Applied Receding Horizon Control of the Caltech Ducted Fan*

Ryan Franz† Mark Milam‡ John Hauser†

{franzr,milam,hauser}@cds.caltech.edu

Abstract

This paper details the application of a constrained receding horizon control strategy to the Caltech Ducted Fan, an indoor vectored-thrust flight experiment. The strategy is used to stabilize the experiment about one operating point, and improved disturbance rejection and region of attraction are shown compared with a scheduled LQR controller. Issues related to non-zero computation times, choice of horizon length and terminal cost are discussed. **Keywords:** real-time optimization, model predictive control, optimal control, nonlinear control, guidance.

1 Introduction

In receding horizon control, an open-loop trajectory is found by solving a finite-horizon optimal control problem. The controls of this trajectory are then applied for a certain fraction of the horizon length, after which the process is repeated. See [5] for a good review of recent work in this field. This approach has been used in the process control industry successfully for some time, where dynamics are relatively slow, but the computing power required and the tendency for naive implementations to unstabilize a system have prevented its use on fast, stability critical nonlinear systems. Recently, however, theoretical results regarding stability properties of receding horizon controllers and ever increasing computing power have revived interest in the scheme.

The application of Receding Horizon Control to aerial vehicles has been proposed and analyzed by several researchers; [10], for example, provide simulation results for stabilization of a helicopter UAV about an open loop trajectory using receding horizon control. The strategy offers many benefits in this environment, such as the inherent ability to deal with constraints in the state and control. Examples of such constraints commonly encountered include static terrain obstacles, dynamic or pop-up threats and saturations on the actuators. Much of the existing work, however, has not addressed the fact that computation times cannot be ignored with the fast dynamics of most aerial platforms.

*Research supported in part by DARPA

†Electrical and Computer Engineering, University of Colorado, Boulder, CO 80309-0425

‡Control and Dynamical Systems, Mail Code 107-81, California Institute of Technology, Pasadena, CA 91125

The goal of the work in this paper was twofold: first, to address issues of implementation with substantial computation times, and second, to provide a validation of theoretical results through implementation on an actual nonlinear experiment. The optimal control problem is solved using NTG, a software package developed at Caltech [8]; a full aerodynamic model of the flying wing is used in order to demonstrate feasibility on a true UAV platform. A gain-scheduled LQR controller is used for comparison because of its good overall performance. This paper deals exclusively with stability, region of attraction and disturbance rejection properties; use of receding horizon in a maneuvering sense will be examined in a future paper.

The progression of the paper is as follows: section 2 provides theoretical background as well as some motivation for the choices made in terms of timing; section 3 describes in detail the two different timing methods used in the experiment; section 4 describes the actual experiment, and finally sections 6 and 7 show simulation and actual results before concluding.

2 Theoretical Background

This section presents a summary of relevant theory, and attempts to motivate our choices for timing made in the sequel. We hope to provide more concrete mathematical foundations for these timing formulations in a future paper.

We begin by defining a system with an ideal feedback law, looking at it in a sampled data sense and then proposing four different methods of applying the feedback. Next we consider the effects of weakening the idealness of the system, where one method becomes the standard theoretical receding horizon formulation, one method becomes a naive implementation with non-zero runtimes, and the remaining two become candidates for actual implementation. We conjecture that the implementable methods will be stable for short enough runtimes and long enough horizons.

Our system is described by

$$\dot{y} = f(y, u) + g(y, u, w) \tag{1}$$

where $f(\cdot, \cdot)$ is the nominal (i.e., model) system vector field and $g(\cdot, \cdot, \cdot)$ describes the effect of the external disturbance w together with that portion of the system dynamics that is not explicitly modeled. Thus, for the purpose of control design, etc., we will use

$$\dot{x} = f(x, u) \tag{2}$$

as the *model* system.

Now, suppose that

$$u = k(x) \tag{3}$$

is a state feedback that exponentially stabilizes the origin for the nominal system (2) and that $V(x)$ is a quadratic Lyapunov function proving such. For example, $k(\cdot)$ might arise as the solution to an infinite horizon optimization problem with $V(\cdot)$ as the corresponding minimum cost (to go). In the case that

the perturbation is nonzero but can be bounded by a constant, one may use Lyapunov arguments to show that the state of the true closed loop system (1), (3) will converge to a neighborhood of the origin.

Next we construct a sampled data feedback structure such that at every time $t_k := k\delta$ we obtain a measurement $y_k := y(t_k)$. At every time step, we calculate a trajectory $x(\cdot; y_k)$, $u(\cdot; y_k)$ by simulating the closed loop model system (2), (3) for a length of time (either δ or 2δ seconds).

We propose the following four methods for applying the resulting *open loop* input trajectory to the actual system (1):

1. apply $u_{[0,\delta]}(y_k)$ (the control trajectory over the interval $t_{sim} \in [0, \delta]$ resulting from a simulation starting at y_k) over the interval $t \in [t_k, t_{k+1}]$. Note that this option requires that the simulation be run in zero time.

2. apply $u_{[0,\delta]}(y_k)$ over the interval $t \in [t_{k+1}, t_{k+2}]$. Note that this option will always involve a delay.

3. apply $u_{[\delta,2\delta]}(y_k)$ over the interval $t \in [t_{k+1}, t_{k+2}]$.

4. apply $u_{[0,\delta]}(x(\delta; y_k))$ over the interval $t \in [t_{k+1}, t_{k+2}]$. Here $x(\delta; y_k)$ represents the state of the system x starting at y_k simulated ahead δ s.

When the system perturbation is identically zero $g(x, u, w) \equiv 0$, we see that options 1, 3, and 4 will be identical. Options 2, 3, and 4 are all implementable if the simulation computation can be completed in less than δ seconds (i.e., faster than real time). Because option 2 involves a delay (even in the no perturbation case), we propose that 3 and 4 will be the best methods with non-zero runtimes. Clearly the performance of the sampled data system schemes with nonzero perturbation will depend on the sample time δ.

As a next step, suppose that we compute the input trajectory $u(\cdot; y_k)$ by solving the finite horizon optimal control problem

$$J_T^*(y(t_k)) = \min_{u(\cdot)} \int_0^T q(x(\tau), u(\tau))\, d\tau + V(x(T)), \quad \dot{x}(t) = f(x(t), u(t)),\ x(0) = y(t_k)$$

where the incremental cost satisfies $q(x, u) \geq c_q(\|x\|^2 + \|u\|^2)$ with $c_q > 0$. If the terminal cost $V(\cdot)$ is chosen to be a control Lyapunov function (CLF) satisfying $\min_u(\dot{(V)} + q)(x, u) \leq 0$ on a neighborhood of the origin, option 1 (with $g(x, u, w) \equiv 0$) is the receding horizon control scheme $\mathcal{RH}(T, \delta)$, analyzed in [3]. Now allowing $g(x, u, w)$ to be nonzero, we discuss some stability properties of this structure.

As in the stability analysis of unperturbed receding horizon control [3], we will use $J_T^*(\cdot)$ as a Lyapunov function. Roughly speaking, we require that $J_T^*(y_k)$ be a strictly decreasing sequence, ensuring the convergence of the state to a (hopefully small) neighborhood of the origin.

Note that $J_T^*(\cdot)$ is Lipschitz continuous with constant K over the compact region of interest. The properties of $q(\cdot, \cdot)$ and $V(\cdot)$ ensure that

$$J_T^*(x(t_k + \delta)) \leq J_T^*(x(t_k)) - Q_\delta(x(t_k)) \tag{4}$$

where the decrement $Q_\delta(\cdot)$ is a positive definite function (given by integrating the optimal incremental cost over a δ second interval).

Suppose, now, that we apply the same open loop control $u(\cdot)$ (e.g., the just computed optimal $u(\cdot)$) to the real and model systems, (1) and (2), with potentially different initial conditions. By a standard argument (using the Bellman-Gronwall lemma, see [4]), we have

$$\|y(t_k+\delta) - x(t_k+\delta)\| \leq e^{L\delta}\|y(t_k) - x(t_k)\| + \frac{b}{L}\left(e^{L\delta} - 1\right) \tag{5}$$

where b is a bound on $\|g(y(t), u(t), w(t))\|$, $t \in [t_k, t_k+\delta]$, and L is a Lipschitz constant for $f(\cdot,\cdot)$. In our current case (option 1), the initial conditions are equal, $y(t_k) = x(t_k)$, so the above equation reduces to

$$\|y(t_k+\delta) - x(t_k+\delta)\| \leq b\delta + \text{h.o.t.} \tag{6}$$

Combining (4) and (5) and noting that $y(t_k) = x(t_k)$, we obtain

$$\begin{aligned} J_T^*(y(t_k+\delta)) &\leq J_T^*(x(t_k+\delta)) + K\|y(t_k+\delta) - x(t_k+\delta)\| & (7)\\ &\leq J_T^*(y(t_k)) - Q_\delta(y(t_k)) + K\frac{b}{L}(e^{L\delta} - 1)\ . & (8)\end{aligned}$$

For small $\delta > 0$, we can bound the terms on the right according to $Q_\delta(x) \geq \frac{\delta}{2}c_y\|x\|^2$ and $K\frac{b}{L}(e^{L\delta} - 1) \leq 2Kb\delta$. We conclude that, for small δ, $J_T^*(y(t))$ will decrease provided that

$$\|y(t)\|^2 \geq \frac{4Kb}{c_q}\ . \tag{9}$$

This determines the *radius* r_b for an invariant sublevel set of $J_T^*(\cdot)$ to which the state of the true system will converge to under the scheme of option 1. A picture of this is shown in fig. 1.

Finally, we extend this discussion to include situations in which $y(t_k) \neq x(t_k)$, which is the case in options 2, 3, and 4. In this case (5) necessarily contains an exponential (in δ) term multiplied by the error in the initial conditions. Performing analysis similar to that detailed above, we obtain the relation

$$\begin{aligned} J_T^*(y(t_k+\delta)) \leq\ & J_T^*(x(t_k+\delta)) - Q_\delta(y(t_k)) + K\tfrac{b}{L}(e^{L\delta} - 1) \\ & + (K(1+e^{L\delta}) + K_Q)\|y(t_k) - x(t_k)\| \end{aligned} \tag{10}$$

where K_Q is a Lipschitz constant for $Q_\delta(\cdot)$. Clearly, mismatches in the initial conditions lead to performance degredations, including an enlargement of the terminal set (r_b increased) as well as potential destabilization. It is therefore of prime importance to minimize the initial condition mismatch to the extent possible. We conjecture that option 2 does not do a good job of this; indeed, even in the no perturbation case such an error is induced by delay. Accordingly, we study options 3 and 4 both in simulation and experimentally on the physical system, and attempt to determine a range of parameters horizon T and period δ providing a desired level of performance.

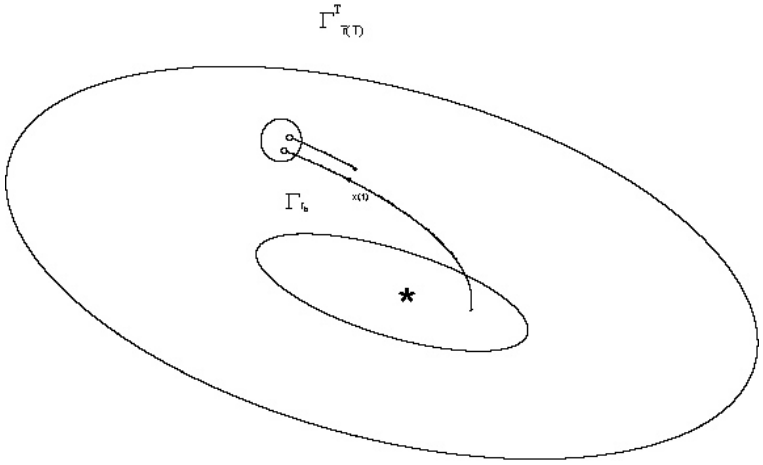

Figure 1: Illstration of sublevel set of $J_T^*(\cdot)$ to which the state of the true system will converge

A final note is meant to justify the use of a "fast as possible" timing scheme, whereby δ is taken as the last computation time and thus is not constant. The reference [3] provides as a result the stability of $\mathcal{RH}(T, \{\delta_k\})$, where $0 \leq \delta_k \leq T$ and $\lim_{l\to\infty} \sum_{k=0}^{l} \delta_k = \infty$.

3 Timing and Optimization Formulation

In some applications of receding horizon, runtimes are insignificant compared to the dynamics of the system. This is not the case on most aerial platforms with current computing power. On our hardware we were able to achieve runtimes between $0.1s$ and $0.2s$ in most cases; we ordinarily run linear controllers at a minimum of $50Hz$. Because of this, the preceding discussion is crucial. Before detailing our timing implementations, we first state explicitly the optimization formulation used in this paper:

$$\min_{x(\cdot),u(\cdot)} \int_0^T x_{err}^T(\tau) Q x_{err}(\tau) + u_{err}^T(\tau) R u_{err}(\tau)\, d\tau + x^T(T) P x(T) \tag{11}$$

subject to

$$x(0) = x_0, u(0) \leq c$$

with $x_{err}(t) = x(t) - x_{ref}(t)$ and $u_{err}(t) = u(t) - u_{ref}(t)$, $x_{ref}(t)$ and u_{ref} constant.

We choose Q and R to be the same as weights used in the past to generate LQR gains with good performance, and P to be the corresponding solution to

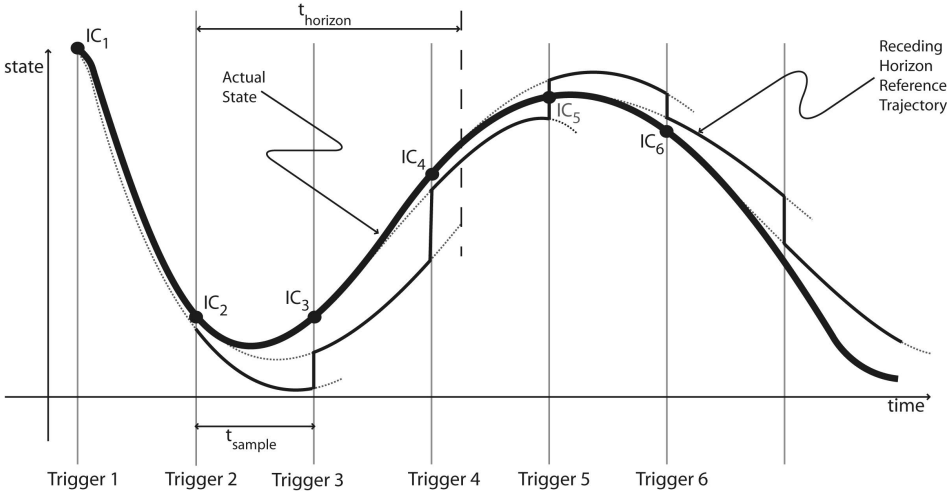

Figure 2: Illustration of timing scheme without prediction

the algebraic Riccati equation resulting in a CLF terminal cost around a certain operating point. The choice of $x(0)$ above is dictated by the choice of timing scheme. We use two different strategies, corresponding to options 3 and 4 above, for choosing these initial constraints. Sections 3.1 and 3.2 describe these further. In addition to these constraints, we can also choose initial constraints on the accelerations and the controls; this is described in section 3.3. Another issue in timing involves how we deal with trajectory computations which fail for some reason; section 3.4 details our solution to this.

3.1 Option 3: No Prediction

The first scheme for choosing the initial constraints in the state is the simplest, as it involves no model prediction. Whenever a computation is triggered, the current state of the system is given as the initial constraint on the state trajectory for the optimization problem. By the time the computation is finished t_{sample} s later, however, the idea is that the system has changed significantly. To attempt to use a valid control, we simply discard the first t_{sample} s of the trajectory, hoping that the resulting start point will coincide roughly with where we were in the previous trajectory. Fig. 2 shows graphically how this process works on one of the states. In this case, the controls corresponding to the line labelled "Receding Horizon Reference Trajectory" are applied to the system. Some of our results apply these controls open-loop, while at other times we apply V_m open-loop while applying a PD feedback rule with θ and $\dot{\theta}$ to compute the δp used. Note that the figure exaggerates certain things for illustration–for example, the horizon length $t_{horizon}$ is in reality much longer than t_{sample}. As with any timing scheme, there are necessarily discontinuities in the resulting control, although a short enough period and an accurate model will minimize

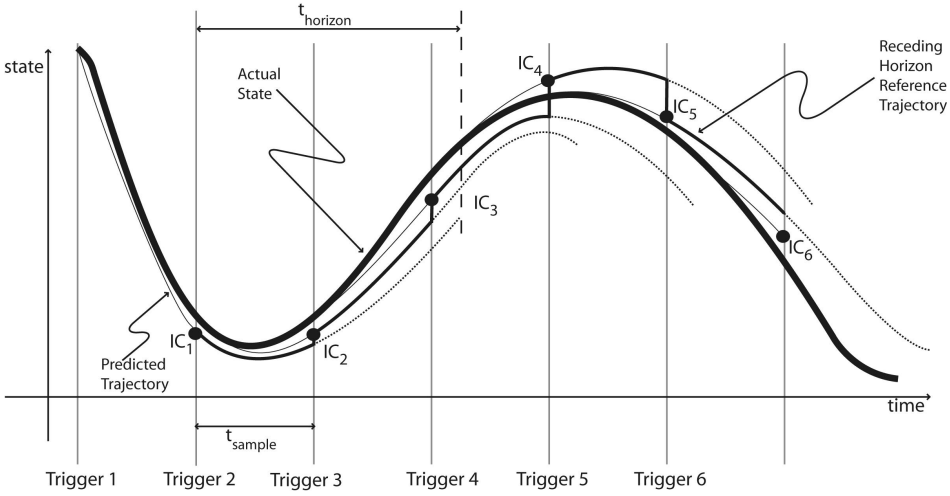

Figure 3: Illustration of timing scheme with prediction

the jumps.

In our implementation, t_{sample} can either be set to some constant, or the computations can be run as "fast as possible", meaning a new computation is triggered immediately after the last one has finished. In this case, t_{sample} varies with the runtime.

3.2 Option 4: With Prediction

The second scheme we examine attempts to minimize discontinuities by using prediction. When a computation is triggered, the current state of the fan is first used as the initial condition for a simulation in which the control trajectory of the previous computation is used as input. This simulation is run for some amount of time t_{sim}; if a fixed period is being used, t_{sim} is simply equal to the t_{sample}, but if a "fast as possible" rule is used, t_{sim} is taken as an average of the past n runtimes. After the simulation is completed, the final values are passed as the initial constraints to the optimization. The resulting trajectory is output from the beginning. Fig. 3 shows this process graphically. Again, the controls corresponding to the line labelled "Receding Horizon Reference Trajectory" are applied to the system, and can be used either open-loop or with feedback around θ and $\dot{\theta}$.

3.3 Initial Constraints on State Derivatives and Forces

A characteristic of the spline representation used to solve the optimal control problem is that, between enforcement points, the values of the states, their derivatives and the controls may not be consistent with the equations of motion

for the system. Because of this, a point on the trajectory is in general not suitable as an initial equality constraint for a successive computation. Nevertheless, experience showed us that some sort of effort in minimizing large jumps in at least the forces is worthwhile. To deal with this, we introduce a degree of freedom on the accelerations by eliminating their initial constraints. We are most interested in minimizing jumps in the controls, so we enforce an inequality constraint $|u_{k+1}(0) - u_k(t_{sample})| < a$ for some a. If a fixed period is used, t_{sample} is simply equal to the period, but if a "fast as possible" rule is used, t_{sample} is taken as an average of the past n runtimes. This approach is compatible with both timing schemes discussed above; graphically, control trajectories always start near the previous trajectory.

3.4 Non-Convergent Trajectory Computations

Unfortunately, not all trajectory computations are guaranteed to converge. Each computation is given the last computed trajectory as an initial guess, which is sometimes not good enough; also, some combinations of initial constraints and cost function are simply degenerate. If a computation returns certain signs of failure, the last good trajectory is simply continued and another computation is triggered. This will certainly fail if a valid trajectory is not computed before the

4 Experimental Setup

The right diagram in fig. 4 shows an overview of the Caltech ducted fan. The experiment consists of a vertical stand and a horizontal boom which holds the actual wing. This setup enables flight on a cylinder of height 2.5 m and radius 2.35 m. Because of a mass of 12.5 kg and a maximum thrust of only 15 N, a counterweight is attached to the boom via a cable and pulleys which reduces the effective gravity to achieve $mg_{eff} = 7\ N$. This allows the system to attain sizable vertical accelerations, while minimizing the force of potential crashes. Mechanical brakes in the vertical direction are used as well to aid in crash landings. Actuation of the ducted fan is accomplished in two ways: by controlling the speed of the propellor mounted inside the cowl, and by vectoring the resulting thrust via a servo controlled bucket.

The left diagram in fig. 4 shows more explicitly the inertial coordinate frame used in this paper. In this frame, the axes are fixed to the ground, and the x and z directions represent horizontal and vertical inertial translations. θ represents the rotation of the ducted fan about the boom axis. All three of these variables are measured via rotary encoders, and the resulting signals are routed to the computing platform via sliprings.

The equations of motion are given by

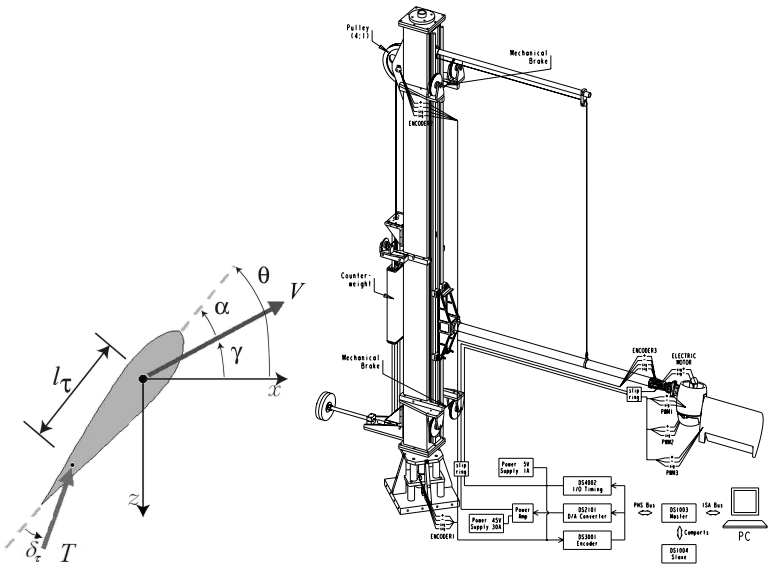

Figure 4: Ducted Fan Schematic Diagram

$$\begin{aligned} m\ddot{x} + F_{X_a} - F_{X_b}\cos\theta - F_{Z_b}\sin\theta &= 0 \\ m\ddot{z} - mg_{eff} + F_{Z_a} + F_{X_b}\sin\theta - F_{Z_b}\cos\theta &= 0 \\ J\ddot{\theta} - M_a + \tfrac{1}{r_s} I_p \Omega \dot{x}\cos\theta - F_{Z_b} r_f &= 0 \end{aligned} \tag{12}$$

where

$$F_{X_a} = D\cos\gamma + L\sin\gamma,\ F_{Z_a} = -D\sin\gamma + L\cos\gamma$$

are the aerodynamic forces. See [7] for a complete derivation of these equations. We chose a spatial represention of the equations of motion in order that we can consider both hover and forward flight modes. F_{X_b} and F_{Z_b} are thrust vectoring body forces; $I_p = 2e^{-5}\ kg\ mm^2$ and $\Omega = 1300\ rad/s$ are the moment of inertia and angular velocity of the ducted fan propeller, respectively. $J = .25\ kg\ m^2$ is the moment of inertia of the ducted fan about the boom, and $r_f = .35\ m$ is the distance from center of mass along the X_b axis to the effective application point of the thrust vectoring force. The angle of attack α is related to the pitch angle θ and the flight path angle γ by

$$\alpha = \theta - \gamma.$$

The flight path angle can be derived from the spatial velocities by

$$\gamma = \arctan\frac{-\dot{z}}{\dot{x}}.$$

The lift (L) ,drag (D), and moment (M) are given by

$$L = qSC_L(\alpha), D = qSC_D(\alpha), \text{ and } M = \bar{c}SC_M(\alpha),$$

respectively. The dynamic pressure is given by $q = \frac{1}{2}\rho V^2$. The norm of the spatial velocity is denoted by V and ρ is the atmospheric density. Figure (5) depicts the coefficients of lift ($C_L(\alpha)$) and drag ($C_D(\alpha)$) and the moment coefficient ($C_M(\alpha)$). These coefficients were determined from a combination of wind tunnel and flight testing.

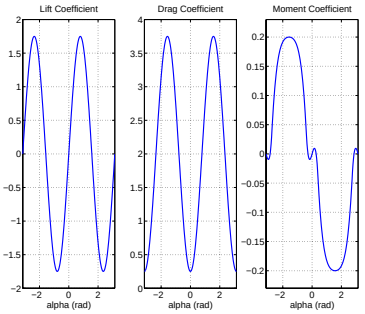

Figure 5: B-spline curve fits to wind tunnel and flight test results for the $C_L(\alpha)$,$C_D(\alpha)$, and $C_M(\alpha)$ aerodynamic coefficients

5 Trajectory Generation Methodology

There are three components to the trajectory generation methodology we propose. The first is to determine a parameterization (output) such that Equation (12) can be mapped to a lower dimensional space (output space). [1] gives information on finding this mapping if the system if flat. The idea is to map dynamic constraints to algebraic ones. Once this is done the cost and constraints can also be mapped to the output space. The second is to parameterize each component of the output in terms of an appropriate B-spline polynomial. Finally, sequential quadratic programming is used to solve for the coefficients of the B-splines that minimize the cost subject to the constraints in output space. See [6, 9] for more details on this approach. The NTG software package is an implementation of this concept. The user provides the cost and the constraints in terms of the outputs and their derivatives as well as the Jacobian of the cost and constraints with respect to each output and the maximum derivative that occurs in each output.

By using this methodology, we can sufficiently reduce the dimension of the nonlinear programming problem to make real-time computation possible. For our system we will choose as outputs $z_1 = x(t)$, $z_2 = z(t)$, $z_3 = \theta(t)$ in solving

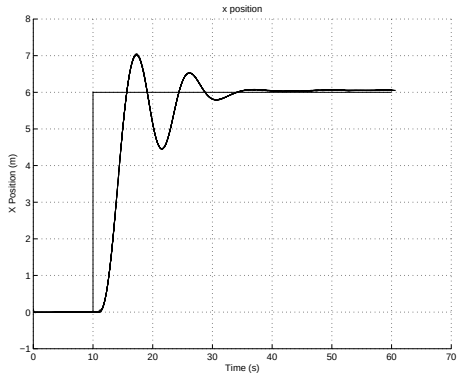

Figure 6: Simulation results in x to a 6 m step

the problem posed in eq. 11. Given these outputs, their derivatives and the control trajectory can be computed easily.

6 Simulation Study

Because of the complex implementation involved in this experiment, a simulation was built in tandem with the real implementation which was functionally equivalent except for the use of a model instead of the real system. The simulation allows us to explore many different configurations without fear of damaging the hardware. Table 1 shows results of identifying the highest acceptable periods for different combinations of timing mode, horizon length and theta controller. The test used for these results was a 20 m step in x, a fairly demanding request which puts the fan into a forward flight state to test out the full features of the model. Acceptable results were chosen as stable and with few qualitative difference from the best results. The theta controller is simply a PD loop around computed theta trajectories which proved necessary in some configurations of the experiment because of theta's fast dynamics.

An interesting feature of these results is that the theta controller significantly improves the performance of the non-prediction timing scheme, while it has a negative effect on the prediction timing system. We attribute this to the fact that the prediction mode relies on knowing the exact inputs to the model as given by the last computation, while the theta controller changes those inputs.

In this range of horizon values, runtimes are consistently between 0.05 s and 0.35 s. Accordingly, we anticipated a much greater chance of success with greater horizon lengths.

Fig. 6 shows the response in x of the simulated system to a 6 m step using no θ controller, no prediction, 1 s horizon and $\delta = 0.2$ s.

	predict		no predict	
horizon	w/o θ	w/ θ	w/o θ	w/ θ
$1.0s$	$0.4s$	$0.1s$	$0.15s$	$0.2s$
$1.5s$	$0.5s$	$0.1s$	$0.2s$	$0.4s$
$2.0s$	$0.65s$	$0.2s$	$0.3s$	$0.7s$
$2.5s$	$0.6s$	$0.45s$	$0.4s$	$0.6s$

Table 1: Maximum acceptable periods as determined in simulation

7 Results

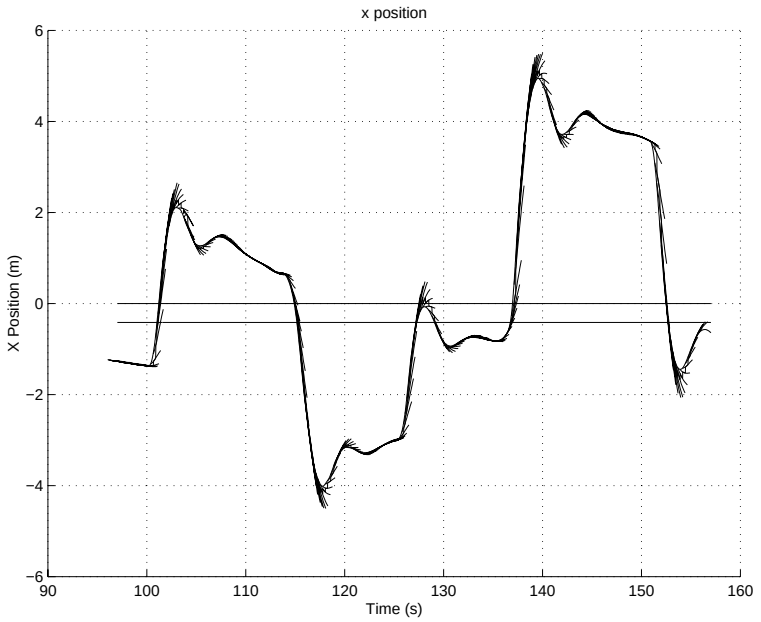

Figure 7: x response to a series of disturbances; $T = 1$ s, no prediction

First we show disturbance rejection and region of attraction properties. Fig. 7 and 8 show x and z responses when the fan is given a serious of rather violent shoves. The receding horizon trajectories can be seen branching off of the curve, especially during periods of high accelerations. The constant reference value is shown as the horizontal line in both plots. The setup used is as follows: a horizon of 1.0 s, no prediction timing mode, "fast as possible" period δ and the θ controller is turned on. Periods between disturbances are most easily seen in the z plot as times when the fan stays fairly constant at zero. The corresponding portions of the x data show that the system can tolerate offstes in x which is to

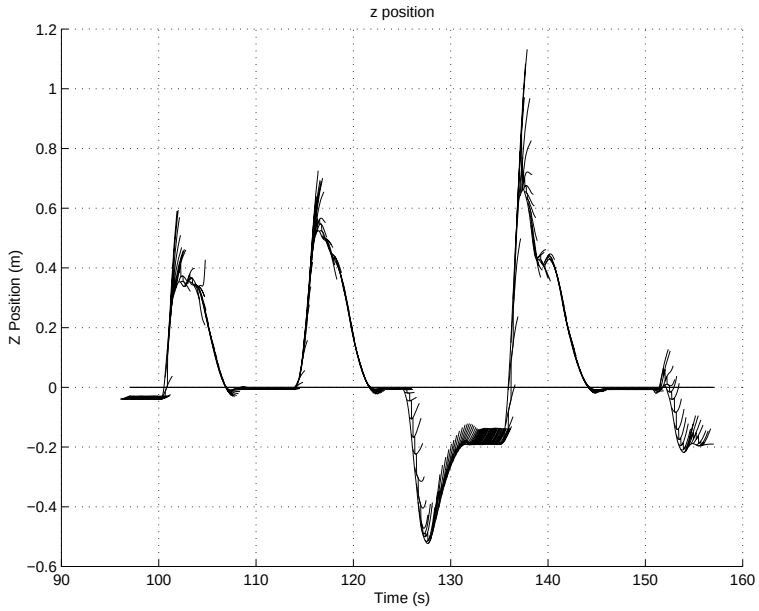

Figure 8: z response to a series of disturbances; $T = 1\ s$, no prediction, w/ θ controller

be expected given the weights Q and R which we chose. From these results we see that the system can recover from fairly strong disturbances, also indicating a large region of attraction. Results of the same tests using a gain scheduled LQR controller are not even worth showing, as the controller could not recover and the system crashed.

Next we show the response to a 6 m step in x, analagous to the simulation shown in fig. 6. Figs. 9 and 10 show the x response using the receding horizon controller and the gain scheduled LQR controller, respectively. The receding horizon controller uses the same parameters as in the above disturbance rejection setup. The response using the gain scheduled LQR controller is similar, but we add that the fan lost more altitude with the LQR controller.

In terms of studying interactions between timing scheme, horizon length and sample period δ, we are currently perfecting the implementation to allow for a detailed investigation. Future revisions of this paper will include results of this investigation.

8 Conclusion

The theoretical discussion in section 2 provides a good framework for evaluating in a qualitative sense different choices for choosing a timing scheme to deal with non-zero runtimes. In particular, two timing paradigms are chosen which seem

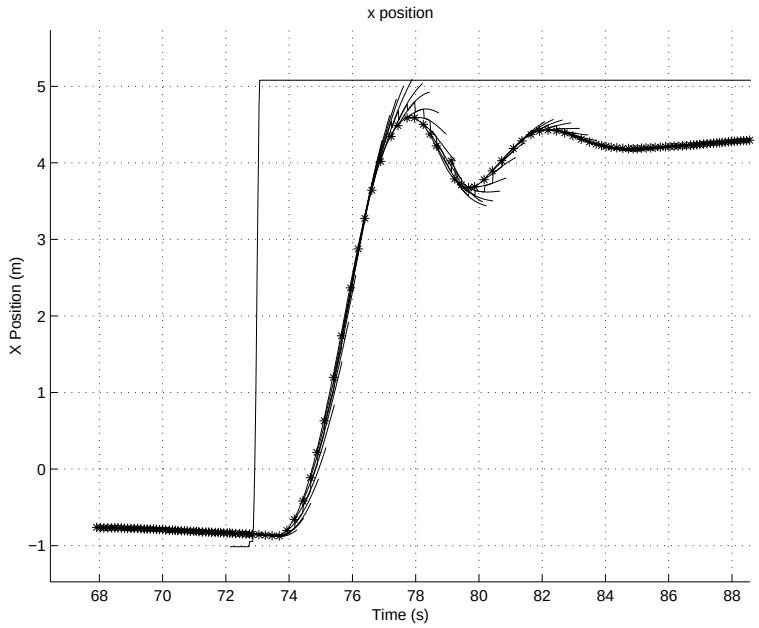

Figure 9: x response to a 6 m step input; $T = 1$ s, no prediction, w/ θ controller

reasonable. Simulation is used to compare and contrast different combinations of timing mode, horizon length and sample period δ. Implementation on the real system, while preliminary, is very encouraging and shows improved disturbance rejection, region of attraction and step response in comparison to a gain scheduled LQR controller. We believe this is a good validation of the theory involved.

Future work will include a comprehensive study of performance using different timing modes, horizons and sample periods. In addition, we seek stronger mathematical results which could indicate theoretically which timing mode is better. Another intriguing possibility is to extend the prediction timing mode by feeding the optimization routine constant updates of what the best predicted state is. The predicted state would increase in accuracy while not changing the resulting optimization solution greatly.

References

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

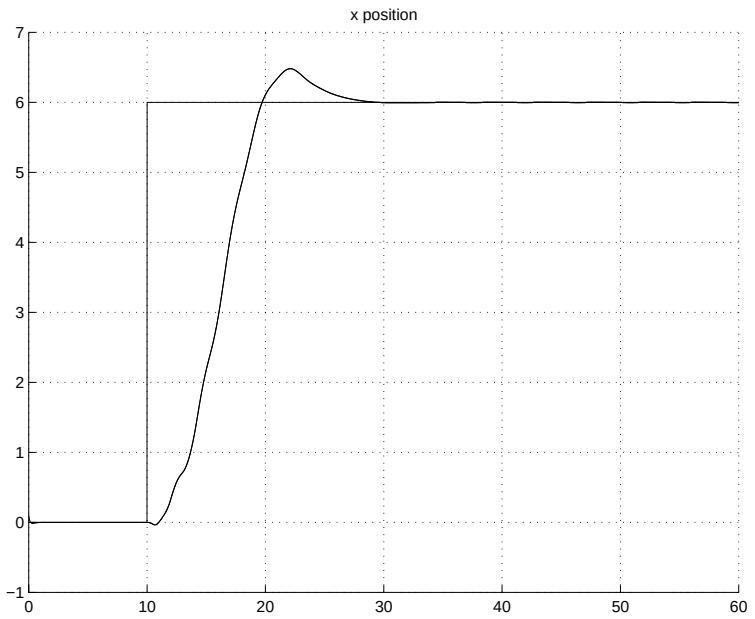

Figure 10: x response to a 6 m step input; $T = 1$ s, no prediction, w/ θ controller

[2] Ali Jadbabaie and John Hauser. On the stability of unconstrained receding horizon control with a general terminal cost. In *Submitted, Systems and Control Letters*, 2001.

[3] Ali Jadbabaie, Jie Yu, and John Hauser. Unconstrained receding-horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46:776–783, 2001.

[4] Hassan Khalil. *Nonlinear Systems*. Prentice Hall, Upper Saddle River, NJ, 1996.

[5] D.Q. Mayne, J. B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

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

[7] Mark Milam and R.M. Murray. A testbed for nonlinear flight control techniques: The caltech ducted fan. In *Conference on Control Applications*, 1999.

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

[9] N. Petit, M. B. Milam, and R. M. Murray. Inversion based constrained trajectory optimization. In 5*th IFAC symposium on nonlinear control systems*, 2001.

[10] Leena Singh and James Fuller. Trajectory generation for a uav in urban terrain, using nonlinear mpc. In *Proceedings of the American Control Conference*, 2001.