2024 IEEE Conference on Control Technology and Applications (CCTA)
August 21-23, 2024. Newcastle upon Tyne, UK

Efficient Path Planning with Soft Homology Constraints

Carlos A. Taveras, Santiago Segarra, and César A. Uribe

2024 IEEE Conference on Control Technology and Applications (CCTA) | 979-8-3503-7094-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/CCTA60707.2024.10666586

***Abstract*— We study the problem of path planning with soft homology constraints on a surface topologically equivalent to a disk with punctures. Specifically, we propose an algorithm, named $\mathcal{H}^*$, for the efficient computation of a path homologous to a user-provided reference path. We show that the algorithm can generate a suite of paths in distinct homology classes, from overall shortest path to the short path homologous to the reference, ordered both by path length and similarity to the reference path. Rollout is shown to improve the results produced by the algorithm. Experiments demonstrate that $\mathcal{H}^*$ can be an efficient alternative to optimal methods, especially for configuration spaces with many obstacles.**

I. Introduction

Planning algorithms are ubiquitous in robotics, serving as the bridge between high-level task specifications and the low-level instructions that completion of the task necessitates [1]. Whether due to a robot's intrinsics, or the presence of obstacles in an environment, most real-world configuration spaces (C-spaces) in motion planning problems have non-trivial topology [2]. Paths in a C-space with non-trivial topology can be partitioned into classes by topological equivalence relations, namely homotopy and homology. Topology-constrained path planning has seen use in several applications, including multi-agent coordination [3], motion planning in dynamic environments [4], and guided AUV navigation [5].

As previously mentioned, homotopy and homology can define equivalence relations on paths. Two paths connecting the same points are homotopic if they can be continuously deformed into one another; they are homologous if their concatenation does not enclose any holes. Homotopy (homology) classes are defined as the set of all homotopic (homologous) paths. The Hurewicz theorem [6] implies that homotopic paths are also homologous (the converse, however, is false; see [7, Figure 2]).

The fields of computational geometry and topology have long studied topology constrained path planning problems over surfaces. An important line of work in these fields is that of finding the shortest path between a pair of points belonging to a designated homotopy class. For example, [8] uses the funnel algorithm [9] to solve this problem for boundary-triangulated 2-manifolds and [10] extends these results to a more general class of combinatorial surfaces. Related work includes solving for shortest homologous cycles [11] and computing a minimal homology basis [12]. Recently, the robotics motion planning community has developed various search [7], [13], [14], sampling [15]–[17], and optimization [4] based topology aware motion planning methods.

Despite these developments, there is no efficient method for producing shortest paths in homology classes that are similar to, but topologically distinct from, a designated class. Towards this end, [7] introduces the H-signature, a homology class invariant, that can be used to create an augmented graph (or road map) over which computing the shortest path in a homology class is trivial. This method can be inefficient, however, as it often requires the computation of paths belonging to homology classes that are very different from the designated class and of little practical relevance (e.g., paths that loop around holes).

Our method enables users to tailor their search to paths in homology classes similar to the designated class efficiently, as experiments validate. Other works related to ours include [4] and [14] which can efficiently construct paths in distinct homotopy classes, however, they do not explicitly aim to minimize the length of the resulting paths as our method and [7] do.

We propose $\mathcal{H}^*$ an efficient search-based algorithm that can produce a sequence of paths in distinct homology classes for C-spaces homeomorphic to a disk with holes, ordered by length and the degree of proximity to a user-designated homology class. We propose a distance metric between homology classes and show that paths that are close in this metric appear topologically similar. As in [15], we only consider paths that do not form complete loops around obstacles/holes.

This paper is organized as follows. Section II establishes the necessary background, notation, and preliminary results. In Section III, we state the objectives of this paper. In Section IV, we propose and justify algorithms for solving the objectives. In Section V, we demonstrate the effectiveness of the proposed algorithms and compare them with the methods proposed in [7]. We conclude in Section VI with a summary.

II. Preliminaries

A. Simplicial Complexes

Let $\mathcal{V} = \{v_1, ..., v_N\}$ be a finite set of elements called vertices. We call a non-empty subset $\sigma \subset \mathcal{V}$ with $|\sigma| = k+1$ a k-simplex, and say it has dimension k. An (abstract) simplicial complex (SC) $\mathcal{X}$ is a collection of simplices that is

Research was sponsored by the Army Research Office and was accomplished under Grant Number W911NF-17-S-0002. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Army or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. Part of this material is based upon work supported by the National Science Foundation under Grants #2211815 and #2213568 and the Ken Kennedy-HPE Cray 2023/24 Fellowship. The authors are with the Dept. of ECE, Rice University. Emails:{ctaveras, segarra, cauribe}@rice.edu.

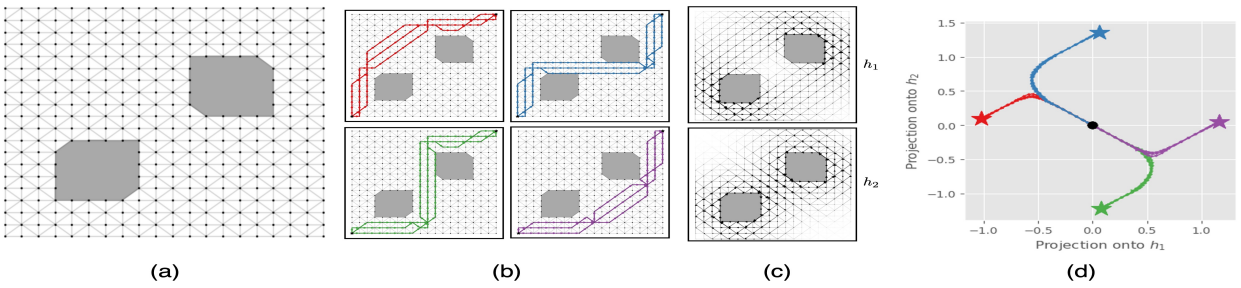

Fig. 1. (a) Simplicial Complex with two holes, (b) four distinct homotopy/homology classes with three paths each, (c) harmonic projection basis vectors, and (d) the progression of the harmonic projection for each path in (b).

closed under restriction [18]; i.e., if $\sigma \in \mathcal{X}$ and $\pi \subseteq \sigma$ then $\pi \in \mathcal{X}$. The SCs in this paper are assumed to be discretized (manifold) surfaces [19] and are thus comprised of $\{0,1,2\}$-simplices which we call nodes, edges and triangles, respectively. Denoting the sets of edges and triangles respectively as $\mathcal{E}$, and $\mathcal{T}$, we can express an SC as $\mathcal{X} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$. Each edge $e \in \mathcal{E}$ is assigned a weight by a function w such that $w(e) > 0$.

We assign an orientation to all nodes, edges and triangles. Nodes have positive orientation by convention. The canonical positive orientation of an edge/triangle is that in which all nodes are in label order (e.g., $[v_1, v_2, v_3]$). A pair of oriented edges/triangles are equivalent if they differ by an even permutation (e.g., $[v_1, v_2, v_3] = [v_3, v_1, v_2]$). An oriented edge/triangle can be negated by any odd permutation of its elements (e.g., $-[v_1, v_2] = [v_2, v_1]$). Orientation and sign can be extended to simplices of any dimension; we denote the set of oriented k-simplices by $\mathcal{X}^k$.

The k-th chain group $\mathcal{C}_k$ of $\mathcal{X}$ is the vector space over $\mathbb{R}$ generated by a basis formed by the oriented k-simplices. As $\mathcal{C}_k$ is a finite-dimensional vector space, its elements, called k-chains, can be represented as vectors with real coefficients. The boundary operator ∂_k is a linear mapping from $\mathcal{C}_k$ to $\mathcal{C}_{k-1}$ which acts on $\sigma \in \mathcal{X}^k$ by $\partial_k(\sigma) = \sum_{i=0}^{m} (-1)^i \sigma^{-i}$ where $\sigma^{-i} = [\sigma_0, ..., \sigma_{i-1}, \sigma_{i+1}, ..., \sigma_m]$ is the $(k-1)$-simplex formed by removing the i-th node of σ. As a linear operator ∂_k can be encoded by a matrix.

Recommended texts on simplicial complexes, their applications, and related concepts include [6], [18], [20].

B. Paths

Let $v_i, v_f \in \mathcal{V}$ be distinct nodes, which we respectively call the source and destination. We represent a path as a sequence of edge-connected nodes $\tau = (\tau^{(0)}, ..., \tau^{(n)})$ where $\tau^{(0)} = v_i$ and $\tau^{(n)} = v_f$. We denote the (weighted) length of τ by

$$W(\tau) = \sum_{i=0}^{n-1} w(\tau^{(i)}, \tau^{(i+1)}) \tag{1}$$

We define simple algebraic operations on paths as follows. Let $\tau_1 = (\tau_1^{(0)}, ..., \tau_1^{(n)})$ and $\tau_2 = (\tau_2^{(0)}, ..., \tau_2^{(m)})$ be paths. Addition is defined by path concatenation $\tau_1 + \tau_2 = (\tau_1^{(0)}, ..., \tau_1^{(n)}, \tau_2^{(1)}, ..., \tau_2^{(m)})$ given that $\tau_1^{(n)} = \tau_2^{(0)}$. Negation is defined as path reversal $-\tau_1 = (\tau_1^{(n)}, ..., \tau_1^{(0)})$ and subtraction by $\tau_1 - \tau_2 = \tau_1 + (-\tau_2)$. The sum of a path and a node is $\tau_1 + (v) = \tau_1 + (\tau_1^{(n)}, v)$ given that $(\tau_1^{(n)}, v) \in \mathcal{E}$.

A path from v_i to v_f can also be represented as a 1-chain $x \in C_1$, in which $\partial_1(x) = -v_i + v_f$. The map Φ can transform a path τ to $x \in C_1$

$$x = \Phi(\tau) = \sum_{i=0}^{n-1} \phi(\tau^{(i)}, \tau^{(i+1)}) \tag{2}$$

where $\phi(v_i, v_j) = [v_i, v_j]$ if $i < j$, and $\phi(v_i, v_j) = -[v_j, v_i]$, otherwise.

Definition 1: (Homologous paths) Paths τ_1 and τ_2 connecting points v_i, v_f are *homologous*, denoted $\tau_1 \sim \tau_2$, if the difference of their 1-chain representations $\Phi(\tau_1) - \Phi(\tau_2)$ is a 2-chain boundary (i.e. $\Phi(\tau_1) - \Phi(\tau_2) \in \text{im}(\partial_2)$).
It can be shown that $\sim$ is an equivalence relation on paths with fixed endpoints. We refer to an equivalence class of homologous paths as a homology class.

C. Hodge Theory, Homology and Harmonics

The composition of consecutive boundary maps is null, that is, for all $x \in \mathcal{C}_{k+1}$, $\partial_k \circ \partial_{k+1}(x) = 0$ [20]. As a consequence, we can define the quotient vector space $H_k = \ker(\partial_k)/\text{im}(\partial_{k+1})$, called the k-th homology group, where $\text{im}(\cdot)$ and $\ker(\cdot)$ respectively denote the image and kernel of an operator. Moreover, the Hodge k-Laplacian can be defined using the boundary operator as

$$L_k = \partial_k^\top \partial_k + \partial_{k+1} \partial_{k+1}^\top. \tag{3}$$

The Hodge Decomposition Theorem [21] states that C_k can be decomposed into three orthogonal subspaces

$$C_k \cong \text{im}(\partial_{k+1}) \oplus \text{im}(\partial_k^\top) \oplus \ker(L_k) \tag{4}$$

where $\oplus$ denotes the direct sum between a pair of subspaces.

Discrete Hodge Theory implies that H_k and $\ker(L_k)$ are isomorphic [22], allowing for an algebraic condition to determine whether a pair of paths are homologous. Toward this, we construct a matrix $H = [h_1 ... h_D]$, where D is the number of holes in the SC, whose columns span $\ker(L_1)$ and define an operator which we call the harmonic projection of a path τ

$$\gamma(\tau) = H^\top \Phi(\tau). \tag{5}$$

Proposition 1: Two paths τ_1 and τ_2 connecting the same points are homologous if and only if $\gamma(\tau_1) = \gamma(\tau_2)$.

Proof: Let $x_1 = \Phi(\tau_1)$ and $x_2 = \Phi(\tau_2)$.

Suppose $\tau_1 \sim \tau_2$. Therefore, $x_1 - x_2 \in \text{im}(\partial_2)$. By construction, $\text{im}(H) = \ker(L_1)$. By Eq. (4), $\text{im}(\partial_2) \cap \ker(L_1) = \{0\}$. Taken together with [23, Theorem 2] we have $x_1 - x_2 \in \ker(H^\top)$, implying that $H^\top x_1 = \gamma(\tau_1) = \gamma(\tau_2) = H^\top x_2$.

Now, let $\gamma(\tau_1) = H^\top x_1 = H^\top x_2 = \gamma(\tau_2)$. Then, $x_1 - x_2 \in \ker(H^\top) = \text{im}(H)^\perp$. By construction $\text{im}(H) = \ker(L_1)$, and by by Eq.(4) we have $x_1 - x_2 \in \text{im}(\partial_2) \oplus \text{im}(\partial_1^\top)$. Since τ_1 and τ_2 share endpoints, $x_1 - x_2 \in \ker(\partial_1) = \text{im}(\partial_1^\top)^\perp$, thus $x_1 - x_2 \in \text{im}(\partial_2)$ and $\tau_1 \sim \tau_2$. ■

Works that leverage harmonic 1-(co)chains to measure topological similarity between paths include [24]–[30]. Proposition 1 is analogous to [7, Lemma 2] as the holomorphic functions in H-signature are harmonic [19].

III. Problem Statement

Let $\mathcal{X} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$ be an oriented 2-dimensional simplicial complex topologically equivalent to a disk with D holes. Let $v_i, v_f \in \mathcal{V}$ respectively denote the source and destination nodes. Let $\bar{\tau} = (\bar{\tau}^{(0)}, ..., \bar{\tau}^{(n)})$ be a path called the reference path with harmonic projection $\bar{\gamma} = \gamma(\bar{\tau})$. We aim to efficiently compute a path τ^* homologous to $\bar{\tau}$ with minimal length; in other words, we want to solve

$$\tau^* = \underset{\tau \in \mathcal{P}}{\arg\min} \quad W(\tau) \;\text{ s.t. }\; \gamma(\tau) = \gamma(\bar{\tau}). \tag{P1}$$

where $\mathcal{P}$ is the set of all edge-connected paths with source v_i and destination v_f.

A search-based algorithm to solve P1 would require the ability to compute several paths connecting v_i to v_f that are the shortest in their respective class. The method proposed in [7] enables this by lifting the problem to an augmentation of the original graph in which an augmented node $v' = (v, H(v))$ consists of both a node $v \in \mathcal{V}$ and an H-signature (harmonic projection) $H(v) \in \mathbb{C}^D$. A τ^* can thus be found by solving for a path connecting $v_i' = (v_i, 0)$ to $v_f' = (v_f, \bar{\gamma})$ in the augmented graph using A^* [31].

Depending on the length of τ^*, however, this algorithm can require constructing extremely large graphs, which we would like to avoid, so we restrict our focus to solutions in the original problem space.

To bypass the homology constraint, we soften it, introducing a penalty to the cost function, as one does in Lagrangian-based optimization [32]. Let

$$\Delta\gamma(\tau_1, \tau_2) = \|\gamma(\tau_1) - \gamma(\tau_2)\|_2 \tag{6}$$

denote the harmonic projection difference between τ_1 and τ_2, where $\|\cdot\|_2$ is the Euclidean norm. We reformulate P1 as

$$\min_{\tau \in \mathcal{P}} \quad C^\alpha(\tau, \bar{\tau}) := W(\tau) + \alpha\Delta\gamma(\tau, \bar{\tau}). \tag{P2}$$

where $\alpha > 0$. This reformulation cannot be solved optimally on the SC using dynamic programming-based (DP) methods as the cost function lacks the optimal substructure that characterizes DP problems [33]. Therefore, we develop a heuristic function to approximate a solution to P2 that leverages the sequential structure of the harmonic projection (as seen in Fig. 1(d)) and use rollout to improve results produced by the heuristic. It can be shown that for sufficiently large α, the minimizers for P1 and P2 are equivalent.

Proposition 2: There exists a sufficiently large $\alpha > 0$ in P2, such that a minimizer of P1 is a minimizer of P2.

Proof: Let τ_1^* denote the shortest path from v_i to v_f and τ^* be a solution to P1 (i.e. τ^* is shortest path such that $\tau^* \sim \bar{\tau}$). Without loss of generality, assume that there are $m \geq 1$ distinct homology classes whose respective shortest paths $\tau_1^*, ..., \tau_m^*$ have shorter length than τ^* (i.e. $W(\tau_1^*) < ... < W(\tau_m^*) < W(\tau^*)$). Proposition 1 states that if $\tau^* \sim \bar{\tau}$ then $\gamma(\tau^*) = \gamma(\bar{\tau})$, hence the cost of τ^* in P2 is

$$C^\alpha(\tau^*, \bar{\tau}) = W(\tau_i^*) + \alpha\Delta\gamma(\tau^*, \bar{\tau}) = W(\tau^*) + 0 = W(\tau^*)$$

which is independent of α. Path τ^* is thus a minimizer of P2 when α^* satisfies $C^{\alpha^*}(\tau^*, \bar{\tau}) < C^{\alpha^*}(\tau_i^*, \bar{\tau})$ for all τ_i^*, in particular, when

$$\alpha^* > \max\{\alpha_i\}_{i=1}^m = \max\left\{\frac{W(\tau^*) - W(\tau_i^*)}{\Delta\gamma(\tau_i^*, \bar{\tau})}\right\}_{i=1}^m. \tag{7}$$

■

We say P2 has a soft homology constraint as it penalizes paths in homology classes that are "farther away" from the desired homology class, as measured by the projection difference $\Delta\gamma$.

IV. Methods

A. The $\mathcal{H}^*$ Algorithm

We introduce $\mathcal{H}^*$, a homology-informed heuristic algorithm best-first search algorithm providing approximate solutions to P2. $\mathcal{H}^*$ is essentially A^* [31] with a heuristic that measures the harmonic projection difference between $\bar{\tau}$ and a partial path from v_i to a node $\tau^{(k)}$. In particular, the cost of a node $\tau^{(k)}$ connected to v_i by path $\tau_k = (\tau^{(0)}, ..., \tau^{(k)})$ is $W(\tau_k) + \alpha\Delta\gamma(\tau_k, \bar{\tau})$. The parameter α thus controls the extent to which the projection difference contributes to the overall cost at a node. Note that when the destination node is reached, its cost coincides with the cost function in P2.

The $\mathcal{H}^*$ heuristic was inspired by the equivalence of homologous paths in harmonic projection (Proposition 1) and an observation about the similarity of sequential structure of paths within a homology class, as can be seen in Fig. 1(d). Notice, in particular, that over each edge in a path in Fig. 1(d), the projection difference tends to decrease until the destination is reached.

We now describe the parameters and steps of the $\mathcal{H}^*$ algorithm. At a stage k, a set A_k of visited nodes is maintained; we denote the set of unvisited nodes by $A_k^c = \mathcal{V} \setminus A_k$. Each node $v \in \mathcal{V}$ is associated with: i) a weight $W_k(v)$, ii) harmonic projection $\gamma_k(v)$, iii) cost $C_k^\alpha(v) = W_k(v) + \alpha\Delta\gamma_k(v)$ where $\Delta\gamma_k(v) = \|\gamma_k(v) - \bar{\gamma}\|_2$, and iv) the node $\text{prev}(v)$ preceding v in the path from v_i to v.

At stage 0, we initialize the aforementioned data to $W_0(v) = \mathbb{I}(v, v_i), \gamma_0(v) = \mathbf{0}_D \in \mathbb{R}^D, C_0^\alpha(v) = W_0(v) + \alpha\Delta\gamma_0(v)$, and prev $= \emptyset$ where $\mathbb{I}(u, v) = 0$ if $u = v$, and ∞, otherwise; $\emptyset$ means that a node has no predecessor. For $k > 0$, we find the node in A_{k-1}^c with least cost,

$$v_k^* = \underset{v \in A_{k-1}^c}{\arg\min}\, C_{k-1}^\alpha(v), \tag{8}$$

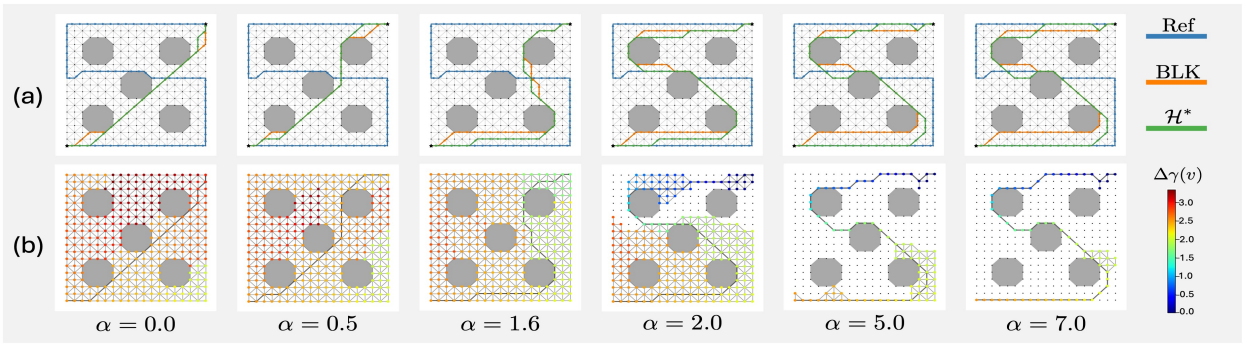

Fig. 2. (a) Paths produced by $\mathcal{H}^*$ for various α and (b) the projection difference of all visited nodes after a path has been found.

add it to the visited node set $A_k = A_{k-1} \cup \{v_k^*\}$, then update the cost of its neighbors, which we denote by $\mathcal{N}(v_k^*) = \{u \in \mathcal{V} \mid (u, v_k^*) \in \mathcal{E}\}$. The cost of $v \in \mathcal{N}(v_k^*) \cap A_k^c$ is

$$C_k^\alpha(v) = \min\{C_{k-1}^\alpha(v), W'(v) + \alpha\|\gamma'(v) - \bar{\gamma}\|_2\} \quad (9)$$

$$W'(v) = W_{k-1}(v_k^*) + w(v_k^*, v) \quad (10)$$

$$\gamma'(v) = \gamma_{k-1}(v_k^*) + \gamma(v_k^*, v) \quad (11)$$

where if $C_k^\alpha(v) < C_{k-1}^\alpha(v)$, then

$$W_k(v) = W'(v), \gamma_k(v) = \gamma'(v), \text{prev}(v) = v_k^*. \quad (12)$$

The search ends when $v_k^* = v_f$, after which, the path from v_i to v_f is reconstructed using prev. We use $\mathcal{H}^*(v_i, v_f, \tau)$ to denote the output of the $\mathcal{H}^*$ algorithm.

B. Fortified Rollout

Solutions to a combinatorial problem provided by a heuristic algorithm (not to be confused with heuristic in the A* sense) can be improved by embedding the heuristic in a rollout framework [34]. Instead of solving for the entire path, as in P2, rollout sequentially constructs the path using a heuristic. We detail how rollout is leveraged for our purposes. At a non-terminal stage k, we maintain a path $\tau_k = (\tau^{(0)}, ..., \tau^{(k)})$, where $\tau^{(0)} = v_i$ and $\tau^{(k)} \neq v_f$. The path is updated by $\tau_{k+1} = \tau_k + (v_{k+1})$, where

$$v_{k+1} = \underset{v \in \mathcal{N}(\tau^{(k)})}{\arg\min}\ C^\alpha(\tau_k + \mathcal{H}^*(v, v_f, \bar{\tau}_k(v)), \bar{\tau}). \quad (13)$$

Because the source node for $\mathcal{H}^*$ in Eq. (13) changes at each stage, the reference path at stage k for a node v is $\bar{\tau}_k(v) = (v) - \tau_k + \bar{\tau}$.

If a heuristic satisfies the sequential consistency property [34, Definition 2.2], rollout produces a path no worse than the one produces by the heuristic alone [34, Proposition 3.2]. Under mild conditions, a heuristic can be made sequentially consistent using the fortified rollout algorithm outlined in [34, Section 4]. The key difference between rollout and fortified rollout is that at every stage, the fortified variant maintains a full path $\tilde{\tau}_k$ from v_i to v_f; in particular, continuing from (13), we have

$$\tilde{\tau}_{k+1} = \arg\min\{C^\alpha(\tilde{\tau}_k, \bar{\tau}), C^\alpha(\tau'_{k+1}, \bar{\tau})\} \quad (14)$$

$$\tau'_{k+1} = \tau_k + \mathcal{H}^*(v_{k+1}, v_f, \bar{\tau}_k(v_{k+1})) \quad (15)$$

where $\tilde{\tau}_0 = \mathcal{H}^*(v_i, v_f, \bar{\tau})$.

While rollout can improve the performance of $\mathcal{H}^*$, it requires several uses of $\mathcal{H}^*$, thereby increasing the effective number of nodes visited to produce a result. We mitigate this by pruning nodes that increase the projection difference by more than some small value ϵ

$$\mathcal{N}(v, \epsilon) = \{u \in \mathcal{N}(v) | \Delta\gamma(\tau_k + (u), \bar{\tau}) < \Delta\gamma(\tau_k, \bar{\tau}) + \epsilon\}. \quad (16)$$

$\mathcal{RH}^*(v_i, v_f, \bar{\tau})$ and $\mathcal{PRH}^*(v_i, v_f, \bar{\tau}, \epsilon)$ denote the outputs of fortified rollout without and with pruning, respectively.

V. Numerical Analysis

We show the utility of $\mathcal{H}^*$ and its variants on various synthetic C-spaces. C-spaces are constructed by triangulating a uniform grid of 19×19 of points on $[-1, 1]^2 \subset \mathbb{R}^2$. Holes are created by removing all simplices within a specified region and edge weights are set to the Euclidean distance between node positions. We construct the Hodge 1-Laplacian L_1 of the C-space and compute $H = [h_1, ..., h_D]$. Reference paths $\bar{\tau}$ are constructed by connecting set key points by shortest path.

For each experiment we compare the length, and # of nodes visited for $\mathcal{H}^*$, $\mathcal{RH}^*$, $\mathcal{PRH}^*$ and BLK [7] [1]. All experiments were run on a 1.1 GHz Quad-Core Intel Core i5 processor. Per node visit, on average, $\mathcal{H}^*$, $\mathcal{RH}^*$ and $\mathcal{PRH}^*$ take 0.001 seconds while BLK takes 0.002 seconds.

A. Illustrative Example

We provide a practical example on an SC with five holes, consisting of 316 nodes. Figure 2(a) shows $\mathcal{H}^*$ results for several α values (in green), given the S-shaped reference path (in blue) with source and destination nodes at the bottom-left and top-right, respectively. We plot the BLK path (in orange) that is homologous to the $\mathcal{H}^*$ path for a given α. Paths produced by $\mathcal{H}^*$ for $\alpha = 0.0, 0.5, 1.6$ and 2.0 belong to distinct homology classes, the last of which is homologous to $\bar{\tau}$. For $\alpha \in \{0.0, 0.5, 1.6, 2.0\}$, the $\mathcal{H}^*$ and BLK paths are equal in length. To achieve this, $\mathcal{H}^*$ visits $315, 315, 315$ and 251 nodes, respectively, whereas BLK visits $14,269$ nodes – two orders of magnitude more visits to achieve the same results.

In addition to efficiency gains, $\mathcal{H}^*$ provides an interpretable interface for producing paths that are increasing similar to $\bar{\tau}$, as measured by $\Delta\gamma(\tau, \bar{\tau})$, as α increases. This ordering allows users to use α to trade off between path

[1]Experiment code available here: https://github.com/ctaveras1999/h-star

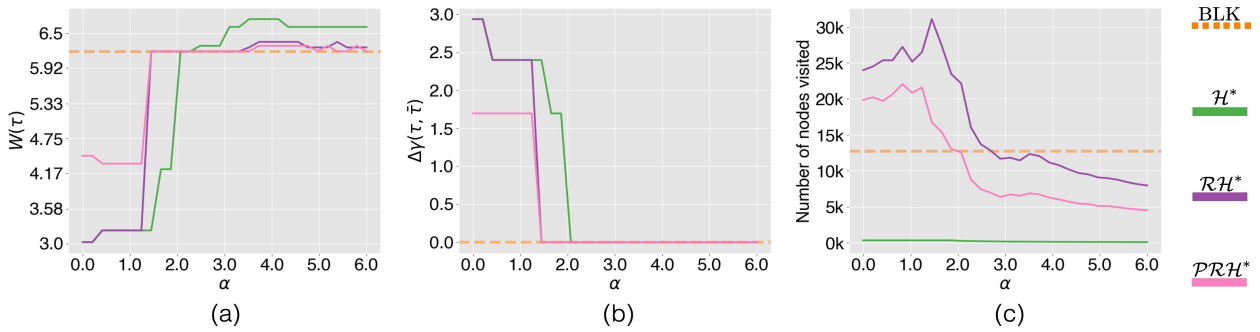

Fig. 3. (a) Path length, (b) projection difference, and (c) number of nodes visited for $\mathcal{H}^*, \mathcal{RH}^*$, and $\mathcal{PRH}^*$ as a function of α. The color-coded (and labeled) horizontal lines each correspond to a distinct homology class in (a) denoting the classes' shortest path length and (b) the projection difference. The horizontal line in (c) shows the number of nodes visited by BLK.

length and topological similarity to the reference path. The $\mathcal{H}^*$ path for $\alpha = 1.6$ has length 4.2 whereas the solution to P1, which is achieved by $\mathcal{H}^*$ for $\alpha = 2$, has length 6.2. If the user is able to ignore the loop around the top left hole, they are provided a similar, but much shorter route, to the destination.

For $\alpha \in \{5, 7\}$, the $\mathcal{H}^*$ paths are homologous to $\bar{\tau}$ both with length 6.6, providing examples of non-optimal paths produced by $\mathcal{H}^*$. These results are achieved after visiting only 86 and 63 nodes, respectively, which is about a quarter of the SC's 316 nodes and two orders of magnitude fewer than BLK. Such a trade-off can be justified for applications in which the speed of a planner is critical, as is the case in many real-time systems.

The larger α is, the earlier $\mathcal{H}^*$ visits nodes with smaller projection difference. Due to this and the sequential structure of the harmonic projections (e.g., Fig. 1(d)), $\mathcal{H}^*$ builds paths to nodes that are closer in harmonic projection at the expense of optimality in path length. Figure 2(b) demonstrates this phenomenon, showing that the harmonic projection difference decreases (or remains unchanged) at each visited node, as α goes from 0 to 2.0. Increasing α arbitrarily promotes increasingly greedy behavior towards minimizing projection difference. This behavior can be seen in Fig. 2(b) as α transitions from 2.5 to 5.0 to 7.0.

B. Characterizing the Effect of α

In this next experiment, we aim to characterize the effect of the parameter α on the behavior of $\mathcal{H}^*$, $\mathcal{RH}^*$, and $\mathcal{PRH}^*$. The SC and reference path in this experiment are identical to that of the first (i.e. Fig. 2(a)). We sample thirty evenly spaced points from 0 to 6 for α and use them to produce paths with $\mathcal{H}^*$ (in green), $\mathcal{RH}^*$ (in purple) and $\mathcal{PRH}^*$ (in pink). Figure 3 (a), (b) and (c), respectively plot the length, projection difference, and number of nodes visited by the algorithms. The (orange) horizontal lines in 3 correspond to the output of BLK (which is independent of α).

For $0 \leq \alpha \leq 2$, the $\mathcal{H}^*$, $\mathcal{RH}^*$, and $\mathcal{PRH}^*$ results are stable (i.e. remain in the same homology class) over several ranges, eventually jumping to homology classes which are closer to that of the reference, at the cost of increased path length. Moreover, for $\alpha < 2$, as shown in Fig. 2(a), $\mathcal{H}^*$ produces a shortest path in each homology class leading up to the desired one. Around $\alpha = 2$, each of the algorithms produce a shortest path in the desired homology class. Figure 3(c) shows that, across α, $\mathcal{H}^*$ visits the least amount of nodes among all methods (315 at most), while achieving results comparable to BLK for each homology class $\mathcal{H}^*$ produces solutions to. From $\alpha < 2$, BLK requires the second least amount of node visits at about 19k, followed by $\mathcal{PRH}^*$, then $\mathcal{RH}^*$ ranging between 30k and 15k. As α increases, the number of nodes visited by $\mathcal{PRH}^*$ and $\mathcal{RH}^*$ trends downwards, requiring less node visits than BLK for $\alpha > 2$. The reason for this is explained in the previous experiment and can be seen in Fig. 2(b).

Importantly, $\mathcal{RH}^*$ and $\mathcal{PRH}^*$ produce shorter length paths than $\mathcal{H}^*$ does for all α in which the algorithm outputs are homologous, as can be seen in Fig. 3(a), demonstrating the utility of the rollout procedure. Finally, Fig. 3(c) shows that node pruning, as proposed in Sec. IV-B, can reduce the number of nodes visited when using rollout.

C. Characterizing the Effect of the Number of Holes

This final experiment studies how the number of holes in an SC affects the number of nodes each algorithm visits to produce a path homologous to the reference. We show that the number of nodes visited by BLK can scale with the number of holes, while our algorithms do not.

Towards this, we generate nine SCs, with between one and nine holes each with fixed hole location and reference path. Examples of these SCs and reference paths can be seen in Fig. 4(c). Figure 4(a) and (b) respectively show the path lengths and number of node visits for each algorithm. For $\mathcal{H}^*, \mathcal{RH}^*$ and $\mathcal{PRH}^*$, the paths were produced by sampling twenty α values between 0 and 3, and choosing the path in the desired homology class with shortest length.

In all cases, the $\mathcal{H}^*$ algorithms produce optimal shortest paths while maintaining a relatively constant number of node visits for each SC. For the first four SCs, $\mathcal{H}^*$ and BLK visit a comparable amount of nodes, while $\mathcal{RH}^*$ and $\mathcal{PRH}^*$ visit significantly more. For each hole added after the fourth, the number of nodes visited by BLK increases almost exponentially. This is because each time a hole is added, short paths that had previously been homologous no longer are. Consequently, the number of homologically distinct paths between v_i and v_f that have shorter length than

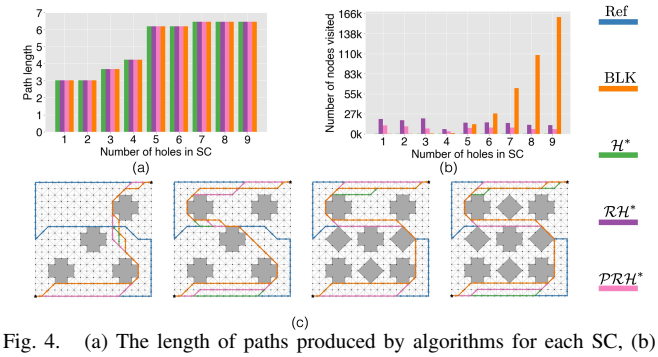

Fig. 4. (a) The length of paths produced by algorithms for each SC, (b) number of nodes visited to produce the paths, and (c) example paths.

that of the desired path can increase as the number of holes increases. For instance, in the five, six, and seven hole cases, BLK respectively produces $102, 252$, and 675 homologically distinct shortest paths before producing the desired path.

The $\mathcal{H}^*$ algorithm, on the other hand, does not scale with the number of holes as it visits each node once, at most. Moreover, $\mathcal{RH}^*$ and $\mathcal{PRH}^*$ scale with the number of nodes in a path and the size of the nodes' neighborhoods, which is unrelated to the number of holes. As such, the $\mathcal{H}^*$-based algorithms can be used as efficient alternatives to BLK in many-obstacled environments.

VI. Conclusion

We present $\mathcal{H}^*$, an efficient heuristic algorithm for solving a relaxation of the problem of finding the shortest path homologous to some user-provided reference path. We show that $\mathcal{H}^*$ can be useful for suggesting paths in homology classes that are similar to, but with shorter shortest path than, the reference class. $\mathcal{RH}^*$ and $\mathcal{PRH}^*$ are introduced as rollout-based variants of $\mathcal{H}^*$ that can improve its results. Experiments demonstrate that $\mathcal{H}^*$ can produce results that are often comparable to [7] and at a significantly reduced computational cost, especially for environments with many obstacles or holes present.

References

[1] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[2] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, and W. Burgard, *Principles of Robot Motion: Theory, Algorithms, and Implementations*, ser. Intel. Rob. and Aut. Agents series. MIT Press, 2005.

[3] W. Wu, S. Bhattacharya, and A. Prorok, "Multi-robot path deconfliction through prioritization by path prospects," in *2020 IEEE Int. Conf. on Rob. and Aut. (ICRA)*, 2020, pp. 9809–9815.

[4] K. Kolur, S. Chintalapudi, B. Boots, and M. Mukadam, "Online motion planning over multiple homotopy classes with gaussian process inference," in *2019 IEEE/RSJ Int. Conf. on Intel. Rob. and Sys. (IROS)*. Macau, China: IEEE, Nov. 2019, p. 2358–2364.

[5] E. Hernandez, M. Carreras, and P. Ridao, "A path planning algorithm for an auv guided with homotopy classes," *Proc. of the Int. Conf. on Aut. Planning and Scheduling*, vol. 21, p. 82–89, Mar. 2011.

[6] A. Hatcher, *Algebraic Topology*. Cambridge: Cambridge University Press, 2002.

[7] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Aut. Rob.*, vol. 33, no. 3, p. 273–290, Oct 2012.

[8] J. Hershberger and J. Snoeyink, "Computing minimum length paths of a given homotopy class," *Comp. Geo.*, vol. 4, no. 2, pp. 63–97, 1994.

[9] B. Chazelle, "A theorem on polygon cutting with applications," in *23rd Ann Symp. on Found. of Comp. Sci. (sfcs 1982)*, 1982, pp. 339–349.

[10] E. C. de Verdière and J. Erickson, "Tightening nonsimple paths and cycles on surfaces," *SIAM Journal on Computing*, vol. 39, no. 8, pp. 3784–3813, 2010.

[11] E. W. Chambers, J. Erickson, and A. Nayyeri, "Minimum cuts and shortest homologous cycles," in *Proc. of the 25th Ann Symp. on Comp. Geo.*, ser. SCG '09. New York, NY, USA: ACM, 2009, p. 377–385.

[12] T. K. Dey, T. Li, and Y. Wang, "Efficient algorithms for computing a minimal homology basis," in *LATIN 2018: Theoretical Informatics*. Springer Int. Publishing, 2018, pp. 376–398.

[13] S. Bhattacharya, D. Lipsky, R. Ghrist, and V. Kumar, "Invariants for homology classes with application to optimal search and planning problem in robotics," *Annals of Mathematics and Artificial Intelligence*, vol. 67, no. 3–4, p. 251–281, Mar 2013.

[14] M. Kuderer, C. Sprunk, H. Kretzschmar, and W. Burgard, "Online generation of homotopically distinct navigation paths," in *2014 IEEE Int. Conf. on Rob. and Automation (ICRA)*. Hong Kong, China: IEEE, May 2014, p. 6462–6467.

[15] D. Yi, M. A. Goodrich, and K. D. Seppi, "Homotopy-aware rrt*: Toward human-robot topological path-planning," in *2016 11th ACM/IEEE Int. Conf. on Human-Robot Interaction (HRI)*, 2016, pp. 279–286.

[16] E. Schmitzberger, J. Bouchet, M. Dufaut, D. Wolf, and R. Husson, "Capture of homotopy classes with probabilistic road map," in *IEEE/RSJ Int. Conf. on Intel. Rob. and System*, vol. 3. Lausanne, Switzerland: IEEE, 2002, p. 2317–2322.

[17] R. Sandstrom, D. Uwacu, J. Denny, and N. M. Amato, "Topology-guided roadmap construction with dynamic region sampling," *IEEE Rob. and Aut. Letters*, vol. 5, no. 4, p. 6161–6168, Oct. 2020.

[18] R. W. Ghrist, *Elementary Applied Topology*. Createspace Seattle, 2014, vol. 1.

[19] K. Crane, F. de Goes, M. Desbrun, and P. Schröder, "Digital geometry processing with discrete exterior calculus," in *ACM SIGGRAPH 2013 courses*, ser. SIGGRAPH '13. New York, NY, USA: ACM, 2013.

[20] V. Nanda, "Computational algebraic topology lecture notes," *URL: https://people.maths.ox.ac.uk/nanda/cat/TDANotes.pdf*, 2021.

[21] L. J. Grady and J. R. Polimeni, *Discrete Calculus*. London: Springer London, 2010.

[22] L.-H. Lim, "Hodge laplacians on graphs," *SIAM Review*, vol. 62, no. 3, pp. 685–715, 2020.

[23] P. D. Lax, *Linear Algebra and Its Applications*, 2nd ed. Hoboken, NJ: Wiley-Interscience, 2007.

[24] M. T. Schaub, A. R. Benson, P. Horn, G. Lippner, and A. Jadbabaie, "Random walks on simplicial complexes and the normalized hodge 1-laplacian," *SIAM Review*, vol. 62, no. 2, pp. 353–391, 2020.

[25] A. Ghosh, B. Rozemberczki, S. Ramamoorthy, and R. Sarkar, "Topological signatures for fast mobility analysis," in *Proc. of the 26th ACM SIGSPATIAL Int. Conf. on Adv. in Geo. Inf. Sys.*, 2018, pp. 159–168.

[26] M. T. Schaub, Y. Zhu, J.-B. Seby, T. M. Roddenberry, and S. Segarra, "Signal processing on higher-order networks: Livin' on the edge... and beyond," *Signal Proc.*, vol. 187, p. 108149, Oct 2021.

[27] X. Yin, C.-C. Ni, J. Ding, W. Han, D. Zhou, J. Gao, and X. D. Gu, "Decentralized human trajectories tracking using hodge decomposition in sensor networks," in *Proc. of the 23rd SIGSPATIAL Int. Conf. on Adv. in Geo. Inf. Sys.* Seattle Washington: ACM, Nov. 2015, p. 1–4.

[28] T. M. Roddenberry and S. Segarra, "HodgeNet: Graph neural networks for edge data," in *Asilomar Conf. on Sig., Sys., and Comp.*, 2019, pp. 220–224.

[29] J. Jia, M. T. Schaub, S. Segarra, and A. R. Benson, "Graph-based semi-supervised & active learning for edge flows," in *ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*, 2019, p. 761–771.

[30] T. M. Roddenberry, N. Glaze, and S. Segarra, "Principled simplicial neural networks for trajectory prediction," in *Proc. of the 38th Int. Conf. on Machine Learning*, ser. Proc. of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, Jul 2021, pp. 9020–9029.

[31] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. on Sys. Science and Cyb.*, vol. 4, no. 2, pp. 100–107, 1968.

[32] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[33] ——, *Dynamic Programming and Optimal Control*, 4th ed., ser. Athena scientific optimization and computation series. Athena Scientific, 2012.

[34] ——, "Rollout algorithms for constrained dynamic programming," *Lab. for Inf. and Dec. Sys. Report*, vol. 2646, 2005.