Map Merging of Oriented Topological Semantic Maps

Jose Luis Susa Rincon Stefano Carpin

Abstract— **In this paper we propose a solution for the problem of merging together partial spatial models relying on our recently introduced Oriented Topological Semantic Maps (OTSM). This problem arises when a group of robots cooperatively explore an environment, and each one independently builds a partial map that must be combined with the others into a full map. Our methodology is inspired by the Warrington's Object Recognition Model, a cognitive model hypothesizing two post-sensory categorical stages working together for object recognition. Accordingly, we use two stages to compare different maps and match them together based on their mutual resemblance. Our method is complemented by a scoring system to measure the likelihood that two vertices in different OTSMs correspond to the same vertex, despite possible errors in labeling, orientation, or topological structure. Our methodology is validated in a simulation informed by an ongoing real robot implementation, thus allowing us to perform various experiments with carefully controlled error sources.**

I. Introduction

Multi-robot research has been on the rise because in many instances multiple robots offer inherent advantages over solutions relying on a single robot. As a classic example, a group of coordinated robots would explore an unknown environment much faster than a single robot. This is a specific problem where the multi-robot solution introduces new challenges not found in the single robot approach. For individual robots, continuous progress in SLAM research has generated sophisticated solutions and in some instances this can be considered to be a solved problem. However, when a group of robots cooperatively explore and map an unknown environment, two approaches can be undertaken to combine the partial results. The robots can either jointly build a spatial model "on the fly," or they can individually build a single map, and then combine their partial models together a posteriori. This last problem, known as *map merging*, is not a direct extension of a single robot problem and is tackled in this paper. Map merging relates to other problems such as structural graph matching or sub graph isomorphism, depending on the type of model considered [7]. The most common practice is to create metric maps and use different techniques to merge them together [4], [10], [18], [3]. However, when the maps utilize a topological or semantic representation, a different approach is needed. In our recent work [17], we introduced a new type of maps called Oriented Topological Semantic Map (OTSM), and in this paper we study how multiple partial OTSMs can be combined when a team of robots cooperatively explore a common indoor environment. Our solution is inspired by the Warrington's object recognition model [19], a cognitive model that describes how humans recognize objects using a two layer system of perceptual and semantic categorization.

In their everyday activities, humans do not rely on metric maps to complete any of their tasks. Instead, a more concise representation to store the structural organization of buildings or other relevant structures is computed quickly and shared, arguably with little effort, with other humans for immediate use. Even in the absence of visual information, humans can still use the same efficient representation and interact with their environment and peers [11]. As we expect robots to operate side-by-side with humans, it makes sense to envision the same capabilities for robots, both for robot-robot interoperation, and to smooth the human-robot interface. In the past, some attempts to merge topological maps have been considered, but to the best of our knowledge this is the first bio-inspired model used to merge a topological map enriched with semantic information and without the use of any metric data. The contributions of this paper are three: 1) we propose a new two-stage method to compare vertices in different OTSMs and measure their resemblance; 2) we present a new merging technique to stitch OTSMs using a semantic and perceptual categorization; 3) we study (in simulation) four different types of errors that affect OTSMs and their impact when merging together partial maps.

The paper is organized as follows. Selected related work is introduced in section II. In section III we present the OTSM models, and then we focus on the algorithm to merge this type of maps. Simulations of our algorithm and its performance are shown in the results section V, and we finalize with conclusions and future work in section VI.

II. Related Work

The vast majority of SLAM related research embraces a single-robot approach. When increasing the number of robots, several challenges arise, like estimating relative poses of the robots, uncertainty of the relative poses, concurrent updates of maps and poses, communications, and others. Saeedi et al. [15] present a complete review for multi-robot SLAM research in the past years.

Early works in map merging aim at combining multiple grid maps into one global map using metric features. Our former work [4] used Hough features for the bidimensional case and the Radon transform for the three dimensional case [5]. Blanco et al. [1] use computer vision techniques to extract features and compare them, in what they call multi-hypothesis RANSAC stage, to match them later to the grid map. Paulik [13], on the other hand, merges a hybrid, feature-metric map finding a transformation matrix to match the

J.L. Susa Rincon and S. Carpin are with the Department of Computer Science and Engineering of the University of California, Merced.

pieces of maps. A function to measure the overlap between pixels is proposed and used to calculate the quality of the matching. Park et al. [12] use geometric information to match shapes and geometric features to find overlapping points between maps. This approach gives an important advantage when eliminating the need of knowing the relative position of the robots. Another method, implemented by Karpov [9], includes communication between the robots and a landmark system to localize the robots and find matches between the sections of map that each robot builds. For improving vehicle positioning, Rohani et al. [14] merge road maps between vehicles in a VANET network. The GPS error from each individual is taken into account to find the matches, and then a dynamic base station is used to help other vehicles to improve their location when they are not part of the network.

Merging topological maps is related to the problem of graph merging, because of the underlying graph representation. When trying to merge these graphs, we have some examples of how we can obtain a global map by combining topological with metric information. For example, Bonanni et al. [2] extract a graph from a 3D metric map and try to match the vertices of the graph. This is the opposite of the common approach of applying geometric transformations to each of the map sections to find the match between them. In our work, we do not need to know the global or the relative pose of the robots to merge the maps. Similarly, when merging topological maps for rectilinear worlds, Huang et al. [8] generate hypotheses of how two graphs match depending on the number of outgoing edges. Then, they augment the topological map with metric information (like local distance) while in our work we do not use any concept of distance.

Finally, there are works showing the advantages of using hybrid maps to encapsulate, at once, more information. Shahbandi et al. [16] propose a multi-modal map alignment where multiple maps of different types can be combined into a global model that contains occupancy grid maps, 3D meshes and also semantic information. Dichtl et al. [6] introduce a new type of map called PolyMap, where an environment is decomposed in polygons that stay in the middle of a grid map and a vector map, taking advantage of the strengths of both, and allowing efficient communication and sharing for multi-robot missions.

III. Background on Oriented Topological Semantic Maps and Warrington's model

A. Oriented Topological Semantic Maps

We here present a short review the *Oriented Semantic Topological Map*. Our recent work presents a more detailed insight about the OSTM as an extension to classic topological maps [17], as well as exploration algorithms to incrementally build them. The general idea of an OTSM is to define a semantically structured environment as a topological map whose edges connect the different locations (e.g. rooms and corridors) and can be traversed using human-like commands to turn "to the right of"and "to the left of," and follow actions like walk through a corridor and get inside/outside rooms. A topological map is modeled as a directed graph $G = (V, E)$, where vertices represent a location in a building (room, corridor), and directed edges connect the places with a direction to traverse them (hence the term *oriented* in the name). It is because of these relationships that we assume edges are directed (a feature will be on the right, or on the left, depending on the direction with which an edge will be traversed). Starting from the observation that in most human inhabited buildings walls and features are arranged orthogonally (as noted and exploited in [12]), we classify vertices based on the their degree, i.e., on the number of other vertices they are connected to: degree 1 are rooms, or corridor dead ends; degree 2 are corridors with only two ways; degree 3 are T intersections between corridors/rooms; and degree 4 are cross intersections.

Edges keep track of the direction in which they are located in relation to the robot orientation. Without loss of generality, we assume that walls and corridors (end therefore edges) are aligned along the four cardinal directions (indicated as N/S/E/W in the following). Every edge is then associated with a label like N-S or E-W to indicate their orientation. Two vertices are along the N-S (north-south) direction if the edge connecting them has the N-S label, and a similar reference is used for E-W direction. Accordingly, we assume the robot has access to a sensor (e.g., compass, IMU, or a combination) to determine its heading, and therefore assigns directions to edges while they are being discovered during the exploration. Navigation between vertices is built upon the concept of "to the right/left of." To define the relationships "to the right/left of" for elements aligned along the N-S direction, we assume that the robot faces W, whereas for elements along the E-W direction we assume the robot faces N. This distinction makes the OTSM more akin human-like spatial models. Robots can then receive instructions to move through the vertices in the same way a human receives or gives directions to reach a target location. For example, when one person gives instructions to a second person about how to go from a room to the closest elevator, a sequence of commands will be given, to turn right or left depending on of the topology of the building and the direction of motion.

Definition 1: An OSTM is defined as $\mathcal{M} = (V, E, \mathcal{L}, \mathcal{D}, \mathcal{S})$ where:

- (V, E) are the vertices and edges of a directed graph;
- $\mathcal{L} : V \rightarrow \mathcal{L}$ is a function that associates a unique semantic label to each of the vertices; in this work we assume that $\mathcal{L}$ includes just two labels, i.e., *room* or *corridor*;
- $\mathcal{D} : E \rightarrow \{\text{E-W}, \text{N-S}\}$ associates a direction to each of the edges (east-west or north-south);
- $\mathcal{S} : E \rightarrow \{L, R\}$ assigns a label L (to the left of) or R (to the right of) to each edge; the meaning is that the oriented edge is to the left/right of the vertex it originates from.

The functions $\mathcal{S}$ and $\mathcal{D}$ are subject to the following consistency constraints:

- If $e = (v_i, v_j) \in E$ is an edge from v_i to v_j and $\mathcal{S}(e) = L$, then $e' = (v_j, v_i) \in E$ and $\mathcal{S}(e') = R$.

And, $e(v_i, v_j) \in E \wedge \mathcal{S}(e) = R \Rightarrow e' = (v_j, v_i) \in E \wedge \mathcal{S}(e') = R$.

- For each pair of adjacent vertices, $\mathcal{D}(v_i, v_j) = \mathcal{D}(v_j, v_i)$.

During map creation, each vertex can be associated with the direction from where it was discovered. Although this direction is only a reference of the robot's moving direction, we add this extra element to the map representation to increase the amount of information available about a location, apart for the label and the information about the edge's direction. To this end, we introduce

- $\mathcal{VD} : V \rightarrow \{\text{E-W}, \text{N-S}\}$ is a function that links a direction to each of the vertex from where they were discovered.

B. Inverse Warrington's Object Recognition Model (IWORM)

The model presented by Warrington [19] inspires our strategy for merging together two or more OTSMs. Warrington hypothesized two post-sensory categorical stages that work together for object recognition. This research showed evidence of how different patient groups presented a deficit in recognizing objects because of left-posterior and right-posterior cerebral lesions. These patients were suffering from different levels of *visual agnosia*, defined as [19], "the inability to recognize or identify common objects that cannot be accounted for by sensory impairment or more generalized cognitive deficits." Warrington's hypothesis of how the human brain processes the visual information proposes that although both sides of the brain do a visual analysis, the right side of the brain's job is to judge the matching as same or different stimuli, while the left side's function is to match objects to pictorial representation for an a posteriori word matching. Visual stimuli are interpreted by the brain to form a shape/contour of an object. This shape can match with at least one shape already stored in the memory that will eventually receive a semantic label that corresponds to a word that has a meaning or significance.

Starting from this work, we posit that in order to effectively evaluate and measure the similarities between two sub-maps and find their matching vertices, we require a two-sided function that can sequentially process their perceptual and semantic information. Inspired by how our brain finds the match of shapes/contours and assigns semantic labels to them, we propose to match the shapes and labels of different sub-maps to score their resemblance. Then, we will use the quality of the resemblance to then merge them together. Since OTSMs embed semantic information in our graphs, we propose to invert the information flow from a semantic to a perceptual categorization, instead of perceptual to semantic as the original Warrington's work proposes for humans. Specifically, we will take the semantic categorization as the comparison of labels and orientations, followed by the perceptual categorization, that will compare the shape of the graphs, analyze the number of neighbors and their respective connections with other vertices in the map.

IV. IWORM Inspired Map Merging of OTSMs

When considering spatial models featuring topological components, map merging can be referred to problems like graph structural matching, and sub graph isomorphism. However, for our problem these approaches cannot be applied directly due to the fact that OTSMs extend the basic graph structure embedding semantic information into the topological map. When merging topological maps, the objective is to obtain a *better* map of an environment by stitching together multiple *sub-maps*. We now describe how multiple OSTMs can be consolidated into a unique map.

Definition 2: A sub-map is represented by a graph $g_i = (V_i, E_i) \subseteq G = (V, E)$, whose vertices V and edges E can be compared with vertices and edges of other sub-maps using a scoring function called IWORM. This will serve to determine the likelihood that a vertex v_i in g_N corresponds to a vertex v_j in g_M (with $N \neq M$).

Without loss of generality, in the following we consider the case where just two sub-maps must be merged. When three or more maps must be combined, subsequent pairwise mergings can be used. Definition 2 establishes that two sub-graphs can be compared and the IWORM function (to be define later) can help to determine correspondences between common vertices. We say that two sub-maps overlap if they share one or more common vertices. These shared vertices are then used to establish a fully connected map. As for other map merging problems, if there is no overlap between the two sub-maps, then no merging can occur and this situation must be properly detected and handled. While an initial analysis of this case will be presented in the results section, it is still a research topic to find a robust way to deal with this problem.

As presented in section III-B, we aim to find a map matching algorithm following a method inspired by the same cognitive process that our human brain does. IWORM is a pattern classifier that uses two types of inputs to match two given sub-graphs. The first layer, called *Semantic Categorization*, compares two sub-graphs g_1 and g_2 in terms of the semantic label and the orientation of each of the vertices. This is a high level matching providing only a limited level of differentiation between vertices. However, as it will be presented later, there are different sources of errors that make it too brittle to rely only on these two features to classify and match vertices. To mitigate this limitation, a second layer is introduced. *Perceptual Categorization* compares the local structure of the sub-graphs. The structure we are looking to match will be the topological properties describing the connections between vertices. Specifically, we examine how many neighbors a vertex has, and how these neighbors, in turn, are connect to others. This structure can be described in terms of degrees of depth, where level one corresponds to the immediate neighbors of a certain vertex, level two corresponds to the neighbors of the immediate neighbors, and so on. Algorithm 1 describes the process to merge two sub-maps g_1, and g_2. The main process iterates over each of the input sub-maps. First, it tries to find correspondences

1: **Algorithm** $mergeGraphs(g_1, g_2)$
2: **for all** Vertices v_i from g_1 **do**
3: **for all** Vertices v_j from g_2 **do**
4: Get $[matchingPairs_{g_1}, \ scoresPairs_{g_1}]$ from $IWORM(v_i, v_j)$
5: **for all** Vertices v_j from g_2 **do**
6: **for all** Vertices v_i from g_1 **do**
7: Get $[matchingPairs_{g_2}, \ scoresPairs_{g_2}]$ from $IWORM(v_j, v_i)$
8: mergeMaps(g_1, g_2);

Algorithm 1: Merging OTSMs

1: **Algorithm** $IWORM(v_i, \ v_j)$
★ {*Semantic Categorization*}
2: $Score_L \leftarrow \mathcal{L}(v_i)$ is semantically similar to $\mathcal{L}(v_j)$
3: $Score_{VD} \leftarrow \mathcal{VD}(v_i)$ is equally oriented to $\mathcal{VD}(v_j)$.
★ {*Perceptual Categorization*}
4: $Score_{NE} \leftarrow \forall\, e_i \,\exists\, e_j$ Number of edges comparison.
5: $Score_{ED} \leftarrow \forall\, \mathcal{D}(e_i) = \mathcal{D}(e_j)$ Edge's direction comparison.
6: **for all** Vertices v_k^i **do**
7: **for all** Vertices v_l^j **do**
8: **if** max_Depth is reached **then**
9: Create matching pairs with the maximum scores.
10: **return** $matchingPairs$, $scoresEdges$
11: **else**
12: Get $scoresEdges$ from $IWORM(v_k^i, v_l^j)$

Algorithm 2: IWORM scores the resemblance of a pair of vertices $v_i, \ v_j$.

between vertices. To this end, it makes a complete pairwise comparison between all vertices[1] . Note that the first vertices of g_1 are matched against vertices of g_2 and then vertices of g_2 are matched against vertices of g_1. This is because the *IWORM* function is not symmetric, i.e., for $v_i \neq v_j$ in general $IWORM(v_i, v_j) \neq IWORM(v_j, v_i)$.

Because we assume that in the acquisition of each map there may be errors, the labeling, orientation and neighbors cannot be assumed to be error free. Consequently, we first create pairs of possible matching vertices and for each pair a score is assigned. To do this, the *IWORM* function performs the local analysis in the graphs to give a value for each correspondence found when comparing labels, orientation, number of neighbors for a certain degree of depth, that will be studied in the numerical validation section. If the semantic label is the same for both vertices, the score will be K, and if the label is different it will be $-K$, where K is a preassigned positive constant (Algorithm 2, Line 2). A similar binary score is assigned when comparing the orientation between the two vertices. This will be K when both vertices share the same orientation or 0 otherwise (Algorithm 2, Line 3). In this case a mismatch receives a 0 score, rather than $-K$ because as we established previously, the orientation of each vertex is relative to the robot's orientation and will only be used to add a bias component that will increase the score when two vertices were discovered the same way. For the number of edges outgoing from a vertex, the score is assigned with the following function $Score_{ED}(v_i)$ where *diffEdges* is the absolute value of the difference between the outdegree of the vertices. The effect of changing the K function will be the subject of future research:

$$\begin{cases} K & \text{if } diffEdges = 0 \\ \left(1 - \frac{diffEdges+1}{K'}\right) \cdot K & \text{if } diffEdges \leq 2 \\ -K & \text{otherwise} \end{cases}$$

Finally, for line 5 of algorithm 2, a score of K is assigned if the number of edges and orientations are the same in both vertices. From line 6, for each of the vertices v_i and v_j we score their neighbors recursively calling the same function *IWORM* on the neighbors k and l from vertices v_i and v_j, noted as v_k^i and v_l^j, respectively. The recursive calls to *IWORM* stop after a fixed number of recursive levels to ensure that the analysis remains local.

After scoring the vertices of each sub-graph, the next step is to validate if a label is found in two different locations, understanding the location as where the vertex is in the topological map and its neighbors.

Finally, we need to merge the two sub-graphs, in line 8 of Algorithm 1. For this purpose, we need to chose which pair of vertices have the same resemblance and in this case decide which information to accept. When we obtain the scores matching g_1 to g_2 (lines 2 and 4), and g_2 to g_1 (lines 5 to 7), we are adding each of the individual K values after comparing label, direction and edges of each pair of vertices, that will result in a final score that is compared with the highest possible score of K. If the final score corresponds to more than 80% of the highest K we assume that those two vertices are the same. By the contrary, if a pair of vertices is matched with less than 80% of the highest score, it means that the semantic and topological information between those two vertices has some discrepancies (it can be the labels, directions, or the edges). In the case where the semantic labels are different, we cannot be certain which one is the correct one, so we arbitrarily pick the label from g_1 and save the label from g_2 as an optional label. This optional label list will be used when merging new maps to the existing final map that, perhaps, can confirm which label is most likely the correct one. If the direction differs between vertices, again, we pick the direction from g_1 and save the direction from g_2 as optional. For the edges we combine the information from g_1 and g_2 together, where the outdegree of the vertices from v_1 defines how many possible vertices can be connected. We then complete this list with the available information about the visited vertices that connect with v_1 and v_2.

A. Sources of error

There are at least four different types of errors that can happen when building OSTMs. Because we are not using

[1]This step is typically not time-consuming because topological representations are compact and maps have a small number of vertices.

metric maps, there are no errors associated with rotation, alignment, or scale. However, we still have a possible translation problem: the location of the same vertex can be different due to an incorrect labeling.

1) *Error type 1:* A semantic labeling error, i.e., a vertex is assigned the wrong label when it is visited for the first time, or when it is revisited it is not recognized as the same vertex and assigned a different label.
2) *Error type 2:* An error with the compass will lead to an incorrect assignment of the direction of a vertex/edge.
3) *Error type 3:* A vertex in a sub-graph can be mistakenly associated with a wrong degree. For example, the vertex v_1 in g_1 is a four way intersection, but the same vertex in g_2 is detected a three-way intersection due to a wrong intersection detection.
4) *Error type 4:* A vertex v_1 in a sub-graph g_1 can be missing, but appear in a second sub-graph v_2. This happens when the vertex is recognized like a previous vertex (error type 1), that already exists in the sub-graph or because the robot passed by the location and never identified it. This is one of the most serious types of error. If the robot misses a vertex, this means that the neighbors will be connected incorrectly, their directions will not match, or we may have unconnected graphs.

V. Simulations

A. Setup

To evaluate the solution we proposed, we studied the problem in Gazebo, so that numerous tests with controlled error conditions could be performed. Consistent with the hypotheses that we operate in an environment with orthogonal walls, we start with the CAD models of one of the buildings in our university (see Figure 1). We defined five start locations in the building as shown in figure 1. Each one represents an initial position for a robot, and at each location the robot used the exploration algorithm described in our former work [17] to build a a partial map. To assess robustness, the formerly identified four different types of errors were introduced while building the partial maps. To evaluate the the IWORM algorithm we chose a very high error rate, where each of these errors had an independent 20% chance of occurring. Consequently, each vertex could be affected by more than one error at once. At each of the five locations we ran the exploration algorithm 20 times and obtained 20 sub-maps. Since the full map contains 40 vertices in total and we have 5 different regions, we chose maps with more than 8 vertices in order to have overlapping regions. For this reason, in each case the exploration algorithm was stopped when 11 vertices were added. Colored regions in figure 1 show the areas within which each robot wandered (red area for red start point, and so on). For each subgraph we exported seven different versions: one without any errors and four with the four type of errors, plus two with all the errors combined with and without error type 4. A good merging would mean that the topology of the resultant merged map closely resembles the ground truth. Similar to what we did in [4], to assess the quality of a merging, we developed a score function to compare the similarities of ground truth and the merged maps. The function assigns one point to a quality value in four cases:

1) 1 point for each vertex that exists in the ground truth map and also exists in the merged map and both have the topological label;
2) 1 point for each vertex that exists in the ground truth map and also exists in the merged map with the same name and direction;
3) 1 point for each vertex that exists in the ground truth map and also exists in the merged map with the same name and number of edges;
4) 1 point for each vertex that exists in the ground truth map and also exists in the merged map with the same name and the direction of all edges is the same;
5) 0 for every other case.

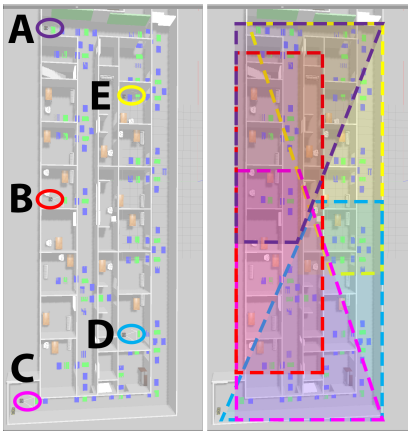

Fig. 1: Left: start locations. Right: overlapping regions.

B. Results

We first tested the merging algorithm with two non overlapping maps to show how much a map can be affected by the four types of errors when trying to merge them. In this case we took the maps produced by the robot starting in position A and tried to merge them with maps from position D (see figure 1 for the non overlapping footprints). Figure 3 compares the final map when merging non overlapping error-free maps against maps with errors 1,2,3, and 4. Blue and orange circles correspond to discovered corridors and room locations, yellow circles indicate open edges that lead to new undiscovered locations and green arrows correspond to the edges that connect a pair of vertices. After merging a map from region A and D, we show how the merging algorithm kept the correct structure of the sub-maps with little impact on the vertices. On the left of the figure we observe how the algorithm correctly merged both pieces of maps without misplacing, or incorrectly connecting the vertices, and on the right, we can see that, despite adding errors, the final map did not suffer drastic changes. Only 3 vertices were affected by the errors (blue circles with

red letters). Since the final map was qualitatively correct, we quantitatively analyzed the performance of the merged maps for each type of error. Figure 2 shows a blue line, labeled *Max*, representing the maximum score obtained with the error-free maps. As we can observe, the maps with only errors in the direction have a very close quality score to the error-free maps; results that validates the theory about the small impact of this type of error when defining an OTSM that does not affect the topology fundamentally. The maps that contained errors type 2, 3 directions, edges remain close to the maximum. However, we see how the label error gives the lowest score, together with the map of combined errors 1+2+3. This is due to how we score the maps based on the label. This also corresponds to the idea that semantics plays a main role when differentiating a vertex from others. It is also expected that the maps with the lowest quality corresponds to the ones with error type 4 and all the errors combined 1+2+3+4.

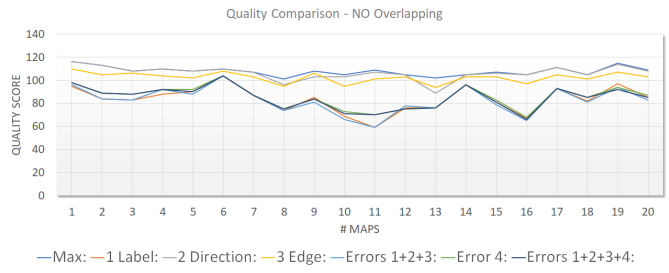

Fig. 2: Comparison of quality from 20 full maps with different errors when maps do not overlap A+D.

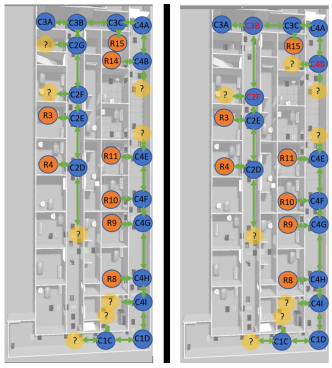

Fig. 3: Left: OTSM Error Free No Overlapping. Right: OTSM Errors 1+2+3+4 No Overlapping

Figure 4 shows again the seven versions of the maps, but this time when we combine all the regions. Similarly, as the previous case for no overlapping maps, we observe that for errors 2, and 3 the quality is close to the maximum. The map built in figure 5 shows a fully connected map. Once again, for error-free maps the final map is perfectly merged, showing how the IWORM function recognized correctly the overlapping vertices and the matching score adequately served to correctly stitch them together. When trying to merge the maps with all errors, we note that the algorithm can merge the graphs, although as shown in Figure 5, there exist some errors in the final map, when we compare it with the error-free map. Red arrows show an incorrect edge between two nodes. These issues come mainly from errors type 1 and 4; a missing vertex affects heavily the connection between vertices and this impacts the map's connectivity.

Finally, we did a quantitative analysis when conducting the IWORM from 1 degree up to 4 degrees of depth search. We found that there is no significant difference between degrees 1 and 2, and only 2.19% difference between 2 and 4 for only the maps with all errors. For all our tests we chose degree 3 to prove the intrinsic iterative capacity of our algorithm, even when a degree of 2 would have been enough. However, we believe this degree can play a more significant role when dealing with bigger environments where there are locations in different parts of the map with similar topologies and semantics that require more detailed differentiation.

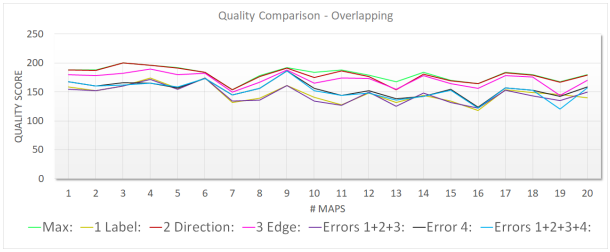

Fig. 4: Comparison of quality from 20 full maps with different errors when maps overlap A+B+C+D+E.

Fig. 5: Left: Final OTSM error-free. Right: Final OTSM with errors 1+2+3+4

VI. Conclusions and Future Work

In this paper we presented a new algorithm to merge OTSM that is inspired by the Warrington's object recognition model [19]. The method was tested using overlapping and no-overlapping pieces of a full map that contains four different type of errors and we showed that it is possible to merge maps with errors in the labels, directions, number of edges detected, and missing vertices. The proposed technique proved to be robust to multiple concurrent errors, even when error rates are much higher than what we observed in practice. For future work, we will explore how different exploration strategies, similar to the ones presented, affect the overall quality of the merged maps and how to assess how partial errors in map merging affect the robots' ability to use the combined map for autonomous navigation.

REFERENCES

[1] J. L. Blanco, J. González-Jiménez, and J. A. Fernández-Madrigal. A robust, multi-hypothesis approach to matching occupancy grid maps. *Robotica*, 31(5):687–701, 2013.

[2] T. M. Bonanni, B. Della Corte, and G. Grisetti. 3-D Map Merging on Pose Graphs. *IEEE Robotics and Automation Letters*, 2(2):1031–1038, 2017.

[3] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider. Coordinated Multi-Robot Exploration. *IEEE Transaction On Robotics*, 21(3), 2005.

[4] S. Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous Robots*, 25(3):305–316, 2008.

[5] S. Carpin and A. Censi. An experimental assessment of the hsm3d algorithm for sparse and colored data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3595–3600, 2009.

[6] J. Dichtl, L. Fabresse, G. Lozenguez, and N. Bouraqadi. PolyMap: A 2D Polygon-Based Map Format for Multi-robot Autonomous Indoor Localization and Mapping. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10984 LNAI, pages 120–131, 2018.

[7] B. Gallagher. Matching structure and semantics: A survey on graph-based pattern matching. *AAAI FS*, 6:45–53, 2006.

[8] W. H. Huang and K. R. Beevers. Topological map merging. *International Journal of Robotics Research*, 24(8):601–613, 2005.

[9] V. Karpov, A. Migalev, A. Moscowsky, M. Rovbo, and V. Vorobiev. Multi-robot exploration and mapping based on the subdefinite models. In A. Ronzhin, G. Rigoll, and R. Meshcheryakov, editors, *Interactive Collaborative Robotics*, pages 143–152. Springer International Publishing, 2016.

[10] H. Li and F. Nashashibi. A new method for occupancy grid maps merging: Application to multi-vehicle cooperative local mapping and moving object detection in outdoor environment. *12th International Conference on Control, Automation, Robotics and Vision*, 2012(December):632–637, 2012.

[11] Q. Liu, R. Li, H. Hu, and D. Gu. Building semantic maps for blind people to navigate at home. *Proceedings of the 8th Computer Science and Electronic Engineering Conference*, pages 12–17, 2017.

[12] J. Park, A. J. Sinclair, R. E. Sherrill, E. A. Doucette, and J. W. Curtis. Map merging of rotated, corrupted, and different scale maps using rectangular features. *Proceedings of the IEEE/ION Position, Location and Navigation Symposium*, pages 535–543, 2016.

[13] M. J. Paulik, J. Overholt, M. Krishnan, Y. Alnounou, and G. Hudas. Occupancy Grid Map Merging using Feature Maps. In *IASTED Technology Conferences*, pages 10.2316/P.2010.706–074., 2016.

[14] M. Rohani, D. Gingras, and D. Gruyer. A novel approach for improved vehicular positioning using cooperative map matching and dynamic base station DGPS concept. *IEEE Transactions on Intelligent Transportation Systems*, 17(1):230–239, 2016.

[15] S. Saeedi, M. Trentini, M. Seto, and H. Li. Multiple-Robot Simultaneous Localization and Mapping: A Review. *Journal of Field Robotics*, 33(1):3–46, 2016.

[16] S. G. Shahbandi, M. Magnusson, and K. Iagnemma. Nonlinear Optimization of Multimodal Two-Dimensional Map Alignment With Application to Prior Knowledge Transfer. *IEEE Robotics and Automation Letters*, 3(3):2040–2047, 2018.

[17] J. L. Susa Rincon and S. Carpin. Time Constrained Exploration Using TopoSemantic Spatial Models: a reproducible approach. *IEEE Robotics and Automation Magazine*, 2019 (accepted for publication).

[18] E. Tsardoulias, A. Thallas, and L. Petrou. Metric map merging using RFID tags & topological information. *CoRR*, abs/1711.06591, 2017.

[19] E. K. Warrington. The selective impairment of semantic memory. *The Quarterly journal of experimental psychology*, 27(4):635–657, 1975.