DeepMetaHandles: Learning Deformation Meta-Handles of 3D Meshes with Biharmonic Coordinates

Minghua Liu[1] Minhyuk Sung[2] Radomir Mech[3] Hao Su[1]
[1]University of California San Diego [2]KAIST [3]Adobe Research

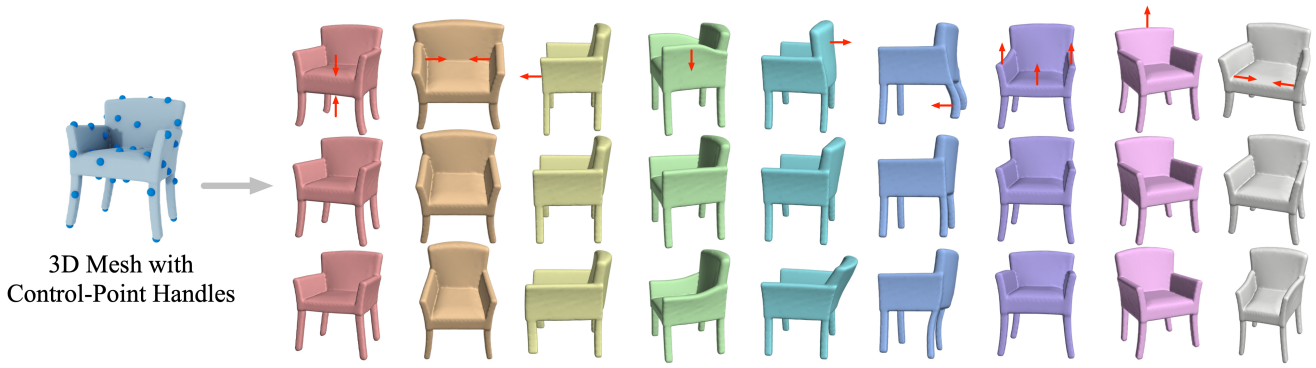

Figure 1: Learned meta-handles for a single chair. Each column indicates a meta-handle and shows three deformations along the direction of that meta-handle, with red arrows highlighting the deformed region. Our method learns intuitive and disentangled meta-handles in an unsupervised fashion, which factorize all the plausible deformations for the shape.

Abstract

*We propose **DeepMetaHandles**, a 3D conditional generative model based on mesh deformation. Given a collection of 3D meshes of a category and their deformation handles (control points), our method learns a set of meta-handles for each shape, which are represented as combinations of the given handles. The disentangled meta-handles factorize all the plausible deformations of the shape, while each of them corresponds to an intuitive deformation. A new deformation can then be generated by sampling the coefficients of the meta-handles in a specific range. We employ biharmonic coordinates as the deformation function, which can smoothly propagate the control points' translations to the entire mesh. To avoid learning zero deformation as meta-handles, we incorporate a target-fitting module which deforms the input mesh to match a random target. To enhance deformations' plausibility, we employ a soft-rasterizer-based discriminator that projects the meshes to a 2D space. Our experiments demonstrate the superiority of the generated deformations as well as the interpretability and consistency of the learned meta-handles. The code is available at* https://github.com/Colin97/DeepMetaHandles.

1. Introduction

3D Meshes can store sharp edges and smooth surfaces compactly. However, Learning to generate 3D meshes is much more challenging than 2D images due to the irregularity of mesh data structures and the difficulty in designing loss functions to measure geometrical and topological properties. For such reasons, to create new meshes, instead of generating a mesh from scratch, recent work assumes that the connectivity structure of geometries is known so that the creation space is restricted to changing the geometry without altering the structure. For example, [37, 36, 48] create new shapes by *deformations* of one template mesh. They, however, limit the scope of the shape generation to possible variants of the template mesh. We thus propose a 3D *conditional* generative model that can take any existing mesh as input and produce its plausible variants. Our approach integrates a *target-driven* fitting component and a conditional generative model. At test time, it allows both deforming the input shape to fit the given target shape and exploring plausible variants of the input shape without a target.

Our main design goals are two-fold: improving the *plausibility* of the output shapes and enhancing the *interpretability* of the learned latent spaces. To achieve the goals, the key is to choose a suitable parameterization of deformations. One option is to follow the recent target-driven deformation network [39, 9, 46, 35], which parameterizes the deformation as new positions of all the mesh vertices. However, such a large degree of freedom often results in the loss of fine-grained geometric details and tends to cause undesirable distortions. Instead of following the above works, we leverage a classical idea in computational geometry, named

deformation handles, to parameterize smooth deformations with a low degree of freedom. Specifically, we propose to take a small set of *control points* as deformation handles and utilize a deformation function defined on the control points and their *biharmonic coordinates* [41].

Not all the translations of the control points lead to plausible deformations. Based on the control-point handles, we aim to learn a low-dimensional deformation subspace for each shape, and we expect the structure of this subspace to exhibit *interpretability*. In contrast to typical generative models, where shape variations are embedded into a latent space implicitly, our method explicitly factorizes all the plausible deformations of a shape with a small number of interpretable deformation functions. Specifically, for each axis of our input-dependent latent space, we assign a deformation function defined with the given set of control points and offset vectors on them so that each axis corresponds to an intuitive deformation direction. Since each axis is explicitly linked to multiple control-point handles, we thus call them *meta-handles*. We enforce the network to learn *disentangled* meta-handles, in the sense that a meta-handle should not only leverage the correlations of the control-point handles, but also correspond to a group of parts that tend to deform altogether according to the dataset. We hope that the disentangled meta-handles allow us to deform each part group independently in downstream applications.

Beyond choosing the parameterization of deformations, we have to overcome the challenge of examining the plausibility. In the popular adversarial learning framework, a straightforward approach would be converting the output mesh to voxels or point clouds and exploiting voxel or point cloud based discriminators. The conversions, however, may discard some important geometric details. In our method, we instead project the shapes into a 2D space with a differentiable *soft rasterizer* [25] and employ a 2D discriminator. We found that this architecture can be trained more robustly, and it captures local details of plausible shapes.

Our deformation-based conditional generative model, named **DeepMetaHandles**, takes random pairs of source and target shapes as input during training. For the source shape, the control points are sampled from its mesh vertices by farthest point sampling, and the biharmonic coordinates [41] for control-point handles are pre-computed. Our network consists of two main modules: MetaHandleNet and DeformNet. The MetaHandleNet first predicts a set of meta-handles for the source shape, where each meta-handle is represented as a combination of control-point offsets. A deformation range is also predicted for each meta-handle, describing the scope of plausible deformations along that direction. The learned meta-handles, together with the corresponding ranges, define a deformation subspace for the source shape. Then, DeformNet predicts coefficients multiplied to the meta-handles, within the predicted ranges, so that the source shape deformed with the coefficients can match the target shape. To ensure the plausibility of variations within the learned subspace, we then randomly sample coefficients within the predicted ranges and apply both geometric and adversarial regularizations to the corresponding deformations.

Fig. 1 shows examples of the learned meta-handles, which interestingly resemble natural deformations of *semantic* parts, such as lifting the armrests or bending the back of a chair. Our experiments also show that the learned meta-handles are consistent across various shapes and well disentangle the shape variation space. Finally, we compare our approach with other target-driven deformation techniques [13, 39, 9, 46] and demonstrate that our method produces superior fitting results.

Key contributions:

- We propose DeepMetaHandles, a 3D *conditional* generative model based on mesh deformation.
- We employ a few control points as deformation handles. Together with their biharmonic coordinates, we can produce smooth but flexible enough deformations.
- We propose to factorize the deformation space with a small number of disentangled meta-handles, each of which provides an intuitive deformation by leveraging the correlations between the control points.
- We improve the plausibility of the deformations by exploiting a differentiable renderer and a 2D discriminator.

2. Related Work

Learning 3D Shape Deformations. 3D shape deformation is a classic subject in computational geometry that has been studied extensively for decades. The problem is typically formulated as an optimization problem minimizing the fitting error from the source to target shape and also some regularization errors (e.g., local rigidity). Recent work, however, has demonstrated how neural networks can be leveraged in the shape deformation not only for improving the fitting accuracy but also for multiple other purposes such as: to fit the source shape to a partial target shape [11] or 2D images [15, 22, 39], to find point-wise correspondences through deformation [8, 9], to predict customized deformation handles for each input shape [46], to cluster shapes given a collection [27], to learn semantic deformations [47], and to transfer deformations [44, 35]. While our approach can also perform target-driven deformation, our main goal is different: to learn a deformation-based conditional generative model. We also remark that our method does not utilize any semantic supervision such as part segmentation, as done by some recent works [35, 45].

3D Shape Generative Models. In light of the success in the 2D image case, deep generative models have also been widely investigated for 3D data. Wu *et al*. [42] was the first proposing a 3D GAN with voxel representation, and

Achlioptas *et al*. [1] and their subsequent work [38, 34] also proposed point-cloud-based GANs. However, these approaches are not able to produce fine-grained geometric details due to the limit of the resolution.While mesh is a preferable representation, generating meshes is very challenging in particular when preventing the generation of non-manifold faces or disconnected components [30]. Hence, previous work, such as the one of Tan *et al*. [37, 36], considers generating novel shapes by deforming a given template mesh, limiting the scope of the generation to the possible variations of the template shape. We propose to overcome this limitation with our *conditional* generative model, which takes any 3D mesh as input to deform. Generative models for 3D shapes have also been investigated to learn possible variations of compositional structures with or without semantic annotations [7, 33, 4, 43, 45, 23, 29]. In this work, we focus on learning geometric variations of the given shape while preserving its topological structure.

3. Method

In this section, we will first briefly review the control-point-based deformation and the biharmonic coordinates [41] technique we use, and introduce how the meta-handles are defined with the control-point handles (Section 3.1). We will then present how we learn the meta-handles in an unsupervised fashion and our neural network architectures (Section 3.2). Lastly, we will introduce our loss functions that guide the emergence of plausible deformations and intuitive factorizations (Section 3.3).

3.1. Biharmonic Coordinates and Meta-Handles

Mesh deformation through directly moving individual vertex is cumbersome and may easily lead to unwanted distortions. We thus leverage deformation handles to parameterize the deformations with a low degree of freedom. The key in the handle-based deformation is to define a proper deformation function that features several desired properties. For instance, no change of handles should result in no deformation; each handle should produce local and smooth deformation; the deformation function should be expressed in closed form. Numerous previous work has introduced different handle-based deformation functions. Many of them are based on solving the *biharmonic* equation defined over the mesh with boundary constraints (given from handles). The resulting deformation functions of these approaches satisfy many desired properties [16, 19]. Also, closed-form expressions with respect to the handles can be easily calculated after a pre-computation. (Please refer to Jacobson *et al*. [17] for more details.)

In our method, we employ a subset of mesh vertices as the deformation handles (*control points*) and restrict the transformations of the handles to pure translations. Given the mesh vertices $\mathbf{V} \in \mathbb{R}^{n\times 3}$ (n vertices) and a set of c control points $\mathbf{C} \in \mathbb{R}^{c\times 3}$, the *linear* map $\mathbf{W} \in \mathbb{R}^{n\times c}$ between them ($\mathbf{V} = \mathbf{WC}$) is often called 'generalized barycentric coordinates' [28, 21, 20, 26]. Wang *et al*. [41] proposed one way to define $\mathbf{W}$ based on the biharmonic functions, which is thus dubbed *biharmonic coordinates*, and we utilize it as our deformation function. Without requiring that control points form a *cage* enclosing the input shape, our deformation handles are flexible and intuitive.

Specifically, we sample c control points from the mesh vertices by farthest point sampling (FPS) over the geodesic distances. The biharmonic coordinates $\mathbf{W}$ are also precomputed. However, the deformation function $f : \mathbb{R}^{c\times 3} \rightarrow \mathbb{R}^{n\times 3}$ defined over the given control points $\mathbf{C}$, $f(\mathbf{C}) = \mathbf{WC}$, has $3c$ degrees of freedom. It may overparameterize the plausible shape variation space, which means there may be lots of implausible deformations, if we randomly translate the control points (see Fig. 2). Also, there may exist strong correlations across the deformations from moving individual control points. For a specific shape (e.g., a chair), all the plausible variants may reside in a lower-dimensional subspace and can be factorized into several meaningful deformation directions (e.g., scaling all chair legs and bending the chair back).

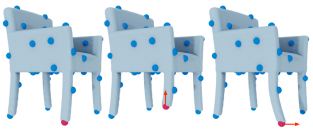

Figure 2: Two deformations resulted from moving the red control point along the arrow directions.

To this end, we propose to find a smaller number of meta-handles to factorize the subspace covering all the plausible deformations. Specifically, each meta-handle $\mathbf{M}_i \in \mathbb{R}^{c\times 3}$ is represented as *offsets* over the c control points:

$$\mathbf{M}_i = [\vec{t}_{i1}, \cdots, \vec{t}_{ic}]^T, \tag{1}$$

where $\vec{t}_{ij} \in \mathbb{R}^3$ indicates the offset of the j-th control point for the i-th meta-handle. In contrast to a single control point that mainly affects a local region of the mesh, each meta-handle is expected to provide a more intuitive deformation direction, which may even correspond to some semantic meanings (See Figs. 1 and 8).

We now use the linear combination of the meta-handles to represent a deformation. Specifically, a new deformation function $g : \mathbb{R}^m \rightarrow \mathbb{R}^{n\times 3}$ is defined with respect to the meta-handles $\{\mathbf{M}_i\}_{i=1\cdots m}$ and their linear combination coefficients $\mathbf{a} = [a_1, \cdots, a_m]$:

$$g\left(\mathbf{a}; \{\mathbf{M}_i\}_{i=1\cdots m}\right) = \mathbf{W}(\mathbf{C}_0 + \sum_{i=1}^{m} a_i \mathbf{M}_i), \tag{2}$$

where $\mathbf{C}_0 \in \mathbb{R}^{c\times 3}$ denotes the rest positions of the given control points. In the context of the conditional generative model, it can be interpreted as that each shape has a m-dimensional input-dependent latent space, where each axis corresponds to a meta-handle describing a specific deformation function in 3D space. A latent code $\mathbf{a}$ can thus be

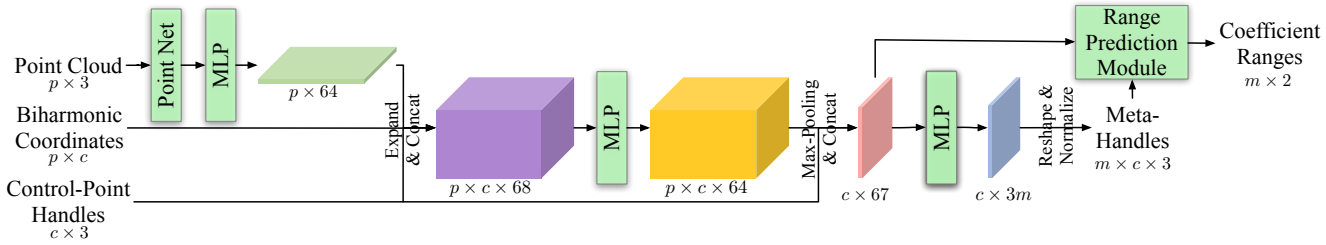

Figure 3: Architecture of MetaHandleNet: it incorporates the information from the shape (point cloud), control-point handles, and biharmonic coordinates by building a 3D tensor, and predicts a set of meta-handles with the corresponding coefficient ranges for the shape.

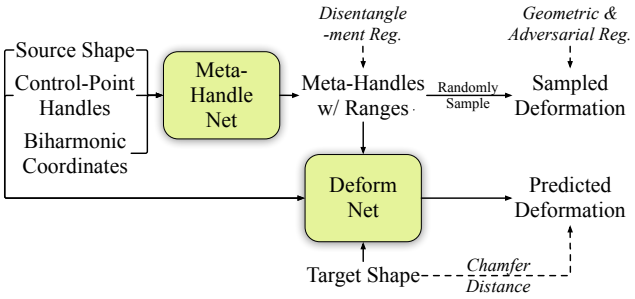

Figure 4: Overview of our method. We learn the meta-handles in an unsupervised fashion.

directly decoded to a deformation of the input mesh as a linear combination of the meta-handles.

Along with the meta-handles, our method also predicts *ranges* $\{[L_i, R_i]\}_{i=1\cdots m}$ of the coefficients associated with each meta-handle. The ranges describe the scope of plausible deformations along the direction of each meta-handle. Any set of coefficients within the *ranges* $\{[L_i, R_i]\}_{i=1\cdots m}$ is thus expected to produce a plausible deformation.

We utilize a small number of meta-handles to learn a low-dimensional compact deformation space. The degrees of freedom of the deformation function g is typically much smaller than that of the deformation function f, i.e., $m \ll 3c$. As a result, the meta-handles are required to not only leverage the correlations of the control-point handles, but also discover the underlying properties of the shape structure (e.g., chair legs are symmetric and should thus be deformed together).

3.2. Network Architecture

We propose to learn the meta-handles in an unsupervised fashion without taking semantic annotations or correspondences across the shapes as input or supervision. As shown in Fig. 4, our method mainly includes three networks: MetaHandleNet, DeformNet, and a discriminator network (discussed in Section 3.3). Taking a pair of randomly sampled shapes within the same category as input, the method predicts a deformation space for the source shape, and finds a deformation within the space to match the target shape. Specifically, MetaHandleNet takes a source shape, its control points, and the precomputed biharmonic coordinates as input and predicts a set of meta-handles as well as the corresponding coefficient ranges. DeformNet then predicts coefficients of the meta-handles so that the resulting deformation of the source shape matches the target shape.

To ease encoding, in MetaHandleNet, we convert the input source mesh to a point cloud (denoted as $\mathbf{P} \in \mathbb{R}^{p\times 3}$) by uniformly sampling p points over the mesh surface. The precomputed biharmonic coordinates are also interpolated from the mesh vertices to the point cloud (i.e., $\mathbf{W} \in \mathbb{R}^{p\times c}$) according to the barycentric coordinates. Fig. 3 illustrates the architecture of MetaHandleNet. It first encodes the point cloud with PointNet [31] and obtains 64-dimensional features per point, which is denoted as $\mathbf{D} \in \mathbb{R}^{p\times 64}$. Then, the point features $\mathbf{D}$, the biharmonic coordinates $\mathbf{W}$, and the rest positions of the control points $\mathbf{C_0} \in \mathbb{R}^{c\times 3}$ are consolidated in a 3D tensor (a purple volume in Fig. 3). Specifically, the 3D tensor has a size of $p \times c \times 68$, and the first $p \times c \times 64$ is packed with the point features $\mathbf{D}$ (repeating for the control points), the next $p \times c \times 1$ is filled with the biharmonic coordinates $\mathbf{W}$, and the last $p \times c \times 3$ is filled with the rest positions of the control points $\mathbf{C_0}$ (repeating for the point cloud). Hence, in this tensor, each pair of a point in $\mathbf{P}$ and a control point has a 68-dimensional feature, which is processed with an MLP. We then aggregate the features across the points through a symmetric function (i.e., max-pooling) to produce 64-dimensional features per control point. The control-point feature is combined again with the rest position of the control point and is then converted to a $3m$-dimensional vector through another MLP, which becomes the offsets for the m meta-handles. We then normalize each metal-handle to unit length to facilitate training. The predicted meta-handles and the 67-dimensional control-point features are then fed into a range prediction module, which outputs a coefficient range $[L_i, R_i]$ for each meta-handle. Please refer to the supplementary materials for the details of the module.

As for the DeformNet, it takes the source shape, target shape, the predicted meta-handles with the coefficient ranges, and the control-point features (extracted from MetaHandleNet) as input, and predicts a coefficient vector $\mathbf{a} \in \mathbb{R}^m$ within the predicted ranges $\Pi_{i=1}^{m}[Li, Ri]$. The pre-

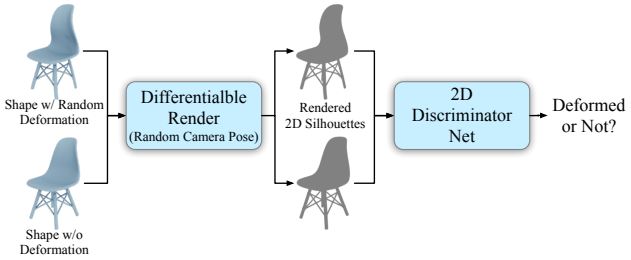

Figure 5: We utilize a soft rasterizer [25] and a 2D discriminator network to penalize unrealistic deformations.

dicted coefficient vector and the meta-handles are then fed into the deformation function g (Equation 2) to decode the deformation for the source shape, which is expected to match the target shape. Similar to MetaHandleNet, DeformNet also builds a 3D tensor to incorporate all the information and utilize shared-weight MLPs and max-pooling to process and aggregate the features. Please refer to the supplementary materials for the details.

3.3. Loss Functions

We consider three objectives when training our network: 1) the deformed input (source) shape matches the given target shape; 2) any deformation sampled from the learned ranges is plausible; 3) the learned meta-handles properly disentangle the deformation space. We thus train our network with the following joint loss function:

$$\mathcal{L} = \mathcal{L}_{fit} + \mathcal{L}_{geo} + \mathcal{L}_{adv} + \mathcal{L}_{disen}. \tag{3}$$

Among the four terms, the fitting loss $\mathcal{L}_{fit}$ corresponds to the first objective and minimizes the Chamfer distance [5] between the deformed source point cloud and the target point cloud.

$\mathcal{L}_{geo}$ and $\mathcal{L}_{adv}$ are geometry loss and adversarial loss, respectively, added for the second objective. In each iteration, we randomly sample a deformation within the predicted ranges, and apply these two losses to penalize implausible deformations.

Specifically, $\mathcal{L}_{geo}$ is further decomposed into:

$$\mathcal{L}_{geo} = \mathcal{L}_{symm} + \mathcal{L}_{nor} + \mathcal{L}_{Lap}, \tag{4}$$

where $\mathcal{L}_{symm}$ is symmetry loss minimizing the Chamfer distance [5] between the deformed point cloud and its reflection along the x-axis (also used in previous works [39, 46]). Given the mesh connectivity, normal loss $\mathcal{L}_{nor}$ and Laplacian loss $\mathcal{L}_{Lap}$ are computed to prevent distortions. $\mathcal{L}_{nor}$ minimizes the angle difference between the face normals of the source mesh and the deformed mesh. $\mathcal{L}_{Lap}$ minimizes $l1$-norm of the difference of Cotangent Laplacian.

It is not enough to guarantee plausible deformation with only geometric regularization. We thus leverage an adversarial loss $\mathcal{L}_{adv}$, which is defined with a soft rasterizer and a 2D discriminator. (A similar adversarial training idea using 2D projection is also introduced by Li *et al*. [24].) As shown in Fig. 5, we feed both randomly deformed shapes and shapes without deformation into a soft rasterizer [25]. The renderer captures a soft silhouette image for each shape from a random view. The images are then fed into a simple 2D convolution neural network to predict whether they come from a deformed shape or not. The 2D discriminator network is jointly trained with MetaHandleNet and DeformNet with a classification loss function. The output probabilities for deformed shapes are used to penalize implausible deformations.

For the third objective, we introduce a disentanglement loss $\mathcal{L}_{disen}$. Inspired by Aumentado-Armstrong *et al*. [2], $\mathcal{L}_{disen}$ is defined with four terms:

$$\mathcal{L}_{disen} = \mathcal{L}_{sp} + \mathcal{L}_{cov} + \mathcal{L}_{ortho} + \mathcal{L}_{SVD}. \tag{5}$$

Specifically, $\mathcal{L}_{sp}$ encourages the meta-handles $\mathbf{M}_i$ and the coefficient vector $\mathbf{a}$ to be sparse by penalizing their $l1$-norm. $\mathcal{L}_{cov}$ penalizes the covariance matrix (calculated for each batch) of the coefficients $\mathbf{a}$. $\mathcal{L}_{ortho}$ encourages meta-handles to cover different parts of the control-point offsets by penalizing "dot products" between the meta-handles. $\mathcal{L}_{SVD}$ encourages the control points to translate in a single direction within each meta-handle. Please refer to the supplementary materials for the details of $\mathcal{L}_{disen}$.

Note that we do not incorporate any explicit loss function for the coefficient ranges. While $\mathcal{L}_{fit}$ motivates the coefficient ranges to expand to cover more plausible deformations, $\mathcal{L}_{geo}$ and $\mathcal{L}_{adv}$ prevent the ranges from excessive expansion by penalizing implausible deformations. The coefficient ranges are thus motivated to learn a trade-off.

4. Experiments

4.1. Target-Driven Deformation

We evaluate our methods on the ShapeNet dataset [3]. We choose 15,522 models from the dataset, which cover three categories: chair, table, and car. Shapes are normalized to fit in a unit sphere. For each shape, we sample $c = 50$ control-point handles by FPS, in order to generally cover most of the surface and allow flexible deformations. We uniformly sample point clouds of the size $p = 4096$ to represent the shapes. We set the number of meta-handles to be $m = 15$. This should be an upper bound since the network can use part of them by setting some ranges to zero. As tetrahedral meshes are required as input to compute the biharmonic weights [40], all the ShapeNet [3] triangular meshes are first fed into Huang *et al*.'s algorithm [14] to become watertight manifolds, and are then fed into TetWild [12] to produce tetrahedral meshes. We use libigl's [18] implementation to compute the biharmonic coordinates, which are then interpolated from the mesh vertices to the sampled point cloud. For the differentiable renderer, we use an implementation from Pytorch3D [32]. We reserve 10% models for testing and the rest for training. For

Figure 6: Qualitative comparison of our method with other deformation methods [13, 39, 10, 46]. Our method allows flexible deformation and fine-grained detail preservation. Our results are also more plausible, especially when the source-target pairs do not share the same structures (see the second and the fourth columns). Please zoom in for details.

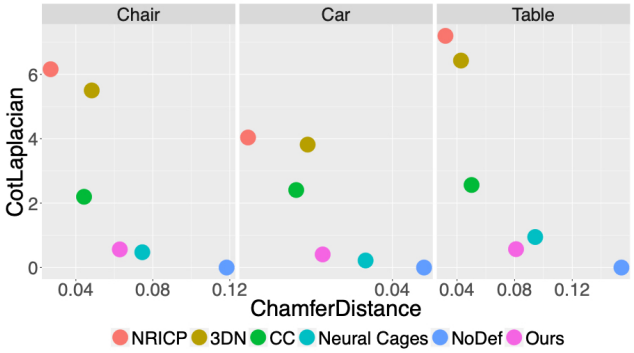

Figure 7: Quantitative comparison of the target-driven deformation. Each 2D point represents one method. The coordinates correspond to the alignment error and the distortion, with the origin being ideal. 'NoDef' indicates undeformed source shapes.

each category, we train a separate model and test it on $3,000$ randomly sampled source-target pairs.

We compare our method to non-rigid ICP (NRICP) [13], a non-neural registration technique which aligns two point clouds by minimizing a smooth deformation energy; 3D deformation network (3DN) [39] and cycle-consistent deformation (CC) [10], two learning-based methods that directly infer per-vertex displacements; and Neural Cages [46], a learnable cage-based deformation method.

Qualitative results are shown in Fig. 6. Although NRICP [13], 3DN [39], and CC [10] do align the source shape to the target shape in most cases, they fail to preserve fine-grained details of the source shape and introduce lots of distortions. The results of Neural Cages [46] look more pleasing, but the cage-based deformation is less flexible than our control-point based deformation. Compared to the Neural Cages [46], our method can achieve more detailed deformation of a local region, such as adjusting the thickness of chair seats (first and fifth columns) and armrests' height (third column). Also, most alternative methods produce unrealistic deformations when the source shape and the target shape do not share similar structures. For example, suppose the source shape has four chair legs, and the target shape is a swivel chair (second and fourth columns). In that case, the alternative methods tend to deform the four chair legs toward the center under the fitting loss's influence, resulting in undesirable deformations. Thanks to the adversarial regularization we employed, our method can avoid such implausible deformations while still aligning the output to the target.

Inspired by Neural Cages [46], we also utilize Chamfer distance [5] between the deformed shape and the target shape (computed over 100,000 uniformly sampled points) to measure the alignment error; and use the difference between cotangent Laplacians of the source shape and the deformed shape ($l1$-norm) to measure the distortion. The

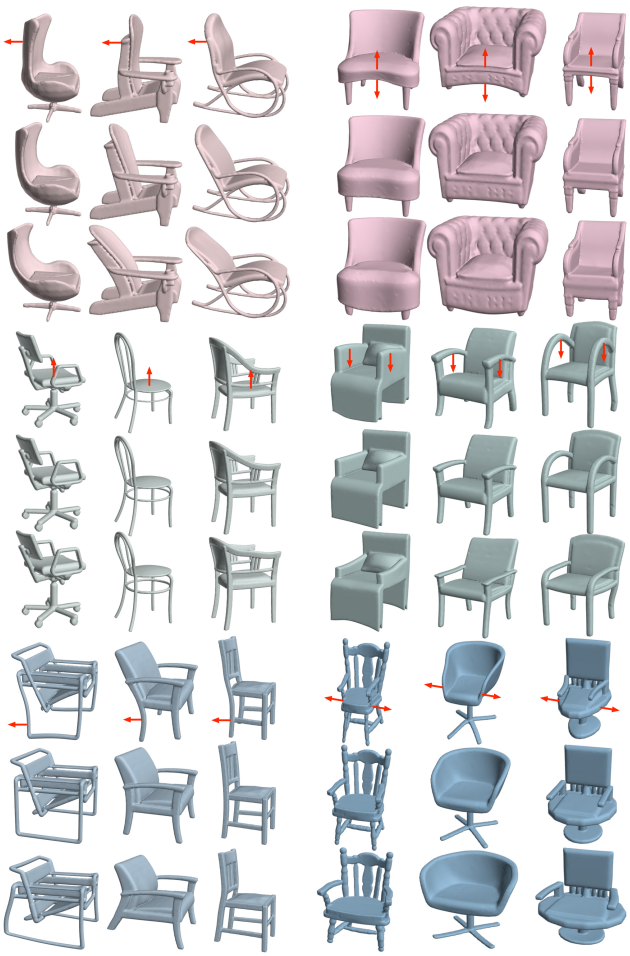

Figure 8: Learned meta-handles across different shapes. The figure includes six meta-handles, and each color indicates a distinct one. For each meta-handle, the figure demonstrates the corresponding deformations on three different shapes, with the red arrows highlighting the deformation direction. The meta-handles are consistent across various shapes.

quantitative results are shown in Fig. 7. As shown in the figure, although NRICP [13], 3DN [39], and CC [10] achieve lower alignment errors, the distortions are much higher. Compared to Neural Cages [46], our method achieves better Chamfer distance with a similar cotangent Laplacian.

4.2. Meta-Handle Deformation Space

Another main contribution of our method is that, for each shape, we learn a set of interpretable meta-handles with the corresponding coefficient ranges, which factorize all the plausible deformations for the shape.

Fig. 1 demonstrates some learned meta-handles of a single shape. Each column shows the deformations along the direction of a meta-handle, with the deformation scale uniformly sampled within the corresponding coefficient range. The red arrows highlight the deformation direction of each meta-handle. As shown in the figure, the learned meta-handles are disentangled and factorize all the plausible deformations for the shape. Although we do not take any semantic annotation or correspondences across different shapes as input or supervision, our method is able to learn some intuitive meta-handles. Specifically, the learned meta-handles are not limited to global scaling. Many of them align with some local semantic parts, such as adjusting the thickness of the chair seat (first column), the height of armrests (fourth column), the length of four chair legs (seventh column), and the height of the chair back (eighth column). Also, many of them involve non-rigid deformation of some parts, such as bending the chair back (fifth column) and two back legs (sixth column), which cannot be achieved through the rigid bounding-box handles proposed by previous methods [6, 35]. To construct a low-dimensional compact deformation space, the learned meta-handles not only leverage correlations between the control-point handles, but also discover the underlying hard constraints (e.g., symmetry) of the shape structure. Meanwhile, the coefficient ranges learn the underlying soft priors (e.g., ratios of part scales) and provide reasonable deformation scopes for meta-handles.

Table 1: Coverage (higher is better) and MMD ($\times 100$, lower is better) comparison between different methods.

	Chair		Car		Table	
	COV ↑	MMD ↓	COV ↑	MMD ↓	COV ↑	MMD ↓
3DN [39]	32.0%	4.56	46.6%	2.91	30.6%	4.26
CC [10]	51.0%	4.26	50.3%	2.79	50.2%	3.88
NC [46]	54.4%	4.23	66.6%	2.65	44.7%	3.85
Ours	64.6%	4.28	76.5%	2.97	54.9%	3.70

We assume that, for different shapes, meta-handles with the same index share similar deformations due to the structure feature of MetaHandleNet. As shown in Fig. 8, our learned meta-handles are consistent across different shapes. Despite geometry details and even global structures being different, each meta-handle can find corresponding regions across various shapes and predict similar deformations, which is interesting as we do not provide any semantic annotation or correspondence information.

Inspired by Achlioptas *et al*. [1], we also employ coverage (COV) and minimum matching distance (MMD) to evaluate our generative model. For a set of generated shapes A and a set of ground truth shapes B, coverage measures the *fraction* of the shapes in B that can be roughly represented within A, while MMD measures *how well* shapes in B can be represented by shapes in A. For both metrics, closeness is computed using Chamfer distance [5]. For each category, we separate 500 shapes for constructing the set A, and the remaining shapes are regarded as set B. For our method, we randomly sample 20 deformations within the learned deformation space of each shape. For baseline methods 3DN [39], CC [10], and Neural Cages [46], we randomly sample 20 target shapes for each shape in A to generate target-driven deformations. The quantitative re-

Table 2: Chamfer distance ($\times 100$) and Cotangent Laplacian ($\times 10$) between different ablated versions (on chair category). For both metrics, lower is better. DoF indicates degrees of freedom.

Meta-handle / Handle	DoF	$\mathcal{L}_{adv}$	CD↓	CotLap↓
Handle	50×3	w/o	4.78	5.60
Meta-handle	15	w/o	5.76	8.61
Handle	50×3	w/	7.98	7.69
Meta-handle	15	w/	6.28	5.75

sults are shown in Table 1. While all the methods have similar MMDs, our method achieves higher coverages, which indicates that our method generates more diverse deformations, and more ground truth shapes can thus be represented within our deformation space.

4.3. Ablation Studies

Meta-Handles. Instead of predicting a set of meta-handles, we can deform a shape by directly predicting the offset of each control-point handle (deformation function f). We compare our method to this variant. As shown in Table 2, when there is no adversarial loss (first and second rows), directly using 50 control-point handles can achieve better fitting error and smaller distortion, since it allows more degrees of freedom for the deformation. However, when applied with the adversarial loss (third and fourth rows), it is harder for the network to find plausible deformations based on 50 control-point handles, while our learned meta-handles provide intuitive deformations resulting in better results. Also, without meta-handles, we cannot directly sample plausible variants of a input shape. The target-driven deformation is less effective in generating diverse deformations and covering all the plausible variants (see the first row of Table 3).

Table 3: Coverage (higher is better) and MMD ($\times 100$, lower is better) for different ablated versions (on Chair category).

	COV↑	MMD↓
w/o meta-handle	48.4%	4.69
w/o $\mathcal{L}_{adv}$	56.3%	4.64
w/o $\mathcal{L}_{disen}$	64.1%	4.14
Ours	64.6%	4.28

Adversarial Regularization. We use both adversarial loss $\mathcal{L}_{adv}$ and normal loss $\mathcal{L}_{nor}$ (part of the geometric loss $\mathcal{L}_{geo}$) to encourage plausible deformations. Fig. 9 demonstrates a qualitative comparison between them. When there is no $\mathcal{L}_{adv}$, the deformation may lose plausibility in order to match the target shape. Although $\mathcal{L}_{nor}$ can also alleviate this issue to some extent, strong $\mathcal{L}_{nor}$ (fourth column) may be too restrictive for the deformation, while $\mathcal{L}_{adv}$ achieves more realistic results and still allows flexible deformations. When $\mathcal{L}_{adv}$ is applied, the fitting error increases (second and fourth row of Table 2) in exchange for more plausible deformations. As shown in Table 3, without $\mathcal{L}_{adv}$, both the coverage and MMD become worse, indicating that $\mathcal{L}_{adv}$ is important for generating diverse and realistic deformations.

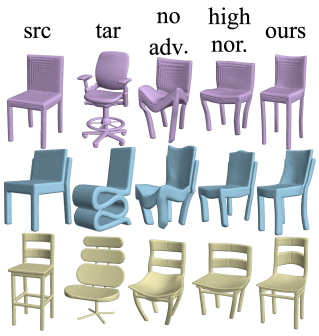

Figure 9: Comparison between $\mathcal{L}_{nor}$ and $\mathcal{L}_{adv}$. Both the third column and the fourth column have no $\mathcal{L}_{adv}$, but the fourth column has higher weight for $\mathcal{L}_{nor}$.

Disentanglement Regularization. We use $\mathcal{L}_{disen}$ to encourage the intuitive factorization of the deformation space. As shown in Fig. 10, when there is no $\mathcal{L}_{disen}$, the deformations along each learned meta-handle are still plausible, since $\mathcal{L}_{geo}$ and $\mathcal{L}_{adv}$ are still applied to the random samples within the space to penalize unrealistic deformations. However, the learned meta-handles are entangled, each meta-handle may deform multiple parts along different directions, and there are overlappings between different meta-handles. In contrast, meta-handles in Fig. 1 provide more intuitive and disentangled deformations. Table 3 quantitatively verifies that $\mathcal{L}_{disen}$ does not affect the diversity and plausibility of the deformation space.

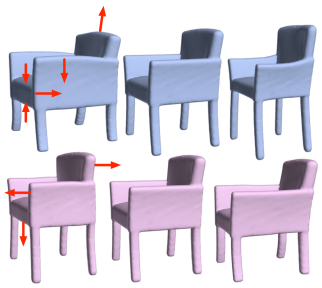

Figure 10: Results w/o $\mathcal{L}_{disen}$, each row indicates a learned meta-handle.

5. Conclusion

We presented **DeepMetaHandles**, a 3D conditional generative model based on mesh deformation. Our method takes automatically-generated control points with biharmonic coordinates as deformation handles, and learns a latent space of deformation for each input mesh. Each axis of the space is explicitly associated with multiple deformation handles, and it's thus called a meta-handle. The disentangled meta-handles factorize all the plausible deformations of the shape, while each of them conforms to an intuitive deformation. We learn the meta-handles unsupervisely by incorporating a target-driven deformation module. We also employ a differentiable render and a 2D discriminator to enhance the plausibility of the deformation.

In our method, the expressibility of the deformation is limited by the given control points. Technically, increasing the number of input control points a lot will result in a memory issue and making the network training more difficult. An interesting future direction would be developing another network that can adaptively sample the control points at appropriate locations and thus enable more fine-grained local deformations.

Acknowledgments. This work is supported in part by gifts from Adobe, Kwai, Qualcomm, and Vivo.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *ICML*, 2018. 3, 7

[2] T. Aumentado-Armstrong, S. Tsogkas, A. Jepson, and S. Dickinson. Geometric disentanglement for generative latent shape models. In *ICCV*, 2019. 5

[3] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015. 5

[4] Anastasia Dubrovina, Fei Xia, Panos Achlioptas, Mira Shalah, Raphael Groscot, and Leonidas Guibas. Composite shape modeling via latent space factorization. In *ICCV*, 2019. 3

[5] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017. 5, 6, 7

[6] Matheus Gadelha, Giorgio Gori, Duygu Ceylan, Radomir Mech, Nathan Carr, Tamy Boubekeur, Rui Wang, and Subhransu Maji. Learning generative models of shape handles. In *CVPR*, 2020. 7

[7] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. Sdm-net: Deep generative network for structured deformable mesh. In *ACM SIGGRAPH Asia*, 2019. 3

[8] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. 3D-CODED: 3D correspondences by deep deformation. In *ECCV*, 2018. 2

[9] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Deep self-supervised cycle-consistent deformation for few-shot shape segmentation. In *Eurographics Symposium on Geometry Processing*, 2019. 1, 2

[10] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Deep self-supervised cycle-consistent deformation for few-shot shape segmentation. In *Eurographics Symposium on Geometry Processing*, 2019. 6, 7

[11] Rana Hanocka, Noa Fish, Zhenhua Wang, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. ALIGNet: Partial-Shape agnostic alignment via unsupervised learning. *ACM Transactions on Graphics*, 2018. 2

[12] Yixin Hu, Qingnan Zhou, Xifeng Gao, Alec Jacobson, Denis Zorin, and Daniele Panozzo. Tetrahedral meshing in the wild. In *ACM SIGGRAPH*, 2015. 5

[13] Haibin Huang, Evangelos Kalogerakis, Siddhartha Chaudhuri, Duygu Ceylan, Vladimir G. Kim, and Ersin Yumer. Learning local shape descriptors from part correspondences with multiview convolutional networks. *ACM Transactions on Graphics*, 2017. 2, 6, 7

[14] Jingwei Huang, Hao Su, and Leonidas Guibas. Robust watertight manifold surface generation method for ShapeNet models, 2018. 5

[15] Dominic Jack, Jhony K. Pontes, Sridha Sridharan, Clinton Fookes, Sareh Shirazi, Frederic Maire, and Anders Eriksson. Learning free-Form deformations for 3D object reconstruction. In *ICCV*, 2018. 2

[16] Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. In *ACM SIGGRAPH*, 2011. 3

[17] Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. Skinning: Real-time shape deformation. In *ACM SIGGRAPH 2014 Courses*, 2014. 3

[18] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library, 2018. https://libigl.github.io/. 5

[19] Alec Jacobson, Tino Weinkauf, and Olga Sorkine. Smooth shape-aware functions with controlled extrema. In *Eurographics Symposium on Geometry Processing*, 2012. 3

[20] Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. In *ACM SIGGRAPH*, 2007. 3

[21] Tao Ju, Scott Schaefer, and Joe Warren. Mean value coordinates for closed triangular meshes. In *ACM SIGGRAPH*, 2005. 3

[22] Andrey Kurenkov, Jingwei Ji, Animesh Garg, Viraj Mehta, JunYoung Gwak, Christopher Bongsoo Choy, and Silvio Savarese. DeformNet: Free-Form deformation network for 3d shape reconstruction from a single image. In *WACV*, 2018. 2

[23] Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas Guibas. Grass: Generative recursive autoencoders for shape structures. In *ACM SIGGRAPH*, 2017. 3

[24] Xiao Li, Yue Dong, Pieter Peers, and Xin Tong. Synthesizing 3d shapes from silhouette image collections using multi-projection generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5535–5544, 2019. 5

[25] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *ICCV*, 2019. 2, 5

[26] Josiah Manson and Scott Schaefer. Moving least squares coordinates. In *Eurographics Symposium on Geometry Processing*, 2010. 3

[27] Eloi Mehr, Ariane Jourdan, Nicolas Thome, Matthieu Cord, and Vincent Guitteny. DiscoNet: Shapes learning on disconnected manifolds for 3d editing. In *ICCV*, 2019. 2

[28] Mark Meyer, Alan Barr, Haeyoung Lee, and Mathieu Desbrun. Generalized barycentric coordinates on irregular polygons. *J. Graph. Tools*, 2002. 3

[29] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J. Mitra, and Leonidas Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. In *ACM SIGGRAPH Asia*, 2019. 3

[30] Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. PolyGen: An autoregressive generative model of 3d meshes. In *ICML*, 2019. 3

[31] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, 2017. 4

[32] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d, 2020. 5

[33] Nadav Schor, Oren Katzir, Hao Zhang, and Daniel Cohen-Or. CompoNet: Learning to generate the unseen by part synthesis and composition. In *ICCV*, 2019. 3

[34] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *ICCV*, 2019. 3

[35] Minhyuk Sung, Zhenyu Jiang, Panos Achlioptas, Niloy J. Mitra, and Leonidas J. Guibas. DeformSyncNet: Deformation transfer via synchronized shape deformation spaces, 2020. 1, 2, 7

[36] Qingyang Tan, Lin Gao, Yu-Kun Lai, Jie Yang, and Shihong Xia. Mesh-based autoencoders for localized deformation component analysis. In *AAAI*, 2018. 1, 3

[37] Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Meshvae: Variational autoencoders for deforming 3d mesh models. In *CVPR*, 2018. 1, 3

[38] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph convolution. In *ICLR*, 2019. 3

[39] Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. 3dn: 3d deformation network. In *CVPR*, 2019. 1, 2, 5, 6, 7

[40] Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. Linear subspace design for real-time shape deformation. 2015. 5

[41] Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. Linear subspace design for real-time shape deformation. In *ACM SIGGRAPH*, 2015. 2, 3

[42] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NeurIPS*, 2016. 2

[43] Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. PQ-NET: A generative part seq2seq network for 3D shapes. In *CVPR*, 2020. 3

[44] Jie Yang, Lin Gao, Yu-Kun Lai, Paul L. Rosin, and Shihong Xia. Biharmonic deformation transfer with automatic key point selection. *Graphical Models*, 2018. 2

[45] Jie Yang, Kaichun Mo, Yu-Kun Lai, Leonidas J. Guibas, and Lin Gao. DSM-Net: Disentangled structured mesh net for controllable generation of fine geometry, 2020. 2, 3

[46] Wang Yifan, Noam Aigerman, Vladimir Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. Neural cages for detail-preserving 3d deformations. In *CVPR*, 2020. 1, 2, 5, 6, 7

[47] Ersin Yumer and Niloy J. Mitra. Learning semantic deformation flows with 3d convolutional networks. In *ECCV*, 2016. 2

[48] Keyang Zhou, Bharat Lal Bhatnagar, and Gerard Pons-Moll. Unsupervised shape and pose disentanglement for 3d meshes. In *European Conference on Computer Vision*, pages 341–357. Springer, 2020. 1