A Partitioned Approach for Efficient Graph–Based Place Recognition

Mattia G. Gollub   Renaud Dubé   Hannes Sommer   Igor Gilitschenski   Roland Siegwart*

***Abstract*— Place recognition is a crucial capability of autonomous vehicles that is commonly approached by identifying keypoint correspondences which are geometrically consistent. This geometric verification process can be computationally expensive when working with 3D data and with increasing number of candidates and outliers. In this work, we propose a technique for performing 3D geometric verification efficiently by taking advantage of the sparsity of the problem. Exploiting the relatively small size of the area around the vehicle, the reference map is first subdivided in partitions, and geometric verifications are only performed across relevant partitions, guaranteeing the sparseness of the resulting consistency graph. A maximum clique detection algorithm is finally applied for finding the inliers and the associated 3D transformation, taking advantage of the low degeneracy of the graph. Through experiments in urban driving scenarios, we show that our method outperforms a state of the art method both asymptotically and in practice.**

I. INTRODUCTION

Efficient and reliable place recognition is one important challenge for enabling fully autonomous driving. With considerable changes in illumination occurring in driving scenarios, it is interesting to consider geometric information for performing place recognition. Autonomous vehicles are therefore often equipped with 3D time of flight sensors which permit a precise estimation of the road environment through the generation of point cloud maps.

One standard approach for recognizing places in 3D point cloud data is to compare a *local map* characterizing the vicinity of the vehicle, to a *target map* representing the full environment. This comparison can be done by extracting different basis elements such as keypoints[1], objects[2], shapes[3] or segments[4]. Place recognition is then performed by identifying correspondences between these basis elements and by verifying these correspondences for geometric consistency. This final 3D geometric verification step can be computationally expensive when working with large maps and with increasing number of correspondence candidates and outliers.

In this work, we formulate the problem of geometric verification as identifying a maximum clique in a *consistency graph* where edges connect correspondences that are geometrically consistent. A simplified example of a consistency graph is illustrated in Fig. 1c. We propose to perform the geometric verification by exploiting two important characteristics of the problem. First, we take advantage of the significant difference in size between the local and target maps by subdividing the latter into partitions and by performing geometric verifications only across relevant partitions. This approach not only effectively reduces the number of consistency tests but also guarantees the sparseness of the resulting consistency graph. In a second step, we exploit this sparsity by leveraging an efficient algorithm for detecting maximum cliques in sparse graphs.

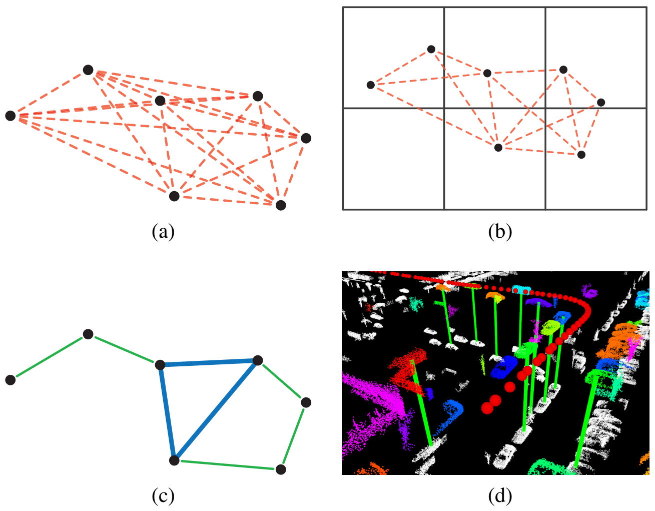

Fig. 1: Steps of the recognition process with minimum geometrically consistent set size $T = 3$: (a) Current approaches need to test all possible correspondence pairs for consistency. Nodes represent the correspondences, while the tested pairs are shown as edges. (b) Our partitioning approach allows to drastically reduce the number of consistency tests. (c) The consistency graph obtained with our method and the recognized T–clique (in blue) of geometrically consistent correspondences. (d) Example of a successful recognition in a urban driving scenario experiment. Correspondences that constitute the maximum consistent set are indicated in green.

The proposed approach is compared to a state of the art baseline method in urban driving scenario experiments which demonstrate that our approach is more efficient. We show through a simple example that the baseline method can return sub-optimal solutions whereas our method always identifies a maximum clique. A derivation of the complexity of our method is presented, demonstrating that it scales linearly with the size of the target map.

To summarize, this paper presents the following contributions:

- A novel partition-based algorithm for efficiently identifying maximum geometrically consistent sets of correspondences.
- An asymptotic complexity analysis of the proposed method.

*Authors are with the Autonomous Systems Lab, ETH, Zurich `{gollubm, rdube, sommerh, igilitschenski, rsiegwart}@ethz.ch`.

This work was supported by the European Union's Seventh Framework Programme for research, technological development and demonstration under the TRADR project No. FP7-ICT-609763.

- An evaluation of the proposed method based on experiments in urban driving scenarios.

The remainder of the paper is structured as follows: Section II provides an overview of the related work in the field of geometric verification for place recognition. Section III describes our partition-based approach to place recognition. The approach is evaluated in Section IV, and Section V finally concludes with a short discussion.

II. RELATED WORK

Numerous methods have been developed for performing global registration of partially overlapping 3D objects [5–7]. Unfortunately such methods cannot always be applied for performing place recognition in real-time, as the target map can be many orders of magnitude bigger than the local map and as the matching process can produce a high fraction of false correspondences.

Chen and Bhanu [8] propose to filter these outliers by clustering correspondences into geometrically consistent sets. Two correspondences c_i and c_j are called *pairwise geometrically consistent* if the difference of the Euclidean distance between the keypoints in the local map and in the target map is below a threshold ϵ, i.e. if

$$|d_l\left(c_i, c_j\right) - d_t\left(c_i, c_j\right)| \leq \epsilon \tag{1}$$

where $d_l\left(c_i, c_j\right)$ and $d_t\left(c_i, c_j\right)$ are the keypoint distances in the local map and in the target map respectively.

This idea is employed in the *recognition* module of the Point Cloud Library (PCL) [9] where geometric consistencies are determined using a bruteforce approach in which all possible correspondence pairs are checked for consistencies. This has an asymptotic complexity of $O\left(n^3\right)$, where n is the number of correspondences. This approach is well suited for scenarios with a low amount of candidates and was employed in our previous work on segment-based place recognition [4]. However, it does not scale well to cases with a large number of candidates and outliers, eg. when doing real-time place recognition in a large target map. We consider this method as our baseline in the experiments of Section IV.

Strategies for efficiently reducing the number of correspondence pairs have been proposed for stereo images and image retrieval. Ayache and Faverjon [10] describe a partitioning scheme for efficiently finding neighbor segments in stereo images. The SCRAMSAC method [11] performs RANSAC only on *spatially consistent* correspondences, i.e. correspondences that have a minimum fraction of matching neighbor features in both images. Both methods rely on assumptions about the disparity between images, thus their accuracy is influenced by the presence of high disparity and strong variation in viewing angles.

Graphical models have successfully been employed in the analogous task of recognizing places based on camera images [12, 13] In the context of geometry-based place recognition, Fernandez-Moral et al. [3] propose to leverage graphical models for executing the geometric verification. An interpretation tree is used to match the local and target graphs where vertices represent planes and edges represent geometric relationships between these planes. Finman et al. [14] also propose a similar graph-matching strategy where vertices represent objects instead of planes. Contrastingly in our approach, vertices and edges respectively represent correspondences and geometric consistencies. Place recognition is then executed by extracting a maximum clique.

III. METHOD

This section presents our approach for performing geometric verification of correspondences in order to recognize places in 3D maps. We treat this task as a graph problem with the goal of identifying a *maximum geometrically consistent set* which is a set of maximum size consisting of correspondences that are all pairwise geometrically consistent. Pairwise geometrical consistency relationships are encoded in an undirected graph $G = (V, E)$ where $V = \{c_i\}$ is the set of correspondences c_i and $E = \{e_{ij}\}$ is the set of undirected edges e_{ij} connecting all consistent pairs of correspondences (c_i, c_j). Once the graph is constructed, identifying a maximum geometrically consistent set is equivalent to finding a maximum clique of G. An example of a consistency graph with its maximum consistent set is given in Fig. 1c.

A. Partition-based consistency graph construction

The naive approach for identifying all pairwise consistencies is testing all possible correspondence pairs. Here we take advantage of our knowledge about the sizes of the target and local maps to reduce the number of tests.

In many recognition applications, like place recognition as in our case, the local map is significantly smaller than the target map. Taking advantage of this information we can define a criterion for reducing the amount of consistency tests a priori. Let ϵ be the tolerance for geometric consistency and b be the diameter of the bounding sphere of the local map keypoints, a pair of correspondences c_i and c_j can then only be consistent if

$$d_t\left(c_i, c_j\right) \leq b + \epsilon \tag{2}$$

Urban driving scenarios usually extend along the earth's surface only (x and y coordinates), but not vertically. Without loss of performance or correctness, we simplify the problem and consider only a bounding cylinder of diameter b. The new relaxed criterion becomes

$$d_t^{2D}\left(c_i, c_j\right) \leq b + \epsilon \tag{3}$$

where $d_t^{2D}\left(c_i, c_j\right)$ computes the distance between the target map keypoints of c_i and c_j projected on the xy–plane[1].

We create a 2D grid partitioning of the correspondences according to the position of their keypoints in the target map, where each partition $p_{u,v}$ has a square base with a side length of $b + \epsilon$ and infinite height. An example of 2D grid partitioning is illustrated in Fig. 1b. Each correspondence c_i is assigned to its partition $P\left(c_i\right) = p_{u,v}$,

$$u = \left\lfloor \frac{k_t\left(c_i\right).x - o.x}{b + \epsilon} \right\rfloor, v = \left\lfloor \frac{k_t\left(c_i\right).y - o.y}{b + \epsilon} \right\rfloor \tag{4}$$

[1]The z-axis is assumed to be roughly gravity aligned.

where $k_t(c_i)$ is the keypoint of c_i in the target map and o is the origin of the grid. A good choice of o is the componentwise minimum of all $k_t(c_i)$.

With the chosen grid size $b+\epsilon$, it is guaranteed that the bounding cylinder of the model is always contained in a squared group of four adjacent partitions. Thus geometric consistency tests are necessary only on a set of candidate pairs of correspondences that is much smaller than $V \times V$:

$$\{(c_i, c_j) \in V \times V \mid i < j \wedge \exists u, v : c_i \in p_{u,v} \wedge c_j \in \mathcal{N}(u,v)\} \tag{5}$$

where $\mathcal{N}(u,v) := \bigcup_{l,m \in \{-1,0,1\}} p_{u+l,v+m}$. i.e. for a given $c_i \in V$ only correspondences in the partition of c_i and the 8–neighbor-partitions need to be tested. Since consistency is a symmetric property, each pair is tested only once, i.e. if $i < j$. Other pairs of correspondences are ignored, since they cannot be consistent. The consistency graph is constructed as an adjacency list and contains all the geometrically consistent correspondence pairs (c_i, c_j) as edges. Fig. 1a shows a set of 7 correspondences where all 21 pairs are tested for consistency. Applying this partitioning strategy reduces the number of tests to 14 (Fig. 1b).

The same approach can be used for cases where the environment extends into the third dimension. In this case, assuming that the local map is bounded by a sphere of diameter b, a 3D grid of cubes with size $b+\epsilon$ is used and consistency tests are performed over the 26–neighborhood of each partition.

B. Maximum clique detection

We consider a recognition to be successful in case the size of the detected maximum geometrically consistent set is greater than or equal to a threshold parameter T. Thus we need to identify a maximum k–clique with $k \geq T$.

A second advantage of enforcing the partitioning constraints in situations where the local map is significantly smaller than the target map is that the sparseness of the consistency graph is guaranteed. With this knowledge we can rely on a class of k–clique detection algorithms that visit the graph in *degeneracy order* to find the maximum geometrically consistent set. A graph G can be characterized by its *degeneracy* (or k–core number) d, which is the smallest number so that every subgraph of G contains a vertex with degree $\leq d$. Each graph has a *degeneracy ordering*, an order in which each vertex has at most d vertices that come later in the ordering.

We use a generalization of the maximal clique listing approach described in [15, 16] to find a maximum clique of G. An outline of the algorithm is presented in Algorithm 1. First, we use the bucket sort algorithm for sorting the vertices in increasing degree order. This is described in detail in [17] and can be performed in $O(|V|)$. Then we iterate on the vertices according to the found order. At each step, the function CLIQUE is called using the current vertex of minimum degree as input. If the function returns a clique that is bigger than the current maximum clique max_clique, the new maximum clique is stored. The vertex is then removed from the graph together with its incident edges and the vertex ordering is updated in $O(d)$. The resulting *degeneracy ordering* in the visit of the vertices guarantees that at each step v has at most d neighbors, bounding the computational load on CLIQUE.

CLIQUE can be any function that returns the biggest clique $C \subseteq G$ such that $|C| \geq T$ and $v \in C$ (or the empty set, if such a clique does not exist). Instances of this function can be any clique detection approach like the branch-and-bound method presented in [18] or successive refinements like [19]. For simplicity, in this work we use a method based on heavy pruning strategies [20]. This algorithm builds a subgraph containing v and its neighbors and recursively visits its subgraphs to determine if a clique of the required size can be built. Vertex degrees are constantly checked to discard fruitless candidates as soon as possible in the recursion.

Algorithm 1 Outline of the algorithm for maximum clique detection. Iterating over the vertices in degeneracy order guarantees that v always has at most d neighbors.

```
1:  procedure MAXIMUMCLIQUE($G$, $T$)
2:      $max\_clique \leftarrow \emptyset$
3:      $sorted\_vertices \leftarrow$ SORTBYDEGREE($G$)
4:      while $|sorted\_vertices| \neq 0$ do
5:          $v \leftarrow sorted\_vertices[0]$
6:          $C$ = CLIQUE($v$, $G$, $T$)
7:          if $|C| \neq 0$ then
8:              $max\_clique \leftarrow C$
9:              $T \leftarrow |C| + 1$
10:         end if
11:         REMOVEVERTEX($G, v$)
12:         UPDATEORDER($G, sorted\_vertices$)
13:     end while
14:     return $max\_clique$
15: end procedure
```

C. Recognition

The condition on the size of the maximum geometrically consistent set can be removed in case it is safe to assume that a true recognition always exists. This is however not always possible, e.g. in loop–closure detection applications. Once an acceptable maximum clique is identified, the relative transformation between the local and target maps can be found using any rigid transformation estimation method. Here we use the least-squares approach described in [21].

D. Asymptotic complexity and scaling

In the consistency graph $G = (V, E)$, let the parameters $n = |V|$ be the number of correspondences and d be the degeneracy of the graph. Assuming that outliers are uniformly distributed in the target map and are present in a high ratio over inliers, we can say that n is proportional to the size of the target map. The average number of correspondences per partition n_p and d are considered constant as they depend on fixed parameters like the density

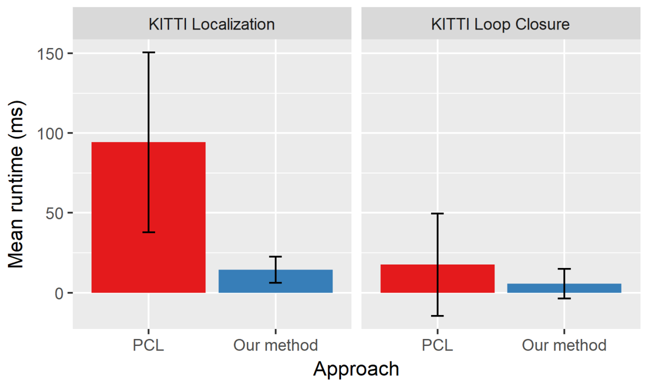

Fig. 2: Runtime comparisons between the reference PCL implementation and our new method. Error bars show the standard deviation of the measurements and are caused by the different amount of candidate correspondences found while the vehicle moves.

of objects in the environment, the size of the local map and the correspondence matching strategy.

Partitioning the correspondences is trivial and can be done in $O(n)$. All the necessary consistency tests can be performed in $O(n_p n)$ since each of the $\frac{n}{n_p}$ partitions requires $O\left(n_p^2\right)$ tests. The bucket sort is done in $O(n)$. Since every vertex visited in the loop has at most d neighbors, each of the n calls to CLIQUE has a worst case complexity of $O(d!)$, thus the complexity of the code in Algorithm 1 is $O(d!n)$. Although the $d!$ term may appear like an important bottleneck, any reasonable implementation for CLIQUE usually performs better. As shown in Table I, maximum clique detection brings a minimal contribution to the total runtime. The estimation of the 3D transformation scales linearly with the number of elements in the geometrically consistent set, thus it is bound by $O(d)$.

The total complexity of our method is $O\left(n\left(d! + n_p\right)\right)$. Under the homogeneity assumption, since d and n_p are parameters that do not depend on the target map's size, we can conclude that the performance of our algorithm scales linearly with the size of the target map.

IV. EXPERIMENTS

We tested our method in the place recognition module of our online LiDAR-based localization and mapping system described in [22]. We compare our new method with our current baseline approach implemented in the PCL (`pcl::GeometricConsistencyGrouping`). Both implementations are benchmarked in two different configurations:

- *Localization*: An autonomous vehicle drives in a known urban scenario, continuously trying to localize inside a static target map.
- *Loop-closure*: The vehicle explores an unknown urban scenario, continuously updating a dynamic target map and trying to detect loop-closures.

Each configuration uses a different dataset of the KITTI collection [23], both datasets have similar sizes. At each call we measure the runtime of the recognition. We use $\epsilon = 0.4m$ as consistency threshold and $T = 6$ as minimum consistent set size.

Figure 2 shows a runtime comparison between our method and the PCL implementation. In the localization test, our new method is 6.57 times faster than the reference method. This improvement comes from two main factors. First, only 32.6% of the consistency tests are performed. In addition, our partitioned approach enables better cache locality when accessing correspondence keypoints, increasing the performance of the consistency test functions.

In the loop-closure setting the speedup decreases to 2.05x. This is due to the fact that during the first part of the experiment the target map is relatively small and prevents partitioning from being effective. In general we observe that partitioning successfully accelerates the recognition process by reducing the number of tested correspondence pairs and our method scales better as the size of the map increases. In future work the method should be tested with bigger maps in order to prove the theorized linear scaling of the method (see Section III-D).

Step	Mean runtime (Localization)	Mean runtime (Loop-Closure)
Partitioning	$0.06 \pm 0.01ms$	$0.03 \pm 0.02ms$
Graph construction	$13.86 \pm 7.95ms$	$8.09 \pm 9.70ms$
Max clique detection	$0.13 \pm 0.08ms$	$0.17 \pm 0.23ms$
Transformation estimation	$< 0.01ms$	$< 0.01ms$
Total (Our method)	$14.33 \pm 8.16ms$	$8.52 \pm 9.98ms$
Total (PCL)	$94.23 \pm 56.28ms$	$17.49 \pm 32.10ms$
Speedup	6.57x	2.05x

TABLE I: Runtimes and speedups of the different steps of our algorithm. Clearly the most important factor is the construction of the consistency graph.

Table I shows the time needed by each step of our method. While partitioning quickly and effectively reduces the number of consistency tests required, the construction of the consistency graph is still the most expensive operation. Detecting a maximum clique is a relatively cheap task thanks to the sparseness of the graph and to the traversal strategy that bounds the complexity of the search.

We performed additional tests to determine the quality of the resulting recognition in the two methods. As metric for the quality we use the size of the identified maximum consistent set. Our method uses an exact algorithm, and is always able to find a consistent set of maximum size. On the other side, the PCL implementation uses a greedy approach and occasionally fails to detect the optimal solution. Fig. 3 shows a tested pathological case: depending on the order of the input correspondences the greedy approach can detect a maximum (3a, in blue) or a maximal (3b, in red) consistent set. Our method detects the maximum clique in both cases. Note that the PCL implementation does not explicitly build a graph, but detects the same pairwise consistency, thus we can use the same visualization.

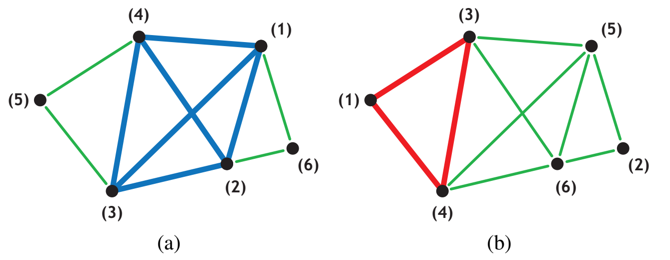

Fig. 3: Detection of consistent sets with $T = 3$ in two graphs based on the same correspondences stored in different orders, indicated by the numbers. The PCL implementation identifies the blue biggest cluster in (a), but can only find a suboptimal solution (in red) if the correspondences are stored differently (b). Our implementation uses exact k–clique detection and finds a maximum consistent set in all cases.

V. CONCLUSION

In this work, we presented a new graph-based method for place recognition in 3D point clouds that improves on our reference baseline in terms of performance. We stated the recognition task as a graph problem and used a novel partitioning approach to significantly reduce the number of consistency tests required. Taking advantage of the sparseness of the consistency graph, we use a clique detection algorithm to identify the biggest set of geometrically consistent correspondences quickly and exactly.

Experiments in urban driving scenarios show that our method performs better than the greedy approach of our selected baseline. We theoretically demonstrate that the runtime of this algorithm scales linearly with the size of the map, enabling fast localization in large environments. Benchmark results show that the present bottleneck of the method is the construction of the consistency graph. In future work we will explore other approaches to further accelerate the construction task. Moreover we would like to evaluate our method on bigger scenes to prove the theorized linear scaling behavior.

REFERENCES

[1] M. Bosse and R. Zlot, "Place recognition using keypoint voting in large 3D lidar datasets," in *IEEE Int. Conf. on Robotics and Automation*, 2013, pp. 2677–2684.

[2] R. F. Salas-Moreno, R. A. Newcombe *et al.*, "Slam++: Simultaneous localisation and mapping at the level of objects," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[3] E. Fernandez-Moral, W. Mayol-Cuevas *et al.*, "Fast place recognition with plane-based maps," in *IEEE Int. Conf. on Robotics and Automation*, 2013, pp. 2719–2724.

[4] R. Dubé, D. Dugas *et al.*, "Segmatch: Segment based place recognition in 3d point clouds," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5266–5272.

[5] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *European Conference on Computer Vision*. Springer, 2016, pp. 766–782.

[6] A. S. Mian, M. Bennamoun, and R. A. Owens, "Automatic correspondence for 3d modeling: an extensive review," *International Journal of Shape Modeling*, vol. 11, no. 02, pp. 253–291, 2005.

[7] Y. Guo, M. Bennamoun *et al.*, "3d object recognition in cluttered scenes with local surface features: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2270–2287, 2014.

[8] H. Chen and B. Bhanu, "3d free-form object recognition in range images using local surface patches," *Pattern Recognition Letters*, vol. 28, no. 10, pp. 1252–1262, 2007.

[9] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–4.

[10] N. Ayache and B. Faverjon, "Efficient registration of stereo images by matching graph descriptions of edge segments," *International Journal of Computer Vision*, vol. 1, no. 2, pp. 107–131, 1987.

[11] T. Sattler, B. Leibe, and L. Kobbelt, "Scramsac: Improving ransac's efficiency with a spatial consistency filter," in *Computer vision, 2009 ieee 12th international conference on*. IEEE, 2009, pp. 2090–2097.

[12] E. Stumm, C. Mei *et al.*, "Location graphs for visual place recognition," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5475–5480.

[13] S. Cascianelli, G. Costante *et al.*, "A robust semi-semantic approach for visual localization in urban environment," in *Smart Cities Conference (ISC2), 2016 IEEE International*. IEEE, 2016, pp. 1–6.

[14] R. Finman, L. Paull, and J. J. Leonard, "Toward object-based place recognition in dense rgb-d maps," in *ICRA Workshop Visual Place Recognition in Changing Environments, Seattle, WA*, 2015.

[15] D. Eppstein, M. Löffler, and D. Strash, "Listing all maximal cliques in sparse graphs in near-optimal time," *Algorithms and computation*, pp. 403–414, 2010.

[16] D. Eppstein and D. Strash, "Listing all maximal cliques in large sparse real-world graphs," *Experimental Algorithms*, pp. 364–375, 2011.

[17] V. Batagelj and M. Zaversnik, "An o (m) algorithm for cores decomposition of networks," 2003.

[18] C. Bron and J. Kerbosch, "Algorithm 457: finding all cliques of an undirected graph," *Communications of the ACM*, vol. 16, no. 9, pp. 575–577, 1973.

[19] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques and computational experiments," *Theoretical Computer Science*, vol. 363, no. 1, pp. 28–42, 2006.

[20] B. Pattabiraman, M. Patwary *et al.*, "Fast algorithms for the maximum clique problem on massive sparse graphs," *arXiv preprint arXiv:1209.5818*, 2012.

[21] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 13, no. 4, pp. 376–380, 1991.

[22] R. Dubé, A. Gawel *et al.*, "An online multi-robot slam system for 3d lidars," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2017.

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