Level Set Mesher: Single-image to 3D reconstruction by following the level sets of the signed distance function

Diego Patiño*, Carlos Esteves† and Kostas Daniilidis‡
GRASP Laboratory, University of Pennsylvania
Email: *diegopc@seas.upenn.edu, †machc@seas.upenn.edu, ‡kostas@cis.upenn.edu

***Abstract*—We present Level Set Mesher, a single-image to 3D reconstruction strategy to deform an initial spherical triangular mesh into the 3D geometry of a target shape. Level Set Mesher offsets each vertex in a discrete number of steps following a learned velocity vector field V modeled as a Graph Attention Network. At each step k, we constrain the deformed vertices to lie on the l_k level set of the target shape's signed distance function to guide the deformation process. We show that our approach accurately estimates the surface's normal of the predicted shapes and reduces mesh's artifacts in the final prediction. We compare our approach with state-of-the-art single-image to 3D reconstruction methods and show improvements in accuracy predictions, resulting in better quality and manifold meshes.**

I. Introduction

Single-image to 3D reconstruction is a promising but challenging area of research in computer vision with many potential applications, including robotic manipulation, self-driving cars, and augmented reality. In recent years, there has been increased interest in 3D reconstruction enabled by advances in deep learning techniques and the availability of large 3D shape datasets [1], [2]. However, due to its under-constrained nature, the subject remains an open problem with exciting challenges.

Existing research has addressed the 3D reconstruction problem by extending two-dimensional Convolutional Neural Networks (CNN) capabilities to an extra dimension. 3D CNNs allow neural network-based approaches to predict occupancy voxel grids [3]–[5] that represent 3D shapes. However, these 3D CNNs pose several challenges due to their computational cost and their capacity to approximate objects with smooth surfaces. Instead, other work uses continuous implicit representations of the target shape in the form of deep learning models that discriminate whether a point x is on the surface of the 3D shape [6], [7].

Another popular strategy for single-image shape reconstruction is deforming the vertices of an initial mesh that encloses the target shape until the deformation converges to the target surface [8]–[11]. These types of methods are easy to formulate and to model. However, they often lead to errors in the prediction's topology when the predicted triangles self-intersect, which resulting in non-manifold meshes.

This paper presents a new deformation approach to single-image 3D reconstruction that aims to reduce wrong surface normals, self-intersecting faces, and other artifacts that make the predicted surfaces non-manifold while maintaining high prediction accuracy. Given a 3D shape Ω, we seek to estimate the evolution of an initial spherical surface along a velocity field $V(x,t)$, such that after a time T, the initial surface converges to $\partial\Omega$. We learn such vector field with the help of an attention-based deep network architecture that takes an RGB image as input to produce a discrete set of deformations from an initial spherical topology into the geometry of $\partial\Omega$.

We are interested in learning a deformation field $V(x,t)$ that smoothly deforms the mesh such that the trajectories of a point $x_i \in S^2$ do not interfere with the trajectory of another point $x_j \in S^2$. Intersecting trajectories can cause mesh artifacts that reduce the quality of the final predictions. To learn a smooth deformation, we constrain V to follow the level sets of the target shape's signed distance function (SDF). The intuition behind this choice is that the gradient of a shape's SDF function is a curl-free vector field. The flow therefore, progresses without producing *unexpected turns* that could cause mesh artifacts. We show how the deformation through V occurs for a single shape in Fig. 1 (left).

We evaluate our approach using the ShapeNet dataset in order to compare our method to related methods in the state-of-the-art. For the comparison, we use metrics that reflect the accuracy of the predictions and the *manifoldness* of the final meshes.

In summary, we make the following **contributions**:

- We show that our level set approach increases the accuracy of predicting 3D shapes from single images by learning to deform an initial topology while progressively following a well-behaved deformation field.
- Our learned deformation vector field prevents collisions of the deforming points' trajectories, maintaining a relatively low amount of mesh artifacts measured as self-intersecting faces.

II. Related Work

This section reviews the most relevant work on single-image to 3D reconstruction. We divide this work into three categories according to their predicted geometry representation.

Occupancy voxel grids and point clouds. Several approaches exist wherein CNN architectures act upon 2D images, after which a 3D CNN decoder predicts an occupancy voxel grids representing the target shape [2]–[5], [12], [13]. These

types of architectures come at a high computational cost forcing authors to use low-resolution voxel grids [14], [15], which hurts the predictions' accuracy. Other work, instead, focuses on reconstructing 3D shapes as point clouds [16], [17] by employing generative CNN models that act on unordered point sets [18]–[20]. Although point clouds methods reduce the computational cost, they require post-processing to produce ready-to-use triangular meshes.

Implicit surfaces. Implicit methods have gained popularity due to their ability to predict shapes without discretization of the output space. They lead to memory-efficient surface representation because the space where the surface is embedded does not need to be explicitly stored. In general, these methods involve a deep neural network that learns the decision boundary of a binary classifier [6], [7], [21], indicating whether a point x is on the surface. Other approaches learn to approximate the SDF on a subset of a Euclidean space around the target shape [22]–[25]. Implicit representations allow reconstructing high-resolution meshes by sampling the embedded space. However, sampling operations are expensive, and we therefore need efficient strategies to overcome such limitations. Moreover, implicit approaches require 3D dense ground truth annotations for training. Further, the question of how to learn implicit representations from image data alone remains unanswered [26].

Mesh deformation. Some authors have opted for directly outputting triangular meshes by predicting a set of triangles with geometric constrains [27], or by deforming an initial topology [8]–[11], [28], [29]. The most representative method in this category is Pixel2Mesh [8]. Pixel2Mesh uses a series of graph neural networks (GNN) to deform and refine an initial ellipsoidal topology until it fits the ground truth mesh. This type of approach avoids post-processing and can be trained end-to-end. However, this method suffers from two key drawbacks: 1) it can only predict genus-zero topologies, and 2) the predictions are prone to mesh artifacts like self-intersecting faces or wrong surface normals orientation. Another example of mesh deformation methods is Mesh-R-CNN [11]. Mesh-R-CNN generates the initial predictions from a volumetric CNN backbone based on Mask-R-CNN [30]. Later, a GNN refines the predicted mesh. In contrast to Pixel2Mesh and other mesh deformation methods, Mesh-R-CNN can handle shapes with multiple topologies (shapes with holes). However, it fails to produce jointly accurate and artifact-free meshes.

III. Method

We introduce Level-Set Mesher (LSM) as a new approach to single-image for 3D reconstruction. Our method predicts 3D shapes by deforming an initial mesh with spherical topology through a deformation flow that follows the level sets of the SDF's target shape.

A. Level set deformation field

Let us consider a 3D shape Ω enclosed in the unit sphere S^2 and assume that its surface $\partial\Omega$ is a two-dimensional smooth manifold. It is possible to evolve the surface of every point $x \in S^2$ into a point on $\partial\Omega$, given a velocity vector field $V : \mathbb{R}^3 \mapsto \mathbb{R}^3$. We model the deformation as an initial value problem (IVP) with

$$\frac{dx}{dt} = V(x,t), \quad t \in [0,1]. \tag{1}$$

Note that Eq. 1 admits a unique solution given that V is continuous and Lipschitz in the deformation interval. This is the case when the gradient vector field is induced by the SDF of $\partial\Omega$. Using the SDF's gradient as the vector field in Eq. 1 leads to smoother trajectories of individual points in S^2 to $\partial\Omega$.

In a discrete setup, we approximate the surface of the sphere with a triangular mesh $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. We deform the vertices $x_i \in \mathcal{V}$ through the vector field using backward finite differences on the left side of Eq. 1. We compute the vertex's position at any deformation step $k = 1, ..., n$, and for each vertex i as:

$$x_i^k = x_i^{k-1} + \Delta t \cdot V(x_i^{k-1}), \tag{2}$$

where $x_i^0 \subset S^2$ and $x_i^n \subset \partial\Omega$. Note that the mesh's topology remains unaltered because the edges do not change during the deformation.

We propose to estimate the deformation from x_i^{k-1} to x_i^k by training a sequence of deep neural networks g_θ^k that approximates the vector field at every vertex location through the n steps of the deformation process.

Our model takes an RGB image $\mathcal{I}$ rendered from $\partial\Omega$ as input, plus the vertices of the initial mesh. Then our model computes:

$$z_i^k = f_\phi(K x_i^{k-1},\, \mathcal{I}) \tag{3}$$

$$x_i^k = x_i^{k-1} + h_\psi^k \left(g_\theta^k \left(x_i^{k-1},\, z_i^k,\, g_\theta^{k-1}\left(...\right)\right)\right). \tag{4}$$

In Eq. 3 and 4, we approximate f_ϕ as a 2D CNN with parameters ϕ. Through f_ϕ, our model extracts visual features z_i^k from the input image $\mathcal{I}$. We project x_i^{k-1} into $\mathcal{I}$ using the projection matrix K to extract features at locations where the vertices project onto the image plane. Later, we concatenate z_i^k with the current vertex position and pass it through g_θ^k, a Multi-head Graph Attention Network (MGAN) that compute vertex features on $\mathcal{G}$. The function g_θ^k follows the MGAN formulation from [31], [32].

The third component of our model is another small MGAN h_ψ^k with $tanh(x)$ activation function. This last component takes the vertex features from g_θ^k and predicts the direction and magnitude of the velocity vector we use to evolve the surface as per Eq. 2. The key observation that led us to use a graph attention mechanism is that points in small neighborhoods follow a similar flow. Thus, at each vertex i, our model learns which neighbors to attend – through the attention re-weighting strategy – to predict the next step in the flow.

Note that we are interested in evolving the surface with a smooth vector field capable of capturing complex details of the target mesh's geometry. We need a smooth deformation to keep the surface's normals pointing outward at every step. In this way, we prevent artifacts in the mesh's triangles induced by *erratic* movements of the vertices.

To achieve smooth trajectories, we constrain the deformations to follow to the gradient vector field $-\nabla\phi(x)$, with ϕ the SDF associated to $\partial\Omega$. We enforce such restriction by keeping the points x_i^k close to the corresponding level set of $\phi(x)$

$$\Gamma^k = \{x \mid \varphi(x) = l_k\}. \tag{5}$$

The gradient vector field controls the vertex's deformation by making them *jump* between level sets until they converge to the target shape's surface. We chose Γ^k at values l_k such that $l_k > l_{k+1}$ and $l_n = 0$. Consequently, Γ^n is the surface of Ω, and Γ^0 is the set of points on the surface of a sphere. Note that the deformation follows the IVP dynamic process from Eq. 1 that is guaranteed to converge to the target surface.

Our method diverges from previous related approaches such as Wang et al. [8] where the vertex deformation is expected to occur at a single step. The remaining steps in Wang et al. aim to predict vertex refinements with no clear definition of how the deformation should behave. Our key insight to develop our strategy is that a progressive deformation allows us to learn increasing level of details in the predictions while avoiding undesired flow behavior like the ones observed in Fig. 1 (right).

B. Losses and regularization

We motivate our model under the assumption that the deformation between intermediate level sets is an easier problem to solve than predicting the offsets between Γ^0 and Γ^n in one single step. Thus, our loss function needs to guide the evolving points until they reach the surface of the target shape, Fig. 1 (left).

To learn the appropriate deformation paths, the loss function needs to fit the intermediate predicted meshes to the corresponding level set. However, mesh to mesh comparison is intractable, and thus we employ a proxy loss function through the Chamfer distance:

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} ||x - y||_2^2 + \sum_{y \in S_2} \min_{x \in S_1} ||x - y||_2^2.$$

Let us consider the set $\hat{\Gamma}^k = \{\hat{x}_i^k\}_i^n$ as an approximation of Γ^k. We train our deformation model by minimizing $d_{CD}(\Gamma^k, \hat{\Gamma}^k)$, with Γ^k sampled from ground truth points from the corresponding level sets.

Additionally, we enforce our learned vector field to further prevent mesh artifacts by regularizing the edge length of the predicted triangles. Edge length regularization is designed to keep magnitude of the deformation small, further reducing any potential self-intersection of the triangles. We define both terms in loss function as:

$$\mathcal{L}_{CD} = \sum_{k=1}^{n} d_{CD}(\hat{\Gamma}^k, \Gamma^k) \tag{6}$$

$$\mathcal{L}_{\mathcal{E}} = \sum_{k=1}^{n} \sum_{j=1}^{|\mathcal{E}^k|} ||e_j^k||. \tag{7}$$

The overall loss of the model is a weighted sum of the two losses:

$$\mathcal{L} = \lambda_{CD}\mathcal{L}_{CD} + \sum_{k=0}^{n-1} \lambda_{\mathcal{E}}^k \mathcal{L}_{\mathcal{E}} \tag{8}$$

C. Architecture

We use a 2D Resnet-50 CNN pre-trained on ImageNet [33] as our visual feature network f_ϕ. We model each g_θ^k as a MGAN with six layers, ReLU activation function, and four attention heads. The output of each g_θ^k is a 128-dimensional feature vector that is the concatenation of the four attention feature vectors of dimension 32 each. We allow self-loops in g_θ^k to guarantee that a vertex x_i^k can attend itself.

The final component of our architecture, h_ψ^k, is a single layer MGAN. For this network we use four attention heads with $tanh(x)$ activation function each. The output of h_ψ^k is a 3-dimensional vector representing the offsets that deform x_k into x_{k+1}. The multi-head features from h_ψ^k are averaged rather than concatenated to guarantee a 3-dimensional vector as output.

We use seven different level sets in our experiments including the shape's surface. Before steps $k = 2$ and $k = 4$, we subdivide the predicted meshes using a Mid-Edge scheme [34], [35]. This scheme helps reduce the memory footprint while capturing finer details in the final stages of the deformation. Please see the Supplementary Material for further discussion on the subdivision strategy.

IV. Experiments

We test our method on a subset of ShapeNet [36] following the evaluation protocol defined in [9]. We compare our results with state-of-the-art approaches and focus on the reconstruction accuracy and smoothness of the output meshes.

A. Experimental Setup

1) Data and pre-processing: We benchmark our model on a subset of the ShapeNetCorev1.0 dataset with only 13 classes as in [3], [8], [11]. We used ShapeNet's original dataset split for training, validation, and testing. All meshes are centered at the origin and normalized to a unit cube. We use the image renderings from Choy et al. [37] – along with their extrinsic matrices – as inputs to our model. Each mesh in the dataset is rendered from 24 camera viewpoints.

ShapeNet does not guarantee that all the ground truth meshes are watertight or winding-consistent. We therefore preprocess the meshes into 2-manifolds using [38]. Later, we sample points from different level sets of the mesh's SDF for every pre-processed watertight mesh.

In our experiments, we use seven level sets per shape. We set Γ^0 to be a sphere of radius $r = 1.25$, and Γ^6 to be the shape's surface. We obtain the remaining Γ^k by sampling the mesh's SDF inside a cube of side $s = 3$ centered on the shape. We observe that outer-level sets only carry information about the object's general shape, whereas the level sets closer to the surfaces exhibit more detailed features. Therefore, we sample level sets at exponentially decaying intervals such that

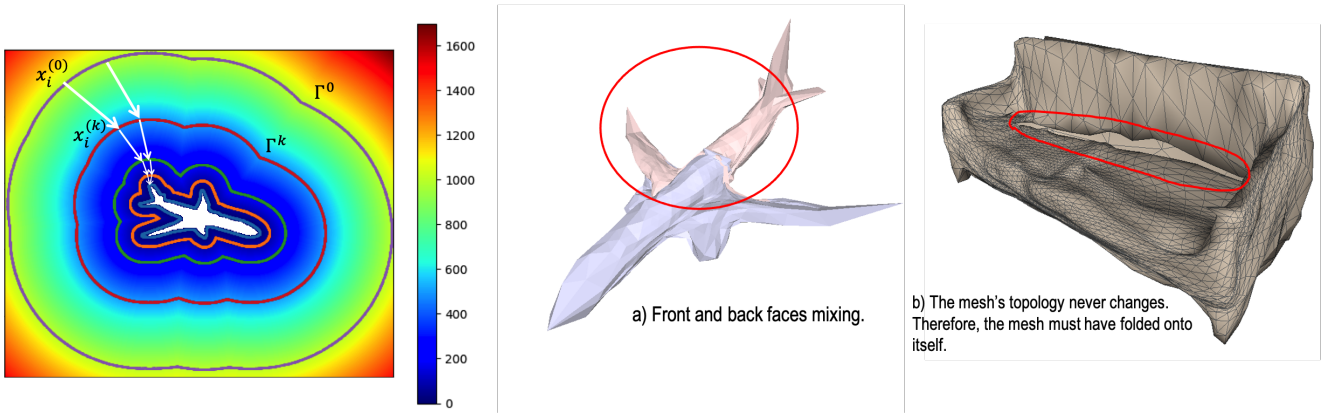

Fig. 1. (Left) Example of a point x_i evolving through the level sets Γ^k of the SDF until reaching the surface of the target shape. (Right) Examples of two self-intersection phenomena when the deformation is not constrained by a proper vector field.

$l_k = 2^{-k}$, for $k = 1, ..., 5$. We provide further details about the level set sampling in the Supplementary Material. Our method takes approximately 0.64 seconds to predict a target shape from its RGB input image at inference time on an NVIDIA v100 GPU.

2) Evaluation metrics: We adopt Pixel2Mesh [9] evaluation protocol and extended it to measure self-intersection and normal consistency on the final predicted meshes. We compute the symmetric squared Chamfer distance between prediction and ground truth for every mesh in the test split, along with two F-scores at τ and 2τ. We use the same threshold values as Pixel2Mesh. We also include a normals' consistency metric defined as the absolute value of the cosine similarity between the predicted normals and their ground truth. Additionally, we include two metrics to evaluate the predictions' manifoldness of each mesh: self-intersection length and self-intersection ratio. The former is computed by adding all of the line segment lengths that result when a pair of triangular faces intersect. The latter is the ratio of the number of faces that self-intersect over the total number of mesh faces.

3) Baselines: We use Pixel2mesh [8] as a baseline because it is the closest approach to ours. We use the implementation provided by [11] and retrain the model following their training recipe on our pre-processed dataset to ensure a fair comparison. As in Pixel2Mesh, we deform an a-priori-defined topology – a sphere –until it fits the target shape. However, we use the level sets data as described in section III to improve the accuracy of our predicted meshes. Our model has more mesh deformation stages than Pixel2Mesh due to the number of level sets we use in our architecture, thus increasing the network's size and capacity. Consequently, we also train a larger version of Pixel2Mesh with five stages as our method for a fair comparison.

We include MeshRCNN [11] as another baseline in our comparative study. Unlike our approach, MeshRCNN can predict meshes with holed topology. We acknowledge that our proposed method produces only genus-0 topology. Therefore, we also show experiments on a subset of ShapeNet that only contains shapes with zero holes to ensure a fair comparison.

Method	Input mesh $(\mathcal{V}, \mathcal{E})$	Output mesh $(\mathcal{V}, \mathcal{E})$
P2M [8]	(162, 320)	(2562, 5120)
P2M, 5 stages	(12, 20)	(2562, 5120)
MeshRCNN [11]	N/A	$\sim$ (2122, 4242)
Ours	(162, 320)	(2562, 5120)

TABLE I
INPUT AND OUTPUT MESH'S SIZE FOR ALL METHODS IN THE STUDY.

4) Training: We train our model for 45 epochs using Adam optimizer [39] with a constant learning rate of $\text{lr} = 1\text{e}{-4}$. We use 64 images per batch, which give us a training time of about 96 hours on two Nvidia v100 GPUs.

We set the Chamfer loss' weight to $\lambda_{CD} = 1$, and the edge regularization loss' weights to $\lambda_{\mathcal{E}}^{0,1} = 0.05$, $\lambda_{\mathcal{E}}^{2,3} = 0.1$, $\lambda_{\mathcal{E}}^{4,5} = 0.2$ to compensate for the bigger size of the predicted meshes at the first steps.

We report details about the input and output meshes for all the baselines in Tab. I. The input images have a resolution of 224x224. We normalize the images with the full dataset's mean and standard deviation. The model begins deforming an initial triangular mesh representing a sphere with 162 vertices and 320 faces.

B. Results and Discussion

We show the main results from our experiments in Tab. II, and Tab. III. The experiments demonstrate that following the level sets on our deformation strategy outperforms the Pixel2Mesh baseline in the metrics that compare the prediction's accuracy to the target object. Note that some categories have low self-intersection ratios of less than 10% faces, and normal consistency achieves significantly higher values than the state-of-the-art, especially compared to Pixel2Mesh.

Starting from a coarser mesh and adding more graph convolutional layers on the base implementation of Pixel2Mesh also aids in reducing the self-intersections. We believe this occurs due to the significantly lower resolution of the input meshes in Pixel2Mesh with five stages. As shown in Tab. I and Fig. 6 in the supplementary material. Notice that in this case, self-intersections are unlikely to appear in the first stages of the

	Full Test Set						No Holes Test Set					
	CD(↓)	$F1^{\tau}(\uparrow)$	$F1^{2\tau}(\uparrow)$	Normal consist.	Self. inter. length	Self. inter. ratio	CD(↓)	$F1^{\tau}(\uparrow)$	$F1^{2\tau}(\uparrow)$	Normal consist.	Self. inter. length	Self. inter. ratio
P2M [8]	0.560	60.58	74.31	0.723	9.31	0.271	0.542	60.94	74.99	0.775	7.22	0.234
P2M, 5 stages	0.541	61.59	75.02	0.737	**5.64**	**0.106**	0.5174	62.16	75.69	0.798	**2.47**	**0.050**
MeshRCNN [11]	0.492	64.62	77.60	0.708	5.32	0.097	0.474	**65.93**	**78.68**	0.759	2.70	0.061
Ours	**0.476**	**64.70**	**77.75**	**0.741**	9.37	0.168	**0.461**	65.44	78.60	**0.797**	6.29	0.122

TABLE II

SINGLE-IMAGE TO 3D RECONSTRUCTION RESULTS. WE SHOW CHAMFER DISTANCE, F1-SCORE, AND NORMAL CONSISTENCY TO EVALUATE THE ACCURACY OF THE PREDICTIONS. WE REPORT SELF-INTERSECTION LENGTH AND RATIO TO EVALUATE THE MANIFOLDNESS OF THE FINAL MESHES.

Category	P2M			P2M, 5 stages			MeshRCNN			Ours		
	$F1^{\tau}(\uparrow)$	Normal consist.	S. I. ratio	$F1^{\tau}(\uparrow)$	Normal consist.	S. I. ratio	$F1^{\tau}(\uparrow)$	Normal consist.	S. I. ratio	$F1^{\tau}(\uparrow)$	Normal consist.	S. I. ratio
bench	59.47	0.742	0.22	60.42	0.763	**0.045**	64.89	0.736	0.066	**73.87**	**0.805**	0.088
chair	53.57	0.765	0.225	55.96	0.795	**0.036**	59.63	0.746	0.053	**68.03**	**0.820**	0.093
lamp	57.10	0.685	0.454	58.42	0.715	0.214	61.48	0.690	**0.109**	**67.79**	**0.730**	0.342
speaker	52.52	0.830	0.147	52.49	0.846	**0.021**	59.18	0.776	0.035	**65.28**	**0.866**	0.046
firearm	60.31	0.641	0.392	62.16	0.653	0.142	61.69	0.635	**0.079**	**67.36**	**0.678**	0.269
table	68.29	0.774	0.169	69.32	0.796	**0.043**	74.03	0.775	0.073	**79.32**	**0.812**	0.126
watercraft	55.53	0.698	0.347	58.44	0.715	0.075	57.2	0.683	**0.068**	**65.04**	**0.756**	0.121
plane	64.44	0.687	0.465	66.48	0.704	0.119	**65.85**	0.675	**0.096**	64.23	**0.732**	0.258
cabinet	62.78	0.825	0.155	62.46	0.840	**0.011**	**70.82**	0.796	0.033	67.59	**0.866**	0.028
car	66.25	0.824	0.159	65.81	**0.845**	**0.028**	72.42	0.779	0.033	**76.48**	0.843	0.035
monitor	54.46	0.814	0.246	55.05	0.839	**0.029**	57.98	0.772	0.048	**66.80**	**0.850**	0.058
couch	55.21	0.806	0.155	55.72	0.825	**0.015**	58.61	0.777	0.033	**69.46**	**0.858**	0.024
cellphone	66.15	0.866	0.244	68.98	0.897	**0.009**	72.18	0.843	0.04	**82.29**	**0.915**	0.027
mean	60.94	0.775	0.234	62.16	0.798	**0.050**	65.93	0.759	0.061	**73.44**	**0.820**	0.112

TABLE III

RECONSTRUCTION RESULTS PER CATEGORY ON THE ZERO-HOLE SHAPES FROM SHAPENET DATASET. WE REPORT PER-CATEGORY AVERAGE OF EVERY F1-SCORE, NORMAL CONSISTENCY, AND THE SELF-INTERSECTION RATIO METRICS.

extended Pixel2Mesh, which carries on to the last stage. The low amount of self-intersections occurs at the expense of the reconstruction accuracy because Pixel2Mesh assumes that the resulting mesh is already a good approximation of the target shape after the first deformation stage. A good approximation of the target shape will not likely be the case with such low resolution. We therefore conclude that our method achieves better reconstruction performance than all baselines while maintaining a relatively low self-intersection ratio.

Tab. III shows per-category reconstruction metrics. In this experiment, we only use shapes with no holes. We report the average F1-score, normal consistency, and self-intersection ratio per class. We bring attention to several categories, such as cabinet, couch, or monitor, where the self-intersection occurs – on average – in less than 10% of the faces. Additionally, we outperform Pixel2Mesh in all metrics and achieve competitive results compared with MeshRCNN. In all cases, the normal consistency was superior to Pixel2Mesh with no modifications and MeshRCNN. Note that our approaches do not use any normal consistency regularization. Thus, we can attribute the higher accuracy of this metric to the smooth deformation through the level sets.

We present qualitative results in Fig. 2. Our results show an increase in accuracy to approximate the ground truth shape while maintaining a relatively low amount of self-intersecting faces.

Additionally, we include Fig. 3 to visualize how our method progressively deforms the vertices from the initial spherical mesh through all the stages in our model. The figure shows how the predicted meshes evolve while adopting the shape of the level sets at each step. The meshes get closer to the target shape following a smooth transformation.

V. CONCLUSIONS

We presented Level Set Mesher, a single-image to 3D reconstruction method that learns to predict 3D shapes by progressively deforming an initial surface constrained to follow the target shape's SDF level sets. We modeled our method as a Multi-head Attention Network, which allowed us to compute the deformation at each vertex as a weighted combination of features in neighbor vertices. Our strategy proved to be effective for 3D reconstruction by improving the accuracy of the predicted shapes while also capturing fine details and reducing artifacts such as self-intersections.

VI. ACKNOWLEDGMENTS

The financial support by the following grants is gratefully acknowledged: NSF TRIPODS 1934960, ARL DCIST CRA W911NF-17-2-0181, ARO MURI W911NF-20-1-0080, and ONR N00014-17-1-2093.

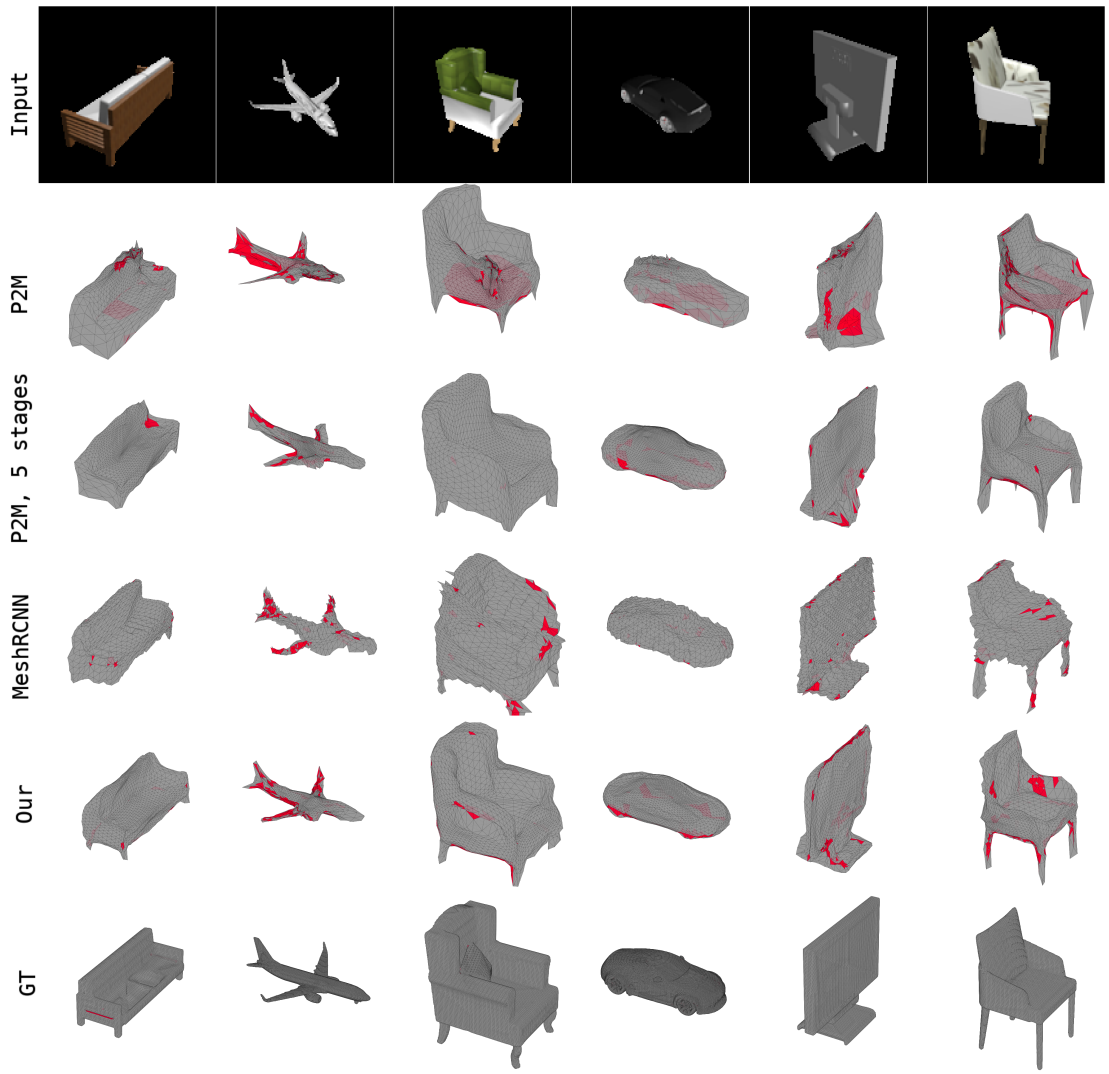

Fig. 2. Qualitative results of the final predictions in the test set. The predicted meshes were rendered highlighting the self-intersecting faces in red. Our approach achieves greater accuracy w.r.t the ground truth meshes with a relatively low ratio of self-intersections.

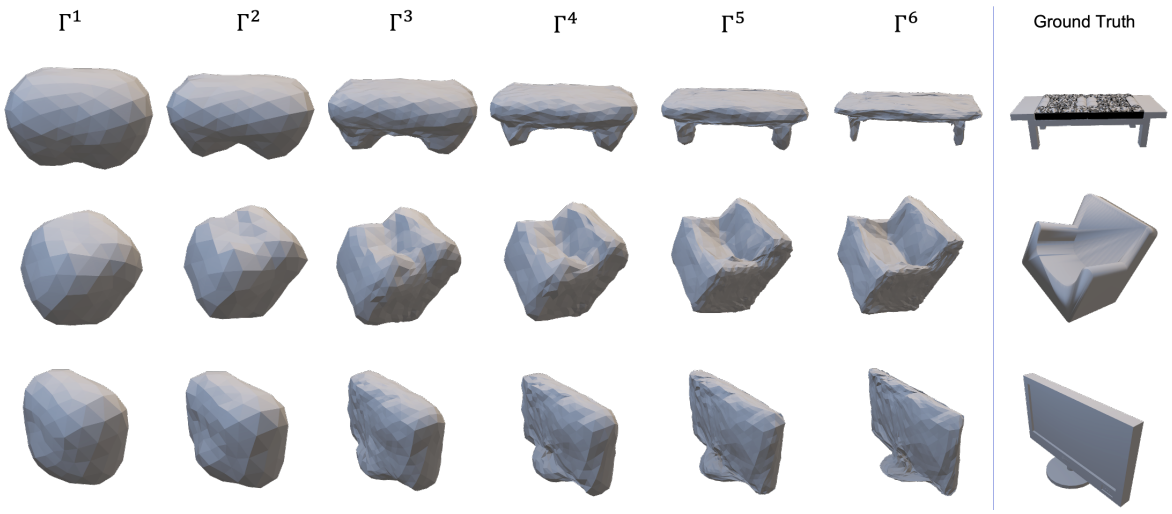

Fig. 3. Evolution of the predicted meshes (test set) until the predictions match the target shape (right). Note how our approach follows a smooth deformation such that self-intersections only appear in the last stages.

REFERENCES

[1] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman, "Pix3d: Dataset and methods for single-image 3d shape modeling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[2] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1912–1920.

[3] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," *ArXiv*, vol. abs/1604.00449, 2016.

[4] C. Häne, S. Tulsiani, and J. Malik, "Hierarchical surface prediction for 3d object reconstruction," *2017 International Conference on 3D Vision (3DV)*, pp. 412–420, 2017.

[5] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2107–2115, 2017.

[6] L. M. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4455–4465, 2019.

[7] S. Saito, Z. Huang, R. Natsume, S. Morishima, A. Kanazawa, and H. Li, "Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2304–2314, 2019.

[8] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," in *ECCV*, 2018.

[9] C. Wen, Y. Zhang, Z. Li, and Y. Fu, "Pixel2mesh++: Multi-view 3d mesh generation via deformation," *ArXiv*, vol. abs/1908.01491, 2019.

[10] E. Smith, S. Fujimoto, A. Romero, and D. Meger, "Geometrics: Exploiting geometric structure for graph-encoded objects," in *ICML*, 2019.

[11] G. Gkioxari, J. Malik, and J. Johnson, "Mesh r-cnn," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9784–9794, 2019.

[12] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. B. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," *ArXiv*, vol. abs/1610.07584, 2016.

[13] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *ECCV*, 2016.

[14] D. Stutz and A. Geiger, "Learning 3d shape completion under weak supervision," *International Journal of Computer Vision*, vol. 128, pp. 1162–1181, 2018.

[15] H. Xie, H. Yao, X. Sun, S. Zhou, S. Zhang, and X. Tong, "Pix2vox: Context-aware 3d reconstruction from single and multi-view images," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2690–2698, 2019.

[16] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Multi-view 3d models from single images with a convolutional network," in *ECCV*, 2016.

[17] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2463–2471, 2017.

[18] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 77–85, 2017.

[19] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *NIPS*, 2017.

[20] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "Pcn: Point completion network," *2018 International Conference on 3D Vision (3DV)*, pp. 728–737, 2018.

[21] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5932–5941, 2019.

[22] K. Genova, F. Cole, D. Vlasic, A. Sarna, W. T. Freeman, and T. A. Funkhouser, "Learning shape templates with structured implicit functions," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7153–7163, 2019.

[23] M. Michalkiewicz, J. K. Pontes, D. Jack, M. Baktashmotlagh, and A. Eriksson, "Implicit surface representations as layers in neural networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[24] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Occupancy flow: 4d reconstruction by learning particle dynamics," in *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019.

[25] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann, "Disn: Deep implicit surface network for high-quality single-view 3d reconstruction," in *NeurIPS*, 2019.

[26] M. Niemeyer, L. M. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision," *ArXiv*, vol. abs/1912.07372, 2019.

[27] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mache approach to learning 3d surface generation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 216–224, 2018.

[28] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia, "Deep mesh reconstruction from single rgb images via topology modification networks," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9963–9972, 2019.

[29] S. Liu, W. Chen, T. Li, and H. Li, "Soft rasterizer: Differentiable rendering for unsupervised single-view mesh reconstruction," *ArXiv*, vol. abs/1901.05567, 2019.

[30] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask r-cnn," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

[31] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio', and Y. Bengio, "Graph attention networks," *ArXiv*, vol. abs/1710.10903, 2018.

[32] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, pp. 18–42, 2017.

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[34] J. Peters and U. Reif, "The simplest subdivision scheme for smoothing polyhedra," *ACM Trans. Graph.*, vol. 16, no. 4, p. 420–431, Oct. 1997. [Online]. Available: https://doi.org/10.1145/263834.263851

[35] A. Habib and J. Warren, "Edge and vertex insertion for a class of c1 subdivision surfaces," *Computer Aided Geometric Design*, vol. 16, no. 4, pp. 223 – 247, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167839698000454

[36] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q.-X. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "Shapenet: An information-rich 3d model repository," *ArXiv*, vol. abs/1512.03012, 2015.

[37] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

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

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.

Level Set Mesher: Single-image to 3D reconstruction by following the level sets of the signed distance function: Supplementary Material

Diego Patiño*, Carlos Esteves† and Kostas Daniilidis‡
GRASP Laboratory, University of Pennsylvania
Email: *diegopc@seas.upenn.edu,
†machc@seas.upenn.edu, ‡kostas@cis.upenn.edu

I. Ablation Study

In this ablation study, we compare the results of our full model with a version wherein we replace the multi-head graph attention network with a regular graph convolution of the same capacity. We show the results in Tab. IV.

II. Level Set Sampling

We follow an Octree-based strategy to sample points on the l-level set of a shape's SDF. First, we sample points on a regular grid where each grid cell has a side of $s_0 = 0.75$. Recall that we use level sets inside of a cube of side 3 surrounding the shape. We sample the SDF at all grid cell centers c. If $|c - l| <= \sqrt{2s_0^2}$, we conclude that cell c potentially contains a level-set point, otherwise we discard the cell. If this condition is true, we proceed to subdivide the cell into eight new ones and apply the condition again for $s_k = \frac{1}{2}s_{k-1}$. The algorithm stops when $s_k < \epsilon$ for a user-defined ϵ. We depict the process in Fig. 4.

III. Face subdivision strategy

In our architecture, we split each face in each mesh into four smaller faces during steps 2 and 4. We do the subdivision using a Mid-edge subdivision scheme [34], [35]. We split each face into four different faces by creating new vertices at the center of each edge. New edges form by connecting the new vertices and keeping both halves of the original edges. The process is depicted in Fig. 5.

IV. Input meshes

In Tab I, we reported the input and output mesh sizes for all methods in the study. Note that we use the same input and output size in our method as in Pixel2Mesh. Additionally, note that we started from a coarser mesh for the Pixel2Mesh with five stages comparison. In Fig. 6, we show an example of the initial meshes as a way to visualize the resolution at different stages of the deformation. We show average values for MeshRCNN because the output mesh's size depends on applying the marching cubes algorithm on the 48x48 voxelized initial prediction. There is no face subdivision in the MeshRCNN architecture.

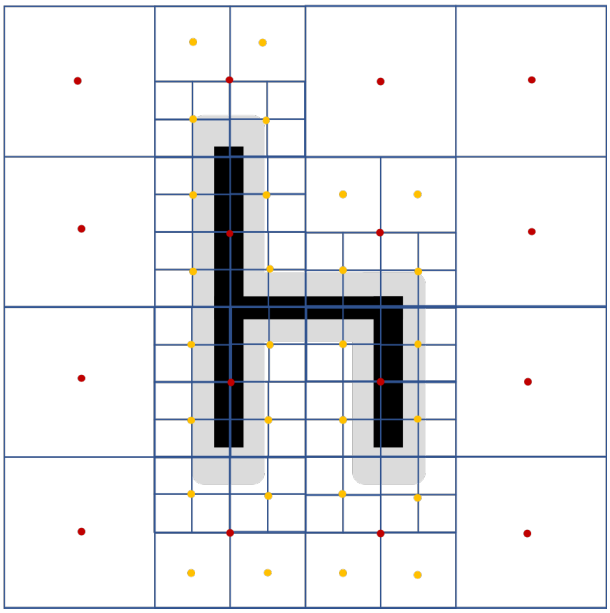

Fig. 4. Level-set sampling strategy. We progressively sample points on the level-sets by using an Octree-based strategy. Red dots are sampled points in the first iteration. Yellow dots are points sampled in the second iterations, just on cells that potentially contain level-set points.

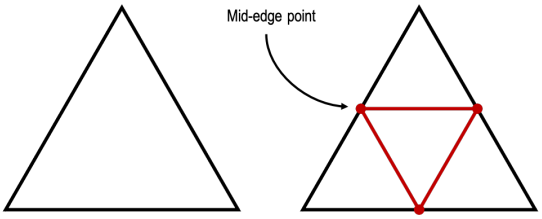

Fig. 5. Subdivision strategy. Every faces split in four through new vertices inserted at mid-edge points.

V. Number of level sets

Note that our method converges to a continuous deformation following the SDF gradient field if we use infinite intermediate layers and infinite level sets in the loss function. However, such scenario is impossible to model. Therefore, it is necessary to follow discrete steps to learn a model that approximates the flow. We chose seven different level sets in our study due to empirical criteria:

1) We desire to maximize the number of layers to create a model deep enough to learn the problem's complex patterns. We believe that more layers (and thus more level sets) are preferable to approximate the ground truth V.
2) Computational resources constrain us given the known memory limitation of Graph Neural Networks. This

	Full Test Set						No Holes Test Set					
	CD(↓)	$F1^{\tau}(\uparrow)$	$F1^{2\tau}(\uparrow)$	Normal consist.	Self. inter. length	Self. inter. ratio	CD(↓)	$F1^{\tau}(\uparrow)$	$F1^{2\tau}(\uparrow)$	Normal consist.	Self. inter. length	Self. inter. ratio
Full model	0.368	71.26	82.82	0.761	10.83	0.149	0.353	73.44	84.34	0.820	7.23	0.112
No MGAN	0.369	70.45	82.56	0.77	10.99	0.180	0.3613	71.56	83.52	0.825	8.26	0.143

TABLE IV
ABLATION STUDY TO SHOW THE CONTRIBUTION OF THE DIFFERENT ELEMENTS OF OUR NETWORK.

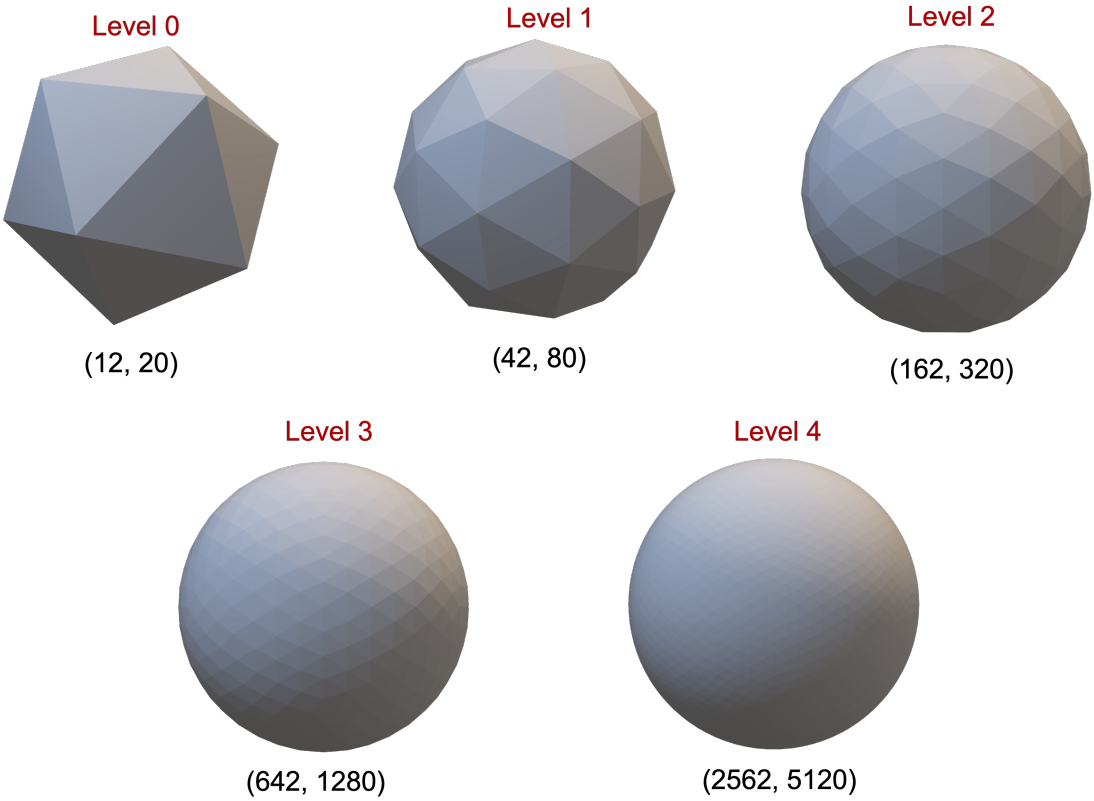

Fig. 6. Illustrative example describing the size and quality of the input meshes. All the meshes derive from an icosahedron whose vertices have been projected to the surfaces of a unit sphere. We use the face subdivision strategy to produce the input meshes in subsequent levels.

limitation prevents us from predicting meshes with a large number of vertices, and prevents us from to using a large number of level sets.

3) We chose a number of stages that could represent the critical level sets where the vector field starts "curving" to reveal more fine details.

Consequently, we have placed the level sets at exponentially at intervals of decreasing length. This was done because outer level sets only hold general cues on the shape, whereas the inner level sets reveal more refined details.

VI. Evolution videos

We provide a video of a ShapeNet model showing the evolution of the mesh through the deformation process. This video highlights the need for a vector field-guided mesh deformation in our level set formulations. In the video, we show our method in comparison with Pixel2Mesh. Note that a) our method smoothly deforms the initial mesh until it converges to the surface of the 3D shape, and b) we avoid self-intersections and topology artifacts in the final predictions. We have provided the video as an attachment of this supplementary material.

VII. Self-intersection metrics

This section provides the mathematical definition of the two metrics we use to assess the quality/manifoldness of the predicted meshes: Self-intersection length and self-intersection ratio.

Recall that two planes in 3D intersect in a straight line. Consider now two faces F_i and F_j from the same triangular mesh with face normals $n_i, n_j \in \mathbb{R}^3$ respectively. We can find the line Q spanned by the intersection of the two faces' planes by solving

$$\begin{bmatrix} n_i^{\intercal} & -n_i^{\intercal} \cdot P_i \\ n_j^{\intercal} & -n_j^{\intercal} \cdot P_j \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{9}$$

where P_i is any point inside triangle i (including the vertices). Note that Eq. 9 has one degree of freedom which corresponds with the free parameter of Q. We define the self-intersection length as the segment of Q that is simultaneously contained within the two triangles. We define the self-intersection rate as the ratio of intersecting faces over the total number of faces in the mesh.