Drag Your GAN: Interactive Point-based Manipulation on the Generative Image Manifold

Xingang Pan
xpan@mpi-inf.mpg.de
Max Planck Institute for Informatics
Germany
Saarbrücken Research Center for Visual Computing, Interaction and AI
Germany

Ayush Tewari
ayusht@mit.edu
MIT CSAIL
USA

Thomas Leimkühler
thomas.leimkuehler@mpi-inf.mpg.de
Max Planck Institute for Informatics
Germany

Lingjie Liu
lingjie.liu@seas.upenn.edu
Max Planck Institute for Informatics
Germany
University of Pennsylvania
USA

Abhimitra Meka
abhim@google.com
Google AR/VR
USA

Christian Theobalt
theobalt@mpi-inf.mpg.de
Max Planck Institute for Informatics
Germany
Saarbrücken Research Center for Visual Computing, Interaction and AI
Germany

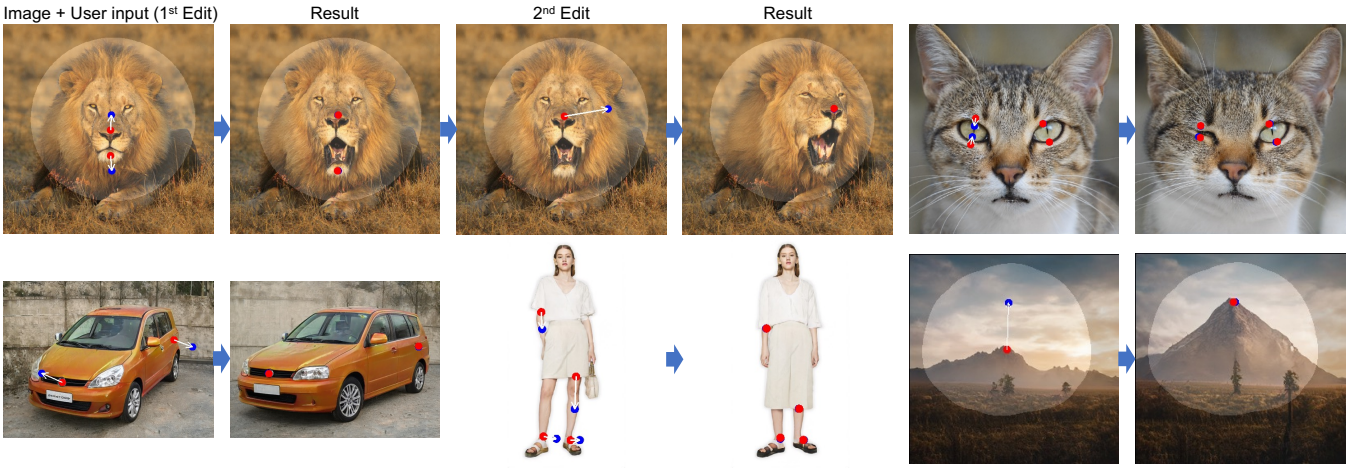

Figure 1: Our approach *DragGAN* allows users to "drag" the content of any GAN-generated images. Users only need to click a few handle points (red) and target points (blue) on the image, and our approach will move the handle points to precisely reach their corresponding target points. Users can optionally draw a mask of the flexible region (brighter area), keeping the rest of the image fixed. This flexible point-based manipulation enables control of many spatial attributes like pose, shape, expression, and layout across diverse object categories.

ABSTRACT

Synthesizing visual content that meets users' needs often requires flexible and precise controllability of the pose, shape, expression, and layout of the generated objects. Existing approaches gain controllability of generative adversarial networks (GANs) via manually annotated training data or a prior 3D model, which often lack flexibility, precision, and generality. In this work, we study a powerful yet much less explored way of controlling GANs, that is, to "drag" any points of the image to precisely reach target points in a user-interactive manner, as shown in Fig.1. To achieve this, we propose *DragGAN*, which consists of two main components: 1) a feature-based motion supervision that drives the handle point to move towards the target position, and 2) a new point tracking approach that leverages the discriminative generator features to keep localizing the position of the handle points. Through *DragGAN*, anyone

SIGGRAPH '23 Conference Proceedings, August 06–10, 2023, Los Angeles, CA, USA

ACM ISBN 979-8-4007-0159-7/23/08.
https://doi.org/10.1145/3588432.3591500

can deform an image with precise control over where pixels go, thus manipulating the pose, shape, expression, and layout of diverse categories such as animals, cars, humans, landscapes, *etc.* As these manipulations are performed on the learned generative image manifold of a GAN, they tend to produce realistic outputs even for challenging scenarios such as hallucinating occluded content and deforming shapes that consistently follow the object's rigidity. Both qualitative and quantitative comparisons demonstrate the advantage of *DragGAN* over prior approaches in the tasks of image manipulation and point tracking. We also showcase the manipulation of real images through GAN inversion.

CCS CONCEPTS

• **Computing methodologies → Computer vision**.

KEYWORDS

GANs, interactive image manipulation, point tracking

ACM Reference Format:
Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. Drag Your GAN: Interactive Point-based Manipulation on the Generative Image Manifold. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings), August 06–10, 2023, Los Angeles, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3588432.3591500

1 INTRODUCTION

Deep generative models such as generative adversarial networks (GANs) [Goodfellow et al. 2014] have achieved unprecedented success in synthesizing random photorealistic images. In real-world applications, a critical functionality requirement of such learning-based image synthesis methods is the controllability over the synthesized visual content. For example, social-media users might want to adjust the position, shape, expression, and body pose of a human or animal in a casually-captured photo; professional movie pre-visualization and media editing may require efficiently creating sketches of scenes with certain layouts; and car designers may want to interactively modify the shape of their creations. To satisfy these diverse user requirements, an *ideal* controllable image synthesis approach should possess the following properties 1) *Flexibility*: it should be able to control different spatial attributes including position, pose, shape, expression, and layout of the generated objects or animals; 2) *Precision*: it should be able to control the spatial attributes with high precision; 3) *Generality*: it should be applicable to different object categories but not limited to a certain category. While previous works only satisfy one or two of these properties, we target to achieve them all in this work.

Most previous approaches gain controllability of GANs via prior 3D models [Deng et al. 2020; Ghosh et al. 2020; Tewari et al. 2020] or supervised learning that relies on manually annotated data [Abdal et al. 2021; Isola et al. 2017; Ling et al. 2021; Park et al. 2019; Shen et al. 2020]. Thus, these approaches fail to generalize to new object categories, often control a limited range of spatial attributes or provide little control over the editing process. Recently, text-guided image synthesis has attracted attention [Ramesh et al. 2022; Rombach et al. 2021; Saharia et al. 2022]. However, text guidance lacks precision and flexibility in terms of editing spatial attributes. For example, it cannot be used to move an object by a specific number of pixels.

To achieve flexible, precise, and generic controllability of GANs, in this work, we explore a powerful yet much less explored interactive point-based manipulation. Specifically, we allow users to click any number of handle points and target points on the image and the goal is to drive the handle points to reach their corresponding target points. As shown in Fig. 1, this point-based manipulation allows users to control diverse spatial attributes and is agnostic to object categories. The approach with the closest setting to ours is UserControllableLT [Endo 2022], which also studies dragging-based manipulation. Compared to it, the problem studied in this paper has two more challenges: 1) we consider the control of more than one point, which their approach does not handle well; 2) we require the handle points to precisely reach the target points while their approach does not. As we will show in experiments, handling more than one point with precise position control enables much more diverse and accurate image manipulation.

To achieve such interactive point-based manipulation, we propose *DragGAN*, which addresses two sub-problems, including 1) supervising the handle points to move towards the targets and 2) tracking the handle points so that their positions are known at each editing step. Our technique is built on the key insight that the feature space of a GAN is sufficiently discriminative to enable both motion supervision and precise point tracking. Specifically, the motion supervision is achieved via a shifted feature patch loss that optimizes the latent code. Each optimization step leads to the handle points shifting closer to the targets; thus point tracking is then performed through nearest neighbor search in the feature space. This optimization process is repeated until the handle points reach the targets. *DragGAN* also allows users to optionally draw a region of interest to perform region-specific editing. Since *DragGAN* does not rely on any additional networks like RAFT [Teed and Deng 2020], it achieves efficient manipulation, only taking a few seconds on a single RTX 3090 GPU in most cases. This allows for live, interactive editing sessions, in which the user can quickly iterate on different layouts till the desired output is achieved.

We conduct an extensive evaluation of *DragGAN* on diverse datasets including animals (lions, dogs, cats, and horses), humans (face and whole body), cars, and landscapes. As shown in Fig.1, our approach effectively moves the user-defined handle points to the target points, achieving diverse manipulation effects across many object categories. Unlike conventional shape deformation approaches that simply apply warping [Igarashi et al. 2005], our deformation is performed on the learned image manifold of a GAN, which tends to obey the underlying object structures. For example, our approach can hallucinate occluded content, like the teeth inside a lion's mouth, and can deform following the object's rigidity, like the bending of a horse leg. We also develop a GUI for users to interactively perform the manipulation by simply clicking on the image. Both qualitative and quantitative comparison confirms the advantage of our approach over UserControllableLT. Furthermore, our GAN-based point tracking algorithm also outperforms existing point tracking approaches such as RAFT [Teed and Deng 2020] and PIPs [Harley et al. 2022] for GAN-generated frames. Furthermore,

by combining with GAN inversion techniques, our approach also serves as a powerful tool for real image editing.

2 RELATED WORK

2.1 Generative Models for Interactive Content Creation

Most current methods use generative adversarial networks (GANs) or diffusion models for controllable image synthesis.

Unconditional GANs. GANs are generative models that transform low-dimensional randomly sampled latent vectors into photorealistic images. They are trained using adversarial learning and can be used to generate high-resolution photorealistic images [Creswell et al. 2018; Goodfellow et al. 2014; Karras et al. 2021, 2019]. Most GAN models like StyleGAN [Karras et al. 2019] do not directly enable controllable editing of the generated images.

Conditional GANs. Several methods have proposed conditional GANs to address this limitation. Here, the network receives a conditional input, such as segmentation map [Isola et al. 2017; Park et al. 2019] or 3D variables [Deng et al. 2020; Ghosh et al. 2020], in addition to the randomly sampled latent vector to generate photorealistic images. Instead of modeling the conditional distribution, EditGAN [Ling et al. 2021] enables editing by first modeling a joint distribution of images and segmentation maps, and then computing new images corresponding to edited segmentation maps.

Controllability using Unconditional GANs. Several methods have been proposed for editing unconditional GANs by manipulating the input latent vectors. Some approaches find meaningful latent directions via supervised learning from manual annotations or prior 3D models [Abdal et al. 2021; Leimkühler and Drettakis 2021; Patashnik et al. 2021; Shen et al. 2020; Tewari et al. 2020]. Other approaches compute the important semantic directions in the latent space in an unsupervised manner [Härkönen et al. 2020; Shen and Zhou 2020; Zhu et al. 2023]. Recently, the controllability of coarse object position is achieved by introducing intermediate "blobs" [Epstein et al. 2022] or heatmaps [Wang et al. 2022b]. All of these approaches enable editing of either image-aligned semantic attributes such as appearance, or coarse geometric attributes such as object position and pose. While Editing-in-Style [Collins et al. 2020] showcases some spatial attributes editing capability, it can only achieve this by transferring local semantics between different samples. In contrast to these methods, our approach allows users to perform fine-grained control over the spatial attributes using point-based editing.

GANWarping [Wang et al. 2022a] also use point-based editing, however, they only enable out-of-distribution image editing. A few warped images can be used to update the generative model such that all generated images demonstrate similar warps. However, this method does not ensure that the warps lead to realistic images. Further, it does not enable controls such as changing the 3D pose of the object. Similar to us, UserControllableLT [Endo 2022] enables point-based editing by transforming latent vectors of a GAN. However, this approach only supports editing using a single point being dragged on the image and does not handle multiple-point constraints well. In addition, the control is not precise, *i.e.*, after editing, the target point is often not reached.

3D-aware GANs. Several methods modify the architecture of the GAN to enable 3D control [Chan et al. 2022, 2021; Chen et al. 2022; Gu et al. 2022; Schwarz et al. 2020]. Here, the model generates 3D representations that can be rendered using a physically-based analytic renderer. However, unlike our approach, control is limited to global pose or lighting.

Diffusion Models. More recently, diffusion models [Sohl-Dickstein et al. 2015] have enabled image synthesis at high quality [Ho et al. 2020; Song et al. 2020, 2021]. These models iteratively denoise a randomly sampled noise to create a photorealistic image. Recent models have shown expressive image synthesis conditioned on text inputs [Ramesh et al. 2022; Rombach et al. 2021; Saharia et al. 2022]. However, natural language does not enable fine-grained control over the spatial attributes of images, and thus, all text-conditional methods are restricted to high-level semantic editing. In addition, current diffusion models are slow since they require multiple denoising steps. While progress has been made toward efficient sampling, GANs are still significantly more efficient.

2.2 Point Tracking

To track points in videos, an obvious approach is through optical flow estimation between consecutive frames. Optical flow estimation is a classic problem that estimates motion fields between two images. Conventional approaches solve optimization problems with hand-crafted criteria [Brox and Malik 2010; Sundaram et al. 2010], while deep learning-based approaches started to dominate the field in recent years due to better performance [Dosovitskiy et al. 2015; Ilg et al. 2017; Teed and Deng 2020]. These deep learning-based approaches typically use synthetic data with ground truth optical flow to train the deep neural networks. Among them, the most widely used method now is RAFT [Teed and Deng 2020], which estimates optical flow via an iterative algorithm. Recently, Harley *et al.* [2022] combines this iterative algorithm with a conventional "particle video" approach, giving rise to a new point tracking method named PIPs. PIPs considers information across multiple frames and thus handles long-range tracking better than previous approaches.

In this work, we show that point tracking on GAN-generated images can be performed without using any of the aforementioned approaches or additional neural networks. We reveal that the feature spaces of GANs are discriminative enough such that tracking can be achieved simply via feature matching. While some previous works also leverage the discriminative feature in semantic segmentation [Tritrong et al. 2021; Zhang et al. 2021], we are the first to connect the point-based editing problem to the intuition of discriminative GAN features and design a concrete method. Getting rid of additional tracking models allows our approach to run much more efficiently to support interactive editing. Despite the simplicity of our approach, we show that it outperforms the state-of-the-art point tracking approaches including RAFT and PIPs in our experiments.

3 METHOD

This work aims to develop an interactive image manipulation method for GANs where users only need to click on the images to define some pairs of (handle point, target point) and drive the handle points to reach their corresponding target points. Our study

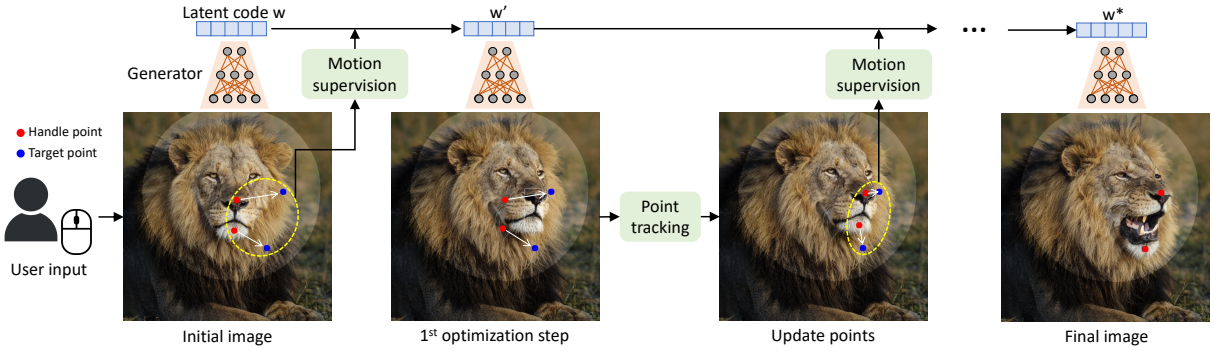

Figure 2: Overview of our pipeline. Given a GAN-generated image, the user only needs to set several handle points (red dots), target points (blue dots), and optionally a mask denoting the movable region during editing (brighter area). Our approach iteratively performs *motion supervision* (Sec. 3.2) and *point tracking* (Sec. 3.3). The motion supervision step drives the handle points (red dots) to move towards the target points (blue dots) and the point tracking step updates the handle points to track the object in the image. This process continues until the handle points reach their corresponding target points.

is based on the StyleGAN2 architecture [Karras et al. 2020]. Here we briefly introduce the basics of this architecture.

StyleGAN Terminology. In the StyleGAN2 architecture, a 512 dimensional latent code $\boldsymbol{z} \in \mathcal{N}(0, \boldsymbol{I})$ is mapped to an intermediate latent code $\boldsymbol{w} \in \mathbb{R}^{512}$ via a mapping network. The space of $\boldsymbol{w}$ is commonly referred to as $\mathcal{W}$. $\boldsymbol{w}$ is then sent to the generator G to produce the output image $\mathbf{I} = G(\boldsymbol{w})$. In this process, $\boldsymbol{w}$ is copied several times and sent to different layers of the generator G to control different levels of attributes. Alternatively, one can also use different $\boldsymbol{w}$ for different layers, in which case the input would be $\boldsymbol{w} \in \mathbb{R}^{l\times 512} = \mathcal{W}^+$, where l is the number of layers. This less constrained $\mathcal{W}^+$ space is shown to be more expressive [Abdal et al. 2019]. As the generator G learns a mapping from a low-dimensional latent space to a much higher dimensional image space, it can be seen as modelling an image manifold [Zhu et al. 2016].

3.1 Interactive Point-based Manipulation

An overview of our image manipulation pipeline is shown in Fig. 2. For any image $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$ generated by a GAN with latent code $\boldsymbol{w}$, we allow the user to input a number of handle points $\{\boldsymbol{p}_i = (x_{p,i}, y_{p,i}) | i = 1, 2, ..., n\}$ and their corresponding target points $\{\boldsymbol{t}_i = (x_{t,i}, y_{t,i}) | i = 1, 2, ..., n\}$ (*i.e.*, the corresponding target point of $\boldsymbol{p}_i$ is $\boldsymbol{t}_i$). The goal is to move the object in the image such that the semantic positions (*e.g.*, the nose and the jaw in Fig. 2) of the handle points reach their corresponding target points. We also allow the user to optionally draw a binary mask $\mathbf{M}$ denoting which region of the image is movable.

Given these user inputs, we perform image manipulation in an optimization manner. As shown in Fig. 2, each optimization step consists of two sub-steps, including 1) *motion supervision* and 2) *point tracking*. In motion supervision, a loss that enforces handle points to move towards target points is used to optimize the latent code $\boldsymbol{w}$. After one optimization step, we get a new latent code $\boldsymbol{w}'$ and a new image $\mathbf{I}'$. The update would cause a slight movement of the object in the image. Note that the motion supervision step only moves each handle point towards its target by a small step but the exact length of the step is unclear as it is subject to complex optimization dynamics and therefore varies for different objects and parts. Thus, we then update the positions of the handle points $\{\boldsymbol{p}_i\}$ to track the corresponding points on the object. This tracking process is necessary because if the handle points (*e.g.*, nose of the lion) are not accurately tracked, then in the next motion supervision step, wrong points (*e.g.*, face of the lion) will be supervised, leading to undesired results. After tracking, we repeat the above optimization step based on the new handle points and latent codes. This optimization process continues until the handle points $\{\boldsymbol{p}_i\}$ reach the position of the target points $\{\boldsymbol{t}_i\}$, which usually takes 30-200 iterations in our experiments. The user can also stop the optimization at any intermediate step. After editing, the user can input new handle and target points and continue editing until satisfied with the results.

3.2 Motion Supervision

How to supervise the point motion for a GAN-generated image has not been much explored before. In this work, we propose a motion supervision loss that does not rely on any additional neural networks. The key idea is that the intermediate features of the generator are very discriminative such that a simple loss suffices to supervise motion. Specifically, we consider the feature maps $\mathbf{F}$ after the 6th block of StyleGAN2, which performs the best among all features due to a good trade-off between resolution and discriminativeness. We resize $\mathbf{F}$ to have the same resolution as the final image via bilinear interpolation. As shown in Fig. 3, to move a handle point $\boldsymbol{p}_i$ to the target point $\boldsymbol{t}_i$, our idea is to supervise a small patch around $\boldsymbol{p}_i$ (red circle) to move towards $\boldsymbol{t}_i$ by a small step (blue circle). We use $\Omega_1(\boldsymbol{p}_i, r_1)$ to denote the pixels whose distance to $\boldsymbol{p}_i$ is less than r_1, then our motion supervision loss is:

$$\mathcal{L} = \sum_{i=0}^{n} \sum_{\boldsymbol{q}_i \in \Omega_1(\boldsymbol{p}_i, r_1)} \|\mathbf{F}(\boldsymbol{q}_i) - \mathbf{F}(\boldsymbol{q}_i + \boldsymbol{d}_i)\|_1 + \lambda \|(\mathbf{F} - \mathbf{F}_0) \cdot (1 - \mathbf{M})\|_1, \tag{1}$$

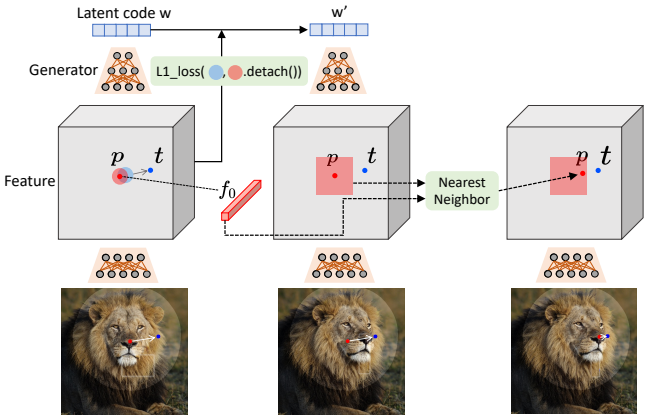

Figure 3: Method. Our motion supervision is achieved via a shifted patch loss on the feature maps of the generator. We perform point tracking on the same feature space via the nearest neighbor search.

where $\mathbf{F}(\boldsymbol{q})$ denotes the feature values of $\mathbf{F}$ at pixel $\boldsymbol{q}$, $\boldsymbol{d}_i = \frac{\boldsymbol{t}_i - \boldsymbol{p}_i}{\|\boldsymbol{t}_i - \boldsymbol{p}_i\|_2}$ is a normalized vector pointing from $\boldsymbol{p}_i$ to $\boldsymbol{t}_i$ ($\boldsymbol{d}_i = 0$ if $\boldsymbol{t}_i = \boldsymbol{p}_i$), and $\mathbf{F}_0$ is the feature maps corresponding to the initial image. Note that the first term is summed up over all handle points $\{\boldsymbol{p}_i\}$. As the components of $\boldsymbol{q}_i + \boldsymbol{d}_i$ are not integers, we obtain $\mathbf{F}(\boldsymbol{q}_i + \boldsymbol{d}_i)$ via bilinear interpolation. Importantly, when performing back-propagation using this loss, the gradient is not back-propagated through $\mathbf{F}(\boldsymbol{q}_i)$. This will motivate $\boldsymbol{p}_i$ to move to $\boldsymbol{p}_i + \boldsymbol{d}_i$ but not vice versa. In case the binary mask $\mathbf{M}$ is given, we keep the unmasked region fixed with a reconstruction loss shown as the second term. At each motion supervision step, this loss is used to optimize the latent code $\boldsymbol{w}$ for one step. $\boldsymbol{w}$ can be optimized either in the $\mathcal{W}$ space or in the $\mathcal{W}^+$ space, depending on whether the user wants a more constrained image manifold or not. As $\mathcal{W}^+$ space is easier to achieve out-of-distribution manipulations (*e.g.*, cat in Fig. 16), we use $\mathcal{W}^+$ in this work for better editability. In practice, we observe that the spatial attributes of the image are mainly affected by the $\boldsymbol{w}$ for the first 6 layers while the remaining ones only affect appearance. Thus, inspired by the style-mixing technique [Karras et al. 2019], we only update the $\boldsymbol{w}$ for the first 6 layers while fixing others to preserve the appearance. This selective optimization leads to the desired slight movement of image content.

3.3 Point Tracking

The previous motion supervision results in a new latent code $\boldsymbol{w}'$, new feature maps $\mathbf{F}'$, and a new image $\mathbf{I}'$. As the motion supervision step does not readily provide the precise new locations of the handle points, our goal here is to update each handle point $\boldsymbol{p}_i$ such that it tracks the corresponding point on the object. Point tracking is typically performed via optical flow estimation models or particle video approaches [Harley et al. 2022]. Again, these additional models can significantly harm efficiency and may suffer from accumulation error, especially in the presence of alias artifacts in GANs. We thus present a new point tracking approach for GANs. The insight is that the discriminative features of GANs well capture dense correspondence and thus tracking can be effectively performed via nearest neighbor search in a feature patch. Specifically, we denote the feature of the initial handle point as $\boldsymbol{f}_i = \mathbf{F}_0(\boldsymbol{p}_i)$. We denote the patch around $\boldsymbol{p}_i$ as $\Omega_2(\boldsymbol{p}_i, r_2) = \{(x, y) \mid |x - x_{p,i}| < r_2, |y - y_{p,i}| < r_2\}$. Then the tracked point is obtained by searching for the nearest neighbor of $\boldsymbol{f}_i$ in $\Omega_2(\boldsymbol{p}_i, r_2)$:

$$\boldsymbol{p}_i := \underset{\boldsymbol{q}_i \in \Omega_2(\boldsymbol{p}_i, r_2)}{\arg\min} \|\mathbf{F}'(\boldsymbol{q}_i) - \boldsymbol{f}_i\|_1. \tag{2}$$

In this way, $\boldsymbol{p}_i$ is updated to track the object. For more than one handle point, we apply the same process for each point. Note that here we are also considering the feature maps $\mathbf{F}'$ after the 6th block of StyleGAN2. The feature maps have a resolution of 256×256 and are bilinear interpolated to the same size as the image if needed, which is sufficient to perform accurate tracking in our experiments. We analyze this choice at Sec. 4.2.

3.4 Implementation Details

We implement our approach based on PyTorch [Paszke et al. 2017]. We use the Adam optimizer [Kingma and Ba 2014] to optimize the latent code $\boldsymbol{w}$ with a step size of 2e-3 for FFHQ [Karras et al. 2019], AFHQCat [Choi et al. 2020], and LSUN Car [Yu et al. 2015] datasets and 1e-3 for others. The hyper-parameters are set to be $\lambda = 20, r_1 = 3, r_2 = 12$. In our implementation, we stop the optimization process when all the handle points are no more than d pixel away from their corresponding target points, where d is set to 1 for no more than 5 handle points and 2 otherwise. We also develop a GUI to support interactive image manipulation. Thanks to the computational efficiency of our approach, users only need to wait for a few seconds for each edit and can continue the editing until satisfied. We highly recommend readers refer to the supplemental video for live recordings of interactive sessions.

4 EXPERIMENTS

Datasets. We evaluate our approach based on StyleGAN2 [Karras et al. 2020] pretrained on the following datasets (the resolution of the pretrained StyleGAN2 is shown in brackets): FFHQ (512) [Karras et al. 2019], AFHQCat (512) [Choi et al. 2020], SHHQ (512) [Fu et al. 2022], LSUN Car (512) [Yu et al. 2015], LSUN Cat (256) [Yu et al. 2015], Landscapes HQ (256) [Skorokhodov et al. 2021], microscope (512) [Pinkney 2020] and self-distilled dataset from [Mokady et al. 2022] including Lion (512), Dog (1024), and Elephant (512).

Baselines. Our main baseline is UserControllableLT [Endo 2022], which has the closest setting with our method. UserControllableLT does not support a mask input but allows users to define a number of fixed points. Thus, for testing cases with a mask input, we sample a regular 16×16 grid on the image and use the points outside the mask as the fixed points to UserControllableLT. Besides, we also compare with RAFT [Teed and Deng 2020] and PIPs [Harley et al. 2022] for point tracking. To do so, we create two variants of our approach where the point tracking part (Sec.3.3) is replaced with these two tracking methods.

4.1 Qualitative Evaluation

Fig. 4 shows the qualitative comparison between our method and UserControllableLT. We show the image manipulation results for

Figure 4: Qualitative comparison of our approach to UserControllableLT [Endo 2022] on the task of moving handle points (red dots) to target points (blue dots). Our approach achieves more natural and superior results on various datasets. More examples are provided in Fig. 10.

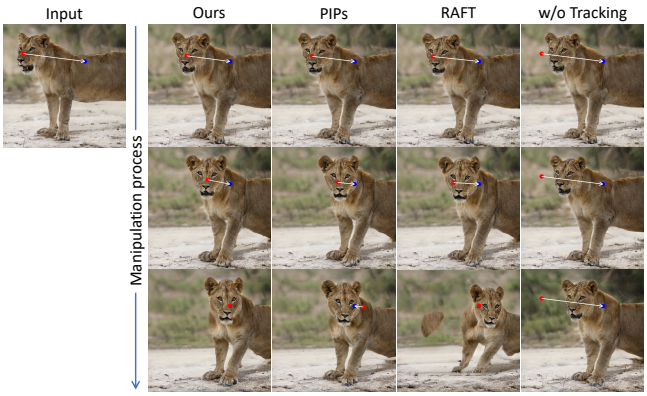

Figure 5: Qualitative tracking comparison of our approach to RAFT [Teed and Deng 2020], PIPs [Harley et al. 2022], and without tracking. Our approach tracks the handle point more accurately than baselines, thus producing more precise editing.

several different object categories and user inputs. Our approach accurately moves the handle points to reach the target points, achieving diverse and natural manipulation effects such as changing the pose of animals, the shape of a car, and the layout of a landscape. In contrast, UserControllableLT cannot faithfully move the handle points to the targets and often leads to undesired changes in the images, *e.g.*, the clothes of the human and the background of the car. It also does not keep the unmasked region fixed as well as ours, as shown in the cat images. We show more comparisons in Fig. 10.

A comparison between our approach with PIPs and RAFT is provided in Fig. 5. Our approach accurately tracks the handle point above the nose of the lion, thus successfully driving it to the target position. In PIPs and RAFT, the tracked point starts to deviate from the nose during the manipulation process. Consequently, they move the wrong part to the target position. When no tracking is performed, the fixed handle point soon starts to drive another part of the image (*e.g.*, background) after a few steps and never knows when to stop, which fails to achieve the editing goal.

Real image editing. Using GAN inversion techniques that embed a real image in the latent space of StyleGAN, we can also apply our approach to manipulate real images. Fig. 6 shows an example, where we apply PTI inversion [Roich et al. 2022] to the real image and then perform a series of manipulations to edit the pose, hair, shape, and expression of the face in the image. We show another real image editing example in Fig. 13.

4.2 Quantitative Evaluation

We quantitatively evaluate our method under two settings, including face landmark manipulation and paired image reconstruction.

Face landmark manipulation. Since face landmark detection is very reliable using an off-the-shelf tool [King 2009], we use its prediction as ground truth landmarks. Specifically, we randomly generate two face images using the StyleGAN trained on FFHQ and detect their landmarks. The goal is to manipulate the landmarks of the first image to match the landmarks of the second image. After manipulation, we detect the landmarks of the final image and compute the mean distance (MD) to the target landmarks. The results are averaged over 1000 tests. The same set of test samples

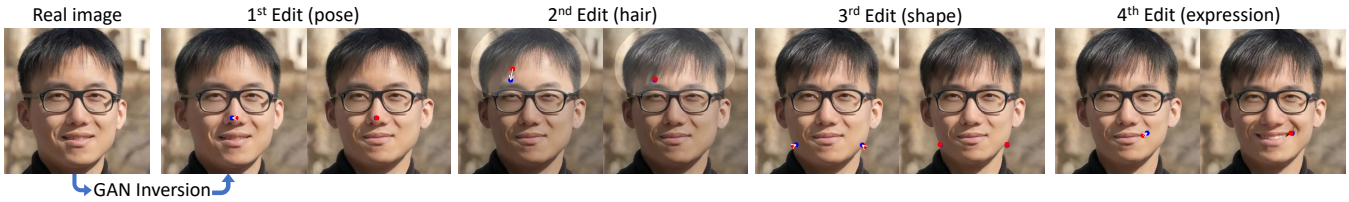

Figure 6: Real image manipulation. Given a real image, we apply GAN inversion to map it to the latent space of StyleGAN, then edit the pose, hair, shape, and expression, respectively.

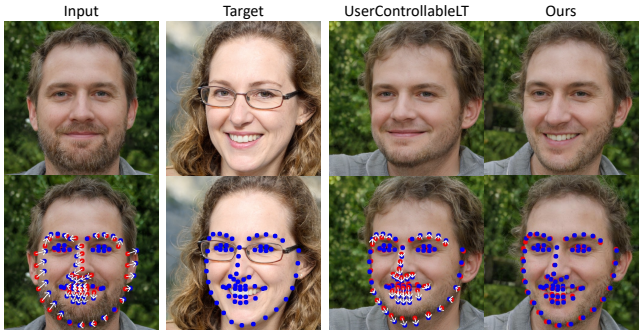

Figure 7: Face landmark manipulation. Compared to UserControllableLT [Endo 2022], our method can manipulate the landmarks detected from the input image to match the landmarks detected from the target image with less matching error.

is used to evaluate all methods. In this way, the final MD score reflects how well the method can move the landmarks to the target positions. We perform the evaluation under 3 settings with different numbers of landmarks including 1, 5, and 68 to show the robustness of our approach under different numbers of handle points. We also report the FID score between the edited images and the initial images as an indication of image quality. In our approach and its variants, the maximum optimization step is set to 300.

The results are provided in Table 1. Our approach significantly outperforms UserControllableLT under different numbers of points. A qualitative comparison is shown in Fig. 7, where our method opens the mouth and adjusts the shape of the jaw to match the target face while UserControllableLT fails to do so. Furthermore, our approach preserves better image quality as indicated by the FID scores. Thanks to a better tracking capability, we also achieve more accurate manipulation than RAFT and PIPs. Inaccurate tracking also leads to excessive manipulation, which deteriorates the image quality as shown in FID scores. Although UserControllableLT is faster, our approach largely pushes the upper bound of this task, achieving much more faithful manipulation while maintaining a comfortable running time for users.

Paired image reconstruction. In this evaluation, we follow the same setting as UserControllableLT [Endo 2022]. Specifically, we sample a latent code $\boldsymbol{w}_1$ and randomly perturb it to get $\boldsymbol{w}_2$ in the same way as in [Endo 2022]. Let $\mathbf{I}_1$ and $\mathbf{I}_2$ be the StyleGAN images generated from the two latent codes. We then compute the optical

Table 1: Quantitative evaluation on face keypoint manipulation. We compute the mean distance between edited points and target points. The FID and Time are reported based on the '1 point' setting.

Method	1 point	5 points	68 points	FID	Time (s)
No edit	12.93	11.66	16.02	-	-
UserControllableLT	11.64	10.41	10.15	25.32	0.03
Ours w. RAFT tracking	13.43	13.59	15.92	51.37	15.4
Ours w. PIPs tracking	2.98	4.83	5.30	31.87	6.6
Ours	**2.44**	**3.18**	**4.73**	**9.28**	2.0

Figure 8: Effects of the mask. Our approach allows masking the movable region. After masking the head region of the dog, the rest part would be almost unchanged.

flow between $\mathbf{I}_1$ and $\mathbf{I}_2$ and randomly sample 32 pixels from the flow field as the user input $\mathcal{U}$. The goal is to reconstruct $\mathbf{I}_2$ from $\mathbf{I}_1$ and $\mathcal{U}$. We report MSE and LPIPS [Zhang et al. 2018] and average the results over 1000 samples. The maximum optimization step is set to 100 in our approach and its variants. As shown in Table 2, our approach outperforms all the baselines in different object categories, which is consistent with previous results.

Ablation Study. Here we study the effects of which feature to use in motion supervision and point tracking. We report the performance (MD) of face landmark manipulation using different features. As Table 3 shows, in both motion supervision and point tracking, the feature maps after the 6th block of StyleGAN perform the best, showing the best balance between resolution and discriminativeness. We also provide the effects of r_1 in Table 4. It can be observed that the performance is not very sensitive to the choice of r_1, and $r_1 = 3$ performs slightly better.

4.3 Discussions

Effects of mask. Our approach allows users to input a binary mask denoting the movable region. We show its effects in Fig. 8. When a mask over the head of the dog is given, the other regions are almost fixed and only the head moves. Without the mask, the

Table 2: Quantitative evaluation on paired image reconstruction. We follow the evaluation in [Endo 2022] and report MSE $(\times 10^2)\downarrow$ and LPIPS $(\times 10)\downarrow$ scores.

Dataset	Lion		LSUN Cat		Dog		LSUN Car	
Metric	MSE	LPIPS	MSE	LPIPS	MSE	LPIPS	MSE	LPIPS
UserControllableLT	1.82	1.14	1.25	0.87	1.23	0.92	1.98	0.85
Ours w. RAFT tracking	1.09	0.99	1.84	1.15	0.91	0.76	2.37	0.94
Ours w. PIPs tracking	0.80	0.82	1.11	0.85	0.78	0.63	1.81	0.79
Ours	**0.66**	**0.72**	**1.04**	**0.82**	**0.48**	**0.44**	**1.67**	**0.74**

Table 3: Effects of which feature to use. x+y means the concatenation of two features. We report the performance (MD) of face landmark manipulation (1 point).

Block No.	4	5	6	7	5+6	6+7
Motion sup.	2.73	2.50	**2.44**	2.51	2.47	2.45
Tracking	3.61	2.55	**2.44**	2.58	2.47	2.45

Table 4: Effects of r_1.

r_1	1	2	3	4	5
MD	2.49	2.51	**2.44**	2.45	2.46

Figure 9: Out-of-distribution manipulations. Our approach has extrapolation capability for creating images out of the training image distribution, for example, an extremely opened mouth and a greatly enlarged wheel.

manipulation moves the whole dog's body. This also shows that point-based manipulation often has multiple possible solutions and the GAN will tend to find the closest solution in the image manifold learned from the training data. The mask function can help to reduce ambiguity and keep certain regions fixed.

Out-of-distribution manipulation. So far, the point-based manipulations we have shown are "in-distribution" manipulations, *i.e.*, it is possible to satisfy the manipulation requirements with a natural image inside the image distribution of the training dataset. Here we showcase some out-of-distribution manipulations in Fig. 9. It can be seen that our approach has some extrapolation capability, creating images outside the training image distribution, *e.g.*, an extremely opened mouth and a large wheel. In some cases, users may want to always keep the image in the training distribution and prevent it from reaching such out-of-distribution manipulations. A potential way to achieve this is to add additional regularization to the latent code $\boldsymbol{w}$, which is not the main focus of this paper.

Limitations. Despite some extrapolation capability, our editing quality is still affected by the diversity of training data. As exemplified in Fig. 14 (a), creating a human pose that deviates from the training distribution can lead to artifacts. Besides, handle points in texture-less regions sometimes suffer from more drift in tracking, as shown in Fig. 14 (b)(c). We thus suggest picking texture-rich handle points if possible.

Social impacts. As our method can change the spatial attributes of images, it could be misused to create images of a real person with a fake pose, expression, or shape. Thus, any application or research that uses our approach has to strictly respect personality rights and privacy regulations.

5 CONCLUSION

We have presented *DragGAN*, an interactive approach for intuitive point-based image editing. Our method leverages a pre-trained GAN to synthesize images that not only precisely follow user input, but also stay on the manifold of realistic images. In contrast to many previous approaches, we present a general framework by not relying on domain-specific modeling or auxiliary networks. This is achieved using two novel ingredients: An optimization of latent codes that incrementally moves multiple handle points towards their target locations, and a point tracking procedure to faithfully trace the trajectory of the handle points. Both components utilize the discriminative quality of intermediate feature maps of the GAN to yield pixel-precise image deformations and interactive performance. We have demonstrated that our approach outperforms the state of the art in GAN-based manipulation and opens new directions for powerful image editing using generative priors. As for future work, we plan to extend point-based editing to 3D generative models.

ACKNOWLEDGMENTS

Christian Theobalt was supported by ERC Consolidator Grant 4DReply (770784). Lingjie Liu was supported by Lise Meitner Postdoctoral Fellowship. This project was also supported by Saarbrücken Research Center for Visual Computing, Interaction and AI.

REFERENCES

Rameen Abdal, Yipeng Qin, and Peter Wonka. 2019. Image2stylegan: How to embed images into the stylegan latent space?. In *ICCV*.

Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. 2021. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (ToG)* 40, 3 (2021), 1–21.

Thomas Brox and Jitendra Malik. 2010. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE transactions on pattern analysis and machine intelligence* 33, 3 (2010), 500–513.

Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. 2022. Efficient Geometry-aware 3D Generative Adversarial Networks. In *CVPR*.

Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. 2021. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*.

Anpei Chen, Ruiyang Liu, Ling Xie, Zhang Chen, Hao Su, and Jingyi Yu. 2022. Sofgan: A portrait image generator with dynamic styling. *ACM Transactions on Graphics (TOG)* 41, 1 (2022), 1–26.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. StarGAN v2: Diverse Image Synthesis for Multiple Domains. In *CVPR*.

Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. 2020. Editing in style: Uncovering the local semantics of gans. In *CVPR*. 5771–5780.

Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. 2018. Generative adversarial networks: An overview. *IEEE signal processing magazine* 35, 1 (2018), 53–65.

Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. 2020. Disentangled and Controllable Face Image Generation via 3D Imitative-Contrastive Learning. In *CVPR*.

Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. 2015. Flownet: Learning optical flow with convolutional networks. In *ICCV*.

Yuki Endo. 2022. User-Controllable Latent Transformer for StyleGAN Image Layout Editing. *Computer Graphics Forum* 41, 7 (2022), 395–406. https://doi.org/10.1111/cgf.14686

Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A Efros. 2022. Blobgan: Spatially disentangled scene representations. In *ECCV*. 616–635.

Jianglin Fu, Shikai Li, Yuming Jiang, Kwan-Yee Lin, Chen Qian, Chen-Change Loy, Wayne Wu, and Ziwei Liu. 2022. StyleGAN-Human: A Data-Centric Odyssey of Human Generation. In *ECCV*.

Partha Ghosh, Pravir Singh Gupta, Roy Uziel, Anurag Ranjan, Michael J Black, and Timo Bolkart. 2020. GIF: Generative interpretable faces. In *International Conference on 3D Vision (3DV)*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NeurIPS*.

Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. 2022. StyleNeRF: A Style-based 3D-Aware Generator for High-resolution Image Synthesis. In *ICLR*.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. 2020. GANSpace: Discovering Interpretable GAN Controls. *arXiv preprint arXiv:2004.02546* (2020).

Adam W. Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. 2022. Particle Video Revisited: Tracking Through Occlusions Using Point Trajectories. In *ECCV*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *NeurIPS*.

Takeo Igarashi, Tomer Moscovich, and John F Hughes. 2005. As-rigid-as-possible shape manipulation. *ACM transactions on Graphics (TOG)* 24, 3 (2005), 1134–1141.

Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. 2017. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*.

Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2021. Alias-Free Generative Adversarial Networks. In *NeurIPS*.

Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *CVPR*. 4401–4410.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and improving the image quality of stylegan. In *CVPR*. 8110–8119.

Davis E. King. 2009. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research* 10 (2009), 1755–1758.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Thomas Leimkühler and George Drettakis. 2021. FreeStyleGAN: Free-view Editable Portrait Rendering with the Camera Manifold. 40, 6 (2021). https://doi.org/10.1145/3478513.3480538

Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. 2021. Editgan: High-precision semantic image editing. In *NeurIPS*.

Ron Mokady, Omer Tov, Michal Yarom, Oran Lang, Inbar Mosseri, Tali Dekel, Daniel Cohen-Or, and Michal Irani. 2022. Self-distilled stylegan: Towards generation from internet photos. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. (2017).

Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. 2021. Styleclip: Text-driven manipulation of stylegan imagery. In *ICCV*.

Justin N. M. Pinkney. 2020. Awesome pretrained StyleGAN2. https://github.com/justinpinkney/awesome-pretrained-stylegan2.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125* (2022).

Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. 2022. Pivotal tuning for latent-based editing of real images. *ACM Transactions on Graphics (TOG)* 42, 1 (2022), 1–13.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2021. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752 [cs.CV]

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. 2022. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. *arXiv preprint arXiv:2205.11487* (2022).

Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. 2020. GRAF: Generative Radiance Fields for 3D-Aware Image Synthesis. In *NeurIPS*.

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. 2020. Interpreting the latent space of gans for semantic face editing. In *CVPR*.

Yujun Shen and Bolei Zhou. 2020. Closed-Form Factorization of Latent Semantics in GANs. *arXiv preprint arXiv:2007.06600* (2020).

Ivan Skorokhodov, Grigorii Sotnikov, and Mohamed Elhoseiny. 2021. Aligning Latent and Image Spaces to Connect the Unconnectable. *arXiv preprint arXiv:2104.06954* (2021).

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*. PMLR, 2256–2265.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising Diffusion Implicit Models. In *ICLR*.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*.

Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. 2010. Dense point trajectories by gpu-accelerated large displacement optical flow. In *ECCV*.

Ryohei Suzuki, Masanori Koyama, Takeru Miyato, Taizan Yonetsuji, and Huachun Zhu. 2018. Spatially controllable image synthesis with internal representation collaging. *arXiv preprint arXiv:1811.10153* (2018).

Zachary Teed and Jia Deng. 2020. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*.

Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. 2020. StyleRig: Rigging StyleGAN for 3D Control over Portrait Images. In *CVPR*.

Nontawat Tritrong, Pitchaporn Rewatbowornwong, and Supasorn Suwajanakorn. 2021. Repurposing gans for one-shot semantic part segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4475–4485.

Jianyuan Wang, Ceyuan Yang, Yinghao Xu, Yujun Shen, Hongdong Li, and Bolei Zhou. 2022b. Improving gan equilibrium by raising spatial awareness. In *CVPR*. 11285–11293.

Sheng-Yu Wang, David Bau, and Jun-Yan Zhu. 2022a. Rewriting Geometric Rules of a GAN. *ACM Transactions on Graphics (TOG)* (2022).

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365* (2015).

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*.

Yuxuan Zhang, Huan Ling, Jun Gao, Kangxue Yin, Jean-Francois Lafleche, Adela Barriuso, Antonio Torralba, and Sanja Fidler. 2021. DatasetGAN: Efficient Labeled Data Factory with Minimal Human Effort. In *CVPR*.

Jiapeng Zhu, Ceyuan Yang, Yujun Shen, Zifan Shi, Deli Zhao, and Qifeng Chen. 2023. LinkGAN: Linking GAN Latents to Pixels for Controllable Image Synthesis. *arXiv preprint arXiv:2301.04604* (2023).

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. 2016. Generative visual manipulation on the natural image manifold. In *ECCV*.

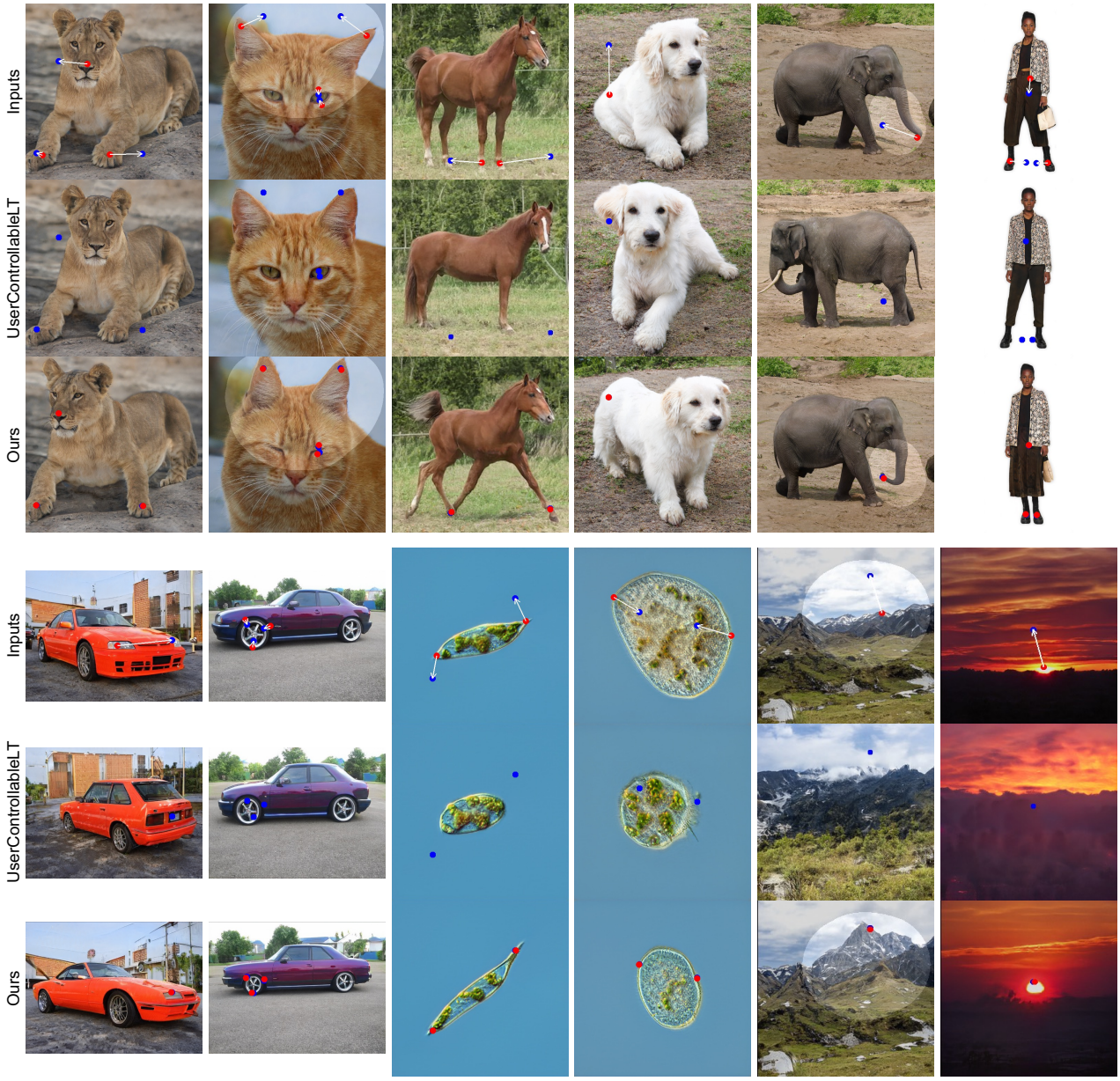

Figure 10: Qualitative comparison. This is an extension of Fig. 4.

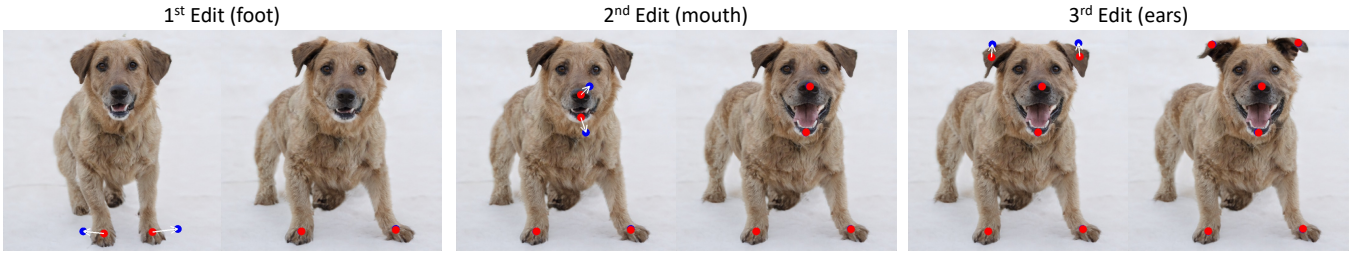

Figure 11: Continuous image manipulation. Users can continue the manipulation based on previous manipulation results.

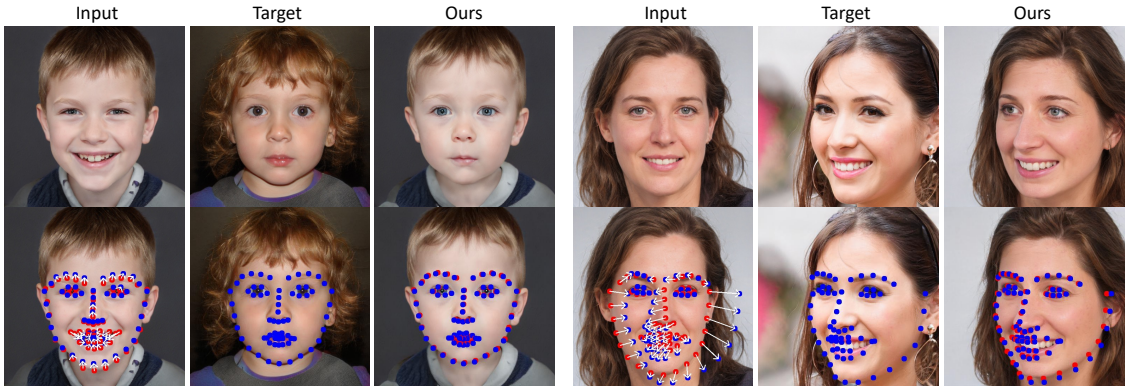

Figure 12: Face landmark manipulation. Our method works well even for such dense keypoint cases.

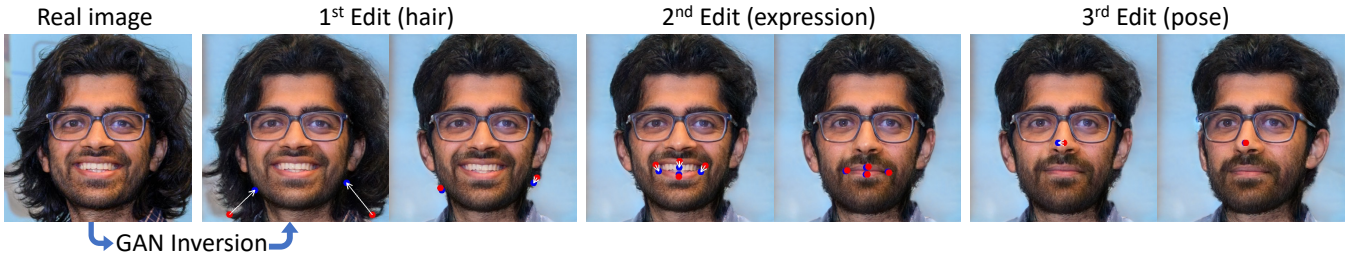

Figure 13: Real image manipulation.

Figure 14: Limitations. (a) the StyleGAN-human [Fu et al. 2022] is trained on a fashion dataset where most arms and legs are downward. Editing toward out-of-distribution poses can cause distortion artifacts as shown in the legs and hands. (b)&(c) The handle point (red) in texture-less regions may suffer from more drift during tracking, as can be observed from its relative position to the rearview mirror.

Figure 15: Effects of the mask. By masking the foreground object, we can fix the background. The details of the trees and grasses are kept nearly unchanged. Better background preservation could potentially be achieved via feature blending [Suzuki et al. 2018].

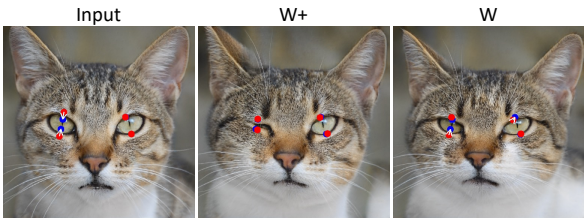

Figure 16: Effects of $\mathcal{W}/\mathcal{W}^+$ space. Optimizing the latent code in $\mathcal{W}+$ space is easier to achieve out-of-distribution manipulations such as closing only one eye of the cat. In contrast, $\mathcal{W}$ space struggles to achieve this as it tends to keep the image within the distribution of training data.