Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

Fast 2D Map Matching Based on Area Graphs

Jiawei Hou, Haofei Kuang and Sören Schwertfeger
School of Information Science and Technology
ShanghaiTech University
Shanghai, China
{*houjw, kuanghf, soerensch*}*@shanghaitech.edu.cn*

***Abstract*— We present a novel area matching algorithm for merging two different 2D grid maps. There are many approaches to address this problem, nevertheless, most previous work is built on some assumptions, such as rigid transformation, or similar scale and modalities of two maps. In this work we propose a 2D map matching algorithm based on area segmentation. We transfer general 2D occupancy grid maps to an area graph representation, then compute the correct results by voting in that space. In the experiments, we compare with a state-of-the-art method applied to the matching of sensor maps with ground truth layout maps. The experiment shows that our algorithm has a better performance on large-scale maps and a faster computation speed.**

***Index Terms*— Map matching, Map segmentation, Topological maps**

I. Introduction

Map matching is an important and challenging task in the robotics field. Merging robot maps with maps in different modalities is beneficial for many robotics applications. For example, matching a robot sensor map with a known layout map could apply the semantic information from the layout map to the robot map, which is very helpful for human robot interaction [1], [2]. This can be extended to semantic localization, where semantic labels in one map can help with the localization of the robot in the current map. Furthermore, matching maps of different modalities, e.g., from robot maps to ground truth maps, is a critical part of robot map evaluation [3]. Another important application of map matching is to help with the cooperation in multi-robot systems [4].

Solving the autonomous map matching problem is interesting and challenging. The process of matching two different maps is similar to image registration. In practice, robot sensor maps can be treated as a picture. However, when an occupancy grid map is interpreted as an image, it has fewer and less descriptive features compared to a general digital image. Thus common image registration algorithms (e.g., feature-based methods) are not suitable to the map matching problem in robotics.

There are three challenges of map matching problem: 1) Matching two different maps in different modalities with image registration only works with quite similar looking maps with big overlap [4]–[7]. 2) These methods cannot deal with the scale problem, which can be present in maps from different sources. Some of these methods are based on the assumption of a rigid transformation. But large-scale maps sometimes are distorted due to problems with the SLAM algorithm, so non-rigid transformations would have to be applied. To address these challenges, topological maps have been used in map matching problem [3], [8]–[10]. Shahbandi et al. [11] propose a state-of-the-art method which is based on region decomposition. This method can solve the different modality and scaling, and it also improves the performance with the distortion problem. 3) The processing time in 2D map matching has to be considered. Especially for large scale maps the computation requirements can be intensive and thus prevent certain applications with real-time requirements, like multi-robot localization and map matching.

In this work, we propose a graph matching algorithm which is based on our Area Graph [12]. Our method addresses the above mentioned problems. The presented algorithm can deal with maps from different modalities and it does not rely on rigid transformation assumption. Also, our method exhibits a fast computation speed. Our algorithm has been implemented in C++ and tested on various maps in different modalities and scenarios. We also compare with the state-of-the-art method and the results show the high performance and efficiency of our algorithm, especially in the large-scale scenario.

This paper is organized as follows. In Section II, we introduce the related work about map matching problems. Section III gives the overview of our algorithm. The details of area segmentation and matching will be discussed in Section IV. Section V shows the multiple results aiming to illustrate the performance and efficiency of our approach for different maps. We conclude our work in Section VI.

II. Related Work

In this section, we review related work concerning map matching. The map matching problem is related to the data association problem in robot mapping. Konolige et al. stated that map merging was a meaningful and challenging problem in the robotics field, but it did not receive as much attention as other robotics problems such as SLAM [13]. In the early stage of this research field, optimization is one of the most popular technique. Map matching is similar to image registration. Feature-based and optimization methods

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

are mainstream methods, such as SIFT [14] and Lucas-Kanade algorithm [15], [16]. Maps can be considered as a picture, and the map matching could be considered as a special case of image registration. However, it offers less distinctive features than general digital pictures due to the self-similarity of the indoor environment. These methods are easily affected to local minimal, particularly in absence of an initial guess. Carpin and Birk model the map matching problem as an optimization problem and then solve it through a stochastic search algorithm [6], [7]. Their subsequent work improves the performance of their previous work through using mechanisms to detect failures and a more complex method to guide the search [5].

Other methods could transform the input cartesian signal to a parametric space. This has the advantage of extracting the structure of the maps in the parametric space. Carpin proposed a method using the Hough transform to model the search space and decompose the transformation into translation and rotation estimation for merging occupancy maps in multi-robot systems [4]. Because of the decomposition, these methods are often fast, non-iterative, and deterministic. However, these approaches are limited to some assumptions: the input maps being merged have been built using the same scale and just work best on maps which are in same modality. These methods are also limited to rigid transformations, and would not satisfy the matching in a large-scale environment because of affection of distortions.

A common approach to address the problem is to model the input maps to an abstract representation and then search on the similarity of instances. Huang et al. proposed an algorithm which is based on graph matching and image registration to merge partial maps with embedded topological maps [9]. Through a similar approach, Wallgrn et al. propose a map matching algorithm using a graph matching based on the Voronoi graph [10]. Schwertfeger et al. developed a method for map matching which is used for map quality assessment in an automated process [3], [8] based on the matching of vertices, which takes their descriptor and the similarity of their neighbors into account. Saeedi et al. use the topological structure to find the relative transforms between two maps through Radon transform and find translation through an edge matching methods [17]–[19].

For fusing prior maps and robot maps, the state-of-the-art approach to the map matching problems must allow the input maps with different modalities. Georgiou et al. state that computing the correspondence between abstract human readable maps with a map which is constructed from a robot's sensors is desired to improve the effectiveness of human-robot cooperation [2]. A related topic is the matching of paths in a topological map, which can then be used for map matching [20]. For finding the correct matching between different modalities of the maps, Shahbandi et al. propose a decomposition-based algorithm to match 2D spatial

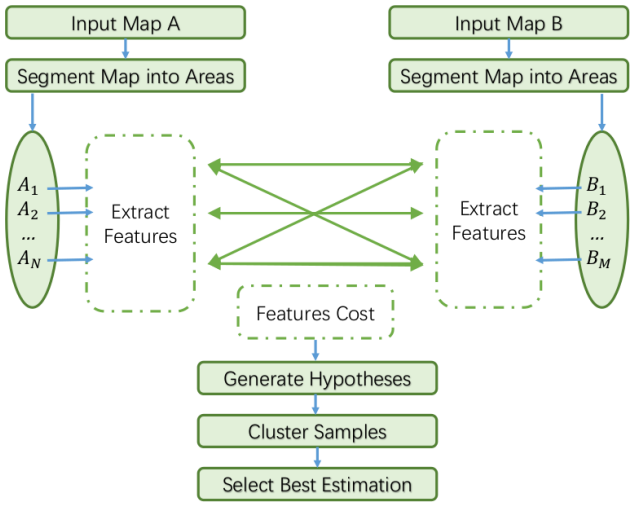

Fig. 1: The pineline of our method.

maps [11]. Through region decomposition, they abstract maps into 2D arrangements which explicitly represent both the boundaries and the region of the open-spaces. It makes the method capable of interpreting maps in different modalities and scale. Their method achieves state-of-the-art performance in different modality map matching tasks, but in order to avoid missing the correct results, they exhaust the search space, which makes their algorithm slow. To address these problems, we propose an area graph based method to utilize the feature of the areas (e.g., corridor or doors).

III. Overview

Given two overlapping maps of the same environment, the workflow of our algorithm is as follows:

1) Segment the given 2D maps as areas, in this work we generate the Area Graph [12] as the segmentation solution.
2) Compute the features for areas segmented from the map. The features used for matching include the size of the area, *passage distance* and *convex hull longest distance* of the areas, where passage distance is the distance between a pair of passages of the same area, which are basically connections between different areas.
3) Compute the area matching cost for each pair of areas by multiplying the feature cost with a weight vector w, and record the matching cost in an adjacency matrix M_{cost}.
4) Find a list of pairwise areas that are considered as matched with each other. For the matched area pairs, estimate the rotation between them and save the angles as candidates.
5) Cluster the rotations candidates to vote the range in which the correct rotation between two maps is located, which is called *best cluster*.

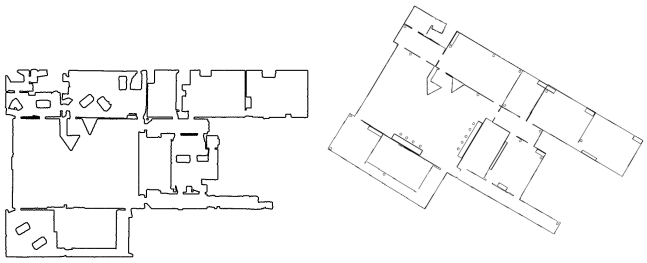

Fig. 2: The maps to be matched. The left map is binarized from a scanned map and the right map is a corresponding artificial map, both of which are from Bormann's dataset [21].

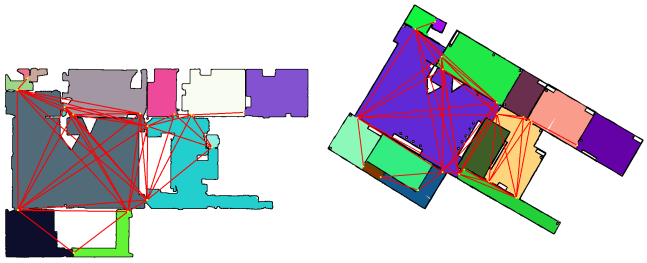

Fig. 3: The maps are divided into different areas, which are represented with different colors. The passages points are highlighted as yellow points, and we connect the passages belonging to the same area with red lines. The passage distance features are computed from the lengths of the red lines.

6) Traverse the samples in the best cluster and compute their overlap area to choose the best transformation.

In Section IV, we introduce these modules in detail. The overview of our algorithm is shown in Fig. 1.

IV. Methodology

We use the two maps shown in Fig. 2 as an example to explain our method.

A. Brief Introduction of Area Graph

In this paper, we segment the given map by generating the Area Graph, which is introduced in our previous work [12]. The Area Graph is generated from a Topology Graph, which in turn is generated from the Voronoi Diagram of the 2D grid map. Additionally the α-shape algorithm is used to detect and merge open areas such as rooms. More details can be found in [12]. Due to the dependence of segmentation, our matching algorithm works with the assumption that the maps can be segmented to an enough number of areas which are not seriously over-segmented. As an example, the segmentation of the maps given in Fig. 2 is visually shown in Fig. 3.

To address the over-segmentation problem, we have developed an algorithm for the generation of furniture free maps [22]. It is based on a mapping robot with two 3D laser scanners: a horizontally scanning and a vertically scanning sensor. The robot additionally has 9 color cameras, an IMU and odometry. All sensors are fully synchronized [23] and calibrated [24]. Fig. 4 shows the results of the Area Graph algorithm on maps from the same dataset, once with a normal 2D grid mapping algorithm and once with the furniture free mapping method.

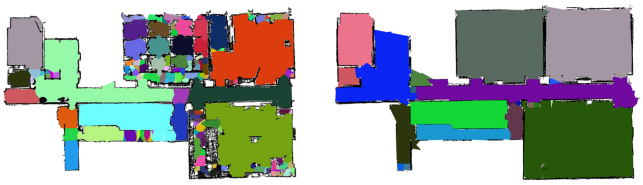

Fig. 4: Furniture free 2D grid mapping: Left: Area Graph of a normal map with furniture. Right: A furniture free map from the same dataset with its Area Graph.

B. Feature Extraction and Cost Computation

For each area in the Area Graph, we compute its *size*, *passage distance* and *convex hull longest distance* as features to match areas between maps.

1) Area Size: The regions in the map are represented as bounded polygons, and the area size is the polygons' most basic geometric property having meaningful recognition. Note the fact that the unit used for area size is m^2, while that for the other two features are m, and the total cost is a weighted sum of the three feature costs. In order to maintain the linearity of the weighted sum, it is reasonable to calculate the area size cost c_a with the square roots of the area sizes:

$$c_a^{ij} = \frac{|\sqrt{a_i} - \sqrt{a_j}|}{\sqrt{\max(a_i, a_j)}}. \quad (1)$$

Here the area cost c_a between two areas is the normalized distance between the area size of the ith region in the first map, donated as a_i, and the area of the jth region in the second map, donated as a_j.

2) Passage Distance: We notice that the distance between doors and the distance between junctions in corridors are fixed, which are called passages. Therefore, we developed the idea that compares the distance between passages as a feature to match areas. In Fig. 3 all the passages in the same area are connected with red lines. We calculate the distance between all passages in each area. The distance between a pair of passages is donated as pd_n^i, where i is the area ID and this is the nth passage distance in the area i, e.g., the N passage distances included in area a_i are $\{pd_1^i, \ldots, pd_N^i\}$. An example is shown in Fig. 5.

Then the minimal difference between passage distances in area i and passage distances in the area j is the passage

Fig. 5: The black numbers show the distance between each pair of passages in the regions. In the left example, the second left-top room with three doors has three passage distances close to the three passage distances involved in the corresponding region of the right map.

Fig. 6: The black numbers show the longest distance in each convex hull of regions.

distance cost c_p^{ij} between the areas i and j:

$$c_p^{ij} = \min_{n,m} \frac{|pd_n^i - pd_m^j|}{\max(pd_n^i, pd_m^j)}. \quad (2)$$

3) Convex Hull Longest Distance: There are some areas that only contain one passage. As a result, the passage distance of these regions cannot be obtained. An example of the convex hull longest distance features of each area is shown in Fig. 6. We obverse that for corresponding areas the their convex hull longest distance are very close, even better than the accuracy of the feature passage distance. However, since this feature can be affected by the scan integrity of the map, while the passage distance is robust, we only use this feature for the regions that cannot obtain passage distance. The longest distance cost between two regions is computed as

$$c_l^{ij} = \frac{|ld_i - ld_j|}{\max(ld_i, ld_j)}. \quad (3)$$

C. Area Matching

The goal of this step is to find a list of potential matching areas according to the features described above.

1) Total Cost between Areas: Each area in map A is compared to all the m areas in map B. For each pair to compare, we compute the feature distance for the three

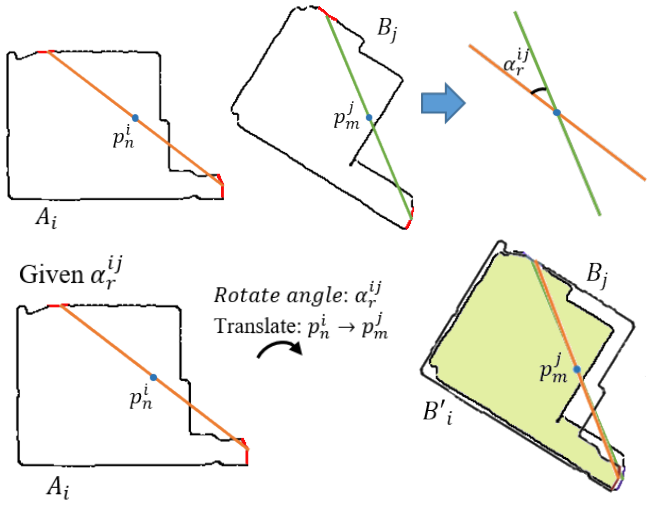

Fig. 7: (a) Compute the transformation between two areas. (b) Transform the area A_i with the given angle and the midpoints of its matched segment and the one of its matched area B_j. The passages are highlighted in red.

features described above, and save in a row vector. Then the total matching cost between this area pair is computed by multiplying the feature cost vector and a weight vector as a weighted sum of all features cost. The matching costs between the area i to all m areas in another map are recorded in C_i.

$$C_i = \begin{bmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ \vdots & \vdots & \vdots \\ d_{m1} & d_{m2} & d_{m3} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} c_{i1} \\ c_{i2} \\ \vdots \\ c_{im} \end{bmatrix}$$

where d_{jf} is the fth feature cost between the area i and the area j in another map for $j = 1, \ldots, m$ and $f = 1, 2, 3$. In Section V-A, the setting of the weight parameters will be discussed.

2) Potential Matching Pairs: For each area in map A, we record its k (e.g., k=3) most similar areas, i.e. the areas with the lowest matching cost to it, in map B, and vice versa to each area in map B. Area i and j are considered as mutual matching areas if and only if area i is one of the k nearest matches of area j and area j is also one of the k nearest matches of area i. We record all the mutual matching pairs in a matched pairs list, which will be used to generate transformation hypotheses.

D. Transformation Estimation

1) Rotation between matched areas: Given a pair of matched areas (A_i, B_j), the transformation between A_i and B_j is obtained from their matched passage distance segments or longest lines segments in their convex hulls. The rotation is computed as the difference of the angles between the two

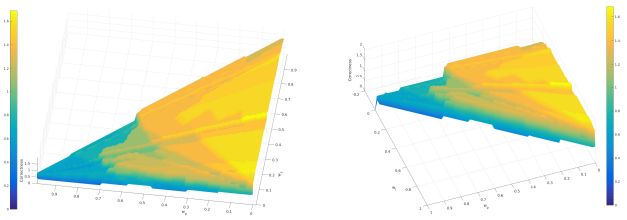

Fig. 8: A heatmap to show the correct matching number for different w vectors. The yellow region represent the high correctness.

segments, donated as α_r^{ij}, with the midpoints of the segments p_n^i and p_m^j as the rotation center, to transform the area A_i in the map A to the map B as B_i'. Then we calculate the overlap between the areas B_i' and B_j and use it as a criterion for judging whether the two areas are matched correctly. This process is illustrated in Fig. 7. Then the rotation angles between correctly matched areas are recorded in a list as rotation hypotheses.

2) Cluster the Rotations: We cluster these rotation hypotheses to vote the range in which the correct rotation between two maps is located. The clustering algorithm is based on the angular distance between the angles. The clustering process starts with setting the first rotation in the list as the center of a new cluster. Given another rotation in the list, if its distance from all centers is more than the set angle threshold (e.g., 3 degrees), then we create a new cluster centered on it, otherwise, add it into the cluster whose center is closest to it and recalculate the center of the cluster. Finally, we iteratively adjust the clusters to ensure that, for each cluster, the distance from the rotations in the clusters to the center does not exceed the set threshold. After the clustering is completed, the range of clusters that have the most angle hypotheses, donated as $Cluster_{max}$, is considered to be the range containing the correct map rotation.

3) Search for the Best Rotation: Given the best rotation cluster $Cluster_{max}$, we traverse its samples. For each sample $(\alpha_r^{ij}, (p_n^i, p_m^j))$, we transform all the areas in map A of matching area pair to match their corresponding areas in map B and sum up their overlap percentage. For a pair of matched regions (i, j), whose area values are a_i and a_j, their overlap area percentage OP is computed by

$$OP = \frac{Overlap\_Area}{\min(a_i, a_j)}.$$

Then we select the sample obtains highest overlap sum as our final transformation estimation between the given maps.

V. Experiments

A. Selection of w

As we mentioned in Section IV-C.1, the matching cost between two areas is obtained by multiplying the feature vector with the weight vector $w = [w_a \ w_p \ w_l]^\top$. Both area value and passage distance are essential to deciding the matchings. We want to know the weighting ratio of these three features in making decisions. Therefore, this experiment is done to select a good w. We run this experiment on Bormann's dataset [21], which includes large indoor environment maps and satisfies the assumption that the maps contain enough rooms.

In order to facilitate the search, and since only the weight ratio is important, we let the weight vector w have an element sum of 1 and set the search step size to 0.01 for each element. To find a good ratio between w_a and w_p, we use the w_a as X-axis, the w_p as Y-axis and the matching correctness of the best cluster as Z-axis to draw a heatmap. Also, we use the w_p as X-axis and the w_l as Y-axis to draw another heatmap. Both heatmaps are shown in Fig. 8. The correctness was obtained by dividing the correct area matchings in the best cluster by the hand-made ground truth.

According to Fig. 8, we find some regulation for the set of w to get a high correctness: w_a and w_p are positively correlated; w_p and w_l are negatively correlated; the lower the w_p was, the higher the correctness was obtained. Therefore, we set $w = [0.1 \ 0.1 \ 0.8]^\top$ to run the comparison in the next experiment, which satisfies the regulation.

B. Comparison with the State-of-the-Art Method

We chose Shahbandi and Magnusson's work [11] to compare with our method. It is a state-of-the-art matching method, which also depends on segmentation. We observe that their method performed better on the maps with strong self-similarity for the reason that it exhausts the search space to find the best solution, which makes their algorithm unsuitable for matching large environments. However, our algorithm is difficult to match the areas through feature matching for high self-similarity maps. More seriously, these maps can fool the overlap calculations, which lead to the wrong hypotheses selection. Their dataset included maps with high self-similarity and the maps with few rooms, contrary to our assumption, so we did not use their dataset.

In this experiment, we compared the methods on Bormann's dataset [21] with layout or artificial maps and robot maps, where one of the map in a pair is transformed so that we have rotation to estimate. The matching result of the two methods are compared with the ground truth are shown in Fig. 9. Besides, we also compare the computation time of the two methods and present the results in Table I. All the experiments are finished on a PC with a Intel Core i7-6700 CPU (3.40GHz $\times$ 8) and 16 GiB memory. The single threaded C++ code is running on the Ubuntu 16.04 operating system. The computation speed experiments provide separate analysis for the segmentation/ Area Graph generation time, matching time and total run time in the dataset. Using the Area Graph generation to segment the maps, the parameter width $w(m)$ to segment the different

maps was set to $\{1.8,\ 2.5,\ 1.5,\ 2.3,\ 2.1,\ 1.8\}$. Though our method can deal with the maps with different scales by inputting resolutions, the other method is not good at this. We scale the input to the same scaling.

Since our matching is effected by the initial sample of the clustering algorithm, for each pair of input maps, we run our matching algorithm twenty times and recorded the correct match rate, as shown in Table I. One can observe that our method achieves a higher correctness. It can be seen that the other method has serious rotation errors: 180° in the first map, 90° in the second and third map. Also, our method shows a faster computation speed for segmentation and matching for all maps. Due to the reason that Shahbandi et al. don't use any geometric feature of the regions to match corresponding regions between maps, but estimate alignment transformation with the "Least-squares estimation" method for each pairwise regions' Oriented Minimum Bounding Boxes, about 90% of the hypotheses are rejected. This process leads to long computation times for large maps. Therefore, our method performs better on large or complex maps.

VI. Conclusions

In this paper, we proposed a fast 2D map matching method based on area feature matching, where we utilized an important characteristic of indoor architecture, that the location of doors and intersections is always fixed, as a novel area matching feature: passage distance. Other features are the area size and the convex hull longest length. We use the weighted sum of those features to find the best matches between areas. Using the overlap of good matched area candidates we finally extract the correctly matched areas and their transformation. The experiment compares our method with another area matching based. The experiments show that our method performs better on maps with a large quantity of rooms, which are normal in real buildings. Our algorithm also has a faster computation time.

As future work we will apply our algorithm to map evaluation.

References

[1] D. Kakuma, S. Tsuichihara, G. A. G. Ricardez, J. Takamatsu, and T. Ogasawara, "Alignment of occupancy grid and floor maps using graph matching," in *2017 IEEE 11th international conference on semantic computing (ICSC)*. IEEE, 2017, pp. 57–60.

[2] C. Georgiou, S. Anderson, and T. Dodd, "Constructing informative bayesian map priors: A multi-objective optimisation approach applied to indoor occupancy grid mapping," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 274–291, 2017.

[3] S. Schwertfeger and A. Birk, "Map evaluation using matched topology graphs," *Autonomous Robots*, p. 1–27, 2015.

[4] S. Carpin, "Fast and accurate map merging for multi-robot systems," *Autonomous Robots*, vol. 25, no. 3, pp. 305–316, 2008.

[5] A. Birk and S. Carpin, "Merging occupancy grid maps from multiple robots," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1384–1397, 2006.

[6] S. Carpin and A. Birk, "Stochastic map merging in rescue environments," in *Robot Soccer World Cup*. Springer, 2004, pp. 483–490.

[7] S. Carpin, A. Birk, and V. Jucikas, "On map merging," *Robotics and autonomous systems*, vol. 53, no. 1, pp. 1–14, 2005.

[8] S. Schwertfeger and A. Birk, "Evaluation of map quality by matching and scoring high-level, topological map structures," in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 2221–2226.

[9] W. H. Huang and K. R. Beevers, "Topological map merging," *The International Journal of Robotics Research*, vol. 24, no. 8, pp. 601–613, 2005.

[10] J. O. Wallgrün, "Voronoi graph matching for robot localization and mapping," in *Transactions on computational science IX*. Springer, 2010, pp. 76–108.

[11] S. G. Shahbandi and M. Magnusson, "2d map alignment with region decomposition," *Autonomous Robots*, vol. 43, no. 5, pp. 1117–1136, 2019.

[12] J. Hou, Y. Yuan, and S. Schwertfeger, "Area graph: Generation of topological maps using the voronoi diagram," in *2019 IEEE International Conference on Advanced Robotics (ICAR)*, 2019.

[13] K. Konolige, D. Fox, B. Limketkai, J. Ko, and B. Stewart, "Map merging for distributed robot navigation," in *Proceedings 2003 IEEE/RSJ international conference on intelligent robots and systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 1. IEEE, 2003, pp. 212–217.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[15] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[16] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[17] S. Saeedi, L. Paull, M. Trentini, M. Seto, and H. Li, "Efficient map merging using a probabilistic generalized voronoi diagram," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 4419–4424.

[18] ——, "Group mapping: A topological approach to map merging for multiple robots," *IEEE Robotics & Automation Magazine*, vol. 21, no. 2, pp. 60–72, 2014.

[19] ——, "Map merging for multiple robots using hough peak matching," *Robotics and Autonomous Systems*, vol. 62, no. 10, pp. 1408–1424, 2014.

[20] S. Schwertfeger and T. Yu, "Matching paths in topological maps," in *9th Symposium on Intelligent Autonomous Vehicles (IAV), IFAC*, IFAC. IFAC, 2016.

[21] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hagele, "Room segmentation: Survey, implementation, and analysis," pp. 1019–1026, 2016.

[22] Z. He, J. Hou, and S. Schwertfeger, "Furniture free mapping using 3d lidars," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2019.

[23] H. Chen, X. Zhao, J. Luo, Z. Yang, Z. Zhao, H. Wan, X. Ye, G. Weng, Z. He, T. Dong, and S. Schwertfeger, "Towards generation and evaluation of comprehensive mapping robot datasets," in *Workshop on Dataset Generation and Benchmarking of SLAM Algorithms for Robotics and VR/AR, 2019 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE Press. IEEE Press, 2019.

[24] H. Chen and S. Schwertfeger, "Heterogeneous multi-sensor calibration based on graph optimization," in *2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, IEEE. IEEE, 2019.

TABLE I: The comparative analyses of computation times between two methods.

	freiburg 101		intel		lab_a		lab_c		lab_d		lab_f		Average	
	Shahbandi et al.	Ours	Shahbandi et al.	Ours	Shahbandi et al.	Ours	Shahbandi et al.	Ours	Shahbandi et al.	Ours	Shahbandi et al.	Ours	Shahbandi et al.	Ours
Segmentation/Arrangement Time (s)	4.77	**1.37**	9.15	**5.91**	8.75	**8.21**	5.36	**2.77**	15.23	**7.97**	25.20	**12.65**	11.41	**6.48**
Matching Time (s)	1.41	**0.004**	3.85	**0.085**	4.56	**1.715**	3.05	**0.069**	18.91	**0.049**	200.96	**0.192**	38.79	**0.352**
Total Times (s)	9.23	**1.39**	15.93	**6.00**	16.43	**9.93**	11.10	**2.84**	39.74	**8.02**	231.79	**12.85**	54.04	**15.32**
Correctness (%)	0	**100**	0	**90**	0	**100**	**100**	**100**	**100**	**100**	**100**	95	50	**97.5**

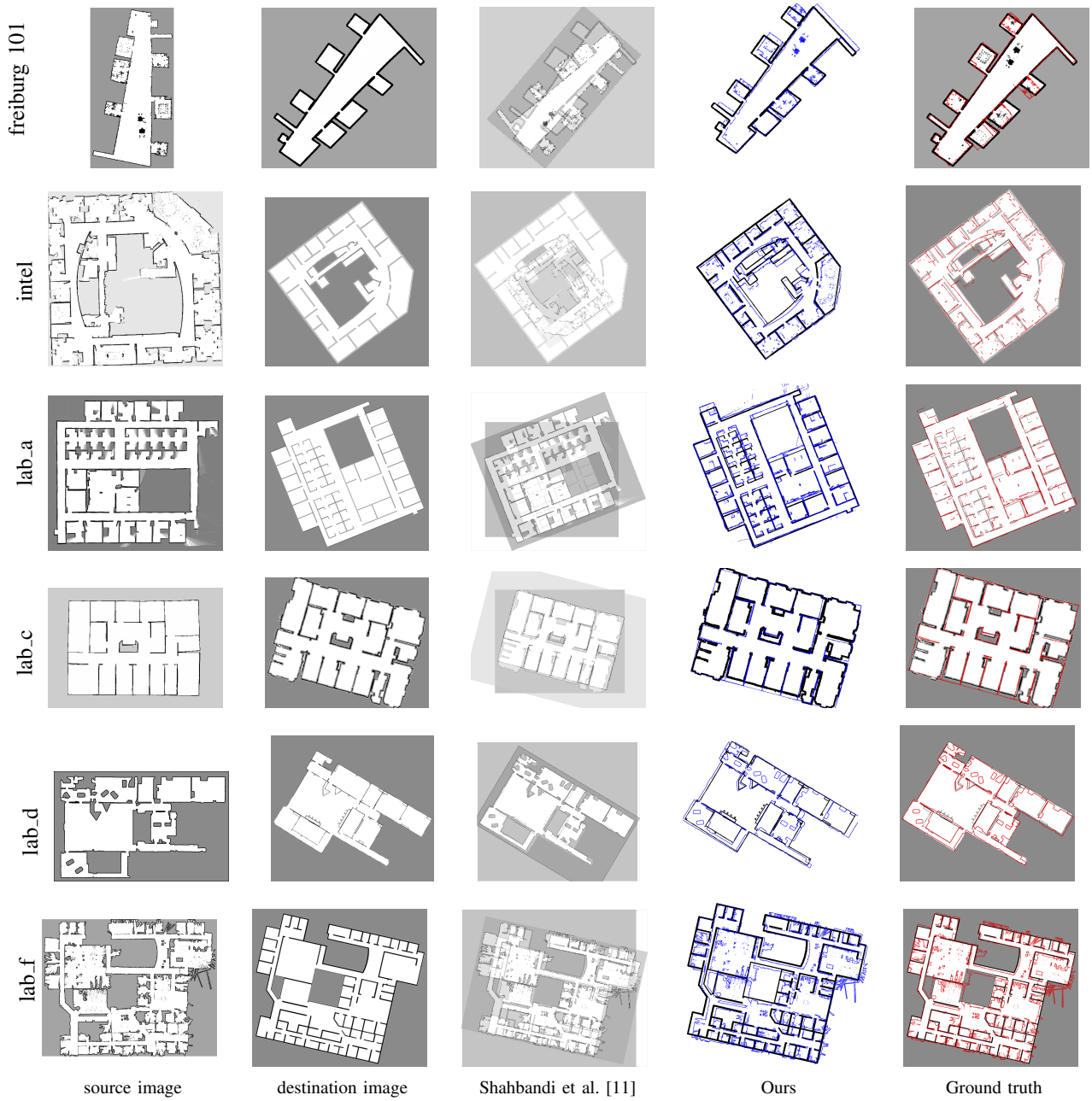

Fig. 9: Experiment result to show the performance of our algorithm. The source maps are occupancy grid maps which were generated by 2D SLAM algorithms. The destination map is a handmade layout image. And the ground truth is generated by manual alignment. The result shows that our algorithm has a better performance compared to the state-of-the-art method.