A State-of-the-Art Review on Topology and Differential Geometry-Based Robotic Path Planning—Part II: Planning Under Dynamic Constraints

Sindhu Radhakrishnan[1] and Wail Gueaieb[1*]

[1*]School of Electrical Engineering and Computer Science, University of Ottawa, 800 King Edward Avenue, Ottawa, K1N 6N5, ON, Canada.

*Corresponding author(s). E-mail(s): wgueaieb@uottawa.ca; Contributing authors: sradh006@uottawa.ca;

Abstract

Path planning is an intrinsic component of autonomous robotics, be it industrial, research or consumer robotics. Such avenues experience constraints around which paths must be planned. While the choice of an appropriate algorithm is application-dependent, the starting point of an ideal path planning algorithm is the review of past work. Historically, algorithms were classified based on the three tenets of autonomous robotics which are the ability to avoid different constraints (static/dynamic), knowledge of the environment (known/unknown) and knowledge of the robot (general/model specific). This division in literature however, is not comprehensive, especially with respect to dynamics constraints. Therefore, to remedy this issue, we propose a new taxonomy, based on the fundamental tenet of characterizing space, i.e., as a set of distinct, unrelated points or as a set of points that share a relationship. We show that this taxonomy is effective in addressing important parameters of path planning such as connectivity and partitioning of spaces. Therefore, path planning spaces may now be viewed either as a set of points or, as a space with structure. The former relies heavily on robot models, since the mathematical structure of the environment is not considered. Thus, the approaches used are variants of optimization algorithms and specific variants of model-based methods that are tailored to counteract effects of dynamic constraints. The latter depicts spaces as points with interconnecting relationships, such as surfaces or manifolds. These structures allow for unique characterizations of paths using homotopy-based methods. The goals of this work, viewed specifically in light with dynamic constraints, are therefore as follows: First, we propose an all-encompassing taxonomy for robotic path planning literature that considers an underlying structure of the space. Second, we provide a detailed accumulation of works that do focus on the characterization of paths in spaces formulated to show underlying structure. This work accomplishes the goals by doing the following: It highlights existing classifications of path planning literature, identifies gaps in common classifications, proposes a new taxonomy based on the mathematical nature of the path planning space (topological properties), and provides an extensive conglomeration of literature that is encompassed by this new proposed taxonomy.

Keywords: Path planning, Robotics, Manifolds, Topology

Abbreviations

APF Artificial Potential Field
CDGT Cell Decomposition and Graph Traversal
DC Dynamic Constraint
EA Evolutionary Algorithm
*HA** Homotopic A*
HB Homotopic Bug
HBM Homotopy Based Method
HCM Homotopy Continuation Methods
HRRT Homotopic RRT
LAG L-Augmented Graph
MBM Model Based Methods
MPC Model Predictive Control
NAES Non-linear Algebraic Equation System
NF Navigation Functions
OA Optimization Algorithm
PPM Path Planning Manifold
PPP Path Planning Problem
PPS Path Planning Space
PRM Probabilistic Road Map
RRT Rapidly exploring Random Tree
SA Sampling Algorithm
SC Static Constraint
SHIO Single Homotopy Inducing Obstacle
VOS Velocity Obstacle Sets

List of Latin Symbols

H-signature Unique measure associated to a homotopic class
$\boldsymbol{p}^{start} \in \mathcal{P}$ Starting point in $\mathcal{P}$
$\boldsymbol{p}^{goal} \in \mathcal{P}$ Destination point in $\mathcal{P}$

1 Introduction

The solutions to Path Planning Problems (PPPs) in robotics depend on the application at hand and some general requirements. For example, one may consider an automation chain in a factory or a semi/fully-autonomous warehouse. Here, robots have to be aware of their surroundings and must meet real-time constraints to interact with humans within their working environment while avoiding collisions with them and any obstacles that may be on the way, such as other autonomous robots, for instance [1]. Naturally, the plethora of path planning literature can be assessed for a suitable choice of a path planning algorithm. Assessing literature shows that such algorithms may be segmented based on certain parameters.

This work is the second in a series of two articles that comprehensively explore path planning algorithms that are centered around topology and differential geometry-based methods. Tackling the problem from this perspective distinguishes our survey from previous works and makes it unique. The first paper in this series [2] focuses on path planning techniques under static constraints, while this article covers techniques that are more suitable for dynamic constraints. Here, we propose that path planning literature may be classified based on the nature of the Path Planning Spaces (PPSs) and whether they allow paths to be characterized uniquely around constraints. We do so specifically for path planning algorithms that deal with Dynamic Constraints (DCs), by first briefly discussing the taxonomies most commonly seen in literature. Since the demerits of existing taxonomies preclude the analysis of the nature of the PPS and the ability to characterize uniqueness of paths, we propose a classification that remedies these demerits. The common classifications is described first, following which the demerits are provided.

The three common taxonomies that divide the path planning literature are as follows: The first category classifies approaches based on whether or not the path planning environment is known apriori. Known environments use as graphs, occupancy grids and any Cell Decomposition and Graph Traversal (CDGT) [3–11] techniques. On the other hand, unknown environments rely on Sampling Algorithms (SAs), such as Probabilistic Road Maps (PRMs) [12–20], Rapidly exploring Random Trees (RRTs) [6–8, 12, 13, 15, 21–28] and other variants of both. Path planning algorithms generally consist of two tasks. The first is generating a path segment towards the goal and the second is evaluation of the path segment for collision avoidance. The second category distinguishes algorithms based on which of the two tasks is optimized for path metrics. For example, Model Based Methods (MBM) [29–31] methods produce model specific paths and then evaluate the produced paths for collision.

Contrarily, force field methods, such as Artificial Potential Fields (APFs) [32–39] and Navigation Functionss (NFs) [40–45] focus on producing collision-free paths, which can later be processed for the robot models' specifics. The third classification separates algorithms based on their ability to avoid Static Constraints (SCs) [3–5, 32, 45] or DCs [46–50]. Avoiding static constraints can be viewed as planning with knowledge of constraints, so it is deliberate collision avoidance. On the other hand, impromptu encounters of unknown static or dynamic constraints prompts the algorithm to reactively avoid them. Usually, this classification consists of variants of algorithms belonging to the first two categories. The classifications may be viewed pictorially in Fig. 1. Clearly, all classifications may share some algorithms, while also possessing some variants that are unique to the classification. The aforementioned distinctions in categories enable the choice of a suitable path planning algorithm for a specific application. However, viewing the PPP via the three categories has its advantages and disadvantages, which are now discussed.

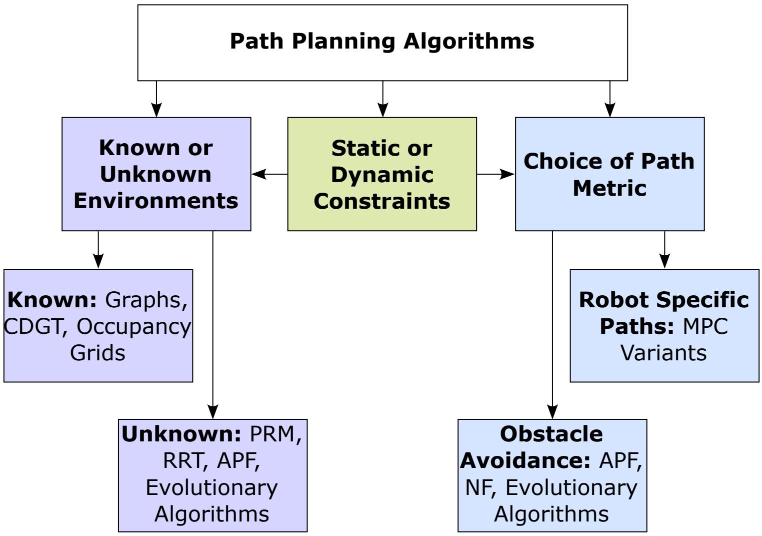


Fig. 1: Three common taxonomies in path panning literature

The three categories enable one to look at the PPP via different resource lenses. One may choose a suitable path planner based on the abundance or dearth of map information, or the precedence of robot specific path segments over the fastest collision avoidance path. However, the classifications still share a significant overlap of approaches, while missing an important branch of path planning literature. To minimize the overlap and make the classification more intuitive from the grassroots level, we propose a classification based on the nature of the PPS. That is, the PPS may be categorized as: (i) not having an explicit mathematical structure, such as a set of points, or (ii) possessing a mathematical structure of smoothness, called topological spaces, such as manifolds. Now, using this taxonomy, the path planning literature can be seen to consist of approaches, such as SAs and Optimization Algorithms (OAs) for the former, and topology-based methods for the latter. This taxonomy is illustrated in Fig. 2. Following such taxonomy then, one can easily insert the original three classifications under the respective division of point-set or manifold-based methods, if such a detailed branch structure is desired.

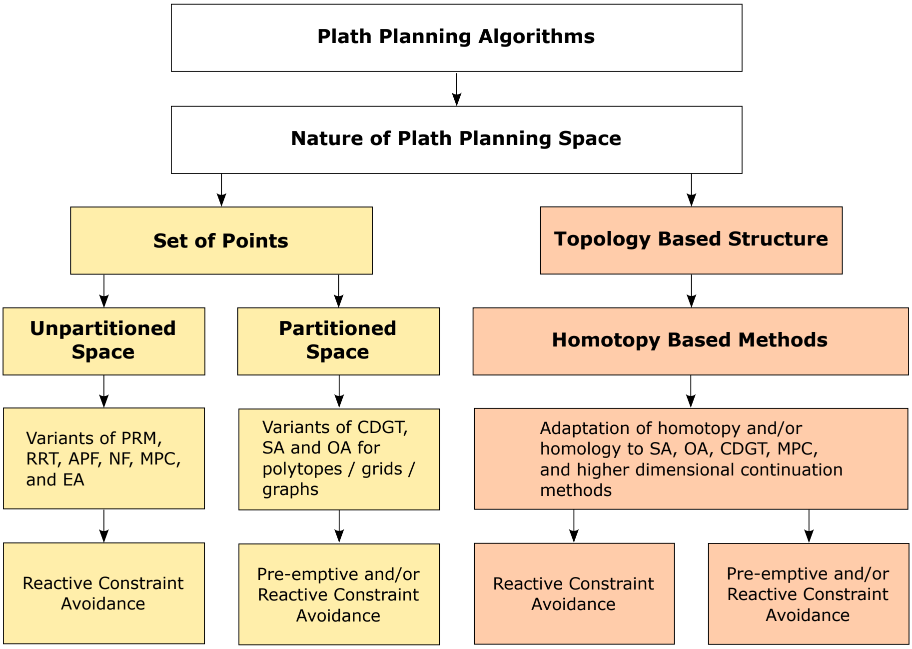


Fig. 2: Taxonomy based on the nature of the PPS

Path planning needs to account for DCs. A DC can be a sudden appearance of a SC that was not known apriori, such as a falling object. Alternatively, it can be the appearance of a moving constraint into the region of the path, where the DC's location at the time of path planning did not interact with the planned path. Such an example can be seen with the motion of another robot meandering into the planned path, or with the encounter of a human being whose presence was not accounted for previously. Using the new taxonomy based on the nature of the PPS, path planning techniques that avoid DCs can broadly be classified into two main approaches. The first group of methods assumes knowledge of the robot's model and/or of the DCs, and performs path planning predominantly (but not exclusively), as a local response. This group of methods is referred to as MBMs in this article. The second group assumes a higher level understanding of the problem and plans/modifies the path as a response to the global path planning problem, without requiring any knowledge of the specific robot model at hand. Typically, this group focuses on path deformation by borrowing concepts from manifold theory, called homotopy deformation; therefore, being referred to as Homotopy Based Methods (HBMs). It can be seen that MBMs typically encompass the methods seen in SAs and OAs, which are typically used when the PPS is modelled as a set of points. Similarly, HBMs are applicable when the PPS can be attributed an underlying topological representation.

This proposed classification is general and encompasses all other traditional classifications as it is based on the nature of the path planning space. Subsequently, it segments the literature into MBMs and HBMs. We provide a comprehensive literature review of those path planning algorithms building on concepts from the latter in light of handling DCs. This effort aims to minimize the gap in robotics literature pertaining to such a specific review of works in the area.

The paper is organized as follows: A brief introduction to MBMs is provided in Section 2. Following this, Section 3 details the approaches pertaining to HBMs. Finally, Section 4 concludes the literature review. A tabulated overview of the proposed taxonomy with works that fall into MBMs and HBMs is seen in Table 1.

Table 1: Proposed taxonomy: Segmentation into MBMs and HBMs

Segmentation Category	OA	CDGT+Model Based Solutions	Analytical Solutions	SA	Recursive Solutions
Model Based Methods	[29–31, 51–61]	[46–50, 62–67]	-	-	-
Homotopy Based Methods	[68–71]	[72–75]	[76–81]	[82–91]	[77–80, 92, 93]

2 Path Planning with Model-Based Methods

MBMs rely on assumed models of the robots, constraints to be avoided and the environment, to plan a path free of said constraints. They may do so either by solving the problem globally or as a reactive mechanism. Global methods that avoid obstacles in motion are usually adaptations of SAs and OAs. Their being global typically implies that knowledge of the number, potential size, initial locations and maximum velocities of DCs are available or may be inferred. Then, paths are planned to avoid the assumed/predicted worst case scenario, which is that of a collision. On the other hand, local planners are reactive in nature. That is, without assuming too much knowledge of the constraints to avoid, these planners only react when a collision is indicated to happen at a user/algorithm defined proximity. These locally reactive algorithms, such as model based control algorithms, graph based solvers, reachability and velocity sets of obstacles, and methods seen as part of OA and SA, are heavily reliant on the quality of sensor data. They also usually resort to some variant of obstacle boundary following, until a clear line of sight towards the target is visible.

Naturally, global and local reactive planners may both be reliant on models of the robot used, constraints expected and the environment, thus giving rise to a variety of model based algorithms. The primary advantage of using model based approaches is that paths generated are suited to be executed for exactly the kind of robot that it was designed for. So the next stage of trajectory tracking becomes easier. The obvious disadvantage however, is that the solutions are restricted to the models defined. Augmentations to MBMs take the form of robot/constraint specific graph based approaches, or alternatively, in the form of OAs, SAs and Evolutionary Algorithms (EAs). The goal of using any optimization based approach in conjunction with MBMs is to combat the issue of solution specificity offered by undiluted MBMs. It is for this reason, that the literature pertaining to MBMs may be segmented generally into two categories: OA based MBMs and core MBMs. Since the focus of this article is HBMs, the literature review pertaining to MBMs will not be elaborated much here. Instead, one may refer to Fig. 2 for a quick introduction to common methods associated with MBMs and to the following for an in-depth understanding.

- OA based MBMs: sampling and learning algorithms [29–31, 51, 52], variants of APFs based approaches in [31, 53–61].
- Core MBMs: grid based design in [62], velocity disks, cones, tangents and Velocity Obstacle Sets (VOS) in [46–50, 63–65]. Note that pure graphical methods that could be used for most models with minor tweaking, such as A*, Dynamic A*, Theta* and other such variants are seen in [66, 67]. They are included here, since they can be modified to suit the model of the robot and the constraints.

3 Homotopy Based Methods

Methods seen in Section 2 are adaptations of general path planning methods adapted to the problem of DC avoidance, with no specific structural assumption of the path or PPS. An alternative class of methods based on concepts of homotopy and homology exists, which will be referred to as HBMs and remains the focus of this section. HBMs rely on the fact that paths planned in a space with constraints follow a certain structure. Roughly explained, the structure shows how the path crosses the region between obstacles, i.e., the sequence in which the crossing occurs and the kinds of turns the path takes while traversing the environment. Knowing the structure of the path helps generate other paths with the same structure, or alternatively, can help choose a different structure of paths. This structure that defines the nature or deformability of the path is called the homotopy class of the path. Having understood the premise of using homotopy classes, one can see that the literature in the area may be distinguished into the following categories.

The first category focuses on the theory behind recognizing and computing homotopy classes for path planning applications, as seen in Section 3.1. This theoretical background constitutes a separate category of literature, as its usage only slowly forayed into robotics. This theory is what propelled other augmentations and discretizations of HBMs to come to fruition. Section 3.2 contains the second category of methods that utilize SAs in conjunction with homotopy classes for path planning and/or exploration. The third category is discussed in Section 3.3, where methods are seen to focus on recursive strategies to generate homotopy classes, as opposed to analytical methods. In Section 3.4, computing paths in homotopy classes are accomplished using optimization based solutions. Finally, the fifth category of methods is analyzed in Section 3.5, where MBM based solutions are seen to better tune homotopy classes.

3.1 Analytical Solutions

The problem of testing for homotopy paths on the plane has existed as a problem in computational geometry [76], and has only recently started moving into the world of robotics for path planning. Works such as [77–80] predominantly focus on identifying and representing homotopy classes for path planning. Authors of [79] succinctly define homotopic trajectories as the following: Two trajectories τ_1 and τ_2 connecting the same $\mathbf{p}^{\text{start}}$ and $\mathbf{p}^{\text{goal}}$ are called homotopic if and only if one can be continuously deformed into the other without intersecting any obstacle, as seen in Fig. 3. Then, sets of homotopic trajectories form homotopy classes. The different homotopy classes are seen in Fig. 3, where the difference in the classes correspond to how they navigate through obstacles. So, homotopy classes can have different implications when being evaluated for path planning. That is, path planners may simply desire to generate a path that is constraint-free, in which case any of the classes may be picked from. Alternatively, a path planner may also require to avoid certain classes due to some heuristic measure, such as the probability of encountering an increased number of DCs, general path length and the class distance from SCs that it encounters along the way. Clearly, there is a benefit to using homotopy classes. One may check if two arbitrary paths are homotopic, or equivalently, verify what homotopic class the path falls into. This however is a challenging problem. One needs to construct such a measure of deformability of every homotopy class, such that paths in the same class bear the same measure, as opposed to paths from other classes.

Characterizing this measure of path deformability around a constraint and developing a method used to compute it forms the theoretical basis of analytical HBMs. This is now explained in some detail.

The authors of [79] propose constructing a functional (linear one-form) for a trajectory, such that its value uniquely identifies the homotopy class. This functional of the form $\mathcal{H}(\tau)$ is called the H-signature of the trajectory τ. It is beneficial that the H-signature takes the form of an integration as such,

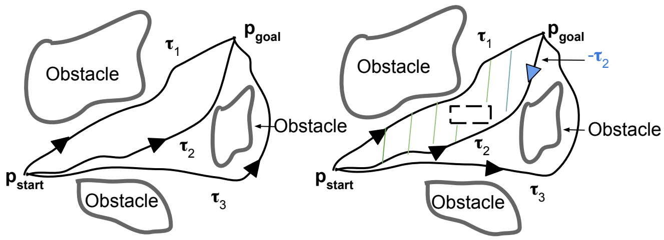


Fig. 3: (a) τ_1 and τ_2 are homotopic paths; τ_1 is not homotopic to τ_3. Thus, there are two homotopic classes, one denoted by τ_1 and another, denoted by τ_3; (b) τ_1 and $-\tau_2$ are homologous.

$\mathcal{H}(\tau) = \int_\tau \mathrm{d}h$, where $\mathrm{d}h$ is also a linear one-form. This way, the chosen one-form may be a heuristic, such as path length, and may be integrated along the path. The authors of [79] compute one such H-signature in their earlier work [77] using complex analysis. In fact, one may rely on the work from [77] to comprehend the use of homotopy classes in path planning, and an excerpt of the work is explained. This explanation is only meant to serve as a high level interpretation of homotopy classes and their relevance to constraint avoidance, and not as a formal mathematical proof. Readers are directed to [77–80] for any formal statements of the concepts explained here.

Across all their works, the authors of [77–80] focus on the following three goals. First, the homotopy and/or homology classes in a PPS are required to be detected. Second, specific mathematical tools are required to represent constraints that need to be avoided and a measure that indicates the homotopic class in which the test path currently lies. Third, this homotopic measure of a path needs to be integrated with path planning for the path planner's needs. In order to accomplish the first goal, the authors of [77] use holomorphic functions (analytic functions in the complex plane) to formulate the path and the PPS in the complex domain. In the complex plane $\mathbb{C}$, assume a simply connected region $\mathcal{R}$ within which there is a closed and oriented contour γ that encloses a point z_0 and a holomorphic function $f : \mathbb{C} \to \mathbb{C}$. Then, the Cauchy Integral theorem affords the following:

$$\oint_\gamma f(z)dz = 0 \quad \text{and} \quad \oint_\gamma \frac{f(z)}{z - z_0} dz = 2\pi i f(z_0)$$

That is, the effect of the holomorphic function f on every point of the path γ, around a point z_0, can be represented as a holomorphic function (acting on the path) with a pole at z_0. This pole indicates that the function is holomorphic everywhere except at the pole. The Residue theorem provides an expression to quantify the effect of a holomorphic function F with M poles, on a contour

γ, enclosing only $m < M$ poles.

$$\oint_\gamma F(z)dz = 2\pi i \sum_{l=1}^{m} \lim_{z \to a_l} (z - a_l) F(z)$$

This shows that the value of the integral of F is solely dependent on the points that the path γ encloses, as seen in Fig. 4. This is of consequence in the context of the PPP, since obstacles/constraints can be represented as such poles. The homotopy class within which the path falls, is entirely dependent on the poles which the path encloses. Recall that homotopic trajectories are defined as continuously deformable paths, without intersecting obstacles. So, if one can define the obstacles as the poles of $F(z)$, then the homotopic classes can be defined. The authors of [77] do so by defining an obstacle marker function as follows:

$$F(z) = \frac{f_0(z)}{(z-\xi_1)(z-\xi_2)\dots(z-\xi_N)} \tag{1}$$

where f_0 is analytic over C, and each of the N poles is a representative of each obstacle or constraint to be avoided. The intuitive understanding of the obstacle marker function can be seen with the function

$$f_i(z) = \frac{f_0(z)}{(z-\xi_1)\dots(z-\xi_{i-1})(z-\xi_{i+1})\dots(z-\xi_N)} \tag{2}$$

defined for each of the N poles. Using $f_i(z)$ in (1) results in $f_i(z) = (z - \xi_i)F(z)$, which is analytic inside regions that do not contain the poles of (2). This, in conjunction with the integral being influenced only by the poles that the path encompasses, provides a way to identify the different homotopic classes of paths in the PPS. The authors of [77] define the homologic signature H-signature, of the path as

$$L(\tau) = \int_\tau F(z)dz \tag{3}$$

for a trajectory $\tau \in C$. In this specific case, the authors refer to the H-signature as the L-value of a trajectory. They define (3) as the unique descriptor of a homotopy class, since the H-signature of trajectories (connecting the same two points) in same homotopy class are equal, while the H-signature of trajectories in different homotopy classes are different. Thus far, the authors of [77] accomplish detecting homotopy classes and representing constraints, initially described as the first two goals. The last goal was to combine such information with path planning methods, for which the authors rely on a graph based representation. They construct an exploratory graph, whose edges and vertices are augmented with the L-value. The L-value of every edge e connecting vertices z_1 and z_2 in the L-Augmented Graph (LAG) is computed either numerically or analytically (as seen in [77]).

As the LAG explores new nodes, the new edges are augmented with L-values. This LAG can be searched for a path using any standard graph traversal

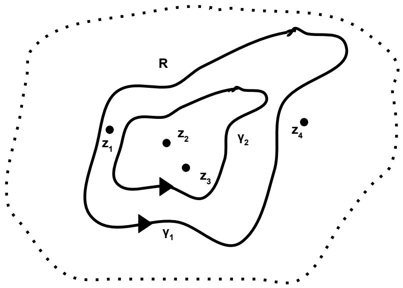


Fig. 4: The integral of a holomorphic function F acting on a closed contour γ depends on the points enclosed by γ. If the closed contour encloses no poles, then the value of the Cauchy Integral is 0, because there is nothing to wind around. On the other hand, if the closed contour encloses some poles, then the contour winds around the poles and does so with a specific direction. The number of poles enclosed and the direction of enclosure is what is quantified using the Cauchy Integral Theorem and consequently, the Residue Theorem. In this example, since γ_1 and γ_2 enclose different numbers of poles, their evaluations using the Residue Theorem will provide different results.

technique. Since every homotopy class has a distinct complex number associated with it, the LAG may be equipped with homotopy classes to either specifically navigate in, or avoid. The authors show the impact of using A* on an exploratory LAG with three cases: without specifying a homotopy class, choosing a specific class and avoiding all classes. In the second case, blocking a homotopy class prompts the search for a solution in another homotopy class. As the authors point out, the exploration of the homotopy classes can be performed in a single run of graph search. Finally, in the third case, where both homotopy classes are blocked, the search is biased towards finding paths in different homotopy classes in the order of their path costs. Such paths may include non-Jordan (self-intersecting) curves. It is important to note that the *H*-signature computed in the LAG are complete invariants for homology classes as opposed to homotopy classes [79]. For the purposes of path planning, homology and homotopy may be considered to provide the same information. Two paths τ_1 and τ_2 (with an orientation opposing that of τ_1) are homologous if τ_1 and τ_2 together form the boundary of a 2-manifold free of constraints. Referring to Fig. 4, one can see that homotopic trajectories are always homologous, and non-homotopic paths are non-homologous. While the converse may not always be true, the authors of [79] note that homology serves as a fair analog to homotopy in robotic path planning.

The authors of [77] extend their work in 2-dimensions to 3-dimensions in [78]. While the work in 2-dimensions using complex analysis is a compact way of generating homotopy classes that is independent of the geometry of constraints and robust to sensor noise [78], the technique applies only to 2-dimensions. The extension predominantly deals with representing obstacles or constraints as surfaces with genus-1 or more [79]. This is of importance since in a 3-dimensional PPS, constraint regions with genus-1 or more can induce more than one homotopy class. Fig. 5 shows examples of such constraints that permit multiple homotopy classes. This is opposed to the behavior of genus-0 surfaces, such as a sphere or a solid cube, which cannot induce the same behaviour. Then, one can no longer use a representative point for the constraint or obstacle in the obstacle marker function. The authors of [78] propose representing such a constraint by its skeleton. They do so by relying on Biot-Savart's Law and Ampere's Law to define a differential 1-form, the integration of which along trajectories allow them to distinguish between different classes of trajectories. By using skeletons of constraints, the authors of [78] are able to exploit the same objectives from the 2-dimensional case but now in 3-dimensions, which are identification of different homotopy classes and combining them with graph traversal algorithms to compute least-cost paths. The resulting algorithm is still independent of geometry and sensor noise, and can be integrated with graph traversal algorithms. A brief summary of their approach is now explained.

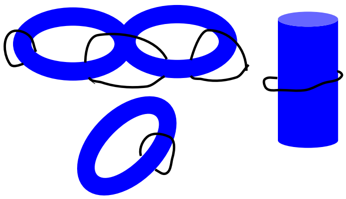


Fig. 5: Surfaces admitting and disallowing multiple homotopy classes

In 2-dimensions, a finite sized obstacle can theoretically induce multiple homotopy classes for trajectories joining two points. However, the notion of homotopy classes in three dimensions can only be induced by obstacles with genus-1 or more, or with obstacles stretching to infinity in two directions. This can be explained with Fig. 5. Note that a path around surfaces without any genus, such as a sphere or a solid cube, can simply be shrunk to a point beyond the size of the surface itself. That is, every path around the surface may be deformed to a point, therefore multiple homotopy classes cannot exist. It is not the case for surfaces with genus-1 or more, since there can exist paths that cannot be deformed into one another. To simplify the concept of homotopy classes and obstacle surfaces in 3-dimensional space, the authors of [78] perform two main tasks. First, they generate a representation in the form of a skeleton for a surface that induces only one homotopy class, called the skeleton of a Single Homotopy Inducing Obstacle (SHIO). A SHIO is defined as a bounded obstacle of genus-1, such as a torus. Subsequently, as a second step, they convert generic obstacles with genus-k into a collection of k SHIOs, each with genus-1. Since the skeleton of the SHIO is used as an equivalent representative point for the obstacle marker in 3-dimensions, one must ensure that the skeleton is a representation that does not alter the topology of the surface. So, the skeleton of a surface is defined as follows. A 1-dimensional manifold, S, is called a skeleton of a SHIO, $\mathcal{O}$, if and only if S is homeomorphic to $\mathbb{S}^1$ (a circle), S is completely contained inside $\mathcal{O}$, and if S and $\mathcal{O}$ are homotopy equivalent. That is, if the obstacle $\mathcal{O}$ is replaced by an equivalent obstacle S, then the homotopy equivalence between two arbitrary trajectories, τ_1 and τ_2, connecting every pair of fixed points in the environment, will remain unchanged [78]. The conversion of genus-k obstacles into k SHIOs, with overlap and each being genus-1, allows k skeletons to be defined for the original obstacle of genus-k. This simplifies the computations of $\boldsymbol{H}$-signature of paths.

The general principle of $\boldsymbol{H}$-signature remains the same irrespective of dimensions. The only difference lies in the principle used to compute it. The authors used complex analysis in the 2-dimensional case. They relied on laws of electromagnetism in the 3-dimensional case. They model the skeleton S of SHIO as a current carrying manifold. This can be performed for every genus-k obstacle, by converting it to k skeletons of SHIO obstacles of genus-1. Next, given every skeleton S_i corresponding to SHIO$_i$, they define a virtual magnetic field vector generated at a point in space, as a result of the current carrying conductor (skeleton). Then, the $\boldsymbol{H}$-signature for one skeleton is defined as the integral of the virtual magnetic field, along the path τ. Should there be M skeletons, then the $\boldsymbol{H}$-signature is a vector consisting of M individual $\boldsymbol{H}$-signatures. Post this step, the approach is similar to that of [77], where an $\boldsymbol{H}$-signature augmented graph is computed. This augmented graph contains allowed and blocked homotopy classes, and any graph traversal algorithm may be used to find a path.

The results of the algorithm as applied to a dynamic environment can be explained as follows. The environment is 2-dimensional, with time as the third dimension. The dynamic obstacles are assumed to move in an oscillatory manner, and their skeletons are traced by the movement of their central representative points. The authors show that the trajectories in the non-trivial homotopy classes go behind the obstacles, a region that would otherwise not be visited by the least cost path without any homotopy class consideration [78].

One of the most recent works in the field of path planning using manifold theory is seen in [81]. The authors of [81] solve the problem of obstacle avoidance by interpolating points on the PPS, which is assumed to be either a Riemannian manifold or a space described by Lie groups. While the method does not explicitly deal with generating paths in homotopy classes, it focuses on admitting only such variations to path segments that comply with the local tangent space. Therefore, the deformations agree with the underlying manifold, and are seen to be $\mathcal{C}^1$), which is not just continuous as required for homotopic classes, but also once continuously differentiable. In the absence of any obstacles, the resulting path is shown to be a geodesic, which satisfies the requirement of the shortest path on a manifold. The quality of deformation of

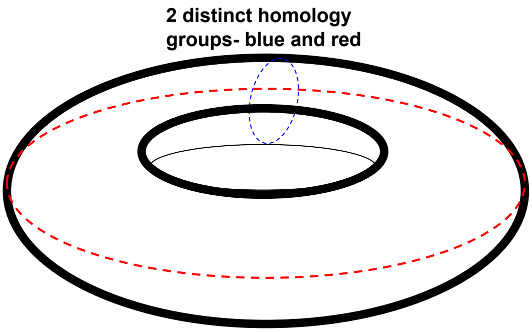


Fig. 6: Distinct Homology Groups Torus

paths is comparable to that of deformations experienced by paths in a homotopic class, since the resulting path still respects the topology of the underlying manifold. Recall from [78] that a homotopy group was induced in the ambient space of the PPS; however, that is not the only homotopy class that can be deduced from the situation. Homology groups that are innate to the surface may also be considered, and these are irrespective of the ambient space, as seen in Fig. 6. This implies that paths or cycles that lie in these groups may be transformed to other non-canonical cycles in the same groups. The resulting cycles or paths still lie on the surface of interest. This digression from the content in [81], is to show that the deformation of path segments therein ensures that the resulting path segment lies on the tangent space of the manifold, and thus is as important to consider as are other works on homotopic classes of paths. The work in [81] is now briefly explained. The premise of the general problem being solved here, is the dynamic interpolation of path segments on the tangent space of a Riemannian manifold. Therefore, it is assumed that a standard metric on the tangent space is defined, and the path segments conform to being C^1 piecewise smooth curves. The path segments obeying the boundary conditions on the tangent space of the manifold are identified as a class of curves, which collectively admits a manifold structure, called the admissible manifold ω. The deformation or interpolation of these path segments are explained with the help of the curvature tensor defined as follows. On a given Riemannian manifold, the influence of a vector field on a curve is given by the covariant derivative of that vector field on the curve. When there are several vector fields acting on the curve, the combined influence of all these vector fields on the curve is given by the curvature tensor, which is a function of all the covariant derivatives of the vector fields and their lie brackets.

For the class of curves in ω, the authors of [81] introduce the notion of a C^1 piecewise smooth one-parameter admissible variation α of a curve $x \in \omega$. The admissible variation α is characterized infinitesimally by the C^1 piecewise smooth vector field along the curve x, called the variational vector field. Due to the boundary conditions imposed, the variational vector field belongs to the tangent space of the curves, as originally defined. This one-parameter variation satisfies a relationship involving the curvature tensor. Then, the authors define a functional J to be minimized to solve the problem defined. The functional J is given by a weighted combination of the velocity and covariant acceleration of the curve x regulated by a parameter ρ, together with an artificial potential function V used to avoid the obstacle. The obstacle is described by a region in the manifold bounded by S, a regular level set of some scalar valued smooth function f. Examples are circles, spheres and ellipsoids. The potential function V is an artificial smooth (or at least C^2) function aimed to produce a fictitious force inducing a repulsion from S. Now that all aspects of the functional J are defined, it is minimized on the admissible manifold ω to produce path segments. The interesting aspect of this work is to observe the nature of the path, if the notion of an obstacle was eliminated from the functional J. That is, when the repulsive force V becomes 0, the resulting path is a geodesic. This shows that using structured approaches to path modifications guarantee a return to definitions of the underlying space, which is of consequence when the PPS is non-Euclidean.

3.2 Sampling Algorithms

Other works also focused on combining homotopy with with other graph based methods, namely SAs and OAs. The study in [82] is one such work that uses homotopy classes to guide path planning algorithms topologically. The objective of [82] is exploration as opposed to path planning in a known environment, so the authors guide their algorithms to explore the environment confined to a specific homotopy class as opposed to the entire space. They exploit three popular algorithms in conjunction with homotopy, which are the Homotopic A* (HA*) [83], the Homotopic RRT (HRRT) [84] and the Homotopic Bug (HB) [85]. They ensure completeness, because in case the goal is not reachable, no homotopy class exists and, consequently, no paths will be generated. The homotopy class of the global optimal path is guaranteed to be generated by the algorithm [82]. A prime advantage of driving exploration in the regions favoured by homotopy classes is that it allows generating good solutions at a fraction of the computation time taken by exhaustive search methods [82]. The authors generate homotopy classes by following the approach outlined in [86] for any 2-dimensional workspace with obstacles, by turning the environment into a topological graph. This allows the systematic generation of homotopy classes using graph-search algorithms. The method first builds a reference frame which determines the topological relationships between obstacles in the workspace and is used to name the homotopy classes. The reference frame is then used to build the topological graph which allows the computation of homotopy classes systematically. Similar to the word generation seen in [80], this approach involves determining a canonical sequence of operations or transformations that correspond to every partition of the environment. Then, paths in different homotopy classes can be seen to have a different sequence of operations with respect to the origin of the reference frame. Such a representative path is called the canonical sequence. The canonical sequence is the simplest representation of a path without changing its topology, and only one canonical sequence exists for each homotopy class, but each canonical sequence can be represented by infinite trajectories in the workspace [82]. Fig. 7 represents one such canonical sequence. Having understood the canonical sequence, the aim of the work is to systematically determine canonical sequences for every homotopy class. For this, the authors construct the topological graph. In the original reference frame, a path is defined according to the segments it crosses, whereas in the graph, it turns into traversing the graph from the source to target nodes. The graph on the right in Fig. 7 depicts the same canonical sequence seen on the left side in Fig. 7, in the topological graph. Following the construction of the topological graph, a breadth-first algorithm is used to traverse the graph. Starting from the source node, it explores every neighbouring node until the goal is found. The difference between the standard breadth-first algorithm and the one used here is that the former stops when all vertices have been visited; the latter however, continues until there are no more homotopy class candidates to explore or the length of the last homotopy class candidate is larger than a given threshold [82].

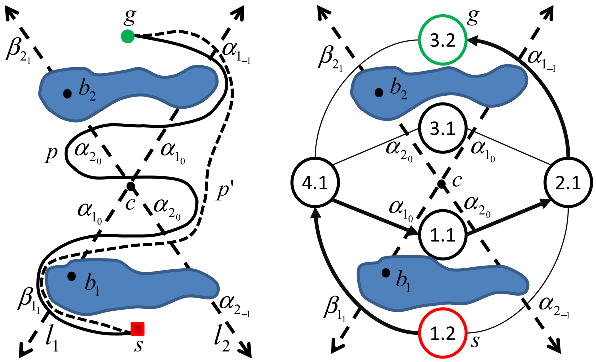
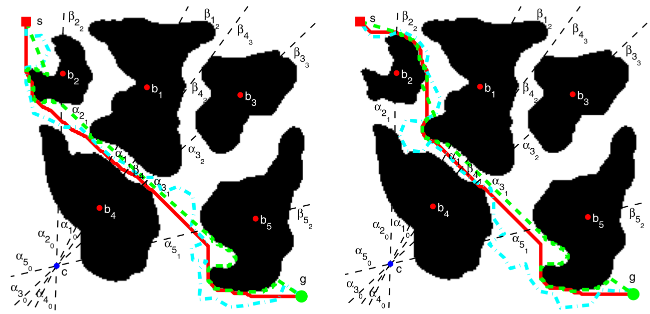


Fig. 7: Topological path in the context of reference frame on the left, and in terms of the topological graph on the right [82]

The number of homotopy classes generated by the algorithm highly depends on the number of the nodes in the topological graph. To set up a ranking system for the classes, the authors use a modified version of the funnel algorithm. It computes a quantitative measure for each homotopy class estimating its quality, by estimating the lower bound of the optimal path in the region of the selected homotopy class [82]. Once the sorting of homotopy classes is complete, three path planning algorithms are used to compute a path in the homotopy class of choice: A*, RRT and Bug. This process implies that the topological graph is converted into a metric path, with the reference frame being the key and only link between the graph and the original environment. The HA* is a graph-based search algorithm based on the A* which only explores the



(a) $\alpha_{2_1}\alpha_{1_1}\beta_{4_1}\alpha_{3_1}\alpha_{5_1}$. (b) $\beta_{2_2}\alpha_{1_1}\beta_{4_1}\alpha_{3_1}\alpha_{5_1}$.

Fig. 8: Paths generated with HA* (red), HRRT (cyan) and HBug (green) algorithms for the two homotopy classes [82]

zones in the workspace that satisfy a given homotopy class, as opposed to the entire space. It does so by checking the intersections with the reference frame before taking into consideration the node as a candidate to be explored. The advantage of HA* is that it is complete. The second variant used is that of the HRRT is based on the RRT. HRRT allows a constrained growing of the tree only in those directions that satisfy a given homotopy class. Similar to HA*, adding a new node into the tree is done only if the topological path traversed belongs to the homotopy class. It can be seen that the HRRT is probabilistically complete for a given homotopy class.

The final algorithm used is that of the HB, based on Bug 2 from [87]. The opportunistic algorithm exploits the already computed lower bound path generated by the funnel algorithm, which was used to determine the preference of homotopy classes based on their lower bound path estimates. The segments in the reference frame constrain the regions the paths can go through, but do not take into account the shape of the obstacles. So should a collision be imminent, the HB performs boundary following according to the homotopy class until the lower bound path leaves the obstacle. The authors note that the HB is complete; however, no explicit assumptions about the obstacles are made. In that case, it is safe to assume that their conclusion may be based on convex obstacle regions only. The efficiency and scalability of the proposed methods have been tested in synthetic scenarios mimicking a cluttered environment. Fig. 8 depicts the paths of two homotopy classes and a clearer example of the HB is seen in Fig. 9. When the authors made an explicit comparison between the homotopic set of three algorithms, i.e., HA*, HRRT and HB, with their original counterparts, the following was observed. The homotopic variants returned an optimal solution typically faster than their original variants. Additionally, they generated an optimal path in all of the computed homotopy classes, and each variant took a certain fraction of time that it took to compute the optimal path. While this time is longer than the original counterparts' performance for optimal paths, note that the original counterparts cannot accomplish this task

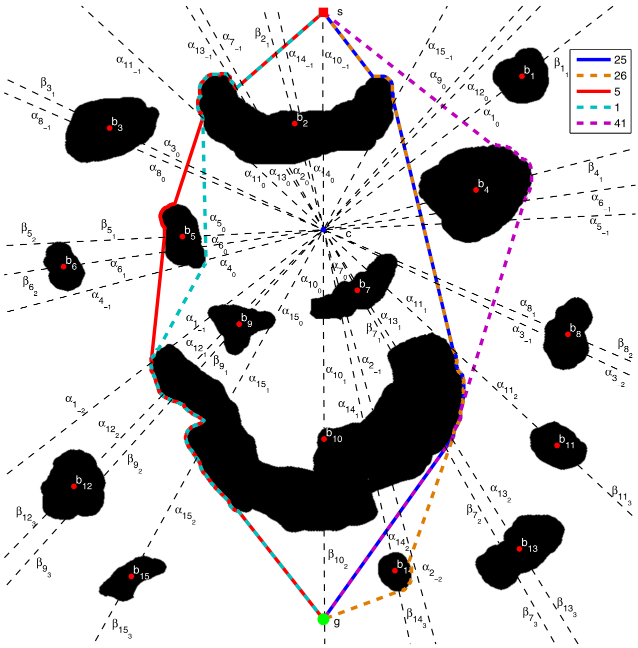
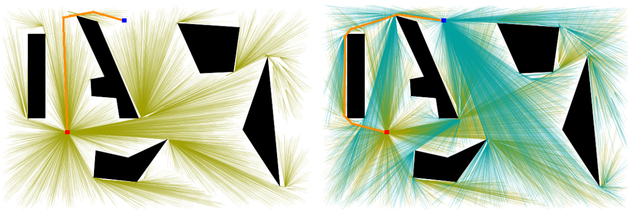


Fig. 9: HBug and homotopy classes [82]. The different coloured lines can be interpreted as representative paths for different homotopy classes. That is, each coloured line connects the start and goal locations, however, differ from one another, based on their winding characteristics. For example, the yellow line winds around/borders obstacle regions b_{10}, b_{b_7} and b_2 in a positive right-hand-frame rotation. The green dotted line also winds around in the same positive direction, however, it winds around an additional obstacle region b_4. Therefore, it represents a different homotopic class. The dotted red line represents yet another different homotopic class, because it winds around a different set of obstacle regions and with a negative winding direction.

at all. The HB offered the best performance among the homotopic variants, with a mere cost of 1.05 times the optimal path cost.

The clear advantage of such discretized graph based approaches of homotopic path planning is the one-to-one correspondence between the cell sequence and the homotopy class. The issue of incorrect or multiple solutions across homotopy classes, is not encountered here. Authors of [88] also use a variant of such discrete representations of homotopy classes to guide their SA of choice, which in this case is the RRT. They propose an algorithm called the Homotopic Aware RRT*, or HARRT* [88], which consists of three steps, as follows. The first involves partitioning the PPS; the second step uses some form of representation to denote unique path classes, and finally, the third step employs a path planning method on this segmented canonically represented space. These three steps use inspiration from works in the area, which are explained as follows. The authors of [88] partition their PPS based on variants of partitioning algorithms used previously, such as those based on reference frames, as seen in [82]. Consequently, they use the canonical representation for every homotopy class, similar to previously described works. The main contribution of this study is the use of homotopic groups to drive a biased exploration using the RRT*. The authors propose an algorithm called the Recursively Embedded Palindrome (REP) to trim newly generated strings or paths, to their minimum representative lengths or sequence. This allowed the authors to identify homotopic equivalence of path segments or branches generated thus far, and allow the expansion to take place in restricted homotopy classes. The authors also show its impact on both the single and bi-directional RRT, as seen in Fig. 10. The authors do recognize the challenge of combining RRT with the REP trim algorithm, since there is a possibility that potentially optimal paths in a homotopy class may be prematurely discouraged from being explored. While it was resolved by using a bi-directional RRT, the authors point out that there may be other configurations of environments that potentially still have the same problem. Fig. 11 shows the optimal solutions returned by HARRT* for six different homotopy classes in another 2-dimensional environment with planar obstacles.



(a) Single directional tree (b) Bidirectional tree

Fig. 10: Impact of Recursively Embedded Palindrome strings in single and bi-directional RRT [88]

Consistent improvements to graph construction and representation of homotopy classes can be seen through works, such as [89, 90]. The study in [89] shows the direction of improvement catering to dynamic environments, where road map construction and path generation need better efficiency, in terms of the next sampled point or direction of growth. In environments with narrow corridors, discovering all such narrow spaces and redundantly connecting them with the road map plays an important role towards the optimality and completeness of the solution. The objective is that the road map must consist of the least number of vertices and edges (for minimizing road map construction and query time), while leading to discovery of all possible homotopic groups of paths between all possible sources and goals. Discovery of all homotopic

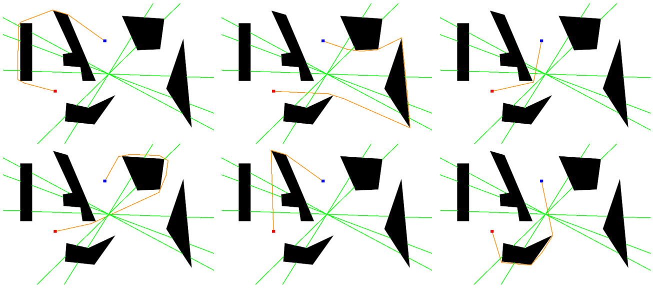


Fig. 11: Paths in six different homotopy classes obtained by HARRT* [88]

groups guarantees a path for every pair of valid source and goal locations, or in simple words, the approach is complete [89].

To achieve the objective, [89] thus focuses on different ways to manipulate graphs to reduce redundant cycles and yet, provide better paths in narrow areas. The author of [89] therefore proposes a better sampling strategy to enable a homotopy conscious road map to be constructed. An extension of this work is seen in [90], which better outlines the contributions in the area of homotopy conscious road maps.

The work in [90] focuses on devising a road map generation method using PRM, such that most homotopic groups are represented in small computation times. The key difference between this approach and other homotopic variants of sampling techniques, is that this approach changes the sampling strategy to detect homotopy classes in an unstructured environment, and possibly unknown as well. Additionally, the work also proposes a difference in the kind of local road maps constructed depending on the nature of the region being sampled. The local road map facilitates redundant connectivity to the rest of the road map, which is an advantage, because it can simultaneously cater to narrow corridors, congested environments and otherwise difficult areas. In order to connect these sampled points, an edge connection strategy is formulated. This strategy discounts the connectivity of an edge, based on the sizes of the road maps that it is directly or indirectly connected to. It saves computational time while simultaneously maintaining nearly the same connectivity of the road map as a k-connected PRM.

The design of the algorithm aims to suit a variety of cases including narrow corridors, open spaces, multiple pathways between obstacles of multiple widths and narrow corridors amidst wide open spaces. Yet another work that attempts to combine efficient sampling with homotopy is [91]. A combination of modified sample generation and discard is used to generate points of the Volumetric Tree* (VT*). Hyperspheres around the robot are used to generate new samples, from which vertices of VT* are created. The authors of [91] claim that the points generated in the hypersphere are homotopic to other points in the same hypersphere, and use this as a justification of generating sufficient points in the VT*. They use the Dynamic Shortest Path Tree to compute paths and also perform homotopy exploration. One must note that [91] does not establish homotopy classes as established in previous works on the subject. Since random sampling and dropout techniques do not explicitly lead to identification of homotopy classes in the entirety of the PPS, previously described works may serve as better references on the matter.

3.3 Recursive/Discretized Solutions

The shift towards discretizing the identification of homotopy classes for path planning started to make its first appearance in the works of [77–79]. The study in [80] is one of the first that discretizes the $\boldsymbol{H}$-signature of a homotopy class and then uses mixed integer linear programming to find the optimal path in the chosen homotopy class. The authors of [80] approach the problem of trajectory generation in 2-dimensions, and divide it into two main parts, as follows. The first part involves describing the homotopy class using a "word", which is a coarse representation of the trajectory, and in turn the homotopy class to which it belongs. In the second part, they determine the most optimal path corresponding to the word describing the homotopy class. Now, each of the aforementioned parts is explained in detail. To accomplish the first part, the authors of [80] partition the PPS, and represent each partition by a letter, as seen in Fig. 12. It is easy to see that any path can be represented by a sequence of letters indicating the parts of the environment that the path crosses, and in turn the constraints/obstacles that the path encloses. Naturally, there can be more than one word associated with a homotopy class, since more than the minimum number of partitions in the same homotopy class can result in several non-unique words that describe that class. The $\boldsymbol{H}$-signature of the trajectory τ with respect to a representative point of obstacle j is

$$H_j(\tau) = \int \frac{1}{z - z_j} dz = \int_{t_0}^{t_f} \frac{1}{\tau_x(t) + i\tau_y(t) - z_j} (\dot{\tau_x}(t) + i\dot{\tau_y}(t))\, dt \tag{4}$$

Note that (4) considers time to evaluate the $\boldsymbol{H}$-signature, as opposed to (3), which does not do so. The $\boldsymbol{H}$-signature for a trajectory encompassing n obstacles is given by $H(\tau) = [H_1(\tau), H_2(\tau), \ldots, H_n(\tau)]^T$, similar to what is seen in the authors' original works in [77–79]. Before the $\boldsymbol{H}$-signature can be used to determine the optimal path, one must describe the homotopy class of interest in detail. Given a source and target location, the resulting path falls into a certain homotopy class, called the desired homotopy class. This initial trajectory helps detect the homotopy class of interest, using which an optimal path in the class can be determined. To find an optimal trajectory satisfying the given homotopy class constraint, they first construct the set of words in the same homotopy class as the required one; this set is denoted by W_h. Then, as part of the second step, every word in W_h undergoes an optimization process that minimizes the time spent by the path in each letter or partition of the PPS. In due course, the most optimal path corresponding to each word is returned such that the path is made up of segments with the least amount of time in every partition. The optimization process imposes the *H*-signature as seen in (4) as a constraint to be adhered to; additionally, the time spent in each partition may also be imposed as a constraint. The resulting problem is non-linear, so the global minimum for the optimized trajectory may not be guaranteed. However, the iterative nature of the algorithm does produce a trajectory whose cost is further minimized, as opposed to that of the initial solution. So, it is considered as an anytime algorithm that provides better solutions with time.

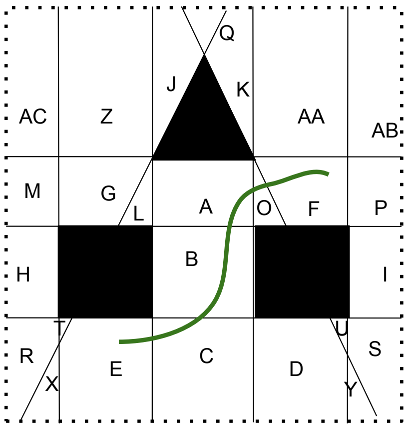


Fig. 12: Word based homotopic path planning

The authors of [80] present and analyze simulations in a planar environment. They compare the impact of imposing homotopy class constraints for four distinct homotopy classes, with not imposing such constraints. The iteratively optimized and generated trajectories in the constraint imposed situations, show that, over time, the sub-optimal trajectories approach the optimal trajectory for its homotopic class. With this work, the authors of [80] have successfully introduced systematic optimization of paths being generated.

One such successful example of discretized homotopic path planning in dynamic environments in a 2-d polygonal space is presented in [92]. The authors of [92] use a combination of cell decomposition and graph theory to effectively create representations of homotopy classes. Similar to earlier works in the field, once the homotopy classes have been found, mixed integer quadratic programming is used to determine an optimized trajectory within a preferred cell sequence associated to a specific homotopy class. The algorithm is seen to comprise the following general steps. First, the PPS is decomposed into convex polygons using well known convex decomposition algorithms. This is of use eventually, since adjacency relationships between decomposed polygons can be employed to represent adjacencies as a graph. Eventually, graph based methods can be used to search the adjacency graph for an optimal path connecting the source and target locations. Adjacencies in the form of adjacent cells or polygons, contain valuable information related to homotopy classes. In the second step, they refine the construction of convex polygons in the interest of creating a minimal vertex graph. This step creates partitions of the space in the form of vertices of convex polygons, such that new vertices are created only if they represent new homotopic information. The third step sees the construction of the adjacency graph, where adjacency of cells is recognized by virtue of shared edges, as opposed to vertices. This is useful to determine sequences of cells, that the path takes and thus determine the homotopy class. The fourth step involves determining what sequences of cells signify a homotopy and how that information can be extracted from the graph. The authors exploit the loop properties of the adjacency graph to determine homotopy classes, similar to the creation of words as seen in previous discretized homotopy algorithms.

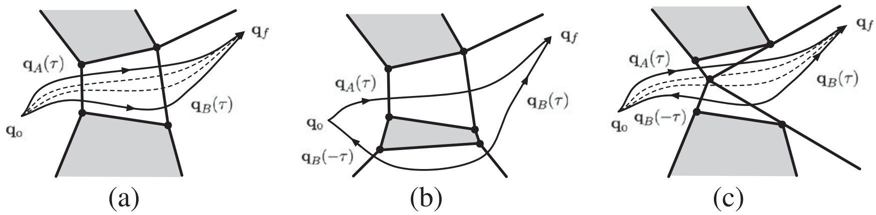


Fig. 13: Different homotopy classes based on different cells [92]

Once they partition the infinite trajectories into finite sets of trajectories, they show that there exists a one-to-one correspondence of loop-less cell sequences with homotopy classes. In other words, a feasible trajectory is homotopic with any other feasible trajectory corresponding to the same loop-less sequence of cells on the graph, and not homotopic with all feasible trajectories in different sequences of cells in the graph. Fig. 13 illustrates these properties. The most important and defining statement of the work is seen with the correlation between the cell sequence and homotopy class, which is as follows. If two feasible trajectories correspond to two different cell sequences, which are loop-less in the adjacency graph constructed through a convex decomposition with the minimal vertex set, they are not homotopic. With this information, the algorithm now proceeds to the final step, which now has the information about which homotopy class is required to be used as a constraint. The algorithm then uses mixed integer programming to optimize the trajectory segment in each convex cell of the homotopy class. The successful use of this algorithm in simulated dynamic environments on a model with non-linear vehicle dynamics can be seen in Fig. 14.

The authors of [93] also use word based homotopic path planning. They propose the Homotopy Informed Preprocessor (HIP) algorithm to find the subset of the PPS containing the optimal path and prune those path candidates that fall outside the subset. The three steps of the HIP are as follows. The first step is to use homotopic invariants to obtain disjoint homotopy classes as a

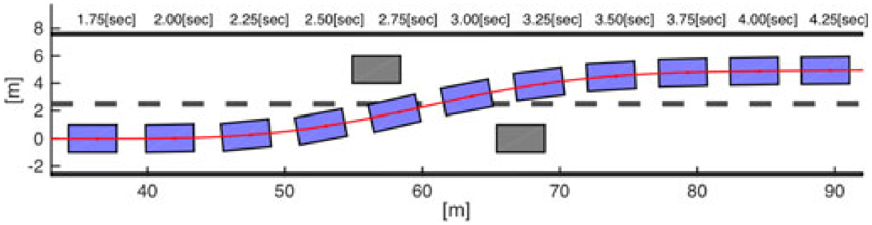


Fig. 14: Paths to avoid dynamic obstacles [92]

partition. The second step uses the Multiple Jump Point Search (Multiple JPS) algorithm to obtain all shortest paths locally in the discrete grid representation of the PPS. From this, they are able to correlate those homotopy classes that produce optimal paths, and these classes are called the optimal collection. The remaining non-optimal classes are filtered during the pruning process. The third step constructs words of the homotopy classes of the optimal collection. The word requirement forms an additional pruning criterion in rejecting those path candidates that do not satisfy the word requirement. This work's primary contribution is in the qualifier used for pruning and sampling. It also shows the use of additional pruning through HIP in conjunction with RRT*.

The clear advantage of such discretized graph based approaches of homotopic path planning over optimization based approaches as will be seen in Section 3.4, is the one-to-one correspondence between the cell sequence and the homotopy class. The issue of incorrect or multiple solutions across homotopy classes, experienced as curve jumping in Non-linear Algebraic Equation System (NAES) is not encountered here.

3.4 Optimization based solutions

Here, approaches that combine optimized recursion are seen to solve for a homotopically valid path. For example, one approach to discretize the homotopic PPP is by using NAES, which can be solved iteratively for a valid path. Works such as [68–71] use different variants of NAES to model the ability to determine a path using homotopic constraints. Since the environment is not discretized in the same manner as was seen with the usage of words in a homotopy class, techniques that rely on NAES are often referred to as Homotopy Continuation Methods (HCM). As the authors of [68] explain, HCMs are a continuous transformation from one trivial problem (simple to solve) to the problem at hand (hard to solve). The general process of the HCM is briefly described first, before explaining how the PPP can be adapted to it. In the general HCM problem, the first step to formulate a homotopy is to establish a nonlinear equation that models the problem to be solved. It is defined as $f(x) = 0$, where $f : \mathcal{R}^n \to \mathcal{R}^n$, consisting of n total variables of the problem, denoted by x. A homotopy map may be written in this form as $H(f(x), \lambda) = 0$, where $H : \mathcal{R}^{n+1} \to \mathcal{R}^n$, and λ is the homotopy parameter. In this homotopy formulation, the value of the homotopy parameter λ determines whether a solution to the NAES has been found. When $\lambda = 0$, the solution to $H^{-1}(0)$, is either known or can be found using numerical methods, and when $\lambda = 1$, it implies that the solution/s to $H(f(x), 1) = f(x)$, can be found. The homotopy map therefore is a continuous function of $0 \leq \lambda \leq 1$. The resulting solution set of $H^{-1}(0)$, forms an initial homotopy path, that can then be traced by numerical continuation techniques. The next step involves setting a specific homotopy denoted by $g(x)$, which is combined with $H(f(x), \lambda)$ to augment the existing NAES. This new set of NAES can be solved by continuously varying λ, resulting in paths, as seen in Fig. 15. Details about the numerical continuation method for NAES are provided in [68]. It is interesting to note that in Fig. 15, trajectory γ_2 is a successful case from the starting point A ($\lambda = 0$) to solution B ($\lambda = 1$), while the remaining paths are cases of failure for the HCM. This matter of incorrect solutions, did not occur so far in the original and discretized forms of homotopic path planning, and must be noted for their reliability in path planning in dynamic environments.

Having obtained a perfunctory understanding of HCM, its application to path planning is now seen. The authors of [68] formulate all parts of the PPS as algebraic equations. Obstacles modeled as circular or rectangular structures are modeled as their respective algebraic counterparts. The PPS is planar and normalized so that the source and target locations can also be represented algebraically. The homotopic constraint is chosen in the form of a repellent behaviour, and an attraction towards two specific initial solutions. As the authors themselves note, this represents some sort of force fields around the real obstacles. The homotopy path cannot cross those fields, however it can pass close to them. This, along with the possibility of multiple solutions must be considered, in the efficacy of using NAES for homotopic path planning. Though the authors show that this method returns paths in static 2-d and 3-d environments, there exists a problem of curve jumping, when parameters of the NAES are non-optimal. An example of the paths generated in 2-d and an example of curve jumping is seen in Fig. 16. Paths in 3-d are seen in Fig. 19. Works, such as [69–71], also rely on NAES to solve the homotopic Path Planning Manifold (PPM). The individual algorithms vary either based on how the robot is represented and the nature of the initial paths used as initial best guesses. The authors of [69] use guiding parameters, such as auxiliary lines, that help keep the path deformation under control. The effect of auxiliary lines essentially adjusts the repulsion parameter, and thus adjusts the deformation of the path, as seen in Fig. 17. The more auxiliary lines there are, the better the deformation of the path and its resulting quality, as seen in Figs. 17 and 18. The caveat with such methods however, is that the algebraic equations representing the PPS and the robot appear to lie on a plane. In such a case then, there is not much of a vast advantage over original homotopic methods using complex analysis or discretized methods using words to represent homotopy classes.

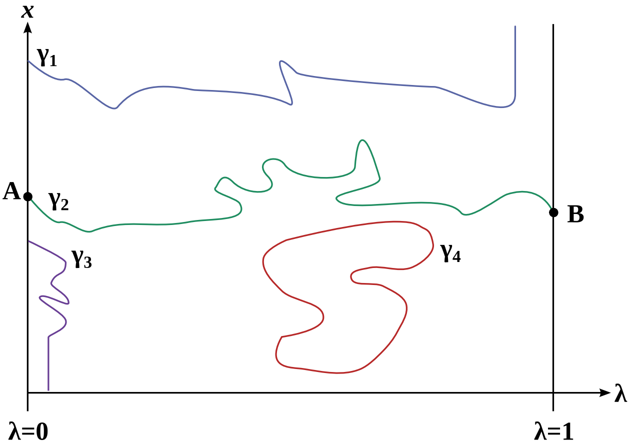
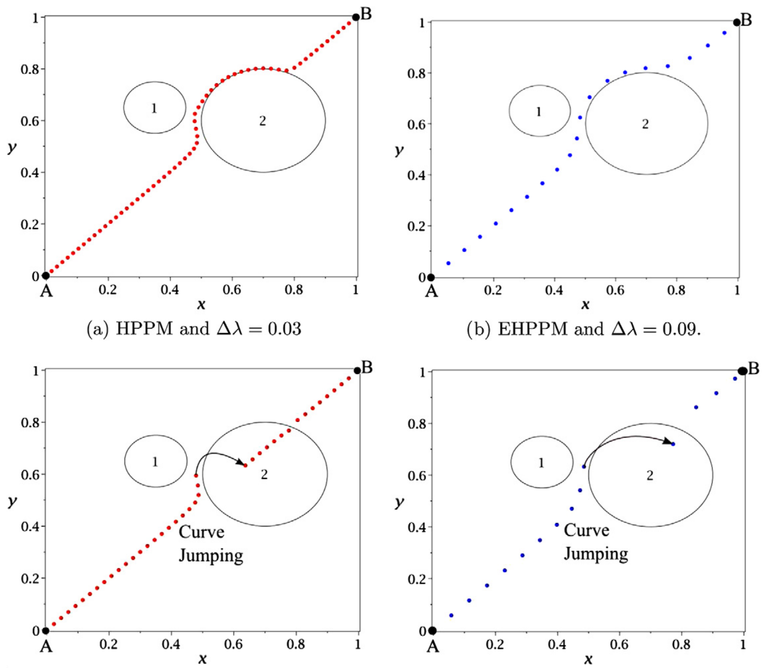


Fig. 15: Different path solutions to the NAES problem [68]

3.5 Model Based Methods

In this section, approaches that utilize knowledge of the robot models in conjunction with homotopic path planning can be seen. Authors of [72] show that integrating homotopy classes with path planning is indeed a structured process. They propose Structured Homotopy (SH) based path planning for a Dubin's vehicle model, which is assumed to have an upper bound on the linear and angular velocities. Such bounds force the Dubin's model to produce paths that are a combination of straight lines and/or circular segments, each of whose length and curvature is bounded by the linear and angular velocities. Yet another work that combines homotopy and Dubin's curves can be seen in [73]. These curves are referred to as the Dubin's curves, and are the kind of paths being used and generated in [72]. Using these curves, a homotopy class called the curvature bounded homotopy class is constructed. This homotopy class constitutes an envelope of the path planning region, where segments of the Dubin's curves can be continuously deformed into other curvature bounded Dubin's path segments. The aim of this is to have a dedicated region in which the initial guess of the Dubin's curve path can be optimized for path length, while still being homotopically the same. In order to do so, the authors choose a specific set of Dubin's curves to work with, for the following reason. The path deformation of such curves when discretized, is a recursive function of the intermediate segments' end points. Called the Structured Homotopy based path planning, this is what finds the solution to a new path in the homotopy class with desired length characteristics. To demonstrate the effectiveness of SH based planning with Dubin's curves, the authors simulate a case where



(a) HPPM and $\Delta\lambda = 0.03$

(b) EHPPM and $\Delta\lambda = 0.09$.

(c) HPPM $\Delta\lambda = 0.04$

(d) EHPPM $\Delta\lambda = 0.10$.

Fig. 16: Illustration of curve jumping [68]

the target locations lie on a manifold, which in this case is that of the annulus. This is of use, since an annulus' vector bundle induces different kinds of Dubin's curves (due to change in curvature), as opposed to a plane.

In [74], homotopy is used for multi-robot path planning, such that no deadlock is encountered when the multiple robots track their paths. They do so by accomplishing two goals. First they show that coordinated trajectories computed by an arbitrary multi-robot motion planner can be used to derive a maximal set of homotopic solutions forming a homotopy class. This was obtained by translating the problem into the coordination space. This set of solutions is composed of all paths in the coordination space that are homotopic to an initial path called the diagonal path. This diagonal path is a path in the coordination space that steers clear of any constraints, similar to a canonical path of a given homotopy class. Viewed from a different perspective, the diagonal path serves as a different method to partition the coordination space, similar to word-based homotopic path planning. Then, the authors derive a collision and deadlock free region in the coordination space, which is also comprised of the maximal set of solutions in the homotopy class. The second contribution involves devising a controller that the system remains in the homotopy class once it has entered it. That is, the homotopy class is formulated as an invariant set. Though multi-robot coordination is out of the scope

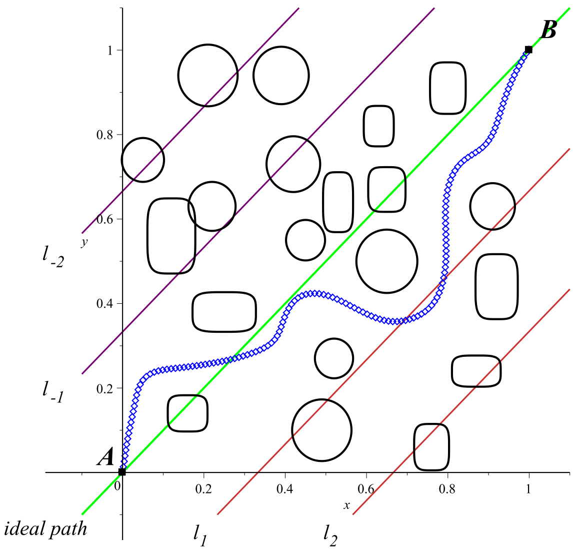


Fig. 17: Impact of auxiliary lines [69]

of this work, experimental results from [74] show that, homotopy classes can be discretized and that they may be robust to perturbations or noise. Original HBMs and their discretized variants may be seen to have an advantage over this method, due to their robustness towards 3-d spaces. The method outlined here, is catered to planar 2-d environments, which are realistic for office like environments, but may not be truly robust as a generic homotopic path planner.

The authors of [75] offer an interesting perspective on homotopy based path planning, that one could consider as an application for dynamic environments. The main contribution of the paper is a switch from one homotopy class to another, should one class no longer be favorable to traverse in. They do so by proposing an algorithm called the Planning Online by Switching Homotopies (POSH). An illustration of this is seen in Fig. 20. POSH builds on inference based trajectory optimization techniques to plan online and dynamically switch homotopy classes. POSH can extract information about multiple homotopy classes, and in light of updated information of the dynamic environment, the appropriate homotopy class can be selected as the new optimal class. This prevents the algorithm from stagnating at a class that recently became sub-optimal due to a new constraint. The authors use Gaussian Process Motion Planning (GPMP) to enable a probabilistic update to the representation of the environment in the form of a graph. Instead of retaining the same optimized trajectory from the initial time step for execution as does GPMP, POSH maintains and updates the entire graph at every time step. Specifically, at any time step the graph is pruned to remove unreachable (in time) states. The graph is then re-optimized with the changes in the environment, and is

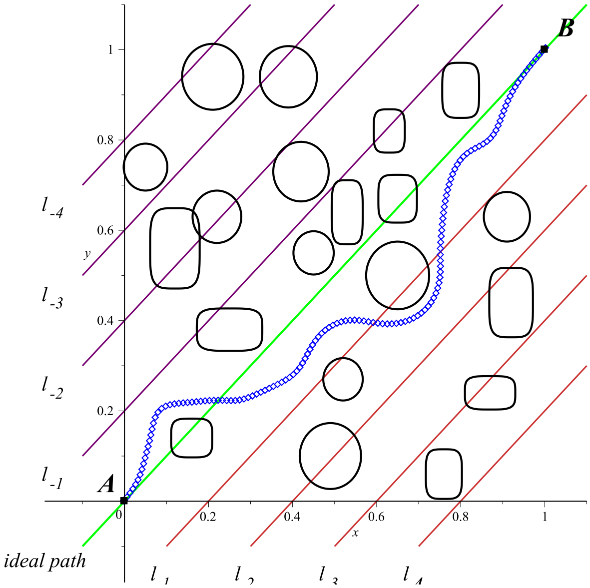


Fig. 18: Impact of increased auxiliary lines [69]

now equipped to find the new optimal trajectory. The details of GPMP may be seen in [75]. They are summarized briefly here. GPMP constructs a factor graph with multiple initial trajectories (chains), each with its own set of path segments, called support states. Adjacent chains have interconnected support states with a prior factor and interpolated factors between two states from two different chains. This allows the robot to switch between chains. An example graph construction with three chains and interconnections is shown in Fig. 21. Note that such interconnecting support states and factors are what represent the multiple paths in the same homotopy class and in different classes, without which deliberate class switching is not possible. The *H*-signature is used as the unique identifier for the homotopy class and is applied to identify the frequency at which the robot changes the homotopy class of its trajectory. The POSH algorithm consists of the following steps. First, using the signed distance field, the obstacle pose is initialized. Next, by computing a factor graph, one can determine the extent to which the nodes are required to be interconnected for better switching or faster solution times; refer to Fig. 22. In the third step, A* is employed to generate the initial best cost path, which will be followed until new information about the environment changes the graph. The fourth step executes the motion planned thus far, and performs an update of the support nodes using the signed distance field. This will help prune useless support states or nodes that are no longer valid. On this graph, A* is then used again to produce the next best path, and the algorithm is recursive, until the goal is reached. Fig. 23 illustrates the algorithm pictorially, while Fig. 24 shows the

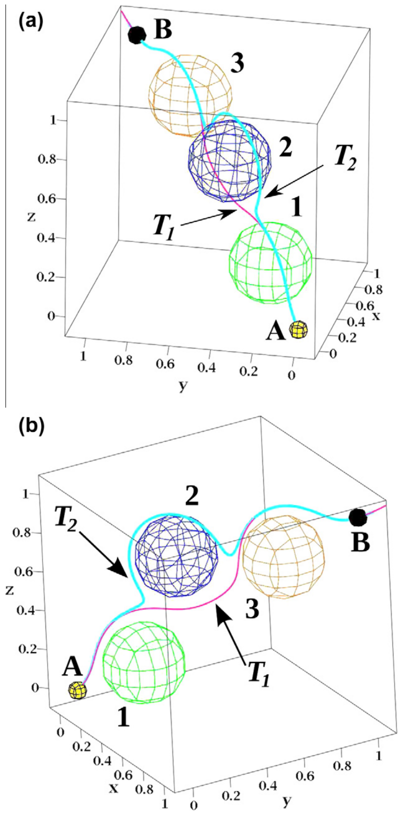


Fig. 19: Paths seen in 3D using NAES [68]

results. The latter shows the response of the algorithm in the dynamic forest environment consisting of numerous objects exhibiting random 2D motion. The dynamic forest shown in Fig. 20 is employed to benchmark the ability of the algorithms to contend with a challenging re-planning problem [75].

4 Conclusion

Viewing the PPP in the context of avoiding DCs in the PPS, showed that there are three main factors to consider: (i) defining DCs, (ii) characterizing the response of the algorithm to DCs, and (iii) the kinds of mathematical tools used to define the chosen response. DCs could be moving objects or the sudden presence/absence of stationary objects. This means that the knowledge of such constraints may be evanescent, thus characterizing the nature of the response provided by the algorithm. Fleeting information about dynamically appearing constraints may require the algorithm to reactively provide a local, remedial solution. This is opposed to global deliberative approaches, which preemptively steer clear of constraints (assuming their knowledge is at least partially known

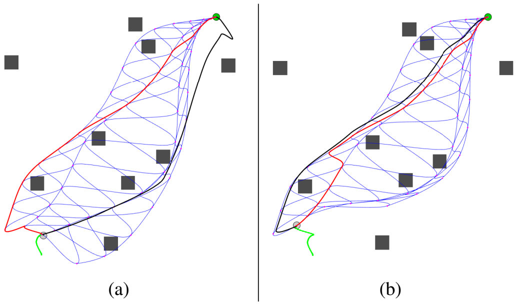


Fig. 20: Illustration of Planning Online by Switching Homotopies (POSH) [75]

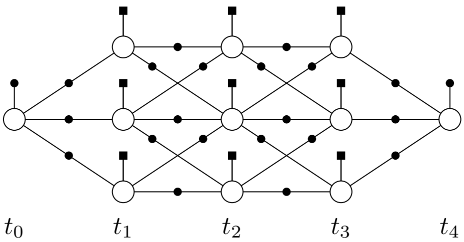


Fig. 21: An example graph construction with three chains and interconnections [75]

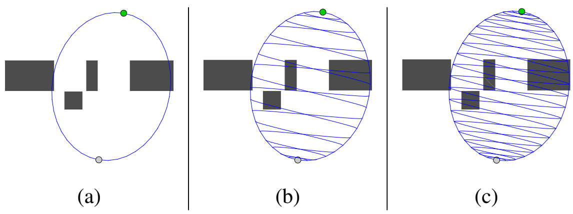


Fig. 22: Different desired levels of inter-connectivity [75]

apriori) and prioritize navigating towards the destination. Global deliberative approaches and local reactive algorithms typically induce slightly different path variations and maneuvers. The former aims to steadily and seamlessly modify the path such that the potential constraint is not encountered at close quarters, whereas the latter aims to generate immediate collision-free movements, such as a blatant turning away or termination of movement. While both kinds of algorithms modify the path either as a whole or in part, the operation involved is that of constructing a deviation in the original pre-constraint path. This

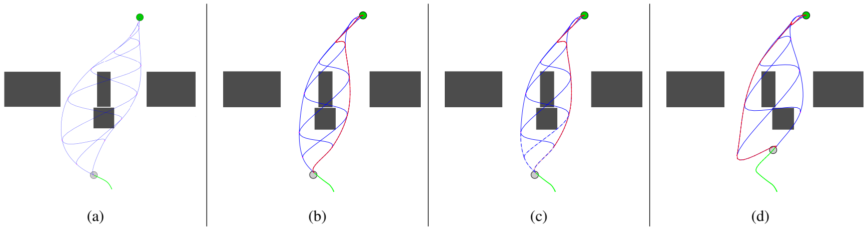


Fig. 23: In (a), the gray circle and green circle are the robot and goal states. In (b), A* is used to find a low-cost trajectory (red) on the graph. In (c), edges that are no-longer viable (dashed blue lines) after the first movement are pruned. These can include the collision-free segment currently chosen as the direction of movement and segments that currently or potentially intersect with obstacles. Finally in (d), as the robot executes its chosen direction of movement, the environment changes with the lower square moving further along the positive x-axis. Post re-optimization, POSH finds a trajectory in a new homotopy class (red). [75]

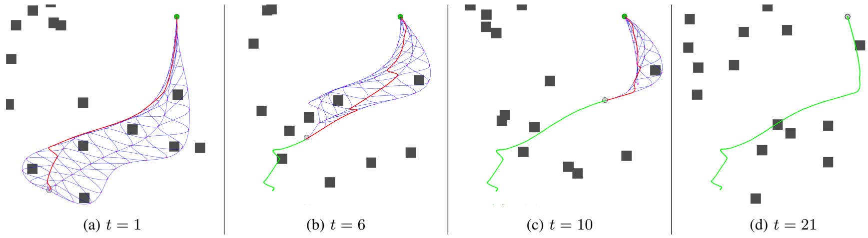


Fig. 24: Another illustration of POSH [75]

deviation is what divides the approaches of the PPP with DCs into MBMs and HBMs.

In order to produce the desired modification of the path, MBMs may rely on all or some of the following: models of the robot, environment, and expected constraints. Some examples of MBMs include a combination and/or variants of Model Predictive Control (MPC) (with Dubin's curves, VOSs), OAs (diffusion, conductivity and APF based methods), SAs (reactive RRTs, PRMs) and CDGT (tangent graphs). The primary advantage of producing path deviations especially suited to the robot model, also means that the algorithm is not model independent. Robustness to different environments and robot models is seen as a desirable feature, since there is minimal modification required to use the algorithm in a new PPP. The other demerit experienced by MBMs is that they inherit the same disadvantages as were originally experienced in the context of SCs. So, issues of sampling affecting the representation of the environment and encounters with non-optimal solutions in the form of oscillations and local minima, still arise. The alternative therefore, is a generic solution that minimizes the pitfalls seen with MBMs, and that group of solutions is called HBMs. HBMs are able to distinguish categories of paths based on how many constraints the path encloses. The knowledge not only allows identifying desired homotopy classes within which ideal paths should lie, but also the deformations that will allow the path to fall into the desired class. When combined with graph traversal based algorithms or used on their own, the original and discretized HBMs are seen to effortlessly identify homotopy classes in environments with several obstacles. Their robustness to noise (although still a function of sensor data), independence of robot model and avoidance of ambiguous solutions (such as local minima or oscillations) serve as important advantages over MBMs.

A high-level comparison of the different types of path planning algorithms that fall under the two main distinctions, i.e. MBMs and HBMs, is shown in Table 2.

Table 2: Comparison of path planning algorithms in the context of DCs

	Algorithms	
Characteristic	MBM	HBM
Requires robot/constraint models	Yes	No
Single/multiple query	Both	Both
Anytime algorithm	Possibly; depends on the variant chosen	Yes
Robust to noise	No	Yes
Robust to stationary and mobile robots	Yes	Yes
Completeness	Probabilistically complete	Complete (resolution-complete after discretization)
Vulnerable to local minima	Yes	No
Multiple possible alternative solutions	No	Yes
Manifold based path planning	No	No

Statements and Declarations

- Funding: This work was partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). Grant RGPIN-2014-06512.
- Competing interests: The authors have no competing interests to declare that are relevant to the content of this article.
- Ethics approval: Not applicable
- Consent to participate: Not applicable
- Consent for publication: Not applicable
- Availability of data and materials: There is no data or materials to share.
- Code availability: Not applicable
- Authors' contributions: All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Sindhu Radhakrishnan. The first draft of the manuscript was written by Sindhu Radhakrishnan and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References

[1] Molina, C.P., Ortego, R.G., Pérez, F.M.: Perspectives on Technological Developments Applied to Robotics, pp. 59–86. Springer, London (2014). https://doi.org/10.1007/978-1-4471-5358-0_5

[2] Radhakrishnan, S., Gueaieb, W.: A state-of-the-art review on topology and differential geometry-based robotic path planning—Part I: planning under static constraints. International Journal of Intelligent Robotics and Applications (Springer) (2023). (Submitted: Nov. 2023)

[3] Lengyel, J., Reichert, M., Donald, B.R., Greenberg, D.P.: Real-time robot motion planning using rasterizing computer graphics hardware. Computer graphics (New York, N.Y.) **24**(4), 327–335 (1990)

[4] Stentz, A.: Optimal and efficient path planning for partially-known environments. In: Proceedings of the 1994 IEEE International Conference on Robotics and Automation, pp. 3310–33174 (1994). https://doi.org/10.1109/ROBOT.1994.351061

[5] Radhakrishnan, S.: Observable 2D SLAM and Evidential Occupancy Grids. Master's thesis, Carleton University (2014)

[6] Kuffner, J.J., LaValle, S.M.: RRT-connect: An efficient approach to single-query path planning. In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), vol. 2, pp. 995–10012 (2000). https://doi.org/10.1109/ROBOT.2000.844730

[7] Ferguson, D., Stentz, A.: Anytime RRTs. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5369–5375 (2006). https://doi.org/10.1109/IROS.2006.282100

[8] Lee, J., Pippin, C., Balch, T.: Cost based planning with RRT in outdoor environments. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 684–689 (2008). https://doi.org/10.1109/IROS.2008.4651052

[9] Jaillet, L., Cortés, J., Siméon, T.: Sampling-based path planning on configuration-space costmaps. IEEE Transactions on Robotics **26**(4), 635–646 (2010)

[10] Masehian, E., Amin-Naseri, M.R.: A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. Journal of Robotic Systems **21**(6), 275–300 (2004)

[11] Qin, L., Yin, Q., Zha, Y., Peng, Y.: Dynamic detection of topological information from grid-based generalized voronoi diagrams. Mathematical Problems in Engineering **2013**, 1–11 (2013)

[12] M.LaValle, S.: Rapidly-exploring random trees: A new tool for path planning. Technical report, Iowa State University, Ames, IA 50011 USA (June 1998)

[13] Kavraki, L.E., Svestka, P., Latombe, J.-C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation **12**(4), 566–580 (1996). https://doi.org/10.1109/70.508439

[14] Elbanhawi, M., Simic, M.: Sampling-based robot motion planning: A review. IEEE Access **2**, 56–77 (2014). https://doi.org/10.1109/ACCESS.2014.2302442

[15] Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. The International Journal of Robotics Research **30**(7), 846–894 (2011)

[16] Hsu, D., Zheng Sun: Adaptively combining multiple sampling strategies for probabilistic roadmap planning. In: IEEE Conference on Robotics, Automation and Mechatronics, 2004., vol. 2, pp. 774–7792 (2004). https://doi.org/10.1109/RAMECH.2004.1438016

[17] Bohlin, R., Kavraki, L.E.: Path planning using lazy PRM. In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), vol. 1, pp. 521–5281 (2000). https://doi.org/10.1109/

ROBOT.2000.844107

[18] Akbaripour, H., Akbaripour, H., Masehian, E., Masehian, E.: Semi-lazy probabilistic roadmap: a parameter-tuned, resilient and robust path planning method for manipulator robots. International Journal of Advanced Manufacturing Technology **89**(5), 1401–1430 (2017)

[19] LaValle, S.M., Branicky, M.S., Lindemann, S.R.: On the relationship between classical grid search and probabilistic roadmaps. The International Journal of Robotics Research **23**(7-8), 673–692 (2004)

[20] Dale, L.K., Amato, N.M.: Probabilistic roadmaps-putting it all together. In: Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164), vol. 2, pp. 1940–19472 (2001). https://doi.org/10.1109/ROBOT.2001.932892

[21] LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. The International Journal of Robotics Research **20**(5), 378–400 (2001)

[22] Atramentov, A., LaValle, S.M.: Efficient nearest neighbor searching for motion planning. In: Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), vol. 1, pp. 632–6371 (2002). https://doi.org/10.1109/ROBOT.2002.1013429

[23] Diankov, R., Kuffner, J.: Randomized statistical path planning. In: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1–6 (2007). https://doi.org/10.1109/IROS.2007.4399557

[24] Lin, Y., Saripalli, S.: Sampling-based path planning for uav collision avoidance. IEEE Transactions on Intelligent Transportation Systems **18**(11), 3179–3192 (2017)

[25] Kuwata, Y., Teo, J., Karaman, S., Fiore, G., Frazzoli, E., How, J.: Motion planning in complex environments using closed-loop prediction. In: AIAA Guidance, Navigation and Control Conference and Exhibit (2008). https://doi.org/10.2514/6.2008-7166. AIAA

[26] Wang, W., Zuo, L., Xu, X.: A learning-based multi-rrt approach for robot path planning in narrow passages. Journal of Intelligent & Robotic Systems **90**(1-2), 81–100 (2017)

[27] Suh, J., Gong, J., Oh, S.: Fast sampling-based cost-aware path planning with nonmyopic extensions using cross entropy. IEEE Transactions on Robotics **33**(6), 1313–1326 (2017)

[28] Noreen, I., Khan, A., Ryu, H., Doh, N.L., Habib, Z.: Optimal path planning in cluttered environment using RRT-AB. Intelligent Service Robotics

11(1), 41–52 (2017)

[29] Stopp, A., Riethmuller, T.: Fast reactive path planning by 2d and 3d multi-layer spatial grids for mobile robot navigation. In: Proceedings of Tenth International Symposium on Intelligent Control, pp. 545–550. IEEE, Monterey, CA, USA (1995)

[30] Mediavilla, M., González, J.L., Fraile, J.C., Ramón Perán, J.: Reactive approach to on-line path planning for robot manipulators in dynamic environments. Robotica **20**(4), 375–384 (2002)

[31] Hossain, M.A., Ferdous, I.: Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique. Robotics and Autonomous Systems **64**, 137–141 (2015)

[32] Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. The International Journal of Robotics Research **5**(1), 90–98 (1986)

[33] Volpe, R., Khosla, P.: Artificial potentials with elliptical isopotential contours for obstacle avoidance. In: 26th IEEE Conference on Decision and Control, vol. 26, pp. 180–185 (1987). https://doi.org/10.1109/CDC.1987.272738

[34] Faverjon, B., Tournassoud, P.: A local based approach for path planning of manipulators with a high number of degrees of freedom. In: Proceedings. 1987 IEEE International Conference on Robotics and Automation, vol. 4, pp. 1152–1159 (1987). https://doi.org/10.1109/ROBOT.1987.1087982

[35] Koditschek, D.: Exact robot navigation by means of potential functions: Some topological considerations. In: Proceedings. 1987 IEEE International Conference on Robotics and Automation, vol. 4, pp. 1–6 (1987). https://doi.org/10.1109/ROBOT.1987.1088038

[36] Khosla, P., Volpe, R.: Superquadric artificial potentials for obstacle avoidance and approach. In: Proceedings. 1988 IEEE International Conference on Robotics and Automation, pp. 1778–17843 (1988). https://doi.org/10.1109/ROBOT.1988.12323

[37] Volpe, R., Khosla, P.: Manipulator control with superquadric artificial potential functions: theory and experiments. IEEE Transactions on Systems, Man, and Cybernetics **20**(6), 1423–1436 (1990). https://doi.org/10.1109/21.61211

[38] Agirrebeitia, J., Avilés, R., de Bustos, I.F., Ajuria, G.: A new APF strategy for path planning in environments with obstacles. Mechanism and

machine theory **40**(6), 645–658 (2005)

[39] Montiel, O., Orozco-Rosas, U., Sepúlveda, R.: Path planning for mobile robots using bacterial potential field for avoiding static and dynamic obstacles. Expert Systems with Applications **42**(12), 5177–5191 (2015)

[40] Koditschek, D.E., Rimon, E.: Robot navigation functions on manifolds with boundary. Advances in Applied Mathematics **11**(4), 412–442 (1990)

[41] Quillen, P., Muñoz, J., Subbarao, K.: Path planning to a reachable state using minimum control effort based navigation functions. The Journal of the astronautical sciences **66**(4), 554–581 (2019)

[42] Kowalczyk, W., Kowalczyk, W., Przybyla, M., Przybyla, M., Kozlowski, K., Kozlowski, K.: Set-point control of mobile robot with obstacle detection and avoidance using navigation function - experimental verification. Journal of Intelligent & Robotic Systems **85**(3), 539–552 (2017)

[43] Kowalczyk, W.: Rapid navigation function control for two-wheeled mobile robots. Journal of Intelligent & Robotic Systems **93**(3-4), 687–697 (2018)

[44] Tao, S., Tan, J.: Path planning with obstacle avoidance based on normalized r -functions. Journal of Robotics **2018**, 1–10 (2018)

[45] Barraquand, J., Latombe, J.-C.: Robot motion planning: A distributed representation approach. The International Journal of Robotics Research **10**(6), 628–649 (2016)

[46] Wu, A., How, J.P.: Guaranteed infinite horizon avoidance of unpredictable, dynamically constrained obstacles. Autonomous Robots **32**(3), 227–242 (2012)

[47] Cockayne, E.J., Hall, G.W.C.: Plane motion of a particle subject to curvature constraints. SIAM Journal on Control **13**(1), 197–220 (1975)

[48] Fiorini, P., Shiller, Z.: Motion planning in dynamic environments using velocity obstacles. The International Journal of Robotics Research **17**(7), 760–772 (1998)

[49] Savkin, A.V., Hoy, M.: Reactive and the shortest path navigation of a wheeled mobile robot in cluttered environments. Robotica **31**(2), 323–330 (2013)

[50] Wada, H., Kinugawa, J., Kosuge, K.: Reactive motion planning using time-layered c-spaces for a collaborative robot pady. Advanced Robotics **35**(8), 490–503 (2021)

[51] Wang, B., Liu, Z., Li, Q., Prorok, A.: Mobile robot path planning in

dynamic environments through globally guided reinforcement learning. IEEE Robotics and Automation Letters **5**(4), 6932–6939 (2020)

[52] Ginesi, M., Meli, D., Roberti, A., Sansonetto, N., Fiorini, P.: Dynamic movement primitives: Volumetric obstacle avoidance using dynamic potential functions. Journal of Intelligent & Robotic Systems **101**(4) (2021)

[53] Murphy, R.R., Hughes, K., Marzilli, A., Noll, E.: Integrating explicit path planning with reactive control of mobile robots using trulla. Robotics and Autonomous Systems **27**(4), 225–245 (1999)

[54] Hughes, K., Tokuta, A., Ranganathan, N.: Trulla : An algorithm for path planning among weighted regions by localized propagations. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 1, pp. 469–476 (1992). https://doi.org/10.1109/IROS.1992.587377

[55] Lamiraux, F., Bonnafous, D., Lefebvre, O.: Reactive path deformation for nonholonomic mobile robots. IEEE Transactions on Robotics **20**(6), 967–977 (2004)

[56] McFetridge, L., Ibrahim, M.Y.: A new methodology of mobile robot navigation: The agoraphilic algorithm. Robotics and Computer-Integrated Manufacturing **25**(3), 545–551 (2009)

[57] Cui, P., Yan, W., Wang, Y.: Reactive path planning approach for docking robots in unknown environment. Journal of Advanced Transportation **2017**, 1–11 (2017)

[58] Hildebrandt, A.-C., Klischat, M., Wahrmann, D., Wittmann, R., Sygulla, F., Seiwald, P., Rixen, D., Buschmann, T.: Real-time path planning in unknown environments for bipedal robots. IEEE Robotics and Automation Letters **2**(4), 1856–1863 (2017)

[59] Bevilacqua, P., Frego, M., Fontanelli, D., Palopoli, L.: Reactive planning for assistive robots. IEEE Robotics and Automation Letters **3**(2), 1276–1283 (2018)

[60] Ataka, A., Lam, H.-K., Althoefer, K.: Reactive magnetic-field-inspired navigation method for robots in unknown convex 3-D environments. IEEE Robotics and Automation Letters **3**(4), 3583–3590 (2018)

[61] Li, X., Zhao, G., Li, B.: Generating optimal path by level set approach for a mobile robot moving in static/dynamic environments. Applied Mathematical Modelling **85**, 210–230 (2020)

[62] Diéguez, A.R., Sanz, R., López, J.: Deliberative on-line local path planning for autonomous mobile robots. Journal of Intelligent & Robotic Systems **37**(1), 1–19 (2003)

[63] van den Berg, J., Overmars, M.: Planning time-minimal safe paths amidst unpredictably moving obstacles. The International Journal of Robotics Research **27**(11-12), 1274–1294 (2008)

[64] Belkhouche, F.: Reactive path planning in a dynamic environment. IEEE Transactions on Robotics **25**(4), 902–911 (2009)

[65] Belkhouche, F., Bendjilali, B.: Reactive path planning for 3-D autonomous vehicles. IEEE Transactions on Control Systems Technology **20**(1), 249–256 (2012)

[66] De Filippis, L., Guglieri, G., Quagliotti, F.: Path planning strategies for UAVs in 3D environments. Journal of Intelligent & Robotic Systems **65**(1), 247–264 (2012)

[67] Duchoň, F., Babinec, A., Kajan, M., Beňo, P., Florek, M., Fico, T., Jurišica, L.: Path planning with modified a star algorithm for a mobile robot. Procedia Engineering **96**, 59–69 (2014)

[68] Vazquez-Leal, H., Marin-Hernandez, A., Khan, Y., Yıldırım, A., Filobello-Nino, U., Castaneda-Sheissa, R., Jimenez-Fernandez, V.M.: Exploring collision-free path planning by using homotopy continuation methods. Applied Mathematics and Computation **219**(14), 7514–7532 (2013)

[69] Eduardo De Cos-Cholula, H., Ulises Diaz-Arango, G., Hernandez-Martinez, L., Vazquez-Leal, H., Sarmiento-Reyes, A., Teresa Sanz-Pascual, M., Leobardo Herrera-May, A., Castaneda-Sheissa, R.: FPGA implementation of homotopic path planning method with automatic assignment of repulsion parameter. Energies (Basel) **13**(10), 2623 (2020)

[70] Diaz-Arango, G., Vazquez-Leal, H., Hernandez-Martinez, L., Manuel Jimenez-Fernandez, V., Heredia-Jimenez, A., Ambrosio, R.C., Huerta-Chua, J., De Cos-Cholula, H., Hernandez-Mendez, S.: Multiple-target homotopic quasi-complete path planning method for mobile robot using a piecewise linear approach. Sensors (Basel, Switzerland) **20**(11), 3265 (2020)

[71] Diaz-Arango, G., Sarmiento-Reyes, A., Hernandez-Martinez, L., Vazquez-Leal, H., Lopez-Hernandez, D.D., Marin-Hernandez, A.: Path optimization for terrestrial robots using homotopy path planning method. In: 2015 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 2824–2827. IEEE, Lisbon, Portugal (2015)

[72] Yao, W., Qi, N., Zhao, J., Wan, N.: Bounded curvature path planning with expected length for dubins vehicle entering target manifold. Robotics and Autonomous systems **97**, 217–229 (2017)

[73] Liu, Y., Qi, N., Yao, W., Zhao, J., Xu, S.: Cooperative path planning for aerial recovery of a UAV swarm using genetic algorithm and homotopic approach. Applied Sciences **10**(12), 4154 (2020)

[74] Gregoire, J., Čáp, M., Frazzoli, E.: Locally-optimal multi-robot navigation under delaying disturbances using homotopy constraints. Autonomous Robots **42**(4), 895–907 (2018)

[75] Kolur, K., Chintalapudi, S., Boots, B., Mukadam, M.: Online motion planning over multiple homotopy classes with gaussian process inference. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2358–2364. IEEE, Macau, China (2019)

[76] Cabello, S., Liu, Y., Mantler, A., Snoeyink, J.: Testing homotopy for paths in the plane. Discrete & Computational Geometry **31**(1), 61–81 (2004)

[77] Bhattacharya, S., Kumar, V., Likhachev, M.: Search-based path planning with homotopy class constraints. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence. AAAI'10, pp. 1230–1237. AAAI Press, Atlanta, Georgia, USA (2010)

[78] Bhattacharya, S., Likhachev, M., Kumar, V.: Identification and representation of homotopy classes of trajectories for search-based path planning in 3D. In: Robotics. MIT Press, Cambridge, Massachusetts, United States (2012)

[79] Bhattacharya, S., Likhachev, M., Kumar, V.: Search-based path planning with homotopy class constraints in 3D. In: Invited Paper for Sub-area Spotlights Track on 'Best-paper Talks', Proceedings of Twenty-Sixth Conference on Artificial Intelligence (AAAI-12) (2012)

[80] Kim, S., Sreenath, K., Bhattacharya, S., Kumar, V.: Trajectory Planning for Systems with Homotopy Class Constraints. In: Latest Advances in Robot Kinematics (ARK), Innsbruck, Austria, pp. 83–90 (2012)

[81] Bloch, A., Camarinha, M., Colombo, L.J.: Dynamic interpolation for obstacle avoidance on riemannian manifolds. International Journal of Control **94**(3), 588–600 (2021)

[82] Hernandez, E., Carreras, M., Ridao, P.: A comparison of homotopic path planning algorithms for robotic applications. Robotics and Autonomous Systems **64**, 44–58 (2015)

[83] Hernandez, E., Carreras, M., Ridao, P., Antich, J., Ortiz, A.: A search-based path planning algorithm with topological constraints. application to an AUV. IFAC Proceedings Volumes **44**(1), 13654–13659 (2011)

[84] Hernández, E., Carreras, M., Ridao, P.: A path planning algorithm for an AUV guided with homotopy classes. In: Proceedings of the Twenty-First International Conference on Automated Planning and Scheduling, vol. 21 (2011)

[85] Hernández, E., Carreras, M., Ridao, P.: A bug-based path planner guided with homotopy classes. ICINCO 2012 - Proceedings of the 9th International Conference on Informatics in Control, Automation and Robotics **2**, 123–131 (2012)

[86] Jenkins, K.D.: The shortest path problem in the plane with obstacles: A graph modeling approach to producing finite search lists of homotopy classes. Master's thesis, Naval Postgraduate School Monterey California (June 1991)

[87] Lumelsky, V.J., Stepanov, A.A.: Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. Algorithmica **2**(1-4), 403–430 (1987)

[88] Yi, D., Goodrich, M., Seppi, K.: Homotopy-aware RRT: Toward human-robot topological path-planning. In: The Eleventh ACM/IEEE International Conference on Human Robot Interaction. HRI '16, pp. 279–286. IEEE Press, Christchurch, New Zealand (2016)

[89] Kala, R.: Homotopy conscious roadmap construction by fast sampling of narrow corridors. Applied Intelligence (Dordrecht, Netherlands) **45**(4), 1089–1102 (2016)

[90] Kala, R.: Homotopic roadmap generation for robot motion planning. Journal of Intelligent & Robotic Systems **82**(3), 555–575 (2016)

[91] Kim, D., Kang, M., Yoon, S.-E.: Volumetric tree: Adaptive sparse graph for effective exploration of homotopy classes. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1496–1503. IEEE, Macau, China (2019)

[92] Park, J., Karumanchi, S., Iagnemma, K.: Homotopy-based divide-and-conquer strategy for optimal trajectory planning via mixed-integer programming. IEEE Transactions on Robotics **31**(5), 1101–1115 (2015)

[93] LIU, Y., ZHENG, Z., QIN, F.: Homotopy based optimal configuration space reduction for anytime robotic motion planning. Chinese Journal of Aeronautics **34**(1), 364–379 (2021)