2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-7281-6212-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/IROS45743.2020.9341299

GOSMatch: Graph-of-Semantics Matching for Detecting Loop Closures in 3D LiDAR data

Yachen Zhu[1], Yanyang Ma[1], Long Chen[1], Cong Liu[1], Maosheng Ye[2], and Lingxi Li[3]

***Abstract*— Detecting loop closures in 3D Light Detection and Ranging (LiDAR) data is a challenging task since point-level methods always suffer from instability. This paper presents a semantic-level approach named GOSMatch to perform reliable place recognition. Our method leverages novel descriptors, which are generated from the spatial relationship between semantics, to perform frame description and data association. We also propose a coarse-to-fine strategy to efficiently search for loop closures. Besides, GOSMatch can give an accurate 6-DOF initial pose estimation once a loop closure is confirmed. Extensive experiments have been conducted on the KITTI odometry dataset and the results show that GOSMatch can achieve robust loop closure detection performance and outperform existing methods.**

I. INTRODUCTION

Loop closure detection is a problem associated with identifying places visited previously. It is a crucial part of Simultaneous Localization and Mapping (SLAM). During SLAM, pairwise scan matching odometry inevitably accumulates pose drift. Reliable loop closure detection is a key technique for SLAM systems to correct the drift error [1]. Though many vision-based methods have been proposed in recent years [2]–[4], it may draw unreliable results in the cases of dramatic changes on illumination [5] or viewpoint [6]. LiDAR, unlike cameras [7], senses the surrounding environment by generating high resolution 3D points with accurate measurements. It can work under unfavorable illumination conditions but also provide more geometric information. Therefore, the LiDAR-based loop closure detection task has attracted significant research attention.

Generally, traditional LiDAR-based methods use local keypoints [8]–[11] or other global features [12]–[18] to extract pointwise descriptors from point clouds and subsequently compare the descriptor of the query scan with the descriptors of historical scans to finally recognize the previously-visited places. These types of approaches place much emphasis on local details but ignore high-level feature constraints. Instead of defining revisited places by geometric points or other point-level features, humans understand the entire scenes from a more macro perspective [19], by recognizing objects and their relative positions in three-dimensional space.

[1]Y. Zhu, Y. Ma, L. Chen, and C. Liu are with the School of Data and Computer Science, Sun Yat-sen University, China. Corresponding author: Long Chen. chenl46@mail.sysu.edu.cn

[2]M. Ye is with the School of Geodesy and Geomatics, Wuhan University, China.

[3]L. Li is with Dept. of Electrical and Computer Engineering, Purdue School of Engineering and Technology, Indiana University-Purdue University Indianapolis, USA.

Inspired by the way humans identify places, we propose a novel approach that utilizes semantic objects in the scenes to detect loop closures. A couple of novel global and local descriptors formed by these semantic objects are introduced. The global descriptors are designed to efficiently seek the top similar loop candidates and the local descriptors are used to calculate the one-to-one correspondences of the semantics in two point clouds. A geometric verification step is then exploited to identify loop closures.

To the best of our knowledge, this is the first work that leverages object-level semantics to detect loop closures in 3D laser data. The main contributions of this paper are summarized as follows:

- We propose GOSMatch, an object-based approach for reliable place recognition in the urban driving environment based on LiDAR-only observations.
- New kinds of global and local descriptors are investigated for efficiently two-step loop searching, which encode spatial relationship between semantic objects.
- We perform extensive experimental validation against other state-of-the-art methods on a large public dataset and also publish the implementation of GOSMatch at https://github.com/zhuyachen/GOSMatch.

The rest of this paper is organized as follows: Section II reviews the previous literature our work relates to. Next, we describe the presented loop closure detection framework in Section III. Section IV shows the experiment results on the KITTI odometry dataset. Finally, we conclude in Section V.

II. RELATED WORK

Although detecting loop closures from 3D point clouds has been intensively studied in the past decades, it remains an open problem in SLAM systems.

A. Traditional LiDAR-based place recognition

One kind of methods utilize local features for place recognition. Bosse and Zlot [8] extract regional shape descriptors from randomly selected keypoints in the point clouds. Subsequently, a voting strategy is used to find the nearest neighbor among the keypoints to recognize places. Steder et al. [9] extract feature description vector of each keypoint in a transformed range image. Then a kd-tree is exploited to process the high dimensional vectors efficiently. To achieve a higher recognition performance, the Normal Aligned Radial Feature (NARF) local descriptor and a bag-of-words (BoW) matching approach are employed in their extension work [10].

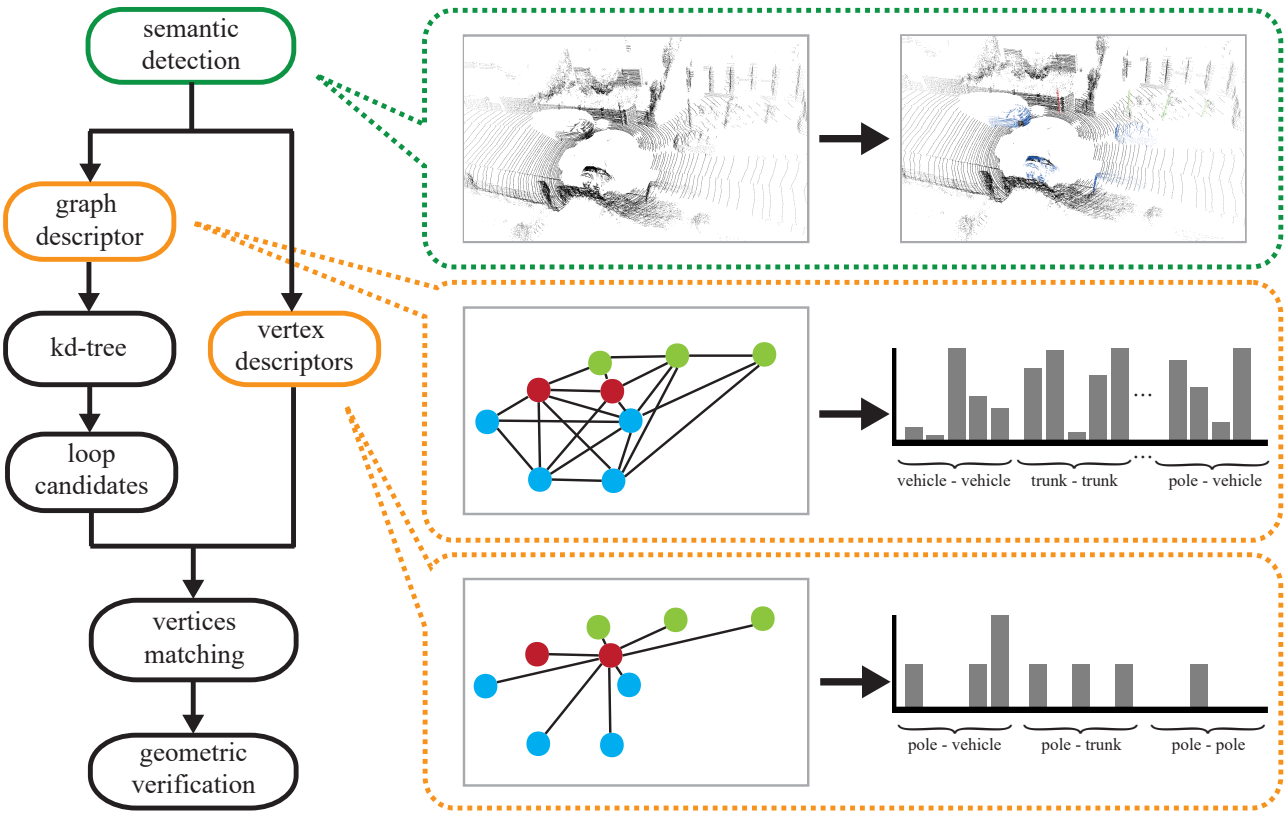


Fig. 1. The block diagram of GOSMatch. First, we segment the semantics from the raw point cloud. Then, a histogram-based *graph descriptor* is established for a fast nearest neighbor search. Note that we omit some edges in the undirected complete graph and we reduce it to a 2D representation for clarity. Next, we compare the *vertex descriptors* of the query point cloud and the retrieved candidates, a 6D initial pose estimation and a loss value are eventually derived from the geometric verification step.

Another kind of typical approaches focus on extracting the global representation of point cloud, which are usually presented in the form of histograms, such as VFH [17], ESF [18] and Z-Projection [16]. An appearance-based method [12] is proposed based on the Normal Distribution Transform (NDT) algorithm. After discretizing the clouds into cubes, the shape properties of each cell can be determined from the covariance matrix. By combining them, a surface shape histogram representing the whole cloud can be obtained. Giseop Kim [15] proposes Scan Context, a descriptor that encodes the max height of points in each divided bin into a 2D matrix. Since matching the matrix-type descriptors requires heavy computational cost, Kim extracts ring key from Scan Context to build kd-tree and speed up the searching process. He et al. [14] project 3D points on multiview 2D planes and concatenate all singular vectors derived from these planes as the point cloud descriptor. Rohling et al. [20] propose a 1D histogram of point range distribution as a global point cloud descriptor. The Wasserstein metric is then used to compare the histograms for place recognition.

B. Place recognition based on high-level descriptors

While local descriptors always lack the overall-description ability and global descriptors suffer from rotational variance problem easily, Shan et al. [21] instead extract more distinctive descriptors like edge and planar features from ground points and segmented points respectively. Then, Iterative Closest Point (ICP) is performed to match feature correspondences when finding a loop closure. Dube et al. propose SegMatch [22] that match clustering segments to obtain a more general and robust solution for place recognition. However, SegMatch takes so much computing cost that the whole loop closure detection framework can only update at 1 Hz.

C. Data association with graph representation

Graph matching plays an important role in coping with the pairwise data association problem. Specifically, Graph representation is a usual approach to describe the objects as well as their topology. In this case, finding the association of the objects between two places is therefore transformed into calculating the vertices and edges correspondences between graphs. However, finding an exact solution to this problem is always NP-hard. Bailey et al. [23] generate a correspondence graph and then search maximum clique in it to obtain exact correspondences between vertices and edges in two graphs. However, this method is only suitable for matching graphs with a small-sized number of vertices.

To avoid the prohibitive computation cost, it is typical to alternatively use tolerable approximate solutions. One

attempt is to work with a graph kernel based on random walk technique [24]. For each node in the graph, it generates numerous walking sequences as a node descriptor, following a matching step to find the node-to-node correspondences. Moreover, Fisher et al. [25] use the graph kernel method proposed in [26] to compare the similarity between relationship graphs for scene recognition.

III. METHOD

The block diagram of the proposed method is depicted in Fig. 1. It consists of four main modules: semantic detection, $graph\ descriptor$ generation, vertices matching, and geometric verification.

A. Semantic Detection

According to [27], valuable semantic features should be stable, distinctive and view-independent. We focus on detecting parked vehicles, trunks, and poles as they are the common semantic features with such attributes in the city road scene. Though the parked vehicle is a kind of potentially movable object, which means parked vehicles may move away and new vehicles may come to stop at some point, the time interval of each loop closure detection task is relatively short in practice. Therefore, in our experiment, we consider the parked vehicles to be as critical as any other semantic features and we assume that the positions of parked vehicles in the same place does not change a lot during two observations.

We employ RangeNet++ [28], a state-of-the-art deep learning architecture specifically designed for the semantic segmentation task in 3D LiDAR data. This end-to-end network can classify each point in the original point cloud. It is important to note that, unlike trunks and poles, it is unable to determine whether vehicles are parked or moving based on one point cloud, thus we utilize a front-end odometry to estimate the speed of the vehicles to simply distinguish between moving vehicles and parked vehicles.

Once we get the semantic labels, the Euclidean clustering algorithm is performed to retrieve objects. For all objects, we compute their centroids to represent their spatial locations in the point cloud.

B. Graph Descriptor Generation

A single LiDAR scan P is described by an undirected complete graph $\boldsymbol{G} =< \boldsymbol{V}, \boldsymbol{E} >$, where $\boldsymbol{V}, \boldsymbol{E}$ represents the vertices set and the edges set respectively. We consider the locations of semantic objects which are gained from the semantic detection module as vertices in $\boldsymbol{G}$ while each edge $e_{ij} =< v_i, v_j >$ in $\boldsymbol{E}$ represents the Euclidean distance between the vertex v_i and v_j. Classified by semantic category, there are three kinds of vertices with different semantic categories in $\boldsymbol{V}$ (vehicle, trunk and pole) and six kinds of edges in $\boldsymbol{E}$ (vehicle-vehicle, trunk-trunk, pole-pole, vehicle-trunk, trunk-pole and pole-vehicle).

The whole histogram-based $graph\ descriptor$ consists of six parts, which correspond to the six edge types in $\boldsymbol{E}$. For instance, we shall introduce the procedure of calculating one of the six parts by using the edges in $\boldsymbol{E}^{pole-trunk}$. We assume a constant bin count b and an interval I range from the shortest edge length l_{min} to the possible longest edge length l_{max}:

$$I = [l_{\min}, l_{\max}] \tag{1}$$

We divide I into mutually exclusive and separated subintervals, the size of each subinterval can be calculated by the following formulation:

$$\Delta I = \frac{1}{b}\left(l_{\max} - l_{\min}\right) \tag{2}$$

Therefore, each bin corresponds to one of the disjunct subintervals

$$I_k = [l_{\min} + k \cdot \Delta I, l_{\min} + (k+1) \cdot \Delta I] \tag{3}$$

Then, this part of $graph\ descriptor$ formed by $\boldsymbol{E}^{pole-trunk}$ can be denoted as:

$$\boldsymbol{h}^{pole-trunk} = (h_0, h_1, \ldots, h_{b-1}) \tag{4}$$

where

$$h_k = \left|\left\{e \in \boldsymbol{E}^{pole-trunk} : l(e) \in I_k\right\}\right| \tag{5}$$

Once the other five kinds of edges are processed, the complete $graph\ descriptor$ $\boldsymbol{H}^{graph}$ for point cloud P can be established by concatenating the six parts:

$$\boldsymbol{H}^{graph} = \left(\boldsymbol{h}^{vehicle-vehicle}, \ldots, \boldsymbol{h}^{pole-vehicle}\right) \tag{6}$$

$graph\ descriptors$ for each historical scan are stored in a database. When coming in a query point cloud, they are used to construct a kd-tree to perform the standard k-nearest neighbor search algorithm for efficiently searching the top similar loop candidates. Then, a list of N possible loop candidates can be obtained.

C. Vertices Matching

In this section, we introduce the $vertex\ descriptor$ to describe one vertex in the graph. Similar to $graph\ descriptor$, $vertex\ descriptor$ is also histogram-based. The difference is that the edges considered in $vertex\ descriptor$ are no longer of the whole graph, but the edges connected to the described vertex v. To build a $vertex\ descriptor$ for $v \in \boldsymbol{V}^{pole}$, there are only three kinds of edges need to be considered because one of the two endpoints that make up a edge have been identified to be a pole. Similar to $graph\ descriptor$, one kind of edges can form one of the three parts of the $vertex\ descriptor$, which can be described as:

$$\boldsymbol{h}^{pole-vehicle} = (h_0, h_1, \ldots, h_{b-1}) \tag{7}$$

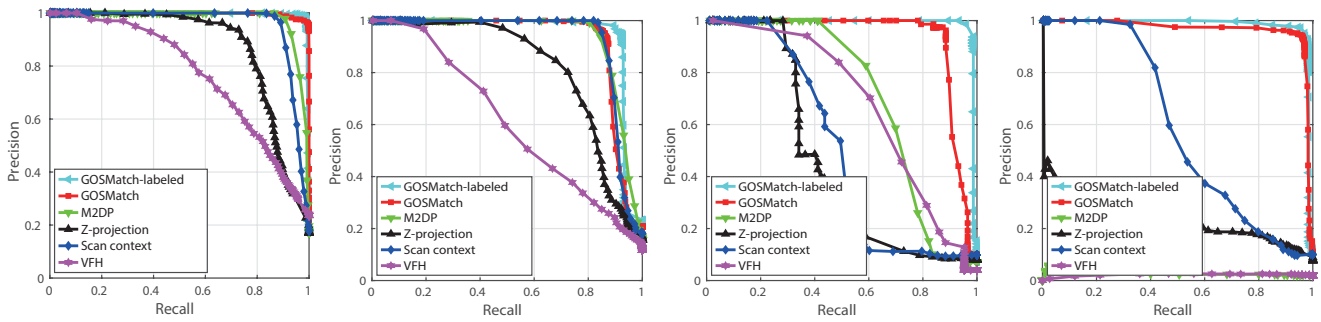
where

$$h_k = \left|\left\{e \in \boldsymbol{E}^{pole-vehicle} : l(e) \in I_k, e \in \boldsymbol{e}_v\right\}\right| \tag{8}$$

where $\boldsymbol{e}_v$ represents the edges that connected to v. After concatenating the three parts, the $vertex\ descriptor$ $\boldsymbol{H}^{vertex}$ for v can be finally obtained:

$$\boldsymbol{H}^{vertex} = \left(\boldsymbol{h}^{pole-vehicle}, \boldsymbol{h}^{pole-trunk}, \boldsymbol{h}^{pole-pole}\right) \tag{9}$$

We then match the $vertex\ descriptors$ in the query point cloud and the ones in the candidate point cloud by Euclidean distance to find the correspondences.



(a) KITTI 00 (b) KITTI 05 (c) KITTI 07 (d) KITTI 08

Fig. 2. Precision-Recall curves for KITTI dataset

D. Geometric Verification

This step focuses on selecting a set of geometrically consistent correspondences for each loop candidate. A typical RANSAC-based algorithm is used to refine the correspondence set [29]. In each RANSAC iteration, we use the SVD method to find a closed-form 6-DOF transformation solution to the Absolute Orientation Problem [30]. If this transformation matrix can lead to more inliers, we update it.

For each candidate, we evaluate the loop closure detection loss using the formulation as follows:

$$loss = \sqrt{\frac{\sum_{i=1}^{|\boldsymbol{C}|}\left(T \cdot \begin{bmatrix} c_{iq} \\ 1 \end{bmatrix} - \begin{bmatrix} c_{ic} \\ 1 \end{bmatrix}\right)^2}{|\boldsymbol{C}|}} \tag{10}$$

where $\boldsymbol{C}$ represents the refined correspondence set between query scan and its loop candidate, T represents the transformation matrix, c_{iq} and c_{ic} represent the i-th 3D points in $\boldsymbol{C}$ that belong to the query scan and the candidate scan respectively.

The candidate scan with the minimum loss $loss_{min}$ is picked to judge that if there exists loop closure according to a threshold β:

$$L_\beta(P_q, P_c) = \begin{cases} \text{true} & \text{if } loss_{min} < \beta \\ \text{false} & \text{otherwise} \end{cases} \tag{11}$$

Note that the two places are considered as a loop closure only if $loss_{min}$ is smaller than β and vice versa. Once a loop closure is confirmed, we regard the transformation matrix as the 6-DOF initial pose.

IV. EXPERIMENTS

In this section, the proposed graph-based algorithm is evaluated on the KITTI odometry dataset [31]. We perform a comparison with other state-of-the-art loop closure detection algorithms including four global descriptors: VFH, M2DP, Scan Context, and Z-projection. We use the C++ implementation of VFH in the Point Cloud Library (PCL) [32], the open-sourced Matlab code of M2DP and Scan Context. Moreover, the proposed GOSMatch approach and the Z-projection method are implemented by ourselves on the Matlab platform. All experiments are conducted on an Intel Core i7-6820HQ with 16 GB RAM and an Nvidia Titan X with 12 GB RAM.

A. Dataset and experiment settings

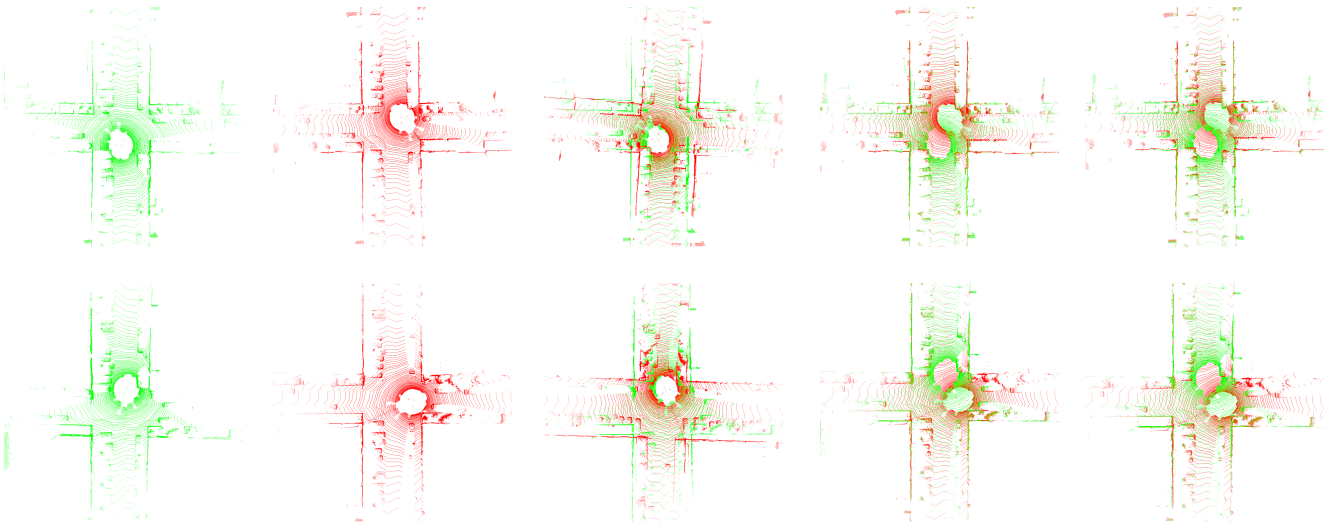
There are a total of 11 sequences with ground truth pose in the KITTI odometry dataset. We select sequence 00, 05, 07, 08 to evaluate the proposed method as they have loops and are collected in the urban environment. Among these four sequences, 08 is the only one that can be used to test the rotation-invariance performance of the algorithms because all real loops in sequence 08 are in the opposite direction.

The RangeNet++ is trained by using the labeled point cloud data from the SemanticKITTI dataset [33], which provides dense point-wise labels for each sequence in KITTI odometry dataset. Specifically, For each sequence we picked, we train one unique model of RangeNet++ by using the remaining 10 sequences.

There is a situation in the KITTI dataset where the car passes through the same wide crossroads twice, once in the upper left corner and another time in the lower right corner. Thus, we consider the detection is a true positive loop closure if the Euclidean distance of two places is less than 15m. For each query, we exclude a total of 100 scans to prevent the query scan from matching to its time-neighbors. Z-Projection, as well as VFH, are both surface normal-based methods, both of them require a normal-computation step. To facilitate the implementation of the code, we adopt the second way for Z-Projection to calculate the normals which are mentioned in [16], and the bin number in Z-Projection is set to 202. Normal radius is the only parameter for VFH, and we set it to be 0.03m in this experiment. The candidate number of Scan Context is set to 50 since Kim et al. [15] illustrated that it had a better performance. For the remaining parameters of Scan Context, we use the default values proposed in the open-sourced code. Similar to Scan Context, we use the parameters mentioned in the available code for M2DP. Considering the semantic segmentation ability in our method, we set $l_{min} = 0, l_{max} = 60, b = 60$ and set $N = 10$ for all validation datasets.

B. Loop closure detection performance

In the first experiment, we assess the performance of GOSMatch. Fig. 2 shows the precision-recall curves [34].



(a) Query Point Clouds (b) Detected Point Clouds (c) ICP without initial pose (d) Alignments from initial pose (e) ICP with initial pose

Fig. 3. The loop closure detection examples in KITTI 00 for the initial pose accuracy. (a) The frame numbers of the query point cloud are 1415 (up) and 4540 (down). (b) The frame numbers of the detected point cloud are 586 (up) and 116 (down). (c) The ICP registration results without the initial poses. (d) Align query and detected point clouds using the initial poses. (e) The ICP registration results with the initial poses.

TABLE I

RECALL AT 100% PRECISION WITH DIFFERENT n VALUES

	$n = 0$	$n = 1$	$n = 2$	$n = 3$	$n = 4$
KITTI 00	95.36%	94.49%	92.87%	92.61%	89.13%
KITTI 05	74.51%	71.14%	67.74%	55.58%	50.61%
KITTI 07	92.01%	89.21%	87.84%	88.61%	86.14%
KITTI 08	78.44%	77.91%	76.49%	68.64%	66.90%

Note that GOSMatch-labeled uses the point cloud data with labels as input while the other five methods use the raw point cloud data as input. Methods that are not object-based, especially Z-Projection and VFH, have shown poor performance. The reason is that these descriptors heavily depends on the accuracy of the normal estimation step. Besides, these types of descriptors only care about the angles formed by the normal with the centroid of the point cloud or with z-axis direction vectors but ignore the geometric space information. In this case, if the probability distribution of normal vectors is similar between two different places, these methods easily derive non-distinctive descriptors.

Scan Context and M2DP perform a comparatively good result in our experiment. However, they are unable to handle the challenging KITTI 08 dataset where all loops are generated when the car revisits the same place in the opposite direction. Only the rotational invariant algorithms can detect loop closures in this sequence with high performance. M2DP can not successfully detect loop closures in KITTI 08 and the precision-recall curve of Scan Context also decreases dramatically.

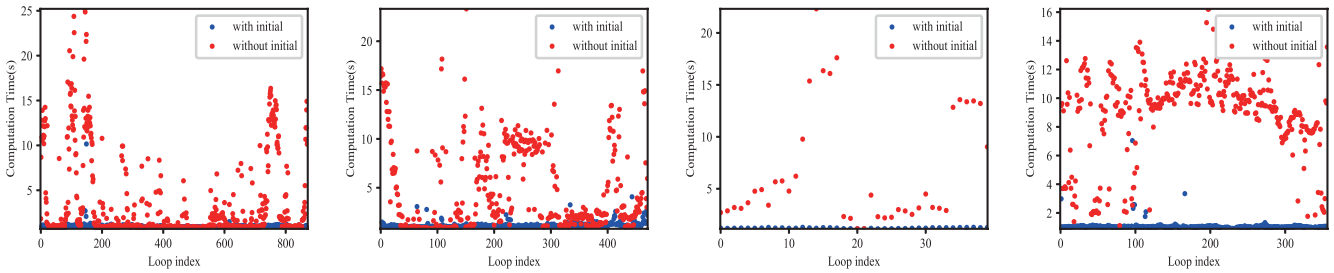
Overall, GOSMatch performs slightly worse than GOSMatch-labeled as the effect of the point classification errors from RangeNet++. Even so, both of them exhibit significant robustness and competitive rotation-invariance performance comparing with other methods. Our approach benefits from stable descriptors formed by valuable semantics. Such descriptors will not be greatly affected even if the viewpoint changes a lot, thus we can perform reliable place recognition.

C. Noise sensitivity

This section focus on evaluating the robustness of GOSMatch against noise. In this experiment, we choose to extract segments directly from the labeled point cloud data provided by SemanticKITTI for avoiding classification noise interference. We randomly remove X objects in each scan to simulate changes in the vehicle positions (e.g. the previously parked vehicles have driven away) and some misidentifications, X is a random variable and $X \sim B(n, \frac{1}{2})$, where n means the maximum number of objects that can be removed in each scan. We test the effect of different n values on the four sequences and we perform 100 runs for each n value in each sequence. The average recall at 100% precision are presented in TABLE I.

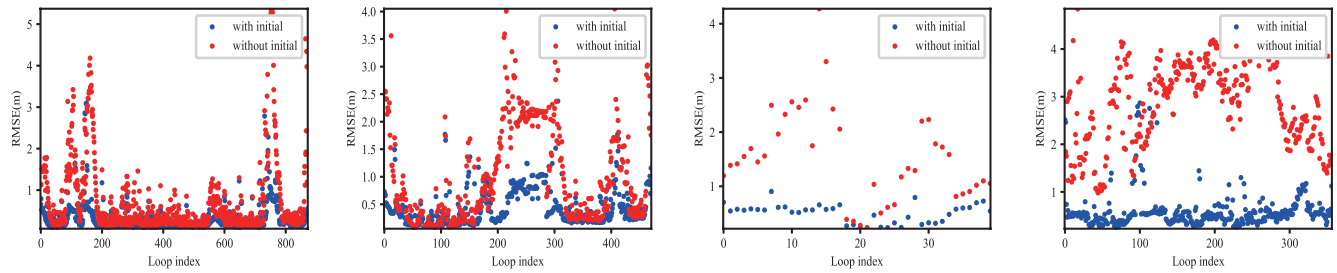
D. Initial pose accuracy

The goal of this section is to evaluate the initial pose obtained from GOSMatch. To get more reliable registration results, the initial pose can be provided to ICP. Some examples from KITTI 00 are depicted in Fig. 3. In Fig. 3 (e), the results show that query and detected point clouds can be



(a) KITTI 00 (b) KITTI 05 (c) KITTI 07 (d) KITTI 08

Fig. 4. Computation time for each KITTI sequence



(a) KITTI 00 (b) KITTI 05 (c) KITTI 07 (d) KITTI 08

Fig. 5. ICP point-to-point RMSE for each KITTI sequence

TABLE II
AVERAGE TIME COSTS ON KITTI 00 IN SECONDS

	Descriptors calculation	Loop searching	Total time per scan
Scan Context	**0.0424**	0.0514	**0.0938**
M2DP	0.2461	0.0220	0.2681
Z-Projection	0.4045	**0.0201**	0.4246
VFH	0.5625	0.0379	0.6004
GOSMatch	0.0183 +0.1892*	0.0278	0.2353

*semantic detection execution time

successfully registered using the initial poses. However, in Fig. 3 (c), we can see that without the provided initial poses, the registration procedures of ICP can easily fail. Besides, the initial poses obtained from GOSMatch are shown in Fig. 3 (d). The alignment results using the initial poses are close to results which are optimized by the ICP algorithm, revealing the high accuracy of the estimated initial pose. The experimental results indicate that GOSMatch is able to predict a precise initial pose when detecting a loop closure.

In our experiment, we also test and verify the effect of the initial pose on the performance of ICP. The computation time and point-to-point RMSE of ICP with and without the initial pose is given in Fig. 4 and Fig. 5 respectively. The ICP registration procedure benefits a lot from the initial pose GOSMatch provided.

E. Computational complexity

The computational requirements for different methods are evaluated on KITTI 00 and the concrete results are shown in Table II. Because our method is different from traditional methods in that GOSMatch requires a semantic detection step, so we separately show the time of calculating the two graphical descriptors and the time of semantic detection. In addition, only the semantic detection module in our method is performed on the GPU, all the other experiments are conducted on the CPU. The searching time of GOSMatch consists of searching k-nearest neighbors, matching *vertex descriptors* and verifying the geometric consistency. It is important to note that the time GOSMatch takes depends heavily on the semantic detection module, therefore a faster object detection algorithm can significantly improve the execution efficiency of GOSMatch.

V. CONCLUSIONS

In this paper, we propose GOSMatch, an algorithm for detecting loop closures in 3D point clouds based on two graphical descriptors. One is designed for efficiently searching the similar loop candidates from the historical point clouds, the other one is for further meticulous matching to give a solution to vertex-to-vertex correspondences between two places. Unlike the previous works, our method performs at a semantic level which is not only robust to environment changes but also able to give an accurate 6D initial pose estimation. The results of the exhaustive evaluation experiments show the potential of using relative position relationship between objects, which is significantly helpful for reliable loop closure detection.

VI. ACKNOWLEDGMENT

This work is supported by National Key R&D Program of China under Grant 2018YFB1305002.

REFERENCES

[1] S. Thrun *et al.*, "Robotic mapping: A survey," *Exploring artificial intelligence in the new millennium*, vol. 1, no. 1-35, p. 1, 2002.

[2] N. Sünderhauf and P. Protzel, "Brief-gist-closing the loop by simple means," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1234–1241.

[3] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[4] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 1643–1649.

[5] L. Chen, L. Sun, T. Yang, L. Fan, K. Huang, and Z. Xuanyuan, "Rgb-t slam: A flexible slam framework by combining appearance and thermal information," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5682–5687.

[6] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[7] L. Chen, Q. Zou, Z. Pan, D. Lai, L. Zhu, Z. Hou, J. Wang, and D. Cao, "Surrounding vehicle detection using an fpga panoramic camera and deep cnns," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, no. 99, pp. 1–13.

[8] M. Bosse and R. Zlot, "Place recognition using keypoint voting in large 3d lidar datasets," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 2677–2684.

[9] B. Steder, G. Grisetti, and W. Burgard, "Robust place recognition for 3d range data based on point features," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 1400–1405.

[10] B. Steder, M. Ruhnke, S. Grzonka, and W. Burgard, "Place recognition in 3d scans using a combination of bag of words and point feature based relative pose estimation," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1249–1255.

[11] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 3212–3217.

[12] M. Magnusson, H. Andreasson, A. Nuchter, and A. J. Lilienthal, "Appearance-based loop detection from 3d laser data using the normal distributions transform," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 23–28.

[13] A. Buyval, A. Gabdullin, R. Mustafin, and I. Shimchik, "Realtime vehicle and pedestrian tracking for didi udacity self-driving car challenge," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2064–2069.

[14] L. He, X. Wang, and H. Zhang, "M2dp: A novel 3d point cloud descriptor and its application in loop closure detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 231–237.

[15] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4802–4809.

[16] N. Muhammad and S. Lacroix, "Loop closure detection using small-sized signatures from 3d lidar data," in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*. IEEE, 2011, pp. 333–338.

[17] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 2155–2162.

[18] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3d object classification," in *2011 IEEE international conference on robotics and biomimetics*. IEEE, 2011, pp. 2987–2992.

[19] M. Kuse and S. Shen, "Learning whole-image descriptors for real-time loop detection andkidnap recovery under large viewpoint difference," *arXiv preprint arXiv:1904.06962*, 2019.

[20] T. Röhling, J. Mack, and D. Schulz, "A fast histogram-based similarity measure for detecting loop closures in 3-d lidar data," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 736–741.

[21] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4758–4765.

[22] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "Segmatch: Segment based place recognition in 3d point clouds," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5266–5272.

[23] T. Bailey, E. M. Nebot, J. Rosenblatt, and H. F. Durrant-Whyte, "Data association for mobile robot navigation: A graph theoretic approach," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 3. IEEE, 2000, pp. 2512–2517.

[24] A. Gawel, C. Del Don, R. Siegwart, J. Nieto, and C. Cadena, "X-view: Graph-based semantic multi-view localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1687–1694, 2018.

[25] M. Fisher, M. Savva, and P. Hanrahan, "Characterizing structural relationships in scenes using graph kernels," in *ACM SIGGRAPH 2011 papers*, 2011, pp. 1–12.

[26] Z. Harchaoui and F. Bach, "Image classification with segmentation graph kernels," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[27] F. Yu, J. Xiao, and T. Funkhouser, "Semantic alignment of lidar data at city scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1722–1731.

[28] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and Accurate LiDAR Semantic Segmentation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[29] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[30] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Josa a*, vol. 4, no. 4, pp. 629–642, 1987.

[31] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361.

[32] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1–4.

[33] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," in *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019.

[34] C. Manning and H. Schutze, *Foundations of statistical natural language processing*. MIT press, 1999.