The Visual Computer
https://doi.org/10.1007/s00371-024-03310-9

ORIGINAL ARTICLE

# StylePart: image-based shape part manipulation

I.-Chao Shen[1] · Li-Wen Su[2] · Yu-Ting Wu[3] · Bing-Yu Chen[2]

Accepted: 10 February 2024


**Abstract**
Direct *part-level* manipulation of man-made shapes in an image is desired given its simplicity. However, it is not intuitive given the existing manually created cuboid and cylinder controllers. To tackle this problem, we present StylePart, a framework that enables direct shape manipulation of an image by leveraging generative models of both images and 3D shapes. Our key contribution is a shape-consistent latent mapping function that connects the image generative latent space and the 3D man-made shape attribute latent space. Our method "forwardly maps" the image content to its corresponding 3D shape attributes, where the shape part can be easily manipulated. The attribute codes of the manipulated 3D shape are then "backwardly mapped" to the image latent code to obtain the final manipulated image. By using both forward and backward mapping, an user can edit the image directly without resorting to any 3D workflow. We demonstrate our approach through various manipulation tasks, including part replacement, part resizing, and shape orientation manipulation, and evaluate its effectiveness through extensive ablation studies.

**Keywords** Image editing · Image generative model · 3D shape generative model · Latent vector mapping function

## 1 Introduction

Manipulating 3D objects derived from *a single photo* poses a considerable challenge. Unlike previous works that directly edit 3D man-made objects directly [1], this task comprises numerous complex subtasks. These tasks include extracting the target shape from the background, estimating the 3D pose, shape, and materials of the object, estimating the lighting conditions in the scene through image observation, and defining the controllers to manipulate the estimated object properties. Recently, advanced deep learning-based methods have been developed to perform numerous subtasks, including object segmentation [2]; object pose [3], shape [4–6], and material estimation [7]; and lighting estimation [8, 9].

Moreover, generative adversarial networks (GANs) have opened up a new high-fidelity image generation paradigm. For example, because of its disentangled style space, StyleGAN [10] can produce high-resolution facial images with unmatched photorealism and support stylized manipulation. Users can manipulate the generated outputs by adjusting the latent code [11–15]. Furthermore, they can also edit a natural image by projecting it into the GAN image latent space, identifying a latent code that reconstructs the input image, and then modifying that code [16, 17].

Semantic 3D controllers have been proposed to allow for semantic parameter control over images. For example, the 3D morphable face model (3DMM) has particularly been used to manipulating face shape, facial expression, head orientation, and scene illumination in both generated [18] and natural [19, 20] human face images. However, the current methods are exclusively applicable to portrait face images, posing a limitation in their usage.

In this paper, we present StylePart, which investigates how to achieve part-based manipulation of man-made objects in a single image. The key insight of our method is to augment a 3D generative model representation that is controllable and

✉ I.-Chao Shen
ichaoshen@g.ecc.u-tokyo.ac.jp

Li-Wen Su
susan31213@gmail.com

Yu-Ting Wu
yutingwu@mail.ntpu.edu.tw

Bing-Yu Chen
robin@ntu.edu.tw

[1] Computer Science Department, The University of Tokyo, Tokyo, Japan

[2] Computer Science Department, National Taiwan University, Taipei, Taiwan

[3] Computer Science Department, National Taipei University, Taipei, Taiwan

Published online: 02 April 2024

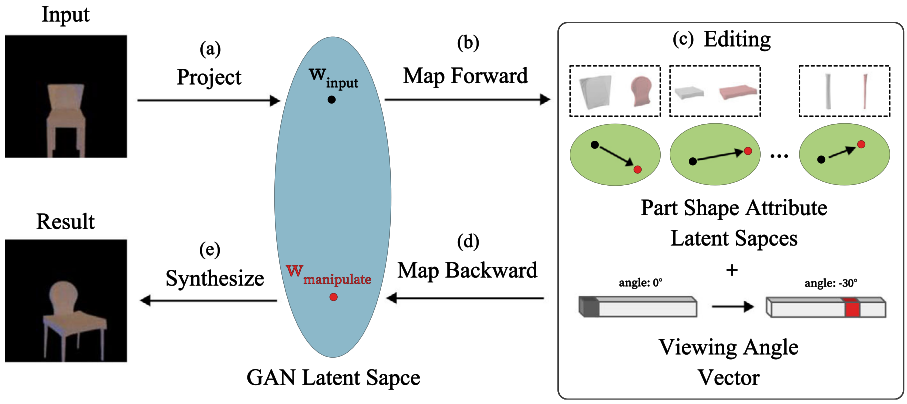



**Fig. 1** **Overview of StylePart. a** We first project the input image into the GAN latent space, and **b** map the projected GAN latent code $\mathbf{w}_{\text{input}}$ to its corresponding shape attributes and viewing angle using a forward shape-consistent latent mapping function. **c** A user can directly manipulate the image shape at the part-level. Then, we obtain the manipulated GAN latent code $\mathbf{w}_{\text{manipulate}}$ by **d** mapping the manipulated attributes to the GAN latent space with a backward mapping function. Finally, we **e** synthesize the final edited image without the need of any 3D workflow

with semantic information with the image latent space. We propose a shape-consistent mapping function that connects the image generative latent space and latent space of the 3D man-made shape attribute. The shape-consistent mapping function is composed of forward and backward mapping functions. We "forwardly map" the input image from the image latent code space to the shape attribute space, where the shape can be easily manipulated. The manipulated shape is "backwardly mapped" to the image latent code space and synthesized using a pretrained StyleGAN (Fig. 1). With the backward mapping, the user can obtain final edited image without resorting to any 3D workflow. This brings a huge advantage so that the users do not need to understand how to manipulate the lighting conditions, materials properties, and every rendering parameter. We also propose a novel training strategy that guarantees the soundness of the mapped 3D shape structure. Overall, our method offers a significant advantage over other methods as it allows for explicit part-based control through the PartVAE code. For example, although pixelNeRF [21] can be used to render novel views of a shape from a sparse set of views, it is not intuitive to manipulate the shape directly. Moreover, compared to baseline conditional VAE, our method achieves better shape reconstruction. Finally, our method includes both forward and backward mapping functions, making it highly versatile for a wide range of image-based shape editing applications as shown in this paper.

Note that we focus on the subtask of extracting controllers to manipulate the shape only, instead of proposing a full image manipulation system that tackle all subtasks altogether. Thus we use images with a simple rendering style without complex lighting conditions and material properties to demonstrate our results. We evaluate our method through shape reconstruction tests, considering four man-made object categories (chair, cup, car, and guitar). We also present several *identity-preserving* manipulated results of three shape part manipulation tasks: including part replacement, part resizing, and orientation manipulation.

## 2 Related work

### 2.1 Image-based shape reconstruction and editing

Three-dimensional modeling based on a single photo has been a challenge in the field of computer graphics and computer vision. Several studies have investigated shape inference from multiple images [22, 23] and single photos for different shape representations, such as voxel [24, 25], point cloud [26], mesh [5, 6, 27], and simple primitives [28]. With advances in deep learning methods, the quality of reconstructed shapes and images has improved tremendously; however, they are usually not semantically controllable. Chen et al. [28] proposed *3-Sweep*, an interactive method for shape extraction and manipulation in a photo. A user can create 3D primitives (cylinders and cuboids) using *3-Sweep* and manipulate the photo content using extracted primitives. Banerjee et al. [29] enable users to manually aligned a publicly available 3D model to guide the completion of geometry and light estimations. Zheng et al. [30] proposed a system that extracts cuboid-based proxies automatically and enable semantic manipulations. Unlike previous methods, our method leverages recent advances in part-based generative shape representations [31, 32] to automatically infer shape attributes. Moreover, the shape parts are more complex than simple cylinders and cuboids used in previous methods [28, 30].

### 2.2 GAN inversion and latent space manipulation

GAN inversion is required to edit a real image through latent manipulation. GAN inversion identifies the latent vector from which the generator can best replicate the input image. Inversion methods can typically be divided into optimization- and encoder-based methods. Optimization-based methods which directly optimize the latent code using a single sample [33–36], whereas encoder-based methods train an encoder over a large number of samples [37–39]. Some recent works have augmented the inversion process with additional semantic constraints [40] and additional latent codes [41]. Among these works, many have specifically considered StyleGAN inversion and investigated latent spaces with different disentanglement abilities, such as $\mathcal{W}$ [42], $\mathcal{W}^+$ [33, 34, 43], and style space ($\mathcal{S}$) [44]. Several works have examined semantic directions in the latent spaces of pretrained GANs. Full-supervision using semantic labels [11, 45] and self-supervised approaches [46–48] have been employed. Moreover, several recent studies have utilized unsupervised methods to obtain semantic directions[12, 13, 49]. A series of works focused on real human facial editing [18, 19] and non-photorealistic faces [50]; they utilized a prior in the form of a 3D morphable face model.

Unlike EditGAN [43], StylePart enables more 3D-aware part-based image editing such as part replacing and shape orientation manipulation. Compared to Liu et al. [51], there is no need to prepare a part-level CAD model beforehand and our method supports more categories of objects.

## 3 Method

We aim to enable users to directly edit the structure of a man-made shape in an image. We first describe the background of the 3D shape representation method, which allows for tight semantic control of a 3D man-made shape. We then describe a new neural network architecture that maps latent vectors between the image and 3D shape domains, and highlight the different shape part manipulation methods of the architecture.

### 3.1 3D shape attribute representation

We adopt a structured deformable mesh generative network (SDM-NET) [31] to represent the man-made shape attributes. The network allows for tight semantic control of different shape parts. Moreover, it generates a spatial arrangement of closed, deformable mesh parts, which represents the global part structure of a shape collection, e.g., chair, table, and airplane. In SDM-NET, a complete man-made shape is generated using a two-level architecture comprising a part variational autoencoder (PartVAE) and structured parts VAE (SP-VAE). PartVAE learns the deformation of a single part using its Laplacian feature vector, extracted through the method established in [52], and SP-VAE jointly learns the deformation of all parts of a shape and the structural relationship between them. In this work, given an input shape $s$ of category $c$ with $n_c$ as the number of parts, we represent its shape attributes $\boldsymbol{S} = (\boldsymbol{P}, \boldsymbol{T})$ using both

- *Geometric* attribute ($\boldsymbol{P} \in \mathbb{R}^{z \times n_c}$): all the PartVAE latent codes of each shape, where $z = 128$ is the dimension of the PartVAE latent code adopted in our work.
- *Topology* attribute ($\boldsymbol{T} \in \mathbb{R}^{2n_c+73}$): the representation vector **rv** in SDM-Net. This vector represents the associated relationships of each part in $s$, including existence, supporting, and symmetry information. Different shapes in the same category will have different **rv**. Please refer to Sect. 3.1 in Gao et al. [31].

### 3.2 Shape-consistent mapping framework

At the core of our image-based shape part manipulation pipeline (shown in Fig. 2) is a cross-domain mapping structure between the image latent space $\mathcal{W}$ and the shape attribute space $\mathcal{S}$. Given an input image containing a man-made object ($I$), the image inversion method is first applied to optimize a GAN latent code ($w_I$) that can best reconstruct the image. Then, the corresponding shape attributes ($\boldsymbol{S}$) are obtained using a forward shape-consistent latent mapping function ($M_F$). The viewing angle of the input image is predicted using a pretrained viewing angle predictor ($M_V$). From the mapped geometric attribute, topology attribute, and the predicted viewing angle, we can perform image-based shape editing tasks, such as part replacement, part deformation, and viewing angle manipulation by manipulating the attribute codes. After manipulating the attribute codes, we use the backward mapping function ($M_B$) to map the shape attributes to an image latent code ($w'$). The final edited image can be synthesized as $I_{w'} = G(w')$.

#### 3.2.1 Image inversion

To obtain the latent code of the input image $I$, we optimize the following objective:

$$w_I = \arg\min_{w} \mathcal{L}_{LPIPS}(I, G(w;\theta)) + \lambda_w \mathcal{L}_{\mathbf{w}_{\text{reg}}}(w), \qquad (1)$$

where $G$ is a pretrained StyleGAN-ADA [53] generator with weight $\theta$, $\mathcal{L}_{LPIPS}$ denotes the perceptual loss [54], and $\mathcal{L}_{\mathbf{w}_{\text{reg}}} = \|w\|_2^2$ denotes the latent code regularization loss. We introduced $\mathcal{L}_{\mathbf{w}_{\text{reg}}}$ above because we observed that there are multiple latent codes that can synthesize the same image.

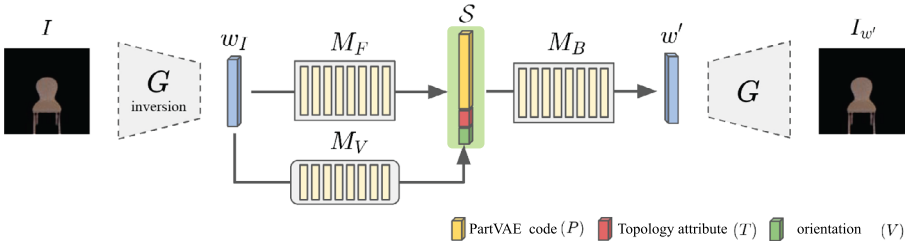



**Fig. 2** The network architecture of our cross-domain mapping framework

By introducing $\mathbf{w}_{\text{reg}}$, we regularize the solution for each input image.

#### 3.2.2 From $\mathcal{W}$ to $\mathcal{S}$

Given a GAN latent code $w_I$, which is inverted from an input image $I$, we obtain the corresponding shape attribute code $\boldsymbol{S} = (\boldsymbol{P}, \boldsymbol{T})$ using the forward mapping network $(\boldsymbol{P}, \boldsymbol{T}) = M_F(\boldsymbol{w_I})$, where the 3D shape generated by $(\boldsymbol{P}, \boldsymbol{T})$ best fits the target man-made shape encoded in the image latent code $\boldsymbol{w_I}$.

#### 3.2.3 From $\mathcal{S}$ to $\mathcal{W}$

Given the shape attribute $\boldsymbol{S}$ and one-hot vector of a viewing angle $\boldsymbol{v}$, the backward mapping function predicts a GAN latent code $\mathbf{w}' = M_B(\boldsymbol{S}, \boldsymbol{v})$, where the synthesized image $I_{w'} = G(w')$ best matches the image containing the target shape described in $\boldsymbol{S}$ with viewing angle $\boldsymbol{v}$. Our mapping functions ($M_F$ and $M_B$) are realized using two eight-layer MLP networks.

### 3.3 Training strategy and loss function

#### 3.3.1 Data preparation

To train our shape-consistent mapping function, both the image-based latent code and the shape attribute code for each man-made object shape are required. In this paper, we use synthetic datasets of four categories (chair, guitar, car, and cup). A dataset contains $N^M$ shapes, formed by interchanging the $M$ parts of $N$ shapes. For each shape, we apply a simple procedure to ensure there is no gap in the interchanged shape. We fix one part of the shape and move the rest parts toward the fixed part until the bounding boxes have an intersection. We render these shapes according to different viewing angles to obtain the paired shapes and images. We sample the viewing angles at 30° intervals on the yaw axis. For the chair and guitar category, $N = 15$ and $M = 3$; for the cup and car categories, $N = 50$ and $M = 2$. There are 3375 shapes and 40, 500 images for the chair and guitar categories, and 2500 shapes and 30, 000 images for the car and cup category. We split the synthetic datasets for each category into 80% training data and 20% testing data For shape $k$, we first prepare the shape attribute code $\boldsymbol{S}_k$. Each part is represented by a feature matrix $f \in \mathbb{R}^{9\times V}$ that describes the per-vertex deformation of a template cube with $V = 3752$ vertices. Let $n_c$ denote the number of the parts for category $c$, and $z_{\text{part}} = 128$ is the dimension of a PartVAE latent code. We embed the feature matrix of part $i$ ($f_i$) into a pretrained PartVAE and obtain $\boldsymbol{P}_i = Enc_{P_i}(f_i)$. We compute topology vector $\boldsymbol{T} \in R^{2n+73}$ for each shape. Finally, for each shape, we concatenate the PartVAE codes of all parts and the topology vector into the final shape attribute vector $\boldsymbol{S} = (\boldsymbol{P}_0, \boldsymbol{P}_1, ..., \boldsymbol{P}_{n-1}, \boldsymbol{T})$.

Next, we prepare the image-based latent code for shape $k$. We render all shapes from 12 viewing angles and use these images to train a StyleGAN2 through adaptive discriminator augmentation [53], which depends on the image's viewing angle. We project the rendered image containing shape $k$ into the pretrained conditional StyleGAN latent space to obtain the corresponding latent code $\boldsymbol{w}_k \in R^{512}$. We collect all $(\boldsymbol{w}_k, \boldsymbol{S})$ pairs for training $M_F$ and $M_B$.

#### 3.3.2 Training for $M_F$

We train the forward mapping function $M_F$ using a two-step process comprising *Laplacian feature reconstruction training* and *size finetuning*. In the Laplacian feature reconstruction training step, we use the following loss function involving the feature vector reconstruction loss ($\mathcal{L}_{\boldsymbol{P}_{\text{recon}}}$) and the topology loss ($\mathcal{L}_{\boldsymbol{T}}$):

$$\mathcal{L}_{M_F} = \mathcal{L}_{\boldsymbol{P}_{\text{recon}}} + \mathcal{L}_{\boldsymbol{T}}, \tag{2}$$

where $\mathcal{L}_{\boldsymbol{P}_{\text{recon}}}$ and $\mathcal{L}_{\boldsymbol{T}}$ are defined as follows:

$$\mathcal{L}_{\boldsymbol{P}_{\text{recon}}} = \sum_{i=1}^{N} ||Dec_{P_i}(\boldsymbol{P'}) - Dec_{P_i}(\boldsymbol{P})||_2^2 \tag{3}$$

$$\mathcal{L}_{\boldsymbol{T}} = \sum_{i=1}^{N} \|\boldsymbol{T'} - \boldsymbol{T}\|_2^2 \tag{4}$$

Here, $(\boldsymbol{P'}, \boldsymbol{T'})$ are the shape attributes predicted by $M_F$, $Dec_{P_i}(\cdot)$ denotes the pretrained PartVAE decoder of the $i$-th part, and $N$ denotes the number of parts. In the size finetuning step, we replace the reconstructed feature vectors from

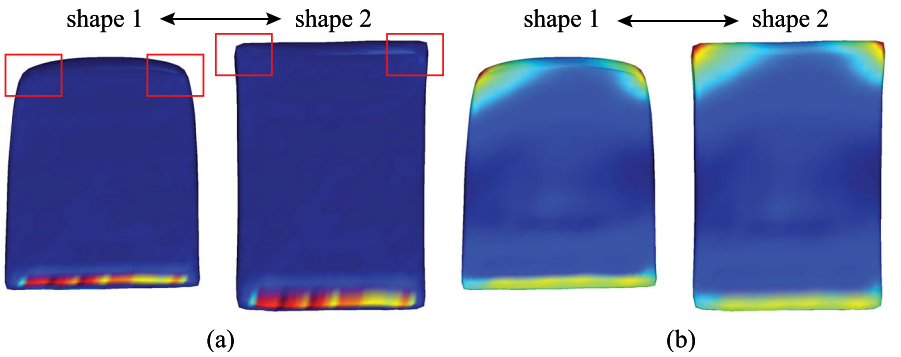



**Fig. 3** **a** With Laplacian feature differences, some deformations cannot be captured (highlighted by red rectangles). **b** On the contrary, vertex coordinate differences capture these deformations more accurately

the Laplacian feature to the vertex coordinates. As shown in Fig. 3, we observed the Laplacian feature vectors are sensitive to local differences but insensitive to global shape differences. The loss function in this step can be written as:

$$\hat{\mathcal{L}}_{M_F} = \mathcal{L}_{\boldsymbol{V}_{\text{recon}}} + \mathcal{L}_{\boldsymbol{T}} \tag{5}$$

$$\mathcal{L}_{\boldsymbol{V}_{\text{recon}}} = \sum_{i=1}^{N} ||\mathcal{T}(Dec_{P_i}(\boldsymbol{P'})) - \mathcal{T}(Dec_{P_i}(\boldsymbol{P}))||_2 \tag{6}$$

where $\mathcal{T}$ transforms a vertex Laplacian feature vector into its vertex coordinates according to the steps described in [52].

#### 3.3.3 Training for $M_B$

Our backward mapping function $M_B$ minimizes the following loss:

$$\mathcal{L}_{M_B} = \mathcal{L}_{\mathbf{w}_{\text{recon}}} + \lambda_1 \mathcal{L}_{\mathbf{w}_{\text{reg}}} \tag{7}$$

where $\lambda_1$ is the weight of the $l_2$ norm regularization term of the image latent code $w'$ (the same weight used in image inversion in Eq. 1), and $\mathcal{L}_{\mathbf{w}_{\text{recon}}}$ is defined as:

$$\mathcal{L}_{\mathbf{w}_{\text{recon}}} = \mathcal{L}_{LPIPS}(G(w'), G(w)) \tag{8}$$

where $G$ is the pretrained Style-ADA generator, $w'$ is the mapped image latent code of $(\boldsymbol{P}, \boldsymbol{T})$, and $\mathcal{L}_{LPIPS}(\cdot)$ is the perceptual loss function [54].

#### 3.3.4 Finetuning for $M_F$ and $M_B$

After training $M_F$ and $M_B$ separately, we finetune them together using the following loss function:

$$\mathcal{L}_{\text{finetune}} = \mathcal{L}_{\boldsymbol{T}} + \mathcal{L}_{\mathbf{w}_{\text{recon}}} + \lambda_1 \mathcal{L}_{\mathbf{w}_{\text{reg}}} + \lambda_2 \mathcal{L}_{\boldsymbol{P}_{\text{reg}}} \tag{9}$$

where $\lambda_2$ is the weight of a shape attribute regularization term which can be written as:

$$\mathcal{L}_{\boldsymbol{P}_{\text{reg}}} = ||M_F(w) - M_{F_{\text{freeze}}}(w)||_2, \tag{10}$$

where $F_{\text{freeze}}$ is the frozen network before finetuning. Here, we introduce a Laplacian feature reconstruction loss with a shape attribute regularization term $\mathcal{L}_{\boldsymbol{P}_{\text{reg}}}$; the loss function differs from the loss functions used to train $M_F$ and $M_B$. In Fig. 4, we show the reconstructed shapes based on PartVAE latent codes predicted with or without $\mathcal{L}_{\boldsymbol{P}_{\text{reg}}}$. $\mathcal{L}_{\boldsymbol{P}_{\text{reg}}}$ prevents substantial deviation of the PartVAE latent codes from reasonable 3D shapes. The regularization term $\mathcal{L}_{\boldsymbol{P}_{\text{reg}}}$ has a significant impact because it helps to align the original latent spaces learned from forward and backward mapping functions that are often not well aligned. By adding this term, we encourage the finetuning process to focus on the final latent space around the learned forward mapping function. This alignment leads to better performance.

## 4 Shape part manipulation

### 4.1 Part replacement

The first manipulation task is to replace parts of the input shape. Given the source image $I_{\text{source}}$, the user aims to replace one part (e.g., chair back) in $I_{\text{source}}$ with the corresponding part in the target image $I_{\text{target}}$. The part replacement procedure of our mapping framework is illustrated in Fig. 5. First, we obtain the image latent codes ($w_{\text{source}}$ and $w_{\text{target}}$) through image inversion and then we use the pretrained $M_F$ to obtain the shape attributes $\boldsymbol{S}_{\text{source}}$ and $\boldsymbol{S}_{\text{target}}$ of the shape in both input images. A user can select which part (e.g., back, seat, or leg) of the shape in $I_{\text{source}}$ he/she wants to replace. The corresponding shape attribute representing the selected part is then replaced with the target shape attribute $\boldsymbol{S}'_{\text{source}} = \{\boldsymbol{P}_0^{\text{source}}, \boldsymbol{P}_1^{\text{target}}, ...\boldsymbol{P}_{n-1}^{\text{source}}\}$. Finally, we synthesize the edited image $I' = G(M_B(\boldsymbol{S}', \boldsymbol{V}))$, where $\boldsymbol{V}$ is the original viewing angle vector. The new shape in the manipulated image will contain the part selected from $I_{\text{target}}$, while the non-selected parts will remain as close as possible to the original parts in $I_{\text{source}}$.

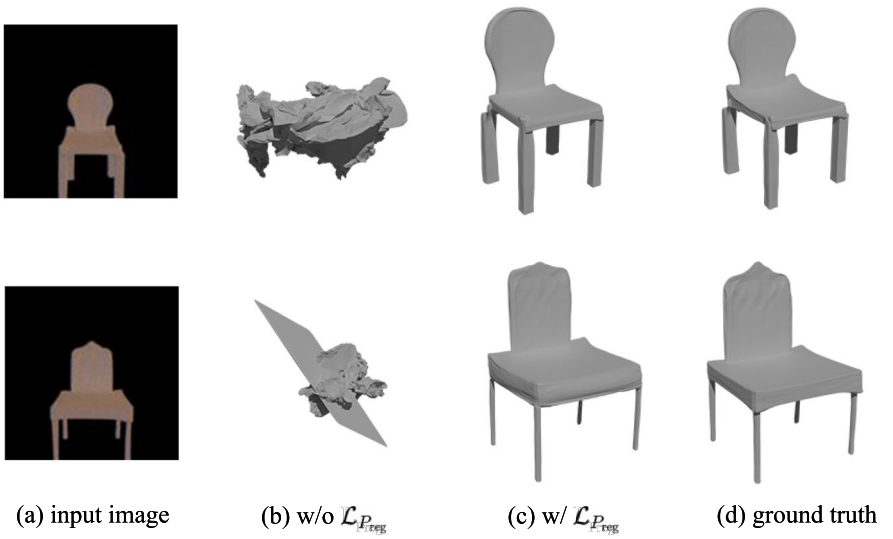



**Fig. 4** **a** Input image $I$; **b** is the mapped shape of $w_I$ without $\mathcal{L}_{\boldsymbol{P}_{\text{reg}}}$; **c** is the mapped shape of $w_I$ with $\mathcal{L}_{\boldsymbol{P}_{\text{reg}}}$; **d** is the target shape rendered in the input image

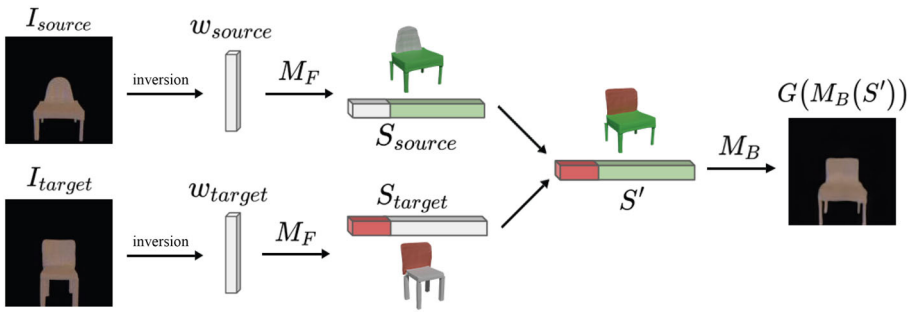



**Fig. 5** Part replacement editing process based on our pipeline

## 4.2 Part resizing

The second manipulation task is part resizing. A user directly resizes a selected part in the input image. We first invert an input image $I$ to obtain its GAN latent code $w_I$ and map $w_I$ to the shape attribute $\boldsymbol{S}$ by $M_F$. The user can resize the selected part by following a certain trajectory in the latent space and obtain the resulted image $I_{\text{resize}}$:

$$I_{resize} = G(w_I + \mathcal{F}_r(r^{\boldsymbol{P}})), \tag{11}$$

where $r^{\boldsymbol{P}}$ is the trajectory of the PartVAE latent code that represents the desired resizing result, and $\mathcal{F}_r$ is a trajectory finetuner function that refines a PartVAE code trajectory into a GAN latent code trajectory.

### 4.2.1 Resize trajectory

Trajectories that fit the desired resize manipulation in $\mathcal{S}$ and $\mathcal{W}$ are obtained using the following procedure. Given a shape $s_i$ in our dataset, we obtain its geometric attribute by $\boldsymbol{P}_{\text{i}} = \text{Enc}_P(\mathcal{T}^{-1}(s_i))$, where $\text{Enc}_P$ is a pretrained PartVAE encoder. Then we apply a specific resizing operation to $s_i$ to obtain the resized shape $\hat{s}_i = \mathcal{R}(s_i)$ and its geometric attribute $\hat{\boldsymbol{P}}_i = \text{Enc}_P(\mathcal{T}^{-1}(\hat{s}_i))$. Thus, we obtain a PartVAE latent trajectory for this resize manipulation of $s_i$ by $r_i^{\boldsymbol{P}} = \hat{\boldsymbol{P}}_i - \boldsymbol{P}_i$. We apply the same resize manipulation to all shapes in our dataset and average all PartVAE latent trajectories to obtain a general trajectory $r^{\boldsymbol{P}} = (1/N)\sum_N(r_i^{\boldsymbol{P}})$ representing the resizing manipulation. By applying $r^{\boldsymbol{P}}$, we observe that the resized images often lose some details, thus impairing shape identify (Fig. 7). For better shape identity after part resizing, we introduce a *GAN space trajectory finetuner* implemented using a four-layer MLP. The main function of this finetuner is to transform the trajectories from $\mathcal{S}$ to $\mathcal{W}$. The finetuner inputs are the GAN latent code of input image $w_I$ and a PartVAE latent trajectory of the target part, while the output is a trajectory in $\mathcal{W}$. The training data of this trajectory finetuner is paired trajectory data $(r^{\boldsymbol{P}}, r_i^{\boldsymbol{W}})$. To collect the paired trajectory data, we first add $r^{\boldsymbol{P}}$ to the PartVAE latent codes of all training shapes and obtain the rendered image latent codes (i.e., $\hat{w}_i$) by optimizing Eq. 1. For shape $i$, we obtain the GAN latent space trajectory $r_i^{\boldsymbol{W}}$ corresponding to $r^{\boldsymbol{P}}$ by $r_i^{\boldsymbol{W}} = \hat{w}_i - w_I$. We train this trajectory finetuner by minimizing the following loss function:

$$\mathcal{L}_r = \|\mathcal{F}_r(w_I, r^{\boldsymbol{P}}) - r_i^{\boldsymbol{W}}\|_2^2. \tag{12}$$

We use the shapes in the datasets mentioned in Sect. 5.1 as the original shapes. For each part, we resize shapes by adding

three weights ($-0.5r^{\boldsymbol{P}}, +0.5r^{\boldsymbol{P}}, +1.0r^{\boldsymbol{P}}$) to the specific shape attribute trajectory. There are 10, 125 (3, 375 × 3) shapes for each part of chair and guitar, and 30, 375 (10, 125 × 3 parts) shapes in total. There are 7, 500 (2, 500 × 3) shapes for each part of car and cup, and a total of 21, 500 (7500 × 3 parts) and 15, 000 (7500. × 2 parts) shapes, respectively.

### 4.3 Shape orientation manipulation

The third editing task is shape orientation manipulation. To achieve this, we first train a shape orientation prediction network ($M_V$) to predict the orientation of an image $I$ from its GAN latent code $w_I$, i.e., $\boldsymbol{V} = M_V(w_I)$. The shape orientation prediction network is an eight-layer MLP trained with the cross-entropy loss. For each category, we use the rendered images described in Sect. 3.3 and collect 36, 000 paired training data $(w, v)$ from 3000 shapes in 12 different orientations.

To manipulate the orientation of input image $I$, we obtain the shape attribute $\boldsymbol{S}$ of the shape in the input image using $M_F$. We obtain the edited image with the manipulated orientation vector: $I' = G(M_B((\boldsymbol{S}, \boldsymbol{V}')))$.

## 5 Experiment and results

### 5.1 Implementation details

For each man-made object category, we trained the StyleGAN-ADA generator using a batch size of 64 and learning rate of 0.0025. Both of our forward and backward mapping networks were trained using the Adam [55] optimizer for 3000 epochs and a batch size 64. All networks were trained using a Tesla V100 GPU. We implemented our pipeline using PyTorch [56]. The forward mapping network were trained for 0.5-3 days, depending on the number of part labels. The backward mapping network was trained for 20 hours, and the forward and backward mapping networks were finetuned together for 10 hours. *Baseline disentangled network* We use a conditional variational autoencoder (cVAE) as the baseline disentangled method (we provide the architecture in the supplemental material) and compare our results with this baseline on image shape reconstruction and the results of viewing angle manipulation quantitatively. For each shape, we include three factors: part identity, part size, and viewing angle. All these factors are modeled as discrete variables and represented as one-hot vectors during training and testing.

### 5.2 Image shape reconstruction

The key contribution of the proposed framework is the mapping functions between the GAN latent space and the shape attribute latent space. We qualitatively and quantitatively evaluated these mapping functions through image shape reconstruction.

#### 5.2.1 With/without size finetuning for $M_F$

As described in Sect. 3.3, after the *Laplacian feature reconstruction training*, we performed an additional *size finetuning* for $M_F$ with vertex coordinate loss for a better reconstruction. To test the effectiveness of the *size finetuning*, we evaluated the reconstruction quality of the front view of the **testing images** described in Sect. 3.3. We sampled 4000 points on each shape and calculated the bidirectional Chamfer distance as our shape reconstruction error metric, and we used perceptual loss as the image reconstruction error metric. Both the image reconstruction and the shape reconstruction errors were lower when the network was trained with *size finetuning* (Table 1).

#### 5.2.2 Finetuning for $M_F$ and $M_B$

After separately training $M_F$ and $M_B$, we performed an end-to-end finetuning to improve the reconstruction results. We compared the full and non-finetuned versions of our method. The full version included the shape attribute regularizer $\mathcal{L}_{\boldsymbol{P}_{\text{reg}}}$ to prevent the shape from collapsing. We tested on about 300 images in each category and obtained the mean perceptual losses of finetuned networks and non-finetuned networks. Through this step, the image reconstruction errors for chair, car, cup, and guitar categories were reduced by 28%, 37%, 27%, and 32% respectively.

### 5.3 Shape part manipulation results

#### 5.3.1 Part replacement results

We randomly picked 12 different chair images and exchanged their parts. We applied several replacement patterns to all possible combinations of the picked 12 images and produced edited images for each category. We show some replacement results in Fig. 6. Detailed results are available in the supplementary material.

#### 5.3.2 Part resizing results

We trained separate GAN latent trajectory finetuners for each part using the resizing dataset discussed in Sect. 4.2. We compared the test results obtained by two methods: using shape attribute trajectory ($r^{\boldsymbol{P}}$) and the GAN space trajectory finetuner ($\mathcal{F}_r$). The comparison results and mean perceptual losses are shown in Fig. 7 and Table 2, respectively. The manipulation result obtained from $\mathcal{F}_r$ showed less perceptual loss and matched the desired resize manipulation better than the results obtained from $r^{\boldsymbol{P}}$. Moreover, we applied multiple

**Table 1** Comparison of the mean shape reconstruction errors ($E_s$) and mean image reconstruction errors ($E_i$) of results obtained from with and without size finetuning, and the baseline conditional VAE (cVAE) method $\mathcal{B}$. We did not report the mean shape reconstruction error for the baseline method because it did not reconstruct an explicit shape

| | Chair | | | Car | | | Cup | | | Guitar | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | w/ | w/o | $\mathcal{B}$ | w | w/o | $\mathcal{B}$ | w | w/o | $\mathcal{B}$ | w | w/o | $\mathcal{B}$ |
| $E_s \downarrow$ | **1.28** | 2.63 | – | **1.45** | 3.07 | – | **0.99** | 1.20 | – | **0.40** | 0.74 | – |
| $E_i \downarrow$ | **0.09** | 0.12 | 0.14 | **0.01** | 0.02 | 0.06 | **0.07** | 0.12 | 0.21 | **0.01** | 0.08 | 0.13 |

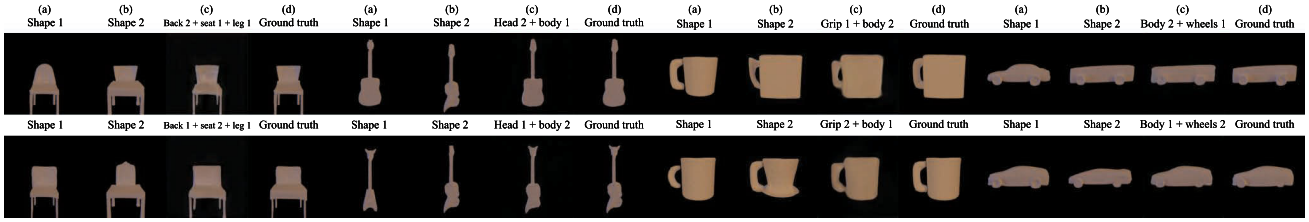



**Fig. 6** Results of part replacement. The images were produced from the attributes of shapes 1 and 2

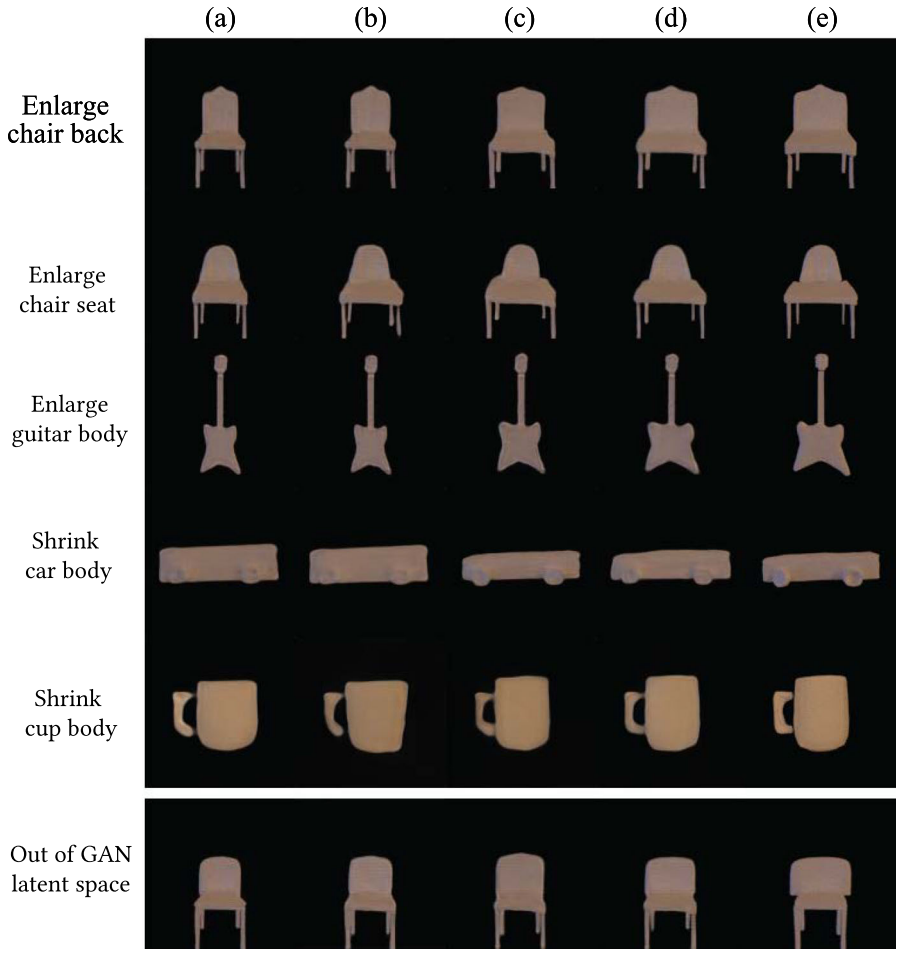



**Fig. 7** Part resizing results for different categories. **a** is the input image, **b** resized image directly obtained using $P + r^{\boldsymbol{P}}$; **c** resized image obtained using GAN space trajectory finetuner; **d** is the inverted result of ground truth, and **e** is the rendered ground truth

trajectory finetuners trained on different parts of the same image to simultaneously resize multiple parts. Regarding the failure case (last row in Fig. 7), it fails because the target resized image was outside the GAN latent space, and the finetuner could not find a good trajectory to fit the target image.

**Table 2** Quantitative results for the part resizing manipulation. Comparison of mean perceptual losses of results obtained using shape attribute trajectory ($r^{\boldsymbol{P}}$) and the GAN space trajectory finetuner ($F_r$)

| | Chair | | | Car | | Cup | | Guitar | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Back | Seat | Legs | Body | Wheel | Body | Grip | Head | Neck | body |
| $r^{\boldsymbol{P}}$ | 8.22 | 11.95 | 5.60 | 13.70 | 7.05 | 14.55 | 12.67 | 1.62 | 2.85 | 4.62 |
| $\mathcal{F}_r$ | 6.18 | 7.73 | 5.33 | 6.34 | 4.42 | 6.88 | 5.85 | 0.96 | 1.58 | 2.08 |

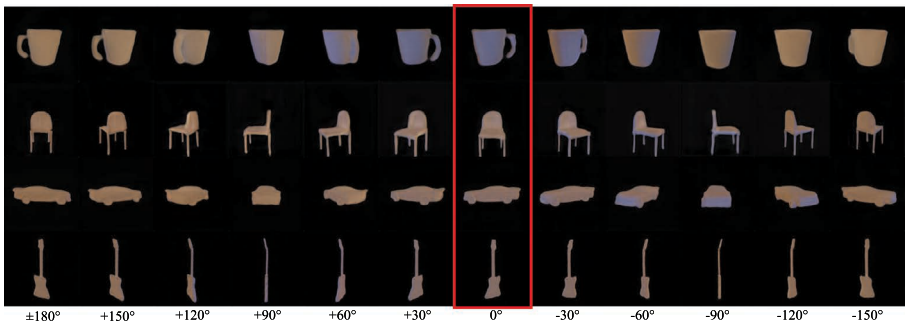



**Fig. 8** Shape orientation manipulation results. Our method can synthesize images of the shape from different viewing angles

**Table 3** Shape orientation manipulation quantitative evaluation

| | Chair↓ | Car↓ | Cup↓ | Guitar↓ |
|---|---|---|---|---|
| Our | 0.0022 | 0.0075 | 0.0029 | 0.0043 |
| $\mathcal{B}$ | 0.0416 | 0.05 | 0.07 | 0.02 |
| pixelNeRF | 0.0074 | 0.022 | 0.0091 | 0.0085 |

### 5.3.3 Shape orientation manipulation results

Figure 8 shows the shape orientation manipulation results. By manipulating the one-hot vector of the shape orientation of a given image, our method can produce the images of different shape orientations. We tested the images described in Sect. 3.3 against the baseline method. For each shape, each method synthesizes images from 12 different orientations as described in Sect. 5.3.3. In Table 3, we show mean perceptual losses across different orientations. The results of our method obtain lower perceptual losses compared to the results of the baseline method (cVAE) and pixelNeRF [21]. We trained a 1-view pixelNeRF using our training data. This result suggested that the explicit shape attribute latent space helps to build up better geometries compared to random latent vectors and implicit geometries learned by radiance field-based method [21]. The shape identities of the manipulated results (Fig. 8) were well maintained. We showed the visual comparisons of our results and the results of pixelNeRF [21] in Fig. 9. The results of pixelNeRF often fail to reconstruct the geometry details of shape parts; thus, the shape identities are not maintained.

## 5.4 Real image results

We tested our pipeline for shape part manipulation of real images. To infer the shape attributes of an object in the input real image $I$, we first colorized the shape region in $I$ so that it had the same appearance as our dataset. We projected the colorized image $\tilde{I}$ into the GAN latent space to obtain its GAN latent code $w_{\tilde{I}}$. Because the shape in $\tilde{I}$ was often outside the domain covered by our training dataset, our pipeline yielded unsatisfactory results (*without shape-specific finetune* row in Fig. 10). To address this problem, we designed a

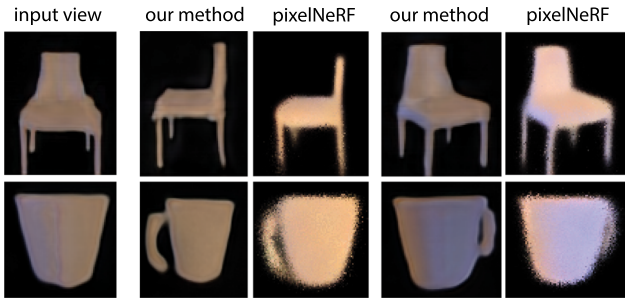



**Fig. 9** Comparison with pixelNeRF [21]. Our method synthesized shapes with more details while pixelNeRF only synthesize blurry shapes using the same input orientation.

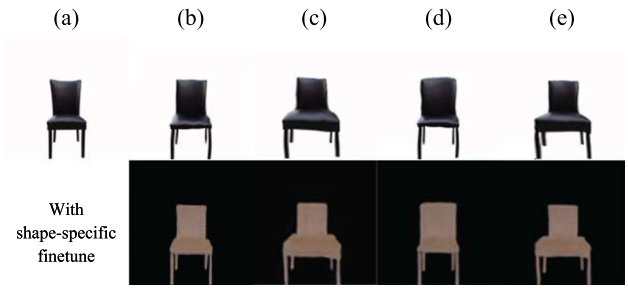



**Fig. 10** The first row shows the manipulated results. The second row shows their shape images inferred through shape-specific finetune. **a** Input real image; **b** shape and image reconstructed using our pipeline; **c** seat replacement result; **d** resizing result of the wider back and **e** the wider seat

shape-specific finetuning process. We identified the best forward and backward mapping network parameters $\theta'_{M_F}$ and $\theta'_{M_B}$ for $\tilde{I}$ by optimizing the following function:

$$\mathcal{L}_{LPIPS}(I', I) + \lambda_3 \theta'_{M_F} - \theta_{M_F} 2 + \lambda_4 \theta'_{M_B} - \theta_{M_B} 2, \qquad (13)$$

where $\theta_{M_F}$ and $\theta_{M_B}$ are the parameters of the pretrained forward and backward mapping functions, and $\lambda_3$ and $\lambda_4$ are the weights of losses representing the distances between the pretrained mapping functions and optimized mapping functions. Figure 10 shows the manipulated results for a real image obtained through the shape-specific finetuning process. After obtaining the manipulated image, we warped the input texture in the source real image using TPS warping. We identified the object contours in the input image and the manipulated images using [57], and sampled points on the contours as the control points for TPS warping.

## 6 Conclusion and limitation

In this paper, we propose a framework for bridging the image latent space and 3D shape attribute space using shape-consistent mapping functions. Furthermore, we show that the mapping functions enable image shape part manipulation subtasks such as part replacement, part resizing, and shape orientation manipulation without the need of any 3D workflow. *Fixed shape attribute space.* Our trained mapping function assumed fixed the number of parts, which is inherited from SDMNet [31]. In future, we plan to explore a more flexible shape attribute space. *Gap between synthesized images and real images.* Our framework focuses on the geometry of the input synthesized image. To manipulate the shape of a real image, we need to infer the appearance of the deformed textures to achieve realistic manipulation results. We plan to learn mapping functions between realistic image with rendered image using image translation network [58].

**Acknowledgements** We thank the anonymous reviewers for their valuable comments. This work was supported in part by JSPS Grant-in-Aid JP23K16921, and National Science and Technology Council, under Grant, NSTC111-2634-F-002-022, 111-2221-E-002-145-MY3, 111-3111-E-002-002, 112-2218-E-002-029, and 111-2222-E-305-001-MY2 and National Taiwan University (NTU112L900902).

**Funding** Open Access funding provided by The University of Tokyo.

**Data availability statement** The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

## Declarations

**Conflict of interest** Following are potential conflicts of interest: Yung-Yu Chuang (https://www.csie.ntu.edu.tw/~cyy/) Takeo Igarashi (https://www-ui.is.s.u-tokyo.ac.jp/~takeo/index.html)

**Research involving Human Participants and/or Animals** This research did not involve human participants.



## References

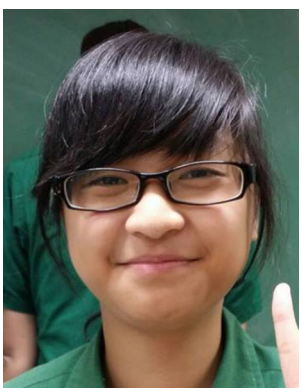
1. Gal, R., Sorkine, O., Mitra, N.J., Cohen-Or, D.: Iwires: An analyze-and-edit approach to shape manipulation. In: ACM SIGGRAPH 2009 Papers. SIGGRAPH '09. Association for Computing Machinery (2009). https://doi.org/10.1145/1576246.1531339
2. He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
3. Song, C., Song, J., Huang, Q.: Hybridpose: 6d object pose estimation under hybrid representations. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
4. Feng, Y., Feng, H., Black, M.J., Bolkart, T.: Learning an animatable detailed 3D face model from in-the-wild images. ACM Trans. Gr. (ToG) Proc. SIGGRAPH **40**(4), 88–18813 (2021)
5. Georgia Gkioxari, J.J. Jitendra Malik: Mesh r-cnn. In: Proceedings of the IEEE international conference on computer vision (ICCV) 2019 (2019)
6. Lei, J., Sridhar, S., Guerrero, P., Sung, M., Mitra, N., Guibas, L.J.: Pix2surf: Learning parametric 3d surface models of objects from images. In: Proceedings of European Conference on Computer Vision (ECCV) (2020). https://geometry.stanford.edu/projects/pix2surf
7. Liu, G., Ceylan, D., Yumer, E., Yang, J., Lien, J.-M.: Material editing using a physically based rendering network. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (spotlight) (2017)
8. Garon, M., Sunkavalli, K., Hadap, S., Carr, N., Lalonde, J.-F.: Fast spatially-varying indoor lighting estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
9. Zhu, Y., Zhang, Y., Li, S., Shi, B.: Spatially-varying outdoor lighting estimation from intrinsics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12834–12842 (2021)
10. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4401–4410 (2019)
11. Shen, Y., Gu, J., Tang, X., Zhou, B.: Interpreting the latent space of gans for semantic face editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9243–9252 (2020)
12. Shen, Y., Zhou, B.: Closed-form factorization of latent semantics in gans. In: CVPR (2021)
13. Härkönen, E., Hertzmann, A., Lehtinen, J., Paris, S.: Ganspace: Discovering interpretable gan controls. arXiv preprint arXiv:2004.02546 (2020)

14. Jahanian, A., Chai, L., Isola, P.: On the "steerability" of generative adversarial networks. In: International Conference on Learning Representations (2020)
15. Chen, A., Liu, R., Xie, L., Chen, Z., Su, H., Jingyi, Y.: Sofgan: A portrait image generator with dynamic styling. ACM Trans. Graph. (2021). https://doi.org/10.1145/3470848
16. Zhu, J.-Y., Krähenbühl, P., Shechtman, E., Efros, A.A.: Generative visual manipulation on the natural image manifold. In: Proceedings of European Conference on Computer Vision (ECCV) (2016)
17. Hin, T.C.L., Shen, I.-C., Sato, I., Igarashi, T.: Interactive optimization of generative image modeling using sequential subspace search and content-based guidance. Comput. Gr. Forum (2021). https://doi.org/10.1111/cgf.14188
18. Tewari, A., Elgharib, M., Bharaj, G., Bernard, F., Seidel, H.-P., Pérez, P., Zollhofer, M., Theobalt, C.: Stylerig: Rigging stylegan for 3d control over portrait images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6142–6151 (2020)
19. Tewari, A., Elgharib, M., Bernard, F., Seidel, H.-P., Pérez, P., Zollhöfer, M., Theobalt, C.: Pie: Portrait image embedding for semantic control. ACM Trans. Gr. (TOG) **39**(6), 1–14 (2020)
20. Tewari, A., Zollhofer, M., Kim, H., Garrido, P., Bernard, F., Perez, P., Theobalt, C.: Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 1274–1283 (2017)
21. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelnerf: Neural radiance fields from one or few images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4578–4587 (2021)
22. Debevec, P.E., Taylor, C.J., Malik, J.: Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '96, pp. 11–20. Association for Computing Machinery, New York, NY, USA (1996). https://doi.org/10.1145/237170.237191
23. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: Exploring photo collections in 3d. In: SIGGRAPH Conference Proceedings, pp. 835–846. ACM Press, New York, NY, USA (2006)
24. Wu, J., Wang, Y., Xue, T., Sun, X., Freeman, W.T., Tenenbaum, J.B.: MarrNet: 3D Shape Reconstruction via 2.5D Sketches. In: Advances In Neural Information Processing Systems (2017)
25. Yan, X., Yang, J., Yumer, E., Guo, Y., Lee, H.: Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 1696–1704. Curran Associates, Inc. (2016)
26. Su, H., Fan, H., Guibas, L.: A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
27. Mao, A., Dai, C., Gao, L., He, Y., Liu, Y.-j.: Std-net: Structure-preserving and topology-adaptive deformation network for 3d reconstruction from a single image. arXiv preprint arXiv:2003.03551 (2020)
28. Chen, T., Zhu, Z., Shamir, A., Hu, S.-M., Cohen-Or, D.: 3-sweep: Extracting editable objects from a single photo. ACM Trans. Gr. (TOG) **32**(6), 1–10 (2013)
29. Kholgade, N., Simon, T., Efros, A., Sheikh, Y.: 3d object manipulation in a single photograph using stock 3d models. ACM Trans. Comput. Gr. **33**(4) (2014)
30. Zheng, Y., Chen, X., Cheng, M.-M., Zhou, K., Hu, S.-M., Mitra, N.J.: Interactive images: Cuboid proxies for smart image manipulation. ACM Trans. Graph. (2012). https://doi.org/10.1145/2185520.2185595
31. Gao, L., Yang, J., Wu, T., Yuan, Y.-J., Fu, H., Lai, Y.-K., Zhang, H.: Sdm-net: Deep generative network for structured deformable mesh. ACM Trans. Gr. (TOG) **38**(6), 1–15 (2019)
32. Mo, K., Guerrero, P., Yi, L., Su, H., Wonka, P., Mitra, N., Guibas, L.: Structurenet: Hierarchical graph networks for 3d shape generation. ACM Trans. Gr. TOG Siggraph Asia **38**(6), 242 (2019)
33. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 4432–4441 (2019)
34. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan++: How to edit the embedded images? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8296–8305 (2020)
35. Creswell, A., Bharath, A.A.: Inverting the generator of a generative adversarial network. IEEE Trans. Neural Netw. Learn. Syst. **30**(7), 1967–1974 (2018)
36. Huh, M., Zhang, R., Zhu, J.-Y., Paris, S., Hertzmann, A.: Transforming and projecting images into class-conditional generative networks. In: European Conference on Computer Vision, pp. 17–34 (2020). Springer
37. Guan, S., Tai, Y., Ni, B., Zhu, F., Huang, F., Yang, X.: Collaborative learning for faster stylegan embedding. arXiv preprint arXiv:2007.01758 (2020)
38. Tov, O., Alaluf, Y., Nitzan, Y., Patashnik, O., Cohen-Or, D.: Designing an encoder for stylegan image manipulation. ACM Trans. Gr. (TOG) **40**(4), 1–14 (2021)
39. Alaluf, Y., Patashnik, O., Cohen-Or, D.: Restyle: A residual-based stylegan encoder via iterative refinement. arXiv preprint arXiv:2104.02699 (2021)
40. Zhu, J., Shen, Y., Zhao, D., Zhou, B.: In-domain gan inversion for real image editing. In: Proceedings of European Conference on Computer Vision (ECCV) (2020)
41. Gu, J., Shen, Y., Zhou, B.: Image processing using multi-code gan prior. In: CVPR (2020)
42. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8110–8119 (2020)
43. Ling, H., Kreis, K., Li, D., Kim, S.W., Torralba, A., Fidler, S.: Editgan: High-precision semantic image editing. In: Advances in Neural Information Processing Systems (NeurIPS) (2021)
44. Wu, Z., Lischinski, D., Shechtman, E.: Stylespace analysis: Disentangled controls for stylegan image generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12863–12872 (2021)
45. Goetschalckx, L., Andonian, A., Oliva, A., Isola, P.: Ganalyze: Toward visual definitions of cognitive image properties. arXiv preprint arXiv:1906.10112 (2019)
46. Jahanian, A., Chai, L., Isola, P.: On the "steerability" of generative adversarial networks. arXiv preprint arXiv:1907.07171 (2019)
47. Plumerault, A., Borgne, H.L., Hudelot, C.: Controlling generative models with continuous factors of variations. arXiv preprint arXiv:2001.10238 (2020)
48. Spingarn-Eliezer, N., Banner, R., Michaeli, T.: Gan "steerability" without optimization. arXiv preprint arXiv:2012.05328 (2020)
49. Wang, B., Ponce, C.R.: The geometry of deep generative image models and its applications. arXiv preprint arXiv:2101.06006 (2021)
50. Wang, C., Chai, M., He, M., Chen, D., Liao, J.: Cross-domain and disentangled face manipulation with 3d guidance. arXiv preprint arXiv:2104.11228 (2021)
51. Liu, Z., Lu, F., Wang, P., Miao, H., Zhang, L., Yang, R., Zhou, B.: 3d part guided image editing for fine-grained object understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11336–11345 (2020)

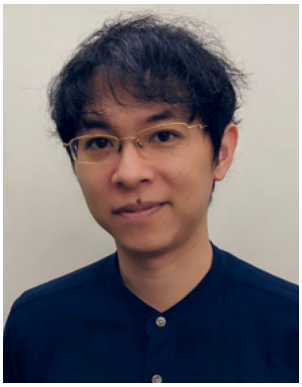
52. Gao, L., Lai, Y.-K., Yang, J., Ling-Xiao, Z., Xia, S., Kobbelt, L.: Sparse data driven mesh deformation. IEEE Trans. Visual. Comput. Gr. (2019)
53. Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., Aila, T.: Training generative adversarial networks with limited data. In: Proc. NeurIPS (2020)
54. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
55. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
56. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019)
57. Suzuki, S., et al.: Topological structural analysis of digitized binary images by border following. Comput. Vis. Gr. Image Process. **30**(1), 32–46 (1985)
58. Hu, R., Su, X., Chen, X., van Kaick, O., Huang, H.: Photo-to-shape material transfer for diverse structures. ACM Trans. Gr. Proc. SIGGRAPH **39**(6), 113–111314 (2022)



**I.-Chao Shen** is an assistant professor at the Graduate School of Information Science and Technology at the University of Tokyo, working with Takeo Igarashi. He did his PhD in the computer graphics group at National Taiwan University, advised by Robin Bing-Yu Chen. He received B.B.A and M.B.A degrees in information management from National Taiwan University, in 2009 and 2011, respectively. He was a JSPS postdoctoral researcher and project assistant professor at The University of Tokyo.



**Li-Wen Su** is a software engineer at Akatsuki Games, Inc, Japan. She was a master student in CMLab, National Taiwan University, Taiwan.



**Yu-Ting Wu** received his B.S. and M.S. from National Chiao Tung University in 2007 and 2009 respectively, and the PhD degree from National Taiwan University in 2014, all in Computer Science. He is currently an assistant professor with the Department of Computer Science and Information Engineering at National Taipei University. His research interests include computer graphics, extended reality, computer vision, and visual effects.

**Bing-Yu Chen** aka Robin Chen, received the B.S. and M.S. degrees in Computer Science and Information Engineering from National Taiwan University, Taipei, Taiwan, in 1995 and 1997, respectively, and the PhD degree in Information Science from The University of Tokyo, Tokyo, Japan, in 2003. He is currently a Distinguished Professor with Department of Computer Science and Information Engineering, Department of Information Management, and Graduate Institute of Networking and Multimedia of National Taiwan University (NTU), and also the Dean of NTU College of Innovation and Design (D-School). His current research interests include Human-Computer Interaction, Computer Graphics, and Image Processing. He is a senior member of ACM and IEEE.