Optimal Trajectory Generation Under Homology Class Constraints

Soonkyum Kim Koushil Sreenath Subhrajit Bhattacharya Vijay Kumar

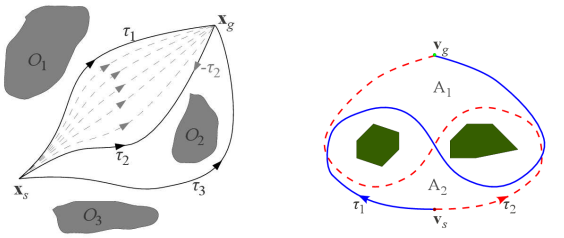
***Abstract*— There are many applications where topology constraints are useful in trajectory generation for mobile robots. In this paper we present a method to generate an optimal trajectory restricted to a particular homology class. The optimality is achieved by formulating the trajectory generation problem as a Mixed-Integer Quadratic Program (MIQP). We introduce binary variables that not only encode information about the satisfaction of geometric constraints, but also incorporate information about the homology class. We define the h-signature, a complete homology class invariant, as a quadratic function of the binary variables, which we subsequently convert to a linear function by variable substitutions. As a result, the suggested trajectory generation problem under homology class constraints can still be formulated as a MIQP, which can be solved by an anytime solver like CPLEX. We illustrate the method with examples of minimum acceleration trajectory generation under different homology class constraints with potential application to differentially-flat systems with a two-dimensional flat output space.**

I. Introduction

Trajectory generation algorithms for robotic systems is one of the most active areas in robotics research. Some literatures focus on finding optimal trajectories in convex or unbounded spaces [1], [2]. However, with development of computational capabilities, significant research interest has been focused on algorithms for generating trajectories in cluttered, non-convex environments with kinematic and dynamic constraints in the form of constraints on communication, coverage, environment, time, etc (see kinodynamic planners [3], RRT trees [4], LQR trees [5], Elastic Roadmaps [6], MILP [7], [8] and references within). Most of the algorithms are developed to find optimal trajectories satisfying feasibility constraints. However, there have also been considerable amount of research interest in algorithms for generating trajectories for multi-agent problems [9], [10], [11]. In such problems it is often required that each robot follows different trajectories to cover or sense the whole work space as in search-and-rescue or surveillance problems. This brings forth the necessity of finding trajectories in topologically different classes. This requires that we impose constraints on the homotopy classes of the trajectories accordingly. However, in many practical robotic problems, homology class constraints act as suitable and convenient substitutes for homotopy class constraints [12]. In this paper we propose an anytime algorithm for finding optimal trajectories in specific homology classes (*i.e.* finding trajectories with homology class constraints), satisfying smoothness conditions, and minimizing a cost function that is not necessarily the length of the trajectory, but involves higher order derivatives.

Early attempts at classifying homotopy classes in two dimensions include geometric methods [13], [14], homotopy preserving probabilistic road-map constructions [15], and triangulation-based path planning [16]. Two trajectories are said to be homotopic if one can be continuously deformed to another without intersecting any obstacle. Each set of trajectories that are homotopic forms an equivalence class, called a homotopy class (see Figure 1(a)). A particular homotopy class can be specified by a representative trajectory in that class. Thus, trajectory generation with homotopy class constraints consists of finding an optimal trajectory in the desired homotopy class, specified by the given representative trajectory, that also respects the kinematic constraints. One can think of applications ranging from multi-robot exploration, where it may be beneficial to deploy each robot in a different homotopy class to ensure maximal coverage and minimal congestion, to single arm motion planning where one may seek paths that go around obstacles one way or the other based on the specific task.

Our objective in this paper is to design optimal trajectory for a robot that minimizes an integral cost functional (which depends on the trajectory), while also respecting kinematic constraints of the system, avoiding obstacles, and constraining the trajectory to a particular homology class. The kinematic constraint that we will consider in particular is that the trajectory needs to be smooth (r-differentiable). We will however not consider bounds on curvature in the present work. Although several of these subproblems have been solved separately (see [3], [4], [5], [17], [18], [12]), there is no literature, to our knowledge, that addresses the combined problem described above. We assume a planar robot of finite radius and with a safety padding around it, and represent a planar trajectory of its center, $q(t)$, by the points $[q_x(t), q_y(t)]$ parametrized by t. We use mixed-integer quadratic programming [19], [20] to check feasibility of each intermediate point on the trajectory and to divide the configuration space into subsets. We start by assuming that the required homology class is specified by a value of the h-signature, which will be defined later in Section III-B. Specifically, the suggested algorithm is an anytime algorithm, and we can achieve an optimal trajectory that is guaranteed to have the desired h-signature value while satisfying the kinematic and topological constraints. This method can be used for trajectory generation for differentially-flat systems [21] with a two-dimensional flat output space, such as a kinematic car [22], or a tricycle robot [23], which not only produces a trajectory respecting the homology constraint, but also provides the nominal feed-forward forces (due to the differential-flatness property,) for use in feedback control for



(a) (b)

Fig. 1. (a) τ_1 is homotopic to τ_2 since there is a continuous sequence of trajectories representing deformation of one into the other. τ_3 belongs to a different homotopy class since it cannot be continuously deformed into any of the other two. (b) Example where the trajectories (τ_1 and τ_2) are homologous, but not homotopic.

trajectory tracking.

The outline for the rest of the paper is as follows: Section II briefly reviews some related previous works like h-signature and MIQP. Section III describes the algorithm to find optimal trajectories that are in a specified homology class and satisfy other kinematic and dynamic constraints. Section IV presents an example of finding optimal trajectories with homology class constraints for a planar mobile robot. Finally, Section V provides concluding remarks.

II. Preliminaries

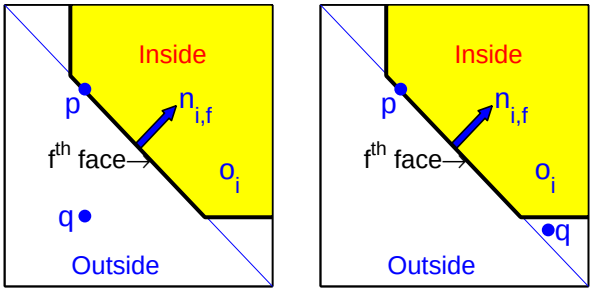
A. Homotopy and Homology Classes for Trajectories

We begin by defining *homologous* trajectories and illustrate the difference between homology and homotopy. Two trajectories, q_1 and q_2, connecting the same start and end points are homologous if and only if the closed loop formed by them, $q_1 \bigsqcup -q_2$ (*i.e.*, q_1 together with q_2 with opposite orientation), forms the boundary of a 2-dimensional region on the plane not containing/intersecting any obstacle. It can be shown that [12] homology is a coarser representation of homotopy, with trajectories that are homotopic being also homologous. Figure 1(b) shows a good example of two trajectories, which are homologous but not homotopic.

A compact representation for the homology classes of trajectories is the *h-signature*, computed using the Cauchy integral theorem and the Residue theorem from complex analysis, as proposed by Bhattacharya et al. [18], [12]. The h-signature of a trajectory, q, with respect to obstacle o_j is defined as

$$H_j(q) = \int \frac{1}{z - z_j} dz \qquad (1)$$
$$= \int_{t_0}^{t_f} \frac{1}{q_x(t) + iq_y(t) - z_j} \left(\dot{q}_x(t) + i\dot{q}_y(t)\right) dt$$

where $z(t) = q_x(t) + iq_y(t)$ is the complex representation of the trajectory, and z_j is the complex representation of an arbitrary point inside obstacle o_j. Then the h-signature with respect to all obstacles is given by $H = [H_1, ..., H_{n_o}]^T$ with n_o obstacles. However, since this h-signature is nonlinear in the optimization variables that we will use for describing the trajectories, we will define the h-signature using a different formulation which is consistent with its definition of being a complete invariant for homology classes.



(a) (b)

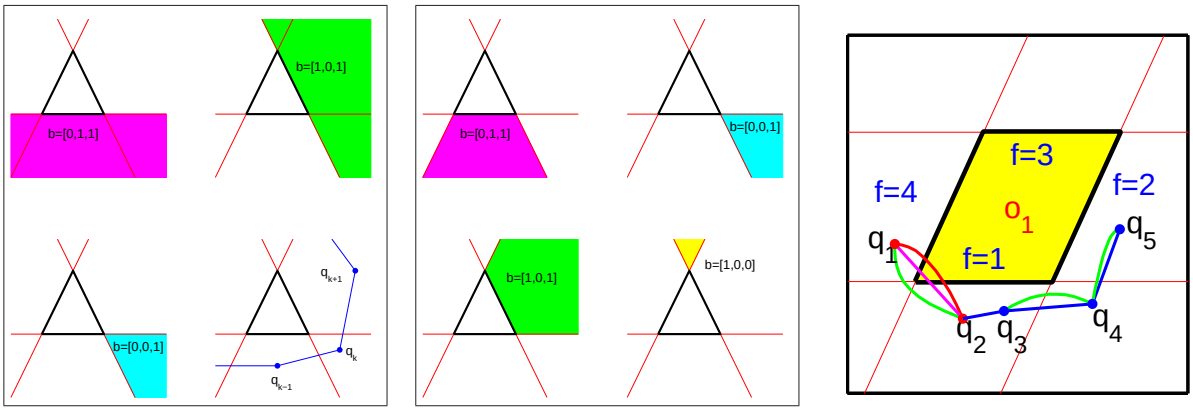
Fig. 2. The normal vector, $n_{i,f}$, of the f^{th} face of obstacle o_i is pointing inward. p is an arbitrary point on the f^{th}face. (a) An example of $q \in Q$ when $b_{i,f} = 0$. (b) An example of $q \in Q$ when $b_{i,f} = 1$.

B. Optimal Trajectory Generation

We consider trajectory planning in a compact subset $Q \subset \mathbb{R}^2$ of a plane. Let $O = \{o_1, o_2, \cdots, o_{n_o}\}$ be a set of convex, pair-wise disjoint *obstacles* in Q (The requirement of convexity of obstacles can be relaxed by considering a set of arbitrarily-shaped obstacles such that their convex hulls are pair-wise disjoint). Each obstacle $o_i \in O$ can be represented by a n_i-sided convex polygon, whose faces define hyperplanes that partition Q into two half-spaces. A binary variable is used to indicate whether a point is on the feasible side of the hyperplane, as described in [20]. So a point $q \in Q$ will be feasible and will avoid collision with an obstacle o_i if there is at least one face $f \in [1, ..., n_i]$ satisfying $n_{i,f} \cdot q \leq s_{i,f}$. Where $n_{i,f}$ is a normal vector to the f^{th} face of obstacle o_i pointing inward, and $s_{i,f} = n_{i,f} \cdot p$, for an arbitrarily chosen point p on the f^{th} face as shown in Figure 2. Similar to obstacle avoidance using binary variables $b_{i,f}$, as described in [20], a given point is feasible with respect to obstacle o_i if

$$n_{i,f} \cdot q \leq s_{i,f} - \delta_r + M b_{i,f} \quad \text{for} \quad f = 1, ..., n_i \qquad (2)$$
$$\sum_{f=1}^{n_i} b_{i,f} \leq n_i - 1,$$

where $b_{i,f} \in \{0, 1\}$ are binary variables (with $b_{i,f} = 0$ indicating that the point lies on the feasible side of the f^{th} face of the i^{th} obstacle as shown in Figure 2(a)), and $M > 0$ is a large positive number. $\delta_r \geq 0$ is the radius of the disk encircling the finite-sized robot, along with some safety-padding around it. The second inequality in (2) implies that the point q will be feasible with respect to at least one face, *i.e.*, for a given i, there exists at least one f such that $b_{i,f} = 0$. Although (2) is a sufficient condition for feasibility, this formulation breaks up Q into overlapping subsets as shown in Figure 3(a). The first three plots in Figure 3(a) illustrate that the subset corresponding to $b = [0, 0, 1]$ is the intersection of the two subsets corresponding to $b = [0, 1, 1]$ and $b = [1, 0, 1]$. Considering the segment of trajectory in the last plot of Figure 3(a), the binary variable vector b_k, corresponding to the point q_k, is not unique, but could be any of $b = [0, 1, 1]$, $b = [0, 0, 1]$ and $b = [1, 0, 1]$. As a result, we can have the same trajectory (represented by the points on it) described by different sets of binary



(a) (b) (c)

Fig. 3. (a) Overlapping subsets divided by values of binary variables representing each face of triangular obstacle. (b) Disjointed cells divided by values of binary variables representing each face but considering additional constraint. (c) An example of parallelogram obstacle. f is the index of each face. Red and magenta curves are infeasible trajectories between two feasible configurations, q_1 and q_2. Adjacent intermediate points(q_3, q_4 and q_5) are satisfying additional constraint.

variables. Such duplication increases the size of the feasible region in the space of binary variables, resulting in redundant searches, and larger computation times. To eliminate such cases, we introduce some additional inequality constraints to build disjointed cells like in Figure 3(b),

$$-n_{i,f} \cdot q \leq -s_{i,f} + \delta_r + M(1 - b_{i,f}) \text{ for } f = 1, ..., n_i. \tag{3}$$

The first inequality of (2) only guarantees that the point q is on the feasible side or outside of f^{th} face when $b_{i,f} = 0$. But the constraint (3) enforces that the point q be on the other side when $b_{i,f} = 1$. Thus, the feasible region, Q, is partitioned into disjoint cells, each of which is bounded by hyperplanes defined by the faces of the obstacles. (see Figure 3(b)). Moreover, each cell can be identified by a unique vector of binary variables, $b = [b_1^T, \ldots, b_{n_o}^T]^T$ where $b_i = [b_{i,1}, \ldots, b_{i,n_i}]^T \in \{0,1\}^{n_i}$.

We parametrize the trajectory by splicing N_s segments of trajectories, each parametrized by linear combination of $N_p + 1$ basis functions,

$$q(t) = \sum_{k=0}^{N_p} c_{j,k}\ e_k(t - t_j) \quad \text{for} \quad t_j \leq t < t_{j+1}, \tag{4}$$

for $j \in [0, ..., N_s - 1]$, $0 = t_0 \leq t_1 \leq ... \leq t_{N_s} = t_f$. Where $e_k(t)$ is any basis function and $c_{j,k}$ are coefficients. The trajectory is restricted to be k_r-times differentiable at the junction of each of the segments of trajectories, $q(t_j)$, for $j \in [1, ..., N_s - 1]$. Further, obstacle avoidance is achieved by enforcing (2) at some equally distributed intermediate points on each segment of trajectories. we choose the cost function to be the integration of the square of the norm of r^{th}-derivative of the trajectory:

$$J(c) = \int_{t_0}^{t_f} \left\| \frac{d^r q(t)}{dt^r} \right\|^2 dt \ = \ c^T H c. \tag{5}$$

where $c = [c_0^T, ...c_{N_s-1}^T]^T$, and H depends only on the choice of the basis functions (note that we could choose a cost function that is a weighted sum of different order derivatives, and still keep it quadratic in c). The optimal trajectory generation problem can then be simplified as the following MIQP (Mixed-Integer Quadratic Program),

$$\begin{aligned} \min_{c,\ \widetilde{b}} \quad & c^T H c \\ \text{s.t.} \quad & A_f c + D_f \widetilde{b} \leq g_f \\ & A_b \widetilde{b} \leq g_b \\ & A_{eq} c = 0 \end{aligned} \tag{6}$$

where $\widetilde{b}$ is the vector formed by stacking all the binary vectors, b_k, corresponding to the intermediate points, q_k, of the trajectory and hence is a coarse representation of the continuous trajectory $q(t)$. The first inequality captures the feasibility constraints of (2) for the intermediate points, the second inequality captures the constraint on sum of binary variables in (2), and $A_{eq}c = 0$ imposes r^{th} order differentiability at the junction of the segments of trajectories and the boundary conditions of initial configuration, $q(0) = q_0$ and final configuration $q(t_f) = q_f$.

To find an optimal trajectory in a specific homology class, we can then add some topological constraints. If we add a constraint on the h-signature of (1), which we described earlier, such that the h-signature of the trajectory, $H(q)$, should be some desired H_d, the quadratic program (6) becomes a non-convex problem. Furthermore, the gradient of the new constraint, $H = H_d$, will be zero almost everywhere, because the value of the h-signature does not change within a particular homology class (*i.e.* the range of the h-signature is a set of discrete variables). So, the resulting problem is a non-convex problem, which is numerically hard to solve based on gradients of cost and constraints. So, we need a different way to enforce topological constraints, and this is described in the next Section.

III. Algorithm Description

In this Section, we will describe our algorithm to generate the optimal trajectory with homology constraints while ensuring that the problem remains a MIQP.

A. Additional feasibility condition

We start by noting that the feasibility (with respect to obstacles) of each intermediate point on the trajectory does not guarantee the feasibility of the whole trajectory. Consider an example with only one parallelogram obstacle as shown in Figure 3(c). In Figure 3(c), two adjacent intermediate points, q_1 and q_2, are both feasible with respect to the given obstacle, o_1. The red curve in Figure 3(c) shows an infeasible curve connecting two points. Of course, the optimal trajectory could be feasible like the green curve. However, the line segment connecting the two points (the magenta curve) is infeasible. To avoid such undesirable cases, we need additional constraint between adjacent intermediate points. The curves in Figure 3(c) illustrates this additional constraint: Considering the corresponding binary variables of each point, q_3 is only feasible with respect to face $f = 1$ and the next intermediate point, q_4 is also feasible with respect to the same face. So, the line segment, connecting these two points is also feasible with respect to face $f = 1$. Moreover, both q_5 and its previous point, q_4, are feasible with respect to face $f = 2$. So the line segment joining them is also feasible. In contrast, consider the case of q_1 and q_2 in Figure 3(c). These two adjacent intermediate points do not share feasibility with respect to a common face – q_1 is feasible with respect to only face $f = 4$ and q_2 is feasible with respect to only face $f = 1$. So we cannot guarantee the feasibility of the line segment connecting these two points.

The above discussion suggests an additional constraint that two consecutive intermediate points should share a common hyperplane with respect to which they are feasible, and this should hold true for each obstacle. In other words, the binary variables corresponding to the adjacent intermediate points should either be the same or differ by only one component, and this condition should be satisfied with respect to all obstacles. This constraint then guarantees the feasibility of a straight line segment connecting the two intermediate points. We write $b_{(o,k)}$ to describe the vector of binary variables for the k^{th} intermediate point formed by stacking together the binary variables for the different faces of the o^{th} obstacle (thus, it is a n_i-sized sub-vector of $\widetilde{b}$). Thus the constraint involving the k^{th} and $k+1^{th}$ intermediate points with respect to o^{th} obstacle can be describe as

$$\begin{aligned}\left\|b_{(o,k)} - b_{(o,k+1)}\right\|_2^2 = \\ b_{(o,k)} \cdot b_{(o,k)} + b_{(o,k+1)} \cdot b_{(o,k+1)} - 2b_{(o,k)} \cdot b_{(o,k+1)} = \\ \sum b_{(o,k)} + \sum b_{(o,k+1)} - 2b_{(o,k)} \cdot b_{(o,k+1)} \quad \leq 1\end{aligned} \tag{7}$$

where, $\sum b$ denotes the sum of the elements of a binary vector, and the last equality holds since $b \cdot b = \sum b$ for a vector of binary variables, $b \in \{0,1\}^n$. This additional constraint on the gradual change of the binary variables along the trajectory plays an important role in formulation of a new h-signature based on binary variables, as described in the next section. However, this constraint is quadratic in the binary variables, and we will discuss how we can reduce this constraint to a linear one in Section III-C.

B. Define h-signature

To find an optimal trajectory contained in a specific homology class, the h-signature of [18] can be used. However, this function is based on Cauchy integral theorem and is both nonlinear with respect to the given trajectory or coefficient of basis functions. So we define a new h-signature keeping in mind the definition of homologous trajectories of [12]. Instead of calculating winding number through Cauchy integral, we can get it by counting how many times the closed curve intersects a reference ray (one dimensional half-hyperplane), originating on the obstacle, in either clockwise or counterclockwise directions. The Figure 4(a) shows an example. Since the closed loop formed by $\tau_1 \bigsqcup -\tau_2$, where $-\tau_2$ means reverting the direction of trajectory, crosses the red ray twice, first in clockwise and then in counterclockwise direction, the winding number of this closed loop becomes zero, and the area enclosed by this closed curve does not contain the obstacle o_1. Thus, in Figure 4(a), τ_1 and τ_2 are homologous. However, the closed loop formed by $\tau_1 \bigsqcup -\tau_3$ intersects the red ray once in clockwise. The winding number of $\tau_1 \bigsqcup -\tau_3$ is one, not zero, and two trajectories are not homologous. This definition is consistent with previous works, [18], [12]. So we define new h-signature of a trajectory with respect to an obstacle, o_i, as the number of intersections with reference ray of o_i in clockwise subtracted by the number of intersections with reference ray of o_i in counterclockwise.

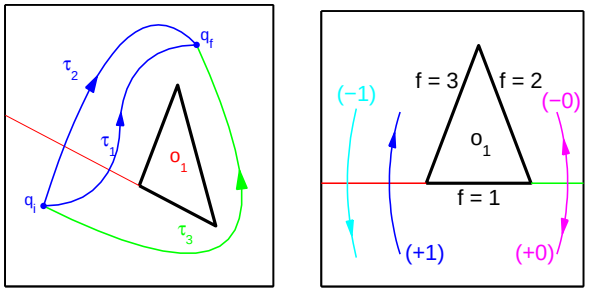
We can choose arbitrary reference ray of each obstacle but for convenience of calculation and notation, we choose the reference ray as the extension of face $f = 1$ in the direction of the last face $f = n_f$ as shown in Figure 4(b). Also, for the consistency of sign of winding number, the faces are numbered in counterclockwise direction like Figure 4(b). Then it is obvious that we need to accumulate the value of $b_{i,1,k+1} - b_{i,1,k}$ for $\forall k$ (where by $b_{i,j,k}$ we mean the binary variable for the k^{th} intermediate point corresponding to the j^{th} face of the i^{th} obstacle). However, to avoid counting the number of intersection with the the other ray obtained by extending the face $f = 1$ in the other direction (the green line in Figure 4(b)), we need to count the case when the two adjacent intermediate points are infeasible with respect to the second face $f = 2$, *i.e.* $b_{i,2,k+1} = b_{i,2,k} = 1$. So, the h-signature with respect to an obstacle, o_i, will be

$$\begin{aligned} h_i(\widetilde{b}) &= \sum_k \frac{b_{i,2,k+1} + b_{i,2,k}}{2} \left(b_{i,1,k+1} - b_{i,1,k}\right). \\ &= \sum_k b_{i,2,k} \left(b_{i,1,k+1} - b_{i,1,k}\right) \end{aligned} \tag{8}$$

The second equality of above equation holds because $b_{i,2,k+1} = b_{i,2,k}$ when $b_{i,1,k+1} \neq b_{i,1,k}$ due to the constraint we defined in (7). The h-signature with respect to all obstacles will be $H = [h_1, ..., h_{n_o}]^T$, where n_o is the number of obstacles. However, this new h-signature is also quadratic in binary variables. We will discuss how we can reduce this quadratic equation to a linear one in the next section.

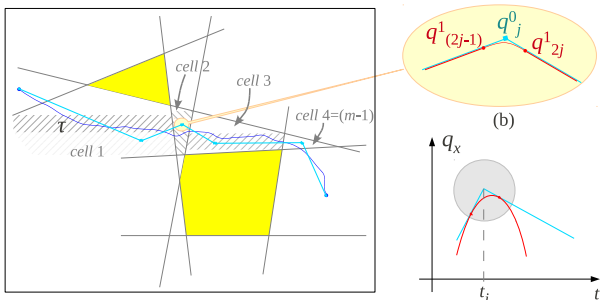
C. Substitution binary variables

As the new constraint in (7) and the h-signature in (8) are quadratic with respect to the binary variables, we introduce some *substitution binary variables* that represent the product of two binary variables. For example, consider the product



(a) (b)

Fig. 4. (a) An example of counting winding number of trajectories. (b) An example of calculating the h-signature with respect to a triangular obstacle.



(a)

Fig. 5. Starting from a piece-wise linear curve (cyan), we can progressively add points, to make the trajectory smoother by increasing the order of differentiability by one at each step.

of two binary variables, $b_i \cdot b_j$, for $b_i, b_j \in \{0,1\}$. Then, we substitute $b_i \cdot b_j$ with a new binary variable $d_{ij} \in \{0,1\}$, on which we impose the following three inequalities,

$$d_{ij} \leq b_i \quad , \quad d_{ij} \leq b_j \quad , \quad -2 + \delta + b_i + b_j \leq d_{ij} \tag{9}$$

where $0 < \delta < 1$ is a design parameter. The first two inequalities in (9) enforce $d_{ij} = 0$ when $b_i = 0$ or $b_j = 0$, respectively. And the last inequality enforces $d_{ij} = 1$ when $b_i = b_j = 1$, because $0 < \delta \leq d_{ij}$. So the above three constraints let us perform the substitution $d_{ij} = b_i \cdot b_j$. Let d be the vector of substitution variables with which we need to replace all the quadratic terms in (7) and (8). Then we can rewrite the feasibility conditions of substitution binary variables, (9), as

$$A_{f,d}\widetilde{b} + B_{f,d}d \leq b_f. \tag{10}$$

Then we can rewrite the quadratic constraint of (7) for the whole trajectory as

$$A_{o,k}\widetilde{b} + B_{o,k}d \leq b_{o,k} \tag{11}$$

for all o and k. And the h-signature calculation of (8) becomes the following linear equation

$$h_i = A_{i,h}d. \tag{12}$$

So, we can reduce all equations containing quadratic terms in the binary variables to linear ones using the substitution binary variables.

D. Finding Optimal Trajectory in a given Homology Class

Since our goal is to design optimal trajectory with homology constraint, we can impose the new constraints of (10), (11), and (12) to the optimal trajectory generation problem (6) to formulate a new MIQP as follows

$$\begin{aligned} \min_{c,\ \widetilde{b},\ d} \quad & c^T H c \\ \text{s.t.} \quad & A_f c + D_f \widetilde{b} \leq g_f \quad , \quad A_b \widetilde{b} \leq g_b \quad , \\ & A_d \widetilde{b} + B_d d \leq b_f \quad , \quad A_o \widetilde{b} + B_o d \leq b_o \quad , \\ & A_{eq} c = 0 \quad , \quad A_h d = H_d \end{aligned} \tag{13}$$

where $\widetilde{b}$ and d are vectors of binary variables as described earlier. The third inequality is the condition of substitution variables (10), the forth inequality is for additional feasibility constraint for continuous change of binary variables (11), and the last equality is for the homology constraint with respect to all obstacles (12). As the resulting problem is MIQP, we can get an anytime solution to this problem through numerical solvers like CPLEX [24]. However, we need enough number of segments of trajectories (N_s) and basis function (N_p) to be able to obtain a feasible trajectory in the given homology class.

Proposition 1 (Completeness Guarantee): Suppose there exists an arbitrary trajectory τ (dark blue curve in Figure 5(a)), not touching any of the obstacles, in the homology class represented by the h-signature of H_d, that crosses the cell boundaries (*i.e.* the hyperplanes) m or less number of times (for avoiding ambiguity we assume τ is generic and that it does not pass through the intersection of 2 or more hyperplanes). With the choice of basis functions $e_k(t) = t^k$ in (4), and with $N_p > r$, it is then sufficient to choose $N_s = 2^r(m-1)+1$ in order to guarantee existence of a solution for the problem in (13) (*i.e.* all the conditions being satisfied, and with finite cost).

Sketch of Proof. Consider the $m-1$ consecutive cells that τ passes through. We choose $m-1$ points, $q_1^0, q_2^0, \cdots, q_{m-1}^0$, respectively in the interior of each of these cells. Now, two such consecutive cells together form a convex region (bounded by the hyperplanes the cells are individually bounded by, except for the one hyperplane that separates them). Thus, the piece-wise linear curve formed by joining these consecutive points (call this q^0) give a trajectory consisting of m segments (cyan curve in Figure 5(a)), connecting the initial and final points, not intersecting any of the obstacles, and is continuous (*i.e.* 0^{th} order differentiable). The affine segments are permitted by the choice of the basis functions (the parametrization may be chosen arbitrarily), thus giving values of coefficients, $c_{j,k}$, in (4) that describe this trajectory. Those, along with the binary vectors corresponding to each of these points, satisfy all the conditions in (13), except for the differentiability condition $A_{eq}c = 0$.

The main idea behind the proof of this proposition is that we can now replace each of the points q_j^0 by two points lying arbitrarily close to it, and thus "smoothen" the curve (Figure 5(b)). This smoothening is possible to achieve with just an unit increase in the degree of the basis functions (which is evident by looking at the individual components $q_x^0(t)$ and $q_y^0(t)$, as illustrated in Figure 5(c)) – in this case, going from linear to quadratic (it is always possible to find a parabola that has two given lines with bounded slope as tangents, and then scale it down such that the contact points with the tangents lie within a small ball around the point of

intersection of the lines).

Thus, now we have a new trajectory (call this q^1), that is smooth everywhere, but not twice differentiable (red trajectory in Figure 5(b)). However, we can continue the same process of smoothening the derivatives of $q(t)$ by adding points in a small neighborhood of the original t_j's, doubling the number of intermediate points at every step. The choice of this neighborhood can be arbitrarily small to ensure that the added points remain in the interior of the same cell. Continuing this until we have r^{th} order differentiability requires $2^r(m-1)$ intermediate points. In this way, we can construct a trajectory that satisfies all the conditions of (13). ■

E. Computational Complexity

The resulting optimal trajectory generation problem is a MIQP, which can be solved by an anytime solver like CPLEX. Thus, if there exists a feasible solution, it will be found by CPLEX. Moreover, with additional time available for computation, a lower cost solution can be found. However, the computation time will increase with the complexity of the given MIQP. So, in this section, we will discuss the computational complexity of the trajectory generation problem (13). The number of continuous variable in the problem is

$$n_c = 2(N_p + 1)N_s \tag{14}$$

where there are N_s segments of trajectories of $N_p + 1$ basis function for each x and y. However, some equality constraints to satisfy initial and final configuration and the continuity between segments of trajectories will reduce the actual number of continuous variables by searching the null space of A_{eq} of (13). Again, the number of binary variables to describe feasibility with respect to each face of obstacle is

$$n_b = N_c \cdot N_f \tag{15}$$

where N_c is the number of intermediate points on the whole trajectory and $N_f = \sum_{i=1}^{n_o} n_i$ is the total number of faces of all obstacles. Then the number of substitution binary variables is

$$n_d = (N_c - 1)N_f + 2(N_c - 1) \tag{16}$$

where each term is related to the quadratic terms in (7) and (8), respectively. So the total number of binary variables will be $n_b + n_d$.

The number of constraint is also an important factor in computational complexity. The number of equality constraint will be

$$n_{eq} = 2(k_r + 1)(N_s - 1) + 2 \times 4 + n_o \tag{17}$$

where the first term represent the continuity between segments of trajectories. The second term represents the equality constraint of initial and final configuration; position and velocity of x and y. The last term is related to the h-signature constraint, which is the same as the number of obstacles. However, this equality constraint will disappear because we search in the null space of this equality constraint while reducing the number of continuous variables in the same manner. Most of the constraints are inequality constraints and we have

$$\begin{aligned} n_{ineq} =& 2N_c \cdot N_f + N_c \cdot n_o + N_c \cdot n_o + 3n_d \\ =& 5N_c \cdot N_f + 2N_c \cdot n_o + 6N_c - 3N_f - 6 \end{aligned} \tag{18}$$

where the first term represents on which side of each face the intermediate point is located – the first equation of (2) and equation of (3). The second term represents the second equation of (2) and the third term represents (7). The last term presents the condition of substitution binary variables (9). So the number of inequality constraint is bilinear with respect to the number of intermediate points and faces of obstacles.

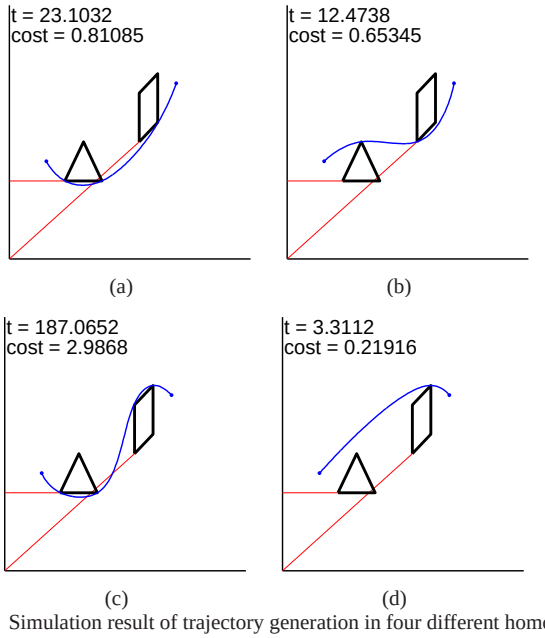


Fig. 6. Simulation result of trajectory generation in four different homology classes with the same initial configuration (left bottom point) and final configurations(right upper point). The first obstacle is parallelogram and the second obstacle is triangle. The actual computation time(sec) and optimal costs are specified on the upper left corners of plots. (a) $H_d = [-1, -1]^T$. (b) $H_d = [-1, 0]^T$. (c) $H_d = [0, -1]^T$. (d) $H_d = [0, 0]^T$.

IV. Simulation Results

To illustrate how the suggested algorithm works, we performed some simulations to generate optimal trajectories of a point robot, $\delta_r = 0$, in various homology classes of a given environment. For all simulations, we use polynomial basis functions, $e_k(t) = t^k$ and minimize the integration of the norm of acceleration of trajectories, *i.e.* we choose $r = 2$ in (5). In the first simulation, we find optimal trajectories with two obstacles under homology constraint. As mentioned in the introduction, the planned trajectory could be for a differentially-flat dynamical robot system such as a kinematic car [22], or a tricycle robot [23].

Figure 6 shows optimal trajectories with the same initial and final configurations but with different desired h-signatures, and consequently different homology classes. For each homology class, the CPLEX solver finds the optimal trajectory.Comparing the computation time and cost of trajectories of each homology class, it can be observed that the optimal trajectory in the homology class corresponding to

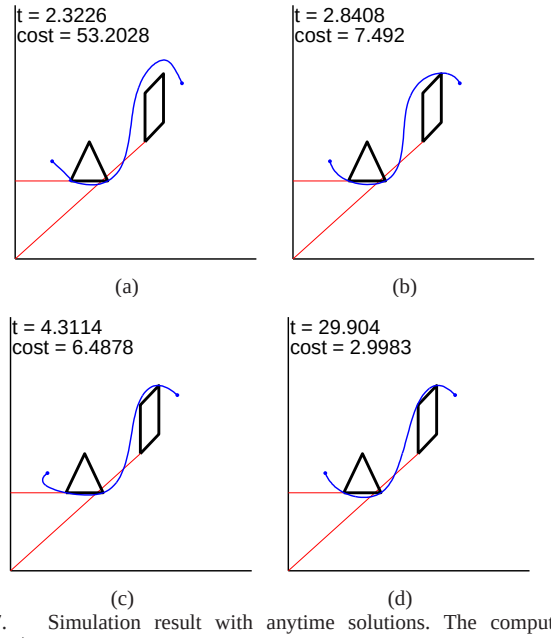


Fig. 7. Simulation result with anytime solutions. The computation time(sec) and optimal costs are specified on the upper left corners of each plot.

lower cost takes less time to compute. This is an expected phenomenon since in a branch-and-bound algorithm the tree of fixed binary variables tends to expand to minimize the cost. In searching for the optimal trajectory in a particular homology class (*e.g.* the one in Figure 6(c)), the algorithm expands the nodes in the tree in such a way that feasible solutions corresponding to other homology classes, but with lower costs (*e.g.* the class in Figure 6(d)), are also obtained in the process.

To show the anytime performance of the suggested algorithm, we performed some simulations with the same environment as earlier, and with the homology constraint of Figure 6(c). The CPLEX solver was terminated at different times to compare the resulting trajectories. As shown in Figure 7, the cost decreases as we allow more computation time. And as illustrated in the previous example in Figure 6, we will get the global optimal trajectory with enough computation time.

Next we present a series of four examples with increasing number of obstacles, and subsequently increasing complexity (see Figure 8). For all examples, we choose six segments of trajectories with nine basis functions, such that the number of continuous variables for optimization, as given by (14), is $n_c = 108$. The optimizer is given a maximum time of one hour to search for a feasible trajectory. The resulting found trajectory will respect the homology constraint and may be either suboptimal or optimal. Figure 8(a) illustrates results for the two obstacle case, showing trajectories in all four different homology classes. Figure 8(e) shows how the cost of each generated trajectory changes as we keep searching – the corresponding trajectories have the same color as in Figure 8(a). It is obvious that the trajectory corresponding to the red curves in Figures 8(a),8(e) is the global optimal one without topological constraints. So it terminates searching solution before the time limit. Note that we could not find global optimal trajectories for all the homology classes within the time limit. Figure 8(e) however shows that a feasible solution was found relatively quickly. With additional computation time, we expect the optimizer to either find the global optimal trajectories in each homology class or guarantee that the current solution is the global optimum.

Figure 8(b) shows the result of simulation with three obstacles. We find suboptimal trajectories in all the eight homology classes. As shown in Figure 8(f), an initial feasible trajectory was found relatively quickly for all but one homology class. For the homology class corresponding to the black curve, the optimizer took over 1000 sec to find a feasible trajectory.

For the four obstacle case shown in Figure 8(c), we found trajectories in nine homology classes, and could not find trajectories in other seven homology classes within the time limit. The missing seven trajectories should pass obstacle 4 on left like one of the green plots in Figure 8(c). Similarly, for the five obstacle case shown in Figure 8(d), we found trajectories in only five homology classes among $2^5 = 32$ possible homology classes. As we found all eight trajectories passing between obstacle 4 and 5, like the trajectories in Figure 8(b), there are feasible trajectories in these homology classes which can be represented with our parametrization and can be found.

As illustrated by these simulations, although the optimizer quickly found initial feasible suboptimal trajectories in various homology classes, it could either not find corresponding optimal solutions for all classes or not find the trajectories in all the possible homology classes within the provided time. A few reasons for this are (a) insufficient computation time for the optimizer, (b) insufficient continuous-time variables for parametrization of longer and winding trajectories in certain homology classes, and (c) fundamental limitation of using a general purpose solver such as CPLEX for this particular scenario. To address the issue of computation time, the algorithm can easily be run longer, but more importantly, the computation time can be improved significantly by providing an initial guess for the optimization. Further, increasing the number of continuous-time variables will also help since Proposition 1 guarantees that there exists a feasible trajectory with a sufficiently large number of segments of trajectories.

V. Conclusion

In this paper, we have presented a method to find a smooth optimal trajectory subject to geometric and kinematic constraints, and restricted to a specific homology class. We used a Mixed-Integer Quadratic Programming(MIQP) formulation to achieve this. The homology constraint is incorporated by calculating the h-signature of the trajectory from its binary variables, which constitute a coarse representation of trajectory. The calculation of the h-signature is quadratic in the binary variables but is reduced to a linear equation by introducing substitution binary variables. The resulting problem then becomes a MIQP, which can be

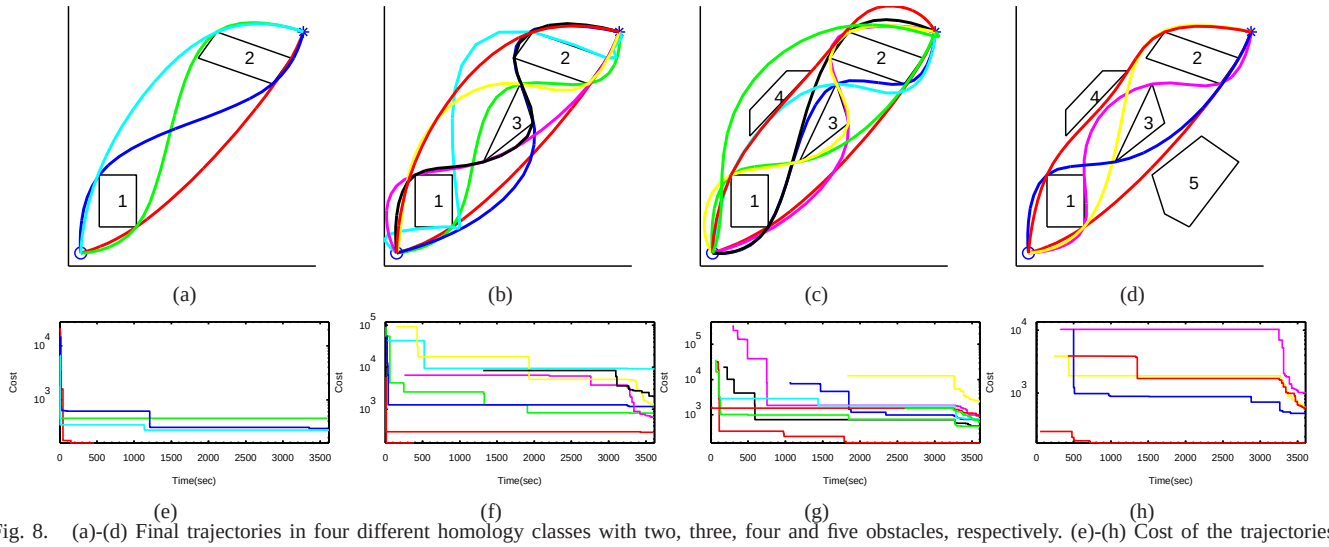


Fig. 8. (a)-(d) Final trajectories in four different homology classes with two, three, four and five obstacles, respectively. (e)-(h) Cost of the trajectories along with computation time with two, three, four and five obstacles, respectively. The plots shows the change in cost with time plotted in log scale.

solved using an anytime numerical solver like CPLEX. We also illustrated the anytime performance of the suggested method with various simulations. However, the substitution binary variables increase the problem size of MIQP, and thus requires more computation time. Clearly reducing the computational complexity of the method is an important direction of future research. In addition, we plan to extend the basic method to three-dimensional settings using the framework used in [12].

ACKNOWLEDGEMENTS

We gratefully acknowledge the support of ONR Grants N00014-07-1-0829 and N00014-09-1-1031 and ARL Grant W911NF-10-2-0016.

REFERENCES

[1] D. J. Balkcom, "Geometric construction of time optimal trajectories for differential drive robots," in *Fourth Workshop on Algorithmic Foundations of Robotics*, 2000, pp. 1–13.

[2] J. E. Bobrow, B. Martin, G. Sohl, E. C. Wang, F. C. Park, and J. Kim, "Optimal robot motions for physical criteria," *J. of Robotic Systems*, vol. 18, p. 2001, 2001.

[3] B. Donald, P. Xavier, J. Canny, and J. Reif, "Kinodynamic motion planning," *J. of ACM*, vol. 40, no. 5, pp. 1048–1066, November 1993.

[4] S. M. Lavalle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, B. R. Donald, K. M. Lynch, and D. Rus, Eds. A K Peters, 2001, pp. 293–308.

[5] R. Tedrake, I. R. Manchester, M. M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums of square verification," *The Int. J. of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, July 2010.

[6] Y. Yang and O. Brock, "Elastic roadmaps–motion generation for autonomous mobile manipulation," *Auton. Robots*, vol. 28, pp. 113–130, January 2010. [Online]. Available: http://dx.doi.org/10.1007/s10514-009-9151-x

[7] M. Earl and R. D'Andrea, "Iterative milp methods for vehicle-control problems," *Robotics, IEEE Transactions on*, vol. 21, no. 6, pp. 1158–1167, 2005.

[8] M. Vitus, V. Pradeep, G. Hoffmann, S. Waslander, and C. Tomlin, "Tunnel-milp: Path planning with sequential convex polytopes," in *AIAA Guidance, Navigation, and Control Conference*, 2008.

[9] Y. Guo and L. E. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *In Proceedings of IEEE International Conference on Robotics and Automation*, 2002, pp. 2612–2619.

[10] M. B. Dias, R. M. Zlot, N. Kalra, and A. T. Stentz, "Market-based multirobot coordination: A survey and analysis," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-05-13, April 2005.

[11] H. Zhang, V. Kumar, and J. Ostrowski, "Motion planning under uncertainty," in *IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May 16-21 1998, this paper was nominated for the best paper award at the 1998 IEEE International Conference on Robotics and Automation.

[12] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Autonomous Robots*, vol. 33, no. 3, pp. 273–290, October 2012, doi: 10.1007/s10514-012-9304-1.

[13] J. Hershberger and J. Snoeyink, "Computing minimum length paths of a given homotopy class," *Comp. Geom. Theory and Applications*, vol. 4, pp. 331–342, 1991.

[14] D. Grigoriev and A. Slissenko, "Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane," in *Int. Symposium on Symbolic and Algebraic Computation*, 1998, pp. 17–24.

[15] E. Schmitzberger, J. L. Bouchet, M. Dufaut, D. Wolf, and R. Husson, "Capture of homotopy classes with probabilistic road map," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, October 2002, pp. 2317–2322.

[16] D. Demyen and M. Buro, "Efficient triangulation-based pathfinding," in *National Conf. on Artificial Intelligence*, 2006, pp. 942–947.

[17] B. Tovar, F. Cohen, and S. LaValle, "Sensor beams, obstacles, and possible paths," in *Algorithmic Foundation of Robotics VIII*, ser. Springer Tracts in Advanced Robotics, G. Chirikjian, H. Choset, M. Morales, and T. Murphey, Eds. Springer Berlin / Heidelberg, 2009, vol. 57, pp. 317–332.

[18] S. Bhattacharya, V. Kumar, and M. Likhachev, "Search-based path planning with homotopy class constraints," in *AAAI Conf. on Artificial Intelligence*, July 2010.

[19] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *European Control Conference*. Citeseer, 2001, pp. 2603–2608.

[20] A. Richards and J. P. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *American Control Conf.*, vol. 3, May 2002, pp. 1936–1941.

[21] R. M. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems: A catalog of prototype systems," in *Proceedings of the 1995 ASME International Congress and Exposition*, 1995.

[22] R. Murray and S. S. Sastry, "Nonholonomic motion planning: Steering using sinusoids," *IEEE Transactions on Automatic Control*, vol. 38, pp. 700–716, 1993.

[23] M. Benayad, G. Campion, V. Wertz, and M. Achhab, "Steering a mobile robot: Selection of a velocity profile satisfying dynamical constraints," *Asian Journal of Control*, vol. 2, no. 4, pp. 219–229, 2000.

[24] *IBM ILOG CPLEX V12.1: User's Manual for CPLEX*, International Business Machines Corporation, 2009.