A New Efficient Topological Approach to Map Merging Based on a Probabilistic Generalized Voronoi Diagram

Sajad Saeedi[†], Liam Paull[‡], Michael Trentini[⋆], and Howard Li[†]

***Abstract*—Simultaneous localization and mapping (SLAM) is required for mobile robots to be able to explore a prior unknown space without a global positioning reference. Multiple robots can achieve exploration tasks more quickly, but with added complexity. A useful representation of the map for SLAM purposes is as an occupancy grid map. In the most general case of multiple robot SLAM, occupancy grid maps from multiple agents must be merged in real-time without any prior knowledge of their relative transformation. In addition, the probabilistic information of the maps must be accounted for and fused accordingly. In this paper, the generalized Voronoi diagram (GVD) is extended to encapsulate the probabilistic information encoded in the occupancy grid map. The new construct called the probabilistic GVD (PGVD) operates directly on occupancy grid maps and is used to determine the relative transformation between maps and fuse them. This approach has three major benefits over past methods: 1) It is effective at finding relative transformations quickly and reliably, 2) the uncertainty associated with transformations used to fuse the maps is accounted for, and 3) the parts of the maps that are more certain are preferentially used in the merging process due to the probabilistic nature of the PGVD.**

***Index Terms*—Simultaneous Localization and Mapping (SLAM), Map Fusion, Voronoi Diagram, Multiple Robots.**

I. Introduction

The ability of an autonomous agent to sense its environment and situate itself within this environment is a cornerstone of mobile robotics. Simultaneous localization and mapping (SLAM) is the process of exploring an unknown environment and concurrently generating a consistent map of the world by fusing available sensor data. As such, the pose of the robot can be estimated and the environment map can be built [2].

Extensive literature exists on SLAM for a single robot, for a review see [3]. However, exploring unknown environments with multiple mobile agents has received comparatively less attention and can have significant advantages, including:

- Exploration and mapping can be done more rapidly.
- A distributed system is more robust to failures [4].
- Results are more accurate due to redundancy of data.

With the stated benefits of multiple robot SLAM comes significant challenges in implementation. The two main problems to be overcome are map merging without a global position reference, and map fusion incorporating map uncertainty. This paper addresses both issues using only information within the maps in a robust and scalable way.

Abstract geometrical perception is a foundation for high-level reasoning and knowledge sharing. When multiple robots are supposed to explore and map an unknown environment cooperatively, there is a need to provide a logical infrastructure so that robots can share their spatial perception and decide about how to use the shared knowledge. If the salient information from the maps is extracted and shared among robots, the speed and accuracy of the mutual perception will improve and communication channels will not be burdened with large amounts of unprocessed data.

A topological map is an abstract world representation in the form of connected paths and intersections. Humans and insects use topological maps to navigate, argue about paths and positions and avoid obstacles [5]. As an example, pigeons have been shown to use highways and their intersections as a topological map to fly over long distances [6].

The general approach taken here is to achieve SLAM through graph matching where a graph is some reduced form topological representation of the higher level map structure [7]. The generalized Voronoi diagram (GVD), described in next sections, is an appropriate graph structure.

A. Multiple-robot SLAM

Past approaches to collaborative SLAM can be generally categorized based on whether they share raw sensor data [8] or processed maps [4]. Sharing raw sensor data results in more flexibility but requires high bandwidth and reliable communication between robots as well as more processing power. In contrast, sharing maps uses less bandwidth and reduces the need to process raw data; however, the performance is dependent on map quality. The latter method is referred to as map merging or map fusion and is the approach taken here.

Multiple robot SLAM can also be categorized based on the method used to process measurement data. In feature-based SLAM [9], which is usually performed by cameras, unique objects called features or landmarks are extracted from measurements and used for localization. The spatial distribution of the features represents a model of the world. Feature-based multiple robot SLAM has been implemented using an information filter [10], an extended Kalman filter [11], and a particle filter [12].

An alternative paradigm is view-based SLAM [13], which uses entire laser scans. Scans are matched using scan matching algorithms. Our approach uses view-based SLAM.

Thrun proposes a probabilistic multiple robot view-based SLAM in [14]. This method is robust; however, the initial poses of the robots are assumed to be approximately known prior to the start of the mission.

A view-based multi-robot SLAM is proposed by Howard using a particle filter [8] where it is assumed that robots will meet each other in the environment to determine their relative poses. This method is moderately fast but demands high computational power and memory since it is based on particle filtering.

Manuscript received on February 10, 2013.

Preliminary results of this work have been presented in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2012 [1].

[†]COBRA Group, University of New Brunswick, Fredericton, Canada, {sajad.saeedi.g, howard}@unb.ca

[‡]CSAIL Lab, Massachusetts Institute of Technology, Cambridge, USA lpaull@csail.mit.edu

[⋆]Defence Research and Development Canada-Suffield, Alberta, Canada, Mike.Trentini@drdc-rddc.gc.ca

An effective and fast approach to multiple robot SLAM rests on the concept of map merging or map fusion [15]. In [4] a solution is presented based on occupancy map merging. This method uses map-distance as a similarity index and tries to find similar patterns in two maps based on a random walk algorithm. The drawback of this method is that it fails especially when there are few similar patterns in both maps. This solution is time consuming and therefore problematic in large scale maps which are common in indoor environments. A similar method is proposed in [16] with simulated annealing and hill climbing used to merge maps. This method becomes ineffective in maps with less overlaps.

The contribution of this research is a novel map fusion algorithm that exploits the properties of the GVD to achieve fast and accurate map fusion for large maps. In addition, the uncertainty in the maps is used to build a probabilistic GVD (PGVD) that encapsulates not only the topological structure of the map, but also the confidences associated with different areas of the map. Once the PGVDs are built, edges are matched using a 2D cross correlation that will preferentially match the areas of the maps that have higher confidences. The resulting transformation can align maps to generate a global map. However, there is an uncertainty associated with this calculated transformation which should also be propagated to the maps. The novel linearized uncertainty propagation (LUP) approach proposed in this research accounts for the uncertainty of the transformation. In LUP, the transformation for each cell of the map is linearized, then the Gaussian uncertainty of the transformation is propagated to the transformed cell. This process is performed on all transformed cells, therefore the resulting map carries the uncertainty of the transformation. Final map fusion is then achieved with an entropy filter. The result is a fast, reliable, and robust method of fusing maps for multiple robot SLAM.

It should be mentioned that the assumption in this work is that the individual maps developed by each robot are accurate and consistent.

The preliminary results of this work were presented in a conference paper [1]. Here we expand on [1] with a more rigorous formulation of LUP in Sec. III, followed by a step-by-step explanation of the map merging process with an example in Sec. IV, and additional results in a highly cluttered environment in Sec. V. To summarize, The proposed novel method for map fusion has a few key advantages:

- It accounts for uncertainties in the occupancy grid maps using the PGVD,
- It considers the uncertainty of the calculated transformation by linearization,
- It is fast and robust compared to other methods,
- It is able to preferentially match areas of the maps that are more certain.

The rest of the paper is organized as follows: Section II presents some background on the GVD and SLAM, Section III discusses transforming grid maps, Section IV introduces the proposed method for topological multiple robot SLAM, Section V presents some experimental results from both online datasets and experiments conducted in the lab, and Section VI makes some general conclusions and discusses future work.

II. Background

The GVD is a type of roadmap [17] which is the locus of points equidistant to at least the two closest obstacles. The GVD has two important properties which will be exploited:

1) The GVD is connected because the set of free space is connected and connectivity is maintained under a deformation retraction [17].
2) The GVD of a map is unique and is invariant to transformations because it is a retraction [17].

The GVD can be interpreted as a topological representation of the map structure that contains the key information intrinsic to the map but in a much more compact form.

There are different methods to generate GVD for a map. Mathematical morphological operations [18] are a fast and reliable method to build a GVD.

In Blum's method introduced in [19], the GVD is developed using a maximal inscribed circle method which is inefficient for large maps.

A dynamic version of the GVD is proposed by Boris et al. [20] which improves performance near non-convex obstacles. Though this method has good results, it is time consuming.

Beeson et al. [21] propose an extended Voronoi graph algorithm to improve the efficiency of building the GVD when the robot is limited with its sensory horizon. This method performs well in noisy environments by eliminating spurious junctions.

A. *Previous Topological Approaches to SLAM*

Voronoi graphs and graph matching in its different forms have many applications in SLAM. There are many topological solutions for single robot SLAM such as the works by Choset et al. [22], [23], annotated generalize Voronoi graph (AGVG) [24], the work by Beeson et al. [25], Bayesian inference [26] and the semantic approach with place labeling by Friedman et al. [27].

Choset et al. [22] propose a topological SLAM algorithm for a single robot based on the GVD. In [24], an AGVG is used for single robot SLAM. The proposed method is based on a matching scheme for solving the data association problem. The method identifies the corresponding parts of the map in two tree-formed Voronoi graphs. One form represents a local observation and the other form represents the internal map of the robot.

In [28], a solution to detect and recognize topological features is proposed using Delaunay triangulations.

In [27], a semantic approach is used by a robot to generate a topological map which can identify different places. Identified places can be used for different autonomy applications.

Multi-robot SLAM based on topological map merging using both structural and geometrical characteristics of the Voronoi graph is proposed in [29]. The assumption in this work is that a robot will be able to recognize areas of the map that correspond to vertices. In this case the topological map is built on the occupied space as opposed to the free space. The method in [29] is claimed to be fast. However, a limitation is that the maps are not updated.

In this work, a skeletonization approach is extended to be probabilistic and used for map matching of multiple robots. A common problem encountered when using skeletonization for SLAM is that finding the skeleton has no closed form solution and heuristic methods for generating it tend to be slow. In our approach, morphological operations are used to generate the GVD as is explained in Sec. IV-B. The proposed approach is fast enough to be used in real-time and guarantees connectedness of the skeleton [18]. In addition, the probabilistic nature of the proposed PGVD allows areas of the maps that are known with higher certainty to be preferentially matched.

III. Transforming Occupancy Grid Maps with an Uncertain Transformation

Generally, any type of uncertainty in this context is identified as having five main causes: environment, sensors, robots, models and computations. Thrun et al. state that “uncertainty arises if the robot lacks critical information for carrying out its task” [3].

Map fusion involves dealing with two types of uncertainty.

- *Uncertainty due to pose and sensor measurements*. This kind of uncertainty is represented within the occupancy grid map. Specifically, it originates from the uncertain pose of each robot and is embedded in the map of the robot. This can be considered as a form of uncertainty at the individual robot level. Throughout this research, the terms uncertainty or map uncertainty refer to this type unless otherwise stated.
- *Uncertainty due to transformation*: If we can estimate the uncertainty in the transformation that relates two maps, then we can account for this uncertainty in the fusion process. In this work, the terms transformation uncertainty, rotation uncertainty and translation uncertainty are used to refer to this type of uncertainty.

In this section, the second type of the uncertainty, transformation uncertainty, is studied. The result of this section is used in subsequent sections. To the authors’ knowledge, there is no similar research in the literature.

The uncertainty associated with the relative transformation matrix is represented as a covariance matrix. The covariance error is propagated through the probabilistic transformation function using the LUP formulation, a novel map merging method introduced in [1] and expanded upon here. This section addresses the propagation of the uncertainty in transformation given the transformation matrix and its covariance as a map is being transformed so that it may be merged with another.

Assume that $q = \begin{bmatrix} q_i & q_j \end{bmatrix}^T$ is a multivariate random variable denoting the position of the cell (i, j) from map m. The probability density function (PDF) of q is shown by $p(q)$ and it is assumed to be a delta function. The mean value of q is shown by $\mu_q = \begin{bmatrix} \mu_{q_i} & \mu_{q_j} \end{bmatrix}^T$. The point q is transformed to another point, r, with the PDF of $p(r)$. The transformation is performed by $f(\cdot)$ defined as:

$$r = f(\psi, x, y, q) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} q + \begin{bmatrix} x \\ y \end{bmatrix}, \qquad (1)$$

where ψ is the rotation angle and x, y are translation elements.

In general, the nonlinearity and uncertainty of the transformation will cause a non-Gaussian distribution for r. However, it is possible to approximate the non-Gaussian distribution with a Gaussian distribution. By linearizing the transformation using the first order Taylor expansion, a Gaussian distribution is assigned to the transformed point:

$$p(r) = |2\pi\Sigma_r|^{-\frac{1}{2}} \exp\{\tfrac{-(r-\mu_r)^T \Sigma_r (r-\mu_r)}{2}\} \sim \mathcal{N}(r; \mu_r, \Sigma_r), \qquad (2)$$

where the mean of the normal distribution, μ_r, is calculated by (1). Assume that the covariance of the transformation matrix has the form:

$$\Sigma_{tr} = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 & \sigma_{x\psi}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 & \sigma_{y\psi}^2 \\ \sigma_{x\psi}^2 & \sigma_{y\psi}^2 & \sigma_{\psi\psi}^2 \end{bmatrix}. \qquad (3)$$

Through linearization, the covariance of the transformed point, Σ_r , is calculated by

$$\Sigma_r = F_{xy\psi} \Sigma_{tr} F_{xy\psi}^T + F_q \Sigma_q F_q^T \qquad (4)$$

where $F_{xy\psi}$ and F_q are the Jacobians of $f(\cdot)$ defined as

$$F_{xy\psi} = \frac{\partial f(\psi, x, y, q)}{\partial(\psi, x, y)}, F_q = \frac{\partial f(\psi, x, y, q)}{\partial(q)}, \qquad (5)$$

and Σ_q is the covariance of the point q which is zero. Note that the covariance Σ_r is a function of q, the location of the cell. Fig. 1 depicts the problem with a hypothetical nonlinear function representing the transformation. The lower right shows the point q which should be transformed by the given transformation function. The position of this point is deterministic, therefore its distribution is a delta function. However, the transformation function is not certain, shown by setting boundaries around the nominal transformation function (upper right). A Gaussian distribution is assigned to the transformed point (upper left).

Equation (4) gives the shape of the Gaussian distribution for every transformed point. Fig. 2 shows this process. The point q from map m is transformed to point μ_r. By applying the Gaussian defined in (4), depicted by $\mathcal{N}(r; \mu_r, \Sigma_r)$, the point μ_r takes a Gaussian form.

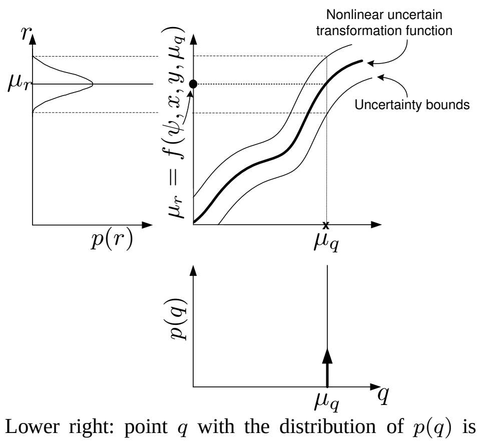
Nonlinear uncertain transformation function
Uncertainty bounds
r
μ_r
$\mu_r = f(\psi, x, y, \mu_q)$
$p(r)$
μ_q
$p(q)$
μ_q
q

Fig. 1. Lower right: point q with the distribution of $p(q)$ is transformed by an uncertain transformation function. Upper right: the uncertainty of the transformation function is shown by setting boundaries around the nominal function. Upper left: after linearization, the transformed point, r will have a Gaussian distribution, $p(r)$.

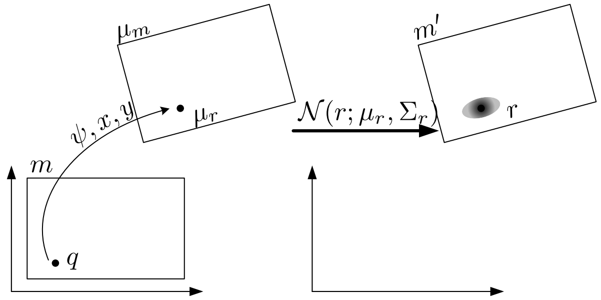

Fig. 2. Linearized transformation uncertainty propagation. Point q is transformed to point μ_r according to the rotation ψ and the translation x, y. Then a Gaussian kernel, $\mathcal{N}(r;\mu_r,\Sigma_r)$ is convolved with the transformed point. Σ_r is calculated through linearization.

The next step is to extend this formulation to all points of the map. Algorithm 1 explains this process. For simplicity, we use $T_{x,y,\psi}(q)$ to denote that the point q is rotated according to ψ and then translated according to (x, y). The same concept applies for transformation of a map, m, shown by $T_{x,y,\psi}(m)$.

First, the map m is transformed according to the given transformation, shown by $T_{x,y,\psi}(\cdot)$ (line 1). This is shown in Fig. 2, where the resulting map is marked by μ_m. In line 2, the final map which includes the uncertainty of the transformation is initialized by μ_m. In line 4, for every point of the transformed map, the covariance is calculated using (4). In line 5, the Gaussian kernel based on the covariance is calculated. Note that the Gaussian kernel is center originated which means the center point of the kernel is $h(0, 0)$. The mean of the kernel is the transformed point, while its 2×2 covariance matrix is calculated by (4). It is important to note that for each point of the transformed map, the kernel takes different values. Then in line 6, every transformed point is convolved with the Gaussian kernel. The operator $\circledast$ denotes the convolution of a Gaussian kernel with the map [18] and defined as:

$$\begin{aligned} m'(k,l) &= \mu_m(k,l) \circledast h(k,l) \\ &= \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} \mu_m(i,j) h(k-i, l-j). \end{aligned} \quad (6)$$

For simplicity of implementation, it is assumed that the size of the kernel is the same as μ_m.

Now m' can be fused with its pair map using the entropy filter method detailed in Sec. IV-E.

Algorithm 1 Linearized Uncertainty Propagation.

```
Require: Occupancy grid map: $m$,
         Transformation: $\psi, x, y$,
         Uncertainty of the transformation: $\Sigma_{tr}$.
Ensure: Transformed occupancy grid map: $m'$.
1: $\mu_m = T_{x,y,\psi}(m)$
2: $m' = \mu_m$
3: for all $\mu_r = [\mu_{r_i}, \mu_{r_j}]^T \in \mu_m$ do
4:     Calculate $\Sigma_r$ using (4)
5:     $h \leftarrow \mathcal{N}(\mu_r, \Sigma_r)$
6:     $m'(\mu_{r_i}, \mu_{r_j}) \leftarrow \mu_m(\mu_{r_i}, \mu_{r_j}) \circledast h(\mu_{r_i}, \mu_{r_j})$
7: end for
```

IV. Probabilistic Map Merging with the GVD

In this section we present the details of the map fusion process with the probabilistic GVD that was introduced in [1].

Consider the case of two mobile robots, R_1 and R_2, equipped with laser rangers exploring an environment and building occupancy grid maps (OGM) [3]. In an OGM representation, each cell in map $m_k(i, j)$, $k = \{1, 2\}$, is a binary random variable (RV) where $p(m_k(i, j) = 1) = p(m_k(i, j))$ is the probability that the cell at location (i, j) is occupied in the map of robot k. It is convenient to represent the OGM using the log odds representation of occupancy [3]:

$$l_k(i,j) = \log \frac{p(m_k(i,j))}{1-p(m_k(i,j))} \quad (7)$$

Without loss of generality, assume that R_2 transmits its local map, map_2 to R_1 through a wireless channel. R_1 is now responsible for incorporating the transmitted map into its own local map, map_1. There are three main challenges that need to be overcome:

1) The relative transformation from map_1 to map_2 needs to be found.
2) The uncertainty of the transformation should be accounted for.
3) The OGM probabilities from map_2 need to be incorporated with the OGM probabilities of map_1.

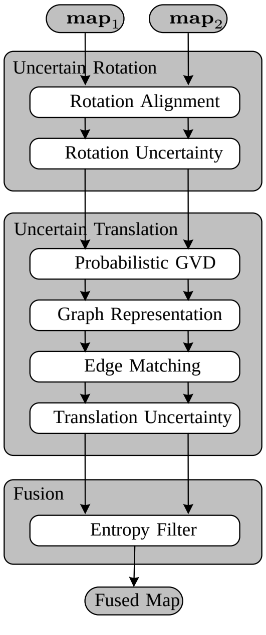

Fig. 3. (**left**) The proposed map fusion algorithm. Two input maps, map_1 and map_2, are fused by finding their relative transformation matrix. No prior information is available regarding the relative position of two respective robots. (**right**) Experimental robots, CoroBot, each equipped with a laser ranger and wheel encoders.

An overview of the elements of the algorithm is shown in Fig. 3. The subsequent sections will describe each of the block components in detail.

A simulated example accompanies each step of the algorithm to aid with explanation. Fig. 4-a shows the simulated environment, where three poles are located inside a rectangular room. The two robots map the room starting near the big pole but moving in opposite directions. Fig. 4-b and Fig. 4-c show the two local maps after some time has passed. Without loss of generality, it is assumed that the second map (Fig. 4-c) is fused into the first map (Fig. 4-b). Free, occupied, and unknown cells are shown by different colors, using the OGM standard. The darker the grid cell, the higher the probability of occupancy.

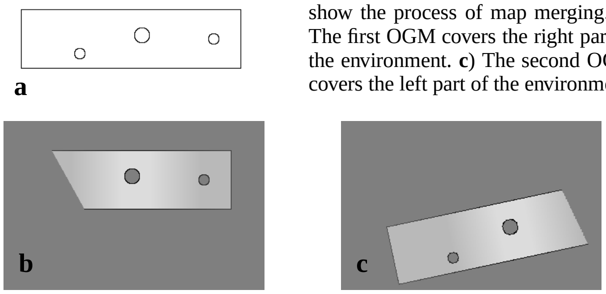

Fig. 4. **a**) A simulated environment to show the process of map merging. **b**) The first OGM covers the right part of the environment. **c**) The second OGM covers the left part of the environment.

A. Uncertain Rotation Alignment

In structured environments like urban or indoor settings, the relative rotation between maps can be found easily using the Radon transform as shown by the authors in previous work [30]. The Radon transform is the projection of the image intensity along a radial line oriented at a specific angle. Peak points in the Radon transform will correspond to straight line segments in the image. As a result, it is possible to resolve the relative rotation, ψ, between two images by looking for peaks in the Radon images of both maps. However, due to environment similarity, four rotation hypotheses are considered and only one is accepted by a similarity index [30]. At the output of this block there are the two aligned maps, m_1 and m_2 with the same size of $M \times N$, given by:

$$m_1 = map_1, \quad m_2 = T_{0,0,\psi}(map_2). \tag{8}$$

The uncertainty in the relative rotation must be accounted for in the subsequent calculation of the relative translation. This is performed using Algorithm 1. This means that after finding the rotation, its uncertainty on the map is propagated using Algorithm 1, assuming zero translation and $\sigma^2_{xx} = \sigma^2_{yy} = \sigma^2_{xy} = 0$. Now the process of finding the relative translation can be done knowing that uncertainty of the rotation is already incorporated into the rotated map.

Fig. 5-b shows the aligned map after applying the uncertain rotation alignment. More details about the rotation alignment process can be found in the authors' previous work [30].

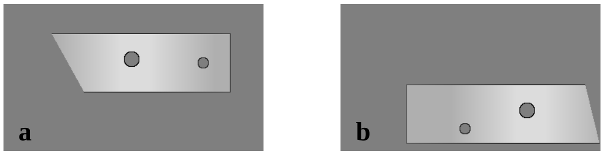

Fig. 5. **a**) The first OGM. **b**) The second OGM after being rotated. The relative rotation is calculated by comparing the Radon images of the maps.

B. Building the Probabilistic GVD

Finding the relative translation between maps is much more challenging than finding the relative rotation. The search for overlapping parts of maps can be slow. As a result, we use a novel probabilistic topological representation of the map as described in this section. The proposed solution is based on the idea that the search for overlaps in the topological space, which represents salient information, is easier, and consequently faster, than in the metric space. The PGVD is found for each of the two maps, m_1 and m_2. This process is completed in two steps:

1) Find the GVD efficiently using mathematical morphological operations on the binary OGM.
2) Compute the associated probabilities of each cell in the GVD based on the actual probabilities in the OGM.

1) Finding the GVD using Mathematical Morphology: Mathematical morphology defines the application of set operations on binary images using convolution between the image and defined masks. It has been used extensively in computer vision and image processing. A set of basic operators are defined in mathematical morphology like erosion, dilation, opening, closing, skeleton and hit-or-miss transform with different properties. The most important property is that they are translation invariant.

The GVD of the binary map is generated using eight D-type hit-and-miss transform masks [18]. Each mask is designed for a particular situation to guarantee the connectedness (using the connectivity-8 model). The GVD is represented as a matrix, $S = [s_{i,j}]$, with the same size as the map, $M \times N$, defined as:

$$[s_{i,j}]_{i=1..M,j=1..N} = \begin{cases} 1 & \text{if } m(i,j) \in \text{ GVD} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Fig. 6-a and Fig. 6-b show the GVDs of the two aligned maps. The probabilistic GVDs will be based on these GVDs.

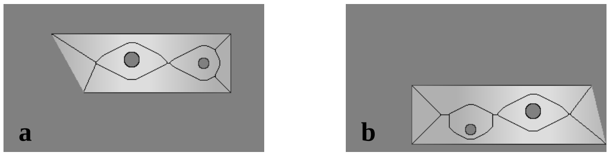

Fig. 6. **a**) GVD of the first map. **b**) GVD of the second map.

2) Finding the Probabilistic GVD: The skeleton developed by the proposed probabilistic GVD approach does reflect the uncertain nature of the OGM. While the GVD structure is generated from a binary version of the OGM, the probabilistic information in the OGM should be reflected in the GVD structure. A new structure, the PGVD is built that combines the structure of the GVD with the probabilistic nature of the OGM.

Definition 1: Contact Points. The *Contact Points* of a cell in the GVD are all the occupied cells in m that have the same distance from the cell as the closest occupied cell.

The probabilities of the contact points in the OGM are used to build the PGVD.

Definition 2: Probabilistic GVD. Each cell in the GVD is represented by a binary RV representing the probability that it has two or more occupied contact points based on their probabilities of occupancy in the OGMs.

Consider that a GVD, S, contains s cells, $G_i, i = 1..s$. Each cell in the GVD has an associated set of n_i contact points in the binary map m.

The PGVD, S^p has the same structure as the GVD, except that each of the s cells is represented as a binary RV, $G_i^p, i = 1..s$ where $p(G_i^p = 1) = p(G_i^p)$ is the probability that cell G_i^p has been correctly placed in the GVD. Each cell in the PGVD has an associated set of contact points $C_i = \{c_1, c_2, ..., c_{n_i}\}$, $i = 1..s$, where these contact points are at the same locations as the contact points of G_i except they are in the OGM so they have associated probabilities of occupancy. Each contact point is a binary RV where $p(c_j = 1) = p(c_j)\ \forall j = 1..n_i$ is the probability that the contact point is occupied taken directly from the OGM.

A cell belongs in the GVD if at least 2 of the contact points are occupied. For each cell in the PGVD G_i^p, we must determine the probability that it has at least 2 occupied contact points based on their probabilities of occupancy.

To do so, another RV $N_{G_i^p}$ is defined which represents the probability distribution of the number of occupied contact points of G_i^p. $p(N_{G_i^p} = k)$ is the probability that cell G_i^p contains k occupied contact points.

The $p(N_{G_i^p} = k)$ is defined as a function of n_i, the number of contact points in the OGM, and k, the number of contact points that are actually occupied:

$$P(N_{G_i^p} = k) = f(n_i, k). \tag{10}$$

The function $f(n_i, k)$ can be defined recursively as:

$$f(n_i, k) = p(c_{n_i})f(n_i - 1, k - 1) + (1 - p(c_{n_i}))f(n_i - 1, k). \tag{11}$$

Intuitively, this equation says that the probability of having k out of n_i contact points occupied is equal to the probability that contact point c_{n_i} is occupied and that $k-1$ of the remaining $n_i - 1$ contact points are occupied, plus the probability that c_{n_i} is not occupied and k of the remaining $n_i - 1$ contact points are occupied.

The base cases for the recursion are cases where all of the cells must be occupied or none of the cells must be occupied:

$$f(n_i, n_i) = \prod_{j=1}^{n_i} p(c_j), \quad f(n_i, 0) = \prod_{j=1}^{n_i} (1 - p(c_j)) \tag{12}$$

By induction it can be proven that $f(n_i, k)$ is a valid probability density function that produces all possible combinations of contact points and sums to 1 [1]. $p(G_i^p)$ is now given by:

$$\begin{aligned} p(G_i^p) &= p(N_{G_i^p} \geq 2) = \sum_{k=2}^{n_1} p(CN_i = k) \\ &= 1 - f(n_i, 0) - f(n_i, 1) \end{aligned} \tag{13}$$

The PGVD is now defined as:

$$S^p = \bigcup_{i=1..s} G_i^p \tag{14}$$

The output from this block will be a PGVD S_k^p for each input map m_k with $k = \{1, 2\}$.

Fig. 7-a and Fig. 7-b show the PGVD of the maps. In contrast with Fig. 6 where the GVD cells are deterministic, these cells are now probabilistic. The probabilities of the cells of the PGVD are represented by their grayscale intensity, darker cells having a higher probability of being true GVD cells.

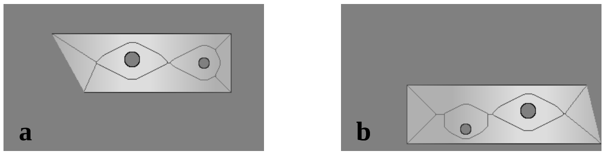

Fig. 7. **a**) PGVD of the first map. **b**) PGVD of the second map.

C. Graph Representation

The PGVD is then processed to be represented as a graph with edges and vertices. A vertex can be identified as any cell in the PGVD with more than two adjacent occupied cells. The set of all vertices is defined as V.

To identify edges in the graph, a dilation mask is applied to each vertex in V. The result of the dilation operation is a new map D that contains the dilated vertices. The edge matrix E is found as $S^p - D$. This operation is performed on each of the two GVDs to produce two edge maps: $E_k, k = \{1, 2\}$.

The probabilistic edge matrices (PEM), E_1^p and E_2^p, have the same structure as the edge matrices, E_1 and E_2, but the cells are the probabilistic ones extracted from the PGVD. These PEMs are now used to find the translational transformation between the two maps m_1 and m_2. Edges with short lengths are removed to avoid processing short edges which have a higher chance of producing false matches.

Fig. 8-a and Fig. 8-b show the PEMs for each map where short edges have been removed. The edges of each map are marked with numbers. The first map has seven edges and the second map has eight edges.

Fig. 9-a and Fig. 9-b show two enlarged edges of the first map, number 3 and number 7. The grayscale intensity of each cell in these edges represents the probability of that cell in the PGVD (The probability that it is a valid GVD cell). Edge number 3 is located in the area of the original map which has high certainty, so the probability of the cells of this edge being in the GVD is high. However, edge number 7 is located at the end corner of the map which is less certain. Therefore, the probability of the cells in this edge is lower.

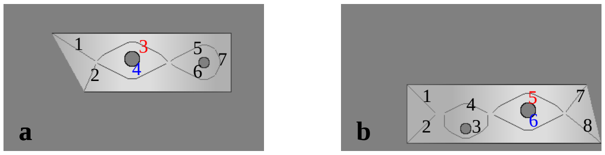

Fig. 8. **a**) PEM of the first map. **b**) PEM of the second map.

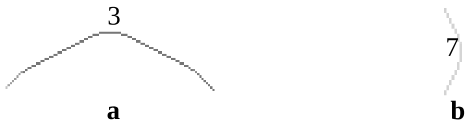

Fig. 9. Two edges from Fig. 8-a **a**) edge number 3. **b**) edge number 7.

D. Edge Matching for Translation Alignment

The edges of each probabilistic edge matrix are matched using a 2D cross correlation. To speed up the cross correlation, each edge from each edge matrix is represented as a sub-matrix of E where its size is such that it is as small as possible while still containing the entire edge. For a given edge matrix, E_k^p

with N_k edges, each edge, $e_k^{i_k}$, $i_k = 1..N_k$ is given by ($m_i \times n_i$ is the size of the sub-matrix containing $e_k^{i_k}$):

$$e_k^{i_k} = E^p_{k(pr_i:pr_i+m_i, pc_i:pc_i+n_i)}. \tag{15}$$

pr_i and pc_i point to the start of the sub-matrix. To do matching, each edge of E_1^p is cross correlated with each edge of E_2^p.

The 2D cross correlation of two matrices $e_1^{i_1}$ and $e_2^{i_2}$ with the respective sizes $(m_1 \times n_1)$ and $(m_2 \times n_2)$ is given as

$$\mathcal{E}^{(i_1,i_2)} = e_1^{i_1} \star e_2^{i_2} \tag{16}$$

where $\mathcal{E}^{(i_1,i_2)}$, the cross correlation matrix of edge $e_1^{i_1}$ and edge $e_2^{i_2}$, has size $(m_1+m_2-1)\times(n_1+n_2-1)$. The operator $\star$ denotes the 2D cross correlation operation.

The maximum value in the $\mathcal{E}^{(i_1,i_2)}$ matrix quantifies the best match between $e_1^{i_1}$ and $e_2^{i_2}$ based on the all possible combinations of translations between the two edge matrices. This value is computed using (16) for every combination of edges: $i_1 = 0..N_1$ and $i_2 = 0..N_2$.

Then we can define the similarity matrix, Γ_1^2, between E_1^p and E_2^p as:

$$\Gamma_1^2 = [\gamma_{i_1 i_2}]_{i_1=1..N_1; i_2=1..N_2}, \tag{17}$$

$$\gamma_{i_1 i_2} = \max(\mathcal{E}^{(i_1,i_2)}). \tag{18}$$

An overview of the formation of the similarity matrix, Γ, is shown in Fig. 10. First, all probabilistic edges from the PEM are cross matched based on (16). For instance, the two edges shown in blue are matched to generate the correlation matrix in blue, $\mathcal{E}^{(4,5)}$. The maximum value of $\mathcal{E}^{(4,5)}$ is stored at the corresponding location in Γ based on (17) and (18).

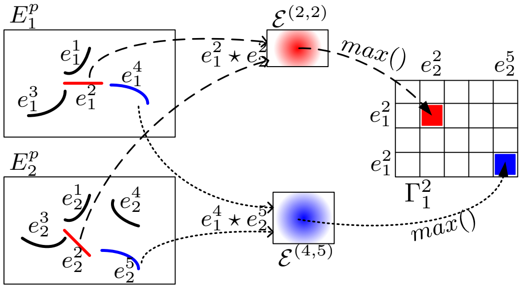

Fig. 10. Formation of the Γ matrix from the PEMs. All edges from the PEMs are cross matched. The peak of each match is stored at the corresponding location in Γ.

The best candidate for a match corresponds to the maximum value in the similarity matrix Γ_1^2:

$$[i_1^*, i_2^*] = \underset{i_1,i_2}{\operatorname{argmax}}(\Gamma_1^2[i_1, i_2]), \tag{19}$$

where i_1^* and i_2^* are the indices of the most likely similar edges from the two maps.

The relative translation is calculated by resolving the translation vector which matched these two edges using the following relation [30]:

$$T = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \mu_x(e_1^{i_1^*}) - \mu_x(e_2^{i_2^*}) \\ \mu_y(e_1^{i_1^*}) - \mu_y(e_2^{i_2^*}) \end{bmatrix} \tag{20}$$

where the $\mu_x(\cdot)$ and $\mu_y(\cdot)$ functions return the mean values of the elements of the input matrix, evaluated along the x and y axes respectively.

The final transformed map is defined as:

$$m_2' = T_{x,y,0}(T_{0,0,\psi}(map_2)) = T_{x,y,\psi}(map_2) \tag{21}$$

For the PEMs in Fig. 8, the Γ matrix, defined in (17), is:

$$\Gamma_1^2 = \begin{bmatrix} 0.77 & 0.26 & 3.26 & 2.35 & 4.71 & 3.23 & 0.35 & 2.31 \\ 0.28 & 0.55 & 1.61 & 0.60 & 0.87 & 0.87 & 1.37 & 0.32 \\ 3.51 & 2.44 & 7.38 & 13.72 & 52.13 & 19.63 & 2.30 & 5.13 \\ 0.71 & 0.71 & 10.94 & 10.15 & 19.63 & \mathbf{52.08} & 0.91 & 1.68 \\ 0.49 & 0.93 & 5.33 & 11.86 & 17.59 & 15.85 & 1.17 & 1.13 \\ 0.77 & 0.49 & 9.90 & 7.32 & 13.32 & 18.86 & 0.62 & 1.79 \\ 0.48 & 0.48 & 2.74 & 0.55 & 0.75 & 0.75 & 1.18 & 0.56 \end{bmatrix}. \tag{22}$$

There are seven rows corresponding to the seven edges in the first map and eight columns corresponding to the eight edges in the second map.

From (19), the peak of the matrix is at $i^\star = 3$ and $j^\star = 5$. This means that edge 3 from the first map is the the best match with edge 5 from the second map (see Fig. 8, where matching pairs of edges have identification numbers with the same color). Moreover, edge 4 from the first map is a good match with edge 6 from the second map. However, both matching results will generate similar translation vectors. Using (20), the translation is $T = [-80, -70]^T$, which is the translation required to fuse the maps. In general, if there exists multiple good match candidates, the translation resulting from each of the matches can be used or the results can be averaged. In the case that there are no matching edges in the maps, the map fusion process cannot be completed. The existence of the solution and verification of the matches are explained further in the following section.

It should be emphasized that the associated probabilities for each cell in the edges of the PGVD play an important role in determining the values from the cross-correlation. Edges with cells with higher probability of being true GVD cells will result in higher cross-correlation values.

1) Existence of a Solution: False matches can be identified by analyzing the results of the 2D cross correlation.

Assume that there is an edge with l_1 cells within $e_1^{i_1^*}$ and an edge with l_2 cells within $e_2^{i_2^*}$. Assume that the best match in the 2D cross correlation results in L matching cells. Define the subset of cells in $e_k^{i_k}$ that were matched as $\varepsilon_k^{i_k}$. We can define the average confidence for the matched edge as:

$$\alpha_k^{i_k} = \frac{1}{L} \sum_{l=1..L} p(\varepsilon_k^{i_k}) \tag{23}$$

By expanding equation (16), $\max(\mathcal{E})$ becomes $\alpha_1^{i_1} \alpha_1^{i_2} L$.

Given two edges with lengths l_1 and l_2 with unknown mutual matching cells, the best matching or maximum value in the correlation matrix should satisfy the following to be considered as a candidate match:

$$\max(\mathcal{E}^{(i_1,i_2)}) > \rho \alpha_1^{i_1} \alpha_1^{i_2} \min(l_1, l_2), \tag{24}$$

where ρ is a desired matching percentage. If there is no element in the Γ_1^2 matrix that satisfies (24), then the matching process fails. For each pair of the edges, i_1 and i_2, the following relation should hold to accept them as a valid match:

$$\frac{\max(\mathcal{E}^{(i_1,i_2)})}{\rho \alpha_1^{i_1} \alpha_1^{i_2} \min(l_1, l_2)} > 1. \tag{25}$$

For our example, $\alpha_1^3 = 0.7402$ and $\alpha_2^5 = 0.7381$. Assuming that the desired ρ is 95%, then

$$\begin{aligned}\rho\alpha_1^3\alpha_1^5 \min(l_3, l_5) &= (0.95)(0.7402)(0.7381)min(96, 96)\\ &= 49.83 < \max(\mathcal{E}^{(3,5)}) = 52.13, \end{aligned} \quad (26)$$

which means the match between edge 3 from the first map and edge 5 from the second map is a valid match. This matching criterion can be calculated for each pair of the edges represented in the Γ matrix. For the given example, if the criterion in (25) is calculated for all edges in the Γ matrix, then the following matrix is the result:

$$\begin{bmatrix} 0.089 & 0.030 & 0.217 & 0.160 & 0.225 & 0.152 & 0.035 & 0.191 \\ 0.030 & 0.060 & 0.156 & 0.059 & 0.061 & 0.060 & 0.122 & 0.031 \\ 0.277 & 0.192 & 0.223 & 0.637 & \textcolor{red}{\mathbf{1.046}} & 0.397 & 0.158 & 0.289 \\ 0.056 & 0.056 & 0.327 & 0.467 & 0.399 & \textcolor{blue}{\mathbf{1.045}} & 0.062 & 0.093 \\ 0.053 & 0.101 & 0.329 & 0.758 & 0.778 & 0.692 & 0.119 & 0.088 \\ 0.085 & 0.053 & 0.627 & 0.473 & 0.605 & 0.846 & 0.060 & 0.141 \\ 0.062 & 0.062 & 0.322 & 0.065 & 0.063 & 0.062 & 0.128 & 0.064 \end{bmatrix}, \quad (27)$$

in which only the elements which are greater than one are valid matches.

The aligned maps after applying the uncertain rotation are used to find the relative translation. After finding the translation, Algorithm 1 is used again to propagate the uncertainty of the translation, assuming zero rotation and $\sigma^2_{\psi\psi} = 0$. We now have a fully probabilistic representation of the required transformation and can proceed to fuse the maps.

E. Map Fusion with the Entropy Filter

After finding the relative transformation between the two maps, the probabilities are combined and filtered to produce the final map. The data that is received in m'_2 is akin to a batch of sensor data and should be incorporated by using the additive property of the log odds representation of occupancy originally defined in (7).

$$l_{fused}(i, j) = l_1(i, j) + l'_2(i, j) \quad (28)$$

for all $i = 1..N, j = i..M$. The probabilities can then be recovered from (7). The entropy filter is applied to the fused map, m_{fused} [31]. The entropy filter compares the original map, m_1 and the fused map m_{fused} and rejects updates that result in higher entropy.

For the case of a discrete binary RV, such as each cell of the OGM, $m(i, j)$ with $p(m(i, j)) = p_{ij}$, the entropy can be described by:

$$H(m(i, j)) = -p_{ij} \log p_{ij} - (1 - p_{ij}) \log (1 - p_{ij}) \quad (29)$$

Mutual information, I_{ij} is defined as the reduction in entropy at location (i, j) between the original map, m_1 and the fused map, m_{fused}:

$$I_{ij} = H(m_1(i, j)) - H(m_{fused}(i, j)) \quad (30)$$

The final map, m_{final} is defined as:

$$m_{final}(i, j) = \begin{cases} m_{fused}(i, j) & I_{ij} \geq 0 \\ m_1(i, j) & I_{ij} < 0 \end{cases} \quad (31)$$

where only values from m_{fused} which result in positive information are kept. It should be noted that, in the case of subsequent map fusion operations, the original maps which contain no information from others should be used. This is to avoid having overconfidence by fusing maps repeatedly. Fig. 11 shows the final fused map after the proposed entropy filter.

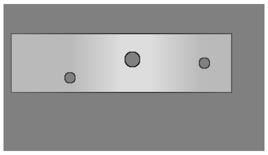

Fig. 11. Final fused maps at the output of the entropy filter.

V. Experimental Results

To demonstrate the effectiveness of the proposed method, four experiments are presented. The first one is performed on a standard online dataset. In the presentation of the results for this experiment, most details of the proposed algorithm are explained. The second experiment is a real world experiment performed with two CoroBot robots in an indoor environment in the basement of the University of New Brunswick (UNB). The third experiment shows the map merging algorithm with three maps of the Department of Electrical and Computer Engineering of UNB. Finally, the last experiment is performed in a simulated MATLAB environment with more obstacles to show the performance of the algorithm for more complex environments.

The three robots used are built by CoroWare, Inc. and each is equipped with high speed Phidget Encoders and a Hokuyo UTM-30LX laser ranger as shown in Fig. 3. SLAM on the individual robots is performed with the particle filtering algorithm presented in [8].

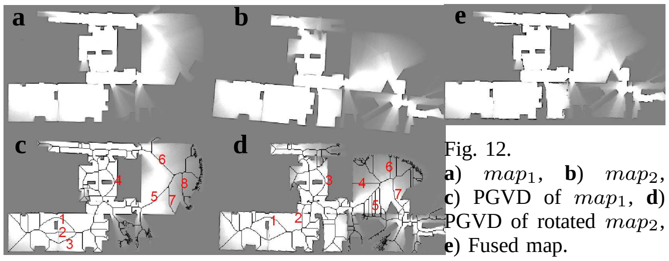

Fig. 12.
a) map_1, **b)** map_2, **c)** PGVD of map_1, **d)** PGVD of rotated map_2, **e)** Fused map.

A. Experiment 1: RADISH Dataset

The first experiment is performed on the RADISH Fort AP Hill dataset [32]. Fig. 12-a and Fig. 12-b show two input maps before alignment. Fig. 12-c and Fig. 12-d show the PGVDs of the maps after being aligned. The alignment is 5.5^o. The marked edges are the filtered edges for cross matching. The Γ_1^2 matrix is shown in (34). From (19), $i^* = 4$ and $j^* = 3$ which means the fourth edge of the first map has the highest similarity to the third edge of the second map. These two edges are used to calculate the translation vector from (20). The final translation vector is $T = [-25, -1]^T$ and the rotation is $\psi = 5.5^o$.

From (23), $\alpha_1^4 = 0.86$ and $\alpha_2^3 = 0.89$ are the average confidences of the matched edges. The lengths of these two edges were 44 and 45 cells. The matching percentage, ρ, was chosen to be 95%. According to (24)

$$(0.95)(0.86)(0.89) \min (44, 45) = 31.99 < 32.9, \quad (32)$$

therefore the match is valid. Note that the match of e_1^1 with e_2^1, given in $\Gamma_1^2(1, 1)$, is disqualified because the average

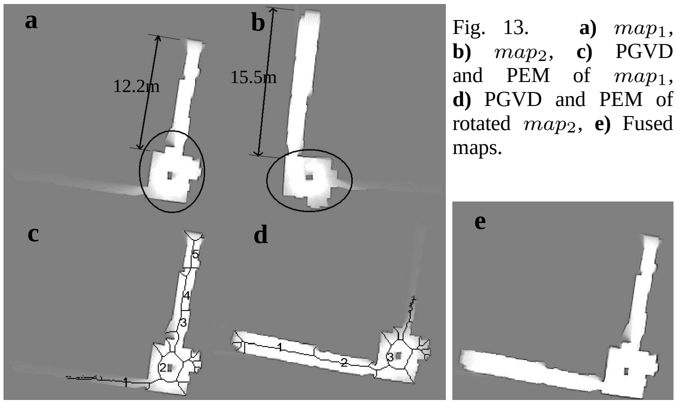

Fig. 13. **a)** map_1, **b)** map_2, **c)** PGVD and PEM of map_1, **d)** PGVD and PEM of rotated map_2, **e)** Fused maps.

confidences of those edges were higher: $\alpha_1^1 = 0.91, \alpha_2^1 = 0.92$ and the lengths of the edges are 56 and 62.

$$(0.95)(0.91)(0.92)\min(56, 62) = 44.54 > 31.6. \tag{33}$$

$$\Gamma_1^2 = \begin{bmatrix} \mathbf{31.6} & 2.0 & 6.4 & 6.1 & 1.9 & 0.8 & 5.4 \\ 12.9 & 1.9 & 2.0 & 11.7 & 0.9 & 0.5 & 1.6 \\ 11.9 & 2.8 & 5.4 & 3.0 & 1.4 & 0.5 & 2.5 \\ 10.7 & 6.7 & \textcircled{\color{red}\mathbf{32.9}} & 1.0 & 5.3 & 1.8 & 8.1 \\ 9.1 & 1.9 & 8.9 & 2.3 & 0.9 & 0.5 & 1.6 \\ 5.9 & 0.4 & 2.1 & 0.4 & 0.4 & 0.2 & 1.9 \\ 3.2 & 9.9 & 3.9 & 0.5 & 11.6 & 4.0 & 6.9 \\ 0.9 & 7.2 & 2.9 & 0.3 & 10.2 & 3.5 & 4.1 \end{bmatrix} \tag{34}$$

To show the benefit of the probabilistic approach, Γ_{det}, the deterministic Γ, is shown in (35). In this case, matching is performed directly on the GVD instead of the PGVD. Edges that were on the periphery of the explored area in the GVD are now matched quite well, when, in reality, those edges do not represent well the structure of the discovered map. As a result, the correct match is much less clear. By comparing those elements from Γ_{det} shown by bold font, with the same elements in Γ_1^2, it is obvious that how the probabilistic structure can eliminate false or weak matches. Fig. 12-h shows the final aligned maps.

$$\Gamma_{det} = \begin{bmatrix} \mathbf{38} & 2 & 9 & 10 & 2 & 2 & 9 \\ 17 & 2 & 2 & \mathbf{30} & 2 & 1 & 2 \\ 16 & 3 & 6 & 5 & 3 & 2 & 3 \\ 11 & 9 & \textcircled{\color{red}\mathbf{40}} & 1 & 10 & 8 & 12 \\ 14 & 2 & 10 & 3 & 2 & 1 & 2 \\ 14 & 1 & 7 & 1 & 1 & 1 & 8 \\ 6 & \mathbf{29} & 9 & 1 & \mathbf{29} & \mathbf{29} & 25 \\ 3 & \mathbf{29} & 9 & 1 & \mathbf{32} & \mathbf{32} & 21 \end{bmatrix} \tag{35}$$

B. *Experiment 2: Two CoroBot Robots*

In this experiment, two robots are used. Fig. 13-a and Fig. 13-b show the OGMs built by the robots. The total area is $81.02m^2$ and the trajectories were $22.5m$ and $25.9m$. In both maps, there are non-overlapping corridors with almost the same size ($15.5m$ and $12.2m$). But the algorithm is capable of rejecting them as matching and finding the transformation based on the overlap. Fig. 13-c and Fig. 13-d depict the PGVD of the aligned maps with selected edges for matching marked with numbers. The edges marked with number 2 in Fig. 13-c and number 3 in Fig. 13-d are used to calculate the translation. Finally, Fig. 13-e shows the final fused map.

C. *Experiment 3: Three CoroBot Robots*

This experiment is performed in a larger environment, with an area of approximately $600m^2$ and three agents are involved. The trajectories of the robots are approximately $60m$, $35m$ and $55m$. By merging the maps, loop closure happens successfully. Fig. 14-a, b and c show the three local maps. Maps of Fig. 14-b and c are fused to Fig. 14-a. The overlapping region is indicated in the figure. The fused global map of the environment is shown in Fig. 14-d.

D. *Experiment 4: Simulated Highly Cluttered Environment*

This experiment is performed in a simulated environment which includes 30 blocks with different sizes and orientations. Two local maps are generated which have 50% overlap. Map merging using the proposed probabilistic approach successfully fuses the local maps shown in Fig. 15-a and 15-b. Fig. 15-c and 15-d demonstrate the PEMs of the aligned maps which are probabilistic GVDs with short edges removed. The first PEM has 44 edges and the second one has 45 edges. A few of the matching pairs of the edges are highlighted with stars of the same color. Finally, Fig. 15-e shows the final maps generated using the proposed entropy filter.

E. *Comparison*

As mentioned, a major benefit of edge matching for map fusion is the low processing time requirement. The proposed method is compared with adaptive random walk (ARW) map merging [4] and Map Segmentation [30] methods.

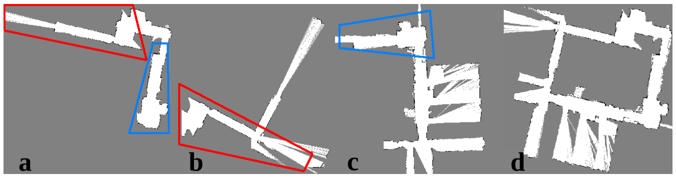

Fig. 14. Three partial maps are fused together to generate a global map. (**a**) is the base map where (**b**) and (**c**) are fused to that. Overlaps are marked with polygons. (**d**) Final fused map which depicts loop closure.

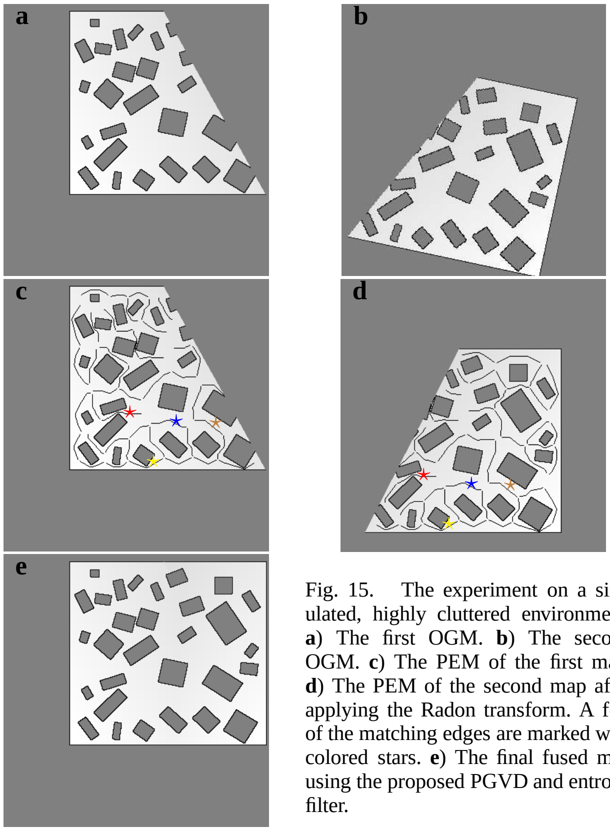

Fig. 15. The experiment on a simulated, highly cluttered environment. **a**) The first OGM. **b**) The second OGM. **c**) The PEM of the first map. **d**) The PEM of the second map after applying the Radon transform. A few of the matching edges are marked with colored stars. **e**) The final fused map using the proposed PGVD and entropy filter.

Table I summarizes the comparison of the processing times and verification for the dataset and real-world experiments. To do an accurate comparison, all algorithms and computations were implemented and performed on a Core2Duo 2.66 GHz laptop. As the results show, the proposed method operates at least eight times faster and the verification index [4] shows the accuracy of the results. The comparison results for the simulation are not provided since the other two methods are not effective for such large and complex environments.

TABLE I
Processing time and efficiency of three experiments, ① Radish dataset, ② two CoroBots, ③ three CoroBots.

Experiment	①	②	③	①	②	③
Method	Processing (sec)			Verification (%)		
PGVD based	12	10	13	95	91	92
Map segmentation [30]	105	83	106	95	92	94
ARW map merging [4]	168	150	152	93	88	92

VI. Conclusion and Future Work

A probabilistic multiple robot SLAM algorithm has been presented that is fast and robust. The probabilistic GVD is used to fuse maps and account for uncertainties in the occupancy grid map. The proposed method has been shown to preferentially fuse areas of the maps that have low uncertainty. Experiments and comparison with other established methods show that it is effective and 9 to 14 times faster.

In future work, it would be interesting to use the structure of the GVD to verify the accuracy of the map matching. If different pairs of edges are matched with high accuracy, then the structure of the GVD could allow us to determine that both sets of edges correspond to the same transformation, increasing the likelihood of correct matching.

Acknowledgement

This research is supported by Natural Sciences and Engineering Research Council of Canada (NSERC) and Canada Foundation for Innovation. The authors would also like to thank Jacqueline Paull for her contributions.

References

[1] S. Saeedi, L. Paull, M. Trentini, M. Seto, and H. Li, “Efficient map merging using a probabilistic generalized Voronoi diagram,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 4419–4424.

[2] H. Durrant-Whyte and T. Bailey, “Simultaneous localization and mapping (SLAM): Part I the essential algorithms,” *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: The MIT press, 2005.

[4] A. Birk and S. Carpin, “Merging occupancy grid maps from multiple robots,” *Proceedings of the IEEE: Special Issue on Multi-Robot Systems*, vol. 94, no. 7, pp. 1384–1387, 2006.

[5] J. L. Gould, “Honey bee cognition,” *Cognition*, vol. 37, pp. 83–103, 1990.

[6] A. W. Siegel and S. H. White, “The development of spatial representations of large-scale environments,” *Advances in Child Development and Behavior*, vol. 10, pp. 9–55, 1975.

[7] H. Bunke, “Graph matching: Theoretical foundations,algorithms, and applications,” in *International Conference on Vision Interface*, 2000, pp. 82–88.

[8] A. Howard, “Multi-robot simultaneous localization and mapping using particle filters,” *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1243–1256, 2006.

[9] R. Smith, M. Self, and P. Cheeseman, “A stochastic map for uncertain spatial relationships,” in *Fourth International Symposium of Robotics Research*, 1987, pp. 467–474.

[10] S. Thrun and Y. Liu, “Multi-robot SLAM with sparse extended information filers,” *Advanced Robotics*, vol. 15, pp. 254–266, 2005.

[11] X. S. Zhou and S. I. Roumeliotis, “Multi-robot SLAM with unknown initial correspondence: The robot rendezvous case,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006, pp. 1785–1792.

[12] A. Gil, O. Reinoso, M. Ballesta, and J. Miguel, “Multi-robot visual SLAM using a Rao-Blackwellized particle filter,” *Robotics and Autonomous Systems*, vol. 58, pp. 68–80, 2010.

[13] D. Hahnel, W. Burgard, D. Fox, and S. Thrun, “An effcient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003, pp. 206–211.

[14] S. Thrun, “A probabilistic on-line mapping algorithm for teams of mobile robots,” *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 335–363, 2001.

[15] S. Carpin, “Fast and accurate map merging for multi-robot systems,” *Autonomous Robots*, vol. 25, pp. 305–316, 2008.

[16] S. Carpin, A. Birk, and V. Jucikas, “On map merging,” *Robotics and Autonomous Systems*, vol. 53, pp. 1–14, 2005.

[17] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations (Intelligent Robotics and Autonomous Agents)*. Cambridge, MA, USA: The MIT Press, 2005.

[18] E. R. Davies, *Machine Vision, Theory, Algorithms, Practicalities, 3rd Edition*. Morgan Kaufmann, 2005.

[19] H. Blum, “A transformation for extracting new descriptors of shape,” *Models for the Perception of Speech and Visual Form*, pp. 362–381, 1967.

[20] B. Lau, C. Sprunk, and W. Burgard, “Improved updating of Euclidean distance maps and Voronoi diagrams,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 281–286.

[21] P. Beeson, N. K. Jong, and B. Kuipers, “Towards autonomous topological place detection using the extended Voronoi graph,” in *Proceedings of the IEEE/RSJ International Conference on Robotics and Automation (ICRA)*, 2004, pp. 4373–4379.

[22] H. Choset and K. Nagatani, “Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 2, pp. 125–137, 2001.

[23] H. Choset, S. Walker, K. Eiamsa-Ard, and J. Burdick, “Sensor-based exploration: incremental construction of the hierarchical generalized Voronoi graph,” *The International Journal of Robotics Research*, vol. 19, no. 2, pp. 125–148, 2000.

[24] J. O. Wallgrum, “Voronoi graph matching for robot localization and mapping,” *Transactions on computational science IX*, vol. 6290, pp. 76–108, 2010.

[25] P. Beeson, J. Modayil, and B. Kuipers, “Factoring the mapping problem: Mobile robot map-building in the hybrid spatial semantic hierarchy,” *The International Journal of Robotics Research*, vol. 29, no. 4, pp. 428–459, 2010.

[26] A. Ranganathan, E. Menegatti, and F. Dellaert, “Bayesian inference in the space of topological maps,” *IEEE Transactions on Robotics*, vol. 22, no. 1, pp. 92–107, 2006.

[27] S. Friedman, H. Pasula, and D. Fox, “Voronoi random fields: Extracting the topological structure of indoor environments via place labeling,” in *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*, 2007, pp. 2109–2114.

[28] D. Silver, D. Ferguson, A. Moms, and S. Thayer, “Feature extraction for topological mine maps,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004, pp. 773–779.

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

[30] S. Saeedi, L. Paull, M. Trentini, and H. Li, “Multiple robot simultaneous localization and mapping,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 853–858.

[31] D. Fox, W. Burgard, S. Thrun, and A. B. Cremers, “Position estimation for mobile robots in dynamic environments,” in *Proceedings of the American Association for Artificial Intelligence (AAAI)*, 1998.

[32] [Online]. Available: http://cres.usc.edu/radishrepository/