Available online at www.sciencedirect.com
SCIENCE @ DIRECT®

Graphical Models 66 (2004) 333–351

Graphical Models

www.elsevier.com/locate/gmod

Combining 3D flow fields with silhouette-based human motion capture for immersive video

Christian Theobalt*, Joel Carranza, Marcus A. Magnor, Hans-Peter Seidel

MPI Informatik, Saarbrücken, Germany

Received 31 January 2004; received in revised form 28 April 2004; accepted 9 June 2004
Available online 12 September 2004

Abstract

In recent years, the convergence of computer vision and computer graphics has put forth a new field of research that focuses on the reconstruction of real-world scenes from video streams. To make immersive 3D video reality, the whole pipeline spanning from scene acquisition over 3D video reconstruction to real-time rendering needs to be researched. In this paper, we describe latest advancements of our system to record, reconstruct and render free-viewpoint videos of human actors. We apply a silhouette-based non-intrusive motion capture algorithm making use of a 3D human body model to estimate the actor's parameters of motion from multi-view video streams. A renderer plays back the acquired motion sequence in real-time from any arbitrary perspective. Photo-realistic physical appearance of the moving actor is obtained by generating time-varying multi-view textures from video. This work shows how the motion capture sub-system can be enhanced by incorporating texture information from the input video streams into the tracking process. 3D motion fields are reconstructed from optical flow that are used in combination with silhouette matching to estimate pose parameters. We demonstrate that a high visual quality can be achieved with the proposed approach and validate the enhancements caused by the the motion field step.
© 2004 Elsevier Inc. All rights reserved.

* Corresponding author. Fax: +49 681 9325 499.
E-mail addresses: theobalt@mpi-sb.mpg.de (C. Theobalt), carranza@mpi-sb.mpg.de (J. Carranza), magnor@mpi-sb.mpg.de (M.A. Magnor), hpseidel@mpi-sb.mpg.de (H.-P. Seidel).

1524-0703/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.gmod.2004.06.009

Keywords: 3D Video; Human motion capture; Optical flow; 3D Motion field; Scene flow

1. Introduction

Recent developments in media technology show that there is a strong interest of the entertainment industry to enhance the classic 2D video into an immersive and interactive 3D medium. The ultimate goal is to create a feeling of immersion by enabling the viewer to choose an arbitrary viewpoint onto the scene in real-time and without loss of visual quality. The range of applications for this technology will be manifold and will allow impressive effects such as virtual fly-arounds in sports videos or actor-dependent viewpoint selection in interactive movies.

Human actors are the central elements of motion picture scenes, and the human visual system is very sensitive to the slightest inaccuracies in a human's motion and look. In consequence, the synthesis of realistic images of humans in motion is a challenging problem in computer graphics. Two aspects of this problem are the creation of natural human motion and the accurate rendering of a person's appearance. Combining the small scale details of skin, muscle, and cloth movement with the large-scale motions of the body into one realistic image has required the development of new techniques which rely on the strengths of both computer vision and computer graphics. Many conventional methods for estimating motion parameters are intrusive, requiring optical markers or complex mechanical setups, and thus require a separation of the generation of realistic motion from the generation of realistic appearance. However, in the field of computer vision, numerous techniques have been developed for non-intrusive motion parameter estimation. Incorporating some of these techniques into computer graphics allows us to capture body appearance and motion at the same time, vastly simplifying the problem of novel image synthesis.

We have developed a method which non-intrusively estimates motion parameters using silhouette information [5]. This method employs a detailed geometric body model and uses features of latest-generation commodity graphics hardware to estimate pose parameters. The estimation is performed by optimizing the overlap between the model silhouette and the silhouettes obtained from multi-view video data. Due to its compartmentalized nature the estimation algorithm lends itself to a parallel implementation.

The reconstructed scene is rendered at video frame rate and allows the viewer to change its viewpoint freely and in real-time. For rendering, the body model is textured with high-detail time-varying textures that are created from the video data by means of image-based techniques.

We have previously demonstrated that a broad range of complex and rapid body motion is robustly captured using silhouette-based techniques [5]. However, improvements are possible in those portions of the body with small-scale details (such as features of the face) whose visual appearance is erroneous due to small pose inaccuracies. To be maximally effective, we enhanced our original silhouette-based tracking algorithm into a hybrid approach that makes use of all available information. We propose to use texture information to augment the silhouette-fitting process.

Optical flow reconstruction is a computer vision technique that employs texture information to compute a 2D motion field in the image plane. Making use of optical flow calculations from multiple input views and an a priori body model, it becomes possible to construct a 3D motion field which estimates body motion. We present a method for extracting hierarchical rigid body transformations from these motion fields and show that it is used best in conjunction with, and not in place of, silhouette-based tracking.

We demonstrate that this new hybrid method improves motion parameter estimation and, consequently, that it has a significant impact on the quality of generated free-viewpoint video sequences.

This article proceeds with an overview of related work in Section 2, which is followed by a general overview of our free-viewpoint video system in Section 3. The main functional components and algorithmic ingredients are outlined in the sections thereafter, beginning with the acquisition setup for multi-view video streams in Section 4. Adaptable body model and multi-view texture generation are presented in Sections 5 and 7, respectively, and the silhouette fitting step in our motion capture algorithm is described in Section 6. The theoretical foundations of optical flow estimation and the reconstruction of 3D motion fields from 2D flows are presented in Section 8. The nuts and bolts of how to compute differential pose update parameters from 3D flow fields are explained in Section 9. Graphical results and a validation of visual improvements by the motion field step are presented in Section 10, and the paper concludes with an outlook in Section 11.

2. Related work

In the scientific literature the problems of capturing human motion from video data and of realistic rendering from novel viewpoints have rarely been tackled in tandem. In computer vision, non-intrusive optical human motion capture has been an active field of research for a long time (see [9,22] for comprehensive reviews). Some methods work on a single 2D image and apply, for example, frame differencing [16] or image skeletonization [12] to fit simple body models to human motion. 3D human motion capture approaches typically employ an explicit human body model consisting of a joint structure and some form of surface representation. Simple shape primitives, such as cylinders [13,26] or superquadrics [10], are commonly used to represent limbs. The body models are fitted to the motion by aligning their projection with features in the image plane, such as image discontinuities. The application of silhouette images for human motion capture has also been considered. In [7] a force field exerted by multiple image silhouettes aligns a 3D body model. In [24] a combination of stereo and silhouette fitting is used to fit a human body model, and in [5] a silhouette-based motion estimation method is described that exploits graphics hardware to maximize model and silhouette overlap. Recently, the application of reconstructed volumetric models (visual hulls) from silhouettes of a moving person for motion capture has also been considered. Ellipsoidal body models [6], kinematic skeletons [20], or skeleton models with attached volume samples [32] are fitted to the volume data.

Recently, Sand et al. [27] have presented a novel method for estimating human body motion and a deformable skin model. They simultaneously record silhouette images and capture motion parameters with a marker-based system in order to estimate a mapping from pose parameter space to the space of skin deformations.

The intent to lay the foundations for the next generation of electronic visual media has brought together researchers from vision and graphics in the field of 3D video. In 3D video, dynamic models of scenes that are recorded from several camera perspectives are reconstructed and rendered from novel viewpoints. Examples of methods that are used to approach that problem are shape-from silhouette-like approaches, such as the visual hull [21,38], stereo-based approaches [23] or a combination of both [28]. Ray-space methods from image-based rendering that approximate the plenoptic function [1] in a scene, such as the lightfield [18] or the Lumigraph [11], can also be considered to be part of the same effort. However, their immense memory requirements and the necessary sampling density demand large camera arrays and huge data storage systems in order to record dynamic scenes [37].

None of the previously mentioned approaches explicitly uses optical flow or incorporates 3D velocity fields into motion parameter estimation or scene reconstruction. The optical flow is the observed 2D motion field in the image plane of a camera resulting from the projection of the 3D velocity field of the recorded moving scene (see [2] for a comparison of optical flow algorithms). The application of 2D optical flow has been investigated in model-based video coding for deriving facial animation parameters of a generic head model [8] or for recovering motion parameters of a body model in a teleconferencing scenario [17]. Using optical flow in one or more camera views for full body human motion estimation is presented in [4]. In their work, the authors use a twist parameterization for rigid body transformations to determine the body pose parameters from 2D information directly by solving a linear system. The algorithm computes pose updates and performs image warping in an iterative procedure. None of these methods explicitly reconstruct a 3D motion field. In [35], an algorithm for computing such a 3D motion field from optical flows in multiple camera views is presented. It has been used to improve voxel-based scene reconstruction [36] and to compute models of intermediate time steps in a sequence of shape-from-silhouette representations [34]. Unfortunately, the methods employing optical flow exhibit robustness problems if the typical optical flow assumptions such as brightness constancy over time and smoothness in a spatial neighborhood are not fulfilled. This happens very easily if the motion in the scene is very rapid and effects such as self-shadowing come into play.

In our work we combine into a single framework techniques that were previously investigated separately. We demonstrate that the combination of a silhouette-based human motion capture algorithm and multi-view texture generation forms a powerful tool for acquisition and realisic rendering of 3D videos of human actors [5]. Furthermore, we describe a method that uses a 3D motion field reconstructed from optical flow to update pose parameters computed via a silhouette-based model fitting procedure [30]. By this means, we combine the strengths of silhouette-based fitting for robust acquisition of a large range of motions with that of motion estimation from texture information for robust computation of small-scale pose corrections. Using texture information, pose updates can be recovered on a small scale.

3. The big picture

In Fig. 1, an overview of the proposed free-viewpoint video system is shown. It is functionally separated into an offline and an online component, the former one comprising the acquisition and motion capture sub-systems, the latter one consisting of the real-time free-viewpoint renderer.

The system takes synchronized multi-view video streams as input and computes the silhouette of the person in each video frame via background subtraction. In an initialization step, the employed body model is adapted to the physical shape of the recorded person. At every time step, the system computes multi-view textures for the body model from the video images using image-based techniques (Section 7). After initialization, the motion capture algorithm iteratively estimates the body pose parameters for each time step. The described motion capture algorithm implements a two-step predictor-corrector scheme. Considering an arbitrary time step $t + 1$, the motion capture algorithm works as follows. Starting with a body pose P_t recovered for time step t, the system first computes an estimate of the pose parameters $P'_{\mathrm{sil},t+1}$ at time $t + 1$ by optimizing the overlap between the projected model and the silhouette images in all camera views. In a second step, estimate $P'_{\mathrm{sil},t+1}$ is augmented

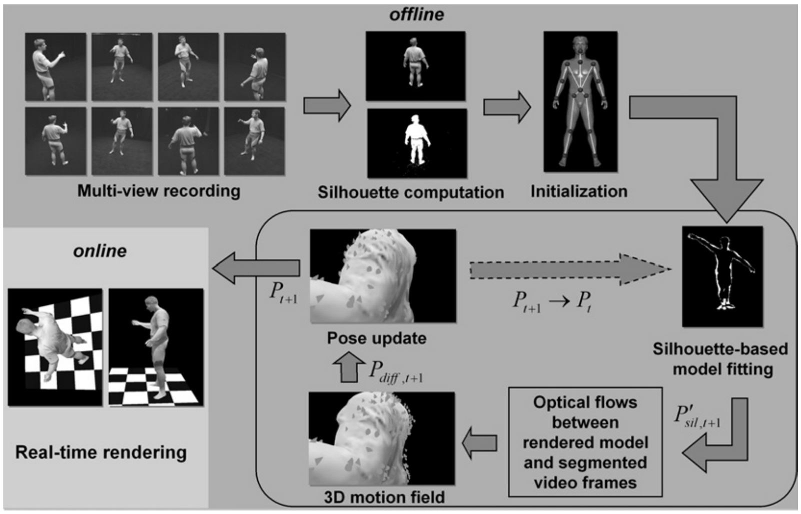

Fig. 1. Overview of the functional units of the free-viewpoint video system. The system consists of an online (light gray background) and an offline component (darker gray). The elements of the motion capture sub-system are shown in the rounded box.

by computing a 3D corrective motion field from optical flows. The model standing in pose $P'_{\text{sil},t+1}$ and textured with the video images from time t is rendered into all camera views. The images of the back-projected model form a prediction of the person's appearance at $t + 1$. The optical flows are computed for each pair of back-projected model view and corresponding segmented video frame at time $t + 1$.

The reconstructed motion field provides an estimate of the correct positions of the limbs of the body model to be conform with the acquired image data. From the motion field we compute a least-squares differential pose update, $P_{\text{diff},t+1}$, i.e., a set of pose parameters that are added to $P'_{\text{sil},t+1}$ to form the final pose estimate P_{t+1} for time t+1. The final pose parameter estimate serves as a starting point in the next iteration.

Once the pose parameters for all time steps are obtained, they are saved into a 3D video file that may be played back from arbitrary perspectives in real-time. The renderer displays the body model in the sequence of recovered body poses and creates the realistic surface appearance by generating multi-view textures from video.

4. Multi-view video acquisition

The video sequences used as input to our system are recorded in our multi-view video studio [31]. IEEE1394 cameras are placed in a convergent setup around the center of the scene. The video sequences used for this paper are recorded from 8 static viewing positions arranged at approximately equal angles and distances around the center of the room. The cameras are synchronized via an external trigger, and all the video data are directly streamed to hard drives by the four control PCs, each of which is connected to two cameras. Video frames are recorded at a resolution of 320×240 at 15 fps. The frame rate is fundamentally limited to 15 fps by the camera hardware's external trigger capabilities. Using Tsai's algorithm [33], the cameras' intrinsic and extrinsic parameters are determined, calibrating every camera into one common global coordinate system. The lighting conditions are controlled and all cameras are color-calibrated.

In each video frame, the person in the foreground is segmented via background subtraction. The algorithm employs per-pixel color-statistics to generate silhouettes [6]. Shadow regions that might lead to an incorrect classification of background pixels as foreground are eliminated via an additional threshold on pixel hue values.

5. Adaptable body model

The body model used is a generic model consisting of a hierarchic arrangement of 16 body segments (head, upper arm, torso, etc.), each of which is represented by a closed triangle mesh (see Fig. 2 and Fig. 4). In total, the surface geometry consists of 21422 triangles. The model's kinematics are defined via an underlying skeleton consisting of 17 joints connecting bone segments. Rigid transformations at each of these joint locations define a specific body pose for the model. These transformations are constrained to imitate the actual motions of the body. Shoulder and hip joints

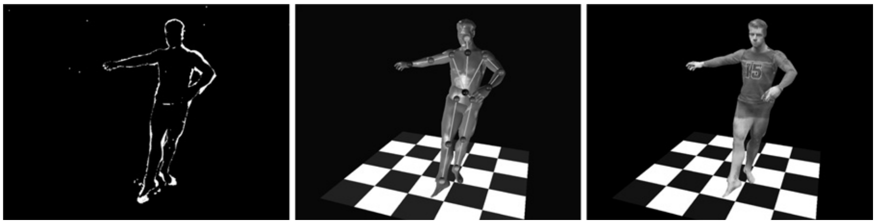

Fig. 2. Exclusive-or operation (XOR) of model and image silhouette (l), body model with underlying kinematic skeleton in recovered body pose (m), and textured body model (r).

are represented by 3 degree-of-freedom (DOF) ball joints, and elbow and knee joints are represented by 1-DOF hinge joints. In total, 35 parameters are needed to completely define a body pose. While the actor stands in a specific initialization pose, the generic body model shape is adjusted to that of the person through a silhouette-based fitting process. Shape-adaptation of the body model is achieved by adjusting body pose parameters and additional scaling parameters in an iterative procedure that converges when optimal conformance of model shape and silhouette images is reached [5]. From this point on, bone lengths and segment geometry are fixed and motion parameter estimation is performed using a combination of silhouette and motion field information.

6. Silhouette-based model fitting

For each new time step of video, the motion capture sub-system begins with the computation of a set of pose parameters $P'_{\mathrm{sil},t+1}$ that maximizes the overlap between the projected model and the input silhouettes. These pose parameters are found by means of a hierarchical non-linear optimization procedure which is initialized using the pose parameters P_t that were found in the preceding time step. The error function that drives motion capture is a per-pixel XOR-operation between the projected model silhouettes and input silhouettes in each camera view (Fig. 2). It can be efficiently computed using current consumer graphics hardware. The pose parameters are found by solving a sequence of optimization problems on subsets of the parameter space. Following the skeleton hierarchy, the pose parameters of the model's root joint, located in the torso segment, are found first. Thereafter, the poses of arms, legs and head are derived and finally the poses of the feet and hands are computed [5].

Recent results show that the silhouette fitting algorithm can be significantly sped up by implementing it as a distributed client-server system [29]. In this parallel implementation, four client PCs and one server PC are used. The independence of poses of body parts on the same level of the skeleton hierarchy, e.g., arms, legs, and head, can be exploited in the parallel implementation. Instead of computing the pose parameters of each arm, each leg, and the head sequentially on a single machine, their poses can be found on five different machines in parallel. The same goes for hands and feet. Following the skeleton hierarchy top to bottom, the server starts by finding the root parameters. On the next two hierarchy levels the work is distributed among the five GPUs and CPUs. A further speedup of the XOR evaluation is gained by considering only sub-regions of the image plane and by excluding unchanging model parts from rendering. In this work, the silhouette-based model fitting is used for robust prediction of pose parameters.

7. Multi-view texture generation

For any body pose, it becomes possible to generate a textured body model by projecting the input camera views onto the body model surface using programmable graphics hardware. The degree to which a specific input view is visible at a given surface location is variable. Per-vertex blending weights are computed based on visibility and the difference in direction between the vertex normal and the input view vector. Ref. [5] addresses texture generation for the purpose of novel viewpoint generation. We make use of texture generation for both rendering and motion parameter estimation. This requires a slight modification. When rendering the textured model for motion field computation, the texture coordinates generated from the previous time step are used.

8. Fundamentals of optical flow and its 3D equivalent

This section briefly reviews the mathematical preliminaries of optical flow and the reconstruction of 3D motion fields.

8.1. 2D optical flow

The optical flow is the projection of the 3D velocity field of a moving scene into the 2D image plane of a recording camera. The determination of the 2D optical flow from spatio-temporal intensity variations in images has been investigated in computer vision for many years [2].

A number of simplifying assumptions are typically made to compute the optical flow from the pixel intensities of two subsequent images. First, it is assumed that the change in image intensity is due to translation in the image plane only (intensity constancy constraint)

$$I(x,t) = I(x - \mathbf{o}t, 0), \tag{1}$$

where $\mathbf{o} = (p, q)^{\mathrm{T}}$ is the optical flow at image point x, I being the image intensity at coordinates x and time t. From the Taylor expansion of (1) and linearization, the *optical flow constraint equation* is derived

$$\nabla I(x,t) \cdot \mathbf{o} + I_t(x,t) = 0, \tag{2}$$

where $I_t(x,t)$ is the temporal derivative. This is an equation in two unknowns which cannot be solved at a single image plane location without additional assumptions. Hence, it is common practice to make additional assumptions about the smoothness of the optical flow field in a local spatial neighborhood to make the problem well-posed.

In the optical flow approach by Lucas and Kanade [19], a weighted least-squares fit to the local first-order constraints (2) is computed by minimizing the functional

$$\sum_{x \in W} W^2(x)[\nabla I(x,t) \cdot \mathbf{o} + I_t(x,t)]^2, \tag{3}$$

where $W(x)$ defines a Gaussian neighborhood around the current position x in the image plane. We use this technique in our system.

8.2. 3D motion fields

The optical flow observed by a camera is only a 2D-projection of the real world 3D motion field. The goal of motion capture is the recovery of the parameters of 3D motion. A reconstructed 3D motion field from optical flows in multiple camera views can be used to compute these parameters. The reconstruction of the 3D motion field, also know as the *scene flow*, from the 2D optical flows is possible using a technique described in [35].

If correspondences in the image plane are known, i.e., it is known to which image coordinates 3D points project in each camera view, the scene flow can be reconstructed by solving a linear system of equations. In our system, the correspondences are known for each vertex because we have an explicit body model and the projection matrices $\mathbf{P}_i$ for each recording camera i are known. The projection matrices describe the relationship between a 3D position of a vertex and its projection into the image plane of the camera, $\mathbf{u}_i = (u_i, v_i)^{\mathrm{T}}$.

The differential relationship between the vertex $\mathbf{x}$ with coordinates $(x, y, z)^{\mathrm{T}}$ and $\mathbf{u}_i$ is described by the 2×3 Jacobian matrix $J_i = \frac{\partial \mathbf{u}_i}{\partial \mathbf{x}_i}$

$$\frac{\mathrm{d}\mathbf{u}_i}{\mathrm{d}t} = J_i \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}. \tag{4}$$

In other words, the Jacobian describes the relationship between a small change in 3D position of a vertex, and the change of its projected image in camera i. The term $\frac{\mathrm{d}\mathbf{u}_i}{\mathrm{d}t}$ is the optical flow observed in camera i, $\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}$ is the corresponding scene flow of the vertex (Fig. 3). Having a mathematical camera model, the Jacobian can be computed analytically (see [35]).

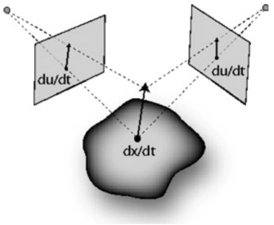

Fig. 3. 3D motion (scene flow) of a surface point and the corresponding observed optical flows in two camera views.

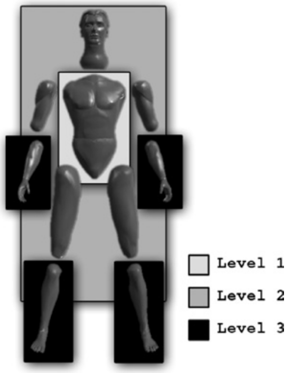

Fig. 4. Body model showing the separate hierarchy levels.

If a vertex is visible from at least two camera views, an equation system of the form $\mathbf{B}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{U}$ can be formulated to solve for the scene flow of this vertex given the optical flows in all camera views

$$\mathbf{B} = \begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} & \frac{\partial u_1}{\partial z} \\ \frac{\partial v_1}{\partial x} & \frac{\partial v_1}{\partial y} & \frac{\partial v_1}{\partial z} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \frac{\partial u_N}{\partial x} & \frac{\partial u_N}{\partial y} & \frac{\partial u_N}{\partial z} \\ \frac{\partial v_N}{\partial x} & \frac{\partial v_N}{\partial y} & \frac{\partial v_N}{\partial z} \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} \frac{\partial u_1}{\partial t} \\ \frac{\partial v_1}{\partial t} \\ \cdot \\ \cdot \\ \frac{\partial u_N}{\partial t} \\ \frac{\partial v_N}{\partial t} \end{bmatrix}, \tag{5}$$

where N is the number of camera views. A least-squares solution to this equation system can be found via singular value decomposition (SVD) [25].

9. Body pose update using 3D motion fields

Many assumptions in optical flow algorithms, such as the brightness constancy assumption and the assumption that the visibility does not change in subsequent images, often break down if the motion in the scene is very fast. In rapidly moving scenes, illumination changes may cause strong differences in the appearance of identical surface elements in two subsequent video frames, and occlusions or disocclusions of parts of a scene are very likely to occur. We concede that a purely motion-field-based tracking system is suitable for a slowly moving subject only. However, by combining optical flow and silhouette information, it becomes possible to bypass some of the limitations of optical flow reconstruction and capture complex, fast motions of the body. A motion field describes the motion of a scene between two time instants. In contrast, our *corrective motion field* describes the corrective motion between a first pose estimate at one time instant obtained via silhouette-based tracking and the correct pose for the same time instant if texture information is taken into account. These motions are small translations and rotations which properly align the model with the input video footage.

We apply the previously described scene flow reconstruction algorithm to compute a corrective motion field at each time step. Let $I_{j,t}$ be the jth input camera view at time t, and P_t be the model pose at time t. The algorithm then proceeds as follows:

- With P_t as the starting point, use silhouette fitting to compute $P'_{\mathrm{sil},t+1}$, which is an estimated pose for time $t+1$.
- Generate $I'_{j,t+1}$ by rendering the model from camera j in pose $P'_{\mathrm{sil},t+1}$ with texture from time t.
- Computation of corrective motion field D: For each model vertex
  - Determine the projection of the vertex into each camera's image plane.
  - Determine vertex visibility in all cameras by comparing the projected z-coordinate to the OpenGL z-buffer value.
  - If a vertex is visible from camera j, compute the optical flow between images $I'_{j,t+1}$ and $I_{j,t+1}$.
  - If a vertex is visible in at least three camera views (more robust reconstruction than with minimum number of two views), compute a least squares solution to an equation system of the form as in (5) by applying an SVD as described in Section 8.2.
- Update $P'_{\mathrm{sil},t+1}$ to conform with motion field to yield P_{t+1}.

The computed corrective 3D motion field D describes vertex position updates that correct slight inaccuracies in the result of the silhouette step. Fig. 6 shows an example of a corrective flow field and the corresponding updated body pose. The remainder of the section describes the derivation of the differential pose updates from $P'_{\mathrm{sil},t+1}$ to P_{t+1} using D.

9.1. Differential pose update

The corrective motion field D can be used to compute differential pose parameter updates for each limb of the body model. For the root, which is located in the torso segment, three differential rotation and three differential translation parameters are computed. All other joints are purely rotational. This includes 3-DOF rotations for the shoulders, hips, and neck, and a 1-DOF rotation for the elbows and knees. Wrist and ankle joints are currently not considered.

By adding each vector in D to the current 3D position of its corresponding vertex, a set of goal positions is defined for each model vertex. The goal is to find the set of differential joint parameters of the body model that best aligns the vertices with these positions. The idea is to compute the differential pose parameter updates for every joint only from the goal positions of the vertices of the attached body segment, e.g., using the upper arm goal positions to find the shoulder parameters.

Both our artificial body model and the real human body are hierarchical kinematic chains. This implies that transformations of joints lower in the hierarchy involve all transformation of preceding joints, too. Taking this into account, we solve for the differential model parameters for one hierarchy level of the model at a time, proceeding from top to bottom (level 1 being the highest level, see Fig. 4). After the pose updates for a higher level are found, the model parameters on this level are updated, leaving all lower levels unchanged. The algorithm proceeds to the next lower level. This method assures that the computed differential update corresponds only to a joint transformation on each level.

9.1.1. Registration method for pose update

Finding a pose update for a joint corresponds to finding a coordinate system transformation between two point sets, a problem know as the *absolute orientation problem* in photogrammetry [14]. For each joint, one point set consists of the current 3D vertex positions of the attached body segment. The second point set defines the goal locations for each vertex in 3D space.

Horn [15] describes a closed form solution to the absolute orientation problem, henceforth referred to as the registration method. In his work, Horn uses quaternions to parameterize rotations. All transformations are computed with respect to the centers of gravity of both point sets. Let $x_{1,i}$ and $x_{2,i}$, $i = \{1, \dots, N\}$ be corresponding points from two point sets, then the solution to the absolute orientation problem in the least-squares sense are the rotation R and translation c that minimize the error function

$$\sum_{i}^{N} \|x_{2,i} - Rx_{1,i} - c\|^2. \tag{6}$$

It is shown that the optimal translation c is defined by the difference between the centroid of set 2 and the rotated centroid of set 1. To find the optimal rotation, the coordinates of the points in both point sets are defined relative to their center of gravity, respectively. It can be shown that the optimal rotation in the sense of (6) can be found by maximizing

$$\sum_{i}^{N} x_{2,i} \cdot Rx_{1,i}. \tag{7}$$

The maximal solution to (7) can efficiently be computed in closed-form using a quaternion parameterization q of the rotation. A quaternion can be regarded as a complex number with one real component and three imaginary components, $q = q_0 + q_x i_x + q_y i_y + q_z i_z$, and can be represented by a 4-component vector. Rotations can be represented by unit quaternions. A detailed description of quaternions is beyond the scope of this paper, hence we refer the reader to the paper by Horn [15].

Using quaternions, the sum (7) can be transformed into the form

$$q^{\mathrm{T}} N q. \tag{8}$$

The matrix N contains entries that are only made up of products of coordinates of corresponding points in the two point sets that need to be registered (see Appendix).

The rotation q that maximizes this sum is the eigenvector that corresponds to the largest eigenvalue of the symmetric 4×4-matrix N. The solution q is a unit vector in the same direction as the eigenvector.

We apply the registration method to compute differential pose updates as follows. The adjustment starts at hierarchy level 1 with the root of the model. To find the corrective model update of the root joint, a differential rotation and translation is computed using the torso segment start and destination positions computed from D. The rotation component is computed by applying the previously described registration method. The corrective translation is simply the optimal translation of the registration method transformed into the global coordinate system.

On the second level of the hierarchy, only differential rotation parameters for 3-DOF shoulder, hip, and head joints need to be computed. The rotations are to be performed around the center of each joint, not around the center of gravity of the vertex positions. However, it is valid to simply use the start and goal vertex coordinates, $x_{1,i}$ and $x_{2,i}$, defined with respect to the local joint coordinate system instead of relative to the centers of gravity. The same algorithm that is part of the registration method applies for finding the optimal rotation. The least-squares rotation for the joint is found as the rotation R that minimizes

$$\sum_{i}^{N} \|x_{2,i} - Rx_{1,i}\|^2. \tag{9}$$

This energy term can be expanded into

$$\sum_{i=1}^{N} \|x_{2,i}\|^2 - 2\sum_{i=1}^{N} x_{2,i} \cdot Rx_{1,i} + \sum_{i=1}^{N} \|x_{1,i}\|^2, \tag{10}$$

which is minimized by maximizing the middle sum. This sum can be maximized by the same quaternion-based eigenvector decomposition method as previously described.

On hierarchy level 3, there are four 1-DOF joints (the elbows and the knees). The body model is designed in such a way that the rotation axis for each of these joints coincides with the X-axis of the local coordinate system. The optimal rotations are found using the same procedure as on hierarchy level 2. The 1-DOF constraint is incorporated by projecting the start and goal vertex positions into the local yz-planes.

In Fig. 5 the different steps of the pose parameter update computation are illustrated using an exaggerated flow field for better visualization.

10. Results and validation

The performance of our system was tested on two multi-view video sequences that were recorded with 8 cameras at a resolution of 320×240 pixels. The sequences show simple gestures that exhibit a large amount of head motion which cannot be accurately recovered using only the silhouette step. All test PCs feature 1.8 GHz Pentium IV Xeon CPUs with 512 MB of main memory and are equipped with Nvidia

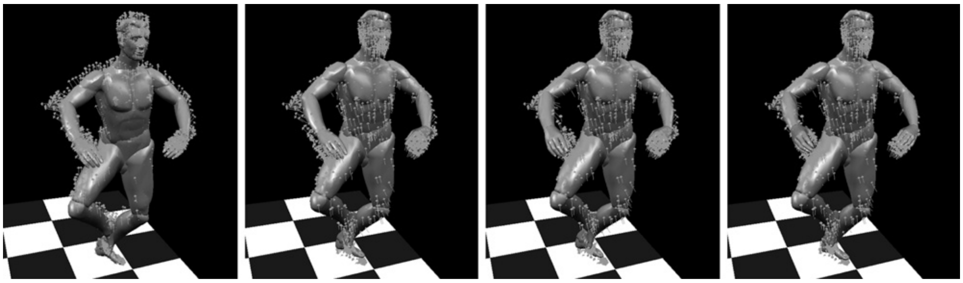

Fig. 5. The pictures from left to right show the silhouette-only model pose with 3D motion field (little arrows), the model after correction on the first hierarchy level, the second and then the third level. The lengths of the motion field vectors are exaggerated.

GeForce3 GPUs. For the different sub-components of the motion capture algorithm we obtained the following timing results:

For both sequences, the silhouette fitting takes between 3 s and 5 s for each time step if the implementation on a single PC is used. If the parallel implementation with five PCs is applied, silhouette fitting times significantly below one second are achieved for a single time step.

The most time-consuming step in the motion field reconstruction is the computation of the optical flows in all camera views. The Lucas–Kanade optical flow algorithm takes, on average, 45 s for the processing of one set of 8 input views if four levels of an image pyramid and a 20×20 Gaussian window are used. These numbers apply if the algorithm is configured to compute scene flow vectors for each model vertex, and thus 8 optical flow vectors are computed for each vertex. The runtime of the optical flow computation strongly depends on the chosen parameters. Speed-ups are gained by reducing the number of image pyramid levels and the size of the Gaussian neighborhood. For only one level in the pyramid and a 10×10-neighborhood, the optical flows in 8 camera views can be computed in 8 s.

A further acceleration is achieved by computing the scene flows only for a subset of the model vertices. However, since our focus lies on producing the maximally possible visual quality, we run the scene flow computation at the highest level of detail. Our system is flexible enough to incorporate any other optical flow method. The reconstruction of the 3D motion field from the 2D optical flows, takes on average 0.34 s for each time step.

The results obtained with both sequences show that the motion field update step can noticeably improve the quality of the reconstructed motion and thus also the reconstructed 3D video. Two pairs of images in Fig. 6 show the textured and untextured body model side by side. The left pair shows the result that is obtained with pure silhouette-based motion capture, the right pair shows the result with the enhanced algorithm. It is obvious that the improved visual quality of the textured model, notably in the face, is caused by the more accurate body pose.

In Fig. 6 some more screen-shots of 3D videos are depicted that were generated with and without motion field correction. As expected the most obvious improve-

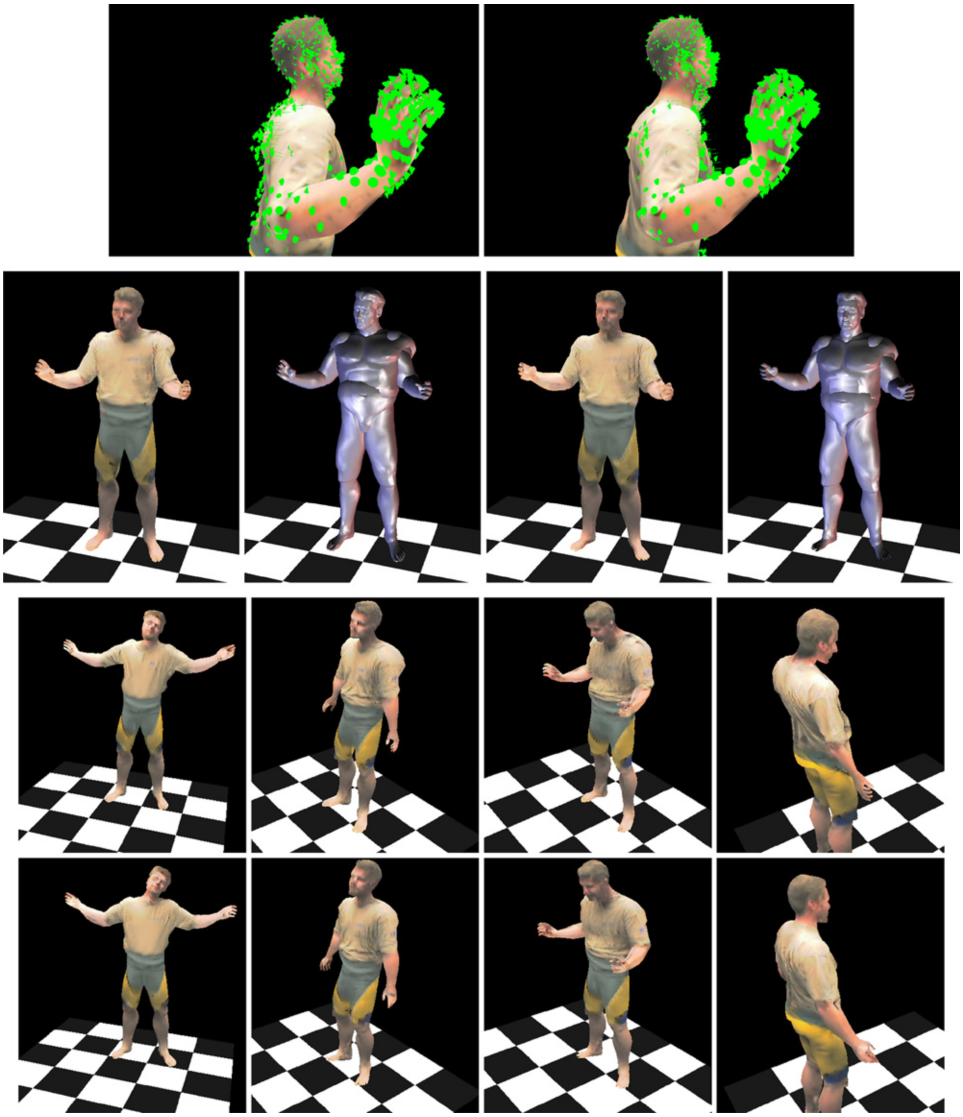

Fig. 6. First row: body model with corrective motion field (green arrows) before (l) and after (r) pose update. Second row: the left image pair shows the textured and untextured body model obtained using only silhouette fitting. The right pair shows the corresponding result with active motion field correction. Third and fourth row: the top images show screen shots of 3D videos reconstructed without motion field correction, the bottom images show visual improvements with active differential pose update.

ments are visible in the face and on the torso. The silhouette step often cannot exactly recover the head orientation. The additional use of the texture information can correct for such small errors. Slight changes in torso orientation are also discovered more robustly if the motion field correction step is applied.

Table 1
Differences in PSNR measurements between free-viewpoint videos that are reconstructed with and without motion field step

	Difference in avg. (dB)	Max. difference (dB)
Sequence 1	0.33	0.81
Sequence 2	0.35	0.93

In order to validate the visual improvements caused by the motion field step we employ a quality measure widely used in research on video encoding. For each time step of video we compute the peak signal-to-noise-ratio (PSNR) [3] in the luminance channel between the 3D video rendered from the input camera perspectives and the segmented recorded input views. On both test sequences, the PSNR is computed for the 3D videos with and without the corrective motion field step.

The difference in the average PSNR between the corrected and uncorrected free-viewpoint videos as well as the maximal observed difference for one single time step of video are summarized in Table 1.

The difference in the average PSNR over all video frames is a measure of reconstruction quality. A positive difference characterizes an improvement of rendering quality with respect to the original video frames. We obtained positive differences between the average PSNRs for both sequences. For one single time step of video the improvements can even be more significant as it is expressed in the values for the maximal observed PSNR difference.

It is interesting to observe that, after only small differences at the beginning, later, in both sequences, the PSNR differences are larger. This confirms the assumption that the correction step improves model fitting over time. Result movies can be downloaded from http://www.mpi-sb.mpg.de/~theobalt/SceneFlowFitting/index.html.

11. Conclusions and future work

In this work, we have presented a new approach for acquisition and rendering of 3D videos of human actors. We have demonstrated that that a large range of complex and rapid body motion can be robustly acquired by means of a silhouette-based marker-less motion capture algorithm. In combination with a renderer that applies a multi-view texture generation method to create realistic surface appearance, a powerful tool for immersive video production is proposed. We have demonstrated that the visual quality of the free-viewpoint videos can be further enhanced by incorporating texture information into the motion tracking process. It was shown empirically and quantitatively that the incorporation of corrective motion fields into the fitting process yields significant improvements, and thus, is a worthwhile enhancement to the process of capturing free-viewpoint video.

In the future, we plan to enhance our approach into a system that can simultaneously reconstruct human actors in the foreground of a scene as well as derive the geometry of the scene background. The concurrent acquisition of motion data and parametric models of surface appearance for relighting is also an issue. A body model estimation procedure similar to the one presented in [27] in order to estimate a skin deformation model from images may also be a worthwhile enhancement.

Acknowledgments

We thank Harald Krytinar and Anna Hagermark from the ballet ensemble of the Staatstheater Saarbrücken for performing in our video room. The authors would also like to express their gratitude to Ming Li from the MPI Informatik for his assistance during multi-view video recording.

Appendix. Structure of matrix *N*

The matrix N needed to compute the optimal rotation in a joint is defined as follows: Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two point sets of size n, each point defined via coordinates (x, y, and z), then a matrix M is defined as follows:

$$M = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}$$

with

$$S_{xy} = \sum_{i=1}^{n} x_{1,i} y_{2,i}.$$

The entries in N are built via arithmetic operations on elements of M:

$$N = [N_1 \quad N_2 \quad N_3 \quad N_4]$$

$$N_1 = \begin{bmatrix} (S_{xx} + S_{yy} + S_{zz}) \\ S_{yz} - S_{zy} \\ S_{zx} - S_{xz} \\ S_{xy} - S_{yx} \end{bmatrix}, \quad N_2 = \begin{bmatrix} S_{yz} - S_{zy} \\ (S_{xx} + S_{yy} + S_{zz}) \\ S_{xy} + S_{yx} \\ S_{zx} + S_{xz} \end{bmatrix},$$

$$N_3 = \begin{bmatrix} S_{zx} - S_{xz} \\ S_{xy} + S_{yx} \\ (-S_{xx} + S_{yy} - S_{zz}) \\ S_{yz} + S_{zy} \end{bmatrix}, \quad N_4 = \begin{bmatrix} S_{xy} - S_{yx} \\ S_{zx} + S_{xz} \\ S_{yz} + S_{zy} \\ (-S_{xx} - S_{yy} + S_{zz}) \end{bmatrix}.$$

References

[1] E.H. Adelson, J.R. Bergen, The plenoptic function and the elements of early vision, in: M. Landy, J.A. Movshon, Computational Models of Visual Processing, 1991.

[2] J. Barron, D. Fleet, S. Beauchemin, Performance of optical flow techniques, IJCV 12 (1) (1994) 43–77.

[3] V. Bhaskaran, K. Konstantinidis, Image and Video Compression Standards, Kluwer, Dordrecht, 1999.

[4] C. Bregler, J. Malik, Tracking people with twists and exponential maps, in: Proceedings of CVPR 98, 1998, pp. 8–15.

[5] J. Carranza, C. Theobalt, M. Magnor, H.-P. Seidel, Free-viewpoint video of human actors, in: Proceedings of ACM SIGGRAPH2003, San Diego, USA, 2003, pp. 569–577.

[6] K. Cheung, T. Kanade, J.-Y. Bouguet, M. Holler, A real time system for robust 3D voxel reconstruction of human motions, in: Proceedings of CVPR, vol. 2, 2000, pp. 714–720.

[7] Q. Delamarre, O. Faugeras, 3D articulated models and multi-view tracking with silhouettes, in: Proceedings of ICCV 99, 1999, pp. 716–721.

[8] P. Eisert, W.T. and, B. Girod, Model-aided coding: a new approach to incorporate facial animation into motion-compensated video coding, Trans. Circ. Syst. Video Technol. 10 (3) (2000) 244–258.

[9] D. Gavrila, The visual analysis of human movement, CVIU 73 (1) (1999) 82–98.

[10] D. Gavrila, L. Davis, 3D model-based tracking of humans in action: a multi-view approach, in: CVPR 96, 1996, pp. 73–80.

[11] S. Gortler, R. Grzeszczuk, R. Szeliski, M. Cohen, The lumigraph, in: Proceedings of ACM SIGGRAPH, 1996, pp. 43–54.

[12] Y. Guo, G. Xu, S. Tsuji, Tracking human body motion based on a stick-figure model, J. Vis. Commun. Image Rep. 5 (1) (1994) 1–9.

[13] D. Hogg, Model-based vision: a program to see a walking person, Image Vis. Comput. 1 (1) (1983) 5–20.

[14] B. Horn, Robot Vision, MIT Press, Cambridge, 1986.

[15] B. Horn, Closed-form solution of absolute orientation using unit quaternions, J. Opt. Soc. Am. 4 (4) (1987) 629–642.

[16] Y. Kameda, M. Minoh, K. Ikeda, Three dimensional motion estimation of a human body using a difference image sequence, in: Proceedings of the Asian Conference On Computer Vision '95, 1995, pp. 181–185.

[17] R. Koch, Dynamic 3D scene analysis through synthesis feedback control, PAMI 15 (6) (1993) 556–568.

[18] M. Levoy, P. Hanrahan, Light field rendering, in: Proceedings of ACM SIGGRAPH, ACM, 1996, pp. 31–42.

[19] B. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision, in: Proceedings of the DARPA IU Workshop, 1981, pp. 121–130.

[20] J. Luck, D. Small, Real-time markerless motion tracking using linked kinematic chains, in: Proceedings of CVPRIP02, 2002.

[21] W. Matusik, C. Buehler, L. McMillan, Polyhedral visual hulls for real-time rendering, in: Proceedings of 12th Eurographics Workshop on Rendering, 2001, pp. 116–126.

[22] T.B. Moeslund, E. Granum, A survey of computer vision-based human motion capture, Comput. Vision Image Und.: CVIU 81 (3) (2001) 231–268.

[23] P. Narayanan, P. Rander, T. Kanade, Constructing virtual worlds using dense stereo, in: Proceedings of ICCV 98, 1998, pp. 3–10.

[24] R. Plaenkers, P. Fua, Tracking and modeling people in video sequences, CVIU 81 (3) (2001) 285–302.

[25] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, Numerical Recipes, Cambridge University Press, Cambridge, 1992.

[26] K. Rohr, Incremental recognition of pedestrians from image sequences, in: Proceedings of CVPR 93, 1993, pp. 8–13.

[27] P. Sand, L. McMillan, J. Popović, Continuous capture of skin deformation, in: Proceedings of ACM SIGGRPAH03, 2003, pp. 578–586.

[28] J. Starck, A. Hilton, Towards a 3D virtual studio for human appearance capture, in: Proceedings of Vision, Video and Graphics, 2003, pp. 17–24.

[29] C. Theobalt, J. Carranza, M. Magnor, H. Seidel, A parallel framework for silhouette-based human motion capture, in: Proceedings of Vision, Modeling and Visualization 2003, Munich, Germany, 2003, pp. 207–214.

[30] C. Theobalt, J. Carranza, M.A. Magnor, H.-P. Seidel, Enhancing silhouette-based human motion capture with 3D motion fields, in: Proceedings of Pacific Graphics 2003, IEEE Computer Society Press, Canmore, Canada, 2003, pp. 185–193.

[31] C. Theobalt, M. Li, M. Magnor, H.-P. Seidel, A flexible and versatile studio for synchronized multi-view video recording, in: Proceedings of Vision, Video and Graphics, 2003, pp. 9–16.

[32] C. Theobalt, M. Magnor, P. Schueler, H.-P. Seidel, Combining 2D feature tracking and volume reconstruction for online video-based human motion capture, in: Proceedings of Pacific Graphics 2002, 2002, pp. 96–103.

[33] R. Tsai, An efficient and accurate camera calibration technique for 3D machine vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'86), 1986, pp. 364–374.

[34] S. Vedula, S. Baker, T. Kanade, Spatio-temporal view interpolation, in: Proceedings of the 13th ACM Eurographics Workshop on Rendering, 2002, pp. 65–75.

[35] S. Vedula, S. Baker, P. Rander, R. Collins, T. Kanade, Three-dimensional scene flow, in: Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV-99) [7], pp. 722–729.

[36] S. Vedula, S. Baker, S. Seitz, T. Kanade, Shape and motion carving in 6D, in: Proceedings of CVPR, 2000, pp. 592–598.

[37] B. Wilburn, M. Smulski, H.-H.K. Lee, M. Horowitz, The light field video camera, in: Proceedings of Media Processors 2002, SPIE Electronic Imaging 2002, 2002.

[38] S. Wuermlin, E. Lamboray, O. Staadt, M. Gross, 3D video recorder, in: Proceedings of Pacific Graphics 2002, IEEE Computer Society Press, 2002, pp. 325–334.