Locally Homotopic Paths: Ensuring Consistent Paths in Hierarchical Path Planning

Tichakorn Wongpiromsarn[1], Marcelo Kallmann[2] and Andreas Kolling[2]

***Abstract*—We consider a local planner that utilizes model predictive control to locally deviate from a prescribed global path in response to dynamic environments, taking into account the system dynamics. To ensure the consistency between the local and global paths, we introduce the concept of *locally homotopic* paths for paths with different origins and destinations. We then formulate a hard constraint to ensure that local paths are locally homotopic to a given global path. Additionally, we propose a cost function to penalize any violation of this requirement, rather than completely prohibiting it. Experimental results show that both variants of our approach are more resilient to localization errors, compared to existing methods that represent the homotopy class constraint as an envelope around the global path.**

***Index Terms*—Motion and Path Planning; Integrated Planning and Control; Optimization and Optimal Control**

I. Introduction

A Common approach for motion planning in robotic and autonomous systems involves decomposing the problem into multiple levels, each addressing different aspects of the problem [1]–[5]. A typical architecture begins at the top of the hierarchy with a high-level planner that is abstracted away from the detailed mechanics of the system and mainly deals with the overall mission and static or slowly changing aspects of the environment. Its primary role is to compute a coarse "global" path for the robot to accomplish its mission. In multi-agent systems, this high-level planner may also be responsible for the coordination among all the robots to avoid congestion, as illustrated in Figure 1. At the lower end of the hierarchy, a low-level planner, which typically operates at a higher frequency, computes a local path in response to dynamic objects, taking into account the mechanics of the system. To respect the global distribution of the paths and ensure the successful completion of the mission, it is important for the local path to be "consistent" with the global path. This is because the construction of the local path may not take into account all the information used when computing the global path, e.g., the coarse plans of the other robots in the system.

The focus of this work is on the low-level planner, which we refer to as the "local" planner that relies on onboard sensors to locally deviate from the global path in response to a dynamically changing environment and its own (potentially imperfect) state estimates. We consider the case where the local planner utilizes model predictive control (MPC) [6]–[8], which uses a model of the system to predict its future states over a finite time horizon. The local planner computes a local path that optimizes certain objectives within specific constraints, generally expressed with respect to predicted future states and control actions. Objectives and constraints can vary depending on the application, but are often concerned with collisions avoidance, metric deviation and progress along the global path, and dynamic feasibility.

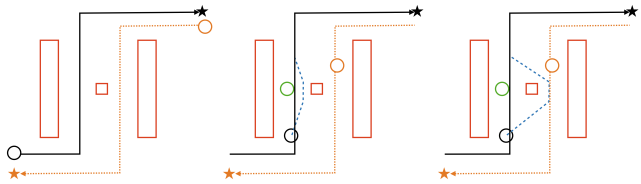

Fig. 1. Examples of global and local paths in multi-agent systems. [Left] The global paths of the two robots (origin denoted by circles, destination by stars) avoid conflicts with each other. The construction of these paths considers only static objects represented by red rectangles and the coarse plans of the other robots in the system. [Middle, right] While following the global path, the black robot encounters a dynamic object (e.g., a human) depicted as a green circle. The local path (dashed blue) of the black robot deviates from the global path to avoid this newly detected object. The middle and right figures illustrate two alternative local paths, both of which start from the robot's current position and are constructed taking into account static and dynamic objects within the sensing range. Due to occlusion, the local planner may not be aware of the presence of the orange robot. Without taking into account the homotopy class of the global path, the local path of the black robot, as illustrated in the right figure, may conflict with the orange robot in order to provide sufficient clearance to the green object.

We propose to utilize the notion of homotopy to establish the consistency between the global and local paths and as a better measure of closeness between the global and local paths. Compared to a standard notion of closeness defined based on a distance metric (which may be captured through a "global path biasing" cost), this concept provides local paths with increased flexibility that is conditioned on the space available. As the global path typically spans from the initial state of the robot to the goal state, its origin and destination are often different from those of the local path, which spans from the current state of the robot to a state at a predefined finite time horizon. Thus, most existing homotopy theory, which considers paths with the same endpoints (i.e., origin and destination), is not directly applicable. To address this, we introduce the concept of locally homotopic paths. Specifically, we consider path planning in the plane with a global path and static and dynamic obstacles within the sensing range

Manuscript received: February, 20, 2024; Revised May, 8, 2024; Accepted June, 27, 2024.

This paper was recommended for publication by Editor Lucia Pallottino upon evaluation of the Associate Editor and Reviewers' comments.

[1] T. Wongpiromsarn holds concurrent appointments with Amazon and Iowa State University. This paper describes the work performed at Amazon. nok@iastate.edu

[2] M. Kallmann and A. Kolling are with Amazon Robotics.

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

as inputs. We then formulate the requirement for local paths to be locally homotopic to the global path as a constraint and cost function for MPC. Our key observation is that the proposed concept of locally homotopic paths leads to a more robust execution in face of localization errors or environment changes, compared to existing approaches that rely on an envelope or metric constraint defined in global coordinates to impose the homotopy class constraint. Throughout the paper, we assume that paths are polygonal, i.e., can be represented by the union of a finite number of connected line segments.

The main contributions of the paper are twofold. First, in Definition 2, we define the concept of *locally homotopic* paths for paths with different endpoints. Second, in Section V and Section VI, we formulate a hard constraint to ensure that local paths are locally homotopic to the global path and a soft constraint that penalizes any violation of this requirement. Experimental results in Section VIII demonstrate the effectiveness of both variants of our approach. These results show that the robot reliably reaches the goal state and stays on the same side of the obstacles as the global path, even in the presence of localization errors, provided that the localization drift does not alter the homotopy class of the global path.

II. Related Work

Existing algorithms for path planning in the plane incorporate the homotopy classes in different ways. For example, Bhattacharya et al. [9] exploits the Cauchy Integral theorem to characterize homotopy classes and proposes a graph-based planning approach to compute an optimal path with homotopy class constraints. Yang et al. [10] defines the concept of *distinguished homotopy* and presents an efficient algorithm to solve the k shortest non-homotopic path planning problem.

Our approach utilizes model predictive control (MPC), which is a versatile technique extensively employed for trajectory generation and tracking [6]–[8], [11], [12]. The popularity of MPC stems from its ability to leverage the model of the system to optimize its future behavior over a finite time horizon, while accounting for constraints on both states and control actions. A key advantage of MPC over graph-based approaches is its ability to handle more precise, nonlinear dynamics.

The construction of the global path can be viewed as solving a combinatorial problem such as determining which side of each obstacle the robot should navigate. These different options lead to multiple local minima, causing difficulty in most gradient-based optimization approaches. Some existing MPC-based approaches address this problem by enumerating all the possible homotopy classes around static obstacles [13], [14]. Other approaches have adopted a local representation of the logical succession of equivalence classes [15], which is to be maintained over time. These approaches, however, can be computationally expensive, especially for multi-agent systems.

More in general, path following has been addressed in previous work in different ways. Howard et al. [16] have focused on navigation in natural terrain by minimizing an objective based on corrective trajectory feasibility and cross-track error. Sheckells et al. [17] have addressed the efficient generation of motion primitives for following a reference path. Our work focuses on scenarios where a robot has to autonomously deviate from unexpected dynamic agents, while being as close as possible to the reference global path in order to not become obstructed by obstacles and to observe global multi-agent path distribution considerations. The proposed approach relies on the concept of locally homotopic paths and demonstrates to be effective in such situations and robust in face of localization errors.

III. Preliminaries

We consider path planning in the plane. Let $X \subset \mathbb{R}^2$ be a topological space. A path in X is a continuous function $\tau : [0,1] \to X$. We say that a path is *polygonal* if its image is the union of a finite number of line segments. A *simple polygon* is the closure of the interior of a simple polygonal closed curve. We assume that X is a polygon with holes [18], [19], i.e., a simple polygon minus the interiors of some other simple polygons, which are called the holes of X.

For any path τ, we let $\overline{\tau}$ denote the reverse of τ, i.e., $\overline{\tau}(s) = \tau(1-s)$ for all $s \in [0,1]$. Given two paths τ_1 and τ_2 with $\tau_1(1) = \tau_2(0)$, their concatenation $\tau_1 \cdot \tau_2$ is the path

$$(\tau_1 \cdot \tau_2)(s) = \begin{cases} \tau_1(2s) & \text{if } s \leq \frac{1}{2} \\ \tau_2(2s-1) & \text{otherwise.} \end{cases}$$

For any $s_1, s_2 \in [0,1]$, a subpath $\tau[s_1, s_2]$ of τ is the path

$$\tau[s_1, s_2](s) = \tau(s(s_2 - s_1) + s_1).$$

The following definition is adapted from [1], [20].

Definition 1. An (endpoint-fixing) *homotopy* between paths τ_1 to τ_2 in X is a continuous function $h : [0,1] \times [0,1] \to X$ such that 1) $h(0,s) = \tau_1(s)$ for all $s \in [0,1]$, 2) $h(1,s) = \tau_2(s)$ for all $s \in [0,1]$, 3) $h(t,0) = \tau_1(0) = \tau_2(0)$ for all $t \in [0,1]$, and 4) $h(t,1) = \tau_1(1) = \tau_2(1)$ for all $t \in [0,1]$.

Conditions 3 and 4 mean that the starting and ending points are hold fixed as h continuously warps τ_1 into τ_2. We say that τ_1 and τ_2 are *homotopic* and write $\tau_1 \sim \tau_2$ if there exists a homotopy between them. Homotopy induces an equivalence relation on the set of all paths from some $x_1 \in X$ to some $x_2 \in X$ [1], [20]. We refer to the equivalence classes as homotopy classes. It is well known [20] that two paths are homotopic if and only if the resulting loop is contractible as stated formally in the following lemma.

Lemma 1. Two paths τ_1 and τ_2 with the same endpoints are homotopic if and only if the loop $\tau_1 \cdot \overline{\tau_2}$ is contractible.

Let $S = \{p_1, p_2, \ldots, p_n\}$ denote the set of *sentinel* points, each chosen arbitrarily inside each of the holes of X. The following lemma, which appears in [21], [22], shows that to test the contractibility of a loop in X, it suffices to work in the simpler space $\mathbb{R}^2 \setminus S$.

Lemma 2. A loop l in X is contractible in $\mathbb{R}^2 \setminus S$ if and only if l is contractible in X.

Combining Lemma 1 and 2, we can check whether two paths τ_1 and τ_2 with the same endpoints are homotopic by

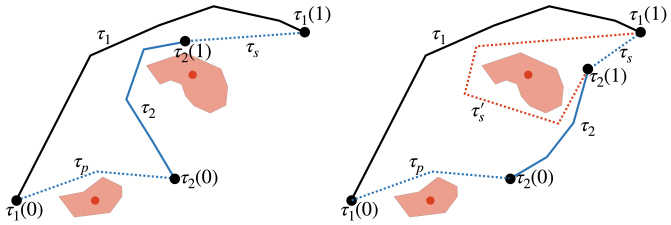

Fig. 2. Construction of τ_p and τ_s in Definition 2. Red polygons are obstacles. Both τ_p and τ_s are monotone with respect to τ_1 based on the standard Euclidean distance. τ_p connects the origin of τ_1 to that of τ_2, while τ_s connects the destination of τ_2 to that of τ_1. [Left] $(\tau_p \cdot \tau_2 \cdot \tau_s) \sim \tau_1$ and thus, by Definition 2, τ_2 is locally homotopic to τ_1. [Right] τ_2 is not locally homotopic to τ_1. First, $(\tau_p \cdot \tau_2 \cdot \tau_s)$ is not homotopic to τ_1. On the other hand, an alternative extension τ'_s results in $(\tau_p \cdot \tau_2 \cdot \tau'_s) \sim \tau_1$ but τ'_s is not monotone with respect to τ_1.

constructing a loop $l = \tau_1 \cdot \overline{\tau_2}$ and checking that no point in S lies in the interior of l.

Our application, however, requires considering a global path and a local path that do not necessarily have common endpoints. As a result, we will define the concept of *locally homotopic* paths. Assume that X is equipped with a function $\rho : X \times X \to \mathbb{R}$ such that (X, ρ) forms a metric space, i.e., $\rho(x_1, x_2)$ defines the distance between x_1 and x_2 for any $x_1, x_2 \in X$. Given a path $\tau : [0,1] \to X$ and a point $x \in X$, we let $\Pi_\tau(x) \in [0,1]$ denote a projection of x onto τ, i.e., $\rho(x, \tau(\Pi_\tau(x))) \leq \rho(x, \tau(s))$ for all $s \in [0,1]$. Given paths τ_1 and τ_2, we say that τ_2 is *monotone* with respect to τ_1 if $\Pi_{\tau_1}(\tau_2(s)) \leq \Pi_{\tau_1}(\tau_2(s'))$ for all $s \leq s'$.

Definition 2. Path τ_2 is *locally homotopic* to path τ_1 if there exist paths τ_p and τ_s such that (L1) $\tau_p(0) = \tau_1(0)$, $\tau_p(1) = \tau_2(0)$, $\tau_s(0) = \tau_2(1)$, and $\tau_s(1) = \tau_1(1)$, (L2) τ_p and τ_s are in X, and (L3) τ_p and τ_s are monotone with respect to τ_1, and (L4) $(\tau_p \cdot \tau_2 \cdot \tau_s) \sim \tau_1$.

Analysis: Conditions (L1)-(L4) ensure that τ_2 can be extended (using τ_p and τ_s) to a path that is in the same homotopy class as τ_1. As illustrated in Figure 2, condition (L1) requires τ_p and τ_s to extend τ_2 to have common endpoints as τ_1. Condition (L4) requires that with the extensions τ_p and τ_s, the resulting path $\tau_p \cdot \tau_2 \cdot \tau_s$ is homotopic to τ_1. Condition (L2) requires τ_p and τ_s to be collision-free. Condition (L3) ensures that the system makes positive progress along τ_1 during these extensions and prevents the system from exhibiting unconventional behavior such as traversing τ_2 in reverse and subsequently circumventing the obstacle from a different side to satisfy (L4) as in the case of τ'_s in Figure 2. Note that if τ_1 and τ_2 have the same endpoints, then they are homotopic if and only if they are also locally homotopic, since τ_p and τ_s in this case are simply paths whose images are points at $\tau_1(0)$ and $\tau_1(1)$, respectively.

Envelope-based methods: A popular approach to ensure that the local path is equivalent to the global path is to impose a constraint, such as an envelope, that corresponds to the homotopy class of the global path. For example, when employing triangulation-based path planning to compute the global path [23], the resulting channel of triangles can be used to form the homotopy class constraint associated with the global path. Figure 3 illustrates how such an envelope can be constructed

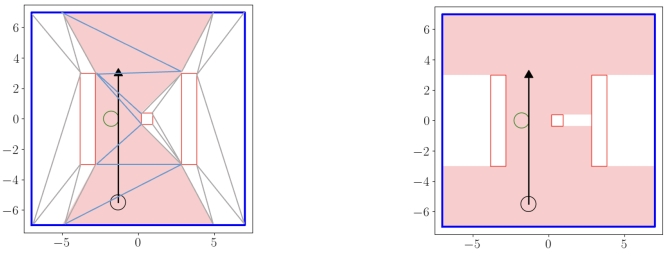

Fig. 3. Construction of path envelope based on Constrained Delaunay Triangulation. [Left] Path envelope constructed from the union of the free triangles containing the path. [Right] Envelope enlarged by adding adjacent convex portions of the free space.

from the union of the free triangles containing the path when the free space is decomposed by a Constrained Delaunay Triangulation (CDT) [24]. As an irregular decomposition, the envelope can contain large or small triangles depending on how the map is defined.

The concept of computing an envelope is also adopted by other methods, with the geometry of the obtained envelope varying according to the underlying planning technique employed [25], [26]. As such an envelope construction typically occurs in the global coordinate frame. A drawback of this approach, as will be later illustrated in Figure 6, is that the resulting constraint may be inaccurate in the presence of localization error. Furthermore, as the local planner has to also deal with dynamic objects, the homotopy class constraint may render the local planning problem infeasible.

IV. Problem Formulation

Consider a system operating in a polygonal world $W \subset \mathbb{R}^2$ containing static obstacles $O_s \subseteq W$ that can be represented by the union of simple polygons. The free space, denoted as $X = W \setminus O_s$ is, therefore, a polygon with holes. Let Z denote the set of states of the system. We assume that a representative position (e.g., the centroid) of the system can be derived from a state and define a function $\mathbb{P} : Z \to \mathbb{R}^2$ such that $\mathbb{P}(z)$ is the position of the system at state z. Given a representative position $p \in \mathbb{R}^2$, we express (using a slight abuse of notation) that $p \in X$ if the entirety of the system resides within X and thus is not in collision with any static obstacles. For instance, a state of a circular differential drive robot may be of the form $z = (x, y, \theta)$, where $(x, y) \in \mathbb{R}^2$ and $\theta \in [0, 2\pi)$ represents the position of the its center and its orientation, respectively. In this case, $\mathbb{P}(z) = (x, y)$ and $\mathbb{P}(z) \in X$ if the distance from the robot's center to any static obstacle exceeds its radius.

Assume that the state of the system evolves according a discrete-time nonlinear time-invariant model

$$z[t+1] = f(z[t], u[t]), \forall t \in \mathbb{N}, \tag{1}$$

where $z[t] \in Z$ and $u[t] \in U$ are the state and control input, respectively, at discrete-time instance t. At each time instance $t \in \mathbb{N}$, the local planner receives the following information: 1) the estimated state $z[t] = z_t \in Z$, 2) the sets $O_s \subset \mathbb{R}^2$ of static obstacles, 3) the set $O_d \subset \mathbb{R}^2$ of dynamic obstacles (e.g., other robots and humans), and 4) the global path $\tau_g : [0,1] \to X$, which is assumed to be polygonal. Note that since τ_g is a path in X, by definition, it ensures

that the system does not overlap with O_s but may overlap with O_d. With this information, the local planner computes the sequence of control actions $\mathbf{u}[t] = (u[t], u[t+1], \ldots, u[t+T])$ that minimizes a given cost function over a given finite time horizon $T \in \mathbb{N}$. Finally, it executes the first value of the optimal control sequence and recomputes the solution at the next time step $t+1$ based on the updated information as O_s, O_d, and τ_g may change over time due to, e.g., localization errors (although we do not explicitly write $O_s[t]$, $O_d[t]$, and $\tau_g[t]$ for the simplicity of the notation).

Specifically, let $\mathbf{z}[t] = (z[t], \ldots, z[t+T+1])$ denote the sequence of states of the system from time t to $t+T+1$. The local planner solves the following constrained optimization problem to obtain the optimal control sequence.

$$\begin{aligned} \min_{\mathbf{u}[t]} \quad & J(\mathbf{z}[t], \mathbf{u}[t], O_s, O_d, \tau_g) \\ \text{s.t.} \quad & z[\tau+1] = f(z[\tau], u[\tau]), \forall \tau \in \{t, \ldots, t+T\}, \\ & z[t] = z_t, \\ & u[\tau] \in U, \forall \tau \in \{t, \ldots, t+T\}, \\ & \mathbb{P}(z[\tau]) \in X, \forall \tau \in \{t, \ldots, t+T+1\}, \\ & g(\mathbf{z}[t], \mathbf{u}[t], O_s, O_d, \tau_g) \leq 0. \end{aligned} \tag{2}$$

Here, J is the cost function that the system tries to minimize. The first three constraints in (2) ensures that the sequence of states respects the system model in (1), starting from the estimated state $z[t] = z_t$ at time t. The fourth condition ensures that the system does not collide with any static obstacles. Finally, the function g describes additional constraints we may impose on the system.

The resulting sequence of states $\mathbf{z}[t] = (z[t], \ldots, z[t+T+1])$ forms a polygonal path $\tau_l : [0,1] \to \mathbb{R}^2$ such that $\tau_l(0) = \mathbb{P}(z[t])$, $\tau_l(1) = \mathbb{P}(z[t+T+1])$, and there exist $s_1, \ldots, s_T \in [0,1]$ with $s_1 \leq s_2 \leq \ldots \leq s_T$ such that $\tau_l(s_i) = \mathbb{P}(z[t+i]) \in X$ for all $i \in \{1, \ldots, T\}$. For the rest of the paper, we will focus on the path, as opposed to the sequence of states, computed by the local planner. As a result, we reformulate the cost function J and the constraint g to take τ_l instead of $\mathbf{z}[t]$ as input. We also ignore their dependence on the control input $\mathbf{u}[t]$ to simplify the notation. With this, the cost function and the constraint are expressed as $J(\tau_l, O_s, O_d, \tau_g)$ and $g(\tau_l, O_s, O_d, \tau_g)$, respectively.

Typically, the constraint g and cost function J are expressed as the summation of multiple terms corresponding to different constraints (e.g, the absence of collision with dynamic obstacles) and objectives (e.g., minimizing the distance to the goal state). In this work, we focus on the specific terms of g and J that we denote as g_h and J_h, respectively. Here, g_h corresponds to the constraint that τ_l is locally homotopic to τ_g, while J_h represents the penalty incurred on τ_l that violates this constraint.

To ensure that τ_l can be extended to a path that is in the same homotopy class as τ_g, our first problem is to impose a hard constraint that τ_l is locally homotopic to τ_g.

Problem 1 (Constraint Formulation)**.** Formulate the constraint $g_h(\tau_l, O_s, O_d, \tau_g)$ to ensure that τ_l is locally homotopic to τ_g.

As a relaxation of Problem 1, we also consider penalizing local paths that are not locally homotopic to the global path.

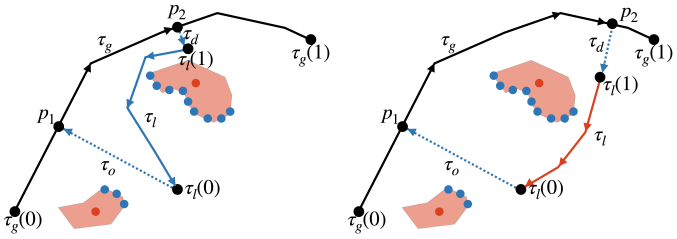

Fig. 4. The construction of projection-based connectors τ_o and τ_d. The set $S(O_s)$ can be constructed from the lidar points (blue dots) or a representative point (red dot) inside each static object (red polygon). The arrows along the paths represent to the loop $l = (\tau_g[s_1, s_2] \cdot \overline{\tau_d} \cdot \overline{\tau_l} \cdot \overline{\tau_o})$. [Left] $(\tau_o \cdot \tau_l \cdot \tau_d)$ is locally homotopic to $\tau_g[s_1, s_2]$ as l does not enclose any point in $S(O_s)$. [Right] $(\tau_o \cdot \tau_l \cdot \tau_d)$ is not locally homotopic to $\tau_g[s_1, s_2]$ as l encloses a point in $S(O_s)$.

Problem 2 (Cost Function Formulation)**.** Formulate the cost function $J_h(\tau_l, O_s, O_d, \tau_g)$ to penalize τ_l that is not locally homotopic to τ_g.

V. Constraint Formulation

According to Definition 2, checking whether path τ_l is locally homotopic to path τ_g requires computing monotone extensions τ_p and τ_s of τ_l such that $(\tau_p \cdot \tau_l \cdot \tau_s) \sim \tau_g$. To avoid this expensive computation, we will utilize the projection Π_{τ_g} to impose a tighter constraint.

Let $s_1 = \Pi_{\tau_g}(\tau_l(0))$, $s_2 = \Pi_{\tau_g}(\tau_l(1))$, $p_1 = \tau_g(s_1)$, and $p_2 = \tau_g(s_2)$, i.e., $p_1, p_2 \in X$ are the projection of the origin and destination, respectively, of τ_l onto τ_g. We then define the connector τ_o as a path in $\mathbb{R}^2$ whose image is a line segment from p_1 to $\tau_l(0)$ and the connector τ_d as a path in $\mathbb{R}^2$ whose image is a line segment from $\tau_l(1)$ to p_2. Specifically, let

$$\begin{aligned} \tau_o(s) &= (1-s)p_1 + s\tau_l(0) \\ \tau_d(s) &= (1-s)\tau_l(1) + sp_2 \end{aligned} \tag{3}$$

Then, we will impose the constraint that

(C1) $\tau_o(s), \tau_d(s), \tau_l(s) \in X$ for all $s \in [0,1]$, and

(C2) $(\tau_o \cdot \tau_l \cdot \tau_d) \sim \tau_g[s_1, s_2]$.

Figure 4 illustrates the construction of s_1, s_2, τ_o, τ_d for different local paths τ_l when ρ is the standard Euclidean distance. The extensions τ_p and τ_s in Definition 2 can then be defined as $\tau_p = (\tau_g[0, s_1] \cdot \tau_o)$ and $\tau_s = (\tau_d \cdot \tau_g[s_2, 1])$. With this construction, it is easy to verify that conditions (L1)-(L3) of Definition 2 are satisfied. Additionally, as τ_p starts with $\tau_g[0, s_1]$ and τ_s ends with $\tau_g[s_2, 1]$, it is easy to check that the satisfaction of condition (C2) implies the satisfaction of condition (L4) of Definition 2.

Let us first consider constraint (C1). Given a path τ in $\mathbb{R}^2$ and region $R \subseteq W$, we define

$$\mathbf{C}(\tau, R) = \begin{cases} 0 & \text{if } \tau(s) \in W \setminus R, \forall s \in [0,1] \\ 1 & \text{otherwise} \end{cases} \tag{4}$$

With this definition $\mathbf{C}(\tau, O_s) = 0$ if and only if $\tau(s) \in X$ for all $s \in [0,1]$. Otherwise, $\mathbf{C}(\tau, O_s) = 1$. Constraint (C1) is then equivalent to $\mathbf{C}(\tau_o, O_s) + \mathbf{C}(\tau_d, O_s) + \mathbf{C}(\tau_l, O_s) \leq 0$.

Next, we will employ Lemma 1 and Lemma 2 to impose constraint (C2). Let $S(O_s) = \{o_1, o_2, \ldots, o_n\} \subseteq O_s$ denote the set of sentinel points of O_s. For the case where O_s is the set of lidar points, we can let $S(O_s)$ be the set of points that

correspond to static objects. If O_s is represented as a set of polygons, each point o_i can be chosen arbitrarily inside each static object. We define the loop $l = (\tau_g[s_1, s_2] \cdot \overline{\tau_d} \cdot \overline{\tau_l} \cdot \overline{\tau_o})$. Applying Lemma 2, constraint (C2) is equivalent to ensuring that o_i does not lie inside l for all $i \in \{1, \ldots, n\}$. This is a standard point-in-polygon problem that is well studied in computational geometry and can be solved using a winding number algorithm [27], [28], which works even for nonsimple polygons. For each $o_i \in S(O_s)$, we define

$$\mathbf{I}(\tau_l, \tau_g, o_i) = \begin{cases} 1 & \text{if } o_i \text{ lies inside } l \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

With this definition, constraint (C2) is equivalent to $\sum_{i \in \{1,\ldots,n\}} \mathbf{I}(\tau_l, \tau_g, o_i) \leq 0$. As a result, we define

$$g_h(\tau_l, O_s, O_d, \tau_g) = \sum_{\tau \in \{\tau_o, \tau_d, \tau_l\}} \mathbf{C}(\tau, O_s) + \sum_{o \in S(O_s)} \mathbf{I}(\tau_l, \tau_g, o), \tag{6}$$

where τ_o and τ_d are constructed from τ_l and τ_g based on (3). Note that the dynamic obstacles O_d have no impact on the homotopy constraint. This is reasonable since they do not impact the construction of the global path τ_g.

The following result ensures that by incorporating g_h as a term of the constraint g in the MPC formulation (2), the resulting local path will be locally homotopic to the global path, provided that the remaining terms of g are non-negative.

Proposition 1. If $g_h(\tau_l, O_s, O_d, \tau_g) = 0$, then τ_l is locally homotopic to τ_g.

VI. Cost Function Formulation

To ensure the feasibility of the optimization problem in (2), we can penalize local paths that are not locally homotopic to τ_g, instead of completely prohibiting such paths. In this case, we assume that each sentinel point $o \in S(O_s)$ is associated with weight $w(o) \in \mathbb{R}_{\geq 0}$ and region $r(o) \subseteq O_s$. The weight $w(o)$ corresponds to the amount of violation of homotopy class constraint associated with o while region $r(o)$ is part of O_s (not necessarily a polygon) associated with o such that $r(o) \cap r(o') = \emptyset$ if $o \neq o'$ and $\bigcup_{o \in S(O_s)} r(o) = O_s$. For example, if O_s is represented as a set of polygons, $w(o_i)$ may represent the size of the corresponding polygon and $r(o_i)$ may simply be the polygon associated with o_i. If O_s is the set of LiDAR points, we can assign $w(o_i) = 1$ for all $i \in \{1, \ldots, n\}$ and define $r(o_i) = \{o_i\}$. As indicated in Section IV, when checking whether a path is in collision with a static obstacle, we need to take into account the entirety of the system (not just its representative point). As a result, having a region that is a point does not pose an issue.

Following the construction of the constraint g in (6), we define the cost function J_h as

$$J_h(\tau_l, O_s, O_d, \tau_g) = \sum_{o \in S(O_s)} w(o) \max \left\{ \mathbf{I}(\tau_l, \tau_g, o), \max_{\tau \in \{\tau_o, \tau_d, \tau_l\}} \mathbf{C}(\tau, r(o)) \right\}. \tag{7}$$

Note that the cost function in (7) employs the maximization operation, contrasting with the summation operation utilized in (6). This is to ensure that for each sentinel point o_i, the violation of condition (C1), which results in $\mathbf{C}(\tau, r(o_i)) > 0$ for some $\tau \in \{\tau_o, \tau_d, \tau_l\}$, and the violation of (C2), which results in $\mathbf{I}(\tau_l, \tau_g, o_i) > 0$, is accounted for only once when computing the cost of τ_l.

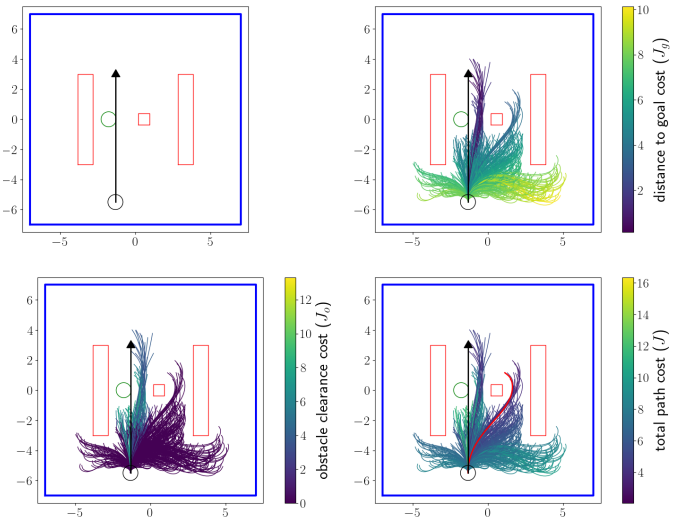

Fig. 5. [Top left] The experiment setup with the world W (blue rectangle), static obstacles O_s (red rectangles), dynamic obstacles O_d (green circle), global path τ_g (black arrow), and robot's initial position (black circle). [Top right, bottom left] Valid candidate paths color-coded based on their associated distance-to-goal (J_g) and obstacle-clearance (J_o) cost, as indicated by the color bar on the plot. [Bottom right] Valid candidate paths color-coded based on their total costs $J = J_g + J_o$ as well as the optimal path computed by the baseline planner shown in red.

Based on the construction of J_h, we can derive the following property of J_h.

Proposition 2. If $J_h(\tau_l, O_s, O_d, \tau_g) = 0$, then τ_l is locally homotopic to τ_g.

VII. Computational Complexity

Let N_g and N_l denote the number of segments in τ_g and τ_l, respectively. The computation of constraint g_h and cost function J_h involves computing the projection $s_1 = \Pi_{\tau_g}(\tau_l(0))$ and $s_2 = \Pi_{\tau_g}(\tau_l(1))$, both of which take $O(N_g)$ time. For each $o \in S(O_s)$, computing $\mathbf{I}(\tau_l, \tau_g, o)$ is equivalent to determining whether o lies within the polygon formed by the loop $l = (\tau_g[s_1, s_2] \cdot \overline{\tau_d} \cdot \overline{\tau_l} \cdot \overline{\tau_o})$. This computation can be efficiently performed in $O(N_g + N_l)$ time using a winding number algorithm [27], [28]. Finally, the computation of $\mathbf{C}(\tau, r(o))$ for each $\tau \in \{\tau_o, \tau_d, \tau_l\}$ represents a standard collision checking problem, whose computational complexity varies depending on the chosen implementation (including the representation of the obstacles) and the specific approximations employed. As collision checking is an integral part of the standard safety constraint in most systems, the computation of $\mathbf{C}(\tau, r(o))$ should not incur additional computational complexity. In summary, excluding collision checking, the computational complexity of computing g_h and J_h is $O(n(N_g+N_l))$, where n is the number of sentinel points of O_s.

VIII. Experimental Results

We consider a circular differential drive robot with radius $R = 0.5$ meters operating in the environment with static

and dynamic obstacles shown in Figure 5. Note that the categorization of obstacles into static and dynamic should not be determined solely by their instantaneous speed. Instead, it should be based on their inherent nature or type. For example, a wall is considered static, while a robot is considered dynamic even if it is stationary. The state $s = (x, y, \theta)$ of the robot evolves according to the following equations.

$$\begin{aligned} x[t+1] &= x[t] + u_v \Delta t \cos(\theta), \\ y[t+1] &= y[t] + u_v \Delta t \sin(\theta), \\ \theta[t+1] &= \theta[t] + u_\omega \Delta t, \end{aligned} \tag{8}$$

where (x, y) is the position of the robot's center, θ is its heading, $u = (u_v, u_\omega)$ is the control input, $u_v \in [0, 2\text{m/s}]$ and $u_\omega \in [-1.674\text{s}^{-1}, 1.674\text{s}^{-1}]$ are the linear and angular speed, respectively, and $\Delta t = 0.1$ second is the duration between consecutive time instances.

We employ NLopt [29] to solve the constrained optimization problem in (2) using the Improved Stochastic Ranking Evolution Strategy (ISRES) [30]. Additionally, we integrate the equilibrium point control technique proposed in [31] that utilizes a compact parametrization of trajectories to efficiently solve the optimization problem. To comprehensively assess our proposed approach, we compare it against two established methods: a baseline and envelope-based methods.

Baseline planner: As a fundamental reference point, we consider a simple local planner with cost function $J = J_g + J_o$, J_g penalizes the distance from the destination of τ_l to the destination of τ_g, while J_o penalizes the violation of the required clearance to dynamic obstacles. We also impose a constraint $g = g_o$ to ensure the absence of collision with dynamic obstacles. Formally,

$$\begin{aligned} J_g(\tau_l, O_s, O_d, \tau_g) &= c_g \rho(\tau_l(1), \tau_g(1)), \\ J_o(\tau_l, O_s, O_d, \tau_g) &= c_o \sum_{s \in H} \max\Big(D - \min_{p \in O_d} \rho\big(\tau_l(s), p\big), 0\Big), \\ g_o(\tau_l, O_s, O_d, \tau_g) &= \sum_{s \in H} \max\Big(R - \min_{p \in O_d} \rho\big(\tau_l(s), p\big), 0\Big). \end{aligned}$$

Here, $c_g, c_o \geq 0$ are the constants that correspond to the weights of the associated cost terms, $\rho(p_1, p_2)$ is the standard Euclidean distance between p_1 and p_2, $H \subset [0, 1]$ is a finite set of path parameters of τ_l where the distance to dynamic obstacles is evaluated, $D \geq R$ is the required clearance from the center of the robot to dynamic obstacles, and $\min_{p \in O_d} \rho\big(\tau_l(s), p\big)$ is the distance from $\tau_l(s)$ to O_d. In all the experiments, we set $c_g = 1$, $c_o = 0.5$, $D = 4R$, and H corresponds to the time instances $t, \ldots, t+T+1$ in (2). While g_o and J_o are similar, their key difference is that g_o imposes a hard constraint while J_o is a soft constraint that encourages the robot to stay away from dynamic obstacles.

Envelope-based planner: The envelope-based planner augments the baseline planner with an additional constraint g_e that represents the envelope around the global path. The envelope defines the restricted homotopy class to be considered and is constructed from a CDT as depicted in Figure 3. The envelope-based planner optimizes the same cost function as the baseline planner ($J = J_g + J_o$) but with constraint $g = g_o + g_e$ to ensure that a valid candidate path is not only collision-free but also within a given envelope.

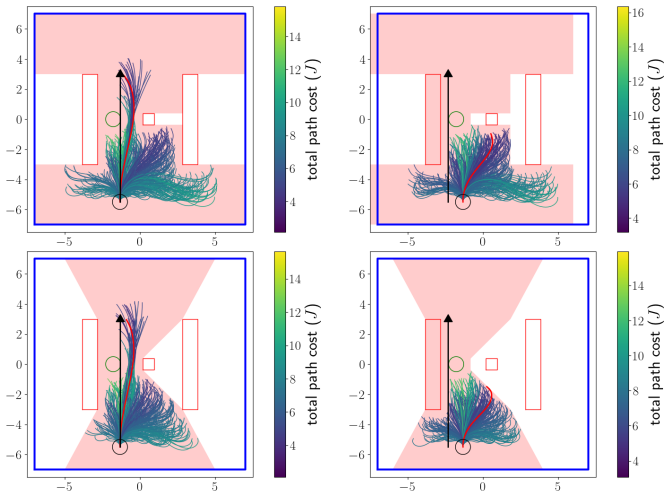

Fig. 6. The result, employing the same setup and annotation as in Figure 5, but with an additional path envelope (red region) that corresponds to the homotopy class constraint. The top and bottom rows correspond to different envelopes constructed as described in Figure 3. [Left column] When there is no localization error, the optimal path is closer to the dynamic obstacle to stay within the given envelope. [Right column] A localization error of 1 meter results in a shift in the global path and the envelope, which impacts the valid candidates and consequently alters the resulting optimal path, preventing it from reaching the destination of the global path.

Single time instance optimization results: First, we configure the timeout of NLopt to 1.0 second to investigate the candidate paths explored by NLopt within this time limit and the resulting optimal path for different formulations of constraint g and cost function J. We say that a candidate path τ_l is "valid" if it satisfies $g(\tau_l, O_s, O_d, \tau_g) \leq 0$ and $\tau_l(s) \in X, \forall s \in H$.

Figure 5 shows the results for the baseline planner. The optimal path τ_l is not locally homotopic to τ_g due to the influence of the clearance cost J_o, which encourages the system to maintain the required clearance D from the dynamic obstacle.

Figure 6 shows the results for the envelope-based planner. Suppose that the global path and the envelope are defined in the global frame, while the static and dynamic obstacles are defined in the local frame (i.e., a frame attached to the robot).[1] As a result, a localization error will cause a shift in the global path and the envelope relative to the robot and the obstacles. Figure 6 demonstrates that errors of this nature can have significant impact on the quality of the local path.

Figure 7 shows the results of the constraint-based variant of our approach, where we incorporate the constraint g_h proposed in (6) instead of an envelope around the path. Specifically, we let $g = g_o + g_h$, while keeping $J = J_g + J_o$. We simply position the sentinel points o_1, o_2, o_3 at the centroid of each static obstacle (the result is unaffected by the location of o_i as long as it is within the associated rectangle). The average computation time for g_h is 140.44 microseconds. In the absence of localization errors, the valid candidate paths and the resulting optimal path closely resemble those in Figure 6. This illustrates that g_h has a comparable effect to imposing an envelope around the global path. As opposed to imposing

[1]This choice of coordinate frames is natural for many systems, as the global path typically aims to reach a destination specified in the global frame, while the robot utilizes onboard sensing for object detection.

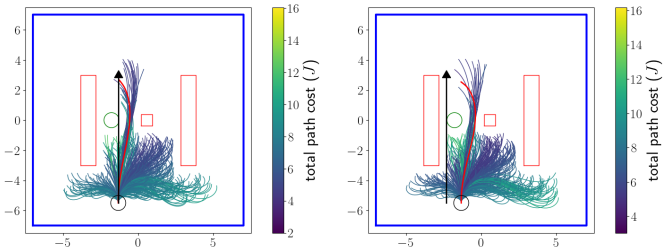

Fig. 7. The result, employing the same setup and annotation as in Figure 5, but with the addition of constraint g_h defined in (6). [Left] With no localization errors, the results are similar to those when imposing an envelope around the global path (Figure 6 left). [Right] A localization error of 1 meter, which causes a shift in the global path, minimally impacts the valid candidates and the resulting optimal path.

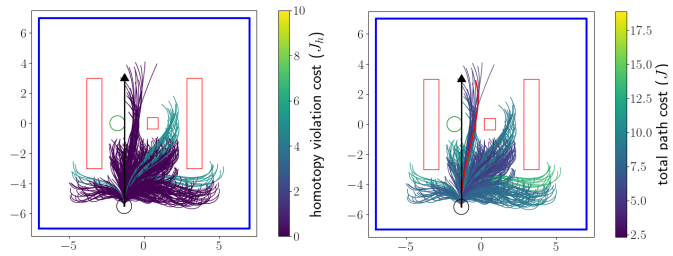

Fig. 8. The result employing the same setup and annotation as in Figure 5, but with the addition of the homotopy violation cost J_h defined in (7). In this case, the total cost $J = J_g + J_o + J_h$.

an envelope around the global path, localization errors do not affect g_h, as long as the shift in the global path does not affect its homotopy class (with respect to the static obstacles).

Finally, we evaluate the cost-based variant of our approach by incorporating the cost function J_h in (7) and setting $J = J_g + J_o + J_h$ and $g = g_o$. For all $i \in \{1, 2, 3\}$, we define $w(o_i) = 5$ and $r(o_i)$ straightforwardly corresponds to the associated rectangle. The average computation time for J_h is 158.38 microseconds. As shown in Figure 8, the candidate paths are similar to those in Figure 5, which include those that are not locally homotopic to τ_g, while the optimal path is similar to the one in Figure 7 and is locally homotopic to τ_g. Figure 9 demonstrates that similar to the case of imposing g_h, localization errors do not affect J_h, as long as the global path remains within the same homotopy class.

Receding horizon planning results: We now simulate the robot by running the local planner in a receding horizon manner, i.e., at each time instance t, execute the first value $u^*[t]$ of the optimal control actions $\mathbf{u}^*[t] = (u^*[t], \ldots, u^*[t+T])$ and recompute the optimal control actions as the robot moves. We configure the timeout of NLopt to 0.1 second to ensure that the robot react promptly the dynamically changing environment. For each formulation of constraint g and cost function J, we perform 50 simulation runs. The resulting paths of the robot, depicted in Figure 10, illustrate the effectiveness of our approach. Both variants, relying on constraint g_h and cost function J_h, demonstrate resilience to localization errors, in contrast to the path envelope approach, which causes the robot to fail to reach the goal state when faced with localization errors. Furthermore, the issue of NLopt not having sufficient time to converge to the optimal solution is observed when

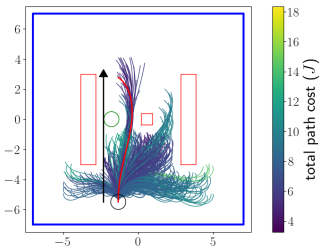

Fig. 9. The result employing the same setup and annotation as in Figure 8, but with a localization error of 1 meter, causing a shift in the global path.

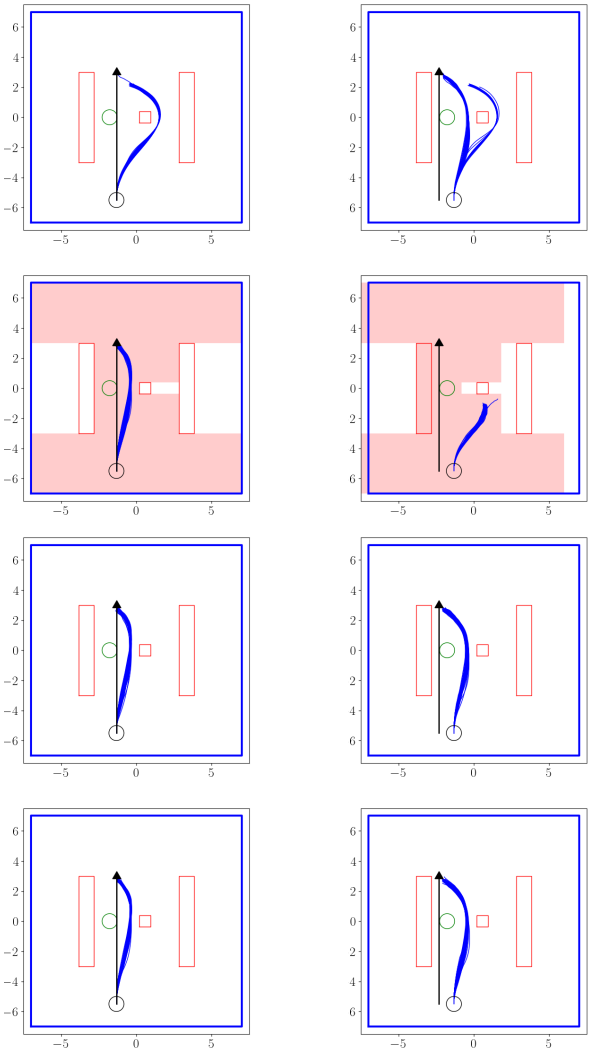

Fig. 10. Simulation results for different formulations of constraint g and cost function J when the local planner is run in a receding horizon manner. The left column corresponds to the case where there is no localization error, while the right column corresponds to the case where there is a 1-meter localization error. [First row] the baseline planner, [second row] the envelope-based planner, [third row] the constraint-based variant, and [fourth row] the cost-based variant of our approach.

employing the baseline planner. As shown in the right plot on the first row of Figure 10, in some runs, the robot remains on the same side of the obstacle as the global path, while in the others, it does not.

Figure 11 shows the results when applying our approach to a more complex setup with a longer-duration mission and more static and dynamic obstacles. To effectively handle missions

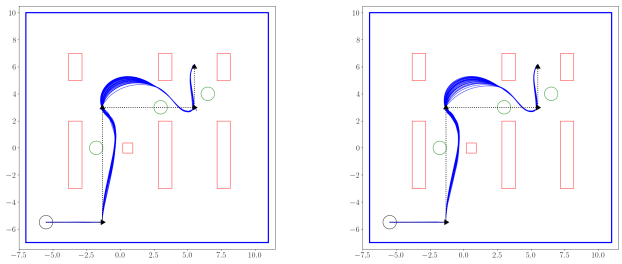

Fig. 11. Simulation results of the cost-based [left] and constraint-based [right] variants of our proposed approach in a more complex scenario.

of considerable length, we implement a common strategy that involves using an intermediate goal, rather than the final destination, when computing the distance-to-goal cost (J_g). Since the global path is a sequence of line segments, each endpoint of these segments serves as a natural intermediate goal. As the robot progresses along the global path and reaches a point sufficiently close to the end of the current path segment, it updates its intermediate goal to be the endpoint of the next path segment. A total of 50 simulation runs were conducted in this challenging scenario to comprehensively evaluate the performance of our approach and verify that the 0.1-second NLopt timeout does not compromise the quality of the solution. The results demonstrate the effectiveness of both variants of our approach in successful navigation of this complex environment, with the path followed by the robot in all simulation runs belonging to the same homotopy class as the global path.

IX. Conclusion and Future Work

We defined the concept of locally homotopic paths for paths with different endpoints to establish the equivalence between global and local paths in hierarchical path planning. We also formulated a hard constraint and a soft constraint suitable for integration into an MPC formulation to ensure that local paths are locally homotopic to a given global path or penalize any violation of this requirement. Experimental results demonstrated the effectiveness of both variants of our approach in comparison with existing approaches that rely on an envelope around the global path to enforce the homotopy class constraint. Future work includes evaluating the proposed constraint in real scenarios, exploring other extensions τ_p and τ_s of local paths and how the introduction of g_h and J_h affect the resulting paths when J and g include more complex terms.

References

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

[2] M. Campbell, M. Egerstedt, J. P. How, and R. M. Murray, "Autonomous driving in urban environments: approaches, lessons and challenges," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4649–4672, 2010.

[3] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *2011 IEEE International Conference on Robotics and Automation*, pp. 1470–1477, 2011.

[4] D. Zarubin, V. Ivan, M. Toussaint, T. Komura, and S. Vijayakumar, "Hierarchical Motion Planning in Topological Representations," in *Robotics: Science and Systems VIII*, The MIT Press, 07 2013.

[5] Y. Qi, B. He, R. Wang, L. Wang, and Y. Xu, "Hierarchical motion planning for autonomous vehicles in unstructured dynamic environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 496–503, 2023.

[6] M. Morari and J. H. Lee, "Model predictive control: past, present and future," *Computers & Chemical Engineering*, vol. 23, no. 4, pp. 667–682, 1999.

[7] A. Bemporad, "Model predictive control design: New trends and tools," in *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 6678–6683, 2006.

[8] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2017.

[9] S. Bhattacharya, V. Kumar, and M. Likhachev, "Search-based path planning with homotopy class constraints," in *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, p. 1230–1237, AAAI Press, 2010.

[10] T. Yang, L. Huang, Y. Wang, and R. Xiong, "Efficient search of the k shortest non-homotopic paths by eliminating non-k-optimal topologies," 2022.

[11] F. Seccamonte, J. Kabzan, and E. Frazzoli, "On maximizing lateral clearance of an autonomous vehicle in urban environments," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 1819–1825, 2019.

[12] Z. Zhang, L. Zheng, Y. Li, Y. Yu, and X. Qiao, "Model predictive control for path following of autonomous vehicle considering model parameter uncertainties," in *2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM)*, pp. 207–212, 2021.

[13] J. Park, S. Karumanchi, and K. Iagnemma, "Homotopy-based divide-and-conquer strategy for optimal trajectory planning via mixed-integer programming," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1101–1115, 2015.

[14] V. Z. Patterson, F. E. Lewis, and J. C. Gerdes, "Optimal decision making for automated vehicles using homotopy generation and nonlinear model predictive control," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1045–1050, 2021.

[15] R. A. Knepper and M. T. Mason, "Improved hierarchical planner performance using local path equivalence," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3856–3861, 2011.

[16] T. M. Howard, R. A. Knepper, and A. Kelly, *Constrained Optimization Path Following of Wheeled Robots in Natural Terrain*, pp. 343–352. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[17] M. Sheckells, T. M. Caldwell, and M. Kobilarov, "Fast approximate path coordinate motion primitives for autonomous driving," in *IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017.

[18] M. Bern and P. Plassmann, "Chapter 6 - mesh generation," in *Handbook of Computational Geometry* (J.-R. Sack and J. Urrutia, eds.), pp. 291–332, Amsterdam: North-Holland, 2000.

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

[20] J. R. Munkres, *Topology*. Prentice Hall, 2 ed., 2000.

[21] J. Erickson, "Testing homotopic paths in the plane." Lecture Note, 2009. Available at: https://jeffe.cs.illinois.edu/teaching/comptop/2009/notes/testing-contractibility.pdf.

[22] S. Cabello, Y. Liu, A. Mantler, and J. Snoeyink, "Testing homotopy for paths in the plane," *Discrete & Computational Geometry*, vol. 31, pp. 61–81, Jan 2004.

[23] M. Kallmann, "Dynamic and robust local clearance triangulations," *Transactions on Graphics*, vol. 33, no. 5, p. 1–17, 2014.

[24] P. L. Chew, "Constrained delaunay triangulations," *Algorithmica*, vol. 4, pp. 97–108, 1989.

[25] R. Geraerts and M. H. Overmars, "The corridor map method: Real-time high-quality path planning," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 1023–1028, 2007.

[26] J. Choi, "Kinodynamic motion planning for autonomous vehicles," *International Journal of Advanced Robotic Systems*, vol. 11, 2014.

[27] D. Sunday, "Inclusion of a point in a polygon." *Available at https://web.archive.org/web/20130126163405/http://geomalgorithms.com/a03-_inclusion.html*, 2013.

[28] G. N. Kumar and M. Bangi, "An extension to winding number and point-in-polygon algorithm," *IFAC-PapersOnLine*, vol. 51, no. 1, pp. 548–553, 2018. 5th IFAC Conference on Advances in Control and Optimization of Dynamical Systems ACODS 2018.

[29] S. G. Johnson, "The NLopt nonlinear-optimization package." https://github.com/stevengj/nlopt, 2007.

[30] T. P. Runarsson and X. Yao, "Search biases in constrained evolutionary optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, pp. 233–243, 2005.

[31] J. J. Park, C. Johnson, and B. Kuipers, "Robot navigation with model predictive equilibrium point control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4945–4952, 2012.