Geometric Granularity Aware Pixel-to-Mesh

Yue Shi Bingbing Ni† Jinxian Liu Dingyi Rong Ye Qian Wenjun Zhang
Shanghai Key Laboratory of Digital Media Processing and Transmission
Shanghai Jiao Tong University, Shanghai 200240, China
{shiyue001, nibingbing, liujinxian, r892546826, qianye001, zhangwenjun}@sjtu.edu.cn

Abstract

Pixel-to-mesh has wide applications, especially in virtual or augmented reality, animation and game industry. However, existing mesh reconstruction models perform unsatisfactorily in local geometry details due to ignoring mesh topology information during learning. Besides, most methods are constrained by the initial template, which cannot reconstruct meshes of various genus. In this work, we propose a geometric granularity-aware pixel-to-mesh framework with a fidelity-selection-and-guarantee strategy, which explicitly addresses both challenges. First, a geometry structure extractor is proposed for detecting local high structured parts and capturing local spatial feature. Second, we apply it to facilitate pixel-to-mesh mapping and resolve coarse details problem caused by the neglect of structural information in previous practices. Finally, a mesh edit module is proposed to encourage non-zero genus topology to emergence by fine-grained topology modification and a patching algorithm is introduced to repair the non-closed boundaries. Extensive experimental results, both quantitatively and visually have demonstrated the high reconstruction fidelity achieved by the proposed framework.

1. Introduction

Mesh is a widely used 3D representation, especially in virtual/augmented reality, animation and game industry, for its capability of modeling geometric details. As an alternative or auxiliary of traditional manual mesh, 3D reconstruction from pixel level attracts a growing attention and has achieved promising results. According to the format of the generated 3D model, existing reconstructed methods can be divided into image to voxel, image to point-cloud, image to mesh, etc [22]. More recently, implicit representation [41, 8, 58] has been used for 3D reconstruction and some new reconstructing forms also come up, such as NeRF [37]. Among these methods, 3D mesh reconstruction attracts much attention because it is the most popular shape representation in game and movie industries. Recent progress in single-view mesh reconstruction proposes to reconstruct a 3D mesh by deforming a template model based on the perceptual features extracted from the input image [51]. Though promising results have been achieved, the reconstructed results are unsatisfactory on local details and non-zero genus objects.

The first challenge in reconstruction is to generate precise and rich details. Existing methods mainly utilize the chamfer distance between two point clouds, which are sampled from predicted mesh and the ground truth respectively, to restrict the reconstruction. Although some other constraints are added, such as normal loss [27] which enforces the consistency of surface normal and edge length regularization to prevent outliers [45]. All of them only focus on the point clouds, without considering the topology of mesh. Replacing mesh with point cloud smooths out local structure details and corresponding local structure information, leading to coarse reconstruction details. Besides, it is not reasonable to treat all parts of a mesh equally for the uneven information distribution on it, which means high-fidelity parts need extra attention. Treating the mesh as point cloud and in-
discriminately overall processing of existing methods show disadvantages in parts with large curvature changes and rich details, like table corners and chair legs, represented by abnormally smooth and even disordered connection on the reconstructed mesh. The lack of details leads to noticeable gaps between reconstructed mesh and manual one, which affects practical application.

Another obstacle is to reconstruct non-zero genus objects, where the holes need to be reproduced. This problem is especially obvious in pixel-to-mesh reconstruction, where initial template is imported and the network is trained to guide movements of vertices. According to topology deformation theory [52, 2], the objects of same genus are homeomorphism and can be transformed into each other through deformation while objects of different genus cannot. This suggests the initial template restricts the reconstruction results, which we show in Figure 2. Thus, how to realize the change of genus is a key problem to precise mesh reconstruction. Mesh edit provides a way out for the change of genus is a key problem to precise mesh reconstruction. The framework consists of an adaptive pruning module and a patching repair module. By the GSE and the Fine-grained Edit modules, our framework realizes geometry granularity-aware reconstruction and reformulates the pipeline of pixel-to-mesh. To the best of our knowledge, our framework is the first to realize integral reconstruction while focusing on the two core issues in reconstruction at the same time. It’s also the first to explicitly extract local features of meshes and realize fine-grained edit.

The contributions of this paper can be summarized as:

- We propose a Geometry Structure Extractor (GSE) for mesh data to locate key areas and explicitly extract local geometry information, which helps to retain structure information of the graph data in Hilbert high-dimensional space.
- We design a Multi-scale Shape Preserving Constraint to facilitate pixel-to-mesh mapping and then validate its effectiveness on a deformation-based reconstruction network.
- We introduce a Fine-grained Mesh Edit module which consists of an adaptive pruning module and a patching repair algorithm that break the restriction of the template mesh and allow objects of various genus to emergence.

2. Related Work

2.1. Mesh Reconstruction from Pixel

Mesh Reconstruction from Pixel can be divided into two routes, including indirect and direct approaches. The former reconstructs 3D model in another format, such as point cloud and voxels, and then translates it into mesh. Point cloud reconstruction [14, 13] and voxel reconstruction [17, 18] are common explicit 3D reconstructions. 3D R2N2 [11] proposed a voxel reconstruction framework, which is the representation of monocular 3D shape generation using deep learning techniques. Besides, AtlasNet [19] based on parametric representation also emerged. But it needs to solve how to stitch multiple meshes tightly together. Recently, implicit representations are widely applied in reconstruction [60, 10, 41, 9, 36, 10]. They are typically created with a pipeline that couples simultaneous localization, mapping-based pose estimation, and depth image integration using Signed Distance Functions (SDFs). While it is possible to produce accurate shapes using above methods, models may suffer from missing parts and holes due to the translation to mesh by Marching Cube algorithm [34, 15].
Comparing with the indirect methods, reconstructing mesh directly can avoid information loss caused by post-processing. Most existing direct mesh reconstruction methods are deformation-based [49, 56], which regress to coordinate movements from the image feature and the ground truth model. Pixel2mesh [51] propose the earliest end-to-end deep learning architecture that produces a 3D shape in triangular mesh from a single color image. Pixel2mesh++ [53, 25] introduces multi-view images and utilizes the distance between images and the projection of the reconstructed model at different angles to refine the reconstruction. However, there are two core issues that hinders the further development of reconstruction. One is the restriction of the initial template. The other is the fine-grained reconstruction of geometry. For the first challenge, Kanazawa et al. [24, 16] propose to find a more likely template by retrieving, which is hard to generalize. Pan et al. [40] reconstructed holes on the mesh by prune faces with large error possibility. However, the prune is coarse and it also leads to non-closed boundaries which make mesh abnormal. For the second problem, Smith et al. [46] utilize adaptive splitting to allow detail to emerge, which alleviates the coarse details problem to some extent. However, it has no special consideration for local geometry structure, equally treating every part of the mesh. Tang et al. [48] roundbreaking realized direct mesh reconstruction by a skeleton-bridged approach. But it cannot be trained end-to-end.

2.2. Geometry Structure Similarity

Measuring the similarity between geometry structures is critical in the mesh reconstruction. Existing reconstruction methods roughly equal the similarity between meshes to the similarity between their sampled point clouds, which leads to the loss of shape information. Actually, mesh is a fully connected graph and its local structures can be described by local subgraphs. Then the geometry similarity can be measured by the similarity of the subgraphs.

Graph similarity is calculated mainly by Kernel methods, which can be summarized into two categories, graph embedding and graph kernel algorithm [38]. The former vectorizes graph, structures and utilizes vector kernels like RBF kernel and Sigmoid kernel, represented by GCNs [6, 12, 35] which are widely applied in pixel to mesh reconstruction [30, 50]. But this kind of methods reduce the dimensional of structure data to vector space and loses a lot of structure information. The graph kernel algorithm [50] directly makes use of graph structure data, which not only retains the advantages of efficient kernel function calculation but also contains the structure information of graph data in Hilbert’s high-dimensional space. According to the difference of kernel function, common graph kernel algorithms can be divided into Graphlet kernels, Weisfeiler-Lehman subtree kernels, and Shortest-PathKernels [30]. We import Wasserstein Weisfeiler-Lehman Graph Kernel, which integrates Wasserstein distance, preserves features of the node, and proposes a graph embedding scheme.

2.3. Mesh Edit

Most researches about pixel to mesh use a genus-0 3D model such as sphere and ellipse as their initial model [24, 1, 42, 49, 59], while some others find proper initial template by retrieval [31, 49]. However, all of them cannot cope with the genus difference between the template and the target. Pan et al. [40] propose to reconstruct holes by estimating the face error and pruning the incorrect part. However, it has two disadvantages. The first is that the number of faces on the template is limited, for which it is very possible to prune a bigger part than the real wrong part. In order to realize fine-grained prune, we adopt the main idea of mesh subdivision [33, 26, 43, 21], which is also utilized by Smith et al. [46] to encourage details to emergence. Secondly, pruning leads to non-closed boundaries and abnormal mesh structure, which will affect its application. This problem can be resolved by repairing the boundaries. There have been some mesh repair methods [3, 23, 44], which can recovery normal mesh by patching. However, they are not suitable for objects born with holes. Using existing filling holes method directly will filling the holes, instead of repair the boundaries smartly. So inspired by methods mentioned above, we propose a novel repair algorithm in section 3.2.

3. Methodology

The existing mesh reconstruction models mainly utilize Chamfer Distance (CD) between the predict mesh and the ground truth to constrain the deformation of the initial template, while some constraints such as normal loss, laplacian loss, and edge loss are accompanied sometimes. All these indicators only describe the similarity between two point sets, which are sampled from faces of predicted and ground truth meshes. Due to the neglect of topology structure, neither of them performs well on complicated shapes. Besides, deformation-based reconstruction is limited by the initial template, the general genus 0 ball cannot reconstruct non-zero topology accurately. Focusing on the two problems, we propose an enhanced deep pixel to mesh framework which explicitly addresses the challenge of fine-grained detail reconstruction. Figure 3 overviews our framework. The framework consists of three modules: 1. Multi-scale Geometry structure Extractor; 2. Geometry-aware Mesh Deformation; 3. Fine-grained Mesh Edit Module. The details of each module are described below.

3.1. Geometry Structure Extractor

Mesh of an object can be taken as a large-scaled undirected graph. However, its connection information and topology structure are rarely used by previous methods. Although deformation-based methods put an ellipsoid into GCN or CNN [32] and utilize losses such as Laplacian loss to restrict the edge of the predicted mesh, they never learn the geometry structure of ground truth meshes. In order to take advantage of the geometry structure of mesh data, we propose an extractor that can detect semantic key points and explore local geometry. Compared with embedding methods
which encode the whole structure into vector space, GSE is more flexible and keen. It focuses on local details and retains more geometry structure information in Hilbert high-dimensional space.

Locate Key Areas According to the law of human perception, we first focus on the overall shape and secondly pay attention to some intuitive key areas. The overall shape have already derived by existing reconstruction methods [51, 53, 40, 46]. We mainly focus on the key areas. On the basis of our empirical visual habits, we prefer to define semantically significant areas as key areas, such as chair legs and table corners. In order to describe the shape of these key areas, we utilize key points, which are approximately the centers of the key areas. Taking key point as root, we derive subgraphs of predefined size around it to locate the key area.

To detect semantic key points, we use the hand-marked data provided by KeypointNet [57] to train a key-points detector. The network is based on Pointconv [54]. Although some key points are not accurate absolutely due to the insufficient generalization caused by limited data categories, this will not bring negative gains to subsequent modules for the loss calculating approach in sections 3.1.

Capture Structures by DeepWalk In order to learn the local geometry structure of mesh data, we decouple local structures from the overall graph. Taking the detected semantic key-point as root, we derive a sub-graph consisting of its nearest neighbors. The edge of the sub-graph is weighted by the descending function of distance between vertices at both ends. Then we explore the geometry of the sub-graph by deep walk [5, 55], which encodes the structure information.

Take the mesh \( M \) in Figure 3 as an example, we detect a set of \( n \) key-points \( K(n) = [k_1, k_2, ..., k_n] \). Then, taking \( k_i \) as root and fix the local scale as \( m \), we derive a set of \( n \) subgraphs \( G(m, n) = [g_1, g_2, ..., g_n] \), where \( g_i \) contains coordinates of vertexes and connecting edges of them \( g_i = \{ V = \{ v_1, v_2, ..., v_m \}, E \} \). Taking \( g_1 \) as an example, rooted by \( k_1 \), it is a sub-graph of the \( M \) and captures the local geometry structure in area1. As a collection of subgraphs \( g_i \), \( G \) contains all feature of areas where the semantic and structural information is rich.

Geometry Similarity Calculation The similarity of two or more meshes is widely applied in classification, detection, and reconstruction tasks. However, most similarity calculations of mesh are manipulating on sampled points, wasting the topology information. In order to further excavate mesh data, we introduce a geometry similarity calculation method.

Given two mesh \( M_1 \) and \( M_2 \), we firstly use the method in 3.1 to locate their key areas \( A = [a_1, a_2, ..., a_n] \) and \( B = [b_1, b_2, ..., b_n] \). Then according to section 3.1, two sets of geometry structures are captured in \( G_1 \) and \( G_2 \). For subgraphs of \( G_1 \) and \( G_2 \), we use \( g_1^1 \) and \( g_2^2 \) to represent them correspondingly. Then we use weighted graph kernel to calculate their similarity, represented by \( S \). First, we need to determine the corresponding points of the two subgraphs \( g_1^1 \) and \( g_2^2 \). Node2vec [20] is imported here to learn the embedding of each point. Specifically, Node2vec algorithm encodes coordinate and connecting relationship of each vertex into the vector \( e \). For \( g_j \) in \( G \), using Node2vec, we address its integral geometry structure vector

\[
g g_j = [e_1, e_2, ..., e_m]
\]

The \( e_n \) here is the embedding of the vertex \( m \). Credit to the process of solving the likelihood problem, vector \( e \) greatly retains geometric structure information.

Then, according to the distance between the feature vectors, we reorder the vertices of \( g_1^1 \) and \( g_2^2 \) into one-to-one
corresponding format. For each pair of vertices, euclidean distance is used to represent the difference between their features. Euclidean distance of each pair of nodes is combined to construct distance matrix D. Euclidean distance is calculated as follows:

\[ d_G(e, e') = \|e - e'\|_2, \]  

(2)

Subsequently, we utilize weighted graph kernel to calculate the similarity of \( g_1^i \) and \( g_2^i \):

\[ S(g_1^i, g_2^i) = e^{-\lambda D}, \lambda > 0. \]  

(3)

Finally, the similarity of the two mesh \( G1 \) and \( G2 \) can be derived by the mean of similarities of all subgraphs.

\[ S(G1, G2) = \frac{1}{m} \sum_{i=1}^{m} S(g_1^i, g_2^i). \]  

(4)

3.2. Geometry-aware Mesh Reconstruction

In this section, we apply geometry structure extracted by GSE in section 3.1 to 3D mesh reconstruction. Existed 3D mesh reconstruction methods mainly focus on the correspondence of the 2D image and sampled 3D points on the mesh and neglect geometry structure, which leads to the lack of details and even abnormal topology. Our deformation networks follow the architectures in Deep Mesh [40]. The features of images are extracted by resnet, and then put into CNNs to guide the movement of vertexes of template ellipsoid. When the error is too large, the attention on the local area will be weakened. In order to obtain feature in various levels, we utilize a multi-scale geometry extractor. The local geometry structure restrict can be described as follows:

\[ L_{gs}(s) = \sum_{k=1}^{s} \alpha_k L(m_k, n), \]  

(7)

where \( s \) is the number of scales and \( \alpha_k \) is the weight of different scales.

3.3. Fine-grained Mesh Edit

Deformation-based reconstruction method is the only approach to gain mesh from pixel-wise directly. However, it is restricted by the initial template, for the reason that only objects of the same genus can be transformed from each other through deformation, which is demonstrated in topology theory [2, 52]. The most universal template is genus 0 ball or ellipsoid for its better generalization, which is destined to be impossible to reconstruct non-zero genus object. Although adopting template by retrieval has been practiced [31, 49], it is still helpless to the change of genus. Inspired by deep mesh, to generate objects of various genus from a genus 0 template, we introduce a fine-grained mesh edit module that is illustrated in Figure 3. The error estimation network divides vertexes into different error levels, according to which we do super-resolution of a different degree. Then, we judge on fine faces and prune error one. Finally, a patching algorithm is implemented to repair details. Compared with deep mesh, our mesh edit module not only enables the face pruning to be performed in a fine-grained manner but also repairs the abnormal meshes with non-closed boundaries. Specially, if the target mesh is genus 0, the estimated errors of faces are generally low. Thus, the edit module will not affect the reconstruction of genus 0 object.

Error estimation and Local Super-resolution

The biggest gap between different genus meshes is the number of holes. Our main idea to span over different genus is to dig holes in proper area. To do this, a classification network is trained to estimate the error property of each face. However, the error is the mean error of sampled points, pruning whole faces directly will lead to excessive delete, coarse boundaries, and unnecessary error. Thus, different from the discriminating problem in deep mesh [40], we set up an error classification and hierarchically super-resolution mechanism. The upper part of the edit module in Figure 3 describes branches of the super resolution. When the error is in the interval of \( \tau_1 \) and \( \tau_2 \), we think the error property is lower and the error area is smaller correspondingly. Then we do super resolution two times, after which one face is divided into sixty faces, if the error is bigger than \( \tau_2 \), we do super resolution only once, after which one face is divided into four faces.

Prune and Repair

As one of the topology-changing approaches, pruning is the key to break through the limitation of the genus 0 initial template in reconstruction. After error
estimation and local super resolution, we will prune neighbor faces of vertices whose error property is higher than \( \tau \).

Pruning operation encourages holes to generate by forcing mesh to change its genus. However, mentioned in deepmesh [40], deleting faces leads to non-closed boundaries, which makes mesh abnormal and unsuitable to general rendering, rigging, and other subsequent applications. In order to solve the problem, we propose the following patching algorithm. By detecting non-closed boundaries and classifying them into two categories, we can fill holes to repair the mesh. The detailed proof and explanation of the algorithm are provided in the supplementary material.

**Algorithm 1** Patching Algorithm

**Input:** A mesh with non-closed boundaries  
**Output:** A closed mesh

Detect a set of circles \( C = \{C_1, C_2, ..., C_t\} \), where \( C_i \) is formed by non-closed boundary edges

for \( C_i \) in \( C \) do

if \( \text{dist}(C_i, C_j) < \gamma \) then

Connect corresponding points on \( C_i, C_j \) and triangulate new surfaces

end if

if \( \text{dist}(C_i, C_j) \geq \gamma \) for any \( j \) then

Connect turning points \( P_1, P_2, ..., P_n \) in turn and generate new circles \( S = \{S_1, S_2, ..., S_m\} \) for \( S_j \) in \( S \) do

Find center \( v_j \) of \( S_j \) and generate triangular surface by connecting \( v_j \) with every vertex of \( S_j \)

end for

\end{algorithm}

3.4. Training Objectives

Our network is restricted by losses of three parts. For the overall shape, we extend commonly used Charmfer Distance to support larger batchsize on multiple GPUs. In order to complete local details, we propose multi-scale setting Geometry Structure loss utilizing GSE in section 3.1. Besides, to avoid abnormal movements of vertexes and guarantee the high quality of reconstructed 3D geometry, we also apply a series of regularization losses.

**Charmer Distance** Charmfer Distance\((CD)\) is a normally used restrict in supervised 3D reconstruction [51, 53, 40]. It measures the similarity of two shapes by calculating the distance between two point sets, which is generally defined as:

\[
L_{\text{CD}} = \sum_{x \in M} \min_{y \in S} \| x - y \|^2 + \sum_{y \in S} \min_{x \in M} \| y - x \|^2, \quad (8)
\]

where \( x \in M \) and \( y \in S \) are respectively the point sets down-sampled from vertices of generated mesh \( M \) and the ground truth points set.

**Regularization Loss.** We employ three regularization techniques defined in [51, 27]. The normal loss \( L_{\text{normal}} \) measures the normal consistency between the generated mesh and ground truth. The smoothness loss \( L_{\text{smooth}} \) flattens the intersection angles of the triangle faces and supports the surface smoothness. And the edge loss \( L_{\text{edge}} \) penalizes the flying vertices and overlong edges to guarantee the high quality of recovered 3D geometry.

**Error estimation loss.** We adopt the error estimation loss in Deep Mesh [40] to train the error estimation network. For every face on the predicted mesh \( M \), we sampled a point set \( \{x \in M\} \) on it. \( f_e \) is the estimated error of the network and \( e_x \) is the corresponding ground truth error.

\[
L_{\text{error}} = \sum_{x \in M} |f_e(x) - e_x|^2, \quad (9)
\]

The final training objective of our system is defined as:

\[
L = \lambda_1 L_{\text{gs}} + \lambda_2 L_{\text{CD}} + \lambda_3 L_{\text{normal}} + \lambda_4 L_{\text{smooth}} + \lambda_5 L_{\text{error}}, \quad (10)
\]

4. Experiments

4.1. Experimental setup

**Dataset** In the GSE module, we use the dataset provided by You et al. [57] to train a key-points discovery network. The dataset contains hand-marked 3D keypoint dataset on 16 categories of the ShapeNet [7], which is a collection of 3D CAD models. In 3D mesh reconstruction, we use five categories of the ShapeNet [7] dataset to train the deformation network. In terms of category selection, we focus on common objects in indoor scenes, which are more valuable for daily applications. We use the rendering images provided by Choy et al. [11] as input. For fair comparison, we use the same training/testing split as in Choy et al. [11].

**Evaluation Metric** We use standard evaluation metrics for 3D shape generation. Following Fan et al. [13], we calculate Chamfer Distance\((CD)\) between points clouds uniformly sampled from the ground truth and our prediction to measure the surface accuracy. We also use F1-score following Wang et al. [51] to measure the completeness and precision of generated shapes. Both metrics are computed between the ground truth point cloud and 10,000 points uniformly sampled from the generated mesh. For CD, the smaller is better. For F-score, the larger is better.

**Implementation Detail** The key-points discovery network in GSE module is a classifier based on pointconv [54], which is trained firstly for only 4 hours on NVIDIA 2080Ti. The reconstruction module including deformation module and edit module is implemented in Pytorch and every submodule is trained separately. We use a batch size of 32 and Adam [28] optimizer at a learning rate of \( 1e^{-3} \) (dropped to \( 1e^{-4} \) after 200 epochs) for 400 epochs. The whole model is trained on NVIDIA 2080Ti for 72 hours. The values of hyper-parameters mentioned above are \( \tau_1 = 0.001, \tau_2 = 0.01, \gamma = 0.1, \lambda_1 = 0.1, \lambda_2 = 1, \lambda_3 = 1e^{-3}, \lambda_4 = 5e^{-7}, \lambda_5 = 0.1 \).

4.2. Results and Comparisons

We first Quantitatively compare the performance of our approach with three state-of-the-art methods, including
Table 1: CD and F1 on the ShapeNet test set. For CD, the smaller is better. For F-score, the larger is better.

<table>
<thead>
<tr>
<th>Category</th>
<th>CD</th>
<th>F1</th>
<th>CD</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>P2M</td>
<td>0.610</td>
<td>0.514</td>
<td>0.389</td>
<td>54.38</td>
</tr>
<tr>
<td>GEOMetrics</td>
<td>0.498</td>
<td>0.404</td>
<td>0.316</td>
<td>66.30</td>
</tr>
<tr>
<td>DeepMesh</td>
<td>0.755</td>
<td>0.629</td>
<td>0.798</td>
<td>51.39</td>
</tr>
<tr>
<td>Ours</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chair</td>
<td>0.514</td>
<td>0.7432</td>
<td>0.798</td>
<td>71.20</td>
</tr>
<tr>
<td>table</td>
<td>0.498</td>
<td>0.6630</td>
<td>0.7432</td>
<td></td>
</tr>
<tr>
<td>monitor</td>
<td>0.624</td>
<td>0.7211</td>
<td>0.6630</td>
<td></td>
</tr>
<tr>
<td>lamp</td>
<td>0.755</td>
<td>0.7482</td>
<td>0.7211</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>0.7564</td>
<td>0.6212</td>
<td>0.4894</td>
<td>71.202</td>
</tr>
</tbody>
</table>

(a) Image (f) Ground truth (b) Pixel2Mesh (c) GEO (d) DeepMesh (e) Ours

Figure 4: Qualitative results. (a) Input image; (b) Pixel2Mesh [51]; (c) GEOMetrics [46]; (d) Deep Mesh [40], the green lines and triangles mark the non-closed boundaries correspondingly; (e) Ours; (f) Ground truth.

Pixel2Mesh [51], GEOMetrics [46] and Deep Mesh [40]. We adopt the widely used Chamfer Distance (CD) and F1 score [29] to quantitatively evaluate the reconstruction accuracy. Results are shown in Tab 1, where CD and F1 are in the unit of $10^{-2}$. Our approach outperforms the other methods in most categories and achieves the best mean score. The most obvious category is chair, which is improved by 24% than the best method before, followed by the table with 22% improved. Although the lamp has the least improvement, probably because of that the large variance within the class weakens shape constraints and the edit module, our method still achieves improvement relative to others. GEOMetrics performs well in F1 score while its CD is the worst. This is probably caused by its sparse vertices strategy. In order to analyze comprehensively, we also compare the quantitative results.

The visual reconstruction results are shown in Fig 4. While Pixel2Mesh can reconstruct the rough shapes, it fails to capture the fine details of the geometry, such as the slim chair legs, square table corner, and smooth curved surfaces, which is due to the lack of partial shape constraints. Thanks to the local face slitting operation, GEOMetrics reconstructs relatively refined geometry. But overlap and intersection make glitches appear in the results. Besides, due to the constrain of the initial genus-0 ball template, both Pixel2Mesh and GEOMetrics can not generate holes under the chair handle or desks, which seriously affects the accuracy of reconstruction for that objects of non-zero genus are widespread in nature. In contrast, our baseline Deep Mesh utilizes topology modification module to break the constrain of the initial module. The results show holes in corresponding parts. However, pruning brings two problems. The one is the non-closed boundaries and faces, which is marked with green line and light green triangles correspondingly in the results of Deep Mesh. The other is the large scale of the prune which brings imprecise cropped contour. In comparison, we
boast highly accurate reconstructions both in global shape and local details. We are able to generate meshes with complex geometry structure, shown in high-fidelity areas such as corners and edges. In addition, our method is able to reconstruct holes with fine-grained boundaries with closed stereoscopic repaired faces, which is especially shown on the right side of Fig 4. We also validate the advance of our method on real dataset Pix3D [47], and compare with Total3DUnderstanding [39]. The results are shown in the supplementary material.

4.3. Ablation Study

Components Analyses Now we conduct controlled experiments to analyze the effectiveness of the proposed geometry structure extractor and fine-grained edit module in our framework. Tab 2 reports the performance of each module by removing one component from the full model.

We first remove the Geometry Structure Extractor. It is observed that for every category, there is still more than 10% improvement on CD compared with the baseline, which is also the best model of the three competitors. This reflects that the fine-grained edit module performs obviously better than the simple topology modification in Deep Mesh. Then we remove the edit module. The results show that monitor and lamp still has a 14.3% and 11.6% improvement correspondingly than the baseline, which validates the effectiveness of the GSE. For chair and table, the results are slightly worse than the baseline, which is because that the larger proportion of the holes in the two categories leads to larger sensitivity to the edit module.

In order to prove the wide applicability of the two modules, we also experiment on the gradually refined reconstruction framework of Pixel2Mesh. The average results of five categories are shown in Tab 3. In these results, GSE achieves great improvement while the editing module is relatively low. This is probably because that the graph convolutional network of Pixel2Mesh does not support the deformation after prune, which hinders the stretching of the border after pruning.

The size of subgraphs In order to decide the suitable settings of GSE, we first investigate the effect of the size of local geometry areas, where Chamfer distance (CD) is used as the measure of the GT mesh and reconstructed mesh. In order to explore how geometry extractor is affected by relevant factors, The architecture without mesh edit module is adopted. Figure 5 plots the result, suggesting that best performance can be achieved when the number of points is 8 in every sub-graph. The result also hints that too small or large sub-graph cannot describe local geometry properly while computing expense is increasing continuously.

<table>
<thead>
<tr>
<th>Category</th>
<th>CD (full model)/(without GSE)/(without Edit)/(full model)/(without GSE)/(without Edit)</th>
<th>F1 (full model)/(without GSE)/(without Edit)/(full model)/(without GSE)/(without Edit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>chair</td>
<td>0.389/0.424/0.573/0.743/0.532/0.589</td>
<td>70.54/75.22/74.67/73.56/71.76/71.34</td>
</tr>
<tr>
<td>table</td>
<td>0.316/0.351/0.432/0.763/0.542/0.560</td>
<td>70.82/74.13/73.64/73.64/73.64/73.64</td>
</tr>
<tr>
<td>bench</td>
<td>0.427/0.433/0.498/0.743/0.539/0.559</td>
<td>71.20/59.89/70.35/70.35/70.35/70.35</td>
</tr>
<tr>
<td>monitor</td>
<td>0.517/0.560/0.539/0.712/0.592/0.578</td>
<td>58.60/58.16/57.08/57.08/57.08/57.08</td>
</tr>
<tr>
<td>lamp</td>
<td>0.798/0.871/0.922/0.593/0.570/0.555</td>
<td>59.13/58.16/57.08/57.08/57.08/57.08</td>
</tr>
<tr>
<td>Mean</td>
<td>0.4894/0.5278/0.5928/0.71202/0.68014/0.66168</td>
<td>71.202/68.014/66.168/66.168/66.168/66.168</td>
</tr>
</tbody>
</table>

Figure 5: Effect of the size of subgraphs and the number of points

The number of key points Under same settings of the first investigation, we explore the effect of the number of key points. The number of key points is changed by randomly increasing non-semantic points or deleting semantic key points. According to the trend in Figure 5, the results will be improved as the number of points increases. But considering the heavily increased calculation and limited computing resources we choose 10 key-points for every model.

The ablation studies on thresholds of the super-resolution and pruning are provided in the supplementary material.

5. Conclusion

In this paper, we presented a geometric granularity-aware pixel-to-mesh framework. The framework contains a Geometry Structure Extractor, which can select high-fidelity areas and capture the local shape information tendentially. The GSE facilitates pixel-to-mesh mapping and resolves coarse details problem caused by the neglect of structural information in previous practices. Furthermore, we address the restrictive constant topology prescribed by the initial mesh object through fine-grained mesh edit, which encourages non-zero genus topology to emergence and repair abnormal mesh. Extensive experimental results have demonstrated that our framework achieves high reconstruction quality. Future research directions include mining genus information in the picture to reconstruct tiny holes on objects and fusing multi-view 2D information to facilitate 3D reconstruction.

6. Acknowledgements

This work was supported by National Science Foundation of China (U20B2072, 61976137).
References


