To appear, J. Guidance, Navigation and Control
February 2000

Differentially Flat Systems with Inequality Constraints: An Approach to Real-time Feasible Trajectory Generation

Nadeem Faiz*, Graduate student
Sunil K. Agrawal†, Associate Professor
Mechanical Systems Laboratory, Department of Mechanical Engineering
University of Delaware, Newark, DE 19716.

Richard M. Murray‡, Associate Professor
Division of Engineering and Applied Science
California Institute of Technology, Pasadena, CA 91125.

Abstract

This paper proposes a real-time planning scheme and its implementation for a class of dynamic systems. The planner is aimed to satisfy the state equations, the path and actuator constraints, and the given initial and terminal constraints. In order to generate trajectories in real-time, three broad steps are performed: (i) the structure of differentially flat systems is used to explicitly encapsulate the state equations into linear differential constraints in the flat space, and appropriately transform the boundary conditions; (ii) using semi-infinite optimization theory, an inner approximation of nonlinear constraints is made to replace these by a set of linear inequalities in the flat space, i.e., by a polytope; (iii) this polytopic representation of the system that satisfies the state equations and the constraints is then parameterized using basis functions and the planning problem is turned around into solution of a set of linear inequalities in the coefficient space of the basis functions. It is then demonstrated that numerically efficient algorithms can be built to solve the planning problem in real-time. The essence of the approach is demonstrated by two examples: (i) an implementation is performed on a spring-mass-damper system to demonstrate the real-time capability of evasion-pursuit; (ii) a VTOL aircraft is used to illustrate the application of this approach in simulation to nonlinear problems.

Keywords: real-time planning, differential flatness, inequality constraints, convex programming, polytope.

*faiz@me.udel.edu
†agrawal@me.udel.edu
‡murray@indra.caltech.edu

1 Introduction

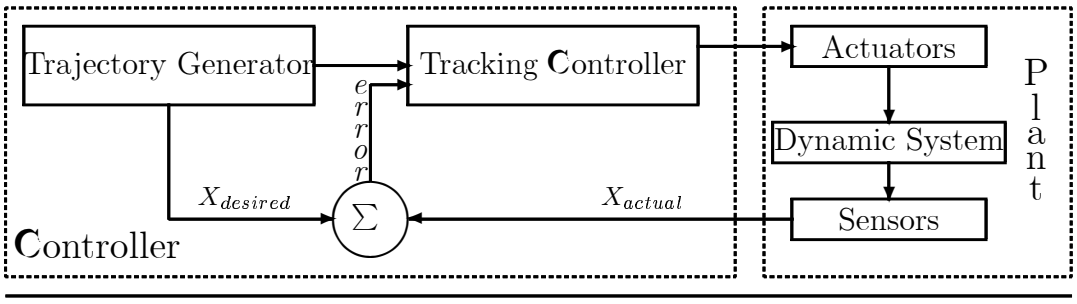

Figure 1: A schematic of a generic motion control system

Figure 1 shows the schematic of a motion controller along with a possible implementation. This two part motion controller consists of a *trajectory generator*, and a *tracking controller*. Here, the role of the trajectory generator is to provide to the tracking controller feasible reference trajectories that satisfy the dynamic equations, path and actuator constraints, and end-point constraints. The tracker's role is to implement these trajectories to a sufficient high degree of accuracy. This two part controller is similar to two degree-of-freedom controllers found in optimal control [20], [21], [15]. Conceptually, the roles of the trajectory generator and tracking controller are different and have their own importance. A good reference command has a higher chance of being implemented by the servo controller. The purpose of this paper is to focus on the *trajectory generator* and propose methods how this block can provide and update reference commands in real-time.

The need to perform computations in real-time comes from the requirement that the system model may be inexact or the constraints and boundary conditions may be changing over time. For instance, a missile targeting a moving object needs to constantly update its trajectory to account for perceived target motion. In such situations, real-time trajectory generation is a necessity to accomplish the task. The methods proposed in this paper for trajectory generation are motivated from their real-time worthiness.

The statement of the problem is as follows: For a dynamic system

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

with the states $\bar{q} \in R^n$ and controls $\bar{u} \in R^m$, build a motion planner to determine in real-time $\bar{q}(t)$ and $\bar{u}(t)$ to take the system from an initial state $\bar{q}(0)$ to a final state $\bar{q}(t_f)$ while satisfying Eq. (1)

and path and actuator constraints

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

where the constraints $\bar{c} \ \in \ \Re^p$. Equations (1) are intrinsic to the physical system and represent mathematically its behavior. Equations (2) are constraints which have been put externally on the system. For instance, an airplane carrying passengers would execute yaw and pitch maneuvers within safe limits for passengers. A cargo carrying airplane would not be subject to the same constraints. A subclass of Eq. (2) are path constraints of the form

$$\bar{h}(\bar{q}) \ \leq \ 0, \ \bar{h} \ \in \ \Re^{p_1}, \tag{3}$$

only dependent on $\bar{q}$. Another class of constraints are actuator saturation limits

$$L_{lower} \ \leq \ \bar{u} \ \leq \ L_{upper}. \tag{4}$$

The system may also be subject to end-point constraints, e.g.,

$$\begin{aligned} \bar{q} &= \bar{q}_0, \text{ at } t = 0 \\ \bar{q} &= \bar{q}_f, \text{ at } t = t_f \end{aligned} \tag{5}$$

The main contribution of this paper is the development of real-time planners which suitably exploit the structure of the dynamic system. This problem is computation intensive, therefore, it is unrealistic to look for a technique applicable to all problems. Instead, we restrict our attention to a class of systems for which the structure of the dynamic system allows canonical transformations to a higher-order form. This higher-order form trivially satisfies the state equations and this class has been labeled as higher-order systems [1] or differentially flat systems ([10], [18]). In some earlier work, it has been hypothesized that the planning problem for this class of systems can be solved in real-time. However, all studies so far are limited to planning in the absence of inequality constraints. The flow of arguments in this paper is as follows:

- Eliminate state equations by transforming the problem to differentially flat space
- Inner approximate the path and actuator constraints by linear inequality constraints
- Solve a trajectory generation problem in the flat space with linear inequality constraints
- Transform the motion plans to the original space.

The organization of the paper is as follows: Section 2 describes higher-order or differentially flat systems. Section 3 considers inequality constraints and proposes schemes to accommodate them to real-time planning. Section 4 outlines trajectory generation when the constraints are approximated by linear forms. Examples, including experiments, are presented in Section 5.

2 Differentially Flat Dynamic Systems

In this paper, only those systems (1) are considered which admit the following alternate form

$$\bar{q} = \bar{\mathfrak{q}}(\bar{y}, \dot{\bar{y}}, \ddot{\bar{y}}, \ldots, \overset{\cdots p_1}{\bar{y}}), \tag{6}$$

$$\bar{u} = \bar{\mathfrak{u}}(\bar{y}, \dot{\bar{y}}, \ddot{\bar{y}}, \ldots, \overset{\cdots p_2}{\bar{y}}), \tag{7}$$

where $\bar{y} \in R^m$ and p_1, p_2 represent the highest derivatives. Systems which have this property have been referred to as higher-order systems [3] or differentially flat systems [10]. Examples of flat systems include controllable linear systems, feedback linearizable systems, and chained systems. Physical examples of differentially flat systems are commonly found in mechanical engineering [18], in electrical engineering [23] as well as in chemical engineering [4].

For differentially flat systems, trajectories consistent with the dynamics can be planned using $\bar{y}$. Further, using Eqs. (6) and (7), the state $\bar{q}(t)$ and input trajectories $\bar{u}(t)$ can be obtained by simple differentiations. This is definitely advantageous over its alternatives, e.g., finding a trajectory consistent with the dynamics using collocation or by shooting techniques. In some recent publications, Agrawal and coworkers have investigated the problem of optimal trajectory generation of classes of differentially flat systems which are not subject to auxiliary constraints ([1], [2], [3], [7]). Also, Murray and coworkers have investigated the feasible motion planning problem for such systems without auxiliary constraints with application to ducted fan experiments [20].

The purpose of this paper is to extend these planning techniques to systems with path and actuator constraints (3) and (4). In the present section specific example classes of differentially flat systems will be considered. Figure 2 shows a planning problem in its original and transformed space. From this figure, it is evident that the state equations in the flat space are trivially satisfied.

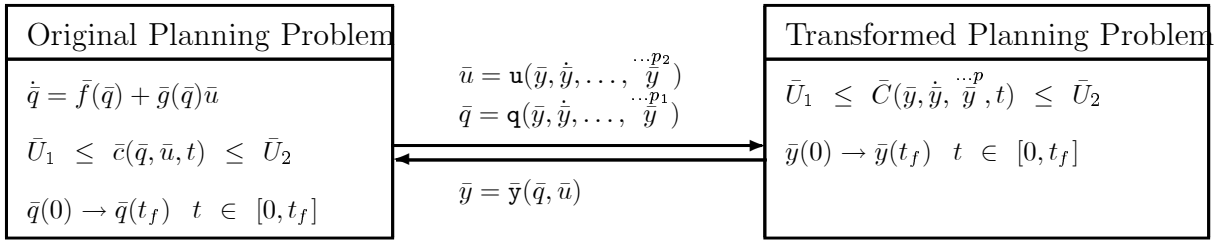

Figure 2: The planning problem in the original and transformed space.

3 Inequality Constraints: Polytopic Approximation

In Section 2, the higher-order property of differentially flat systems was used to transform the state equations to the flat space where these were trivially satisfied. As was illustrated in Figure 2, the inequality constraints are also transformed to the flat space and these equations provide the following feasible region:

$$S = \left\{ \bar{y} : \ \bar{U}_1 \ \leq \ \bar{C}(\bar{y}, \dot{\bar{y}}, ..., \overset{\cdots p}{\bar{y}}, t) \ \leq \ \bar{U}_2 \right\} \tag{8}$$

In general, the set S is non convex in the space of $\bar{y}$ and its derivatives. In order to determine S, one requires to solve nonlinear inequalities which is quite computationally expensive. In general, the boundary of set S is a collection of hypersurface patches obtained from Eq. (8). All feasible trajectories of the problem must lie entirely within this set S. An attempt to build a real-time trajectory generation scheme by solving for S is simply impossible because of computation burden.

With the goal to provide a real-time solution procedure, simplifications are proposed in this section. We will distinguish between three classes: (i) linear systems with linear inequality constraints; (ii) linear systems with nonlinear inequality constraints; and (iii) nonlinear systems with linear or nonlinear inequality constraints. These three problem classes encompass most problems of interest. It can be shown that for linear systems, on invoking a transformation to the flat space, linear inequalities remain linear [8]. This is not the case for linear systems with nonlinear constraints. However, if the constraint set is convex in the original space, under a linear transformation to the flat space, the set S will remain convex [5]. It is well known that a convex set can be well represented by a system of linear inequalities by selecting an arbitrary number of points on the boundary of this set. In summary, linear systems with linear inequalities or

convex nonlinear inequalities have a natural alternate representation of a set of linear inequalities (a polytope) in the flat space.

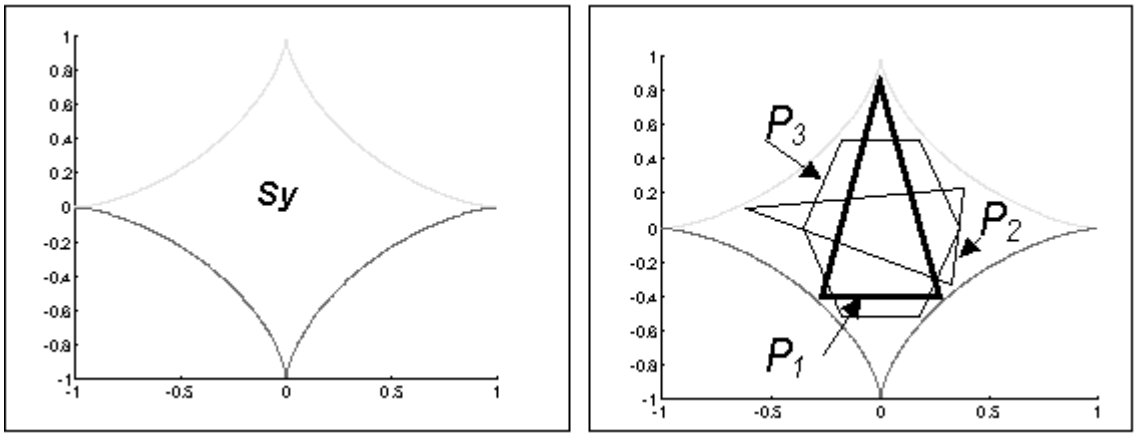

Figure 3: Polytope P_1, P_2, P_3 embedded within a non convex set S.

However, for linear systems with general nonlinear inequalities or nonlinear systems, a natural polytopic representation is not possible. Hence, in this paper, we propose to replace S by a polytope that lies entirely within it. We recall that polytopes are sets enclosed by linear inequalities. Figure 3 shows an example where polytopes P_1, P_2, P_3 are embedded within a non convex region S in two dimensions. Since each polytope is embedded within the feasible region, all points within it is feasible. As will be shown in following sections, trajectories can be generated in real-time for polytopic sets in a rather convenient manner. One challenge, however, is to approximate S by a polytope that encloses the maximum region. This is posed as an optimization problem and solved using results of semi-infinite optimization theory as described below [12].

A formal problem statement is *Given a feasible set*

$$\mathbf{S}_y = \left\{ \bar{y} : \ \bar{U}_1 \ \leq \ \bar{g}(\bar{y}, \dot{\bar{y}}, ..., \overset{...p}{\bar{y}}, t) \ \leq \ \bar{U}_2 \right\}, \tag{9}$$

find the largest polytope P *such that* $P \subset S$. Such a polytope can be written in the following form:

$$M_0 \ \bar{y} + M_1 \ \dot{\bar{y}} + ... + M_{p-1} \ \overset{...(p-1)}{\bar{y}} + M_p \ \overset{...p}{\bar{y}} + e \leq 0 \tag{10}$$

where for l, number of facets in the polytope, M_i are $(l \times m)$ matrices, and e denotes a constant $(l \times 1)$ vector.

3.1 Semi-infinite Optimization

In order to ensure that the polytope P always lies in $\mathbf{S}_y$, every point within the polytope must be verified to satisfy each constraint in Eq. (9). However, there are infinite such points and it is impossible to verify the constraints at every point. An approximate solution is to check for satisfaction of the constraint at a finite set of points.

The choice of these finite points is an open question. A good approximation scheme will use points which are *sufficient guaranteers* for all other points in P. Namely, the satisfaction of the constraints at these finite points serves a guarantee for the satisfaction of the constraints in P. One means of choosing the points is to cover the polytope with a finite grid of equally spaced points. A finer grid results in a larger number of grid points and a better approximation of the region. However, a larger number of grid points increases the number of constraints and adds to the total computation effort needed. Also the grid points must be "sufficient" points as explained.

Semi-infinite Optimization Theory provides a formal technique to solve optimization problems with semi-infinite constraints. It will provide a formal framework in generation of the finite points needed to determine a finite number of constraints. The following terminology is needed.

$$\begin{aligned} P &= \text{Polytope P} \\ \mathbf{V} &= \text{Set of Vertices of polytope P} \\ \texttt{Volume}\ P(\mathbf{V}) &= \text{Volume of Polytope P with vertices } \mathbf{V} \end{aligned}$$

From convexity theory it follows that for vertices $\mathbf{V} = \{\bar{V}_1, \bar{V}_2, \ldots, \bar{V}_q\}$, and a vector $\bar{\lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_q]$, every point $\bar{y} \in P$ is represented

$$\begin{aligned} \bar{y} &= \bar{V}_1\lambda_1 + \bar{V}_2\lambda_2 + \ldots + \bar{V}_q\lambda_q,\ \lambda_j \in [0,1],\ \sum_{j=1}^{q} \lambda_j = 1 \\ &\equiv (\mathbf{V}, \bar{\lambda}) \end{aligned} \tag{11}$$

Then $\bar{C}(\mathbf{V}, \bar{\lambda})$ represents the inequality constraints evaluated at each point in the polytope.

Problem Definition

The polytopic approximation problem is cast in an optimization theory framework and is stated as a semi-infinite optimization problem (SNP) as

$$(SNP) \equiv \max\{\texttt{Volume}\ P(\mathbf{V}) |\ \mathbf{V} \in \mathbf{D}\}, \tag{12}$$

$$\mathbf{D} = \{\mathbf{V} |\ \bar{\mathbf{C}}(\mathbf{V}, \bar{\lambda}) \leq \bar{\mathbf{0}}, \bar{\lambda} \in \mathbf{B}\} \tag{13}$$

$$\mathbf{B} = \{\bar{\lambda} \in \mathbb{R}^q = (\lambda_1, \lambda_2, \ldots, \lambda_q) |\ \lambda_j \in [0,1], \sum_{j=1}^{q} \lambda_j = 1\} \tag{14}$$

From a mathematical point of view, the semi-infinite polytopic optimization problem is a problem of semi-infinite parameter optimization, i.e. an optimization problem with a finite number of optimization variables $\mathbf{V}$ and an infinite number of inequality constraints which are parameterized by the parameter $\lambda_j \in [0,1]$. The domain of set $\mathbf{D}$ is exactly the feasible set $\mathbf{S_y}$. The set $\mathbf{B}$ always lies in the positive domain, i.e. $\mathbf{B} \subset \Re^{q^+}$, and is upper bounded by the unit hyperplane, $\lambda_1 + \lambda_2 + \cdots + \lambda_q = 1$.

Local Reduction Technique

For efficiently determining an optimal solution of (SNP), the problem is reduced to a finite-dimensional parameter optimization problem where the number of constraints is finite. We follow the *Local reduction technique* proposed by Hettich and Kortanek [12] for semi-infinite programming. A new problem (NP) is defined as

$$(NP) \quad \equiv \quad \max\{\texttt{Volume}\ P(\mathbf{V}) |\ \mathbf{V} \in \mathbf{D}\}, \tag{15}$$

$$\mathbf{D} = \{\mathbf{V} |\ \bar{\mathbf{C}}(\mathbf{V}, \bar{\lambda}) \leq \bar{\mathbf{0}}, \bar{\lambda} \in \hat{\mathbf{B}}\}, \tag{16}$$

$$\hat{\mathbf{B}} = \{\bar{\lambda}_1, \bar{\lambda}_2, \ldots, \bar{\lambda}_r\} \ \subset \ \mathbf{B}, \tag{17}$$

and a finite-dimensional optimization problem is obtained. The local reduction technique then specifies conditions for the equivalence of the locally transformed finite-dimensional parameter optimization problem $(NP(\hat{\mathbf{V}}))$ and the semi-infinite parameter optimization problem (SNP). Evaluating the constraints at the finite number of points in set $\hat{\mathbf{B}}$, results in a finite number of constraints to be verified. Satisfaction of these constraints is a sufficient condition for the satisfaction of all the constraints in (SNP). A full treatment of the local reduction technique is available in the paper by Hettich and Kortanek [12].

3.2 Examples of Polytopic Approximation

A general purpose program has been developed to implement the local reduction technique. The program is written in MATLAB [17] and uses NPSOL [11], a nonlinear programming solver. Some results from this program are shown in Figure 4. The polytopic approximation is an off-line process and needs to be performed only once.

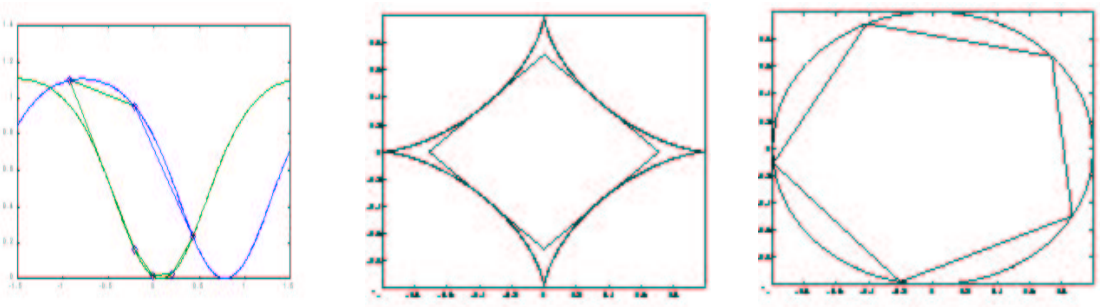

Figure 4: Polytopic approximation of some non convex regions.

4 Trajectory Generation

From the last two sections, one arrives at a simplified problem where the state equations are trivially satisfied and the nonlinear inequalities are replaced by linear inequalities. The problem now is to develop trajectories for this new system (10) subject to boundary conditions. A statement of the problem is to find a solution $y(t)$ over the time $[t_0, t_f]$ such that it satisfies Eq. (10) and boundary conditions of $\bar{y}$, $\dot{\bar{y}}$,..., $\overset{\cdots(p-1)}{\bar{y}}$ at t_0 and t_f. In this study, the solution $\bar{y}(t)$ is characterized as

$$y_j(t) = \Phi_{j0}(t) + \sum_{k=1}^{n_j} a_{jk}\Phi_{jk}(t), \qquad j = 1, \ldots, m, \tag{18}$$

where $\Phi_{i0}(t)$ satisfy all boundary conditions on $y_i(t)$, $\phi_{ij}(t)$ are scalars that satisfy the boundary conditions homogeneously, and a_{ij} are m-dimensional vectors of mode coefficients [9]. With this form, $y_i(t)$ satisfies all boundary conditions of the problem regardless of the values of a_{ij}.

On substituting $\bar{y}(t)$ from Eq. (18) in Eq. (10), and defining M_{i_j} as the j-th row of matrix M_i, the form for the inequalities becomes

$$\sum_{i=0}^{p}\sum_{j=1}^{m}\sum_{k=1}^{n_j} M_{i_j} a_{jk}\phi_{jk}^{(i)}(t) + \sum_{i=0}^{p}\sum_{j=1}^{m} M_{i_j}\Phi_{j0}^{(i)}(t) + \bar{e} \leq \bar{0}. \tag{19}$$

This equation should be valid at all points in the domain $[t_0, t_f]$, i.e., it represents an infinite number of constraints on the mode coefficients a_{11},...,a_{mn_m}. It must be noted that these constraints are linear in the elements of the mode coefficients a_{ij} and nonlinear in time.

The feasible trajectories are then characterized as follows: (i) satisfy Eq. (19) exactly or approximately within the domain by using a finite set of constraints; (ii) since the mode coefficients a_{11},...,a_{mn_m} appear linearly in these constraints, the feasible solution space of a_{11},...,a_{mn_m} is a new polytope whose vertices can be characterized in real-time by a simple computational procedure;

(iii) select a point within this new polytope and form the trajectory $\bar{y}(t)$ according to Eq. (18). This trajectory $\bar{y}(t)$ will be consistent with the state equations, constraints, and the boundary conditions posed on the problem.

4.1 Finite Constraint Set

We propose here briefly a few schemes to satisfy Eq. (19) within the domain by using a finite number of constraints. More details of implementation are available in [8].

4.1.1 Collocation Scheme

In the first scheme, a finite number of collocation points is selected within the domain of interest. At each of the collocation points, the constraint functions and a finite number of their derivatives are bounded to ensure satisfaction of Eq. (19). In order to describe this method intuitively, consider a single inequality $|g(t)| < c$. Given $g(t)$, $g(t + \Delta T)$ can be approximated by

$$g(t + \Delta T) = g(t) + g^{(1)}(t)\Delta T + ... + g^{(r)}(t)\frac{(\Delta T)^r}{r!} \tag{20}$$

where $0 \leq \Delta T < 1$ and terms of the order $(\Delta T)^{r+1}$ and higher are considered negligible. If the functions $g^{(i)}(t)$ satisfy appropriate bounds, i.e., $|g^{(i)}(t)| \leq g_i$, $i = 0, ..., r$, $g(t+\Delta T)$ is bounded in the neighborhood of t in the following way:

$$|g(t + \Delta T)| \leq g_0 + g_1\Delta T + ... + g_r\frac{(\Delta T)^r}{r!} \tag{21}$$

As a result, given the original bound c for $g(t)$ and a choice for derivative bounds g_i, $i = 1, ..., r$, a modified bound g_0 can be determined

$$g_0 = c - [g_1\Delta T + ... + g_r\frac{(\Delta T)^r}{r!}] \tag{22}$$

with the following interpretation: If $|g(t)| \leq g_0$ and its derivatives are bounded according to $|g^{(i)}(t)| \leq g_i$, $i = 1, ..., r$, then $|g(t + \Delta T)| \leq c$ in the neighborhood of t under the specific assumptions on terms of order $(\Delta T)^{r+1}$ and higher.

Using this reasoning, for the system described in Eq. (19), $N + 2$ equally spaced collocation points t_0, t_1, ..., t_N, t_f are chosen such that $t_0 < t_1 < t_2 < ... < t_N < t_f$ and $0 \leq \Delta T < 1$ is the spacing between the collocation points. At a collocation point, for every constraint, a bound on its r derivatives is specified and an appropriate bound on the constraint is computed using Eq. (22). For the l inequality constraints of Eq. (19), on bounding their r derivatives at $N + 2$ collocation

points, results in a total of $(N+2)(r+1)l$ linear inequalities on the km elements of the mode coefficients a_{11},...,a_{mn_m}. In general, these linear inequalities enclose a convex polyhedral region in the coefficient space which represent the feasible trajectories of the dynamic system within the characterization of Eq. (18).

4.1.2 Conservative Scheme

In this scheme, the elements of the vector $\Phi_0(t)$ and the mode functions $\phi_j(t)$ are chosen prudently to come up with inequalities on the elements of a_{11},...,a_{mn_m} that satisfy Eq. (19) at every point in the domain of interest. Along these lines, it may be worth noting that polynomial functions 1, t, t^2,...,t^k possess the property that they are positive in the domain $0 \leq t_0 < t \leq t_f$ and their derivatives are closed within the set. Some special linear combinations of polynomial functions, e.g., **C**hebyshev's polynomials, also share this property.

In order to get an intuitive understanding of the method, consider a single linear inequality $g(t) \leq c$. Let the function $g(t)$ be parameterized using suitable basis functions so that the constraint is written in the following form,

$$g(t) - c = \alpha_1\gamma_1(t) + \alpha_2\gamma_2(t) + ... + \alpha_k\gamma_k(t) \leq 0 \tag{23}$$

where α_i are linear functions of the parameters used to express $g(t)$ and $\gamma_i(t)$ are functions of time. Now, if each $\gamma_i(t)$ possessed the property that it is non-negative over $[t_0, t_f]$, a sufficient condition for the inequality (23) to hold in the domain of interest is that the coefficients $\alpha_i \leq 0$. It must be noted that the basis functions that are used to parameterize $g(t)$ need not be the same as $\gamma_i(t)$. For example, one could use **C**hebyshev's polynomials to express $g(t)$ while select $\gamma_i(t)$ to be the simple polynomials 1, t, t^2,...,t^k.

Using this reasoning, with proper choice of $\Phi_0(t)$ and $\phi_j(t)$, the component equations of (19) are written in the form of Eq. (23). Then, appropriate linear inequalities are imposed on the elements of a_{11},...,a_{mn_m} to guarantee satisfaction of the constraints everywhere in the domain. These linear inequalities again enclose a convex polyhedral region in the coefficient space which represent the feasible trajectories of the dynamic system.

4.1.3 Min-Max Scheme

In this scheme, a set of sufficient conditions are obtained such that Eqs. (19) are satisfied at every point. In order to illustrate this idea, consider one such inequality after the solution form has

been imposed

$$c_0(t) + c_1(t)a_1 + c_2(t)a_2 + \ldots + c_k(t)a_k \ \leq \ 0, \quad t \in [0, t_f]. \tag{24}$$

where $c_j(t)$ are time-varying coefficients over $t \in [t_0, t_f]$ and a_k are the mode coefficients. Each c_j has a minimum and maximum over $[t_0, t_f]$. Using these extrema, one could replace Eq. (24) by a single sufficient conditions that is valid over each set of possible coefficient sign distributions, i.e., when each a_k is less than or greater than zero. In general, this procedure could be repeated for every constraint until a set of sufficient conditions is found for the entire problem

4.2 Linear Inequalities and Convex Sets

From the three schemes described, it is clear that finding a solution of Eq. (19) is equivalent to characterizing the convex feasible region bounded by I linear inequalities on the elements of a_{11},...,a_{mn_m} in $\mathcal{R}^{mk}$. In general, the forms of the inequalities arising from the different schemes of Section 4.1 will be different, however, the inequalities will still be linear. This section briefly outlines the computational aspects of determining the feasible region of a set of inequalities ([22], [5]).

Every constraint partitions $\mathcal{R}^{mk}$ in 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 (i) empty, if the inequalities are inconsistent, (ii) a bounded polytope, or (iii) an unbounded convex region. These possibilities are illustrated for a 2-dimensional space in Figure 5. It is implicitly assumed in this section that the system of inequalities is consistent, i.e., the convex set $\mathcal{K}$ is non-empty. In order to mathematically characterize the feasible region $\mathcal{K}$, one needs to define some terminologies and mathematical procedures.

In general, a system of linear inequalities is non homogeneous, i.e., contains constant terms. We label such a system as $\mathcal{N}(\mathcal{I})$. Correspondingly, one can define a system of homogeneous inequalities, i.e., all constant terms are set to zero, labeled as $\mathcal{H}(\mathcal{I})$. Also, one defines $\mathcal{N}(\mathcal{E})$ corresponding to $\mathcal{N}(\mathcal{I})$ and $\mathcal{H}(\mathcal{E})$ corresponding to $\mathcal{H}(\mathcal{I})$, where all inequalities are replaced 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 called *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 which are not normal. A normal system of inequalities

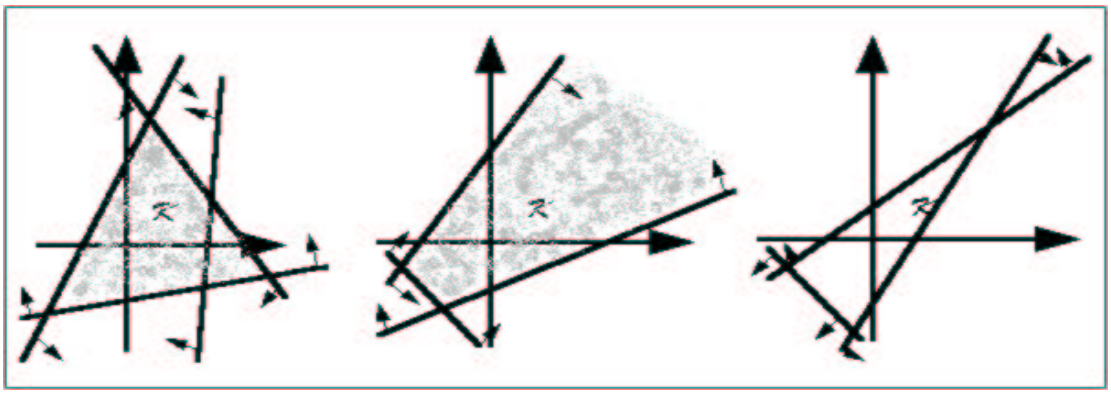

Figure 5: **C**onvex sets $\mathcal{K}$ in two variables with (i) consistent feasible region, (ii) unbounded feasible region, (iii) inconsistent feasible region.

$\mathcal{N}(\mathcal{I})$ possesses *vertices*, which are defined as points of $\mathcal{K}$ which are not interior points of any line segment lying entirely within $\mathcal{K}$.

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

$$\langle P_1, P_2, ..., P_H \rangle = s_1 P_1 + s_2 P_2 + ... + s_H P_H$$
$$s1,\ s2,\ ..., s_l \geq 0 \quad \& \quad s_1 + s_2 + ... + s_l = 1 \tag{25}$$

In order 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,...,N_C. The convex cone is defined as $(N_1, ..., N_C)$, where

$$(N_1, N_2, ..., N_C) = r_1 N_1 + r_2 N_2 + ... + r_C N_C$$
$$r1,\ r2,\ ..., r_C \geq 0 \tag{26}$$

Proposition 1: If the system of linear inequalities $\mathcal{N}(\mathcal{I})$ is normal and the convex hull of its vertices, as defined in Eq. (8), is $\langle P_1, P_2, ..., P_H \rangle$ and the convex cone, as defined in Eq. (26), is $(N_1, N_2, ..., N_C)$,

then the domain $\mathcal{K}$ has the following analytical form

$$\mathcal{K} = \langle P_1, P_2, ..., P_H \rangle + (N_1, N_2, ..., N_C) \tag{27}$$

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

Proposition 2: If the system of linear inequalities $\mathcal{N}(\mathcal{I})$ is not normal, then

$$\mathcal{K} = \mathcal{K}_r + \mathcal{L} \tag{28}$$

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

4.3 Real-time Computations

For the solution to be real-time, the trajectory must be determined at a rate faster than the response time of the system. In general, boundary conditions at the final time are either supplied by the user or determined by on-board sensors. Similarly, the current conditions are known from sensors. It is assumed that the form of the inequalities during trajectory recomputation does not change but the end conditions do. A typical example of this is *pursuit-evasion* problem. From the steps in the previous sections, the trajectory determination problem is reduced to finding a solution of Eq. (19). The schemes of Sections 4.1 further modified the problem to finding intersection of I half spaces characterized by $\mathcal{N}(\mathcal{I})$. From the point of view of determination of a feasible trajectory for the system, any solution of the mode coefficients that satisfies $\mathcal{N}(\mathcal{I})$ is acceptable. However, our approach in this paper is to characterize the entire set of feasible solutions $\mathcal{K}$ enclosed by $\mathcal{N}(\mathcal{I})$ using convexity theory so that the feasible trajectory can be selected in a judicious way.

In general, the inequalities $\mathcal{N}(\mathcal{I})$ arising from the procedures of last section are normal and $I > mk$. From the characterization of $\mathcal{K}$, the main computation steps are to find the vertices P_i of the convex hull and N_i of the convex cone. In order to compute the vertices P_i, one requires solving ${}^{I}C_{mk}$ sets of mk linear equations in mk variables, followed by I 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 $mk-1$ linear equations in $mk-1$ variables, followed by $2I$ inequality checks to determine feasibility with respect to $\mathcal{H}(\mathcal{I})$. These solutions must be done in real-time as end-conditions may change continuously.

A significant computational effort goes into solution of the vertices P_i as end conditions change, since ${}^{I}C_{mk}$ sets of mk linear equations in mk variables are solved. A similar computational burden is faced in the computation of N_i. However, it is important to note a property stated in the following proposition:

Proposition 3: If the only difference in two trajectory computations is a new set of end conditions then the set $\mathcal{N}(\mathcal{I})$ in the two computations differ only in their homogeneous terms. As a consequence, to compute P_i, it is no longer necessary to solve in real-time ${}^{I}C_{mk}$ sets of mk linear equations in mk variables. Instead, the corresponding matrix inverses may be computed off-line and stored. In real-time, a single matrix-vector computation is required to determine a P_i. Also, due to the above mentioned property of $\mathcal{N}(\mathcal{I})$, $\mathcal{H}(\mathcal{I})$ in the two computations are identical. As a result, the computation of N_i needs to be performed only once even when the boundary conditions are changed.

As a consequence of the above observations, the real-time trajectory generation becomes even more attractive as boundary conditions are changed. The computations do not require matrix inversions or solution of simultaneous linear equations in real-time but only matrix-vector multiplications.

5 Examples

The methods proposed in this paper are illustrated by two examples in this section. 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 VTOL aircraft, which is also a simplified model for a V-22 Osprey aircraft.

5.1 Mass-spring-damper system

This example is motivated from an experiment of a rectilinear spring-mass-damper system shown in Figure 6. The system is driven by a single motor acting on mass 1 while the positions and

rates of the three masses are available to the controller through encoders. In order 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 remaining system which now consists of masses 1 and 2. This experiment is designed to demonstrate real-time planning and on-line implementation during situations such as "pursuit-evasion".

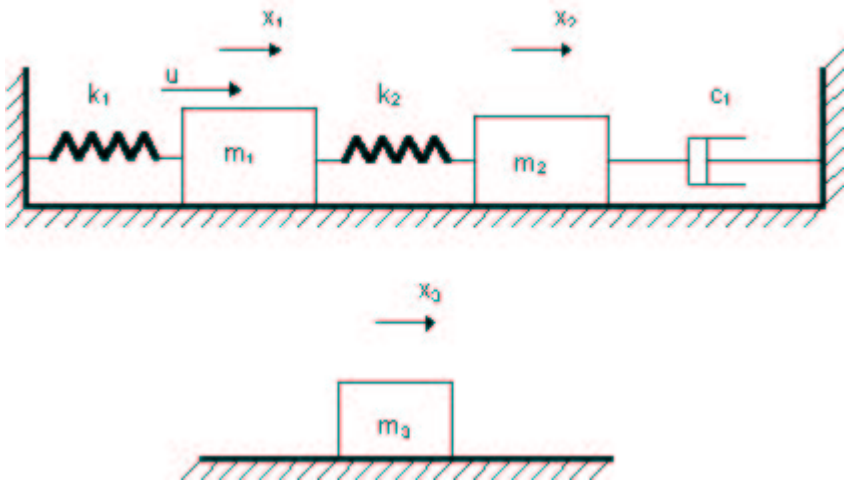

Figure 6: A schematic model of the three mass-spring-damper system.

A schematic of this system is shown in Figure 6. It consists of two degrees-of-freedom and a single input, i.e., $n = 4$ and $m = 1$. The equations of motion for this system are:

$$\begin{aligned} m_1\ddot{x}_1 &= -k_2(x_1 - x_2) - k_1 x_1 + u \\ m_2\ddot{x}_2 &= k_2(x_1 - x_2) - c x_2^{(1)} \end{aligned} \tag{29}$$

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

$$x_2 = y \tag{30}$$

$$x_1 = \frac{m_2}{k_2}\ddot{y} + \frac{c}{k_2}\dot{y} + y \tag{31}$$

$$u = \frac{m_1 m_2}{k_2}\overset{\cdots 4}{y} + \frac{m_1 c_2}{k_2}\overset{\cdots 3}{y} + [m_1 + \frac{m_2(k_1+k_2)}{k_1}]\ddot{y} + \frac{c_2(k_1+k_2)}{k_2}\dot{y} + k_1 y \tag{32}$$

From the physical limitations, the motion of the system must satisfy the following 2-sided constraints: $|x_1| \le x_{1l}$, $|x_2| \le x_{2l}$, and $|u| \le u_l$. On substituting from Eqs. (32), we get

$$|y| \le x_{2l}$$

$$|\tfrac{m_2}{k_2}\ddot{y} + \tfrac{c}{k_2}\dot{y} + y| \leq x_{1l}$$

$$|\tfrac{m_1 m_2}{k_2}\overset{\cdots}{y}^4 + \tfrac{m_1 c_2}{k_2}\overset{\cdots}{y}^3 + [m_1 + \tfrac{m_2(k_1+k_2)}{k_1}]\ddot{y} + \tfrac{c_2(k_1+k_2)}{k_2}\dot{y} + k_1 y| \leq u_l \tag{33}$$

The boundary conditions at t_0 on x_1, $\dot{x}_1$, x_2, $\dot{x}_2$ are known using the encoder readings. At t_f, the boundary conditions on x_2 and $\dot{x}_2$ are obtained from sampled position and rate of slider 3 using a specific heuristic. As a result, four boundary conditions on $\overset{\cdots}{y}^3$, $\ddot{y}$, $\dot{y}$, y at t_0 and two boundary conditions on $\dot{y}$ and y at t_f are known. The statement of the problem is to find a feasible trajectory for the dynamic system that satisfies Eqs. (33) and the appropriate boundary conditions on derivatives of y.

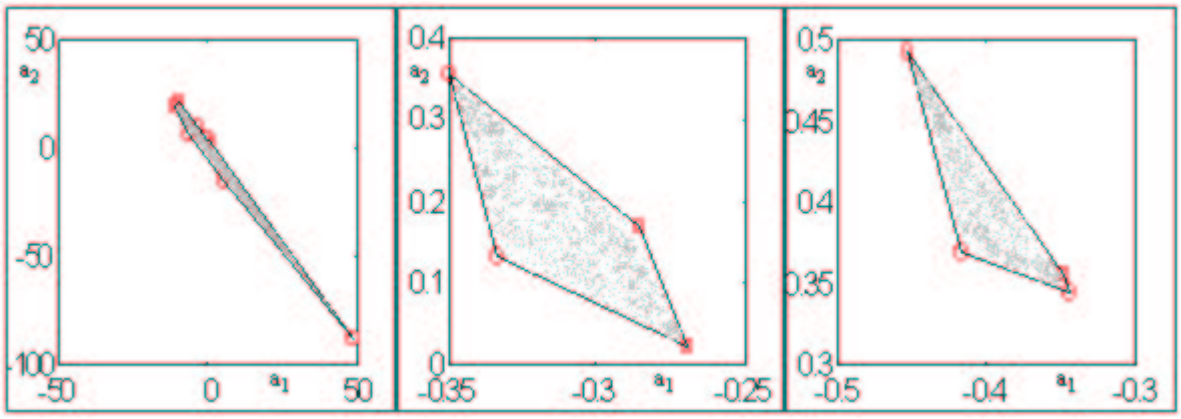

Figure 7: Example 1 feasible regions with collocation scheme: (i) six collocation points, (ii) six collocation points with bounded first derivatives, (iii) six collocation points with bounded first and second derivatives.

First, we want to note that due to the 2-sided nature of the constraints (33), the conservative scheme can not be used. This scheme gives a set of inconsistent inequalities. As a result, the collocation scheme is used to develop the feasible solution selected to have the following form:

$$y(t) = \phi_0(t) + a_1\phi_1(t) + a_2\phi_2(t) \tag{34}$$

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 $\phi_2 = t^5(1-t)^2$, assuming $t_0 = 0$ and $t_f = 1$. The coefficients b_i are determined from the six boundary conditions specified on the problem. Due to the forms of $\phi_1(t)$ and $\phi_2(t)$, $y(t)$ is admissible regardless of the values for a_1 and a_2. **C**onsistent with the physical set up, 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 was implemented in simulation using six equally spaced collocation points.

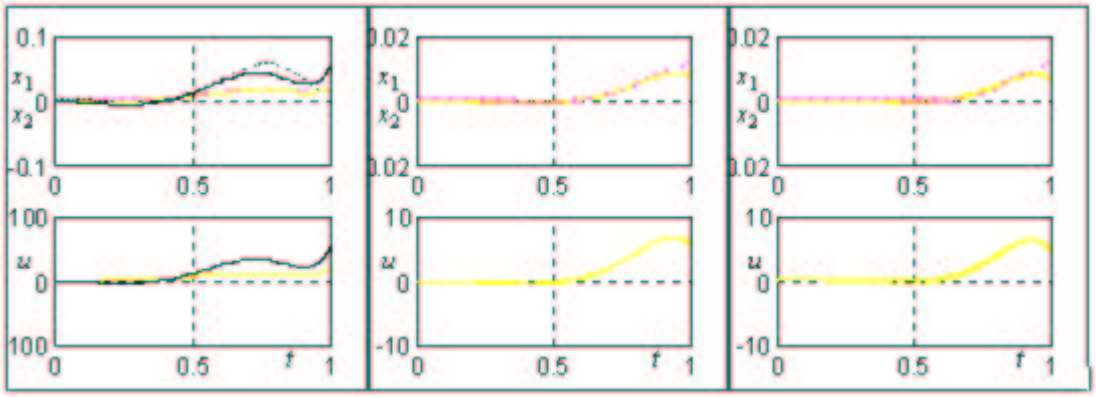

Figure 8: Constraint satisfaction with a vertex point and the centroid of Figure 7: (i) six collocation points, (ii) six collocation points with bounded first derivatives, (iii) six collocation points with bounded first and second derivatives.

Figure 7 shows the feasible regions with the following specifications: (i) constraints are satisfied at the collocation points, (ii) constraints and their first derivatives are bounded at the collocation points, (iii) constraints and their first and second derivatives are bounded at the collocation points. With 6 constraints, 6 collocation points, and two 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 the figure, 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 Figure 8. From these plots, we notice that as higher derivatives of the constraints are bounded, the trajectories satisfy the constraints throughout the domain.

5.1.1 Hardware Implementation

A Pentium 166 Mhz PC running Windows 95 was used to host ECPUSR software and its scripting language compiler. The real-time trajectory generator and tracker were written within 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.

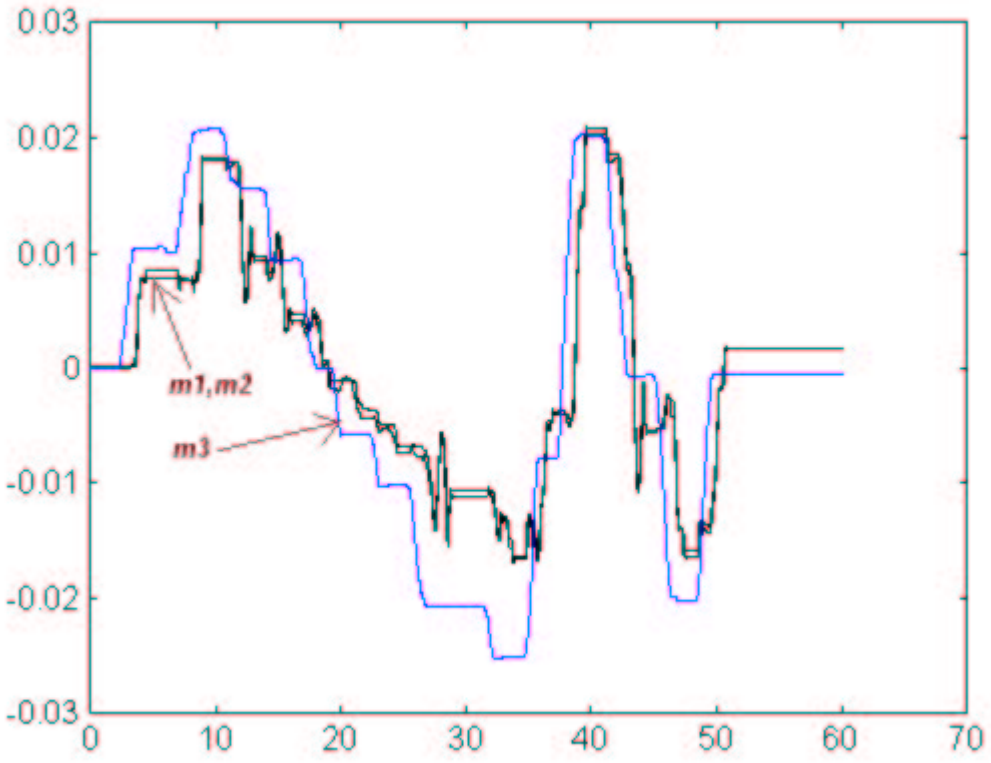

Figure 9: The data of a typical 'pursuit-evasion' experiment where mass 3 is moved around and mass 2 follows mass 3 while satisfying the constraints. The plot overlays the commanded motion of mass 3 and followed motion of mass 2 and mass 1.

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 milliseconds and the data is implemented over the next 0.98 seconds.

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

$$u(t) = \beta_4 y_d^{(4)}(t) + \beta_3 y_d^{(3)}(t) + \beta_2 y_d^{(2)}(t) + \beta_1 y_d^{(1)}(t) + \beta_0 y_d(t) + \gamma_3 e^{(3)}(t) + \gamma_2 e^{(2)}(t) + \gamma_1 e^{(1)}(t) + \gamma_0 e(t) \qquad (35)$$

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

following equation

$$e^{(4)} + \frac{\beta_3 + \gamma_3}{\beta_4} e^{(3)}(t) + \frac{\beta_2 + \gamma_2}{\beta_4} e^{(2)}(t) + \frac{\beta_1 + \gamma_1}{\beta_4} e^{(1)}(t) + \frac{\beta_0 + \gamma_0}{\beta_4} e(t) = 0 \tag{36}$$

from any initial conditions approaches zero with a desired transient dynamics. Figure 5.1.1 shows the typical 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.

5.2 V-22 Osprey aircraft

The V-22 Osprey is a tiltrotor aircraft designed and built jointly by Boeing and Bell Helicopters. This new kind of aircraft features a modification to conventional thrust mechanisms. It is fitted with tilt-rotors making it unique in that it takes off and lands like a helicopter, but once airborne converts to a turboprop airplane capable of high-speed, high-altitude flight. The operation of the tilt-rotors in take-off, landing, and forward propulsion is best explained with the aid of a diagram. Figure 10 depicts the tiltrotor action.

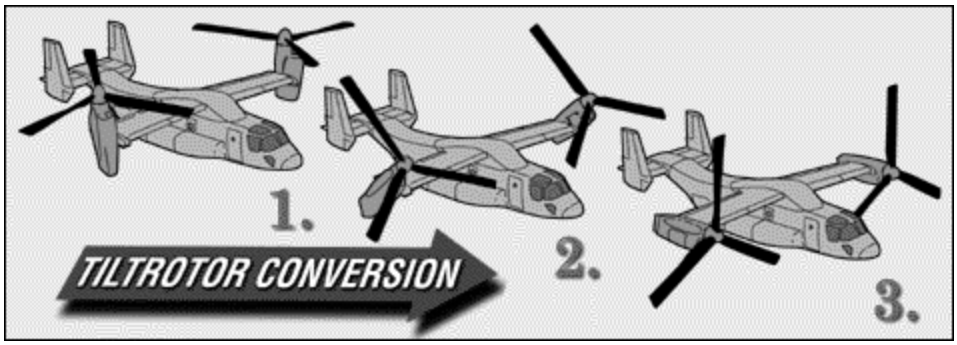

Figure 10: Tilt-rotor operation. Picture courtesy, The Boeing Company. http://www.boeing.com

In the present work we are interested in VTOL operation of the V-22. This includes the flight phase from take-off to the point where the desired altitude is acheived and the rotors are horizontal.

5.2.1 V22-Osprey modelling

For the purposes of this work a simplified representation of the Osprey is formulated. The first simplifying assumption is restriction of motion to the vertical plane. To determine the dynamic

parameters of the V-22 physical data available from Boeing is used. Based on these dimensions the center of mass was found to be located at $x = 0.435\ m, y = 0\ m$ from the geometrical center of the fuselage. Assumming a mass of $17000\ kg$, the inertia about the center of mass was found to be $256388.6\ kgm^2$. To dynamically model the aircraft Figure 11 is used.

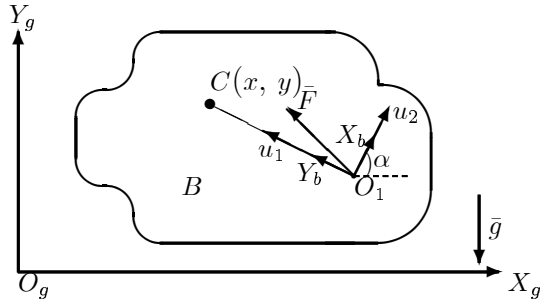

Figure 11: V-22 modeled as a planar body with two thrusters

Figure 11 shows the aircraft modelled as a planar body B with a single thruster F. The position of the center 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 $\mathbf{C}$ 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, a distance l from C, $\bar{Y_b}$ along O_bC, and $\alpha = \angle(\bar{X_g}\bar{X_b})$. The orientation of the thruster is independently controller by the angle $\theta = \angle(\bar{X_b}\bar{F})$. The thruster force F is resolved into its two independent components $u_1,\ u_2$, aligned along axis Y_b and X_b respectively. $u_2 = F\cos\theta$ and $u_1 = F\sin\theta$. The equations of motion of body B are:

$$\begin{aligned} m\ddot{x} &= -u_1\sin\alpha + u_2\cos\alpha, \\ m\ddot{y} &= u_1\cos\alpha + u_2\sin\alpha - mg, \\ J\ddot{\alpha} &= lu_2. \end{aligned} \tag{37}$$

5.2.2 Problem Description

In a typical flight planning problem, the aircraft is to take off from a position and reach a predetermined altitude, while staying within its operating limits and maintaining a clear flight trajectory (i.e. avoiding any obstacles). In order to produce planned motions, all forces applied to the body, must be consistent with the dynamic equations. Only then can we be certain that the system will move as desired, let alone reach prescribed points. The system dynamic equations represented in Eq. (37) represents a constraint that must be satisfied at all times.

Besides system equation constraints, physical and operating constraints are also present. To better explain these depict a typical flight planning situation in Figure 12

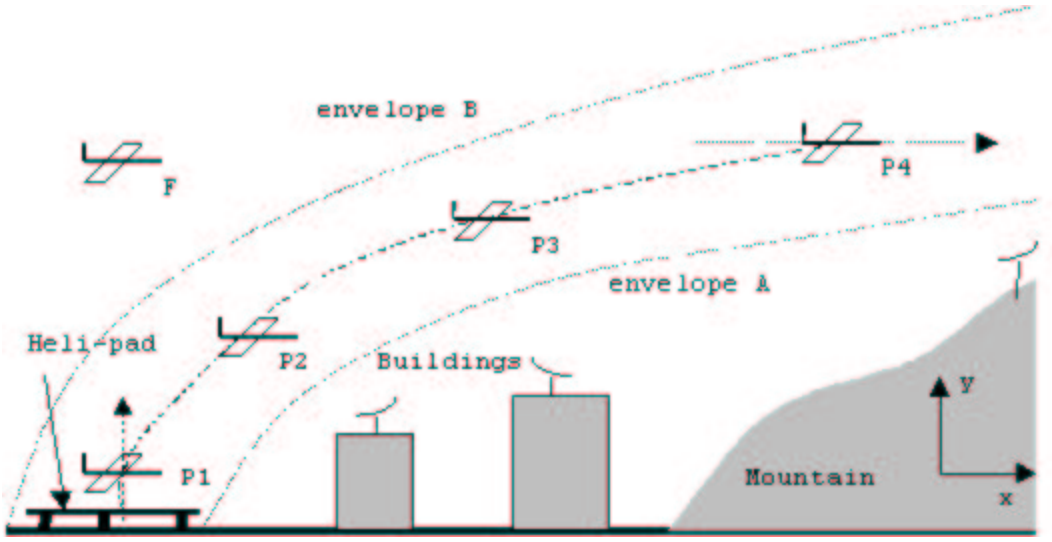

Figure 12: A typical flight planning situation.

Depicted is an airplane in take-off from a postion P1 on a helipad. The airplane is to reach a position P4 in a prescribed final time. P2, P3 are feasible intermediate points. The space is cluttered. Physical obstacle like buildings and a mountain are to be avoided. The airplane must also maintain an approximate flight lane to avoid intrusion in paths of other aircraft. Envelope A is the lower bound of the flight lane, avoiding physical obstacles. Envelope B is the upper limit of the flight lane. The airplane must at all times in flight lie within the physical limits of the flight lanes. Denoting the center of mass of the aircraft as x, y, the entire flight enevelope is mathematically modelled as follows:

$$\begin{array}{rcccl} \text{Envelope B} & y & \leq & 1500 \sin\frac{\pi x}{2200} & \\ \text{Envelope A} & 750 \sin\frac{\pi(x-100)}{2500} & \leq & y & \\ \text{Physical limits on } x & 0 & \leq & x & \leq 1000 \\ \text{Physical limits on } y & 0 & \leq & y & \end{array} \tag{38}$$

The final time is assumed to be 1000 seconds. The airplane will also have operating constraints on maximum rates of accelerations and thrust produced. The vertical velocity is a limited quantity

and will have a maximum limit. We choose to retain this constraint as representative of thrust limitations. From technical specifications on the V-22 available from Boeing we have

$$\dot{y} \;\leq\; 10\text{ms}^{-1} \tag{39}$$

Initial and final conditions are also specified

$$x(0) = 50, \quad y(0) = 1, \quad \dot{x}(0) = 0, \quad \dot{y}(0) = 0, \quad \ddot{x}(0) = 0 \tag{40}$$

$$x(t_f) = 850, \quad y(t_f) = 850, \quad \dot{x}(t_f) = 0, \quad \dot{y}(t_f) = 0, \quad \ddot{y}(t_f) = 0 \tag{41}$$

These conditions account for the fact that during take-off the tilt-rotors are directed vertically and there is no component of horizontal acceleration, while at the final point, the tilt-rotors are directed horizontally and there is no component of vertical acceleration.

In the present form the constraint equations are nonlinear and will not be solvable in real-time for solution. The constraint structure needs to simplified, and inorder to do so, first the system equations will be simplified, followed by a simplification of the position, rate and obstacle constraints.

5.2.3 Differentially flat form

Defining $v_1 \equiv \frac{u_1}{m}$, $v_2 \equiv \frac{u_2}{m}$, $\lambda \equiv \frac{ml}{J}$, a simplified version of the equations of motion are obtained:

$$\begin{aligned} \ddot{x} &= -v_1 \sin\alpha + v_2 \cos\alpha, \\ \ddot{y} &= v_1 \cos\alpha + v_2 \sin\alpha - g, \\ \ddot{\alpha} &= \lambda u_2. \end{aligned} \tag{42}$$

Now defining two functions x_f and y_f as in [?]:

$$x_f = x - \frac{1}{\lambda}\sin\alpha, \qquad y_f = y + \frac{1}{\lambda}\cos\alpha, \tag{43}$$

it is possible to express all states and inputs in the system of Eq. (42), in terms of x_f and y_f and their derivatives (i.e. in a differentially flat form). Greater details are available in [8] and also available "on-line" at http://mechsys4.me.udel.edu/publications.html.

5.2.4 State, rate and obstacle constraints

Using the transformations in Eq. (43), the constraints in Eq. (44) are transformed to the flat space. A closer examination of these transformed constraints will reveal that the nonlinear terms

are much smaller than the linear terms when considered over the domain of interest. For instance $\frac{1}{\lambda} \approx 20$, and $0 \leq x \leq 1000$, $0 \leq y \leq 1000$. Ignoring the nonlinear terms in the expressions results in a simplification which is both useful and justified for the present problem. Therefore we have

$$x_f \approx x, \quad y_f \approx y, \quad \dot{x}_f \approx \dot{x}, \quad \dot{y}_f \approx \dot{y}$$

and the nonlinear constraints are as

$$\begin{array}{rcl} y_f & \leq & 1500 \ \sin\frac{\pi x_f}{2200} \\ 750 \ \sin\frac{\pi(x_f-100)}{2500} & \leq & y_f \\ 0 & \leq & x_f \qquad \leq \ 1000 \\ 0 & \leq & y_f \\ \dot{y}_f & \leq & 10 \end{array} \tag{44}$$

Using the polytopic approximation scheme the constraints are replaced by linear inequality constraints. Figure 13 depicts the original nonlinear feasible region, followed by the polytopic approximation within the region The resulting 5-gon is represented by 5 linear inequalities

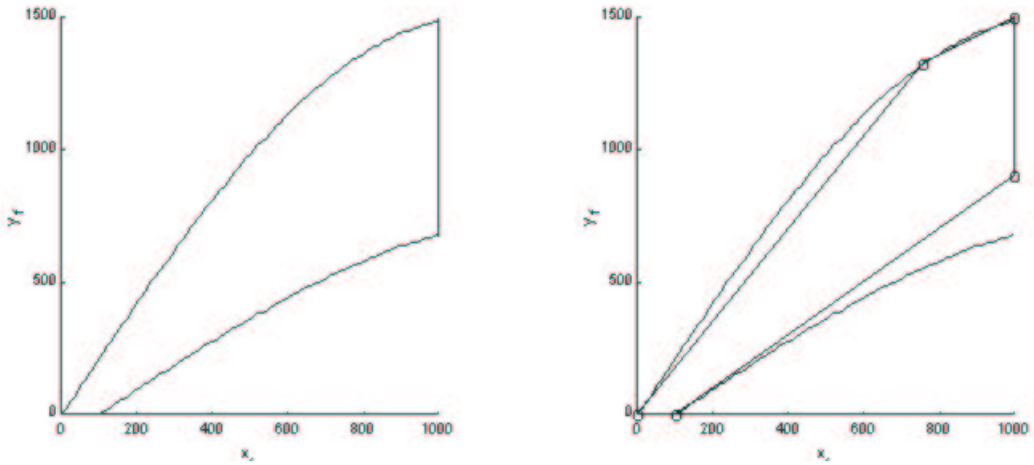

Figure 13: Feasible region followed by polytopic approximation.

$$0 \leq y_f, \ x_f \leq 1000, \ 0.01 \ x_f - 0.01 \ y_f \leq 1, \ -0.000928 \ x_f + 0.001285 \ y_f \leq 1, \ y_f \leq 1.7457 \ x_f. \tag{45}$$

5.2.5 Trajectory Generation

The linear inequalities in Eq. (45) together with limits on rates, are satisfied in trajectory generation using the collocation scheme. Here a solution form is chosen for the flat outputs in weigthed residual form as:

$$x_f = \Phi_{10} + a_{11}\Phi_{11} + a_{12}\Phi_{12}, \quad y_f \ = \ \Phi_{20} + a_{21}\Phi_{21} + a_{22}\Phi_{22}, \tag{46}$$

where Φ_{10}, Φ_{20} are termed the initial solution and chosen to satisfy the boundary conditions in Eq. (44), Φ_{11}, Φ_{12}, and Φ_{21}, Φ_{22}, termed the basis functions, are chosen to be homogenous at the boundaries (i.e. they have no contribution to any of the boundary terms). The coefficients a_{11}, a_{12}, a_{21}, a_{22} are chosen to satisfy the inequality constraints. In the present example polynomials in t are chosen for the initial solution and basis functions

$$\begin{aligned}
\Phi_{10} &= a_0 + a_1 t + a_2 t(t_f - t) + a_3 t^2(t_f - t) + a_4 t^2(t_f - t)^2 \\
\Phi_{12} &= b_0 + b_1 t + b_2 t(t_f - t) + b_3 t^2(t_f - t) + b_4 t^2(t_f - t)^2 \\
\Phi_{11} &= t^3(t_f - t)^2, \quad \Phi_{21} = t^2(t_f - t)^3, \quad \Phi_{12} = \Phi_{22} = t^3(t_f - t)^3
\end{aligned} \tag{47}$$

Using the collocation method with 5 collocation points chosen evenly over $t \in [0, 1000]$, linear inequality constraints in the coefficient variables a_{11}, a_{12}, a_{21}, a_{22} are obtained. The resulting inequalities are then solved simulataneously to determine all solutions. The solutions form the vertices of a polytope in a_{11}, a_{12}, a_{21}, a_{22} space. The ceontroid of the polytope is chosen, as the final value for the coefficients. In the present problem 63 vertices were obtained, the solution being evaluated in under 2 seconds. Note also that we choose to determine all polytope vertices, and then evaluate the centroid, as a final values. Strictly, any one polytope vertice is also a feasible solution, and had we chosen to accept the first generated polytope vertice as our final solution, evaluation time is vastly reduced. The solution was implemented in a high-level package as is MATLAB. **C**hoosing instead to implement a solution in **C** language or FORTRAN, a greater savings in execution times are expected. It has been observed in such implementations, that a 1000 fold savings in computation time is acheived. Figure 14 depict the feasible trajectories generated.

6 Conclusion

This paper presented a methodology to generate feasible trajectories of linear and nonlinear dynamic systems which admit higher-order (or flat) forms in the presence of inequality constraints. The methodology consists of three steps: (i) inner approximate the constraint set in the higher-order space by linear inequalities; (ii) choose an admissible form of the solution with a set of basis function and through collocation transform the semi-infinite problem into a problem with finite number of inequality constraints; (iii) identify a feasible polytope in the coefficient space of the basis functions such that all points within it are feasible solutions. The inner approximation of

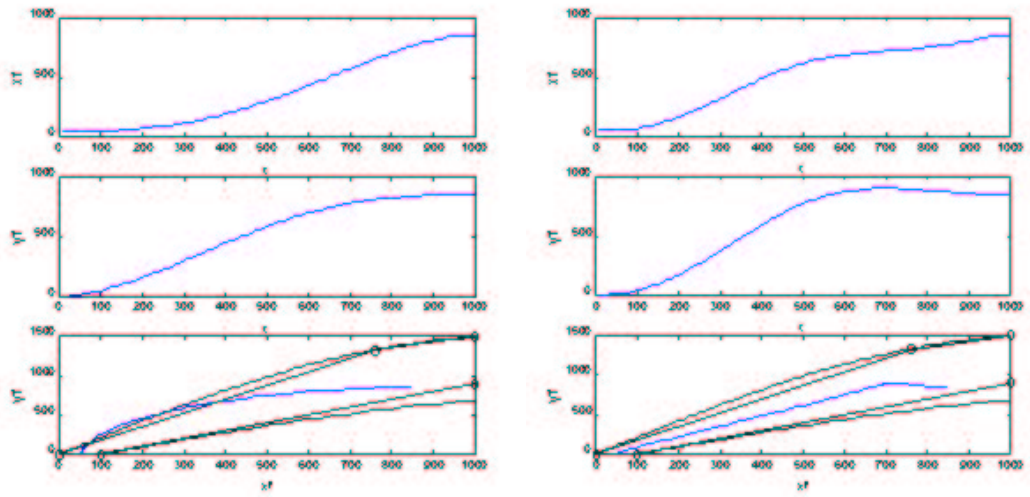

Figure 14: Figure on left depicts the initial trajectories, generated using only the initial conditions. Notice that a constraint violation occurs in the period of 50 to 250 seconds (in x_f-y_f plot). After running the problem through the collocation scheme an improved trajectory satisfying the constraints at all points is achieved. This is shown in the figure on the right.

the nonlinear constraints by linear constraints can be done once off-line assuming that the set of inequalities do not change during motion. The feasible trajectories within the linear inequalities can be determined in real-time because of the need to solve only simultaneous linear equations. The method was successfully implemented in hardware to simulate a 'pursuit-evasion' maneuver with a linear spring-mass-damper system. Also, simulation results were provided for a VTOL aircraft. We believe that this methodology can be extended for real-time planning of nonlinear systems such as air vehicles and missiles.

7 Acknowledgment

The support of National Science Foundation Presidential Faculty Fellowship '94 to the second author is gratefully acknowledged. This work was also partially supported by Air Force during sabbatical leave of the second author at Caltech. The authors also acknowledge helpful discussions with Dr. Maximilian Schlemmer.

References

[1] 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, Transactions of ASME*, Vol. 118, No. 4, 1996, pp. 786-791.

[2] Agrawal, S. K., and Xu, X., "A New Higher-Order Method for Optimization of a Class of Linear Time-Varying Dynamic Systems", *Journal of Vibrations and Control* Vol. 3, No. 4., 1997, pp. 379-396.

[3] Agrawal, S. K., and Faiz, N., 1998, "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, 11-28.

[4] F. Allgöwer and Doyle, F.J. III, "Nonlinear process control - which way to the promised land?", *Proc. Chemical Process Control V*, Lake Tahoe, CA, 1996.

[5] Boyd, S. and Vandenberghe, L., *Convex Optimization: EE364 Lecture Notes*, Stanford University, 1996.

[6] *Manual for Model 210/210a Rectilinear Control System*, ECP Educational Products Inc, 1996.

[7] Faiz, N. and Agrawal, S. K., "Optimal Control of 2-Input Chained Systems using Higher-Order Method", *1998 American Control Conference*, Philadelphia, pp. 6-7, 1998.

[8] Faiz, N., *Real-time and optimal trajectory generation for nonlinear systems*, Doctoral Dissertation, Department of Mechanical Engineering, University of Delaware, 1999.

[9] Fletcher, C. A. J., *Computational Galerkin Methods*, Springer Verlag, 1984.

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

[11] Gill, P.E., Murray,W., Saunders, M.A., Wright,M.H.: *User's Guide for NPSOL (Version 4.0): A Fortran package for Nonlinear Programming*, Stanford Business Software Inc., Mountain View, CA. Hauser, J. and Meyer, D. G., "Trajectory morphing for nonlinear systems", *1998 American Control Conference*, Philadelphia, pp. 2065-2070, 1998.

[12] Hettich, R., Kortanek, K., "Semi-infinite Programming: Theory, methods and applications", *SIAM Review*, Vol. 35, No. 3, 1993, pp. 380-429.

[13] Horst, R., "On the Convexification of Nonconvex Programming Problems", *European Journal of Operational Research*, Vol. 15, 1984, pp. 382-392.

[14] Isidori, A., *Nonlinear Control Systems*, Springer-Verlag, 3rd edition, 1995.

[15] Latombe, J. C., *Robot Motion Planning*, Kluwer Academic Publishers, Boston, 1991.

[16] Martin, P., Devasia, S., and Paden, B., "A different look at output tracking: control of a VTOL aircraft", *Automatica*, Vol. 32, No. 1, pp. 101-107, 1996.

[17] *MATLAB 5.1*, The Math Works Inc., Natick, MA.

[18] Murray, R. M., Rathinam, M., and Sluis, W., "Differential flatness of mechanical control systems: a catalog of prototype systems", *Proceedings of the 1995 ASME International Mechanical Engineering Congress and Exposition*, Vol. 57-1, 1995.

[19] Nieuwstadt, M. J., Murray, R. M., "Approximate Trajectory Generation for Differentially Flat Systems with Zero Dynamics", *34th IEEE Conference on Decision and Control*, 1996.

[20] Nieuwstadt, M. J. and Murray, R. M., "Real-time trajectory generation for differentially flat systems", *International Journal of Robust and Nonlinear Control*, Vol. 8, No. 11, pp. 995-1020, 1998.

[21] Shin, K. G. and McKay, N. D., "Minimum time control of robotic manipulators with geometric constraints", *IEEE Transactions on Automatic Control*, Vol. 30, No. 6, pp. 531-541, 1985.

[22] Solodovnikov, A. S., *System of Linear Inequalities*, University of Chicago press, Chicago, 1980.

[23] Ilic'-Spong, M. I., Marino, R., Peresada, S. M., and Taylor, D. G., "Feedback linearizing control of switched relucatnce motors", *IEEE Transactions on Automatic Control*, Vol. 32, No. 5, 1987, pp. 371-379.

[24] Spong, M. W., Vidyasagar M., *Robot dynamics and control*, John Wiley and Sons, 1986.

[25] Zhan, W., Tarn, T. J., Isidori, A., "A canonical dynamic extension algorithm for noniteraction with stability for affine nonlinear systems", *System and Control Letters*, Vol. 17, 1991, pp. 177-184.