A Probabilistic Approach to Automated Construction of Topological Maps using a Stochastic Robotic Swarm

Ragesh K. Ramachandran[1], Sean Wilson[1] and Spring Berman[1]

***Abstract*—In this paper, we present a novel procedure for constructing a topological map of an unknown environment from data collected by a swarm of robots with limited sensing capabilities and no communication or global localization. Topological maps are sparse roadmap representations of environments that can be used to identify collision-free trajectories for robots to navigate through a domain. Our method uses uncertain position data obtained by robots during the course of random exploration to construct a probability function over the explored region that indicates the presence of obstacles. Techniques from topological data analysis, in particular the concept of persistent homology, are applied to the probability map to segment the obstacle regions. Finally, a graph-based wave propagation algorithm is applied to the obstacle-free region to construct the topological map of the domain in the form of an approximate Generalized Voronoi Diagram. We demonstrate the effectiveness of our approach in a variety of simulated domains and in multi-robot experiments on a domain with two obstacles.**

***Index Terms*—Swarms; Mapping; Probability and Statistical Methods**

I. Introduction

SWARMS of low-cost, expendable robots can potentially be used in a variety of applications, such as exploration, environmental sensing, disaster response, and search-and-rescue operations, that require mapping an unknown, possibly hazardous environment in order to perform desired tasks. The robots may not be equipped with sufficient power to have onboard GPS or inter-robot communication devices, or they may be deployed in GPS-denied environments (e.g., underground or indoors) with unreliable communication. These resource limitations would preclude the use of conventional techniques of exploration and mapping, such as simultaneous localization and mapping (SLAM) [1], [2].

For such scenarios, we address the specific problem of finding safe robot trajectories using uncertain localization data acquired by robots with onboard odometry. Toward this end, we use the robots' data to construct a *topological map*, which is a sparse representation of an environment that encodes all of its topological features, such as holes that represent obstacles, and provides a collision-free path through the environment in the form of a roadmap [3]. If the domain is embedded in $\mathbb{R}^2$, then the topological map can be mathematically described as a one-dimensional *deformation retract* of the domain [4]. A topological map can be constructed in the form of a *Generalized Voronoi Diagram (GVD)* [5]. A GVD provides all possible path homotopies in an environment containing obstacles and indicates the maximum clearances from obstacles and the domain boundary. In addition, since GVDs are graphs, standard graph search algorithms can be used for planning on GVDs. Due to the computational complexity of computing exact GVDs, algorithms have been developed to generate *approximate GVDs (AGVDs)* in practice [6].

The novel contribution of this paper is an automated procedure, which combines existing techniques from algebraic topology and graph search, for generating the topological map of an unknown, GPS-denied environment using data from a swarm of robots with limited sensing and no inter-robot communication. This procedure is an extension of our work in [7], which also computes a probabilistic map of a domain using uncertain localization data from randomly exploring robots. However, the approach in [7] only estimates the number of topological holes (obstacles) in the domain and does not construct a topological map, as we do in this paper. The estimate in [7] is derived from a *Rips complex* filtration [4] with a heuristically chosen filtration parameter, whereas the topological map in this paper is constructed from a probability-based filtration which outputs the optimal filtration parameter. Our prior work [8] also presents a method for mapping GPS-denied environments using a swarm of robots with stochastic behaviors. This approach employs optimal control of partial differential equation models rather than the topological techniques used in this paper. Although the method in [8] only requires robot data on encounter times with features of interest, it is limited in application to domains with a few sparsely distributed features.

We employ tools from topological data analysis (TDA) to segment the obstacle regions in the domain by constructing a probability-based filtration on the domain's free space, thereby simultaneously computing the optimal filtration parameter and estimating the number of topological features in the domain. TDA has previously been used for super-level set estimation of probability densities [9]. Next, we use a graph-based *wave propagation algorithm* [10], [11] to construct the topological

Manuscript received: September 12, 2016; Revised November 28, 2016; Accepted December 24, 2016.

This paper was recommended for publication by Editor Nak Young Chong upon evaluation of the Associate Editor and Reviewers' comments. This research was supported by NSF Award CMMI-1363499, ONR Young Investigator Award N00014-16-1-2605, and DARPA Young Faculty Award D14AP00054.

[1]Ragesh K. Ramachandran, Sean Wilson, and Spring Berman are with the School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ 85287, USA rageshkr@asu.edu, Sean.T.Wilson@asu.edu, spring.berman@asu.edu

Digital Object Identifier (DOI): see top of this page.

map of the domain in the form of an AGVD. This map can then be used by humans or more expensive robots to navigate safely through the environment. The data-gathering portion of our procedure is decentralized, in that the robots act autonomously and a central supervisor is not required to control their individual operations. After this phase, a central server is needed to construct the AGVD from the collected robot data, since the robots do not have the resources to perform these computations onboard. While we only consider 2D environments in this paper, our techniques can be extended to 3D environments as well.

Although constructing a GVD from a given configuration space is a well-studied topic [12], [13], there is little work on constructing a GVD using a swarm of resource-restricted robots. Compared to existing methods that do address this problem, our procedure uses robots with fewer requirements. Most of the existing work on using robots to generate topological maps requires the robots to have sophisticated sensing and localization capabilities [12], [14]. Other work on exploration and topological mapping uses robots with limited sensing capabilities, but does not scale well in a swarm robotic framework [15]. A similar approach to ours is presented in [11], which constructs an AGVD of a domain by combining a graph search algorithm with a coverage algorithm based on concepts from algebraic topology. Unlike our procedure, this approach requires each robot to communicate with a central server that commands its actions. In [16], a simplicial approximation of a region of interest is obtained as a topological map by constructing dual pairs of nerves using relevant visibility and observation covers. This strategy requires the robots to detect, identify, and store landmarks in the domain such as obstacle corners and edges, and therefore requires the robots to have higher sensing and processing capabilities than in our procedure. The approach in [17] generates a point cloud of the domain's free space in a coordinate-free manner and employs persistent homology to compute topological features in the domain. This strategy requires inter-robot communication, and each robot must have a unique identifier that is recognized by other robots.

II. Background

Topological Data Analysis (TDA) [18] provides algorithmic and mathematical tools for studying topological and geometric attributes of noisy data in a coordinate-free manner. Detailed treatments of the theoretical and computational aspects of TDA are given in [19], [20]. Techniques from TDA have been previously applied to problems in robotics [21], sensor networks [22], and localization [23]. The key idea underlying TDA is that data have an inherent *shape* that encodes important information regarding their global structure. TDA uses the mathematical framework of *algebraic topology* [20] to characterize topological features embedded in the data. For many applications, data are obtained as a *point cloud* comprised of noisy samples of an intensity map in a Euclidean space. TDA can be used to compute prominent topological features of a point cloud, which can be presented graphically in a compact fashion with *persistence diagrams* [24] and *barcode diagrams* [25].

A central concept in TDA is *persistent homology* [19], which enables the study of *global* attributes of a space from *local* computations on noisy data that are obtained by sampling the space. Persistent homology can be used to filter topological features that are persistent over a large range of scales. To any topological space $\mathbf{T}$, one can associate a set of vector spaces called *homology groups*, denoted by $H_k(\mathbf{T})$, $k = 0,1,2,...,dim(\mathbf{T})-1$. Every vector space in this set encodes information regarding a particular topological feature of $\mathbf{T}$. The dimension of each vector space $H_k(\mathbf{T})$, denoted by β_k, gives the number of independent topological features represented by that vector space. The dimensions β_k are topological invariants [20] and are commonly referred as *Betti numbers*. In addition, β_k represents the number of independent k-dimensional cycles in $\mathbf{T}$. For instance, if $\mathbf{T}$ is embedded in $\mathbb{R}^2$, denoted by $\mathbf{T} \hookrightarrow \mathbb{R}^2$, then β_0 gives the number of connected components in $\mathbf{T}$ and β_1 indicates the number of holes in $\mathbf{T}$. Similarly, if $\mathbf{T} \hookrightarrow \mathbb{R}^3$ then β_0, β_1, and β_2 specify the numbers of connected components, tunnels, and voids in $\mathbf{T}$, respectively.

Another key concept in algebraic topology is the *abstract simplicial complex*. Although this complex is in general defined on topological spaces constructed on arbitrary sets, here we restrict its definition to Euclidean spaces and use notation from [24]. We say that vectors $v_0, v_1, ..., v_k \in \mathbb{R}^n$ are *affinely independent* if the vectors $v_1 - v_0, ..., v_k - v_0$ are linearly independent. A k-simplex $\sigma \subset \mathbb{R}^n$ can be defined as the convex hull of $k+1$ affinely independent vectors $\{v_0, v_1, ..., v_k\}$, called *vertices*, and is often represented as $\sigma = [v_0, v_1, ..., v_k]$. A *face* τ of the simplex σ is the convex hull of a non-empty subset of $\{v_0, v_1, ..., v_k\}$. This relationship is commonly denoted as $\tau \leq \sigma$. A *simplicial complex* κ is defined as a finite collection of simplices σ such that (1) $\sigma \in \kappa$, (2) $\tau \leq \sigma$ implies that $\tau \in \kappa$, and (3) $\sigma, \sigma' \in \kappa$ implies that $\sigma \cap \sigma'$ is empty or is a face of both σ and σ'.

Simplicial complexes provide discrete representations of an underlying topological space using a combinatorial structure, which can be expressed algebraically with linear operators (matrices). This algebraic structure can be exploited to develop algorithms for homological computations. If $f : \kappa \to \mathbb{R}$ is a function such that $\tau \leq \sigma$ implies that $f(\tau) \leq f(\sigma)$, then $f^{-1}((-\infty, \delta])$ is a simplicial complex denoted by κ_δ and $\delta_1 \leq \delta_2$ implies that $\kappa_{\delta_1} \subseteq \kappa_{\delta_2}$, yielding a *filtration of simplicial complexes* with δ as the *filtration parameter*. The persistent homology is obtained by varying the value of the filtration parameter, computing the generators of homology groups (the basis of the homology group vector spaces) for each simplicial complex obtained for the parameter value, and identifying the persistent homology generators.

A *barcode diagram* is a graphical representation of $H_k(\mathbf{T})$ in terms of its homology generators and can be used to identify the persistent topological features of $\mathbf{T}$. This diagram plots a set of horizontal line segments whose x-axis spans a range of δ values and whose y-axis depicts the homology generators in an arbitrary ordering. The Betti number β_k gives the number of generators of $H_k(\mathbf{T})$. The number of arrows in the diagram indicates the number of persistent topological features of $\mathbf{T}$. Figure 1 illustrates an example of a barcode diagram.

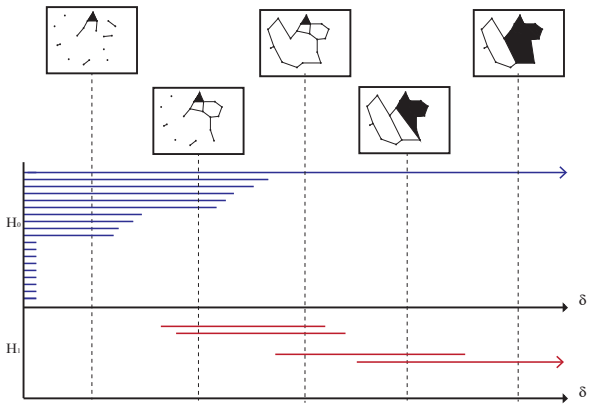


Fig. 1. An example barcode diagram of a filtration. $\beta_k(\delta_i)$ is the number of horizontal segments in the barcode for $H_k(\mathbf{T})$ that intersect the dashed line at $\delta = \delta_i$. The arrows in H_0 and H_1 indicate the persistent topological features. The shaded regions contain the 2-simplices (triangles).

III. Problem Statement

We consider the problem of generating a topological map of a bounded, GPS-denied 2D environment using data collected by a swarm of N robots. We assume that the boundary of the domain is known, but its interior is unknown. We also assume that each robot can identify features and other robots that fall within its local sensing range, enabling it to avoid collisions, and can estimate its position and orientation with uncertainty using measurements from a compass and wheel encoders. After the swarm is deployed into the domain, each robot performs a correlated random walk while estimating its global position using its onboard odometry and refining its measurements using a Kalman filter. At fixed time intervals, each robot records its estimated position and the associated covariance matrix corresponding to the uncertainty of the estimate. After a time T, which we assume is sufficiently large for the robots to thoroughly cover the domain, all robots travel to a common location where their stored data is retrieved.

The robots follow the standard odometry motion model described in [26]. Each robot has a constant speed v and an orientation $\theta(t)$ at time t with respect to a global frame. The velocity and position vectors of a robot at time t are defined as $\mathbf{V}(t) = [v_x(t),\ v_y(t)]^T = [v\ \cos(\theta(t)),\ v\ \sin(\theta(t))]^T$ and $\mathbf{X}(t) = [x(t),\ y(t)]^T$, respectively. At time $t = 0$, the beginning of a deployment, each robot is provided with accurate measurements of its position $\mathbf{X}(0)$ and orientation $\theta(0)$. At the start of every time step Δt during the deployment, each robot generates a uniform random number between 0 and 1 and randomly chooses a new orientation $\theta(t) \in [-\pi, \pi]$ if the number is below a predefined threshold p_{th}. During a time step, the displacement of a robot is described by the equation

$$\mathbf{X}(t+\Delta t) = \mathbf{X}(t) + \mathbf{V}(t)\Delta t + \mathbf{W}(t), \tag{1}$$

where $\mathbf{W}(t) \in \mathbb{R}^2$ is a vector of independent, zero-mean normal random variables that are generated at time t. These variables model randomness in the robot's motion due to sensor and actuator noise.

IV. Topological Map Generation Procedure

A. Procedure Description

In this section, we present a three-step procedure for extracting a topological map of the domain in the form of an AGVD from the data obtained by the robots.

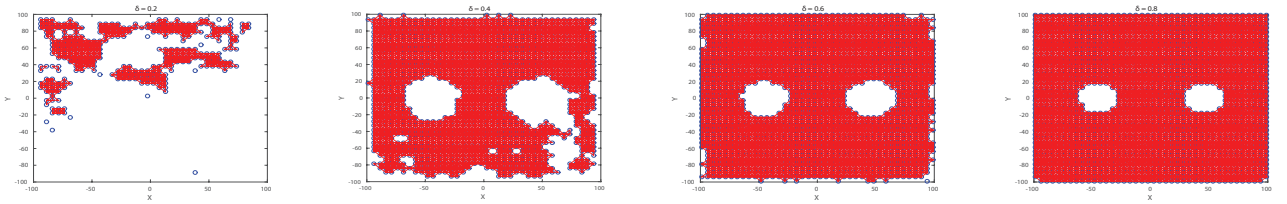
1) Estimation of the Number of Obstacles: We first discretize the domain into a high-resolution uniform grid of M cells, as in occupancy grid mapping algorithms [26], and use the robots' data to assign a probability p_i^f to each grid cell m_i, $i \in \{1,...,M\}$, of being *free*, or unoccupied by an obstacle. We first presented this computation in [7] and summarize it here. During a deployment, the data obtained by robot $j \in \{1,...,N\}$ at time $t_k \in [0,T]$, $k \in \{1,...,K\}$, consists of the tuple $d_k^j = \{\mu_k^j, \Sigma_k^j\}$, where $\mu_k^j \in \mathbb{R}^2$ is the mean of the robot's estimate of its position in Cartesian coordinates at time t_k, and $\Sigma_k^j \in \mathbb{R}^{2\times 2}$ is the covariance matrix of its position estimate at this time. The probability p_{ijk} that robot j occupied grid cell m_i at time t_k is computed for all robots, cells, and measurement times. This discrete probability is calculated by integrating the Gaussian distribution with mean μ_k^j and covariance matrix Σ_k^j over the region $[x_i^l, x_i^u] \times [y_i^l, y_i^u]$ occupied by cell m_i. A score $s_i \in [0,\infty)$ is then assigned to cell m_i according to the equation

$$s_i = \sum_{j=1}^{N}\sum_{k=1}^{K} \log\left(\frac{1}{1-p_{ijk}}\right). \tag{2}$$

This score is rescaled to a value $s_i^C \in [0,C]$, where C is chosen such that the value of $1-\exp(C)^{-1}$ is close to one. The rescaling improves numerical stability when converting the scores to probabilities, especially for values near zero and one. Finally, we compute the probability of cell m_i being free as $p_i^f = 1-(\exp(s_i^C))^{-1}$.

Next, we identify the persistent topological features in the domain and find the *optimal threshold* α_{opt} for which all grid cells with $p_i^f < \alpha_{opt}$ belong to an obstacle. As discussed in Section II, we generate a filtration of simplicial complexes based on a parameter δ in order to compute the persistent homology. Let α denote a given threshold for identifying grid cells m_i that belong to obstacles (the "obstacle grid cells"), according to $p_i^f < \alpha$. We define the filtration parameter δ as $1-\alpha$ in order to be consistent with the conditions described in Section II. Thus, the value of δ varies from 0.1 to 0.9 when the threshold α varies from 0.9 to 0.1. We construct the simplicial complex κ_δ by selecting the center points of the grid cells with $p_i^f \geq \alpha = 1-\delta$ and constructing 1-simplices and 2-simplices from these points (the 0-simplices). The 1-simplices are generated by taking each element of the 0-simplices and pairing it with its immediate vertical, horizontal, and diagonal neighbors (8-connectivity) if the neighbors are elements of the 0-simplices. Thereafter, the 2-simplices are constructed by taking every subset of three elements in the 1-simplices that form a triangle. Figure 2 shows the filtration constructed for the domain in Figure 4(b), which was used in the multi-robot experiments described in Section V-B.

Barcode diagrams are extracted from these filtrations, and the number of barcode arrows in each homology group corresponds to the number of topological features in the domain that are encoded by that particular group. The *optimal filtration*



(a) Simplicial complex, $\delta = 0.2$ (b) Simplicial complex, $\delta = 0.4$ (c) Simplicial complex, $\delta = 0.6$ (d) Simplicial complex, $\delta = 0.8$

Fig. 2. Filtration used to generate the barcode diagram Figure 6(b) for the domain shown in Figure 4(b). The red triangles are the 2-simplices that are constructed from the centers of the grid cells in the domain discretization.

parameter δ_{opt} is defined as the minimum value of δ for which all the topological features are captured by the corresponding simplicial complex. Alternately, it is the value of δ for which there exists no barcode segment other than arrows in any of the homology groups for all values of the filtration parameter greater than this value. Thus, the optimal threshold can be defined as $\alpha_{opt} = 1 - \delta_{opt}$. In practice, we can compute δ_{opt} by taking the maximum value of δ that is spanned by the non-arrow barcode segments in all the homology groups.

We used the MATLAB-based JavaPlex package [27] to perform the persistent homology computations and generate the barcode diagrams. Persistent homology was computed only for dimensions zero and one, since higher dimensions are not relevant for our application.

2) Obstacle Segmentation: In the second step, we use the information gathered in the previous step to segment the portion of the domain that is occupied by obstacles and identify the obstacle grid cells. By definition, the grid cells with $p_i^f < \alpha_{opt}$ belong to an obstacle. Since we have determined the number of obstacles N_O in the domain (the number of arrows in H_1 in the barcode diagram), a straightforward approach would be to use a K means clustering algorithm on the center points of these grid cells. However, since K means techniques are highly sensitive to the choice of randomly initialized points, it is difficult to guarantee the correct classification of the obstacle grid cells.

Instead, to classify the grid cells in each obstacle, we develop an algorithm that takes as input (1) the number of obstacles in the domain, and (2) an *obstacle graph* $\mathcal{G}_O$ whose vertices are the center points of the obstacle grid cells. Denoting the set of these points by C, the edges of $\mathcal{G}_O$ are defined by pairing every element in C with its immediate vertical and horizontal neighbors (4-connectivity) if the neighbors are elements of C. We initialize an open list L with all the elements in C and loop through the following procedure N_O times. We choose an element randomly from L and perform a breadth-first search on $\mathcal{G}_O$ with this element as starting point. The resulting set of elements, denoted by V, consists of the grid cells contained in a single obstacle. Then L is updated by removing those elements in L which are also in V. After the obstacles are segmented in this way, the boundaries of each obstacle are identified as the elements in C that belong to the same obstacle and have fewer than four neighbors, according to $\mathcal{G}_O$. We denote the set of elements along the boundary of the i^{th} obstacle by $\partial\mathcal{O}_i$.

Algorithm 1 Topological Map Generation

Input: Free region graph $\mathcal{G}_f$ with vertex set $V(\mathcal{G}_f)$, $\{\mathcal{B}_k\} = \{\partial\mathcal{O}_i\} \cup \{\partial\mathcal{D}_i\}$

Output: gvd_nodes: subset of $V(\mathcal{G}_f)$ which constitutes the discrete AGVD (topological map)

L = $V(\mathcal{G}_f)$ ▷ Initialize the open list
▷ L is a min heap with ∞ priority $\forall$ elements at start
src[v] = ∞ for all v $\in L$ ▷ Distance to obstacles and domain boundaries
label[v] = -1 for all v $\in L$ ▷ Label variable
gvd_nodes = $\emptyset$
mark = 0
for $\mathcal{B}_k \in \{\mathcal{B}_k\}$ **do**
  for v $\in \mathcal{B}_k$ **do**
    ▷ Label the obstacle and domain boundaries
    label[v] = mark
    src[v] = 0 ▷ Set distance to 0
    L[v] = 0 ▷ Decrease priority of v
  end for
  mark = mark + 1
end for
while $L \neq \emptyset$ **do**
  n = $\arg\min_{\hat{n} \in L}$ src[$\hat{n}$] ▷ Min element of L
  if src[n] == ∞ **then**
    break
  end if
  L = L − n ▷ Remove the min element
▷ Find element in nbh, the expanded neighbor set, with a different label
  q = $\arg\min_{\hat{q} \in \text{nbh(n)}}$ {src[$\hat{q}$] | label[$\hat{q}$] $\neq -1$ & label[$\hat{q}$] $\neq$ label[n]}
  if (src[q] + 1 == src[n]) or (src[q] == src[n]) **then**
    add q to gvd_nodes ▷ Vertex of AGVD
  end if
  U = {u $\in$ nbh(n) | u $\in L$, src[u] > src[n] + dist(n,u)}
  for u $\in$ U **do**
    src[u] = src[n] + dist(n,u) ▷ Update distance
    label[u] = label[n] ▷ Assign same label to neighbor
    L[u] = src[n] + dist(n,u) ▷ Change priority
  end for
end while
return gvd_nodes

3) Voronoi Diagram Construction: In the third step, we develop a new implementation of the wave propagation algorithm to generate the approximate Generalized Voronoi Diagram (AGVD). Let $\partial\mathscr{D}_j$ denote the set of grid cell centers that are closest to the j^{th} edge of the domain boundary. Since we have assumed that the domain boundary is known, we have prior knowledge of the $\partial\mathscr{D}_j$. We define a *free region graph* $\mathscr{G}_f$ whose vertex set $V(\mathscr{G}_f)$ contains the center points of grid cells that lie in the union of the obstacle-free region with the obstacle boundary sets, $\partial\mathscr{O}_i$. This graph is constructed in the same manner as the obstacle graph. We also define the sets $\{\mathscr{B}_k\} = \{\partial\mathscr{O}_i\} \cup \{\partial\mathscr{D}_j\}$ for each obstacle i and each boundary edge j. The distance between $v, w \in V(\mathscr{G}_f)$ is given by a known function $dist(v,w)$, which is based on the discretization of the domain.

The pseudocode of the algorithm is outlined in Algorithm 1. The inputs to the algorithm are the free region graph $\mathscr{G}_f$ and the set $\{\mathscr{B}_k\}$. The algorithm starts by initializing an open list with the vertices of $\mathscr{G}_f$. Then the src-value of each element in $\{\mathscr{B}_k\}$, defined as the distance from the element to the closest obstacle or domain boundary edge, is set to zero and a label is assigned to it based on which $\mathscr{B}_k \in \{\mathscr{B}_k\}$ it belongs to (lines 7-16). The remainder of the algorithm is a modified form of Dijkstra's algorithm [28]. Until the open list is non-empty, at every iteration we choose the element with the minimum src-value from the open list and check whether any of the neighbors of this element can be a part of the AGVD (lines 24-27). We update the src-values of the neighbors of this element and copy its label to its neighbors. The algorithm outputs the set of vertices in $V(\mathscr{G}_f)$, which constitutes the topological map of the domain in the form of a discrete AGVD. Figure 3 illustrates the progress of the algorithm when it is applied to the domain in Figure 4(b).

B. Computational Complexity

The computational complexity of an algorithm is a key factor in determining its feasibility for real-time implementation. Here, we analyze the complexity of each of the main computational blocks in our procedure. From our analysis, we conclude that the worst-case complexity of our approach is $\mathscr{O}(M^{2.372})$, which is same as the worst-case computational complexity of the method in [17], a state-of-the-art mapping technique for a swarm of resource-constrained robots with stochastic motion and no global localization.

1) Probability map generation: In order to generate the probability map of the domain, we need to process the data from each robot for every grid cell. Therefore, if N robots each collect K items of data over a domain that is discretized into M grid cells, and we use this data to compute the probability map, then the cost of computation for this block is $\mathscr{O}(NKM)$. This cost can be reduced by processing the data from each robot in parallel.

2) Simplicial complex construction and barcode generation: The simplicial complex generated from the centers of the M grid cells will have a size proportional to M. The worst-case computational complexity of persistent homology is $\mathscr{O}(M^{2.372})$, but for most practical applications it is close to $\mathscr{O}(M)$ [19].

3) Obstacle segmentation: The most computationally expensive part in this block is the breadth-first search (BFS) performed on the obstacle graph. The computational complexity of BFS on a graph with V vertices and E edges is $\mathscr{O}(V+E)$ [28]. Since the obstacle graph is constructed from a subset of the grid cells of the domain, the number of vertices in the obstacle graph will be a constant factor times M. Thus, the resulting cost becomes $\mathscr{O}(M+E)$. Since the obstacle graph is planar, the number of edges will be a constant factor times V, reducing the cost to $\mathscr{O}(M)$.

4) AGVD extraction using the wave propagation algorithm: The cost of this block can be evaluated by analyzing Algorithm 1. The most computationally expensive part of the algorithm is the loop from lines 17 to 35. As before, the number of vertices in the free region graph is a constant factor times M. Extracting the minimum element in line 18 will cost $\mathscr{O}(\log M)$ due to the heap implementation [28]. During each iteration, a vertex is popped out, and the loop ends when all the vertices are popped. Since the statements inside the loop each have a sub-linear cost, the overall cost of this block is $\mathscr{O}(M \log M)$.

C. Comparison to Other Mapping Algorithms

Table I compares key properties of our approach to those of several existing probabilistic sparse map methods. The properties of these methods are described as in [29]. In the table, the *uncertainty* field states how uncertainty is represented in the resulting map. The *convergence* field describes the convergence properties of the algorithms under suitable assumptions. The *incremental* field indicates whether an algorithm can build the map incrementally or not. The *correspondence* field specifies whether the method can accommodate mapping similar features in the environment. Lastly, the *handles raw data* field states whether the method can construct maps from raw sensor data, or whether the data first requires pre-processing and filtering.

V. Results

A. Simulation Results

We applied our procedure to generate the topological maps of the simulated domain shown in Figure 4. All computations and simulations except for persistent homology computations were done in Python. The simulated robotic swarm consisted of 50 point robots, each with a sensing radius of 0.06 m, an average speed of $v = 0.2$ m/s, and $p_{th} = 0.2$. The robots explored a 2 m $\times$ 2 m domain over a time period of 160 s. At the beginning of each simulation, the robots were placed at random positions near the domain boundary. The robots followed the motion model Equation (1) while dispersing throughout the domain, with the covariance matrix of the random variables in $\mathbf{W}(t)$ set to a diagonal matrix with 0.1 on the diagonal. Upon encountering an obstacle, the domain boundary, or another robot within its sensing radius, a robot randomly chose a different direction to avoid a collision. The robot data obtained after each simulated swarm deployment was used to construct the AGVD of the domain.

TABLE I
COMPARISON OF OUR APPROACH TO SEVERAL PROBABILISTIC SPARSE MAP GENERATION METHODS DESCRIBED IN [29]

	Kalman	Hybrid	Occupancy Grid	Dogma	Our Approach
Map representation	landmark locations	point obstacles	occupancy grids	occupancy grids	occupancy grids
Robot sensor noise	Gaussian	any	any	any	any
Requires exact robot poses	no	no	yes	yes	no
Uncertainty	posterior poses and map	maximum likelihood map	posterior map	posterior map	posterior map
Convergence	strong	no	strong	weak	weak
Incremental	yes	yes	yes	no	yes
Correspondence	no	yes	yes	yes	yes
Handles raw data	no	yes	yes	yes	yes

The outputs at different stages of the procedure for each domain are displayed in the following figures: the contour plots of p_i^f (Figure 5), the barcode diagrams (Figure 6), the obstacle segmentation (Figure 7), and the computed AGVD superimposed over the actual maps (Figure 4). As mentioned previously, the numbers of arrows in the barcode diagram for dimensions 0 (H_0) and 1 (H_1) give the numbers of connected components and features (obstacles) in the domain, respectively. The results in Figure 6 estimate the correct number of topological features and report the value of δ_{opt} for each case. Figure 7 demonstrates that the obstacle segmentation technique described in Section IV-A2, based on the α_{opt} obtained from Figure 6, successfully identifies each distinct obstacle in the domains. Finally, Figure 4 displays the topological map generated for each domain, which display collision-free trajectories among the obstacles as expected. These results show that our procedure can accurately construct topological maps for different scenarios, even though it relies on uncertain robot position data.

To study the failure cases of our approach, we also simulated a complex 20 m $\times$ 20 m domain explored by 300 point robots for 200 s. All other parameters were the same as in the previous simulations. The results are presented in Figure 8. The topological map generated from the robot data does not reveal the narrow gap between the two rectangular obstacles in the center of the domain, since there is a low probability of robots passing through the gap while recording localization estimates. Also, the large size of the domain prevents the robots from obtaining reliable localization data about certain regions before their odometry noise become too high for the associated robot position data to provide any useful information. As Figure 8(c) shows, the lack of reliable data about these regions causes the procedure to incorrectly identify free space in these regions as obstacles. Although the resulting map in Figure 8(d) does not accurately represent the deformation retract of the domain, it does provide a conservative set of collision-free trajectories for the robots. A possible way to obtain accurate topological maps over large domains is to construct the maps locally and patch them together, similar to the approach in [17].

B. Experimental Results

We also tested our procedure in experiments with four Pheeno mobile robots [30] that explored a 1.5 $\times$ 2.1 m rectangular arena with two obstacles. We first analyzed this experimental data in [7] using the approach to topological feature identification presented in that paper. The robots were initially assigned random positions, and they were controlled to move with a linear velocity of 10 cm/s. Other details of the experimental setup are described in [7]. The plots in Figure 9 show that our procedure is effective at building the topological map (AGVD) of the experimental arena.

VI. CONCLUSIONS

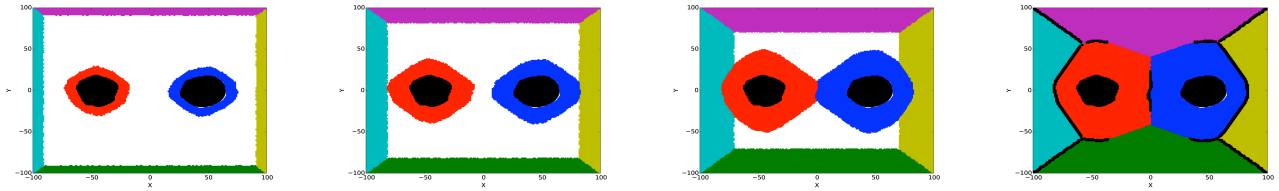
We have formulated a novel procedure for automatically generating the topological map of an unknown environment using data collected by a robotic swarm without global localization or communication. Our procedure constructs an approximate Generalized Voronoi Diagram, which yields collision-free paths through the environment for safe robot navigation. Our proposed methodology was validated through simulations of domains with different numbers of obstacles and illustrated with experiments using mobile robots. In future work, we plan to experimentally test our approach on larger, more complex domains with larger numbers of robots. We will also extend our approach to construct a metric map of an unknown environment using techniques from manifold learning [31], which uses local information to infer the global structure of a domain.

ACKNOWLEDGMENT

R.K.R. thanks Dr. Subhrajit Bhattacharya for his valuable input on this work and Dr. Ganesh P. Kumar for verifying the time complexity analysis of the approach.

REFERENCES

[1] P. Robertson, M. Angermann, and B. Krach, "Simultaneous localization and mapping for pedestrians using only foot-mounted inertial sensors," in *Int'l. Conf. on Ubiquitous Computing (Ubicomp)*, 2009, pp. 93–96.
[2] S. Thrun, "A probabilistic online mapping algorithm for teams of mobile robots," *Int'l. Journal of Robotics Research*, vol. 20, no. 5, pp. 335–363, 2001.
[3] S. Thrun and A. Bü, "Integrating grid-based and topological maps for mobile robot navigation," in *Proc. 13th Nat'l. Conf. on Artificial Intelligence - Vol. 2*, ser. AAAI'96. AAAI Press, 1996, pp. 944–950.
[4] R. Ghrist, *Elementary Applied Topology*, 1st ed. Createspace, 2014.
[5] F. Aurenhammer, "Voronoi diagrams–a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, pp. 345–405, Sep. 1991.
[6] K. E. Hoff, III *et al.*, "Fast computation of generalized Voronoi diagrams using graphics hardware," in *Proc. 26th Annual Conf. on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '99. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999, pp. 277–286.
[7] R. K. Ramachandran, S. Wilson, and S. Berman, "A probabilistic topological approach to feature identification using a stochastic robotic swarm," in *To appear in the Int'l. Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2016.



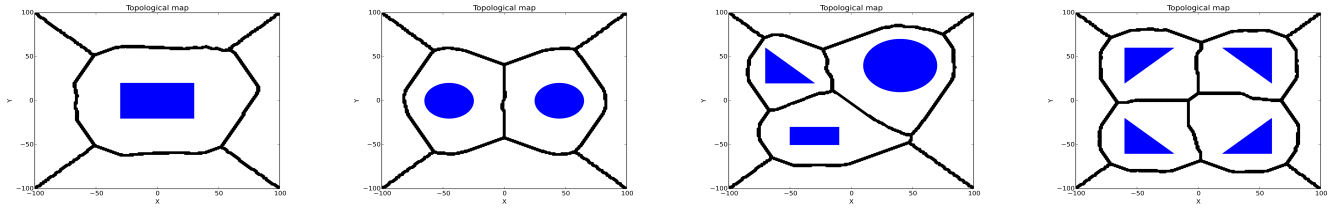
(a) Waves starting to propagate from different edges and obstacles are marked with different colors.

(b) Waves from opposite edges of the domain start to collide with waves from the obstacles

(c) Waves propagate further

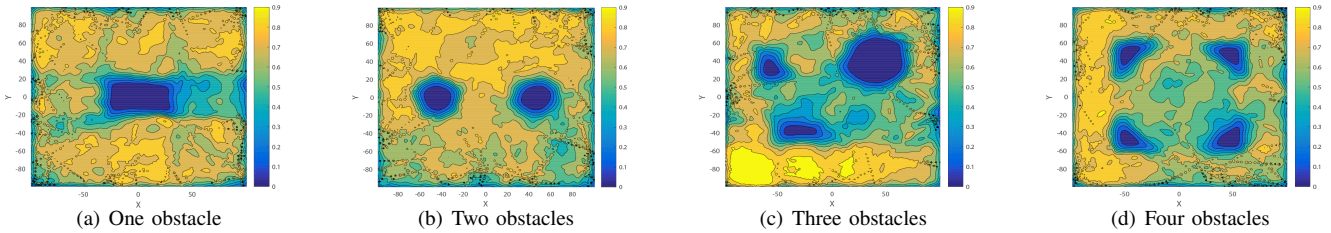
(d) AGVD of the domain is constructed from the wave intersection regions (marked in black)

Fig. 3. Stages of the wave propagation algorithm for constructing a topological map (AGVD) of the domain shown in Figure 4(b).



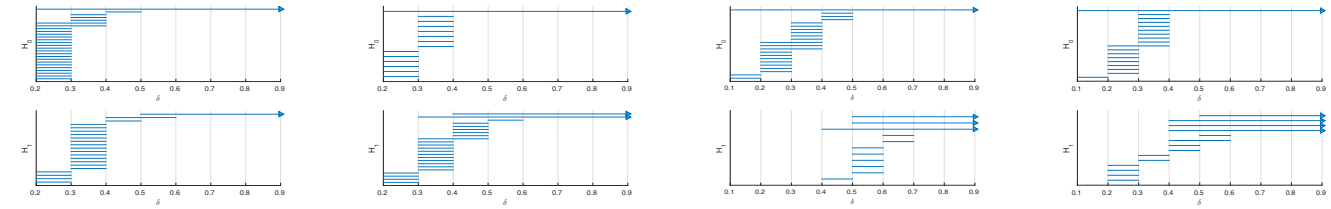
(a) One obstacle (b) Two obstacles (c) Three obstacles (d) Four obstacles

Fig. 4. Actual maps with obstacles in blue and topological maps shown as black lines.



(a) One obstacle (b) Two obstacles (c) Three obstacles (d) Four obstacles

Fig. 5. Contour plots of p_i^f, the probability that grid cell m_i is free, over all grid cells of discretized domains. Colorbar values range from 0 to 0.9.



(a) One obstacle; $\delta_{opt} = 0.6$ (b) Two obstacles; $\delta_{opt} = 0.6$ (c) Three obstacles; $\delta_{opt} = 0.7$ (d) Four obstacles; $\delta_{opt} = 0.6$

Fig. 6. Barcode diagram for each domain, generated from the filtration described in Section IV-A1, with δ_{opt} computed for each case.

[8] R. K. Ramachandran, K. Elamvazhuthi, and S. Berman, "An optimal control approach to mapping GPS-denied environments using a stochastic robotic swarm," in *Int'l. Symp. on Robotics Research (ISRR)*, 2015.

[9] F. T. Pokorny *et al.*, "Persistent homology for learning densities with bounded support," in *Advances in NIPS 25*, P. Bartlett *et al.*, Eds., 2012, pp. 1826–1834.

[10] N. Kalra, D. Ferguson, and A. Stentz, "Incremental reconstruction of generalized Voronoi diagrams on grids," *Robot. Auton. Syst.*, vol. 57, no. 2, pp. 123–128, Feb. 2009.

[11] R. Ramaithitima *et al.*, "Automated creation of topological maps in unknown environments using a swarm of resource-constrained robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 746–753, July 2016.

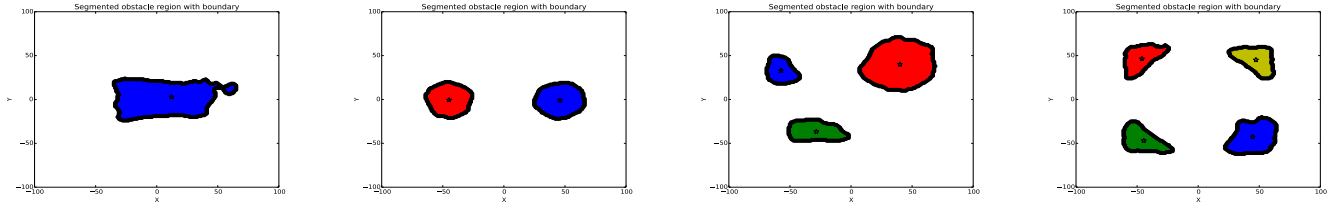
[12] H. Choset and K. Nagatani, "Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 125–137, 2001.

[13] E. G. Tsardoulias *et al.*, "Construction of minimized topological graphs on occupancy grid maps based on gvd and sensor coverage information," *Journal of Intelligent & Robotic Systems*, vol. 75, no. 3, pp. 457–474, 2014.

[14] J. Kim, F. Zhang, and M. Egerstedt, "A provably complete exploration strategy by constructing Voronoi diagrams," *Autonomous Robots*, vol. 29, no. 3, pp. 367–380, 2010.

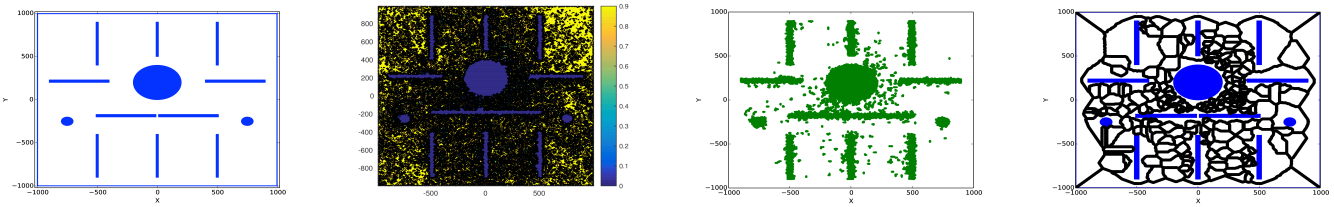
[15] B. Tovar, L. Guilamo, and S. M. LaValle, "Gap navigation trees: Minimal representation for visibility-based tasks," in *Algorithmic Foundations of Robotics VI*. Springer, 2004, pp. 425–440.

[16] R. Ghrist *et al.*, "Topological landmark-based navigation and mapping,"



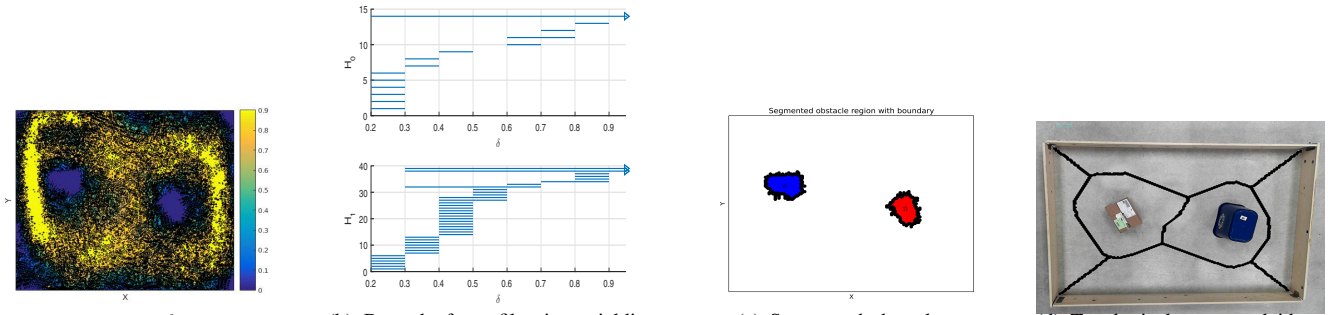
(a) One obstacle (b) Two obstacles (c) Three obstacles (d) Four obstacles

Fig. 7. Obstacles in each domain, segmented based on the probabilities p_i^f using values of α_{opt} obtained from the barcode diagrams in Figure 6. Each colored region represents the interior of an obstacle, with the obstacle boundary marked in black.



(a) Actual map (b) Contour plot of p_i^f; colorbar values range from 0 to 0.9 (c) Segmented obstacles (d) Topological map

Fig. 8. Simulation results for a large, complex environment.



(a) Contour plot of p_i^f; colorbar values range from 0 to 0.9 (b) Barcode from filtration, yielding $\delta_{opt} = 0.9$ (c) Segmented obstacles (d) Topological map overlaid on the experimental arena

Fig. 9. Multi-robot experimental results.

University of Pennsylvania, Department of Mathematics, Tech. Rep. Vol. 8, 2012.

[17] A. Dirafzoon, A. Bozkurt, and E. Lobaton, "A framework for mapping with biobotic insect networks: From local to global maps," *Robotics and Autonomous Systems*, 2016.

[18] G. Carlsson, "Topology and data," *Bulletin of the American Mathematical Society*, vol. 46, no. 2, pp. 255–308, 2009.

[19] H. Edelsbrunner and D. Morozov, "Persistent homology: Theory and practice," *the European Congress of Mathematics*, pp. 31–50, 2012.

[20] A. Hatcher, *Algebraic Topology*. New York, NY: Cambridge University Press, 2002.

[21] S. Bhattacharya, R. Ghrist, and V. Kumar, "Persistent homology for path planning in uncertain environments," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 578–590, June 2015.

[22] R. Ghrist and A. Muhammad, "Coverage and hole-detection in sensor networks via homology," in *IPSN*, 2005.

[23] M. Robinson, *Topological Signal Processing*. Springer-Verlag Berlin Heidelberg, 2014.

[24] H. Edelsbrunner and J. L. Harer, *Computational topology : an introduction*. Providence (R.I.): American Mathematical Society, 2010.

[25] R. Ghrist, "Barcodes: the persistent topology of data," *Bulletin of the American Mathematical Society*, vol. 45, no. 1, pp. 61–75, 2008.

[26] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[27] H. Adams, A. Tausz, and M. Vejdemo-Johansson, "javaPlex: A research software package for persistent (co)homology," in *Mathematical Software – ICMS*. Springer Berlin Heidelberg, 2014, pp. 129–136.

[28] R. Sedgewick and K. Wayne, *Algorithms, 4th Edition*. Addison-Wesley, 2011.

[29] S. Thrun, "Robotic mapping: A survey," in *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[30] S. Wilson *et al.*, "Pheeno, a versatile swarm robotic research and education platform," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 884–891, July 2016.

[31] Y. Ma and Y. Fu, *Manifold Learning Theory and Applications*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2011.