FP14 4:50

Proceedings of the 34th
Conference on Decision & Control
New Orleans, LA - December 1995

Approximate Trajectory Generation for Differentially Flat Systems with Zero Dynamics

Michiel J. van Nieuwstadt
vannieuw@indra.caltech.edu

Richard M. Murray
murray@indra.caltech.edu

Division of Engineering and Applied Science
California Institute of Technology
Pasadena, CA 91125

Abstract

This paper describes algorithms to generate trajectories for differentially flat systems with zero dynamics. Zero dynamics in flat systems occur when the flat outputs are not the tracking outputs. This means that the output trajectories can be fully parametrized by the flat outputs, but that there is some additional freedom left. This freedom can be exploited to minimize a cost criterion. We parametrize the differentially flat outputs by basis functions, and solve for the parameters so as to track a prescribed trajectory approximately while minimizing a cost function. We give examples of such systems, and present simulations and experimental data. We focus on implementation issues and point out the computational cost involved in the various problems.

1. Introduction

In this paper we describe algorithms to generate trajectories for a class of nonlinear systems. This is part of a general control paradigm for nonlinear systems depicted in Figure 1.

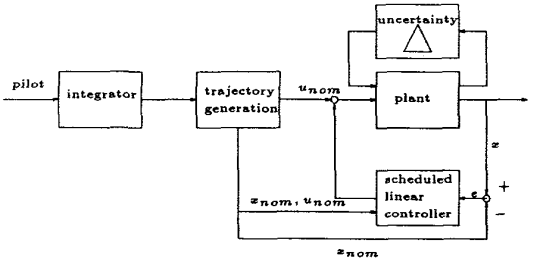

Fig. 1: Paradigm for nonlinear control.

In this framework, the operator gives a rate of change of the tracking variables, or a desired destination in tracking space. The trajectory generation module generates a nominal state space trajectory and a nominal control input. This part of the controller can be run at a rate lower than the sampling rate, since the dynamics of the operator are typically much slower than those of the plant. The plant is linearized around the nominal trajectory, and a linear controller is used to stabilize the plant around this trajectory and deal with uncertainty. The advantage of linearizing the plant around a trajectory as opposed to using a coordinate transformation is that in the latter case it is often impossible to get a good uncertainty description that makes physical sense. The linear controller runs at a higher rate, since it needs to stabilize the plant dynamics. Note that the linear controller needs to have information about the nominal state to compute the appropriate linearization. All this should happen in real time.

We only consider trajectory generation over a finite horizon. The infinite horizon asymptotic tracking problem was proven to be unsolvable in many interesting cases in [7]. Note that this finite horizon may shift as time proceeds, to accomodate real time trajectory generation. All trajectory generation algorithms are to some extent acausal: we need some information about the trajectory in future time to obtain sufficient smoothness. The degree of acausality depends, among other things, on the computation time of the trajectory. If trajectory computation takes a long time, we need to know the trajectory a long time ahead. Therefore computation time is of prime influence on the nature of the generated trajectories.

In general, the trajectory generation problem cannot be solved analytically. An exception is formed by linear systems. For general systems we can only solve the generation problem by repeatedly integrating the system equations and trying to minimize some error between the computed trajectory and the desired trajectory. Differentially flat systems are systems in which all states and inputs can be expressed as functions of some outputs and their derivatives [6, 9, 16]. They have the useful property that there is a 1 to 1 mapping between trajectories in output space and trajectories in state space. This means that we can plan a trajectory in the lower dimensional output space and lift it to state space. In the output space we have no dynamics, so trajectory generation is particularly easy.

In general the flat outputs that parametrize all system trajectories are not the outputs that we want to track. In [11] it is suggested to redefine the outputs to the flat outputs, so that the tracking problem becomes trivial. However, this may have undesired effects for the zero dynamics of the original system. Tracking the flat ouputs will allow exact tracking but might drive the zero dynamics of the original outputs to undesirable magnitude. If we maintain the original outputs, we can still parametrize all system trajectories with the flat outputs, but in general for each trajectory of the tracking outputs, we can find more than one trajectory of the flat outputs. This freedom can be used to advantage to minimize an additional cost function. Typically, we pick this cost function to bound the magnitude of the coordinates describing the zero dynamics. We discuss advantages and drawbacks of this approach, and

*Research supported in part by NASA, by NSF Grant CMS-9502224 and AFOSR Grant F49620-95-1-0419

†Keywords: differential flatness, trajectory generation, nonlinear control, real time control

compare with the stable inversion proposed in [5, 3, 2].

Although the scheme in Figure 1 is quite common, [12, 11]. implementation issues are usually ignored. We added an operator and an integrator box to the scheme to indicate emphasis on online input. We focus on digital implementation and real time feasibility. Work to this effect was presented in [12], where trajectory computations were done in pseudo real time, which we will loosely define to mean update rates 4 orders of magnitude slower than the controller rate. All algorithms will describe which numerical computations have to be performed, and give an indication of the computational cost.

The organization of the paper is as follows. In section 2 we review flatness. In section 3 we give examples of physical systems that are flat with respect to different outputs than the natural tracking outputs. In section 4 we present different trajectory generation problems and show simulations. In section 5 we show experimental data for the Caltech ducted fan. In the last section we summarize our conclusions and indicate future research.

2. Differential Flatness

Differential flatness was introduced by Fliess et al. in a differential algebraic context [6, 9]. In [16] it was reinterpreted in an exterior differential systems setting. We will briefly reiterate this definition here. See [15, 1] for a more detailed treatment of exterior diffential systems and their connections with control theory.

We write a nonlinear system

$$\dot{x} = f(x, u, t) \tag{1}$$

as a set of one forms

$$I = \{dx_1 - f_1(x, u, t)dt, \ldots, dx_n - f_n(x, u, t)dt\} \tag{2}$$

where $(x, u, t) \in M$, a manifold of appropriate dimension. The *Pfaffian system* generated by these one forms on the manifold M is the module it generates over the ring of smooth functions on M, i.e., $C^\infty(M)$.

If $\pi : B \to M$ is a fiber bundle, we can define a new system on the manifold B by adding one forms ω_i to I. The system $J = (I, \omega_i)$ on the higher dimensional manifold B thus created is called a *Cartan prolongation* of the system I if

1. every solution curve of I lifts to a unique solution curve of J
2. $\pi^*(I) \subset J$.

Roughly, a prolongation of a control system corresponds to adding feedback.

An *independence condition* for a Pfaffian system is a one form that is not allowed to vanish on any of the solution curves of the system. This allows us to model time in physical systems, by adding dt as an independence condition to our system: time is restricted to always increase for a physical solution.

The *trivial system* is the Pfaffian system $(0, dt)$, that is, a system with no constraints and with independence condition dt.

Two systems I_1 and I_2 are *absolutely equivalent* if they have Cartan prolongations J_1 and J_2 defined on fiber bundles B_1, B_2 respectively, that are equivalent, i.e. there is a diffeomorphism $\phi : B_1 \to B_2$ such that $\phi^*(J_2) = J_1$. This allows us to establish a notion of equivalence between systems of different dimensions.

A system is *differentially flat* if it is absolutely equivalent to the trivial system. In particular this means that every curve in the flat output space has a unique counterpart in the state and input space of the original system. This allows us to generate trajectories in the flat output space where we don't have to deal with constraints, and then lift them to our original system.

Let z denote the flat outputs, which are functions of the states, the inputs and their derivatives. It can be shown that the above definition in exterior differential systems setting implies the following [16]

$$\begin{aligned} z &= \psi(x, u, u^{(1)}, \ldots, u^{(l)}) \\ (x, u) &= \phi(z, z^{(1)}, \ldots, z^{(l)}) \end{aligned} \tag{3}$$

for some l. That is, we can express the state and the inputs as functions of the flat outputs and their derivatives. This was the original definition in the differential algebraic setting.

In [16, 10] it was shown that in an open and dense set differential flatness is equivalent to dynamic feedback linearizability. At first sight differential flatness does not seem to introduce a new concept. However, feedback linearization is a notion that only applies to an equilibrium point. There are systems (e.g. some driftless systems) that are differentially flat away from an equilibrium point, but not feedback linearizable around that equilibrium point. This indicates that for the trajectory generation problem it is more interesting to look at the system structure around a trajectory, rather than around an equilibrium. Flatness means that around a trajectory the system looks like a linear system.

Even if a system is feedback linearizable around an equilibrium, it might be that the corresponding linearizing outputs are not the outputs we want to track. In this case the linearization does not help us to generate trajectories. This is the case we are particularly interested in. Tracking the outputs of interest will result in, possible unstable, zero dynamics. Parametrizing the flat outputs allows us to trade off the magnitude of the zero dynamics with the tracking error.

3. Examples

This section presents examples of flat systems with zero dynamics.

3.1. The kinematic car

Consider the well known equations of motion for a kinematic car (see Figure 2)

$$\begin{aligned} \dot{x} &= \cos\theta \cos\phi \, v_1 \\ \dot{y} &= \sin\theta \cos\phi \, v_1 \\ \dot{\theta} &= \frac{1}{l} \sin\phi \, v_1 \\ \dot{\phi} &= v_2. \end{aligned} \tag{4}$$

Here, (x, y) is the position of the rear axle, θ is the angle between the horizontal and the rear car, ϕ is the steering angle, v_1 is the forward velocity of the front

wheels, v_2 is the steering angle velocity and l is the distance between front and rear axle.

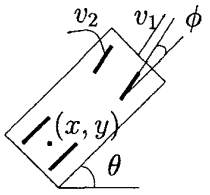

Fig. 2: The kinematic car.

This system is flat with flat outputs (x, y), the position of the rear axle. If we want to back up a truck into a loading dock these outputs are the same as the tracking outputs. For other problems, e.g. when the driver is trying to negotiate a window at a drive-thru restaurant, it might be more appropriate to generate a trajectory for the front cab of the car. Then the tracking outputs are $(x + l\cos\theta, y + l\sin\theta)$, and the zero dynamics can be parametrized by ϕ. In general it is desirable to keep ϕ small. This can be achieved by setting up a cost criterion that minimizes a weighted integral of the tracking error and the magnitude of ϕ. Note that ϕ can be expressed in terms of the flat outputs as

$$\phi = \arctan(\frac{1}{l}(\dot{x}^2 + \dot{y}^2)^{3/2}, \dot{x}\ddot{y} - \dot{y}\ddot{x}). \tag{5}$$

3.2. The Ducted Fan Engine

The ducted fan is a model of a thrust vectored aircraft mounted on a stand, (which introduces some parasitic dynamics), as shown in see Figure 3. See [4] for a detailed description of this apparatus.

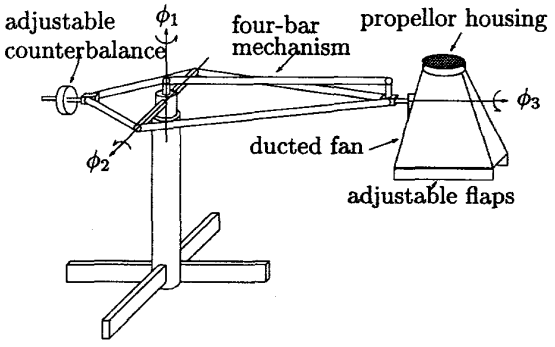

Fig. 3: Ducted fan with stand.

The ducted fan is mounted on a stand with a counterweight that moves in as the fan moves up. This results in inertial masses m_x and m_y in the x and y direction respectively, that change with the y coordinate. We do not take the variation of these inertial masses with y into account but take their value around hover. The counterweight also results in an effective weight m_z different than the masses in x and y direction.

We can apply any force on the center of mass by adjusting the magnitude and the direction of the thrust. After shifting the control variables to compensate for gravity, and decomposing them into a parallel and perpendicular component, the equations of motion are:

$$\begin{aligned} m_x\ddot{x} &= -m_z g\sin\theta + \cos\theta u_1 - \sin\theta u_2 \\ m_y\ddot{y} &= m_z g(\cos\theta - 1) + \sin\theta u_1 + \cos\theta u_2 \\ J\ddot{\theta} &= r u_1 \end{aligned} \tag{6}$$

where (x, y) are the coordinates center of the center of mass, θ is the angle with the vertical, u_1 is the force perpendicular to the fan body, u_2 is the force parallel to the fan body, r is the distance between the center of mass and the point where the force is applied, g is the gravitational constant, $m_{x,y}$ is the inertial mass of the fan in the (x, y) direction respectively, m_z is the gravitational mass of the fan, and J is the moment of inertia of the fan. The tracking outputs are the (x, y) coordinates of the center of mass.

Note that these equations are almost identical to the ones presented in [8] and [11], except for the small parameter ϵ multiplying the u_2 term that occurs in those references. We do not impose this restriction here.

In [11] the flat outputs were shown to be:

$$\begin{aligned} x_f &= x - \frac{J}{m_x r}\sin\theta \\ y_f &= y + \frac{J}{m_y r}\cos\theta \end{aligned} \tag{7}$$

Note that these outputs are not fixed in body coordinates We can dynamically feedback linearize this system by the following dynamic extension:

1. Add u_2 as a state, and let $\dot{u}_2 = u_3$.
2. Apply the following input transformation: $u_4 = -2u_1\dot{\theta} + u_3$.
3. Add u_4 as a state, and let $\dot{u}_4 = u_5$.

The extended system is

$$\begin{aligned} m_x\ddot{x} &= -m_z g\sin\theta + \cos\theta u_1 - \sin\theta u_2 \\ m_y\ddot{y} &= m_z g(\cos\theta - 1) + \sin\theta u_1 + \cos\theta u_2 \\ J\ddot{\theta} &= r u_1 \\ \dot{u}_2 &= u_4 + 2u_1\dot{\theta} \\ \dot{u}_4 &= u_5 \end{aligned} \tag{8}$$

with new inputs (u_1, u_5). The coordinate transformation

$$(x_f, \ldots, x_f^{(3)}, y_f, \ldots, y_f^{(3)}) \mapsto (x, y, \theta, \dot{x}, \dot{y}, \dot{\theta}, u_2, u_4) \tag{9}$$

has determinant

$$-\frac{(m_z g r - J\dot{\theta}^2 + r u_2)^2}{m_x^2 m_y^2 r^2} \tag{10}$$

which is nonzero around the origin. The determinant of the I/O decoupling matrix,

$$\det(\frac{\partial(x_f^{(4)}, y_f^{(4)})}{\partial(u_1, u_5)}) = \frac{-m_z g r + J\dot{\theta}^2 - r u_2}{J m_x m_y}, \tag{11}$$

is nonzero around the origin. Therefore the extended system has well defined relative degree around the origin, and is feedback linearizable. It is interesting to note that both the decoupling matrix and the coordinate transformation become singular if no gravity is present. The system will still be flat in zero gravity, since flatness is not restricted to an equilibrium point:

in an open and dense set the system will still have linear structure. We will not explictly use the feedback linearization, since the form in equation (8) is sufficient for trajectory generation purposes.

Note that the zero dynamics (with respect to outputs (x, y))

$$\ddot{\theta} = \frac{m_z g r}{J} \sin\theta \tag{12}$$

are unstable. Imposing a bound on θ will impose a bound on the zero dynamics. The variable θ can be expressed in terms of the flat outputs as

$$\tan\theta = \frac{-m_x \ddot{x_f}}{m_y \ddot{y_f} + m_z g}. \tag{13}$$

3.3. The axially symmetric rigid body with 3 forces

Consider the a rigid body symmetric about the y-axis with 3 forces acting in 1 point on the y-axis, depicted in Figure 4. This could model an underwater vehicle or a zeppelin.

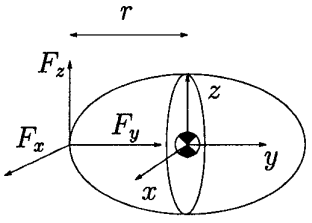

Fig. 4: The axially symmetric body with 3 forces acting in a point.

Due to the axial symmetry we ignore the rotation about the y-axis. We have no actuating torque there, and do not measure the angle. In the language of geometric mechanics, we reduce the dynamics by the symmetry around the y-axis. Recall Euler's equations for a rigid body in body coordinates [13],

$$J\dot{\omega}^b + \omega^b \times J\omega^b = \tau^b \tag{14}$$

Where J is the inertia matrix which will be diagonal with $J_x = J_z$ due to axial symmetry. ω^b is the rotational velocity in body coordinates. Put the origin at the center of mass, and let the forces act at the point $(0, r, 0)$, (note that r is negative in the picture), then $\tau = (rF_z, 0, -rF_x)$.

Suppose we observe the point $p^b = (0, -\frac{J_x}{mr}, 0)$ on the body. This is the equivalent of the center of oscillation for the ducted fan. Our observation in a spatial frame is

$$p^s = p_c^s + Rp^b \tag{15}$$

where p_c^s are the coordinates of the center of mass in the spatial frame, and R is the rotation matrix from body to spatial coordinates. Since we ignore rotation about y, R has 2 unknown parameters, say ϕ and θ. Differentiating equation (15) gives:

$$\dot{p}^s = \dot{p}_c^s + R\dot{p}^b + R\hat{\omega}^b p^b \tag{16}$$

where $\hat{\omega}p = \omega \times p$. Note that $\dot{p}^b = 0$, since p^b is a body fixed point.

Differentiating equation (16) and using Euler's equations and $F^s = RF^b = m\ddot{p}_c^s$ gives

$$m\ddot{p}^s = R \begin{pmatrix} -J_y \omega_x^b \omega_y^b / r \\ F_y + J_x((\omega_x^b)^2 + (\omega_z^b)^2)/r \\ -J_y \omega_y^b \omega_z^b / r \end{pmatrix} - \begin{pmatrix} 0 \\ mg \\ 0 \end{pmatrix} \tag{17}$$

Now we need the extra assumption that $\omega_y^b \equiv 0$. Then we know the direction of the second column of R, that is, we know the attitude up to a rotation about the y-axis, which we ignore. This gives us θ and ϕ. One more differentiation gives ω_x^b, ω_z^b, then equation (17) gives us F_y, and the Euler equations give us F_x, F_z.

If we want to track the center of mass instead of the center of oscillation, we will have unstable zero dynamics, entirely analogous to the ducted fan.

Note that even though we have no direct actuation of the roll rotation, we can rotate about the y-axis by performing a sequence of noncommuting rotations about the x and z axes.

4. Trajectory generation problems for differentially flat systems

In this section we will enumerate different trajectory generation problems and investigate their computational cost. All code is written in C, all computation times refer to an Intel 486 DX2 CPU, running at 66 MHz, while the numerical routines are slightly modified version of those provided in the Numerical Recipes in C [14]. Throughout this section we will denote flat outputs by z and tracking outputs by y. We will be looking at trajectories over a finite time interval (t_0, t_1). We will approximate trajectories by polynomials, since this allows us to perform derivative calculations symbolically.

All examples are based on the ducted fan presented in section 3. The extended system for this example has 8 states and 2 inputs.

The values of the parameters for a ducted fan that was built in our lab are: $g = 9.8$ m/s^2, $r = 0.25$ m, $J = 0.0475m^2$kg, $m_x = 4.19$ kg, $m_y = 3.71$ kg, $m_z = 0.27$ kg.

4.1. Point to point steering problems

The easiest tracking problem is where we want to steer from one point in state space to another point in state space. For this problem it is irrelevant whether the flat outputs are the tracking outputs or not, since we are given the entire state at two points in time. Suppose we want to steer from $x(t_0) = x_0$ to $x(t_1) = x_1$. Assume the inputs and their derivatives at both times are also specified. Then we can compute the flat outputs and their derivatives at the initial and final times. We parametrize the flat outputs z as

$$z_i(t) = A_{ij}\phi_j(t) \tag{18}$$

(using implicit summation) where the $\phi_j(t)$ are some basis functions. We need to solve for the coefficients A_{ij} in the following equations:

$$\begin{aligned} z_i(t_0) &= A_{ij}\phi_j(t_0) & z_i(t_f) &= A_{ij}\phi_j(t_0) \\ &\vdots & &\vdots \\ z_i^{(l)}(t_0) &= A_{ij}\phi_j^{(l)}(t_0) & z_i^{(l)}(t_f) &= A_{ij}\phi_j^{(l)}(t_f) \end{aligned} \tag{19}$$

N	Time ([s])
50	0.66
100	1.05
200	1.76
300	2.58

Table 1: Computation time for point to point steering.

We need enough basis functions so that these equations have a solution. If the dimension of the state is n and the dimension of the input is m, then we need $2(n + m(l + 1))$ coefficients. The point to point steering problem then amounts to solving a system of linear equations with $2(n + m(l + 1))$ unknowns.

After we compute the coefficients A_{ij} we need to compute the trajectory at a number of time points for our real time implementation from equation (3). The more time points we have, the more accurate our controller will be. Solving for the states and inputs amounts to solving a nonlinear systems of equations for each desired trajectory point. Since this is an iterative process, the computation time is hard to predict. It is clear however that this computation will be much longer than the computation of the coefficients from equation (19). Table 1 lists the computation time vs. number of time points for the ducted fan when each flat output is parametrized by 8 polynomials. The numerical algorithm used is a Newton's method. If we don't require to fix the inputs at both ends, we need the flat outputs and 3 derivatives, so that we need 8 polynomials for each flat output for the equations (19) to have a solution.

Figure 5 shows the trajectory for an initial state $(x, y, \theta, \dot{x}, \dot{y}, \dot{\theta}) = (0, 0, 0, 0, 0, 0)$ at $t_0 = 0$, and a final state $(x, y, \theta, \dot{x}, \dot{y}, \theta) = (1.0, 0, 0, 0, 0, 0)$ at $t_1 = 3.0$.

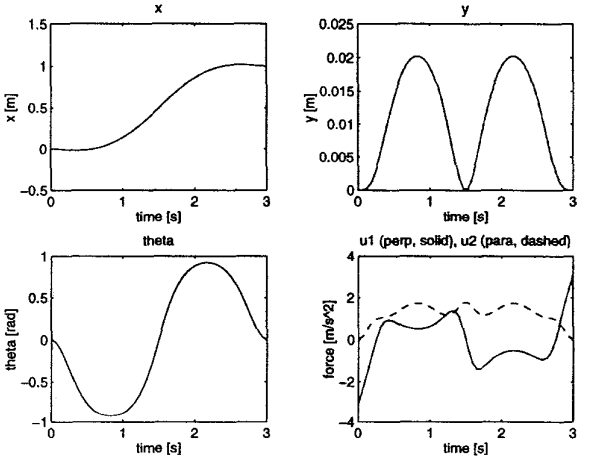

Fig. 5: Trajectory for a point to point steering problem.

4.2. Approximate trajectory generation for flat systems

If the tracking outputs are the flat outputs and we are given a trajectory to track, we still want to parametrize the flat outputs by basis functions. The reason is that computation of the states requires numerical differentiation of the flat outputs, up to several orders, which is an ill conditioned computation. Large magnitudes of the derivatives can prevent convergence of the nonlinear solver of equation (3).

This leads to the least squares problem:

$$\min_A \int_{t_0}^{t_1} \big(z(t) - A\phi(t)\big)^* W(t) \big(z(t) - A\phi(t)\big) dt \tag{20}$$

where A_{ij} is the coefficient of basis function j in flat output i, $\phi(t)$ is the vector of basis functions evaluated at time t, and $W(t)$ is a time varying weighting matrix. This minimization has the closed form solution:

$$A = M^{-1} L \tag{21}$$

where $M_{ij} = \int_{t_0}^{t_1} f_i(t) W_{ij}(t) f_j(t) dt$ and $L_{ij} = \int_{t_0}^{t_1} f_i(t) W_{ij}(t) z_j(t) dt$. Note that the problem is decoupled with respect to the different outputs: we can compute the coefficients for each flat output separately. In our implementation we approximate integration by summation. The computation of M^{-1} only has to be performed once, so that there is no great savings in picking orthogonal basis functions.

Since we are only minimizing an integrated error, the resulting trajectory does not necessarily start at the state we are at. However, we can fix initial and final conditions (or conditions at any time for that matter), by imposing linear constraints on the coefficients A_{ij} exactly as in equation (19). Suppose the linear constraints on A are given by $z = FA$. Then we have to find a particular solution to these equations, say $A_0 = F^\dagger z$, and we can reparametrize $A = A_0 + F^\perp A_1$, where $F^\perp$ is a basis for the null space of F. The composed problem will still be linear in A_1.

4.3. Approximate tracking for nonmimimum phase systems

Now suppose that the flat outputs are not the tracking outputs. Then we can track a trajectory and still have some freedom left. We can use this freedom to minimize a cost function. Typically this cost function bounds the magnitude of the zero dynamics.

This leads to the minimization problem:

$$\min_A \int_{t_0}^{t_1} \big(y(A,t) - A\phi(t)\big)^* W(t) \big(y(A,t) - A\phi(t)\big) + \lambda J(z, z^{(1)}, \ldots, z^{(l)}) dt \tag{22}$$

where λ trades off the tracking accuracy against the zero dynamics. With $\lambda = 0$ we will track exactly, within the accuracy of the basis function parametrization, but we have no control over the zero dynamics. With λ large we will have poor tracking but small zero dynamics. The minimization problem (22) is in general a nonconvex nonlinear minimization problem, so that we cannot guarantee convergence to a global minimum. We approximate the integral with a discrete sum. For each time point in this sum we need to compute the flat outputs and the states from the coefficients A_{ij}, and then evaluate the integrand. This results in long computation times. Computation time depends on the particular problem and the required accuracy. After the coefficients A_{ij} have been found, we still have to compute the state-input trajectory at a number of time

NPOL	Time ([s])	cost
(4,4)	92.88	8.38
(6,6)	175.81	6.45
(8,8)	357.57	6.45

Table 2: Computation times for approximate tracking with zero dynamics.

points. This latter computation will only take a fraction of the time needed to compute the minimum of the cost (22).

For the ducted fan, the cost criterion we are trying to minimize is :

$$\int_{t_0}^{t_1} (y(t) - Af(t))^*(y(t) - Af(t)) + \lambda\theta^2 dt \tag{23}$$

where the entries of f are polynomials. The trajectory $y(t)$ is depicted in the first 2 windows of Figures 6 and 7. Figure 6 shows the results for $\lambda = 0.1$. Figure 7 shows the results for $\lambda = 1.0$. The effect of increasing λ is as expected: it decreases the magnitude of θ at the expense of performance.

Table 2 shows the computation time in seconds for this trajectory. The pair NPOL refers to the number of polynomials for each of the flat outputs. The column "cost" refers to the achieved value of the minimization criterion (23).

We used a Powell direction set method with numerical approximation of the gradient. This method is quite slow but resulted in lower values for the cost than conjugate gradient methods, where an explicit formula for the gradient is needed, and variable metric methods, which use an approximate Hessian. The latter two methods had a tendency to get stuck in local minima. If the computation time has to be short, we can settle for the higher cost value. As we can see, the computation times are quite long. They are definitely too long for real time applications, but are still feasible for pseudo real time applications. Also, increasing the number of basis functions very soon stops giving a lower cost function and increases computation time enormously. We expect that tuning the minimization method will result in much shorter computation times. The same remarks for fixing the initial conditions hold as in the previous subsection, except that we now have a nonlinear minimization problem with linear constraints. Using the same parametrization $A = A_0 + F^{\perp}A_1$, we minimize over A_1.

This approach is to be contrasted with [11] where it is suggested to track the flat outputs disregarding the excursions of θ. It should also be contrasted with [3, 2, 5], where an iterative solution to the tracking problem with unstable zero dynamics is proposed. This solution offers exact tracking, and needs a preliminary input trajectory (prologue) to bring the state to a starting point on the unstable zero dynamics manifold, and an epilogue to bring the state from the stable zero dynamics manifold to an equilibrium. During the actual tracking no bounds on the zero dynamics are imposed. The solution presented here is computationally much simpler. The iterative solution in [2] involves repeated solution of a differential equation over the entire time interval. It also requires numerical differentiation of the desired

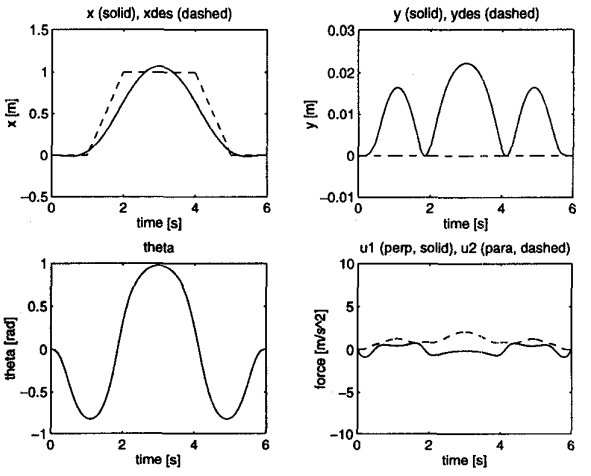

Fig. 6: Trajectory for cost minimization with non minimum phase outputs, $\lambda = 0.1$.

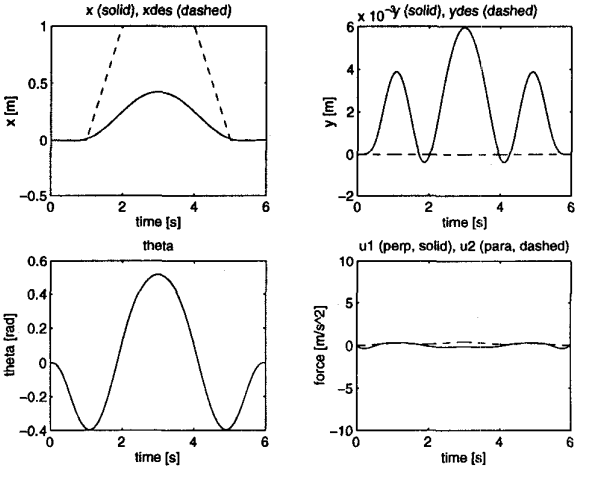

Fig. 7: Trajectory for cost minimization with non minimum phase outputs, $\lambda = 1.0$.

trajectory, since the zero dynamics are dependent on this trajectory and its derivatives.

Our does not require a prologue to bring the zero dynamics to the unstable zero dynamics manifold. We can bound the zero dynamics during tracking, at the cost of a larger tracking error. The obvious drawbacks of our solution are that it only works with flat systems and only offers approximate tracking.

5. Experimental Data

In this section we present experimental data to validate the nonlinear control paradigm depicted in Figure 1. The data is taken with the Caltech ducted fan, described in section 3. We compare a 1 degree of freedom design (Figure 9), where only the desired output is fed forward, to a 2 degree of freedom design (Figure 8), where we feed forward the entire state and input space trajectory. In both cases we use the same LQR controller to stabilize the system around the trajectory. We use the point to point steering technique from section 4 repeatedly to compute the following trajectory for the approximate flat model of the ducted fan.

- Stay at $(0, 0)$ hover for 1 seconds.
- Steer from $(0, 0)$ to $(1, 0)$ meter in 4 seconds.
- Stay at $(1, 0)$ hover for 1 seconds.
- Steer from $(1, 0)$ to $(0, 0)$ meter in 4 seconds.
- Stay at $(0, 0)$ hover for 5 seconds.

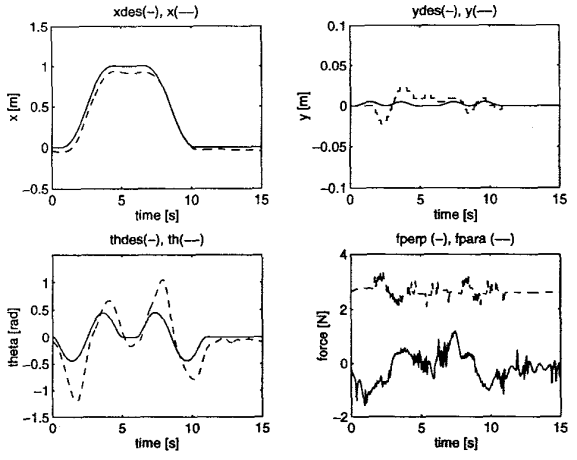

Fig. 8: Two degree of freedom.

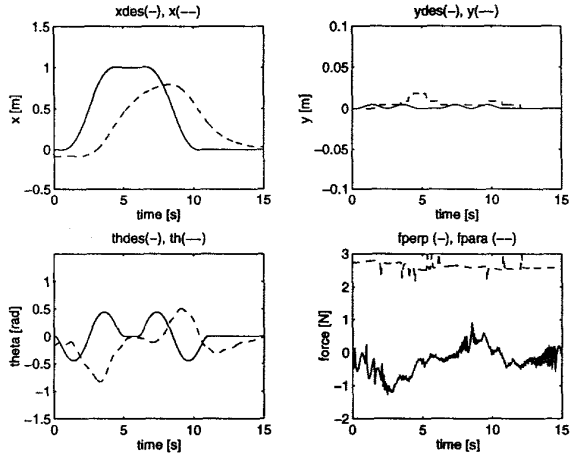

Fig. 9: One degree of freedom.

The 2 degree of freedom design gives a much more aggressive response, showing the validity of this apporoach.

6. Summary and Conclusions

This paper studies implementation issues related to a 2 degree of fredom nonlinear control paradigm. In this paradigm one module generates a trajectory around which another module stabilizes the system. We presented experimental data for the Caltexh ducted fan to validate this approach.

For differentially flat systems, we reviewed different trajectory generation problems and evaluated them on their computational cost. We presented approximate trajectory tracking for flat systems with zero dynamics as an alternative to stable system inversion and discussed its advantages and drawbacks.

Future research will focus on real time implementations of these algorithms.

References

[1] R.L. Bryant, S.S. Chern, R.B. Gardner, H.L. Goldschmidt, and P.A. Griffiths. *Exterior Differential Systems.* Springer Verlag, 1991.

[2] D. Chen. An iterative solution to stable inversion of nonminimum phase systems. In *ACC Proceeedings*, pages 2960–2964, June 1994.

[3] D. Chen. Output tracking of nonlinear nonmimimun phase systems. In *CDC Proceeedings*, pages 2340–2345, December 1994.

[4] H. Choi, P. Sturdza, and R.M. Murray. Design and construction of a small ducted fan engine for nonlinear control experiments. In *ACC Proceeedings*, pages 2618–2622, June 1994.

[5] S. Devasia and B. Paden. Exact output tracking for nonlinear time-varying systems. In *CDC Proceedings*, pages 2346–2355, December 1994.

[6] M. Fliess, J. Lévine, Ph. Martin, and P. Rouchon. On differentially flat nonlinear system. In *NOLCOS*, pages 408–412, 1992.

[7] J.W. Grizzle, M.D. Di Benedetto, and F. Lamnabhi-Lagarrigue. Necessary conditions for asymptotic tracking in nonlinear systems. *IEEE TAC*, 39(9):1782–1795, September 1994.

[8] J. Hauser, S. Sastry, and G. Meyer. Nonlinear control design for slightly nonminimum phase systems: Application to V/STOL aircraft. *Automatica*, 28(4):665–679, 1992.

[9] Ph. Martin. *Contribution à l'étude des systèmes différentiellemant plats.* PhD thesis, L'Ecole Nationale Supérieure des Mines de Paris, 1993.

[10] Ph. Martin. Endogenous feedbacks and equivalence. In *MTNS 93*, Regensburg, Germany, August 1993.

[11] Ph. Martin, S. Devasia, and B. Paden. A different look at output tracking: control of a VTOL aircraft. In *CDC Proceedings*, pages 2376–2381, December 1994.

[12] G. Meyer, L.R. Hunt, and R. Su. Nonlinear system guidance in the presence of transmission zero dynamics. Technical Report Draft 5, NASA, October 1994.

[13] R.M. Murray, Z. Li, and S.S. Sastry. *A Mathematical Introduction to Robotic Manipulation.* CRC Press, 1994.

[14] W.H. Press, W.T. Vetterling, S.A. Teukolsky, and B.P. Flannery. *Numerical Recipes in C.* Cambridge University Press, 1992.

[15] W. Sluis. *Absolute Equivalence and its Applications to Control Theory.* PhD thesis, University of Waterloo, Waterloo, Ontario, 1992.

[16] M. van Nieuwstadt, M. Rathinam, and R.M. Murray. Differential flatness and absolute equivalence. In *CDC conference*, pages 326–333, December 1994.