2005 American Control Conference
June 8-10, 2005. Portland, OR, USA

Flight Test of a Receding Horizon Controller for Autonomous UAV Guidance

Tamás Keviczky and Gary J. Balas*

Abstract— **This paper describes autonomous unmanned aerial vehicle (UAV) guidance technologies developed and demonstrated in a flight test sponsored by the DARPA Software Enabled Control program. The flight experiment took place in June 2004 using a Boeing UAV testbed and demonstrated important autonomy capabilities enabled by a receding horizon guidance controller and fault detection filter. These technologies were implemented using an application programming interface developed for real-time receding horizon control applications (RHC API) under Boeing's Open Control Platform (OCP) embedded software environment.**

This paper describes the receding horizon controller (RHC) design process and demonstration scenarios which were designed to exercise and evaluate the primary functionalities of the control system. Simulation results of the key capabilities are shown and compared with recorded flight data for evaluation purposes.

I. Introduction

Interest in the use of unmanned aerial vehicles (UAVs) has grown significantly in the last decade. Currently most UAVs are used for military surveillance and reconnaissance in practice but the range of possible civilian applications is also promising. Current UAV systems are either remotely operated by a human pilot or rely on rudimentary guidance technologies that limit the number of vehicles used and the flexibility needed to accomplish complex mission tasks as outlined in [1].

One of the main technological goals of improving the capabilities of currently operational UAVs is to increase their autonomy by providing more advanced guidance systems. In particular, to enable the application of advanced control design methods that respect constraints of the vehicle dynamics and allow systematic design of contingencies (e.g. reconfiguration) in case a fault occurs in the system. These technologies are essential elements of a high-performance, high-risk autonomous aerial vehicle system.

Advances in software and computational power have provided the basis that is critical in implementing state-of-the-art control algorithms. To this date, very few advanced guidance and fault detection concepts have been evaluated in flight tests on actual vehicles.

The technologies described in this paper were developed and implemented as part of the Defense Advanced Research Projects Agency (DARPA) Software Enabled Control (SEC) program [2]. The overall goals of the project included development of the Open Control Platform (OCP) middleware [3] that enables seamless integration of control algorithms in real-time embedded environments, and demonstration of advanced guidance capabilities on a full-scale aircraft equipped with actual future UAV avionics and software.

The paper is organized as follows. First, a brief description of Boeing's UAV testbed is given in Section II, followed by the receding horizon guidance controller design in Section III. Section IV explains the experimental scenario that formed the basis of the flight demonstration. High fidelity simulation results and flight test data are presented in Section V.

II. UAV testbed and the OCP software environment

The aircraft used for the flight test experiments is a modified T-33 two-seat jet trainer equipped with components of the Boeing X-45 unmanned aerial vehicle avionics and an autopilot. The autopilot provides several control functionalities such as altitude, velocity, heading and turn-rate tracking. These different command types and functionalities are accessible through Boeing's Open Control Platform (OCP) [3]. The OCP served as a software integration framework for flight code development and desktop simulations as well, besides hosting and interfacing the final flight code implementation.

A standard laptop computer served as the on-board, real-time flight controller hosting the OCP and the guidance algorithms. The guidance system produces the velocity, turn-rate and altitude commands of the aircraft to follow a reference trajectory.

In the testing and control design phase, simulations were performed using a nonlinear black-box executable model of the T-33 aircraft integrated with the autopilot provided by Boeing. This open vehicle simulation model represents the nonlinear closed-loop aircraft dynamics and will be referred to as DemoSim.

The RHC-based guidance system was designed based on identified linear time-invariant closed-loop vehicle dynamics using DemoSim and the techniques presented in [4]. The nonlinear operational constraints of the vehicle were approximated by linear expressions. Reference [5] contains a detailed description of the identification process.

A. DemoSim modeling

The inputs to the identified LTI DemoSim model are the following command signals: ground speed command V_{cmd} (i.e. total velocity w.r.t. ground), turn rate command

* Department of Aerospace Engineering and Mechanics, University of Minnesota, 107 Akerman Hall, 110 Union Street S.E., Minneapolis, MN 55455, {keviczky,balas}@aem.umn.edu

This work was funded by the Defense Advanced Research Projects Agency under the Software Enabled Control program, Dr. John Bay Program Manager. The contract number is USAF/AFMC F33615-99-C-1497, Dale Van Cleave is the Technical Contract Monitor.

$\dot{\chi}_{cmd}$ and altitude rate command $\dot{h}_{cmd}$. The model outputs are ground speed V, heading χ and flight path angle γ. The variables ζ and η will denote the local north and east coordinates, respectively. The linear DemoSim dynamics was identified at the following input-output trim values: $V_{\text{DStrim}} = 505$ ft/s, $\dot{\chi}_{\text{DStrim}} = 0$ rad/s, $\dot{h}_{\text{DStrim}} = 0$ ft/s, $\chi_{\text{DStrim}} = \frac{\pi}{2}$ rad, $\gamma_{\text{DStrim}} = 0$ rad.

B. Prediction model

The prediction model was chosen to accommodate two important requirements. It has to provide a reasonably accurate description of the dynamic relationship between the control inputs of the test platform and the output signals of interest, which include position coordinates for tracking performance and other variables used for describing maneuvering constraints. At the same time, it has to be simple enough to limit the complexity of the optimization problem that is solved online in a receding horizon fashion.

These objectives were captured by constructing the prediction model from the identified LTI DemoSim dynamics and a flat-earth kinematic model. Using linearized kinematics that were updated at every time step, a discrete-time linear time-invariant prediction model was derived. Although this model was fixed throughout the prediction horizon of the optimization problem, the updates to the linearized kinematics part rendered the prediction model a linear parameter-varying system, which depended on the current velocity, heading and flight path angle values.

The continuous-time nonlinear prediction model, comprised of the LTI dynamics and the nonlinear kinematics, is depicted in Figure 1. The tilded variables represent deviations from trim values of the DemoSim model:

$$\left[V, \dot{\chi}, \dot{h}\right]_{cmd} = \left[V, \dot{\chi}, \dot{h}\right]_{\text{DStrim}} + \left[\tilde{V}, \tilde{\dot{\chi}}, \tilde{\dot{h}}\right]_{cmd}, \tag{1a}$$

$$\left[V, \chi, \gamma\right] = \left[V, \chi, \gamma\right]_{\text{DStrim}} + \left[\tilde{V}, \tilde{\chi}, \tilde{\gamma}\right]. \tag{1b}$$

A schematic diagram of the linearized prediction model is shown in Figure 2. Note the nonlinear kinematics are linearized around fixed V_0, χ_0, γ_0 values that represent current measurements. The inputs fed into the linearized kinematics model represent the differences between the true values predicted by the LTI DemoSim model (after addition of trim values) and the current measurements used for linearization:

$$\left[\tilde{V}, \tilde{\chi}, \tilde{\gamma}\right]_0 = \left[V, \chi, \gamma\right]_{\text{DStrim}} + \left[\tilde{V}, \tilde{\chi}, \tilde{\gamma}\right] - \left[V, \chi, \gamma\right]_0 \tag{2}$$

The linearized model was discretized using the discrete-time identified DemoSim dynamics and approximating the continuous-time nonlinear kinematics around fixed V_0, χ_0, γ_0 values with the following forward-Euler discretized linear system

$$\begin{bmatrix} \tilde{\zeta}(k+1) \\ \tilde{\eta}(k+1) \\ \tilde{h}(k+1) \end{bmatrix} = \tilde{A} \begin{bmatrix} \tilde{\zeta}(k) \\ \tilde{\eta}(k) \\ \tilde{h}(k) \end{bmatrix} + \tilde{B} \begin{bmatrix} \tilde{V}_0(k) \\ \tilde{\chi}_0(k) \\ \tilde{\gamma}_0(k) \end{bmatrix} \tag{3}$$

where $\tilde{A} = I_3$ and

$$\tilde{B} = T_s \cdot \begin{bmatrix} C_{\chi_0} C_{\gamma_0} & -V_0 S_{\chi_0} C_{\gamma_0} & -V_0 C_{\chi_0} S_{\gamma_0} \\ S_{\chi_0} C_{\gamma_0} & V_0 C_{\chi_0} C_{\gamma_0} & -V_0 S_{\chi_0} S_{\gamma_0} \\ S_{\gamma_0} & 0 & V_0 C_{\gamma_0} \end{bmatrix} \tag{4}$$

In equation (4) the sampling time T_s is 0.5 seconds and $C_{(\cdot)}$, $S_{(\cdot)}$ are short notation for $\cos(\cdot)$ and $\sin(\cdot)$, respectively.

The addition of DemoSim trim values and the subtraction of the current measurement values at the output of the LTI DemoSim model were implemented by augmenting the LTI dynamics with extra states, which were subtracted from the model outputs. The state values were updated every time step based on the difference between DemoSim trim values and current measurements. Denoting the original discrete-time LTI DemoSim matrices with $A_{\text{DS}}, B_{\text{DS}}, C_{\text{DS}}, D_{\text{DS}}$ and using $A_{\text{DS0}}, B_{\text{DS0}}, C_{\text{DS0}}, D_{\text{DS0}}$ to denote the augmented dynamics that adjusts the output values based on current measurements (V_0, χ_0, γ_0), leads to

$$A_{\text{DS0}} = \begin{bmatrix} A_{\text{DS}} & 0 \\ 0 & I_3 \end{bmatrix}, \quad B_{\text{DS0}} = \begin{bmatrix} B_{\text{DS}} \\ 0 \end{bmatrix}, \tag{5}$$

$$C_{\text{DS0}} = \begin{bmatrix} C_{\text{DS}} & -I_3 \end{bmatrix}, \quad D_{\text{DS0}} = D_{\text{DS}}.$$

The estimated time-delays associated with the pilot model and data processing by the on-board avionics were accounted for by augmenting the discrete-time DemoSim dynamics with extra states to arrive at matrices A_d, B_d, C_d, D_d. The command input time-delays were characterized as integer multiples of the sampling time T_s.

Since the nonlinear kinematics part of the prediction model was always linearized around the current measurements of V_0, χ_0, γ_0, the outputs of this augmented, modified LTI DemoSim model could be fed directly into the linearized kinematics model. In other words, the complete linear prediction model could be obtained by the following augmentation of the state-space matrices

$$A = \begin{bmatrix} A_d & 0 \\ \tilde{B} C_d & \tilde{A} \end{bmatrix}, \quad B = \begin{bmatrix} B_d & 0 \\ \tilde{B} D_d & 0 \end{bmatrix}, \tag{6}$$

$$C = \begin{bmatrix} 0 & I_3 \end{bmatrix}, \qquad D = \begin{bmatrix} 0 & I_3 \end{bmatrix}.$$

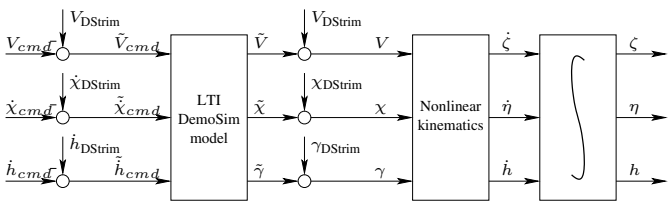


Fig. 1. Nonlinear prediction model.

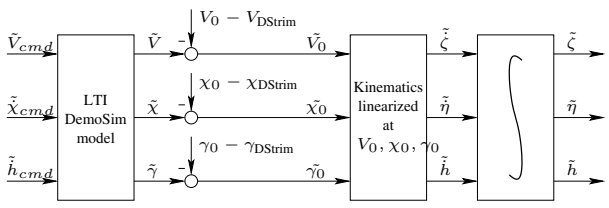


Fig. 2. Linearized prediction model.

Additional disturbance inputs were added to (6), to model additive output disturbance that affects the plant.

The states associated with the DemoSim dynamics "part" of the prediction model were updated using a linear time-invariant observer that relied on the control inputs sent from the RHC controller to the plant $(V_{cmd}, \dot{\chi}_{cmd}, \dot{h}_{cmd})$ and measurements of ground speed V, heading angle χ and flight path angle γ. Note the flight path angle could not be measured directly on the test platform, so a conversion from ground speed and altitude rate measurements was performed to obtain flight path angle according to $\gamma = \arcsin \frac{\dot{h}}{V}$. The input-output scheme of the RHC controller and the observer are shown in Figure 3.

The T-33 test platform avionics provides GPS position measurements in terms of latitude λ, longitude Λ and altitude h using the WGS-84 system. These measurements were converted to north ζ, east η and altitude h values to be compatible with the coordinate frame of the reference trajectory. The north and east coordinates were obtained by the transformation of the geodetic measurements (GPS) into an NEU (north-east-up) local Cartesian coordinate frame that had its origin at the point in space where the RHC controller is engaged $(\lambda_0, \Lambda_0, h_0)$. The geodetic GPS altitude measurements however were used directly, without any further conversion. The states associated with the position integrators of the linearized prediction model were always initialized at zero (i.e. current position).

The output signals of the entire linear prediction model were assigned to three objective groups denoted by u, z and y. These correspond to hard actuator or other input constraints, maneuvering limits and tracking performance, respectively. The commanded input signals are denoted by r. The prediction model in (6) has only y outputs for tracking performance.

$$y = \begin{bmatrix} \tilde{\zeta} & \tilde{\eta} & \tilde{h} \end{bmatrix}^T, \qquad z = A_z \begin{bmatrix} V & \dot{\chi} & \dot{\gamma} \end{bmatrix}^T, \quad (7)$$
$$u = r = \begin{bmatrix} \tilde{V} & \tilde{\dot{\chi}} & \tilde{\dot{h}} \end{bmatrix}^T_{cmd}, \quad \Delta r = \begin{bmatrix} \Delta\tilde{V} & \Delta\tilde{\dot{\chi}} & \Delta\tilde{\dot{h}} \end{bmatrix}^T_{cmd}.$$

Note that outputs z used to define maneuvering limits were not included in the flight code. These output constraints were implemented only for testing in high-fidelity simulations, which are described in more detail in [5], including the definition of A_z, b_z linear constraint parameters.

The parameter-variance of the prediction model, due to linearization of nonlinear kinematics, is characterized by the nominal velocity V_0, heading χ_0 and flight path angle γ_0, around which the kinematic model was linearized. The parameter dependence is present in the $\tilde{B}$ matrix of the linearized kinematics model (4). Denoting the vector of parameters with $\varrho(k) = [V_0(k) \;\; \chi_0(k) \;\; \gamma_0(k)]^T$, the linearized discrete-time prediction models have the form

$$x(k+1) = A_k x(k) + B_k r(k) \quad (8)$$
$$\begin{bmatrix} y(k) & z(k) & u(k) \end{bmatrix}^T = C_k x(k) + D_k r(k)$$

where the parameter dependency of the prediction model is indicated by the subscript k, meaning $A_k = A(\varrho(k))$, $B_k = B(\varrho(k))$, $C_k = C(\varrho(k))$, $D_k = D(\varrho(k))$.

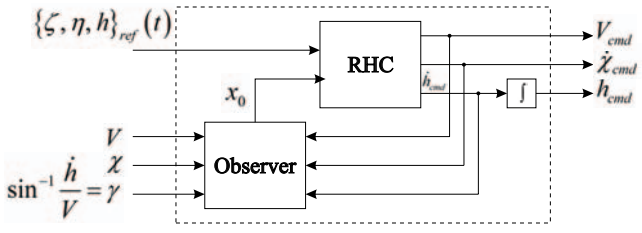


Fig. 3. RHC controller and observer.

III. RHC-BASED GUIDANCE DESIGN

The objective of the RHC control design was to track a time-stamped three-dimensional position reference trajectory in the presence of constraints that limit the maneuverability of the aircraft based on the identified guidance-level vehicle model described in the previous section. Vehicle guidance, trajectory tracking had to be performed while explicitly accounting for the actual vehicle maneuvering capabilities such as flight-envelope and dynamics constraints.

A. Problem formulation

The position reference trajectory was specified in terms of north, east, and altitude coordinates in a local NEU coordinate frame relative to the point in space where the controller is engaged. The reference position vector elements were placed 0.5 seconds apart from each other in time. Denote the reference position trajectory values by $\zeta_{ref}(k), \eta_{ref}(k), h_{ref}(k)$ at time step k. The LTI prediction model based RHC controller was required to track a "linearized" position reference trajectory $\tilde{\zeta}_{ref}(k), \tilde{\eta}_{ref}(k), \tilde{h}_{ref}(k)$. This was generated by subtracting the simulated output of the nonlinear kinematics from the original reference trajectory. The "nominal" simulated trajectory was calculated using fixed V_0, χ_0, γ_0 values based on the current measurements used for linearization. The linear reference trajectory was therefore obtained by

$$\left[\tilde{\zeta}, \tilde{\eta}, \tilde{h}\right]_{ref}(k) = \left[\zeta, \eta, h\right]_{ref}(k) - \left[\zeta, \eta, h\right]_0(k) \quad (9)$$

where "nominal" simulated trajectories were calculated by

$$\begin{bmatrix} \zeta_0 \\ \eta_0 \\ h_0 \end{bmatrix}(k) = \sum_{i=1}^{k} T_s \cdot \begin{bmatrix} \dot{\zeta}_0 \\ \dot{\eta}_0 \\ \dot{h}_0 \end{bmatrix}(V_0, \chi_0, \gamma_0). \quad (10)$$

The state values that represent the constant additive output disturbance in the prediction model were updated every time step based on the following disturbance filter

$$d(k+1) = 0.99 d(k) + 0.01 d_{in}(k) \quad (11)$$

where the input $d_{in}(k)$ was determined from the following error equation

$$d_{in}(k) = \underbrace{\begin{bmatrix} \zeta(k) \\ \eta(k) \\ h(k) \end{bmatrix}}_{\text{pos. meas.}} - \underbrace{\begin{bmatrix} \zeta_0(k|k-1) \\ \eta_0(k|k-1) \\ h_0(k|k-1) \end{bmatrix}}_{\text{nom. pos. pred.}} - \underbrace{\begin{bmatrix} \tilde{\zeta}(k|k-1) \\ \tilde{\eta}(k|k-1) \\ \tilde{h}(k|k-1) \end{bmatrix}}_{\text{pred. lin. output}} \quad (12)$$

where $\zeta_0(k|k-1), \eta_0(k|k-1), h_0(k|k-1)$ are one-step ahead position predictions based on the nonlinear kinematics model and $V_0(k-1), \chi_0(k-1), \gamma_0(k-1)$ measurements at time $k-1$.

The optimization problem setup is based on the linear MPC formulation of [6] with some modifications. In most linear predictive controllers, the performance is specified by the following quadratic cost function to be minimized, which will also be adopted here:

$$J(k) = \sum_{i=1}^{H_p} \left\| \hat{y}\left(k+i \mid k\right) - y_{ref}\left(k+i \mid k\right) \right\|_Q^2 + \\ + \sum_{i=0(\delta H_c)}^{H_c-1} \left\| \Delta r\left(k+i \mid k\right) \right\|_R^2 + \rho\varepsilon \tag{13}$$

where $\hat{y}\left(k+i \mid k\right)$ is the i-step ahead prediction of the outputs based on data up to time k. H_p denotes the number of steps in the output prediction horizon. These predictions of the outputs are functions of future control increments $\Delta r\left(k+i \mid k\right)$ for $i = 0, \delta H_c, 2\delta H_c, \ldots, H_c - 1$. The integer number of samples H_c is called the control horizon, the control signal is allowed to change only at integer multiples of δH_c samples and is set to be constant for all $i \geq H_c$. This means that the future control signal has the form of a stairstep function with steps occuring at δH_c intervals. The reference signal y_{ref} represents the desired outputs, Q and R are suitably chosen weighting matrices. For this specific application, the optimization problem was specified using the following parameters

$$H_c = 1, \quad \delta H_c = 1, \quad H_p = 40,$$
$$Q = \text{diag}(0.01, 0.01, 1), \quad R = \text{diag}(10, 5 \cdot 10^6, 1).$$

The slack variable ε and its weight ρ are used for softening constraints. The exact purpose of the slack variable and weight in the problem formulation will be clarified shortly.

In order to obtain the predictions for the signals of interest, a model of the process is needed. By using a linear model, the resulting optimization problem of minimizing $J(k)$ will be a quadratic programming (QP) problem, for which fast and numerically reliable algorithms are available. The linearized prediction model, introduced in Section II-B, is augmented with extra states as follows

$$\overbrace{\begin{bmatrix} \hat{x}(k+1) \\ \hat{d}(k+1) \\ r(k) \end{bmatrix}}^{\hat{\xi}(k+1)} = \overbrace{\begin{bmatrix} A_k & 0 & B_k \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}}^{\mathcal{A}_k} \overbrace{\begin{bmatrix} \hat{x}(k) \\ \hat{d}(k) \\ r(k-1) \end{bmatrix}}^{\hat{\xi}(k)} + \overbrace{\begin{bmatrix} B_k \\ 0 \\ I \end{bmatrix}}^{\mathcal{B}_k} \Delta r(k)$$
$$\underbrace{\begin{bmatrix} \hat{y}(k) \\ \hat{z}(k) \\ \hat{u}(k) \end{bmatrix}}_{\hat{w}(k)} = \underbrace{\left[\begin{array}{c|c|c} C_k & I & D_k \\ & 0 & \end{array}\right]}_{\mathcal{C}_k} \underbrace{\begin{bmatrix} \hat{x}(k) \\ \hat{d}(k) \\ r(k-1) \end{bmatrix}}_{\hat{\xi}(k)} + \underbrace{D_k}_{\mathcal{D}_k} \Delta r(k) \tag{14}$$

Three integrators are added to convert the control changes Δr into actual control commands r, each one associated with the command inputs of velocity, turn rate and altitude rate. A simple disturbance model is incorporated to the state space description of the prediction model in equation (14), which assumes constant disturbances are acting on outputs. The constant disturbance estimates are obtained by filtering the difference between measured and predicted outputs, according to equation (12).

As in most applications, there are level and rate limits on control inputs. These are enforced as hard constraints

$$\underline{u} \leq \hat{u}\left(k+1 \mid k\right), \ldots, \hat{u}\left(k+H_p \mid k\right) \leq \overline{u} \tag{15}$$
$$\underline{\Delta r} \leq \Delta r\left(k\right), \Delta r\left(k+\delta H_c\right), \ldots, \Delta r\left(k+H_c\right) \leq \overline{\Delta r} \tag{16}$$

since the RHC algorithm has almost direct control over them (the optimization variables are the changes in control inputs). Hence there is no modeling uncertainty associated with this aspect of the prediction model. Another type of constraint is also considered in this specific application example represented by certain maneuvering limits on the aircraft. The controller has to be versatile enough to handle these limits that might be system-state dependent or change according to different stages of a mission. The most important maneuvering constraints of the T-33 testbed were characterized based on the DemoSim open vehicle executable model. These constraints arose mainly from vertical acceleration and bank angle limits. Although the relationship between the control inputs and these variables is nonlinear, it could be approximated reasonably well with linear expressions. This allowed their incorporation into the RHC problem formulation as output constraints. It is vital that these limits are treated as soft constraints, since disturbances and model mismatch can easily lead to infeasibility problems if hard constraints are put on these type of output signals. Details of the soft output constraint formulation can be found in [5] and are omitted in this paper due to space limitations. Note also that the actual flight code was flown without the inclusion of these type of constraints.

The numerical values of the u output signal limits (representing hard constraints) were specified as

$$\underline{u} = \begin{bmatrix} -100 \text{ ft/s} \\ -0.035 \text{ rad/s} \\ -1000 \text{ ft/s} \end{bmatrix}, \quad \overline{u} = \begin{bmatrix} 100 \text{ ft/s} \\ 0.035 \text{ rad/s} \\ 1000 \text{ ft/s} \end{bmatrix}. \tag{17}$$

Hard constraints were put on the optimization variables as well, specifically

$$\underline{\Delta r} = \begin{bmatrix} -4 \text{ ft/s}^2 \\ -0.01 \text{ rad/s}^2 \\ -1000 \text{ ft/s}^2 \end{bmatrix}, \quad \overline{\Delta r} = \begin{bmatrix} 4 \text{ ft/s}^2 \\ 0.01 \text{ rad/s}^2 \\ 1000 \text{ ft/s}^2 \end{bmatrix}. \tag{18}$$

Constraint softening for the outputs z is accomplished by introducing an additional slack variable ($\varepsilon \geq 0$) that allows some level of constraint violation if no feasible solution exists

$$\underline{z} - \varepsilon \leq \hat{z}\left(k+1 \mid k\right), \ldots, \hat{z}\left(k+H_p \mid k\right) \leq \overline{z} + \varepsilon \tag{19}$$

It is beneficial to use an ∞-norm (maximum violation) penalty on constraint violations (as shown in (13) and (19)),

because it gives an "exact penalty" method if the weight ρ is large enough. This means that constraint violations will not occur unless no feasible solution exists to the original "hard" problem. If a feasible solution exists, the same solution will be obtained as with the "hard" formulation.

B. Implementation within the OCP

The RHC guidance algorithm was implemented using an application programming interface developed specifically for receding horizon control algorithms (RHC API) in a real-time environment using the OCP. The RHC API provides an interface to an online mathematical program solver by formulating a generic optimization problem for receding horizon applications. Implementation of the control algorithms is performed using three separate threads. One of these is used to enable anytime scheduling of the online optimization solver while satisfying real-time requirements with the other two hard real-time tasks. The reader is referred to [5] for further details.

After creating the prediction model and formulating the RHC problem, the optimization problem was translated to the formulation used by the RHC API. The optimization yielded $\Delta\tilde{V}_{cmd}, \Delta\tilde{\dot{\chi}}_{cmd}, \Delta\tilde{\dot{h}}_{cmd}$ values, which were integrated to get $\tilde{V}_{cmd}, \tilde{\dot{\chi}}_{cmd}, \tilde{\dot{h}}_{cmd}$. The obtained $\tilde{\dot{h}}_{cmd}$ value was further integrated to get $\tilde{h}_{cmd}$. Trim values were added to arrive at $V_{cmd}, \dot{\chi}_{cmd}, h_{cmd}$ values, which could be directly implemented on the T-33 autopilot.

C. Remarks

The problem formulation used for control design is a natural extension of a fixed LTI model based RHC. The prediction at a certain time step is based on a linear model that best describes the plant at the actual flight condition, assuming that flight condition dependent linear models are available for prediction. A fixed LTI model is used over the entire prediction horizon but it is updated according to the values of the scheduling parameters $\varrho(k)$ every time the horizon is propagated and the optimization is resolved based on new measurement data. This approach leads to a QP problem where the state matrices $\mathcal{A}_k, \mathcal{B}_k, \mathcal{C}_k, \mathcal{D}_k$ describing the internal model change in each implementation cycle. A flight condition dependent description of the plant can be obtained either by freezing the scheduling parameters of a quasi-LPV model [7], or interpolating over a database of linearized models. In other cases the nonlinear prediction model is simple enough to lend itself to "online" linearization, while still retaining a reasonable prediction accuracy. This latter approach was followed in our control solution.

We note if an accurate prediction of the parameters that the linear models depend on is available, this would allow for the prediction model to vary over the prediction horizon. The optimization problem could still be formulated as a quadratic program using different state matrices of the internal model at each time step. Obtaining a reasonable prediction of the scheduling parameters is not always easy, one could experiment with solving the problem first with the fixed LTI model based RHC method and use the solution as the prediction for the scheduling parameters. Our investigations indicate, that this extra effort doesn't lead to significant improvement for the specific application example and horizon lengths considered. Moreover, even though the optimization problem complexity is retained, the additional computational overhead could undermine real-time implementation of these ideas if the parameter-dependent models are calculated using interpolation over a collection of linear systems.

IV. Demonstration scenarios

The flight tests took place at Edwards Air Force Base in the Mojave desert in June 2004. A description of the flight demonstration experiment scenario is provided next. The timeline of events starts with engaging the RHC controller in a pre-specified area near the ingress point once the starting conditions are met. The controller tracks a time-stamped position reference trajectory while respecting constraints on the vehicle dynamics. At a certain point along the reference trajectory, a pop-up threat can be invoked by a ground operator, which results in a switch to an alternative reference trajectory that avoids the threat. After the target is reached with a specified heading, a simulated fault was inserted into the system, which was detected shortly thereafter at a trajectory segment designed specifically for this purpose. After detection, the fault was removed and the aircraft returned to the egress point. A schematic figure of the experiment plan can be found in [5] and is omitted here for brevity. Figure 4(a) shows the ground track of the reference trajectory that had to be tracked eventually.

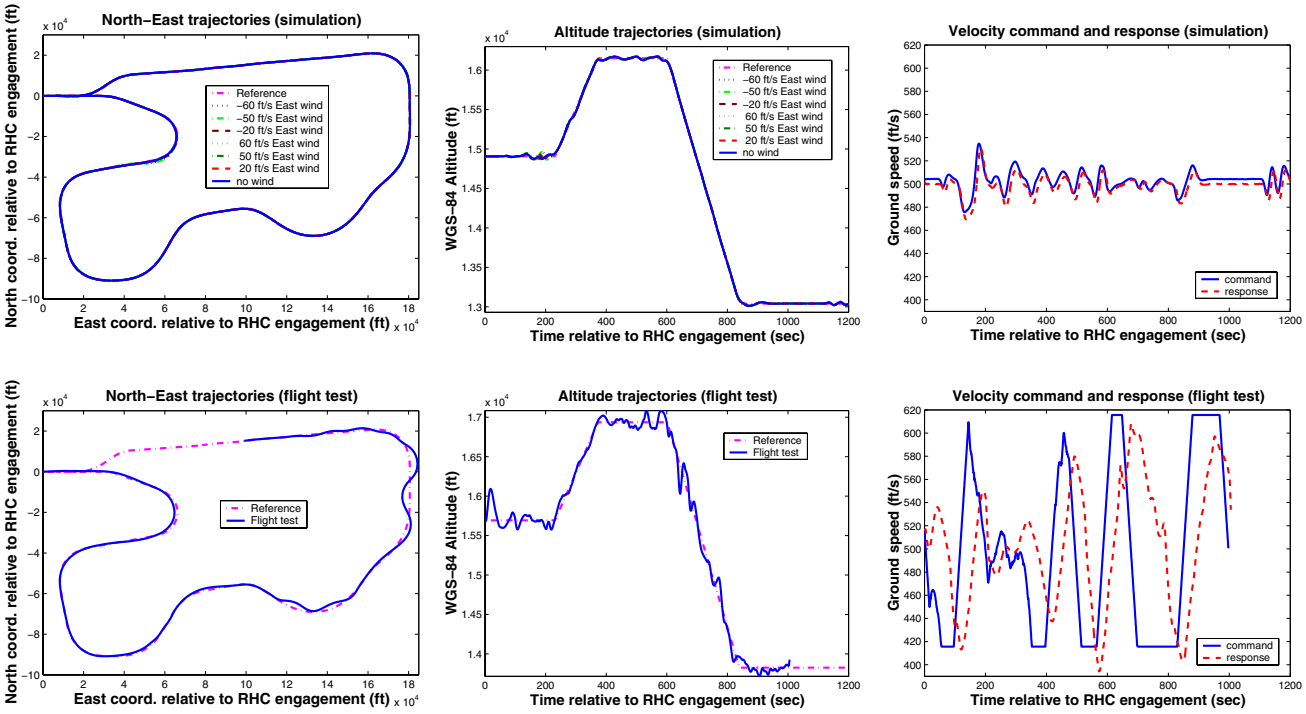
In a second, more ambitious scenario, the fault is not removed from the system after detection, and the RHC controller is reconfigured to adapt to the faulty vehicle dynamics. Constraints are also adjusted to restrict the aircraft's maneuvers. Description of the fault detection filter design and test results are omitted in this paper and can be found in [5], along with simulations that show successful reconfiguration of the RHC controller after a fault.

V. Simulation and experimental results

Several universities and aerospace companies were involved in the two-week long flight demonstrations of the DARPA SEC project, each having their own experimental flight scenario to be tested. Due to difficulties with asset scheduling at the base, our team eventually had only two flight tests that could be evaluated.

The receding horizon guidance controller was tested in simulation with different wind conditions (up to 60 ft/s) and showed excellent robustness. Constraint enforcement was demonstrated by saturating turn rate command, whereas true airspeed was adjusted according to the various wind conditions, to achieve the necessary ground speed required for accurate tracking of the position reference. The main limitations of good performance at higher wind velocities were posed by flight envelope constraints.

The top plots in Figures 4(a)-4(b) show simulated tracking performance in the north-east coordinate frame and in terms of altitude under different wind conditions. The bottom plots in Figures 4(a)-4(b) illustrate the flight test results. The main reason for deviations from the reference



(a) North, east tracking performance. (b) Altitude tracking performance. (c) Velocity command and response.

Fig. 4. Comparison of simulation and flight test results. The top plots show simulation results under different wind conditions, the bottom plots represent flight test data.

trajectory and degraded tracking performance during flight test was that automatic speed control was not available on the test platform. Airspeed was controlled using manual adjustments to the throttle by the pilot, who was cued by a three-state LED indicator whether to increase, decrease or maintain the velocity of the aircraft based on commanded and actual speed measurements. The dead-zone of the "maintain speed" status indicator was approximately 30 ft/s. Another factor influencing the outcome of experiments was that flight tests were conducted in the presence of strong winds (25-30 knots).

Figure 4(c) suggests an average delay of 50-100 seconds in the velocity command channel, which was not modeled in the RHC controller. The controller was tested only up to 10-20 samples (5-10 seconds) of unmodeled additional delay compared to the prediction model and showed acceptable degradation of performance.

VI. Conclusions

The RHC-based guidance system showed amenable robustness properties inspite of the dramatic difference between the velocity tracking behavior of the prediction model and the real system. Performance analysis of flight test data and simulation results with varying wind conditions suggest that the RHC guidance law would have excellent tracking performance using an autonomous speed control system. Further high-fidelity simulations showed that output constraints could be also accommodated using soft constraint formulation. This, along with the observed successful reconfiguration experiments, suggests that the proposed approach provides a high-performance, versatile guidance technology for future unmanned aircraft systems.

Acknowledgement

The authors gratefully acknowledge discussions with George Papageorgiou at Honeywell Labs on the particular RHC formulation to be used in the final flight test.

References

[1] Office of the Secretary of Defense, "Unmanned aerial vehicles roadmap 2002-2027," Department of Defense, Tech. Rep., 2002.

[2] J. S. Bay, B. S. Heck, *et al.*, "Special section: Software-enabled control," *IEEE Control Systems Magazine*, vol. 23, no. 1, Feb. 2003.

[3] J. Paunicka, B. Mendel, and D. Corman, "The OCP – An open middleware solution for embedded systems," in *Proc. American Contr. Conf.*, Arlington, VA, June 2001, pp. 3345–3350.

[4] T. Keviczky and G. J. Balas, "Receding horizon control of an F-16 aircraft: a comparative study," in *Proc. European Control Conf.*, Cambridge, UK, 2003.

[5] T. Keviczky, R. Ingvalson, H. Rotstein, A. Packard, O. R. Natale, and G. J. Balas, "An integrated multi-layer approach to software enabled control: Mission planning to vehicle control," Dept. of Aerospace Eng. and Mechanics. University of Minnesota, Minneapolis. Dept. of Mechanical Eng. University of California, Berkeley, Tech. Rep., Nov. 2004. [Online]. Available: http:// www.aem.umn.edu / people / faculty / balas / darpa_sec / SEC.Publications.html

[6] J. M. Maciejowski, *Predictive Control with Constraints*. Prentice Hall, 2002.

[7] M. Huzmezan and J. M. Maciejowski, "Reconfiguration and scheduling in flight using quasi-LPV high-fidelity models and MBPC control," in *Proc. American Contr. Conf.*, 1998.