Persistent Homology for Path Planning in Uncertain Environments

Subhrajit Bhattacharya* Robert Ghrist† Vijay Kumar‡

Abstract

We address the fundamental problem of goal-directed path planning in an uncertain environment represented as a probability (of occupancy) map. Most methods generally use a threshold to reduce the gray scale map to a binary map before applying off-the-shelf techniques to finding the best path. This raises the somewhat ill-posed question, what is the right (optimal) value to threshold the map? We instead suggest a *persistent homology* approach to the problem – a topological approach in which we seek the homology class of trajectories that is *most persistent* for the given probability map. In other words, we want the class of trajectories that is free of obstacles over the largest range of threshold values. In order to make this problem tractable and practical, we use homology in $\mathbb{Z}_2$ coefficients (instead of the standard $\mathbb{Z}$ coefficients), and describe how graph search-based algorithms can be used to find trajectories in different homology classes. Our simulation results demonstrate the efficiency and practical applicability of the algorithm proposed in the paper.

I. Introduction

A. Related Work

Motion planning in uncertain environments is an important field of research in robotics. Uncertainty naturally arises in unknown environments, in the presence of process and/or observation noise, and unknown dynamics of the environment. For mobile robots, the most common representation of the environment is an occupancy grid in which each cell is assigned a value of probability of occupancy [27], [26], [14]. Path planning is generally solved by first designating all cells with a probability of occupancy below some threshold to be free (and traversable), while others above the threshold are designated as occupied [24], [4], [23]. Graph search algorithms such as Dijkstra's [11] and A* [17] can be used for finding shortest paths in this graph representation of the environment. However, in this approach, it is unclear how to select this threshold. Low values of threshold result in suboptimal paths while higher values may result in unsafe trajectories.

Alternatively, weights/costs can be assigned to the edges of the graph according to the probability value at the location of the edge, thus penalizing paths that tend to pass through regions with high possibility of occupancy [28], [9]. However, these approaches lack robustness because there are cases when the penalties for some edges are not high enough to offset the incentive offered by a shorter path, and the resulting plan may very well pass through regions with high probability of occupancy.

We note that incremental search algorithms like D* [25] can be used in conjunction with either of the above approaches when the probability map can be updated using incoming sensor data. However these algorithms do not address the fundamental question of how to plan a path for a given probability map.

* Department of Mathematics, University of Pennsylvania. subhrabh@math.upenn.edu

† Department of Mathematics and Department of Electrical and Systems Engineering, University of Pennsylvania. ghrist@math.upenn.edu

‡ Department of Mechanical Engineering and Applied Mechanics, University of Pennsylvania. kumar@seas.upenn.edu

In this paper we consider the fundamental problem of finding a safe trajectory between two points in an environment, given the probability of occupancy/inaccessibility at each point in the environment. We take a topological approach to solving this problem using concepts from the field of *persistent homology* [12], [29], [15], and find the homology class of trajectories that is *most persistent* for a given probability map. We formulate this problem using coefficients in $\mathbb{Z}_2$ (integer coefficients modulo 2) and reduce it to a discrete graph search which can be solved efficiently.

B. Problem Description

We are given a workspace, $W \subseteq \mathbb{R}^n$, and a (continuous) probability distribution on it $P : W \to [0,1]$. This is the probability that a point in W is inaccessible (otherwise known as the occupancy probability). In robotics applications, such a probability map may be obtained from sensor readings and filtering algorithms [27], [14]. Define the ϵ-thresholded space[1] $U^\epsilon = \{q \in W \mid P(q) \leq \epsilon\}$ for $\epsilon \in [0,1]$. This is the subset of W where the probability of occupancy is less than or equal to ϵ – the *free* space assuming that we threshold the probability map at ϵ. For convenience we also define the obstacle set, $O^\epsilon := W - U^\epsilon$. Typically [24], [4], [23] the problem of planning trajectories for a robot in W, given the probability distribution P, boils down to finding an "optimal" ϵ^* that, on one hand should minimize net probability (or some indicator of probability) that a trajectory planned in U^{ϵ^*} will be invalid (or will require substantial change/re-planning during the course of execution), while on the other hand should take into account the objective of minimizing the cost or length of the trajectory.

In this paper we suggest that the correct solution approach to this problem is not to find an optimal value for ϵ. Instead, using concepts from *persistent homology* [12], [29], [15], we suggest processing *all* ϵ-values and extracting the most *persistent* class of trajectories.

We assume some background in *algebraic topology* and *differential topology*. Although we give a quick overview of some of the required concepts in the next section, for more details the reader may refer to the standard texts, [18] and [5], on these respective subjects.

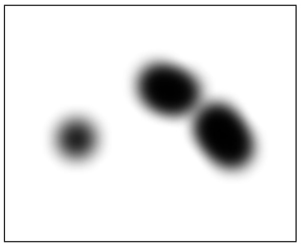
II. Preliminaries

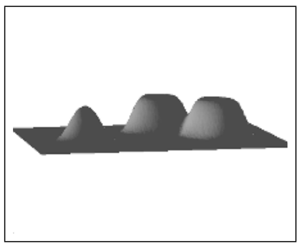
In this section we review some of the standard concepts from topology and algebraic topology [18], [22] and its basic application in search-based path planning [2], [3].

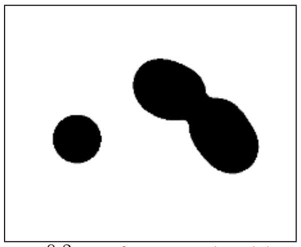
A. Homology

We specialize to (persistent) homology of 1-dimensional curves, which constitute trajectories. This will be indicated as a subscript 1 in various notations for groups that will follow. The homology of a space X, $H_1(X)$, consists of equivalence classes of cycles ("closed loops") [18]. The definition is as follows: In a topological space X (in this paper, X will be the free configuration space, U^ϵ), one considers the set $Z_1(X)$ of all *cycles* (closed oriented loops), given a group structure by means of a formal sum with $\mathbb{Z}$-coefficients. The *boundary group*, $B_1(X)$, is the subgroup of $Z_1(X)$ generated

[1]Throughout this paper the threshold parameter in superscripts of various symbols will indicate an indexing, and not exponential or power.


(a) Probability map, $P : W \to [0,1]$. White: $P = 0.0$, black: $P = 1.0$.


(b) Surface $M = \{[x,y,z] \mid [x,y] \in W, z = P([x,y])\} \subset \mathbb{R}^3$.


(c) $U^{0.2} = \{q \in W \mid P(q) \leq 0.2\}$ (in white).

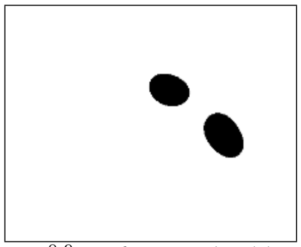
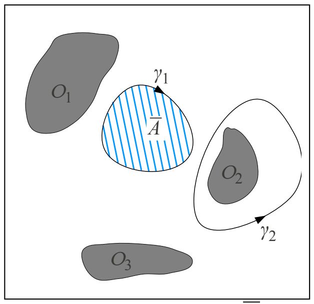

(d) $U^{0.9} = \{q \in W \mid P(q) \leq 0.9\}$ (in white).

Fig. 1: Probability map, P, and the change in topology of U^ϵ as ϵ is changed.

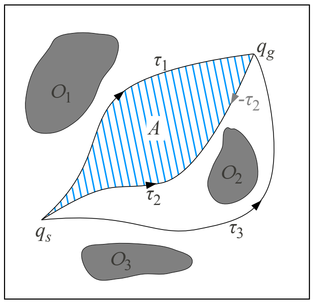
as boundary cycles of a 2-d oriented region (formally, a 2-chain) in X. The (first) homology group of X is then defined as the quotient group $H_1(X) = Z_1(X)/B_1(X)$. What this means is that two cycles, γ_1 and γ_2, are equivalent (written $[\gamma_1] = [\gamma_2]$) if their difference is the boundary of an oriented 2-d region; furthermore, any cycle which bounds a 2-d region represents the zero class $\mathbf{0} \in H_1(X)$ (see Figure 2(a)). For comprehensive definitions, see [18].

This definition can be naturally extended for defining *homology classes of trajectories* connecting fixed start and goal points [2], [3]. We say trajectories τ and τ' connecting fixed start (q_s) and goal (q_g) points, belong to the same homology class if $(\tau \sqcup -\tau')$ is a boundary (*i.e.* $[\tau \sqcup -\tau'] = \mathbf{0} \in H_1(X)$). Thus, in Figure 2(b), since $(\tau_1 \sqcup -\tau_3) \in Z_1(X)$ is not a boundary, τ_1 and τ_3 are not homologous. But since $(\tau_1 \sqcup -\tau_2) \in B_1(X) \subseteq Z_1(X)$, we say τ_1 and τ_2 are homologous. There is however a caveat in the later definition: The set of homology classes of paths/trajectories (connecting fixed end points) defined this way does not have a preferred $\mathbf{0}$ (trivial) element, and thus does not form a *group* (unlike the set of homology classes of cycles). It however forms a set on which $H_1(X)$ acts freely and transitively.

Note 1 (Some remarks on the distinction between homotopy an homology [18]). The "first homotopy group" (denoted $\pi_1(X)$) as well as the "first homology groups" (denoted $H_1(X)$) give classifications of closed loops in an arbitrary topological space, X, and both have higher dimensional generalizations ($\pi_n(X)$ and $H_n(X)$). However, the main distinction between homotopy and homology are that *homotopy groups*, in general, are non-abelian (non-commutative) groups. Homotopy yields a classification with a finer resolution. A *homology group*, on the other hand, is always an abelian (commutative) group and yields a coarser classification of the closed loops. As a result of this, homology groups can be given a vector space-like structure (R-modules, to be more precise) unlike homotopy groups, and lend themselves to computation using standard tools in linear algebra. Unlike homotopy, there are multiple closely related *homology theories* (*e.g.* simplicial homology, singular homology, De Rham cohomology), all or any of which can be used for efficient computations of homology.
Homotopy and homology are used to classify trajectories connecting fixed points in analogous manner [2] – both form spaces on which respectively the homotopy and homology groups act freely and transitively. However, as mentioned, in many contexts homotopy is significantly more difficult computationally, and being non-abelian does not give some of the nice properties that homology gives. It's the very reason why "persistent homology", and not "persistent homoopy" has been vastly studied and could be developed as a fundamental tool in computational topology [12], [29], [15], [8].
The fact that the distinction between *homotopy* and *homology* is mainly in the resolution of the classification can be seen from the Hurewicz theorem – abelianization of the first homotopy group gives us the first homology group (*i.e.* $H_1(X) \approxeq \pi_1(X)/[\pi_1(X), \pi_1(X)]$, where $[\pi_1(X), \pi_1(X)]$ is the *commutator subgroup* of $\pi_1(X)$). As a consequence, all closed loops that belong to the same homotopy class also belongs to the same homology class, but the converse is not necessarily true. Furthermore, the $\mathbb{Z}_2$ coefficients which we will use in this paper, is typically used in the context of homology, and not homotopy. [2] provides an intuitive and simple explanation with examples of this distinction, and shows how the concept of homology can be leveraged for path planning. An interested reader may refer to [18] for a more formal/algebraic description.



(a) γ_1 is the boundary of $\overline{A}$. Thus $\gamma_1 \in B_1(X) \subset Z_1(X)$, and its homology class, $[\gamma_1] = \mathbf{0} \in H_1(X)$. But γ_2 cannot be expressed as a boundary. So, $\gamma_2 \in Z_1(X), \gamma_2 \notin B_1(X)$, and hence $[\gamma_2] \neq \mathbf{0}$ (non-trivial homology class).



(b) $(\tau_1 \sqcup -\tau_2)$ is the boundary of A. Thus, $[\tau_1 \sqcup -\tau_2] = \mathbf{0} \in H_1(X)$. We say τ_1 is homologous to τ_2 or that they belong to the same homology class. But $(\tau_1 \sqcup -\tau_3) \notin B_1(X)$. Thus τ_3 belongs to a different homology class.

Fig. 2: Homology classes of cycles and trajectories.

$\mathbb{Z}_2$ *coefficients:* The standard definition of homology groups is with coefficients in $\mathbb{Z}$: $H_1(X)$ is just a shorthand for $H_1(X;\mathbb{Z})$, which explicitly mentions the coefficient group $\mathbb{Z}$. Such coefficients detect *winding* about an obstacle with orientation (*cf.* "winding numbers"). For example, in Figure 3, since there is a single obstacle in the plane, the homology group $H_1(X)$ is isomorphic to $\mathbb{Z}$. The homology classes of the cycles shown are then $[\gamma_1] = 1 \in H_1(X)$, $[\gamma_2] = 2 \in H_1(X)$, $[\gamma_3] = 3 \in H_1(X)$ — the number in each case being the corresponding winding number. Other coefficients are possible and are introduced at the chain level [18]. For example, with coefficients in $\mathbb{Z}_2 = \mathbb{Z}/2\mathbb{Z}$, the homology group, denoted $H_1(X;\mathbb{Z}_2)$, in Figure 3 is isomorphic to $\mathbb{Z}_2$, with a loss of resolution: $[\gamma_1]_{\mathbb{Z}_2} = [\gamma_3]_{\mathbb{Z}_2} = 1 \in H(X;\mathbb{Z}_2)$ and $[\gamma_2]_{\mathbb{Z}_2} = 0 \in H(X;\mathbb{Z}_2)$ (since the quotient map $\mathbb{Z} \to \mathbb{Z}_2$ records even/odd parity). In this case, the homology class of a cycle that winds around the obstacle twice is in the trivial class, while a cycle that winds around thrice will be in the same class as winding once, and no distinction is made between clockwise and counterclockwise windings. We exploit this coefficient change in the context of robot path planning in Section IV-A.

B. A Homology Invariant

As described in [2], [3] homology invariants of cycles (with coefficients in $\mathbb{R}$) in a topological space X can be constructed as integrals of closed differential 1-forms that generate the *de Rham cohomology group*, $H^1_{dR}(X)$. In particular, in $X = (\mathbb{R}^2 - \mathcal{O})$ (where $\mathcal{O}$ is the set of obstacles consisting of n connected components), we can choose the vector of differential 1-forms, $\omega = [\,\mathrm{d}\theta_1,\, \mathrm{d}\theta_2, \cdots,\, \mathrm{d}\theta_n]^T$ where, $\mathrm{d}\theta_j := \frac{(x-x_j)\ \mathrm{d}y - (y-y_j)\ \mathrm{d}x}{(x-x_j)^2+(y-y_j)^2}$ is the angle subtended by a differential element at (x,y) from the *representative point* (x_j, y_j) — one for each connected component of the obstacles (Figure 4(a)). (Note that one can write $\mathrm{d}\theta_j = Im(\frac{\mathrm{d}z}{z-z_j})$ where $z = x+iy, z_j = x_j + iy_j$ are complex representation of the coordinates – a formulation presented in [2].)

We thus define the *$\mathcal{H}$-signature* of a path τ (which may or may not be a cycle) as $\mathcal{H}(\tau) := \int_\tau \omega$. Then, if γ is a cycle in $Z_1(X)$, the i^{th} element of $\mathcal{H}(\gamma)$ gives the winding number of γ about the i^{th} obstacle. This computes a *complete invariant* for homology classes of cycles: $\mathcal{H}(\gamma_1) = \mathcal{H}(\gamma_2) \iff [\gamma_1] = [\gamma_2] \in H_1(X)$, with $\mathcal{H}(\gamma) = 0$ iff $[\gamma] = \mathbf{0} \in H_1(X;\mathbb{R})$.

Because we fix a coefficients of 1 on the paths in constructing chains and cycles, and the way we construct ω (multiplying the vector of $\mathrm{d}\theta_i$ by $\frac{1}{2\pi}$), $\mathcal{H}(\gamma)$ takes values in $\mathbb{Z}^n$ for every cycle γ (*i.e.* the elements of $\mathcal{H}(\gamma)$ have integer values for cycles γ, each computing the winding number about a connected component of an obstacle). Thus, in Figure 4(a), $\mathcal{H}(\gamma) = [0,1]^T$ and $\mathcal{H}(\gamma') = [0,2]^T$.

C. $\mathcal{H}$-augmented Graph

The idea behind construction of a graph, $\mathcal{G}$, for use in search algorithms such as Dijkstra's or A* [10], [20], for finding optimal trajectories for robots, is to sample points from the free space, X, call them *vertices*, and establish edges between *neighboring* vertices (Figure 4(b)). Paths in the graph are then curves in X on which the differential 1-form, ω, can be integrated, and $\mathcal{H}$-signature can be computed. By convention we assume a coefficient of 1 on every edge (1-simplices) on any path in the graph.

The $\mathcal{H}$-augmented graph [2], [3] essentially augments the information of the $\mathcal{H}$-signature of paths leading up to the vertices from a fixed base-point (the start point, q_s, for all practical purposes), so that the homology classes of paths can be 'tracked' during the execution of the search algorithm. More precisely, the $\mathcal{H}$-augmented graph, $\mathcal{G}_\mathcal{H}$, has its vertex set as points sampled from a *covering space* [18] of X (we will call this covering space $X_\mathcal{H}$), for which the covering map is $p : (q,h) \mapsto q$ for every point $q \in X$ and h the $\mathcal{H}$-signature of some curve in X joining q_s to q (see Figure 7(a-b)). Note that for a given q, the set of $\mathcal{H}$-signatures of paths connecting q_s to q (values that h can assume) is countable (one for each homology class), and that p indeed is a covering map.

Since search algorithms like A* and Dijkstras can find only unique paths connecting two vertices in a graph, paths in different homology classes connecting two given vertices, q_s and q_g cannot be found by running the algorithms in $\mathcal{G}$. However these paths are *lifted* to different paths with different end points in $\mathcal{G}_\mathcal{H}$ (Figure 7(b)). Thus the algorithm needs to find paths to different *goal* vertices, $(q_g, h), (q_g, h+n_1), (q_g, h+n_2), \cdots$ (where $n_i \in \mathbb{Z}^n$), each of which project to paths in $\mathcal{G}$ connecting q_s and q_g, but in different homology classes (say τ_1 and τ_2 in Figure 7(b)).

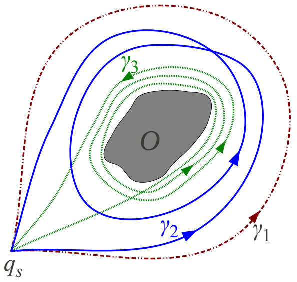
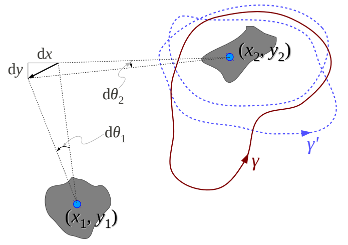


Fig. 3: A space with a single obstacle and cycles that loop around it multiple times. $[\gamma_1] \neq [\gamma_3] \neq [\gamma_2] \in H_1(X)$ (homology group with coefficients in $\mathbb{Z}$), but $[\gamma_1]_{\mathbb{Z}_2} = [\gamma_3]_{\mathbb{Z}_2} \neq [\gamma_2]_{\mathbb{Z}_2} \in H_1(X;\mathbb{Z}_2)$.



(a) Homology invariants can be computed by integrating $\frac{1}{2\pi}[\,\mathrm{d}\theta_1,\, \mathrm{d}\theta_2, \cdots,\, \mathrm{d}\theta_n]^T$ over cycles.

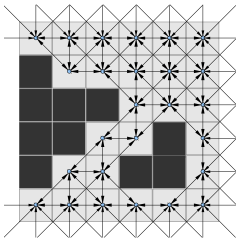

(b) Graph formed by uniform discretization of configuration space and connecting each vertex with its neighbors. Dark gray indicate obstacles.

Fig. 4

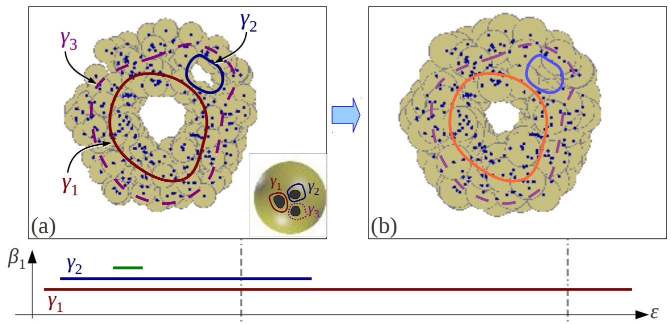


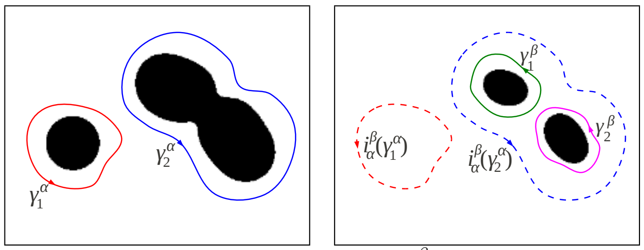
Fig. 5: Change in topology of the union of ϵ-balls (yellow disks) centered around data points (dark dots) as ϵ is increased. (a) and (b) show the union of balls with different values of ϵ. The barcode diagram for H_1 is shown below.

III. Persistent Homology

Persistent homology was pioneered for the topological analysis of point-cloud data [8], [12]. Given a collection of data points in a high dimensional space, one wants to infer an underlying subspace on (or near) which the data points lie, as well as its qualitative features. One can imagine growing *balls* of radius ϵ around each data point, and tracking the unions of all those balls as a 1-parameter family of spaces. Of course, the choice of the radius ϵ will dictate the topology (*cf.*, Figure 5(a)). Persistent homology considers the rank of the homology (H_1 in our setting) as a function of ϵ. This rank is called the *Betti number* β_1.

We use a similar idea for computing persistent features of the free space for robot path planning, when we are given an occupancy probability map $P : W \to [0,1]$. As we vary the probability threshold, ϵ, in computing U^ϵ, homology classes will appear and disappear. Consider different homology classes in $H_1(U^\alpha)$ (represented by cycles, for example, $\gamma_1^\alpha, \gamma_2^\alpha$, as shown in Figure 6(a)). For a $\beta > \alpha$ we have the inclusion map $i_\alpha^\beta : U^\alpha \hookrightarrow U^\beta$. Thus upon passing through the inclusion map, the cycles in U^α (γ_1^α and γ_2^α) are valid cycles in U^β (dashed curves in Figure 6(b)). However, as can be observed in the example of Figure 6, γ_1^α, after passing through the inclusion map, becomes trivial (a boundary) in U^β. Persistent homology groups are defined to capture this information:

Definition 1 (Persistent Homology [13], [29]). The $(\beta–\alpha)$-persistent first homology group of U^α is defined to be the set of homology



γ_1^α
γ_2^α
γ_1^β
γ_2^β
$i_\alpha^\beta(\gamma_1^\alpha)$
$i_\alpha^\beta(\gamma_2^\alpha)$

(a) γ_1^α and γ_2^α are two non-trivial cycles in U^α

(b) In U^β ($\supset U^\alpha$) the cycle γ_1^α becomes trivial, whereas γ_2^α '*splits*' such that its homology class, $[i_\alpha^\beta(\gamma_2^\alpha)] = [\gamma_1^\beta] + [\gamma_2^\beta] \in H_1(U^\beta)$.

Fig. 6: Illustration of how homology classes in U^α can either become trivial or 'split' upon inclusion in U^β for $\alpha < \beta$.

classes in $H_1(U^\alpha)$ that survive or *persist* into U^β. Formally it is defined as $H_1^{\alpha,\beta-\alpha} = Z_1(U^\alpha)/\left(B_1(U^\beta) \cap Z_1(U^\alpha)\right)$ — that is, we consider all cycles in U^α and then quotient out the ones that are *trivial* (*i.e.*, boundaries) in U^β (after passing through inclusion map).

Thus, in the example of Figure 6, $[\gamma_1^\alpha], [\gamma_2^\alpha] \in H_1(U^\alpha)$ are non-zero homology classes in U^α. But in the $(\beta-\alpha)$-persistent homology, $[\gamma_1^\alpha]^{\alpha,\beta-\alpha} = 0 \in H_1^{\alpha,\beta-\alpha}$ and $[\gamma_2^\alpha]^{\alpha,\beta-\alpha} \in H_1^{\alpha,\beta-\alpha}$ is non-zero — implying that the cycle γ_1^α does not 'survive' into U^β, but γ_2^α does.

Persistent homology is typically visualized in form of a diagram known as *barcode*, representing changes in Betti number, β_1 (bottom of Figure 5) – each horizontal bar represents a homology generator. The longest surviving (over the longest range of ϵ) generators indicate the most *persistent* topological feature of the space underlying the dataset. See [8], [12] for details. In its usual presentation, at a particular value of ϵ (say α), each bar above ϵ corresponds to a generator of $H_1(U^\alpha)$. In our case, however, since we are interested in finding the longest surviving homology classes of trajectories, each bar will correspond to a homology class.

IV. Theoretical and Algorithmic Tools

A. $\mathcal{H}2$-augmented Graph, $\mathcal{G}_{\mathcal{H}2}$

As described earlier, the merit of using $\mathbb{Z}_2$ coefficients is that if a cycle winds around an obstacle w number of times, then it is mapped to the same homology class as a cycle that winds around the particular obstacle ($w \mod 2$) times (assuming winding numbers about other obstacles are the same). This is useful in a search algorithm like A* or Dijkstra's when we want to avoid computation of trajectories that "loop around" obstacles multiple times, since such trajectories are highly suboptimal and mostly irrelevant in most robotics application (an issue addressed in [2] only informally).

In order to systematically attain this in graph search we need to alter the topology of the $\mathcal{H}$-augmented graph to reflect the fact that two trajectories (starting at q_s) leading up to q, and whose $\mathcal{H}$-signatures are $h + 2u$ and $h + 2w$ (for some $u, w \in \mathbb{Z}^n$) have the same goal vertex in the augmented graph (see Figure 7(b-c)). Thus the modification we need to make to the $\mathcal{H}$-augmented graph is that we set $(q, h) \equiv (q, h')$ (*i.e. identify* the vertices as same) whenever $h - h'$ is a vector of even integers (*i.e.*, we *glue* points in the covering space $X_\mathcal{H}$ which are at "alternate levels", and thus obtain a new covering space $X_{\mathcal{H}2}$). We call this the $\mathcal{H}2$-augmented graph, and the underlying covering space $X_{\mathcal{H}2}$ (the '2' here indicate that the *lifts* of paths starting at q_s have end points that are unique for every $\mathbb{Z}_2$-coefficient homology class of the path). The $\mathcal{H}$-signatures, mod 2, are called the $\mathcal{H}2$-signatures, and for a trajectory τ it is basically equal to $\mathcal{H}2(\tau) = (\mathcal{H}(\tau) \mod 2)$. (*Note:* $x \mod 2 = x - 2\lfloor \frac{x}{2} \rfloor$, $\forall x \in \mathbb{R}$, where $\lfloor \cdot \rfloor$ is the *floor*). The following proposition formally justifies the claim that this construction correctly computes the $\mathbb{Z}_2$-coefficient homology invariants:

Proposition 1. *Suppose γ_1 and γ_2 are closed loops with homology classes $[\gamma_1], [\gamma_2] \in H_1(X;\mathbb{Z})$. (These classes, as described in Section II-B, can be explicitly represented as the $\mathcal{H}$-signatures of γ_1 and γ_2 respectively.) Suppose $[\gamma_1]_{\mathbb{Z}_2}, [\gamma_2]_{\mathbb{Z}_2} \in H_1(X;\mathbb{Z}_2)$ are the $\mathbb{Z}_2$-coefficient homology classes of the same closed loops. Then $[\gamma_1]_{\mathbb{Z}_2} = [\gamma_2]_{\mathbb{Z}_2}$ if and only if $([\gamma_1] - [\gamma_2]) \mod 2 = 0$ (equivalently, $(\mathcal{H}(\gamma_1) - \mathcal{H}(\gamma_2)) \mod 2 = 0$, which, using the new notation, is equivalent to $\mathcal{H}2(\gamma_1) = \mathcal{H}2(\gamma_2)$).*

Proof. First recall that $H_1(X;\mathbb{Z}) \approx \mathbb{Z}^n$ is a $\mathbb{Z}$-module and $[\gamma_1]$ and $[\gamma_2]$ are elements in it. Thus, by definition, $[\gamma_1] \mod 2$, $[\gamma_2] \mod 2$, and $([\gamma_1] - [\gamma_2]) \mod 2$ are elements of $H_1(X;\mathbb{Z})/(2\mathbb{Z}\ H_1(X;\mathbb{Z}))$ (generally speaking, if M is a R-module, and $I \subseteq R$ is an ideal, then $IM = \{\alpha m \mid m \in M, \alpha \in I\}$ is a sub-module, and $M/IM = \{m + IM \mid m \in M\}$ is the quotient module).

A fundamental theorem [19] gives us the following isomorphism: $H_1(X;\mathbb{Z})/(2\mathbb{Z}\ H_1(X;\mathbb{Z})) \approx H_1(X;\mathbb{Z}) \otimes \mathbb{Z}_2$ (generally, $M/IM \approx M \otimes (R/I)$). Furthermore, since the topological spaces under consideration (subsets of $\mathbb{R}^2$) are *torsion free*, by the *Universal Coefficient Theorem* [18] we have $H_1(X;\mathbb{Z}) \otimes \mathbb{Z}_2 \approx H_1(X;\mathbb{Z}_2)$. Thus we have the isomorphism $H_1(X;\mathbb{Z})/(2\mathbb{Z}\ H_1(X;\mathbb{Z})) \underset{\approx}{\xrightarrow{\rho}} H_1(X;\mathbb{Z}_2)$. It is easy to check that this isomorphism can be explicitly expressed in terms of cycles, γ, as $\rho : ([\gamma] \mod 2) \mapsto [\gamma]_{\mathbb{Z}_2}$. Thus it follows that only $0 \in H_1(X;\mathbb{Z})/(2\mathbb{Z}\ H_1(X;\mathbb{Z}))$ maps to $0 \in H_1(X;\mathbb{Z}_2)$ under the map ρ, thus proving our claim. □

The explicit construction of $\mathcal{H}2$-augmented graph from $\mathcal{G}$, like that of $\mathcal{H}$-augmented graph [2], is described as follows: We assume that we are given a set of points sampled from the original free space, X, which constitute the vertex set, $\mathcal{V}_\mathcal{G}$. A set of unordered pairs of vertices that are 'neighbors' to each other constitute the edge set, $\mathcal{E}_\mathcal{G}$. This defines a discrete graph representation of the free space X (Figure 4(b)), and we write it as $\mathcal{G} = (\mathcal{V}_\mathcal{G}, \mathcal{E}_\mathcal{G})$. We furthermore assume that the chosen base-point, q_s, is in $\mathcal{V}_\mathcal{G}$. From such a graph we can describe a $\mathcal{H}2$-augmented graph, $\mathcal{G}_{\mathcal{H}2} = (\mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}, \mathcal{E}_{\mathcal{G}_{\mathcal{H}2}})$ as follows:

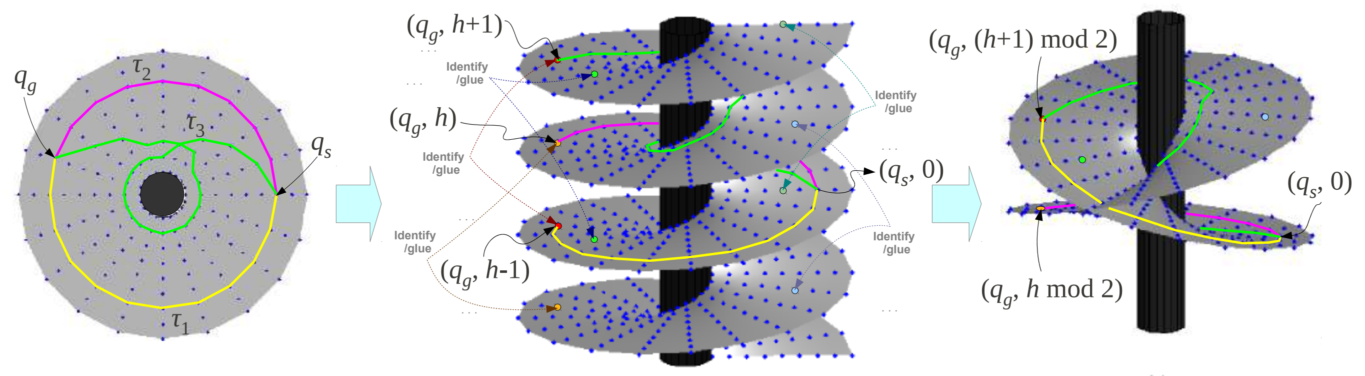
1) *General description:* Vertices in $\mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}$ are pairs of the form (q, h), where $q \in \mathcal{V}_\mathcal{G}$ and $h \in \oplus_{i=1}^n \mathbb{R}/2\mathbb{Z}$ (note that unlike the $\mathcal{H}$-augmented graph, h does not take value in $\mathbb{R}^n$).
2) *Base vertex:* $(q_s, \mathbf{0})$ is a vertex in $\mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}$.
3) *Recursive construction:* For a vertex $v = (q, h) \in \mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}$ and edge $[q, q'] \in \mathcal{E}_\mathcal{G}$, there is a vertex $v' = \left(q',\ (h + \mathcal{H}([q, q'])) \mod 2\right) \in \mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}$ and an edge $[v, v'] \in \mathcal{E}_{\mathcal{G}_{\mathcal{H}2}}$. [By 'mod 2' of a vector we mean the element-wise modulo operation.]

The cost of edges in the $\mathcal{H}2$-augmented graph are chosen to be the same as their projected counterparts on to $\mathcal{G}$. That is, $C_{\mathcal{G}_{\mathcal{H}2}}([(q, h), (q', h')]) = C_\mathcal{G}([q, q'])$ (where $C_G : \mathcal{E}_G \to \mathbb{R}_+$ is the cost function for graph G).

The consequence of constructing the $\mathcal{H}2$-augmented graph are the following:

- $\mathcal{H}2$-augmented graph, unlike the $\mathcal{H}$-augmented graph, is finite and bounded if $\mathcal{G}$ is finite.
- The paths obtained by searching in the $\mathcal{H}2$-augmented graph using an optimal search algorithm like A* or Dijkstra's will return only a finite number of paths – one for each homology class (the optimal one in the class) with coefficients in $\mathbb{Z}_2$.
- Optimal path connecting $(q_s, \mathbf{0})$ to $(q_g, h \mod 2)$ in $\mathcal{G}_{\mathcal{H}2}$, when projected back to $\mathcal{G}$, give optimal path in $\mathcal{G}$ restricted to homology class corresponding to $\mathcal{H}2$-signature $(h \bmod 2)$.

U^ϵ-specific notations: It is obvious that the function $\mathcal{H}$, and hence $\mathcal{H}2$, depend on the particular space, X, and hence the choice of the *representative points* (x_i, y_i) in the obstacle set, $O = W - X$. When we compute the $\mathcal{H}2$-signature for a path in U^ϵ, in order to explicitly



(a) $\mathcal{V}_{\mathcal{G}}$ (b) $\mathcal{V}_{\mathcal{G}_{\mathcal{H}}}$ (c) $\mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}$

Fig. 7: Vertex sets (blue dots) of $\mathcal{G}$, $\mathcal{G}_{\mathcal{H}}$ and $\mathcal{G}_{\mathcal{H}2}$, and the spaces that they live on: X, $X_{\mathcal{H}}$ and $X_{\mathcal{H}2}$ (light gray surfaces): (a) The vertices of graph $\mathcal{G}$ formed by cylindrically-uniform sampling of points in a disk-shaped region of a plane, X, (light gray) with a single disk-shaped obstacle (black). Also shown 3 paths. (b) The vertex set of the corresponding $\mathcal{H}$-augmented graph, $\mathcal{G}_{\mathcal{H}}$, sampled from $X_{\mathcal{H}}$. Vertices in $\mathcal{G}_{\mathcal{H}}$ are of the form (q, h) for $q \in \mathcal{V}_{\mathcal{G}}$ and h the $\mathcal{H}$-signatures corresponding to homology classes of paths connecting q_s to q. Lifts of corresponding paths are shown. (c) The vertex set of the $\mathcal{H}2$-augmented graph, $\mathcal{G}_{\mathcal{H}2}$, sampled from $X_{\mathcal{H}2}$. This graph is obtained by *identifying* or *gluing* every vertex $(q, h) \in \mathcal{V}_{\mathcal{G}_{\mathcal{H}}}$ with vertices of the form $(q, h + 2u)$ $\forall u \in \mathbb{Z}$ (note that the image shows an immersion of $\mathcal{G}_{\mathcal{H}2}$ in $\mathbb{R}^3$). Lifts of corresponding paths are shown.

indicate that the space under consideration is U^ϵ, we will write $\mathcal{H}2^\epsilon$ as the function that computes the $\mathcal{H}2$-signature of trajectories in U^ϵ (with representative points on $O^\epsilon := W - U^\epsilon$), and call it $\mathcal{H}2^\epsilon$-signature. Likewise, we write $\mathcal{G}^\epsilon_{\mathcal{H}2} = (\mathcal{V}^\epsilon_{\mathcal{G}_{\mathcal{H}2}}, \mathcal{E}^\epsilon_{\mathcal{G}_{\mathcal{H}2}})$ to denote the $\mathcal{H}2$-augmented graph obtained from discretization of U^ϵ.

B. Algorithm for Generating the Barcode

As discussed earlier, in the standard representation of a barcode diagram, at a particular value of ϵ each bar corresponds to a generator (representative cycles of elements of a generating set) of the homology group of U^ϵ. However, in our presentation, each bar will correspond to a homology class of trajectories, thus the number of bars will be equal to the number of homology classes (with $\mathbb{Z}_2$ coefficients) in U^α. Thus in Figure 9, at $\epsilon = 0.2$, we observe that there are 4 trajectories in different homology classes. In the barcode diagram shown below it, at the mark of $\epsilon = 0.2$, one can observe 4 bars corresponding to these classes. Likewise for $\epsilon = 0.55$ and $\epsilon = 0.9$. The colors of the bars in the barcode diagram correspond to the colors of the trajectories in the figures above.

1) Computations at each ϵ: In practice we start at $\epsilon = 1.0$ and decrease the value of ϵ at regular intervals of δ (the reason for not going the other way starting from 0.0 will become clear later), and for every value of ϵ we perform the following operations (using OpenCV [6] and YAGSBPL [1] libraries):

i. Compute U^ϵ (as a binary image, which is naturally represented as a graph – see Figure 4(b)), and thus have a representation for $\mathcal{G}^\epsilon$,
ii. Identify connected components of the obstacles (that are not connected to the boundary of the environment and is not enclosed in a disconnected patch of the free space isolated from q_s and q_g) — say, n^ϵ counts of them — and place a *representative point* inside each and call them $p^\epsilon_1, p^\epsilon_2, \cdots, p^\epsilon_{n^\epsilon}$, (for computing a representative point inside an obstacle we perform a scan along a single line parallel to the X axis passing through the mid-height of the obstacle, and choose a point on the line that lies inside the obstacle),
iii. Starting from $(q_s, \mathbf{0} \bmod 2)$, expand all vertices in the $\mathcal{H}2$-augmented graph, $\mathcal{G}^\epsilon_{\mathcal{H}2}$, using an optimal search algorithm (this uses the underlying graph, $\mathcal{G}^\epsilon$, which is obtained as a uniform square 8-connected discretization as in Figure 4(b)), and store paths to vertices of the form $(q_g, *)$ (where '$*$' are $\mathcal{H}2$-signatures of the respective found paths), and we do this until every vertex in $\mathcal{G}^\epsilon_{\mathcal{H}2}$ has been *expanded*.

Thus, at the end of each search, we have a set of m^ϵ trajectories and corresponding $\mathcal{H}2$-signatures: $\{\tau^\epsilon_i, h^\epsilon_i\}_{i=1,2,\cdots,m^\epsilon}$ (with $h^\epsilon_i \in (\mathbb{R}/2\mathbb{Z})^{n^\epsilon}$). This set contains all the homology classes of trajectories in U^ϵ connecting the given points.

This algorithm is straightforward. The more challenging part is to associate this data from ϵ to the data obtained for $\epsilon - \delta$.

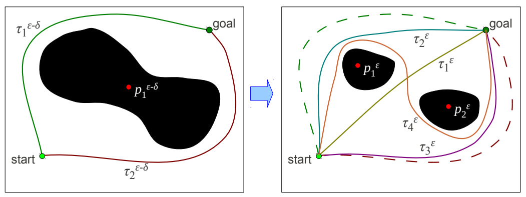
2) Relating Data Between ϵ and $\epsilon - \delta$: We first make a few simple observations about the change in topology as we go from $U^{\epsilon-\delta}$ to U^ϵ (*i.e.*, increase the parameter by a value of δ) – see Figure 8:

0. Obstacles can shrink in size, without changing topology of the free space or the obstacles, and hence not changing the number of homology classes of trajectories,
1. Obstacles can disappear, resulting in some of the homology classes of trajectories to *disappear* as well (this is illustrated in Figure 8(b) and can be observed in Figure 9 in the transition from $U^{0.85}$ to $U^{0.86}$).
2. Obstacles can *split*, resulting in potential *creation* of new homology classes of trajectories (this is illustrated in Figure 8(a) and can be observed in Figure 9 in transition from $U^{0.5}$ to $U^{0.51}$). The *splitting* can happen in a part of a single obstacle as well (or multiple obstacles simultaneously), due to which a previously disconnected region of the free space (and features contained inside it) can get connected to the free space containing q_s and q_g (illustrated in Figure 8(c)(ii)).
3. A disconnected component of the free region (that is isolated from the rest of the free space where start, goal and trajectories reside) can appear inside an obstacle (illustrated in Figure 8(c)(i)).

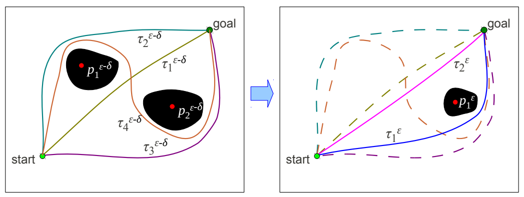
The justification behind the fact that only these three types of events ('1', '2' and '3') can create changes in global topology comes from Morse theory [21], with points 1, 2, and 3 corresponding to critical points with Morse index 2, 1, and 0 respectively. The assertions about the relation between the topology of the obstacles and that of the free space follow from Alexander duality [18], [3], as is obvious.

Suppose upon performing search in $\mathcal{G}^\epsilon_{\mathcal{H}2}$ we find m^ϵ counts of trajectory and $\mathcal{H}2$-signature pairs, $\{\tau^\epsilon_k, h^\epsilon_k\}_{k=1,2,\cdots,m^\epsilon}$. Likewise, suppose searching in $\mathcal{G}^{\epsilon-\delta}_{\mathcal{H}2}$ gives us $m^{\epsilon-\delta}$ counts of trajectories and their $\mathcal{H}2$-signatures, $\{\tau^{\epsilon-\delta}_j, h^{\epsilon-\delta}_j\}_{j=1,2,\cdots,m^{\epsilon-\delta}}$. The main objective of the following algorithm (algorithm *ComputeCorrespondences*) is to relate these two sets of data. We achieve this in two stages:

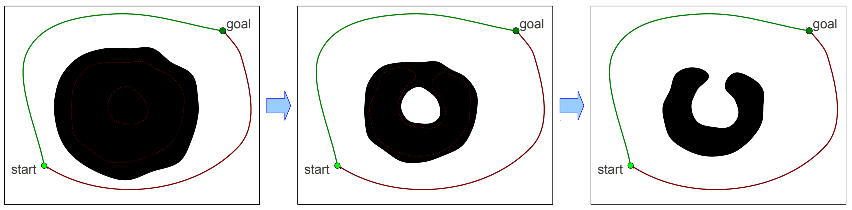
Stage I: Since $U^{\epsilon-\delta} \subset U^\epsilon$, a trajectory $\tau^{\epsilon-\delta}_j$ in $U^{\epsilon-\delta}$ is a valid trajectory in U^ϵ as well (formally, via the inclusion map $i^\epsilon_{\epsilon-\delta}$, which we will assume implicitly) – see Figure 8(a). This induces a *well-defined map* [18] between the homology classes



(a) An obstacle splits into multiple obstacles. Note the unambiguous correspondences: $\tau_1^{\epsilon-\delta} \mapsto \tau_2^{\epsilon}$ and $\tau_2^{\epsilon-\delta} \mapsto \tau_3^{\epsilon}$.


goal
$\tau_2^{\epsilon\text{-}\delta}$
$\tau_1^{\epsilon\text{-}\delta}$
$p_1^{\epsilon\text{-}\delta}$
$p_2^{\epsilon\text{-}\delta}$
$\tau_4^{\epsilon\text{-}\delta}$
$\tau_3^{\epsilon\text{-}\delta}$
start
goal
τ_2^{ϵ}
p_1^{ϵ}
τ_1^{ϵ}
start

(b) An obstacle disappears. Note the ambiguity in correspondences simply by topological consideration: $\tau_1^{\epsilon-\delta}, \tau_2^{\epsilon-\delta} \mapsto \tau_2^{\epsilon}$ and $\tau_3^{\epsilon-\delta}, \tau_4^{\epsilon-\delta} \mapsto \tau_1^{\epsilon}$.



(i) (ii)

(c) A new disconnected region of the free space is created, which eventually gets linked to the free region.

Fig. 8: Change in topology (as ϵ is increased).

of the trajectories: $\iota_{\epsilon-\delta}^{\epsilon} \;:\; H_1(U^{\epsilon-\delta}; \mathbb{Z}_2) \;\to\; H_1(U^{\epsilon}; \mathbb{Z}_2)$. So the obvious first step in computing the correspondences (Lines 4-12 of algorithm *ComputeCorrespondences*) is to compute the $\mathcal{H}2$-signatures of $\tau_j^{\epsilon-\delta}$, $\forall j \in \{1, 2, \cdots, m^{\epsilon-\delta}\}$ when they are viewed as trajectories in U^{ϵ} (*i.e.* compute $\mathcal{H}2^{\epsilon}(\tau_j^{\epsilon-\delta})$, $\forall j = 1, 2, \cdots, m^{\epsilon-\delta}$), and compare them with the $\mathcal{H}2$-signatures $\{h_k^{\epsilon}\}_{k=1,2,\cdots,m^{\epsilon}}$ (which, by computation, is an exhaustive set of $\mathcal{H}2$-signatures of trajectories in U^{ϵ} connecting the given points). Thus we establish a map $\overline{C}_I \;:\; \{1, 2, \cdots, m^{\epsilon-\delta}\} \;\to\; \{1, 2, \cdots, m^{\epsilon}\}$ defined as $j \;\mapsto\; k$ iff $\mathcal{H}2^{\epsilon}(\tau_j^{\epsilon-\delta}) = h_k^{\epsilon}$. Thus, corresponding to each homology class in $U^{\epsilon-\delta}$ we find a homology classes in U^{ϵ}.

Stage II: However the relationship obtained using the above process may map multiple different elements in $\{1, 2, \cdots, m^{\epsilon-\delta}\}$ to a same element in $\{1, 2, \cdots, m^{\epsilon}\}$ (this happens when, for example, an obstacle vanishes while going from $\epsilon - \delta$ to ϵ. See Figure 8(b)). We thus need to resolve this conflict in order to match the classes in $H_1(U^{\epsilon-\delta})$ with those in $H_1(U^{\epsilon})$ unambiguously.

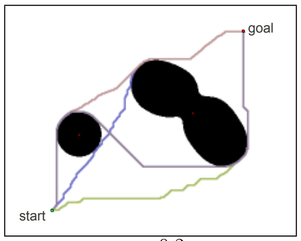
Note: In computing persistent homology for cycles in a filtered simplicial complex, the standard algorithms [13], [29] choose a preferred basis (as a set of cycles) that generate the homology group. This preferred basis is such that if two or more different elements from the basis chosen for $H_1(U^{\epsilon-\delta})$ maps to a single element of $H_1(U^{\epsilon})$ under the inclusion map (the conflict situation described above), then it is always the element $\mathbf{0} \in H_1(U^{\epsilon})$ that they map to. We first notice that this choice is intrinsically related to a metric (for example, in Figure 5(a), any choice of two cycles out of γ_1, γ_2 and γ_3 will be a valid basis for generating the first homology group of the space. The space is typologically a sphere with 3 punctures, as shown in the the inset figure, and no choice of a pair from $\{\gamma_1, \gamma_2, \gamma_3\}$ can be given preference over the others as far as topology is concerned). In particular, the choice of the basis depends on the order in which simplices are inserted into the complex. The consequence is that cycles in the preferred basis tend to be ones encircling *smaller* holes (which get *filled* first).
In our case, where we are concerned with finding persistent homology classes of trajectories, we make the following observations:

i. We desire to compute the persistence of all individual homology classes of trajectories (and not just the Betti numbers, as done in standard persistence computation for cycles). Thus choosing a basis for homology group is not sufficient – we wish to consider all elements in the set of homology classes (finitely many of them when coefficients are in $\mathbb{Z}_2$).
ii. There is no preferred '**0**' element among the homology classes of trajectories.
iii. It is most natural to use a metric for measuring distance between the paths for directly resolving the aforesaid conflict.

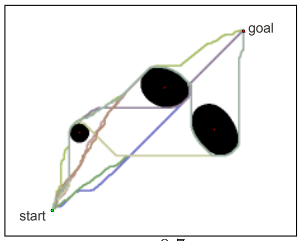
Thus, when the aforesaid relationship, $\overline{C}_I$, maps, say, $j_1, j_2, \cdots \in \{1, 2, \cdots, m^{\epsilon-\delta}\}$ to the same element, say, $k_0 \in \{1, 2, \cdots, m^{\epsilon}\}$, we resolve the conflict by comparing the *distances* between the trajectories (Lines 13-20 of algorithm *ComputeCorrespondences*).

Algorithm 1: Pseudocode for *ComputeCorrespondences*.

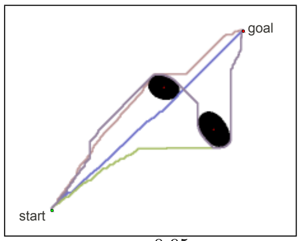
```
$\overline{C}_{II}$ = ComputeCorrespondences
        $(\{\tau_j^{\epsilon-\delta}\}_{j=1,2,\cdots,m^{\epsilon-\delta}},\ \{\tau_k^{\epsilon}, h_k^{\epsilon}\}_{k=1,2,\cdots,m^{\epsilon}},\ \mathcal{H}2^{\epsilon})$:
Inputs: i. Trajectories at $\epsilon - \delta$,
        ii. Trajectories and their $\mathcal{H}2^{\epsilon}$-signatures obtained in $\mathcal{G}_{\mathcal{H}2}^{\epsilon}$, and,
        iii. The function $\mathcal{H}2^{\epsilon}$ for computing $\mathcal{H}2$-signatures in $U^{\epsilon}$.
Output: Correspondences between the homology classes of trajectories at
        $\epsilon - \delta$ and those obtained at $\epsilon$.
        Expressed as $\overline{C}_{II} : \{1, 2, \cdots, m^{\epsilon-\delta}\} \to \{1, 2, \cdots, m^{\epsilon}\} \cup 0$.
1.  Compute $\overline{h}_i = \mathcal{H}2^{\epsilon}(\tau_i^{\epsilon-\delta}),\ \forall i = 1, 2, \cdots, m^{\epsilon-\delta}$.
2.  Initiate $\overline{C}_I$ := array of length $m^{\epsilon-\delta}$, $\overline{C}_{II}$ := array of length $m^{\epsilon-\delta}$.
3.  Initiate $\overline{c}$ := array of length $m^{\epsilon}$, $\overline{c}(i) = 0,\ \forall i \in \{1, 2, \cdots, m^{\epsilon}\}$.
    // First stage in computing correspondences. $\overline{C}_I$ can map multiple
    // elements in its domain to same element in $\{1, 2, \cdots, m^{\epsilon}\}$.
4.  for $j = 1, 2, \cdots, m^{\epsilon-\delta}$
5.      for $k = 1, 2, \cdots, m^{\epsilon}$
6.          if $\overline{h}_j == h_k^{\epsilon}$
7.              $\overline{C}_I(j) := k$
8.              $\overline{c}(k) + +$
9.              break for   // Break inner for loop, since there can't be
                            // another $k'$ such that $h_{k'}^{\epsilon} = h_k^{\epsilon} = \overline{h}_j$.
10.         end if
11.     end for
12. end for
    // Second stage in computing correspondences. The only
    // element in its codomain that $\overline{C}$ can map to multiple times is '0'.
13. for $k = 1, 2, \cdots, m^{\epsilon}$
14.     if $\overline{c}(k) > 0$
15.         Let $J := \{j \mid \overline{C}_I(j) = k\}$   // All classes in $U^{\epsilon-\delta}$ that map to
                                                     // the $k^{th}$ class in $U^{\epsilon}$ under inclusion $i_{\epsilon-\delta}^{\epsilon}$
16.         $j^* := \mathrm{argmin}_{j \in J}\ d_{Hf}(\tau_j^{\epsilon-\delta}, \tau_k^{\epsilon})$
17.         Set $\overline{C}_{II}(j^*) := k$
18.         Set $\overline{C}_{II}(j) := 0,\ \forall j \neq j^*, j \in J$
19.     end if
20. end for
21. return $\overline{C}_{II}$
```



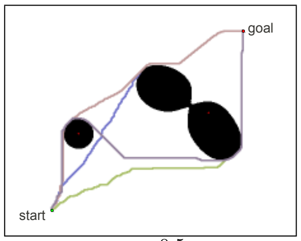
(a) $U^{0.2}$.



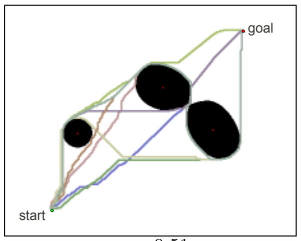
(b) $U^{0.7}$.



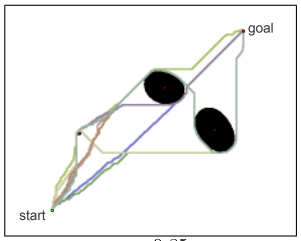
(c) $U^{0.95}$.



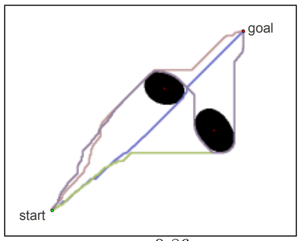
(d) $U^{0.5}$.



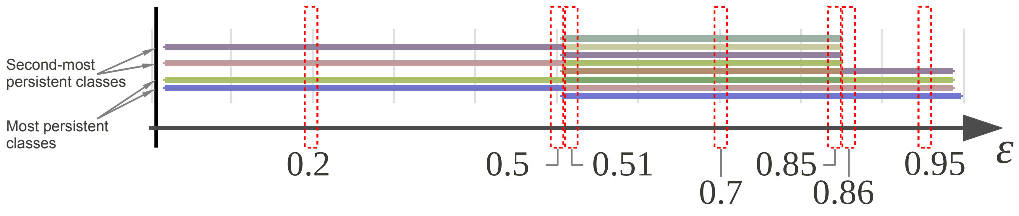
(e) $U^{0.51}$.



(f) $U^{0.85}$.



(g) $U^{0.86}$.



(h) Barcode diagram.

Fig. 9: (a), (d), (e), (b), (f), (g), (c): U^ϵ in increasing order of epsilon, and optimal trajectories in different homology classes. (h): Barcode diagram for homology classes of trajectories. Note that the colors of the bars at a particular value of ϵ correspond to the colors of the trajectories in the corresponding U^ϵ. In generating these sets of barcodes we used $\delta = 0.01$.

Out of $\tau_{j_1}^{\epsilon-\delta}, \tau_{j_2}^{\epsilon-\delta}, \cdots$, we choose the one that is *closest* to $\tau_{k_0}^{\epsilon}$ for establishing the correspondence, and all others are marked as *dead* (mapped to 0 in algorithm *ComputeCorrespondences*). Thus, in Figure 8(b), although the classes of both $\tau_1^{\epsilon-\delta}$ and $\tau_2^{\epsilon-\delta}$ would map to the class of τ_2^{ϵ} under the inclusion map (Stage I), since $\tau_1^{\epsilon-\delta}$ is *closer* to τ_2^{ϵ} the class of $\tau_1^{\epsilon-\delta}$ maps to the class of τ_2^{ϵ}, while the class of $\tau_2^{\epsilon-\delta}$ is simply declared *dead* in U^ϵ.

For comparing the distance between trajectories we use the Hausdorff metric [16] which can be used to measure the distance between arbitrary sets A and B in a metric space:

$$d_{Hf}(A,B) = \max\left\{ \sup_{a\in A} \inf_{b\in B} d_E(a,b), \quad \sup_{b\in B} \inf_{a\in A} d_E(a,b) \right\}$$

where d_E is the Euclidean distance on the workspace W.

Thus, *ComputeCorrespondences* computes the correspondences between the $m^{\epsilon-\delta}$ counts of homology classes of trajectories in $U^{\epsilon-\delta}$ and m^ϵ counts of homology classes of trajectories in U^ϵ. This lets us construct the barcode diagram incrementally.

C. *Heuristic Function taking Advantage of Previous Searches*

As noted earlier, at every ϵ we compute all the $\mathbb{Z}_2$-homology classes of trajectories by expanding all the vertices in $\mathcal{G}_{\mathcal{H}2}^{\epsilon}$. However, if we know the number of disconnected components of obstacles present in the active workspace (obstacles which are not connected to the boundary of W and are not enclosed in a disconnected component of the free space isolated from q_s and q_g), say n^ϵ, then we can predict that the number of homology classes of trajectories will be 2^{n^ϵ}. This can be seen as follows: Since there are n^ϵ representative points, there are n^ϵ components in a $\mathcal{H}2$-signature. Suppose $h = [(h_1 \bmod 2), (h_2 \bmod 2), \cdots, (h_{n^\epsilon} \bmod 2)]^T$ is the $\mathcal{H}2$-signatures of one of the trajectories connecting q_s to q_g. Then the set of possible $\mathcal{H}2$-signatures of trajectories connecting the same points will be $[((h_1+u_1) \bmod 2), ((h_2+u_2) \bmod 2), \cdots, ((h_{n^\epsilon}+u_{n^\epsilon}) \bmod 2)]^T$, for each $u_i \in \{0,1\}$. This is a total of 2^{n^ϵ} classes.

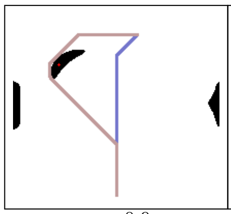
With this knowledge we can stop the graph search once all the 2^{n^ϵ} classes have been found during a single search, without having to expand all the vertices in the $\mathcal{H}2$-augmented graph. Thus it is useful to use A* search algorithm with an admissible heuristic function to speed the search (which would have been irrelevant if all the vertices in the graph had to be expanded). In particular, if we perform the searches in decreasing order of ϵ, then for the search in $\mathcal{G}_{\mathcal{H}2}^{\epsilon-\delta}$ it is possible to exploit the search result obtained at $\mathcal{G}_{\mathcal{H}2}^{\epsilon}$ to design an admissible heuristic function due to the following observation:

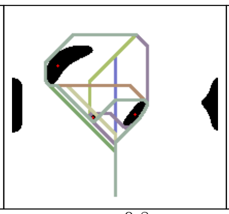
> The length of the shortest path connecting $(q_s, \mathbf{0})$ to a vertex $(q,h) \in \mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}^{\epsilon-\delta}$ in $\mathcal{G}_{\mathcal{H}2}^{\epsilon-\delta}$ is greater than or equal to the length of the shortest path connecting q_s to $q \in \mathcal{V}_{\mathcal{G}}^{\epsilon-\delta}$ in $\mathcal{G}^{\epsilon-\delta}$, which in turn is greater than or equal to the length of the shortest path connecting q_s to $q \in \mathcal{V}_{\mathcal{G}}^{\epsilon}$ in $\mathcal{G}^\epsilon$ (since $\mathcal{G}^{\epsilon-\delta}$ is a subgraph of $\mathcal{G}^\epsilon$ due to $U^{\epsilon-\delta} \subset U^\epsilon$). Also, note that the length of the shortest path connecting q_s to $q \in \mathcal{V}_{\mathcal{G}}^{\epsilon}$ is simply the minimum of the "g-scores" of vertices of the form $(q,*) \in \mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}^{\epsilon}$ when searching in $\mathcal{G}_{\mathcal{H}2}^{\epsilon}$ using A* algorithm starting from $(q_s, \mathbf{0})$. Let's call this value $\mathfrak{h}^\epsilon(q_s, q)$.

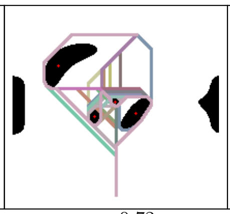
Thus $\mathfrak{h}^\epsilon(q_s, q)$ can be used as the heuristic for vertex $(q,h) \in \mathcal{V}_{\mathcal{G}_{\mathcal{H}2}}^{\epsilon-\delta}$ if the search is started from q_g (rather than q_s) when searching in $\mathcal{G}_{\mathcal{H}2}^{\epsilon-\delta}$ using A* algorithm (since heuristics function requires to return an underestimate of the least cost to the target vertex). Thus in the searches we alternate between starting the search from q_s and starting it from q_g. In the later cases the orientation of the trajectories obtained from the search and the signs of the $\mathcal{H}2$-signatures of the trajectories need to be flipped before calling *ComputeCorrespondences*.

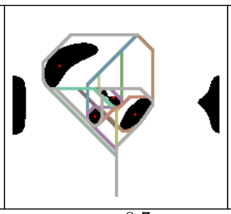
D. *Noisy Probability Distributions*

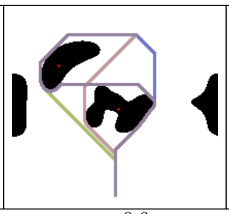
As described in the previous section, the number of $\mathbb{Z}_2$-coefficient homology classes grows exponentially with the number of *representative points*. If the probability distribution, P, is constructed

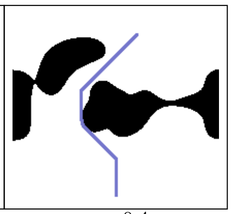

(a) $U^{0.9}$.

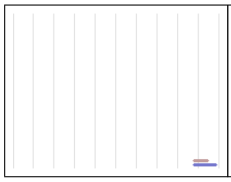

(b) $U^{0.8}$.

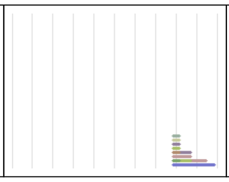

(c) $U^{0.72}$.

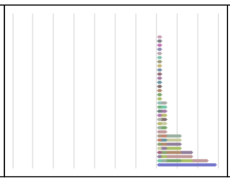

(d) $U^{0.7}$.

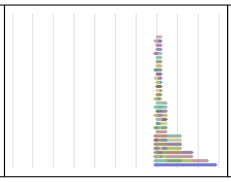

(e) $U^{0.6}$.

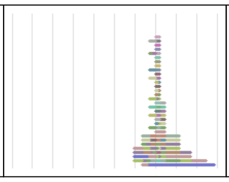

(f) $U^{0.4}$.

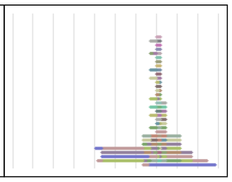

(g)

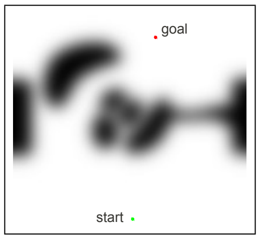

(h)


(i)

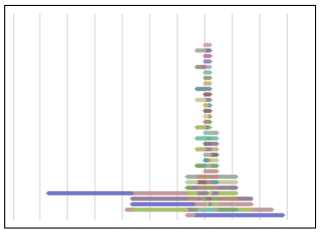

(j)


(k)


(l)



(m) Probability map.


(n) Final complete barcode. Note the longest bar (5th from the bottom).

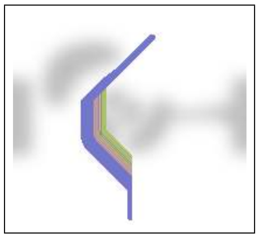

(o) Overlay of trajectories corresponding to the most persistent class (longest bar) obtained from the barcode.

Fig. 10: (a)-(f): U^ϵ for different values of ϵ in decreasing order. (g)-(l): The corresponding barcode diagram in the process of being constructed as ϵ is decreased. The final barcode is in (n). The decrement step in the value of ϵ used was $\delta = 0.01$. Probability map is shown in (m). The trajectories in the most persistent homology class shown in (o) clearly indicates the most reliable set of trajectories to follow.

from the readings of a noisy sensor, an ϵ-thresholded space, U^ϵ may contain many small irrelevant obstacles created only due to the presence of noise in the probability map. Placing a representative point on each of those only makes the computation complexity grow exponentially at certain small ranges of ϵ, where a large number of obstacles and homology classes may pop into existence and die shortly thereafter. This computation is mostly redundant because the short-lived homology classes do not contribute towards our main subject of interest – the most persistent homology classes. In order to handle this issue we use two techniques. In the following sub-sections we describe them and justify their computationally correctness.

1) Ignoring Obstacles Smaller than a Threshold Size: When computing trajectories in U^ϵ one can choose not to put *representative points* on components of obstacles that are smaller than a certain size (say, diameter smaller than a certain value) – $O^\epsilon_{small} \subseteq O^\epsilon$. Subsequent discretization and search in the $\mathcal{H}2$-augmented graph will only return feasible trajectories that have different $\mathcal{H}2$-signatures purely because of the presence of obstacles that are not small (Figure 11). Topologically, the operation performed on U^ϵ for computing the $\mathcal{H}2$-signatures is that of taking a small *tubular neighborhood* of every connected component of O^ϵ_{small}, and *identifying* each to a point. In the resulting *quotient spaces* [22] (which we will informally refer to as $U^\epsilon/\sim_{\mathcal{N}(O^\epsilon_{small})}$, or simply $U^\epsilon/\sim$) the small obstacles are essentially non-existent when making distinction between the topological classes of trajectories.

The size-based choice of the *small* obstacles guarantee that every obstacle that is small at a particular value of ϵ will remain small at an $\epsilon' < \epsilon$. This defines an inclusion map $\bar{\iota}^\beta_\alpha : U^\alpha/\sim_{\mathcal{N}(O^\alpha_{small})} \hookrightarrow U^\beta/\sim_{\mathcal{N}(O^\beta_{small})}$, which indices a well-defined map between the homology classes: $\bar{\iota}^\beta_\alpha : H_1(U^\alpha/\sim_{\mathcal{N}(O^\alpha_{small})};\mathbb{Z}_2) \rightarrow H_1(U^\beta/\sim_{\mathcal{N}(O^\beta_{small})};\mathbb{Z}_2)$ (just as the inclusion map i^β_α induced the map ι^β_α between the homology classes of the unquotiented spaces). This observation justifies the comparison of the homology classes of trajectories obtained in U^α with those obtained in U^β using the algorithm *ComputeCorrespondences*, even when we choose not to place representative points on the *small* obstacles.

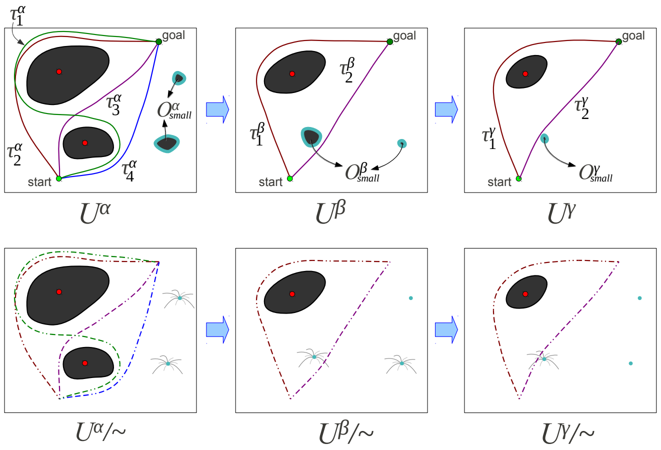


Fig. 11: Ignoring small obstacles ($\alpha < \beta < \gamma$). Top row: The computation of trajectories in different $\mathbb{Z}_2$-coefficient homology classes with *small obstacles* not being used in making distinction between the homology classes. Bottom row: The corresponding abstract topological spaces in which each connected component of the neighborhoods of small obstacles are identified to points. Note how the inclusion maps are well-defined: $U^\alpha/\sim \;\hookrightarrow\; U^\beta/\sim \;\hookrightarrow\; U^\gamma/\sim$. This lets us establish well-defined correspondence between the $\mathbb{Z}_2$-coefficient homology classes: $\tau_1^\alpha, \tau_2^\alpha \mapsto \tau_1^\beta$; $\tau_3^\alpha, \tau_4^\alpha \mapsto \tau_2^\beta$ and $\tau_1^\beta \mapsto \tau_1^\gamma$; $\tau_2^\beta \mapsto \tau_2^\gamma$.

2) Adaptively Change ϵ step: As described in the previous sections, in computing the bar diagram we start at a value of 1.0 for ϵ, and decrease it at each iteration by an amount δ, and establish correspondences between the classes obtained at the consecutive values of ϵ. However, given an upper bound, n_{max}, on the number of obstacles that we can computationally deal with, we can choose to skip a value of ϵ if the number of obstacles at that value is larger than that bound. For example, say we compute the homology classes in $U^{\epsilon'}$. Following that, suppose at the beginning of computation for $U^{\epsilon'-\delta}$ we find that the number of *non-small* obstacles, $n^{\epsilon'-\delta} > n_{max}$. We can then skip the search in $\mathcal{G}_{\mathcal{H}2}^{\epsilon'-\delta}$, and instead move on to $U^{\epsilon'-2\delta}$. If the number of non-small obstacles in there is within the bounds, we compute the homology classes in $U^{\epsilon'-2\delta}$ and compare them with ones obtained at ϵ' to establish correspondence (using the *ComputeCorrespondences* algorithm). In general we can establish correspondence between $U^{\epsilon'}$ and $U^{\epsilon'-s\delta}$ by skipping $(s-1)$ intermediate steps.

However, since this may result in comparison of homology classes between environments with large (metric) differences, we perform an additional check on the Hausdorff distance between a pair of corresponding trajectories determined by Line 16 of *ComputeCorrespondences* algorithm — we establish correspondence (*i.e.* set $\overline{C}_{II}(j^*) := k$ as in Line 17) only if $d_{Hf}(\tau_{j^*}^{\epsilon-s\delta}, \tau_k^{\epsilon}) < d_{thresh}$. Otherwise we set $\overline{C}_{II}(j^*) := 0$ — *i.e.* the j^{*th} class in $U^{\epsilon-s\delta}$ is marked as *dead*.

V. Results

We implemented the above algorithm in C++ programming language. All computations were performed on a dual core machine with processor clock speed of 2.6GHz and 4MB memory.

Figure 9 shows the results obtained for the probability map shown in Figure 1(a). The optimal trajectories generated at different value of ϵ, along with the entire barcode diagram are shown in figures Figure 9(a),(d),(e),(b),(f),(g),(c) (in order of the value of ϵ). At each ϵ the graphs $\mathcal{G}^{\epsilon}$ are created out of 400×320 uniform square discretization of the workspace. The entire computation (including thresholding, finding representative points and trajectory computations at all the 100 values of ϵ, as well as the generation of the barcode diagram, along with all the graphics output) took about 246s.

Figure 10 shows results in another probability map. In this case the workspace was 200×181 discretized and $\delta = 0.01$ was chosen. The total computation time was about 67s. The accompanying video shows the barcode in the process of being built as trajectories in different homology classes are obtained in decreasing order of ϵ.

Figure 12 shows a 400×320 discretized environment with two "blobs" of high probability of occupancy joined by a "bridge" of very low probability of occupancy. Note that the most persistent class (the bottom-most bar in the barcode diagram) dies at a low value of threshold due to the presence of the bridge, and two other classes survive at even lower values of ϵ. Note how the most persistent class does not survive at the lowest value of ϵ.

Figure 13 shows an example of an indoor environment where the probability map (of size 188×142 discretization units) contains significant amount of noise. Setting the *small obstacle* criteria at 5% of the maximum map dimension, $n_{max} = 7$ and d_{thresh} at 20% of the maximum map dimension, we computed the shown bar diagram in about 129s (including all steps as well as generation of the plots). The overlay of the trajectories in the most persistent classes are also shown. Note how, due to the noise, $U^{0.10}$ does not allow any feasible trajectory, and as ϵ increases, a large number of spurious classes show up due to the noise.

VI. Conclusion and Future Direction

In this paper we propose an approach to path planning in uncertain environments that is fundamentally different from existing approaches. We formulate the problem in terms of finding optimal trajectories in different $\mathbb{Z}_2$-coefficient homology classes, and compute the persistent homology classes of trajectories from a given occupancy probability map. Paths belonging to more persistent homology classes are more robust to uncertainty in the sense that they are less likely to require changes in homology classes. Our paper is the first to formulate path planning under uncertainty using a topological framework invoking the powerful tools of persistent homology.

A natural question is if this framework can be extended to integrate models of sensing and allowing us to go beyond simplistic static probability-based representations of occupancy. In this paper we restricted ourselves to the requirement that $U^{\alpha} \subset U^{\beta}$ whenever $\alpha < \beta$ so that we have the inclusion map $i_{\alpha}^{\beta} : U^{\alpha} \hookrightarrow U^{\beta}$. However, if such a relationship is not available between a sequence of spaces, one can always consider the inclusion maps $U^{\alpha} \hookrightarrow U^{\alpha} \cup U^{\beta} \hookleftarrow U^{\beta}$, and thus relate the homology classes between U^{α} and U^{β} via the homology classes in $U^{\alpha} \cup U^{\beta}$. This principle will compute a *zig-zag persistence* [7] for the sequence of spaces and can be used to pursue online re-planning as updates to the probability map become available from sensor data. Refinement of the proposed algorithm and its implementatuion for improved efficiency is also within the scope of future work.

References

[1] Subhrajit Bhattacharya. A template-based c++ library for large-scale graph search and planning, 2011. See http://subhrajit.net/index.php?WPage=yagsbpl.

[2] Subhrajit Bhattacharya, Maxim Likhachev, and Vijay Kumar. Topological constraints in search-based robot path planning. *Autonomous Robots*, pages 1–18, June 2012. DOI: 10.1007/s10514-012-9304-1.

[3] Subhrajit Bhattacharya, David Lipsky, Robert Ghrist, and Vijay Kumar. Invariants for homology classes with application to optimal search and planning problem in robotics. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 67(3-4):251–281, March 2013. DOI: 10.1007/s10472-013-9357-7.

[4] Subhrajit Bhattacharya, Nathan Michael, and Vijay Kumar. Distributed coverage and exploration in unknown non-convex environments. In *Proceedings of 10th International Symposium on Distributed Autonomous Robotics Systems*. Springer, 1-3 Nov 2010.

[5] R. Bott and L.W. Tu. *Differential Forms in Algebraic Topology*. Graduate texts in mathematics. Springer-Verlag, 1982.

[6] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly, Cambridge, MA, 2008.

[7] G. Carlsson and V. de Silva. Zigzag Persistence. *ArXiv e-prints*, nov 2008.

[8] Gunnar Carlsson. Topology and data. *Bull. Amer. Math. Soc.*, 46:255–308, 2009.

[9] Benjamin Cohen, Sachin Chitta, and Maxim Likhachev. Search-based planning for dual-arm manipulation with upright orientation constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT Press, 2nd edition, 2001.

[11] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[12] Herbert Edelsbrunner and John L. Harer. *Computational Topology*. American Mathematical Society, 2009.

[13] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. *Discrete and Computational Geometry*, 28(4):511–533, November 2002.

[14] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schultz, and B. Stewart. Distributed multirobot exploration and mapping. *Proc. of the IEEE*, 94(7):1325–1339, July 2006.

[15] Robert Ghrist. Barcodes: The persistent topology of data. *Bull. Amer. Math. Soc.*, 45:61–75, 2008.

[16] M. Gromov, J. Lafontaine, and P. Pansu. *Metric structures for Riemannian and non-Riemannian spaces*. Progress in mathematics. Birkhäuser, 1999.

[17] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science, and Cybernetics*, SSC-4(2):100–107, 1968.

[18] Allen Hatcher. *Algebraic Topology*. Cambridge Univ. Press, 2001.

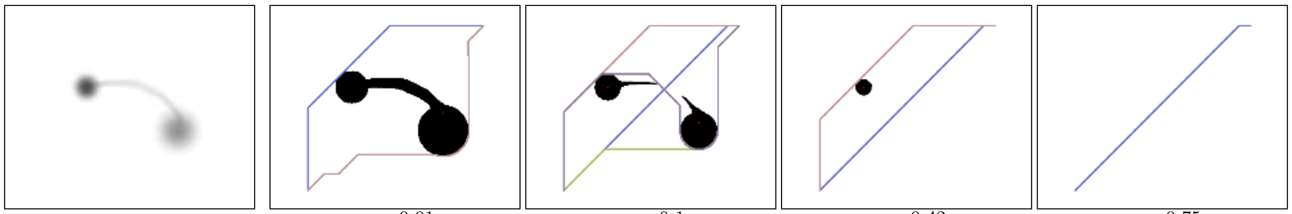
[19] M. Hazewinkel, N. Gubareni, and V.V. Kirichenko. *Algebras, Rings and Modules*. Number pt. 1 in Algebras, Rings and Modules. Springer, 2004.

[20] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.

[21] J.W. Milnor. *Morse Theory*. Annals of mathematics studies. Princeton University Press, 1963.

[22] James Munkres. *Topology*. Prentice Hall, 1999.

[23] C. Stachniss. *Exploration and Mapping with Mobile Robots*. PhD thesis, University of Freiburg, Freiburg, Germany, April 2006.



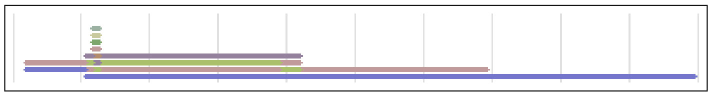

(f) Barcode diagram.

Fig. 12: An example in which the most persistent class (the bottom-most bar in the barcode diagram) dies at a low value of ϵ due to the presence of the bridge, although two other classes survive.

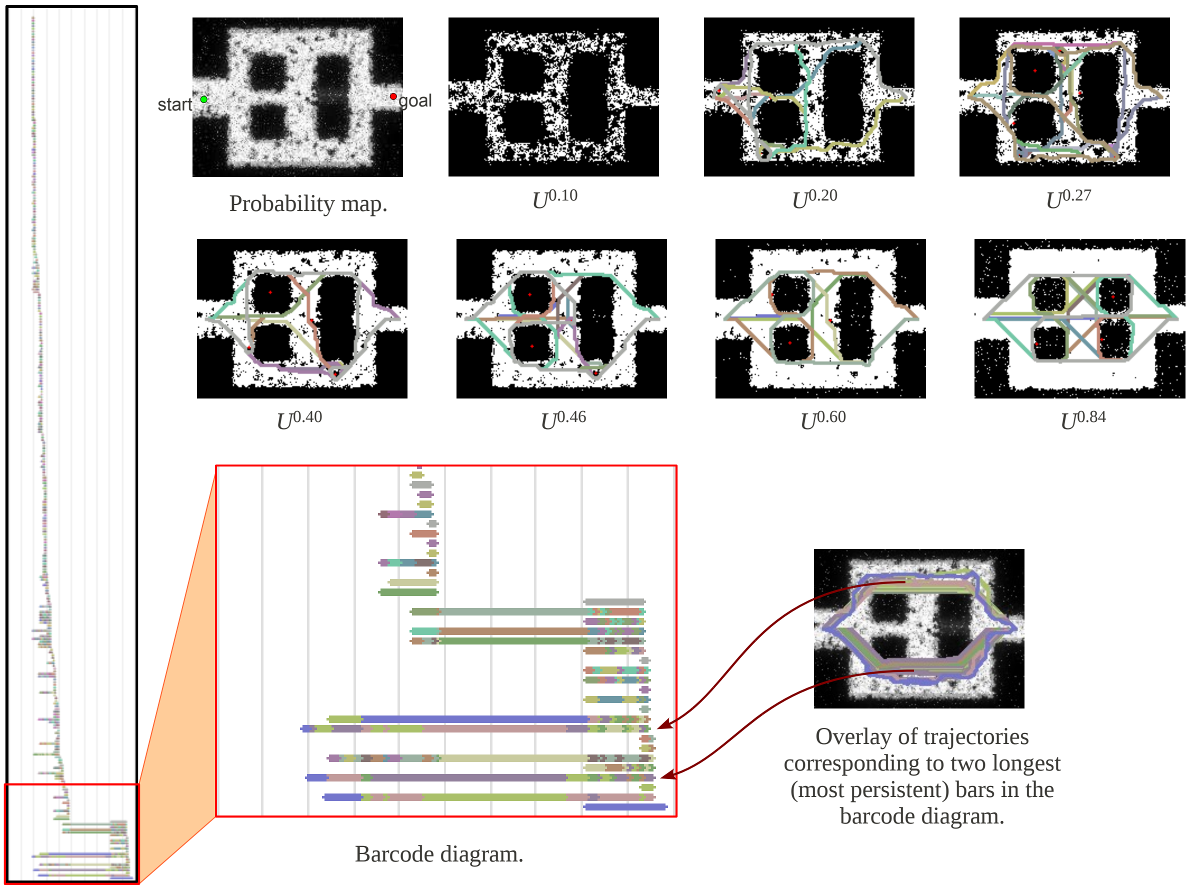


Fig. 13: An example with a noisy probability map. Observe the large number of short-lived classes created due to noise.

[24] C. Stachniss. *Robotic Mapping and Exploration*. Springer Tracts in Advanced Robotics. Springer, 2009.

[25] A. Stentz. The focussed D* algorithm for real-time replanning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1652–1659, 1995.

[26] P. Svestka and M. H. Overmars. Probabilistic path planning. Technical Report UU-CS-1995-22, Department of Information and Computing Sciences, Utrecht University, 1995.

[27] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[28] Paul Vernaza, Maxim Likhachev, Subhrajit Bhattacharya, Sachin Chitta, Aleksandr Kushleyev, and Daniel D. Lee. Search-based planning for a legged robot over rough terrain. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 2380–2387, 12-17 May 2009.

[29] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete Comput. Geom*, 33(2):249–274, February 2005.