Analytic Marching: An Analytic Meshing Solution from Deep Implicit Surface Networks

Jiabao Lei [1 2] **Kui Jia** [1 2]

Abstract

This paper studies a problem of learning surface mesh via implicit functions in an emerging field of deep learning surface reconstruction, where implicit functions are popularly implemented as multi-layer perceptrons (MLPs) with rectified linear units (ReLU). To achieve meshing from learned implicit functions, existing methods adopt the de-facto standard algorithm of marching cubes; while promising, they suffer from loss of precision learned in the MLPs, due to the discretization nature of marching cubes. Motivated by the knowledge that a ReLU based MLP partitions its input space into a number of linear regions, we identify from these regions *analytic cells* and *analytic faces* that are associated with zero-level isosurface of the implicit function, and characterize the theoretical conditions under which the identified analytic faces are guaranteed to connect and form a *closed, piecewise planar surface*. Based on our theorem, we propose a naturally parallelizable algorithm of *analytic marching*, which marches among analytic cells to *exactly* recover the mesh captured by a learned MLP. Experiments on deep learning mesh reconstruction verify the advantages of our algorithm over existing ones.

1. Introduction

This paper studies a geometric notion of object surface whose nature is a 2-dimensional manifold embedded in the 3D space. In literature, there exist many different ways to represent an object surface, either explicitly or implicitly (Botsch et al., 2010). For example, one of the most popular representations is *polygonal mesh* that approximates a smooth surface as piecewise linear functions. Mesh representation of object surface plays fundamental roles in many applications of computer graphics and geometry processing, e.g., computer-aided design, movie production, and virtual/augmented reality.

As a parametric representation of object surface, the most typical *triangle mesh* is explicitly defined as a collection of connected faces, each of which has three vertices that uniquely determine plane parameters of the face in the 3D space. However, parametric surface representations are usually difficult to obtain, especially for topologically complex surface; queries of points inside or outside the surface are expensive as well. Instead, one usually resorts to implicit functions (e.g., signed distance function or SDF (Curless & Levoy, 1996; Park et al., 2019)), which subsume the surface as zero-level isosurface in the function field; other implicit representations include discrete volumes and those based on algebraic (Blinn, 1982; Nishimura et al., 1985; Wyvill et al., 1986) and radial basis functions (Carr et al., 2001; 1997; Turk & O'Brien, 1999). To obtain a surface mesh, the continuous field function is often discretized around the object as a regular grid of voxels, followed by the de-facto standard algorithm of marching cubes (Lorensen & Cline, 1987). Efficiency and result regularity of marching cubes can be improved on a hierarchically sampled structure of octree via algorithms such as dual contouring (Ju et al., 2002).

The most popular implicit function of SDF is traditionally implemented discretely as a regular grid of voxels. More recently, methods of deep learning surface reconstruction (Park et al., 2019; Xu et al., 2019) propose to use deep models of Multi-Layer Perceptron (MLP) to learn continuous SDFs; given a learned SDF, they typically take a final step of marching cubes to obtain the mesh reconstruction results. While promising, the final step of marching cubes recovers a mesh that is only an approximation of the surface captured by the learned SDF; more specifically, it suffers from a trade-off of efficiency and precision, due to the discretization nature of the marching cubes algorithm.

Motivated by the established knowledge that an MLP based

[1]School of Electronic and Information Engineering, South China University of Technology, Guangzhou, Guangdong, China [2]Pazhou Lab, Guangzhou, 510335, China. Correspondence to: Kui Jia <kuijia@scut.edu.cn>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

on rectified linear units (ReLU) (Glorot et al., 2011) partitions its input space into a number of linear regions (Montúfar et al., 2014), we identify from these regions *analytic cells* and *analytic faces* that are associated with zero-level isosurface of the MLP based SDF. Assuming that such a SDF learns its zero-level isosurface as a *closed, piecewise planar surface*, we characterize theoretical conditions under which analytic faces of the SDF *connect and exactly form* the surface mesh. Based on our theorem, we propose an algorithm of *analytic marching*, which marches among analytic cells to recover the *exact mesh* of the closed, piecewise planar surface captured by a learned MLP. Our algorithm can be naturally implemented in parallel. We present careful ablation studies in the context of deep learning mesh reconstruction. Experiments on benchmark datasets of 3D object repositories show the advantages of our algorithm over existing ones, particularly in terms of a better trade-off of precision and efficiency.

2. Related works

Implicit surface representation To represent an object surface implicitly, some of previous methods take a strategy of divide and conquer that represents the surface using atom functions. For example, blobby molecule (Blinn, 1982) is proposed to approximate each atom by a gaussian potential, and a piecewise quadratic meta-ball (Nishimura et al., 1985) is used to approximate the gaussian, which is improved via a soft object model in (Wyvill et al., 1986) by using a sixth degree polynomial. Radial basis function (RBF) is an alternative to the above algebraic functions. RBF-based approaches (Carr et al., 2001; 1997; Turk & O'Brien, 1999) place the function centers near the surface and are able to reconstruct a surface from a discrete point cloud, where common choices of basis function include thin-plate spline, gaussian, multiquadric, and biharmonic/triharmonic splines.

Methods of mesh conversion The conversion from an implicit representation to an explicit surface mesh is called isosurface extraction. Probably the simplest approach is to directly convert a volume via greedy meshing (GM). The defacto standard algorithm of marching cubes (MC) (Lorensen & Cline, 1987) builds from the implicit function a discrete volume around the object, and computes mesh vertices on the edges of the volume; due to its discretization nature, mesh results of the algorithm are often short of sharp surface details. Algorithms similar to MC include marching tetrahedra (MT) (Doi & Koide, 1991) and dual contouring (DC) (Ju et al., 2002). In particular, MT divides a voxel into six tetrahedrons and calculates the vertices on edges of each tetrahedron; DC utilizes gradients to estimate positions of vertices in a cell and extracts meshes from adaptive octrees. All these methods suffer from a trade-off of precision and efficiency due to the necessity to sample discrete points from the 3D space.

Local linearity of MLPs Among the research studying representational complexities of deep networks, Montúfar et al. (2014) and Pascanu et al. (2014) investigate how a ReLU or maxout based MLP partitions its input space into a number of linear regions, and bound this number via quantities relevant to network depth and width. The region-wise linear mapping is explicitly established in (Jia et al., 2019) in order to analyze generalization properties of deep networks. A closed-form solution termed OpenBox is proposed in (Chu et al., 2018) that computes consistent and exact interpretations for piecewise linear deep networks. The present work leverages the locally linear properties of MLP networks and studies how zero-level isosurface can be identified from MLP based SDFs.

3. Analytic meshing via deep implicit surface networks

We start this section with the introduction of Multi-Layer Perceptrons (MLPs) based on rectified linear units (ReLU) (Glorot et al., 2011), and discuss how such an MLP as a nonlinear function partitions its input space into linear regions via a compositional structure.

3.1. Local linearity of multi-layer perceptions

An MLP of L hidden layers takes an input $\boldsymbol{x} \in \mathbb{R}^{n_0}$ from the space $\mathcal{X}$, and layer-wisely computes $\boldsymbol{x}_l = \boldsymbol{g}(\boldsymbol{W}_l \boldsymbol{x}_{l-1})$, where $l \in \{1, \dots, L\}$ indexes the layer, $\boldsymbol{x}_l \in \mathbb{R}^{n_l}$, $\boldsymbol{x}_0 = \boldsymbol{x}$, $\boldsymbol{W}_l \in \mathbb{R}^{n_l \times n_{l-1}}$, and $\boldsymbol{g}$ is the point-wise, ReLU based activation function. We also denote the intermediate feature space $g(\boldsymbol{W}_l \boldsymbol{x}_{l-1})$ as $\mathcal{X}_l$ and $\mathcal{X}_0 = \mathcal{X}$. We compactly write the MLP as a mapping $\boldsymbol{T}\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{W}_L \dots \boldsymbol{g}(\boldsymbol{W}_1 \boldsymbol{x}))$. Any k^{th} neuron, $k \in \{1, \dots, n_l\}$, of an l^{th} layer of the MLP $\boldsymbol{T}$ specifies a functional defined as

$$a_{lk}(\boldsymbol{x}) = \pi_k \boldsymbol{g}(\boldsymbol{W}_l \boldsymbol{g}(\boldsymbol{W}_{l-1} \dots \boldsymbol{g}(\boldsymbol{W}_1 \boldsymbol{x}))),$$

where π_k denotes an operator that projects onto the k^{th} coordinate. All the neurons at layer l define a functional as

$$\boldsymbol{a}_l(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{W}_l \boldsymbol{g}(\boldsymbol{W}_{l-1} \dots \boldsymbol{g}(\boldsymbol{W}_1 \boldsymbol{x}))).$$

We define the support of $\boldsymbol{T}$ as

$$\text{supp}(\boldsymbol{T}) = \{\boldsymbol{x} \in \mathcal{X} | \boldsymbol{T}\boldsymbol{x} \neq \boldsymbol{0}\}, \qquad (1)$$

which are instances of practical interest in the input space. Support $\text{supp}(a_{lk})$ of any neuron a_{lk} is similarly defined.

For an intermediate feature space $\mathcal{X}_{l-1} \in \mathbb{R}^{n_{l-1}}$, each hidden neuron of layer l specifies a hyperplane H that partitions $\mathcal{X}_{l-1}$ into two halves, and the collection of hyperplanes $\{H_i\}_{i=1}^{n_l}$ specified by all the n_l neurons of layer l form a *hyperplane arrangement* (Orlik & Terao, 1992). These hyperplanes partition the space $\mathcal{X}_{l-1}$ into multiple linear regions whose formal definition is as follows.

Definition 1 (Region/Cell)**.** *Let $\mathcal{A}$ be an arrangement of hyperplanes in $\mathbb{R}^m$. A region of the arrangement is a connected component of the complement $\mathbb{R}^m - \bigcup_{H \in \mathcal{A}} H$. A region is a cell when it is bounded.*

Classical result from (Zaslavsky, 1975; Pascanu et al., 2014) tells that the arrangement of n_l hyperplanes gives at most $\sum_{j=0}^{n_{l-1}} \binom{n_l}{j}$ regions in $\mathbb{R}^{n_{l-1}}$. Given fixed $\{\boldsymbol{W}_l\}_{l=1}^L$, the MLP $\boldsymbol{T}$ partitions the input space $\mathcal{X} \in \mathbb{R}^{n_0}$ by its layers' recursive partitioning of intermediate feature spaces, which can be intuitively understood as a successive process of space folding (Montúfar et al., 2014).

Let $\mathcal{R}(\boldsymbol{T})$, shortened as $\mathcal{R}$, denote the set of all linear regions/cells in $\mathbb{R}^{n_0}$ that are possibly achieved by $\boldsymbol{T}$. To have a concept on the maximal size of $\mathcal{R}$, we introduce the following functionals about activation states of neuron, layer, and the whole MLP.

Definition 2 (State of Neuron/MLP)**.** *For a k^{th} neuron of an l^{th} layer of an MLP $\boldsymbol{T}$, with $k \in \{1, \dots, n_l\}$ and $l \in \{1, \dots, L\}$, its state functional of neuron activation is defined as*

$$s_{lk}(\boldsymbol{x}) = \begin{cases} 1 & \text{if } a_{lk}(\boldsymbol{x}) > 0 \\ 0 & \text{if } a_{lk}(\boldsymbol{x}) \leq 0, \end{cases} \tag{2}$$

which gives the state functional of layer l as

$$\boldsymbol{s}_l(\boldsymbol{x}) = [s_{l1}(\boldsymbol{x}), \dots, s_{ln_l}(\boldsymbol{x})]^\top, \tag{3}$$

and the state functional of MLP $\boldsymbol{T}$ as

$$\boldsymbol{s}(\boldsymbol{x}) = [\boldsymbol{s}_1(\boldsymbol{x})^\top, \dots, \boldsymbol{s}_L(\boldsymbol{x})^\top]^\top. \tag{4}$$

Let the total number of hidden neurons in $\boldsymbol{T}$ be $N = \sum_{l=1}^L n_l$. Denote $\mathbb{J} = \{1, 0\}$, and we have the state functional $\boldsymbol{s} \in \mathbb{J}^N$. Considering that a region in $\mathbb{R}^{n_0}$ corresponds to a realization of $\boldsymbol{s} \in \mathbb{J}^N$, it is clear that the maximal size of $\mathcal{R}$ is upper bounded by 2^N. This gives us the following labeling scheme: for any region $r \in \mathcal{R}$, it corresponds to a unique element in $\mathbb{J}^N$; since $\boldsymbol{s}(\boldsymbol{x})$ is fixed for all $\boldsymbol{x} \in \mathcal{X}$ that fall in a same region r, we use $\boldsymbol{s}(r) \in \mathbb{J}^N$ to label this region. Results from (Montúfar et al., 2014) tell that when layer widths of $\boldsymbol{T}$ satisfy $n_l \geq n_0$ for any $l \in \{1, \dots, L\}$, the maximal size of $\mathcal{R}(\boldsymbol{T})$ is lower bounded by $\left(\prod_{l=1}^{L-1} \lfloor n_l / n_0 \rfloor^{n_0}\right) \sum_{j=0}^{n_0} \binom{n_L}{j}$, where $\lfloor \cdot \rfloor$ ignores the remainder. Assuming $n_1 = \cdots = n_L = n$, the lower bound has an order of $\mathcal{O}\left((n/n_0)^{(L-1)n_0} n^{n_0}\right)$, which grows exponentially with the network depth and polynomially with the network width. We have the following lemma adapted from (Jia et al., 2019) to characterize the region-wise linear mappings.

Lemma 3 (Linear Mapping of Region/Cell, an adaptation of Lemma 3.2 in (Jia et al., 2019))**.** *Given a ReLU based MLP $\boldsymbol{T}$ of L hidden layers, for any region/cell $r \in \mathcal{R}(\boldsymbol{T})$, its associated linear mapping $\boldsymbol{T}^r$ is defined as*

$$\boldsymbol{T}^r = \prod_{l=1}^L \boldsymbol{W}_l^r \tag{5}$$

$$\boldsymbol{W}_l^r = diag(\boldsymbol{s}_l(r)) \boldsymbol{W}_l, \tag{6}$$

where $diag(\cdot)$ diagonalizes the state vector $\boldsymbol{s}_l(r)$.

Intuitively, the state vector $\boldsymbol{s}_l$ in (6) selects a submatrix from $\boldsymbol{W}_l$ by setting those inactive rows as zero.

3.2. Analytic cells and faces associated with the zero level isosurface of an implicit field function

Let $F : \mathbb{R}^3 \to \mathbb{R}$ denote a scalar-valued, implicit field of signed distance function (SDF). Given F, an object surface $\mathcal{Z}$ is formally defined as its zero-level isosurface, i.e., $\mathcal{Z} = \{\boldsymbol{x} \in \mathbb{R}^3 | F(\boldsymbol{x}) = 0\}$. We also have the distance $F(\boldsymbol{x}) > 0$ for points inside $\mathcal{Z}$ and $F(\boldsymbol{x}) < 0$ for those outside. While F can be realized using radial basis functions (Carr et al., 2001; 1997; Turk & O'Brien, 1999) or be approximated as a regular grid of voxels (i.e., a volume), in this work, we are particularly interested in implementing F using ReLU based MLPs, which become an increasingly popular choice in recent works of deep learning surface reconstruction (Park et al., 2019; Xu et al., 2019).

To implement F using an MLP $\boldsymbol{T}$, we stack on top of $\boldsymbol{T}$ a regression function $f : \mathbb{R}^{n_L} \to \mathbb{R}$, giving rise to a functional of SDF as

$$F(\boldsymbol{x}) = f \circ \boldsymbol{T}(\boldsymbol{x}) = \boldsymbol{w}_f^\top \boldsymbol{g}(\boldsymbol{W}_L \dots \boldsymbol{g}(\boldsymbol{W}_1 \boldsymbol{x})),$$

where $\boldsymbol{w}_f \in \mathbb{R}^{n_L}$ is weight vector of the regressor. Since F represents a field function in the 3D Euclidean space, we have $n_0 = 3$. Analysis in Section 3.1 tells that the MLP $\boldsymbol{T}$ partitions the input space $\mathbb{R}^3$ into a set $\mathcal{R}$ of linear regions; any region $r \in \mathcal{R}$ that satisfies $\boldsymbol{x} \in \text{supp}(\boldsymbol{T}) \ \forall\ \boldsymbol{x} \in r$ can be uniquely indexed by its state vector $\boldsymbol{s}(r)$ defined by (4). For such a region r, we have the following corollary from Lemma 3 that characterizes the associated linear mappings defined at neurons of $\boldsymbol{T}$ and the final regressor.

Corollary 4. *Given a SDF $F = f \circ \boldsymbol{T}$ built on a ReLU based MLP of L hidden layers, for any $r \in \mathcal{R}(\boldsymbol{T})$, the associated neuron-wise linear mappings and that of the final regressor are defined as*

$$\boldsymbol{a}_{lk}^r = \pi_k \prod_{i=1}^l \boldsymbol{W}_i^r \tag{7}$$

$$\boldsymbol{a}_F^r = \boldsymbol{w}_f^\top \boldsymbol{T}^r = \boldsymbol{w}_f^\top \prod_{i=1}^L \boldsymbol{W}_i^r, \tag{8}$$

where $l \in \{1, \dots, L\}$, $k \in \{1, \dots, n_l\}$, and $\boldsymbol{T}^r$ and $\boldsymbol{W}_i^r$ are defined as in Lemma 3.

Corollary 4 tells that the SDF F in fact induces a set of region-associated planes in $\mathbb{R}^3$. The plane $\{\boldsymbol{x} \in \mathbb{R}^3 | \boldsymbol{a}_F^r \boldsymbol{x} = 0\}$ and the associated region r have the following relations, assuming that they are in general positions. In the following, we use the normal $\boldsymbol{a}_F^r$ to represent the plane for simplicity.

- *Intersection* $\boldsymbol{a}_F^r$ splits the region r into two halves, denoted as r^+ and r^-, such that $\forall \boldsymbol{x} \in r^+$, we have $\boldsymbol{a}_F^r \boldsymbol{x} > 0$ and $\forall \boldsymbol{x} \in r^-$, we have $\boldsymbol{a}_F^r \boldsymbol{x} \leq 0$.
- *Non-intersection* We either have $\boldsymbol{a}_F^r \boldsymbol{x} > 0$ or $\boldsymbol{a}_F^r \boldsymbol{x} < 0$ for all $\boldsymbol{x} \in r$.

Let $\{\tilde{r} \in \widetilde{\mathcal{R}}\}$ denote the subset of regions in $\mathcal{R}$ that have the above relation of intersection, an illustration of which is given in Figure 1. It is clear that the zero-level isosurface $\mathcal{Z} = \{\boldsymbol{x} \in \mathbb{R}^3 | F(\boldsymbol{x}) = 0\}$ defined on the support (1) of $\boldsymbol{T}$ can be only in $\widetilde{\mathcal{R}}$. To have an analytic understanding on any $\tilde{r} \in \widetilde{\mathcal{R}}$, we note from Corollary 4 that the boundary planes of $\tilde{r}$ must be among the set

$$\{H_{lk}^{\tilde{r}}\} \text{ s.t. } H_{lk}^{\tilde{r}} = \{\boldsymbol{x} \in \mathbb{R}^3 | \boldsymbol{a}_{lk}^{\tilde{r}} \boldsymbol{x} = 0\}, \tag{9}$$

where $l = 1, \dots, L$ and $k = 1, \dots, n_l$; for any $\boldsymbol{x} \in \tilde{r}$, it must satisfy $\text{sign}(\boldsymbol{a}_{lk}^{\tilde{r}} \boldsymbol{x}) = (2 s_{lk}(\tilde{r}) - 1)$, which gives the following system of inequalities

$$(\boldsymbol{I} - 2\text{diag}(\boldsymbol{s}(\tilde{r}))\boldsymbol{A}^{\tilde{r}} \boldsymbol{x} = \begin{bmatrix} (1 - 2s_{11}(\tilde{r}))\boldsymbol{a}_{11}^{\tilde{r}} \\ \vdots \\ (1 - 2s_{lk}(\tilde{r}))\boldsymbol{a}_{lk}^{\tilde{r}} \\ \vdots \\ (1 - 2s_{Ln_L}(\tilde{r}))\boldsymbol{a}_{Ln_L}^{\tilde{r}} \end{bmatrix} \boldsymbol{x} \preceq 0, \tag{10}$$

where $\boldsymbol{I}$ is an identity matrix of compatible size, $\boldsymbol{A}^{\tilde{r}} \in \mathbb{R}^{Ln_L \times 3}$ collects the coefficients of the inequalities, and the state functionals s_{lk} and $\boldsymbol{s}$ are defined by (2) and (4). We note that for some cases of region $\tilde{r}$, there could exist redundance in the inequalities of (10). When the region is bounded, the system (10) of inequalities essentially forms a polyhedral cell defined as

$$\mathcal{C}_F^{\tilde{r}} = \{\boldsymbol{x} \in \mathbb{R}^3 | (\boldsymbol{I} - 2\text{diag}(\boldsymbol{s}(\tilde{r}))\boldsymbol{A}^{\tilde{r}} \boldsymbol{x} \preceq 0\}, \tag{11}$$

which we term as *analytic cell of a SDF's zero-level isosurface*, shortened as *analytic cell*. We note that an analytic cell could also be a region open towards infinity in some directions.

Given the plane functional (8), we define the polygonal face that is an intersection of analytic cell $\tilde{r}$ and surface $\mathcal{Z}$ as

$$\mathcal{P}_F^{\tilde{r}} = \{\boldsymbol{x} \in \mathbb{R}^3 | \boldsymbol{a}_F^{\tilde{r}} \boldsymbol{x} = 0, (\boldsymbol{I} - 2\text{diag}(\boldsymbol{s}(\tilde{r}))\boldsymbol{A}^{\tilde{r}} \boldsymbol{x} \preceq 0\}, \tag{12}$$

which we term less precisely as *analytic face of a SDF's zero-level isosurface*, shortened as *analytic face*, since it is possible that the face goes towards infinity in some directions. With the analytic form (12), $\mathcal{Z}$ realized by a ReLU based MLP thus defines a piecewise planar surface, which could be an approximation to an underlying smooth surface when the SDF F is trained using techniques presented shortly in Section 5.

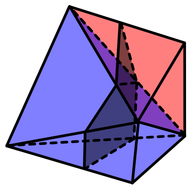

Figure 1. An illustration that two analytic cells (respectively colored as blue and red) connect via a shared boundary plane, on which their associated analytic faces intersect to form a mesh edge.

3.3. A closed mesh via connected cells and faces

We have not so far specified the types of surface that $\mathcal{Z} = \{\boldsymbol{x} \in \mathbb{R}^3 | f \circ \boldsymbol{T}(\boldsymbol{x}) = 0\}$ can represent, as long as they are piecewise planar whose associated analytic cells and faces respectively satisfy (11) and (12). In practice, one is mostly interested in those surface type representing the boundaries of non-degenerate 3D objects, which means that the objects do not have infinitely thin parts and a surface properly separates the interior and exterior of its object (cf. Figure 1.1 in (Botsch et al., 2010) for an illustration). This type of object surface usually has the property of being continuous and closed.

A closed, piecewise planar surface $\mathcal{Z}$ means that every planar face of the surface is connected with other faces via shared edges. We have the following theorem that characterizes the conditions under which analytic faces (12) in their respective analytic cells (11) guarantee to connect and form a closed, piecewise planar surface.

Theorem 5. *Assume that the zero-level isosurface $\mathcal{Z}$ of a SDF $F = f \circ \boldsymbol{T}$ defines a closed, piecewise planar surface. If for any region/cell $r \in \mathcal{R}(\boldsymbol{T})$, its associated linear mapping $\boldsymbol{T}^r$ (5) and the induced plane $\boldsymbol{a}_F^r = \boldsymbol{w}_f^\top \boldsymbol{T}^r$ (8) are uniquely defined, i.e., $\boldsymbol{T}^r \neq \beta \boldsymbol{T}^{r'}$ and $\boldsymbol{a}_F^r \neq \beta \boldsymbol{a}_F^{r'}$ for any region pair of r and r', where β is an arbitrary scaling factor, then analytic faces $\{\mathcal{P}_F^{\tilde{r}}\}$ defined by (12) connect and exactly form the surface $\mathcal{Z}$.*

Proof. See the supplementary material for the proof. Given the assumed conditions, the proof can be sketched by first showing that each planar face on the surface $\mathcal{Z}$ captured by the SDF $F = f \circ \boldsymbol{T}$ uniquely corresponds to an analytic face of an analytic cell, and then showing that for any pair of planar faces connected on $\mathcal{Z}$, their corresponding analytic faces are connected via boundaries of their respective analytic cells. □

We note that the conditions assumed in Theorem 5 are practically reasonable up to a numerical precision of the learned weights in the SDF $F = f \circ \boldsymbol{T}$. The proof of theorem also suggests an algorithm to identify the polygonal faces of a surface mesh learned by F, which is to be presented shortly.

4. The proposed analytic marching algorithm

Given a learned SDF $F = f \circ \boldsymbol{T}$ whose zero-level isofurface $\mathcal{Z} = \{\boldsymbol{x} \in \mathbb{R}^3 | F(\boldsymbol{x}) = 0\}$ defines a closed, piecewise planar surface, Theorem 5 suggests that obtaining the mesh of $\mathcal{Z}$ concerns with identification of analytic faces $\mathcal{P}_F^{\tilde{r}}$ in $\{\mathcal{C}_F^{\tilde{r}} | \tilde{r} \in \widetilde{\mathcal{R}}\}$. To this end, we propose an algorithm of *analytic marching* that marches among $\{\mathcal{C}_F^{\tilde{r}} | \tilde{r} \in \widetilde{\mathcal{R}}\}$ to identify vertices and edges of the polygonal faces $\{\mathcal{P}_F^{\tilde{r}} | \tilde{r} \in \widetilde{\mathcal{R}}\}$, where the name is indeed to show respect to the classical discrete algorithm of marching cubes (Lorensen & Cline, 1987).

Specifically, analytic marching is triggered by identifying at least one point $\boldsymbol{x} \in \mathcal{Z}$. Given the parametric model F and an arbitrarily initialized point $\boldsymbol{x} \in \mathbb{R}^3$, this can be simply achieved by solving the following problem with stochastic gradient descent (SGD)

$$\min_{\boldsymbol{x} \in \mathbb{R}^3} |F(\boldsymbol{x})|. \tag{13}$$

For a point $\boldsymbol{x}$ with $F(\boldsymbol{x}) = 0$, its state vector $\boldsymbol{s}(\boldsymbol{x})$ can be computed via (4), which specifies the analytic cell $\mathcal{C}_F^{\tilde{r}_{\boldsymbol{x}}}$ (11) and analytic face $\mathcal{P}_F^{\tilde{r}_{\boldsymbol{x}}}$ (12) where $\boldsymbol{x}$ resides. Initialize an active set $\mathcal{S}^{\bullet} = \emptyset$ and an inactive set $\mathcal{S}^{\circ} = \emptyset$. Push $\boldsymbol{s}(\boldsymbol{x})$ into $\mathcal{S}^{\bullet}$. Analytic marching proceeds by repeating the following steps.

1. Take an active state $\boldsymbol{s}_i$ from $\mathcal{S}^{\bullet}$, which specifies its analytic cell $\mathcal{C}_F^{\tilde{r}_i}$ and analytic face $\mathcal{P}_F^{\tilde{r}_i}$.

2. To compute the set $\mathcal{V}_{\mathcal{P}}^{\tilde{r}_i}$ of vertex points associated with $\mathcal{P}_F^{\tilde{r}_i}$, enumerates all the pair $(H_{lk}^{\tilde{r}_i}, H_{l'k'}^{\tilde{r}_i})$ of boundary planes $\{H_{lk}^{\tilde{r}_i}\}$ defined by (9), with $l = 1, \dots, L$ and $k = 1, \dots, n_l$. Each pair $(H_{lk}^{\tilde{r}_i}, H_{l'k'}^{\tilde{r}_i})$, together with $\mathcal{P}_F^{\tilde{r}_i}$, defines the following system of three equations
$$\boldsymbol{B}^{\tilde{r}_i}\boldsymbol{x} = 0, \tag{14}$$
where $\boldsymbol{B}^{\tilde{r}_i} = [\boldsymbol{a}_{lk}^{\tilde{r}_i}; \boldsymbol{a}_{l'k'}^{\tilde{r}_i}; \boldsymbol{a}_F^{\tilde{r}_i}] \in \mathbb{R}^{3\times 3}$.

3. System (14) gives a potential vertex $\boldsymbol{v}$ corresponding to the boundary pair $(H_{lk}^{\tilde{r}_i}, H_{l'k'}^{\tilde{r}_i})$. Validity of $\boldsymbol{v}$ is checked by the boundary condition (10) of the cell $\mathcal{C}_F^{\tilde{r}_i}$; if it is true, we have a vertex $\boldsymbol{v} \in \mathcal{V}_{\mathcal{P}}^{\tilde{r}_i}$. All the valid vertices obtained by solving (14) form $\mathcal{V}_{\mathcal{P}}^{\tilde{r}_i}$ of the face $\mathcal{P}_F^{\tilde{r}_i}$, whose pair-wise edges are defined by those on the same boundary planes.

4. Record all the boundary planes $\{\widehat{H}_{lk}^{\tilde{r}_i}\}$ of $\mathcal{C}_F^{\tilde{r}_i}$ that give valid vertices. Proof of Theorem 5 tells that the surface $\mathcal{Z}$ is in general well positioned in $\{\mathcal{C}_F^{\tilde{r}} | \tilde{r} \in \widetilde{\mathcal{R}}\}$ (cf. proof of Theorem 5 for details), and the analytic cell connecting $\mathcal{C}_F^{\tilde{r}_i}$ at a boundary plane $\widehat{H}_{lk}^{\tilde{r}_i}$ has its state vector switching only at the k^{th} neuron of layer l, which gives a new state $\hat{\boldsymbol{s}}_i$ and thus a new analytic cell.

5. Push $\boldsymbol{s}_i$ out of the active set $\mathcal{S}^{\bullet}$ and into the inactive set $\mathcal{S}^{\circ}$. Push $\{\hat{\boldsymbol{s}}_i | \hat{\boldsymbol{s}}_i \notin \mathcal{S}^{\circ}\}$ into the active set $\mathcal{S}^{\bullet}$.

The algorithm of analytic marching proceeds by repeating the above steps, until the active set $\mathcal{S}^{\bullet}$ is cleared up.

Algorithmic guarantee Theorem 5 guarantees that when the SDF F learns its zero-level isosurface Z as a closed, piecewise planar surface, identification of all the analytic faces forms the closed surface mesh. The proposed analytic marching algorithm is anchored at the cell state transition of the above step 4, whose success is of high probability due to a phenomenon similar to the blessing of dimensionality (Gorban & Tyukin, 2018). More specifically, it is of low probability that edges connecting planar faces of $\mathcal{Z}$ coincide with those of analytic cells (cf. proof of Theorem 5 for detailed analysis).

4.1. Analysis of computational complexities

Assume that the SDF $F = f \circ \boldsymbol{T}$ is built on an MLP of L hidden layers, each of which has n_l neurons, $l = 1, \dots, L$. Let $N = n_1 + \dots, n_L$. For ease of analysis, we assume $n_1 = \dots = n_L = n$ and thus $N = nL$. The computations inside each analytic cell concern with computing the boundary planes, solving a maximal number of $\binom{N}{2}$ equation systems (14), and checking the validity of resulting vertices, which give a complexity order of $\mathcal{O}(n^3L^3)$. Improving step 2 of the algorithm with pivoting operation (Avis & Fukuda, 1991) avoids enumeration of all pairs of boundary planes, reducing the complexity to an order of $\mathcal{O}(|\mathcal{V}_{\mathcal{P}}|n^2L^2)$, where $|\mathcal{V}_{\mathcal{P}}|$ represents the number of vertices per face and is typically less than 10. We know from (Montúfar et al., 2014) that the maximal size of the set $\mathcal{R}(\boldsymbol{T})$ of linear regions in general has an order of $\mathcal{O}\left((n/n_0)^{(L-1)n_0}n^{n_0}\right)$, where n_0 is the dimensionality of input space. Since our focus of interest is the 2-dimensional object surface embedded in the 3D space, we have $n_0 = 2$ and thus the maximal size of $\mathcal{R}(\boldsymbol{T})$, which bounds the maximal number of analytic cells, in general has an order of $\mathcal{O}\left((n/2)^{2(L-1)}n^2\right)$. Overall, the complexity of our analytic marching algorithm has an order of $\mathcal{O}\left((n/2)^{2(L-1)}|\mathcal{V}_{\mathcal{P}}|n^4L^2\right)$, which is exponential w.r.t. the MLP depth L and polynomial w.r.t. the MLP width n.

The above analysis shows that the complexity nature of our algorithm is the complexity of SDF function, which is completely different from those of existing algorithms, such as marching cubes (Lorensen & Cline, 1987), whose complexities are irrelevant to function complexities but rather depend on the discretized resolutions of the 3D space. Our algorithm thus provides an opportunity to recover highly precise mesh reconstruction by learning MLPs of low complexities. Alternatively, one may resort to network compression/pruning techniques (Han et al., 2015; Luo et al., 2017), which can potentially reduce network complexities with little sacrifice of inference precision.

4.2. Practical implementations and parallel efficiency

The proposed analytic marching can be naturally implemented in parallel. Instead of triggering the algorithm from a single $\boldsymbol{x}$ by solving (13), we can practically initialize as many as possible such points from the 3D space, and the algorithm would proceed in parallel. The parallel implementation can also be enhanced by simultaneously marching towards all the neighboring cells of the current one (cf. steps 4 and 5 of the algorithm). Since the SDF $F = f \circ \boldsymbol{T}$ is learned from training samples of ground-truth object mesh, *its zero-level isosurface is practically not guaranteed to be exactly the same as the ground truth, and in many cases, it is not even closed*; consequently, the condition assumed in Theorem 5 is not satisfied. In such cases, initialization of multiple surface points would help recover components of the surface that are possibly isolated, whose efficacy is verified in Section 6.

5. Training of deep implicit surface networks

For any $\boldsymbol{x} \in \mathbb{R}^3$, let $d(\boldsymbol{x})$ denote its ground-truth value of signed distance to the surface. We use the following regularized objective to train a SDF $F = f \circ \boldsymbol{T}$,

$$\min_{F=f\circ\boldsymbol{T}} \mathbb{E}_{\boldsymbol{x}\sim\mathbb{R}^3} \left|F(\boldsymbol{x})-d(\boldsymbol{x})\right| + \alpha\left|\|\partial F(\boldsymbol{x})/\partial\boldsymbol{x}\| - 1\right|, \quad (15)$$

where α is a penalty parameter. The unit gradient regularizer follows (Michalkiewicz et al., 2019), which aims to promote learning of a smooth gradient field.

6. Experiments

Datasets We use five categories of "Rifle", "Chair", "Airplane", "Sofa", and "Table" from the ShapeNetCoreV1 dataset (Chang et al., 2015), 200 object instances per category, for evaluation of different meshing algorithms. The 3D space containing mesh models of these instances is normalized in $[-1, 1]^3$. To train an MLP based SDF, we follow (Xu et al., 2019) and sample more points in the 3D space that are in the vicinity of the surface. Ground-truth SDF values are calculated by linear interpolation from a dense SDF grid obtained by (Sin et al., 2013; Xu & Barbič, 2014).
Implementation details Our training hyperparameters are as follows. The learning rates start at 1e-3, and decay every 20 epochs by a factor of 10, until the total number of 60 epochs. We set weight decay as 1e-4 and the penalty in (15) as $\alpha = 0.01$. In all experiments, we trigger our algorithm from 100 randomly sampled points. As described in Section 4.2, our algorithm naturally supports parallel implementation, which however, has not been customized so far; *the current algorithm is simply implemented on a CPU (Intel E5-2630 @ 2.20GHz) in a straightforward manner. For comparative algorithms such as marching cubes (Lorensen & Cline, 1987), their dominating computations of evaluating SDF values of sampled discrete points are implemented on a GPU (Tesla K80)*, which certainly gives them an unfair advantage. Nevertheless, we show in the following a better trade-off of precision and efficiency from our algorithm, even under the unfair comparison.
Evaluation metrics We use the following metrics to quantitatively measure the accuracies between recovered mesh results and ground-truth ones: (1) Chamfer Distance (CD), (2) Earth Mover's Distance (EMD), (3) Intersection over Union (IoU), and (4) F-score (F), where poisson disk sampling (Bowers et al., 2010) is used to sample points from surface. For the measures of IoU and F-Score, the larger the better, and for CD and EMD, the smaller the better. These metrics provide complementary perspectives to compare different algorithms. Additionally, wall-clock time and number of faces in the recovered meshes are reported as reference.

6.1. Ablation studies

Analysis in Section 4 tells that our proposed analytic marching is possible to exactly recover the mesh captured by a learned MLP. *We note that the zero-level isosurface of a learned MLP only approximates the ground-truth mesh*, and the approximation quality mostly depends on the capacity of the MLP network, which is in turn determined by network depth and network width. We study in this section how the network depth and width affect the recovery accuracies and efficiency of our proposed algorithm.

We conduct experiments by fixing two groups respectively of the same numbers of MLP neurons, while varying either the numbers of layers or the numbers of neurons per layer. The first group uses a total of 360 neurons, whose depth/width distributions are D4-W90, D6-W60, and D8-W45, where "D" stands for depth and "W" stands for width. The second group uses a total of 900 neurons, whose depth/width distributions are D10-W90, D15-W60, and D20-W45. Results in Table 1 tell that under different evaluation metrics, accuracies of the recovered meshes consistently improve with the increased network capacities, and the numbers of mesh faces and inference time are increased as well. Given a fixed number of neurons, it seems that properly deep networks are advantageous in terms of precision-efficiency trade off. Since the experimental settings fall in the (relatively) shallower range, polynomial increase of network width dominates the computational complexity, as analyzed in Section 4.1.

Results in Table 1 are from the experiments on the 200 instances of "Rifle" category. Results of other object categories are of similar quality. We summarize in Table 2 results of all the five categories based on an MLP of 6 layers with 60 neurons per layer (the D6-W60 setting in Table 1), which tell that the surface and/or topology complexities of "Chair" are higher, and those of "Airplane" are lower.

6.2. Comparisons with existing meshing algorithms

In this section, we compare our proposed analytic meshing (AM) with existing algorithms of greedy meshing (GM), marching cubes (MC) (Lorensen & Cline, 1987), marching tetrahedra (MT) (Doi & Koide, 1991), and dual contouring (DC) (Ju et al., 2002), where MC is the de-facto standard meshing solution adopted by many surface meshing applications, and DC improves over MC with ingredients including partitioning the 3D space with a hierarchical structure of octree. These comparative algorithms are all based on discretizing the 3D space by evaluating the SDF values at a regular grid of sampled points; consequently, their meshing accuracies and efficiency depend on the sampling resolution. We thus implement them under a range of sampling resolutions from 32^3 to a GPU memory limit of 512^3.

Figure 2 shows that among these comparative methods, marching cubes in general performs better in terms of a balanced precision and efficiency. However, under different evaluation metrics, recovery accuracies of these methods are upper bounded by our proposed one. As noted in the implementation details of this section, the dominating computations of these methods are implemented on GPU, which gives them an unfair advantage of computational efficiency. Nevertheless, results in Figure 2 tell that even under the unfair comparison, our algorithm is much faster at a similar level of recovery precision.

The quantitative results in Figure 2 are averaged ones over all the object instances of all the five categories, using an MLP of 6 layers with 60 neurons per layer (the D6-W60 setting in Table 1). We finally show qualitative results in Figure 3, where mesh results of an example object are presented. More qualitative results are given in the supplementary material. Our proposed algorithm is particularly advantageous in capturing geometry details on high-curvature surface areas.

7. Conclusion

In this work, we contribute an analytic meshing solution from learned deep implicit surface networks. Our contribution is motivated by the established knowledge that a ReLU based MLP partitions its input space into a number of linear regions. We identify from these regions analytic cells and analytic faces that are associated with the zero-level isosurface of the learned MLP based implicit function. We prove that under mild conditions, the identified analytic faces are guaranteed to connect and form a closed, piecewise planar surface. Our theorem inspires us to propose a naturally parallelizable algorithm of analytic marching, which marches among analytic cells to exactly recover the mesh captured by a learned MLP. Experiments on benchmark dataset of 3D object repositories confirm the advantages of our algorithm over existing ones.

Computational complexity of our algorithm depends on the capacities of deployed MLPs, which are in turn determined by network depth and width. It is thus promising to use network compression/pruning techniques to reduce the model complexities, with little sacrifice of precision, which could further improve the inference efficiency of our proposed algorithm. We will pursue this direction in future research.

Acknowledgements

This work is supported in part by National Natural Science Foundation of China (Grant No. 61771201), Program for Guangdong Introducing Innovative and Enterpreneurial Teams (Grant No. 2017ZT07X183), Guangdong R&D key project of China (Grant No. 2019B010155001), and the Guangzhou Key Laboratory of Body Data Science under Grant 201605030011.

References

Avis, D. and Fukuda, K. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. volume 8, pp. 98–104, 01 1991.

Blinn, J. F. A generalization of algebraic surface drawing. *ACM Trans. Graph.*, 1(3):235–256, July 1982.

Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Levy, B. *Polygon Mesh Processing*. CRC Press, 2010.

Bowers, J., Wang, R., Wei, L.-Y., and Maletz, D. Parallel poisson disk sampling with spectrum analysis on surfaces. In *ACM SIGGRAPH Asia*, 2010.

Carr, J., Beatson, R., and Fright, W. Surface interpolation with radial basis functions for medical imaging. *IEEE Transactions on Medical Imaging*, 16(1), 2 1997.

Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., and Evans, T. R. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 67–76, 2001.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su,

Table 1. Ablation studies by varying the numbers of layers and the numbers of neurons per layer for the trained MLPs of SDF. "D" stands for network depth and "W" for width. Results are from the 200 instances of "Rifle" categories. F-scores use $\tau = 5 \times 10^{-3}$.

Architecture	CD($\times 10^{-1}$)	EMD($\times 10^{-3}$)	IoU(%)	F@τ(%)	F@2τ(%)	Face No.	Time(sec.)
D4-W90	6.16	8.60	86.5	84.8	95.4	97865	17.0
D6-W60	5.29	8.00	86.9	85.5	95.8	100179	15.0
D8-W45	5.67	8.12	85.7	84.2	95.4	93482	9.75
D10-W90	3.64	5.85	90.2	88.1	98.6	421840	173
D15-W60	3.85	6.23	88.9	87.4	97.5	336044	86.8
D20-W45	4.21	6.89	86.1	86.6	95.5	253970	47.3

Table 2. Results of all the five categories based on an MLP of 6 layers with 60 neurons per layer (the D6-W60 setting in Table 1). F-scores use $\tau = 5 \times 10^{-3}$.

Category	CD($\times 10^{-1}$)	EMD($\times 10^{-3}$)	IoU(%)	F@τ(%)	F@2τ(%)	Face No.	Time(sec.)
Rifle	5.29	8.00	86.9	85.5	95.8	100179	15.0
Chair	7.27	8.01	89.6	52.9	96.1	182624	25.3
Airplane	3.25	5.32	89.4	89.8	97.9	131074	19.0
Sofa	6.78	6.65	96.6	53.2	97.9	156863	22.0
Table	6.98	7.16	90.7	51.6	96.8	163395	22.7
Mean	5.91	7.03	90.6	66.6	96.9	146827	20.8

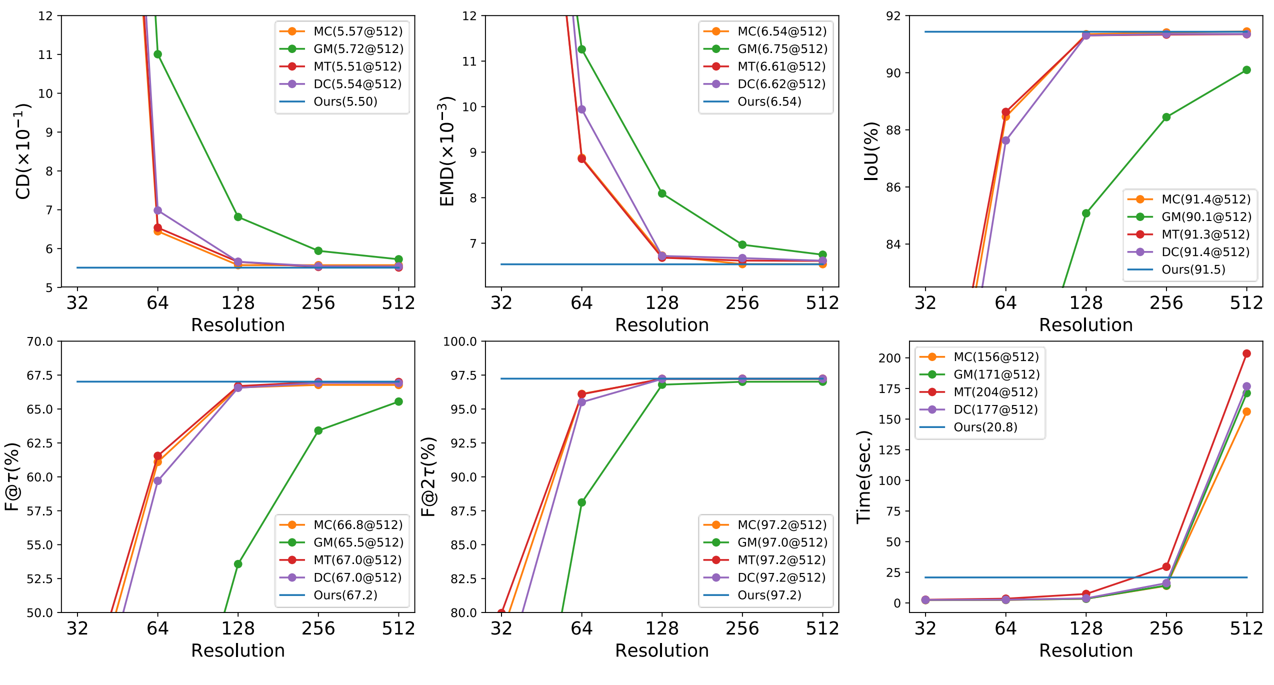

Figure 2. Quantitative comparisons of different meshing algorithms under metrics of recovery precision and inference world-clock time. For greedy meshing (GM), marching cubes (MC), marching tetrahedra (MT), and dual contouring (DC), results under a resolution range of discrete point sampling from 32^3 to a GPU memory limit of 512^3 are presented, and the dominating computations of their sampled points' SDF values are implemented on GPU. Numerical results of this figure are given in the supplementary material.

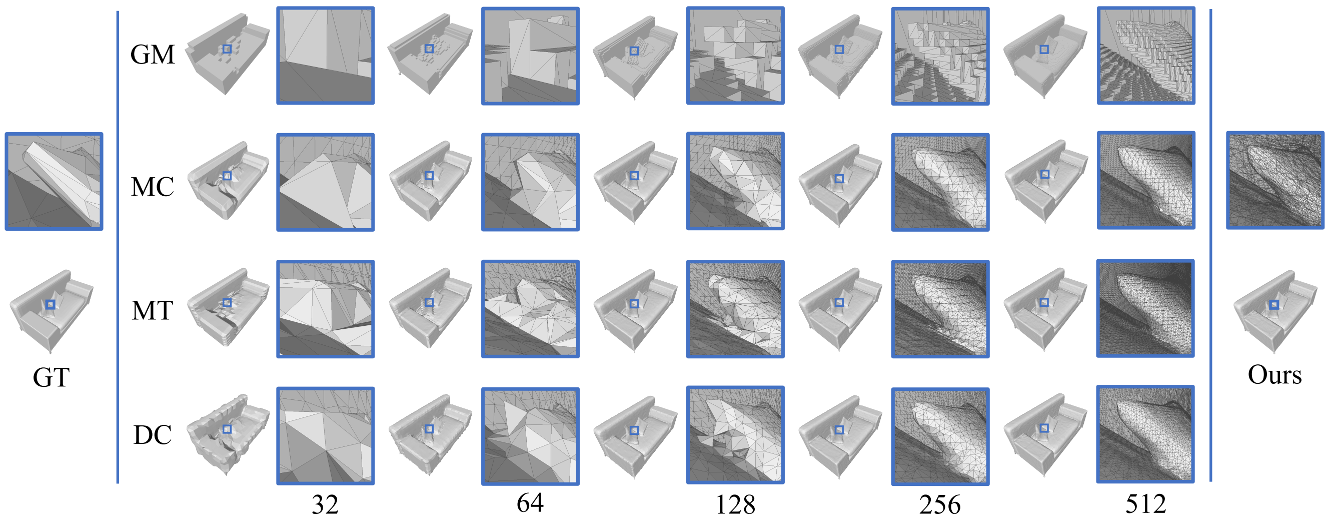

Figure 3. Qualitative comparisons of different meshing algorithms. For greedy meshing (GM), marching cubes (MC), marching tetrahedra (MT), and dual contouring (DC), results under a resolution range of discrete point sampling from 32^3 to a GPU memory limit of 512^3 are presented.

H., Xiao, J., Yi, L., and Yu, F. ShapeNet: An Information-Rich 3D Model Repository. *arXiv:1512.03012*, 2015.

Chu, L., Hu, X., Hu, J., Wang, L., and Pei, J. Exact and consistent interpretation for piecewise linear neural networks: A closed form solution. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1244–1253, 2018.

Curless, B. and Levoy, M. A volumetric method for building complex models from range images. In *SIGGRAPH*, pp. 303–312. ACM, 1996.

Doi, A. and Koide, A. An efficient method of triangulating equivalued surfaces by using tetrahedral cells. *IEICE Transactions on Information and Systems*, 74, 01 1991.

Glorot, X., Bordes, A., and Bengio, Y. Deep sparse rectifier neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2011.

Gorban, A. and Tyukin, I. Blessing of dimensionality: mathematical foundations of the statistical physics of data. *arXiv:1801.03421*, 2018.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pp. 1135–1143, 2015.

Jia, K., Li, S., Wen, Y., Liu, T., and Tao, D. Orthogonal deep neural networks. *IEEE Transactions on Pattner Analysis and Machine Intelligence*, 2019.

Ju, T., Losasso, F., Schaefer, S., and Warren, J. Dual contouring of hermite data. *ACM Trans. Graph.*, 21(3):339–346, July 2002.

Lorensen, W. E. and Cline, H. E. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH*, pp. 163–169. ACM, 1987.

Luo, J.-H., Wu, J., and Lin, W. Thinet: A filter level pruning method for deep neural network compression. In *IEEE international conference on computer vision*, pp. 5058–5066, 2017.

Michalkiewicz, M., Pontes, J. K., Jack, D., Baktashmotlagh, M., and Eriksson, A. Deep level sets: Implicit surface representations for 3d shape inference. *arXiv:1901.06802*, 2019.

Montúfar, G., Pascanu, R., Cho, K., and Bengio, Y. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 2924–2932, 2014.

Nishimura, H., Hirai, M., Kawai, T., Kawata, T., Shirkawa, I., and Omura, K. Object modeling by distributed function and a method of image generation (in japanese). *The Transactions of the Institute of Electronics and Communication Engineers of Japan*, 68, 01 1985.

Orlik, P. and Terao, H. *Arrangements of Hyperplanes*. Springer Berlin Heidelberg, 1992.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. Deepsdf: Learning continuous signed distance functions for shape representation. *arXiv:1901.05103*, 2019.

Pascanu, R., Montúfar, G., and Bengio, Y. On the number of response regions of deep feed forward networks with piece-wise linear activations. In *International Conference on Learning Representations*, 2014.

Sin, F., Schroeder, D., and Barbič, J. Vega: Non-linear fem deformable object simulator. *Computer Graphics Forum*, 32, 2013.

Turk, G. and O'Brien, J. F. Shape transformation using variational implicit functions. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 335–342, 1999.

Wyvill, G., McPheeters, C., and Wyvill, B. Data structure for soft objects. *The Visual Computer*, 2:227–234, 08 1986.

Xu, H. and Barbič, J. Signed distance fields for polygon soup meshes. *Proceedings - Graphics Interface*, pp. 35–41, 01 2014.

Xu, Q., Wang, W., Ceylan, D., Mech, R., and Neumann, U. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *arXiv:1905.10711*, 2019.

Zaslavsky, T. Facing up to arrangements: Face-count formulas for partitions of space by hyperplanes. *Number 154 in Memoirs of the American Mathematical Society*, 1975.