JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 24, No. 2, March–April 2001

Trajectory Planning of Differentially Flat Systems with Dynamics and Inequalities

Nadeem Faiz* and Sunil K. Agrawal†
University of Delaware, Newark, Delaware 19716
and
Richard M. Murray‡
California Institute of Technology, Pasadena, California 91125

Trajectory planning of dynamic systems, in near real time, is important for aerospace systems, especially uncrewed air vehicles and launched munitions. Trajectory plans that do not consider the governing dynamic equations, applicable path, and actuator constraints may be unrealizable during execution. A trajectory planning scheme is proposed for a class of dynamic systems, referred to as differentially flat systems. The planner is motivated from online computations and is aimed to satisfy the state equations, path and actuator constraints, and given initial and terminal constraints. The essence of the approach is demonstrated by two examples: 1) a hardware implementation on a spring–mass–damper system to demonstrate real-time capabilities during pursuit and 2) trajectory planning of a planar vertical takeoff and landing aircraft to illustrate the application to nonlinear problems.

I. Introduction

THE need to compute system trajectories in near real time stems from the possibility of a system having to react quickly to changes in external environment or the goal. For example, an uncrewed air vehicle assigned to surveillance of an enemy facility must quicky modify its trajectory in the presence of an enemy radar. Similarly, it may need to modify its trajectory to make a detailed investigation of a site of interest. If only the geometry of the vehicle and its environment are used to compute a smooth trajectory, it will usually be inconsistent with the governing dynamic equations. As a result, such trajectories may become unflyable during execution. This paper considers trajectory planning of dynamic systems to satisfy explicitly the dynamic equations and inequalities on states and inputs.

Even though the theory presented is applicable to more general structures of dynamic systems, we assume that the system has the following control affine nonlinear model

$$\dot{q} = f(q) + g(q)u \tag{1}$$

where $q \in R^n$ are the states, $u \in R^m$ are the controls, $f(q)$ is an n-dimensional smooth vector field, and $g(q)$ is a smooth $n \times m$ matrix function. It is well known that dynamic systems that do not have a control affine form may still be expressed in the Eq. (1) format by introducing new state and input variables, often related to the derivatives of original variables. We impose the following aditional path and actuator constraints on the motion of the system:

$$c(q, \dot{q}, u) \leq 0, \qquad \forall t \in [0, t_f] \tag{2}$$

where $c \in \Re^{n_c}$. The trajectory planning problem is to determine in near real time $q(t)$ and $u(t)$ to take the system from an initial state $q(0)$ to a goal, assuming such paths exist, while satisfying Eqs. (1) and (2).

Planning has been an important research area in robotics and astonautics. A large body of literature exists on path planning of robots.[1] In these studies, geometries of the robot and environment are used to find a set of feasible joint coordinates between the initial point and the goal. Once the via points are determined, a smooth joint trajectory is formed using conservative estimates of joint speeds and joint accelerations.[2] The input torque constraints are satisfied through conservative estimates of joint accelerations. In aerospace, optimal control theory is often used to minimize a cost functional and to satisfy the system dynamic equations and inequalities.[3,4] It is well known, however, that most approaches to optimal control for nonlinear systems are computation intensive and are usually not suited for real-time implementations.

Briefly, the steps adopted in this paper are as follows: 1) The structure of the system dynamic equations is exploited to map the inequality constraints to the higher-order space of output functions and their derivatives. 2) The constraints are inner approximated offline in this higher-order space by a polytope using results of semi-infinite optimization theory. 3) The feasible trajectories are characterized within a class of admissible functions. 4) A finite collocation grid in time is chosen where the inequalities are satisfied. 5) The convex set of coefficients associated with the basis functions is solved online. These five steps are essentially described in the following sections followed by examples and experiments.

Some salient features of this paper that distinguish it from other work in the trajectory planning area are the following: 1) The state equations are explicitly accounted for during trajectory planning. 2) The constraints are taken as general functions of states, derivatives of states, and inputs. 3) The computations are divided into offline and online, with the goal to have a hard time bound on the online computations. 4) The approach is demonstrated in simulation and in hardware with simple experiments. This approach for real-time planning of dynamic systems in the higher-order space of output functions using polytopic form of the constraints is completely novel.

II. Flatness and Constraints in This Space

Classes of dynamic systems, referred to as differentially flat systems, have the special property that Eq. (1) admits the following diffeomorphic representation over the state space[5]:

$$q = q\left(z, z^{(1)}, \ldots, z^{(p-1)}\right) \tag{3}$$

$$u = u\left(z, z^{(1)}, \ldots, z^{(p)}\right) \tag{4}$$

where $z \in R^m$ are suitably chosen output functions that are determined using results from linear and nonlinear systems theory. In Eqs. (3) and (4), $p-1$ and p represent the highest derivatives of z in the expression for q and u, respectively. The integer p is a property

Received 14 October 1999; revision received 5 June 2000; accepted for publication 31 July 2000.

*Graduate Student, Mechanical Systems Laboratory, Department of Mechanical Engineering.
†Associate Professor, Mechanical Systems Laboratory, Department of Mechanical Engineering.
‡Associate Professor, Department of Mechanical Engineering, Division of Engineering and Applied Science.

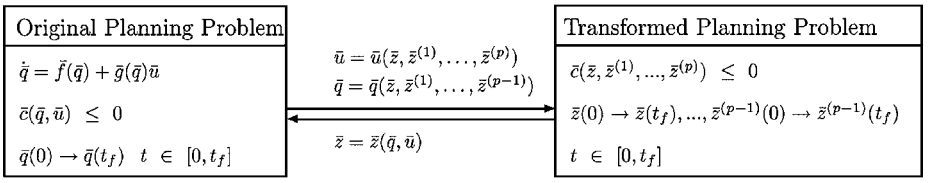

Fig. 1 Trajectory planning problem in the original and transformed space.

of the system. Among others, flat systems include controllable linear systems, as well as nonlinear systems linearizable by static and dynamic feedback. Hence, the theory presented in this paper can be applied to a variety of engineering systems that include mechanical, electrical, and chemical systems.

For differentially flat systems, a trajectory in the space of outputs $z(t)$ and its derivatives is consistent with the dynamics if it satisfies the boundary conditions. The states $q(t)$ and inputs $u(t)$ can be computed from $z(t)$ according to Eqs. (3) and (4). Differentially flat systems have been studied in the context of trajectory generation.[5,6] However, these studies were restricted to cases where the motion is not subject to inequality constraints. In recent publications, Agrawal and Veeraklaew[7] and Agrawal and Faiz[8] have investigated the problem of optimal trajectory generation for classes of differentially flat systems, but without auxiliary constraints. This paper extends the planning in the presence of auxiliary constraints.

Figure 1 shows the trajectory planning problem with auxiliary constraints in the original and transformed spaces of $z(t)$ and its derivatives. The state equations in this flat space are trivially satisfied, and the constraints have the following form:

$$c\left(z, z^{(1)}, \ldots, z^{(p)}\right) \leq 0 \tag{5}$$

The salient features of the constraints in the space of z, $z^{(1)}$, ..., $z^{(p)}$ are as follows: 1) The governing dynamic equations are already incorporated in the constraints. 2) A feasible set $\boldsymbol{S}$ in the space of z, $z^{(1)}, \ldots, z^{(p)}$, characterized by these constraints, is usually nonconvex. A feasible trajectory for the system must lie entirely within $\boldsymbol{S}$. However, because of the nonlinear nature of the constraints, it is impossible to estimate the required computation time to obtain such a feasible trajectory. Therefore, we explore alternative procedures to achieve this goal.

In this paper, we propose to inner approximate $\boldsymbol{S}$ by a polytope P that lies entirely within $\boldsymbol{S}$. We recall that polytopes are sets enclosed by linear inequalities. Hence, this approximation allows the problem to be solved later by linear programming as opposed to nonlinear programming. Corresponding to Eq. (5), P has the form

$$M_0 z + M_1 z^{(1)} + \cdots + M_p z^{(p)} + \boldsymbol{e} \leq 0 \tag{6}$$

where $M_0, M_1, \ldots, M_p$ are $(l \times m)$ matrices, $\boldsymbol{e}$ a constant $(l \times 1)$ vector, and l is the number of facets chosen to inner approximate the polytope. As evident, different polytopes with the same number of facets can be embedded within $\boldsymbol{S}$. For example, if $\boldsymbol{S}$ was a circle, one can potentially enclose a small or a large triangle, the three face polytopes. Similary, one can enclose in the circle a polygon of more than three sides such as a square, a pentagon, etc.

However, from a practical point of view, one would like to choose a P that encloses the maximum volume of $\boldsymbol{S}$. A challenge then is to approximate $\boldsymbol{S}$ by a polytope of maximum volume for a given number of facets for the polytope. This is posed as an offline optimization problem and solved using results of semi-infinite optimization theory to be described.

III. Inner Approximation of a Set by Polytopes

In this section, a feasible set is inner approximated by linear inequality constraints. By the use of the theory of semi-infinite optimization, the volume of a polytope with a given number of facets is maximized. We refer to this procedure as polytopic approximation and is implemented offline in MATLAB® using the nonlinear programming solver NPSOL®. The problem posed in the higher-order space $y = [z' \;\; z'^{(1)} \;\; \ldots \;\; z'^{(p)}]'$ can be stated as follows: Given nonlinear functions $c_i(y) \leq 0$, $i = 1, \ldots, n_c$, define a feasible set $\boldsymbol{S} = \{y \mid c_i(y) \leq 0, i = 1, \ldots, n_c\}$. Find an inner approximation of $\boldsymbol{S}$ by P of a fixed number of facets such that $P \subset \boldsymbol{S}$ and P attains the largest volume.

Global optimization problems seek to minimize $f(y)$ subject to $y \in D$, where D is the feasible domain. In outer approximation, D is approximated from the outside by a sequence of nested polytopes $D_1 \supset D_2 \supset \cdots \supset D_k \supset D$ such that $\min f(D) \leftarrow \min f(D_k)$ (Refs. 9–11). Dual approaches, called inner approximation, have also been suggested where a feasible set D is approximated from the inside by a sequence of expanding polytopes $D_1 \subset D_2 \subset \cdots \subset D_k \subset D$ such that $\min f(D_k) \to \min f(D)$ (Refs. 12 and 13). A limitation of these approaches is that the set D is assumed to be convex. In our work, the theoretical framework of Ref. 14, based on semi-infinite optimization theory, is used to develop a computational approach to embed a polytope, with a fixed number of facets, of largest volume within D. This approach works for nonconvex sets D.

A. Semi-Infinite Optimization Theory

For a polytope P to lie within $\boldsymbol{S}$, every point within the polytope must satisfy the constraints $c_i(y) \leq 0$, $i = 1, \ldots, n_c$. However, there are infinite points within a polytope, and it is computationally impossible to verify the constraints at all of these points. Hence, it is important to find points within P that are sufficient guaranteers of the inequalities for all other points in P. Semi-infinite optimization theory provides a framework to tackle such problems, that is, infinite constraints are replaced by an appropriate finite set of constraints. This section very briefly summarizes these results.

For a polytope P, we define $V = [V_1 \;\; V_2 \;\; \cdots \;\; V_q]^T$ as the set of vertices and Volume $P(V)$ as its volume. A point $y \in P$ can be written as a weighted sum of the vertices:

$$y = V_1\lambda_1 + V_2\lambda_2 + \cdots + V_q\lambda_q \tag{7}$$

where λ_i are scalar weighting coefficients satisfying the following properties:

$$\lambda_j \in [0, 1], \qquad \sum_{j=1}^{q} \lambda_j = 1$$

Further, we define $\bar{\lambda} = [\lambda_1 \;\; \lambda_2 \;\; \ldots \;\; \lambda_q]^T$ and consider $\bar{\lambda} \in B$ if the preceding two conditions are satisfied. The constraints are also written in a vector form: $c(V, \bar{\lambda}) = [c_1(V, \bar{\lambda}) \;\; c_2(V, \bar{\lambda}) \;\; \cdots \;\; c_{n_c}(V, \bar{\lambda})]^T$.

We define now two problems, respectively denoted as semi-infinite optimization problem (SNP) and finite-dimensional optimization problem (NP).

SNP:

$$\max\{\text{Volume } P(V) \mid V \in D\} \tag{8}$$

where $D = \{V \mid c(V, \bar{\lambda}) \leq 0, \bar{\lambda} \in B\}$.

NP:

$$\max\{\text{Volume } P(V) \mid V \in D^*\} \tag{9}$$

where $D^* = \{V \mid c(V, \bar{\lambda}) \leq 0, \bar{\lambda} \in B^*\}$. Here $B^* \subset B$ and consists of discrete $\{\bar{\lambda}_1^*, \bar{\lambda}_2^*, \ldots, \bar{\lambda}_r^*\}$ that correspond to points in P where the constraints $c_i(y^*)$ have maxima. The local reduction technique in semi-infinite optimization theory specifies conditions on the local equivalence of SNP and NP.[14] For a $\hat{V} \in D$, let $I(\hat{V}) = \{\bar{\lambda}_1, \bar{\lambda}_2, \ldots, \bar{\lambda}_r\}$ be a finite set where one of the constraints is active. For each $\bar{\lambda}_l \in I(\hat{V})$, if there exists a continuous mapping $\beta_l : U_{\hat{V}} \longrightarrow U_{\bar{\lambda}_l}$, where $U_{\hat{V}}$ and $U_{\bar{\lambda}_l}$ are, respectively, neighborhoods of $\hat{V}$ and $\bar{\lambda}_l$, the

auxiliary problems $\max_{\bar{\lambda}}\{c_i(\hat{V}, \bar{\lambda}) \mid \bar{\lambda} \in U_{\bar{\lambda}l}\}$ has only a finite number of optima. This property allows replacement of SNP by NP. Also, $\tilde{V}$ is a local optimal of SNP if and only if $\tilde{V}$ is a local optimal of NP. For further details, refer to Ref. 14.

B. Optimization Loop

The results of Sec. III.A allow us to solve SNP by solving NP using the following steps:

1) For the iteration step $k = 1$, any set of vertices $V_k \in D$ are chosen. For $k > 1$, use the vertices V_k from the previous iteration.

2) For a $B_i \subset B$, solve the auxiliary problems $\max_{\bar{\lambda}}\{c_i(V, \bar{\lambda}) \mid \bar{\lambda} \in B_i\}$ as nonlinear programming problems. Collect these optima as $\{\bar{\lambda}_1^*, \bar{\lambda}_2^*, \ldots, \bar{\lambda}_r^*\}$.

3) Formulate the locally reduced problem

$$\text{NP}(V_k): \quad \max\left\{\text{Volume } P(V) \mid V \in U_{V_k}\right\}$$
$$\text{s.t.} \quad c\left(V, \bar{\lambda}_1^*, \bar{\lambda}_2^*, \ldots, \bar{\lambda}_r^*\right) \le 0 \tag{10}$$

Calculate an optimal solution $\tilde{V}_k$ of $\text{NP}(V_k)$.

4) Repeat from step 1 using $V_k = \tilde{V}_k$ until no further improvement is observed, that is, $\tilde{V}_k \approx \tilde{V}_{k-1}$. In this case, $\tilde{V}_k$ is in the immediate vicinity of $\tilde{V}_{k-1}$ and the locally reduced problem $\text{NP}(V_k)$ is a local optimal for the corresponding SNP.

C. Algorithm Implementation

The polytopic approximation is performed offline. The steps of the computer implementation are 1) creation of an initial polytope, 2) use of local reduction techniques to set up a finite grid, 3) solution of the finite-dimensional optimization problem, and 4) iterations to determine an optimal polytope.

In the implementation, we use polytopes with fixed numbers of facets and vertices. We allow the locations of vertices and orientation of facets to change. We assume that the polytope has N_f facets and N_v vertices. If working in a space of dimension d, there is a total of $(N_f + N_v)d$ variables. Additionally, we define F to be the set of facets of the polytope and Q as the set of equations describing the polytope.

The initial polytope is chosen by selecting vertices on the surface of a unit hypersphere and creating a convex hull H of these vertices. From this convex hull, we extract all subsequent lower dimensional facets. For example, if H was a tetrahedron, it has four bounding planes (the 2 facets), six edges (the 1 facets) and four vertices (the 0 facets). From each facet, we then determine the set of vertices V and the constraint equations Q.

The volume of the polytope is computed by breaking it down into simplices. The volume of an n simplex with vertices $\{V_1, V_2, \ldots, V_{n+1}\}$ is given by[15]

$$\det[V_1 - V_{n+1}, V_2 - V_{n+1}, \ldots, V_n - V_{n+1}]/n! \tag{11}$$

To use this result, the following steps are performed:

1) The centroid V_c of the polytope is computed, where

$$V_c = \frac{1}{N_v}\sum_{i=1}^{N_v} V_i$$

2) For each facet, Delaunay tessellation is performed to form the individual Delaunay cells. Along with the centroid of the polytope, each Delaunay cell forms a simplex.

3) Evaluate the volume of each simplex and add for all facets.

In the optimization, a fixed number points is chosen on each facet. By the use of these points, a finite grid of points is created to cover the facet. The constraints are evaluated at each point on the grid to obtain a set of nonlinear equations in terms of vertice coordinates. The optimization is performed using NPSOL. A local reduction step from SNP to NP preceeds the volume maximization step. During volume maximization, the optimization parameters are the elements of the sets V and F.

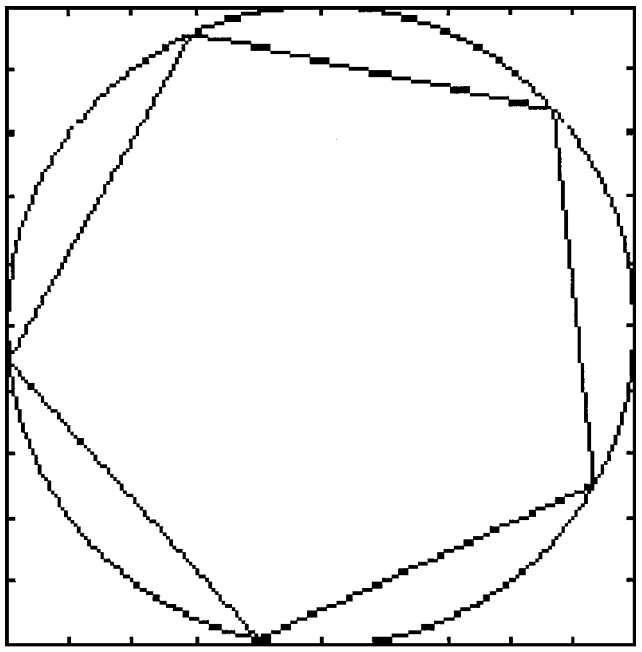

Fig. 2 Largest pentagon inscribed in a unit circle (Sec. III.D, problem 1).

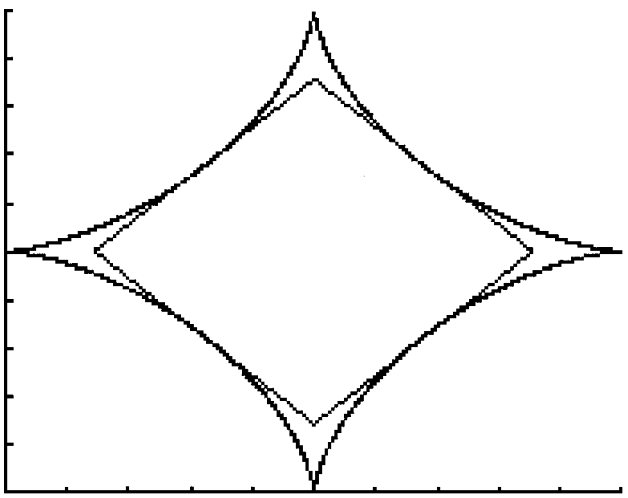

Fig. 3 Largest quadrilateral in a prescribed non-convex region (Sec. III.D, problem 2).

D. Results

The proposed polytopic approximation has been successfully applied to problems of varying dimension. In all cases, the initial vertices were specified by choosing N_v vertices to lie on a unit hypersphere. Here, we present a few results: 1) The largest pentagon contained in a unit circle is an intuitively obvious problem. Figure 2 describes the largest pentagon in a circle obtained by the program. 2) A nonconvex region is described by the equation $D = \{x \mid x \in \mathbb{R}^2,\ x_1^{2/3} + x_2^{2/3} - 1 \le 0\}$. The largest quadrilateral within D using the algorithm is shown in Fig. 3. 3) For a feasible region D given by $\sin 2x_1 + \tan x_2 - 1 \le 0$ and $-\sin(2x_1 - 30) - \tan x_2 + 1 \le 0$, the largest hexagon within D is shown in Fig. 4. 4) Consider a feasible region D given by $x_2^2 + x_3^2 - \sin x_1 \cos x_2 - 1 \le 0$ for which the feasible region D is shown in Fig. 5. The largest polytope, with 8 vertices and 12 facets, lying within D is shown in Fig. 5. The polytopic approximation program has also been successfully applied to problems in higher dimensions such as 4, 6, and 9.

IV. Trajectory Generation

With the use of the results of Sec. III, the nonlinear inequalities are inner approximated by a set of linear inequalities in the higher-order

space of $z(t)$ and its derivatives. The objective now is to develop consistent trajectories $z(t)$ over $[0, t_f]$ to steer the system with given boundary conditions. We choose $z(t)$ to have the following form:

$$z(t) = \Phi_0(t) + \sum_{j=1}^{k} \bar{a}_j \phi_j(t) \tag{12}$$

where $z(t)$ is an admissible solution.[16] Here, $\Phi_0(t)$ are m-dimension vector functions that satisfy boundary conditions of $z(t)$ and its derivatives, $\phi_j(t)$ are scalar functions that satisfy the boundary conditions in their homogeneous form, and $\bar{a}_j$ are m-dimension vectors of constant coefficients. Here, $\phi_j(t)$ are chosen such that together with $\Phi_0(t)$, they span a complete set. There are many possible choices of these basis functions as described in Ref. 7 and other related papers by the authors.

When the form of $z(t)$ from Eq. (12) is substituted in Eq. (6), the inequalities become

$$\sum_{i=0}^{p} \sum_{j=1}^{k} M_i \bar{a}_j \phi_j^{(i)}(t) + \sum_{i=0}^{p} M_i \Phi_0^{(i)}(t) + \boldsymbol{e} \leq 0 \tag{13}$$

Here, a term such as $\phi_j^{(i)}(t)$ represents the ith time derivative of $\phi_j(t)$. Equation (13) must be valid at all points in $[0, t_f]$, that is, it represents an infinite number of constraints on the coefficients $\bar{a}_1, \ldots, \bar{a}_k$. We observe that, with the form of the solution in Eq. (12), the coefficients $\bar{a}_j$ in Eq. (13) appear linearly while the time t appears nonlinearly. A number of schemes may be used to transform these infinite constraints into a finite number of constraints. We use a collocation grid in time to form a finite number of linear inequality constraints in the elements of the coefficients $\bar{a}_j$. In Eq. (12), the

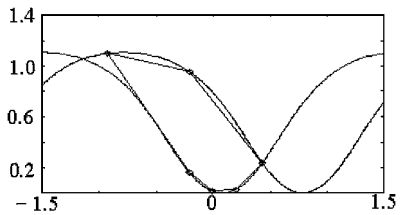

Fig. 4 Largest hexagon within the specified nonconvex region (Sec. III.D, problem 3).

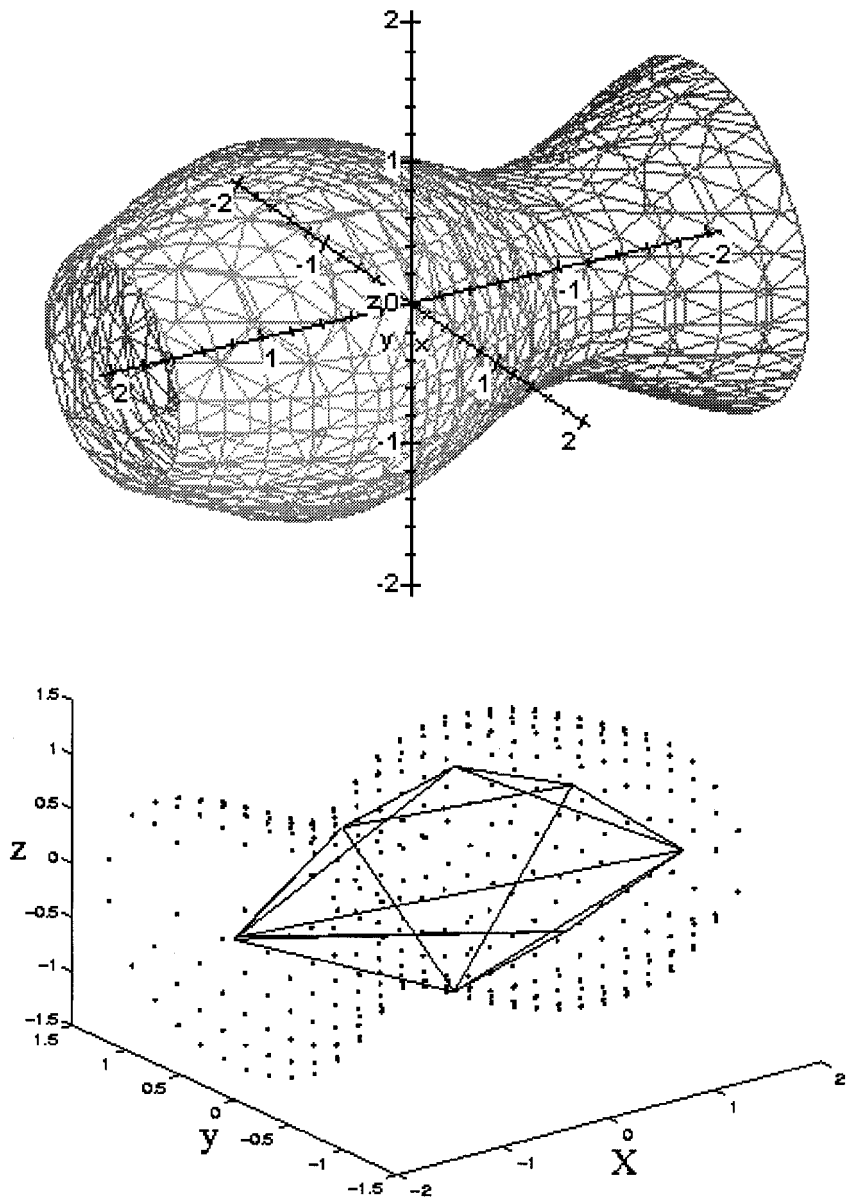

Fig. 5 Largest eight-sided polytope in the given nonconvex region (Sec. III.D, problem 4).

number of modes k can be taken to be infinitely large. However, in practice, one would work with only a finite number of modes.

A. Collocation Scheme

A finite collocation grid is selected within $[0, t_f]$. At each collocation point, the constraint functions are staisfied. If needed, one can ensure satisfaction of the constraints in between the collocation points by bounding a finite number of derivatives of the constraints at the collocation points. For the system described in Eq. (13), $N+2$ collocation points $t_0, t_1, \ldots, t_N, t_f$ are chosen such that $t_0 < t_1 < t_2 < \cdots < t_N < t_f$. The l inequality constraints of Eq. (13) result in a total of $(N+2)l$ linear inequalities on the km elements of the mode coefficients $\bar{a}_1, \ldots, \bar{a}_k$. In general, these linear inequalities enclose a convex polyhedral region in the coefficient space and characterize the feasible trajectories of the dynamic system within the form of Eq. (12).

B. Linear Inequalities and Convex Sets

It is clear that finding a solution of Eq. (13) is equivalent to characterizing the convex feasible region bounded by $I = (N+2)l$ linear inequalities on the mk elements of $\bar{a}_1, \ldots, \bar{a}_k$. This section briefly summarizes the computational aspects of determining the feasible region enclosed by a set of inequalities. For more details, the readers are referred to Ref. 17.

Every constraint partitions $\mathcal{R}^{mk}$ into a half space. Let the intersection of these half spaces be denoted by a convex set $\mathcal{K}$. It is possible for $\mathcal{K}$ to be 1) empty, if the inequalities are inconsistent, 2) a bounded polytope, or 3) an unbounded convex region. These possibilities are shown in Fig. 6. It is implicitly assumed in this section that the system of inequalities is consistent, that is, the convex set $\mathcal{K}$ is nonempty. To characterize the feasible region $\mathcal{K}$, one needs to define some terminologies and mathematical procedures.

In general, a system of linear inequalities is nonhomogeneous, that is, contains constant terms. We label such a system as $\mathcal{N}(\mathcal{I})$. Corresponding to $\mathcal{N}(\mathcal{I})$, one can define a homogeneous system $\mathcal{H}(\mathcal{I})$ with all constant terms set to zero. Also, one can define $\mathcal{N}(\mathcal{E})$ corresponding to $\mathcal{N}(\mathcal{I})$ and $\mathcal{H}(\mathcal{E})$ corresponding to $\mathcal{H}(\mathcal{I})$, by replacing the inequalities by equalities. The characteristics of $\mathcal{K}$ depend on the properties of these four systems: $\mathcal{N}(\mathcal{I})$, $\mathcal{H}(\mathcal{I})$, $\mathcal{N}(\mathcal{E})$, and $\mathcal{H}(\mathcal{E})$.

A system of inequalities $\mathcal{N}(\mathcal{I})$ is labeled as normal if the corresponding system of homogeneous equations $\mathcal{H}(\mathcal{E})$ has only null solution. We will first consider the case where the system is normal and later extend these ideas to systems that are not normal. A normal system of inequalities $\mathcal{N}(\mathcal{I})$ possesses vertices, which are points of $\mathcal{K}$ that are not interior to any line segment lying entirely within $\mathcal{K}$.

To find the vertices of $\mathcal{K}$, from $\mathcal{N}(\mathcal{E})$ consisting of I equations, we solve all combinations of mk equations that have unique solution points. In general, there are ${}^{I}C_{mk}$ such combinations. From these solutions, we discard those that do not satisfy the original inequalities $\mathcal{N}(\mathcal{I})$. Let us assume that this process results in a set of points $P_1, \ldots, P_H$. These H points are the vertices of $\mathcal{K}$. Using these H points, a convex hull $\langle P_1, P_2, \ldots, P_H \rangle$ is defined as follows:

$$\langle P_1, P_2, \ldots, P_H \rangle = s_1 P_1 + s_2 P_2 + \cdots + s_H P_H$$
$$s1, s2, \ldots, s_l \geq 0, \qquad s_1 + s_2 + \cdots + s_l = 1 \tag{14}$$

To fully characterize $\mathcal{K}$, one needs to check the existence of a polyhedral cone. This is done by solving all combinations of $mk-1$ equations from $\mathcal{H}(\mathcal{E})$. There are ${}^{I}C_{mk-1}$ such choices, and corresponding to each, we pick a solution and its negative and check if $\mathcal{H}(\mathcal{I})$ is satisfied by any of these two points. All points that satisfy $\mathcal{H}(\mathcal{I})$ through this procedure are retained and are labeled as $N_1, \ldots, N_C$. The convex cone is defined as $(N_1, \ldots, N_C)$, where

$$(N_1, N_2, \ldots, N_C) = r_1 N_1 + r_2 N_2 + \cdots + r_C N_C$$
$$r1,\ r2,\ \ldots, r_C \geq 0 \tag{15}$$

If the system of linear inequalities $\mathcal{N}(\mathcal{I})$ is normal with the convex hull of its vertices $\langle P_1, P_2, \ldots, P_H \rangle$ and the convex cone $(N_1, N_2, \ldots, N_C)$, then the domain $\mathcal{K}$ has the following analytical form:

$$\mathcal{K} = \langle P_1, P_2, \ldots, P_H \rangle + (N_1, N_2, \ldots, N_C) \tag{16}$$

From this characterization, if $(N_1, N_2, \ldots, N_C)$ is empty, the domain $\mathcal{K}$ is a convex bounded polytope. If $\langle P_1, P_2, \ldots, P_H \rangle$ is empty, the domain $\mathcal{K}$ is a convex cone. In general, $\mathcal{K}$ is a semibounded convex region. If both $(N_1, N_2, \ldots, N_C)$ and $\langle P_1, P_2, \ldots, P_H \rangle$ are empty, $\mathcal{K}$ is empty.

If the system of inequalities $\mathcal{N}(\mathcal{I})$ is not normal, the corresponding system of homogeneous equations $\mathcal{H}(\mathcal{E})$ contains a higher dimensional subspace $\mathcal{L}$ besides the origin. For such a case, $\mathcal{K}$ has the following form:

$$\mathcal{K} = \mathcal{K}_r + \mathcal{L} \tag{17}$$

where $\mathcal{K}_r$ is the convex region enclosed by a reduced set of inequalities $\mathcal{N}_r(\mathcal{I})$ that is obtained from $\mathcal{N}(\mathcal{I})$ by setting an appropriate number of variables to zero that equals the dimension of $\mathcal{L}$ (Ref. 17).

C. Computation Time

It is assumed in this section that the inequalities do not change during trajectory recomputation but the goal point does. A motivating example is a pursuit problem. A chaser continuously pursues a target and applies inputs to follow the target motion. In general, the inequalities $\mathcal{N}(\mathcal{I})$ can assumed to be normal and $I > mk$. To characterize $\mathcal{K}$, the main computation steps are to find the vertices P_i of the convex hull and N_i of the convex cone. To compute the vertices P_i, one requires solving ${}^{I}C_{mk}$ sets of linear equations in mk variables, followed by inequality checks to determine feasibility with respect to $\mathcal{N}(\mathcal{I})$. In the computation of vertices N_i, one requires solving ${}^{I}C_{mk-1}$ sets of linear equations in $mk-1$ variables, followed by inequality checks to determine feasibility with respect to $\mathcal{H}(\mathcal{I})$.

Because of the nature of the involved computations, that is, solution is required of sets of linear equations, one can put a time stamp on the completion time in accordance with the capabilities of the computer. However, such a time of completion is impossible to determine if the constraints are kept in their original nonlinear form. Also, if the problem had not been posed in the higher-order space of z and its derivatives, the state equations would have to be additionally satisfied at collocation points. Because we are looking for a feasible solution, from a practical point of view, the solution can be often obtained using only a few mode functions. The number of collocation points can be kept to a minimum from computational considerations of the problem. We demonstrate these notions through examples in the next section.

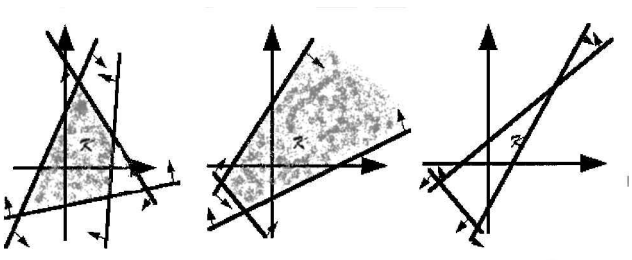

Fig. 6 Convex sets $\mathcal{K}$ in two variables with a consistent feasible region, an unbounded feasible region, and an inconsistent feasible region.

V. Examples

The methods proposed in this paper are illustrated by two examples. In the first example, a laboratory experiment is used to highlight the real-time implementation aspects. This example has upper and lower bound constraints on states and input and is quite representative of constraints of a large number of physical systems. The second example is of a planar vertical takeoff and landing (PVTOL) aircraft, which captures a simplified model for a V-22 Osprey aircraft.

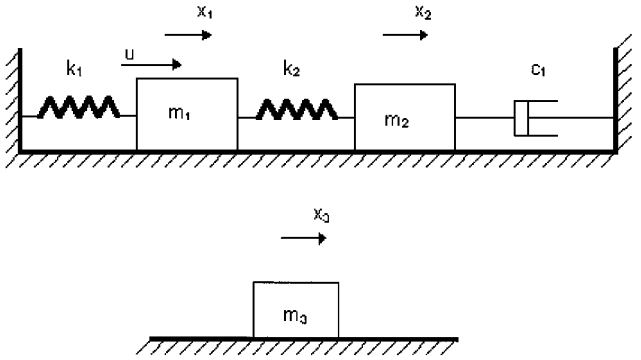

Fig. 7 Schematic model of the three-mass–spring–damper system.

A. Mass–Spring–Damper System

This example is motivated from a rectilinear spring–mass–damper experiment shown in Fig. 7. The system is driven by a single actuator acting on mass 1 while the positions and rates of the three masses are available to the controller through encoders. To demonstrate aspects of real-time planning, mass 3 is physically disconnected from the other masses. It is then used as a hand held joystick to command the motion of the rest of the system, which now consists of masses 1 and 2. This experiment is designed to demonstrate real-time planning and online implementation during pursuit situations.

A schematic of this system is shown in Fig. 7. It consists of two degrees of freedom and a single input, that is, $n = 4$ and $m = 1$. The equations of motion for this system are

$$m_1 x_1^{(2)} = -k_2(x_1 - x_2) - k_1 x_1 + u$$

$$m_2 x_2^{(2)} = k_2(x_1 - x_2) - c x_2^{(1)} \tag{18}$$

From the structure, these equations can be rewritten in the following higher-order form without change of coordinates:

$$x_2 = z \tag{19}$$

$$x_1 = (m_2/k_2)z^{(2)} + (c/k_2)z^{(1)} + z \tag{20}$$

$$u = (m_1 m_2/k_2)z^{(4)} + (m_1 c_2/k_2)z^{(3)} + \{m_1 + [m_2(k_1 + k_2)/k_1]\}z^{(2)} + [c_2(k_1 + k_2)/k_2]z^{(1)} + k_1 z \tag{21}$$

From the physical limitations, the motion of the system must satisfy the following two-sided constraints: $|x_1| \le x_{1l}$, $|x_2| \le x_{2l}$, and $|u| \le u_l$. On substituting from Eqs. (21), we get

$$|z| \le x_{2l}, \qquad |(m_2/k_2)z^{(2)} + (c/k_2)z^{(1)} + z| \le x_{1l}$$

$$|(m_1 m_2/k_2)z^{(4)} + (m_1 c_2/k_2)z^{(3)} + \{m_1 + [m_2(k_1 + k_2)/k_1]\}z^{(2)} + [c_2(k_1 + k_2)/k_2]z^{(1)} + k_1 z| \le u_l \tag{22}$$

The boundary conditions at t_0 on x_1, $\dot{x}_1$, x_2, and $\dot{x}_2$ are known using the encoder readings. At t_f, the boundary conditions on x_2 and $\dot{x}_2$ are determined from sampled position and rate of slider 3. As a result, four boundary conditions on $z^{(3)}$, $z^{(2)}$, $z^{(1)}$, and z at t_0 and two boundary conditions on $z^{(1)}$ and z at t_f are known. The statement of the problem is to find a feasible trajectory for the dynamic system that satisfies Eqs. (22) and the appropriate boundary conditions on derivatives of z.

We select the feasible solution of $z(t)$ to have the following form:

$$z(t) = \phi_0(t) + a_1\phi_1(t) + a_2\phi_2(t) \tag{23}$$

where $\phi_0(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^3(1 - t) + b_5 t^3(1 - t)^2$, $\phi_1(t) = t^4(1 - t)^2$, and $\phi_2 = t^5(1 - t)^2$, while assuming $t_0 = 0$ and $t_f = 1$. The coefficients b_i are determined from the six boundary conditions specified on the problem. Because of the forms of $\phi_1(t)$ and $\phi_2(t)$, here $z(t)$ is admissible regardless of the values for a_1 and a_2. Consistent with the physical setup, the following parameters were used in the simulation and hardware implementation (all in mks units): $m_1 = 0.237$, $m_2 = 0.49$, $k_1 = 800$, $k_2 = 100$, $c = 1.0$, $x_{1l} = 0.03$, $x_{2l} = 0.03$, and $u_l = 52.5$. For a representative set of boundary conditions, $(x_2, \dot{x}_2, x_1, \dot{x}_1)^T|_0 = (0.0, -0.0015, 0.0, 0.0)^T$ and $(x_2, \dot{x}_2)^T|_1 = (0.011, 0.0015)^T$, variants of the algorithm were implemented in the simulation using six equally spaced collocation points.

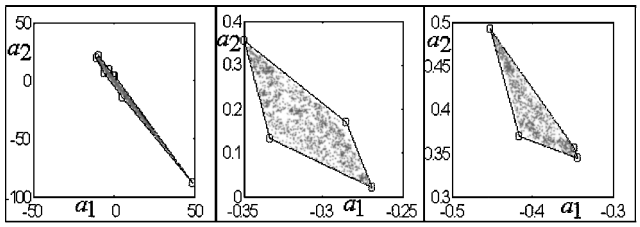

Fig. 8 Example 1 (Sec. V), feasible regions with collocation scheme: six collocation points, six collocation points with bounded first derivatives, and six collocation points with bounded first and second derivatives.

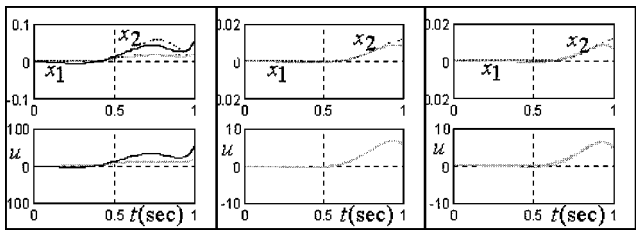

Fig. 9 Example 1 (Sec. V), constraint satisfaction of a trajectory with coefficient at a vertex point and the centroid of Fig. 8: six collocation points, six collocation points with bounded first derivatives, and six collocation points with bounded first and second derivatives.

Figure 8 shows the feasible region in the coefficient space with the following specifications: 1) constraints are satisfied at the collocation points, 2) constraints and their first derivatives are bounded at the collocation points, and 3) constraints and their first and second derivatives are bounded at the collocation points. With 6 constraints, 6 collocation points, and 2 mode functions, the convex sets $\mathcal{K}$ in the three cases are described, respectively, by 36, 72, and 108 linear inequalities in two variables. All computations were performed symbolically using MAPLE. From Fig. 8, one observes that the size of the feasible region reduced progressively as further bounds on the derivatives of the constraints are added. For each case, trajectories corresponding to a vertex point and centroid of the feasible region were checked for constraint satisfaction. The results are plotted in Fig. 9. From these plots, we notice that, as higher derivatives of the constraints are bounded, the trajectories satisfy the constraints throughout the domain.

A Pentium 166-MHz personal computer running Windows 95 was used to host ECPUSR software and its scripting language compiler.[18] The real-time trajectory generator and tracker were written within the ECPUSR environment. As a result, the experiments were limited by compiler restrictions in terms of character length of the program. The experiments were carried out using 3 equally spaced collocation points. With 6 constraints, 3 collocation points, and 2 modes, the feasible space was characterized by a system of 18 linear inequalities in 2 variables.

The controller samples the position and speed of mass 3 every second and uses this to assign goal position and speed of mass 2 to be attained at the end of next second. The current position and speed of masses 1 and 2 are used to generate the four initial conditions on derivatives of y. In ECPUSR environment, the trajectory generation takes roughly 20 ms, and the data are implemented over the next 0.98 s.

In the hardware implementation, an exponential trajectory tracker is designed to make $z(t)$ asymptotically follow $z_d(t)$. The form of the input $u(t)$ is

$$u(t) = \beta_4 z_d^{(4)}(t) + \beta_3 z_d^{(3)}(t) + \beta_2 z_d^{(2)}(t) + \beta_1 z_d^{(1)}(t) + \beta_0 z_d(t) + \gamma_3 e^{(3)}(t) + \gamma_2 e^{(2)}(t) + \gamma_1 e^{(1)}(t) + \gamma_0 e(t) \quad (24)$$

where $e(t) = z_d(t) - z(t)$, β_i are the coefficients of $z^{(i)}$ in the higher-order model of the system, and γ_i are the feedback gains selected to make the error dynamics stable, that is, the solution $e(t)$ of the equation

$$e^{(4)} + [(\beta_3 + \gamma_3)/\beta_4]e^{(3)}(t) + [(\beta_2 + \gamma_2)/\beta_4]e^{(2)}(t) + [(\beta_1 + \gamma_1)/\beta_4]e^{(1)}(t) + [(\beta_0 + \gamma_0)/\beta_4]e(t) = 0 \quad (25)$$

approaches from an initial condition to zero with a desired transient dynamics. Figure 10 shows the results of an experiment where mass 3 is moved over a length of time and mass 2 follows the motion of mass 3 while satisfying the six constraints. From Fig. 10 it is evident that the proposed algorithm has been successfully implemented in real-time. A movie clip of this experiment is also available at http://mechsys4.me.udel.edu.

B. PVTOL Aircraft

We now address a PVTOL aircraft that captures some essential dynamics of vertical takeoff operation of V-22 Osprey, a tiltrotor aircraft designed by The Boeing Company and Bell Helicopters Textron, Inc. We analyze a phase of its motion in the vertical plane. Some relevant data are taken from the documented data of V-22. Figure 11 shows a PVTOL as a planar body B with a single thruster F. The position of the center of mass C in inertial frame (O_g, X_g, Y_g) is (x, y). The mass of body B is m, and its inertia at C about an axis perpendicular to the plane of the body is J. Frame (O_b, X_b, Y_b) is attached to body B with O_b at a distance l from C. $\bar{Y}_b$ is along O_bC and the angle $\alpha = \angle(\bar{X}_g \bar{X}_b)$. The orientation of the thruster is independently controlled by the angle $\theta = \angle(\bar{X}_b \bar{F})$. The thruster force F is resolved into its two independent components u_1 and u_2, aligned along axis Y_b and X_b, respectively, where $u_2 = F\cos\theta$ and $u_1 = F\sin\theta$. The equations of motion of body B are

$$m\ddot{x} = -u_1 \sin\alpha + u_2 \cos\alpha$$
$$m\ddot{y} = u_1 \cos\alpha + u_2 \sin\alpha - mg, \qquad J\ddot{\alpha} = lu_2 \quad (26)$$

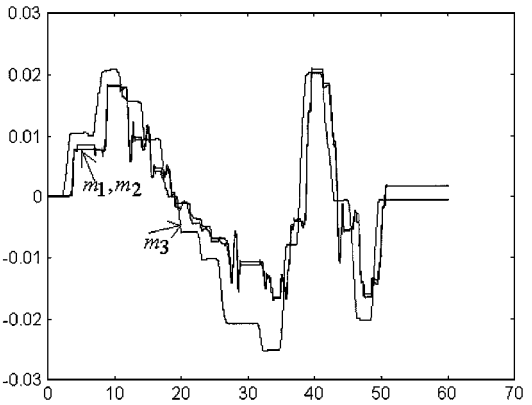

Fig. 10 Data of a pursuit experiment where mass 3 is moved by a user and mass 2 follows mass 3 while satisfying the constraints; overlays the commanded motion of mass 3 and followed motion of masses 2 and 1.

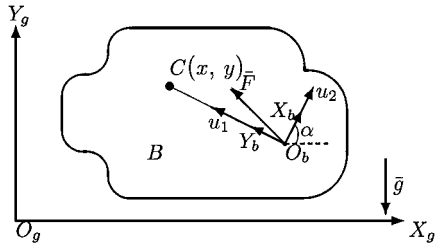

Fig. 11 PVTOL modeled as a planar body with two thrusters.

The problem is motivated from a made up scenario of a motion where the aircraft takes off from a helipad and reaches a predetermined altitude, while maintaining a safe flying envelope shown in Fig. 12. The planned trajectories must satisfy the flying envelope constraints as well as the system equations (26). The constraints are mathematically modeled as follows:

Envelope A:

$$750 \sin\frac{\pi(x - 100)}{2500} \le y \quad (27a)$$

Envelope B:

$$y \le 1500 \sin\frac{\pi x}{2200} \quad (27b)$$

Physical limits on x:

$$0 \le x \le 1000 \quad (27c)$$

Physical limits on y:

$$0 \le y \quad (27d)$$

The final time is assumed to be 1000 s, and the boundary conditions are selected in mks units as

$$x(0) = 50, \qquad y(0) = 1, \qquad \dot{x}(0) = 0$$
$$\dot{y}(0) = 0, \qquad \ddot{x}(0) = 0 \quad (28)$$
$$x(t_f) = 850, \qquad y(t_f) = 850, \qquad \dot{x}(t_f) = 0$$
$$\dot{y}(t_f) = 0, \qquad \ddot{y}(t_f) = 0 \quad (29)$$

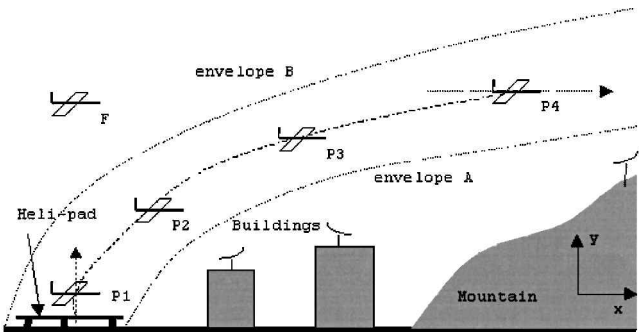

Fig. 12 Scenario of a motion planning problem.

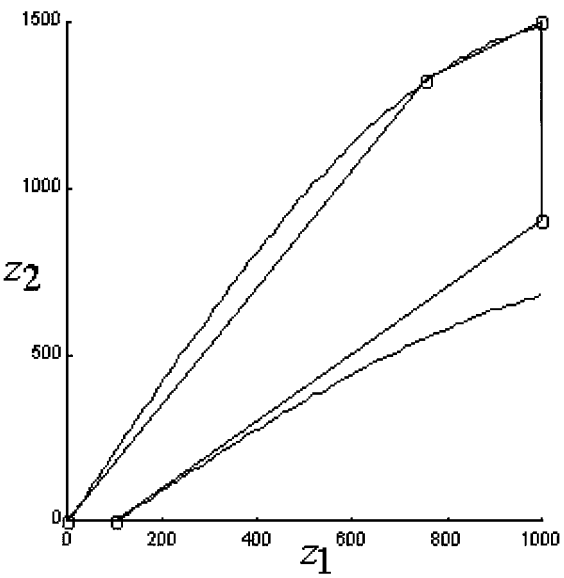

Fig. 13 Polytopic approximation of the feasible region for example 2 (Sec. V).

For the given model, $l = 0.435$ m, $m = 17{,}000$ kg, and $J = 256{,}388.6\,\text{kgm}^2$.

For this system, the flat outputs z_1 and z_2 are[19]

$$z_1 = x - (1/\lambda)\sin\alpha, \qquad z_2 = y + (1/\lambda)\cos\alpha \tag{30}$$

where $\lambda = ml/J$. The nonlinear inequality constraints (27) are mapped into the flat space, and Fig. 13 shows the polytopic approximation within the region. The approximating polytope with five sides is represented by five linear inequalities:

$$0 \le z_2, \qquad z_1 \le 1000, \qquad 0.01z_1 - 0.01z_2 \le 1$$
$$-0.000928z_1 + 0.001285z_2 \le 1, \qquad z_2 \le 1.7457z_1 \tag{31}$$

The linear inequalities in Eq. (31) are satisfied by the trajectory using the scheme presented in this paper. The forms of the solution for the flat outputs are chosen as

$$z_1 = \Phi_{10} + a_{11}\Phi_{11} + a_{12}\Phi_{12}, \qquad z_2 = \Phi_{20} + a_{21}\Phi_{21} + a_{22}\Phi_{22} \tag{32}$$

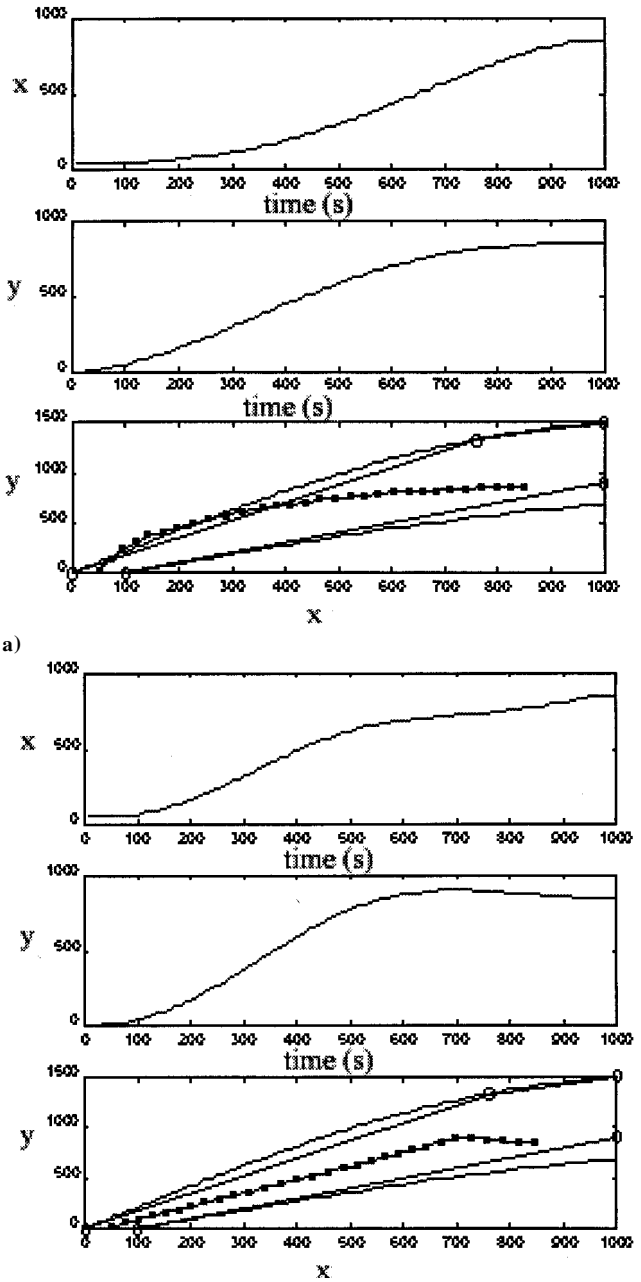

Fig. 14 Example 2 (Sec. V): a) trajectories using only Φ_{10} and Φ_{20} violate the constraints between 50 and 250 s, b) on collocation, the trajectory satisfies the constraints at all points.

where Φ_{10} and Φ_{20} satisfy the boundary conditions. Their expressions are taken as

$$\Phi_{10} = a_0 + a_1 t + a_2 t(t_f - t) + a_3 t^2(t_f - t) + a_4 t^2(t_f - t)^2$$
$$\Phi_{20} = b_0 + b_1 t + b_2 t(t_f - t) + b_3 t^2(t_f - t) + b_4 t^2(t_f - t)^2$$
$$\Phi_{11} = \Phi_{21} = t^2(t_f - t)^3, \qquad \Phi_{12} = \Phi_{22} = t^3(t_f - t)^3 \tag{33}$$

When five collocation points are used evenly over $t \in [0, 1000]$, linear inequalities in the coefficients a_{11}, a_{12}, a_{21}, and a_{22} are obtained. This set is then solved to determine the vertices of a polytope in a_{11}, a_{12}, a_{21}, and a_{22} space. The centroid of the polytope is then selected for trajectory generation. Figure 14 shows the feasible trajectories. From Figs. 14, it is clear that a trajectory using only Φ_{10} and Φ_{20} violates the constraints. However, on imposing the constraints through collocation, the constraints are satisfied over time.

VI. Conclusions

This paper presented a method to generate feasible state and control trajectories of differentially flat systems subject to inequality constraints. For such systems, we observe that the system dynamic equations can be embedded into the constraints in the higher-order space of flat outputs. In this space, the constraints are usually nonlinear and nonconvex. As a result, the nonlinear inequalities are inner approximated by a set of linear inequalities, performed offline using results from semi-infinite optimization theory. The solution for the flat variables are determined within a space of finite number of basis functions. Because of the structure of linear inequalities and collocation grid in time, the feasible solution is characterized by a set of linear inequalities in the mode coefficients associated with the basis functions. This computation is performed online, and a completion time can be computed a priori, determined by the number of modes and size of the collocation grid. This approach was demonstrated in hardware on a pursuit maneuver with a spring–mass–damper system. Also, simulation results were provided for a PVTOL, motivated from Osprey aircraft with simplified dynamic models. Even though the feasibility of the approach has been successfully demonstrated and the key steps have been identified, there is room to refine the steps to make the algorithm computationally more attractive.

Acknowledgments

The support of National Science Foundation Presidential Faculty Fellowship to the second author is gratefully acknowledged. This work was also partially supported by Air Force Office of Scientific Research during sabbattical leave of the second author at the California Institute of Technology.

References

[1]Latombe, J. C., *Robot Motion Planning*, Kluwer Academic, Norwell, MA, 1991.

[2]Lin, C., Chang, P., and Luh, J. Y. S., "Formulation and Optimization of Cubic Polynomial Joint Trajectories for Industrial Robots," *IEEE Transactions on Automatic Control*, Vol. AC-28, No. 12, 1983, pp. 1066–1073.

[3]Bryson, A. E., and Ho, Y. C., *Applied Optimal Control*, Hemisphere, Washington, DC, 1975, pp. 1–125.

[4]Kirk, D. E., *Optimal Control Theory: An Introduction*, Electrical Engineering Series, Prentice–Hall, Engelwood Cliffs, NJ, 1970, pp. 105–315.

[5]Fliess, M., Levine, J., Martin, P., and Rouchon, P., "Flatness and Defect of Nonlinear Systems: Introductory Theory and Examples," *International Journal of Control*, Vol. 61, No. 6, 1995, pp. 1327–1361.

[6]Nieuwstadt, M. J., and Murray, R. M., "Real Time Trajectory Generation for Differentially Flat Systems," *International Journal of Robust and Nonlinear Control*, Vol. l8, No. 11, 1998, pp. 995–1020.

[7]Agrawal, S. K., and Veeraklaew, T., "A Higher-Order Method for Dynamic Optimization of a Class of Linear Time-Invariant Dynamic Systems," *Journal of Dynamic Systems, Measurements, and Control*, Vol. 118, No. 4, 1996, pp. 786–791.

[8]Agrawal, S. K., and Faiz, N., "A New Efficient Method for Optimization of a Class of Nonlinear Systems Without Lagrange Multipliers," *Journal of Optimization Theory and Applications*, Vol. 97, No. 1, 1998, pp. 11–28.

[9]Gomory, R. E., "Solving Linear Programming Problems in Integers," *Combinatorial Analysis*, 1960, pp. 211–215.

[10]Cheney, E. W., and Goldstien, A. A., "Newton's Method for Convex Programming and Tchebycheff Approximation," *Numerische Mathematik*, Vol. 1, 1959, pp. 253–268.

[11]Kelley, J. E.,"The Cutting Plane Method for Solving Convex Programs," *SIAM Journal*, Vol. 8, 1960, pp. 703–712.

[12]Tuy, H., "Concave Programming Under Linear Constraints," *Soviet Mathematics*, Vol. 5, 1964, pp. 1437–1440.

[13]Glover, F., "Polyhedral Annexation in Mixed Integer and Combinatorial Programming," *Mathematical Programming*, Vol. 8, 1975, pp. 161–188.

[14]Hettich, R., and Kortanek, K., "Semi-Infinite Programming: Theory, Methods and Applications," *SIAM Review*, Vol. 35, No. 3, 1993, pp. 380–429.

[15]Büeler, B., Enge, A., and Fukada, K., "Exact Volume Computation for Polytopes: A Practical Study," *Polytopes—Combinatorics and Computation*, edited by G. Kalai and G. M. Ziegler, DMV-Seminars Series, Birkhäuser Verlag, 1999.

[16]Fletcher, C. A. J., *Computational Galerkin Methods*, Springer-Verlag, Berlin, 1984.

[17]Solodovnikov, A. S., *System of Linear Inequalities*, Univ. of Chicago Press, Chicago, 1980.

[18]Manual for Model 210/210a Rectilinear Control System, ECP Educational Products, Inc., Bell Canyon, CA, 1996.

[19]Martin, P., Devasia, S., and Paden, B., "A Different Look at Output Tracking: Control of a VTOL Aircraft," *Automatica*, Vol. 32, No. 1, 1996, pp. 101–107.