3-D Contour Deformation for the Point Cloud Segmentation

Sheng Xu, Wen Han, Weidu Ye, and Qiaolin Ye

***Abstract*— The 3-D point cloud segmentation has played an important role in spatial structure analysis. Nowadays, segmentation methods either use a primitive-based strategy to fit points in predefined geometric shapes or group points based on their attributes (e.g., spatial distance). However, the required segmentation results, e.g., primitive level or object level, depend on the application. Therefore, this letter develops a semiautomatic method to extract contours for the users' desired segmentation. First, we initialize a 3-D closed curve for the target. Second, we calculate the internal and external force based on the proposed vector flow to deform the curve. The deformation equation is solved based on the Euler equation and calculated iteratively. Finally, the curve is converged as object contours. After one removes contours, those disjoint points are grouped as the users' desired instances. Experiments are conducted on various point clouds to demonstrate the effectiveness in terms of accuracy and consistency. Our quantitative evaluation outperformed selected primitive- and object-based methods, which presents a new viewpoint to the point cloud processing.**

***Index Terms*— 3-D point clouds, contour deformation, energy minimization, segmentation, semiautomatic, vector flow.**

I. Introduction

POINT cloud segmentation is the process of partitioning input data into multiple disjoint regions and has been a fundamental work in the 3-D structure study, e.g., geometric analysis [1], point cloud registration [2], and tree delineation [3], [4].

Commonly used global segmentation methods aim to find the optimal label configuration for all points according to energy functions. The method [5] chooses normalized-cut to refine segmentation results and reduces the rate of oversegmentation. The method [6] formulates the energy function using binary variables to indicate whether a point belongs to the foreground or background region. Those two graph-based methods have obtained impressive performances in the segmentation of light detection and ranging (LiDAR) point clouds but require initial numbers of objects. The method [7] proposes a new optimal-vector-field (OVF) to infer boundary cues for the plane segmentation, which seems a new perspective to the segmentation. Although deep learning methods have achieved promising results in the point cloud segmentation, supervised learning methods [8] require a training process to set labels for models. Users need to segment a large number of 3-D objects manually, which is difficult for the generalization of different scenes. Besides, one important issue has not been fully discussed yet in terms of segmentation. The definition of an individual instance is related to one's demand. For example, in building detection or extraction, an ideal instance means an individual house or apartment, but, in building modeling, an ideal element is expected to be a surface.

The motivation of this letter is to segment point clouds into individual regions based on the coordinate information by taking users' demands into consideration. We propose a semiautomatic energy-based method to extract contours from point clouds. Different from the existing segmentation methods, we split objects into different levels based on users' desires.

Manuscript received April 4, 2021; revised May 15, 2021; accepted July 18, 2021. This work was supported in part by the Natural Science Foundation of Jiangsu Province under Grant BK20200784, in part by the Natural Science Foundation of the Higher Education Institutions of Jiangsu Province under Grant 19KJB520010, and in part by China Postdoctoral Science Foundation under Grant 2019M661852. *(Corresponding author: Qiaolin Ye.)*

Sheng Xu is with the College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China, and also with the College of Landscape Architecture, Nanjing Forestry University, Nanjing 210037, China.

Wen Han and Qiaolin Ye are with the College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China (e-mail: yqlcom@njfu.edu.cn).

Weidu Ye is with the School of Computer Science and Engineering, Southeast University, Nanjing 211189, China.

Color versions of one or more figures in this letter are available at https://doi.org/10.1109/LGRS.2021.3098924.

Digital Object Identifier 10.1109/LGRS.2021.3098924

II. Method

A. Energy Function Formulation

Denote the input point clouds as $\bar{\Omega}$, and the segmentation is to decompose input data as $\bar{\Omega} = \Omega_1 \cup \Omega_2 \cup \Omega_3, \ldots, \cup\Upsilon$, where Ω_i means to be inner a disjoint region and Υ stands for contours between different regions, which is to be extracted by our model. The contour function consists of an internal term F_{int} and an external term F_{ext}. F_{int} presents the smoothness constraint for contour extractions. F_{ext} forces the current contour to surface boundaries. Inspired by the tradition snake model [9], we define a 3-D curve as $\mathbf{X}(s) = [x(s), y(s), z(s)], s \in [0, 1]$, where s means the arc length for the representation of continuous curves. The deformation energy is calculated as

$$E = \int_1^0 F_{\text{int}}(s) + F_{\text{ext}}(s)ds$$

$$= \int_1^0 \alpha|\mathbf{X'}(s)| + \beta|\mathbf{X''}(s)| + F_{\text{ext}}(s)ds \quad (1)$$

where parameters α and β aim to tune the contour's tension and rigidity based on the first $\mathbf{X'}(s)$ and second $\mathbf{X''}(s)$ derivatives of $\mathbf{X}(s)$ with respect to s, respectively.

According to the Euler equation, we treat the contour as the function of time t and s and the partial derivative of $\mathbf{X}$ with respect to t; (1) can be written as

$$x_t(s,t) = \alpha x''(s,t) - \beta x''''(s,t) + \mathbf{V} \tag{2}$$

where $\mathbf{V} = -\nabla F_{\text{ext}}$ stands for a vector flow to evaluate the external force and defined as $[u(x,y,z), v(x,y,z), w(x,y,z)]$.

To obtain cues for calculating external force, let us calculate the difference of the point density D between neighbors as

$$\begin{aligned} f(x,y,z) &= (D(\Delta x + x, \Delta y + y, \Delta z + z) - D(x,y,z))^2 \\ &= (\Delta x, \Delta y, \Delta z)\mathbf{M}(\Delta x, \Delta y, \Delta z)^\top \end{aligned} \tag{3}$$

where $\mathbf{M}$ is

$$\begin{pmatrix} \left(\frac{\partial D}{\partial x}\right)^2 & \left(\frac{\partial D}{\partial x}\right)\left(\frac{\partial D}{\partial y}\right) & \left(\frac{\partial D}{\partial x}\right)\left(\frac{\partial D}{\partial z}\right) \\ \left(\frac{\partial D}{\partial x}\right)\left(\frac{\partial D}{\partial y}\right) & \left(\frac{\partial D}{\partial y}\right)^2 & \left(\frac{\partial D}{\partial y}\right)\left(\frac{\partial D}{\partial z}\right) \\ \left(\frac{\partial D}{\partial x}\right)\left(\frac{\partial D}{\partial z}\right) & \left(\frac{\partial D}{\partial y}\right)\left(\frac{\partial D}{\partial z}\right) & \left(\frac{\partial D}{\partial z}\right)^2 \end{pmatrix}. \tag{4}$$

Equation (3) uses the difference method to calculate the gradient information. In order to calculate the information difference between points, we use the voxelization technique to reorganize input data as cubes at the size of 1 cm $\times$ 1 cm $\times$ 1 cm. Δx, Δy, and Δz are step sizes between voxels, which are set as 1 to calculate the neighbor gradient difference.

In point clouds, edges are the intersection of two surfaces. Thus, if gradients of a point are large in one and only one direction, this point is on a planar, i.e., $\lambda_1 \gg 0$, $\lambda_2 \approx 0$, and $\lambda_3 \approx 0$; if gradients of a point are large in different directions, this point tends to be on the intersection of planes, i.e., $\lambda_1 \approx \lambda_2$ and $\lambda_1 \gg \lambda_3$, or $\lambda_1 \approx \lambda_2 \approx \lambda_3$ and $\lambda_1 \gg 0$. We calculate the edge information based on the work of [10] as

$$F = \left(\frac{\lambda_1\lambda_2}{\lambda_1+\lambda_2} + \frac{\lambda_1\lambda_3}{\lambda_1+\lambda_3} + \frac{\lambda_3\lambda_2}{\lambda_3+\lambda_2}\right) \cdot (\lambda_1+\lambda_2+\lambda_3)^2. \tag{5}$$

As the edge information, F has the property that magnitudes of points are large in surface intersections and are small inner surfaces. We visualize each term of (5) in Fig. 1. As shown in Fig. 1(d), only when two of eigenvalues are large, F has a high value indicating edge points.

Now, the solution of the vector flow $\mathbf{V}$ is calculated by minimizing the energy functional

$$\begin{aligned} &\int\int\int \lambda \cdot H_0(u,v,w) + H_1(u,v,w)dxdydz \\ &\quad = \int\int\int \lambda \cdot \left(\nabla u \cdot \nabla u^\top + \nabla v \cdot \nabla v^\top + \nabla w \cdot \nabla w^\top\right) \\ &\qquad + F \cdot (\mathbf{V} - F)^2 dxdydz \end{aligned} \tag{6}$$

where $H_0(u,v,w)$ is the homogeneous term to prevent the external force from being zero when the current curve is far from edges, and $H_1(u,v,w)$ is the heterogeneous term to enlarge the external force to push the curve to contours. The coefficient λ is to balance terms. The achieved (6) helps (2) converge to contours for splitting different regions.

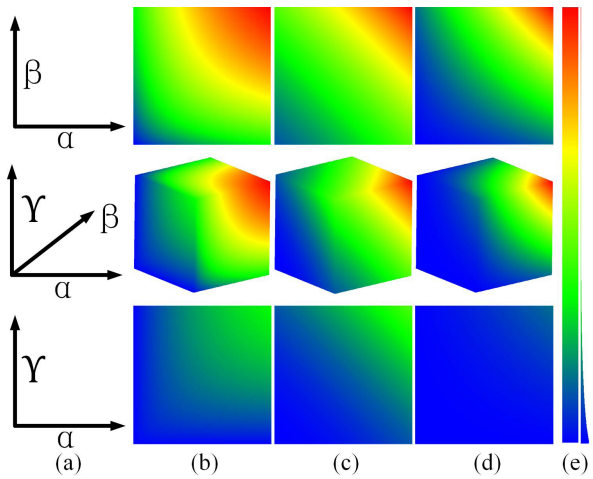

Fig. 1. Energy visualization of F in (5) at the top view, the side view, and the front view, respectively. (a) Axis of eigenvalues. (b) Visualization of the first term on the right-hand side of F. (c) Visualization of the second term on the right-hand side of F. (d) Visualization of F. (e) Bar of energy values from low to high.

B. Optimization and Implementation

This section aims to present the numerical solution to optimize (2). In the implementation, the curve is described as a point set $C = \{c_1, c_2, c_3, \ldots, c_n\}$, and each c_i stands for a point at (x_i, y_i, z_i). Now, we update the coordinate of points for updating the curve. It is worth noting that the convergence performance of curves significantly depends on the initial contour provided by the users manually. Users are required to set an initial curve that is spatially close to the demand contour. This step will increase the deformation performance, including accuracy and efficiency. The internal force is calculated by the difference method as

$$\begin{aligned} F_{\text{int}} = \alpha((c_i - c_{i-1}) - (c_{i+1} - c_i)) \\ - \beta((c_{i-2} - 2c_{i-1} + c_i) - 2(c_{i-1} - 2c_i + c_{i+1}) \\ + (c_i - 2c_{i+1} + c_{i+2})). \end{aligned} \tag{7}$$

Let

$$\begin{aligned} \mathbf{A} &= (-\beta, -\alpha + 4\beta, 2\alpha - 6\beta, -\alpha + 4\beta, -\beta) \\ \mathbf{C} &= (c_{i-2}, c_{i-1}, c_i, c_{i+1}, c_{i+2})^\top \end{aligned}$$

and based on the work of [9], we set (2) equal to the product of a step size as

$$\mathbf{A} \cdot \mathbf{C}_t + \mathbf{V}_{t-1} = -\kappa(\mathbf{C}_t - \mathbf{C}_{t-1}). \tag{8}$$

Therefore

$$\mathbf{C}_t = (\mathbf{A} + \kappa\mathbf{I})^{-1}(\kappa \cdot \mathbf{C}_{t-1} - \mathbf{V}_{t-1}). \tag{9}$$

Again, let us use the Euler equation on (6) for the minimization of $\mathbf{V}$, and u is calculated as

$$\frac{\partial H_1}{\partial u} - \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\partial H_0}{\partial u_x}\right) = (u - F) \cdot F - \lambda\nabla(\nabla u) = 0. \tag{10}$$

Note that the solution of (6) is decoupled, and we can achieve v and w separately in the same way. Let us

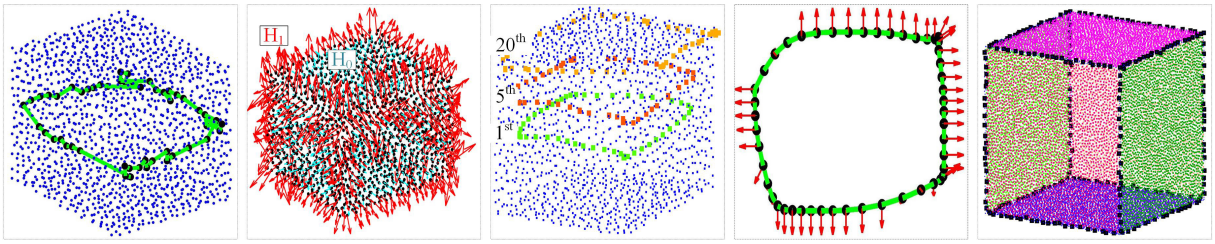

Fig. 2. Contour deformation. (a) Input point clouds and the initial curve. (b) Vector flow in homogeneous H_0 and heterogeneous regions H_1. (c) Update of a curve at the first, fifth, and 20th iterations. (d) Formulated force at boundaries. (e) Final segmentation results. Disjoint surfaces visualized by different colors.

treat u, v, and w as functions of time and update them as

$$u_t(x,y,z) = u_{t-1}(x,y,z) + \lambda\nabla(\nabla u_{t-1}(x,y,z)) - (u_{t-1}(x,y,z) - F(x,y,z)) \cdot F(x,y,z). \quad (11)$$

In the segmentation implementation, we first update the coordinate of the curve iteratively to arrive at contours, and then, we remove points from contours. Finally, we group disjoint regions as individual desired instances based on the Euclidean distance directly.

III. Experiments and Evaluations

A. Results of the Proposed Method

Experimental scenes include point clouds of synthetic data, red, green, blue, depth (RGBD) data, airborne laser scanning (ALS) data, and mobile laser scanning (MLS) data. Synthetic point clouds come from the 3-D model in SketchUp (https://www.sketchup.com/). As shown in Fig. 2(a), first, we initialize a curve based on the users' desires, i.e., each surface in this case. Second, we calculate the internal and external forces to update the curve. Fig. 2(b) visualizes vector flows for H_0 and H_1 in homogeneous and heterogeneous regions, respectively. Fig. 2(c) demonstrates the update of a curve at the first, fifth, and 20th iterations. Fig. 2(d) shows the force for updating the curve. Finally, we achieve the contour of the top surface. Similarly, we set initial curves for the rest of the surfaces and group disjoint points based on the converged contours, as shown in Fig. 2(e).

RGBD point clouds come from the Scenes Dataset v.2 [11]. This dataset consists of common indoor scenes, e.g., chairs, coffee tables, and sofa. The input scene is a point cloud set created by aligning a set of video frames, as shown in Fig. 3(a). The challenge is to split points into different scales of instances. As shown in Fig. 3(b), individual object instances are segmented successfully. In the existing methods, surfaces of sofas are easy to be oversegmented as different instances, which are not reasonable in 3-D scene understanding. In our experiment, chairs, sofas, and desks are segmented as individual instances (furniture), while scatter points are regarded as the background and grouped as one instance.

ALS point clouds come from the Dublin project (doi: 10.17609/N8MQ0N). Data were obtained at an average flying altitude of 300 m. As shown in Fig. 4, we split each plane as an individual instance. The challenge is to split "V" type roofs into one unit rather than conducting the plane segmentation directly, which is simple for us by setting desired curves.

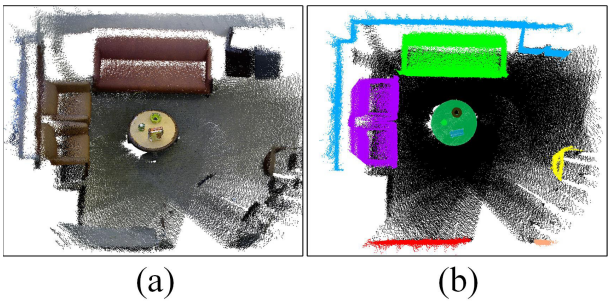

Fig. 3. Performance on the RGBD point clouds. (a) Input indoor scenes. Colors are achieved by the registration of points and pictures. (b) Segmentation results of RGBD data.

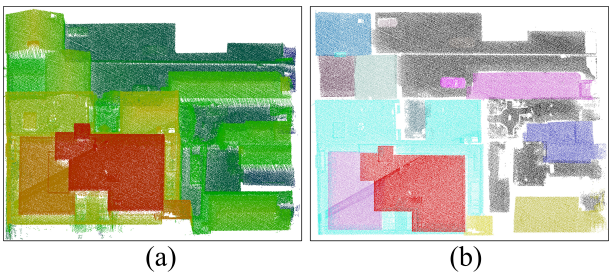

Fig. 4. Performance on airborne LiDAR point clouds. (a) Input outdoor scenes. Color means the elevation of points. (b) Segmentation results of ALS data.

MLS point clouds come from [10]. This experiment is more complex than the abovementioned cases because data are collected with noise and present as different geometric shapes and scales. We choose different methods to show our superiority qualitatively. Fig. 5(a) demonstrates the input scene and the manual ground truth. From Fig. 5(b) to (e) are the methods of 3DNCut [5], MinCut [6], plane extraction by agglomerative clustering (PEAC) [12], OVF [7], and ours. MinCut is from PointCloudLibrary (www.pointclouds.org/); 3DNCut is extended from the normalized-cut (www.cis.upenn.edu/∼jshi/software/). PEAC is achieved based on the software of [12] (www.merl.com/research/). Our results are shown in Fig. 5(f). Our method succeeds in splitting *HouseSet* as

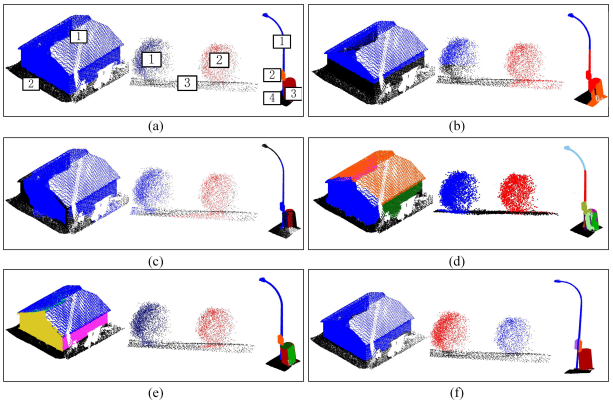

Fig. 5. Performance on mobile LiDAR point clouds. (a) Input street scenes and their ground-truth labels. (b) Results of 3DNCut [5]. (c) Results of MinCut [6]. (d) Results of PEAC [12]. (e) Results of OVF [7]. (f) Results of ours.

one individual instance and group *TrafficlightSet* into the pole, traffic sign, and trash bin separately.

There are four energy-based methods in the comparison. 3DNcut and MinCut are based on the graph theory optimization to split objects. OVF is based on the optimization of normal vector flow to distinguish between connectivity and nonconnectivity regions. PEAC is based on the optimization of the proximity matrix to group points in an unsupervised clustering framework. Our method is based on the minimization of the internal and external terms in the functional globally. 3DNCut and MinCut are two instance segmentation methods. As shown in the comparison, our method is better than those instance segmentation methods visually in terms of completeness and correctness. This is because of our superiority in splitting overlapping regions between different objects. PEAC and OVF are two plane segmentation methods, which performs well in addressing the segmentation of overlapping regions. However, the consistency of those methods is lower than ours because of the ignorance of users' demands. Quantitative evaluation of the accuracy and consistency is shown in Section III-B.

B. Quantitative Comparison and Evaluation

Suppose that the multiobject segmentation result is $R = \{r_1, r_2, r_3, \ldots, r_i\}$ and the manual ground truth is $G = \{g_1, g_2, g_3, \ldots, g_j\}$. Each r_i or g_j means the point set of a segment. There are i segments in R and j segments in G. The abovementioned two point sets R and G mean the achieved segmentation result and ground truth, respectively. The evaluation of our multiobject segmentation is based on the label accuracy and consistency.

The segmentation completeness p and correctness r are defined as

$$p = \frac{1}{i}\sum_{n=1}^{i}\left(\frac{\max_{m=1}^{j}|g_m \bigcap r_n|}{|r_n|}\right)$$
$$r = \frac{1}{j}\sum_{n=1}^{j}\left(\frac{\max_{m=1}^{i}|r_m \bigcap g_n|}{|g_n|}\right) \quad (12)$$

where "||" means the cardinality of a set. Both the criteria p and r belong to a set-based evaluation method ranging from 0 to 1. In order to balance the completeness and correctness, the segmentation accuracy is based on n_d and F_1-score n_f to measure the difference of points between G and R as

$$n_d = \min(p, r), n_f = \frac{2 \times (p \cdot r)}{(p + r)}. \quad (13)$$

Besides the set-based assessment for the accuracy evaluation, we introduce two metrics for the consistency evaluation, which can evaluate our performance on users' desires. The first is a pair-based assessment to consider statistics over pairs of items by measuring the consistency of points in R with respect to G. The introduced probabilistic rand index (PRI) [13] ranges from 0 to 1 and is calculated as

$$\frac{1}{\binom{|C|}{2}}\sum_{\{c_1,c_2\}\in C}\begin{cases}1, & \text{if } \{c_1,c_2\}\in r_i \text{ and } \{c_1,c_2\}\in g_j\\ 1, & \text{if } \{c_1,c_2\}\notin r_i \text{ and } \{c_1,c_2\}\notin g_j\\ 0, & \text{others}\end{cases} \quad (14)$$

where c_1 and c_2 are any two different points from input point clouds.

The second is an information-based assessment to measure the amount of label consistency information in R that is not contained in G. The introduced variation of information (VI) [14] ranges from $e^{-(\log_2 i + \log_2 j)}$ to 1 and is calculated as

$$n_{vi} = e^{-\sum_{n=1}^{i}(f_H(r_n)+f_H(g_J)-2\times f_I(r_i,g_J))} \quad (15)$$

where $J = \arg\max_{n=1}^{j}(f_I(r_i, g_n))$, f_H is the entropy function, and f_I is the mutual information function.

To show the superiority of the proposed algorithm, we conduct the quantitative comparison using MLS point points. In this experiment, $k = 40$, $\alpha = 0.5$, $\beta = 0.1$, $\kappa = 1$, and $\lambda = 0.2$. 3DNcut tends to divide the input scene into several parts evenly, which fails to segment object instances with varied sizes. MinCut requires human–computer interaction to set the optimal radius size for the foreground. PEAC obtains plane features effectively but is difficult to deal with complex surfaces. OVF segments point into different planes well but require a postprocessing step for instance segmentation. The proposed algorithm succeeds in detecting object instances with respect to the ground truth. Corresponding to qualitative results in Fig. 5(f), Table I shows the average accuracy and consistency of methods on input scenes, including *HouseSet*, *BushSet*, and *TrafficlightSet*. Comparison results show that our method is more accurate than all compared methods, especially in terms of n_{pri} and n_{vi}, which highly depends on users' desires.

TABLE I
QUANTITATIVE COMPARISON OF THE ACCURACY AND CONSISTENCY (%)

Index	3DNCut	MinCut	PEAC	OVF	Proposed
n_d	79.12	79.32	80.87	90.23	91.13
n_f	82.15	84.51	85.50	93.11	93.15
n_{pri}	51.50	73.10	59.82	58.04	80.89
n_{vi}	31.78	45.83	16.24	10.72	61.00

C. Discussion

There are five key parameters required to be set by users in the segmentation, namely, k, α, β, κ, and λ. k depends

on the density of input data for setting neighbor points. α is the elasticity parameter to control the consistency of curves, and β is the rigidity parameter to constrain the smoothness of curves. Those two parameters depend on the geometric shape of objects. κ is the viscosity parameter for tuning the update speed. λ relies on the ratio of homogeneous and heterogeneous regions for balancing terms.

For the purpose of the parameter analysis, we range all parameters from -10% to 10% with respect to the suggested values and observe deformation results. The analysis is conducted by floating one parameter and fixing the rest of the parameters. If one increases the κ, the algorithm will be converged rapidly but may lose boundary details. We suggest a small λ for highlighting the weight of external force on heterogeneous regions. The proposed algorithm is not sensitive to the number of neighbors because our external force can distinguish points from homogeneous and heterogeneous regions based on functional minimization globally. α and β decide the curve shape and are required to be tuned by users based on input scenes.

In terms of the computational complexity, in each iteration, the internal term is calculated in $\mathcal{O}(N)$, where N is the number of voxels. In the calculation of external terms, we are required to achieve eigenvalues, as shown in (5), at the complexity of $\mathcal{O}(N^3)$. Although the algorithm is iterative, the eigenvalue of each voxel is stable. Therefore, the computational complexity is $\max\{\mathcal{O}(N^3), \mathcal{O}(k \cdot N)\}$, where k is the number of iterations and usually far less than N^2.

Experimental scenes contain types of laser scanning point clouds to test our generalization, including RGBD point clouds, ALS point clouds, and MLS point clouds. RGBD point clouds are widely used in the indoor scene analysis. ALS and MLS point clouds are frequently used in the outdoor scene analysis, e.g., building, traffic facilities, and vegetation. RGBD, ALS, and MLS point clouds are at the surface density of about 500, 300, and 1000 points/m^2, respectively. Since our results depend on the topological information, we are not sensitive to densities. However, if there are gaps caused by data incompleteness or occlusion, the proposed algorithm regards gaps as boundaries incorrectly, which means the spatial distribution is important to contour detection.

For drawbacks, we have the following issues in the implementation. First, the proposed method is not fully automatic. To obtain objects based on the users' desires, one is required to set curves manually. The accuracy and convergence speed highly depend on the initial curve. Second, in the case of boundaries that are not closed, the proposed method tends to split the target into several instances, which requires a merging process manually. Third, the proposed method fails to segment linear objects, e.g., power lines.

IV. Conclusion

This work proposes a contour extraction method to segment objects from point clouds. We update an initial curve by optimizing the deformation function and converge the curve at contours. Our novelty lies in splitting objects based on the users' desires, which is suitable for 3-D point cloud scenes. The validation is performed on different kinds of point clouds, and evaluation is based on various metrics. In terms of the correctly segmented points, we achieve the F_1-score of 93.15% showing a good balance between completeness and correctness. Besides, we use metrics to measure the consistency of points and labels in our results with respect to the reference, which demonstrates the similarity between results and the users' desires. The achieved point and label consistencies are 80.89% and 61.00%, respectively. Experiment results prove that our accuracy and consistency are better than the compared methods, which indicates that the proposed model can benefit more applications in the area of point cloud processing.

In future work, we plan to focus on the improvement of automatic point cloud segmentation. Promising techniques are given as follows.

1) Add prior knowledge of rules, topology, or geometric regularities information for setting the initial curve automatically.
2) Try supervised learning methods to find contours for the segmentation in complex scenes.

References

[1] S. Xia, D. Chen, R. Wang, J. Li, and X. Zhang, "Geometric primitives in LiDAR point clouds: A review," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 685–707, 2020.

[2] Z. Dong *et al.*, "Registration of large-scale terrestrial laser scanner point clouds: A review and benchmark," *ISPRS J. Photogramm. Remote Sens.*, vol. 163, pp. 327–342, May 2020.

[3] T. Yun *et al.*, "Individual tree crown segmentation from airborne LiDAR data using a novel Gaussian filter and energy function minimization-based approach," *Remote Sens. Environ.*, vol. 256, Apr. 2021, Art. no. 112307.

[4] T. Yun *et al.*, "Simulation of multi-platform LiDAR for assessing total leaf area in tree crowns," *Agricult. Forest Meteorol.*, vols. 276–277, Oct. 2019, Art. no. 107610.

[5] Y. Yu, J. Li, H. Guan, C. Wang, and J. Yu, "Semiautomated extraction of street light poles from mobile LiDAR point-clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 3, pp. 1374–1386, Mar. 2014.

[6] A. Golovinskiy and T. Funk, "Min-cut based segmentation of point clouds," in *Proc. IEEE 12th Int. Conf. Comput. Vis. Workshops, ICCV Workshops*, Sep. 2009, pp. 39–46.

[7] S. Xu, R. Wang, H. Wang, and R. Yang, "Plane segmentation based on the optimal-vector-field in LiDAR point clouds," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, May 18, 2020, doi: 10.1109/TPAMI.2020.2994935.

[8] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[9] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comput. Vis.*, vol. 1, no. 4, pp. 321–331, 1988.

[10] S. Xu, R. Wang, and H. Zheng, "Road curb extraction from mobile LiDAR point clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 996–1009, Feb. 2017.

[11] K. Lai, L. Bo, and D. Fox, "Unsupervised feature learning for 3D scene labeling," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 3050–3057.

[12] C. Feng, Y. Taguchi, and V. R. Kamat, "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 6218–6225.

[13] R. Unnikrishnan, C. Pantofaru, and M. Hebert, "Toward objective evaluation of image segmentation algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 929–944, Jun. 2007.

[14] M. Meila, "Comparing clusterings by the variation of information," in *Colt*, vol. 3. Seattle, WA, USA: Univ. Washington, 2003, pp. 173–187.