Trajectory Planning for Systems with Homotopy Class Constraints

Soonkyum Kim, Koushil Sreenath, Subhrajit Bhattacharya and Vijay Kumar

Abstract There are various applications where homotopy 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 homotopy class, which is specified by a given representative trajectory. The optimality is achieved by formulating the trajectory generation problem as a Mixed-Integer Quadratic Program (MIQP). We partition the configuration space into nonoverlapping cells and model each cell in the partition with integer variables and inequality constraints. We associate with any sequence of integer variables a *word*, so that each trajectory can be mapped to a word. We then construct a set of all words that are homotopically equivalent to a given word. For each word, we fix the integer variables of the MIQP to find the optimal time distribution in each cell, by solving a QP for each iteration, to obtain the locally optimal trajectory in the specified homotopy class. We illustrate an example of minimum acceleration trajectory generation on a plane with different homotopy class constraints.

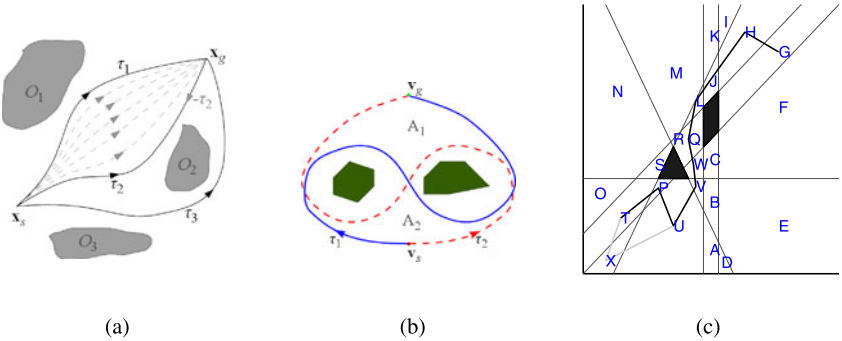
Key words: Trajectory planning, homotopy constraint, optimal trajectory

1 Introduction

Early attempts at classifying homotopy classes in two dimensions include geometric methods [5, 6], homotopy preserving probabilistic road-map constructions [10], and triangulation-based path planning [3]. Two trajectories are said to be homotopic if one can be continuously deformed to another without any intersection with obstacles. 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 the homotopy class. Thus, trajectory generation with a

Soonkyum Kim · Koushil Sreenath · Subhrajit Bhattacharya · Vijay Kumar
Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, USA,
e-mail: {soonkyum,koushils,subhrabh,kumar}@seas.upenn.edu

J. Lenarčič, M. Husty (eds.), *Latest Advances in Robot Kinematics*,
DOI 10.1007/978-94-007-4620-6_11,



(a) (b) (c)

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. (c) An example of a trajectory corresponding to the word TPUVWQLJHG.

homotopy class constraint 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 way based on the specific task.

In this paper, we use mixed-integer quadratic programming [8] to partition the configuration space into non overlapping cells and represent each cell by a label or a *letter*. We use the notion of *words*, constructed out of the letters, to coarsely represent trajectories and relate them with their homotopy classes. First, all words corresponding to a particular homotopy class are constructed. Then, for each word, a quadratic program (QP) is solved to find a trajectory that spends equal amounts of time in each cell specified by the letters of the word. Finally, this trajectory is iteratively refined to obtain the locally optimal trajectory in the specified homotopy class.

2 Preliminaries

Our objective in this paper is to design an 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 homotopy class. Although several of these subproblems have been solved separately, (see [1, 2, 4, 7, 11, 12]), there is no literature, to our knowledge, that addresses the combined problem described above.

We will start by assuming that the required homotopy class is specified by an initial representative trajectory in the homotopy class. Specifically, we will derive a locally optimal trajectory that is *homotopic* to the representative trajectory while also satisfying the kinematic and geometric constraints. We represent a planar trajectory, $q(t)$, by the points $[q_x(t), q_y(t)]$ parametrized by t.

2.1 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, $q_1 \sqcup -q_2$ (*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 obstacles. It is well known that homology is a "coarser" representation of homotopy [2], 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 formulation for computing the homology class as *h-signatures* using Cauchy integral theorems from complex analysis is carried out by Bhattacharya et al. [1, 2]. The h-signature of a trajectory, q, with respect to obstacle o_j is defined as

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

where $z(t) = q_x(t) + i q_y(t)$ 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 about all obstacles is given by $H = [H_1, \ldots, H_{n_o}]^T$ with n_o obstacles.

2.2 Optimal Trajectory Generation

We consider trajectory planning in a compact subset $Q \subset \mathbb{R}^2$ on 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 an arbitrarily-shaped obstacle as a union of convex obstacles.) 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 can be used to represent either side of the hyperplane as in [8]. 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 $f \in [1, \ldots, n_i]$ satisfying $n_{i,f} \cdot q \leq s_{i,f}$. Where $n_{i,f}$ is a normal vector to the fth face of obstacle o_i pointing inward and $s_{i,f} = n_{i,f} \cdot p$, for any point p on the fth face. For the obstacle o_i, all points $q \in Q$ outside the obstacle must satisfy [9]:

$$n_{i,f} \cdot q \leq s_{i,f} + M b_{i,f} \quad \text{for} \quad b_{i,f} \in \{0,1\}, \quad f = 1, \ldots, n_i \tag{2}$$
$$\sum_{f=1}^{n_i} b_{i,f} \leq n_i - 1,$$

where M is a large positive number. The second inequality in (2) implies that the point should be feasible about at least one face; at least one f such that $b_{i,f} = 0$. This formulation breaks up Q into overlapping regions.

The problem of finding a trajectory, $q(t)$, that avoids obstacles, respects kinematic constraints and is restricted to a specified homotopy class, is formulated as follows. The trajectory is obtained by splicing N_s sub-trajectories, each parametrized by linear combination of $N+1$ basis functions,

$$q(t) = \sum_{k=0}^{N} c_{j,k}\, e_k(t) \quad \text{for} \quad t_j \leq t < t_{j+1}, \tag{3}$$

for $j \in [0, \ldots, N_s - 1]$, $0 = t_0 \leq t_1 \leq \ldots \leq t_{N_s} = t_f$. Where $e_k(t)$ is a suitably chosen basis function and $c_{j,k}$ are coefficients. Throughout this paper, we use $e_k(t) = (t - t_j)^k$. The trajectory is restricted to be k_r-times continuously differentiable at each of the intermediate points, $q(t_j)$, for $j \in [0, \ldots, N_s - 1]$. Further, obstacle avoidance is achieved by enforcing (2) at some equally distributed intermediate points on each subtrajectories. The cost function is the integration of norm of rth-derivative of trajectory:

$$J(c) = \int_{t_0}^{t_f} \left\| \frac{d^r q(t)}{dt^r} \right\|^2 dt \;=\; c^T H c. \tag{4}$$

where $c = [c_0^T, \ldots\ c_{N_s-1}^T]^T$. The optimal trajectory generation problem can then be simplified as the following mixed-integer quadratic program,

$$\begin{aligned} \min_{c,\, b} \quad & c^T H c \\ \text{s.t.} \quad & A_f c + D_f b \leq g_f, \quad A_b b \leq g_b, \quad A_{eq} c = 0 \end{aligned} \tag{5}$$

where b is the vector of binary variables. 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 rth order differentiability at the intermediate points as well as the boundary condition of initial configuration, $q(0) = q_0$ and final configuration $q(t_f) = q_f$.

To find an optimal trajectory in a specific homotopy or homology class, we now incorporate topological constraints. If we add a constraint on h-signature, which we defined earlier, such that the h-signature of the trajectory, $H(q)$, should be some desired H_d, the quadratic program (5) becomes a nonlinear problem. Furthermore, the gradient of the new constraint, $H = H_d$ will be zero almost everywhere, because the value of h-signature does not change within a particular homology class (*i.e.* the range of the h-signature is a set of discrete values). So, the resulting problem

is NP and it is numerically hard to find a solution based on gradients of cost and constraints. So, we need another way to enforce topological constraints, and this is carried out as follows.

Although (2) is a sufficient condition for feasibility, we introduce an additional inequality so as to obtain a partition of Q.

$$-n_{i,f} \cdot q \leq -s_{i,f} + M(1 - b_{i,f}) \quad \text{for} \quad b_{i,f} \in \{0,1\}, \quad f = 1,\ldots,n_i. \tag{6}$$

The first inequality of (2) only guarantees that the point q is on one side of face when $b_{i,f} = 0$. With the additional constraint (6), the binary variable, $b_{i,f}$, uniquely determines in which half space a point q is on. So we can divide the work space with hyperplanes of obstacles by value of binary variables.

As a result, a set of connected *cells* is built, whose union is the feasible space, Q, and the intersection is only the extended lines of faces of the obstacles (see Figure 1(c)). Each cell can be identified by a unique *letter*, representing the vector of binary variables with one binary variable for each face of each obstacle. Every point in a particular cell will have the same letter representation. It must be noted that not all binary vectors define valid cells, and hence letters. The collection of all possible valid letters is defined as an *alphabet*.

Determining homotopy class of a trajectory is non-trivial. However, we use location information of intermediate intermediate point, each represented by a letter in the alphabet. Assembling the sequence of letters corresponding to each intermediate point of the trajectory and removing trivial repetitions will results in a *word*, which is a coarse representation of the trajectory. For example, the path shown in Figure 1(c) can be represented by the word $TPUVWQLJHG$. This can then be used to restrict trajectories to a homotopy class as will be seen in Section 3.

3 Algorithm Description

We have broken the problem of optimal trajectory generation into two parts. First we find a *word* that is a coarse representation of the trajectory and use this to restrict the homotopy class of the trajectory, and next find an optimal trajectory with this restriction. The following sections present the algorithm in more detail.

3.1 Finding Words in the Same Homotopy Class

To find an optimal trajectory satisfying a given homotopy class constraint, we first construct W_h, the set of words of the same homotopy class with the required one. We construct W_h by starting with the word for the given initial trajectory; $W_h = \{w_0\}$. Then we choose a word $w_c \in W_h$ and expand the chosen word as follows. For example let $w_c = TPUVWQLJHG$ as in Figure 1(c). We choose two letters, say T and U. If there is an alternative path, like TXU(the gray plot in Figure 1(c)), for the path TPU, we construct the closed loop by reversing the new path, and obtain

$TPUXT$ after removing duplicating letters. If the length of the closed loop is less than six, no obstacle lies in the closed loop (since we need to visit at least six cells to encircle a triangle). So we replace the path between the two chosen letters with the new path, and an expanded word representing the same homotopy class is achieved, $w_1 = TXUVWQLJHG$. The new word is added into W_h. We repeat this expansion until there are no more new words.

3.2 Finding the Optimal Trajectory

For a given word, $w_c \in W_h$, we parameterize the trajectory with N_s subtrajectories, where N_s is same as the length of w_c. Each subtrajectory is restricted to be in a particular cell specified by the corresponding letter in the word. Thus, all the binary variables, b_c, of the trajectory generation problem of (5) are fixed by the given word w_c, to reduce the optimization problem to

$$\begin{aligned} \min_{c} \quad & c^T H c \\ \text{s.t.} \quad & A_f c \leq \tilde{g}_f, \quad A_{eq} c = 0, \end{aligned} \tag{7}$$

which is obtained by substituting b_c in (5) and $\tilde{g}_f = g_f - D_f b_c$. As the resulting problem (7) is a quadratic program, we can find the global optimal trajectory for all words in W_h, which are in the given homotopy class.

However, it is not trivial to find the spending time in each cell to minimize the cost of the whole trajectory. To refine the trajectory further, we can adjust the time spent in each cell. With the final time, t_f, fixed, we can find an optimal time distribution by solving

$$\begin{aligned} \min_{t_j} \quad \min_{c} \quad & c^T H c \quad \left(\text{s.t.} \quad A_f c \leq \tilde{g}_f, \ A_{eq} c = 0\right) \\ \text{s.t.} \quad & t_j \leq t_{j+1} \quad \text{for} \quad j = [0, \ldots, N_s - 1], \quad t_0 = 0, \quad t_{N_s} = t_f. \end{aligned} \tag{8}$$

As this problem is a nonlinear program, we cannot guarantee the global minimum. However, the trajectory is iteratively refined by starting with $\Delta t_j = t_{j+1} - t_j = \frac{t_f}{n_w}$ for $j \in [0, \ldots, N_s - 1]$ and solving (8) by an interior-point method. Although we can find an initial solution without iteration, a better trajectory can be obtained by iterating the time distribution. Moreover, since the optimization cost reduces with more iterations, this method can be considered as an anytime algorithm that produces better solutions with more time.

4 Simulation Results

To illustrate how the suggested algorithm works, we performed some simulations to generate optimal trajectories with various homotopy classes. In this simulation, we

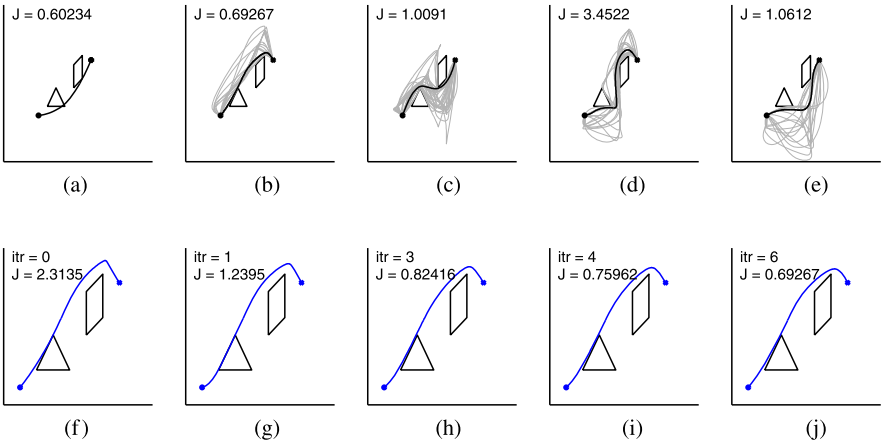


Fig. 2 (a) Optimal trajectory without homotopy constraints. (b)–(e) Trajectories with four different homotopy class constraints. The thick black curve is the optimal trajectory in each homotopy class and thin gray curves are the suboptimal trajectories for each word. The cost (J) for each case is specified on the upper left corners of plots. (f)–(j) Effect of varying the time distribution in each cell through iterations of the optimization (8). The number of iterations (itr) and cost are also specified on the upper left corner of each plot. Note that the cost converges to the local optimal cost of the case (b) in 6 iterations.

fix the final time $t_f = 10$ and find optimal trajectories for four homotopy classes. To reduce the computation time, we limit the maximum length of word to twelve.

The plot of Figure 2(a) shows the result of solving (5) without homotopy class constraints, resulting in an optimal cost of 0.60234. The plots of Figure 2(b)–2(e) show the result of solving (8) with four different homotopy class constraints, resulting in optimal costs that are greater than the global optimal one. When we search for trajectories with the same homotopy as the optimal trajectory achieved without homotopy class constraints (Figure 2(a)), the obtained optimal trajectory (Figure 2(e)) is a local optimal one with a larger cost. This disparity occurs due to restricting the trajectory to pass through certain cells and the fact that it is hard to find global optimal time distribution in each cell. The most optimal trajectory with homotopy class constraints lies in a different homotopy class from the global optimal one (Figure 2(b)). However, this is due to the symmetric arrangement of initial/final location of the trajectory and arrangement of obstacles.

With a fixed time distribution for each cell, the optimization reduces to a quadratic program for each word, which can be solved efficiently. To see the effect of optimizing the time distribution, we begin with a trajectory in the particular homotopy class of Figure 2(b) with equal time distribution over all the cells and iteratively optimize time distribution. The plots of Figure 2(f)–2(j) illustrate the changes in the trajectory and the corresponding cost with each iteration. Although this nested optimization is computationally expensive, with each iteration we get closer to the local optimal solution, resulting in an algorithm with *anytime* properties.

5 Conclusion

In this paper, we have presented a method to find an optimal trajectory subject to kinematic constraints, obstacle avoidance, and restricted to a specific homotopy class. This has been achieved by suitably modifying a MIQP to partition the configuration space and by constructing a coarser representation of the trajectory in the form of a word to represent the homotopy class. The set of all words representing the same homotopy class is constructed, and a nested optimization is carried out to find a locally optimal trajectory restricted to a homotopy class. Although we have only illustrated examples in the plane, work in [2] suggests obvious extensions to 3-D.

References

1. Bhattacharya, S., Kumar, V., Likhachev, M.: Search-based path planning with homotopy class constraints. In: AAAI Conf. on Artificial Intelligence (2010)
2. Bhattacharya, S., Likhachev, M., Kumar, V.: Identification and representation of homotopy classes of trajectories for search-based path planning in 3D. In: Robotics: Science and Systems (2011)
3. Demyen, D., Buro, M.: Efficient triangulation-based pathfinding. In: National Conf. on Artificial Intelligence, pp. 942–947 (2006)
4. Donald, B., Xavier, P., Canny, J., Reif, J.: Kinodynamic motion planning. J. ACM **40**(5), 1048–1066 (1993)
5. Grigoriev, D., Slissenko, A.: 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, pp. 17–24 (1998)
6. Hershberger, J., Snoeyink, J.: Computing minimum length paths of a given homotopy class. Comput. Geom. Theory Appl. **4**, 331–342 (1991)
7. Lavalle, S.M., Kuffner, J.J.: Rapidly-exploring random trees: Progress and prospects. In: Donald B.R., Lynch K.M., Rus D. (eds.) Algorithmic and Computational Robotics: New Directions, pp. 293–308. A K Peters (2001)
8. Richards, A., How, J.P.: Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In: American Control Conf., vol. 3, pp. 1936–1941 (2002)
9. Richards, A., How, J.P., Schouwenaars, T., Feron, E.: Plume avoidance maneuver planning using mixed integer linear programming. In: AIAA Guidance Navigation and Control Conference (2001)
10. Schmitzberger, E., Bouchet, J.L., Dufaut, M., Wolf, D., Husson, R.: Capture of homotopy classes with probabilistic road map. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 2317–2322 (2002)
11. Tedrake, R., Manchester, I.R., Tobenkin, M.M., Roberts, J.W.: LQR-trees: Feedback motion planning via sums of square verification. Int. J. Robot. Res. **29**(8), 1038–1052 (2010)
12. Tovar, B., Cohen, F., LaValle, S.: Sensor beams, obstacles, and possible paths. In: Chirikjian G., Choset H., Morales M., Murphey T. (eds.) Algorithmic Foundation of Robotics VIII. Springer Tracts in Advanced Robotics, vol. 57, pp. 317–332. Springer, Berlin/Heidelberg (2009)