecting representative ones – give more accurate action recognition. Although we have a more compact representation with the representative rectangles, the probability distribution of the limb locations is more accurately estimated when we use all samples of rectangles in the histogram calculation.

Table 1. The accuracies of the matching methods with respect to rectangle elimination, with 15° angular bins over 3×3 grid. Although with eliminated rectangles we have a more sparse representation of the body, rectangles wrongly extracted become more significant in that case. Using all extracted rectangles gives a more robust estimation about where the actual body parts are, since body part regions are likely to produce more rectangles, resulting in denser rectangular regions.

Matching Method	Eliminated	All
FrameVoting	0.6173	0.9630
GlobalHist	0.9506	0.9630
SVM	0.9383	0.9506
DTW	0.9877	**1.0000**

4.3 Torso Detection

We can make a separate search for the torso, omit this region and form our pose descriptors based only on the candidate limb locations. In Table 2, we show the effect of torso detection on the overall accuracies. We observe that with frame voting and global histogramming methods, torso detection and exclusion helps, however, SVM and DTW classifiers suffer from slight performance degradation.

4.4 Granularity of Angular Bins

We also evaluated the choice of orientation angles when forming the histogram. Table 3 shows the results using 15° angular bins versus 30° bins. Our results indicate that there is a slight loss of information when we go from fine level orientations (i.e. 15° bins) to a coarser level (30°).

Table 2. The accuracies of the matching methods with respect to torso detection. The results presented here are over the eliminated rectangles with 15° angular bins and 3×3 grid. Note that torso detection can be useful in the case of FrameVoting and GlobalHist methods whereas SVM and DTW methods suffer from a slight performance loss.

Matching Method	No Torso	With Torso
FrameVoting	0.6173	0.6790
GlobalHist	0.9506	0.9630
SVM	0.9383	0.9259
DTW	0.9877	0.9506

Table 3. The accuracies of the matching methods with respect to angular bins. The original rectangle search is done with 15° tilted rectangular filters. To form 30° histograms, we group rectangles that fall into the same angular bins. These results demonstrate that as we move from fine to coarser scale of angles, there is a slight loss of information and thus 30° HORs become less discriminative than 15° HORs.

Matching Method	15°	30°
FrameVoting	0.9630	0.9506
GlobalHist	0.9630	0.9383
SVM	0.9506	0.9383
DTW	**1.0000**	0.9506

4.5 Grid Size

When forming the histograms of oriented rectangles, we place an $N \times N$ grid over the silhouette of the subject and form spatial histograms for each grid region. The choice of N effects the size of the feature vector (thus execution time of the matching), and the level of detail of the descriptor. Table 4 compares using 2×2 grid versus 3×3 grid. One can try further levels of partitioning, even form pyramids of these partitions. However, too dense partitioning will not make sense, since the subregions should be large enough to contain rectangle patches. Our results over this dataset indicate that, 3×3 gives better performance compared to 2×2. However, if execution time is crucial, choice of $N = 2$ will still work to a certain degree of performance.

4.6 Overall Evaluation and Comparison

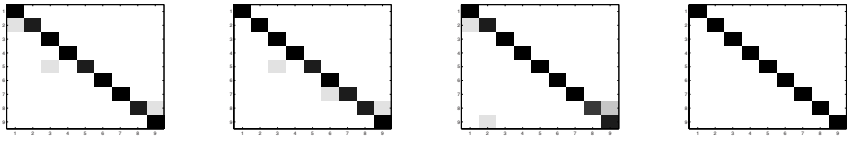
Overall, we achieve the best results with DTW matching. This is not surprising, because the subjects do not perform actions with uniform speeds and lengths. Thus, the sequences need aligning. DTW matching accomplishes this alignment over the bins of the histogram separately, making alignment of limb movements also possible. Action speed differences between body parts are handled this way.

We reach a perfect accuracy (100%) over Blank action dataset, using all extracted rectangles and the torso region with 15° angular bins over a 3×3 partitioning. Figure 9 shows the confusion matrices of each method. Blank *et al.*

Table 4. The accuracies of the matching methods with respect to $N \times N$ grids (with 15° angular bins, no rectangle or torso elimination). We have compared 2×2 and 3×3 partition grids. Our results show that 3×3 grid is more effective when forming our oriented-rectangles based pose descriptor.

Matching Method	2×2	3×3
FrameVoting	0.9136	0.9630
GlobalHist	0.8765	0.9630
SVM	0.9012	0.9506
DTW	0.9136	**1.0000**

report classification error rates of 0.36% and 3.10% for this dataset. Recently, Niebles and Fei Fei [17] evaluate their hierarchial model of spatial and spatio-temporal features over this dataset, acquiring an accuracy of 72.8%.



(a) (b) (c) (d)

Fig. 9. Confusion matrices for each matching method using original rectangle distributions with no torso detection and 15° angular bins over a 3×3 grid. (a) Frame by frame voting : 1 jump sequence classified as bend, 1 one-hand wave sequence classified as jump-in-place and 1 run sequence misclassified as walk. (b) Global histogramming : 1 one-hand wave sequence misclassified as jump-in-place, 1 jumpjack sequence misclassified as two-hands-wave and 1 run sequence misclassified as walk. (c) SVM classification : 1 jump sequence is classified as bend, 2 run sequences classified as walk, 1 run sequence misclassified as jump. (d) DTW classification achieves 100% accuracy.

We should also note that frame by frame voting and global histogramming with our pose descriptor produce surprisingly good results. This suggests that we can still achieve satisfactory classification rates even if we ignore the time domain and look at the frames separately, or as a whole.

5 Conclusions and Future Work

In this paper, we have approached to the problem of human action recognition from a bag-of-features perspective and proposed a new pose-descriptor based on the orientation of body parts. Our pose-descriptor is simple and effective; we extract the rectangular regions from a human silhouette and form spatial oriented histogram of these rectangles. We show that by effective classification of such histograms, robust human action recognition is possible. We demonstrate

the effectiveness of our method over the dataset of Blank *et al.* [1]. Our results are directly comparable/superior than the results presented over this dataset.

The matching methods we present in this study suggest that we may not need a perfect modeling of the dynamics of human actions in order to reach satisfactory results. The questions behind the success of frame by frame voting or global histogramming methods are: "Do we really need dynamics of an action to recognize it correctly?", or "Can we simply recognize the actions by looking at representative frames or signatures of them?". Our experiments show that human pose encapsulates many useful information for the action itself, therefore, one can start with a good pose estimator, before going into the details of dynamics.

Future work includes application of our pose-descriptor to more complex datasets and still images. We also plan to explore the view-invariance case, by means of orthographic projections of rectangular regions. In addition, we will explore finer scale angular bins with varying spatial formations.

References

1. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. In: ICCV, pp. 1395–1402 (2005)
2. Bobick, A., Davis, J.: The recognition of human movement using temporal templates. IEEE T. Pattern Analysis and Machine Intelligence 23(3), 257–267 (2001)
3. Brand, M., Oliver, N., Pentland, A.: Coupled hidden markov models for complex action recognition. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 994–999. IEEE Computer Society Press, Los Alamitos (1997)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Conf. on Computer Vision and Pattern Recognition, vol. I, pp. 886–893. IEEE Computer Society Press, Los Alamitos (2005)
5. Efros, A.A., Berg, A.C., Mori, G., Malik, J.: Recognizing action at a distance. In: ICCV 2003, pp. 726–733 (2003)
6. Fei-Fei, L., Perona, P.: A bayesian heirarcical model for learning natural scene categories. In: IEEE Conf. on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2005)
7. Forsyth, D., Fleck, M.: Body plans. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 678–683. IEEE Computer Society Press, Los Alamitos (1997)
8. Forsyth, D., Arikan, O., Ikemoto, L., O'Brien, J., Ramanan, D.: Computational studies of human motion i: Tracking and animation. Foundations and Trends in Computer Graphics and Vision 1(2/3) (2006)
9. Freeman, W., Roth, M.: Orientation histograms for hand gesture recognition. In: International Workshop on Automatic Face and Gesture Recognition (1995)
10. Hong, P., Turk, M., Huang, T.: Gesture modeling and recognition using finite state machines. In: Int. Conf. Automatic Face and Gesture Recognition, pp. 410–415 (2000)
11. Hongeng, S., Nevatia, R., Bremond, F.: Video-based event recognition: activity representation and probabilistic recognition methods. Computer Vision and Image Understanding 96(2), 129–162 (2004)
12. Hu, W., Tan, T., Wang, L., Maybank, S.: A survey on visual surveillance of object motion and behaviors. IEEE transactions on systems, man, and cybernetics c: applications and reviews 34(3) (2004)

13. Ikizler, N., Forsyth, D.: Searching video for complex activities with finite state models. In: IEEE Conf. on Computer Vision and Pattern Recognition (2007)
14. Leung, T., Malik, J.: Representing and recognizing the visual appearance of materials using three-dimensional textons. Int. J. Computer Vision 43(1), 29–44 (2001)
15. Ling, H., Okada, K.: Diffusion distance for histogram comparison. In: IEEE Conf. on Computer Vision and Pattern Recognition, vol. 1, pp. 246–253 (2006)
16. Monay, F., Gatica-Perez, D.: Modeling semantic aspects for cross-media image retrieval. IEEE T. Pattern Analysis and Machine Intelligence (accepted for publication)
17. Niebles, J.C., Fei-Fei, L.: A hierarchical model of shape and appearance for human action classification. In: IEEE Conf. on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2007)
18. Oliver, N., Garg, A., Horvitz, E.: Layered representations for learning and inferring office activity from multiple sensory channels. Computer Vision and Image Understanding 96(2), 163–180 (2004)
19. Pinhanez, C., Bobick, A.: Pnf propagation and the detection of actions described by temporal intervals. In: DARPA IU Workshop, pp. 227–234 (1997)
20. Pinhanez, C., Bobick, A.: Human action detection using pnf propagation of temporal constraints. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 898–904. IEEE Computer Society Press, Los Alamitos (1998)
21. Polana, R., Nelson, R.: Detecting activities. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 2–7. IEEE Computer Society Press, Los Alamitos (1993)
22. Ramanan, D., Forsyth, D., Zisserman, A.: Strike a pose: Tracking people by finding stylized poses. In: IEEE Conf. on Computer Vision and Pattern Recognition, vol. I, pp. 271–278. IEEE Computer Society Press, Los Alamitos (2005)
23. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. Int. J. Computer Vision 40(2), 99–121 (2000)
24. Siskind, J.M.: Reconstructing force-dynamic models from video sequences. Artificial Intelligence 151, 91–154 (2003)
25. Sivic, J., Russell, B., Efros, A., Zisserman, A., Freeman, W.: Discovering object categories in image collections. In: Int. Conf. on Computer Vision (2005)
26. Sminchisescu, C., Kanaujia, A., Li, Z., Metaxas, D.: Conditional models for contextual human motion recognition. In: Int. Conf. on Computer Vision, pp. 1808–1815 (2005)
27. Wilson, A., Bobick, A.: Parametric hidden markov models for gesture recognition. IEEE T. Pattern Analysis and Machine Intelligence 21(9), 884–900 (1999)
28. Yu-Gang Jiang, C.-W.N., Yang, J.: Towards optimal bag-of-features for object categorization and semantic video retrieval. In: Int. Conf. Image Video Retrieval (2007)

Human Motion Recognition Using Isomap and Dynamic Time Warping

Jaron Blackburn and Eraldo Ribeiro

Computer Vision and Bio-Inspired Computing Laboratory
Department of Computer Sciences
Florida Institute of Technology
Melbourne, FL 32901, USA
{jblackburn,eribeiro}@fit.edu
http://www.cs.fit.edu/~eribeiro

Abstract. In this paper, we address the problem of recognizing human motion from videos. Human motion recognition is a challenging computer vision problem. In the past ten years, a number of successful approaches based on nonlinear manifold learning have been proposed. However, little attention has been given to the use of isometric feature mapping (Isomap) for human motion recognition. Our contribution in this paper is twofold. First, we demonstrate the applicability of Isomap for dimensionality reduction in human motion recognition. Secondly, we show how an adapted dynamic time warping algorithm (DTW) can be successfully used for matching motion patterns of embedded manifolds. We compare our method to previous works on human motion recognition. Evaluation is performed utilizing an established baseline data set from the web for direct comparison. Finally, our results show that our Isomap-DTW method performs very well for human motion recognition.

Keywords: human motion recognition, non-linear manifold learning, dynamic time warping.

1 Introduction

The automatic recognition of human motion from videos is a challenging research problem in computer vision. The interest in obtaining effective solutions to this problem has increased significantly in the past ten years motivated by both the rise of security concerns and increased affordability of digital video hardware. Recent works in the computer vision literature have proposed a number of successful motion recognition approaches based on nonlinear manifold learning techniques [17,8,23]. Nonlinear manifold learning techniques aim at addressing simultaneously the inherent high-dimensionality and non-linearity of representing human motion patterns. However, within this category of methods, little attention has been given to the use of isometric feature mapping (Isomap) [20]. In this paper, we bridge this gap by proposing a new method for automatic recognition of human motion and actions from single-view videos.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 285–298, 2007.

Our approach uses non-linear manifold learning of human silhouettes in motion. The approach is similar to the ones proposed by [17,8,23]. However, we cast the problem of recognizing human motion as the one of matching motion manifolds. Our matching procedure is based on an adapted multidimensional dynamic time warping (DTW) matching measurement [22,2].

Our contribution in this paper is twofold. First, we demonstrate the applicability of Isomap for dimensionality reduction in human motion recognition. Secondly, we show how an adapted dynamic time warping algorithm can be successfully used for matching motion patterns in the Isomap embedded manifold. To accomplish our goals, we commence by assuming that the observed human motion patterns can be represented by point-wise trajectories in a lower dimensional space using isometric non-linear manifold mapping. Our proposed algorithm starts by learning Isomap representations of known motion patterns from a set of training images. The learning of the manifold projection mapping is accomplished by means of an invertible radial basis function (RBF) mapping as described in [8]. The initial Isomap projection does not encode any temporal relationship between image frames. Temporal information is introduced into the learned manifold after the projection to the manifold space. The nonlinear manifold augmented with temporal information will then form the learned motion pattern to be used for the recognition of novel motion sequences. Finally, recognition is accomplished by means of a nearest-neighbor classification scheme based on a dynamic time warping score. Figure 1 illustrates sample output from each of the three main steps of the method (i.e., Preprocessing, Model Generation, Recognition). The process in the figure is briefly described as follows. A single video-frame post preprocessing is provided as an example of the functionality performed in this step. In the model generation step, the Isomap projection and the addition of time are shown. Additionally, a comparison of the Isomap projection ($\circ$) to the inverse RBF learned projection ($\times$) is illustrated. During the recognition step, the learned projection is used to map the test sequence ($\bullet$) into the lower dimensional space. Finally, the DTW moves the projected data (solid line) to the temporally aligned data (thin dotted line) to perform the match to the template (thick dashed line).

Our experiments show that the use of Isomap with DTW performs very well for human motion recognition. We test our method on a set of standard human motion sequences widely used in the literature. Finally, we provide a comparison between our approach and recently published methods [4,17,3,23]. Specifically, we apply our algorithm to the data set created by [3] for direct comparison to both [3] and [23]. The data set is also similar enough in nature to compare our results to the approaches presented in [17] and the single-view case in [4]. We show that our method obtains superior results to [4,17,3], and obtains the same 100% recognition rate as the Hidden Markov Model method proposed by [23].

The remainder of this paper is organized as follows. In Section 2, we commence by providing a brief survey of the related literature on human motion recognition. Section 3 describes the details of our motion recognition method.

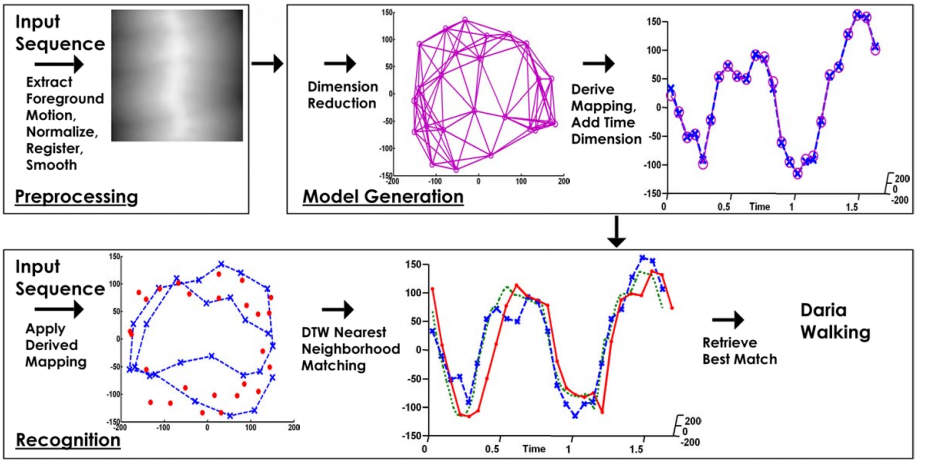


Fig. 1. Motion manifold creation and recognition using DTW. The purple ∘ with solid lines denote a projected training sequence. The blue × with dashed lines denote a learned motion template. The red • with and without solid lines denote a projected test sequence. The green dotted lines denote DTW aligned test data.

Then, in Section 4, we show our preliminary results using the proposed method. Finally, in Section 5, we present our conclusions and directions for future work.

2 Related Literature

The literature on the problem of recognizing human motion from videos sequences is extensive [1,11,17,8,23]. In this paper, we focus ourselves on the methods addressing the specific problem of recognizing human motion from image sequences without the use of markers, tracking devices, or special body suits. In general, such methods can be broadly classified into multiple-view and single-view methods. Multiple-view methods address the motion recognition problem using image sequences obtained from multiple cameras placed at different spatial locations [4,10,18]. The strength of these methods is their power to resolve ambiguous human motion patterns that may result from self-occlusion and viewpoint-driven appearance changes. However, multiple view approaches usually require the availability of synchronized camera systems and controlled camera environments. On the other hand, single-view methods rely only on information provided by a single video camera [3,4,6,8,9,14,15,17,18,23]. Under the single-view assumption, human motion recognition becomes a significantly more challenging and ill-posed problem.

In general, single-view motion recognition is performed using three main steps. The first of these consists of an image processing step in which the image is filtered to reduce the presence of noise (e.g., background, acquisition noise) and to enhance the presence of useful features (e.g., contours, textures, skeletons).

The second step aims at representing the motion information obtained from a sequence of extracted features. The motion information from human activities is inherently both highly non-linear and high-dimensional. As a result, this step will usually try to obtain relevant (i.e., discriminative) motion information using a reduced dimensional space-time representation. The representation can be accomplished, for instance, by making use of explicit measurements on the image to which a pre-determined model is fitted (i.e., skeleton-based methods [18,10], part-segmentation-based methods [15,6,14,9]).

More recently, research in human motion recognition has shifted toward the concept of identifying a motion directly from appearance rather than fitting the visual input to a physical model [4,17,3,23]. Indeed, most of these works have avoided direct feature extraction techniques as they tend to be sensitive to variations such as color, texture, and clothing. Instead, recent work has focused on the use of silhouettes or other high-level abstractions from the raw input data. In this paper, we propose an approach that falls under this later category. Work in silhouette-based human motion recognition can be grouped in terms of the main steps used to approach the problem: image preprocessing, motion pattern representation, and recognition or matching approach.

We begin by discussing the image preprocessing step. This is usually the first step of any recent approach to human motion activity identification. Here, the image foreground (i.e., moving object) is extracted by means of motion segmentation techniques. Standard techniques include the ones based on Motion-History Images (MHI) [4,17,3]. Motion history -based representations allow for simultaneous description of both the dynamics of the motion and the shape of objects. However, as pointed out by Bobick and Davis [4], MHI-based methods are not suited for representing the underlying motion when the observed object returns to similar positions (e.g., cyclic motion patterns). Alternatively, object's silhouette information alone can be used as an input for recognition systems. Wang and Suter [23] used silhouettes as the input to their recognition method. Elgammal and Lee [8] also used silhouettes without motion history. In this paper, we use a similar smoothing technique as the one presented in [8]. However, our distance function representation places a higher weight on the moving object's medial-axis. This reduces the influence of variations in silhouette's contours.

Human motion information is inherently both highly dimensional and complex. Therefore, dimensionality reduction is a standard procedure in the preprocessing of motion data for recognition. Here, the key idea is to find a suitable reduced representation of the motion while maintaining sufficient discriminating data for performing the recognition. To accomplish these goals, past works have used simple data reduction techniques such as principal component analysis (PCA) [17] and Locality Preserving Projections (LPP) [13,23]. The main advantage of these linear approaches is their ability to produce a direct mapping to the embedding space. Nevertheless, the nature of human motion is highly non-linear. Indeed, for complex motions of long duration, recent advances in non-linear dimensionality reduction techniques provide significant improvements of human motion recognition. Techniques in this group include the Isometric feature

mapping (Isomap) [20] and the Local Linear Embedding (LLE) [19]. Evidence of the effectiveness of these non-linear manifold learning methods for human motion recognition has been widely reported in the computer vision literature [23,3,8].

Finally, the recognition step in most motion recognition methods aim at determining the maximum similarity between an unobserved test sequence and pre-learned motion models. Some methods use distance measurements such as the Mahalanobis distance [4] or the Hausdorff distance to establish matches between the learned templates and test sequences [17,3,23]. Methods using the Hausdorff distance are sensitive to non-isometrically similar datasets (i.e., the Hausdorff distance compares each point from one set to every point in the second set regardless of temporal sequence). In order to address this limitation, Wang and Suter [23] propose the use of the Hausdorff distance only as a baseline for a Hidden Markov Model (HMM) matching procedure. Additional important works using HMM for human motion analysis and recognition include [12,5,24]. HMM allows for a principled probabilistic modeling of the temporal sequential information. An alternative way to approach the matching of data sequences is to use Dynamic Time Warping (DTW) [22,2]. DTW has been used in the context of matching data sequences in several applications such as speech recognition, economics, and bio-informatics. DTW provides an approximate similarity measurement while allowing for matching partially identical sequences.

The method proposed in this paper uses an adapted DTW algorithm to perform recognition by matching trajectories on a non-linear manifold space representation. Our paper aims at demonstrating the effectiveness of the Isomap-DTW combination. To the best of our knowledge, this combination has not yet been explored in the human motion recognition literature. In several cases Isomap has been dismissed in favor of Local Linear Embedding or other algorithms mostly due to the greater focus on the local relationship perservation. In other cases Isomap has been dismissed due to the lack of an inverse mapping which other algorithms readily elucidate. The inverse mapping issue has been solved by Elgammal and Lee [8]. Additionally, DTW has also been dismissed in favor of HMM. Our work demonstrates the potential of using Isomap and DTW for matching motion manifolds to accomplish accurate human motion recognition. Next, we describe the details of our motion recognition method.

3 Our Method

In this section, we describe the details of the steps of our method.

3.1 Data Preprocessing

The selection of Isomap for our algorithm imposes a restriction on the input data set. Isomap asymptotically converges for a large class of nonlinear manifolds. The convergence is achieved when the input data has a large enough frequency of coverage within the high dimension space. Consequently, Isomap must be supplied an input data set sufficiently representative to create a meaningful

embedded manifold space. This is a reasonable restriction for any machine learning problem and, given a specific domain, the required amount of input needed for adequate characterization can be obtained via experimentation. Given that a large enough representative set is required, there are two actions that can be taken to aid in preconditioning of the data. The first action is to select a more constrained search space and the second is to generalize the hypothesis sets remaining within the reduced search space. These two techniques aid in decreasing the amount of training data that may be required.

Constraining the Search Space. In many cases the search space can be preconditioned to a much smaller set. One common preconditioning used for images is the reduction of color representation to gray scale representation. Thoughtful manipulation of the search space not only aids in reducing the representative data set needed for learning the manifold space, but can also increase the robustness of the learned mapping.

In the particular case of human motion several recent works established successful results by reducing the space to the silhouette of subjects [3,8]. This discards much of the data associated with internal clothing details, and removes all background data from the search space. The end result focuses the observed dimensionality to strictly the motion performed.

For this particular problem domain, the registration of the silhouettes in the image frames also limits the size of the search space. This preprocessing discards the motion caused by translation and further constrains the space to the motions relative to the internal deformation of the shape. Nevertheless, a simplistic resizing alteration could change the aspect ratio of the subject, and result in an undesirable change to the internal deformation. In our implementation, the registration is performed by isolating the foreground silhouette using a simple background subtraction operation. A bounding box is then constructed for each frame that encompasses the foreground pixels. The largest frame size is chosen to represent the standard frame size for the entire sequence. Finally, all remaining frames are aligned (center pixel) to the center of the standard selected frame.

Generalize the Hypothesis Sets. After the initial search space reduction, generalization is performed by converting the silhouette to a gray level gradient using a distance transform similar to [8]. In our method, we perform the distance transform so that the highest values are assigned to the silhouette's most medial axis points. Once the smoothing is completed, the intensity range in all images is re-scaled to a predefined maximum value (e.g., 255). The result of this preprocessing step is illustrated in Figure 2. Gray scale images are used, however, the color versions illustrate effect on the silhouette's medial axis. The smoothing decreases the variance between subtle differences of similar images, such as those caused by clothing and hair variance. Data sets containing both large volumes and small volumes with significant amount of discriminative features for recognition in smaller volumes may be sensitive to this preprocessing. For human motion, this does not seem to be an issue, and we believe this preprocessing increases the overall robustness of recognition.

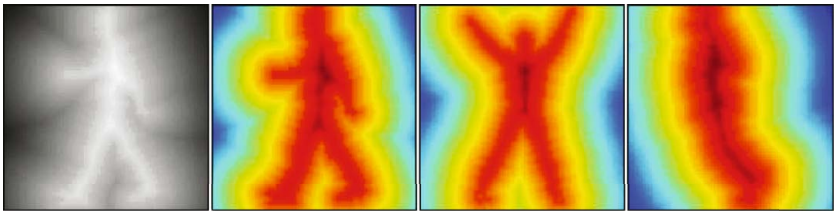


Fig. 2. Sample preprocessed data: Walk (gray scale), Walk, Jack, Jump (color)

3.2 Motion Pattern Learning Using Isomap

In this part of our algorithm, we use the isometric feature mapping or Isomap [20] to obtain template models of the observed motions. Here, our goal is to build a model representation of each motion pattern in our training set. These models will later be used in the matching step to accomplish recognition of unknown motion patterns. The key idea here is to use Isomap as a means of representing the actual intrinsic dimensionality of the analyzed data. Elgammal and Lee [8] used Locally Linear Embedding (LLE) as a manifold learning technique in their motion recognition work. However, Isomap manifolds have been reported to retain more global relationships than its LLE counterpart [7]. This part of our method is divided into two main steps. First, an Isomap manifold is created for each motion available in the training dataset. Secondly, radial basis function mappings are estimated for mapping the learned Isomap manifold space back to the template images. These functions admit an inverse map that allows for the extraction of the manifold embedding for new images. These steps are detailed as follows.

Isometric mapping of silhouette patterns. In this step, we will use Isomap to build a manifold representation of our motion sequence. The input data used by this step is a set of smoothed silhouette images obtained by the preprocessing step of our method. Let $Y = \{y_i \in \mathbb{R}^d, i = 1, \ldots, N\}$ be the set of preprocessed image data (i.e., smoothed silhouette images), and $X = \{x_i \in \mathbb{R}^m, i = 1, \ldots, N\}$ be the corresponding embedding points. The embedded points X are determined using the following three-step Isomap algorithm: **(1)** Create a weighted graph G of points in Y with weights $d_Y(i, j)$ representing the pairwise distance between neighbors. In our algorithm, a neighborhood is defined by the k-nearest neighbors; **(2)** Estimate the pairwise geodesic distances $d_X(i, j)$ between all manifold points by finding the shortest path distances in the graph G. These shortest path distances are denoted by $d_G(i, j)$; **(3)** Finally, apply classical multidimensional scaling (MDS) on D_G to map the data onto an m-dimensional Euclidean space X. It is worth pointing out that $d_X(i, j)$ and $d_Y(i, j)$ are Euclidean pairwise distances within manifold space while $d_G(i, j)$ represents the actual geodesic distances. The coordinate vectors $\mathbf{x}_i$ in X are chosen by minimizing the following L^2 cost function:

$$E = \sqrt{\sum_{ij} \left[\tau\left(d_G(i,j)\right) - \tau\left(d_X\left(i,j\right)\right)\right]^2} \tag{1}$$

where τ in an operator that converts distances to inner products as described in [20]. The use of this operator supports efficiency in the optimization process.

However, the above embedding procedure does not directly allow for the mapping of new images onto the same manifold. In order to address this issue, Elgammal and Lee [8] proposed the use of an approximate invertible mapping from the embedded space to the image space. This mapping is based on radial basis functions. For completeness, this mapping is briefly described next. Further details of this method can be found in [8].

Learning embedded-space-image mappings. The main goal in this step is to obtain an invertible approximate mapping between the embedded manifold space and the image space. Let $t_j \in \mathbb{R}^m, j = 1, \ldots, N_t$ be a set of N_t cluster centers in the embedding space obtained by using a K-Means clustering algorithm. In this paper, we choose N_t such that $N_t = \frac{3}{4}N$. The radial basis function interpolants $f^k : \mathbb{R}^m \rightarrow \mathbb{R}^d$ can be found and satisfy the condition $y_i^k = f^k(x_i)$. Here, k is the k^{th} dimension (pixel) in the image space. More specifically, the interpolant is given by:

$$f^k\left(x\right) = p^k\left(x\right) + \sum_{i=1}^{N_t} w_i^k \phi\left(|x - t_i|\right) \tag{2}$$

Equation 2 can also be written in matrix form as:

$$f\left(x\right) = B \cdot \psi\left(x\right) \tag{3}$$

where B is a $d \times (N_t + m + 1)$ dimensional matrix, and ψ is given by:

$$\psi = \left[\phi\left(|x - t_1|\right) \ldots \phi\left(|x - t_{N_t}|\right) \;\; 1 \;\; x^{\mathsf{T}}\right]^{\mathsf{T}} \tag{4}$$

Finally, B can be obtained by solving the linear system:

$$\begin{pmatrix} A & P_x \\ P_t^{\mathsf{T}} & 0_{(m+1)\times(m+1)} \end{pmatrix} B^{\mathsf{T}} = \begin{pmatrix} Y \\ 0_{(m+1)\times d} \end{pmatrix} \tag{5}$$

where A is $N \times N_t$ matrix with $A_{ij} = \phi\left(|x_i - t_j|\right)$, $i = 1 \ldots N$, $j = 1 \ldots N_t$, ϕ is the thin-plate spline $\phi\left(u\right) = u^2 \log\left(u\right)$, P_x is a $N \times (m+1)$ matrix with i^{th} row $\left[1 \; x_i^{\mathsf{T}}\right]$, and P_t is a $N_t \times (m+1)$ matrix with i^{th} row $\left[1 \; t_i^{\mathsf{T}}\right]$.

The mapping in Equation 5 can be inverted by calculating the Moore-Penrose pseudo-inverse of the matrix B:

$$\psi\left(x\right) = (B^{\mathsf{T}}B)^{-1}B^{\mathsf{T}}y \tag{6}$$

This function can be used to map each training image-frame to the embedded template space. The final motion model manifold is then created by reintroducing the time dimension into the manifold representation. This is accomplished by

assigning each frame its corresponding time from the original sequence. The motion manifold construction is now complete, and test frames can be efficiently converted to each of the template manifold spaces before entering the recognition phase. The recognition step is described as follows.

3.3 Recognition

We perform recognition by means of a matching function based on dynamic time warping [22,2]. We adapted the original DTW framework to allow for the matching of motion patterns in manifold space. The key modifications in the DTW algorithm are the following. First, we interpolate both the model template and test manifolds to have the same number of points. Secondly, we use a multi-dimensional version of the DTW with an adapted scoring system using the basic Sakoe-Chiba band constraint. These few modifications permit the DTW algorithm to adjust to nonlinear variations in the input motion patterns. Our main modifications to the DTW algorithm are described as follows.

Interpolation of Inputs. This is a preprocessing step used to improve the quality of the input data before proceeding with the actual DTW alignment. There are several sources of spatial and temporal variations that need to be considered. First, temporal synchronization of video frames cannot be guaranteed (e.g., cameras of various frame rates). A second source of noise is related to the spatial and temporal variations that occur whenever humans perform the same motion repeatedly. The original DTW algorithms does not require same size sequences. Also, uniformity in the sampling rate of the manifolds' time-series is not required. However, results tend to improve when sequences are of similar sampling rates. This interpolation step allows the time aligning properties of DTW to more accurately compensate for the nonlinear variants by matching to anticipated intermediary missing frames.

Adapted Distance Measure. The standard DTW distance measurement is obtained by integrating the values along a path of a distance matrix relating the final manifold points to the initial manifold points. This path search is performed in a dynamic programming manner. In the standard DTW algorithm, all visited nodes contribute to the final distance reported. However, our distance measure only aggregates distances associated with transitions to the next state of the template into the final distance measure. We have modified this distance function slightly to remove additions which simply indicate the time warping is keeping the test manifold in the same state for a longer duration to remain synchronous with the template.

4 Experimental Results

In this section, we evaluate the effectiveness of our motion recognition method. Our main goal here is to show that our method is able to recognize a number of motion patterns acquired by a single camera. To accomplish this goal we provide

a comparison between our method and two recently published motion recognition methods [3,23]. For this comparative study, we use the same dataset used by the methods in [3,23]. The data set contains a collection of nine individuals performing ten distinct actions. The actions and the corresponding labels used in our experiments are the following: bending over (Bend), jumping jack (Jack), hopping across the screen (Jump), jumping up and down in place (Pjump), running (Run), stepping sideways to one direction (Side), hopping on one foot across the screen (Skip), walking (Walk), waving one arm (Wave1), and waving both arms (Wave2). We divided the data set into training subset and testing subset. These two subsets cover all individuals performing all actions. However, in the case of the Bend action, the data set did not contain enough frames to allow for the creation of two distinct action subsets. We addressed this problem by sampling every other frame in the Bend sequence to create the training and testing subsets. Additionally, in some cases, the starting point of the motion was significantly different (e.g., half-cycle sequence). This was addressed by manually stitching the two halves of incomplete motions into a single test motion. The resulting datasets were then used in the experiments described in this section.

We began by preprocessing each sequence to extract the foreground motion information. For simplicity, we used a background subtraction method to facilitate the extraction of the moving foreground silhouette. For cases where a clean background is not available, a more robust foreground segmentation method can be used [21]. The resulting silhouette images were both normalized and registered as described in Section 3.1. In our experiments, we evaluated the performance of the proposed method for images of varying sizes. The sizes used were 16×16, 24×24, and 32×32 pixels. Once the processed sequences were at hand, we compared our Isomap-based method against both the LLE and the LPP dimensionality reduction techniques. For all methods, the local manifold similarity was based on the K-nearest neighbors. Here, the K neighborhood was chosen as suggested in [23]. Accordingly, we used values of K ranging from 5 to 15 to ensure at least an overlap ranging from 10 to 15, respectively. Each motion manifold space created by these embeddings contained two dimensions and were generated from the images without taking any temporal information into consideration. Temporal information was subsequently reintroduced creating manifolds such as those illustrated in Figure 3. The manifolds in Figure 3 also illustrate the use of linear extrapolation between subsequent data points to define the motion manifold. A sampling of 64 evenly-spaced data points were taken from both the learned motion manifold and the test motion manifold for input to the DTW algorithm. A standard sequence size of 64 was chosen to represent approximately twice the size of the largest number of frames for any of the motions in the experiment's dataset. This sampling rate allows the DTW algorithm to perform alignment to interpolated frames that are missing in the learned models due to temporal misalignment in the frame sequence. The algorithm's power to extract meaningful intermediate frames is illustrated in Figure 4. With the exception of a few degraded cases each motion sequence is recognizable despite only the first and last

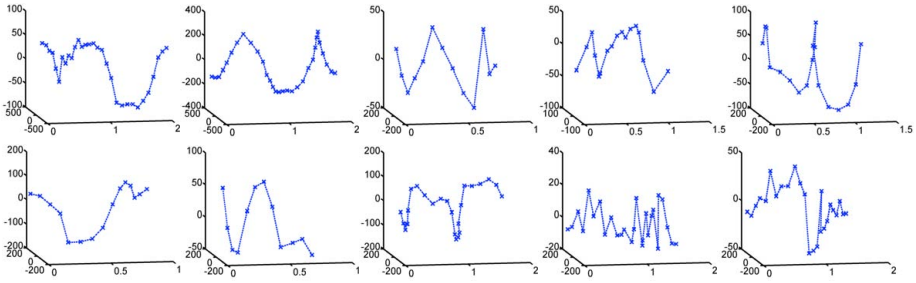


Fig. 3. Motion manifolds for Daria. Top row: Bend, Jack, Jump, Pjump, Run. Bottom row: Side, Skip, Walk, Wave1, Wave2.

silhouettes of each sequence falling exactly on a projected data point. Temporal misalignment and missing frames are common issues in many of the analyzed videos. The DTW was constrained using a Sakoe-Chiba band of 25%. Figure 5 illustrates that our proposed method using Isomap-DTW achieved almost exact recognition rates for the tested activities.

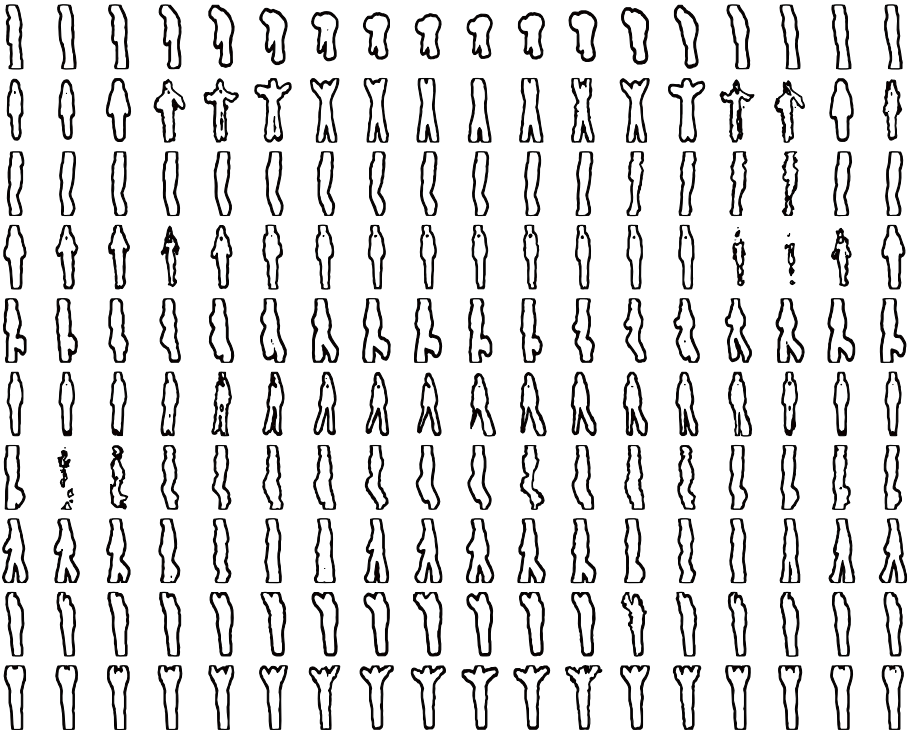


Fig. 4. Silhouette contour of the projection from manifold space to image space

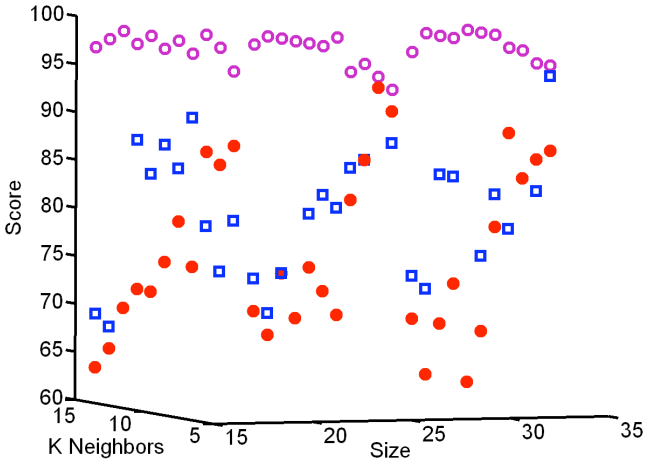


Fig. 5. Isomap(○), LLE(●) and LPP(box) Overall Activity Recognition with a Sakoe-Chiba's band of 25%. The image size is the width and height of the images after preprocessing which is also equivalent to $\sqrt{d}$.

The recognition scores in Figure 5 represent the percentage of motions correctly identified. The size of the images, the k-neighborhood sizes and the dimensionality reduction techniques used were varied for comparison. The results in Figure 5 provide evidence to support our claim that the global perservation of the Isomap data reduction technique can elucidate more meaningful manifolds for recognition via DTW. The recognition results shown in Figure 5 using Isomap with DTW are superior to those reported in [4,3]. Moreover, our results were equivalent to the ones obtained using supervised LPP-Hausdorrf-distance, unsupervised-LPP-HMM, and supervised-LPP-HMM [23]. However, our algorithm achieves this same high recognition rate with smaller image size, smaller neighborhood size, and no supervision. It is worth pointing out that, although Masoud *et al.* [17] utilized a different action database, the motions performed were comparable to the ones used in our experiments. Additionally, the best results reported in [17] were only in the lower 90% range, while our algorithm achieved 100% at several occasions. Also, although our experiments utilized periodic sequences, our method does not require motion periodicity. The specific dataset was used for comparison purposes only.

The subjects used for training are identical to the subjects used for testing. As a result, we are currently unable to infer the generalization capabilities of the proposed method with respect to recognizing unseen subjects. While we are not covering this specific issue in this paper, it is expected that models for one individual may be able to elucidate matches to similar motions performed by other individuals not captured for a particular model.

Finally, our results for all other tested Isomap configurations consistently achieved activity recognition rates above 95%. This demonstrates that, without

any experimental tuning, our technique performs very well in comparison to other established human motion recognition methods.

5 Conclusions and Future Work

In this paper, we presented a method for recognizing human action and motion patterns. Our method works by matching motion projections in Isomap non-linear manifold space using dynamic time warping (DTW). Dynamic time warping has been used in the past in many sequence alignment applications. However, the application of DTW to matching human motion manifolds has been somewhat unexplored. Moreover, we showed that Isomap manifold learning combined with DTW can be an effective way to both represent and match human motion patterns.

Our algorithm achieved accurate activity recognition results using an adapted implementation of DTW with a basic Sakoe-Chiba band optimization. Our experiments established the potential of the method for human motion recognition.

Future work includes the improvement of the computational efficiency of our recognition method by introducing indexing mechanisms such as the one suggested in [16]. Additionally, we plan to investigate the use of statistical neighborhood approach in our adapted DTW to help improve the classification results for both LLE and LPP.

References

1. Aggarwal, J.K., Cai, Q.: Human motion analysis: a review. Computer Vision and Image Understanding 73(3), 428–440 (1999)
2. Berndt, D.J., Clifford, J.: Finding patterns in time series: a dynamic programming approach. Advances in Knowledge Discovery and Data Mining, 229–248 (1996)
3. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. In: IEEE International Conference on Computer Vision, Washington, DC, USA, pp. 1395–1402 (2005)
4. Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. IEEE Trans. Pattern Anal. Mach. Intell. 23(3), 257–267 (2001)
5. Bregler, C.: Learning and recognizing human dynamics in video sequences. In: CVPR 1997. Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition, p. 568. IEEE Computer Society Press, Los Alamitos (1997)
6. Bregler, C., Malik, J.: Learning appearance based models: Mixtures of second moment experts. In: Advances in Neural Information Processing Systems, vol. 9, p. 845. The MIT Press, Cambridge (1997)
7. de Silva, V., Tenenbaum, J.B.: Global versus local methods in nonlinear dimensionality reduction. In: Becker, S., Thrun, S., Obermayer, K. (eds.) NIPS, pp. 705–712. MIT Press, Cambridge (2002)
8. Elgammal, A., Lee, C.-S.: Inferring 3D body pose from silhouettes using activity manifold learning. In: IEEE International Conference on Computer Vision and Pattern Recognition, Los Alamitos, CA, USA, vol. 02, pp. 681–688 (2004)

9. Fanti, C., Zelnik-Manor, L., Perona, P.: Hybrid models for human motion recognition. In: CVPR. IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, vol. 1, pp. 1166–1173 (2005)
10. Gavrila, D., Davis, L.: 3D model-based tracking of humans in action: A multi-view approach. In: CVPR. Conference on Computer Vision and Pattern Recognition, June 18–20, 1996, San Francisco, CA, USA (1996)
11. Gavrila, D.M.: The visual analysis of human movement: a survey. Computer Vision and Image Understanding 73(1), 82–98 (1999)
12. Green, R.D., Guan, L.: Quantifying and recognizing human movement patterns from monocular video images-part i: a new framework for modeling human motion. IEEE Trans. Circuits Syst. Video Techn. 14(2), 179–190 (2004)
13. He, X., Niyogi, P.: Locality preserving projections. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems 16, MIT Press, Cambridge (2004)
14. Heisele, C., Woehler, B.: Motion-based recognition of pedestrians. In: IEEE International Conference on Pattern Recognition, Washington, DC, USA, vol. 2, p. 1325. IEEE Computer Society Press, Los Alamitos (1998)
15. Ju, S.X., Black, M.J., Yacoob, Y.: Cardboard people: A parameterized model of articulated motion. In: International Conference on Automatic Face and Gesture Recognition, Killington, Vermont, pp. 38–44 (1996)
16. Keogh, E.J., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and Information Systems 7(3), 358–386 (2005)
17. Masoud, O., Papanikolopoulos, N.: A method for human action recognition. Image Vision Computing 21(8), 729–743 (2003)
18. Rehg, J., Kanade, T.: Digiteyes: Vision-based human hand tracking. Technical Report CMU-CS-93-220, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA (December 1993)
19. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500), 2323–2326 (2000)
20. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290, 2319–2323 (2000)
21. Tian, Y.-L., Lu, M., Hampapur, A.: Robust and efficient foreground analysis for real-time video surveillance. In: CVPR. IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, vol. 1, pp. 1182–1187 (2005)
22. Vintsyuk, T.K.: Speech discrimination by dynamic programming. Cybernetics and Systems Analysis 4(1), 52–57 (1968)
23. Wang, L., Suter, D.: Analyzing human movements from silhouettes using manifold learning. In: IEEE International Conference on Video and Signal Based Surveillance, Washington, DC, USA, p. 7 (2006)
24. Wilson, A.D., Bobick, A.F.: Parametric hidden markov models for gesture recognition. IEEE Trans. Pattern Anal. Mach. Intell. 21(9), 884–900 (1999)

Behavior Histograms for Action Recognition and Human Detection

Christian Thurau

Czech Technical University, Faculty of Electrical Engineering
Department for Cybernetics, Center for Machine Perception
121 35 Prague 2, Karlovo náměstí, Czech Republic
thurau@cmp.felk.cvut.cz

Abstract. This paper presents an approach for human detection and simultaneous behavior recognition from images and image sequences. An action representation is derived by applying a clustering algorithm to sequences of Histogram of Oriented Gradient (HOG) descriptors of human motion images. For novel image sequences, we first detect the human by matching extracted descriptors with the prototypical action primitives. Given a sequence of assigned action primitives, we can build a histogram from observed motion. Thus, behavior can be classified by means of histogram comparison, interpreting behavior recognition as a problem of statistical sequence analysis. Results on publicly available benchmark-data show a high accuracy for action recognition.

1 Introduction

Human behavior recognition is important for a number of applications, e.g. video annotation, surveillance, or personal assistance systems for elder or disabled persons. The main task in behavior analysis from image sequences is to correctly assign an observed motion sequence to a set of prototypical known behavior classes. If we assume that behavior consists of sequences of atomic actions, we have to address three disjoint problems; (a) how to detect humans in images, (b) how to adequately represent atomic actions, and (c) how to classify a sequence of atomic actions into behavioral classes.

The main contribution of this paper is an approach for human detection and behavior recognition from image sequences by means of statistical sequence analysis. In previous work [1], we already reported on the idea of behavior recognition by means of histogram comparison of bags of action primitives. Here, we further extend and improve these ideas. By directly utilizing the proposed action representation for detecting humans, we no longer require background subtraction, tracking, or global movement compensation. Thereby, we can not only classify image sequences, but also single images. Recognition rates on publicly available action-data [2] exceed up-to-date approaches, see e.g. [3]. However, we still achieve lower recognition rates than approaches that use background subtraction, and global movement compensation, as for instance [2,1].

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 299–312, 2007.

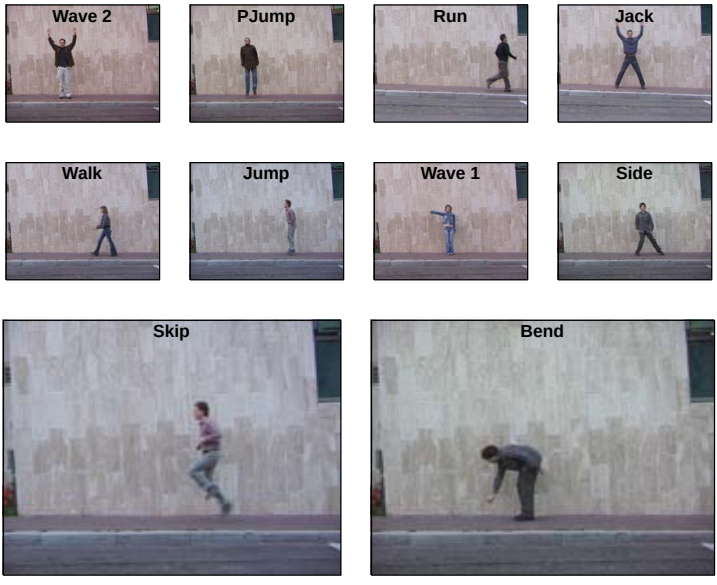


Fig. 1. For the experiments we considered 10 different behaviors performed by 9 subjects. The task is to detect the human within the image and to assign the sequence to one of the known behavioral categories, e.g. 'bend', 'jump', or 'run'.

For detecting humans in images it could been shown that *Histograms of Oriented Gradients (HOG)* [4] are very distinctive and robust. In the presented work, we detect humans by comparing extracted HOG descriptors to template HOG descriptors of human poses. The templates are automatically clustered from a set of training sequences. In the following, we will refer to these templates as *action primitives* [5] or *basic action units* [6]. We select the best matching action primitives for a novel image, and thereby create a sequence of action primitive indices. Consequently, we can treat behavior recognition as a problem of sequence comparison where we compare novel sequences to a set of labeled training sequences. Since behavior is a temporal phenomenon, we express the sequential observation of basic action units using *n-grams*, a popular representation used in statistical text analysis or bioinformatics. Finally, we classify behaviors by means of histogram comparison.

In Section 3.1 we explain how to derive an action representation using the concept of movement primitives, and we introduce a n-gram based action representation. Section 3.2 introduces behavior histograms and a classifier for action primitive sequences. We validate the approach in a human tracking scenario in Section 4. Finally, we close this contribution with a conclusion in Section 5 and an outlook on future work in Section 6. First, however, we will briefly summarize related approaches.

2 Related Work

The most recent approaches related to ours are [3,7,8]. In [3], a hierarchical bags of features approach classifies videos or single images into action categories. In [7] human action-categories are learned by representing a video sequence as a collection of spatial-temporal words by extracting space-time interest points. In [8] actions of Hockey players are represented and tracked using HOG-descriptors and Hidden Markov Models. In the proposed approach, we also represent basic actions as HOG-descriptor based templates. In contrast to [4], we consider histograms of basic action templates for behavior recognition. Furthermore, our approach is inspired by the recent progress on bags of visual words.

The action representation proposed in this paper is similar to the biological concept of *movement primitives*. Results in behavioral biology indicate that instead of directly controlling motor activation commands, movements are the outcome of a combination of simpler movement primitives [9,10]. Furthermore, cognitive psychology explained the representation of motor skills in human long-term memory by hierarchies (but also sequences) of basic-action concepts [6]. It is interesting to note that different disciplines decided for different terminologies for describing very similar concepts of motor control. In this paper, we use the terminology proposed in [6] and [11,5]. In this context, behavior is the outcome of appropriate sequencing of action primitives.

In robotics, a movement primitive is often referred to as a computational element in a sensorimotormap that transforms desired limb-trajectories into actual motor commands [12]. Related to our approach is the work of Fod et al. [13] or Lee et al. [14], where the concept of movement primitives is applied in a computational approach by linear superposition of clustered primitive motor commands that finally resulted in complex motion in simulated agents. For learning and synthesizing human behavior in simulated game worlds, we previously used a probabilistic sequencing of action primitives [5]. In this paper, we derive action primitives in a similar way as proposed in [13] or [5].

The idea of representing actions by a set of basic building blocks is not new. Especially for recognition tasks, it is getting more and more popular. For a recent overview we recommend [11,15]. Numerous approaches on recognizing behavior based on keyframes, movement primitives, or action primitives exist. For example, Moeslund et al. [16] recognized actions by comparing strings of manually found motion primitives, where the motion primitives are extracted from four features in a motion image. In [17], silhouettes images are extracted using optical flow data, afterwards they are grouped in pairs and used to construct a grammar for action recognition. [18] further expands ideas on designing a human action recognition grammar. [2] expresses human action as three-dimensional shapes that result from spatial silhouette movements with respect to time, and use these extracted features for classifying activity. [19] compares histograms of n-grams sequences of complex actions, to recognize higher-level behaviors, where n-grams are build using a set of basic hand crafted events. Goldenberg et al. [20] use PCA to extract eigenshapes from silhouette images for behavior classification.

3 Behavior Recognition

In the following, we outline our approach for behavior recognition, see also Figure 2 for a schematic outline. First, we will explain how to extract action primitives from behavioral observations. Then, we will transfer sequences of action primitives to a more meaningful descriptor by incorporating the temporal context using n-grams. Finally, we will explain how histograms of behaviors can be constructed, and how they are used for classification of novel image sequences.

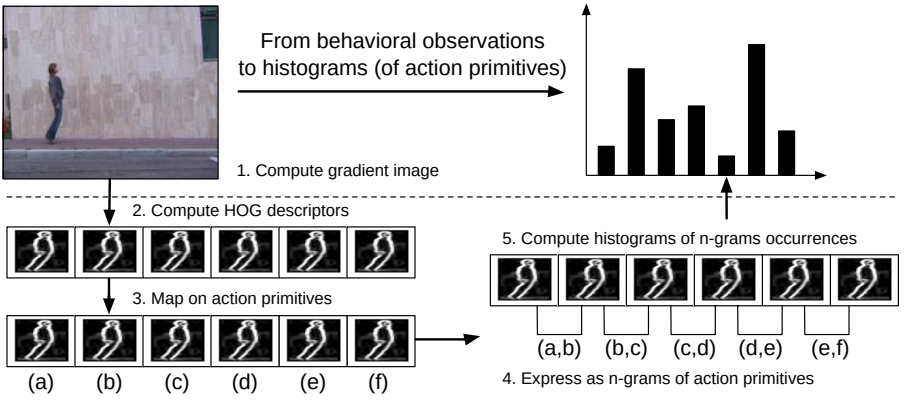


Fig. 2. Construction of behavior histograms: First, the gradient image for a detector window is computed. Then, we compute HOG-descriptors and combine them in a feature vector. We map the feature vector on the best matching prototypical action-primitive. Consecutive action-primitives are afterwards combined in n-gram structures. Finally, the resulting histogram of n-gram occurrences is computed for the whole sequence.

3.1 Basic Action Representations

For describing actions we use templates consisting of HOG-descriptors [4]. It could be shown that these outperform other features for human detection in static images. Basically, the HOG-descriptor is a locally normalized orientation histogram, similar to the popular SIFT features [21]. In this work, we use it to (a) find humans in image sequences, and (b) as a descriptor for action primitives. Figure 3 briefly summarizes the descriptor, for details we recommend [4]. The detector window used in our experiments consists of 40×80 pixels.

The idea of action primitives assumes that complex motion can be constructed using a set of basic action units. In order to derive a meaningful set of action primitives, we apply k-means clustering to the HOG-descriptors extracted from the training sequences. For clustering, we use the cosine distance measure. The training data consists of centered and aligned images showing various human activities. Training images are extracted at a size of 40×80 pixels. Thus, one descriptor corresponds to one activity, see Figure 4 for example sequences.

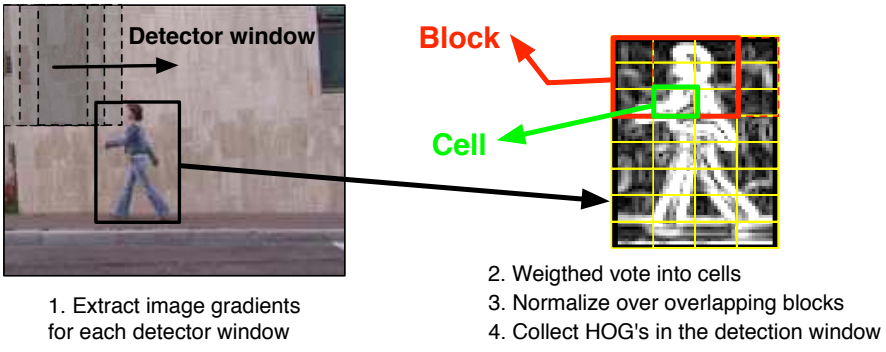


Fig. 3. The detector window (40×80 pixels) is divided into cells of size 10×10. For the detector window the image gradient is computed using a simple filter mask ($[-1, 0, 1]$). Every pixel calculates a vote for an edge orientation histogram for its cell, weighted by the magnitude (as suggested in [4], we used 9 orientation bins). Groups of 3×3 cells form a so called block. Blocks overlap with each other. For each block, the orientation histograms of the included cells are concatenated and normalized using L2-norm. The final descriptor is the concatenated vector of all normalized blocks for the detector window (note that the suggested parameters vary from [4]).

Clustering results in a set of k action primitives $[\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k], \boldsymbol{a}_i \in \mathbb{R}^m$, where each action primitive $\boldsymbol{a}_i$ corresponds to the concatenated HOG-descriptors within the detector window. By assigning each HOG-descriptor $\boldsymbol{h}_i$ of a Sequence $H = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_l]$ of length l to its best matching prototypical action unit $\boldsymbol{a}$, we can express an image sequence of human motions H as a sequence of action primitives A_H. For measuring descriptor similarities, we use the Kullback-Leibler divergence, thus

$$A_H = [\boldsymbol{a}_{\text{argmin}_i D_{KL}(\boldsymbol{a}_i || \boldsymbol{h}_1)}, \ldots, \boldsymbol{a}_{\text{argmin}_i D_{KL}(\boldsymbol{a}_i || \boldsymbol{h}_l)}] \tag{1}$$

where the Kullback-Leibler divergence $D_{KL}(\boldsymbol{a} \;||\; \boldsymbol{h})$ for two normalized histograms $\boldsymbol{a}$ and $\boldsymbol{h}$ is defined as

$$D_{KL}(\boldsymbol{a} \;||\; \boldsymbol{h}) = \sum_i a_i \log \frac{a_i}{h_i} \tag{2}$$

From sequences of action primitives A_H, we can build behavior histograms by counting occurrences of individual action primitives.

Since the action primitives effectively represent specific poses of humans, we can also use them for finding humans in images. In [4] HOG-descriptors of humans and of the background were used to train a SVM for human/non-human classification, providing very accurate human detection. In this work, we decided for a similar approach. Therefore, besides the prototypical action descriptors, we extract a set of k (we use the same number of prototypes as for the pose templates) prototypical descriptors for the image background $[\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k], \boldsymbol{b}_i \in \mathbb{R}^m$,

Fig. 4. Exemplary training sample sequences for extracting action primitives and building behavior histograms. The 40×80 pixel detector window is centered around the human. The HOG-descriptors of all training images are used for clustering action primitives.

by means of k-means clustering of background HOG-descriptors. Based on the Kullback-Leibler divergence and a simple nearest neighbor criterion, each pixel $P_{x,y}$ of a novel image can be recognized as a specific action primitive or as background. For this, we first compute the HOG descriptors $\boldsymbol{h}_{x,y}$ for the detector window (see also Figure 3) centered around each pixel $P_{x,y}$. Then, each $P_{x,y}$ is assigned to one of the k action primitives or set to 0 for background, thus

$$P_{x,y} = \begin{cases} j & D_{KL}(\boldsymbol{a}_j \mid\mid \boldsymbol{h}_{x,y}) \leq D_{KL}(\boldsymbol{b}_l \mid\mid \boldsymbol{h}_{x,y}) \\ 0 & D_{KL}(\boldsymbol{a}_j \mid\mid \boldsymbol{h}_{x,y}) > D_{KL}(\boldsymbol{b}_l \mid\mid \boldsymbol{h}_{x,y}) \end{cases} \tag{3}$$

where

$$j = \text{argmin}_i \, D_{KL}(\boldsymbol{a}_i \mid\mid \boldsymbol{h}_{x,y})$$
$$l = \text{argmin}_i \, D_{KL}(\boldsymbol{b}_i \mid\mid \boldsymbol{h}_{x,y})$$

We found that usually a few pixels within the center of a human are recognized as an action primitive. Consequently, we assume the center of mass of successfully recognized pixels to be the location of a human. We assign every image in which at least one action is successfully recognized to a specific action primitive index. Obviously, this is only works for a single human within the images. Although this approach is rather simple, it showed to be sufficient for the later presented experiments.

While prototypical poses (action primitives) can be detected within a single image, for successful behavior recognition the temporal context of action unit sequences is crucial. Therefore, we represent motion sequences by means of n-grams of action primitives. n-grams are often used for sequence analysis in text mining, or bioinformatics. Basically, they provide a sub-sequencing of length n for a given sequence. For example, a sequence $\boldsymbol{A}^{\text{uni}}$ of action primitive unigrams (n is set to 1)

$$\boldsymbol{A}^{\text{uni}} = [\boldsymbol{a}_3, \boldsymbol{a}_3, \boldsymbol{a}_8, \boldsymbol{a}_6, \boldsymbol{a}_5, \boldsymbol{a}_5] \tag{4}$$

where $\boldsymbol{a}_3, \boldsymbol{a}_8, \boldsymbol{a}_6, \boldsymbol{a}_5$ correspond to specific action primitives, could be simply expressed as a sequence of overlapping bigrams $\boldsymbol{A}^{\text{bi}}$ (n is set to 2)

$$\boldsymbol{A}^{\text{bi}} = [(\boldsymbol{a}_3\boldsymbol{a}_3), (\boldsymbol{a}_3\boldsymbol{a}_8), (\boldsymbol{a}_8\boldsymbol{a}_6), (\boldsymbol{a}_6\boldsymbol{a}_5), (\boldsymbol{a}_5\boldsymbol{a}_5)] \tag{5}$$

or trigrams $\boldsymbol{A}^{\text{tri}}$ (n is set to 3)

$$\boldsymbol{A}^{\text{tri}} = [(\boldsymbol{a}_3\boldsymbol{a}_3\boldsymbol{a}_8), (\boldsymbol{a}_3\boldsymbol{a}_8\boldsymbol{a}_6), (\boldsymbol{a}_8\boldsymbol{a}_6\boldsymbol{a}_5), (\boldsymbol{a}_6\boldsymbol{a}_5\boldsymbol{a}_5)] \tag{6}$$

Figure 2 shows an example on how a sequence of frames can be represented using a sequence of n-grams of action primitives. With increasing n, the number p of possible instances of n-grams increases exponentially, since $p = k^n$ where k denotes the number of action primitives. Since we are dealing with human motion which is bound to physical limitations, we usually observe only a limited number of possible action primitive subsequences (human movement is rather smooth). Thus, the actual number nk of n-gram instances, which were observed during the training phase, tends to be much lower.

3.2 Behavior Histograms for Recognizing Action Classes

In practice, for comparison of different n-gram streams, often the histograms i.e. the occurrences of specific n-gram instances are used. Thus, analyzing sequences of n-gram action primitives is very similar to the idea of bags of visual words. Given a sequence $\boldsymbol{A}$ of n-grams of action primitives, a histogram $\phi(\boldsymbol{A})$ is defined as a mapping from a finite alphabet Σ (corresponding to all unique instances of n-gram action primitives $[a_1, \ldots, a_{nk}]$), so that

$$\phi(\boldsymbol{A})\colon \Sigma^* \rightarrow \mathbb{N}^+ \cup 0, \phi(\boldsymbol{A})_i := \mathrm{occ}(a_i, \boldsymbol{A}) \tag{7}$$

where $i = 1, \ldots, nk$, and nk denotes the total number of action primitives n-grams extracted during the training phase, and $\mathrm{occ}(a_i, \boldsymbol{A})$ denotes the number of occurrences of a_i in $\boldsymbol{A}$. For every training sequence $[\boldsymbol{A}_1, \ldots, \boldsymbol{A}_T]$, where T denotes the total number of training sequences, a separate histogram $\phi(\boldsymbol{A}_i), i = 1, \ldots, T$ is computed. Since each training sequence shows one specific behavior executed by one subject, the histograms represent a behavior and will be further referred to as a *behavior histograms*.

We classify novel image sequences using a nearest neighbor search over the training behavior histograms, where the choice of similarity measure is crucial. For histograms, commonly used measures are KL, L2, or L1. For comparison of single-histogram class models of visual words it could been shown that the Kullback-Leibler (KL) divergence outperforms alternative similarity measures, as for instance L2 or χ^2 [22]. Consequently, since we are facing a similar problem, we decided for KL divergence for measuring histogram similarity.

For a novel sequence $\boldsymbol{A}_{\mathrm{new}}$, the task now is to select a suitable class label based on histogram comparison to stored behavior histograms. For the experiments, we classify into a behavioral class using a 1-NN criterion, i.e. by finding the class histogram $\phi(\boldsymbol{A}_j)$, with

$$j = \operatorname*{argmin}_i D_{KL}(\phi(\boldsymbol{A}_{\mathrm{new}}) \mid\mid \phi(\boldsymbol{A}_i)) \tag{8}$$

where $i = 1, \ldots, T$, and $\phi(\boldsymbol{A}_i)$ denotes a normalized training histogram. We decided for a 1-NN since we can usually only access a very limited number of class histograms. Moreover, we observed a high inner class variability, e.g. for some people walking might look more similar to the average running behavior. Therefore, deciding on the single best matching exemplar histogram gave the best

results. Future work might as well consider histogram integration, for example by means of single-histogram class models [22].

For the histogram similarity measure, the actual length of the image sequence is not important. This means that we can classify single images as well as complete sequences using the same approach. Obviously, for single images the KL divergence comes down to a simple and intuitive formula, where we select by finding the behavior histogram $\phi(\boldsymbol{A}_j)$ for a single recognized action primitive (or n-gram instance) a_l so that

$$j = \underset{i}{\operatorname{argmin}} D_{KL}(\phi(\boldsymbol{a}_l) \mid\mid \phi(\boldsymbol{A}_i)) = \log \frac{1}{a_{i,l}} \tag{9}$$

where $a_{i,l}$ denotes the corresponding histogram entry l for the behavior histogram i. It is important to note that we can only achieve a true single image classification by using unigrams (n-grams of length 1), otherwise the temporal context will be included. Interestingly, including the temporal context for very short sequences (e.g. two frames in the case of bi-grams) does not lead to a better classification rate for these frames as compared to single frame classification. One of the reasons might be overfitting due to the larger amount of symbols (e.g. 50 symbols for unigram action primitives can result in up to 2500 bigram symbols). However, for longer sequences, the usages of n-grams results in better classification rates. The next section will give a detailed report on the experimental results.

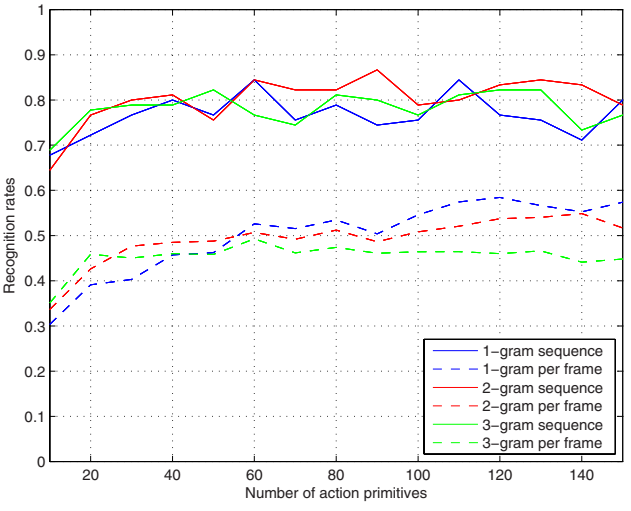
4 Experimental Results

For validation of our approach we use publicly available benchmark-data [2]. The action-data-set consists of 10 different behaviors performed by 9 subjects. Figure 1 illustrates the different behaviors.

We performed a leave one out cross validation where we excluded the behavioral observations from one subject in each run. The remaining 80 sequences (8 subjects and 10 behaviors) were used as training data, for extracting action primitives, prototypical background descriptors, and constructing behavior histograms. We assigned each test sequence to the training behavior histogram with the highest similarity. Moreover, we did a frame-wise classification of every image in a sequence using the same classification procedure. For evaluation, we computed the average classification rates among all subjects.

Using integral histograms [23] for computing the gradient image histograms, we can process up to 3 frames per second in a first basic Matlab implementation. However, we are confident that more sophisticated and faster approaches for computing the HOG descriptors, e.g. [24], will result in better performance. Right now the computation of the HOG-descriptor is clearly the computationally most expensive part in our approach.

In order to get optimal results we varied the cell, and block size for the HOG-descriptor, and the size of the detector window. The last sections already contained details on parameter selection. However, for the approach it seems more



	1-gram	2-gram	3-gram
Max. rate sequence	84.44	**86.66**	82.22
Max. rate per frame	**57.45**	48.63	45.82
# action primitives	110	90	50

Fig. 5. Average classification rates for varying number of action primitives and various length n-grams. The table shows the overall best average classification rates for uni-, bi-, and trigrams. While the rates are comparable, the usage of bi-, or trigrams usually requires less action-primitives.

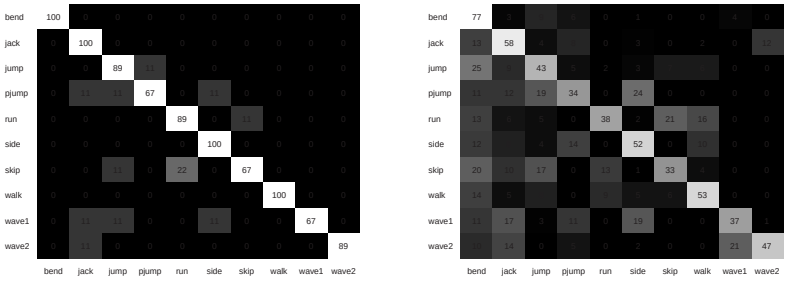


Fig. 6. Confusion matrix for $k = 90$ action primitives using bigrams (columns correspond to predictions while rows correspond to the actual class). The left Figure shows the recognition rates for complete sequences. Here, we achieved an average recognition rate of 86.66% for classifying based on complete sequences. The right figure show the per frame recognition rates, 48.63% of all individual frames were correctly classified.

important to evaluate the influence of the length n of action primitive n-grams, and the number of used action primitives. Figure 5 gives results for varying number of action primitives.

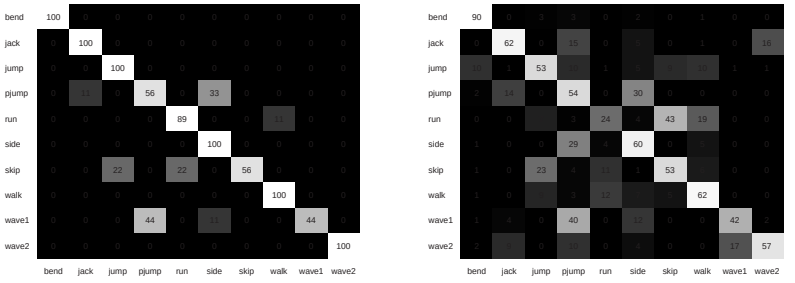


Fig. 7. Confusion matrix for $k = 110$ action primitives using unigrams. The left Figure shows the recognition rates for complete sequences. Here, we achieved an average recognition rate of 84.44% for classifying whole sequences. The right figure show the per frame recognition rates, 57.45% of all individual frames were correctly recognized.

Table 1. Average classification rates for leave-one-out cross validation of 10 different behaviors performed by 9 subjects (excluding all behaviors of one subject for each run). Each column contains the best results reported on in the specific paper. Note that the results for [2,1] are for subsequences and not the whole sequence, however, according to our experience, we can expect a performance increase for classifying the whole sequence. In [2,3], the 'skip' behavior was excluded.

	Thurau et al. [1]	Blank et al. [2]	Niebles et al. [3]	Our approach
Classification rate	99.10%	99.61%	72.8	86.66
Per frame classification rate	–	–	55.0	57.45
No background subtraction	–	–	x	x
Centered/aligned training sequences	x	x	-	x
No movement compensation	–	–	x	x
Human detection	–	–	–	x

For a complete sequence, the best average classification rate of 86.66% was achieved using bigrams and 90 action primitives, Figure 6 shows the confusion matrix. For frame-wise classification we achieved recognition rates of 57.45% using unigrams and 110 action-primitives, the confusion matrix can be seen in Figure 7. Interestingly, unigrams gave the best results for frame-wise classification. One reason might be, that the number of basic actions increases with an increasing n, which could lead to overfitting and worse classification results. Consequently, dependent on the application, different parameters should be selected.

Compared to the recent approach of Niebles et al. [3] we achieve almost 15% higher recognition rates for classifying complete sequences. However, compared to [2,1] the recognition rates are lower. In contrast to [2,1] (and similar to [3]) , we do not require background subtraction, external tracking solutions, or global movement compensation. However, we require a set of centred and aligned

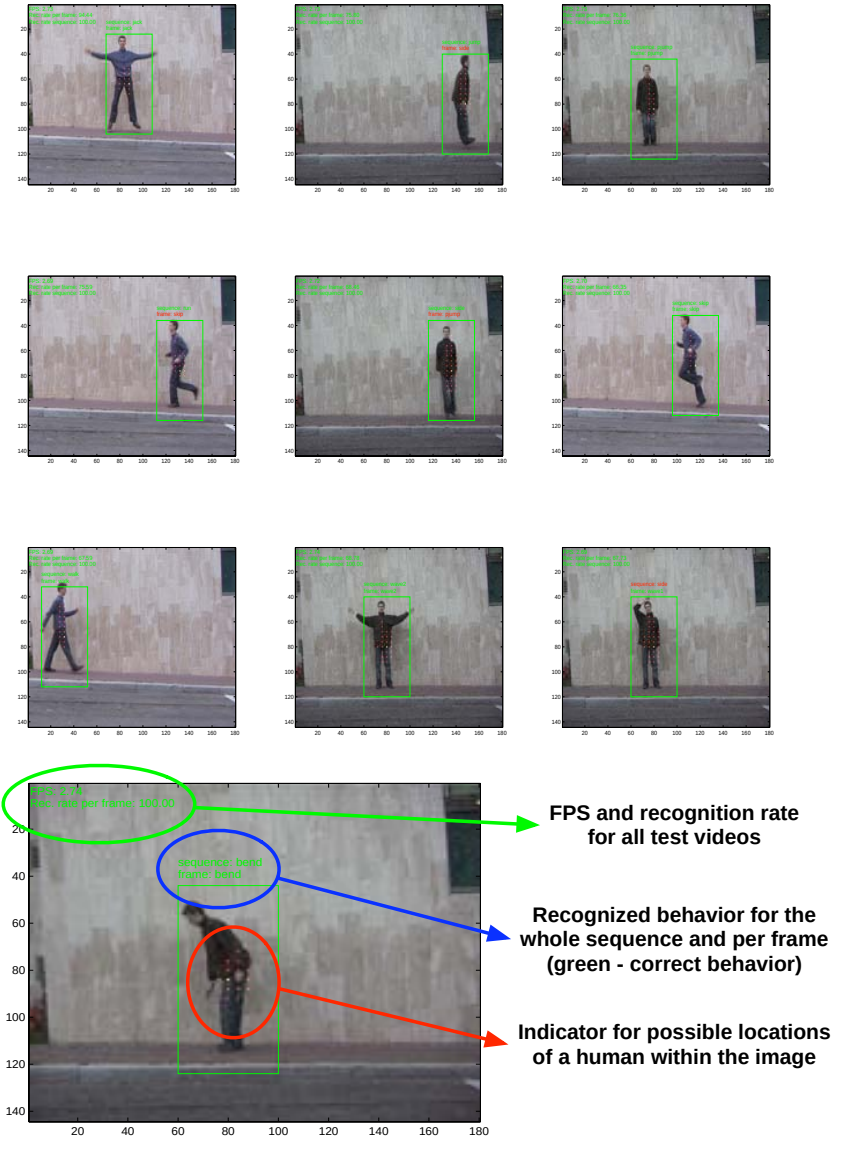


Fig. 8. Sample video sequences as the outcome of our approach. The human is automatically detected, and his current action is recognized for a whole sequence and for single frames.

human motion sequences for training, which makes the training phase more demanding as [3]. Table 1 briefly compares the four papers. Figure 8 shows the augmented image sequences as the outcome of our behavior recognition system.

It is interesting to note that we only used static image features, we did not incorporate optical flow or similar dynamic image features. In [3] it was argued that dynamic features outperform static features for recognizing action categories. Effectively, we could show that sufficiently descriptive and discriminative static image features and a statistical sequence analysis can already result in very accurate recognition of action classes. Nevertheless, we also believe that dynamic image features could improve classification results.

5 Conclusion

In this paper, we proposed a novel approach for behavior recognition by means of behavior histograms of action primitives. Action primitives are clustered, prototypical human pose representations. They can be detected for every frame in an image sequence where a human is visible. For representing action primitives, we used Histogram of Oriented Gradients. Since the HOG-descriptor allows for robust detection of humans in images, we used the set of action primitives to find humans via a nearest-neighbor search. To obey the temporal context of behaviors, subsequent action primitives are combined in n-gram structures. Finally, we expressed behaviors as histograms of the n-gram instances that were found in the image sequence.

In a series of experiments, we could show the applicability for behavior recognition. We classified behaviors based on the Kullback-Leibler divergence of behavior histograms. Recognition rates on a publicly available benchmark-data showed a high accuracy for classifying motion sequences. We were also able, with a lower precision, to recognize action categories from single images.

6 Future Work

To concentrate on the principal idea of expressing behavior as histograms of action primitives, we tried to keep the approach as simple as possible. Although the current recognition rates are already accurate, one could think of many possible extensions. To speed up processing time, we would suggest the cascade of HOGs approach [24]. In addition, in order to deal with inner-class variability, and to speed up processing time, we would suggest the usage of single-histogram class models [22]. For more accurate human detection, we could use more sophisticated classifiers, e.g. SVMs as reported on in [4]. Moreover, we could incorporate dynamic features aside from the HOG descriptor, e.g. oriented histograms of flow [25].

So far, we were not dealing with humans at different scales or different views. Different scales could be handled by using variable size detector windows as in the original HOG approach. For different views, we would suggest the usage of more versatile training data, i.e. already incorporate different views for building behavior histograms.

Acknowledgments

We would like to thank Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri for making their behavior dataset publicly available [2]. Christian Thurau is supported by a grant from the European Community under the EST Marie-Curie Project VISIONTRAIN MRTN-CT-2004-005439.

References

1. Thurau, C., Hlaváč, V.: n-grams of Action Primitives for Recognizing Human Behavior. In: Kropatsch, W., Kampel, M., Hanbury, A. (eds.) CAIP 2007. LNCS, vol. 4673, pp. 93–100. Springer, Heidelberg (2007)
2. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as Space-Time Shapes. In: Tenth IEEE International Conference on Computer Vision, IEEE Computer Society Press, Los Alamitos (2005)
3. Niebles, J.C., Fei-Fei, L.: A Hierarchical Model of Shape and Appearance for Human Action Classification. In: CVPR 2007. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2007)
4. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: CVPR 2005. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2005)
5. Thurau, C., Bauckhage, C., Sagerer, G.: Synthesizing Movements for Computer Game Characters. In: Rasmussen, C.E., Bülthoff, H.H., Schölkopf, B., Giese, M.A. (eds.) DAGM 2004. LNCS, vol. 3175, pp. 179–186. Springer, Heidelberg (2004)
6. Schack, T., Mechsner, F.: Representation of motor skills in human long-term memory. Neuroscience Letters 391, 77–81 (2006)
7. Niebles, J.C., Wang, H., Fei-Fei, L.: Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words. In: BMVC 2006. British Machine Vision Conference (2006)
8. Lu, W.L., Little, J.J.: Simultaneous tracking and action recognition using the pca-hog descriptor. In: CRV 2006. The Third Canadian Conference on Computer and Robot Vision (2006)
9. Ghahramani, Z.: Building blocks of movement. Nature 407, 682–683 (2000)
10. Wolpert, D.M., Ghahramani, Z., Flanagan, J.R.: Perspectives and problems in motor learning. TRENDS in Cognitive Sciences 5(11), 487–494 (2001)
11. Krüger, V., Kragic, D., Ude, A., Geib, C.: Meaning of action (2007)
12. Thoroughman, K., Shadmehr, R.: Learning of action through adaptive combination of motor primitives. Nature 407, 742–747 (2000)
13. Fod, A., Matarić, M., Jenkins, O.: Automated Derivation of Primitives for Movement Classification. Autonomous Robots 12(1), 39–54 (2002)
14. Lee, C.S., Elgammal, A.: Human motion synthesis by motion manifold learning and motion primitive segmentation. In: Perales, F.J., Fisher, R.B. (eds.) AMDO 2006. LNCS, vol. 4069, pp. 464–473. Springer, Heidelberg (2006)
15. Moeslund, T., Reng, L., Granum, E.: Finding Motion Primitives in Human Body Gestures. In: Gibet, S., Courty, N., Kamp, J.F. (eds.) GW 2005. LNCS (LNAI), vol. 3881, pp. 133–144. Springer, Heidelberg (2006)
16. Moeslund, T., Fihl, P., Holte, M.: Action Recognition using Motion Primitives. In: The 15th Danish conference on pattern recognition and image analysis (2006)

17. Ogale, A.S., Karapurkar, A., Aloimonos, Y.: View-invariant modeling and recognition of human actions using grammars. In: ICCV Workshop on Dynamical Vision (2005)
18. Guerra-Filho, G., Aloimonos, Y.: A Sensory-Motor Language for Human Activity Understanding. In: HUMANOIDS 2006. 6th IEEE-RAS International Conference on Humanoid Robots, pp. 69–75. IEEE Computer Society Press, Los Alamitos (2006)
19. Hamid, R., Johnson, A., Batta, S., Bobick, A., Isbell, C., Coleman, G.: Detection and Explanation of Anomalous Activities: Representing Activities as Bags of Event n-Grams. In: CVPR 2005. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society Press, Los Alamitos (2005)
20. Goldenberg, R., Kimmel, R., Rivlin, E., Rudzsky, M.: Behavior classification by eigendecomposition of periodic motions. Pattern Recognition 38, 1033–1043 (2005)
21. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 20, 91–110 (2003)
22. Schroff, F., Criminisi, A., Zisserman, A.: Single-Histogram Class Models for Image Segmentation. In: Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (2006)
23. Porikli, F.: Integral histogram: A fast way to extract higtograms in cartesian spaces. In: CVPR. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 829–836. IEEE Computer Society Press, Los Alamitos (2005)
24. Zhu, Q., Avidan, S., Yeh, M., Cheng, K.: Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In: IEEE Computer Society Conference on Computer vision and Pattern Recognition, pp. 1491–1498. IEEE Computer Society Press, Los Alamitos (2006)
25. Dalal, N., Triggs, B., Schmid, C.: Human detection using oriented histograms of flow and appearance. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 428–441. Springer, Heidelberg (2006)

Learning Actions Using Robust String Kernels

Changjiang Yang, Yanlin Guo, Harpreet Sawhney, and Rakesh Kumar

Sarnoff Corporation
Princeton, NJ 08543
{cyang,yguo,hsawhney,rkumar}@sarnoff.com

Abstract. This paper presents an action analysis method based on robust string matching using dynamic programming. Similar to matching text sequences, atomic actions based on semantic and structural features are first detected and coded as spatio-temporal characters or symbols. These symbols are subsequently concatenated to form a unique set of strings for each action. A similarity metric using longest common subsequence algorithm is employed to robustly match action strings with variable length. A dynamic programming method with polynomial computational complexity and linear space complexity is implemented. An effective learning scheme based on similarity metric embedding is developed to deal with matching strings of variable length. Our proposed method works with limited amount of training data and exhibits desirable generalization property. Moreover, it can be naturally extended to detect compound behaviors and events. Experimental evaluation on our own and a commonly used data set demonstrates that our method allows for large pose and appearance changes, is robust to background clutter, and can accommodate spatio-temporal behavior variations amongst different subjects while achieving high discriminability between different behaviors.

1 Introduction

Human action analysis from visual data is important in visual surveillance and video retrieval applications. Many applications can benefit from the detection and recognition of target actions using exemplar videos or specific semantic descriptions. Due to various factors such as large individual variations in clothes, accessories, appearance, scale, and viewpoint, human body articulation and self-occlusion, and dynamic background clutter, action analysis remains a very challenging problem. Majority of approaches require extensive training [1,2,3] and poorly handle complicated actions and activities.

In this paper, we present an action analysis method that requires limited training and exhibits desirable generalization property. It can also be naturally extended to detect compound human actions and events. The proposed method bears an analogy to text sequence analysis, which has been widely studied an has proved to be a powerful methodology. In our method, atomic actions based on semantic (face, hands, etc.) and structural features (spatio-temporal stable features) are first detected and coded as spatio-temporal characters or symbols.

A. Elgammal et al. (Eds.): Human Motion 2007, LNCS 4814, pp. 313–327, 2007.

These symbols are subsequently concatenated to form a unique set of strings for each action. A similarity metric using a *longest common subsequence* (LCS) algorithm with dynamic programming is employed to robustly match action strings. An effective kernel method based on similarity metric embedding is developed to match strings of variable length.

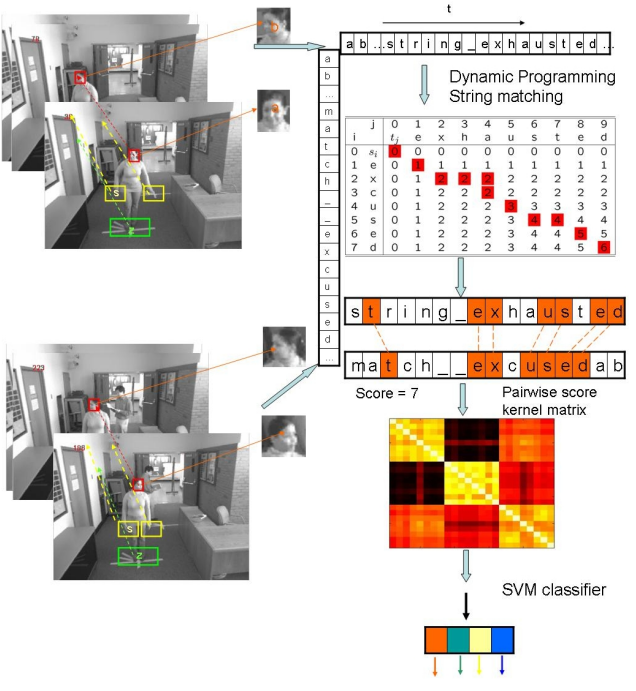


Fig. 1. System diagram

Key characteristics of our approach that alleviate the necessity of extensive training are:

1. Compact representation of human actions using a combination of spatio-temporal informative semantic and structural features (symbols or characters, we use both terms interchangeably in the rest of paper.) and their concatenations (strings). Usually our vocabulary set consists of characters in the order of tens. A character symbolizes feature type, position, and movement. For example, left hand moves toward up-left direction around head position is embedded as one character, as shown in Fig 1. The configuration of characters in one frame characterizes human pose and articulation at one time instance. The temporal order of actions is maintained with string representation and corresponding matching technique.
2. Decomposition of learning from 3D spatio-temporal volumetric data into the semi-2D feature level and semi-1D string matching level. Semi-2D features

(characters) are learned and detected with one (semantic features) or two frames (structural features). Subsequent semi-1D string matching is performed both along temporal axis (temporal order) and in spatial domain (character configuration). This decomposition enables learning with reduced dimensionality in lower dimensions and achieves better generalization property.

3. In the matching level, string kernel matching based on similarity metric embedding also achieves effective dimension reduction in learning.

Since our string kernel matching is a generic matching technique, the proposed method can be naturally extended to handle more complicated and compound actions. Only the vocabulary of atomic actions needs to be enhanced or modified, but action matching and recognition scheme remains intact. Specifically, atomic action units can occupy the full spectrum of spatio-temporal features, from low-level interest points, body parts, to people and vehicle, or even a simple atomic action itself.

The robustness of our approach with respect to pose and appearance change, occlusion, and background clutter is achieved through the combination of semantic and structural features and the coarse quantization of feature positions and orientations in optical flow field. Pixel-wise optical flow has been demonstrated as a simple yet powerful feature for capturing motion independent of appearance. The inclusion of reliable semantic features such as face and hands with their position attribute makes the scheme robust to pose changes. Missing or erroneous atomic actions (characters) due to occlusion and background clutter is compensated for in the robust string matching with LCS algorithm, where a longest sequence is found among all subsequences of a set of sequences. Also, since no rigid constraint is imposed amongst features, and only feature spatial configuration and temporal order is used in the matching, our scheme allows for large individual variations among different subjects.

Finally, the overall action detection and recognition system can be implemented in real time due to the compact representation and fast detection of semantic and structural features, as well as the fast implementation of LCS algorithm using dynamic programming.

We review the related work in Section 2, and explain technical details in Section 3. Experimental results are given in Section 4, and we conclude in Section 5.

2 Related Work

There are two major approaches in suspicious activity detection: supervised v.s. unsupervised methods. Unsupervised methods are suitable for situations where it is not possible or realistic to explicitly define and build all activity models [4,5]. When target activity models are well defined and constrained, the model-based supervised approach can be effective. Our method falls into the model-based category.

In the model-based approach, typically features are extracted and tracked from the video. The extracted features are used to develop models for the target

activities, either by exemplar videos or human specified rules and context. A common choice is to use Hidden Markov Models [6] or other graphical models which quantize image features into a set of discrete states and model how states change in time. There are many known disadvantages and limitations in performing visual tracking such as manual initialization, poor long-term stability, etc.

Davis and Bobick [7] propose a probabilistic syntactic approach for extended action detection and interactions between multiple agents. A two level detection strategy is adopted where the lower level detections are performed using standard independent probabilistic event detectors to propose candidate detections of low-level features, and a longer range stochastic context-free grammar parsing mechanism is used in the higher lever to disambiguate uncertain low-level detections, and allow the inclusion of a priori knowledge. Our approach is largely inspired by this approach, but does not require the specification of rigid activity grammars.

Our conversion of atomic activity element (features) into character and matching with strings also inspired by [8] where matches on descriptors are computed using vector quantization with visual analogy of words, and inverted file systems and document rankings are used for object and scene retrieval.

An important consideration in action analysis is the selection of appropriate feature for motion pattern description and matching. Appearance features such as image intensity, image spatial gradients or temporal gradients [9,10] are useful features, but the appearance features are not always preserved across different sequences. Pixel-wise optical flow has been demonstrated as a simple yet powerful feature for capturing motion independent of appearance. Special treatment [1] needs to be applied to deal with the noisy nature of optical flow vectors.

Several methods exist for recognizing human actions directly from global image measurements such as optical flow or spatio-temporal gradients [11,7,1,12]. Various factors such as spatial resolution, global camera motion, dynamic background clutter, may affect the recognition performance of these methods. Local space-time features or interest points provide compact and informative representations of patterns in an image, they are robust to occlusion and background clutter, and can be adapted to the size, frequency and the velocity of moving patterns, and therefore suitable for recognizing complex motion patterns. Laptev and Lindeberg [13] extend the notion of spatial interest points into the spatio-temporal domain where the image values have significant local variations in both space and time. Scale-invariant spatio-temporal descriptors (the 3rd order local jets of normalized spatio-temporal Gaussian derivatives) are developed and used to classify events. Schuldt et al. [3] construct video representations in terms of local space-time features and integrate such representations with SVM classification schemes for event recognition. They introduce a new video database containing six human actions performed by 25 people in four different scenarios and demonstrate promising results. Our method adopts local features to capture actions and behaviors and use this commonly acknowledged data set for evaluation.

In addition to low level structural features, for human action detection, semantic features such as face and hands can also be utilized to effectively capture human motion patterns. Most recently, Ke et al. [2] generalize the notion of 2D rectangle features used by Viola and Jones [14] to 3D spatio-temporal volumetric features. A real time event detector is constructed by learning a cascade of filters based on volumetric features that efficiently scans video sequences in space and time. We propose to use the combination of semantic and structural spatio-temporal features as our basic visual atomic action representation.

The cornerstone of our algorithm, string matching, has been widely used in bioinformatics applications, where the string is used to represent DNA genomes as sequences of nucleotides, proteins as sequences of amino acids. Along with the increasing popularity in bioinformatics and many others, a great deal of effort has been devoted to string analysis in the past few years [15,16,17]. Such biologically meaningful technique inspired us to develop the behavior analysis algorithm using the string matching which we will describe in detail in the following section.

3 Proposed Algorithm

As mentioned in Section 2, there are many existing action and event recognition algorithms. Some detection algorithms apply quite rigid constraints on temporal continuation [18,2], while other methods ignore the temporal order constraints and use bag-of-words [19]. A desirable approach should maintain the temporal order of the action sequences, at the mean time allows for some degree of tolerance on partial matches and insertions of irrelevant symbols. Such considerations motivate us to propose the action recognition algorithm based on string matching. The thrust of our method is to map an action sequences to a string sequences. Then many powerful string computational techniques can be employed to analyze the human behaviors. The most popular one is to compare the strings by counting the common substrings they include: the more substrings in common, the more similar they are. This idea leads to the *longest common subsequence problem* (LCS) [20]: finding a longest sequence which is a subsequence of all sequences. It is a classic computer science problem (one example is the *unix* utility *diff*) and has many applications in bioinformatics [15].

An important aspect of LCS is that the substrings do not need to be contiguous, only the order of the substrings is maintained, which enables the LCS strike a good balance between the rigid temporal order constraint based methods and bag-of-words based methods. Besides, as explained in the following, the LCS can be computed efficiently [20] with dynamic programming. It also can be naturally put into the framework of kernel methods, which greatly improves its generalization capability.

The main components of our approach consists of the three parts (as shown in Fig.2): first, the atomic actions are extracted and mapped into symbols. Symbols are concatenated sequentially into strings. Then string matching algorithm is applied to find the LCS. A naive implementation of the longest common subsequence algorithm will result in exponential computational complexity. For-

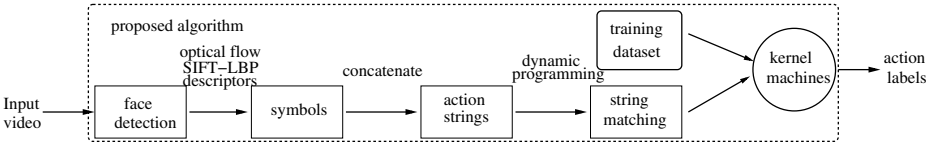


Fig. 2. The main components of the proposed algorithm

tunately, there are fast algorithms which are based on dynamic programming reducing the complexity greatly. Last, we use the LCS as the similarity measure between two strings to induce a kernel function (called string kernel). Such string kernel can represent quite complex behaviors, and in the mean time, it inherits the good generalization of kernel methods.

3.1 Symbol Extraction

To reliably match the actions, we need to define an atomic action vocabulary, on which the symbols in the strings are defined. There are a variety of ways to extract the symbols. Roughly they can be categorized into two classes: semantic model and learning based extraction(such as AdaBoost feature selection [2]). For our application, human behavior analysis, we adopt the former as our focus of attention and anchor points. We also developed a novel structural feature called SIFT-LBP that can effectively represent action characteristics in the optical flow field.

Focus of Attention. The first step of human behavior analysis is to find the interest region of the person. A variety of methods exist to detect the person. Based on our specific domain, we choose to detect the human faces using the face detector [14]. Then we use the detected face as the anchor point to find the body parts. The reason we choose the face detection is that it is fast, reliable and can effectively find the person from cluttered environment. The detected face with its movement also serves as atomic action symbols. Compared with the other implicit focus-of-attention approaches [1,2,3], this method is more meaningful and precise.

SIFT-LBP Descriptor. Once we find the person, we need a descriptor to describe the actions. Many descriptors come directly from the interest point detection algorithm, such as space-time interest point [13]. Some are inspired by the Haar-feature used in face detection [14], such as volumetric features [2]. Space-time interest points based on finding the space-time Harris corners, in practice are quit effective to find the sparse features. However, in some circumstances, they are too rare for action recognition, as observed by Dollar et al. [21,19]. In addition, the claim that the corners detected are actually the ones we want for action recognition is still questionable.

Our approach uses the optical flow as the primary feature, which is relatively insensitive to appearance and illumination variations. Optical flow is a widely

used features for the action recognition [2,1,12]. Here we use the sparse optical flow [22] as our feature vectors. The real-time speed of optical flow is another advantage for practical system.

Inspired by the ideas of the SIFT descriptor [21] and the Local Binary Patterns(LBP) [23], we design a feature descriptor (called SIFT-LBP) for the optical flow: we first partition the human body into $m \times n$ blocks (empirically we choose 2×3), and find the dominant optical flow orientation in each block. Then we apply LBP to encode the position and the orientation information into a binary number.

The LBP is an excellent measure of the spatial structure of image texture. It was first introduced by Ojala et al. [23] for texture analysis. Approximately at the same time, Zabih and Woodfill [24] also proposed a similar idea called *census transform*. The definition of LBP is as follows: given a pixel at position (x_c, y_c), LBP is a sequence of binary comparisons of intensities between the pixel and its neighbors $p = 0, \cdots, N-1$, which is encoded into a binary number:

$$LPB(x_c, y_c) = \sum_{p=0}^{N-1} s(g_p - g_c)2^p, \tag{1}$$

where g_c and g_p are the intensities of pixel at (x_c, y_c) and its neighbors, and the step function

$$s(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ 0 \text{ if } x < 0 \end{cases} \tag{2}$$

An example of the LBP is shown in Fig. 3. By definition, LBP is invariant to affine changes in illumination and potentially to rotations, which makes it robust.

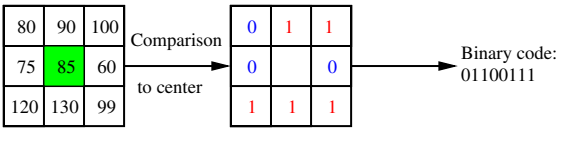


Fig. 3. An example of LBP algorithm

To make the LBP more suited for action detection and classification application, we make the following changes. First, instead of comparing with the center block, we compare with the mean value of all blocks, which addresses the problem that the LBP measure heavily relies on the intensity of center pixel. Second, we encode not only the position information into the binary code, but also orientation information by quantizing the angles. Such encoding method is a mixture of the SIFT descriptor and the LBP. In practice, we find that the SIFT-LBP is very robust against pose and illumination changes and its computational cost is low.

3.2 String Matching Using Dynamic Programming

Once we generate the strings from the actions, we need to find the LCS between them. Given two strings s with length $|s|$ and t with length $|t|$, the LCS algorithm is to find a common substring with maximal length.

A brute-force approach to solving the LCS problem is to enumerate all substrings of s then check each of them to see if it is a substring of s, at the mean time keeping the longest one. There are $2^{|s|}$ substrings in s, so the computational complexity of this approach is of exponential order, which makes it computationally prohibitive even for sequences of modest length.

The LCS problem has an important property: optimal-substructure, which allows it to be efficiently solved using dynamic programming [20], as shown in Eq.(3):

$$c(i,j) = \begin{cases} 0 & \text{if } i = 0 || j = 0 \\ c(i-1, j-1) + 1 & \text{if } s_i = t_j \\ \max(c(i, j-1), c(i-1, j)) & \text{if } s_i \neq t_j \end{cases} \tag{3}$$

where $c(i, j)$ is a table size of $|s| \cdot |t|$, keeping track of the longest substring found. The dynamic programming can be implemented by filling the table bottom up. A concrete example of string matching using dynamic programming is shown in Table 1 where each entity represents the optimal solution to the sub-problems. The final solution is marked in red and the right-bottom corner of the table is the length of LCS.

Both the computational and storage complexity are $|s| \cdot |t|$, much less than previous exponential order. If we only need to find the length of the LCS, we can further reduce the storage size to $\min(|s|, |t|)$. Currently, the fastest algorithm for longest common subsequences runs in time $O(|t| \log |r| + f \log \log \min(f, |s||t|/f))$ [20]. Typically, the quadratic complexity algorithm is fast enough to match the two action strings.

3.3 String Kernel Machines

There are a lot of variations in the action sequences, even performed by the same person. To deal with the large variations, a large amount of data is needed to train the system, which makes direct comparison between strings not effective and computationally expensive. Kernel methods provide an effective alternative to handle the variations and richness of the raw data [25,26].

The standard kernel methods require the input data vectors to be of the same length. However the lengths of action strings can vary dramatically from one action to another, which makes the standard kernel methods not feasible. To deal with this type of data, kernel trick [25] has been used to avoid explicitly computing the kernel function. Instead a similarity measure between each pair of strings is computed using string matching. This similarity measure embeds the finite string space to the infinite feature space through the kernel trick such that

$$K(s,t) = LCS(s,t) \tag{4}$$

where s and t are a pair of strings.

Table 1. An example of string matching using dynamic programming. The table keeps track of the optimal solutions of the subproblems. The final solution of LCS is marked in red and the length of LCS is 6.

	j	0	1	2	3	4	5	6	7	8	9
i		t_j	e	x	h	a	u	s	t	e	d
0	s_i	0	0	0	0	0	0	0	0	0	0
1	e	0	1	1	1	1	1	1	1	1	1
2	x	0	1	2	2	2	2	2	2	2	2
3	c	0	1	2	2	2	2	2	2	2	2
4	u	0	1	2	2	2	3	3	3	3	3
5	s	0	1	2	2	2	3	4	4	4	4
6	e	0	1	2	2	2	3	4	4	5	5
7	d	0	1	2	2	2	3	4	4	5	6

The feature mapping (4) introduces bias by the length of the strings. We can remove this effect by normalizing the feature vectors in the feature space with a new embedding defined as $\hat{\phi}(s) = \phi(s)/\|\phi(s)\|$. So the new kernel is

$$\hat{K}(s,t) = \frac{K(s,t)}{\sqrt{K(s,s)K(t,t)}}. \tag{5}$$

Once we have the kernel function, we can use kernel machines to classify the actions by training. Support Vector Machines [25] method is chosen for this purpose. The support vectors condense the information from all the sequences by removing the irrelevant ones. Another benefit is that the computational cost can be substantially reduced if we select the features(strings) by thresholding the weights of support vectors.

4 Experimental Results

The first set of experiments verify our algorithm's capability to recognize actions performed at different poses. We collected a database of 80 video sequences (194× 411 at 25 fps) performed by eight different persons, each performing three natural actions including "turning head", "wiping head", "fidgeting hands". There are also eight confuser actions such as "making a phone call". The pose variation

Table 2. Classification confusion matrix using our method. Trace = 400.01.

	box	clap	wave	walk	jog	run
box	50	16.67	0	0	0	0
clap	16.67	83.33	16.67	0	0	0
wave	33.33	0	66.67	0	0	0
walk	0	0	16.67	66.67	33.33	33.33
Jog	0	0	0	0	66.67	0
Run	0	0	0	33.33	0	66.67

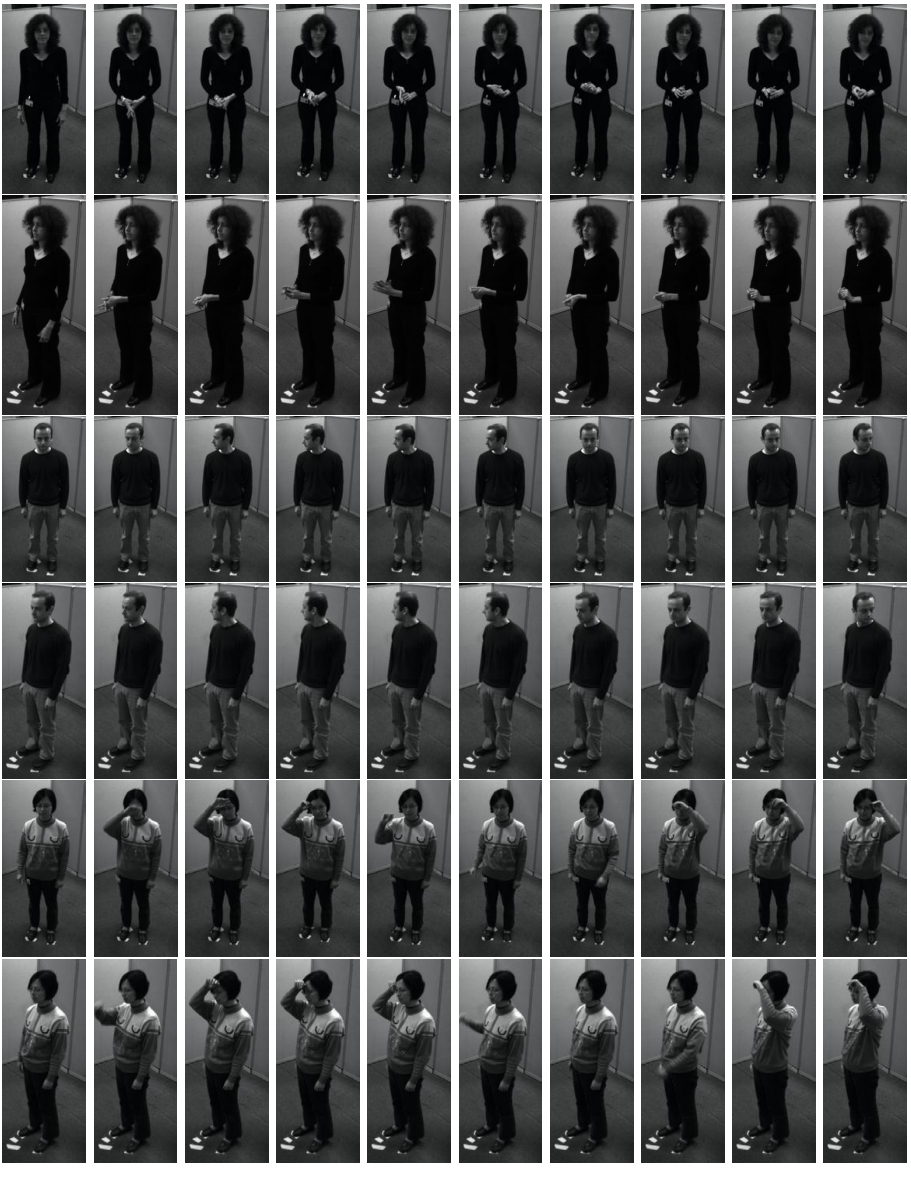


Fig. 4. Examples of video sequences. Note that the pose and appearance variations are quite large.

includes 0°, 30° and 45°. Some sample image sequences are shown in Fig.4. Note that the pose and appearance variations are quite large.

The first experiment is tested on the sequences of pose 0°. For each sequence we detect the faces and body parts. Then we apply the SIFT-LBP descriptors as described in section 3 to find the corresponding symbols in each frame and

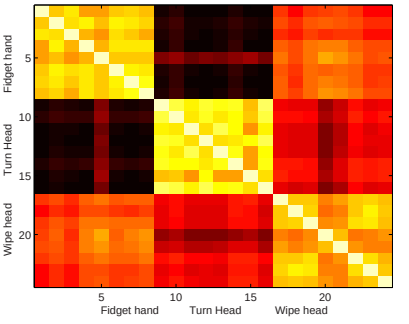


Fig. 5. Confusion matrix of the pairwise comparison of actions using string matching at pose 0°. The brighter the entry is, the more similar the actions are.

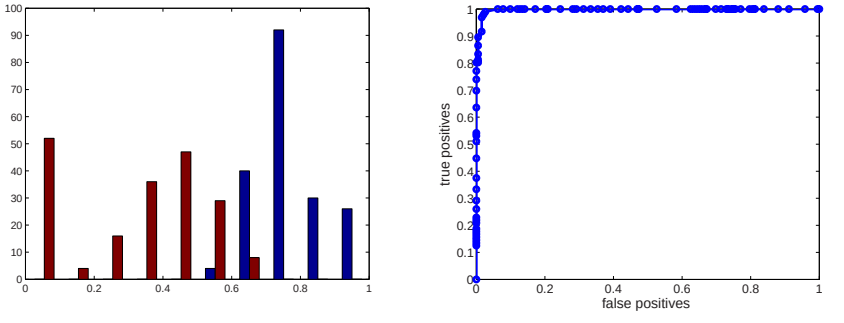


Fig. 6. P-same-P-different plot and the ROC for action comparison using string matching at pose 0°. Red is the different and blue is the same.

generate action string for each video sequence. Then we compare each pair of the strings using dynamic programming based string matching. Fig. 5 plots the confusion matrix of the action matching, where each cell is the normalized score of string matching. The confusion matrix shows apparent diagonal block structure. To verify this block effect, we plot a histogram showing the same v.s. different w.r.t. the normalized string matching score (P-same-P-different plot) and the corresponding ROC curve in Fig. 6. Finally we perform leave-one-out procedure and the average classification accuracy is 100%.

To demonstrate the algorithm's ability to handle pose variation, we conduct the second experiment on actions performed at poses 0°, 30°, and 45°. Following the same procedure as above, we obtain a confusion matrix as shown in Fig. 7. Again we can see a clear diagonal block structure in the confusion matrix. The P-same-P-different plot and the ROC are shown in Fig. 8. Similarly we perform leave-one-out procedure and the average classification accuracy is 98.611%.

We also tested our algorithm on the publicly available KTH human action dataset [3] and achieved comparable performance to state-of-art, while the

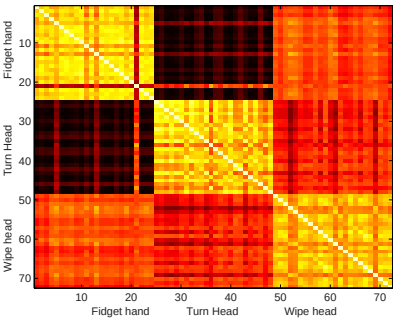


Fig. 7. Confusion matrix of the pairwise comparison of actions using string matching at pose 0°, 30°, and 45°. The brighter the entry is, the more similar the actions are.

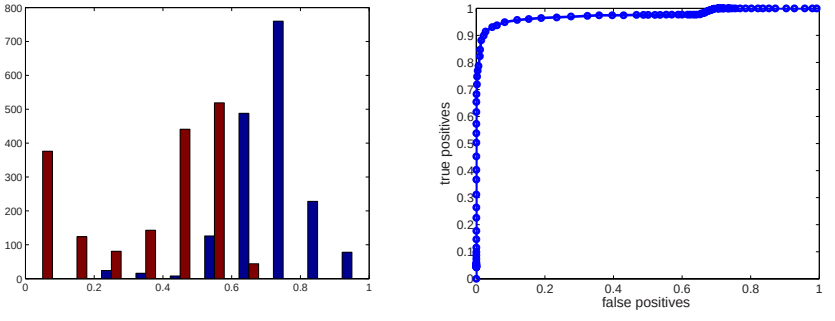


Fig. 8. P-same-P-different plot and the ROC for action comparison using string matching at pose 0°, 30°, and 45°. Red is the different and blue is the same.

computational complexity of our algorithm is much lower. Besides the database is lack of pose variations and background changes, so our method is not fully exploited. The confusion matrix is shown in Table 2. The trace of the confusion matrix is used as one of the measures of classification performance, ranging from 100 (random classification) to 600(perfect classification). The trace of our algorithm is 400.01 which is better than Ke's method [2] (377.8) and worse than Schuldt's method [3](430.3).

The time complexity of the algorithm is very modest. All the algorithms are implemented in C++. The optical flow algorithm and SIFT-LBP run at speed of frame-rate. The average string matching time is 0.12 millisecond on a PC with Pentium M 2.13GHZ CPU.

5 Conclusions and Discussions

This paper presents an action analysis method based on robust string matching using dynamic programming and kernel methods. The key contributions of this paper are as follows:

First, we present a new framework utilizing the powerful and flexible string analysis techniques for action analysis. The action sequences or event sequences are so dynamic and versatile that makes the traditional data types such as fixed length vectors inadequate to handle. In contrast, the string representation is more flexible and more expressive. More advanced data structure such as trees, graphs can be applied similarly.

Second, we apply the dynamic programming to find the LCS between a pair of strings, which gives a good indication about the similarity of actions sequences. The LCS algorithm is quite flexible since it tolerates some degree of the repetition, missing or wrong symbols in the strings due to the imperfection of observations.

Third, we naturally integrate the string matching into the kernel method framework using the similarity measure based on the LCS. So it inherits the desirable generalization capability and efficiency from the kernel methods.

Last, we design a new symbol extraction method called SIFT-LBP. It combines the ideas from SIFT descriptor and the LBP and is suitable for the action recognition. To design descriptors manually or to find them automatically is a disputable topic. On one hand, the manually designed descriptors are encoded the human knowledge about the problem to attack, so they are more meaningful, more concise and more informative. Besides, it requires no training, another advantage. So in this sense, a good design is potentially better than the automatically found ones. On the other hand, the automation of the feature selection saves a lot of human effort and it also encodes the information from the training data. In this paper, based on our scenario, we chose the former and obtained satisfactory results. Of course, it is worthwhile to enclose this technique to our framework.

Another advantage of our method is the speed and simplicity. We implemented the algorithm in C++ and achieved frame-rate speed on ordinary PC. The initial experimental results confirmed the robustness and efficiency of our methods.

There are many potential extensions of the framework. One example may be a Google video [8] like application, where bag-of-features are used. The string matching technique will bring the video search to event mining level if the order information is properly utilized. Another one may be the generic object category recognition. The ordinary feature vector is notoriously hard to represent the heterogenous objects. In contrast, the string representation is much benign to them. The context information can be naturally put into this framework.

References

1. Efros, A.A., Berg, A.C., Mori, G., Malik, J.: Recognizing action at a distance. In: International Conference on Computer Vision, Nice, France, pp. 726–733 (2003)
2. Ke, Y., Sukthankar, R., Hebert, M.: Efficient visual event detection using volumetric features. In: International Conference on Computer Vision, vol. I, pp. 166–173 (October 2005)

3. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: A local svm approach. In: International Conference on Pattern Recognition, Cambridge, United Kingdom, vol. 3, pp. 32–36 (August 2004)
4. Zhong, H., Shi, J., Visontai, M.: Detecting unusual activity in video. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. II, pp. 819–826. IEEE Computer Society Press, Los Alamitos (2004)
5. Boiman, O., Irani, M.: Similarity by composition. In: Neural Information Processing Systems, Vancouver, Canada (2006)
6. Yacoob, Y., Black, M.: Parameterized modeling and recognition of activities. Computer Vision and Image Understanding(CVIU) 73, 232–247 (1999)
7. Davis, J.W., Bobick, A.F.: The representation and recognition of action using temporal templates. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 928–934. IEEE Computer Society Press, Los Alamitos (1997)
8. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: International Conference on Computer Vision, Nice, France, vol. 2, pp. 1470–1477 (October 2003)
9. Schodl, A., Szeliski, R., Salesin, D.H., Essa, I.: Video textures. In: Proceedings of the conference on Computer graphics and interactive techniques, pp. 489–498 (2000)
10. Zelnik-Manor, L., Irani, M.: Event-based analysis of video. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. II, pp. 123–130. IEEE Computer Society Press, Los Alamitos (2001)
11. Black, M.J., Jepson, A.D.: Eigentracking: Robust matching and tracking of articulated objects using view-based representation. International Journal of Computer Vision 26(1), 63–84 (1998)
12. Shechtman, E., Irani, M.: Space-time behavior based correlation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 405–412. IEEE Computer Society Press, Los Alamitos (2005)
13. Laptev, I., Lindeberg, T.: Space-time interest points. In: International Conference on Computer Vision, pp. 432–439 (2003)
14. Viola, P., Jones, M.: Robust real-time object detection. International Journal of Computer Vision 57(2), 137–154 (2004)
15. Gusfield, D.: Algorithms on Strings, Trees and Sequences–Computer Science and Computational Biology. Cambridge University Press, Cambridge (1997)
16. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. J. Mach. Learn. Res. 2, 419–444 (2002)
17. Leslie, C.S., Eskin, E., Cohen, A., Weston, J., Noble, W.S.: Mismatch string kernels for discriminative protein classification. Bioinformatics 20(4), 467–476 (2004)
18. Ivanov, Y., Bobick, A.: Recognition of visual activities and interactions by stochastic parsing. IEEE Trans. on Pattern Analysis and Machine Intelligence 22, 852–872 (2000)
19. Dollar, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: ICCV VS-PETS, Beijing, China, pp. 65–72 (2005)
20. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press and McGraw-Hill (2001)
21. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
22. Shi, J., Tomasi, C.: Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, pp. 593–600. IEEE Computer Society Press, Los Alamitos (1994)

23. Ojala, T., Pietikainen, M., Harwood, D.: A comparative study of texture measures with classification based on feature distributions. Pattern Recognition 29, 51–59 (1996)
24. Zabih, R., Woodfill, J.: Non-parametric local transforms for computing visual correspondence. In: European Conference on Computer Vision, vol. II, pp. 151–158 (1994)
25. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
26. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)

Author Index