INVERSION BASED CONSTRAINED TRAJECTORY OPTIMIZATION

Nicolas Petit, Mark B. Milam, Richard M. Murray

Control and Dynamical Systems
Mail Code 107-81
California Institute of Technology
Pasadena, CA 91125.
{npetit,milam,murray}@cds.caltech.edu

Abstract: A computationally efficient technique for the numerical solution of optimal control problems is discussed. This method utilizes tools from nonlinear control theory to transform the optimal control problem to a new, lower dimensional set of coordinates. It is hypothesized that maximizing the relative degree in this transformation is directly related to minimizing the computation time. Firm evidence of this hypothesis is given by numerical experiments. Results are presented using the Nonlinear Trajectory Generation (NTG) software package.

Keywords: Real-time optimization, optimal control, inversion, nonlinear optimization

1. INTRODUCTION

Computationally efficient trajectory optimization is an enabling technology for many new facets of engineering. Formation flying of satellites, see (TechSat 21 n.d.), and trajectory generation of unmanned aerial vehicles, see (Sweetman 1997), are two examples where the tools of real-time trajectory optimization would be extremely useful. In (Milam *et al.* 2000), a new technique is presented to solve such problems. The technique is based on mapping the system dynamics, objective, and constraints to a lower dimensional space. An optimization problem is solved in the lower dimensional space, and then the optimal states and inputs are recovered from the inverse mapping. However, no concrete criterion was provided to determine the appropriate coordinate change. In this paper, we address the case of single-input single-output systems and show how the role of the relative degree of the system affects real-time trajectory generation optimization problems. The relative degree of the system will directly relate to the amount of inversion used in the optimization problem. The computational implications of inversion are investigated. By example, we conclude that more inversion significantly increases the speed of execution with no loss in the rate of convergence.

For either open-loop reference trajectory design or receding horizon techniques, this example illustrates that the choice of adequate variables for representing a system and its dynamics is crucial in the context of implementation of real-time trajectory generation.

Section 2 describes the class of optimization problems under consideration. Two classical methods for solving optimization problems are presented and compared to the new technique. Section 3 provides numerical investigations for an example that exhibits the explicit relation between relative degree and computation time for a single-input single-output system. A summary and future directions for multi-input multi-output systems are provided in the final section.

2. PROBLEM FORMULATION AND PROPOSED METHOD OF SOLUTION

2.1 *Optimal Control Problem*

Consider the nonlinear control system

$$\begin{aligned} &\dot{x} = f(x) + g(x)u, \\ &\mathbb{R} \ni t \mapsto x \in \mathbb{R}^n, \mathbb{R} \ni t \mapsto u \in \mathbb{R} \end{aligned} \tag{1}$$

where all vector fields and functions are real-analytic. It is desired to find a trajectory of (1), i.e. $[t_0, t_f] \ni t \mapsto (x,u)(t) \in \mathbb{R}^{n+1}$, that minimizes the performance index

$$\begin{aligned} J(x,u) =& \phi_f(x(t_f), u(t_f)) + \phi_0(x(t_0), u(t_0)) \\ &+ \int_{t_0}^{t_f} L(x(t), u(t))dt, \end{aligned}$$

[1] Work supported in part by the Institut National de Recherche en Informatique et en Automatique (INRIA), AFOSR grant F49620-99-1-0190 and DARPA contract F33615-98-C-3613.

where L is a nonlinear function, subject to a vector of initial, final, and trajectory constraints

$$\begin{aligned} lb_0 &\leq \psi_0(x(t_0), u(t_0)) \leq ub_0, \\ lb_f &\leq \psi_f(x(t_f), u(t_f)) \leq ub_f, \\ lb_t &\leq S(x,u) \leq ub_t, \end{aligned} \tag{2}$$

respectively. For conciseness, we will refer to this optimal control problem as

$$\begin{cases} \min\limits_{(x,u)} J(x,u) \\ \text{subject to} \\ \dot{x} = f(x) + g(x)u, \\ lb \leq c(x,u) \leq ub. \end{cases} \tag{3}$$

2.2 *Solution Technique*

2.2.1. *Classical collocation* A numerical approach to solving this optimal control problem is to use the direct collocation method outlined in (Hargraves and Paris 1987). The idea behind this approach is to transform the optimal control problem into a nonlinear programming problem. This is accomplished by discretizing time into a grid of $N-1$ intervals

$$t_0 = t_1 < t_2 < \ldots < t_N = t_f \tag{4}$$

and approximating the state x and the control input u as piecewise polynomials $\hat{x}$ and $\hat{u}$, respectively. Typically a cubic polynomial is chosen for the states and a linear polynomial for the control on each interval. Collocation is then used at the midpoint of each interval to satisfy Eq. (1). Let $\hat{x}(x(t_1)^T, ..., x(t_N)^T)$ and $\hat{u}(u(t_1), ..., u(t_N))$ denote the approximations to x and u, respectively, depending on $(x(t_1)^T, ..., x(t_N)^T) \in \mathbb{R}^{nN}$ and $(u(t_1), ..., u(t_N)) \in \mathbb{R}^N$ corresponding to the value of x and u at the grid points. Then one solves the following finite dimension approximation of the original control problem (3)

$$\begin{cases} \min\limits_{y \in \mathbb{R}^M} F(y) = J(\hat{x}(y), \hat{u}(y)) \\ \text{subject to} \\ \dot{\hat{x}} - f(\hat{x}(y), \hat{u}(y)) = 0, \quad lb \leq c(\hat{x}(y), \hat{u}(y)) \leq ub, \\ \qquad \forall t = \dfrac{t_j + t_{j+1}}{2} \quad j = 1, \ldots, N-1 \end{cases} \tag{5}$$

where $y = (x(t_1)^T, u(t_1), \ldots, x(t_N)^T, u(t_N))$, and $M = \dim y = (n+1)N$.

2.2.2. *Inverse dynamic optimization* (Seywald 1994) suggested an improvement to the previous method (see also (Bryson 1999) page 362). Following this work, one first solves a subset of system dynamics in (3) for the the control in terms of combinations of the state and its time derivative. Then one substitutes for the control in the remaining system dynamics and constraints. Next all the time derivatives $\dot{x}_i$ are approximated by the finite difference approximations

$$\dot{\hat{x}}(t_i) = \frac{x(t_{i+1}) - x(t_i)}{t_{i+1} - t_i}$$

to get

$$\left.\begin{aligned} p(\dot{\hat{x}}(t_i), x(t_i)) &= 0 \\ q(\dot{\hat{x}}(t_i), x(t_i)) &\leq 0 \end{aligned}\right\} \quad i = 0, ..., N-1.$$

The optimal control problem is turned into

$$\begin{cases} \min\limits_{y \in \mathbb{R}^M} F(y) \\ \text{subject to} \\ p(\dot{\hat{x}}(t_i), x(t_i)) = 0 \\ q(\dot{\hat{x}}(t_i), x(t_i)) \leq 0 \end{cases} \tag{6}$$

where $y = (x(t_1)^T, \ldots, x(t_N)^T)$, and $M = \dim y = nN$. As with the Hargraves and Paris method, this parameterization of the optimal control problem (3) can be solved using nonlinear programming.

The dimensionality of this discretized problem is lower than the dimensionality of the Hargraves and Paris method, where both the states and the input are the unknowns. This induces substantial improvement in numerical implementation.

2.2.3. *New Numerical Approach* In fact, it is usually possible to reduce the dimension of the problem further. Given an output, it is generally possible to parameterize the control and a part of the state in terms of this output and its time derivatives. In contrast to the previous approach, one must use more than one derivative of this output for this purpose.

When the whole state and the input can be parameterized with one output, one says that the system is flat; see the work of (Fliess *et al.* 1995, Fliess *et al.* 1999). When the parameterization is only partial, the dimension of the subspace spanned by the output and its derivatives is given by r the *relative degree* of this output.

Definition 1. ((Isidori 1989)). *A single input single output system*

$$\begin{cases} \dot{x} = f(x) + g(x)u \\ y = h(x) \end{cases} \tag{7}$$

is said to have relative degree r *at point* x_0 *if* $L_g L_f^k h(x) = 0$, *in a neighborhood of* x_0, *and for all* $k < r-1$ $L_g L_f^{r-1} h(x_0) \neq 0$ *where* $L_f h(x) = \sum_{i=1}^n \frac{\partial h}{\partial x_i} f_i(x)$ *is the derivative of* h *along* f.

Roughly speaking, r is the number of times one has to differentiate y before u appears.

Result 1. ((Isidori 1989)). Suppose the system (7) has relative degree r at x^0. Then $r \leq n$. Set

$$\begin{aligned} \phi_1(x) &= h(x) \\ \phi_2(x) &= L_f h(x) \\ &\vdots \\ \phi_r(x) &= L_f^{r-1} h(x). \end{aligned}$$

If r is strictly less than n, it is always possible to find $n-r$ more functions $\phi_{r+1}(x),...,\phi_n(x)$ such that the mapping

$$\phi(x) = \begin{pmatrix} \phi_1(x) \\ \vdots \\ \phi_n(x) \end{pmatrix}$$

has a Jacobian matrix which is nonsingular at x^0 and therefore qualifies as a local coordinates transformation in a neighborhood of x^0. The value at x^0 of these additional functions can be fixed arbitrarily. Moreover, it is always possible to choose $\phi_{r+1}(x),...,\phi_n(x)$ in such a way that $L_g\phi_i(x) = 0$, for all $r+1 \leq i \leq n$ and all x around x^0.

The implication of this result is that there exists a change of coordinates $x \mapsto z = (z_1, z_2, ..., z_n)$ such that the systems equations may be written as

$$\begin{cases} \dot{z}_1 = z_2 \\ \dot{z}_2 = z_3 \\ \quad \vdots \\ \dot{z}_{r-1} = z_r \\ \dot{z}_r = b(z) + a(z)u \\ \dot{z}_{r+1} = q_{r+1}(z) \\ \quad \vdots \\ \dot{z}_n = q_n(z) \end{cases}$$

where $a(z)$ is nonzero for all z in a neighborhood of $z^0 = \phi(x^0)$. In these new coordinates, any optimal control problem can be solved by a partial collocation, i.e. collocating only $(z_1, z_{r+1}, ..., z_n)$ instead of a full collocation $(z_1, ..., z_r, z_{r+1}, ..., z_n, u)$. Inverting the change of coordinates, the state and the input $(x_1, ..., x_n, u)$ can be expressed in terms of $(z_1, ..., z_1^{(r)}, z_{r+1}, ..., z_n)$. This means that once translated into these new coordinates, the original control problem (3) will involve r successive derivatives of z_1.

It is not realistic to use finite difference approximations as soon as $r > 2$. In this context, it is convenient to represent $(z_1, z_{r+1}, ...z_n)$ as B-splines. B-splines are chosen as basis functions because of their ease of enforcing continuity across knot points and ease of computing their derivatives. A pictorial representation of such an approximation is given in Figure 1. Doing so we get

$$\begin{aligned} z_1 &= \sum_{i=1}^{p_1} B_{i,k_1}(t) C_i^1 \\ z_{r+1} &= \sum_{i=1}^{p_{r+1}} B_{i,k_{r+1}}(t) C_i^{r+1} \\ &\vdots \\ z_n &= \sum_{i=1}^{p_n} B_{i,k_n}(t) C_i^n \\ \text{with } p_j &= l_j(k_j - m_j) + m_j \end{aligned}$$

where $B_{i,k_j}(t)$ is the B-spline basis function defined in (de Boor 1978) for the output z_j with order k_j, C_i^j are the coefficients of the B-spline, l_j is the number of knot intervals, and m_j is number of smoothness conditions at the knots. The set $(z_1, z_{r+1}, ..., z_n)$ is thus represented by $M = \sum_{i\in\{1,r+1,...,n\}} p_i$ coefficients.

In general, w collocation points are chosen uniformly over the time interval $[t_o, t_f]$, (though optimal knots placements or Gaussian points may also be considered). Both dynamics and constraints will be enforced at the collocation points. The problem can be stated as the following nonlinear programming form:

$$\begin{cases} \min\limits_{y\in\mathbb{R}^M} F(y) \\ \text{subject to} \\ \dot{z}_{r+1}(y) - q_{r+1}(z)(y) = 0 \\ \quad \vdots \\ \dot{z}_n(y) - q_n(z)(y) = 0 \text{ for every } w \\ \quad lb \leq c(y) \leq ub \end{cases} \tag{8}$$

where

$$y = (C_1^1, \ldots, C_{p_1}^1, C_1^{r+1}, \ldots, C_{p_{r+1}}^{r+1}, \ldots, C_1^n, \ldots, C_{p_n}^n),$$

and $M = \sum_{i\in\{1,r+1,...,n\}} p_i$. The coefficients of the B-spline basis functions can be found using nonlinear programming.

2.2.4. *Comparisons* Our approach is a generalization of inverse dynamic optimization. Let us summarize the different ways we can write the optimal control problem:

- "Full collocation" solving problem (5) by collocating $(x, u) = (x_1, ..., x_n, u)$ without any attempt of variable elimination. After collocation the dimension of the unknowns space is $\mathcal{O}(n+1)$.
- "Inverse dynamic optimization" solving problem (6) by collocating $x = (x_1, ..., x_n)$. Here the input is eliminated from the equation using one derivative of the state. After collocation the dimension of the unknowns space is $\mathcal{O}(n)$.
- "Flatness parametrization" (Maximal inversion), our approach, solving problem (8) in the new coordinates collocating only $(z_1,\ z_{r+1}, ..., z_n)$. Here we eliminate as many variables as possible and replace them using the first r derivatives of z_1. After collocation, the dimension of the unknowns space is $\mathcal{O}(n-r+1)$.

3. NUMERICAL INVESTIGATIONS

3.1 *The software package NTG*

The software package called Nonlinear Trajectory Generation (NTG) is designed to solve optimal control problems using the new approach presented in this paper, see (Milam *et al.* 2000).

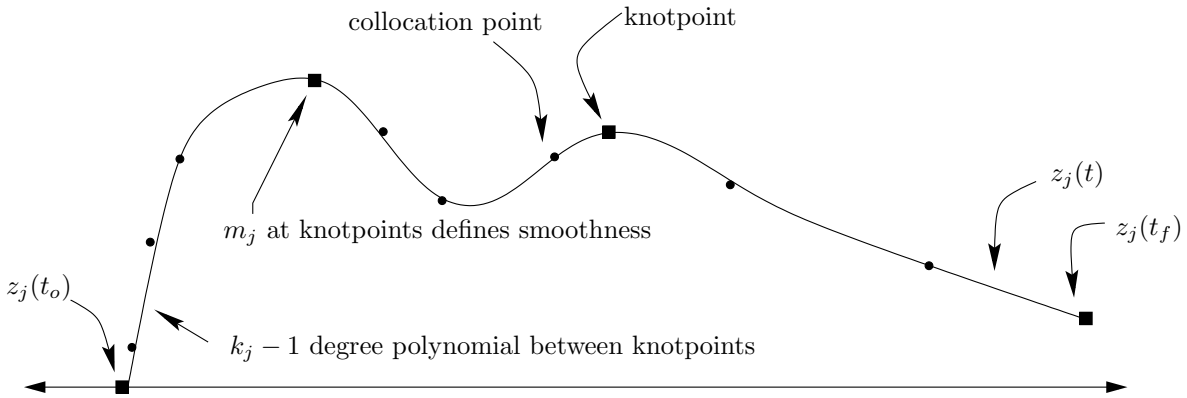

Fig. 1. Spline representation of a variable.

NTG has been developed using the C programming language. The sequential quadratic programming package NPSOL by (Gill *et al.* n.d.) is used as the nonlinear programming solver in NTG. When specifying a problem to NTG, the user is required to state the problem in terms of some choice of outputs and its derivatives. The user is also required to specify the regularity of the variables, the placement of the knot points, the order and regularity of the B-splines, and the collocation points for each output.

3.2 *Example*

The example presented quantitatively illustrates the implications of transforming the optimization problem to the lowest space possible. The system under consideration is an academic example, without any particular physical meaning, chosen to contain various nonlinearities. Without loss of generality, its triangular form is chosen for the purpose of simplicity of the presentation. By considering different outputs with increasing relative degrees we will write different formulations of the optimal control problem. Starting from full collocation, we will end with the flatness parameterization. Finally, runs done with these different approaches will be compared.

3.2.1. *Presentation of the system* We consider the following fifth order single input dynamics

$$\begin{cases} \dot{x}_1 = 5x_2 \\ \dot{x}_2 = \sin x_1 + x_2^2 + 5x_3 \\ \dot{x}_3 = -x_1x_2 + x_3 + 5x_4 \\ \dot{x}_4 = x_1x_2x_3 + x_2x_3 + x_4 + 5x_5 \\ \dot{x}_5 = -x_5 + u \end{cases}$$

and the following optimal control problem: find $[0,1] \ni t \mapsto (x,u)(t)$ that minimizes

$$J = \int_0^1 \left(x_1^2(s) + \frac{1}{100} u^2(s) \right) ds \tag{9}$$

subject to the constraints

$$\begin{aligned} (x_1,x_2,x_3,x_4,x_5,u)(0) &= (0,0,0,0,0,0) \\ (x_1,x_2,x_3,x_4,x_5,u)(1) &= (\pi,0,0,0,0,0) \\ \forall i =\in \{1,2,3,4,5\}, |x_i| &\leq 100 \\ |u| &\leq 100. \end{aligned}$$

To solve this problem by collocation, it is possible to use the three different approaches presented in the previous section

- "Full collocation". One must consider $(x_1, x_2, x_3, x_4, x_5, u)$ as unknowns.
- "Inverse dynamic optimization". For this example, we can solve for u by the following

  $$u = \dot{x}_5 + x_5.$$

  Thus the whole system variables are parameterized by $(x_1, x_2, x_3, x_4, x_5)$.
- "Flatness parameterization" (Maximal Inversion). Consider the variable x_4. Two differentiations give

  $$\begin{aligned} \dot{x}_4 =& x_1x_2x_3 + x_2x_3 + x_4 + 5x_5 \\ \ddot{x}_4 =& 5x_2^2x_3 + (x_1+1)(\sin x_1 + x_2^2 + 5x_3)x_3 \\ &+ (x_1+1)x_2(-x_1x_2 + x_3 + 5x_4) \\ &+ x_1x_2x_3 + x_2x_3 + x_4 + 5x_5 - 5x_5 + 5u. \end{aligned} \tag{10}$$

  The system where x_4 is the output has relative degree 2. By Result 1, it is possible to parameterize all the system by x_4 and 3 more variables. Here we can choose (x_1, x_2, x_3, x_4) for this parameterization.

  It is easy to check that when x_3 is the output the system has relative degree 3. The whole system can be parameterized by (x_1, x_2, x_3).

  Similarly, when x_2 is chosen as the output, the system has relative degree 4 and it is possible to parameterize all its variables by (x_1, x_2).

  At last the system with x_1 as output has relative degree 5. The system is flat, i.e. it is possible to parameterize all its variables by x_1 only. In the optimal control problem one can replace x_2, x_3, x_4, x_5 and u by combinations of $x_1, \dot{x}_1, \ddot{x}_1, x_1^{(3)}, x_1^{(4)}, x_1^{(5)}$.

As a summary the different formulations of the optimal control problem are represented in Figure 2.

Choice of output	Relative degree	Variables for complete parameterization	Differential equation to be satisfied
u	0	$(x_1, x_2, x_3, x_4, x_5, u)$	5
x_5	1	$(x_1, x_2, x_3, x_4, x_5)$	4
x_4	2	(x_1, x_2, x_3, x_4)	3
x_3	3	(x_1, x_2, x_3)	2
x_2	4	(x_1, x_2)	1
x_1	5	(x_1)	0

Fig. 2. The different formulations of the optimal control problem for the example. Top: full collocation. Bottom: flatness parametrization.

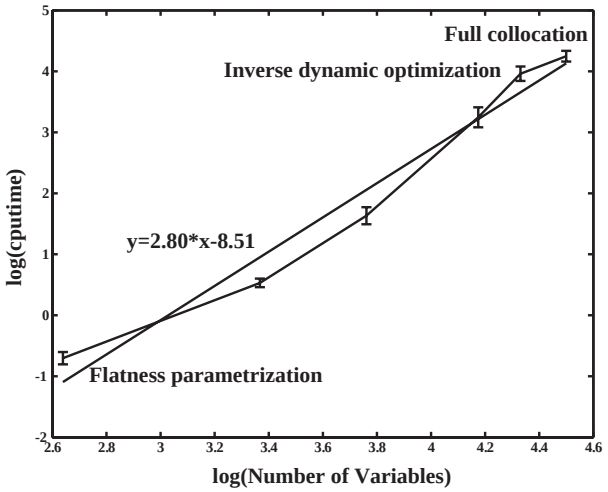

Fig. 4. log(Number of variables) versus log(cpu-time). In each case 200 runs were done with random initial guesses. The *variance* of the results is represented by the error bar. The slope of the linear regression of the mean values of cpu-time is 2.80 .

3.2.2. *Experiments and results* All these formulations of the optimal control problem were coded in C. Symbolic gradients were provided. These problems were solved using the presented NTG package. Every solution was double checked by a numerical integration of the dynamics of the system. We only considered as valid solutions providing the following absolute precision $3.10 \leq x_1(1) \leq 3.20$. This requirement combined with a search for speed of execution induced the choices of the NTG parameters. To perform as fair as possible comparisons we also give in each case the convergence rate, i.e. the percentage of runs ending with an optimal solution satisfying the constraints. This is to prove that the use of inversion does not induce any particular degradation in terms of numerical sensibility.

In each case we did 200 runs with random initial conditions. All tests were conducted on a PC under Linux (Red Hat 6.2) with a Pentium III 733MHz processor.

The results detailed in figure 3 show that the cpu-time is exponentially decreasing with the relative degree. The slowest problem to solve is the one fully collocated and the fastest is the "fully inverted" (flat) one.

Interpretation NTG relies on NPSOL, the Fortran nonlinear programming package developed by (Gill *et al.* n.d.) NPSOL solves nonlinear programming problems using a sequential programming (SQP) algorithm, involving major and minor iterations. At each major iteration a new quadratic programming (QP) problem is defined that approximates both the nonlinear cost function and the nonlinear constraints. This QP problem is solved during the minor iterations. The overall cpu-time required is highly related to the sum of all the minor iterations.

Inspecting the runs, we concluded that the successive QP subproblems are generally well conditioned in all cases.

In this example, each variable is represented by approximatively 15 coefficients. Therefore the number of variables is a decreasing function of the relative degree, see Figure 3.

It is known, see (Gill *et al.* 1994), that the cost of solving a well-conditioned QP problem grows as a cubic function of the number of variables. In Figure 4 one can see that this is a good explanation of the differences in experimental cpu-time, the slope of the linear regression of the mean values of cpu-time versus the number of variables being 2.80 .

4. CONCLUSION AND PERSPECTIVES

In this paper we explained the advantage of inversion in constrained trajectory optimization.

The chosen example illustrates a general methodology. The role of the relative degree was clearly visible for the fifth order single input nonlinear dynamics considered here, both from a theoretical and a computational point of view.

This problem is quite an extreme situation, and is not the most frequent case in engineering.

In mechanical problems for instance, we are more likely to meet two or three of degrees of freedom systems. In the case of fully actuated systems, a known fact is known that these systems are flat, i.e. full inversion is possible and given by the Euler-Lagrange equations

$$\frac{d}{dt}\frac{\partial L(q,\dot{q})}{\partial \dot{q}} - \frac{\partial L(q,\dot{q})}{\partial q} = u.$$

Relative degree	Cpu-time (s.) (average)	Number of Variables (after collocation)	Rate of convergence (%)
0	70.0	90	79.5
1	52.2	76	92.5
2	25.7	65	85.0
3	5.1	43	99.5
4	1.7	29	91.5
5	0.50	14	92.0

Fig. 3. Main results. The cpu-time is an exponential decreasing function of the relative degree. Top: full collocation. Bottom: flatness parametrization.

In the under actuated case, things are more complicated.

Consider a simplified model of a Uninhabited Combat Air Vehicle (UCAV), commonly referred to as the *planar ducted fan*

$$\begin{aligned} m\ddot{x}\cos\theta - (m\ddot{z} - mg)\sin\theta &= F_{Xb} \\ m\ddot{x}\sin\theta + (m\ddot{z} - mg)\cos\theta &= F_{Zb} \\ (J/r)\ddot{\theta} &= F_{Zb} \end{aligned}$$

where the 6-dimensional state is $(x, \dot{x}, z, \dot{z}, \theta, \dot{\theta})$ and the inputs are F_{Xb} and F_{Zb}.

With (x, z) as output, the system has relative degree $(2, 2)$ because

$$\begin{aligned} \ddot{x} &= \frac{1}{m}\left(F_{Xb}\cos\theta + F_{Zb}\sin\theta\right) \\ \ddot{z} &= g + \frac{1}{m}\left(-F_{Xb}\sin\theta + F_{Zb}\cos\theta\right) \end{aligned}$$

and the Jacobian $J_{F_{Xb},F_{Zb}} = \frac{1}{m^2} \neq 0$, i.e. two independent combinations of the inputs appear in the second derivatives of the outputs. Following our methodology it is possible to parameterize the whole system by (x, z) and $6 - 2 - 2 = 2$ other quantities, namely $\theta, \dot{\theta}$. This is a straightforward extension of the approach presented here. Any optimal control problem for this system can be solved by a partial collocation, i.e. collocating only $(x, z, \theta, \dot{\theta})$. The new formulation involves first and second derivatives of x and z because the relative degree is $(2, 2)$.

Further this system is flat as shown in (Martin *et al.* 1996). Choosing $z_1 = x + \frac{J}{rm}\cos\theta$ and $z_2 = x - \frac{J}{rm}\sin\theta$ gives a full parameterization of all the variables of the system. Yet this parameterization involves $(z_1, \dot{z}_1, ..., z_1^{(4)}, z_2, \dot{z}_2, ..., z_2^{(4)})$. The notion of relative degree is not defined in this example. Though the total number of derivatives required, $4 + 4 = 8$ which exceeds the dimension of the state, can be deduced from another argument i.e. the dynamic extension algorithm. For a general multi-input multi-output system the question of the number of derivatives required is still open.

We are currently working on this multi-input multi-output question from a theoretical point of view and investigating the computational consequences of inversion in this case. It is possible that the best parameterization may not be the flat one. The parameterization is highly dependent on the constraints imposed on the problem. These issues are of interest in our next area of application, real-time trajectory generation for the Caltech ducted fan experiment, see (Milam and Murray 1999).

5. REFERENCES

Bryson, A. E. (1999). *Dynamic optimization*. Addison Wesley.

de Boor, C. (1978). *A Practical Guide to Splines*. Springer-Verlag.

Fliess, M., J. Lévine, P. Martin and P. Rouchon (1995). Flatness and defect of non-linear systems: introductory theory and examples. *International Journal of Control* **61**(6), 1327–1360.

Fliess, M., J. Lévine, P. Martin and P. Rouchon (1999). A Lie-Bäcklund approach to equivalence and flatness of nonlinear systems. *IEEE Trans. Auto. Cont.* **44**(5), 928–937.

Gill, P. E., W. Murray and M. A. Saunders (1994). *Large-scale SQP Methods and their Application in Trajectory Optimization*. Chap. 1. Birkhäuser Verlag.

Gill, P. E., W. Murray, M. A. Saunders and M. Wright (n.d.). *User's Guide for NPSOL 5.0: A Fortran Package for Nonlinear Programming*. Systems Optimization Laboratory. Stanford University, Stanford, CA 94305.

Hargraves, C. and S. Paris (1987). Direct trajectory optimization using nonlinear programming and collocation. *AIAA J. Guidance and Control* **10**, 338–342.

Isidori, A. (1989). *Nonlinear Control Systems*. Springer-Verlag.

Martin, P., S. Devasia and B. Paden (1996). A different look at output tracking: control of a VTOL aircraft. *Automatica* **32**(1), 101–107.

Milam, M. B. and R. M. Murray (1999). A testbed for nonlinear control techniques: The Caltech Ducted Fan. In: *1999 IEEE Conference on Control Applications*.

Milam, M. B., K. Mushambi and R. M. Murray (2000). A new computational approach to real-time trajectory generation for constrained mechanical systems. In: *IEEE Conference on Decision and Control*.

Seywald, H. (1994). Trajectory optimization based on differential inclusion. *J. Guidance, Control and Dynamics* **17**(3), 480–487.

Sweetman, B. (1997). Fighters without pilots. *Popular Science* pp. 97–101.

TechSat 21 `//www.vs.afrl.af.mil/vsd/techsat21/`