DeepDT: Learning Geometry From Delaunay Triangulation for Surface Reconstruction

Yiming Luo †, **Zhenxing Mi** †, **Wenbing Tao** *

National Key Laboratory of Science and Technology on Multi-spectral Information Processing
School of Artifical Intelligence and Automation, Huazhong University of Science and Technology, China
yiming_luo@163.com, mizhenxing.henan@gmail.com, wenbingtao@hust.edu.cn

Abstract

In this paper, a novel learning-based network, named DeepDT, is proposed to reconstruct the surface from Delaunay triangulation of point cloud. DeepDT learns to predict inside/outside labels of Delaunay tetrahedrons directly from a point cloud and corresponding Delaunay triangulation. The local geometry features are first extracted from the input point cloud and aggregated into a graph deriving from the Delaunay triangulation. Then a graph filtering is applied on the aggregated features in order to add structural regularization to the label prediction of tetrahedrons. Due to the complicated spatial relations between tetrahedrons and the triangles, it is impossible to directly generate ground truth labels of tetrahedrons from ground truth surface. Therefore, we propose a multi-label supervision strategy which votes for the label of a tetrahedron with labels of sampling locations inside it. The proposed DeepDT can maintain abundant geometry details without generating overly complex surfaces , especially for inner surfaces of open scenes. Meanwhile, the generalization ability and time consumption of the proposed method is acceptable and competitive compared with the state-of-the-art methods. Experiments demonstrate the superior performance of the proposed DeepDT.

Introduction

Surface reconstruction from 3D point clouds is a long-standing problem in computer vision and graphics (Kazhdan and Hoppe 2013). A lot of previous methods use an implicit function framework (Curless and Levoy 1996; Kazhdan, Bolitho, and Hoppe 2006; Kazhdan and Hoppe 2013). They typically discretize space in the bounding box of the input point cloud with a voxel grid or an adaptive octree. Then they compute an implicit function from input points. The result surface is extracted from the grids as an iso-surface of the implicit function by Marching Cubes (MC) (Lorensen and Cline 1987). However, solving equations of large scale in implicit methods can be time-consuming. The maximum resolution of the octree depth also influences the efficiency. There is a quadratic relation between resolution and run time as well as memory usage. Post-processing steps are also needed to clean up the excess part in result surfaces sometimes (trimming), which are sensitive to corresponding parameters and make batch processing of point clouds impractical. This family of implicit approaches is sometimes limited by their sensitivity to noise, outliers, non-uniform sampling or even simply by the lack of reliable and consistent normal estimates and orientation (Labatut, Pons, and Keriven 2009). Another common method uses a Delaunay triangulation of a point cloud to subdivide the space into uneven tetrahedrons. As analyzed in (Amenta and Bern 1999), under the assumption of sufficiently dense point clouds, a good approximation of the surface is contained in the Delaunay triangulation. Therefore, opposed to the approaches that try to fit a surface in a continuous space, surface reconstruction based on Delaunay triangulation can now be reduced to selecting an adequate subset of the triangular facets in the Delaunay triangulation. Researchers have spent a lot of efforts in finding the object surface in a Delaunay triangulation. A commonly used strategy is to classify the tetrahedrons as inside/outside the surface using graph cuts. Visibility information is required in such a process. A 2D example is shown in Figure 1. Such method builds a directed graph based on the neighborhood of adjacent tetrahedrons in the Delaunay triangulation. Visibility terms make this kind of approaches more accurate and robust to outliers.

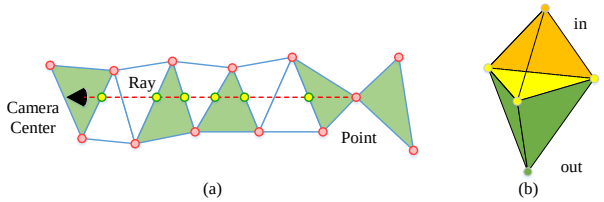

Figure 1: A 2D example of reconstructing surface by in/out labeling of tetrahedrons. The visibility information is integrated into each tetrahedron by intersections between viewing rays and tetrahedrons. A graph cuts optimization is applied to classify tetrahedrons as inside or outside the surface. Result surface is reconstructed by extracting triangular facets between tetrahedrons of different labels.

However, such method still faces problems. Some tetrahedrons lying behind points risk being mislabeled because

† Equal contribution.

* Corresponding author.

no viewing rays from the camera center to the points will ever intersect them. Therefore, such method could generate overly complex bumpy surfaces with many unexpected handles inside the model, failing to get clean inner surfaces. What's more, such a method fails to cope with arbitrary point clouds without visibility information. It suggests that correct labeling of tetrahedrons without visibility information is still a bottleneck to overcome.

Most recently, the rapid development of deep learning and improvement of large-scale 3D datasets make it possible to train neural networks for 3D shapes. A variety of learning-based surface reconstruction methods have been proposed, such as learning to refine Truncated Signed Distance Function (TSDF) on voxel by voxel networks or octree grids by octree networks (Dai, Ruizhongtai Qi, and Nießner 2017; Riegler et al. 2017; Cao et al. 2018), learning occupancy and SDF functions from point clouds (Mescheder et al. 2019; Park et al. 2019). Learning-based methods are inclined to introduce more geometry priors into surface reconstruction for better performance. However, existing learning-based methods still face problems: (1) Most of them still rely on voxel or octree structures, which are not prone to maintain high computational efficiency. (2) Their ability to generate fine-grained structures and details is still limited due to the low-resolution girds or the latent vector of a fixed size. (3) Most of them still rely on global features of training data excessively, which leads to generalization problems.

Another excellent work, SSRNet (Mi, Luo, and Tao 2020), the state-of-the-art learning-based method in Surface Reconstruction from Point Clouds (SRPC) task to our known, constructs local geometry-aware features, which leads to accurate classification for octree vertices and outstanding generalization capability among different datasets. However, it is worth noting that not all inputs are dense enough and there may be missing data in the input.

This inspires us to explore a better learning-based method, which is able to finish reconstruction effectively and efficiently without visibility information. In this paper, we propose DeepDT, a novel learning-based method for surface reconstruction from point clouds based on Delaunay triangulation. We construct Delaunay triangulation from the point cloud and replace graph cuts with a network for accurate labeling of tetrahedrons. The network takes a point cloud and its Delaunay triangulation as input, and learns to classify the tetrahedrons as inside/outside. In our method, the key insight is that the inside/outside labeling of tetrahedrons can be determined via local geometry information of points and the structural information of Delaunay triangulation. We first encode features of points with the geometry feature extraction module. It exploits local K nearest neighbors to encode a local geometry feature for each point. Meanwhile, it directly encodes features representing in/out information, such as signed distances. Therefore, it is able to capture local geometry details of the surface. We then augment the graph of Delaunay triangulation with geometry features by a point-to-graph feature aggregation operation. We use an attention mechanism to aggregate geometry features into graph nodes. It can automatically select important geometry features for in/out classification of tetrahedrons. Subsequently, we apply a graph filtering module to the feature augmented graph. The module exchanges information among neighboring tetrahedrons with Graph Convolution Networks (GCN) (Kipf and Welling 2016) and thus adds structural constraints to the label prediction. Our network is trained in a supervised manner. However, it brings about a problem that we cannot directly get the ground truth labels of tetrahedrons from ground truth surfaces. Since tetrahedrons are likely to have complicated intersections with triangular surfaces, we cannot obtain the relative position relationship between the tetrahedron and the triangular mesh. To tackle this problem, we propose a multi-label supervision strategy. We obtain the in/out labels of tetrahedrons through a multi-label supervision strategy which utilizes the labels of multiple reference locations sampled inside them. As analyzed above, our method enjoys a combination of novelty as follows.

- Our method integrates geometry and graph structural information for more robust tetrahedron labeling.
- Local geometry features together with global graph structural regularization benefits our method with good accuracy and generalization capability.
- The multi-label supervision mechanism makes it possible to train a high quality model without ground truth labels of tetrahedrons or visibility information.

Experiments on challenging data show that our method can complete the reconstruction accurately and efficiently, and restore the geometric details of the input data with noise and complex topology. It is also proved that our method can generalize well across datasets of different styles.

Related Work

Geometric reconstruction methods Implicit reconstruction methods (Levin 2004; Guennebaud and Gross 2007; Fuhrmann and Goesele 2014; Kazhdan and Hoppe 2013; Curless and Levoy 1996; Carr et al. 2001; Turk and O'Brien 2002) attempt to approximate the surface as an implicit function from which an isosurface is extracted by Marching Cubes (MC) (Lorensen and Cline 1987). (Hoppe et al. 1992) estimates a tangent plane for each point with k-nearest neighbors. The implicit function is defined as the signed distance to the tangent plane of the closed point. Poisson surface reconstruction method (PSR) (Kazhdan, Bolitho, and Hoppe 2006) fits the vector field of point normals with gradient of an in/out indicator function. Since the discretization is regular, these methods are mostly suitable for compact point clouds with tight bounding boxes. Further more, a post-processing step is needed to obtain an explicit surface from an implicit function, known as ray tracing and MC.

Another common strategy to complete reconstruction task works with Delaunay tetrahedralization of the input points instead of voxel-based volumetric representation (Vu et al. 2011; Jancosek and Pajdla 2011; Hoppe et al. 2013; Hiep et al. 2009). (Labatut, Pons, and Keriven 2007) is the first to propose a reconstruction method (L-Method) that classifies tetrahedrons as inside/outside by visibility information and graph cuts. Many later methods mostly focus on adding different terms to the graph cuts in order to improve the surface quality (Labatut, Pons, and Keriven 2009; Jancosek and

Pajdla 2014; Zhou, Shen, and Hu 2019). The strategy of visibility constraints and graph optimization has achieved remarkable performance on challenging data with noise and outliers. However, it fails to cope with arbitrary point clouds without visibility information. Even with visibility information, it is usually not enough to label tetrahedrons behind a point correctly.

Learning-based methods Recently developing deep learning techniques have given rise to lots of learning-based reconstruction methods (Liao, Donné, and Geiger 2018; Groueix et al. 2018). The networks based on voxel or octree grids (Riegler et al. 2017; Dai, Ruizhongtai Qi, and Nießner 2017; Riegler, Osman Ulusoy, and Geiger 2017; Cao et al. 2018) usually face efficiency issues so they can only reconstruct surfaces of compact objects at a relatively low resolution. Instead of optimizing TSDF on grids, some learning-based methods directly learn a continuous function from point clouds. The ONet encodes the point cloud into a global latent vector. It predicts occupancy values for 3D locations by decoding the latent vector concatenated with 3D coordinates. DeepSDF (Park et al. 2019) is similar to ONet, but it predicts signed distance values. The result surface is also extracted by MC from an octree. Learning-based methods using voxel or octree grids to discretize the surface are sharing similar problems with the traditional implicit methods. Moreover, most existing learning-based methods encode structural details into fixed-size latent vectors. Although they can usually generate a final surface that roughly retains the shape of the input, they may not be able to capture geometric details of complex topologies. In addition, most existing methods learn too many global features, which makes them perform well on categories similar to the training set. However, they usually have difficulty maintaining the generalization ability well across different datasets.

Method

In this section, we describe the detailed network architecture of DeepDT. Figure 2(a) is a 2D illustration of our main pipeline. The input of our network includes a point cloud $\mathcal{P}$ (black dots in the left of Figure 2(a)) and its Delaunay triangulation $\mathcal{D}$ (black triangles in the left of Figure 2(a)). $\mathcal{P}$ is a set of 3D points with their normals. $\mathcal{D}$ is a set of tetrahedrons. The four vertices of each tetrahedron are points in the $\mathcal{P}$. Each tetrahedron also has four neighbor tetrahedrons sharing common triangular facets. The Delaunay triangulation structure forms a graph $\mathcal{G}$ (red dots connected by red lines in the left of Figure 2(a)), with tetrahedrons as nodes and common triangular facets connecting two adjacent tetrahedrons as edges. Our network can be divided into two parts. One is mainly about geometry feature extraction. The other is for graph feature aggregation and filtering.

Geometry Feature Extraction

The point features in our method provide initial geometry information for in/out classification of tetrahedrons. Therefore, the layer of geometry feature extraction module should have three properties to better complete local surface feature encoding. (1) It must directly provide in/out information with respect to the implicit surface. (2) It should encode in/out information locally in order to capture the geometry details. (3) It has to be computationally efficient in order to process larger point clouds. The computational efficiency and local encoding are related to the network architecture while the in/out information is determined by the input features.

Local Surface Feature Encoding We compute signed distances among each point and its neighbors as the raw input feature of our network. For each reference point $\mathbf{p}_i$, we search its K nearest neighbors $\mathbf{p}_i^k, k = 1, ..., K$. The neighbor $\mathbf{p}_i^k$ with normal $\mathbf{n}_i^k$ can be seen as a tangent plane t_i^k approximating the local surface near $\mathbf{p}_i^k$. Note that normals of points are all normalized so their lengths are equal to 1. As illustrated in Figure 2c, signed distance d_i^k between $\mathbf{p}_i$ and tangent plane can be calculated as:

$$d_i^k = (\mathbf{p}_i - \mathbf{p}_i^k) \cdot \mathbf{n}_i^k, \quad \|\mathbf{n}_i^k\| = 1 \tag{1}$$

The symbol $\cdot$ means dot product of vectors.

In addition to the signed distances, we also include relative normals in the input feature in order to provide richer information about local surface geometry. For each reference point $\mathbf{p}_i$ with normal $\mathbf{n}_i$ and one of its neighbor tangent plane t_i^k, we decompose $\mathbf{n}_i$ into two vectors $\mathbf{v}_i^k$ and $\mathbf{h}_i^k$. , relative to the tangent plane. $\mathbf{v}_i^k$ is perpendicular to t_i^k and $\mathbf{h}_i^k$ is parallel to $\mathbf{t}_i^k$. They can be calculated as:

$$\mathbf{v}_i^k = (\mathbf{n}_i \cdot \mathbf{n}_i^k)\mathbf{n}_i^k, \ \mathbf{h}_i^k = \mathbf{n}_i - \mathbf{v}_i^k, \ \|\mathbf{n}_i\| = \|\mathbf{n}_i^k\| = 1 \tag{2}$$

The symbol $\cdot$ means dot product of vectors. The calculation is illustrated in Figure 2c. After computing the signed distances and relative normals, the surface feature s_i^k between $\mathbf{p}_i$ and $\mathbf{p}_i^k$ is encoded as:

$$s_i^k = \text{MLP}(d_i^k \oplus \mathbf{v}_i^k \oplus \mathbf{h}_i^k) \tag{3}$$

MLP means Multi-Layer Perceptron. $\oplus$ means concatenation. By directly encoding the local surface features, the point features can provide in/out information for the classification of tetrahedrons. As a reward for using only local geometry features rather than global features, our network generalizes well across different datasets.

After designing Local Surface Feature Encoding, we have completed the most important part of geometric feature extraction. In Figure 2a, the geometry feature extraction network consists of multiple geometry feature extraction layers shown in Figure 2b. Besides, random downsampling expands the receptive fields. The l-th multiple geometry feature extraction layer takes the reference point $\mathbf{p}_i$, the previous feature F_i^{l-1} and K nearest neighbors $\mathbf{p}_i^k, k = 1, ..., K$ as input. It first encodes local surface features for $\mathbf{p}_i$ with respect to each neighbor point. Then F_i^{l-1} is repeated K times and concatenated with K local surface features. After that, the K features are aggregated by an attention pooling layer, which aims to use the attention mechanism to help automatically learn important local features. A group of MLP layers are applied to the aggregated feature and output result feature F_i^l as l-th feature of $\mathbf{p}_i$.

Feature Augmented Graph

Point to Graph Feature Aggregation After extracting the features for each point, we build a feature augmented

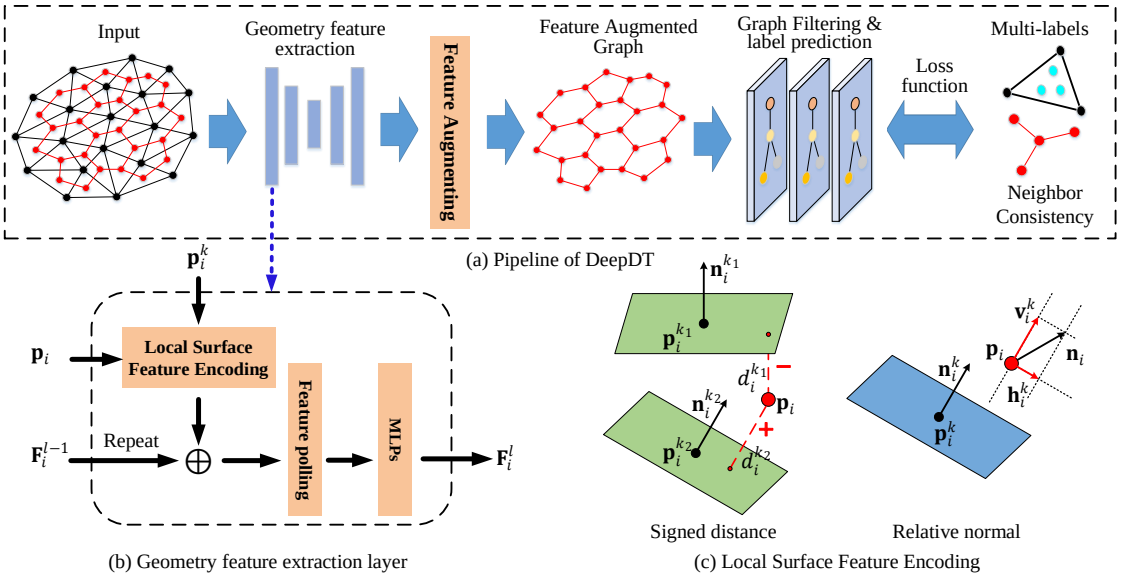

Figure 2: (a) The pipeline of our method. The input includes a point cloud (black dots) and its Delaunay triangulation (black triangles). The Delaunay triangulation structure forms a graph (red dots connected by red lines). (b) Detailed illustration of geometry feature extraction layer, $\oplus$ means concatenation. (c) Calculation of signed distances and relative normals.

graph from the point features and the graph $\mathcal{G}$. The graph directly derives from the Delaunay triangulation $\mathcal{D}$ of the point cloud. The graph nodes and graph edges correspond to Delaunay tetrahedron and the triangular facets between adjacent tetrahedrons. Each tetrahedron in $\mathcal{D}$ consists of four vertices from the point cloud. Therefore, we construct the tetrahedron features in the graph by aggregating the geometry features of the four vertices. Let T_i be the feature of ith node in $\mathcal{G}$ with its four point features $F_i^j (j = 1, 2, 3, 4)$. We construct T_i from the four vertex features by an attention mechanism, which is able to channel-wisely select geometry features important for the classification of tetrahedrons. We first learn weights for each channel of the four features. Let $F_i = \{F_i^j, j = 1, 2, 3, 4\}$ be the set of four features with shape $(4, C)$. W_i is the set of four weight vectors with the same shape $(4, C)$. C means the number of channels. The Softmax operation is applied to W_i in its first dimension. Then corresponding channels are weighted averaged.

$$W_i = \text{Softmax}(\text{MLP}(F_i)) \tag{4}$$

$$T_i = \sum W_i \odot F_i \tag{5}$$

$\odot$ is the element-wise product of W_i and F_i. $\sum$ is applied to the first dimension. T_i has the shape of $(1, C)$.

A special case is that the Delaunay triangulation has some infinite tetrahedrons which have three vertices in the 3D point cloud and share an infinite vertex. To solve this problem, we directly set the feature of the infinite vertex as zeros.

Graph Filtering The raw tetrahedron features in the graph $\mathcal{G}$ constructed by aggregating geometry features encode the inside/outside information of tetrahedrons. They are constructed from point features independently so they do not contain enough neighborhood graph information. They could contain noise and inconsistency due to the existence of noise in the point clouds. The classical graph cuts based methods have demonstrated that local neighborhood smoothing constraint in a graph is important for robust label prediction for graph nodes. Therefore, after the construction of graph features, we apply the multi-layer Graph Convolutional Network to the graph in order to integrate more local graph structural constraints. The GCN layers refine the tetrahedron features by exchanging information among neighboring tetrahedrons. It is able to encode more graph structural information for label prediction. The last layer of the GCNs outputs a 2-channel prediction vector for each tetrahedron. The softmax operation is finally applied to the vectors in order to get the probability for a tetrahedron to be inside/outside.

Loss Functions

Multi-label Supervision A straightforward supervision for our network is the ground truth labels of the tetrahedrons. However, due to the very complicated spatial relations the tetrahedrons have with the triangles, it is not trivial to label the tetrahedrons via the ground truth triangular surface. The tetrahedrons may have intersections with multiple triangles so the classification is ambiguous. One compromise approach is to use classical graph cuts based methods for label generation. However, this will limit the accuracy potential of the network. In our method, we propose a multi-label supervision method. Since it is easy to check whether a 3D location is inside/outside a surface, we randomly sample N reference locations (N_ref) in a tetrahedron and get

their inside/outside labels. Then we use these labels to supervise the labeling process of the tetrahedron. Each label gives a "vote" for each tetrahedron to determine in/out label. With the aforementioned architecture and multiple "votes", our method is able to encode more accurate relations among the tetrahedron labels, the point geometry and Delaunay triangulation structure. It is worth noting that these reference locations only provide labels for each tetrahedron. The locations themselves are not used in our network.

To train our network with multi-label supervision, we compute a multi-label loss L_m. It is a classification loss minimizing the error between predicted probabilities and multi-labels. With a Delaunay triangulation of N tetrahedrons, we sample N_ref reference locations for each tetrahedron T_i and get N_ref labels. The f_{ij} is the binary cross-entropy loss between the prediction probabilities of T_i and its jth label l_i^j. L_m gives more help to label predication with N_ref rising. This will be very beneficial to the prediction of very large tetrahedrons. It can be denoted by:

$$L_m = \frac{1}{N \times N\_ref} \sum_{i=1}^{N} \sum_{j=1}^{N\_ref} f_{ij} \tag{6}$$

Neighbor Consistency Constraint In classical graph cuts based methods, the label smoothness of neighbor tetrahedron helps to reconstruct more smooth surfaces. In our method, we explicitly introduce a regularization loss for more consistent labeling among adjacent tetrahedrons. This loss encourages more smooth surfaces and also helps to classify the very large tetrahedrons connecting points far from each other. With a Delaunay triangulation of N tetrahedrons, the ith tetrahedron T_i has four neighbor tetrahedrons $T_i^j, (j = 1, 2, 3, 4)$. The g_{ij} is the cross-entropy between the prediction probabilities between T_i and its jth neighbor T_i^j. Then the multi-label loss in computed as:

$$L_n = \frac{1}{4N} \sum_{i=1}^{N} \sum_{j=1}^{4} g_{ij} \tag{7}$$

With λ_1, λ_2 balancing the two losses, our loss function can be summarized as:

$$L = \lambda_1 L_m + \lambda_2 L_N \tag{8}$$

Surface Extraction And Data Preparation

After getting the label of each tetrahedron, the output triangular surface is generated by taking the triangular facets between adjacent tetrahedrons with different labels. We then post-process the result surface by Laplacian-based smoothing method for a more smooth surface and better visual effects. We prepare training data from a point cloud and its ground truth surface. First, Delaunay triangulation is constructed from the point cloud using the Computational Geometry Algorithms Library (CGAL) (Boissonnat et al. 2000). The point indices for each tetrahedron and the adjacent matrix used for graph convolution are derived trivially from the Delaunay structure. Then N_ref locations are randomly sampled in each tetrahedrons. We compute the inside/outside labels of these locations with respect to the ground truth surface and record these location labels.

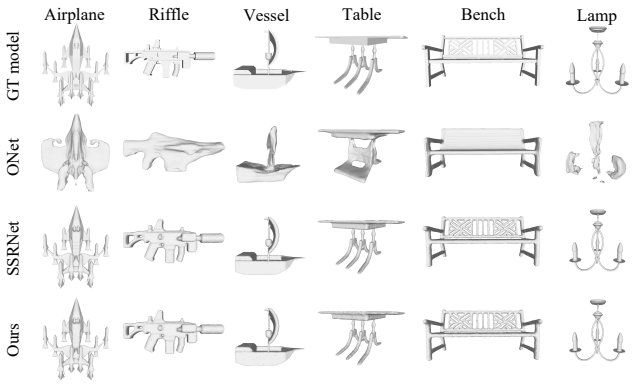

Figure 3: Qualitative results of ShapeNet test data.

Experiments

In this part, we perform a series of experiments on datasets of different scales to qualitatively and quantitatively evaluate our DeepDT from different perspectives.

Datasets & Evaluation Metrics We select three datasets with quite different types to comprehensively compare our DeepDT with traditional methods and other state-of-the-art learning-based methods. They are ShapeNet (Chang et al. 2015), DTU (Jensen et al. 2014) and Stanford 3D. [1] We choose the Chamfer-L_1 distance and the normal consistency (NC) score for experiments on ShapeNet. For DTU dataset, we follow the DTU Completeness metric given by DTU and also take Chamfer Distances (CD) into consideration. We compute the CD between the ground truth point cloud and the vertices of the result surface that is expressed by the triangular meshes. For each point in a cloud, CD finds the nearest point in the other point set, and averages the square of distances up. When it comes to Stanford 3D, we evaluate Chamfer Distances in generalization capability tests. In the following experiments, unless otherwise specified, the number of reference locations (N_ref) is set to 5, and the λ_1, λ_2 are set to 0.9 and 0.1, respectively.

Results on ShapeNet

In this section, we compare our method with several state-of-the-art learning-based methods on ShapeNet : DMC (Liao, Donné, and Geiger 2018), ONet, LDIF (Genova et al. 2020), CEISR (Poursaeed et al. 2020), SSRNet. For fair comparison, we adopt the same train/validation/test split (about $30K$/$4.5K$/$9K$ shapes) as the mentioned methods. Since data in ShapeNet is synthetic, we apply noise using a Gaussian distribution with zero mean and standard deviation 0.05 to the point clouds as ONet and SSRNet did. In order to get the labels of sampled reference locations, we reconstruct surfaces through PSR (octrees depth=9) to generate training data. Quantitative evaluation results for meshes generated by DeepDT are reported in Table 1. We can find that the NC score of our method is comparable to that of the state-of-the-art learning-based method in SRPC task, SSRNet, and

[1] http://graphics.stanford.edu/data/3Dscanrep/

Metric\Method	DMC	ONet	LDIF	CEISR	SSRNet	Ours
Chamfer-L_1	1.17	0.79	0.40	0.41	0.24	**0.20**
NC	0.848	0.895	–	0.902	**0.967**	**0.967**

Table 1: Quantitative results on ShapeNet. We evaluate methods with the Chamfer-L1 distance (lower is better) and Normal Consistency (higher is better).

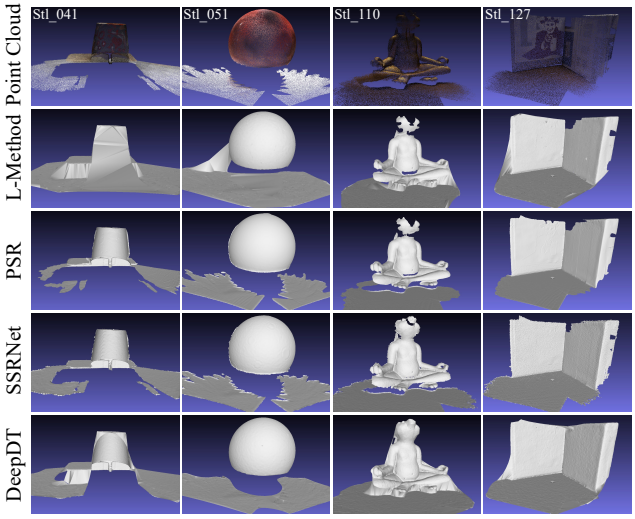

Figure 4: Qualitative results on DTU test scans

our method performs the best among all mentioned methods with the evaluation metric of Chamfer-L_1.

Figure 3 shows that our method outperforms ONet in recovering shape details, especially highlighting the ability to retain as many details as possible on extremely complex objects. We can find that ONet can still barely cope with simple topologies well, but its performance on complex topologies is hard to satisfy us who expect it to perform better. This is mainly because the use of the global latent vector encoder makes ONet lose a lot of geometric details, but geometric details are essential for reconstructing complex topologies. This also reminds us that most learning-based methods that encode the entire shape using a global latent vector are facing a non-negligible problem, that is, the geometry details can be captured by the feature space which is limited by the fixed size of the latent feature vector. We cannot solve this problem simply from the perspective of downsampling the input.

Results on 3D Scans of Larger Scales

In this section, we train and test DeepDT on DTU to verify the performance of our method on large 3D scan dataset. For each scan, we still use PSR (octree depth=10) with trimming value 8 to get labels of reference locations. We focus on whether the network can still maintain good reconstruction performance without using a sufficiently dense input. We do not use the entire point cloud, just a sample of two hundred thousand. Here we also cite traditional surface reconstruction method (L-Method) and evaluate its performance on the same data. Although noise is inevitable in large 3D scanning datasets, we still apply Gaussian noise with zero mean and standard deviation 0.001 to the point clouds in DTU. Since L-Method needs to obtain visible sight information from the process of generating a point cloud from the depth map fusion, we apply the same noise to the depth map that it uses.

Metric\Method	L-Method	PSR	SSRNet	Ours
DC-Mean	0.38	0.35	0.30	0.37
DC-Var.	0.60	0.44	0.09	0.15
CD-Mean	1.21	1.17	1.46	0.68

Table 2: Quantitative results on DTU. The lower the DTU Completeness (DC), the better. The Chamfer Distances (CD, lower is better) are in units of 10^{-8}.

Figure 4 demonstrates that the two traditional methods (PSR and L-Method) and the learning-based method (SSRNet) all generally preserve the geometry structure and complete the task well. Compared with L-Method, our method can avoid generating too many large triangular patches (Stl_051), which should not be presented in the final surface and should be removed. Furthermore, one notable detail is that DeepDT can adaptively fill holes that need to be filled to a certain extent, rather than overfilling or remaining holes (Stl_127, Stl_110). Especially for open scene reconstruction tasks, our method can well avoid the phenomenon that some strange handles that should have been removed are not removed due to the wrong labeling of the tetrahedrons inside the surface. It can be seen in the first column of Figure 4 that L-method cannot handle the inner surface of an open scene well, leaving many triangular patches that close the open scene to be removed. However, DeepDT reconstructs a smooth inner surface and reproduces the original shape of the open scene well. This greatly optimizes the visual effects of the final surface and validates our design of DeepDT.

Table 2 shows that DeepDT gets the competitive results. It is worth mentioning that the results of PSR and SSRNet are performed on the original point cloud without sampling or adding noise (We obtain PSR and SSRNet results from SSRNet paper). This means that DeepDT maintains good performance without using dense input, which benefits the method with the opportunity to explore better efficiency.

Efficiency

We test the efficiency of DeepDT on 1 GeForce RTX 2080 Ti GPU in an Intel Xeon(R) CPU system with 40×2.20 GHz cores. Here we choose DTU that better reflects the general

Data	CD Mean			CD RMS		
	ONet	SSRNet-S	DeepDT-S	ONet	SSRNet-S	DeepDT-S
Armadillo	93.46	0.028	**0.021**	168.59	0.131	**0.028**
Bunny	94.88	0.064	**0.038**	165.44	0.134	**0.095**
Dragon	40.69	0.053	**0.029**	74.99	0.208	**0.081**

Table 3: Generalization performance test on Stanford 3D. The Chamfer Distances (CD, lower is better) are in units of 10^{-6}.

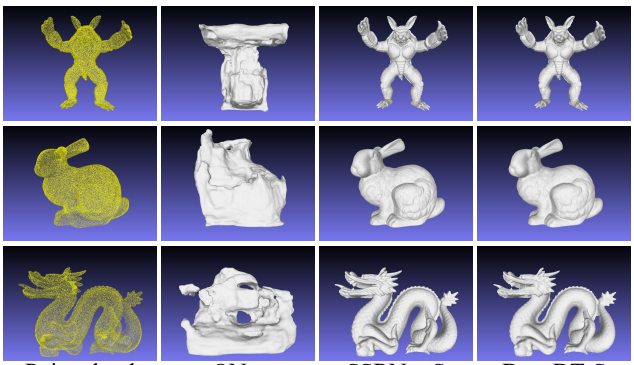

Figure 5: Qualitative performance of ONet, SSRNet-S and DeepDT-S on Stanford 3D.

efficiency. We also make a comparison with the learning-based method, SSRNet. We monitor the time consumption from inputting the point cloud to generating the final surface of each scan. It takes an average of 8.985 seconds for DeepDT to complete a scan, and 200,000 points yield an average of 1,248,801 tetrahedrons. SSRNet deals with the whole input with an average of 2.53 M points(1.20M vertices to be processed, which takes an average of 139.652 seconds). It can be accelerated by using more GPUs or reducing the input. However, the acceleration degree will not change linearly with the decrease of input points. This is mainly because SSRNet predicts the labels of vertices in an octree generated from the given point cloud. Even though the input data gets sparse, the spatial distribution of the point cloud does not change significantly, and the number of vertices will not decrease synchronously and linearly. Therefore, a sparse sample of the input will not significantly influence the efficiency of SSRNet.

Generalization Capability

For a learning-based method, it is important to have an outstanding generalization ability, which reflects whether the network can resist the threat from overfitting on a dataset. Therefore, we perform experiments on the Stanford 3D dataset to verify whether DeepDT can complete the task well on a new dataset without retraining or changing network parameters. We test DeepDT and SSRNet on Stanford 3D with the model trained on ShapeNet (DeepDT-S and SSRNet-S) to better assess the generalization ability. We still apply Gaussian noise with zero mean and standard deviation 0.001 to the point clouds in Stanford 3D. Here we also set the results of ONet as an evaluation reference. Quantitative and qualitative results are reported in Table 3 and Figure 5. The results of SSRNet are performed on the original dense point cloud without sampling or adding noise (We obtain SSRNet results from the paper of SSRNet).

We can see that ONet fails to reconstruct the details and it cannot even restore the general shape. However, model SSRNet-S and DeepDT-S are able to better restore the geometric details of the surface without retraining. It is worth mentioning that the difference between the Stanford 3D dataset and the ShapeNet dataset is very large. This reflects the powerful generalization ability of SSRNet and DeepDT, making it possible to complete the reconstruction without changing the parameters or retraining on a new dataset.

Discussion and Conclusion

In this paper, we propose DeepDT, a novel learning-based method for surface reconstruction from Delaunay triangulation. The network integrates geometry features together with graph structural information for accurate labeling of tetrahedrons. In contrast to state-of-the-art traditional reconstruction method, L-Method, DeepDT needs no visibility information and outputs accurate meshes. SSRNet adopts the strategy of dividing the input so that it can be adapted to sufficiently dense input to obtain better results. However, DeepDT is temporarily unable to use the same strategy to handle excessively large input. With its excellent performance, DeepDT just uses a sample of 200,000 points instead of the entire input to obtain competitive results in a more efficient way than SSRNet. Both DeepDT and SSRNet can complete the reconstruction of dense input well, but DeepDT is also good at dealing with the situation where there is not enough dense input. We have also noticed that when the input data is not dense enough, DeepDT witnessed a drop on texture finesse and clarity. An interesting direction could be to improve the visual effect of the DeepDT when maintaining good quantitative evaluation results. We hope that our method will inspire more learning-based methods that no longer simply adapt the network to all the input data or reduce the input at the expense of accuracy to cope with reconstruction tasks, but focus on the quality, efficiency and generalization ability, making great contributions to daily applications.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant 61772213, Grant 61991412 and Grant 91748204.

References

Amenta, N.; and Bern, M. 1999. Surface Reconstruction by Voronoi Filtering. *Discrete and Computational Geometry* 22(4): 481–504.

Boissonnat, J.-D.; Devillers, O.; Teillaud, M.; and Yvinec, M. 2000. Triangulations in CGAL. In *Proceedings of the sixteenth annual symposium on Computational geometry*, 11–18.

Cao, Y.-P.; Liu, Z.-N.; Kuang, Z.-F.; Kobbelt, L.; and Hu, S.-M. 2018. Learning to Reconstruct High-quality 3D Shapes with Cascaded Fully Convolutional Networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 616–633.

Carr, J. C.; Beatson, R. K.; Cherrie, J. B.; Mitchell, T. J.; Fright, W. R.; McCallum, B. C.; and Evans, T. R. 2001. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 67–76. ACM.

Chang, A. X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* .

Curless, B.; and Levoy, M. 1996. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 303–312.

Dai, A.; Ruizhongtai Qi, C.; and Nießner, M. 2017. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5868–5877.

Fuhrmann, S.; and Goesele, M. 2014. Floating scale surface reconstruction. *ACM Transactions on Graphics (ToG)* 33(4): 46.

Genova, K.; Cole, F.; Sud, A.; Sarna, A.; and Funkhouser, T. 2020. Local Deep Implicit Functions for 3D Shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Groueix, T.; Fisher, M.; Kim, V. G.; Russell, B. C.; and Aubry, M. 2018. A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 216–224.

Guennebaud, G.; and Gross, M. 2007. Algebraic point set surfaces. In *ACM Transactions on Graphics (TOG)*, volume 26, 23. ACM.

Hiep, V. H.; Keriven, R.; Labatut, P.; and Pons, J.-P. 2009. Towards high-resolution large-scale multi-view stereo. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 1430–1437. IEEE.

Holzmann, T.; Maurer, M.; Fraundorfer, F.; and Bischof, H. 2018. Semantically aware urban 3d reconstruction with plane-based regularization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 468–483.

Hoppe, C.; Klopschitz, M.; Donoser, M.; and Bischof, H. 2013. Incremental Surface Extraction from Sparse Structure-from-Motion Point Clouds. In *BMVC*, 94–1.

Hoppe, H.; Derose, T.; Duchamp, T.; Mcdonald, J. W.; and Stuetzle, W. 1992. Surface reconstruction from unorganized points 26(2): 71–78.

Jancosek, M.; and Pajdla, T. 2011. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*, 3121–3128. IEEE.

Jancosek, M.; and Pajdla, T. 2014. Exploiting Visibility Information in Surface Reconstruction to Preserve Weakly Supported Surfaces. *International Scholarly Research Notices* 2014: 798595–798595.

Jensen, R.; Dahl, A.; Vogiatzis, G.; Tola, E.; and Aanæs, H. 2014. Large scale multi-view stereopsis evaluation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 406–413. IEEE.

Kazhdan, M.; Bolitho, M.; and Hoppe, H. 2006. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7.

Kazhdan, M.; and Hoppe, H. 2013. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)* 32(3): 29.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* .

Labatut, P.; Pons, J.-P.; and Keriven, R. 2007. Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In *2007 IEEE 11th international conference on computer vision*, 1–8. IEEE.

Labatut, P.; Pons, J.-P.; and Keriven, R. 2009. Robust and efficient surface reconstruction from range data. In *Computer graphics forum*, volume 28, 2275–2290. Wiley Online Library.

Levin, D. 2004. Mesh-independent surface interpolation. In *Geometric modeling for scientific visualization*, 37–49. Springer.

Liao, Y.; Donné, S.; and Geiger, A. 2018. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2916–2925.

Lorensen, W. E.; and Cline, H. E. 1987. Marching cubes: A high resolution 3D surface construction algorithm. In *ACM siggraph computer graphics*, volume 21, 163–169. ACM.

Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; and Geiger, A. 2019. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4460–4470.

Mi, Z.; Luo, Y.; and Tao, W. 2020. SSRNet: Scalable 3D Surface Reconstruction Network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Park, J. J.; Florence, P.; Straub, J.; Newcombe, R.; and Lovegrove, S. 2019. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 165–174.

Poursaeed, O.; Fisher, M.; Aigerman, N.; and Kim, V. G. 2020. Coupling Explicit and Implicit Surface Representations for Generative 3D Modeling. *ECCV* .

Riegler, G.; Osman Ulusoy, A.; and Geiger, A. 2017. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3577–3586.

Riegler, G.; Ulusoy, A. O.; Bischof, H.; and Geiger, A. 2017. Octnetfusion: Learning depth fusion from data. In *2017 International Conference on 3D Vision (3DV)*, 57–66. IEEE.

Turk, G.; and O'Brien, J. F. 2002. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics (TOG)* 21(4): 855–873.

Vu, H.-H.; Labatut, P.; Pons, J.-P.; and Keriven, R. 2011. High accuracy and visibility-consistent dense multiview stereo. *IEEE transactions on pattern analysis and machine intelligence* 34(5): 889–901.

Zhou, Y.; Shen, S.; and Hu, Z. 2019. Detail Preserved Surface Reconstruction from Point Cloud. *Sensors* 19(6): 1278.