Homology-Class Guided Rapidly-Exploring Random Tree For Belief Space Planning

Ran Hao, M. Cenk Çavuşoğlu

Abstract— **In this work, an efficient homology guided belief space planning method for obstacle-cluttered environments is presented. The proposed planner follows a two-step approach. First, a *h-signature* guided rapidly-exploring random tree (HRRT) algorithm is proposed to provide nominal trajectories in different homology classes by constructing homology aware sub-trees in a parallel manner. The HRRT planner is extended to a *h-signature* guided RRT* algorithm, where an inter-homology-class rewire procedure is proposed, increasing the probability of discovering homology classes in narrow space/passages. The iLQG-based belief space planning algorithm is then employed to find locally optimal trajectories minimizing uncertainties in each homology class.**

I. Introduction

In this paper, a homology- and uncertainty-aware trajectory planner, towards providing a global optimal solution for belief space planning problem by exploring different homology classes in an obstacle-cluttered environment is proposed. Given robot free configuration space $\mathcal{C}_f$, homotopy inequivalent trajectories are defined as the set of trajectories connecting the same start and goal configurations that cannot continuously deform into one another [1]. In an obstacle-scattered partially observable environment, the state of the robot can be described by the conditional probability distribution of the state [2], i.e., *belief*, and the planning algorithms are designed to generate optimal policies in belief space such that the motion and sensing uncertainties are minimized. In such environment, the awareness of homotopy classes of the trajectories can benefit the belief propagation and robot decision making from a global perspective, enhancing the safety and optimality of the planning performance. Current belief space planning methods like [3]–[6] produce locally optimized trajectories in belief space within the same homotopy class, where the trajectories generated are not homotopy aware. This paper presents a efficient belief planning strategy employing the awareness of the homotopy class, towards generating global optimal solutions to the partially observable Markov decision process problem. In practice, determining homotopy inequivalent trajectories is computationally intractable, in most literature, a computational efficient alternative, homology, is used for path planning with topological constraints [7]–[10]. In a 2D plane, two trajectories f_1 and f_2 connecting the same start and goal configurations are of the same homology type if and only if $f_1 \sqcup -f_2$ forms the complete boundary of a 2D manifold in $\mathcal{C}_f$ without containing any obstacle [1], [11].

The belief space planning method proposed follows a two step approach. First, a homology class guided rapidly-exploring random tree (HRRT) planner is presented in this paper. The HRRT planner finds homology inequivalent nominal trajectories using the homology class guided sub-trees which explore different homology classes in a parallel manner. This planner is then extended into a homology class guided RRT*(HRRT*) planner, in which a proposed inter-homology-class rewire procedure is to further improve the probability of discovering new homology classes in narrow passages/space. The iterative Linear Quadratic Gaussian (iLQG) based belief space planner [5] is then employed to locally optimize the nominal trajectories found by the HRRT/HRRT* planner in belief space.

The rest of this paper is organized as follows. Related work is presented in Section II. In Section III, the homology-guided nominal trajectory generation using the HRRT/HRRT* planner is presented. The iLQG based belief space planning is reviewed in Section IV. The experimental examples, including experimental comparison of the HRRT/HRRT* planner with other homology-aware planners from the literature are presented in Section V, followed by conclusions in Section VI.

II. Related Works

In the context of motion planning with homotopy class constraint, recent studies have focused on finding the optimal/shortest paths in different homotopy types using sample based or search based methods. In [7], [8], a *h-signature* augmented graph is constructed by discretizing the environment, and the shortest path is searched via graph search algorithms such as Dijkstra's or A*. The *h-signature* is proposed as a homotopy invariant and computed using Cauchy integral theorem. In [12], [13], the *h-signature* is defined as the "reduced words" which is constructed by tracking the direction and the times an inquiry trajectory intersects the "rays" extended from a point defined on each of the obstacles. In [14], a WA-RRT/RRT* planner is proposed by sampling the winding angle augmented states in standard the RRT/RRT* algorithm. In [9], a PI-RRHT* algorithm is proposed to provide optimal trajectories in different homology classes for stochastic dynamics, where sample states are augmented with *h-signatures*. In [11], the authors propose a Timed-Elastic-Band approach optimizing trajectories in different homology classes, where a *h-signature* based depth-first search over

This work was supported in part by the National Science Foundation under grants CISE IIS-1524363 and CISE IIS-1563805, and National Institutes of Health under grant R01 HL153034.

RH and MCC are with the Department of Electrical, Computer, and Systems Engineering, Case Western Reserve University, Cleveland, OH. They can be reached via email at rxh349@case.edu, and mcc14@case.edu respectively.

a exploration graph generated from a discretized Voronoi diagram or a waypoint sampling strategy, is proposed.

Belief space planning has been extensively studied in the past decade for solving the partially observable Markov decision process (POMDP). Platt et al. [3] propose a Linear quadratic regulation (LQR) approach in belief space where Gaussian belief state dynamics are employed. Van Den Berg et al. [5] propose an iLQG based belief space planner which computes a locally optimal solution to a POMDP problem with continuous state and action spaces. This method does not assume maximum likelihood observations. In [15], a rapidly-exploring random belief tree (RRBT) is proposed to incrementally construct a tree in belief space. The RRBT planner is able to explore different homotopy classes and provide global optimal trajectories, however, the exploration of the belief tree is not homotopy aware. In [16], a stochastic extended linear quadratic regulator (SELQR) is proposed and extended in belief space, which optimizes the approximation of the cost-to-come and cost-to-go iteratively. This method is able to find a locally optimized policy in different homotopy classes in belief space, the trajectory optimized still lacks homotopy awareness. These methods cannot serve the needs for certain robotic missions, e.g., exploring space in different homotopy classes safely for multi-agent systems.

The homology guided belief space planning method proposed in the present paper locally optimizes multiple homology aware trajectories in a parallel manner by employing the proposed HRRT/HRRT* planner towards providing global optimal trajectories in belief space. The present HRRT/HRRT* planner is different from the WA-RRT/RRT* planner [14] in that our planner explores multiple homology classes using sub-tree expansions in a parallel manner. The PI-RRHT* planner [9] provides homology guided optimal trajectories for stochastic dynamics, however, this planner does not take into account the sensing uncertainty. The proposed homology guided belief space planner optimizes trajectories over both motion and sensing uncertainties in belief space. Additionally, the nominal trajectory generation procedure is different from the PI-RRHT* planner [9]. First, we present both feasible and optimal homology class exploration strategies for the nominal trajectory generation, while in [9], only optimal homology class exploration strategy is proposed. Second, the "rewire" procedure proposed in the HRRT* planner is different from that in the PI-RRHT* planner, as the HRRT* planner includes an extra inter-homology-class rewire procedure. Using this procedure, the present HRRT* planner is able to discover new homology classes and increase the connectivity of the sample nodes in narrow passages. The performance of the present HRRT* planner in comparison to the PI-RRHT* planner [9] and WA-RRT* planner [14] in different experimental scenarios is presented in Section V.

III. Homology-Guided Nominal Trajectory Generation

In this section, the *h-signature* guided RRT and RRT* algorithms are proposed. The *h-signature* is defined as a topological invariant for the trajectories within the same homology class [8], [12], [17], i.e., the *h-signature* uniquely identifies the homology class of a given trajectory. The algorithms proposed aim to find nominal homology inequivalent trajectories parallelly through the expansion of the *h-signature* augmented sub-trees. In this paper, the *h-signature* is computed in 2D space, as higher dimensional cases can be computed via 2D projections as proposed in [14].

A. h-signatures and Winding Numbers in 2D

Let the free configuration space $\mathcal{C}_f$ and obstacles $\mathbb{O} = \{\mathbb{O}_1, \mathbb{O}_2, ..., \mathbb{O}_M\}$ be subsets of $\mathbb{R}^2$. Given a continuous curve f, the winding number $w(p_i, f),\ i = 1,2,...,M$ is used to describe the number of times the curve winds around a winding center $p_i \in \mathbb{O}_i$ [14], [18], where the winding center p_i can be defined at a point on the obstacle $\mathbb{O}_i$ or can be found through filtration of simplicial complexes [14]. Given a winding center p_i, the winding number $w(p_i, f)$ of a continuous trajectory $f(t) = (x(t), y(t)),\ t \in [0,1]$ in 2D plane can be computed as [18]:

$$w(p_i, f) = \frac{1}{2\pi} \int_0^1 \frac{\dot{y}(t)x(t) - y(t)\dot{x}(t)}{x(t)^2 + y(t)^2} dt. \tag{1}$$

In sample based motion planning algorithms, e.g. RRT/RRT*, trajectories are constructed through connecting a series of piecewise linear curves $\{f_j\}_{j=1,...,N-1}$. Let $P_0,\ P_1, ...,\ P_N$ be a series of intersecting points connecting the piecewise linear curves, and $v_0,\ v_1, ...,\ v_N$ be the series of vectors where $v_k = P_k - p_i,\ k = 1, ..., N$, the winding number can then be approximated as the sum of the signed angles of the piecewise linear curves revolve against the winding center [18]:

$$\begin{aligned} w(p_i, f) &= \frac{1}{2\pi} \sum_{k=1}^{N-1} \int_0^1 \frac{\dot{y}_k(t)x_k(t) - y_k(t)\dot{x}_k(t)}{x_k(t)^2 + y_k(t)^2} dt \\ &= \frac{1}{2\pi} \sum_{k=1}^{N-1} \cos^{-1} \frac{v_k \cdot v_{k+1}}{||v_k|| \, ||v_{k+1}||} \cdot \mathrm{sign} \begin{vmatrix} v_k^x & v_{k+1}^x \\ v_k^y & v_{k+1}^y \end{vmatrix}. \end{aligned} \tag{2}$$

By definition, if the winding center resides outside a closed curve, the winding number is zero. Otherwise, if the winding center resides inside the closed curve, the winding number is either 1 if the curve travels counterclockwise, or -1 if the curve travels clockwise [18]. The winding number can be used to identify trajectories that are homology inequivalent, as it provides the orientation of how the curve travels around the winding center (the obstacle). This heuristic is then applied to the design of the *h-signature*.

In this paper, we compute the *h-signature* as the set of integers computed from the winding numbers. Specifically, given winding numbers $w = [w(p_1, f),\ w(p_2, f),\ ...,\ w(p_M, f)] \in \mathbb{R}^M$, the *h-signature* is computed as:

$$h = [sgn'(w(p_1, f)), sgn'(w(p_2, f)), ..., sgn'(w(p_M, f))], \tag{3}$$

Algorithm 1: *h-signature* Guided RRT (HRRT) Algorithm

```
Input : $q_0$: the root configuration, $N$: the number
        of iterations to expand the tree,
        $\{p_j\}_{j=1,2,...,M}$: a series of winding centers
1  $V_1 \leftarrow (q_0, \mathbf{0}^M)$, $E_1 \leftarrow \emptyset$, $T$.Append($(V_1, E_1)$)
2  $H$.Append($\mathbf{0}^M$), $N_h \leftarrow 1$
3  for $i = 1, 2, ..., N$ do
4      $q_{rand} \leftarrow$ RandomSample($\mathcal{C}_f$)
5      for all $i_h = 1, 2, ..., N_h$ do
6          $x_{nearest} \leftarrow$ NearestNeighbor($T_{i_h}$, $q_{rand}$)
7          $q_{new} \leftarrow$ Steer($x_{nearest}.q$, $q_{rand}$)
8          if ObstacleFree($x_{nearest}.q$, $q_{new}$) then
9              $w \leftarrow x_{nearest}.w$ +
                ComputeWindingNumber($x_{nearest}.q$, $q_{new}$,
                $\{p_j\}_{j=1,2...,M}$)
10             $h^* \leftarrow sgn'(w)$
11             $T$, $i_{h^*}$, $H$, $N_h \leftarrow$ SubTreeExtension($i_h$, $T$,
                $h^*$, $x_{nearest}$, $H$, $N_h$)
12             $x_{new} \leftarrow (q_{new}, w)$, $T_{i_{h^*}}.V \leftarrow T_{i_{h^*}}.V \cup x_{new}$,
13             $T_{i_{h^*}}.E \leftarrow T_{i_{h^*}}.E \cup (x_{nearest}, x_{new})$
14         end
15     end
16 end
Output: Tree $T = \{(V_{i_h}, E_{i_h})\}_{i_h=1,2,...,N_h}$
```

where a modified *signum* function sgn' is defined as:

$$sgn'(x) = \begin{cases} -(m-1) & \text{if } -(m-1) < x < -(m-2) \\ \vdots & \vdots \\ -1 & \text{if } -1 < x < 0 \\ 0 & \text{if } x = 0. \\ 1 & \text{if } 0 < x < 1 \\ \vdots & \vdots \\ m-1 & \text{if } m-2 < x < m-1 \end{cases} \tag{4}$$

This definition modulates the number of turns the trajectories wind around the winding centers to m, i.e., the *h-signature* computed with modulo of m can only distinguish the trajectories winding up to $m-1$ times around the winding centers, similar to the winding number modulo m defined in [14]. We will use the notation $h = sgn'(w)$ for convenience in the rest of the paper.

The definition of the *h-signature* presented in this section is a numerical version of the definition proposed in [12], [17], instead of constructing the "words" by tracking how the trajectories intersects the "rays" extended from a point on the obstacles, the number of turns and the directions of the trajectory winds around the winding centers are used as "intersecting" criteria.

B. h-signature Guided RRT Algorithm

In this section, the proposed *h-signature* guided Rapidly-Exploring Random Tree [19] (HRRT) algorithm generating

Algorithm 2: SubTreeExtension

```
Input : $H$, $i_h$, $T$, $h^*$, $x_{nearest}$, $N_h$
1  if $h^* \notin H$ then
2      $H \leftarrow H$.Append($h^*$)
3      $i_{h^*} \leftarrow N_h + 1$, $N_h \leftarrow N_h + 1$
4      $T_{i_{h^*}} \leftarrow$ NewSubTree($T$, $i_{h^*}$, $x_{nearest}$)
5  else if $h^* \neq H(i_h)$ then
6      $i_{h^*} \leftarrow$ Find($h^*$, $H$)
7      $T_{i_{h^*}} \leftarrow$ ConnectSubTree($T$, $i_{h^*}$, $x_{nearest}$)
8  else
9      $i_{h^*} \leftarrow i_h$
10 end
Output: $H$, $T$, $N_h$, $i_{h^*}$
```

feasible trajectories in different homology classes in a parallel manner is presented. The algorithm incrementally grows a tree by randomly sampling states in the configurations space, as in the standard RRT algorithm [20]. Two adaptations are made to construct a homology-aware tree. First, the vertices and edges of the tree are augmented by winding numbers [14]. Second, the tree is constructed by growing the *h-signature* augmented sub-trees, where each sub-tree explores a different homology class. During the random tree expansion, new sub-trees will be constructed based on existing sub-trees as new homology classes are discovered.

The HRRT algorithm is presented in Algorithm 1. The tree T is represented by a set of vertices V and edges E [20]. Similar to [14], the set of vertices V and edges E are augmented by the winding numbers and projected to the winding number augmented space $\mathcal{W} \subset \mathbb{R}^M$, with $\mathcal{V} \subseteq \mathcal{C}_f \times \mathcal{W}$ and $E \subseteq \mathcal{V} \times \mathcal{V}$. Each vertex $x = (q, w) \in \mathcal{V}$ has a state $q \in \mathcal{C}_f$ and a winding number vector $w \in \mathcal{W}$. The algorithm takes as input a root configuration q_0, the number of iterations N for tree expansion, and the winding centers $\{p_j\}_{j=1,2,...,M}$ in the $2D$ plane or the projected 2D plane for higher dimensional cases.

The tree T is initialized with the first *h-signature* guided sub-tree (V_1, E_1) in Line 1, where the vertex has the state of the root node with the *h-signature* of a zero vector $\mathbf{0}^M$. In Line 2, an array H collecting *h-signatures* during the expansion of the sub-trees is also initialized with the *h-signature* of the root node, and the size N_h of the *h-signature* array is initialized as 1. The main expansion of the HRRT is presented in Lines 3-16. In Line 4, a random state q_{rand} is first sampled in configuration space $\mathcal{C}_f$. The algorithm then extends all the sub-trees towards the random state q_{rand}. For the i_h-th sub-tree $T_{i_h} = (V_{i_h}, E_{i_h})$, in Line 6, a nearest neighbor search is first performed and returns the nearest neighbor $x_{nearest}$ in the sub-tree. A new state q_{new} is returned by *Steer()* function in Line 7. Line 8 checks if the edge between q_{new} and $x_{nearest}.q$ is obstacle free, if true, the algorithm computes the winding number vector $w \in \mathcal{W}$ given the line segment from $x_{nearest}.q$ to q_{new} using (2) in function *ComputeWindingNumber()*, and the *h-signature* h^* using (3), in Lines 9 and 10. In Line 11, a *SubTreeExtension()* function is performed, as presented in Algorithm 2, where three types of procedures are performed given the *h-signature*

h^* computed in Line 10. The function takes the input of the index i_h of the sub-tree T_{i_h}, the nearest vertex $x_{nearest}$, the current tree T, the *h-signature* h^*, and the *h-signature* array H and its size N_h. First, if the *h-signature* h^* is not found in the existing homology classes, a new homology class is discovered, where the new *h-signature* h^* is appended to H and a new sub-tree $T_{i_{h^*}} = (V_{i_{h^*}}, E_{i_{h^*}})$ is initialized, as shown in Lines 2-4. In Line 4, the *NewSubtree()* function adds and connects the vertex $x_{nearest}$ and all its ancestry vertices (via parent tracing) to the new sub-tree. If h^* is found but does not match the *h-signature* $H(i_h)$ of the nearest vertex $x_{nearest}$, the i_{h^*}-th sub-tree will add and connect the vertex $x_{nearest}$ and its ancestry vertices until a vertex belongs to the i_{h^*}-th sub-tree or the root vertex is found, as in Lines 6-7. Specifically, the *Find()* function returns the index i_{h^*} of the sub-tree $T_{i_{h^*}}$ whose signature matches h^*, and the *ConnectSubtree()* function connects the vertex $x_{nearest}$ and all its ancestry vertices (via parent tracing) to the i_{h^*}-th sub-tree. Otherwise, h^* matches the *h-signature* of the nearest vertex $x_{nearest}$, and i_{h^*} is updated to the index of the i_h-th sub-tree. This function returns the index i_{h^*} of the sub-tree for extension of the new vertex, the updated tree and the *h-signature* array. Finally, in Lines 12-13, the new winding number augmented vertex x_{new} is added to its designated sub-tree $T_{i_{h^*}}$. In each expansion iteration, the algorithm expands all sub-trees $(V_{i_h}, E_{i_h})_{i_h=1,...N_h}$ in a parallel manner, new homology classes and sub-trees are discovered through random sampling in the free configuration space as in standard RRT algorithm.

C. h-signature Guided RRT* Algorithm

The HRRT algorithm can then be extended into *h-signature* guided RRT* (HRRT*) algorithm by adding a rewire procedure every time a new vertex is added to the sub-trees, as presented in Algorithm 3, in a manner similar to the original RRT* algorithm proposed in [21].

The HRRT* algorithm is adapted from the *Extend* procedure of the original RRT* algorithm (Algorithm 4 in [21]) by taking into account *h-signatures* during tree extension and rewiring. Specifically, the algorithm first generates a new state in free configuration space and computes the *h-signature* h^* given the nearest neighbor state, as given in Lines 4-10. The $Near()$ function used in Line 11 returns the neighbor vertices Q_{near} with the same *h-signature* (h^*) in the sub-tree within a ball of radius $\varepsilon(|V_h|)$. The *UpdateMinimumCostNeighbor()* function in Line 12 then takes the neighbor vertices Q_{near} and the new state q_{new} as input and returns the vertex x_{min} with the same *h-signature* h^* that leads to a minimum cost trajectory to the new state q_{new}. (The readers are referred to Lines 6-12 of the *Extend* Algorithm in [21] for details of this process.) The winding number vector w is then updated given the line segment from the minimum cost state $x_{min}.q$ to q_{new}, and a new *h-signature* h^{**} is calculated in Lines 13-14. The designated sub-tree $T_{i_{h^{**}}}$ is then extended based on the *h-signature* computed as in Lines 15-16.

The *Rewire* procedure is performed in Line 17, in a manner similar to the standard rewire procedure from Algorithm 4

Algorithm 3: *h-signature* Guided RRT* Algorithm

```
   Input : $q_0$, $N$, $\{p_j\}_{j=1,2,...,M}$
1  $V_1 \leftarrow (q_0, \mathbf{0}^M)$, $E_1 \leftarrow \emptyset$, $T$.Append($(V_1, E_1)$)
2  $H$.Append($\mathbf{0}^M$), $N_h \leftarrow 1$
3  for $i = 1, 2, ..., N$ do
4      $q_{rand} \leftarrow$ RandomSample($\mathcal{C}_f$)
5      for all $i_h = 1, 2, ..., N_h$ do
6          $x_{nearest}.q \leftarrow$ NearestNeighbor($T_{i_h}$, $q_{rand}$)
7          $q_{new} \leftarrow$ Steer($x_{nearest}.q$, $q_{rand}$)
8          if ObstacleFree($x_{nearest}.q$, $q_{new}$) then
9              $w \leftarrow x_{nearest}.w +$
                 ComputeWindingNumber($x_{nearest}.q$, $q_{new}$,
                 $\{p_j\}_{j=1,2...,M}$)
10             $h^* \leftarrow sgn'(w)$
11             $Q_{near} \leftarrow$ Near($h^*, T_{i_h}, q_{new}$)
12             $x_{min} \leftarrow$
                 UpdateMinimumCostNeighbor($Q_{near}, q_{new}$)
13             $w \leftarrow x_{min}.w +$
                 ComputeWindingNumber($x_{min}.q$, $q_{new}$,
                 $\{p_j\}_{j=1,2...,M}$)
14             $h^{**} \leftarrow sgn'(w)$
15             $T$, $i_{h^{**}}$, $H$, $N_h \leftarrow$ SubTreeExtension($i_h$, $T$,
                 $h^{**}$, $x_{nearest}$, $H$, $N_h$)
16             $x_{new} \leftarrow (q_{new}, w)$, $T_{i_{h^{**}}}.V \leftarrow T_{i_{h^{**}}}.V \cup$
                 $x_{new}$, $T_{i_{h^{**}}}.E \leftarrow T_{i_{h^{**}}}.E \cup (x_{min}, x_{new})$
17             $T_{i_{h^{**}}} \leftarrow$ Rewire($T_{i_{h^{**}}}$, $h^{**}$, $x_{min}$, $x_{new}$,
                 $Q_{near}$)
18             $T$, $H$, $N_h \leftarrow$
                 InterHomologyClassRewire($T$, $h^{**}$, $i_{h^{**}}$,
                 $x_{new}$, $H$, $N_h$)
19         end
20     end
21 end
   Output: Tree $T = \{(V_{i_h}, E_{i_h})\}_{i_h=1,2,...,N_h}$
```

in [21]. One modification is that the *Rewire* procedure in HRRT* algorithm only rewires the neighbor vertices from Q_{near} that can be reached from x_{new} with smaller cost and maintains the same *h-signature* h^{**} after the rewire to x_{new}.

Additionally, an "InterHomologyClassRewire" procedure is performed at the end of each sub-tree extension, as presented in Algorithm 4. This procedure is able to increase the probability of discovering new homology classes by actively extending vertices that belong to different homology classes, as shown in the example scenario given in Fig. 1. In Line 1, the *InterHomologyNearestNeighbor()* function first searches the vertices in the neighborhood of x_{new} within the radius of $\varepsilon(|V_{h|})$ that carry a different *h-signature* from h^{**} (as computed in Line 14, Algorithm 3), and finds all vertices that can be accessed through x_{new} with a smaller cost. The vertex $x^{\dagger}_{min}$ that has the smallest cost being accessed through x_{new} is returned by the function. If such a vertex exists, the new winding numbers $w^{\dagger}$ and the *h-signature* $h^{\dagger}$ are then calculated, as in Lines 3-4. In Line 5, the *SubTreeExtension()* procedure is executed given the new *h-signature* $h^{\dagger}$, and returns the index $i_{h^{\dagger}}$ of the designated sub-tree for the new vertex extension. Finally, a new vertex $x^{\ddagger}_{min}$ is created given

Algorithm 4: InterHomologyClassRewire

```
    Input   : $T$, $h^{**}$, $i_{h^{**}}$, $x_{new}$, $H$, $N_h$
1   $x^{\dagger}_{min} \leftarrow$ InterHomologyNearestNeighbor($h^{**}, T, x_{new}$)
2   if $x^{\dagger}_{min}$ exists then
3       $w^{\dagger} \leftarrow x_{new}.w$ + ComputeWindingNumber($x_{new}.q$,
          $x^{\dagger}_{min}.q$, $\{p_j\}_{j=1,2...,M}$)
4       $h^{\dagger} \leftarrow sgn'(w^{\dagger})$
5       $T$, $i_{h^{\dagger}}$, $H$, $N_h$ $\leftarrow$ SubTreeExtension($i_{h^*}$, $T$, $h^{\dagger}$,
          $x_{new}$, $H$, $N_h$)
6       $x^{\ddagger}_{min} \leftarrow (x^{\dagger}_{min}.q,\ w^{\dagger})$
7       $T_{i_{h^{\dagger}}}.V \leftarrow T_{i_{h^{\dagger}}}.V \cup x^{\ddagger}_{min}$, $T_{i_{h^{\dagger}}}.E \leftarrow T_{i_{h^{\dagger}}}.E \cup (x^{\ddagger}_{min},$
          $x_{new})$
8   end
    Output: $T$, $H$, $N_h$
```

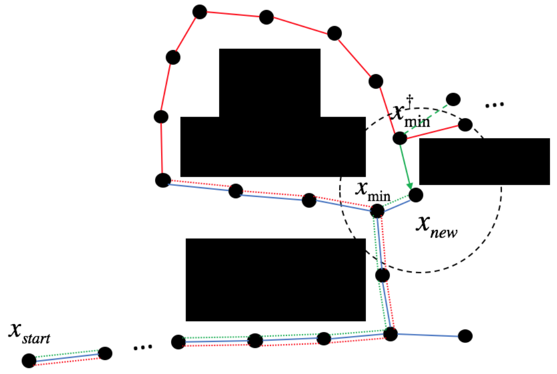


Fig. 1. An example scenario for Inter-Homology-Class Rewire. Suppose there are two homology classes discovered, where the red sub-tree is grown out the blue sub-tree and the ancestry vertices that belongs to the blue sub-tree are added to the red sub-tree. During the expansion of the red sub-tree, suppose a new vertex x_{new} is generated from the vertex x_{min}, which consequently expands the blue sub-tree. The "Inter-Homology-Class Rewire" procedure is then performed, where a vertex $x^{\dagger}_{min}$, carrying a different *h-signature* from x_{new}, finds a smaller cost path by accessing from x_{new}. By connecting a copy of $x^{\dagger}_{min}$ to x_{new}, a new homology class is discovered and a new sub-tree (represented by green lines) is initialized.

the state of x_{min} and the new winding number $w^{\dagger}$, and added to the corresponding sub-tree $T_{i_{h^{\dagger}}}$, as in Lines 6-7.

IV. iLQG Based Belief Space Planning

In this section, the iLQG based belief space planner is briefly reviewed. The readers are referred to [5] for complete discussion. The iLQG based belief space planner uses parametrized belief state representation and applies the iLQG controller to perform value iteration. Specifically, let motion noise m_t and sensing noise n_t be modeled by Gaussian noises, i.e., $m_t \sim N(0,I)$ and $n_t \sim N(0,I)$. The stochastic robot dynamics is given as $x_{t+1} = f(x_t, u_t, m_t)$, and observation model is given as $z_t = h(x_t, n_t)$, where $x_t \in \mathcal{C}_f$ denotes the state vector, $z_t \in \mathcal{Z}$ denotes the observation vector, $u_t \in \mathbb{U}$ is the control input. The belief b_t is represented by Gaussian distributions and is defined as $b_t = (\hat{x}_t^T, vec(\sqrt{\Sigma_t}))$, where Σ_t is the variance of the Gaussian distribution $\mathcal{N}(\hat{x}_t^T, \Sigma_t)$, and $\hat{x}_t$ denotes the estimated state as the system is partially observable. The *vec* function is used to vectorize the variance matrix. The iLQG controller employs the extended Kalman

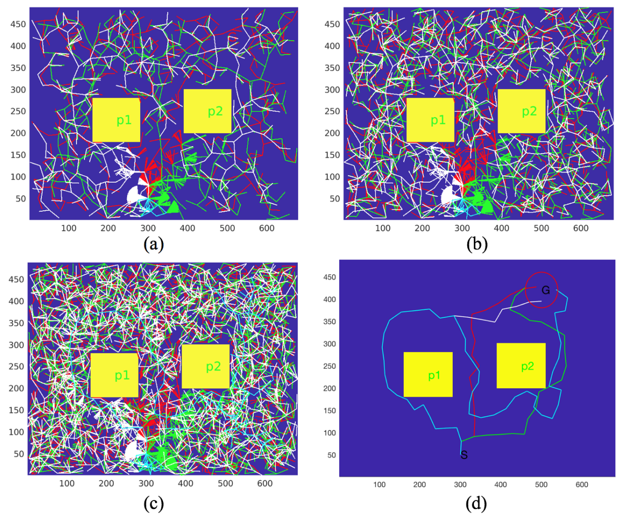


Fig. 2. The expansion of the HRRT tree in an example 2D scenario under the modulo of 2. (a)-(c) The expansion of the four sub-trees at iteration of 500, 1000, and 2000, respectively. (d) The trajectories in 4 different homology classes discovered by the HRRT algorithm after 2000 iterations (0.2 s), where the cyan trajectory is discovered during the expansion of the white sub-tree. The *h-signatures* of the trajectories plotted in red, green, white, cyan are $(1,-1)$, $(1,1)$, $(-1,-1)$, and $(-1,1)$, respectively.

filter (EKF) to propagate the belief dynamics. Value iteration is performed to locally optimize the control policies u_t and iteratively minimize the expected value of the user-defined cost function, where a backward sweep and a forward sweep are executed in each iteration.

V. Results

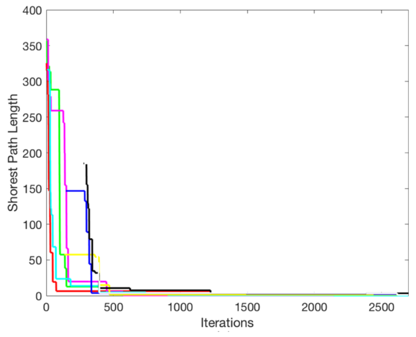
The proposed methods are evaluated in simulations of motion planning scenarios. First, the performance of the HRRT/HRRT* planner is evaluated in 2D scenarios using the winding number modulo $m = 2$, followed by examples of uncertainty reasoning in 2D and 3D spaces using the homology guided belief space planning method.

A. Planning Results of HRRT/HRRT*

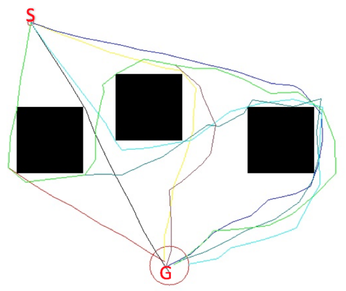
The HRRT/HRRT* algorithms are implemented in C++ on Ubuntu 20.04 operating system. The computer is equipped with Intel® Core™ i9-11900 CPU @ 2.50GHz and 32.0 GiB memory. First, the HRRT sub-tree expansion in 500, 1000, and 2000 iterations are presented in Fig. 2 (a)-(c), where the 4 different sub-trees are marked in 4 different colors. The homology inequivalent nominal trajectories found by the HRRT planner are given in Fig. 2 (d).

The performance of the HRRT* is tested in a 2D environment with 3 rectangular obstacles, as presented in Fig. 3. All 2^3 homology inequivalent trajectories are discovered by the HRRT* after 2700 iterations (2.5 s) as shown in Fig. 3 (b). Fig. 3 (a) shows the shortest distance reduction of the optimal trajectories in each homology classes vs expansion iterations.

The performance of the HRRT* is then compared with PI-RRHT* [9] and WA-RRT*[14] in 3 and 4 obstacles scenarios over 20 trials, as presented in Fig. 4, where a narrow passage between two of the obstacles is created in the 3 obstacles scenario. Fig. 4 (a) and (b) present the 3 and 4 obstacles



(a)



(b)

Fig. 3. Generating homology inequivalent trajectories in 2D environment with 3 obstacles. (a) The shortest distance of the optimal trajectory in each homology class to the goal state vs expansion iterations. (b) All 2^3 homology inequivalent trajectories found by HRRT* algorithm under the modulo of 2 after 2700 iterations (2.5 s).

TABLE I

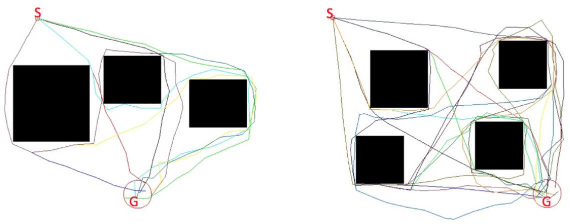
TIME FOR ALL HOMOLOGY INEQUIVALENT TRAJECTORIES TO ARRIVE THE GOAL REGION IN THE 3- AND 4-OBSTACLE SCENARIOS

Scenarios	HRRT*	PI-RRHT*	WA-RRT*
8-Homology Classes	0.53 s	0.98 s	12.74 s
16-Homology Classes	2.26 s	2.43 s	94.42 s

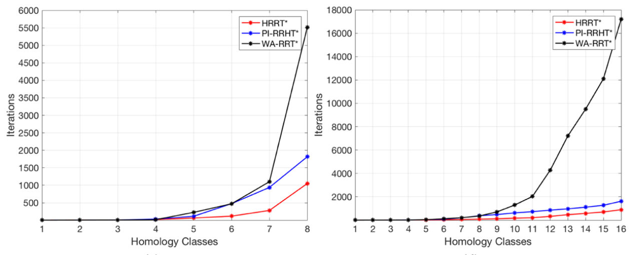
scenarios and the homology inequivalent trajectories discovered by the HRRT*. The average homology class discovery rate of HRRT*, PI-RRHT*, and WA-RRT* over 20 trials given scenarios in (a) and (b) are presented in Fig. 4 (c) and (d), respectively. In the 3 obstacles environment with one narrow passage, HRRT* finds all 8 homology classes in 1050 iterations on average, PI-RRHT* finds the 8 homology classes in 1814 iterations on average, and WA-RRT* finds the 8 homology classes in 5516 iterations on average. In the 4 obstacles scenario, HRRT* finds all 16 homology classes in 890 iterations on average, PI-RRHT* finds the 8 homology classes in 1616 iterations on average, and WA-RRT* finds the 8 homology classes in 17188 iterations on average. As shown in Fig. 4 (c) and (d), HRRT* improves the homology class discovery speed compared to PI-RRHT* and WA-RRT* in both scenarios. Table. I presents the average time for all homology inequivalent trajectories to arrive at the goal region in scenarios (a) and (b) over 20 trials.

B. Uncertainty Reasoning Via Homology-Guided Belief Space Planning

The first belief space planner example presented is a car-like robot in a 2D light-dark environment, where the robot



(a) (b)



(c) (d)

Fig. 4. (a) 8 homology inequivalent trajectories found using HRRT* in a 3 obstacles scenario with a narrow passage. (b) 16 homology inequivalent trajectories found using HRRT* in a 4 obstacles scenario. (c) The homology classes discovered (x-axis) by HRRT*, PI-RRHT*, and WA-RRT* vs exploration iterations (y-axis) in scenario (a). (d) The homology classes discovered by HRRT*, PI-RRHT*, and WA-RRT* vs exploration iterations in scenario (b).

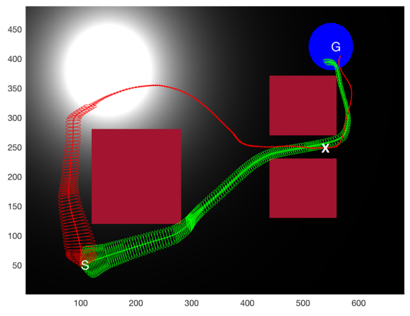


Fig. 5. The light-dark example where the robot's sensing noise is reduced when its positional distance to the beacon is smaller. The shortest trajectory in green homology class is unsafe to pass the narrow passage, while the trajectory in the red homology class is able to safely pass the narrow passage with smaller uncertainty.

needs to pass a narrow passage to pick up an object marked by "X", as presented in Fig. 5, then reach to the goal region (mark by blue). The sensing noise decreases exponentially with respect to its relative position to the light region. Two homotopy inequivalent nominal trajectories passing through the passage found by the HRRT* planner are locally optimized in belief space using the iLQG based belief space planner. As shown in Fig. 5, the shortest trajectory in the green homology class has higher probability of colliding, compared to the optimized trajectory in the red homology class, which is able to gain better sensing and is thus preferable for the mission. This illustrates the advantage of the HRRT* planner based algorithm in identifying a globally optimal solution by generating homology inequivalent nominal trajectories which are subsequently locally optimized.

In the second example, we present a highway car passing scenario, as shown in Fig. 6 (a). In this scenario, the red car needs to pass the front car while two car from left and right lane are in close distance. The state space in this problem is

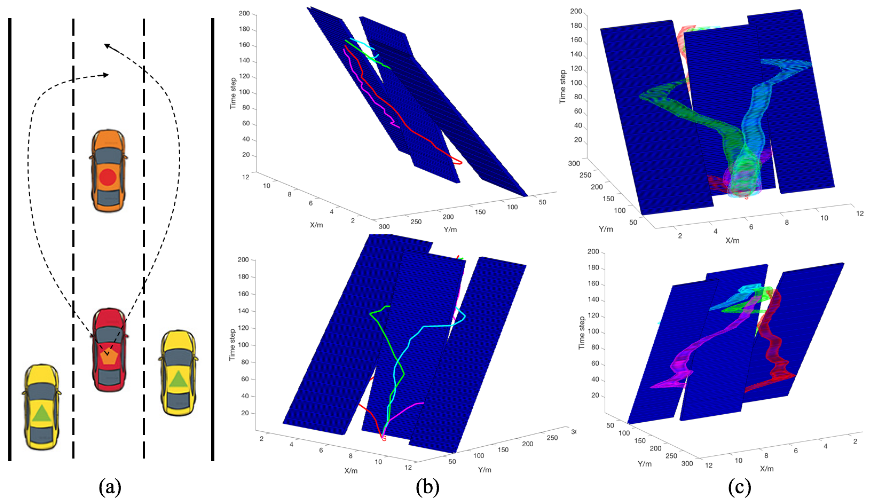


Fig. 6. (a) The highway car passing scenario where the red car needs to choose a lane to pass the front car with two cars chasing up from the left and right lanes. The red car needs to decide which lane to pass and when to pass, i.e., pass after or before the yellow cars. (b) The car passing scenario is a 3D planning problem, with lateral (x-axis) and longitudinal (y-axis) positions and time (z-axis). The motions of the front and side cars are represented by the blue rectangles. Four different homology inequivalent trajectories are found using the HRRT planner, represented in 4 different colors. The red and magenta trajectories complete the passing before the yellow cars from the left and the right lanes, respectively. The green and cyan trajectories complete the passing after the yellow cars from the left and the right lanes, respectively. (c) The optimized trajectories in 4 homology classes using iLQG based belief space planner.

3-dimensional (x, y, t), as the obstacles are not static. In this scenario, the red car has 4 choices for passing, i.e., make the passing before or after the yellow car in the left or the right lane. Suppose the front car is moving with speed of 22 m/s, the yellow cars in the left and right lane are moving with the speed of 40 m/s and 31 m/s, respectively. The motion of the front car and the two cars from the left and right lanes vs 200 future time steps are presented in Fig. 6 (b), represented by the blue parallelepipeds. As shown in Fig. 6 (b), 4 nominal trajectories are found using the HRRT planner, where the topological task projection method proposed in [14] is employed to identify winding centers in x-y and y-z planes.

The iLQG belief space planner is then employed to find the locally optimal trajectories that minimizes the probability of potential collisions, the optimized trajectories are presented in Fig. 6 (c). The motion noise of the red car is scaled to the reciprocal of the distance to the nearest car at a given time step. As shown in Fig. 6 (c), the trajectory in the red homology class has a high probability of collision with the front car, while the trajectory in the cyan and magenta homology classes have high probabilities of collision with the front and the right lane cars. The trajectory in green homology class is a safer option for passing, as it has a smaller probability of collision with the front or the left lane cars, compared to the trajectories in the red, cyan, and magenta homology classes.

VI. Conclusions

This paper presents a homology guided belief space motion planning method in obstacle-cluttered partial observable environment. The proposed method uses a two step approach. The proposed HRRT/HRRT* algorithms efficiently explore different homology classes and find homology inequivalent nominal trajectories. HRRT* increases the homology class discovery speed by exploiting the parallel sub-trees expansion in different homology classes and the active inter-homology sub-tree growth, compared to the PI-RRHT* and WA-RRT* algorithms. Then, the iLQG based belief space planner is employed to locally optimize each of the homology inequivalent nominal trajectories over the motion and sensing uncertainties. Experimental examples for uncertainty reasoning and decision making in obstacle-cluttered environments using the proposed method are presented. Although the present homology guided belief space planning method uses iLQG based belief space planner, the HRRT/HRRT* can be employed by other local belief space planning methods, e.g., [3], [6], [22], [23].

References

[1] A. Hatcher, *Algebraic topology*. Cambridge University Press, 2002.

[2] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[3] R. Platt Jr, R. Tedrake, L. Kaelbling, and T. Lozano-Perez, "Belief space planning assuming maximum likelihood observations," 2010.

[4] T. Erez and W. D. Smart, "A scalable method for solving high-dimensional continuous pomdps using local approximation," in *Conference on Uncertainty in Artificial Intelligence*, 2010, pp. 160–167.

[5] J. Van Den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1263–1278, 2012.

[6] L. Janson, E. Schmerling, and M. Pavone, "Monte carlo motion planning for robot trajectory optimization under uncertainty," in *Robotics Research*. Springer, 2018, pp. 343–361.

[7] S. Bhattacharya, "Search-based path planning with homotopy class constraints," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, no. 1, 2010.

[8] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Autonomous Robots*, vol. 33, no. 3, pp. 273–290, 2012.

[9] J.-S. Ha and H.-L. Choi, "A topology-guided path integral approach for stochastic optimal control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4605–4612.

[10] F. T. Pokorny and D. Kragic, "Data-driven topological motion planning with persistent cohomology." in *Robotics: Science and Systems*, 2015.

[11] C. Rösmann, F. Hoffmann, and T. Bertram, "Integrated online trajectory planning and optimization in distinctive topologies," *Robotics and Autonomous Systems*, vol. 88, pp. 142–153, 2017.

[12] S. Bhattacharya and R. Ghrist, "Path homotopy invariants and their application to optimal trajectory planning," *Annals of Mathematics and Artificial Intelligence*, vol. 84, no. 3, pp. 139–160, 2018.

[13] V. Ranganeni, O. Salzman, and M. Likhachev, "Effective footstep planning for humanoids using homotopy-class guidance," in *Twenty-Eighth International Conference on Automated Planning and Scheduling*, 2018.

[14] F. T. Pokorny, D. Kragic, L. E. Kavraki, and K. Goldberg, "High-dimensional winding-augmented motion planning with 2d topological task projections and persistent homology," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 24–31.

[15] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 723–730.

[16] W. Sun, J. van den Berg, and R. Alterovitz, "Stochastic extended lqr for optimization-based motion planning under uncertainty," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 437–447, 2016.

[17] O. Salzman and D. Halperin, "Optimal motion planning for a tethered robot: Efficient preprocessing for fast shortest paths queries," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4161–4166.

[18] K. Hormann and A. Agathos, "The point in polygon problem for arbitrary polygons," *Computational geometry*, vol. 20, no. 3, pp. 131–144, 2001.

[19] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[20] S. M. LaValle and S. A. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 912–925, 1998.

[21] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *Robotics Science and Systems VI*, vol. 104, no. 2, 2010.

[22] E. Schmerling and M. Pavone, "Evaluating trajectory collision probability through adaptive importance sampling for safe motion planning," *arXiv preprint arXiv:1609.05399*, 2016.

[23] S. Patil, G. Kahn, M. Laskey, J. Schulman, K. Goldberg, and P. Abbeel, "Scaling up gaussian belief space planning through covariance-free trajectory optimization and automatic differentiation," in *Algorithmic foundations of robotics XI*. Springer, 2015, pp. 515–533.