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.

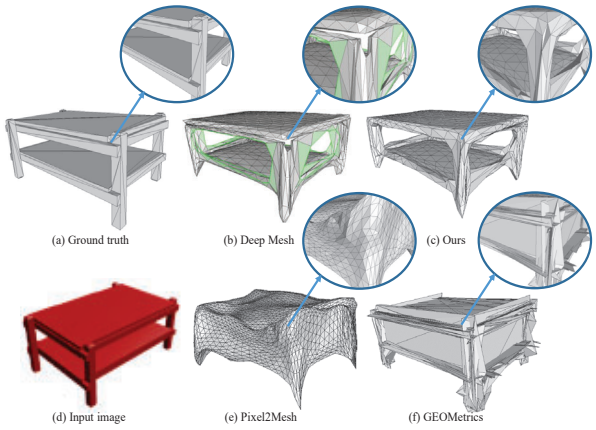


Figure 1: Monocular reconstruction results of the state of the art deformation-based learning approaches. (a) Ground truth; (b) Deep Mesh [40], the green lines and triangles mark the non-closed boundaries correspondingly; (c) Ours; (d) Input image; (e) Pixel2Mesh [51]; (f) GEOMetrics [46]. Relatively, our method can better deal with the topology in details and reconstruct holes of the object without non-closed boundaries.

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-

†Corresponding author

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, but the refinement of the pruned area and ensuing open boundary issues are also shown in related attempts [40]. Although other methods, such as point/voxel and SDF reconstruction [14, 11, 41] seem to avoid this problem, they may suffer from missing parts and holes when transformed to mesh using Marching Cubes algorithms [4]. Therefore

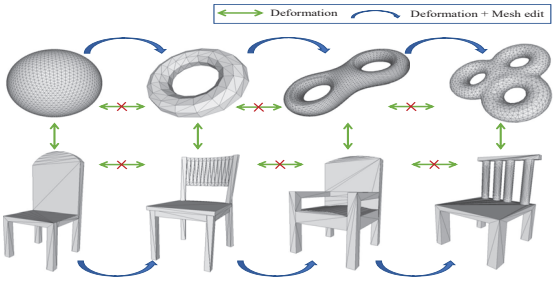


Figure 2: Translation relationship between different genus topologies.

in this paper, we aim to address the above mentioned limitations by introducing geometric granularity-aware pixel-to-mesh framework with a fidelity-selection-and-guarantee strategy. The model selects high-fidelity parts and realizes partially re-sampling, which are used to guide subsequent deformation and in turn guarantee the fidelity of reconstruction results. The framework consists of a Multi-scale Geometry Structure Extractor (GSE), a geometry-aware deformation network, and a fine-grained mesh edit module. Firstly, in order to capture important local geometry features, we propose the multi-scale geometry extractor. It locates visually perceptible high-fidelity areas by detecting semantic key points around with rich details and complex structure. Then it captures the geometry structures of the key areas by forming subgraphs rooted by the key points. Leveraging the subgraphs sets of given meshes, geometry extractor measures local structural similarity of the meshes using weighted graph kernel [50]. Then, to improve the reconstruction details of existing methods, we introduce the Geometry Structure Extractor into monocular 3D mesh reconstruction. Note that the GSE is general and can be extended to other tasks related to mesh data, such as classification and retrieval. In the deformation stage, multi-scale local geometry features captured by the extractor restrict and guide movements of vertexes on the template mesh. After deformation, to break through the restrict of the genus 0 initial template on reconstructed objects, we introduce a Fine-grained Edit Module. In order to realize fine-grained edit, faces are subdivided adaptively according to the estimated error degrees. Then after the second error estimation, pruning is operated. Finally, the edit module refines the boundaries and provides a Patching Algorithm to repair the non-closed boundaries. 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 retrieval, 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] roundbreakingly 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 Laplician 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

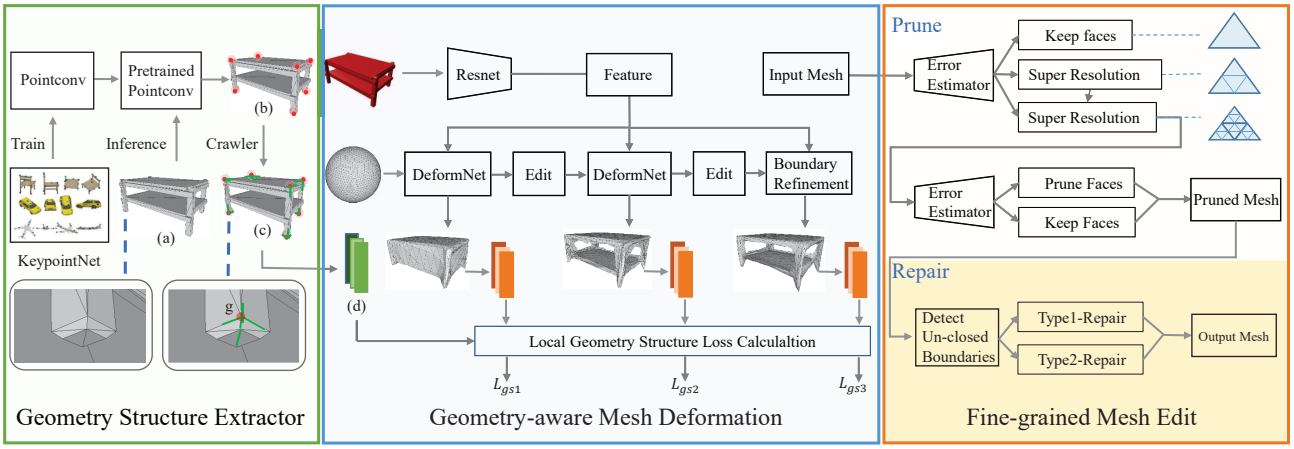


Figure 3: The overview of our framework. When a ground truth mesh (a) is input to the pretrained Geometry Structure extractor, its key areas are located, which is marked by red areas in (b). Then, subgraphs describing local shapes are captured in (c) and then translated into vectors (d), which provide geometry constraints for mesh deformation. The Deformation Module progressively moves the mesh vertices, and the Edit Module hierarchically modifies the topology to approximate the target object model. After two rounds of deformation and edit, boundaries are refined and the non-closed mesh is repaired.

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 $M1$ and $M2$, we firstly use the method in 3.1 to locate their key areas $A = [a_1, a_2, \cdots, a_n]$ and $B = [b_1, b_2, \cdots, b_n]$. 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_i^1 and g_i^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_i^1 and g_i^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

$$gs_j = [e_1, e_2, ..., e_m]. \tag{1}$$

The e_m 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_i^1 and g_i^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:

$$d_E(e, e') = \|e - e'\|_2 . \tag{2}$$

Subsequently, we utilize weighted graph kernel to calculate the similarity of g_i^1 and g_i^2.

$$S(g_i^1, g_i^2) = e^{-\lambda D}, \lambda > 0. \tag{3}$$

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

$$S(G1, G2) = \frac{1}{m} \sum_{i=1}^{m} S(g_i^1, g_i^2). \tag{4}$$

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. In addition to the existed constraints which have been widely in mesh reconstruction [53, 40], GSE preferentially re-samples local shape features which enable the network to learn more geometry structure of mesh data.

First, we can detect n key points by GSE on the ground truth mesh. For each of them, GSE finds the nearest vertices on the predicted mesh and saves them as n key points of the predicted mesh. Then, GSE extract a graph whose size is predefined m around every key point. All m vertexes of graphs on the ground truth mesh form a set $G_{gt}(m, n)$ while the one on the predicted mesh is $G_{pred}(m, n)$. The geometry structure of the ground truth mesh can be represented as $GS_{gt}(m, n) = [gs_1, gs_2, ..., gs_n]^T$, while the geometry structure of the predicted mesh is $GS_{pred}(m, n) = [gs_1^{'}, gs_2^{'}, ..., gs_n^{'}]^T$. Hence, we have v_i and $v_i^{'}$ in gs_i and $gs_i^{'}$. In this geometry vector space, use c_i to represent the correspondence vertex of v_i.

$$c_i = \left\{ v_i^{'} | i = argmin L(e_i, e_i^{'}), i = 1, 2, ..., n \right\}, \tag{5}$$

where $\mathcal{L}$ is a distance function. According to the equation 5, we can reorder vertices in the matrix $G_{pred}(m, n)$ and save it as $G_{pred-ordered}(m, n)$. Finally, the similarity of local geometry structures on predicted mesh and the ground truth can be measured.

$$\mathcal{L}(m, n) = \mathcal{L}\left\{ G_{gt}(m, n), G_{pred-ordered}(m, n) \right\}, \tag{6}$$

The n here is determined by the number of detected key points and the m is the scope of the local area. If m is too small, we cannot capture complete structures. But if it 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:

$$\mathcal{L}_{gs}(s) = \sum_{k=1}^{s} \alpha_k L(m_k, n), \tag{7}$$

where s is the number of scales and α_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 rejudge 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 τ_1 and τ_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 τ_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 τ.

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 $C = \{C_1, C_2, ..., C_l\}$, where C_i is formed by non-closed boundary edges
for C_i in C **do**
  if $dist(C_i, C_j) < \gamma$ **then**
    Connect corresponding points on C_i, C_j and triangulate new surfaces
  end if
  if $dist(C_i, C_j) \geqslant \gamma$ 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

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.

Charmfer 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:

$$\mathcal{L}_{CD} = \sum_{x \in M} \min_{y \in S} \|x - y\|_2^2 + \sum_{y \in S} \min_{x \in M} \|y - x\|_2^2, \quad (8)$$

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 $\mathcal{L}_{normal}$ measures the normal consistency between the generated mesh and ground truth. The smoothness loss $\mathcal{L}_{smooth}$ flattens the intersection angles of the triangle faces and supports the surface smoothness. And the edge loss Ledge 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.

$$\mathcal{L}_{error} = \sum_{x \in M} |f_e(x) - e_x|^2, \quad (9)$$

The final training objective of our system is defined as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{gs} + \lambda_2 \mathcal{L}_{CD} + \lambda_3 \mathcal{L}_{normal} + \lambda_4 \mathcal{L}_{smooth} + \lambda_5 \mathcal{L}_{error}. \quad (10)$$

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.

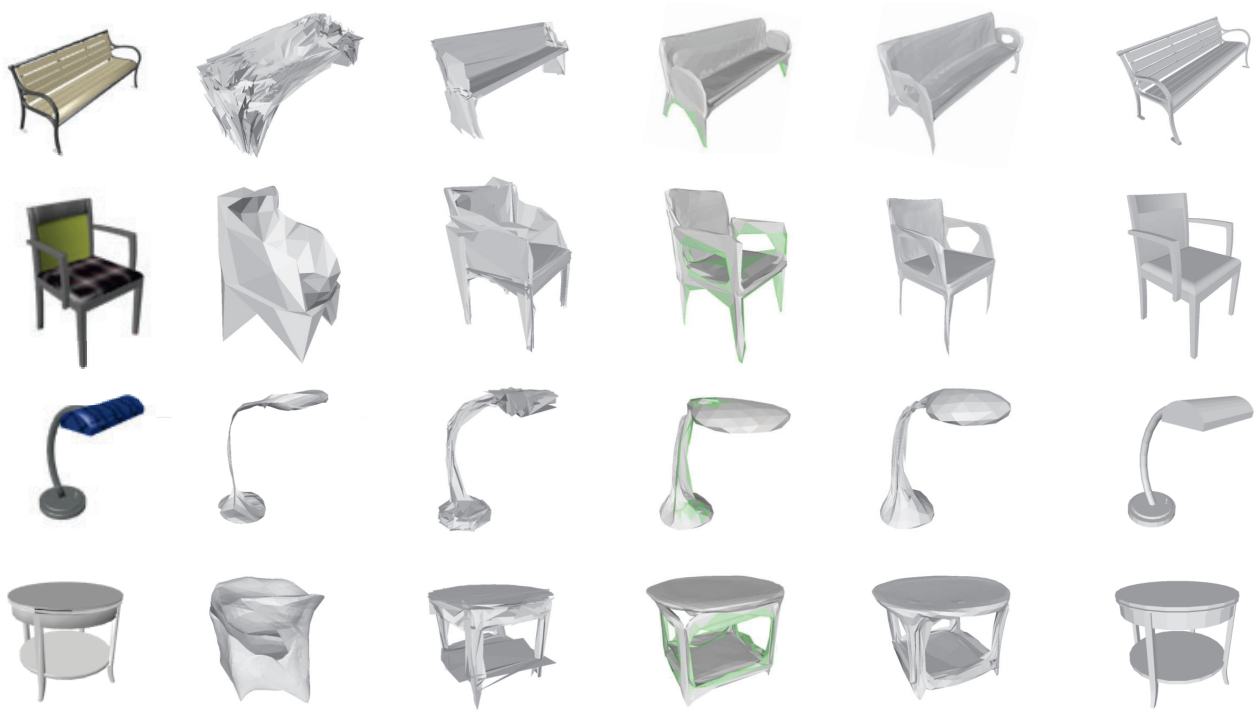
Implementation Detail The key-points discovery network in GSE module is a classifier based on pointconv [54], which is trained firstly for only 4 hours on NVIDIA 2080Ti. The reconstruction network including deformation module and edit module is implemented in Pytorch and every submodule is trained separately. 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, \tau = 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.

Category	CD				F1			
	P2M	GEOMetrics	DeepMesh	Ours	P2M	GEOMetrics	DeepMesh	Ours
chair	0.610	0.823	0.514	**0.389**	54.38	56.61	59.19	**74.32**
table	0.498	0.797	0.404	**0.316**	66.30	66.33	73.42	**76.54**
bench	0.624	0.690	0.516	**0.427**	57.57	72.11	71.55	**74.82**
monitor	0.755	0.793	0.629	**0.517**	51.39	59.50	57.64	**71.20**
lamp	1.295	0.813	1.043	**0.798**	48.15	58.65	56.75	**59.13**
Mean	0.7564	0.7832	0.6212	**0.4894**	55.558	62.64	63.71	**71.202**



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

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 $1e-2$. 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

Table 2: Ablation study that evaluates the contribution of GSE and Fine-grained Edit module to the performance of the framework.

Category	CD			F1		
	Ours (full model)	Ours (without GSE)	Ours (without Edit)	Ours (full model)	Ours (without GSE)	Ours (without Edit)
chair	**0.389**	0.424	0.573	**74.32**	72.67	58.01
table	**0.316**	0.351	0.432	**76.54**	75.22	71.76
bench	**0.427**	0.433	0.498	**74.82**	74.13	73.64
monitor	**0.517**	0.560	0.539	**71.20**	59.89	70.35
lamp	**0.798**	0.871	0.922	**59.13**	58.16	57.08
Mean	**0.4894**	0.5278	0.5928	**71.202**	68.014	66.168

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. But 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 3: Ablation study that evaluates the effectiveness of GSE and the edit module on Pixel2Mesh.

	P2M	P2M+GSE	P2M+Edit
CD	0.756	0.659	0.603
F1	55.56	57.94	59.87

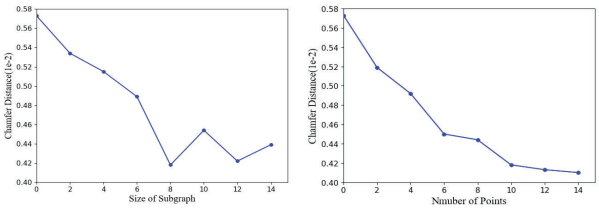


Figure 5: Effect of the size of subgraphs and the number of points

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 tendentiously. 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

[1] Oladapo Afolabi, Allen Y Yang, and S Shankar Sastry. Extending deepsdf for automatic 3d shape retrieval and similarity transform estimation. *arXiv e-prints*, pages arXiv–2004, 2020.

[2] Olga Anosova and Vitaliy Kurlin. Introduction to periodic geometry and topology, 2021.

[3] Marco Attene, Marcel Campen, and Leif Kobbelt. Polygon mesh repairing: An application perspective. *ACM Computing Surveys (CSUR)*, 45(2):1–33, 2013.

[4] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.

[5] Aleksandar Bojchevski, Oleksandr Shchur, Daniel Zügner, and Stephan Günnemann. Netgan: Generating graphs via random walks. In *International Conference on Machine Learning*, pages 610–619. PMLR, 2018.

[6] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, Jul 2017.

[7] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[8] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. Bsp-net: Generating compact meshes via binary space partitioning, 2020.

[9] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. Bsp-net: Generating compact meshes via binary space partitioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 45–54, 2020.

[10] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019.

[11] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

[12] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering, 2017.

[13] H. Fan, S. Hao, and L. Guibas. A point set generation network for 3d object reconstruction from a single image. *IEEE*, 2017.

[14] Haoqiang Fan, Hao Su, and Leonidas 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*, pages 605–613, 2017.

[15] Kui Fu, Jiansheng Peng, Qiwen He, and Hanxiao Zhang. Single image 3d object reconstruction based on deep learning: A review. *Multimedia Tools and Applications*, 80(1):463–498, 2021.

[16] David Fuentes-Jimenez, David Casillas-Perez, Daniel Pizarro, Toby Collins, and Adrien Bartoli. Deep shape-from-template: Wide-baseline, dense and fast registration and deformable reconstruction from a single image. *arXiv preprint arXiv:1811.07791*, 2018.

[17] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016.

[18] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9785–9795, 2019.

[19] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018.

[20] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks, 2016.

[21] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[22] Yiwei Jin, Diqiong Jiang, and Ming Cai. 3d reconstruction using deep learning: a survey. *Communications in Information and Systems*, 20(4):389–413, 2020.

[23] Tao Ju. Fixing geometric errors on polygonal models: a survey. *Journal of Computer Science and Technology*, 24(1):19–29, 2009.

[24] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018.

[25] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. *arXiv preprint arXiv:1708.05375*, 2017.

[26] Kestutis Karčiauskas and Jörg Peters. A new class of guided c2 subdivision surfaces combining good shape with nested refinement. In *Computer Graphics Forum*, volume 37, pages 84–95. Wiley Online Library, 2018.

[27] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. 2017.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[29] A. Knapitsch, J. Park, Q. Y. Zhou, and V. Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *Acm Transactions on Graphics*, 36(4):78, 2017.

[30] Nils M Kriege, Fredrik D Johansson, and Christopher Morris. A survey on graph kernels. *Applied Network Science*, 5(1):1–42, 2020.

[31] Andrey Kurenkov, Jingwei Ji, Animesh Garg, Viraj Mehta, JunYoung Gwak, Christopher Choy, and Silvio Savarese. Deformnet: Free-form deformation network for 3d shape reconstruction from a single image. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 858–866. IEEE, 2018.

[32] Y. Li, S. Hao, X. Guo, and L. Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[33] Hsueh-Ti Derek Liu, Vladimir G Kim, Siddhartha Chaudhuri, Noam Aigerman, and Alec Jacobson. Neural subdivision. *arXiv preprint arXiv:2005.01819*, 2020.

[34] William E. Lorensen and Harvey E. Cline. *Marching Cubes: A High Resolution 3D Surface Construction Algorithm*, page 347–353. Association for Computing Machinery, New York, NY, USA, 1998.

[35] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 832–840, 2015.

[36] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[37] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020.

[38] Marion Neumann, Plinio Moreno, Laura Antanas, Roman Garnett, and Kristian Kersting. Graph kernels for object category prediction in task-dependent robot grasping. In *Online Proceedings of the Eleventh Workshop on Mining and Learning with Graphs*, pages 0–6, 2013.

[39] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, Jian Chang, and Jian Jun Zhang. Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 55–64, 2020.

[40] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep mesh reconstruction from single rgb images via topology modification networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9964–9973, 2019.

[41] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.

[42] Jhony K Pontes, Chen Kong, Sridha Sridharan, Simon Lucey, Anders Eriksson, and Clinton Fookes. Image2mesh: A learning framework for single image 3d reconstruction. In *Asian Conference on Computer Vision*, pages 365–381. Springer, 2018.

[43] Reinhold Preiner, Tamy Boubekeur, and Michael Wimmer. Gaussian-product subdivision surfaces. *ACM Transactions on Graphics (TOG)*, 38(4):1–11, 2019.

[44] E Pérez, S. Salamanca, P Merchán, and A Adán. A comparison of hole-filling methods in 3d. *International Journal of Applied Mathematics and Computer Science*, 26(4), 2016.

[45] Taiping Qu, Yangming Yue, Qirui Zhang, Cheng Wang, Zhiqiang Zhang, Guangming Lu, Wei Du, and Xiuli Li. Baenet: A brain age estimation network with 3d skipping and outlier constraint loss. In *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pages 399–403. IEEE, 2020.

[46] Edward J Smith, Scott Fujimoto, Adriana Romero, and David Meger. Geometrics: Exploiting geometric structure for graph-encoded objects. *arXiv preprint arXiv:1901.11461*, 2019.

[47] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018.

[48] Jiapeng Tang, Xiaoguang Han, Junyi Pan, Kui Jia, and Xin Tong. A skeleton-bridged deep learning approach for generating meshes of complex topologies from single rgb images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4541–4550, 2019.

[49] Mikaela Angelina Uy, Jingwei Huang, Minhyuk Sung, Tolga Birdal, and Leonidas Guibas. Deformation-aware 3d model embedding and retrieval. In *European Conference on Computer Vision*, pages 397–413. Springer, 2020.

[50] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

[51] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018.

[52] P. Wang, L. Lu, and K. Bertoldi. Topological phononic crystals with one-way elastic edge waves. *Physical Review Letters*, 115(10):104302, 2015.

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

[54] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019.

[55] Yun-Peng Xiao, Yu-Kun Lai, Fang-Lue Zhang, Chunpeng Li, and Lin Gao. A survey on deep geometry learning: From a representation perspective. *Computational Visual Media*, 6(2):113–133, 2020.

[56] Wang Yifan, Noam Aigerman, Vladimir G Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. Neural cages for detail-preserving 3d deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 75–83, 2020.

[57] Yang You, Yujing Lou, Chengkun Li, Zhoujun Cheng, Liangwei Li, Lizhuang Ma, Cewu Lu, and Weiming Wang. Keypointnet: A large-scale 3d keypoint dataset aggregated from numerous human annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13647–13656, 2020.

[58] Zerong Zheng, Tao Yu, Qionghai Dai, and Yebin Liu. Deep implicit templates for 3d shape representation, 2020.

[59] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 850–859, 2019.

[60] Michael Zollhöfer, Patrick Stotko, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb. State of the art on 3d reconstruction with rgb-d cameras. In *Computer graphics forum*, volume 37, pages 625–652. Wiley Online Library, 2018.