Object tracking method based on joint global and local feature descriptor of 3D LIDAR point cloud

Qishu Qian (钱其姝)[1,2], Yihua Hu (胡以华)[1,2,*], Nanxiang Zhao (赵楠翔)[1,2], Minle Li (李敏乐)[1,2], Fucai Shao (邵福才)[3], and Xinyuan Zhang (张鑫源)[1,2]

[1]*State Key Laboratory of Pulsed Power Laser Technology, National University of Defense Technology, Hefei 230037, China*

[2]*Anhui Provincial Key Laboratory of Electronic Restriction, National University of Defense Technology, Hefei 230037, China*

[3]*The Military Representative Bureau of the Ministry of Equipment Development, Central Military Commission in Beijing, Beijing 100191, China*

**Corresponding author: skl_hyh@163.com*

Received October 21, 2019; accepted February 28, 2020; posted online May 11, 2020

To fully describe the structure information of the point cloud when the LIDAR-object distance is long, a joint global and local feature (JGLF) descriptor is constructed. Compared with five typical descriptors, the object recognition rate of JGLF is higher when the LIDAR-object distances change. Under the situation that airborne LIDAR is getting close to the object, the particle filtering (PF) algorithm is used as the tracking frame. Particle weight is updated by comparing the difference between JGLFs to track the object. It is verified that the proposed algorithm performs 13.95% more accurately and stably than the basic PF algorithm.

Keywords: object tracking; LIDAR; global and local feature descriptor; point cloud.
doi: 10.3788/COL202018.061001.

Object tracking technology is widely used both in civil and military fields, including unmanned driving[1], intelligent robots[2,3], ballistic missile tracking[4], and so on[5]. Particle filtering (PF)[6] is a filtering method based on the Monte Carlo method and recursive Bayesian estimation. It is not restricted by system conditions and can effectively estimate parameters and filter states under nonlinear and non-Gaussian conditions. Therefore, PF has been widely used in practical moving object tracking systems[6–8].

The data obtained by LIDAR[9] has high accuracy and is not easily affected by illumination changes[10]. In addition to the application of PF to object tracking in two-dimensional (2D) space, the application in three-dimensional (3D) space is also very rich. However, on the one hand, many researchers only take 3D information obtained by LIDAR as a supplementary means of 2D image information in the research process. Seldom have researchers been able to fully explore the potential structure information of 3D data. Choi and Christensen[11] used photometric (color) and geometric (3D points and surface normals) features to determine the likelihood of each particle. Held *et al.*[12] comprehensively used the descriptor of 3D shape, color, and motion to fully describe the object. In the other hand, Zhou[13] directly used the original point cloud information after segmentation for robots to grab dynamic objects successfully. However, for an object with higher speed, as the LIDAR-object distance and the angle of the LIDAR platform change, the original data of point cloud will change accordingly. Such changes have influences on the direct use of the original point cloud information, which will decrease the stability of the object description and tracking process.

In addition, the above studies were all completed in a close LIDAR-object distance, which means that the amount of point cloud data used was relatively abundant. As the LIDAR-object distance becomes longer, the amount becomes smaller, which increases the difficulty in describing the object. Seldom are studies under the above situation. In 2D space, joint color features are used to fully conduct the description when the camera-object distance is long. Ning *et al.*[14] built the joint color material to enhance the object description. Li *et al.*[15] also proposed the joint color space descriptor to enhance the robustness of postulate estimation of a point cloud scene in the PF framework.

In this Letter, LIDAR is assumed to be in an airborne platform for object tracking from an initial distance of 3 km from the scene. There is no open access to the data obtained under the situation used for tracking; thus, the data for experiment in this Letter was undocumented before. The LIDAR-object distance continuously changes during the movement of the platform. The number of laser beams of the LIDAR is 64. The repetition frequency and frame frequency of the LIDAR is 10 kHz and 20 Hz, respectively. The structure of the proposed method is shown in Fig. 1. Firstly, an object is chosen. To fully describe the object, an $n \times 341$-dimensional joint global and local feature (JGLF) descriptor is proposed. With the framework of PF, an object tracking method is proposed. Comparing the Bhattacharyya distances[16] between the JGLF of initial particles and the chosen object, particle weights are calculated, and the particle resample is done. Upon getting the new particles, the object will be tracked.

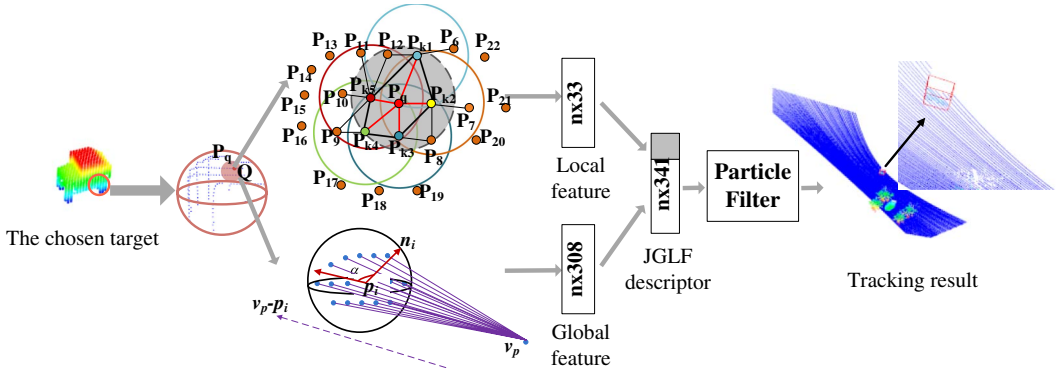

Fig. 1. Proposed object tracking method of point cloud based on JGLF.

Local or global feature descriptors of the 3D LIDAR point cloud cannot perform well when the data is little. In order to make the most of 3D structure information, the JGLF descriptor is proposed. When the platform is far away from the object, and the imaging resolution remains the same, the JGLF descriptor performs better. The process of calculating the JGLF descriptor is as follows.

The relationship between one point and other points in its k neighborhood is calculated one by one among the 3D dataset P. For the point P_q of point cloud P, a 1×33-dimensional eigenvector is obtained after the calculation of fast point feature histogram (FPFH) (P_q)[17]. Then, the points that have been used for calculating the local features are successively included in the set $Q^q\,(q = 1, 2, ..., n)$. For each data set Q^q, it is considered as a whole to extract global features and calculate the viewpoint feature histogram (VFH) (Q^q)[18] with a 1×308 eigenvector.

The sum value is first calculated to obtain the sum of all the feature eigenvectors in FPFH (P_q) and VFH (Q^q):

$$\text{sum} = \sum \text{FPFH}(P_q) + \sum \text{VFH}(Q^q). \tag{1}$$

The construction matrix calculation of the JGLF descriptor is

$$\text{JGLF}(P_q) = \text{FPFH}(P_q) \times T_A + \text{VFH}(Q^q) \times T_B. \tag{2}$$

T_A and T_B can be described by

$$T_A = \begin{bmatrix} \underbrace{1/33\text{sum} \quad \cdots \quad 1/33\text{sum}}_{33} & \underbrace{0 \quad \cdots \quad 0}_{308} \\ \vdots \quad \ddots \quad \vdots \quad \vdots \quad \ddots \quad \vdots \\ \underbrace{1/33\text{sum} \quad \cdots \quad 1/33\text{sum}}_{33} & \underbrace{0 \quad \cdots \quad 0}_{308} \end{bmatrix}, \tag{3}$$

$$T_B = \begin{bmatrix} \underbrace{0 \quad \cdots \quad 0}_{380} & \underbrace{1/308\text{sum} \quad \cdots \quad 1/308\text{sum}}_{33} \\ \vdots \quad \ddots \quad \vdots \quad \vdots \quad \ddots \quad \vdots \\ \underbrace{0 \quad \cdots \quad 0}_{380} & \underbrace{1/308\text{sum} \quad \cdots \quad 1/308\text{sum}}_{33} \end{bmatrix}. \tag{4}$$

PF[7,19–21] uses particles with weights to approximate the state distribution of the object at time k. One possible state of the object is represented by the equations of station and measurement:

$$\begin{aligned} x_k &= f_k(x_{k-1}, v_{k-1}), \\ y_k &= h_k(x_k, n_k). \end{aligned} \tag{5}$$

The PF algorithm consists of six steps: particle initialization, state prediction, sequential importance sampling, weight calculation, weight normalization, and particle resample, as is shown in Fig. 2. In the first particle initialization step, the total number of the particle is set as n. The object to be tracked is selected, and its JGLF descriptor is extracted using the method shown in Fig. 1. State estimation is made according to $X_k = X'_{k-1} + v_{k-1}$, where v_{k-1} stands for the random Gaussian noise. It is assumed that the importance distribution $q(k)$ at moment k is only related to the state value $x(k-1)$ at moment k and the measured value $y(k)$ at moment k:

$$q(x_k|x_{0:k-1}, y_{1:k}) = q(x_k|x_{k-1}, y_k), \tag{6}$$

$$w_k^{(i)} \propto w_{k-1}^{(i)} \cdot \frac{p(y_k|x_k^{(i)}) \cdot p(x_k^{(i)}|x_{k-1}^{(i)})}{q(x_k^{(i)}|x_{k-1}^{(i)}, y_k)}. \tag{7}$$

The Bhattacharyya distance of JGLF between the particle swarm and the object is introduced to measure the particle weight in the proposed algorithm. The larger the weight is, the higher the similarity between the particles and the object is. In the PF algorithm, the weight

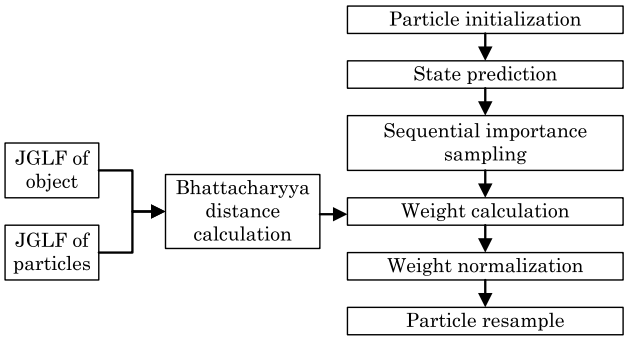

Fig. 2. Flow chart of the proposed object tracking method.

represents the importance of each particle. The weight of each particle is characterized by the Bhattacharyya distance. Assume that there are two $n \times m$-dimensional vectors $h(n, m)$ and $g(n, m)$:

$$b_k(h, g) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sqrt{h(i, j) \times g(i, j)}. \tag{8}$$

b_k is called the Bhattacharyya coefficient, and its value range is [0,1]. The Bhattacharyya distance is calculated as

$$d(k) = \sqrt{1 - b_k(h, g)}. \tag{9}$$

If the calculated $d(k)$ value is smaller, and the two histograms are more similar, then the similarity between the two point clouds is higher.

Thus, the particle weight should be increased. Therefore, the weight is calculated as

$$w_k^{(i)} = \frac{1}{[1 - d(k)^2]^2}. \tag{10}$$

Then, the weights of each particle are normalized as

$$w_k^{(i)} = w_k^{(i)} \Big/ \sum_{i=1}^{N} w_k^{(i)}. \tag{11}$$

N_{eff} is the effective particle number, and N_T is the threshold of the particle number; and if $N_{\text{eff}} < N_T$, the following calculation is performed:

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_s} (w_k^{(i)})^2}. \tag{12}$$

A group of particles are generated again in the update area. First, according to the following formula, the cumulative probability is calculated and normalized:

$$\begin{aligned} c_k^{(0)} &= 0, ..., c_k^{(i)} = c_k^{(i-1)} + w_k^{(i)}, \\ c_k^{(i)} &= c_k^{(i)} \Big/ \sum_{i=1}^{N} c_k^{(i)}. \end{aligned} \tag{13}$$

Then, a set of random numbers that obey uniform distribution $u \sim U(0, 1)$ is generated. $u_{\min}$, which is the smallest and satisfies $c_k^{(i)} \geq u_{\min}$, is found. The particle state value is set as $x_{k-1}^{(i)} = x_{k-1}^{(u_{\min})}$. After completing the above steps, the average value of the object state can be expressed as

$$E(x_k) = \sum_{i=1}^{N_s} w_k^{(i)} x_k^{(i)}. \tag{14}$$

The end of object tracking is related to the length of the data sequence. In order to verify the performance of the proposed algorithm, experiments are implemented with Visual Studio 2017 (VS2017) and the Point Cloud Library (PCL)[22] on a computer with a main frequency of 3.5 GHz and a memory of 8 G. The experiments include testing the object recognition ability of the JGLF descriptor and the proposed tracking algorithm. Besides, since there is no open access for data obtained from the aircraft platform to track the object, data used in this Letter are simulated by the software Blender[23]. To obtain data, one can infer to the operation introduction of the software.

In order to evaluate the effect of the proposed algorithm, the intersection ratio R of a single frame and the tracking accuracy and running time are calculated to, respectively, evaluate the accuracy, stability, and real-time performance of the algorithm. The intersection ratio of a single frame characterizes the ratio of the coincidence part between the particle bounding box and the actual object point cloud and the number N_g of the actual object point cloud in a single frame. The calculation equation[24] is

$$R = \frac{\sum_{(x_i, y_i, z_i) \in A \cap G} p_i}{N_g} \times 100\%. \tag{15}$$

(x_i, y_i, z_i) represents the spatial position of the point P_i, which is the overlapping space of the calculated particle set A and the corresponding true dataset G. The larger the ratio is, the better the tracking in a single frame performs. The threshold of R is set to 50%. When the calculated ratio of a frame is larger than 50%, then the frame is considered as being successfully tracked.

The tracking accuracy S in the overall tracking process is the ratio of the number of frames satisfying the threshold condition to the total number of frames. The average running time is calculated to evaluate the real-time performance of the entire process. In different simulation scenarios, the feature template libraries are trained in advance to compare the object recognition capabilities of the JGLF descriptor, VFH[18], clustered VFH (CVFH)[17], global radius-based surface descriptor (GRSD)[25], ensemble of shape functions (ESF)[26], and FPFH[27] descriptor by feature matching. The average object recognition rate is used as the evaluation standard, and the result is shown in Fig. 3.

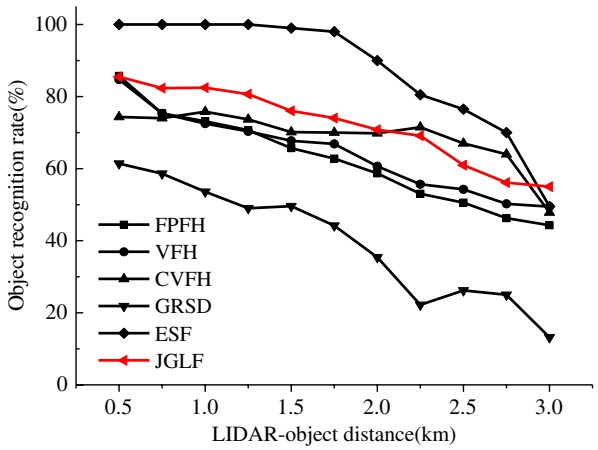

Fig. 3. Comparison of the object recognition rate at different distances between LIDAR and the object.

From Fig. 3, as the LIDAR-object distance becomes longer, the object recognition capability of each descriptor decreases. The ESF descriptor performs best, then the JGLF descriptor. Only observing FPFH and VFH, when the LIDAR-object distance is from 0.5 km to 1.5 km, the average object recognition rate of FPFH is higher than that of VFH. While the LIDAR-object distance exceeds 2 km, VFH performs better, which proves that the ability of single local or global features to describe the continuous change of the LIDAR-object distance is still insufficient. CVFH performs more segmentation and clustering in advance in the calculation process, so its performance is relatively better, but this descriptor itself has strong instability, which can explain the increase in the range of 2.25 to 2.5 km.

From Table 1, it can be seen that the average object recognition rate of JGLF is 15.5% lower than that of ESF, but it is still the second accurate descriptor, and it performs 33 ms faster than ESF. Besides, JGLF is more stable than ESF.

The tracking result of the proposed method is shown in Fig. 4.

In Fig. 4, each tracking result is drawn in a bounding box. As the object moves and the LIDAR-object distance becomes shorter, the tracking bounding box can track the object well in order to more clearly observe the difference between the particles generated by the proposed algorithm and the basic PF algorithm. In Fig. 4(f), black particles represent actual point cloud data, blue particles represent the results obtained by the basic PF algorithm, and red particles represent the results obtained by the PF tracking algorithm based on the JGLF proposed in this Letter. The whole spatial distribution of red particles is closer to the actual object point cloud than that of the blue ones.

The red line in Fig. 5 represents the threshold of tracking accuracy, and it is set as 50%. The blue line represents the intersection ratio in each frame using the proposed algorithm, and the green line represents the intersection ratio in each frame using the basic PF algorithm. In the beginning, the LIDAR-object distance is long, so the result is relatively poor because of the small amount of point cloud data. As LIDAR and the object move closer, the performance becomes better. However, results of the basic one still show that in some frames it works badly, which is mainly caused by the lack in describing the difference between the object and particles. During the whole process, the tracking accuracy of the basic PF algorithm is 84.87%, while the tracking accuracy of the proposed PF algorithm in this Letter is 98.82% with a 13.95% improvement. The 13.95% increase proves that using a JGLF can make better use of the information of the point cloud.

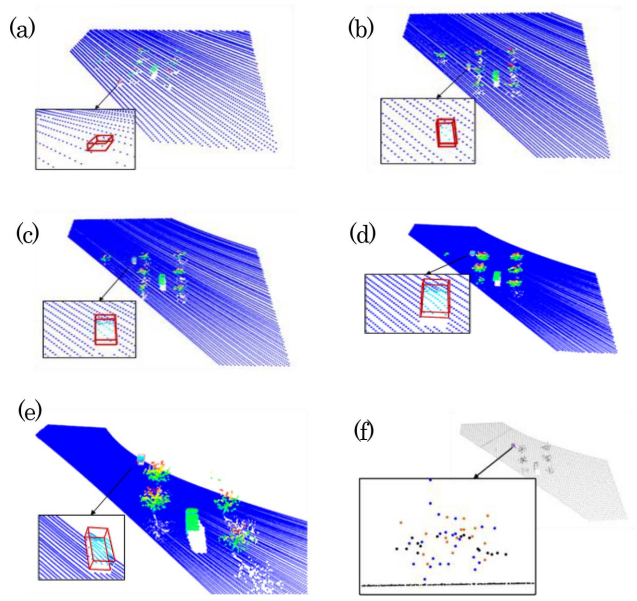

Fig. 4. Results of the PF point cloud tracking algorithm based on JGLF in frame n: (a) $n = 1$, (b) $n = 61$, (c) $n = 121$, (d) $n = 181$, and (e) $n = 241$. (f) Comparison between particles in frame 186.

Table 1. Object Recognition Ability Comparison of Six Descriptors

	Object Recognition Rate (%)		Average Running Time (ms)
	Mean	Standard Deviation	
FPFH	62.37	13.17	5
VFH	64.34	11.25	3.6
CVFH	68.91	7.79	4.5
GRSD	39.85	16.28	31
ESF	87.59	16.71	39
JGLF	72.09	10.81	6

In Table 2, the tracking results of five algorithms are shown. The five algorithms are the basic PF algorithm, PF algorithms based on FPFH, VFH, CVFH, and JGLF in this Letter. It can be seen from Table 2 that the PF algorithms based on FPFH, VFH, CVFH, and JGLF all have better performance in tracking accuracy and stability than the basic one. As the proposed algorithm combined both global and local features, it achieved the highest tracking accuracy and the average intersection ratio of a single frame. Compared with three other algorithms in tracking accuracy, the increase is 13.95%, 9.78%, 8.06%, and 7.57%, respectively. For the mean value of R of each single frame, the increase is 15.2%, 7.34%, 5.8%, and 7.68%, respectively. The proposed algorithm is second to the one based on CVFH in the stability of each single frame. Considering the increase of calculation complexity, the running time of the proposed algorithm is only increased by 0.52 ms. Since the LIDAR obtains data with the interval of 50 ms, the running time of the

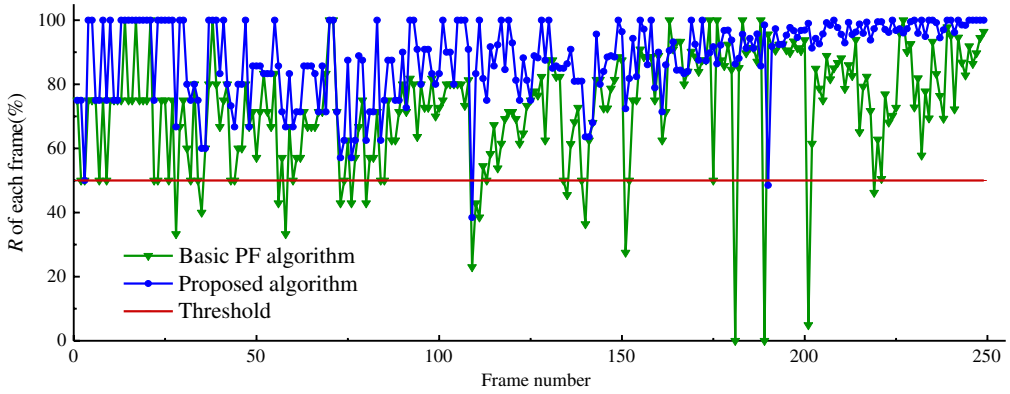

Fig. 5. Comparison of the object tracking effect between the two algorithms.

Table 2. Comparison of Tracking Results of Five Algorithms

	Tracking Accuracy (%)	R of Single Frame (%)		Average Running Time (ms)	CPU Utilized Percent (%)
		Mean	Standard Deviation		
Basic algorithm	84.87	72.81	17.75	12.44	5
Algorithm based on FPFH	89.04	80.67	12.47	12.71	7
Algorithm based on VFH	90.76	82.21	14.59	12.68	7
Algorithm based on CVFH	91.25	80.33	9.07	12.82	8
Proposed algorithm	98.82	88.01	11.96	12.96	8

proposed algorithm is 12.96 ms, accounting for only about 26% of the time interval of data acquisition, which proves that the proposed one has a good real-time performance.

In summary, when it comes to object tracking in a long distance, the lack of object information can be a difficult problem to tackle. With the development of LIDAR, the structure information of the 3D point cloud can meet the above needs. Under the situation with only a little 3D LIDAR point cloud data because the LIDAR platform is far from the object, the JGLF descriptor is proposed. Compared with current feature descriptors of the 3D LIDAR point cloud, the proposed descriptor performs better in the accuracy, stability, and real-time performance of object recognition. Using PF for the frame, an object tracking algorithm is proposed. The comparison experiments prove a better performance of the proposed algorithm, as it increases the tracking accuracy to 98.82%. At the same time, the results show that when object tracking has to be operated in a long distance, using the 3D LIDAR point cloud can be helpful to improve the accuracy and stability of tracking results.

This work was supported by the National Natural Science Foundation of China (Nos. 61271353 and 61871389), Foundation of State Key Laboratory of Pulsed Power Laser Technology (No. SKL2018ZR09), and Major Funding Projects of National University of Defense Technology (No. ZK18-01-02).

References

1. T. Chen, B. Dai, D. Liu, and J. Song, in *International Congress on Image & Signal Processing* (IEEE, 2016), p. 1566.
2. M. Braun, M. Hofele, J. Schanz, S. Ruck, M. Pohl, R. Börret, and H. Riegel, Proc. SPIE **10909**, 109090V (2019).
3. Z. Y. Zhou, J. J. Wang, Z. F. Zhu, D. H. Yang, and J. Wu, Optik **158**, 639 (2018).
4. M. Yu, L. Gong, H. Oh, W. H. Chen, and J. Chambers, IEEE Trans. Aerosp. Electron. Syst. **54**, 1066 (2017).
5. J. Zhao, G. Xiao, X. Zhang, and D. P. Bavirisetti, Chin. Opt. Lett. **17**, 031001 (2019).
6. S. P. Sangale and S. B. Rahane, in *IEEE Inventive Computation Technologies International Conference* (2016), p. 1.
7. S. Sedai, M. Bennamoun, D. Q. Huynh, and A. Gaussian, IEEE Trans. Image Process. **22**, 4286 (2013).
8. U. Fatima, J. P. S. Yadav, and R. K. Goel, Int. J. Comput. Appl. **173**, 30 (2017).
9. Z. D. Chen, R. W. Fan, G. C. Ye, T. Luo, J. Y. Guan, Z. G. Zhou, and D. Y. Chen, Chin. Opt. Lett. **16**, 041101 (2018).
10. M. Li, Y. Hu, N. Zhao, and L. Guo, IEEE Geosci. Remote Sens. Lett. **16**, 962 (2019).
11. C. Choi and H. I. Christensen, in *International Conference on Intelligent Robots and Systems* (IEEE, 2014), p. 1084.
12. D. Held, J. Levinson, S. Thrun, and S. Savarese, Int. J. Robot. Res. **35**, 30 (2016).

13. B. N. Zhou, *Research of Fast Object Tracking Algorithm in 3D Point Cloud Environment*, Master Thesis (Wuhan University of Science and Technology, 2018).
14. J. Ning, L. Zhang, D. Zhang, and C. Wu, Int. J. Pattern Recogn. Artif. Intell. **23**, 1245 (2009).
15. S. Li, S. Koo, and D. Lee, in *IEEE Conference on IEEE/RSJ International* (2015), p. 6079.
16. S. Bi, M. Broggi, and M. Beer, Mech. Syst. Signal Process. **117**, 437 (2019).
17. A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R. B. Rusu, and G. Bradski, in *IEEE International Conference on Computer Vision Workshops* (2011), p. 6.
18. R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, in *IEEE/RSJ International Conference on Intelligent Robots and Systems* (2010), paper 11689992.
19. R. B. Rusu, KI-Kunstliche Intell. **24**, 345 (2010).
20. I. Leizea, H. Álvarez, and D. Borro, Comput. Vision Image Understand. **133**, 51 (2015).
21. A. Ahmad and P. Lima, Robot. Auton. Syst. **61**, 1084 (2013).
22. A. Aldoma, IEEE Robot. Autom. Mag. **19**, 80 (2012).
23. Blender, https://www.blender.org/.
24. K. Ragab, Int. J. Pattern Recogn. Artif. Intell. **30**, 1660004 (2016).
25. Z. C. Marton, D. Pangercic, N. Blodow, and M. Beetz, Int. J. Robot. Res. **30**, 1378 (2011).
26. W. Wohlkinger and M. Vincze, in *IEEE International Conference on Robotics and Biomimetic* (2012).
27. R. B. Rusu, N. Blodow, and M. Beetz, in *IEEE International Conference on Robotics and Automation* (2009), p. 3212.