Saliency Guided Subdivision for Single-View Mesh Reconstruction

Hai Li[1*] Weicai Ye[1*] Guofeng Zhang[1†] Sanyuan Zhang[2] Hujun Bao[1]
[1]State Key Lab of CAD&CG, Zhejiang University
[2]College of Computer Science and Technology, Zhejiang University
{garyli, yeweicai, zhangguofeng, syzhang, baohujun}@zju.edu.cn

Abstract

In this paper, we present a novel deep architecture to recover a 3D shape in triangular mesh from a single image based on mesh deformation. Most existing deformation-based methods produce uniform mesh predictions by repeatedly applying global subdivision but fail to require the highlighted details due to the memory limits. To address this problem, we propose a novel saliency guided subdivision method to achieve the trade-off between detail generation and memory consumption. Instead of using local geometric cues such as curvature, we introduce a global point-based saliency voting operation to guide the adaptive mesh subdivision and deformation explicitly. Moreover, we propose the oriented chamfer loss to mitigate the mesh self-intersection problem in subdivision. We further make our network configurable and explore the best structure combination. Extensive experiments show that our method can both produce visually pleasing results with fine details and achieve better performance compared to other state-of-the-art methods.

1. Introduction

Inferring 3D shapes from 2D images is essential to achieve the ultimate goal that makes machines see the world like humans. As a standard representation of 3D objects, triangle mesh-based learning methods for reconstruction have garnered traction recently, although there exist several potential representations such as voxel[4, 30, 12, 38, 28, 33, 37], point cloud[7, 16, 24, 39, 8], implicit surface[9, 21, 26, 22, 3], primitives[11, 6, 27, 32], etc. The reason is that the approaches based on these representations require extra non-trivial post-processing to generate the final 3D meshes. A pioneer work, Pixel2Mesh [34] progressively deforms an ellipsoid to a desired surface mesh without complex surface extraction procedure. However, the global 1-to-4 subdivision in their method results in the rapidly increasing vertices and faces which bring huge memory consumption and constrain the depth of subdivision. GEOMetrics[29] proposes an adaptive face splitting method to mitigate this problem by choosing high curvature faces to subdivide during deformation. This may lead to inconsistency between the chosen splitting faces with the detailed areas in the real model.

To address the aforementioned problem, we propose a saliency guided architecture to achieve the trade-off between detail generation and memory consumption. The saliency here is expressed by the density of the point cloud which is generated by the saliency deformation subnet.

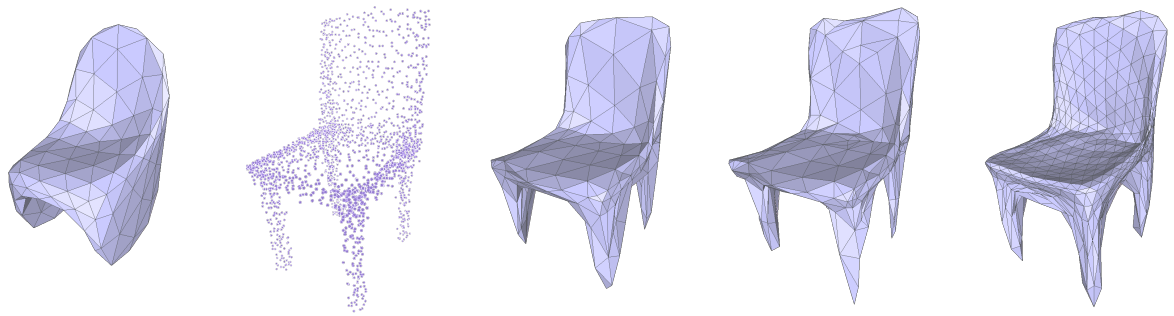
Although the saliency can be retained by diverse representations, such as heat map, curvature map, etc, here we regard it as the point cloud distribution. That is, the denser the point cloud is, the higher the probability of subdivision is, vice versa.

Our insight is that combining various 3D presentations can exploit the advantages that each representation brings. Although mesh can represent a watertight surface, it is also constrained by local smoothness and topological shape. In contrast, points can move freely in the 3D space but lack suitable geometry constraints. Based on this knowledge, we develop a point-based saliency deformation network to deform the uniformly distributed point to the target position, which represents the saliency. Then points are leveraged to guide the mesh subdivision through a voting strategy.

To mitigate the mesh self-intersection problem easily brought in mesh deformation, we introduce an oriented chamfer loss and adopt the metric Mesh-Intersection-Ratio named MIR in our evaluation to estimate the extent of the mesh self-intersection. Besides, we modularize our model to explore configurations that satisfy the best performance. The contribution of this paper can be summarized as follows:

- We propose a novel saliency guided subdivision network for mesh reconstruction by taking advantage of point cloud and mesh representation to achieve the trade-off between detail generation and memory consumption.

*Equal contribution
†Corresponding author



(a) coarse mesh (b) saliency point (c) adaptive subdivision x1 (d) adaptive subdivision x2 (e) global subdivision

Figure 1. Deform a coarse shape (a) to more meticulous shapes (c) (d) by repeatedly using adaptive subdivision under the guidance of saliency points (b). Our method can generate finer meshes with better details but require much fewer faces compares to the global subdivision (e).

.

- We introduce an oriented chamfer loss to reduce the self-intersection phenomenon and propose the MIR metric to verify the effectiveness.
- Our model is configurable and easy to explore the best structure combination. Extensive experiments demonstrate that our method can not only produce more visually pleasing results with fine details but also achieve better performance compared to previous methods.

2. Related Work

2.1. 3D Reconstruction

Due to the variety of 3D representations, the learning-based reconstruction methods are separated into several branches. The early works [4, 37, 38] prefer to employ volumetric representation for voxels have compact and ordered structure which could be fed into the encoder-decoder network directly. However, the memory consumption of 3D voxels highly constrains the resolution. Wang et al. 2018[35] proposed an Octree-based method to reduce the number of voxels by introducing complicate tree structure into training, which increases the computation complexity. Fan et al. 2017[7] first proposed to utilize point cloud, which significantly reduces the burden of memory. Although points can capture the properties of shape, its sparseness makes surface extraction fail in some cases. The same problem happens on some implicit surface methods [21]. To overcome this problem, Groueix et al. 2018[11] adopts the learnable surface patches to compose the shape while [35] exploits adaptive octree based patches to represent the whole surface. These methods also need extra post-processing because their output surfaces are not watertight. Wang et al. 2018[34] proposed a mesh deformation based method to output watertight surface without further post-processing directly. The following work [10] attempts to deal with holes using the low-resolution volume, while [25] directly prunes faces on mesh which deviates significantly from the ground truth. Wen et al. 2019[36] further extended Pixel2Mesh to multi-view scenery. Our method is also based on mesh deformation but pays more attention to mesh subdivision.

2.2. Mesh Adaptive Subdivision

Although mesh representation is compact and scalable, it still needs massive faces to present detail parts. In Pixel2Mesh [34], the global 1-to-4 subdivision is carried out hierarchically. The rapidly increased vertices and faces bring massive memory consumption which constrains the depth of subdivision. GEOMetrics [29] proposed an adaptive face splitting method to mitigate this problem. They choose the high curvature regions at the end of each deformation block and add new vertices in the middle of these faces according to Sqrt(3) subdivision [18]. The main drawback of this strategy is that the splitting faces may not be consistent with the detailed areas in the real model, thanks to lacking guidance from the global information. However, our saliency region is trainable and supervised by the real detail location selected from the ground truth. Then the mesh deformation network produces finer mesh based on the saliency.

3. Proposed Method

Our objective is to design a network to reconstruct a simplified and complete surface mesh directly from a single 2D image without further post-process. The framework is illustrated in Figure 2. The whole network extends Pixel2Mesh [34] to a configurable framework and mainly consists of three sub-networks, namely image feature extraction network, saliency point deformation network and mesh deformation network. The feature map extracted by the image feature network is associated with points and mesh vertices via projection in the other two networks. The saliency point

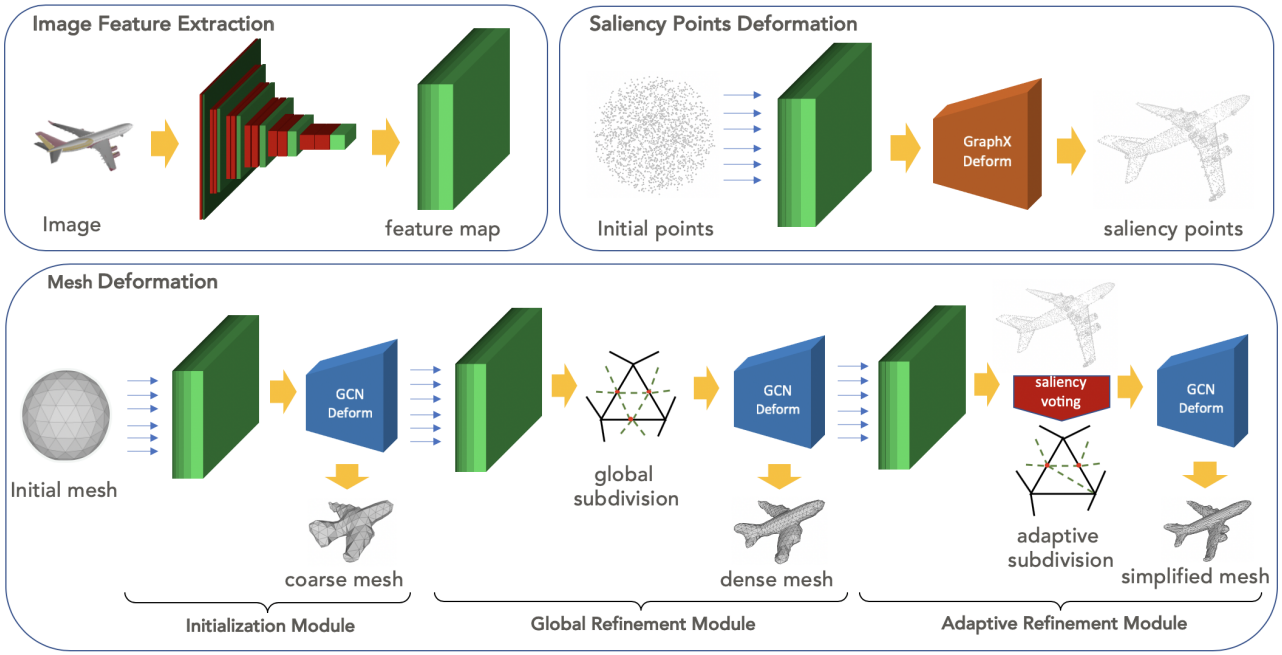


Figure 2. Overview of our framework. Our network consists of three sub-networks: an image feature extraction network (upper-left), a saliency point deformation network (upper-right) and a mesh deformation network (bottom). The mesh deformation network is further divided into three modules: initialization module, global refinement module, and adaptive refinement module. Our network support configurable settings which mean the last two modules of deformation network can be iterated as many times as desired respectively.

deformation network deforms the initial point set with local point feature and image feature to the salient regions. The output point set further guides the selection of subdivision areas through a saliency voting strategy. The mesh deformation network deforms an initial mesh with local mesh feature and image feature to a final shape mesh in a coarse to fine manner. The details of each component are described in the following.

3.1. Feature Association via Projection

We use a VGG-16 encoder network to extract image features from the input image (Figure 2 top-left). Since we cannot know which part of the information will be used in the other networks, we construct a feature map that concatenates the features from conv0_2, conv1_3, conv3_3, conv4_3, and conv5_4 via inverse pooling operation to ensure the feature map has the same height and width as the input image. The generated feature map contains both low level local features and high level global features. We then project the 3D points and mesh vertices from the other two sub-networks into the feature map using the projection matrix. To retrieve the corresponding image feature from specific location, the bilinear interpolation strategy is utilized to gather features from nearby pixels. A fully connected layer is then used to distill the image feature for different hierarchies and the local 3D position coordinates are then concatenated to compose the 3D features for points and mesh vertices.

3.2. Saliency Deformation SubNet

We uniformly sample 2048 points in a unit sphere as an initial point cloud, these points first get corresponding features via projection described in Section 3.1. Then a point set deformation network is leveraged to get a complete point cloud with detail information shown in Figure 1(b). The salient points are sparse in non-detailed areas but much denser in fine-detailed areas such as sharp edges. We adopt the GraphX Deformation module proposed in PCDNet [23] as our point deformation network. GraphX Deformation network is constructed by multiple GraphX layers which shared similar idea with $\mathcal{X}$-conv [19]. The mathematical form of GraphX is defined as

$$f_k^{l+1} = \sigma(\mathrm{W}^{\mathrm{T}}(\sum_{f_i^l \in \mathcal{F}^l} w_{ik} f_i^l + b_k) + \mathrm{b}) \qquad (1)$$

where f_k^l denotes the kth point feature of l layer in $\mathcal{F}^l$, $w_{ik}, b_k \in \mathbb{R}$ are trainable mixing weight and bias scalar. W, b are trained weight matrix and bias vector of the fully-connected layer, σ is activation function. Unlike PCDNet [23], our point deformation branch does not directly calculate the final position of 3D points, but offsets of initial points and supervised by simplified points generated in Section 4.1.2 with the saliency loss demonstrated in Section 8.

We found that this modification can generate better point clouds, and can reduce the phenomenon of point clouds being over-aggregated. These points are distributed around the highly detailed location, which can be a good guidance to subdivide and deform the mesh (Figure 2 top-right).

3.3. Mesh Deformation SubNet

Mesh Deformation deforms an initial sphere with 176 vertices and 320 faces to a desired shape mesh. For different purposes, we can further divide this subnet into three modules: initialization module, global refinement module and adaptive refinement module (Figure 2 bottom). The first two modules are similar to Pixel2Mesh [34]. The initialization module deforms the sphere into a coarse shape without subdivision (Figure 1(a)) while the global refinement module takes in the coarse shape and deforms into a much nicer one via 1-to-4 global subdivision. The main difference compared to Pixel2Mesh is that we train the model through the supervision of the per-vertex offset instead of the absolute position, which produce the more stable and visually appealing results, shown in Figure 1(e). The deformation network is a stack of ResGCN layers. The vanilla GCN layer proposed in [17] has the following propagation:

$$\mathrm{H}^{l+1} = \sigma(\mathrm{D}^{-\frac{1}{2}}\mathrm{A}\mathrm{D}^{-\frac{1}{2}}\mathrm{H}^{l}\mathrm{W}^{l}) \tag{2}$$

where σ is activation function, A is adjacent matrix with self connection, $\mathrm{D} = \sum \mathrm{A}_{ij}$, and W is learnable weight. A ResGCN layer is proposed in Pixel2Mesh to enhance local 3D features by skip connection [13].

3.3.1 Saliency Voting & Adaptive Subdivision Module

The global refinement module increases the number of faces exponentially, which causes enormous memory cost and makes continuous subdivision impossible. To address this problem, we borrow the idea from traditional adaptive subdivision and only subdivide the desired area. Since the generated point clouds from the saliency deformation subnet explicitly reflect the distribution of the saliency region. To align the two representations, we need first to identify the area on mesh consistent with saliency points.

We introduce a saliency voting operation to select desired areas. We first compute the nearest edge on the mesh for each point from saliency points and a voting strategy is then utilized to pick up top K voted edges. The chosen edges are split into two edges by adding a new vertex in the middle. To keep the valence of vertex and prevent the appearance of cracks, we adopt the Green Rule from algorithm [1] to generate new edges which are detailed in Figure 3. The usage of top K edge selection and the Green Rule can keep the equal edge size for each mesh in a mini-batch.

The initialization module is always needed while the last two modules can be combined freely. For example, we can remove the global subdivision module and only stack adaptive subdivision modules which significantly reduce the face number but present high-resolution surfaces in complex areas and leave the flat areas in low-resolution (Figure 1(c) and Figure 1(d)). The exploration of several combinations is detailed in Section 4.4.

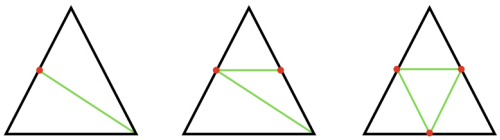


Figure 3. A red dot represents a new vertex generate from the subdivision, and a green line represents a new edge generated by Green Rule. The figures from left to right show the edge generation for different new vertex number.

3.4. Loss Function

The output of the network is a triangle mesh $M = (V, E)$, where $V = v \in \mathbb{R}^3$ is a set of vertex and $E \subseteq V \times V$ is a set of edges. To penalize the irregular phenomenons and enforce the geometric constraints on the predict mesh surface, we adopt the following loss functions.

Edge loss is the key to make vertices on the mesh distribute uniformly by punishing the over large edges (Equation 3).

$$L_{edge}(V, E) = |E|^{-1} \sum_{(v,v')\in E} \|v - v'\|^2 \tag{3}$$

Laplace loss is used as local smoothness constraint. However, over-smoothing could cause a loss of accuracy and lack of details.

$$L_{lap}(V) = |V|^{-1} \sum_{v\in V} \|v - \sum_{k\in N(v)} |N(v)|^{-1} \cdot k\|^2 \tag{4}$$

where $N(v)$ denotes the neighbors of vertex v.

Move loss aims to fix the topology between two continuous modules' output mesh vertices V and V' and keep shape consistency through the whole network.

$$L_{move}(V, V') = |V|^{-1} \sum_{v\in V, v'\in v'} |v' - v| \tag{5}$$

3.4.1 Oriented Chamfer Loss

The chamfer loss is widely used in deep shape reconstruction problem by matching two unordered point set. The common form of chamfer loss is shown in Equation 6 while $P, Q \in \mathbb{R}^3$ denote two point sets and $\Lambda_{P,Q}$ denotes the near-

est point pairs.

$$L_{chamfer}(P,Q) = |P|^{-1} \sum_{(p,q)\in\Lambda_{P,Q}} \|p-q\|^2 + |Q|^{-1} \sum_{(q,p)\in\Lambda_{Q,P}} \|q-p\|^2 \quad (6)$$

In the mesh deformation based method, we can get not only the position of sampled points but also their normal direction since they all lie on the surface. If two points have opposite normal direction, this loss will cause the problem like self-intersection. To this end, we propose an oriented chamfer loss to force the normal consistency of matching point pairs. Unlike other methods [34, 10] who penalize the inconsistency normals after nearest points matching, we directly plug an oriented constraint in the matching process

$$\Lambda^*_{Q,P} = \{(p, argmin_{(q|n_p \cdot n_q > \theta)} \|p-q\|)\} \quad (7)$$

where n_p, n_q are the normal of sampled points on predict and ground truth mesh and θ is a predefined threshold. We denote chamfer loss with the oriented constraint as $L_{oriented}$.

3.4.2 Saliency Loss

To supervise the saliency point deformation, we compute the weighted chamfer loss between the saliency points from point cloud deformation network with the simplified points generated from Mesh Decimation (Section 4.1.2). The different weights w_q, w_p for two matching directions control the density of aggregation.

$$L_{saliency}(P,Q) = w_p \cdot |P|^{-1} \sum_{(p,q)\in\Lambda_{P,Q}} \|p-q\|^2 + w_q \cdot |Q|^{-1} \sum_{(q,p)\in\Lambda_{Q,P}} \|q-p\|^2 \quad (8)$$

3.5. Implementation Details

Our network is implemented in Pytorch and we use Adam Optimizer with decay as 5e-6 and set the batch size as 6 on a Titan Xp GPU. The model is trained for 70 epochs totally with learning rate 0.75e-4. For the adaptive refinement module, we chose the top $\frac{1}{5}$ of the total number of edges ordered by the voting operation for subdivision. We sample 10k points on the predicted meshes using the sampling method presented in Section 4.1.3.

The total loss function of our system is defined as follows:

$$L_{total} = \lambda_1 L_{edge} + \lambda_2 L_{lap} + \lambda_3 L_{move} + \lambda_4 L_{oriented} + \lambda_5 L_{saliency} \quad (9)$$

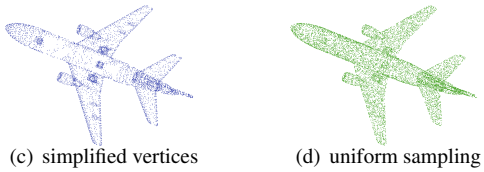
We set threshold $\theta = 0$ in the oriented chamfer loss, and set $\lambda_1 = 0.3, \lambda_2 = 0.3, \lambda_3 = 0.1, \lambda_4 = 1.0, \lambda_5 = 1.0$. To encourage the vertices to grow to the salient areas, we remove the edge loss in the adaptive subdivision module and weaken the move loss for vertices adjacent to newly generated vertices.

4. Experiments

To evaluate the performance of the proposed method, we have conducted experiments across 13 classes of ShapeNet [2] dataset like the other state-of-the-art methods. We further study different structure combinations and explore the best structure.

4.1. Data Preparation

(a) origin (b) watertight



(c) simplified vertices (d) uniform sampling

Figure 4. Data pre-processing example. (a) the original shape mesh from ShapeNet, (b) the extracted watertight 2-manifold surface, (c) the vertices after mesh decimation, (d) the point cloud after uniform sampling.

In our experiments, we adopt ShapeNetCore from ShapeNet [2] and follow the train/test split provided by Pixel2Mesh[34]. The mesh models in ShapeNet cannot be directly exploited since some of them have complicated inner structure and not guarantee the correct 2-manifold topology (Figure 4(a)). The pre-processing is essential to generate a suitable training and evaluation set. We take the following procedures to create our dataset.

4.1.1 Surface Extraction

We adopt the surface generation method proposed in [15] which utilizes octree structure to represent original mesh and a modified marching cube [20] algorithm to generate 2-Manifold surface (Figure 4(b)).

4.1.2 Mesh Decimation

To get proper supervision for saliency deformation, we apply mesh decimation to the 2-Manifold surface which suc-

Category	3D-R2N2	PSG	P2M	GEOMetrics	P2M*	GEOMetrics*	Ours
Plane	41.46	68.20	71.12	89.00	83.62	80.90	**84.75**
Bench	34.09	49.29	57.57	72.11	66.75	65.98	**71.35**
Cabinet	49.88	39.93	60.39	59.52	55.62	51.73	**56.44**
Car	37.80	50.70	67.86	74.64	71.52	69.22	**72.43**
Chair	40.22	41.60	54.38	56.61	56.85	57.41	**61.82**
Monitor	34.38	40.53	51.39	59.50	50.63	48.75	**52.11**
Lamp	32.35	41.40	48.15	58.65	56.49	56.05	**60.32**
Speaker	45.30	32.61	48.84	49.53	44.27	41.85	**46.52**
Firearm	28.34	69.96	73.20	88.30	86.44	84.04	**87.19**
Couch	40.01	36.59	51.90	59.54	54.33	53.70	**57.56**
Table	43.79	53.44	66.30	66.33	64.77	63.27	**68.18**
Cellphone	42.31	55.95	70.24	73.65	62.70	59.83	**65.05**
Watercraft	37.10	51.28	55.12	68.32	62.54	64.52	**66.90**
Mean	39.01	48.58	59.72	67.37	64.26	63.01	**67.20**

Table 1. Quantitative Reconstruction Results. The leftmost 4 columns are reported with the F1 score. The rightmost 3 columns are evaluated on our dataset and unified evaluation metric to make fair comparison. The proposed method outperforms Pixel2Mesh and GEOMetrics in the unified evaluation.

cessively collapses the edges [14] sorted by quadratic error. In this way, the remaining vertices represent the saliency or high detailed region (Figure 4(c)).

4.1.3 Mesh Sampling

Uniform sampling on surface is necessary since we need to use the chamfer loss to constrain the deformation. We make use of the differentiable sampling method (Equation 10) applied in [29, 10, 36].

$$r = (1 - \sqrt{u}) \cdot v_1 + \sqrt{u} \cdot (1 - w) \cdot v_2 + \sqrt{u} \cdot w \cdot v_3 \quad (10)$$

Where v_1, v_2, v_3 are the vertices of the face. u and w are sampled from a uniform distribution. The probability of chosen face is proportional to its area. We sample 10k points as our ground truth (Figure 4(d)).

4.1.4 Rendering

We rendered each model from 2 random viewpoints with a fixed radius. The rendered image size is 224 $\times$ 224. To make our model robust in various light conditions, we also add several random light sources during rendering.

4.2. Metrics

4.2.1 F score

According to the report [31], F-score explicitly evaluates the distance between object surfaces and is defined as the harmonic mean between precision and recall. We sample 10k points from the predicted mesh and set $\tau = 0.0001$ as the threshold. To compare with previous works, we set the scaling factor to 0.57.

4.2.2 Intersection Ratio

We further propose the Mesh-Intersection-Ratio metric named MIR to measure the extent of mesh self-intersection phenomenon. We first detect the self-intersection faces via MeshLab [5] and then compute the ratio of the self-intersection area to the total area. However, in the adaptive scenario, different subdivision depth leads to a highly unequal face area. Therefore, we additionally adopt the self-intersection face number ratio as complementary

$$MIR_{area} = \frac{intersect\ face\ area}{total\ face\ area} \quad (11)$$

$$MIR_{size} = \frac{intersect\ face\ number}{total\ face\ number} \quad (12)$$

4.3. Reconstruction Comparisons

We evaluate the performance of the proposed method quantitatively by comparing the ability to recover the target object from a single image against other single-view reconstruction works [4, 7, 34, 29]. The quantitative results are shown in Table 1. We mainly compare with two mesh deformation based method Pixel2Mesh and GEOMetrics. However, we found they adopted the inconsistent datasets and different evaluation strategies, which made the results ambiguous. To make fair comparison, we have modified their original implementations [1] [2] denoted as P2M* and GEOMETRIC* to match our dataset and evaluated them in the unified metric (rightmost 3 columns). Our framework for achieving optimal results includes 1 global optimization module and 1 adaptive optimization module. As shown, our

[1] https://github.com/nywang16/Pixel2Mesh

[2] https://github.com/EdwardSmith1884/GEOMetrics without latent loss

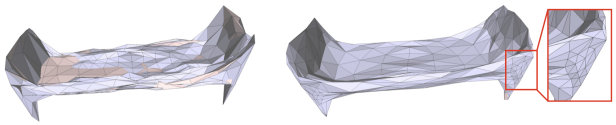


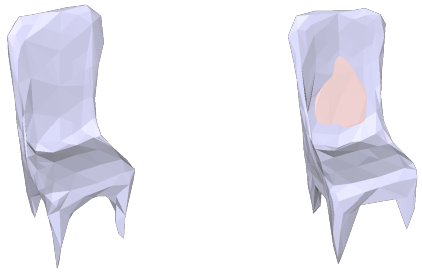
Figure 5. Result comparison between GEOMetrics (left) and ours (right). Our method faithfully subdivides corners or other detail areas while GEOMetrics's subdivision areas are unstable. Meanwhile, our method has less self-intersection compared with GEOMetrics.

Metric	Vertex Size	Face Size	F1 Score
P2M*	2466	4928	64.26
GEOMetrics*	558	1112	63.01
Ours(2a)	**411**	**818**	64.64
Ours(1g1a)	1026	2048	**67.20**
Ours(2g)	2562	5120	66.58

Table 2. The result of different module combinations. a and g stand for adaptive refinement module and global refinement module respectively.

network achieves much higher performance than previous approaches and improves scores in all classes.

Qualitative reconstruction results for those having meticulous structure are shown in Figure 7. We acquire highly accurate reconstruction of the input objects effectively capturing not only the global structure but local details. Figure 5 shows the subdivision result between GEOMetrics [29] and ours. Compared to GEOMetrics which produces the ambiguous and more self-intersection subdivision results, our saliency guided subdivision network produces more pleasing and stable results that concentrate on the corner or other detail region.



(a) w/o oriented chamfer loss (b) w/ oriented chamfer loss

Figure 6. Visualize the effectiveness of the oriented chamfer loss, the pink part stands for the reverse normal caused by intersection.

4.4. Structure Exploration

We make our model configurable and further explore the performance under different configurations. The quantitative results are shown in Table 2. We summarize the reasons for this result. The global refinement module enforces the smoothness and edge length constraint which suppresses the growth of high detailed regions and the adaptive refinement module increases the vertices in more sharp areas like edges or corners. The result reveals that too few vertices and faces fail to well capture the global structure, while too much global refinement modules hinder the representation of local details. Since one global and one adaptive refinement modules take a good balance of both, therefore obtaining the best performance.

The vertex and face number can be easily computed since each global subdivision module generates 4 new faces from each original face and each adaptive subdivision module generates one new vertex from one-fifth of original edges described in Section 3.5. Figure 1 reveals that the proposed saliency guided subdivision architecture brings about stunning results with few faces compared to the global subdivision method.

Category	Ours (w/o oriented) (area / face / size)	Ours (w/ oriented) (area / face / size)
Chair	0.0808 / 0.0495 / 818	**0.0483** / **0.0325** / 818

Table 3. MIR comparison of the oriented chamfer loss

Category	GEOMetrics* (area / face / size)	Ours (w/ oriented) (area / face / size)
Mean	0.2382 / 0.3151 / 1112	**0.0951** / **0.0733** / 818

Table 4. MIR for GEOMetrics* and our method

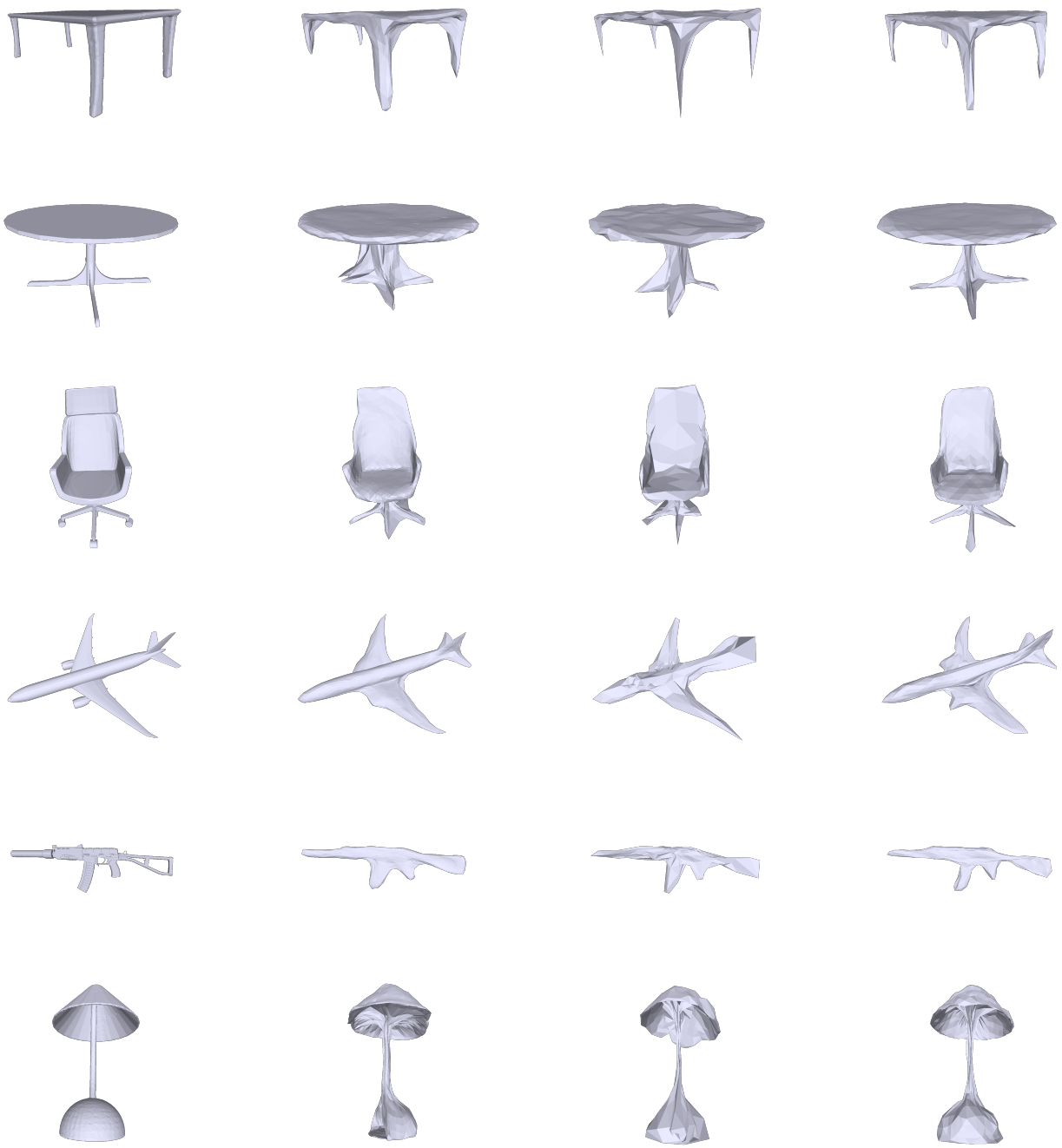
4.5. Ablation Study

To validate the effectiveness of the oriented chamfer loss, we first evaluate the proposed model using 2 adaptive refinement modules on single chair category. The quantitative results in Table 3 shows that the intersection ratio and size decrease significantly with the oriented chamfer loss. Area and face represent intersection face area ratio and face number ratio, respectively. Size denotes the average face number for each class. The quality result in Figure 6 demonstrates that the oriented chamfer loss mitigates the problem of surface overlap effectively.

We further compare our model with 2 adaptive subdivision module against GEOMetrics*. The results are shown in Table 4 depicts that our model produces fewer faces but gets far fewer intersection ratio.

5. Conclusions

In this paper, we propose a novel adaptive subdivision architecture for mesh reconstruction guided by saliency points which take advantage of both representations and achieve the trade-off between detail generation and memory con-



(a) GroundTruth (b) Pixel2Mesh[34] (c) GEOMetrics[29] (d) Ours

Figure 7. Qualitative results by our network. Compare to Pixel2Mesh* and GEOMetrics*, our method generates meshes with better details.

sumption. The proposed oriented chamfer loss significantly reduce the self-intersection phenomenon. The experiments demonstrate that our method can not only produce more visually appealing results but also achieve better performance compared to the state-of-the-art methods.

Acknowledgement

This work was partially supported by NSF of China (Nos. 61822310 and 61672457), and the Fundamental Research Funds for the Central Universities (No. 2019XZZX004-09).

References

[1] R. Bank and A. Sherman. Some refinement algorithms and data structures for regular local mesh refinement. *Applications of Mathematics and Computing to the Physical Sciences*, 1999. 4

[2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[3] Z. Chen and H. Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1

[4] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proceedings of the European Conference on Computer Vision*, 2016. 1, 2, 6

[5] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. Meshlab: an open-source mesh processing tool. In *Proceedings of the Eurographics Italian Chapter Conference*, 2008. 6

[6] B. Deng, K. Genova, S. Yazdani, S. Bouaziz, G. Hinton, and A. Tagliasacchi. Cvxnets: Learnable convex decomposition. *arXiv preprint arXiv:1909.05736*, 2019. 1

[7] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2, 6

[8] M. Gadelha, R. Wang, and S. Maji. Multiresolution tree networks for 3d point cloud processing. In *Proceedings of the European Conference on Computer Vision*, 2018. 1

[9] K. Genova, F. Cole, D. Vlasic, A. Sarna, W. T. Freeman, and T. Funkhouser. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 1

[10] G. Gkioxari, J. Malik, and J. Johnson. Mesh r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2, 5, 6

[11] T. Groueix, M. Fisher, V. G. Kim, B. Russell, and M. Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2

[12] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. In *Proceedings of the International Conference on 3D Vision*, 2017. 1

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 4

[14] H. Hoppe. Progressive meshes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, 1996. 6

[15] J. Huang, H. Su, and L. Guibas. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018. 5

[16] E. Insafutdinov and A. Dosovitskiy. Unsupervised learning of shape and pose with differentiable point clouds. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2018. 1

[17] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations*, 2017. 4

[18] L. Kobbelt. 3-subdivision. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, 2000. 2

[19] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen. Pointcnn: Convolution on x-transformed points. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2018. 3

[20] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, 1987. 5

[21] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2

[22] M. Michalkiewicz, J. K. Pontes, D. Jack, M. Baktashmotlagh, and A. Eriksson. Deep level sets: Implicit surface representations for 3d shape inference. *arXiv preprint arXiv:1901.06802*, 2019. 1

[23] A.-D. Nguyen, S. Choi, W. Kim, and S. Lee. Graphx-convolution for point cloud deformation in 2d-to-3d conversion. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3

[24] D. Novotny, D. Larlus, and A. Vedaldi. Learning 3d object categories by looking around them. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017. 1

[25] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia. Deep mesh reconstruction from single rgb images via topology modification networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2

[26] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1

[27] D. Paschalidou, A. O. Ulusoy, and A. Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1

[28] E. Smith and D. Meger. Improved adversarial systems for 3d object generation and reconstruction. *arXiv preprint arXiv:1707.09557*, 2017. 1

[29] E. Smith, A. Romero, S. Fujimoto, and D. Meger. Geometrics: Exploiting geometric structure for graph-encoded objects. In *Proceedings of the IEEE International Conference on Machine Learning*, 2019. 1, 2, 6, 7, 8

[30] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017. 1

[31] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 6

[32] S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik. Learning shape abstractions by assembling volumetric primitives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[33] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[34] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision*, 2018. 1, 2, 4, 5, 6, 8

[35] P.-S. Wang, C.-Y. Sun, Y. Liu, and X. Tong. Adaptive O-CNN: A Patch-based Deep Representation of 3D Shapes. *ACM Transactions on Graphics*, 2018. 2

[36] C. Wen, Y. Zhang, Z. Li, and Y. Fu. Pixel2mesh++: Multi-view 3d mesh generation via deformation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2, 6

[37] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2017. 1, 2

[38] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Proceedings of the Conference on Neural Information Processing Systems*, 2016. 1, 2

[39] K. Yin, H. Huang, D. Cohen-Or, and H. Zhang. P2p-net: Bidirectional point displacement net for shape transform. *ACM Transactions on Graphics*, 2018. 1