A Survey on 3D Mesh Reconstruction from Single Image

Emre Girgin
Department of Computer Engineering
Bogazici University
Istanbul, Turkey, 34342
`emre.girgin@boun.edu.tr`

1 Introduction

There are many ways to represent 3D objects on computers. Some famous ones are voxels, point clouds, meshes, and implicit functions. Depending on the application domain, each of them has pros and cons. This study investigates the mesh representation reconstruction by exploiting just a single image of an object. This methodology has two main goals: The first one is to represent an object as a mesh, and the other is to do that using just a single image.

Firstly, the mesh representation has several advantages over the other alternatives. The triangular meshes are lightweight and ready to use for many domains like the movie and game industries. The existing tools in the field can render the resulting triangles directly, which makes it the industry standard.[10] Also, the object's surface is already constructed, and it is not needed to analyze the object's shape as it is needed for point clouds, which is non-trivial. Moreover, the surface is more detailed as opposed to voxels. In order to increase the resolution of the voxels, octree representations are introduced. However, these representations have lots of data to be processed which is non-preferable due to time constraints.

Secondly, extracting the object from a single image is superior to its counterpart, the Multi-View Geometry (MVG). MVG has some apparent disadvantages over the single view reconstruction. The main challenge in MVG is to reconstruct the unseen parts of the object. This coverage problem can not be overcome because of the problem's nature. Furthermore, MVG depends on the object's appearance because it can not reconstruct the textureless and non-Lambertian surfaces like transparent and reflective.[10]

Traditionally, the 3D mesh reconstruction from images is conducted using Multi-View Stereo (MVS) and Structure from Motion (SfM) methods. While MVS and MVG resemble each other, all of the methods mentioned require multiple images with tiny points of view differences. Also, these methods need correspondences between 2D images requiring the camera parameters to be known or estimated. The MVS method had a slight advantage over the other methods where the depth channel of the image is present. These intermediate 2.5D (RGBD) images gave the 2D convolutional neural networks a chance to estimate the mesh structure, acting as pioneers in the deep learning based techniques used in the state-of-the-art. [1]

The modern methods of 3D mesh reconstruction from a single image exploit the non-convex optimization properties of the deep learning methods. There is not a unique solution to reconstruction from a single image since it is ill-posed.[1] However, most mesh reconstruction algorithms adopt a coarse-to-fine deformation framework starting from a base template.

The structure of this study is as follows: in the related work section, some of the modern deep learning based works are summarized and compared. In the next section after that, a popular and widely known paper called "Pixel2Mesh" is examined in detail. The conclusion section wraps up the presented methods and discusses if the problem is solved or if the issues persist.

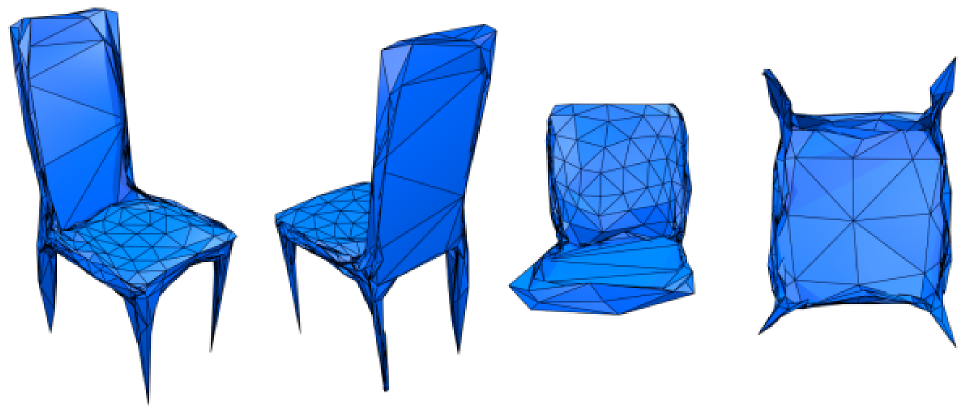

Figure 1: An example of adaptively splitted faces of GEOMetrics[8]. The flat regions of the chair are represented with less number of triangles whereas the parts with high curvature has more number of triangles. Figure from [8].

2 Related Work

In this section, some of the existing methodologies in the literature is presented. Also in Table 1 their comparison is provided.

2.1 GEOMetrics

Smith et al. [8](2019) proposed a modified GCN and a new objective function minding the global appearance of the mesh. Their architecture takes an RGB image and a 3D mesh as inputs and extracts some features through the CNN. Then, projecting the mesh vertices on those features calculates the features corresponding to the vertices. Then the calculated features and the mesh is fed to a Zero-Neighbor GCN. 0N-GCN is a pruned GCN where the features do not depend on the neighbors' coordinates but only on their features. This no vertex communication network prevents being smoothed by the neighboring surfaces. Then the resulting mesh is fine-tuned in the adaptive face splitting module. The adaptive face splitting module upsamples the mesh regarding the local curvature of the surface. This module adds new vertices to the triangles where their curvature is large. The curvature of each face is calculated as the average between its surface normal and its neighboring faces'. This methodology encourages the local complexity of the parts where additional detail is required and divides the flat regions into fewer number triangles. (See Figure 1) It helps to decrease the total number of vertices and space complexity. This feature extraction, mesh deformation, and adaptive face splitting modules are repeated three times to refine the mesh predictions incrementally.

The loss functions of the study gave the name of it. GEOMetrics stands for Geometrically Exploited Object Metrics, and they invented two new objective functions. They introduced a differentiable surface sampling loss which compares the predicted and target surfaces without minding the vertex positions. They sample a set of points on the triangles uniformly and calculate the Chamfer Distance using those sampled points only. This point-to-surface loss measures the distance between the mesh surface and the point. The other loss function they used is the global mesh loss. They use a pretrained encoder-decoder architecture that maps meshes to latent space and recovers them as voxels. The mean squared error in the latent space gives the coarse distance between two mesh models since they had to produce similar voxels. They also utilized Laplacian and edge-length regularization.

2.2 Topology Modification Network

Similar to Pixel2Mesh[10], GEOMetrics[8], and Easymesh[9], a work of Pan et al. [6](2019) designed a deformation based framework. However, their end-to-end trained neural network was able to modify the initial topology as well.(See Figure 2) The mean is that they can reconstruct any genus even though it starts from a genus-0. This Topology Modification Network makes tears on the mesh surface to change the current topology. Their architecture is composed of three sub-modules. A

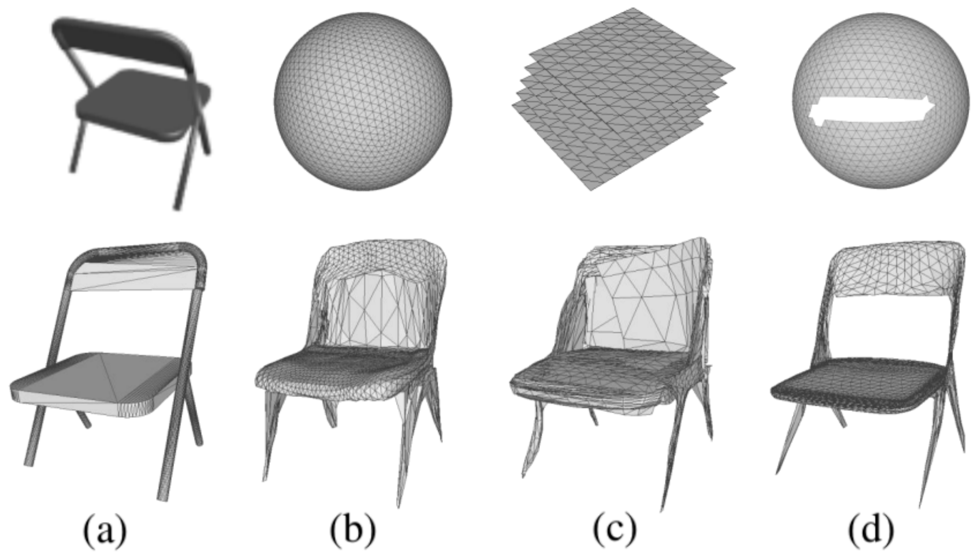

Figure 2: The Topology Modification Network [6] can adapt to the different topologies, although it starts from genus-0. (a) Groundtruth mesh and image (b) Fixed topology deformation (c) Multiple template meshes (d) Dynamically modified topology. Figure from [6]

mesh deformation block takes the initial mesh genus-0 and regresses the offset in the geometry. Then a topology modification network takes that mesh and determines an error score for each face on the mesh by an error estimation network. Then, depending on these estimated errors, a pruning mechanism removes some of the triangles from the mesh. The pruning threshold is decreased by scaling a discount factor. This mesh deformation and topology modification modules are repeated two times. Then boundary refinement module adjusts the vertices at the jagged regions caused by the topology modification at the end of the framework to have a more visually appealing result. All the modules have the extracted features of the input image processed by an encoder.

They have several loss functions used altogether. They adopted Chamfer distance and a mean squared error for error estimation for loss functions and had four additional regularization terms: Boundary, normal, smoothness, and edge-length. Boundary loss penalizes the zigzags and enforces the boundaries at the jagged regions. Normal loss provides surface normal consistency, and smoothness loss minimizes the angle between two neighboring faces. The edge-length loss minimizes the edges of the triangles and prevents flying vertices.

2.3 EasyMesh

Sun et al. [9] (2020) proposed a Generative Adversarial Neural Network setup to reconstruct 3D mesh from a single image. The framework consists of two stages: a silhouette detector and a mesh reconstruction network. Both stages are trained separately, so there is no need to have image and 3D mesh pairs to supervise. Instead, they require a raw image to silhouette and silhouette to 3D mesh pairs. Also, they introduced a new data structure to store the mesh, which is the coordinates of the points sampled from the surface, called geometry images with 3 channels each of them representing x, y, z coordinates of the vertices. This geometry image format allows 2D convolutional operations and point cloud processing techniques to be used since it also preserves the neighboring information. After obtaining the silhouette, a series of residual blocks estimates an initial mesh in geometry image format. While the initial mesh is forming, the camera's viewpoint is also predicted in parallel. Then, a shape deformation network both regresses the coordinates of the vertices and augments the number of points. The mesh obtained at the end is rendered as a set of silhouettes, and a discriminator decides if they seem like a real object or not.

Since their work includes the solution of many subproblems, they have several loss functions. They adopted Chamfer Distance, normal loss, and edge length regularization for the mesh. For the GAN setting, they stuck with the Wasserstein loss. They also used the squared Euclidean distance between groundtruth and predicted the loss function for viewpoints estimation and silhouette rendering.

2.4 Residual MeshNet

While most of the studies deform a 3D initial mesh, a work from Pan et al. [7] (2018) adopts a different type of initial mesh. Their architecture starts from a 2D unit square, and consecutive MLPs map those 2D points into 3D space. The shape properties of the object came from the features extracted from a ResNet-18. The MLPs are computed in parallel paths generating multiple meshes consistent with each other, corresponding to the different patches of the whole final mesh. They also added shortcut connections between MLPs so that each block predicts the offset with respect to the previous one.

As the loss functions, they measured the Chamfer Distance, and to ensure the multiple deformed meshes are consistent, they introduced a pairwise consistency loss across meshes. This loss function takes the average distance between a point and a plane where the plane is the closest one to the point except the triangle from which the point is coming so that the continuity in the overlapping areas is preserved.

2.5 Deep Marching Cubes

Marching cubes (MC) are an explicit surface reconstruction algorithm from a given implicit representation. However, this operation is not differentiable and not suitable for deep learning pipelines. Liao et al. [4] (2018) designed an end-to-end deep learning framework inspired by the MC. Previous implicit function based mesh reconstruction algorithms transform a given input representation to an intermediate representation like (TSDF) by neural networks and then applies the traditional MC. However, since the neural network's output is the implicit representation, the loss function had to be defined on it but not on the mesh object itself. This type of loss function leads to noisy mesh objects. Their differentiable marching cubes layer can be attached at the network's end and exploit arbitrary loss functions. Their work is suitable for any topology and can also process sparse unstructured 3D data. The output of the network is occupancy probabilities and vertex displacements.

They defined an arbitrary loss function composed of four sub losses. Point to mesh loss tries to minimize the expected error over the distribution representing the point to mesh distance. It is calculated by measuring the Euclidean distance between the triangle closest to a given point. Also, their occupancy loss encourages the voxels at the boundary to be unoccupied and the main parts to be occupied since they build a fixed-sized voxel grid while feature extraction. They also have smoothness loss, limiting the occupancy pairwise and a curvature loss, minimizing the expected disparity in normal vector orientations.

2.6 Neural 3D Mesh Renderer

Kato et al.[2] (2017) proposed a neural network based approach to make rasterization operation differentiable. They approximated the gradients for rasterization by interpolation. The value of a pixel while rendering a triangle depends on the position of vertices. This means the value of the pixel changes when the vertices move. Therefore the color change of the pixel is a function of the position of the vertices. However, this change is discrete and not differentiable. To overcome this, they interpolated the pixel value based on the vertex's position. They were able to rasterize a given mesh with a neural network.

They used that rasterization neural network to generate 3D meshes from a single image. By starting an initial spherical mesh, they progressively deformed it. The deformed mesh is rendered using the neural renderer, and the resulting silhouette is compared with the groundtruth silhouette. The mesh generator network is updated by using the calculated loss because the whole process is differentiable due to the neural renderer. As the loss functions, they used silhouette and smoothness loss. The silhouette loss is the negative intersection over union, and smoothness loss satisfies the angle between neighboring faces that are close to 180 degrees. They also used their renderer to transfer style in 3D.

Method	Year	Architecture	Loss Function	Dataset	Evaluation Metrics
GEOmetrics[8]	2019	CNN + GCN	Surface Sampling + Global + Laplacian + Edge Length	ShapeNet	F1 Score
Residual MeshNet [7]	2018	ResNet18 + MLP	CD + Pairwise Consistency	ShapeNetCore	CD
Deep Marching Cubes[4]	2018	Encoder Decoder	Geometric + Occupancy + Smoothness + Curvature	ShapeNet	CD + Accuracy + Completeness
Topology Modification Network[6]	2019	Encoder Decoder	CD + Error Estimation + Boundary + Normal + Smoothness + Edge-Length	ShapeNet	CD + EMD
Soft Rasterizer[5]	2019	Encoder Decoder	Silhouette + Color + Laplacian	ShapeNet	3D IoU
Neural 3D Mesh Renderer[3]	2017	VGG	Silhouette + Smoothness	ShapeNetCore	3D IoU
Pixel2Mesh[10]	2018	G-ResNet + VGG	CD + Normal + Laplacian + Edge Length	ShapeNet	CD + EMD + F1
EasyMesh[9]	2020	GAN	CD + Normal + Edge Length	ShapeNet	CD

Table 1: A comparison on methods reconstructing 3D meshes from a single image.

2.7 Soft Rasterizer

Similar to Neural Mesh Renderer, Liu et al. (2019) developed a neural network based rasterization algorithm and used it to reconstruct 3D mesh from a single image. However, their way of rasterization differs. NMR estimates the gradients in a hand-crafted manner (interpolation) while backpropagation, whereas SoftRas changes the forward rasterization procedure. They directly render meshes via differentiable functions and signal error through backpropagation. They model each triangle's contribution to a pixel as a probability distribution, and an aggregation function fuses that contributions. The probability distribution is calculated by measuring the Euclidean distance between a pixel and a vertex of the triangle faces. The aggregation function calculates the convex combination of the background and triangle color to find the final value.

This truly differentiable rendering pipeline helps to reconstruct the meshes in an unsupervised manner. Given an input image, the color and shape generator encoder-decoder networks regress the color and the mesh. Then the SoftRas network takes them and returns the silhouette and the colored image. Besides, since the probability distribution is calculated for every triangle, including the unseen ones, this method has a higher performance for predicting unseen parts in the input image. They used negative IoU as silhouette loss and classical L_1 loss for the colored image as the loss function. Also, they applied Laplacian regularization.

3 Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images

Pixel2Mesh is an end-to-end deep learning framework that produces 3D mesh representations of an object whose a single image is provided. The algorithm deforms a mesh in a coarse-to-fine manner, starting with an initial topology. The method projects the computed geometry onto the extracted features of the input image and regresses the vertex coordinates based on the matches.

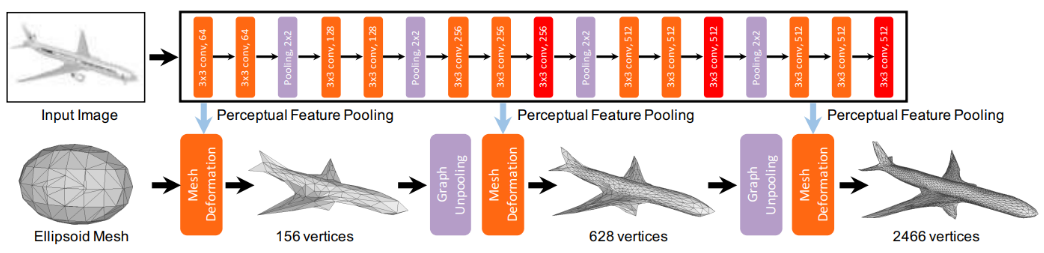

Figure 3: The overview of the Pixel2Mesh architecture. The information flow from Image Feature Extractor (upper) to the Mesh Deformation Network (lower) is conducted via the Perceptual Feature Pooling Layers. Figure from [10].

3.1 Architecture Overview

The system comprises two sub-networks: image feature extractor and cascaded mesh deformation network. (See Figure 3) The image feature extractor is a VGG-16 network whose classifier is discarded. The cascaded mesh deformation block is more sophisticated compared to the former. It starts with an initial mesh object and gradually refines the residual, which is the geometry. The cascaded structure of this network is composed of three mesh deformation blocks following one after another. The resolution of the mesh is increased in each mesh deformation block up to a fixed number. Therefore, the number of deformation blocks controls the trade-off between model complexity and mesh quality. Also, there is a vertical connection between the image extractor and the mesh deformation block, carrying the supervision information of the extracted image features to the coordinates of the regressed points. The positions of these connections are fixed. The mesh deformation network has some fundamental components: the initial ellipsoid, mesh deformation block, and the unpooling layer.

3.2 Initial Mesh

The initial mesh is an ellipsoid placed in front of the camera. This ellipsoid represents the mean of all objects to be tried to construct, which are closed-loop 3D surfaces. The initial mesh has genus 0, meaning that there are no holes on the surface. Therefore, since the topology is fixed at the beginning of the network, this framework can not regress the other objects with different genera like donuts or cups. Also, scene reconstruction of a room or stadium is also not possible with this approach. However, the network can reconstruct all the classes presented in the dataset, like planes, cars, etc. The ellipsoid starts with 156 vertices and reaches 2466 at the end.

3.3 Mesh Deformation Network

The mesh deformation network consists of three mesh deformation blocks with a Perceptual Feature Pooling layer and a G-ResNet. This block takes the point coordinates, 3D features, and image features and returns the new coordinates and new 3D features.

The perceptual feature pooling layer is responsible for carrying the extracted features from the image extractor to the 3D vertex coordinate regressor by projecting onto them. Since there will be an offset error for each projection, the value of the corresponding position is estimated by interpolation. The computed features are concatenated with the features coming from the previous block and processed by the G-ResNet. Since there are no features before the first block, the vertex coordinates are also given to the first block as features.

The G-ResNet is 14 layers of graph convolutional neural network (GCN) with 128 channels. The main difference between vanilla GCN and G-ResNet is the skip connections like the ones in the ResNet architectures. The vanilla GCN computes the features as a combination of its and its neighbors' features coming from the previous layer. However, this method allows for exchanging information between neighbors, and it may be a problem resembling the small perceptive field in the 2D CNNs. To overcome this challenge, a series of connections to the distant points are added to increase the information exchange. Additionally, a final layer is added at the end of the G-ResNet to regress the 3D coordinates.

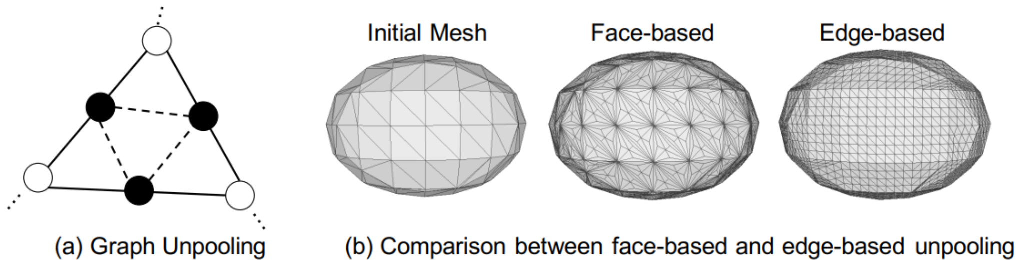

Figure 4: (a) The proposed upsampling method in Pixel2Mesh. (b) A comparison to other methods in terms of uniform distribution. Figure from [10].

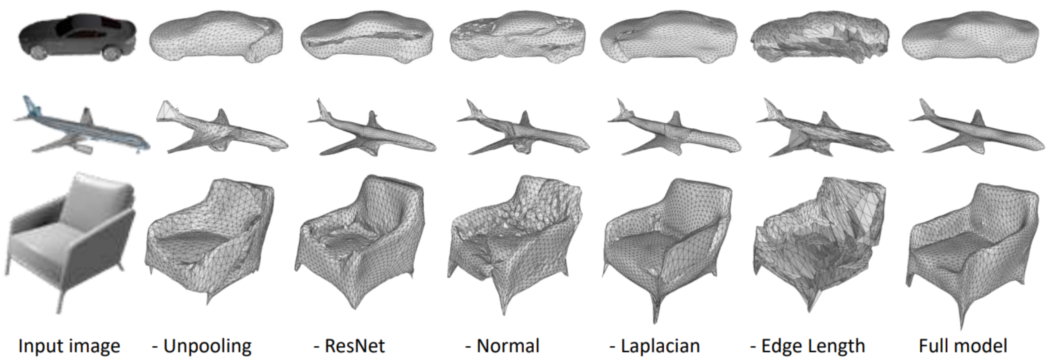

Figure 5: The effect of each component in Pixel2Mesh is visualized. Figure from [10].

After the new coordinates are regressed, the mesh is processed by the graph unpooling layer to increase the resolution. It upsamples using an edge-based method which adds new vertices at the middle of each edge of a triangle and connects them.(See Figure 4) This method provides a uniform upsampling and increases the resolution by 4. This upsampling mechanism is needed because when the target number of vertices is given as the initial resolution, the algorithm makes some mistakes at the beginning of the network and can not fix them later on.

3.4 Loss Functions

As for the loss functions, Pixel2Mesh uses two loss functions and two regularization methods. The first loss function is the Chamfer loss which sums the distance between each predicted point to its closest groundtruth counterpart. This calculation is done vice versa because this operation is not symmetrical. The study also has the normal loss, which forces the edges of the triangles to be perpendicular to the surface normal. This loss function provides the surface normal consistency, and its weight is pretty tiny compared to the Chamfer Loss. The regularization terms are Laplacian and edge-length. The Laplacian regularization encourages the neighboring pixels to make similar movements and prevents the potential self-intersection. It also provides geometric consistency. The other regularization term, edge-length, minimizes the distance between a vertex and its neighbors to avoid outliers and flying vertices. The effect of each component on the final output is demonstrated in Figure 5.

3.5 Evaluation Metrics

The Chamfer Distance, Earth Mover's Distance, and F1 Score are utilized as the evaluation metrics. Although their work, Pixel2Mesh, outperformed many previous results, the authors also discussed a critical deficiency in this domain. They were able to increase their scores by removing some of the components, whereas the visual quality of the output dropped. Their conclusion was the existing evaluation metrics in the domain can not measure the quality of the surface smoothness or continuity. Therefore, the invention of a more sophisticated new metric is a must to have a fair comparison between methods.

4 Conclusion

Reconstructing 3D meshes from a single image has many advantages over other types of representations and multi-view counterparts. Studies in recent years showed remarkable success in this domain. While Pixel2Mesh [10] utilizes the power of Graph Convolutions, GEOMetrics[8] extended this path and was able dynamically to adapt the face sizes depending on the level of detail in the reconstructed region. EasyMesh[9] designed a Generative Adversarial solution to the reconstruction problem, while ResMeshNet [7] generated multiple mesh patches and linked them. On the other side, while all other works fix the topology, the Topology Modification Network [6] could estimate any topology, although it starts from genus-0. Some studies tried to combine traditional methods with modern deep learning architectures. Deep Marching Cubes [4] contains a differentiable layer inspired by the Marching Cubes algorithm that converts intermediate implicit function representations to 3D meshes. On the other side, Neural Mesh Renderer [2] and Soft Rasterizer [5] designed a neural network based rasterization operation and used them to reconstruct meshes by inverting the process.

However, the variety of those solutions and their level of performance signals that the 3D mesh reconstruction from a single image problem is not entirely solved yet. Although different neural network architectures were used, it is unclear whether the deformation based methods are the proper way of reconstruction. Also, there is a need for various loss functions that are hard to tune especially determining their weights. While Chamfer Distance is widely used, it does not contain any inputs about the surface properties. This absence brings us to another significant deficiency: the lack of a direct surface comparison metric. Two mesh surfaces can be almost identical, with the vertices at completely different positions. Future works in that domain should focus more on a metric that quantifies the surface similarity rather than the vertices, and the performance of deformation based methods should be examined on the more complex topologies.

References

[1] G. Fahim, K. Amin, and S. Zarif. Single-view 3d reconstruction: a survey of deep learning methods. *Computers & Graphics*, 94:164–190, 2021.

[2] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3907–3916, 2018.

[3] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3907–3916, 2018.

[4] Y. Liao, S. Donne, and A. Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925, 2018.

[5] S. Liu, T. Li, W. Chen, and H. Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7708–7717, 2019.

[6] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia. Deep mesh reconstruction from single rgb images via topology modification networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9964–9973, 2019.

[7] J. Pan, J. Li, X. Han, and K. Jia. Residual meshnet: Learning to deform meshes for single-view 3d reconstruction. In *2018 International Conference on 3D Vision (3DV)*, pages 719–727. IEEE, 2018.

[8] E. J. Smith, S. Fujimoto, A. Romero, and D. Meger. Geometrics: Exploiting geometric structure for graph-encoded objects. *arXiv preprint arXiv:1901.11461*, 2019.

[9] X. Sun and Z. Lian. Easymesh: An efficient method to reconstruct 3d mesh from a single image. *Computer Aided Geometric Design*, 80:101862, 2020.

[10] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 52–67, 2018.