GEOMetrics: Exploiting Geometric Structure for Graph-Encoded Objects

Edward J. Smith 1 Scott Fujimoto 1 2 Adriana Romero 3 1 David Meger 1

Abstract
Mesh models are a promising approach for encoding the structure of 3D objects. Current mesh reconstruction systems predict uniformly distributed vertex locations of a predetermined graph through a series of graph convolutions, leading to compromises with respect to performance or resolution. In this paper, we argue that the graph representation of geometric objects allows for additional structure, which should be leveraged for enhanced reconstruction. Thus, we propose a system which properly benefits from the advantages of the geometric structure of graph-encoded objects by introducing (1) a graph convolutional update preserving vertex information; (2) an adaptive splitting heuristic allowing detail to emerge; and (3) a training objective operating both on the local surfaces defined by vertices as well as the global structure defined by the mesh. Our proposed method is evaluated on the task of 3D object reconstruction from images with the ShapeNet dataset, where we demonstrate state of the art performance, both visually and numerically, while having far smaller space requirements by generating adaptive meshes.

1. Introduction
Surfaces in our physical world exhibit highly non-uniform curvature; compare a plane’s wing to a microscopic screw that fastens its engine. Traditionally, deep 3D understanding systems have relied upon representations such as voxels and point clouds, which capture structure with either uniform volumetric or surface detail (Choy et al., 2016; Wu et al., 2016; Fan et al., 2017). By representing unimportant, or uninteresting, regions with high detail, these systems scale poorly to higher resolutions and object complexity (see Figure 1a and 1b). While efforts have been made to rectify this issue through intermediary representations (Tatarchenko et al., 2017; Häne et al., 2017; Smith et al., 2018; Zhang et al., 2018), these either continue to rely on sparse volumetric units, or maintain uniform detail through alternative representations.

A triangle mesh is a graph-based shape representation that encodes 3D structure through a set of vertices and corresponding planar faces. Recently, advances in deep learning on graphs have enabled mesh-based 3D shape methods (Kato et al., 2017; Wang et al., 2018; Kanazawa et al., 2018; Jack et al., 2018; Henderson & Ferrari, 2018; Groueix et al., 2018b). However, these approaches produce mesh predictions which uniformly space vertices and faces over their surface, providing no significant improvement over the previously highlighted representations (see Figure 1c) and hence, not exploiting the mesh representation advantages. In particular, by placing many vertices in regions of fine detail while using large triangles to summarize nearly planar regions, one could define adaptive meshes (see Figure 1d), enabling flexible scaling by effectively localizing complexity in object surfaces, and allowing for the 3D structure

Figure 1. Comparison of 3D shape encoding techniques, including their respective encoding sizes for the level of quality viewed.

\(^1\)Department of Computer Science, McGill University, Montreal, Canada \(^2\)Mila Québec AI Institute \(^3\)Facebook AI Research. Correspondence to: Edward Smith <edward.smith@mail.mcgill.ca>.

of complicated shapes to be encoded with smaller space requirements and higher precision.

Moreover, these deep learning mesh-based approaches rely on Graph Convolutional Networks (GCNs) (Kipf & Welling, 2016; Defferrard et al., 2016; Cucurull et al., 2018; Wang et al., 2018). Although effective in many node/graph classification and regression tasks, we argue that GCNs may be inadequate for understanding, reconstructing or generating 3D structure as they may induce over-smoothing while aggregating neighboring information at vertex level (Li et al., 2018). This aggregation bias could in turn lead to a harder learning problem when vital information held at each vertex cannot be derived from its neighbors, and as a direct consequence must not be lost.

Last, an important question when reconstructing 3D objects is how to define a loss between a prediction and its target. A common approach is to employ the Chamfer Distance over some parametrization of the two surfaces (Barrow et al., 1977; Insafutdinov & Dosovitskiy, 2018; Wang et al., 2018; Groueix et al., 2018a; Sun et al., 2018; Fan et al., 2017). However, this loss penalizes the point positions exclusively, and thus, its direct application to mesh vertex positions leads poor accuracy, as no information of the faces they define is provided and the placement of vertices over a surface is, to a large degree, arbitrary. In addition, this local loss function takes no consideration of the global structure of the predicted object, preventing class-specific attributes to emerge and creating global inconsistencies.

Therefore in this paper, we aim to address the above-mentioned limitations by introducing an adaptive mesh reconstruction system, called Geometrically Exploited Object Metrics (GEOMetrics), which properly capitalizes on the advantages and geometric structure of graph-encoded objects. GEOMetrics reformulates graph convolutional layers to prevent vertex smoothing. Moreover, it incorporates an adaptive face splitting heuristic allowing non-uniform detail to emerge. Finally, it introduces a training objective operating both on the local surfaces defined by vertices, via a differentiable sampling procedure, as well as the global structure defined by the graph, through a perceptual loss reminiscent of that of style transfer applications (Gatys et al., 2016; Johnson et al., 2016). To the best of our knowledge, our system is the first deep approach to describing shape as an adaptive mesh, through advances in geometrically-aware graph operations. We extensively evaluate our system on the task of 3D object reconstruction from single RGB images and show that the interplay of our introduced components encourages mesh reconstructions, which properly localize detail, while maintaining structural consistency. As a result, we are able to obtain mesh predictions which outperform previous methods and have far smaller space requirements.

The contributions of this paper can be summarized as:

- We introduce the Zero-Neighbor GCN (0N-GCN), an extension of Kipf & Welling (2016), which allows the information at each vertex to be maintained, and as a result better suits the understanding and reconstruction of 3D meshes.
- We present an adaptive face splitting procedure to encourage local complexity to emerge when reconstructing meshes, taking advantage of the mesh flexible scaling (see Figure 1d).
- We propose a training objective, which operates locally and globally over the surface to produce mesh reconstructions, which are highly accurate and benefit from the graceful scaling of mesh representations.
- We highlight through extensive evaluation the substantial benefits provided by the previous contributions and show, on the task of 3D object reconstruction from single RGB images, that by properly exploiting the meshes' properties and geometry, our GEOMetrics system is able to notably outperform prior methods visually and quantitatively, while requiring far less vertices/faces.

Note that the above-mentioned contributions are not specific to the reconstruction system nor the chosen task and thus, can be easily adapted to arbitrary mesh problems. Code for our system is publicly available on a GitHub repository, to ensure reproducible experimental comparison.¹

## 2. Related Work

### 3D Mesh Reconstruction.

Mesh models have only recently been used in generation and reconstruction tasks due to the challenging nature of their complex definition (Wang et al., 2018). Recent mesh approaches rely on graph representations of meshes, and use GCNs (Kipf & Welling, 2016) to effectively process them. Our work most closely relates to Neural 3D Mesh Renderer (Kato et al., 2017) and Pixel2Mesh (Wang et al., 2018), which use deformations of a generic pre-defined input mesh, generally a sphere, to form 3D structures. Similarly, Atlas-Net (Groueix et al., 2018a) uses deformations over a set of primitive square faces to form 3D shapes. Conceptually similar, there exists numerous papers using class-specific input meshes which are deformed with respect to the given input image (Pontes et al., 2017; Kanazawa et al., 2018; Jack et al., 2018; Henderson & Ferrari, 2018; Groueix et al., 2018b; Kar et al., 2015). While effective, these approaches require prior knowledge on the target class or access to a model repository.

### Graph Convolutional Networks.

The great success of convolutional neural networks in numerous image-based tasks (He et al., 2016; 2017; Huang et al., 2017; Jégou et al., 2017; Casanova et al., 2018) has led to increasing efforts to extend deep networks to domains where graph-structured

¹https://github.com/EdwardSmith1884/GEOMetrics
data is ubiquitous. Early attempts to extend neural networks to deal with arbitrarily structured graphs relied on recursive neural networks (Frasconi et al., 1998; Gori et al., 2005; Scarselli et al., 2009). Recently, spectral approaches have emerged as an effective alternative which formulates the convolution as an operation on the spectrum of the graph (Henaff et al., 2015; Bruna et al., 2014; Bronstein et al., 2017; Levie et al., 2017). Methods operating directly on the graph domain have also been presented. Defferrard et al. (2016) proposed to approximate the filters using the Chebyshev polynomials applied on the Laplacian operator. This approximation was further simplified by Kipf & Welling (2016). Finally, several works have been introduced exploring well-established deep learning ideas and improving previously reported results (Duvenaud et al., 2015; Hamilton et al., 2017; Monti et al., 2017; Veličković et al., 2018).

3D Object Representation. Deep learning approaches for understanding 3D shapes have, for a long time, employed voxels as a default 3D object representation (Choy et al., 2016; Wu et al., 2016; Smith & Meger, 2017; Tulsiani et al., 2017; Wu et al., 2017; 2018). While straightforward to use, voxels induce a cubic computational cost, scaling poorly to higher resolutions and complex objects. Numerous computationally efficient approaches have arisen, such as octree methods (Riegler et al., 2017; Tatarchenko et al., 2017; Häne et al., 2017), which represent voxel objects with adaptive degrees of detail. Most similar to mesh models are point clouds methods, which represent 3D objects through a set of points in 3D space (Fan et al., 2017; Qi et al., 2017; Insafutdinov & Dosovitskiy, 2018; Novotny et al., 2017). Point clouds represent only the surface information of 3D objects, making them more efficient and scalable. However, as they do not define surface information beyond each point’s local neighborhood they must be uniformly sampled over a surface, and so to encode high levels of detail, the sampling density over the entire surface must increase.

3. Background

In this section, we review GCNs (Kipf & Welling, 2016), a key component for mesh generation and reconstruction systems, and outline how the Chamfer Distance has been previously employed as a loss for mesh reconstruction.

3.1. Graph Convolutional Networks

Let $\mathcal{G}$ be a graph with $N$ vertices defined by an adjacency matrix $A \in \mathbb{R}^{N \times N}$ and a vertex feature matrix $H \in \mathbb{R}^{N \times F}$, where $F$ denotes the number of features. A GCN layer takes as input both $A$ and $H$, and produces a new vertex feature matrix $H' \in \mathbb{R}^{N \times F'}$ with $F'$ features as follows:

$$H' = \sigma(AHW + b),$$  

where $\sigma$ is an arbitrary activation function, and $W \in \mathbb{R}^{F \times F'}$ and $b \in \mathbb{R}^F$ are the learnable weight matrix and bias vector, respectively. By stacking multiple such layers, information is exchanged throughout the graph such that, after $k$ layers, the information at a given vertex will, for the first time, reach its $k^{th}$ depth neighbor. Alternatively, $k^{th}$ depth information can be immediately reached using the $k^{th}$ power of adjacency matrix in Eq. 1 (Defferrard et al., 2016; Levie et al., 2017; Cucurull et al., 2018). Note that, as discussed in Section 1, GCN layers equate any given vertex to a summary of its neighbourhood, and their repeated application may over-smooth important local information (Li et al., 2018).

3.2. Chamfer Loss: Vertex-To-Point Loss

The Chamfer Distance between predicted and ground truth objects has become a standard metric for 3D reconstruction (Wang et al., 2018; Insafutdinov & Dosovitskiy, 2018; Groueix et al., 2018a; Sun et al., 2018; Fan et al., 2017). This loss is defined as:

$$\mathcal{L}_{\text{Chamfer}} = \sum_{p \in \hat{S}} \min_{q \in S} \|p - q\|_2 + \sum_{q \in S} \min_{p \in \hat{S}} \|p - q\|_2$$  

(2)

and is computed between two sets of points, $\hat{S}$ and $S$, sampled from the predicted surface and the ground truth surface. As discussed in Section 1, this metric performs poorly when directly applied to two sets of mesh vertices as it does not take into account the faces which they define, and because of the difficult learning problem associated with matching highly arbitrary vertex placement on a surface. To avoid these issues, Wang et al. (2018) define $\hat{S}$ as a large set of predicted vertex positions, and $S$ as a large, pre-computed set of points sampled from the ground truth surface (see Figure 3a). Defining the ground truth set as a dense uniform sampling over the target surface avoids issues with inconsistent vertex positions across similar objects. However, this leads to predictions with a high number of vertices packed tightly over the full surface. In this way, the mesh predictions resemble a point cloud, and thus fail to take advantage of the graceful scaling properties of their representation.

4. GEOMetrics Mesh Reconstruction

In this section, we describe our pipeline for reconstructing adaptive meshes from single images and outline our proposed 0N-GCN as well as our suggested adaptive face splitting.

4.1. Mesh Reconstruction from Images Pipeline

Figure 2 depicts our mesh reconstruction module, which takes as input a mesh model, defined by a set of vertex positions and an adjacency matrix, together with an RGB image
depicting an object view and outputs a new mesh prediction. The module is composed of three distinct phases: feature extraction, mesh deformation and face splitting, which are cascaded $m = 3$ times to obtain incrementally refined mesh predictions. Note that the initial module takes as input a predefined mesh model (e.g. a sphere), whereas each subsequent module is fed the preceding module’s prediction. In this manner, the initial mesh is iteratively deformed and updated to match the input image.

Our feature extraction is based on the method proposed by Wang et al. (2018), where the input image is passed through a deep CNN and the features from 4 intermediary layers are outputted. The feature vector for each vertex of the mesh is then defined by projecting the vertices of the input mesh onto the CNN outputs and extracting their corresponding features. In addition, each vertex feature vector is provided its 3D coordinates $x, y, z$ and also, if available, the final feature vector it possessed in the preceding reconstruction module. Our mesh deformation consists of a graph convolutional model, which takes as input a mesh and deforms it by making a residual prediction for the position of each vertex. The residual prediction is then added to the original position to complete the deformation. The graph convolutional model of this mesh deformation phase is made up of a series of the proposed 0N-GCN layers (see Subsection 4.2 for details). Finally, our face splitting phase, described in Subsection 4.3, encourages local complexity to emerge in regions that require additional detail.

4.2. Zero-Neighbor Graph Convolutional Networks

As described in Subsection 3.1, a potential shortcoming of the standard GCN formulation is that a vertex has no capacity to maintain and directly draw conclusions upon its own information, as this information is smoothed with outside influence at each layer. This outside influence, while useful in global graph understanding contexts, may be detrimental when vital information held at each vertex cannot be derived from its neighbors. This situation is exemplified by meshes, where, if optimally defined, every vertex defines some new surface structure (see Figure 1d). To rectify this problem, we define a Zero-Neighbor update, in which a fraction of a vertex’s feature vector are not updated with the neighbors’ information. This is accomplished by, instead of applying higher powers of the adjacency matrix to reach further depths, taking the adjacency matrix to the power 0 (equivalent to the identity matrix) to exchange with no further depths:

$$H' = HW, \quad H'' = \sigma([AH'_{0:M}A^0H'_{i} + b]),$$

(3)

where $\cdot || \cdot$ denotes concatenation between vectors and $i$ is a feature index. This 0N-GCN provides a soft middle ground between full exchange of information and no vertex communication, where the network can choose how heavily a portion of the features of a given vertex will be influenced by the rest of the graph.

4.3. Adaptive Face Splitting

In the final step of each reconstruction module, the mesh’s set of vertices is redefined over its surface, by adding vertices in regions of high detail. To do so, we introduce a face splitting method, which adaptively increases the set of vertices and the connections between them by analyzing the local curvature of the surface at each mesh face. The curvature at each face is computed by taking the average of the angle between a face’s normal and its neighboring faces’ normals. For a given face $f$, made up of vertices $v_1, v_2, v_3$, its face normal $N_f$ is calculated as:

$$N_f = \frac{e_1 \times e_2}{\|e_1 \times e_2\|},$$

(4)

where $e_1 = v_1 - v_2$ and $e_2 = v_3 - v_2$. The curvature $C_f$ at face $f$ is then computed as:

$$C_f = \frac{180}{|H_f|\pi} \sum_{i \in H_f} \arccos (N_f \cdot N_i),$$

(5)
where $\mathcal{H}_f$ is the set of neighboring faces of $f$. All faces with curvature over a given threshold, $\alpha$, are then selected to be updated. A selected face is updated by adding a new vertex to its center and connecting it to its 3 original vertices, creating 3 new faces. As the new vertex positions are defined by the positions of already existing vertices, the gradients from each vertex are easily defined to flow back through all previous modules. In this way, each reconstruction module is able to identify areas of the current mesh which require increased detail and prescribe them a higher vertex density. This allows the mesh to fully take advantage of the scaling properties of its representation, by concentrating the generation process in areas of high detail.

5. GEOMetrics Losses

In this section, we describe the key contributions made to the mesh prediction task when considering 3D geometry. In particular, we introduce a training objective, considering the local topology and the global structure to produce mesh predictions that properly benefit from the graph representation.

5.1. Differentiable Surface Sampling Losses

We introduce a differentiable sampling procedure which enables us to penalize vertices by the surface they implicitly define, rather than random position. This approach allows predicted meshes to match the target surface without emulating the target vertex positions, which are entirely arbitrary when randomly sampled from the ground truth, while also optimally positioning their vertices and faces.

To do so, we define a discrete probability distribution based on the relative area of each face and sample $n$ times from this distribution to determine the number of points to sample per face. Then, we sample the previously chosen number of points uniformly over each corresponding surface. More precisely, given a triangular face defined by vertices $v_1, v_2, v_3$, following Osada et al. (2002), a point $r$ can be sampled uniformly from the surface of the triangle as:

$$ r = (1 - \sqrt{u})v_1 + \sqrt{u}(1 - w)v_2 + \sqrt{uw}v_3, \quad (6) $$

where $u$, $w \sim U(0, 1)$. This formulation allows us to differentiate through the random selection via the reparametrization trick (Kingma & Welling, 2013; Rezende et al., 2014), as the sampling procedure is defined by a deterministic transformation on the vertex coordinates and the independent stochastic terms.

We apply this sampling procedure to both the predicted mesh and the ground truth and define an alternative Chamfer loss, which operates over these sampled points, rather than the predicted vertices (see Figure 3b):

$$ L_{PtP} = \sum_{p \in \hat{S}} \min_{q \in S} \|p - q\|^2_2 + \sum_{q \in S} \min_{p \in \hat{S}} \|p - q\|^2_2, \quad (7) $$

where $\hat{S}$ and $S$ are the sampled points of the predicted mesh and the ground truth, respectively. An algorithmic description of the entire process is provided in the supplementary material. Note this loss differs from the vertex-to-point loss in that it properly penalizes the surface of the predicted mesh instead of the predicted vertex’s positions.

Building on these ideas, we define an improved loss term to more accurately compare the surfaces of two meshes:

$$ L_{PtS} = \sum_{p \in \hat{S}} \min_{f \in \hat{M}} dist(p, \hat{f}) + \sum_{q \in S} \min_{f \in M} dist(q, f), \quad (8) $$

where $\hat{M}$ and $M$ are the predicted and ground truth meshes, $\hat{f}$ and $f$ the faces, $\hat{S}$ and $S$ the set of points sampled from the surfaces of $\hat{M}$ and $M$, and $dist$ is a function computing the distance between a point and a triangular face\(^2\). This loss is shown in Figure 3c, and an algorithmic description can be found in the supplementary material. Note that this loss provides an exact measure of the distance between a

\(^2\)Calculated using an optimized adaptation of the Distance Between Point and Triangle in 3D algorithm (Eberly, 1999), details provided in the supplementary material.
GEOMetrics: Exploiting Geometric Structure for Graph-Encoded Objects

Figure 5. Renderings from a single reconstructed chair, demonstrating the variety of different local vertex densities which are produced by our approach.

point and a mesh surface. This is in contrast to the Chamfer loss in Eq. 2 and the point-to-point loss in Eq. 7, which are faster to compute, but can drastically lose accuracy if too few points are sampled on either surface. A quantitative analysis of the improvement from these losses is provided in the supplementary material through a toy problem.

5.2. Global Encoding of Graphs

In order to consider the global structure of an object during the reconstruction process, we introduce a global mesh loss. This loss relies on features extracted from a pre-trained mesh-to-voxel model, which is designed as an encoder-decoder network. The mesh-to-voxel encoder takes as input a mesh graph and produces a latent embedding, from which the 3D object is reconstructed, through the decoder, in a voxelized format. In this manner, objects with structural similarity in voxel space, will have similar latent representations, without requiring similar placement of vertices. The proposed global mesh loss is then defined as

$$L_{\text{Latent}} = \| E(M) - E(\hat{M}) \|^2_2$$ (9)

where $E$ corresponds to the encoder function of the mesh-to-voxel network. This process is depicted in Figure 4. The encoder network of the mesh-to-voxel model is built by stacking 0N-GCN layers, followed by a max pooling operation applied to the set of vertices as in Ciregan et al. (2012) to produce a single fixed length latent representation. The decoder is a 3D deconvolutional network (Choy et al., 2016), in following with the network defined in Smith & Meger (2017), to perform image to voxel mappings. The complete mesh-to-voxel network is pre-trained by minimizing the mean squared error on the voxelized representations prior to being used in the GEOMetrics system.

5.3. Optimization Details

Finally, we present the complete training objective for our mesh reconstruction system. This function combines our differentiable surface sampling losses, our global structure loss, along with two regularization techniques defined by Wang et al. (2018): an edge length minimizing regularizer $L_{\text{Edge}}$ and a Laplacian-maintaining regularizer $L_{\Delta \lambda}$, pushing the predicted mesh to be smooth and visually appealing.

The final loss function of our system is defined as:

$$L = \gamma_1 L_{\text{Latent}} + \gamma_2 L_{\text{PtS}} + \gamma_3 L_{\text{Edge}} + \gamma_4 L_{\Delta \lambda},$$ (10)

where $\gamma_i$ are hyper-parameters weighting the importance of each term. During the initial stages of training we approximate the $L_{\text{PtS}}$ term using the defined $L_{\text{PtP}}$ loss function for faster computation. Note that the loss $L$ is applied to the output of each mesh reconstruction module.

6. Experiments

In this section, we demonstrate our algorithm’s ability to reconstruct the surface information of 3D objects from single RGB images by taking advantage of the benefits of the mesh representation. We evaluate on this task across 13 classes of the ShapeNet (Chang et al., 2015) dataset. In addition, we present an ablation study to demonstrate how our algorithm’s individual components contribute to its overall performance.

6.1. Dataset

The dataset consists of mesh models, voxel models, and RGB images computed from 13 large classes of CAD models found in the ShapeNet dataset (Chang et al., 2015). Mesh models were computed from the CADs by removing all texture information and downscaling their size so that each model possesses less than 2000 vertices, where possible. From these mesh models, voxelized counter parts were produced at $32^3$ resolution. From each CAD model, 24 RGB images were produced, from random viewpoints, with the camera projection matrix recorded for use in the feature
Table 1. Results on ShapeNet 3D object reconstruction reported as per class surface sampling F1 scores and mean F1 score.

<table>
<thead>
<tr>
<th>Category</th>
<th>3D-R2N2 (Choy et al., 2016)</th>
<th>PSG (Fan et al., 2017)</th>
<th>N3MR Vertices</th>
<th>Pixel2Mesh Vertices</th>
<th>Ours Vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plane</td>
<td>41.46</td>
<td>68.20</td>
<td>62.10</td>
<td>2466</td>
<td>89.00</td>
</tr>
<tr>
<td>Bench</td>
<td>34.09</td>
<td>49.29</td>
<td>35.84</td>
<td>642</td>
<td>72.11</td>
</tr>
<tr>
<td>Cabinet</td>
<td>49.88</td>
<td>39.93</td>
<td>21.04</td>
<td>642</td>
<td>57.57</td>
</tr>
<tr>
<td>Car</td>
<td>37.80</td>
<td>50.70</td>
<td>36.66</td>
<td>642</td>
<td>60.39</td>
</tr>
<tr>
<td>Chair</td>
<td>40.22</td>
<td>41.60</td>
<td>30.25</td>
<td>642</td>
<td>54.38</td>
</tr>
<tr>
<td>Monitor</td>
<td>34.38</td>
<td>40.53</td>
<td>28.77</td>
<td>642</td>
<td>51.39</td>
</tr>
<tr>
<td>Lamp</td>
<td>32.35</td>
<td>41.40</td>
<td>27.97</td>
<td>642</td>
<td>48.15</td>
</tr>
<tr>
<td>Speaker</td>
<td>45.30</td>
<td>32.61</td>
<td>19.46</td>
<td>642</td>
<td>48.84</td>
</tr>
<tr>
<td>Firearm</td>
<td>28.34</td>
<td>69.96</td>
<td>52.22</td>
<td>642</td>
<td>74.64</td>
</tr>
<tr>
<td>Couch</td>
<td>40.01</td>
<td>36.59</td>
<td>25.04</td>
<td>642</td>
<td>73.20</td>
</tr>
<tr>
<td>Table</td>
<td>43.79</td>
<td>53.44</td>
<td>28.40</td>
<td>642</td>
<td>51.90</td>
</tr>
<tr>
<td>Cellphone</td>
<td>42.31</td>
<td>55.95</td>
<td>27.96</td>
<td>642</td>
<td>66.30</td>
</tr>
<tr>
<td>Watercraft</td>
<td>37.10</td>
<td>51.28</td>
<td>43.71</td>
<td>642</td>
<td>70.24</td>
</tr>
<tr>
<td>Mean</td>
<td>39.01</td>
<td>48.58</td>
<td>33.80</td>
<td>642</td>
<td>59.72</td>
</tr>
</tbody>
</table>

6.3. Single Image Reconstruction

We evaluate our method’s performance quantitatively by comparing its ability to reconstruct mesh surfaces from single RGB image to an array of high performing 3D object reconstruction algorithms. To do this, we sample points from both the surface of the predicted object and the ground-truth object and compute the F1 score. In following with (Wang et al., 2018), precision and recall are calculated using the percentage of sampled points which exists within a threshold of 0.0001 any sampled point in the compared surface. State of the art results of mesh approaches, N3MR (Kato et al., 2017) and Pixel2Mesh (Wang et al., 2018), a point cloud method, PSG (Fan et al., 2017) and a voxel baseline, 3D-R2N2 (Choy et al., 2016), are reported from Wang et al. (2018). We also compare mesh-based approaches in terms of space requirements (number of vertices). The results of this comparison are summarized in Table 1. As shown, our GEOMetrics system boasts far higher performance than previous approaches, with an average increase in F1 score of 7.65 points across all classes, and improved score in all classes but one, where we experience a negligible drop of 0.87 points. In addition, in all cases, our system requires notably less vertices than the previous mesh-based state of the art Pixel2Mesh, e.g. cellphone objects require 6 × less vertices, whereas lamp objects require 3 × less vertices. With an average of 574.06 vertices used across all classes, the vertex requirements drop as much as 4.3 × on average, highlighting the potential of the adaptive face splitting. Moreover, when compared to point cloud and voxel baselines, we also exhibit state of the art results.

Qualitative reconstruction results for each of the 13 classes are displayed in Figure 7. We boast highly accurate reconstructions of the input object, effectively capturing both global structure and local detail. In addition, we render an

---

1 Architecture details provided in the supplementary material.

2 See supplementary material for additional visualizations.
un-smoothed chair reconstruction in Figure 5 with its edges heavily outlined, demonstrating the obtained diverse vertex density across a single object and highlighting the way our system represents simple surfaces with a small number of faces, and shifts to higher density where required. Lastly, Figure 6 depicts a visual comparison between GEOMetrics and Pixel2Mesh reconstructions, where we can observe how GEOMetrics is able to provide reconstructions with higher detail (e.g. the sharpness of the chair legs).

6.4. Single Image Reconstruction Ablation Study

In this subsection, we study the influence of our system’s components and demonstrate their individual importance by comparing our full method’s results on the chair class to ablated versions of the system. We assess the impact of our 0N-GCN layers by replacing them with standard GCNs (Kipf & Welling, 2016) in both the mesh reconstruction as well as the mesh-to-voxel models. We validate the effectiveness of the proposed adaptive face splitting by substituting it by a procedure in which all faces are split at the end of each reconstruction module, keeping uniform detail and maintaining approximately the same the number of vertices as the full approach. We then check the importance of each one of the newly introduced losses by removing them at training time. Note that, when the face sampling losses \( L_{\text{PtP}} \) and \( L_{\text{PtS}} \) are removed, we replace them by the vertex-to-point loss \( (L_{\text{VtP}}) \) proposed by Wang et al. (2018). Finally, we compare our method to the Pixel2Mesh, when roughly equivalent in terms of number of vertices.

The results of this ablation study are reported in Table 2. As shown in the table, the biggest effect comes from the introduction of the adaptive face splitting, with a drop of 6.23 points when replacing it with a uniform splitting heuristic. Moreover, assisting the model to give more importance to vertex features through the 0N-GCN also appears to be relevant. The losses proposed to train the whole system also play an important role, as ignoring them leads to a decrease in performance of 3.69 points and 1.02 points, respectively. Moreover, training Pixel2Mesh baseline to use as few vertices as GEOMetrics leads to notably worse performance. These results empirically justify the contributions of our GEOMetrics system.

<table>
<thead>
<tr>
<th>Ours</th>
<th>GCN</th>
<th>Unif. Split.</th>
<th>No ( L_{\text{latent}} )</th>
<th>( L_{\text{VtP}} )</th>
<th>Pixel2Mesh</th>
</tr>
</thead>
<tbody>
<tr>
<td>56.61</td>
<td>54.57</td>
<td>50.33</td>
<td>55.59</td>
<td>52.92</td>
<td>38.13</td>
</tr>
</tbody>
</table>

Table 2. GEOMetrics ablation compared to full method (ours) and Pixel2Mesh. Results reported as mean F1 score on the chair class.

7. Conclusion

In this paper, we presented GEOMetrics, a novel approach for adaptive mesh reconstruction, which focuses on exploiting the geometry of the mesh representation. The GEOMetrics system reformulates GCNs to explicitly preserve local vertex information and incorporates an adaptive face splitting procedure to enhance local complexity when necessary. Furthermore, the system is trained by introducing a training objective which operates both locally and globally at mesh level, and capitalizes on the geometric structure of graph-encoded objects. We demonstrated the potential of the approach through extensive evaluation on the challenging task of 3D object reconstruction from single images of the ShapeNet dataset. Finally, we reported visually appealing state of the art results, outperforming existing mesh-based methods by a large margin, while requiring (on average) as many as \( 4.3 \times \) less vertices. Future research directions include addressing the restrictive constant topology prescribed by the initial mesh object through reconstruction and generation methods, which adapt the topology to match the target mesh.
Acknowledgements

This work was supported by the National Sciences and Engineering Research Council (NSERC) Canadian Robotics Network (NCRN)

References


Eberly, D. Distance between point and triangle in 3d. Geometric Tools, 1999.


Sun, X., Wu, J., Zhang, X., Zhang, Z., Zhang, C., Xue, T., Tenenbaum, J. B., and Freeman, W. T. Pix3d: Dataset and


