SCIENCE CHINA
Information Sciences

• MOOP •

December 2020, Vol. 63 224101:1–224101:3
https://doi.org/10.1007/s11432-018-9689-9

Semantic part segmentation of single-view point cloud

Haotian PENG, Bin ZHOU*, Liyuan YIN, Kan GUO & Qinping ZHAO

State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, Beijing 100191*, China*

Received 15 September 2018/Accepted 30 November 2018/Published online 28 September 2020

Citation Peng H T, Zhou B, Yin L Y, et al. Semantic part segmentation of single-view point cloud. Sci China Inf Sci, 2020, 63(12): 224101, https://doi.org/10.1007/s11432-018-9689-9

As a classic topic in computer graphics, the semantic part segmentation of 3D data is helpful for 3D part-level editing and modeling. Single-views point cloud is the raw format of 3D data. Giving each point a semantic annotation in single-view point cloud, i.e., single-view point cloud semantic part segmentation, is meaningful and challenging.

In the last decades, many studies have focused on extracting effective geometric descriptors or training high-level features to perform this semantic segmentation task [1]. However, these features have primarily been extracted according to the 3D shape topology. Most traditional methods are inapplicable to 3D point cloud. Few studies have focused on part-level point cloud semantic segmentation. A few researchers have completed this task based on deep learning [2–4]. However, such methods often use multi-view synthetic point clouds as input, rather than single-view point cloud.

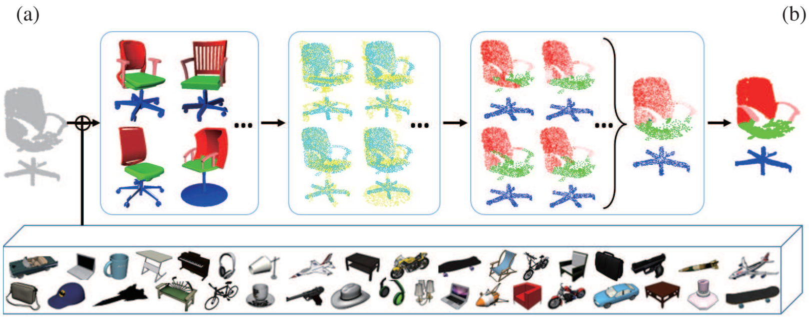
To address this problem, we propose transferring semantic annotations from synthetic models to single-view point cloud. The pipeline of our method is shown in Figure 1. After establishing a database of 3D synthetic CAD models, we matched single-view point cloud with all the models in the database to find models for guidance. Then, we registered the single-view point cloud with the guidance models and built point-level correspondences between them. Finally, we transferred the annotations from the guidance models to the input point cloud. We tested our method on both, synthetic and real scanned single-view point cloud datasets. The results indicate that our method can effectively segment incomplete point cloud with higher annotation accuracy and less calculation time in comparison to the existing state-of-the-art methods.

Category independent matching. Using the single-view point cloud T as the input, we implemented an orthogonal projection and extracted features from the projection referring to 3D-Model-Retrieval approach [5]. We also extracted features from the orthogonal projections of the universal set of the matching database. This approach measures the similarity between the 3D models via visual similarity of their projection. A total of 20 orthogonal projections of an object are encoded via both Zernike moments and Fourier descriptors as features required for later matching. These 20 projection views are distributed uniformly and can roughly represent the shape of a 3D model. Note that we used only one projection of the point cloud T to extract descriptors because our input is a single-view point cloud.

We compared the descriptor of T to that of each orthogonal projection in the database in Euclid space and selected the minimum dissimilarity value to be the difference between T and one model in the database. We ranked the values of the obtained difference to accquire the matching list of T. The matching process is category independent. We chose the top 10 models on the matching list as the guidance models.

Point cloud registration. We sampled the guid-

* Corresponding author (email: zhoubin@buaa.edu.cn)



(b)

Category	PointNet	shapePFCN	ICP	Ours
Airplane	79.2	90.3	81.6	88.3
Bag	93.1	94.6	90.8	95.5
Cap	83.1	94.5	77.3	82.3
Car	77.5	86.7	67.3	72.1
Chair	90.6	82.9	85.7	91.3
Earphone	83.4	84.9	68.4	76.3
Guitar	92.7	91.8	94.3	94.9
Knife	89.9	82.8	83.4	91.0
Lamp	83.7	78.0	76.2	78.4
Laptop	96.1	95.3	79.4	96.5
Motorbike	70.3	87.0	69.9	73.5
Mug	94.6	96.0	87.1	96.2
Pistol	90.5	91.5	86.8	92.0
Rocket	70.9	81.6	74.5	80.9
Skateboard	91.1	91.9	93.6	92.5
Table	91.7	84.8	86.7	93.9

Figure 1 (Color online) (a) Architecture of our approach; (b) comparison using the synthetic dataset.

ance models into point cloud, denoted S, using the triangle faces. Ideally each point in T has a corresponding point in S. They have the same part label and are the shortest distance from each other.

We calculated the point-to-plane distance between S and the rotated T to select the best rotation matrix: $\text{dist} = \sum_t ((M \times T_i - S_i) \times n_i)^2$, wherein T_i and S_i are the coordinates of the point cloud, M is the rotation matrix from the point cloud T to the point cloud S, and n_i is the unit normal of S.

By shrinking the basic angle, we gradually obtained the rotation angle for each axls; therefore, we obtained the rotation matrix from T to S: $R_{\text{angle}} = R_x(\alpha_x \times \theta_o) \times R_y(\alpha_y \times \theta_o) \times R_z(\alpha_z \times \theta_o)$, wherein R_x is the rotation matrix, θ_o is the basic angle, α_x is the multiple coefficients, and R_{angle} is the final rotation angle.

It is observed that the rotation angles are more distributed in the upper hemisphere in the world coordinate system. We do not need to traverse all rotation angles at the three axes. Accordingly, we narrowed the range and used progressive refinement to effectively reduce the calculation cost.

After calculating the rotation matrix, we used binary search to find the transformed vector from the rotated T to S. Because the point clouds S and T are scaled into (1, 1) in the coordinate system, we parallelly translate S in (1.5, 1.5) coordinate space to find the transformed vector. In this retrieval process we cannot obtain guidance models that are extremely similar to the input point cloud in topology; however we can obtain the annotations transferred from different guidance models.

After obtainning the rotation and transform matrices, we can establish a point-level correspondence between S and T. Every point in T has a corresponding point in S whose distance is the shortest.

Annotation transfer. One point carries only one semantic annotation; however, each point cloud T has N top guidance models. To obtain the final annotation transfer result we used the energy function: $E(L) = \min(E_d(L) + E_n(L))$ where L is an annotation transformation between a point pair (p, q) in T and a guidance point cloud S.

E_d is the distance cost. Not all points in T can be labeled using only one guidance model; however these points can obtain labels from other guidance models or from their neighbors. For this purpose, we used a threshold:

$$E_d(L) = \begin{cases} +\infty, & \text{if } D(p,q) \geqslant \text{th1}, \\ \min_{q \epsilon S} D(p,q), & \text{if } D(p,q) < \text{th1}, \end{cases} \tag{1}$$

where D is the Euclidean distance.

E_n is the normal cost. It is the binarization of normal angles between the point pairs. This item ensures that the point in the segment boundary has the correct label transfer. ang is the angle between these two normals.

$$E_n(L) = \begin{cases} +\infty, & \text{if } \text{ang}(p,q) \geqslant \text{th2}, \\ \min_{q \epsilon S} \text{ang}(p,q), & \text{if } \text{ang}(p,q) < \text{th2}. \end{cases} \tag{2}$$

We used a point-level graph cut optimization to refine mistakes near the boundary. Using a k-nearest search we built a point adjacency graph with each node in the graph representing a label. To refine the existing label L to L', we used the following graph cut algorithm: $E'(L') = E'_d(L') + \lambda_s E'_s(L')$. The factors λ_s is set to 0.5. The data cost E'_d is set to preserve the original segment label.

$$E'_d(L') = \begin{cases} 0, & \text{if } L(p) = L'(p), \\ \frac{D(p,b(p))}{\max_i D(i,b(p))}, & \text{if } L(p) \neq L'(p), \end{cases} \tag{3}$$

where $D(p, b(p))$ is the distance from point p (with label L') to the same label boundary (L'), and $D(i, b(p))$ is the distance of all points from label L' to the label boundary.

The smooth term $E'_s(L')$ is set to constrain the neighboring points, tending them to share the same label.

$$E'_s(L') = \begin{cases} 0, & \text{if } L'(p) = L'(p'), \\ \text{ang}(N_p, N_{p'})/\pi, & \text{if } L'(p) \neq L'(p'), \end{cases} \tag{4}$$

where p and p' are neighbors in T searched for via a k-nearest search.

Experiments and comparisons. We built a dataset of synthetic single-view point clouds to verify our method. The database was obtained from the ShapeNet core [6]. The database contains 16 model categories, which include more than 15000 3D CAD models, including cars, guitars, airplanes, laptops and tables. In addition, we collected 4 model categories from the COSEG dataset [7] to enrich the diversity of the database, which include vases, chairs, and lamps.

We compared our approach to PointNet [3] and shapePFCN [8]. In addition, we compared the ICP algorithm [9] to our registration algorithm. Figure 1(b) shows that we achieve more desirable results in a the majority of categories.

We also developed a real scanned dataset called SVPCD (single-view point cloud dataset). Additional contents can be found in the supplementary documents.

Conclusion and discussion. In this study, we proposed a part-level semantic segmentation annotation method for single-view point cloud using the guidance of labeled synthetic models. We performed experiments and demonstrated that our method could achieve excellent results with multiple datasets. Moreover, we established a SVPCD dataset captured from the real world to prove the effectiveness of our approach.

Even though single-view point cloud always have holes and missing structures, our matching and registration processes ensure that the semantic annotations are successfully transferred from synthetic models to the point cloud. However, our method includes several limitations. First, our method dependes on labeled synthetic models, which are difficult to establish. Second, we only focus on single object part-level semantic segmentation problem and are not able to apply our method to the 3D scene data.

Many interesting future extensions are possible based on our work. For example, solving the 3D scene point cloud semantic segmentation problem or taking the RGB color images into consideration to assist understanding point cloud.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61502023, 61532003).

Supporting information Videos and other supplemental documents. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Kalogerakis E, Hertzmann A, Singh K. Learning 3D mesh segmentation and labeling. ACM Trans Graph, 2010, 29: 102

2 Charles R Q, Su H, Kaichun M, et al. Pointnet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, 2017. 77–85

3 Qi C R, Li Y, Su H, et al. PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Proceedings of the 31st Conference on Neural Information Processing Systems, 2017. 1–10

4 Li Y Y, Bu R, Sun M C, et al. PointCNN: convolution on X-transformed points. In: Proceedings of the 32nd Conference on Neural Information Processing Systems, 2018. 1–11

5 Chen D Y, Tian X P, Shen Y T, et al. On visual similarity based 3D model retrieval. Comput Graph Forum, 2003, 22: 223–232

6 Chang A X, Funkhouser T, Guibas L J, et al. Shapenet: an information-rich 3D model repository. 2015. ArXiv: 1512.03012

7 van Kaick O, Tagliasacchi A, Sidi O, et al. Prior knowledge for part correspondence. Comput Graph Forum, 2011, 30: 553–562

8 Kalogerakis E, Averkiou M, Maji S, et al. 3D shape segmentation with projective convolutional networks. In: Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR), 2017. 6630–6639

9 Rusinkiewicz S, Levoy M. Efficient variants of the ICP algorithm. In: Proceedings of International Conference on 3D Digital Imaging and Modeling, 2001. 145–152