Pacific Graphics 2018
H. Fu, A. Ghosh, and J. Kopf
(Guest Editors)

Volume 37 (2018), Number 7

ParamNet: Towards A Network of *Parameterization Prediction* for Shape Generation from a Single Image

ID: paper1147

Abstract

We propose an end-to-end deep learning framework that maps from a unit sphere surface to the target surface, given a single color image. While mesh or surface models are widely used in many applications, it is challenging to apply the successful deep neural networks to directly produce surface models. Unlike the methods that use volumetric or point representations, our method represents a 3D shape by a mapping from a fixed parameter domain. In our framework, the mapping is progressively carried out by the parameterization network*, given parameters that are predicted by the* semantic network *from a single color image. By integrating mesh-based operation (e.g. Laplacian smooth) and mesh related losses into the network, our framework directly outputs high-quality meshes. Experiments show that our method not only produces mesh models that are more visually appealing, but also achieves comparable 3D shape estimation errors compared to the state of the art.*

CCS Concepts

•***Computing methodologies*** → ***Shape representations; Neural networks; Mesh models;***

1. Introduction

Inferring 3D shape from a single view image is a traditional problem for computer vision. In computer graphics, 3D modeling with a given image has also been extensively studied. Though it have been studied for decades, the problem remains challenging, due to the fact that 3D-to-2D projection is not invertible, and large portions of the 3D shape features are excluded in the 2D image.

Recently, great success has been achieved for 3D shape generation from a single color image using deep learning techniques [CXG*16, FSG16]. By using convolutional layers on regular grids or multi-layer perception on unordered 3D coordinates, the estimated 3D shape is represented as either a volume occupancy [CXG*16] or point cloud [FSG16] in neural networks. However, both representations lose important surface details, and they do not recover continuous surface forms.

While meshes are more suitable for modeling shape details, deformation and rendering with various materials, efforts have been made to produce mesh models using deep neural networks. Integrating mesh morphing techniques, an end-to-end trainable network is proposed to generate 3D meshes of a specific class of objects, such as faces [DSK17]. However, the use of morphing technique limits the network's generalization ability on general objects.

In order to develop an end-to-end trainable network that is capable of producing meshes for multiple classes, we propose a brand new framework. Our framework is composed of two neural networks, the parameterization network and the semantic network. The semantic network predicts network parameters for the parameterization network, which maps the unit sphere surface to the target surface.

We propose such framework under two major motivations:

a) We want to separate the 3D shape space and semantic feature space. Such separation can make the learned shape operation more interpretable, while in traditional networks like MLP and CNN, the features from middle layers are usually not directly interpretable. In other words, the effect of learned shape operation can be easily visualized by intermediate output as shown in parameterization network in Figure 1. Such separation also makes it more intuitive to integrate traditional shape operations (e.g. Laplacian smooth) into the network.

b) On one hand, we prefer to represent target surface by mapping from a fixed parameter domain. In this way, we can do triangulation only on the point samples from a fixed domain of our chosen (In implementation presented in this paper, we choose sphere surface as the fixed parameter domain) and transfer the triangulation to target surface to generate mesh. On the other hand, given the definition of our problem, we need to depend the mapping on input image. Using one network to carry out the mapping and using another network to predict its network parameters from input image is a viable and novel solution that meets both requirements.

In summary, our contributions are

- Introduction of parameterization networks that enable mesh generation by representing the 3D shape as a spherical uniform distribution plus a learned/predicted mapping.
- Exploring the idea that uses semantic network to predict param-

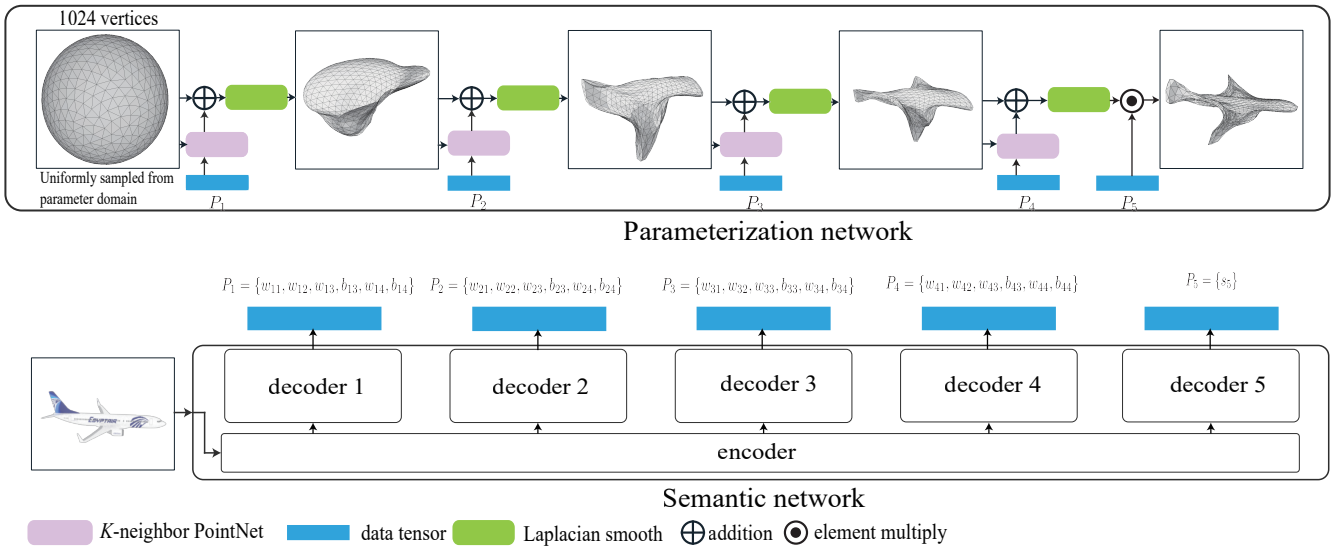

Figure 1: *The overview of the network: The proposed framework consists of two networks. One is the parameterization network that maps points sampled from parameter domain to target shape. The other is the semantic network that takes image as input and predict parameters for the parameterization network.*

eters for parameterization network. Such structure relate input image to parameterization network.

- Though bijective is not ensured for the predicted mapping now, with the ability to integrate mesh based operation and losses, our framework push the neural network towards an actual *parameterization prediction* network that would allow more mesh operation former studied in computer graphics to be integrated into neural networks.

2. Related Work

3D reconstruction and modeling from a single image has been extensively studied as the problem of *shape from monocular cues*, including shadings [ZTCS99], focuses [FSBO08, FS05], and textures [Alo88]. These methods usually recover 2.5D surfaces from 2D images. Learning-based approaches, especially deep learning methods, can acquire more complicate priors by learning from datasets and recover much more complete 3D shape from a single image.

2.1. General Learning Approaches

As far as we known, early work of learning approaches can be traced back to [HEH07] and [SSN07]. These methods learn to segment and classify regions in image and finally produce 3D scene by folding the 2D image. More recent techniques break down the problem to two stages [SHM*14,HWK15]. One is to retrieve shape components from a large dataset, and the other is to assemble the components and deform the assembled shape to fit the observed image. These methods need to segment the shape into components for the database. However, shape retrieval from images itself is an challenging problem due to the loss of information during 3D-to-2D projection. [KTCM15] avoid the retrieval step by learning a deformable 3D shape for each category and learn to predict deformation from input image for these specific categories.

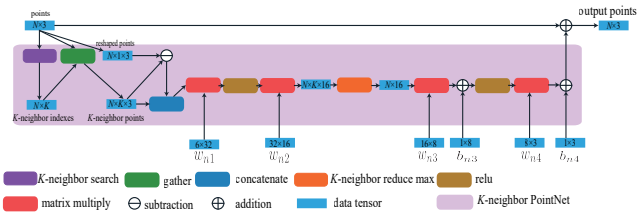

Figure 2: *The internal structure of K-neighbor PointNet.*

2.2. 3D Neural Networks

Most recently, researchers have developed techniques to represent 3D shapes inside deep learning frameworks.Unlike images, 3D shapes are not canonical functions on well-organized grids. This leads to exploration on various representations of 3D shapes.

Volume Occupancy An intuitive way to apply convolutional network in 3D is to use volume occupancy of regular 3D grids to represent 3D shapes [WSK*15], and it is subsequently use for 3D shape generation [CXG*16, GFRG16]. The main disadvantage of volumetric representation was the large memory consumption due to the raising of dimension when extending 2D grids to 3D. Octree representation is proposed to support higher resolution outputs with limited memory, and used for shape generation [TDB18] and shape analysis [WLG*17].

Point Clouds Compared to regular 3D grids, point clouds is not

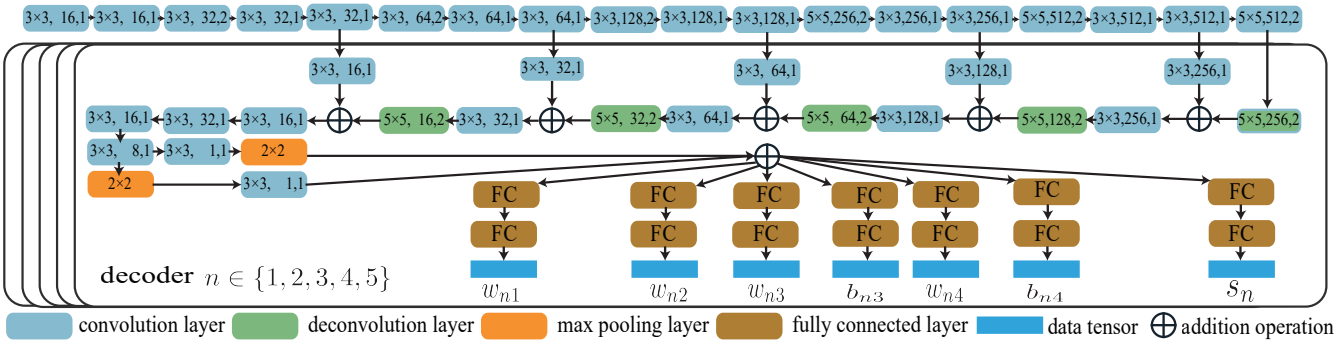

Figure 3: *Semantic network: The semantic network takes image as input and predict parameters for the K-neighbor PointNet and the global scale operation in the parameterization network. The semantic network have separate decoders for each K-neighbor PointNet and and the global scale operation.*

limited by fixed local connections. Many networks have been proposed to take unordered 3D point sets as input and extract geometric features from 3D point set for classification or segmentation [CSKG17, QYSG17, LBSC18]. The first attempt to generate a set of discrete points from a single image using neural networks was made by [FSG16], however, it is non-trivial to construct continuous surface models from the predicted point sets. Since the local variation of point positions are not continues in the predicted point sets.

Meshes Meshes are widely used in game and movie industries. Spherical parameterization techniques like [HFL18] have been well studied and In addition to vertex positions, the mesh representation contains local structure of vertices. However, the mesh representation is not well supported in current neural networks. To generate mesh models by using neural networks, composition weights of a series of base meshes are predicted by networks [PKS*17] and [DSK17]. Since it is only possible to choose/learn base meshes for specific class of object, these two networks only generates meshes for a specific class of object, such as face. [KUH17] aims to make a differentiable approximation for 3D mesh rendering process, which we believe is a significant step for both graphics and vision community. However, in its application for mesh generation, it only take silhouette as supervision, hence does not perform well for complicated objects, such as cars, airplanes, etc. In comparison, our network draws a lot of experiences from techniques designed for point cloud representation, and it directly produces meshes as output.

2.3. Parameterization in Learning

The idea of utilizing surface parameterization in neural networks has been explored [SUHR17, SBR16]. Typically, a non-trainable procedure is involved for the creation of geometry image. Manifold surfaces are required as training data so that the shapes can be parameterized using spherical parameterization and turned into geometry image. However, the public datasets like ShapeNet [CFG*15] contain meshes that are not manifold surfaces. In comparison, we represent 3D surfaces by surface parameterization. Our proposed network predicts a mapping from the parameter domain to the target surface. Manifold surfaces are not necessary as supervision. Out network can be trained with point clouds as supervision, but it directly produces mesh models.

2.4. Learning to Learn and Filter Prediction

It is not a new idea to learn neural network that predict the network parameters of another. It is one of the approaches found for *learning to learn*. A few methods [Sch92, BHV*16, JDBTG16] , in particular, have been proposed to apply such idea in their work. Our framework adopt this idea to separate the semantic feature space and the 3D shape space. In this way, the learned shape operation becomes more interpretable. In other words, the effect of learned shape operation can be directly visualized by the intermediate result of the parameterization network as shown in Figure 1. Such separation also makes it more intuitive to add traditional shape operations (e.g. Laplacian smooth) to the network.

3. Our Method

In this section, we first explain the general framework of our proposed network, and then we elaborate the structure details part by part.

3.1. Network overview

Our original idea about the *parameterization prediction* network was to use a semantic network that takes a single image as input to predict a mapping from the parameter domain to the target surface.

In the proposed framework shown in Figure 1, the mapping is actually expressed by the parameterization network. Instead of directly predicting the mapping, the semantic network predicts parameters for the parameterization network. The entire framework is end-to-end trainable.

The parameterization network is built by stacking several *K*-neighbor PointNet (explained in Sec 3.2). Each *K*-neighbor PointNet takes 3D point set as input and predicts a offset for each point. In this way, the parameterization network actually maps a randomly sampled point set to the target shape.

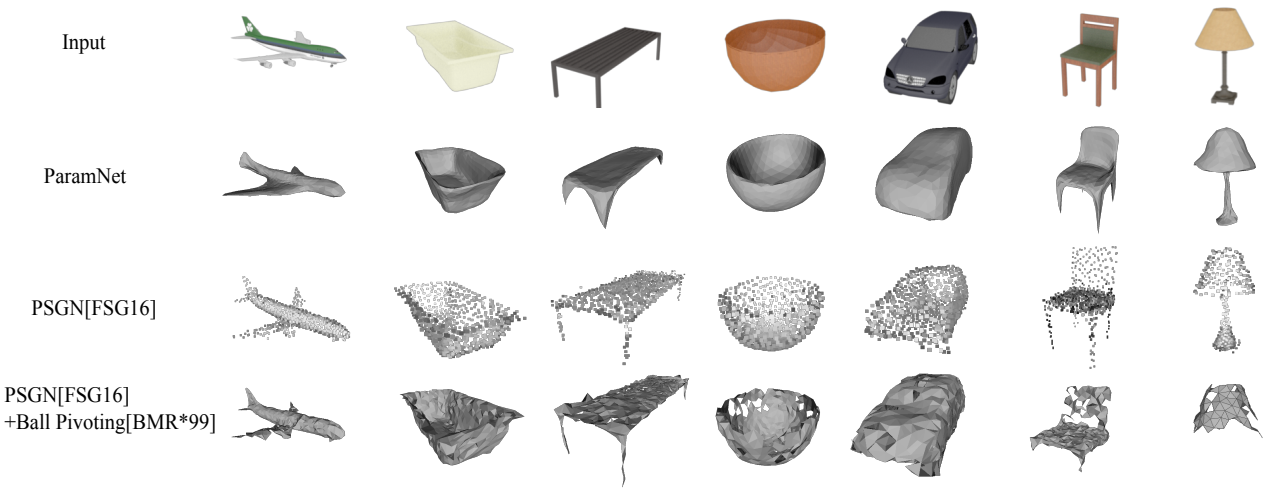

Figure 4: *The comparison of visual results*

The semantic network is built on convolution, deconvolution and fully connected layers. It takes a single image as input to predict the parameters (i.e.$\{P_l\}$ for each layer in Figure 1) for the parameterization network. In this way, the semantic network links the input image and the surface parameterization.

3.2. Parameterization Network

In the proposed framework, we use unit sphere surface as parameter domain. As shown in Figure 1, at the beginning of the parameterization network $N = 1024$ points are uniformly sampled from the parameter domain. (Here we choose $N = 1024$ to be same as in [FSG16]. More points can be used to express more fine details and for our framework increasing N do not require the increasing of network parameter.) Triangulation is applied on these sampled points. The edge connections built by triangulation are later used in the Laplacian smooth layer and the edge length regularization term. These connections are also used to connect output points to mesh.

***K*-neighbor PointNet for parameterization network** An essential block in parameterization network is the K-neighbor PointNet. Figure 2 shows its structure. K-neighbor PointNet is inspired by and named after PointNet [CSKG17] and its follow-up PointNet++ [QYSG17]. In order to take unordered point set as input, K-neighbor PointNet uses either point-wise operations on each point or symmetric functions on multiple points. As shown in Figure 2, the K-neighbor search operation finds the K nearest neighboring points for each point inside the N input points, and outputs the indexes tensor. We employed the Bitonic Sort [Bat68] for its GPU implementation. Then the *gather* operation re-organize the input points based on these indexes and forge the tensor of "K-neighbor points" as shown in Figure 2. In the tensor of "K-neighbor points", the coordinates of K-neighbor points for the N input points are placed togather and results in a tensor with shape of $N \times K \times 3$ The implementation for *gather* operation is available in tensorflow. Subtracting the original input points from K-neighbor points we can get the local relative coordinates of K-neighbor points. By concatenating K-neighbor point coordinates and their local relative coordinates, we get the $N \times K \times 6$ tensor as geometric features. These features go through a series of operations as shown in Figure 2 to predict point-wise 3D offsets for each input point. Unlike the original PointNet [CSKG17], who extracts maximum value among all point as feature vector for input point set, our K-neighbor PointNet extract maximum value among every K-neighborhood as feature vector for each point. This is shown by the K-neighbor reduce max operation in Figure 2.

Laplacian Smoothing Another essential building block for our parameterization network is the Laplacian smoothing layer. Laplacian smoothing is a traditional mesh based operation that is is commonly used in mesh processing. We integrate it into our network because most objects have locally smooth surfaces. By applying a Laplacian smoothing to the mapped/deformed surface after each K-neighbor PointNet, we improve the visual quality of the produced mesh.

The new position of each vertex $\mathbf{x}^*$ in the smoothed mesh is computed as Eq. (1), where $\mathcal{N}(\mathbf{x})$ represents the one-ring neighborhood of vertex $\mathbf{x}$. This Laplacian smoothing operation is local linear and therefore differentiable.

$$\mathbf{x}^* = \frac{1}{|\mathcal{N}(\mathbf{x})|} \sum_{\mathbf{y} \in \mathcal{N}(\mathbf{x})} \mathbf{y} \tag{1}$$

Our ablation study shows that the Laplacian smooth significantly improves the regularity of the generated surface and makes the output mesh visually more appealing.

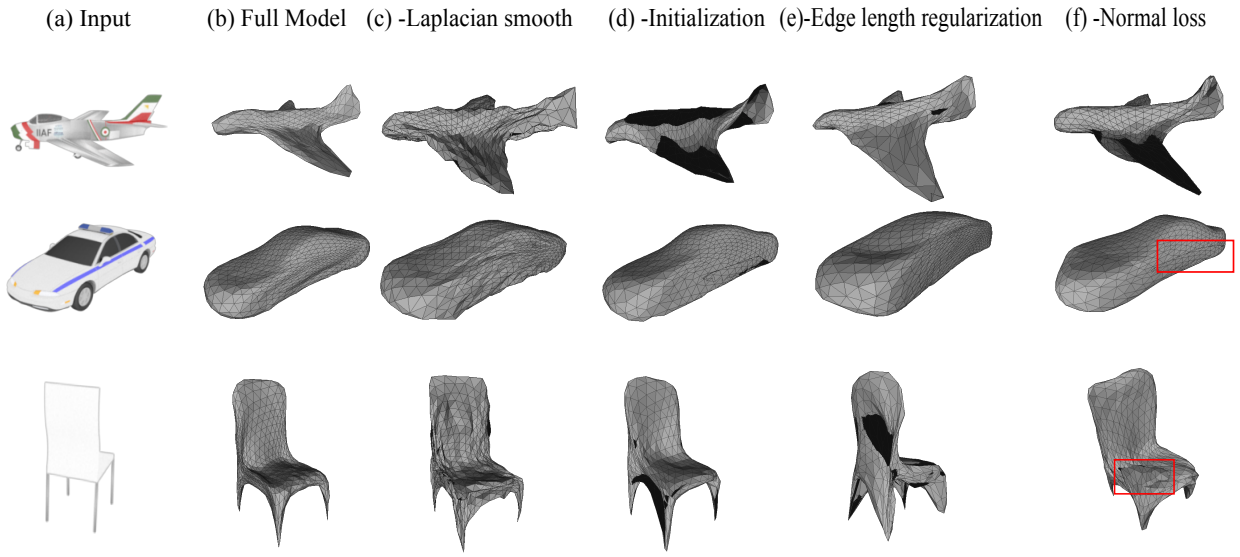

Figure 5: *Visual result for ablation study: In some cases, the view angle is adjusted for better exposure of imperfection. The black regions in the meshes are regions with flipped triangles*

Table 1: *Ablation study with respect to different components*

Models	Full Model	-Laplation smooth	-Initialization	-Edge length regularization	-Norm loss
Chamfer	0.297	0.394	0.424	0.405	0.390
EMD	0.834	1.369	1.524	4.195	1.415

3.3. Semantic Network

In our semantic network, we use convolution layers to extract semantic features from the input image, as shown in Figure 3. Our semantic network adopts the U-shape structure that is similar to UNet [RFB15]. To fit with our parameterization framework, we use five separate decoder branches to predict different parameters for each block in our parameterization network. These predicted parameters (i.e. $P_n = \{\mathbf{W}_{n1}, \mathbf{W}_{n2}, \mathbf{W}_{n3}, \mathbf{b}_{n3}, \mathbf{W}_{n4}, \mathbf{b}_{n4}, s_n\}$) are plugged into the parameterization network in the way shown in Figure 1.

3.4. Loss Function

The whole loss we use is defined in Eq. (2). It is composed of four terms: Chamfer loss L_{chmf}, L2 regularization L_{reg}, Edge length regularization L_{edge}, and a normal loss L_{norm}.

$$L(M_p, M_g) = L_{chmf} + L_{norm} + L_{edge} + \alpha L_{reg}. \tag{2}$$

We directly borrow the Chamfer distance from [FSG16] as the Chamfer loss to measure the shape differences between the predicted mesh M_p and the ground truth points M_g, as defined in Eq.(3). In Eq.(3), $\mathbf{x}$ are vertices on output mesh and $\mathbf{y}$ are the points from the ground truth.

$$L_{chmf} = \sum_{\mathbf{x}\in M_p} \min_{\mathbf{y}\in M_g} ||\mathbf{x}-\mathbf{y}||_2^2 + \sum_{\mathbf{y}\in M_g} \min_{\mathbf{x}\in M_p} ||\mathbf{x}-\mathbf{y}||_2^2, \tag{3}$$

where $\mathbf{x}$ are vertices of output mesh and $\mathbf{y}$ are the points from the groundtruth point clouds.

Besides of the distance of vertices, we use the **Normal loss** L_{norm} to better guide the predicted surface to approach the groundtruth. As defined in Eq. (4), the $\mathbf{n}_i$ and $\mathbf{n}_j$ are the normals at vertices $\mathbf{y}_i$ and $\mathbf{y}_j$. $\mathbf{y}_i$ and $\mathbf{y}_j$ are respectively closest vertices to $\mathbf{x}_i$ and $\mathbf{x}_j$ in the groundtruth. They are found when calculating the Chamfer loss (3). This normal loss encourages the edges on the predicted meshes to remain perpendicular to the corresponding normals of its two end points.

$$L_{norm} = \sum_{(i,j)\in\mathcal{E}} \Big((\mathbf{x}_i - \mathbf{x}_j)\cdot\mathbf{n}_i\Big)^2 + \Big((\mathbf{x}_i - \mathbf{x}_j)\cdot\mathbf{n}_j\Big)^2. \tag{4}$$

To discourage over stretched triangles in the output mesh, we add an **Edge Length Regularization** term to minimize the variance of the edge lengths for the output mesh. As defined in (5), $\mathcal{E}$ is the set of edges, recorded as a set of paired vertex indexes (i.e. (i,j)).

The **L2 Regularization** loss L_{reg} is applied for the parameters in our semantic network.

$$L_{edge} = \sum_{(i,j)\in\mathcal{E}} (\mathbf{x}_i - \mathbf{x}_j||_2 - \frac{1}{|\mathcal{E}|} \sum_{(i,j)\in\mathcal{E}} ||\mathbf{x}_i - \mathbf{x}_j||_2)^2 \tag{5}$$

Table 2: *Ablation study with respect to number of K-neighbor PointNet*

Number of K-n PointNet	2	3	4	5
Chamfer	0.425	0.411	0.297	0.313
EMD	1.304	1.360	0.834	1.067
Training Time	3d20h	4d2h	4d10h	4d23h

3.5. Training

Dataset. We train and evaluate our models on the ShapeNet [CFG*15]. Specifically, we use the ShapeNetCore55.v2 which contains over 50k manually created and cleaned 3D CAD models in 55 category. The images for training and testing are rendered in 16 random angles to provide synthetic training data for the model. In total, 51,856 shape models are covered. For the training/validation/testing split, we follow the CSV file provided by the ShapeNet website and resulted in 36,622/5,110/10,124 shapes. The 3D CAD objects are stored as meshes, and we re-sample the meshes into point sets. In order to capture only the shape surface, we uses the code from [WLG*17] and execute "virtual scan" for the re-sampling. For each view angles, we sample 2 different point samples as ground truth for training and testing. In other words, for each CAD model we generate 32 pairs of image and point set as our data in training/validation/testing.

Network Super Parameters Unless otherwise stated, the network super parameters are chosen as follows for implementation of our framework presented in this paper. We use four K-neighbor PointNet, whose $K = 16$.

Network Initialization Even with the well-designed loss functions, the proposed network might output surfaces with severe global self-intersection, when we train it from scratch. To alleviate this problem, we adopt a *network initialization* step to reset the parameterization network to start from a state that produce surface without self-intersection. In other words, for any input image, we first expect the network to map the parameter domain to a smaller sphere surface (radious=0.5), instead of predicting a shape to approach the ground truth. Thus, the loss function for this initialization step is defined as:

$$L_{init} = \sum_{\mathbf{x} \in M_p} \min_{\mathbf{y} \in S_{0.5}} ||\mathbf{x} - \mathbf{y}||_2^2. \quad (6)$$

We do this training of initialization by one epoch on the training data.

Training configuration Unless otherwise stated, our networks are trained with following configurations. We use Adam optimizer [KB14] with learning rate of $3e-5$. We use $\alpha = 1e-5$ for training loss. We use 32 as size for mini-batch. For each training iteration, we re-sample $N = 1024$ points from the parameter domain and do triangulation with them. These sampled points are shared inside a mini-batch and therefore triangulation is also shared across the mini-batch. For comparison, our network and [FSG16] are both trained 8 epochs on the training data. Typically, it takes four and a half day to train our network on 4 Tesla K80.

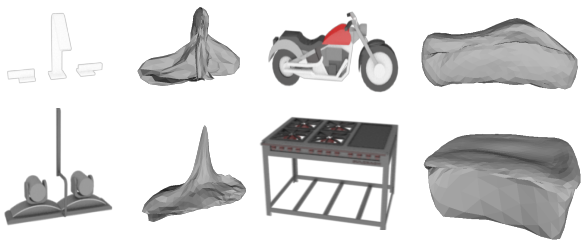

Figure 6: *Failure cases: In some cases, the view angle is adjusted for better exposure of imperfection*

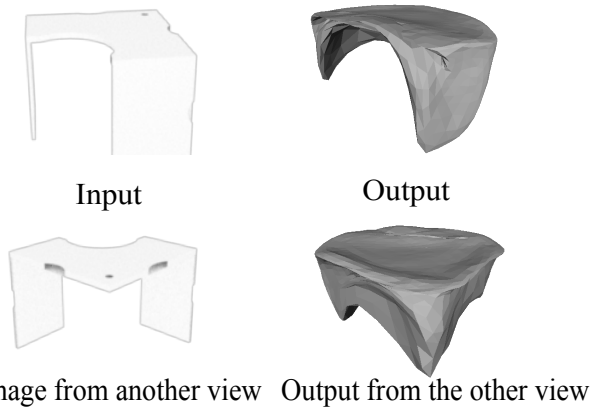

Figure 7: *Failure case: In this case, the network have output an extra leg that doesn't exist for the dresser*

4. Results and Discussions

In this section, we show both qualitative and quantitative results to evaluate the effectiveness of our network, and compare it with state-of-the-art methods.

We first compare our approach with the state-of-the-art point set generation network (PSGN) [FSG16]. In Table 3, we compare the our approach with PSGN [FSG16] in quantitative evaluation over all category of objects in testing data. The results shows that our approach achieves over all comparable performance with PSGN [FSG16] under Chamfer distance and present evidently better performance under Earth mover distance. For categories like bus, bottle, dishwasher etc, who has a simple that is homotopy equivalent to sphere surface. Our network outperform PSGN [FSG16] under both criteria. For the categories like headphone mike etc, who has more various shapes and a lot of which are not homotopy equivalent to sphere. Our network performs worse than PSGN [FSG16] under Chamfer distance.

In Figure 4, we compare the visual results of our framework (ParamNet) and PSGN [FSG16]. In Figure 4, we also show mesh

Figure 8: *More visual results randomly picked from testing data*

Table 3: *Comparison with point set generation network [FSG16]: The Category id from the dataset [CFG*15] is only listed for look up*

Category	Category id	Chamfer		EMD	
		ParamNet	PSGN [FSG16]	ParamNet	PSGN [FSG16]
aircraft	02691156	0.068	0.058	0.248	0.502
dustbin	02747177	0.162	0.166	0.425	1.947
bag	02773838	0.398	0.453	0.921	3.258
basket	02801938	0.861	0.708	1.545	2.186
bathtub	02808440	0.072	0.070	0.175	0.472
bench	02828884	0.063	0.063	0.239	0.641
bed	02818832	0.362	0.291	0.865	1.523
birdhouse	02843684	0.855	0.624	1.972	4.332
shelf	02871439	0.100	0.081	0.407	1.475
bottle	02876657	0.075	0.084	0.289	1.340
bowl	02880940	0.726	0.423	0.954	1.646
bus	02924116	0.035	0.036	0.306	0.602
dresser	02933112	0.082	0.078	0.168	0.799
camera	02942699	0.818	0.752	1.889	3.124
can	02946921	0.165	0.180	0.778	4.247
cap	02954340	1.466	2.738	2.795	3.559
car	02958343	0.051	0.047	0.176	0.399
cellphone	02992529	0.143	0.128	0.965	7.583
chair	03001627	0.042	0.038	0.117	0.605
clock	03046257	0.152	0.119	0.387	1.194
keyboard	03085013	0.450	0.456	1.470	2.626
dishwasher	03207941	0.204	0.236	0.501	2.975
monitor	03211117	0.079	0.067	0.224	0.811
headphone	03261776	0.738	0.590	3.906	4.499
hydrant	03325088	0.177	0.151	0.748	1.123
file cabinet	03337140	0.121	0.113	0.325	1.949
guitar	03467517	0.014	0.015	0.278	1.234
helmet	03513137	0.789	1.199	1.693	2.387
vase	03593526	0.088	0.082	0.305	1.244
knife	03624134	0.034	0.029	0.280	2.089
lamp	03636649	0.089	0.084	0.503	0.869
laptop	03642806	0.174	0.154	0.537	1.262
speaker	03691459	0.125	0.117	0.315	0.806
mailbox	03710193	0.258	0.252	1.245	4.337
mike	03759954	1.599	1.301	3.149	6.842
microwave	03761084	0.305	0.302	0.675	2.647
motorcycle	03790512	0.153	0.139	0.522	1.495
mug	03797390	0.269	0.188	0.481	1.568
piano	03928116	0.234	0.227	0.693	1.587
pillow	03938244	0.614	0.526	0.996	2.101
handgun	04090263	0.049	0.049	0.304	0.790
planter	03991062	0.124	0.139	0.301	0.839
printer	04004475	0.500	0.413	1.064	1.716
remote	04074963	0.106	0.105	0.846	5.940
missile	04099429	0.220	0.187	1.874	4.113
skateboard	04225987	0.369	0.295	1.696	2.232
sofa	04256520	0.051	0.052	0.172	0.321
stove	04330267	0.184	0.214	0.488	1.528
table	04379243	0.077	0.075	0.271	0.537
tower	04460130	0.620	0.735	1.714	3.812
train	04468005	0.129	0.122	0.638	1.140
ship	04530566	0.067	0.057	0.273	0.620
washer	04554684	0.205	0.203	0.482	1.957
average	-	**0.297**	0.298	**0.834**	2.086

generated by ball-pivoting [BMR*99] from the output point set of PSGN [FSG16]. These visual results show that it is difficult to recover continuous surface from the point set generated by PSGN [FSG16], since it does not explicitly consider local structure of the generated surface, while our method can generate continuous surfaces for various objects.

4.1. Ablation study

In this subsection, we do ablation study to investigate the functionality of each component in our framework. The evaluation results are shown in Table 1 and Table 2. Some visual result for the ablation study are shown in Figure 5. Now we discuss these results as follows:

Laplacian smoothing. The Laplacian smoothing operation forces the output surface to have more regular local structure. Without Laplacian smoothing, the output surfaces become much more rough, as shown in Figure 5(c).

Network initialization. The initialization step resets the network to a state that outputs surface without self-intersection. As shown in Figure 5(d), severe triangle flipping and self-intersection happens when the initialization training is skipped.

Edge length regularization The edge length regularization term in the loss is designed to discourage over stretched triangles by minimizing the variance of edge length. According to our observation, when edge length regularization term is removed from the loss, the network tend to produce mesh that has more large triangles as shown by the airplane example in Figure 5(e). Without such suppression for triangle over-stretching, the output becomes more easily to have self-intersection and triangle flipping, such as the chair example in Figure 5(e).

Normal loss The normal loss is designed to guide the learning by using the normal from groundtruth. According to our observation, the normal information is evidently useful to prevent some local self-intersection as highlighted by red rectangles in Figure 5(f).

Number of *K*-neighbor PointNet The number of K-neighbor PointNet is an important super parameter for our framework of networks. Based on the experiment results shown in Table 2, we choose to use four K-neighbor PointNet to construct the parameterization network, since the test error no longer diminishes but the training time increases when we use more (i.e. five).

4.2. Visual Result on Real Images

We also visually test our model on Online Products dataset [SXJS16]. As the examples shown in Figure 9, our network trained on synthetic data can be generalized to produce shape from real image.

4.3. Limitations and future work

A major limitation for our framework is that it cannot handle the surface generation for objects that are not homotopy equivalent to sphere surface. This limitation arises from the design of our framework and is clearly shown in the cases in Figure 6. For this issue, we are planning to combine our framework with the methods like [TSG*17, NLX18]. In this way, we can generate objects with several separate parts and therefore represent objects with more complicate topology.

Figure 9: *Results on real image: the background of the real images are manually masked when input into our network*

Another limitation is that our semantic network failed to correctly infer the parameter for shape from the input image. Figure 7 shows an example of such failure case. This limitation arises from the ambiguity in the 2D to 3D inference. For this issue, we are planning to try VAE method [KW13] or min-of-n loss proposed by [FSG16] to encode the ambiguity.

5. Conclusions

We propose an end-to-end trainable framework of networks that can learn to generate shape from single image without knowing the type of the shape in image. Our approach can generate better continuous surface than the state-of-art PSGN [FSG16].

References

[Alo88] ALOIMONOS J.: Shape from texture. *Biological Cybernetics 58*, 5 (Apr 1988), 345–360. URL: https://doi.org/10.1007/BF00363944, doi:10.1007/BF00363944. 2

[Bat68] BATCHER K.: Sorting networks and their applications. 307–314. 4

[BHV*16] BERTINETTO L., HENRIQUES J. F., VALMADRE J., TORR P., VEDALDI A.: Learning feed-forward one-shot learners. In *Advances in Neural Information Processing Systems* (2016), pp. 523–531. 3

[BMR*99] BERNARDINI F., MITTLEMAN J., RUSHMEIER H., SILVA C., TAUBIN G.: The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics 5*, 4 (Oct. 1999), 349–359. URL: http://dx.doi.org/10.1109/2945.817351, doi:10.1109/2945.817351. 9

[CFG*15] CHANG A. X., FUNKHOUSER T., GUIBAS L., HANRAHAN P., HUANG Q., LI Z., SAVARESE S., SAVVA M., SONG S., SU H., XIAO J., YI L., YU F.: *ShapeNet: An Information-Rich 3D Model Repository*. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 3, 6, 8

[CSKG17] CHARLES R. Q., SU H., KAICHUN M., GUIBAS L. J.:

Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), pp. 77–85. doi:10.1109/CVPR.2017.16. 3, 4

[CXG*16] CHOY C. B., XU D., GWAK J., CHEN K., SAVARESE S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. *CoRR abs/1604.00449* (2016). URL: http://arxiv.org/abs/1604.00449. 1, 2

[DSK17] DOU P., SHAH S. K., KAKADIARIS I. A.: End-to-end 3d face reconstruction with deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), vol. 00, pp. 1503–1512. URL: doi.ieeecomputersociety.org/10.1109/CVPR.2017.164, doi:10.1109/CVPR.2017.164. 1, 3

[FS05] FAVARO P., SOATTO S.: A geometric approach to shape from defocus. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 3 (March 2005), 406–417. doi:10.1109/TPAMI.2005.43. 2

[FSBO08] FAVARO P., SOATTO S., BURGER M., OSHER S. J.: Shape from defocus via diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30*, 3 (March 2008), 518–531. doi:10.1109/TPAMI.2007.1175. 2

[FSG16] FAN H., SU H., GUIBAS L.: A point set generation network for 3d object reconstruction from a single image. 1, 3, 4, 5, 6, 8, 9

[GFRG16] GIRDHAR R., FOUHEY D. F., RODRIGUEZ M., GUPTA A.: Learning a predictable and generative vector representation for objects. In *Computer Vision – ECCV 2016* (Cham, 2016), Leibe B., Matas J., Sebe N., Welling M., (Eds.), Springer International Publishing, pp. 484–499. 2

[HEH07] HOIEM D., EFROS A. A., HEBERT M.: Recovering surface layout from an image. *International Journal of Computer Vision 75*, 1 (Oct 2007), 151–172. URL: https://doi.org/10.1007/s11263-006-0031-y, doi:10.1007/s11263-006-0031-y. 2

[HFL18] HU X., FU X.-M., LIU L.: Advanced hierarchical spherical parameterizations. *IEEE Transactions on Visualization and Computer Graphics 24*, 6 (2018), 1930–1941. 3

[HWK15] HUANG Q., WANG H., KOLTUN V.: Single-view reconstruction via joint analysis of image and shape collections. *ACM Trans. Graph. 34*, 4 (July 2015), 87:1–87:10. URL: http://doi.acm.org/10.1145/2766890, doi:10.1145/2766890. 2

[JDBTG16] JIA X., DE BRABANDERE B., TUYTELAARS T., GOOL L. V.: Dynamic filter networks. In *Advances in Neural Information Processing Systems* (2016), pp. 667–675. 3

[KB14] KINGMA D., BA J.: Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. URL: http://arxiv.org/abs/1412.6980. 6

[KTCM15] KAR A., TULSIANI S., CARREIRA J., MALIK J.: Category-specific object reconstruction from a single image. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015), pp. 1966–1974. doi:10.1109/CVPR.2015.7298807. 2

[KUH17] KATO H., USHIKU Y., HARADA T.: Neural 3d mesh renderer. *CoRR abs/1711.07566* (2017). URL: http://arxiv.org/abs/1711.07566, arXiv:1711.07566. 3

[KW13] KINGMA D. P., WELLING M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013). 9

[LBSC18] LI Y., BU R., SUN M., CHEN B.: Pointcnn. *CoRR abs/1801.07791* (2018). URL: http://arxiv.org/abs/1801.07791, arXiv:1801.07791. 3

[NLX18] NIU C., LI J., XU K.: Im2struct: Recovering 3d shape structure from a single RGB image. *CoRR abs/1804.05469* (2018). URL: http://arxiv.org/abs/1804.05469, arXiv:1804.05469. 9

[PKS*17] PONTES J. K., KONG C., SRIDHARAN S., LUCEY S., ERIKSSON A. P., FOOKES C.: Image2mesh: A learning framework for single image 3d reconstruction. *CoRR abs/1711.10669* (2017). URL: http://arxiv.org/abs/1711.10669, arXiv:1711.10669. 3

[QYSG17] QI C. R., YI L., SU H., GUIBAS L. J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems 30*, Guyon I., Luxburg U. V., Bengio S., Wallach H., Fergus R., Vishwanathan S., Garnett R., (Eds.). Curran Associates, Inc., 2017, pp. 5105–5114. URL: http://papers.nips.cc/paper/7095-pointnet-deep-hierarchical-feature/learning-on-point-sets-in-a-metric-space.pdf. 3, 4

[RFB15] RONNEBERGER O., FISCHER P., BROX T.: U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (Cham, 2015), Navab N., Hornegger J., Wells W. M., Frangi A. F., (Eds.), Springer International Publishing, pp. 234–241. 5

[SBR16] SINHA A., BAI J., RAMANI K.: Deep learning 3d shape surfaces using geometry images. In *Computer Vision – ECCV 2016* (Cham, 2016), Leibe B., Matas J., Sebe N., Welling M., (Eds.), Springer International Publishing, pp. 223–240. 3

[Sch92] SCHMIDHUBER J.: Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation 4*, 1 (1992), 131–139. 3

[SHM*14] SU H., HUANG Q., MITRA N. J., LI Y., GUIBAS L.: Estimating image depth using shape collections. *ACM Trans. Graph. 33*, 4 (July 2014), 37:1–37:11. URL: http://doi.acm.org/10.1145/2601097.2601159, doi:10.1145/2601097.2601159. 2

[SSN07] SAXENA A., SUN M., NG A. Y.: Learning 3-d scene structure from a single still image. In *2007 IEEE 11th International Conference on Computer Vision* (Oct 2007), pp. 1–8. doi:10.1109/ICCV.2007.4408828. 2

[SUHR17] SINHA A., UNMESH A., HUANG Q., RAMANI K.: Surfnet: Generating 3d shape surfaces using deep residual networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), vol. 00, pp. 791–800. URL: doi.ieeecomputersociety.org/10.1109/CVPR.2017.91, doi:10.1109/CVPR.2017.91. 3

[SXJS16] SONG H. O., XIANG Y., JEGELKA S., SAVARESE S.: Deep metric learning via lifted structured feature embedding. In *Computer Vision and Pattern Recognition (CVPR)* (2016). 9

[TDB18] TATARCHENKO M., DOSOVITSKIY A., BROX T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Oct. 2018), vol. 00, pp. 2107–2115. URL: doi.ieeecomputersociety.org/10.1109/ICCV.2017.230, doi:10.1109/ICCV.2017.230. 2

[TSG*17] TULSIANI S., SU H., GUIBAS L. J., EFROS A. A., MALIK J.: Learning shape abstractions by assembling volumetric primitives. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017), pp. 1466–1474. doi:10.1109/CVPR.2017.160. 9

[WLG*17] WANG P.-S., LIU Y., GUO Y.-X., SUN C.-Y., TONG X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (SIGGRAPH) 36*, 4 (2017). 2, 6

[WSK*15] WU Z., SONG S., KHOSLA A., YU F., ZHANG L., TANG X., XIAO J.: 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015), pp. 1912–1920. doi:10.1109/CVPR.2015.7298801. 2

[ZTCS99] ZHANG R., TSAI P.-S., CRYER J. E., SHAH M.: Shape-from-shading: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*, 8 (Aug 1999), 690–706. doi:10.1109/34.784284. 2