AIAA Guidance, Navigation, and Control Conference
08 - 11 August 2011, Portland, Oregon

AIAA 2011-6310

Nonlinear Model Predictive Controller for Navigation, Guidance and Control of a Fixed-Wing UAV

Gonzalo A. Garcia[1] and Shahriar Keshmiri[2]
The University of Kansas, Lawrence, KS, 66045

A Nonlinear Model Predictive Controller (NMPC) is designed to perform guidance, navigation, and control of a fixed-wing UAV in waypoint trajectory tracking. The inherent nonlinear nature of the aircraft together with the limited desired operation ranges of both controls and states variables make the use of an on-line control law computation more appropriate. The MPC theory allows an optimal on-line solution, when enough onboard computational power is available. Using nonlinear guidance laws for both lateral and longitudinal planes, extended Kalman filter estimations if required, and a complete nonlinear model of the UAV, the NMPC generates on-line optimal or suboptimal control signals to keep the UAV in trajectory following.

Nomenclature

UAV	=	unmanned aerial vehicle
(N)MPC	=	(nonlinear) model predictive control
DOF	=	degree of freedom
MIMO	=	multi input, multi output
SQP	=	sequential quadratic programming
ΔT	=	time interval
λ	=	Lagrange multipliers

I. Introduction

CHANGES in the dynamic condition due to the inherent nonlinear nature of the UAV, together with the nonlinearity in the aerodynamics forces and moments and the nonlinear nature on natural disturbances, can be addressed by an on-line scheduling version of pre-computed linear controllers (see for example Ref. 1 for robust designs). These pre-computed control laws lack the ability to adjust to unforeseen events, like unexpected changes in UAV dynamics due to some failure. Adaptive control assisted by system identification[2], could overcome unanticipated alterations in either actuators, sensors or the UAV itself, by redesigning an on-line new controller or set of controllers that better reflect the new dynamics.

As virtually any controlled system, control and state variables are nominally desired to be restricted to bound ranges to vary, including their rates of change. These restrictions worsened by unexpected failures impose severe constraints to fulfill by the controller, augmenting the nonlinearity of the system, and reducing the stability reserve. These possibly time varying constraints impose an extra difficulty for controller's scheduling strategy and even for on-line adaptive control.

The MPC approach, as a numerical on-line controller computation, is able to tackle the previous drawbacks in an integrated way, while searching for an optimal control solution (or suboptimal if not enough available computational power) under the new dynamics and conditions[3]. Giving the required computational resources that must be available for on-line search, MPC is able to include all restrictions without coarse approximations or simplifications. It is able to use the full nonlinear description of the plant, either static or time-varying and also is able to incorporate the possibly time-varying set of constraints during in each numeric iteration. This ability is due the inherent nature of its design concept, since it updates every predefined time interval the numerical search parameters of the control solution. By definition the MPC is a nonlinear adaptive controller that is able, given the computational power, to calculate as close as possible an optimal control law.

[1] Graduate Research Assistant, Department of Aerospace Engineering, gagarcia@ku.edu, AIAA Student Member.
[2] Assistant Professor, Department of Aerospace Engineering, keshmiri@ku.edu, AIAA Member.

The MPC and NMPC have been applied successfully in several areas, where the Chemical Industry has being the first. Their applications to unmanned vehicles are more recent and there are still promissory advances to make. In Ref. 4, NMPC was applied to control a submarine in the longitudinal plane, where the task was to follow the sea bed at a constant distance with no prior knowledge of it. Following this approach Ref. 5 formulated a NMPC algorithm for a rotorcraft-based UAV, showing superior tracking performance over conventional multi-loop proportional-derivative controllers. Ref. 6 applied linear MPC to a small helicopter easily incorporating control and state constraints reducing the computational burden typically of a non-convex nonlinear programming. Another NMPC application is given in Ref. 7 where a receding horizon control algorithm is used for lateral road following using passive sensors. NMPC have also been applied to compute evasive maneuvers in pursuit/evasion games[8] using a gradient-descent method.

Most of the past developments utilize the NMPC formulation in the outer loop, for navigation and guidance purposes, leaving the inner loop control task to some commercial autopilot. Although this approach reduces the computational power requirement by reducing the size of the nonlinear model to be employed by the NMPC, it generates a potential undesired coupling between both loops affecting the stability. Ref. 9, that uses the same configuration utilizing NMPC as a high-level tracking controller (outer loop), proved that the complete closed-loop system remains stable if a given initial condition can be driven into a predefined set of final states, by the solution of a finite-horizon control problem. In their work, the inner loop is handled by a commercial low-level controller, and the NMPC minimizes the lateral cross track error to straight waypoint segments, converting the control law into a regulating problem.

Another interesting on-line application is found in Ref. 10. The NMPC was used to control an inverted pendulum. The control law is able to work in the entire state space including both inputs and state constraints. This control problem proves to be interesting for the KU Aerospace Engineering Flight System team, because it shares salient features with UAVs in general. It displays fast dynamics with highly nonlinear behavior and it is unstable about the operating point showing a non-minimum phase nature. The on-line search uses a Sequential Quadratic Programming (SQP). This method solves a series of sub-problems intended to minimize a quadratic function of the objective, subject to linearization of the constraints. This application included one control input and four states, two of them directly measured and two estimated.

The work of the present paper designs a NMPC for integrated navigation, guidance and control on a fixed-wing UAV. The key idea is to augment the original nonlinear description of the aircraft including as new states the guidance law output variables developed in Ref. 11. The guidance law exposed in Ref. 11 converts the inertial position and velocity of the UAV, into determinate lateral and longitudinal angular errors, with respect to a moving point within the current segment of a trajectory. These angular errors after appropriate transformation are converted into attitude commands and are feed into the inner loop controller (and H-infinity controller) for their further diminishing. This correction forces the UAV to the current segment. This configuration implies two separate loops, designed separately, that could eventually produce cross coupling in some circumstances threatening the stability.

The proposed NMPC design optimally tackles this weakness. It makes the control problem an integrated solution merging consequently both outer and inner loops as one absolute loop, performing guidance, navigation and control at the same time, with the help of an extended Kalman filter[12] to estimate unobserved signals or to enhance poorly measured states, if needed. Four inputs and four outputs are included into a full MIMO NMPC system.

The present work incorporates this guidance law as part of the nonlinear model, adding new variables to the system representation. Now the system to be controlled is not just the UAV itself following an external reference calculated from the trajectory and its navigational status via an outer loop feedback, but it is an augmented nonlinear model which includes the information of the trajectory creating new variables, in this specific case, angular error states.

Now the control task switches from a reference tracking problem to a regulation problem, where the angular errors need to be driven to zero in order to perform the trajectory following. This is a very important simplification in terms of MPC. The regular application of this technique, which carries out a prediction ahead in time of the behavior of the system, requires knowing in advance the future reference values, which is not always possible.

The paper includes in Section II a description of the UAV and its augmentation with the guidance laws, Section III exposes the NMPC formulation for our current problem including an explanation of the numerical search method SQP, and Section IV with simulations results.

II. UAV Description for Guidance, Navigation and Control

A. Rigid Body 6DOF UAV Description

A 33% scale Yak-54 aircraft is modeled as a 6-DOF rigid body, following the approach in Ref. 13. It considers both rotational and translational motion centered about the center of gravity. The modeling of the UAV is done using body and inertial coordinate system, and the model as initially twelve states. Their dynamic is described with the following set of nonlinear differential equation set where all variables are total variables

$$
\begin{aligned}
\dot{U} &= RV - QW - g\sin\theta + \left(X_T + X_A\right)/m \\
\dot{V} &= -RU + PW + g\sin\phi\cos\theta + Y_A/m \\
\dot{W} &= QU - PV + g\cos\phi\cos\theta + Z_A/m \\
\dot{\phi} &= P + \tan\theta\left(Q\sin\phi + R\cos\phi\right) \\
\dot{\theta} &= Q\cos\phi - R\sin\phi \\
\dot{\psi} &= \left(Q\sin\phi + R\cos\phi\right)/\cos\theta \\
\dot{P} &= \left(J_{xz}\left[J_x - J_y + J_z\right]PQ - \left[J_z\left(J_z - J_y\right) + J_{xz}^2\right]QR + J_z L_A + J_{xz}N_A\right)/\left(J_x J_z - J_{xz}^2\right) \\
\dot{Q} &= \left(\left[J_z - J_x\right]PR - J_{xz}\left[P^2 - R^2\right] + M_A\right)/\left(J_x J_z - J_{xz}^2\right) \\
\dot{R} &= \left(\left[\left(J_x - J_y\right)J_x + J_{xz}^2\right]PQ - J_{xz}\left[J_x - J_y + J_z\right]QR + J_{xz}L_A + J_x N_A\right)/\left(J_x J_z - J_{xz}^2\right) \\
\dot{p}_N &= U\cos\theta\cos\psi + V\left(-\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi\right) + W\left(\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi\right) \\
\dot{p}_E &= U\cos\theta\sin\psi + V\left(\cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi\right) + W\left(-\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi\right) \\
\dot{p}_H &= U\sin\theta - V\sin\phi\cos\theta - W\cos\phi\cos\theta
\end{aligned}
\tag{1}
$$

where U,V,W are the body velocity components, ϕ,θ,ψ the inertial attitude components, P,Q,R the body angular rate components, p_N, p_E, p_H the inertial position components, L_A, M_A, N_A the body aerodynamic angular moment components (no thrust moments assumed), X_A, Y_A, Z_A the aerodynamic body force components, X_T the thrust body force (assumed in the x axis only), J_x, J_y, J_z the moments of inertia, J_{xz} the product of inertia and m the aircraft mass.

The aerodynamic forces and moments are not available for direct measurement, so they are calculated using the component build-up method mentioned in Ref. 14. In the build-up method, a trim point is selected to calculate the stability and control derivatives. In principle this implies a departure from the pure nonlinear description of the aircraft, but it turns to be unavoidable given the lack of the exact values when the dynamic condition moves away from the chosen operating point. Beside the trim point these aerodynamic forces X_A, Y_A, Z_A and aerodynamic moments L_A, M_A, N_A are dependent on a determined subset of perturbed states and perturbed control surface deflections[14]. The control surfaces are defined as $\delta_A, \delta_R, \delta_E$ aileron, rudder and elevator, respectively. The thrust force X_T is also modeled as a linear function, in this case depending on the perturbation about the trim point of the throttle deflection δ_T. The proportionality constant is extracted from flight test data[14, 15].

Given the suite of sensors installed onboard the UAV, the airflow angles α, β and the airspeed V_T are measured instead of the body velocity U,V,W. So after measurement is done the following transformation is utilized[13]

$$
\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} V_T\cos\alpha\cos\beta \\ V_T\sin\beta \\ V_T\sin\alpha\cos\beta \end{bmatrix} \tag{2}
$$

In terms of control action $\boldsymbol{u}^T = \{\delta_T, \delta_E, \delta_A, \delta_R\}$ and states $\boldsymbol{x}^T = \{U,V,W,\phi,\theta,\psi,P,Q,R,p_N,p_E,p_H\}$, the following relation represents the complete dynamics from inputs to states defined in Eqs. (1) and (2)

$$
\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) \tag{3}
$$

The output equation, relating the states to the available measurements, is defined as

$$\boldsymbol{y}(t) = \boldsymbol{h}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) \tag{4}$$

where the function h indicates how the measurement are conformed. These outputs could include measurement of any state of the system, or a combination among them. A specific output equation could be used for determinate purposes like an extended Kalman filter formulation, giving the availability of the appropiate sensors, and a specific output relation could be used for a NMPC is needed.

Note that the states U,V,W and V_T,α,β are interchangeable. Also the dynamics of the actuators can be included in the set of equations (not shown here). An extended Kalman filter[12] is used in order to estimate unmeasured state variables and to filter up to some degree the additive noise of those measured.

B. Model Augmentation with Guidance Laws

The main variables to be used in the NMPC formulation are two angular errors to be defined in the sequel. So the task of the NMPC controller will be to attempt to diminish these errors within the prediction horizon, generating a feasible control sequence, fulfilling the constraints. This decreasing will assure a closer trajectory following, as will be explained later. But these angular errors are not part of the nonlinear model in Eq. (1), so they need to be formulated and incorporated as new variables.

The trajectory is composed of straight segments delimited by a list of an arbitrary amount of waypoints. The philosophy of the chosen trajectory following is to keep the UAV as close as possible to the "current" segment indirectly in terms of lateral and longitudinal cross track errors and to be able to "switch" to the incoming segment in advance to avoid excessive overshoots. The logic to determine the current segment is simple. It assumes that the UAV will start from the first segment (subtended between the first two waypoints). Once the UAV is below a predefined distance to the end of the segment, it will change to the following segment. It is interesting to note that this logic uses only two consecutive waypoints, leaving the possibility of modifying on-line the future waypoints as needed.

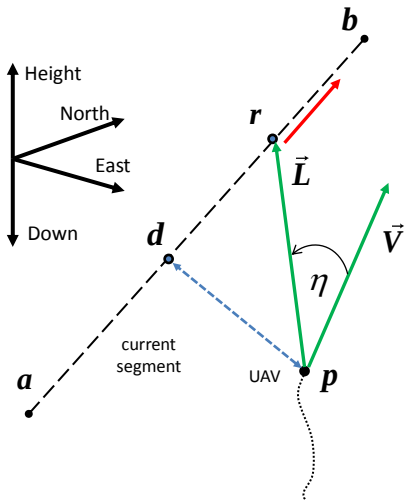

Figure 1. Geometry of the Guidance Law.

The guidance logic to be used is the one illustrated in Ref. 11 which was derived from a lateral nonlinear guidance defined in Ref. 16. This was extended to the longitudinal plane and modified with the addition of new features.

The guidance module calculates the angle η between the inertial velocity vector $\vec{\boldsymbol{V}}$ and the vector $\vec{\boldsymbol{L}}$ subtended from the current position of the UAV $\boldsymbol{p}$ and an imaginary point $\boldsymbol{r}$, running through the segment between $\boldsymbol{a}$ to $\boldsymbol{b}$, see Fig. 1. This point $\boldsymbol{r}$ is located at fixed distance ahead from the perpendicular point $\boldsymbol{d}$ with respect to the UAV, and as long as $\boldsymbol{d}$ moves, $\boldsymbol{r}$ moves in the same direction (see red arrow) too. Analyzing the geometry in Fig, 1, it is evident that a small or zero angle η is equivalent to a small or zero cross track error (distance between points $\boldsymbol{p}$ and $\boldsymbol{d}$), and the inverse is also true. In this application, this error angle η is decomposed into two angles[12], lateral η_{LAT} and longitudinal η_{LON} angles referred to the current segment inertial attitude. This decomposition is not mandatory and the NMPC minimization could be done with the 3D angle η as well (not done in this work).

In order to reformulate both angles referred to the current segment $\boldsymbol{ab}$, both vectors $\vec{\boldsymbol{V}}$ (to be differentiate from the body velocity component V) and $\vec{\boldsymbol{L}}$ need to be rotated from the inertial frame $\boldsymbol{I}$ to a segment frame called $\boldsymbol{S}$, see Fig. 2. The inertial speed $\vec{\boldsymbol{V}}$ is defined as

$$\vec{\boldsymbol{V}}(t) = \begin{bmatrix} \dot{p}_N(t) \\ \dot{p}_E(t) \\ \dot{p}_H(t) \end{bmatrix} \tag{5}$$

In order to define $\vec{\boldsymbol{L}}$ it is necessary to define the points $\boldsymbol{d}$ and $\boldsymbol{r}$. Define $\boldsymbol{d}$ parametrically as

$$\boldsymbol{d}(t)=\begin{bmatrix} d_N(t) \\ d_E(t) \\ d_H(t) \end{bmatrix}=\boldsymbol{a}(t)+q(t)\left[\boldsymbol{b}(t)-\boldsymbol{a}(t)\right] \tag{6}$$

$$q(t)=\frac{\left[\boldsymbol{p}(t)-\boldsymbol{a}(t)\right]\cdot\left[\boldsymbol{b}(t)-\boldsymbol{a}(t)\right]}{\left|\boldsymbol{b}(t)-\boldsymbol{a}(t)\right|^2} \tag{7}$$

where $|\cdot|$ indicates the distance between two points. Points $\boldsymbol{a}$ and $\boldsymbol{b}$ are time dependent as they change when the segment is changed, but they are constant otherwise. The parameter $0<q(t)<1$ determines the point $\boldsymbol{d}$ between $\boldsymbol{a}$ and $\boldsymbol{b}$ perpendicular to $\boldsymbol{p}$. The point $\boldsymbol{r}$ is located at a fixed distance toward the end of the segment[11], so it can be defined parametrically as well as

$$\boldsymbol{r}(t)=\begin{bmatrix} r_N(t) \\ r_E(t) \\ r_H(t) \end{bmatrix}=\boldsymbol{d}(t)+m(t)\left[\boldsymbol{b}(t)-\boldsymbol{d}(t)\right] \tag{8}$$

$$m(t)=\frac{\left|\boldsymbol{r}(t)-\boldsymbol{d}(t)\right|}{\left|\boldsymbol{b}(t)-\boldsymbol{d}(t)\right|} \tag{9}$$

where, even though $\boldsymbol{r}$ and $\boldsymbol{d}$ are time dependent, their distance $|\boldsymbol{r}\text{-}\boldsymbol{d}|$ is by construction a given fix quantity. The parameter $m(t)$ sets the point $\boldsymbol{r}$ at this fixed distance of $|\boldsymbol{r}\text{-}\boldsymbol{d}|$ toward the end of the segment $\boldsymbol{b}$.

Given the characterization of the guidance laws, the point $\boldsymbol{r}$ is acting as a pseudo-target located in the trajectory, always moving to the right (moving away of) direction ahead of the UAV.

Ref. 11 transforms each angle error η_{LON} and η_{LAT} into attitude commands θ_{CMD} and ϕ_{CMD} respectively, sending them to the inner loop. In the present work these angles go directly into the NMPC cost function without any further conversion.

Based on their geometric definition, they should always vary within $-\pi<\eta_{LON,LAT}<\pi$. A simple logic unwraps the lateral angle, so if it surpasses one of the limits, it is shifted pertinently in order to stay within the range. On the other hand the longitudinal angle should not overcome any of the limits to avoid the UAV to start flying upside down. This restriction needs to be included in the numerical search as a state constraint to assure its accomplishment.

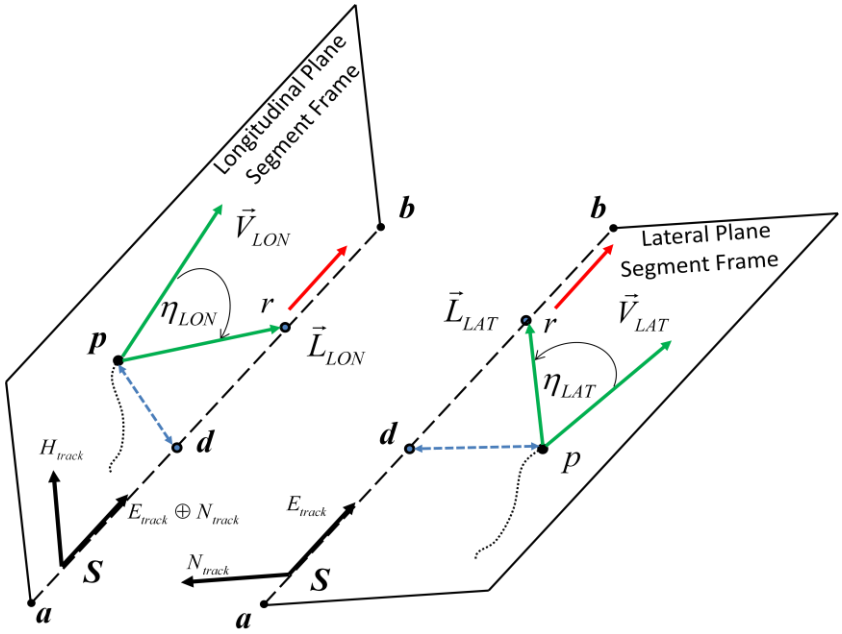

Figure 2. Geometry of Longitudinal and Lateral Guidance Laws.

From Fig. 1 $\vec{\boldsymbol{L}}$ is defined as

$$\vec{\boldsymbol{L}}(t) = \boldsymbol{r}(t) - \boldsymbol{p}(t) = \begin{bmatrix} r_N(t) - p_N(t) \\ r_E(t) - p_E(t) \\ r_H(t) - p_H(t) \end{bmatrix} \tag{10}$$

Each segment has two rotations giving the particular location of its waypoints $\boldsymbol{a}$ and $\boldsymbol{b}$, with respect to the inertial frame $\boldsymbol{I}$. These rotations angles α_{track} and β_{track} are show in Fig. 3. The segment frame $\boldsymbol{S}$ follows these rotations as shown in Fig. 3 and its axial location is irrelevant, only its rotations.

In order to get the error angles η_{LON} and η_{LAT} both vectors $\vec{\boldsymbol{V}}$ and $\vec{\boldsymbol{L}}$ are converted from the inertial frame $\boldsymbol{I}$ to the segment frame $\boldsymbol{S}$ by a standard matrix operation. The following operation $\boldsymbol{C}_I^S$ is applied to both vectors $\vec{\boldsymbol{V}}$ and $\vec{\boldsymbol{L}}$

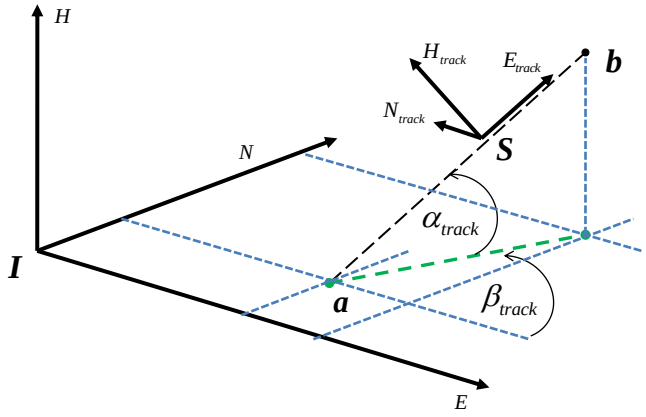

Figure 3. Segment Rotations.

$$\boldsymbol{C}_I^S = \begin{bmatrix} \cos\alpha_{track}\cos\beta_{track} & \sin\beta_{track} & \sin\alpha_{track}\cos\beta_{track} \\ -\cos\alpha_{track}\sin\beta_{track} & c\cos\beta_{track} & -\sin\alpha_{track}\sin\beta_{track} \\ -\sin\alpha_{track} & 0 & \cos\alpha_{track} \end{bmatrix} \tag{11}$$

Using the previous transformation $\boldsymbol{C}_I^S$ of Eq. (11), the inertial vectors $\vec{\boldsymbol{V}}$ and $\vec{\boldsymbol{L}}$ defined in Eqs. (5) and (10), respectively, referred to the segment frame $\boldsymbol{S}$ become

$$\vec{\boldsymbol{V}}_{track}(t) = \begin{bmatrix} \dot{p}_{N\,track}(t) \\ \dot{p}_{E\,track}(t) \\ \dot{p}_{H\,track}(t) \end{bmatrix} \tag{12}$$

$$\vec{\boldsymbol{L}}_{track}(t) = \begin{bmatrix} \left[r_N(t) - p_N(t)\right]_{track} \\ \left[r_E(t) - p_E(t)\right]_{track} \\ \left[r_H(t) - p_H(t)\right]_{track} \end{bmatrix} \tag{13}$$

For the lateral plane of the segment frame $\boldsymbol{S}$, η_{LAT} is defined as (see Fig. 4a)

$$\eta_{LAT}(t) = \arctan\left(\frac{\left[r_N(t) - p_N(t)\right]_{track}}{\left[r_E(t) - p_E(t)\right]_{track}}\right) - \arctan\left(\frac{\dot{p}_{N\,track}}{\dot{p}_{E\,track}}\right) \tag{14}$$

And for the longitudinal plane, η_{LON} is defined as (see Fig. 4b)

$$\eta_{LON}(t) = \arctan\left(\frac{\left[r_H(t) - p_H(t)\right]_{track}}{\left[r_E(t) - p_E(t)\right]_{track} \oplus \left[r_N(t) - p_N(t)\right]_{track}}\right) - \arctan\left(\frac{\dot{p}_{H\,track}}{\dot{p}_{E\,track} \oplus \dot{p}_{N\,track}}\right) \tag{15}$$

where the operator $\oplus$, used to reduce the length of the expressions, is defined as

$$x \oplus y = \sqrt{x^2 + y^2} \tag{16}$$

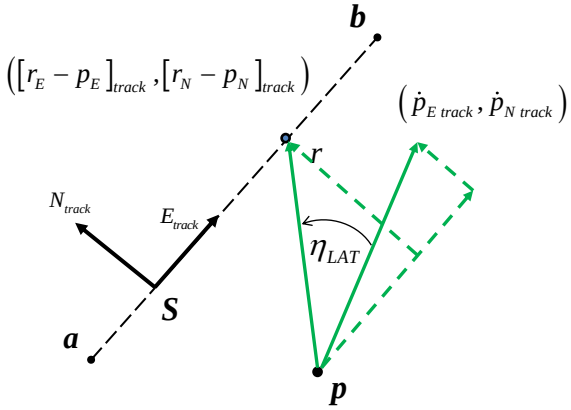

Figure 4a. Lateral Angle Calculation.

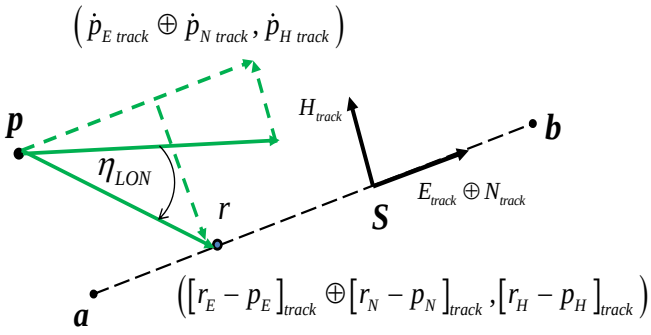

Figure 4b. Longitudinal Angle Calculation.

The system model is augmented by including in the state vector $\boldsymbol{x}$ the quantities η_{LAT} and η_{LON} as new states. The state vector is redefined as a vector of fourteen elements

$$\boldsymbol{x}^T = \{U, V, W, \phi, \theta, \psi, P, Q, R, p_N, p_E, p_H, \eta_{LAT}, \eta_{LON}\} \tag{17}$$

C. UAV Model Discretization

Given the inherent discrete nature of the numerical search of MPC, the description of the UAV needs to be discretized, in particular the time derivatives. The choice of an appropriate discretization method is relevant. Having no unlimited sampling rate availability, bounded by the computational power capability of managing larger amounts of data and its numerical calculations, the discretization scheme should be robust enough to keep numerical stability in the face of relatively small sampling intervals. In this research, with the premise to prove the concept, the Euler dicretization method was chosen. The system equation Eq. (3), now augmented with the new states, is expressed as

$$\boldsymbol{x}(t+h) \approx \boldsymbol{x}(t) + \Delta T \cdot \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{18}$$

where ΔT is the sampling interval. The choice of this parameter together with others like horizon length and precision of the numerical convergence, have a direct impact on the NMPC performance, in terms of accuracy and in terms of work load and time consumption. For the NMPC formulation, Eq. (18) is expressed generically as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta T \cdot \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{f}_d(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{19}$$

where k is the time index of the discretized variables.

It is interesting to note that the NMPC is able to work with nonlinear time-varying systems as well. The only requirement is to have an updated accurate expression of the system $\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)$ and $\boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)$ as it changes with time, what could be provided by some system identification means. This capacity converts the on-line controller into a nonlinear adaptive controller too. The function $\boldsymbol{f}$ and consequently $\boldsymbol{f}_d$ is assumed time invariant during the numerical minimization. If there is noticeable change during the search, it should be considered for update for the next cycle.

III. Nonlinear Model Predictive Control Formulation

A. NMPC Formulation

Following a standard NMPC formulation[3, 17] the receding-horizon control problem P_N is defined as

$$\mathrm{P}_N = \min_{\boldsymbol{u}} \left\{ V\left(\boldsymbol{x}, \boldsymbol{u}\right) \right\} \tag{20}$$

where the cost function $V\left(\boldsymbol{x}_0, \boldsymbol{u}\right)$ is constructed as

$$V\left(\boldsymbol{x}_0, \boldsymbol{u}\right) = \sum_{k=1}^{N} l\left(\boldsymbol{x}_{k+1}, \boldsymbol{u}_k\right) \tag{21}$$

and where $\boldsymbol{x}_k \in \mathrm{R}^n$ and $\boldsymbol{u}_k \in \mathrm{R}^m$ are the state and control vectors at each instant k, with $\boldsymbol{x}_0 \in \mathrm{R}^n$ the system state vector at the beginning of the minimization. The control and state sequences must satisfy the following constraints

$$\begin{aligned} \boldsymbol{u}_k &\in \boldsymbol{U} \\ \boldsymbol{x}_k &\in \boldsymbol{X} \end{aligned} \tag{22}$$

where usually $\boldsymbol{U}$ is a convex, compact subset of R^m an $\boldsymbol{X}$ a convex, closed subset of R^n, containing their origins[3]. These constraints are imposed by the nature of the real system under control, as it has inherent limitations in the size and rate of change of its signals. These sets normally contain equality and inequality constraints accounting for maximum and minimum values and their dynamics. For this research the function $l\left(\boldsymbol{x}_{k+1}, \boldsymbol{u}_k\right)$ is defined as

$$l\left(\boldsymbol{x}_{k+1}, \boldsymbol{u}_k\right) = \boldsymbol{x}_{k+1}{}^T \boldsymbol{Q}_{k+1}\, \boldsymbol{x}_{k+1} + \boldsymbol{u}_k{}^T \boldsymbol{R}_k\, \boldsymbol{u}_k \tag{23}$$

leaving the cost function $V\left(\boldsymbol{x}_0, \boldsymbol{u}\right)$ as

$$V\left(\boldsymbol{x}, \boldsymbol{u}\right) = \sum_{k=1}^{N} \boldsymbol{x}_{k+1}{}^T \boldsymbol{Q}_{k+1}\, \boldsymbol{x}_{k+1} + \boldsymbol{u}_k{}^T \boldsymbol{R}_k\, \boldsymbol{u}_k \tag{24}$$

where the matrix $\boldsymbol{Q}_k \in \mathrm{R}^{n \times n}$, positive semi-definite, and $\boldsymbol{R}_k \in \mathrm{R}^{m \times m}$, positive definite, could be time-variant. The objective of these matrices is to impose relative penalization among the states and the inputs. As shown in Eq. (21) the time index is restarted each cycle given the time invariance of the system. And the length of the horizon, where the numerical minimization is to be preformed, is given by N time steps ahead. This parameter is a very important design element, having a direct impact in the numerical search.

It is interesting to realize that this formulation collapses into the famous linear quadratic controller (LQR) when the system under control is linear, which has a nicely analytical solution approach avoiding falling into the computational expenses of any numerical search.

For an arbitrary cost function, as opposed to a quadratic function, an analytical solution formulation is hard to find, being necessary to perform numerical calculations during the search. At each cycle the output of the search algorithm is the control sequence $\boldsymbol{u} = \left\{ \boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_N \right\}$ that minimizes the cost function $V\left(\boldsymbol{x}, \boldsymbol{u}\right)$. The first or first elements of the input sequence are applied to the UAV, depending on the computational power, and then a new cycle is restarted with a new initial condition $\boldsymbol{x}_0$. The best case corresponds to applying only the first element and doing a search at each sample time. This allows finding a "better" closed loop condition, allowing the update of the state measurement each sample time. Failing to do this implies longer periods of open loop condition, as no new measurement is considered while the pre-computed elements of the input sequence are applied. This drawback could be mitigated with an accurate model of the external disturbances.

Given the predictive nature of Eq. (19) future states are unequivocally defined by the initial conditions $\boldsymbol{x}_0$ and $\boldsymbol{u}_0$ and the control input sequence $\boldsymbol{u} = \left\{ \boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_N \right\}$ to be determine. Using Eq. (19) consecutively the future states are

$$
\begin{aligned}
\boldsymbol{x}_1 &= \boldsymbol{f}_d\left(\boldsymbol{x}_0, \boldsymbol{u}_0\right) \\
\boldsymbol{x}_2 &= \boldsymbol{f}_d\left(\boldsymbol{x}_1, \boldsymbol{u}_1\right) \\
&\vdots \\
\boldsymbol{x}_N &= \boldsymbol{f}_d\left(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}\right) \\
\boldsymbol{x}_{N+1} &= \boldsymbol{f}_d\left(\boldsymbol{x}_N, \boldsymbol{u}_N\right)
\end{aligned}
\tag{25}
$$

Solving for $\boldsymbol{x}_{N+1}$, gives

$$
\boldsymbol{x}_{N+1} = \boldsymbol{f}_d(\boldsymbol{f}_d(\cdots \boldsymbol{f}_d(\boldsymbol{f}_d(\boldsymbol{f}_d(\boldsymbol{x}_0, \boldsymbol{u}_0), \boldsymbol{u}_1), \boldsymbol{u}_2)\cdots, \boldsymbol{u}_{N-1}), \boldsymbol{u}_N) \tag{26}
$$

making it apparent that compound function to be minimized depends on $\boldsymbol{x}_0$ and $\boldsymbol{u}_0$, know quantities at each cycle, and on the control input sequence $\boldsymbol{u} = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_N\}$, to be determined. This can be sketched in Figure 5. With this expression the receding-horizon problem now is expressed as

$$
\mathrm{P}_N = \min_{\boldsymbol{u}} \{V(\boldsymbol{u})\} \tag{27}
$$

For nonlinear systems there is no analytical solution, and the numerical solution $\boldsymbol{u} = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_N\}$ is found in an iterative search with large amount of calculations. This is a non-trivial approach that much of its success depends in the choice of the right algorithm and design parameters, including horizon length, and accuracy of algorithm the convergence. The solution is assumed at a determine iteration number, based on a predefined criteria, and the search is stopped without optimizing the convergence, which could take too many iterations. This numerical solution is clearly suboptimal for the given problem formulation, solution gets closer to the optimal as iterations increases. In the present research the number of iterations will be carefully defined, given the direct impact in the feasibility of the problem. Ref. 10, where an inverted pendulum was controlled by a NMPC algorithm, with a horizon of $N = 60$, a rather simple nonlinear model and a sample time of $\Delta T = 25ms$, only four iterations were used.

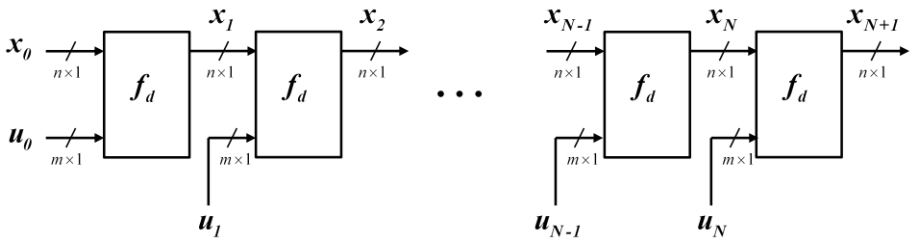

Figure 5. System Composition.

The chosen approach to solve the non-trivial non-convex optimization problem, finding a possibly local minimum, is the Sequential Quadratic Programming (SQP) [18, 19].

For this particular application, and given the nature of the control actuators (control surfaces and engine), the input sequence constraints $\boldsymbol{u}_k \in \boldsymbol{U}$ can be simplified and bounded by the following expression

$$
\boldsymbol{c}(\boldsymbol{u}) < 0 \tag{28}
$$

where it is assumed that this vector equation $\boldsymbol{c}(\boldsymbol{u}) = \left[c_1(\boldsymbol{u}) \quad \cdots \quad c_p(\boldsymbol{u})\right]^T$ is able to accommodate all the existing input constraints. In order to apply the SQP technique, the constraints must be included as part of the cost function with the help of Lagrange multipliers $\boldsymbol{\lambda} = \left[\lambda_1 \quad \cdots \quad \lambda_p\right]^T$. The new objective function is so constructed and shown below

$$
L(\boldsymbol{u}, \boldsymbol{\lambda}) = V(\boldsymbol{u}) - \boldsymbol{\lambda}^T \boldsymbol{c}(\boldsymbol{u}) \tag{29}
$$

where the constraints are added only when they are valid, this is when they are violated, being added as equality constraints. Otherwise they are removed from the augmented objective function, which is evaluated based on the sign of the respective multipliers.

SQP solves the non-convex problem with a series of sub-problems each one minimizing a quadratic approximation of the objective function with a linearized approximation of the constraints. In the case of no valid constraints for a particular iteration SQP reduces to a Newton-type numerical method. Adapting the SQP method from Ref. 19 to the present case, at each iteration the following correction is made

$$\begin{bmatrix} \boldsymbol{u}^{i+1} \\ \boldsymbol{\lambda}^{i+1} \end{bmatrix}_{(mN+p)\times 1} = \begin{bmatrix} \boldsymbol{u}^{i} \\ \boldsymbol{\lambda}^{i} \end{bmatrix}_{(mN+p)\times 1} + \begin{bmatrix} \Delta\boldsymbol{u}^{i} \\ \Delta\boldsymbol{\lambda}^{i} \end{bmatrix}_{(mN+p)\times 1} \tag{30}$$

where the subscript $i = 1, 2, \cdots$ indicates the number of iterations and p the number of inequality constraints valid at iteration i, and where $\left[\boldsymbol{u}_1^i; \boldsymbol{u}_N^i; \cdots; \lambda_1^i; \cdots; \lambda_p^i\right]$ is a column vector in R^{mN+p} (borrowing the column notation from Matlab[20]). As a main feature of the SQP method, the constraints need to be checked every iteration in order to remove the ones inactive, and based on it, update Eq. (30). The corrections are obtained by solving the following system of equations (where the superscript i was omitted for better clarity)

$$\begin{bmatrix} \nabla_{uu}^2 V & -\boldsymbol{A}^T \\ \boldsymbol{A} & 0 \end{bmatrix}_{(mN+p)\times(mN+p)} \begin{bmatrix} \Delta\boldsymbol{u}^{i} \\ \Delta\boldsymbol{\lambda}^{i} \end{bmatrix}_{(mN+p)\times 1} = \begin{bmatrix} -\nabla_u V + \boldsymbol{A}^T\boldsymbol{\lambda} \\ \boldsymbol{c} \end{bmatrix}_{(mN+p)\times 1} \tag{31}$$

where the matrix $\boldsymbol{A}$ is the Jacobian of the constraints vector $\boldsymbol{c}$ and is defined as

$$A^T(\boldsymbol{u}) = \left[\nabla c_1(\boldsymbol{u}) \quad \cdots \quad \nabla c_p(\boldsymbol{u})\right] \tag{32}$$

Remains the calculation of the Hessian and the gradient of $V(\boldsymbol{u})$, $\nabla_{uu}^2 V$ and $\nabla_u V$, respectively, in order to solve Eq. (31). Noticing the quadratic symmetry of the cost function $V(\boldsymbol{u})$ (see Eq. 24), then it is expressible as[10]

$$V(\boldsymbol{u}) = \boldsymbol{e}^T(\boldsymbol{u})\boldsymbol{e}(\boldsymbol{u}) \tag{33}$$

where the error vector $\boldsymbol{e}(\boldsymbol{u})$ is defined as

$$\boldsymbol{e}(\boldsymbol{u}) = \begin{bmatrix} \boldsymbol{Q}_2^{1/2}\boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{Q}_{N+1}^{1/2}\boldsymbol{x}_{N+1} \\ \boldsymbol{R}_1^{1/2}\boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{R}_N^{1/2}\boldsymbol{u}_N \end{bmatrix}_{(nN+mN)\times 1} \tag{34}$$

Now, the Hessian and gradient of the cost function can be extracted starting from a first order Taylor expansion of the error vector $\boldsymbol{e}(\boldsymbol{u})$

$$e(\boldsymbol{u}+\Delta\boldsymbol{u}) \approx \boldsymbol{e}(\boldsymbol{u}) + \boldsymbol{J}(\boldsymbol{u})\Delta\boldsymbol{u} \tag{35}$$

where is $\boldsymbol{J}(\boldsymbol{u})$ Jacobian matrix of $\boldsymbol{e}(\boldsymbol{u})$, defined as

$$J(u)=\begin{bmatrix} Q_2^{1/2}\frac{\partial x_2}{\partial u_1^T} & 0 & 0 & \cdots & 0 \\ Q_3^{1/2}\frac{\partial x_3}{\partial u_1^T} & Q_3^{1/2}\frac{\partial x_3}{\partial u_2^T} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ Q_N^{1/2}\frac{\partial x_{N+1}}{\partial u_1^T} & Q_N^{1/2}\frac{\partial x_{N+1}}{\partial u_2^T} & Q_N^{1/2}\frac{\partial x_{N+1}}{\partial u_3^T} & \cdots & Q_N^{1/2}\frac{\partial x_{N+1}}{\partial u_N^T} \\ R_1^{1/2} & 0 & 0 & 0 & 0 \\ 0 & R_2^{1/2} & 0 & 0 & 0 \\ 0 & 0 & R_3^{1/2} & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & R_N^{1/2} \end{bmatrix} \tag{36}$$

Applying Eq. (35) into Eq. (33)

$$\begin{aligned} V(u+\Delta u) &= \left(e(u)+J(u)\Delta u\right)^T\left(e(u)+J(u)\Delta u\right) \\ &= e^T(u)e(u)+2\Delta u^T J(u)^T e(u)+\Delta u^T J^T(u)J(u)\Delta u \end{aligned} \tag{37}$$

The gradient is identified as $2J^T(u)e(u)$ and the Hessian as $2J^T(u)J(u)$, looking at Eq. (37) as a second order Taylor expansion of the cost function $V(u)$. So gradient and Hessian can be obtained by forming the error vector $e(u)$ and the Jacobian matrix $J^T(u)$.

B. NMPC Numerical Solution Technique

In order to calculate the elements of both the error vector $e(u)$ and the Jacobian $J(u)$, prediction in time is needed, hence the name of MPC. As suggested by Eq. (30) an initial value for the numerical search is needed, for both the control vector u^1 and for the multiplier vector λ^1. This is rather arbitrary but helps the numerical search to start from the previous values obtained in the previous cycle, speeding the search. At the beginning the multipliers are set to zero.

The error vector $e(u)$ is obtained by simply running the prediction ahead in time N steps using Eq. (25) and the initial values for the control signal u^1. The case of the Jacobian $J(u)$ is more involved. Partial variations of the states x_k need to be calculated, with respect to variations in the control vector u_k which is numerically computed. This means discretizing the partial derivatives.

IV. Simulations Results

The main purpose of this simulation is to prove the ideas developed in this research. Specially the ability of the NMPC to provide an integrated task of guidance, navigation and control. For comparison, a previous controller design[11], an H-infinity controller, is employed.

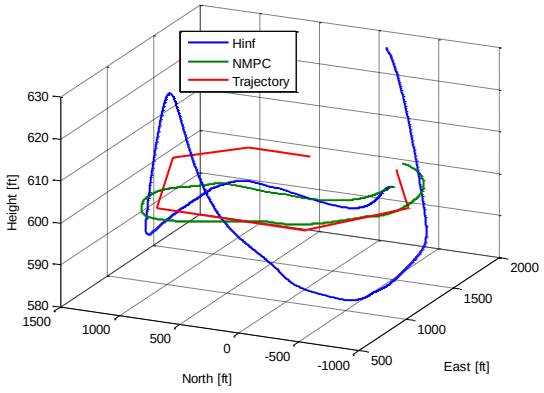

Figure 6a. 3D View of Flight.

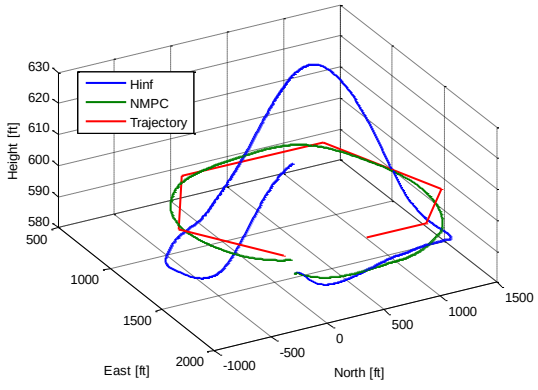

Figure 6b. 3D View of Flight.

Figure 6 show two different 3D views of the trajectory following by both designs. The initial conditions are set identically for both controllers and for the aircraft. This is a flight of 43 seconds, starting slightly off the trajectory. Both graphs show better performance of the NMPC in both lateral and longitudinal planes over the H-infinity. It should be kept in mind that given the highly stable nature of the One-third Scaled YAK-54 RC model used here, NMPC does not show a huge improvement with respect to H-infinity, already giving acceptable results, and its inherent merits should be highlighted when controlling a nonlinear aircraft in a wider range of operation.

The horizon length N chosen is 10 samples, with a sample time ΔT of 0.1 seconds (implying a flight time horizon of 1 second) and using only one iteration (see last row in Table 1). As it is shown, the algorithm takes an average real time of 0.085 seconds, which is below the sampling time (basic requirement of the algorithm to be feasible), but close to it. The code was written in Matlab[20] and run in a standard desktop computer, so it should be able to be optimized before migrating it to an embedded microcontroller, in order to increase the gap between the times, or even allowing a higher sampling rate, higher horizon length and more iteration, which is desirable.

Table 1 shows the searching time (last column) for different sample times, iterations and horizon length combinations. The vast majority of time consuming computations are located in the Hessian calculations.

Table 1. NMPC Algorithm Time (based on average of runs)

Horizon Length N (samples)	Sample Time ΔT (seconds/samples)	Flight Time Horizon $N \cdot \Delta T$ (seconds)	Iterations (times)	Algorithm Time (seconds)
50	0.05	2.5	3	3.8
20	0.1	2	1	0.22
10	0.05	0.5	3	0.2
10	0.05	0.5	2	0.14
10	0.05	0.5	1	0.086
10	0.1	1	3	0.2
10	0.1	1	2	0.14
10	0.1	1	1	0.085

The optimal solution as stated by the MPC theory implies an infinite number of iterations, until the cost function $\boldsymbol{V}(\boldsymbol{u})$ is minimized. This is obviously impracticable, so a finite number needs to be chosen. In Ref. 10 the number was four. In the present research the maximum iterations tested were three, as indicated in Table 1. Even though the more the better, one iteration was acceptable in terms of performance.

Table 2 shows the cost function $\boldsymbol{V}(\boldsymbol{u})$ value for each number of iterations and for the H-infinity solution. This is based on a horizon length of 100 samples and a sample time of 0.05 seconds simulation (not shown in Table 1). It should be understood that in general these values are highly dependent on the choice of weighting matrices, as is evident in Eqs. (23) and (24), besides the algorithm parameters discussed before, and also in the initial conditions of the runs.

Table 2. Cost Function Values

Method	Number of iterations	Cost Function $\boldsymbol{V}(\boldsymbol{u})$
H-infinity	-	2,174
NMPC	1	625
NMPC	2	542
NMPC	3	538

Handling constraints is one of the most attractive features of MPC and NMPC. It is the MPC algorithm that searches the minimum of the cost function explicitly not violating the constraints. The result is a commanded control signals that is not going to hit the actuator limits. In general linear controllers, like H-infinity, are design based on a linear representation of the system to control and does not capture these nonlinearities. The most probable effect is wind-up, causing loss of stability of the entire system or at least degradation on the performance. Figure 7 shows the behavior of the engine's throttle subject to an increase of 30% in the desired trajectory speed. While NMPC command does not surpass the limit (100% of the engine's throttle), the H-infinity goes over it winding-up. This effect is evidenced in the time the controller takes to go back to "normal" values (after second 10).

For this particular case where control surfaces deflection are hardly saturated, only the upper limit of the throttle was included in the NMPC algorithm, which is more likely to hit the limit, though reducing the number of possible constraints to be handled by it. Their time consuming is not significant, so the inclusion of more constraints should not severely affect the overall computing time.

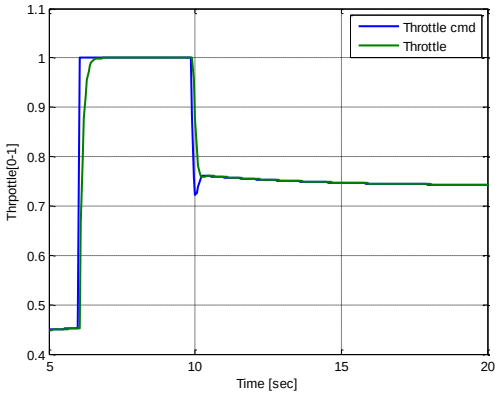

Figure 7a. Throttle Behavior with NMPC.

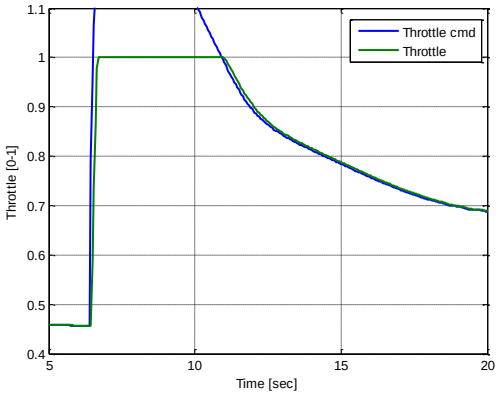

Figure 7b. Throttle Behavior with H-infinity.

V. Conclusion

The problem of guidance, navigation and control normally treated separately are integrated into one. The attitude guidance necessary to follow a predefined trajectory are formulated as states and incorporated into the state vector of the system. The model of the UAV is augmented with two new states, which correspond to angle errors in both longitudinal and lateral planes. The minimization of these two new states will force the UAV to approach asymptotically to the trajectory.

The formulation of these angles includes the information of the segment to track making it possible to reduce the control problem from a reference tracking to a regulation problem. This is especially useful in MPC and NMPC given its predictive nature which besides the prediction of the future states, which is done with the UAV model, it needs the knowledge of the future reference signal to perform the numerical search. Converting the problem to a regulation problem where the objective is to reduce to zero the signals of interest, avoids this difficulty.

Using an integrated solution for guidance, navigation and control evades any possibility of undesired cross-coupling between inner and outer loops, which although close related, are designed separately in conventional approaches.

Having an updated and accurate nonlinear model of the UAV, the NMPC is able to provide a good solution for controlling the UAV, taking care of constraints and time changing conditions for a broader range of operation.

Acknowledgments

The authors would like to thank CReSIS and NSF grant # NSF40011 for their support.

References

[1]Nichols, R., Reichert, R. and Rugh, W., "Gain Scheduling for H-Infinity Controllers: A Flight Control Example," *IEEE Transactions on Control Systems Technology*, Vol. 1, No. 2, 1993.

[2]Chandler, P. R., Patcher, M., and Mears, M., "System Identification for Adaptive and Reconfigurable Control," *AIAA Journal of guidance, Control and Dynamics*, Vol. 18, No. 3, 1995.

[3]Mayne, D. Q., Rawlings, C., Rao, C., and Scokaert, P. O. M., "Constrained Model Predictive Control: Stability and Optimality," *Automatica*, Vol. 36, No. 6, 2000.

[4]Sutton, G., and Bitmead, R., "Experiences with Model Predictive Control applied to a Nonlinear Constrained Submarine," *Proceedings of the 37th IEEE Conference on Decision & Control*, Vol. 2, Tampa, Florida, 1998, pp. 1370-1375.

[5]Kim, H. J., Shim, D. H., and Sastry, S., "Nonlinear Model Predictive Tracking Control for Rotorcraft-based Unmanned Aerial Vehicles," *Proceedings of the American Control Conference*, Vol. 5, Anchorage, Alaska, 2002, pp. 3576-3581.

[6]Castillo, C. L., Moreno, W., and Valavanis, K. P., "Unmanned Helicopter Waypoint Trajectory Tracking using Model Predictive Control," *Proceedings of the 15th Mediterranean Conference on Control and Automation*, Athenas, 2007, pp. 1-8.

[7]Frew, E., "Comparison of Lateral Controllers for Following Linear Structures Using Computer Vision," *American Control Conferences*, Minneapolis, 2006.

[8]Eklund, J. M., Sprinkle, J., and Sastry, S., "Implementing and Testing a Nonlinear Model Predictive Tracking Controller for Aerial Pursuit/Evasion Games on a Fixed Wing Aircraft," *American Control Conferences*, Vol. 3, 2005, pp. 1509-1514.

[9]Kang, Y., and Hedrick, J. K., "Linear Tracking for a Fixed-Wing UAV Using Nonlinear Model Predictive Control," *IEEE Transactions on Control Systems Technology*, Vol. 17, No. 5, 2009.

[10]Mills, A., Wills, A., and Ninness, B., "Nonlinear Model Predictive Control of an Inverted Pendulum," *American Control Conferences*, St. Louis, 2009, pp. 2335-2340.

[11]Garcia, G., Keshmiri, S., and Colgren, R., "Advanced H-Infinity Trainer Autopilot, "*AIAA Modeling and Simulation Technologies Conference*, Toronto, Ontario, 2010.

[12]Garcia, G., and Keshmiri, S. "Design and Application of an Extended Kalman Filter in a Flight System Control Development, "*AIAA Infotech@Aerospace*, St. Louis, Missouri 2011.

[13]Stevens, B., and Lewis, L., *Aircraft Control and Simulation*, 2nd ed., John Wiley & Sons, Hoboken, NJ, 2003.

[14]Leong, H. I., Jager, R., Keshmiri, S., and Colgren, R., "Development of a Training Platform for UAVs Using a 6DOF Nonlinear Model with Flight Test Validation," *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Honolulu, Hawaii, 2008.

[15]Keshmiri, S., Leong, H. I., Jager, R., and Hale, R., "Modeling and Simulation of the YAK-54 Scaled Unmanned Aerial Vehicle Using Parameter System Identification," *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Honolulu, Hawaii, 2008.

[16]Park, S., Deyst, J., and How, J., "A New Nonlinear Guidance Logic for Trajectory Tracking," *AIAA Guidance Navigation and Control Conference and Exhibit*, Providence, Rhode Island, 2004.

[17]Bemporad, A., and Morari, M., *Robustness in Identification and Control*, Springer, London, 1999, Chap. 7.

[18]Boggs, P. T., and Tolle, J. W., "Sequential Quadratic Programming," *Acta Numerica*, pp. 1-51, 1996.

[19]Nocedal, J., and Wright, S. J., *Numerical Optimization*, Springer-Verlag, New York, 1999.

[20]The MathWorks, Matlab Software Package.