

Evaluation of Map Quality by Matching and Scoring High-Level, Topological Map Structures

Sören Schwertfeger and Andreas Birk¹

Abstract—Mapping is an important task for mobile robots. But assessing the quality of maps in a simple, efficient and automated way is not trivial and an ongoing research topic. A new approach on map evaluation is presented here. It is based on Topology Graphs as a topological, abstracted representation of 2D grid maps. The Topology Graphs are derived from Voronoi Diagrams that get post-processed to capture the high-level spatial structures. Based on a similarity metric on vertices in Topology Graphs, the vertices can be matched across maps and spatial (dis)similarities and hence errors in the mapping can be identified and measured. More precisely, the vertex-similarity is the basis to match the structures of Topology Graphs up to the identification of subgraph isomorphisms through wave-front propagation. This allows to determine important map quality attributes up to very challenging structural elements like brokenness, i.e., the number of locally correct partitions in the candidate map and their relative placement towards each other. Experiments with real robot generated maps including examples from various teams in the RoboCup Rescue competition are used to validate the usefulness of this method for map quality assessment.

I. INTRODUCTION

Maps are often essential to enable the robot to perform its tasks, for example for autonomous navigation and path planning. Maps also assist an operator of a remotely teleoperated robot in locating the robot in the environment by providing information of features of interest (corners, hallways, rooms, etc.). It is hence of interest to be able to assess the quality of maps and mapping approaches, to identify working solutions as well as open problems.

Maps generated by mobile robots are abstractions of the real world, which always contain inaccuracies or errors. There has been great progress in mapping in the last two decades, especially with respect to Simultaneous Localization and Mapping (SLAM) techniques. But especially on extended missions or in unstructured environments, maps still often contain large errors. Furthermore, the usefulness of a map not only depends on its quality but also on the application. In some domains certain errors are negligible or not so important. That is why there is not just one measurement for map quality. Different attributes of a map should be measured separately and weighed according to the needs of the application [1]. Those attributes can include:

- Coverage: How much area was traversed/visited.
- Resolution Quality: To what level/detail are features visible.
- Global Accuracy: Correctness of positions of features in the global reference frame.
- Relative Accuracy: Correctness of feature positions after correcting (the initial error of) the map reference frame.
- Local Consistencies: Correctness of positions of different local groups of features relative to each other within each group.
- Brokenness: How often is the map broken.

Note that the Resolution Quality is not only dependent on the actual size of the grid cells of the map but is also influenced by the quality

of the localization. If there are pose errors between scans of the same object, its features blur or completely vanish.

Ground truth information is typically used for map evaluation. There are notable exceptions that solely rely on the appearance of a single map, e.g., [2] where suspicious and plausible arrangements of 3D planes are considered. But if an objective analysis, e.g., for a comparison of mapping algorithms, is required then subjective appearance criteria have their limits - it can never be fully ensured whether the appearance criterion really holds in the experimental environment unless the ground truth is studied as well.

One option is to use ground truth robot paths, e.g., [3] and [4] compare those paths with those estimated by SLAM algorithms. But obtaining highly accurate robot paths can be a difficult problem. Using ground truth maps can be easier and also allows for the comparison of maps generated by different robots that may have gone different paths, used different sensors, and so on.

Therefore, most other approaches to map evaluation use accurate maps of the environment as ground truth, which is particularly easy if simulations are used [5], [6]. One can, for example, measure the alignment error of virtual scans in the ground truth map [7] or use image-processing techniques. But image similarity methods [8] that try to register maps or parts of them with image-processing have their limitations. Though this can even be used in the best case to determine certain attributes like the level of brokenness [9] it is always a pixel-level processing that suffers from local noise and smaller errors that severely challenge the registration of matching regions of ground truth and the evaluated map.

An alternative is to try to detect features or places in the maps. The Harris corner detector, the Hough transform and SIFT are for example used in [10] while SURF and a room detection approach are applied in [11]. Those approaches have in common, that they use the pose of the detected features to determine the map quality by matching them to their counterparts in the ground truth. To ease this process, there is also the option to use artificial markers in the environment, so called fiducials [12], [13]. Just the positions of those fiducials has to be known to calculate the map attributes mentioned above. The main disadvantage of this approach to map evaluation is, that the mapped area has to be populated with those markers.

Here we propose the use of graphs that try to capture the topological structure of maps - hence Topology Graphs. The graphs are derived from post-processed Voronoi Diagrams. We present a way to first match the Topology Graphs of two maps and then evaluate this match to get a similarity metric. It is shown through a validation with real world maps, among others from various teams from RoboCup Rescue competitions and response robot field tests, that the metric allows to assess the quality of maps as it captures important quality aspects. All maps used in the evaluations of our algorithms have been produced by teams unrelated to the authors in various test scenarios designed by third parties.

The rest of this paper is structured as follows: First the Topology Graph is introduced and it is described how it is created and

¹Both authors are with the Department of Electrical Engineering and Computer Science, Jacobs University Bremen, 28759 Bremen, Germany [s.schwertfeger, a.birk] at jacobson-university.de

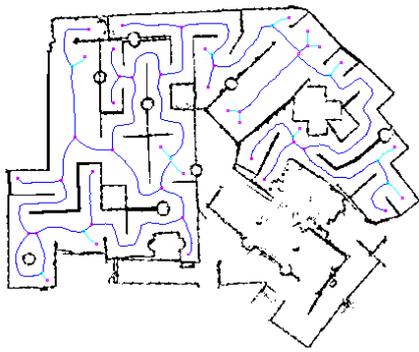


Fig. 1. A Topology Graph in a map generated by one of the teams at the RoboCup Rescue 2010 competition. Turquoise edges and vertices are spurious edges or vertices.

attributed. Then the matching of two Topology Graphs is explained, concentrating on the algorithm to find isomorphisms. Next it is described how a match of the Topology Graphs of a ground truth map and a robot map can be used to measure the quality of the robot map. An experiment is then showing the performance of the map brokenness computation, followed by the conclusions.

II. TOPOLOGY GRAPH

The map evaluation algorithm presented here makes use of what we call Topology Graphs, as they are designed to represent the topological structure of the environment. In the end, the topologies of a ground truth map and the robot-generated map are compared and their differences analyzed. The topological information we are interested in are passage ways like hallways, their junctions and dead ends, and their general connectivity.

Generating the Topology Graph is a form of skeletonization. The input to the algorithm is a two dimensional grid-map which is colored to occupied and free space [13]. After some simplification and pruning, the desired Topology Graph can be extracted from the Voronoi Diagram. As an example of the desired output, Figure 1 shows a map and its Topology Graph; the data is from one of the teams participating in the RoboCup Rescue Competition 2010.

A. Voronoi Diagram

The creation of the Topology Graph is based on the Generalized Voronoi Diagram (GVD). It is a partition of the space into cells. The cells enclose a site, and in this application the site is an obstacle point from the map. The Voronoi cell associated with a site is the set of all points in whose distance to the site is not greater than their distance to all other sites. In this work the interest is not so much in the Voronoi cells but in the graph that is defined by the boundary of said cells.

The generation of the Topology Graph involves several post-processing steps that are illustrated using an example map (see Figure 4 for a complete version together with the resulting Topology Graph), which was generated by a team in an experimental setting known as the random maze at the Response Robot Evaluation Exercise (RREE) 2010. The map has 4398 points and a resolution of 5cm per pixel.

B. Creating the Topology Graph

The input points for the computation of the Voronoi Diagram (VD) are the center points of the occupied cells from the grid map. CGAL is used for that [14]. In the first step edges whose geometrical distance to the closest obstacle (as determined during

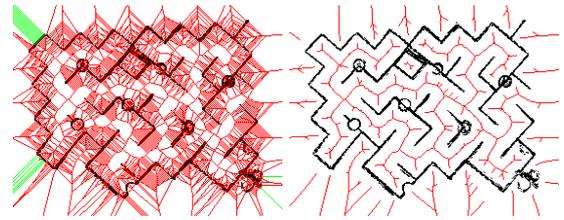


Fig. 2. The CGAL Voronoi graph. A nearly unfiltered graph (left) and the graph filtered for the distance to the obstacles (right). A minimum distance of 30cm is configured. This reduces the number of CGAL half edges from 10280 to 2352.

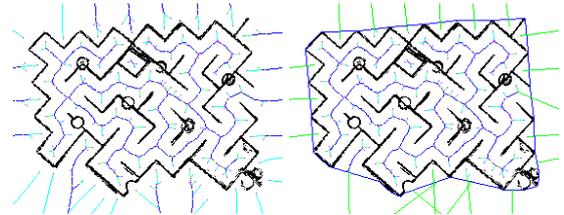


Fig. 3. An unfiltered Topology Graph (left). Turquoise edges lead to dead end vertices. The left graph has 291 Vertices, 584 half edges. The graph filtered for all vertices outside the alpha shape enclosing the map is shown on the right. Ray edges are colored green. An alpha value of 2500 (2.5 m) was used, resulting in an alpha polygon consisting of 133 segments. The filtered graph features 211 vertices and 424 half edges.

the VD generation) is shorter than a certain threshold are filtered out (see Figure 2). The threshold is chosen according to the minimum width of hallways/openings that should be represented in the Topology Graph.

The initial Topology Graph is generated by looking at the vertices from the Voronoi Diagram. All vertices with a degree of one or more than two are used as vertices in the Topology Graph. The Topology Graph edges are then created according to the edges of the VD, merging together edges to vertices that were not copied over to the Topology Graph (those vertices have a degree of two and are thus connected to exactly two edges, which are merged in turn).

The graph is then further filtered from short dead-end edges. Also, we are just interested in the topology inside the grid map, so the parts outside of the map are to be removed. This is achieved by first determining the outer shape of the map using the Alpha Shape algorithm [15]. The longest polygon generated by the Alpha Shape is the one defining the outer boundary. All vertices outside this polygon are removed. Figure 3 shows in the right image the alpha polygon and the filtered graph. Also edges and vertices that are not part of the biggest graph are deleted. At last vertices in close proximity to each other are joined together. The new vertex gets the average position of the joined vertices. The edges coming from other vertices to the joined vertices are connected to the new one.

C. Attributing the Topology Graph

Several attributes for the edges and vertices of the Topology Graph are calculated. Those are used during the later steps of vertex similarity calculation and the graph matching.

The length of the path along an edge is calculated. As can be seen for example in Figure 1, the path of an edge between two vertices often does not follow a straight line but is winding along the corridor. Other attributes for the edges are the average and minimum distance of the path to the obstacles. Dead end edges (edges that are

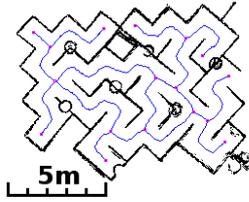


Fig. 4. The resulting Topology Graph with 17 vertices and 36 half edges. The added length of all edges (only counting one of the two half edges forming an undirected edge) is 52 m.

connected to a vertex with a degree of one) are marked. Dead end edges that are shorter than a certain threshold and their vertices are marked as “spurious”. This is because for different maps of the same environment, those might or might not be created in the Topology Graphs, due to minor differences in the map. During the graph matching those vertices could then be ignored.

The edges connected to a vertex are not just saved as an unordered set. Instead they are put into a ring buffer, according to their incidence angles to that vertex. The ring buffer does not store global angles but only the difference between neighboring angles, thus making this information rotation independent. The vertex is then attributed with the smallest and biggest angle between two neighboring vertices from the ring buffer.

D. Stability of the Topology Graph

The stability of the Topology Graph is important - the graphs have to be similar such that the graph matching can work. The pure Voronoi Diagram might have problems with stability, for example with two doors on opposite sides of a corridor, where one or two vertices might be created and the order of the vertices might also differ. But due to the merging of vertices in close proximity, those ambiguities are avoided in the Topology Graph.

So the stability is generally very good in “normal” environments where there are at least some obstacles. Only in large open areas, minor changes in the environment might lead to drastic changes in the Topology Graph, due to the nature of the underlying VD.

The experiments performed by the authors at different field tests and competitions with various 3rd party maps show that the graphs of the same, real life environments, represented by different grid maps, are very similar even if the resolution or amount of errors differ.

III. MATCHING OF TWO TOPOLOGY GRAPHS

In order to compare the Topology Graph of the ground truth map and the one from the robot-generated map, those graphs have to be matched against each other. The matching is done on the vertices as well as the graph structure. The positions of the vertices in the map are not used at all for this matching step, because those can be severely wrong due to the errors in the map.

A. Vertex Similarity

Calculating some similarity between two vertices of two different graphs is a first step for matching the graphs - this is done between all vertices from both graphs. The concept of two approaches is shortly presented here.

The first approach calculates a similarity value by comparing the values of attributes of the vertices. Those attributes were generated while building the graph. Those are, for example, the number of edges connected to a vertex, the biggest and smallest angle between

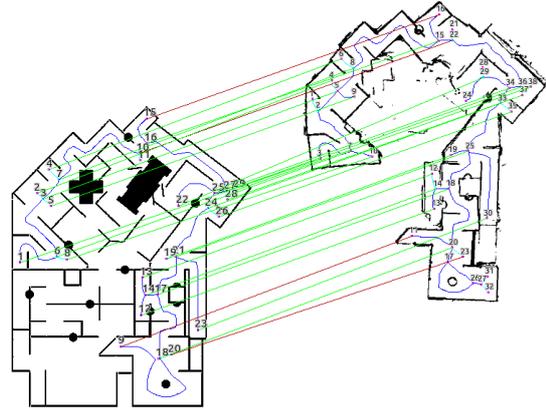


Fig. 5. Example match between a ground truth map (left) and a robot generated map (right) using the isomorphism algorithm and then the grow neighbors approach. There are 28 assignments made and one assignment is missing from the left graph (vertex 11).

the edges of a vertex, the lengths of those edges, the minimum and average distance to obstacles along the edges and to the vertex, etc.

The second approach takes all obstacle points within a certain radius to the vertex from the grid map and compares those two point clouds from the two vertices using the Iterative Closest Point (ICP) algorithm [16]. Since ICP is sensitive to the initial rotation of the point clouds, the algorithm is started with a number (e.g. 12) of different starting angles. The lowest mean squared error of the ICP runs is then used to calculate the similarity between the vertices.

The results of both approaches can be improved by taking the similarities of the neighboring vertices into account. The first approach can be computed many orders of magnitude faster than the ICP approach, while delivering only slightly less accurate results. If computation time is not such a big issue, the ICP method (for which the computation for one pair of graphs is typically less than one minute) might be preferred.

B. Find Isomorphisms

Two Topology Graphs are matched by first finding an isomorphism and then applying a neighbor growing approach. Finding an isomorphism is a quite fast and strict way to find parts of the maps that do not diverge in the connections of the graph. The second, more heuristic approach is the neighbor growing algorithm which makes heavy use of the vertex similarity. It partially ignores the constraints that guide the isomorphism algorithm and uses a similarity value for the whole match to guide the search.

The Graph Isomorphism is a concept of graph theory. We are only looking for subgraphs from both graphs that form an isomorphism, which is called the maximum common subgraph isomorphism problem. In our concrete application we deal with planar graphs (no edges intersect on a 2D plane) and we have a very good heuristic (the Vertex Similarity), thus easing the solution. But the graphs may contain spurious edges and vertices which may have to be ignored during the matching as they are random artifacts, which makes the problem harder. All these factors motivate why we developed a custom solution for the graph matching problem at hand. There are two rules for finding the isomorphism in our method, namely the number and geometric order of the connected edges of the vertices matched must be correct and the length of the edges must coincide at least to a certain degree.

For every vertex (which has not yet been assigned), it is tried to find a new isomorphism starting at said vertex v_x . This works by

Algorithm 1 The Wavefront Propagation

```
function wavefrontProp(wavefront, currentVertexAssignment)
while not wavefront.empty() do
  currentVertexPair = wavefront.pop()
  goodAssignments = getGoodAssignments( currentVertexPair, cur-
  rentVertexAssignment )
  if goodAssignments.size() == 1 then
    addToVertexAssignmentAndWavefront( goodAssign-
    ments.top(), wavefront, currentVertexAssignment )
    continue
  else if goodAssignments.empty() then
    continue
  end if
  // goodAssignments has more than one entry - recursion!
  for all assignment in goodAssignments do
    wavefrontCopy = wavefront
    assignmentCopy = currentVertexAssignment
    addToVertexAssignmentAndWavefront( assignment, wavefront-
    Copy, assignmentCopy )
    wavefrontPropagation( wavefrontCopy, assignmentCopy )
  end for
end while

function getGoodAssignments( currentVertexPair, currentVertexAs-
  signment )
  varGoodAssignments.empty()
  edgeAssignmentTable = vertexTable[ currentVertexPair.first ][ cur-
  rentVertexPair.second ].edgeTable
  for all edgeAssignment in edgeAssignmentTable do
    if checkedEdgeAssignmentForConsistency( edgeAssignment, cur-
    rentVertexAssignment ) then
      varGoodAssignments.insert( edgeAssignment )
    end if
  end for
  return varGoodAssignments
```

trying to match this vertex with every vertex from the other graph that has the same number of connected edges.

The Wavefront Propagation algorithm explained below is then applied to this pair. Should this algorithm result in a found isomorphism, it is checked if the number of matched vertices in this assignment is at least as big as a threshold. If this is the case, the result is accepted. The biggest accepted assignment for v_x is then added to the final vertex assignments. Typically there will be just one accepted assignment for v_x , since the algorithm will only return very good matching isomorphisms. Only if there are large scale repetitive patterns in the environment that are reflected in the topology graph, taking the maximum match can help finding the correct match.

1) *Wavefront Propagation*: In the Wavefront Propagation algorithm, vertices that have edges to other, not yet matched vertices form the wavefront. In the beginning, just the proposed pair is in the frontier.

In Algorithm 1 an edge assignment table is used in the function `getGoodAssignments`. One edge assignment is the pairwise matching of edges of one vertex from the first graph with edges of the matched vertex from the second graph. The edge assignment table consists of all permutations of those matches.

`checkEdgeAssignmentForConsistency` tests if one edge assignment is consistent with the current vertex assignment. This is only true when the edges are equal in number and are matched in the correct (geometric) order. Otherwise the function immediately returns false. Then it is checking if the vertices that the edges are leading to are already matched and if so, if they



Fig. 6. Eight example maps (map 1 to map 8 from left to right and from top down) generated in the context of RoboCup Rescue. Map 1 is shown in big in Figure 9.

are matched with the correct vertices. Additionally, the length of the edges are compared. The recursive step only happens for the first match inserted into the wavefront, since afterwards `getGoodAssignments` will return at most one assignment. This is because all new matches added to the wavefront will already have at least one matched neighbor (from which they were added). Since the matching of the edges is strict with respect to the order of edges connected to a vertex, at most one assignment can be returned.

The found graph match can be extended by a neighbor growing approach. It is also a wavefront approach which has relaxed constraints and is heavily guided by the vertex similarity. Figure 5 shows an example match between the ground truth map from RoboCup and a robot generated map using the isomorphism and neighbor growing approaches.

IV. MAP EVALUATION USING MATCHED TOPOLOGY GRAPHS

Once the matching of the structures of two Topology Graphs is done this still has to be turned into a metric for assessing their similarity. Evaluating the map quality after matching the Topology Graph G of the map with the Topology Graph G' of a reference map is similar to the approach of the Fiducial Map Metric [13]. The map attributes are calculated as follows:

Coverage: Calculate the percentage of vertices from the ground truth Topology Graph matched to vertices from the graph of the robot map.

Global Accuracy: Correctness of positions of the matched vertices in the global reference frame.

Relative Accuracy: Correctness of positions of matched vertices after correcting (the initial error of) the map reference frame using Horn's algorithm [17].

Local Consistencies: The shortest graph distances between any two vertices in the ground truth graph are calculated. All vertex pairs are permuted. The pairs are put into different difficulty classes according to their distance over the graph. The geometric distance between the two vertex positions is calculated and compared to the geometric distance between the two matched vertices from the robot generated map. The errors of those comparisons are averaged for each of the difficulty classes.

Brokenness: It is defined as "the degree with which a map can be partitioned into regions that are locally consistent with ground truth but 'off' relative to each other" [9].

From the experience of the authors from field tests like RREE or competitions like RoboCup Rescue, even state of the art SLAM

TABLE I

COMPARISON: TOPOLOGY MAP EVALUATION TO HUMAN ASSESSMENT.

Map #	Human Rank	Top.Gr. Rank	Cov.	Rel. Acc.	Consistency		
					Short	Med.	Long
5	1-2	2	2	1	2-3	1-2	1
6	1-2	1	1	2-3	1	4	2
1	3	4	3	4	4	1-2	5
4	4	3	4	2-3	2-3	3	3
3	5	5	5	5	6	6	7
2	6-8	6	6	7	7	7	4
7	6-8	7	7	6	5	5	6
8	6-8	8	8	8	8	8	8

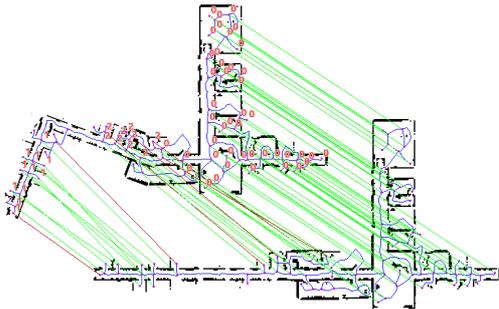


Fig. 7. The Ground Truth Map (Map 0 - bottom) matched to Map 2 (top), which has two broken areas. The green matches are made by the Topology Graph Isomorphism and the red matches by the Neighbor Growing. The numbers represent the brokenness group the vertex belongs to.

algorithms occasionally produce maps with broken regions. One example of a real-world case of a broken map can be seen in Figure 9. This map was generated from real sensor data. The upper part of the map is slightly broken, as can be seen V-shaped wall indicated with the arrow.

Calculating the brokenness using a Topology Graph that is matched against a ground truth Topology Graph is relatively simple. A subset of connected vertices, a *Brokenness Group*, has to be found whose minimal mean square error regarding the location of the vertices compared to the matched ground truth vertex locations does not exceed a certain threshold. This is done using a Wavefront Propagation algorithm. The minimal mean square error is easily computed using Horns algorithm. This subset is saved. The process is iterated over the remaining vertices that are not yet part of a subset. A subset is only accepted if it contains at least a certain number of vertices. The number of subsets found minus one is then the brokenness degree as defined in [9].

V. EXPERIMENTS

First the results of the approach is compared to a human assessment. It can be noted that the proposed metric generates rankings of map qualities that are similar to human judgments. Figure 6 shows eight maps generated in the context of RoboCup Rescue - the very good map from Figure 1 is used as the ground truth map. Table I shows a human ranking of the quality of the maps as well as their automatic scoring.

In a second experiment, it is shown that Topology Graphs can even be used to compute challenging map quality attributes like brokenness. The six maps that were used in [9] for brokenness experiments are also used here. Map 0 (Figure 7) serves as the ground map while Maps 1 to 5 (Figures 7 and 8) are applied to the map metric. The computation is done in pixel coordinates, because

TABLE II

MATCHING OF TOPOLOGY GRAPHS WITH GROUND TRUTH.

Map	# Vertices	# Half Edges	Matching Vertices with Ground Truth		
			Isomorphism	Neighbour Growing	Missing
0	75	176	-	-	-
1	74	172	58	8	8
2	76	178	61	6	8
3	79	188	49	9	17
4	79	186	50	9	16
5	79	186	47	9	19

the scaling factor to real world was not encoded in the images. The configuration values are 80 for the SquaredErrorThreshold and 5 for the MinNumberVertices value.

In Figure 7 Map 2 is shown together with the matches to the ground truth map and the found Brokenness Groups. The numbers represent the Brokenness Group the vertex belongs to. The biggest group with 44 matches (which also means 44 vertices per map) is number 0 - this part represents the unbroken map. Horns algorithm also calculates the transformation between two point sets. The angle from this transformation is -69.5 degrees for set 1 and 21.8 degrees for set 2, which is correct.

Table II gives statistics about the matching between the ground truth and the broken maps. In Table III the different Brokenness Groups for the matches are presented. Maps 1 and 2 deliver the expected result and detect a brokenness of 1 and 2, respectively. For maps 3, 4 and 5 the left-most brokenness cannot be detected. It is very small and only represented by three vertices. Since MinNumberVertices is 5, this broken part cannot form its own set. All other broken parts from maps 3, 4 and five are nicely detected.

In Table IV the results for the other map attributes can be found. The inherent ranking of the maps (maps with higher numbers being more often broken and thus worse than lower number maps) is nicely reflected in the Consistency attributes. The coverage of all maps, including the ground truth, is the same. Due to the brokenness some vertices cannot be matched and thus the coverage is not 100% for the maps. But still the value is and remains quite high for the different maps.

One can see that the Relative Accuracy does also not change too much with the level of brokenness. This is because sometimes the next brokenness is bend towards the "correct" direction, thus

TABLE III

THE DIFFERENT BROKENNESS GROUPS.

Map	Detected Brokenness	Set Number	# Matches	Squared Error	Angle in $^{\circ}$
1	1	0	44	20.8	0.0
		1	18	57.3	20.6
2	2	0	44	58.5	0.1
		1	11	8.9	-69.5
		2	9	79.9	21.8
3	2	0	35	68.5	0.1
		1	8	69.9	23.4
		2	6	44.4	-70.8
4	3	0	28	77.7	0.3
		1	10	79.7	7.4
		2	9	79.9	21.8
		3	7	77.3	-70.5
5	4	0	19	76.8	0.7
		1	9	71.4	7.4
		2	9	79.9	21.8
		3	7	77.3	-70.5
		4	6	53.8	7.8

TABLE IV
BROKENNESS EXPERIMENTS RESULTS.

Map	Coverage	Relative Accuracy	Consistency		
			Short	Medium	Long
1	0.88	0.71	0.95	0.88	0.41
2	0.89	0.62	0.94	0.84	0.29
3	0.77	0.63	0.87	0.68	0.25
4	0.79	0.63	0.86	0.70	0.17
5	0.75	0.59	0.82	0.53	0.13

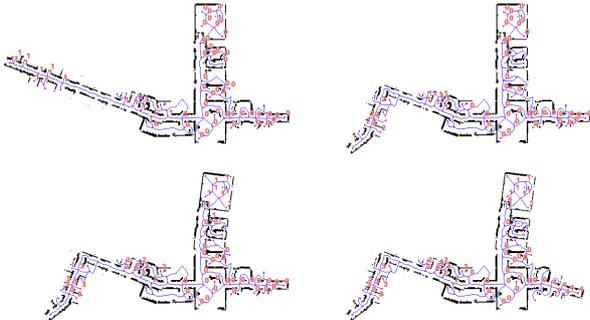


Fig. 8. The brokenness groups (1, 3, 4 and 5) for the different maps.

potentially even improving the Relative Accuracy. As expected, the Consistency attributes are the strongest indicator for the brokenness. Three Local Consistency attributes have been calculated: One for graph-distances between two vertices from the ground truth graph (Map 0) of less than 60 pixel (short), one for distances between 60 and 180 pixel (medium) and one for higher distances (long). The medium and long range Consistency values significantly decrease in value with every brokenness that is added to the map. Since the broken parts do not have any other errors, the short range Consistency is quite high and only slowly decreasing (due to the errors around the start areas of the broken parts).

Another example of a nicely detected brokenness is shown in Figure 9. Here a small brokenness of 7.6 degrees was detected.

VI. CONCLUSIONS

A novel approach for map evaluation using Topology Graphs has been proposed in this paper. The steps to building the Topology Graph representation of a 2D grid map and how to match two Topology Graphs have been shown. The different map evaluation attributes have been defined and methods on how to compute them using the matched graphs were proposed. The algorithms were demonstrated using real life examples - one experiment has been conducted that in particular highlights the assessment of a highly

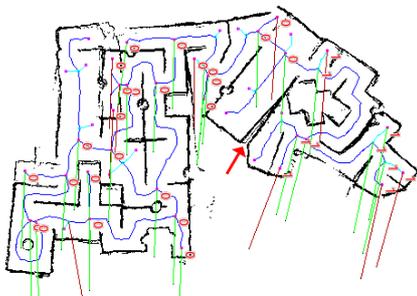


Fig. 9. Map from the RoboCup Mapping Challenge 2010 matched to the ground truth map from Figure 1. The parallel match-lines indicate a good match. The small map brokenness (see arrow at the V-shaped wall) was detected.

non-trivial map quality attribute in form of brokenness.

The main advantage of the Topology Map Metric is that it abstracts from the grid representation with the occupied cells and just works on the topology. It thus does not matter how the walls and obstacles are shown and small scale errors and artifacts on the walls do not effect the map score. Wrong initial orientations of the map and big broken parts are no problem to detect and compensate for in this map metric - those are very difficult situations for many image-based approaches. No detection of features, neither naturally occurring or artificially placed ones, is needed and the computation is fast enough for practical purposes - less than five seconds for each example in this paper on a standard laptop.

In the future an extension of the approach to 3D maps is planned.

REFERENCES

- [1] D. C. Lee, *The Map-Building and Exploration Strategies of a Simple Sonar-Equipped Mobile Robot: An Experimental, Quantitative Evaluation*, ser. Distinguished Dissertations in Computer Science. Cambridge University Press, 1996.
- [2] M. Chandran-Ramesh and P. Newman, "Assessing map quality using conditional random fields," in *Field and Service Robotics, Springer Tracts in Advanced Robotics*, C. Laugier and R. Siegwart, Eds. Springer, 2008.
- [3] O. Wulf, A. Nuchter, J. Hertzberg, and B. Wagner, "Ground truth evaluation of large urban 6d slam," oct. 2007, pp. 650–657.
- [4] R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner, "On measuring the accuracy of slam algorithms," 2009.
- [5] C. Scrapper, R. Madhavan, and S. Balakirsky, "Stable navigation solutions for robots in complex environments," in *IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR)*, 2007, pp. 1–6.
- [6] B. Balaguer, S. Balakirsky, S. Carpin, and A. Visser, "Evaluating maps produced by urban search and rescue robots: lessons learned from robocup," *Autonomous Robots*, vol. 27, pp. 449–464, 2009.
- [7] R. Lakaemper and N. Adluru, "Using virtual scans for improved mapping and evaluation," *Auton. Robots*, vol. 27, no. 4, pp. 431–448, 2009.
- [8] I. Varsadan, A. Birk, and M. Pfingsthorn, "Determining map quality through an image similarity metric," in *RoboCup 2008: Robot World-Cup XII, Lecture Notes in Artificial Intelligence (LNAI)*, L. Iocchi, H. Matsubara, A. Weitzenfeld, and C. Zhou, Eds. Springer, 2009, pp. 355–365.
- [9] A. Birk, "A quantitative assessment of structural errors in grid maps," *Autonomous Robots*, vol. 28, pp. 187–196, 2010.
- [10] A. I. Wagan, A. Godil, and X. Li, "Map quality assessment," in *PerMIS '08: Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*. New York, NY, USA: ACM, 2008, pp. 278–282.
- [11] J. Pellenz and D. Paulus, "Mapping and Map Scoring at the RoboCupRescue Competition," Quantitative Performance Evaluation of Navigation Solutions for Mobile Robots (RSS 2008, Workshop CD), 2008.
- [12] S. Schwertfeger, A. Jacoff, C. Scrapper, J. Pellenz, and A. Kleiner, "Evaluation of maps using fixed shapes: The fiducial map metric," in *Proceedings of PerMIS*, 2010.
- [13] S. Schwertfeger, A. Jacoff, J. Pellenz, and A. Birk, "Using a fiducial map metric for assessing map quality in the context of robocup rescue," in *International Workshop on Safety, Security, and Rescue Robotics (SSRR)*. IEEE Press, 2011.
- [14] M. Karavelas, "2D Voronoi diagram adaptor," in *CGAL User and Reference Manual*, 3.8 ed. CGAL Editorial Board, 2011, http://www.cgal.org/Manual/3.8/doc_html/cgal_manual/packages.html#Pkg:VoronoiDiagramAdaptor2.
- [15] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *Information Theory, IEEE Transactions on*, vol. 29, no. 4, pp. 551–559, jul 1983.
- [16] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb 1992.
- [17] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, vol. 4, no. 4, pp. 629–642, 1987.

© 2013 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other users, including reprinting/ republishing this material for advertising or promotional purposes, creating new collective works for resale or redistribution to servers or lists, or reuse of any copyrighted components of this work in other works.

Schwertfeger, S., and A. Birk, "Evaluation of Map Quality by Matching and Scoring High-Level, Topological Map Structures", IEEE International Conference on Robotics and Automation (ICRA): IEEE Press, 2013.

<http://dx.doi.org/10.1109/ICRA.2013.6630876>

Provided by Sören Schwertfeger
ShanghaiTech Advanced Robotics Lab
School of Information Science and Technology
ShanghaiTech University

<http://robotics.shanghaitech.edu.cn/people/soeren>
<http://robotics.shanghaitech.edu.cn>
<http://sist.shanghaitech.edu.cn>
<http://www.shanghaitech.edu.cn/eng>

File location

<http://robotics.shanghaitech.edu.cn/publications>