Deep Implicit Templates for 3D Shape Representation

Zerong Zheng Tao Yu Qionghai Dai Yebin Liu

Department of Automation, Tsinghua University, Beijing, China

Abstract

Deep implicit functions (DIFs), as a kind of 3D shape representation, are becoming more and more popular in the 3D vision community due to their compactness and strong representation power. However, unlike polygon mesh-based templates, it remains a challenge to reason dense correspondences or other semantic relationships across shapes represented by DIFs, which limits its applications in texture transfer, shape analysis and so on. To overcome this limitation and also make DIFs more interpretable, we propose Deep Implicit Templates, a new 3D shape representation that supports explicit correspondence reasoning in deep implicit representations. Our key idea is to formulate DIFs as conditional deformations of a template implicit function. *To this end, we propose Spatial Warping LSTM, which decomposes the conditional spatial transformation into multiple point-wise transformations and guarantees generalization capability. Moreover, the training loss is carefully designed in order to achieve high reconstruction accuracy while learning a plausible template with accurate correspondences in an unsupervised manner. Experiments show that our method can not only learn a common implicit template for a collection of shapes, but also establish dense correspondences across all the shapes simultaneously without any supervision.*

1. Introduction

Representing 3D objects effectively and efficiently in neural networks is fundamental for many tasks in computer vision, including 3D model reconstruction, matching, manipulation and understanding. In the pioneering studies, researchers have adopted various traditional geometry representations, including voxel grids [53, 55, 52, 54, 23, 46], point clouds [62, 1, 58, 33] and meshes [63, 2, 50, 51, 22, 14, 5]. In the past several years, deep implicit functions (DIFs) have been proposed as an alternative [40, 38, 11, 18, 20, 56, 26, 9, 24, 47, 15]. Compared to traditional representations, DIFs show expressive and flexible capacity for representing complex shapes and fine geometric details, even in challenging tasks like human digitization [43].

Unfortunately, the implicit nature of DIFs is also its Achilles' Heels: although DIFs are good at approximating individual shapes, they provide no information about the relationship between two different ones. One can easily establish vertice-to-vertice correspondences between two shapes when using mesh templates [21, 50, 14], but that is difficult in DIFs. The lack of semantic relationship in DIFs poses significant challenges for using DIFs in downstream applications such as shape understanding and editing.

To overcome this limitation, we propose *Deep Implicit Templates*, a new way to interpret and implement DIFs. The key idea is to decompose a conditional deep implicit function into two components: a template implicit function and a conditional spatial warping function. The template implicit function represents the "mean shape" for a category of objects, while the spatial warping function deforms the template implicit function to form specific object instances. On one hand, as both the template and the warping field are defined in an implicit manner, the advantages of deep implicit representations (compactness and efficiency) are preserved. On the other hand, with the template implicit function as an intermediate shape, the warping function automatically establishes dense correspondences across different object.

More recently, some techniques use a set of primitives to represent 3D shapes in order to capture structure-level semantics [19, 18, 22, 14]. The primitives can be either manually defined [19, 22] or learned from data [18, 14]. We emphasize that our method is essentially different from them in two ways. First, our method decomposes the implicit representations into a template implicit function and a continuous warping field. Compared to element-based methods, our decomposition not only provides a complete, global template for the training data, enabling many applications such as uv mapping and keypiont labeling, but also makes the latent shape space more interpretable as we can inspect how shapes deform. Second, our method directly builds up accurate correspondences in the whole 3D space, while element-based methods rely on interpolation or feature matching to compute dense correspondences. Overall, our method provides more flexibility and scalability to control the template and/or its deformation: one can, for example, replace the template in our framework with a custom designed one without losing the representation power, or

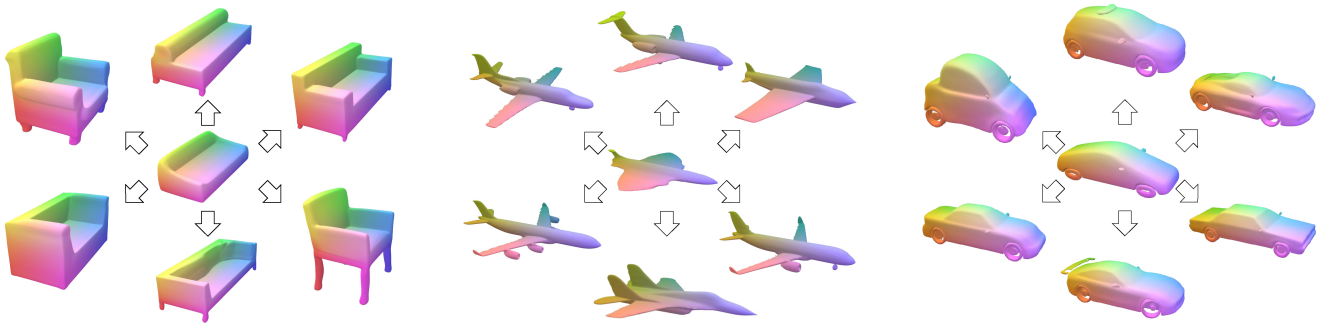


Figure 1: Example results of our representation. Our approach is able to factor out a plausible template (middle) from a set of shapes (surroundings), and builds up dense correspondences (color-coded) across all the shapes automatically without any supervision.

apply additional semantic constraints on the spatial deformation for specific problems such as dynamic human modeling (Sec.6).

Training deep implicit templates is not straight-forward, because we have no access to either the ground-truth mapping between the templates and shape instances, or dense correspondence annotations across different shapes. Our ultimate goal is to make deep implicit templates an effective representation that can: 1) represent training shapes accurately, 2) establish plausible correspondences across shapes and 3) generalize to unseen data. However, without proper design and regularization, the network may not be able to learn such a representation in an unsupervised manner. We make several technical contributions to address these challenges. In terms of network architecture, we propose *Spatial Warping LSTM*, which decomposes the conditional spatial warping into multi-step point-wose transformation, guaranteeing the generalization capacity and the representation power of our warping function. In addition, we introduce *a progressive reconstruction loss* for our Spatial Warping LSTM, which further improves the reconstruction accuracy. *Two-level regularization* is also proposed to obtain plausible templates with accurate correspondences in an unsupervised manner. As shown in the experiments, our method can learn a plausible implicit template for a set of shapes, with conditional warping fields that accurately represent shapes while establishing dense correspondences among them without any supervision (See Fig.1). Overall, the proposed Deep Implicit Templates significantly expands the capability of DIFs without losing its advantages, making it a more effective implicit representation for 3D learning tasks. Code is available at `https://github.com/ZhengZerong/DeepImplicitTemplates`.

2. Related Work

Deep Implicit Functions (DIFs). Implicit functions represent shapes by constructing a continuous volumetric field and embedding shapes as its iso-surface [7, 45, 49]. In recent years, implicit functions have been introduced into neural networks [40, 38, 11, 20, 56, 43, 26, 9, 24, 47, 15] and show promising results. For example, DeepSDF [40] proposed to learn an implicit function where the network output represents the signed distance of the point to its nearest surface. Other approaches defined the implicit functions as 3D occupancy probability and turned shape representation into a point classification problem [38, 11, 56]. Some latest studies proposed to blend multiple local implicit functions in order to generalize to more complex scenes as well as to capture more geometric details [26, 9, 47]. The training loss used in DeepSDF is also improved for more accurate reconstruction [15]. DualSDF [24] extended DeepSDF by introducing a coarse layer to support shape manipulation. Occupancy Flow [39], from another aspect, extended 3D occupancy functions into 4D domains, but this method is restricted to represent temporally continuous 4D sequences. Overall, implicit functions are promising for representing complex shapes and detailed surfaces, but it remains difficult to reason dense correspondences between different shapes represented by DIFs. In contrast, our method and other concurrent works [35, 13] overcomes this limitation and expands the capability of DIFs by introducing dense correspondences across shapes into DIFs.

Elementary Structures. Elementary representations, which aim to describe complex shapes uing a collection of simple shape elements, have been extensively studied for many years in computer vision and graphics [27, 34, 44, 25]. In this direction, previous methods usually require complicated non-convex optimization for primitive fitting. In order to improve the efficiency and effectiveness, various deep learning techniques were adopted, such as recurrent networks [64], differential model estimation [32] and unsupervised training losses [48]. Some recent approaches introduced more complex shape elements, such as multiple charts [5], part segmentations [31, 16, 41], axis-aligned 3D Gaussians [19], local deep implicit functions [18], superquadrics [42], convex decomposition [12], box-homeomorphic parts [17] or learnable shape primitives [14, 29, 48, 61]. Although elementary representation is compact, it is challenging to produce consistent fitting across different shapes. In addition, local elements cannot

be used in many tasks (e.g., shape completion) due to the lack of global knowledge.

Template Learning. Fitting global shape templates instead of local primitives has also attracted a lot of research efforts, as mesh-based templates are able to efficiently represent similar shapes such as articulated human bodies [36, 4, 3, 65]. They are very popular in many data-driven 3D reconstruction studies. For example, articulated human body templates, such as SMPL [36], are now widely used in a lot of human modeling studies [28, 30, 63, 2]. Ellipsoid meshes as a much simpler template can also be deformed to represent various objects [50, 51]. Given a predefined template, some recent techniques like 3D-Coded [21] learned to perform shape matching in unsupervised manner. Although using mesh-based templates is convenient, mesh-based templates are unable to deal with topological changes and require dense vertices to recover surface details [50]. Moreover, when representing shapes that are highly different from the template, mesh-based templates suffer from deformation artifacts (e.g., triangle intersection, extreme distortion, etc.) due to large number of degrees of freedom for mesh vertex coordinates. In contrast, the template in our method is defined in an implicit manner, thus possessing more powerful representation capacity than mesh-based templates. To the best of our knowledge, our work is the first one to learn implicit function-based templates for a collection of shapes.

3. Overview

Our Deep Implicit Template representation is designed on the basis of DeepSDF [40], which is a popular DIF-based 3D shape representation. In this section, we first review DeepSDF for clarity and then describe the overall framework of our approach.

3.1. Review of DeepSDF

The DeepSDF representation defines a surface as the level set of a signed distance field (SDF) $\mathcal{F}$, e.g. $\mathcal{F}(\boldsymbol{p}) = 0$, where $\boldsymbol{p} \in \mathbb{R}^3$ denotes a 3D point and $\mathcal{F}\colon \mathbb{R}^3 \mapsto \mathbb{R}$ is a function approximated using a deep neural network. In practice, in order to represent multiple object instances using one neural network, the function $\mathcal{F}$ also takes a condition variable $\boldsymbol{c}$ as input and thus can be written as:

$$\mathcal{F}(\boldsymbol{p}, \boldsymbol{c}) = s\colon \boldsymbol{p} \in \mathbb{R}^3, \boldsymbol{c} \in \mathcal{X}, s \in \mathbb{R} \tag{1}$$

where $\boldsymbol{c} \in \mathcal{X}$ is the condition variable that encodes the shape of a specific object and can be custom designed in accordance of applications [11, 40, 38, 43]. With this SDF represented by $\mathcal{F}$, the object surface can be extracted using Marching Cube [37]. In DeepSDF [40], the condition variable $\boldsymbol{c}$ is a high-dimensional latent code and each shape instance has a unique code. All latent codes are firstly initialized with Gaussian noise and then optimized in parallel with network training.

3.2. Deep Implicit Templates

In DeepSDF, the shape variance is directly represented by the changes of SDFs themselves. Different from this formulation, we think that, given a category of shapes represented by SDFs, their shape variance can be reflected by the differences of these SDFs relative to a template SDF that captures their common structure. This key idea leads to our formulation of *Deep Implicit Templates*, which decomposes the conditional signed distance function $\mathcal{F}$ into $\mathcal{F} = \mathcal{T} \circ \mathcal{W}$, i.e.,

$$\mathcal{F}(\boldsymbol{p}, \boldsymbol{c}) = \mathcal{T}(\mathcal{W}(\boldsymbol{p}, \boldsymbol{c})) \tag{2}$$

where $\mathcal{W} : \mathbb{R}^3 \times \mathcal{X} \mapsto \mathbb{R}^3$ maps the coordinate of $\boldsymbol{p}$ to a new 3D coordinate, while $\mathcal{T} : \mathbb{R}^3 \mapsto \mathbb{R}$ outputs the signed distance value at this new 3D coordinate.

Intuitively, $\mathcal{W}$ is a conditional spatial warping function that warps the input points according to the latent code $\boldsymbol{c}$, while the function $\mathcal{T}$ itself is an implicit function representing a common SDF which is irrelevant to $\boldsymbol{c}$. Therefore, for a set of objects, the shape represented by $\mathcal{T}(\cdot)$ can be regarded as their common template SDF; in the following context we call it an *implicit template*. To query the signed distance at $\boldsymbol{p}$ for a specific object defined by $\boldsymbol{c}$, the spatial warping function $\mathcal{W}$ first transforms $\boldsymbol{p}$ to its *canonical position* in the implicit template, followed by $\mathcal{T}$ querying its signed distance. In other words, the implicit template is warped according to the latent codes to model different SDFs. [1]

Compared to the original formulation in Eqn.(1), the main advantage of the decomposition in Eqn.(2) is that it naturally induces correspondences between the implicit template and object instances, and accordingly, correspondences across different object instances. As shown in Sec.6, this feature offers more possibility for applying deep implicit functions in many applications.

However, implementing and training the decomposed network is not straight-forward. Specifically, without proper design and regularization, the network tends to overfit to a complicated transformer with an over-simplified implicit template, which further result in inaccurate correspondences. Our goal is to learn an optimal template that can represent the common structure for a set of objects, together with a spatial transformer that establishes accurate dense correspondences between the template and the object instances. Moreover, the learned Deep Implicit Templates should also preserve the representation power and the generalization capacity of DeepSDF, and thus support mesh interpolation and shape completion. In Sec.4 we will discuss how we achieve these goals.

[1] In the strict sense of the term, a warping of an SDF is not an SDF in general. However, we adopt two constraints to make sure that the warped SDF can still approximate the target SDF: (1) we use truncated SDF that only varies in a small band near the surface, and (2) we normalize all the meshes into the same scale. Therefore, we loosely use the term "SDF" in this paper and regard a warping of an SDF as another SDF.

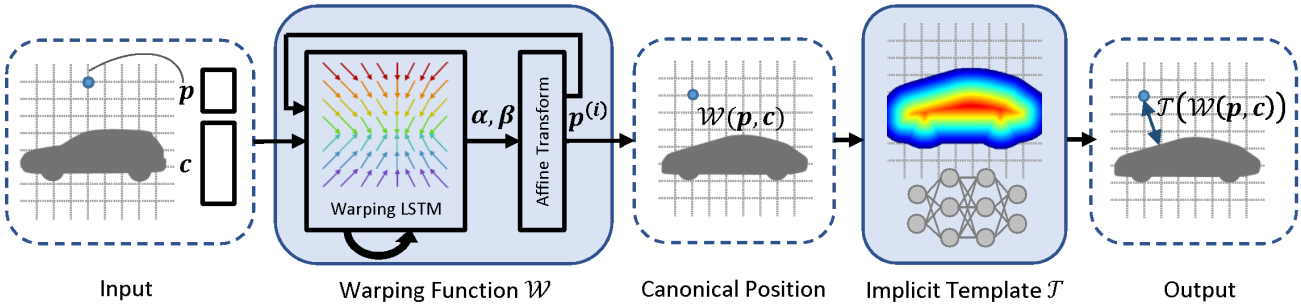


Figure 2: Method overview. Our method decomposes the DIF representation into a warping function and an implicit template. The warping function transforms point samples of shape $\boldsymbol{c}$ to their canonical positions, which are then mapped to SDF values by the implicit template.

4. Methodology

4.1. Network Architecture

Similar to DeepSDF [40], we implement our implicit template, $\mathcal{T}$ in Eqn.(2), using a fully-connected network. For the spatial warping function $\mathcal{W}$, we empirically found that an MLP implementation leads to unsatisfactory results (Sec.5.4). To deal with this challenge, we introduce a Spatial Warping LSTM, which decomposes the spatial transformation for a point $\boldsymbol{p}$ into multi-step point-wise transformations:

$$(\boldsymbol{\alpha}^{(i)}, \boldsymbol{\beta}^{(i)}, \boldsymbol{\phi}^{(i)}, \boldsymbol{\psi}^{(i)}) = \text{LSTMCell}(\boldsymbol{c}, \boldsymbol{p}^{(i-1)}, \boldsymbol{\phi}^{(i-1)}, \boldsymbol{\psi}^{(i-1)}), \tag{3}$$

where $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ are the output and cell states, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the transformation parameters, and the superscript (i) means the results of the i-th step. The position of $\boldsymbol{p}$ is updated as:

$$\boldsymbol{p}^{(i)} = \boldsymbol{p}^{(i-1)} + (\boldsymbol{\alpha}^{(i)} \odot \boldsymbol{p}^{(i-1)} + \boldsymbol{\beta}^{(i)}). \tag{4}$$

where $\odot$ means element-wise product and $\boldsymbol{p}^{(i)} = \boldsymbol{p}$. We iterate this process in Eqn.(3-4) for S steps (in all experiments we set $S = 8$), and the point coordinate at the final step yields the output of the warping function $\mathcal{W}$:

$$\mathcal{W}(\boldsymbol{p}, \boldsymbol{c}) = \boldsymbol{p}^{(S)} \tag{5}$$

This multi-step formulation takes inspiration from the iterative error feedback (IEF) loop [8], where progressive changes are made recurrently to the current estimate. The difference is that our LSTM implementation allows us to aggregate information from multiple previous steps while IEF makes independent estimations in each step. The network architecture is illustrated in Fig.2.

4.2. Network Training

The training loss for our network is composed of two components, a reconstruction loss and a regularization loss:

$$\mathcal{L} = \mathcal{L}_{rec} + \mathcal{L}_{reg} \tag{6}$$

Below we will discuss them in details.

s	1	2	3	4	5	6	7	8
ϵ_s	-	0.025	-	0.01	-	0.0025	-	0
λ_s	-	0	-	0.1	-	0.2	-	0.5

Table 1: Parameter settings of the progressive reconstruction loss for different warping steps. For efficiency, we only construct loss every other steps.

4.2.1 Progressive Reconstruction Loss

As we decompose the spatial warping field into multiple steps of point-wise transformation using our Spatial Warping LSTM, we expect that the network learns a progressive warping function, which starts from obtaining smooth shape approximations and then gradually strives for more local details. To this end, we take inspiration from the shape curriculum in [15] and impose a *progressive reconstruction loss* upon the outputs of our network for different numbers of warping steps, aiming that the network recovers more geometric details when taking more transformation steps. Mathematically, the loss term for the outputs with s warping steps is defined as:

$$\mathcal{L}_{rec}^{(s)} = \sum_{k=1}^{K} \sum_{i=1}^{N} L_{\epsilon_s, \lambda_s} \left(\mathcal{T}\left(\boldsymbol{p}^{(s)}\right), v_{k,i} \right) \tag{7}$$

where $\boldsymbol{p}^{(s)}$ is defined as in Eqn.4, $v_{k,i}$ the ground-truth SDF value of $\boldsymbol{p}_i$ for the k-th shape, N the number of SDF samples for one shape and K the number of shapes. $L_{\epsilon,\lambda}(\cdot,\cdot)$ is a curriculum training loss with ϵ and λ controlling its smoothness level and hard example weights; please refer to [15] for detail definition. In Tab.1 we present the parameter settings for different warping steps. Our progressive reconstruction loss is the sum of all levels of $\mathcal{L}_{rec}^{(s)}$:

$$\mathcal{L}_{rec} = \sum_{s \in \{2,4,6,8\}} \mathcal{L}_{rec}^{(s)} \tag{8}$$

4.2.2 Regularization Loss

Ideally, the spatial warping function is supposed to establish plausible correspondences between the template and the ob-

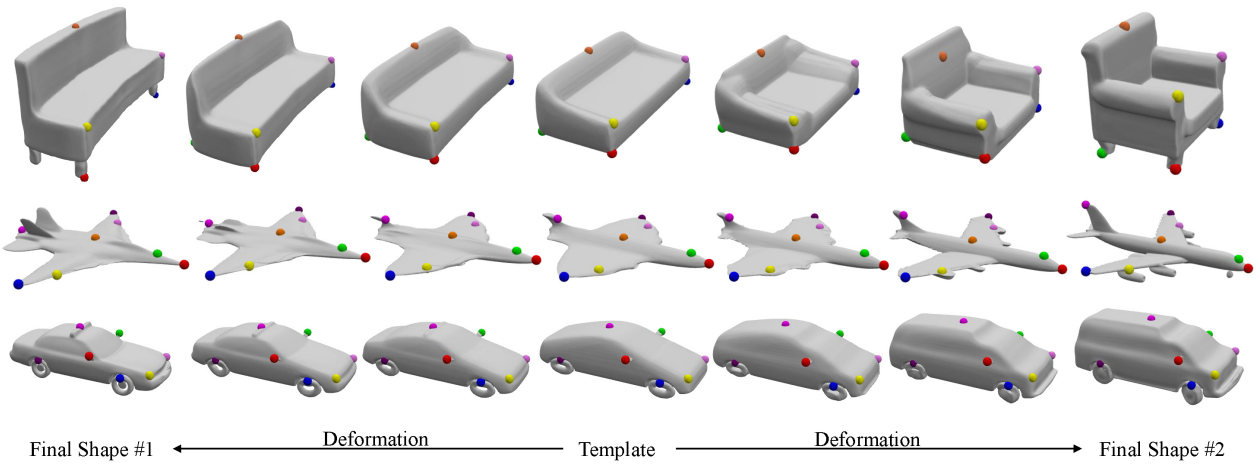


Figure 3: Demonstration of our representation. We manually select several points on the learned templates (middle) and track their traces when the templates deforms to form different shapes (leftmost and rightmost). Note that the deformation is defined and performed in an implicit manner; here we visualize the intermediate results by interpolating the warping fields. The selected points and their movements are rendered as small colored balls.

ject instances, while the implicit field template should capture the common structure for a set of objects. To achieve this goal, we introduce two regularization terms on point-wise and point-pair levels for the warping function.

Point-wise regularization. We assume that all meshes are normalized to a unit sphere and aligned in a canonical pose. Therefore, we introduce a point-wise regularization loss that is used to constrain the position shifting of points after warping. It is defined as:

$$\mathcal{L}_{pw} = \sum_{k=1}^{K}\sum_{i=1}^{N} h\left(\|\mathcal{W}(\boldsymbol{p}_i, \boldsymbol{c}_k) - \boldsymbol{p}_i\|_2\right), \tag{9}$$

where $h(\cdot)$ is the Huber kernel with its hyper-parameter δ_h.

Point pair regularization. Although spatial distortion is inevitable during template deformation, extreme distortions should be avoided. To this end, we introduce a novel regularization loss for the point pairs in each shape:

$$\mathcal{L}_{pp} = \sum_{k=1}^{K}\sum_{i\neq j} \max\left(\frac{\|\Delta\boldsymbol{p}_i - \Delta\boldsymbol{p}_j\|_2}{\|\boldsymbol{p}_i - \boldsymbol{p}_j\|_2} - \epsilon, 0\right), \tag{10}$$

where $\Delta\boldsymbol{p} = \mathcal{W}(\boldsymbol{p}, \boldsymbol{c}) - \boldsymbol{p}$ is the position shift of $\boldsymbol{p}$ and ϵ is a parameter controlling the distortion tolerance. Intuitively, if $\epsilon = 0$, $\mathcal{L}_{pp}$ is reduced to a strict smoothness constraint enforcing translation consistency on neighboring points. We set $\epsilon = 0.5$ in our experiments to construct a relaxed formulation of smoothness loss, which is found important to prevent shape structures from collapsing (Sec.5.4).

Our final regularization loss is defined as:

$$\mathcal{L}_{reg} = \lambda_{pw}\mathcal{L}_{pw} + \lambda_{pp}\mathcal{L}_{pp} + \frac{1}{\sigma^2}\sum_{k=1}^{K}\|\boldsymbol{c}_k\|_2^2, \tag{11}$$

where the last term is the same magnitude constraint on the latent codes as in DeepSDF [40].

5. Experiments

5.1. Experimental Setup

We train Deep Implicit Templates on ShapeNet dataset [10], following [40] for data pre-processing. In Sec.5.2, we first show that our model is capable of representing shapes in high quality with dense correspondences. For comparison in Sec.5.3, we select several strong baselines that use different types of representations: DeepSDF (deep implicit functions) [40], SIF (structured implicit functions) [19] , AtlasNet (mesh parameterization) [22], PointFlow (point clouds) [57] and DualSDF (two-level implicit functions) [24]. We mainly evaluate them and our method in terms of reconstruction, correspondence and interpolation. Finally we evaluate our technical contributions in Sec.5.4. More results, experiments and details are presented in the supplemental materials.

5.2. Results

We demonstrate some results of our approach in Fig.1, Fig.3 and Fig.4. In Fig.1, we present the learned template shapes as well as the dense correspondences between the templates and object instances. The results show that our method can learn to abstract a template that captures the common structure for a collection of shapes, and also establish plausible dense correspondences that indicates the semantic relationship across different shapes. The results also show that our representations can deal with large deformations and describe objects with completely different structures. In Fig.3, we provide a more clear landscape on how the templates deforms to describe different objects. Fig.4

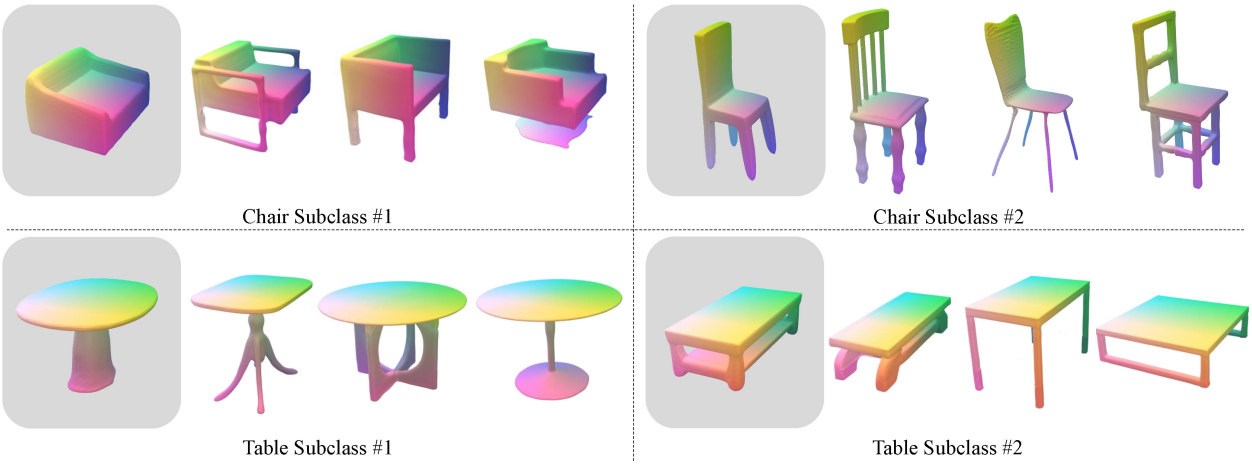


Figure 4: More demonstration of our representation. For objects that show very high structure varieties, like chairs and tables, we can train individual deep implicit templates for different subclass. The learned templates are rendered in gray background and correspondences are shown with the same color.

	CD Mean					CD Median			EMD Mean		
Model \ Shape Class	Airplanes(K)	Airplanes	Sofas	Cars	Chairs	Airplanes	Sofas	Chairs	Airplanes	Sofas	Chairs
AtlasNet-Sph [22]*	-	0.19	0.45	-	0.75	0.079	0.33	0.51	0.038	0.050	0.071
AtlasNet-25 [22]*	-	0.22	0.41	-	0.37	0.065	0.31	0.28	0.041	0.063	0.064
SIF [19]*	-	0.44	0.80	1.08	1.54	-	-	-	-	-	-
DeepSDF [40]†	0.05	0.14	0.12	0.11	0.24	0.061	0.08	0.10	0.035	0.051	0.055
C-DeepSDF [15]†	0.03	0.07	0.11	0.06	**0.16**	0.033	0.07	**0.06**	**0.026**	**0.044**	**0.048**
DualSDF [24]†	0.19	0.22	-	-	0.45	0.14	-	0.21	0.041	-	0.055
Ours (w/o Prog. Loss)†	0.042	0.104	0.117	0.093	0.23	0.040	0.075	0.113	0.031	0.047	0.055
Ours†	**0.025**	**0.053**	**0.093**	**0.052**	0.20	**0.027**	**0.061**	0.071	0.029	0.046	0.049

Table 2: Reconstruction accuracy of different representations on known (K) shapes and unknown shapes for various object categories. Lower is better. (Mean and median Chamfer distance multiplied by 10^3). We highlights methods based on auto-decoders (†) from methods that use an encoder-decoder network architectures (∗).

further presents the representation power of our method as well as the accurate correspondences established by our method.

5.3. Comparison

Reconstruction. We report the reconstruction results for known and unknown shapes (i.e., shapes belonging to the train and test sets) in Tab.2. We use two metrics for accuracy measurement, i.e., Chamfer Distance (CD) and Earth Mover Distance (EMD). As the numeric results show, our method is able to achieve comparable reconstruction performance when compared to state-of-the-art methods. Furthermore, we observe that our method is able to produce slightly better results for object classes that have a strong template prior (e.g., cars, airplanes and sofas), but degenerates for objects that vary enormously in shape structures (e.g., chairs).

Interpolation. Similar to DeepSDF, our learned shape embedding is continuous and supports shape interpolation in the latent space (Fig.5). Note that unlike DeepSDF that directly interpolates signed distance fields, our representation actually *interpolates the spatial warping fields* because the template is independent from the latent code. The results in Fig.5 (bottom row) show that the generative capability of DeepSDF is well preserved in our representation.

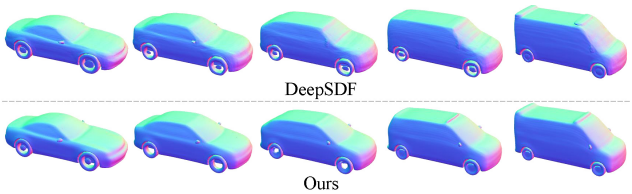


Figure 5: Shape interpolation results of DeepSDF and our method. Given the latent code of different meshes (leftmost and rightmost), we interpolate the latent codes linearly and generate the corresponding meshes (2nd - 4th colume). Models are rendered using their surface normals.

Correspondences. With the dense correspondence provided by our method, one can perform keypoint detection on point clouds by transferring the keypoint labels in training shapes to test ones. Therefore, we use this as a *surrogate* to evaluate the accuracy of correspondences, as there is no large-scale dense correspondence annotation available. We

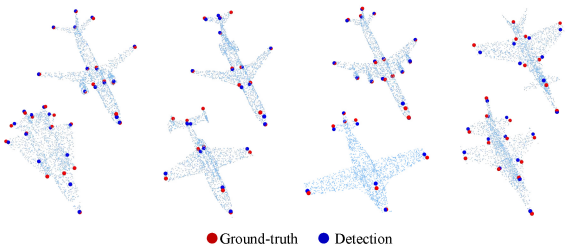


Figure 6: Keypoint detection results by our method. Note that although our network is not trained for keypoint detection, it can still output accurate detection.

Model \ Shape Class	Planes	Cars
SIF [19]	0.267 / 0.446	0.262 / 0.402
PointFlow [57]	0.286 / 0.573	0.083 / 0.326
AltasNet-25 [22]	**0.411** / 0.628	0.206 / 0.361
Ours (w/o point-pair reg)	0.306 / 0.474	0.305 / 0.480
Ours	0.365 / **0.652**	**0.345** / **0.530**

Table 3: Correspondence accuracy comparison. We report the PCK scores with threshold of 0.01 / 0.02 for keypoint detection of different methods. Higher is better.

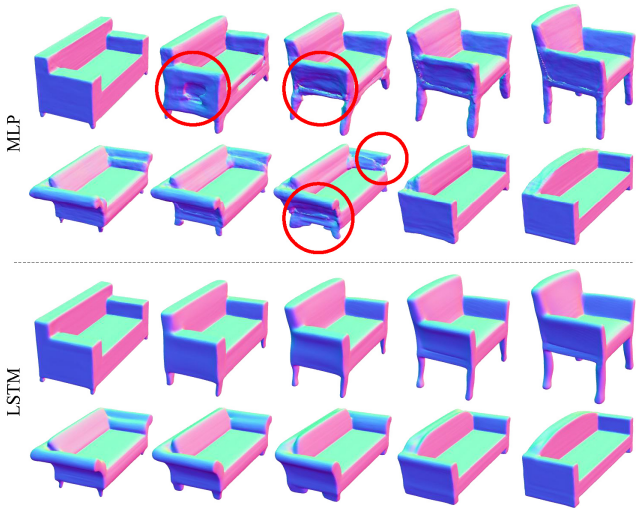


Figure 7: Interpolation capacity comparison of different architectures for the spatial warping function $\mathcal{W}$. Models are rendered using their surface normals.

collect keypoint annotations from KeypointNet [60] and use the percentage of correct keypoints (PCK) [59] as the metric for our experiments. Tab.3 presents the PCK scores under different error distance threshold, showing that our method outperforms other representations. In Fig.6, we can see that, although our network has no access to the keypoint annotations or correspondence annotations during training, it still establishes accurate corresponding relationship across different objects in the same category.

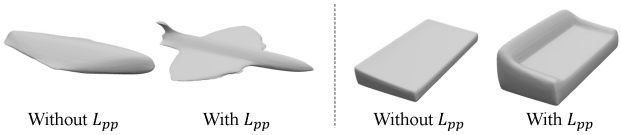


Figure 8: Evaluation of our point-pair regularization. Without the point-pair regularization, the network tends to learn over-simplified templates for planes and sofas.

5.4. Ablation Study

Spatial Warping LSTM. We evaluate our choice of spatial warping LSTM by replacing it with an MLP-based implementation. The numbers of the MLP neurons are $(259, 512, 512, 512, 512, 512, 6)$. The comparison on interpolation capacity of different network architectures is presented in Fig.7. We empirically find that the MLP implementation is prone to over-fitting and cannot generalize well for latent space interpolation. We think that it is because the large non-uniform deformations between the template and object instances are more easily to learn in a gradual manner, but much harder when an MLP try to learn the transformation in a single step.

Progressive Reconstruction Loss. We evaluate the effect of our progressive reconstruction loss in terms of reconstruction accuracy in Tab.2 (last two rows). To construct the evaluate baseline, we replace the progressive reconstruction loss with the original ℓ_1 loss in [40] and only apply supervision on the final output in Eqn.5. The numeric results show that Note that we use the same settings of shape curriculum as [15]; interestingly, we find that our method can even outperform [15] on some object categories although we add additional regularization that may adversely impact the reconstruction. We speculate that this is because we fix the network structure during training and apply multi-level supervision simultaneously, while [15] gradually increases the network depth and changes loss parameters as training proceeds, which may lead to some extent of instability for network training.

Point-pair Regularization. To evaluate our point-pair regularization loss, we train a baseline network without it. As shown in Fig.8, without the point-pair regularization, the networks tends to learn over-simplified templates. Although this phenomenon has no impact on the reconstruct accuracy, it leads to inaccurate correspondences since the detailed structures like plane wings and sofa backrests collapse into tiny regions on the templates, as proven in Tab.3 (last two rows). Therefore, the proposed two-level regularization is essential in our representation of deep implicit templates.

6. Extensions and Applications

6.1. Extensions

To prove the flexibility and the potential of our Deep Implicit Templates, we show how our method can be easily extended when more constraints are available.

Figure 9: Results on clothed humans. With SMPL model as the manually specified template (leftmost), our method can learn to generate human models in various clothes.

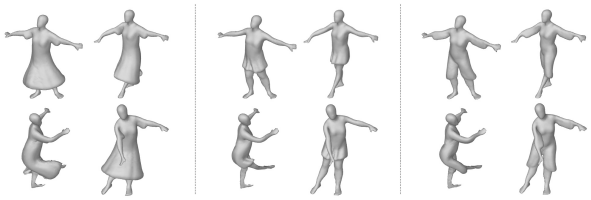


Figure 10: Human model animation. The dense correspondences between the SMPL template and the clothed human models allows automatic calculation of skining weights.

User-defined templates. For example, we can replace the learned template in our method with a manually-specified one. In this case, we just need to construct an additional loss to train the template implicit function $\mathcal{T}$. Specifically, the loss is defined as:

$$\mathcal{L}_{temp} = \sum_{i=1}^{N} |\mathcal{T}(\boldsymbol{p}_i) - s_i|, \tag{12}$$

where $(\boldsymbol{p}_i, s_i)$ are the SDF training pairs extracted from the specified template model. In Fig.9, we demonstrate an example where we specify SMPL model [36] as the template and train our network to model various clothed humans. With the correspondences we can transfer skinning weights from SMPL model to clothed humans, allowing animation for various clothed human models as in Fig.10.

Correspondence annotations. Furthermore, although our method is designed to learn dense correspondence without any supervision, our method can incorporate with sparse/dense correspondence annotations when provided. To this end, we just need to introduce an additional loss to enforce the known correspondences:

$$\mathcal{L}_{corr} = \sum_{\substack{1\le k,l\le K \\ k\neq l}} \sum_{(\boldsymbol{p},\boldsymbol{q})\in \mathrm{C}_{k,l}} \|\mathcal{W}(\boldsymbol{p}, \boldsymbol{c}_k) - \mathcal{W}(\boldsymbol{q}, \boldsymbol{c}_l)\|_2^2, \tag{13}$$

where $\mathrm{C}_{k,l}$ is the correspondence set of the k-th and l-th objects. Fig.11 demonstrate the qualitative and quantitative results of using the correspondence loss when training our network on D-FAUST dataset [6].

6.2. Applications

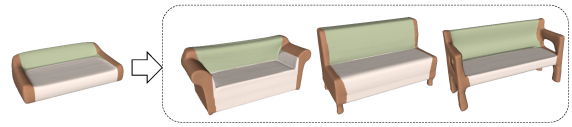
To show the effectiveness and practicability of our method, we investigate how Deep Implicit Templates can be used in applications. With the dense correspondences between the template and the object instances, we can transfer mesh attributes, such as texture (Fig.12(a)), or mesh stretching operation (Fig.12(b)) to multiple objects. Our method can also be used in the applications that have been demonstrated in DeepSDF [40] such as shape completion.

(a) Reconstruction results.

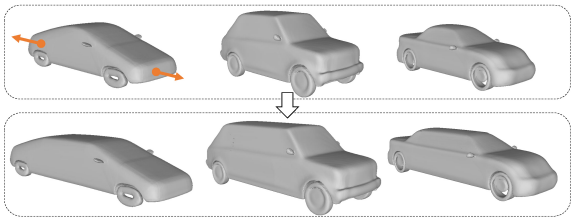
	Without $\mathcal{L}_{corr}$	With $\mathcal{L}_{corr}$
Error	0.337	0.085

(b) Correpsondence error [39]. Low is better.

Figure 11: Results with the correspondence loss. If correspondence annotations are available, our method can learn to reconstruct human models with more accurate correspondences.



(a) Texture transfer.



(b) Shape manipulation.

Figure 12: Our representation can be used in various applications. Please zoom in for better views.

7. Conclusion

We have presented Deep Implicit Templates, a new 3D shape representation that factors out implicit templates from deep implicit functions. To the best of our knowledge, this is the first method that explicitly interprets the latent space for a class of objects as a meaningful pair: an implicit template with its conditional deformations. Several technical contributions on network architectures and training losses are proposed to not only recover accurate dense correspondences without losing the properties of deep implicit functions, but also learn a plausible template capturing common geometry structures. The experiments further demonstrate some promising applications of Deep Implicit Templates. To conclude, we have demonstrated that a semantic interpretation of deep implicit functions leads to a more powerful representation, which we believe is an inspiring direction for future research.

Acknowledgement This paper is supported by the NSFC No.61827805.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J Guibas. Learning representations and generative models for 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017. 1

[2] Thiemo Alldieck, Gerard Pons-Moll, Christian Theobalt, and Marcus Magnor. Tex2shape: Detailed full human body geometry from a single image. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 1, 3

[3] Brett Allen, Brian Curless, and Zoran Popovic. The space of human body shapes: reconstruction and parameterization from range scans. *ACM Trans. Graph.*, 22(3):587–594, 2003. 3

[4] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. SCAPE: shape completion and animation of people. *ACM Trans. Graph.*, 24(3):408–416, 2005. 3

[5] Heli Ben-Hamu, Haggai Maron, Itay Kezurer, Gal Avineri, and Yaron Lipman. Multi-chart generative surface modeling. *ACM Trans. Graph.*, 37(6):215:1–215:15, 2018. 1, 2

[6] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bodies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 8

[7] Jonathan C. Carr, Richard K. Beatson, Jon B. Cherrie, Tim J. Mitchell, W. Richard Fright, Bruce C. McCallum, and Tim R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH*, pages 67–76. ACM, 2001. 2

[8] João Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4733–4742. IEEE Computer Society, 2016. 4

[9] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local SDF priors for detailed 3d reconstruction. *CoRR*, abs/2003.10983, 2020. 1, 2

[10] 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. *CoRR*, abs/1512.03012, 2015. 5

[11] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 3

[12] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey E. Hinton, and Andrea Tagliasacchi. Cvxnets: Learnable convex decomposition. *CoRR*, abs/1909.05736, 2019. 2

[13] Yu Deng, Jiaolong Yang, and Xin Tong. Deformed implicit field: Modeling 3d shapes with learned dense correspondence, 2020. 2

[14] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. In *Advances in Neural Information Processing Systems*, pages 7433–7443, 2019. 1, 2

[15] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevatia, and Leonidas J. Guibas. Curriculum deepsdf, 2020. 1, 2, 4, 6, 7

[16] Noa Fish, Melinos Averkiou, Oliver van Kaick, Olga Sorkine-Hornung, Daniel Cohen-Or, and Niloy J. Mitra. Meta-representation of shape families. *ACM Trans. Graph.*, 33(4):34:1–34:11, 2014. 2

[17] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao(Richard) Zhang. SDM-NET: Deep generative network for structured deformable mesh. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2019)*, 38(6):243:1–243:15, 2019. 2

[18] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas A. Funkhouser. Deep structured implicit functions. *CoRR*, abs/1912.06126, 2019. 1, 2

[19] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T. Freeman, and Thomas A. Funkhouser. Learning shape templates with structured implicit functions. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 7153–7163. IEEE, 2019. 1, 2, 5, 6, 7

[20] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. *CoRR*, abs/2002.10099, 2020. 1, 2

[21] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *15th European Conference on Computer Vision (ECCV)*, volume 11206, pages 235–251, 2018. 1, 3

[22] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation. *CoRR*, abs/1802.05384, 2018. 1, 5, 6, 7

[23] Christian Hane, Shubham Tulsiani, and Jitendra Malik. Hierarchical surface prediction for 3d object reconstruction. In *2017 International Conference on 3D Vision, 3DV*, pages 412–420. IEEE Computer Society, 2017. 1

[24] Zekun Hao, Hadar Averbuch-Elor, Noah Snavely, and Serge Belongie. Dualsdf: Semantic shape manipulation using a two-level representation. *arXiv*, 2020. 1, 2, 5, 6

[25] Qi-Xing Huang, Vladlen Koltun, and Leonidas J. Guibas. Joint shape segmentation with linear programming. *ACM Trans. Graph.*, 30(6):125, 2011. 2

[26] Chiyu Max Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas A. Funkhouser. Local implicit grid representations for 3d scenes. *CoRR*, abs/2003.08981, 2020. 1, 2

[27] Adrien Kaiser, José Alonso Ybáñez Zepeda, and Tamy Boubekeur. A survey of simple geometric primitives detection methods for captured 3d data. *Comput. Graph. Forum*, 38(1):167–196, 2019. 2

[28] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *IEEE CVPR*, 2018. 3

[29] Vladimir G. Kim, Wilmot Li, Niloy J. Mitra, Siddhartha Chaudhuri, Stephen DiVerdi, and Thomas A. Funkhouser. Learning part-based templates from large collections of 3d shapes. *ACM Trans. Graph.*, 32(4):70:1–70:12, 2013. 2

[30] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis. Convolutional mesh regression for single-image human shape reconstruction. In *CVPR*, 2019. 3

[31] Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao (Richard) Zhang, and Leonidas J. Guibas. GRASS: generative recursive autoencoders for shape structures. *ACM Trans. Graph.*, 36(4):52:1–52:14, 2017. 2

[32] Lingxiao Li, Minhyuk Sung, Anastasia Dubrovina, Li Yi, and Leonidas J. Guibas. Supervised fitting of geometric primitives to 3d point clouds. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2652–2660. Computer Vision Foundation / IEEE, 2019. 2

[33] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. PU-GAN: A point cloud upsampling adversarial network. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 7202–7211. IEEE, 2019. 1

[34] Yangyan Li, Xiaokun Wu, Yiorgos Chrysanthou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. Globfit: consistently fitting primitives by discovering global relations. *ACM Trans. Graph.*, 30(4):52, 2011. 2

[35] Feng Liu and Xiaoming Liu. Learning implicit functions for topology-varying dense 3d shape correspondence. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4823–4834. Curran Associates, Inc., 2020. 2

[36] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: a skinned multi-person linear model. *ACM Trans. Graph.*, 34(6):248:1–248:16, 2015. 3, 8

[37] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, 1987. 3

[38] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 3

[39] Michael Niemeyer, Lars M. Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 5378–5388. IEEE, 2019. 2, 8

[40] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 3, 4, 5, 6, 7, 8

[41] Despoina Paschalidou, Luc Gool, and Andreas Geiger. Learning unsupervised hierarchical part decomposition of 3d objects from a single rgb image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[42] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 10344–10353. Computer Vision Foundation / IEEE, 2019. 2

[43] Shunsuke Saito, , Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 1, 2, 3

[44] Ruwen Schnabel, Patrick Degener, and Reinhard Klein. Completion and reconstruction with primitive shapes. *Comput. Graph. Forum*, 28(2):503–512, 2009. 2

[45] Chen Shen, James F. O'Brien, and Jonathan Richard Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. *ACM Trans. Graph.*, 23(3):896–904, 2004. 2

[46] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *IEEE International Conference on Computer Vision, ICCV*, pages 2107–2115. IEEE Computer Society, 2017. 1

[47] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Carsten Stoll, and Christian Theobalt. PatchNets: Patch-Based Generalizable Deep Implicit 3D Shape Representations. *European Conference on Computer Vision (ECCV)*, 2020. 1, 2

[48] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1466–1474. IEEE Computer Society, 2017. 2

[49] Greg Turk and James F. O'Brien. Modelling with implicit surfaces that interpolate. *ACM Trans. Graph.*, 21(4):855–873, 2002. 2

[50] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single RGB images. In *European Conference on Computer Vision (ECCV)*, volume 11215 of *Lecture Notes in Computer Science*, pages 55–71. Springer, 2018. 1, 3

[51] Chao Wen, Yinda Zhang, Zhuwen Li, and Yanwei Fu. Pixel2mesh++: Multi-view 3d mesh generation via deformation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV*, pages 1042–1051. IEEE, 2019. 1, 3

[52] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3d shape reconstruction via 2.5d sketches. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 540–550, 2017. 1

[53] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 82–90, 2016. 1

[54] Jiajun Wu, Chengkai Zhang, Xiuming Zhang, Zhoutong Zhang, William T. Freeman, and Joshua B. Tenenbaum. Learning shape priors for single-view 3d completion and reconstruction. In *European Conference on Computer Vision (ECCV)*, volume 11215 of *Lecture Notes in Computer Science*, pages 673–691. Springer, 2018. 1

[55] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In

IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015, pages 1912–1920. IEEE Computer Society, 2015. 1

[56] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomír Mech, and Ulrich Neumann. DISN: deep implicit surface network for high-quality single-view 3d reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 490–500, 2019. 1, 2

[57] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. *arXiv*, 2019. 5, 7

[58] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 206–215. IEEE Computer Society, 2018. 1

[59] Li Yi, Hao Su, Xingwen Guo, and Leonidas J. Guibas. Syncspeccnn: Synchronized spectral CNN for 3d shape segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6584–6592. IEEE Computer Society, 2017. 7

[60] Yang You, Yujing Lou, Chengkun Li, Zhoujun Cheng, Liangwei Li, Lizhuang Ma, Cewu Lu, and Weiming Wang. Keypointnet: A large-scale 3d keypoint dataset aggregated from numerous human annotations. *CoRR*, abs/2002.12687, 2020. 7

[61] Mehmet Ersin Yümer and Levent Burak Kara. Co-abstraction of shape collections. *ACM Trans. Graph.*, 31(6):166:1–166:11, 2012. 2

[62] Maciej Zamorski, Maciej Zięba, Piotr Klukowski, Rafał Nowak, Karol Kurach, Wojciech Stokowiec, and Tomasz Trzciński. Adversarial autoencoders for compact representations of 3d point clouds. *arXiv preprint arXiv:1811.07605*, 2018. 1

[63] Hao Zhu, Xinxin Zuo, Sen Wang, Xun Cao, and Ruigang Yang. Detailed human shape estimation from a single image by hierarchical mesh deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 3

[64] Chuhang Zou, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem. 3d-prnn: Generating shape primitives with recurrent neural networks. In *IEEE International Conference on Computer Vision, ICCV*, pages 900–909. IEEE Computer Society, 2017. 2

[65] Silvia Zuffi and Michael J. Black. The stitched puppet: A graphical model of 3d human shape and pose. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3537–3546. IEEE Computer Society, 2015. 3