POLITECNICO DI TORINO
Repository ISTITUZIONALE

NMPC and genetic algorithm based approach for trajectory tracking and collision avoidance of UAVs

Original
NMPC and genetic algorithm based approach for trajectory tracking and collision avoidance of UAVs / DE FILIPPIS, Luca; Guglieri, Giorgio. - In: INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING AND APPLICATIONS. - ISSN 1751-648X. - STAMPA. - 5:1(2013), pp. 173-183. [10.1504/IJICA.2013.055935]

Availability:
This version is available at: 11583/2505187 since:

Publisher:
Inderscience Enterprises Ltd.

Published
DOI:10.1504/IJICA.2013.055935

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

20 April 2024

NMPC and genetic algorithm-based approach for trajectory tracking and collision avoidance of UAVs

Luca De Filippis* and Giorgio Guglieri

Dipartimento di Ingegneria Meccanica e Aerospaziale,
Politecnico di Torino,
Torino 10129, Italy
E-mail: luca.defilippis@polito.it
E-mail: giorgio.guglieri@polito.it
*Corresponding author

Abstract: Research on unmanned aircraft is improving constantly the autonomous flight capabilities of these vehicles in order to provide performance needed to employ them in even more complex tasks. UAV path planning (PP) system plans the best path to perform the mission and then it uploads this path on the flight management system (FMS) providing reference to the aircraft navigation. Tracking the path is the way to link kinematic references related to the desired aircraft positions with its dynamic behaviours, to generate the right command sequence. This paper presents a non-linear model predictive control (NMPC) system that tracks the reference path provided by PP and exploits a spherical camera model to avoid unpredicted obstacles along the path. The control system solves online (i.e., at each sampling time) a finite horizon (state horizon) open loop optimal control problem with a genetic algorithm. This algorithm finds the command sequence that minimises the tracking error with respect to the reference path, driving the aircraft far from sensed obstacles and towards the desired trajectory.

Keywords: model predictive control; trajectory tracking; collision avoidance; genetic algorithms; UAV.

Reference to this paper should be made as follows: De Filippis, L. and Guglieri, G. (2013) 'NMPC and genetic algorithm-based approach for trajectory tracking and collision avoidance of UAVs', *Int. J. Innovative Computing and Applications*, Vol. 5, No. 3, pp.173–183.

Biographical notes: Luca De Filippis is a Graduate Research Assistant affiliated with the Department of Mechanics and Aerospace of Politecnico di Torino. He received his PhD degree in 2012 on Path Planning and Collision Avoidance Algorithms for UAVs. His research interests include path planning and collision avoidance systems for unmanned vehicles, control-system design, space dynamics and control, optimisation techniques and evolutionary algorithms, human supervising (HS) and human machine interfaces (HMIs).

Giorgio Guglieri is an Associate Professor of Flight Mechanics at Politecnico di Torino where he belongs to the steering committee for the doctoral course in aerospace engineering. He is currently the Quality Manager for the EASA Part66 certification (Italian Aviation Authority – ENAC) for the Aerospace Engineering degree. He is the author of several articles and reviewer for several aerospace international journals. He is involved in several research activities on flight control system design, HMI interfaces design and tele-operations, UAV design, optimal path planning and collision avoidance, advanced multi-task PC-based HMI, optimal control design based on genetic algorithms (bounded by handling qualities requirements and mission constraints), simulation of spacecraft re-entry vehicle dynamics, lunar landing, and rendezvous/docking, including GNC modelling.

This paper is a revised and expanded version of a paper entitled 'NMPC and genetic algorithm-based approach for trajectory tracking of UAVs' presented at the 5th International Conference on Bioinspired Optimisation Methods and their Applications, Bohinj, Slovenia, 24–25 May 2012.

1 Introduction

Unmanned aerial vehicles (UAV) represent one of the most studied fields of research in aeronautics and robotics. Characteristics and performance of these aircraft excited wide industrial and academic engagement to improve their autonomy in order to cope with even more complex missions. Planning the path and control the UAV in order to follow the desired trajectory accomplishing with its mission is a challenging task. Different PP and tracking systems have been developed, that exploit wide range of techniques providing encouraging results. However, UAV dynamics is

non-linear and control systems able to optimise aircraft performance are desirable in order to plan complex trajectories and in turn solve challenging tracking problems.

In the last decades, wide research has been done on receding horizon control (RHC) techniques (Garcia et al., 1989) to cope with:

a intrinsically non-linear dynamic systems

b high quality requirements

c growing use of robotic systems in any working division.

Linear models are not sufficient to describe adequately any dynamic system particularly when performance close to constraint boundaries are desirable. Non-linear model predictive control (NMPC) allows the use of accurate models and more complex problem formulations that provide better prediction and optimisation (Allgöwer et al., 2004). Deep investigation is carried on to consolidate theoretical background and to proof fundamental properties of NMPC (i.e., feasibility, stability and robustness). In ground and flight robotics, many applications to tracking and collision avoidance problems can be found. Sprinkle et al. (2004) presented a NMPC system applied to trajectory tracking for pursuit/evasion games between two fixed wing UAVs. This control technique works on the 'planning' level. In other words an optimal trajectory is provided to the autopilot in order to perform mission tasks. Our interest is on the other hand in a control technique able to work at lower level generating optimal commands so that the desired path is tracked with the UAV. Kang and Hedrick (2009) implemented an interesting NMPC system to cope with trajectory tracking problems. They designed a high-level tracking controller for a small fixedwing UAV and they studied close-loop stability extracting some performance properties of the control strategy adding an outer loop to the inner control loop. Theoretical and mathematical support is fundamental to convert a simple problem solving approach in a deeper investigation able to provide general concepts on a control technique, however, this approach to the problem requires simplifications that could mismatch with a real implementation problem.

Genetic algorithms (GAs) are optimisation techniques based on biological principles of natural selection. They are able to cope with any kind of problem even when its features are not completely understood. Another important merit of evolutionary optimisation is the wide range of potential solutions that the algorithm is able to evaluate with respect to classic formulations (Holland, 1992). These features make GA particularly useful when optimisation problems like the one met in NMPC formulations with large state and solution spaces need to be solved. Tian et al. (2005) combine MPC with GA to implement an algorithm for UAV cooperative search. The authors subdivide the environment with hexagonal cells and use MPC to predict future states of the UAV swarm in order to minimise searching area through the best aircraft distribution over it.

This paper describes a novel approach to trajectory tracking and collision avoidance that matches GA features with NMPC. To the authors knowledge applications of NMPC tested on real problems exploit model-predictive to tackle collision avoidance and formation flight. In these works, predictive features are needed to generate optimised trajectories to avoid unpredicted obstacles or coordinate the path of UAV swarms preventing collisions. On the other hand, studies where MPC is applied to trajectory tracking are oriented to investigate the predictive-control properties in terms of stability and robustness, simplifying the reference trajectory and the tracking strategy in favour of theoretical contributions. Then the tracking system described here aims to cope with real applications where the PP system cooperates with the lower level navigation system in order to steer the UAV over the best trajectory in complex and performance demanding conditions.

The non-deterministic approach of evolutionary optimisation together with critical time constraints for trajectory tracking of UAVs discouraged applications of NMPC with GA. However, new theoretical contributions and wide experimental results obtained in these years encouraged the authors to fuse these two techniques. As a matter of fact gradient-based optimisation, commonly used on MPC, is robust and theoretically solid but it forces to simplify the problem formulation, preventing the application to more complex and critical tracking tasks. SBX crossover and polynomial mutation are chosen as genetic operators for the GA here described. SBX is a well known crossover method developed by Deb and Agarwal (1995), more advanced solutions were developed in the following years. Ono and Kobayashi (1997) implemented the unimodal normally distributed crossover operator (UNDX), where three parent solutions are used to create two or more children solutions. Then Herrera et al. (1996) introduced real coded cross-over based on fuzzy connectives and Deb et al. (2002) proposed another real coded crossover called parent centric crossover (PCX) operator. Even if more advanced crossover operators are described in literature, providing comparison about their performance. SBX is still used in specific applications linked to path planning (PP) of UAVs (Pehlivanoglu et al., 2007) and it is chosen here as a preliminary approach that will be improved with further investigations.

2 Reference path

The higher-level PP system that provides the path the aircraft must follow exploits Kinetamic A* algorithm (De Filippis and Guglieri, 2012). The output of the PP system is a sequence of waypoints used as a reference in order to steer the aircraft towards the path (Figure 1). Kinematic A* includes a simple kinematic model of the vehicle to evaluate the moving cost between waypoints in a tri-dimensional environment. Movements are constrained with minimum turn radius and maximum rate of climb.

Figure 1 Valley way out test (3D view) (see online version for colours)

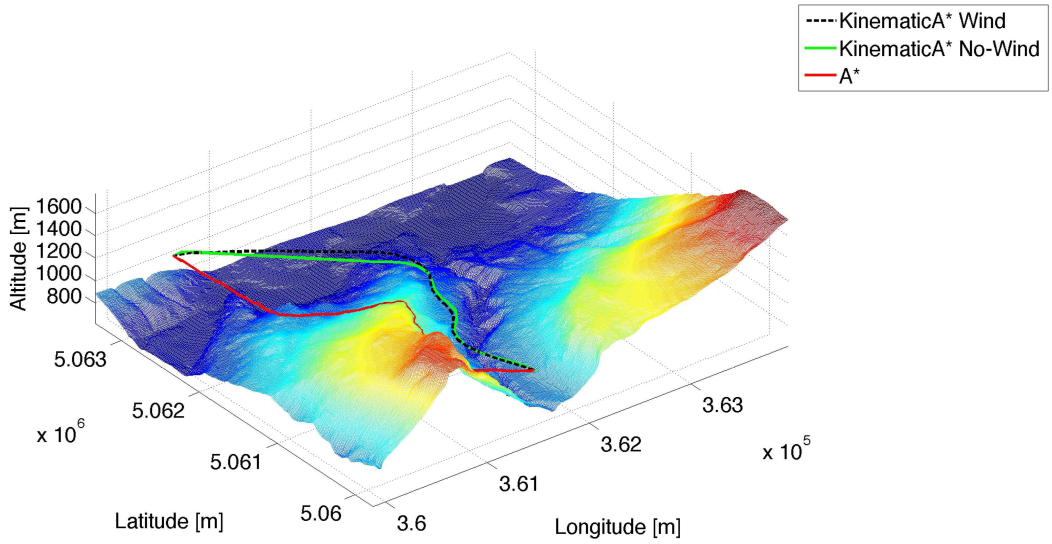
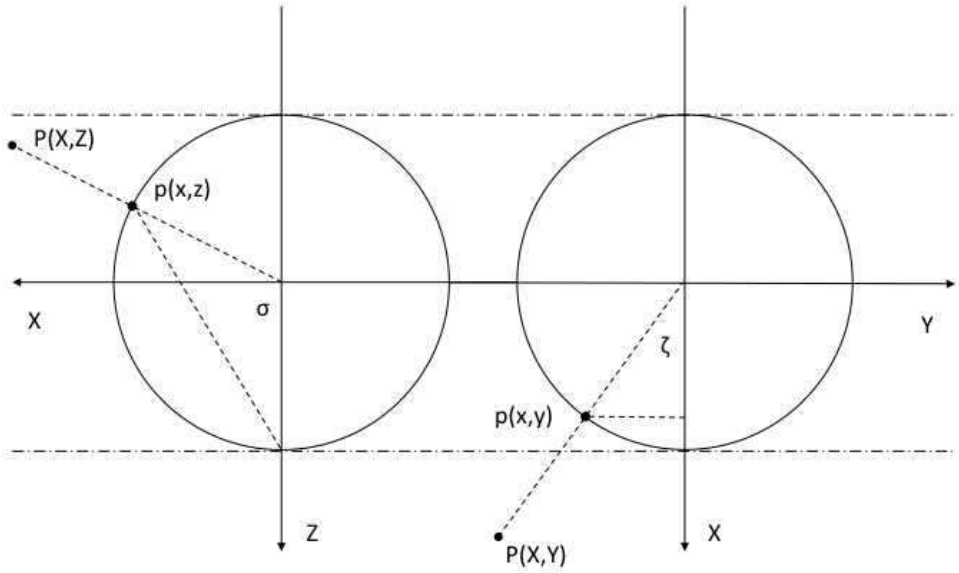


Figure 2 Feature representation in the image surface



The aircraft model used to generate the waypoints sequence is given by equations:

$$\begin{cases} \dot{X} = V \cdot \cos\chi \cdot \cos\gamma_{\max} \cdot w \\ \dot{Y} = V \cdot \sin\chi \cdot \cos\gamma_{\max} \cdot w \\ \dot{Z} = V \cdot \sin\gamma_{\max} \cdot w \\ \dot{\chi} = \dfrac{V}{R} \cdot u \end{cases} \quad (1)$$

where (X, Y, Z) is the position vector and V is the constant ground speed of the aircraft. Command variable w $(-1 \leq w \leq 1)$ modules the climb angle between its minimum and maximum values coincident with $\gamma_{\max}$. The second command u $(-1 \leq u \leq 1)$ on the other hand modules the turn speed with respect to the minimum turn radius R.

KA* output is a waypoint sequence with each point represented by the state vector $(X, Y, Z, \gamma, \chi, V)$. The NMPC system predicts future aircraft positions over the prediction horizon, then the tracking task is performed trying to reduce the error between predicted and reference positions. For each time step a receding fraction of the reference path is extracted by the full path and provided to the NMPC system as a reference. NMPC finds the optimal command which reduces the tracking error.

3 Sense and avoid system

S&A strategy is inspired by visual servoing techniques commonly adopted in robotics. The robotic system is controlled moving a set of visual features (designed from image measurements) on the image plane so that a desired reference is followed. Mathematically, visual servoing task is the reduction to zero of an error expressed in the image as the difference between actual feature and desired one.

A spherical camera is assumed to represent the sensing system exploited here. As a matter of fact large perceptual

field is fundamental to cope with collision avoidance problems and standard perspective cameras do not have a sufficient view angle. Then spherical view do not need to keep features in the field of view providing a wider controllability.

In spherical cameras image plane becomes a surface represented by a unit sphere and image features (i.e., intruders) are considered points projected on this sphere. This assumption is quite usual for this kind of approaches. Majority of fixed-wing UAVs have kinematic and dynamic constraints that affect their time to react. To implement a safe recovery manoeuvre the aircraft has to sense the intruder when it is some kilometres far and it has to change immediately its trajectory flying far enough from it. Then at detection stage the obstacle in the image feature is very small and it remains so for the most part of collision avoidance manoeuvre.

Figure 2 shows image unit sphere and projection of a feature on it. Reference system adopted for the camera is assumed coincident with aircraft body frame and located in its CG. A feature in real world is in $P(X, Y, Z)$ position relative to the camera reference system while its projection on the image surface is in $p(x, y, z)$. Parameters identifying feature on the sphere are relative longitude ζ and latitude σ angles. Relative range would be required to determine relative position, but this information can not be achieved from image.

Kinematic relations describe feature vector in image surface $f(\sigma, \zeta)$ and equations that link camera dynamics with feature-vector variation have been determined (Corke, 2010):

$$\begin{Bmatrix} \dot{\sigma} \\ \dot{\zeta} \end{Bmatrix} = [\boldsymbol{J}_{V_c}] \cdot \begin{Bmatrix} V_x^c \\ V_y^c \\ V_z^c \end{Bmatrix} + [\boldsymbol{J}_{\omega_c}] \cdot \begin{Bmatrix} \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{Bmatrix} \tag{2}$$

with:

$$\bar{V}_c = \begin{Bmatrix} V_x^c \\ V_y^c \\ V_z^c \end{Bmatrix} \tag{3}$$

$$\bar{\omega}_c = \begin{Bmatrix} \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{Bmatrix} \tag{4}$$

$$[\boldsymbol{J}_{V_c}] = \begin{bmatrix} -\dfrac{\cos\sigma \cdot \cos\zeta}{R} & -\dfrac{\cos\sigma \cdot \sin\zeta}{R} & \dfrac{\sin\sigma}{R} \\ \dfrac{\sin\zeta}{R \cdot \sin\sigma} & -\dfrac{\cos\zeta}{R \cdot \sin\sigma} & 0 \end{bmatrix} \tag{5}$$

$$[\boldsymbol{J}_{\omega_c}] = \begin{bmatrix} \sin\zeta & -\cos\zeta & 0 \\ \dfrac{\cos\sigma \cdot \cos\zeta}{\sin\sigma} & \dfrac{\cos\sigma \cdot \sin\zeta}{\sin\sigma} & -1 \end{bmatrix} \tag{6}$$

where $\bar{V}_c$ is the camera linear speed vector, $\bar{\omega}_c$ is the camera angular speed vector and R is range. Equation (2) relates feature angles variation with camera dynamics and thanks to the assumptions already expressed in turn also with the aircraft dynamics. This is the sensor model exploited to predict feature behaviour on image surface according with aircraft dynamic evolution. In other words, the sensor model is exploited to predict future aircraft behaviours such that predicted feature would reach a prescribed position on image surface.

A reference feature position is needed to trigger the avoidance manoeuvre, minimising the error between the current and the desired feature position. Then the S&A system has to provide the right reference to the GS in order to move the feature towards the reference position on image surface and in turn avoid the obstacle performing the recovery manoeuvre.

Figure 3 Recovery manoeuvre (see online version for colours)

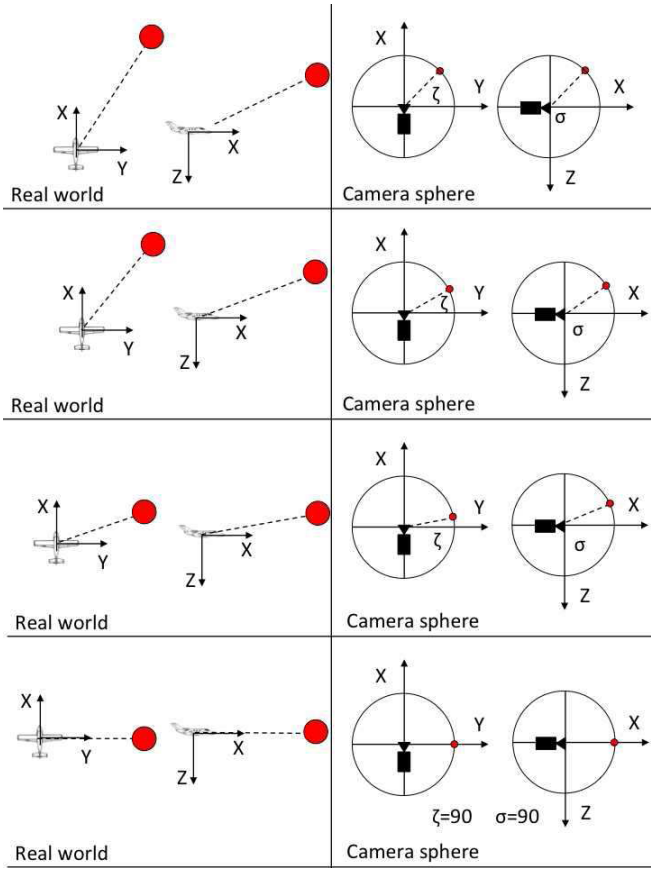


Figure 3 shows a recovery manoeuvre where an obstacle is detected on the right side of the aircraft and slightly above it. The reference feature-vector assigned to perform the manoeuvre is $f(90, 90)$. Then the aircraft is forced to increase the altitude up to drive σ to 90 degrees and turn on the left so that γ reaches the same value.

4 NMPC formulation

NMPC acts solving a finite horizon open-loop optimal control problem in real-time. The cost function is the function minimised with the optimal commands. This function characterises the problem containing variables that

represent the optimisation task. A classical quadratic function has been selected here made of two terms:

- *State error*: this term evaluates the error between reference and predicted states. It is the cost term needed to introduce the mission task inside optimisation problem. As a matter of fact reference states change whether tracking or collision avoidance problems are faced by the aircraft. The PP reference is mainly represented by the desired position provided with KA*, while the collision avoidance manoeuvre involves the desired feature position on the image surface.
- *Command*: this term evaluates the amount of command needed to perform the predicted manoeuvre.

The cost function depends from the initial states, measured with sensors at each time-step and from a predicted control sequence. Indicating with (*) the predicted variables over the prediction horizon (T_p) the task is to find:

$$\min_{\bar{U}^*(\cdot)} J\left(\bar{X}_0, \bar{U}(\cdot)\right) = \int_t^{t+T_p} \left(\hat{\bar{X}}^*(\tau)^T \cdot \boldsymbol{Q} \cdot \hat{\bar{X}}^*(\tau) + \bar{U}^*(\tau)^T \cdot \boldsymbol{R} \cdot \bar{U}^*(\tau) d\tau\right) \tag{7}$$

with:

$$\hat{\bar{X}}^*(\tau) = \bar{X}^*(\tau) - \bar{X}_{ref}(\tau) \tag{8}$$

where $\bar{X}_0 = \bar{X}(t) \in \Re^n$ is the initial-state vector and $\bar{U}^*(\cdot)\Re^m$ is the predicted-command vector. $\bar{X}_\tau^* \in \Re^n$ is the predicted-state vector and $\bar{X}_{ref}(\tau) \in \Re^n$ is the reference-state vector. Finally, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are diagonal matrices of gains weighting state-variables effects over the cost function.

The command horizon T_c defines the horizon of optimal commands generated for each optimisation loop and it commonly differs from T_p. The command strategy is then the other fundamental element of NMPC. The classic command vector of an aircraft contains:

$$\bar{U} = \begin{cases} \delta_e, & \delta_{e_{\min}} \le \delta_e \le \delta_{e_{\max}} \\ \delta_a, & \delta_{a_{\min}} \le \delta_a \le \delta_{a_{\max}} \\ \delta_r, & \delta_{r_{\min}} \le \delta_r \le \delta_{r_{\max}} \\ Th, & 0 \le Th \le 1 \end{cases} \tag{9}$$

The command strategy is just the strategy to build the command signal over the command horizon and in general over the prediction horizon. A linear command variation has been chosen here over the command horizon. Particularly, a piecewise linear function is built over the prediction horizon based on functions:

$$\bar{U}_i^*(\tau) = \bar{U}_{i_0}^* + \bar{A}_i * \tau \; 1 \le i \le n_c \tag{10}$$

where $\bar{U}_i^*(\tau)$ is the ith linear function, $\bar{U}_{i_0}^*$ is the ith initial command value and $\bar{A}_i$ is the ith function slope.

The horizon of commands is then arranged subdividing the time interval in a number of steps (n_c) according with the command horizon and the command frequency (hz_c). As an example with T_p = 2 [s], T_c = 1 [s] and hz_c = 2 [1/s] two linear functions (n_c = 2) are built over the command horizon:

$$\begin{aligned} \bar{U}_1^*(\tau) &= \bar{U}_0 + \bar{A}_1 * \tau \quad t-1 \le \tau \le T_c/2 \\ \bar{U}_2^*(\tau) &= \bar{U}_{T_c/2} + \bar{A}_2 * \tau \quad T_c/2 \le \tau \le T_c \end{aligned} \tag{11}$$

and the command over the time step $T_p - T_c = 1[s]$ given by:

$$\bar{U}^*(\tau) = \bar{U}_{n_c}^*(T_c) \; T_c \le \tau \le T_p - T_c \tag{12}$$

This command sequence has been chosen to guarantee continuity of command functions and in turn of external forces and couples acting on aircraft. Commands generated with (10) are bounded with disequalities in (9) but $\bar{A}_i$ vector must be bounded too. Maximum command variation over unitary time-step is chosen such that:

$$\Delta\bar{U}_{\min}^* \le \bar{A}_i \le \Delta\bar{U}_{\max}^* \tag{13}$$

$$\Delta\bar{U} = \begin{cases} \Delta\delta_{e_{\min}} \le \Delta\delta_e \le \Delta\delta_{e_{\max}} \\ \Delta\delta_{a_{\min}} \le \Delta\delta_a \le \Delta_{r_{\max}} \\ \Delta\delta_{r_{\min}} \le \Delta\delta_r \le \Delta\delta_{r_{\max}} \\ \Delta Th_{\min} \le \Delta Th \le Th_{\max} \end{cases} \tag{14}$$

Solving online the optimisation problem the linear-function slopes in (10) are chosen in order to minimise the cost function.

Defining the state vector as:

$$\bar{X} = \{u \;\; v \;\; w \;\; p \;\; q \;\; r \;\; \phi \;\; \theta \;\; \psi \;\; x_V \;\; y_V \;\; z_V \;\; \omega_p\} \tag{15}$$

the complete optimisation problem prescribes to find:

$$\bar{U}_{opt}^* = f\left(\tau, \bar{U}_0, \bar{A}_{opt}, T_c, T_p, n_c\right) \tag{16}$$

so that equation (7) is satisfied according with system of equations (9) and (14) but also with the aircraft non-linear equations of motion:

$$\dot{\bar{X}} = f\left(\bar{X}, \bar{U}\right) \tag{17}$$

5 GA algorithm

Referring to the problem formulation already presented the individuals representing the solutions are the aircraft command slopes and particularly the chromosomes of each individual are:

$$chromosome = \left[\Delta\delta e, \Delta\delta a, \Delta\delta r, \Delta Th\right] \tag{18}$$

while the fitness function J is described in (7).

Classic GA structure implemented to cope with this problem collects:

- *Population initialisation*:

$$chromosome = \Delta\bar{U}_{\min} + \left(\Delta\bar{U}_{\max} - \Delta\bar{U}_{\max}\right) \cdot \bar{R}d \tag{19}$$

where $0 \le \bar{R}d \le 1$ is a random vector of 4 numbers. For each individual equation (19) provides the four genes

composing the chromosome bounded between its minimum and maximum value. After the initial population is build equation (7) provides the fitness value of each individual.

- Selection: to perform this task half of the total population is randomly selected. Two individuals are randomly chosen and the one with the lower cost is selected for reproduction. Then other two solutions are taken and the selection is repeated up to obtain a number of individuals equal to half of the total population. This technique is called *tournament selection* and it is preferable to select individuals sorted with respect to their fitness value because it introduces a random component into the selection logic.

Algorithm 1 Crossover

Chose a distribution index η_c
Chose a random number $u \in [0; 1)$
if $u \leq 0.5$ **then**
    $\beta_q = (2 \cdot u)^{\frac{1}{\eta_c+1}}$
else
    $\beta_q = \left(\frac{1}{2 \cdot (1-u)}\right)^{\frac{1}{\eta_c+1}}$
end if
for $i = 1.4$ **do**
    $g_{c_1}^i = 0.5 \cdot \left[(1+\beta q) \cdot g_{p_1}^i + (1-\beta q) \cdot q_{p_2}^i\right]$
    $g_{c_2}^i = 0.5 \cdot \left[(1-\beta q) \cdot g_{p_1}^i + (1-\beta q) \cdot q_{p_2}^i\right]$
end for

where $g_{c_j}^i$ is the ith gene of the jth child and $g_{p_j}^i$ is the ith gene of the jth parent.

- *Crossover*: Simulated binary crossover (SBX) (Deb and Agarwal, 1995) is chosen to exchange the genetic pool. This mechanism uses a probability distribution around two parents to create two children. Unlike other methods SBX uses a probability distribution similar to the probability typical of crossover operators used in binary-coded algorithms. The fundamental merit of SBX is its self-adaptive power that guarantees to the offspring to do not narrow near the previous optimal solution (typical phenomenon due to weak diversity inside population). Then with SBX children closer to their parents are more likely to be created and the diversity inside the offspring is proportional to the one inside the previous generation. This is guaranteed fixing a distribution index η_c that can be any positive real number. Large values of η_c increase probability to have children close to their parents and vice-versa. Procedure presented with Algorithm 1 is implemented to create children from their parents with SBX.
- *Mutation*: a polynomial mutation technique based on a probability distribution similar to the SBX one is implemented. The reasons to choose this mutation operator are the same as for the crossover one. Fixing a distribution index η_m from one individual one child is obtained with the Algorithm 2.

Algorithm 2 Mutation

for $i = 1.4$ **do**
    Chose a random number $u_i \in [0, 1)$
    if $u_i \leq 0.5$ **then**
        $\Delta g_c^i = (2 \cdot u)^{\frac{1}{\eta_m+1}} - 1$
    else
        $\Delta g_c^i = 1 - \left(2 \cdot (1-u)\right)^{\frac{1}{\eta_m+1}}$
    end if
    $g_c^i = g_p^i + \Delta g_c^i$
end for

where g_c^i and Δg_c^i are the ith child gene and mutation. While g_p^i is the ith parent gene.

- *Evaluation*: the fitness function used to evaluate the individuals is the cost function provided in (7).
- *Update*: the update scheme is performed joining old and offspring populations and sorting them with respect to the fitness value. Then a set of individuals equal to the population size is chosen and used for the next algorithm cycle.

The offspring population size is half of the total population and its composition is made with a fixed percentage of mutated individuals. As a matter of fact when the selection phase is complete mutation is performed up to obtain the prescribed percentage of individuals over the total amount then crossover is performed up to complete the offspring population.

Convergence condition is satisfied when the algorithm converges to the same solution for a prescribed number of times. In more details each time the population is updated the cost value of each individual is introduced inside a vector:

$$\bar{F} = \left[f_1, f_2, f_3, ..., f_{N-1}, f_N\right] \tag{20}$$

and the following equation inequality is evaluated:

$$f_{best} - \frac{\sum_{i=1}^{N-1} f_i}{N} \leq Toll \tag{21}$$

where f_i is the cost linked to the ith individual, *fbest* is the cost linked to the best individual, N is the population size and *Toll* is a fixed tolerance, set to 10^{-15}.

When inequality (21) is verified, the algorithm convergence to a local minimum is assumed (i.e., the whole population has the same cost value). However, the population is further updated to explore, through the mutation operator, solution-space regions far from the one where the local minimum has been found. If the algorithm

converges to the same local minimum for a given number of times (i.e., five times for the simulations presented in the results), full convergence is assumed.

6 Results

To implement the tests Kinematic A* algorithm is exploited to generate the reference path on the DEM of a mountainous area. The area in the North-West of Italy is inside the alpine region 'Valle d'Aosta' and it includes wide orographic obstacles. The aircraft is forced to climb and turn all along the path to maintain distance from ground because of continuous-obstacle distribution.

GA has 48 individuals that compose population and convergence tolerance fixed to 10^{-3}. NMPC has 40 Hz integration frequency, 1 Hz command frequency, 1 s integration horizon and 1 s command horizon. Simulation starts with the aircraft in trim condition and command bounds are ± 0.5 rad for elevator, aileron and rudder (equal to ± 29 deg), 0 – 1 for throttle. Command slopes are then ± 1 rad/s for each aerodynamic surface and ± 1 for throttle.

6.1 Tracking task

Figure 4 represents the tracking error on the three axis that evidences altitude loss during first 2 seconds and that shows the system is able to track the reference path with high accuracy. The real trajectory in red completely overlaps the reference one in green. The error on the *Z*-axis is high when the simulation begins because of small trim-condition inaccuracies. However, the altitude mismatch is just 2 metres and it is quickly reduced by the control system.

Figure 4 Comparison between reference and real path on each round axis (see online version for colours)

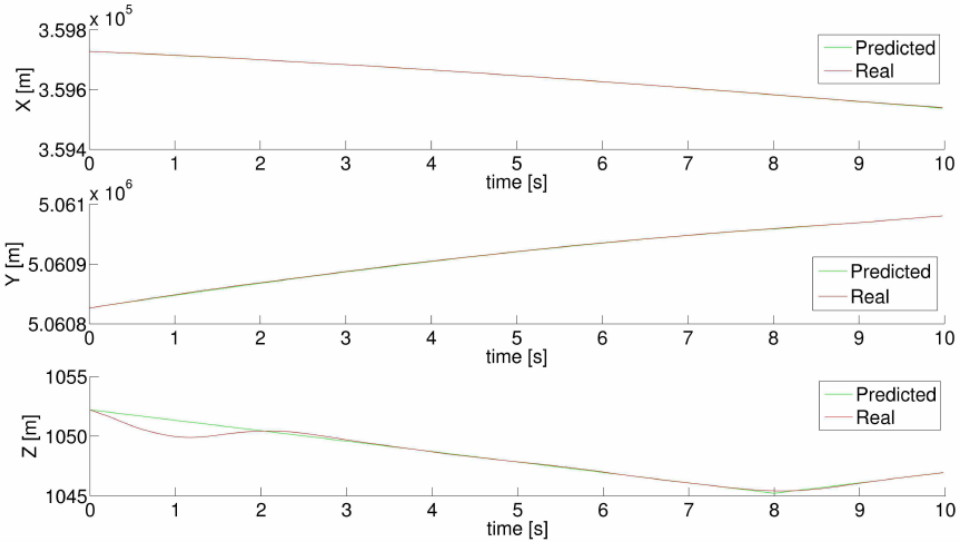


Figure 5 Comparison between predicted and real speed (see online version for colours)

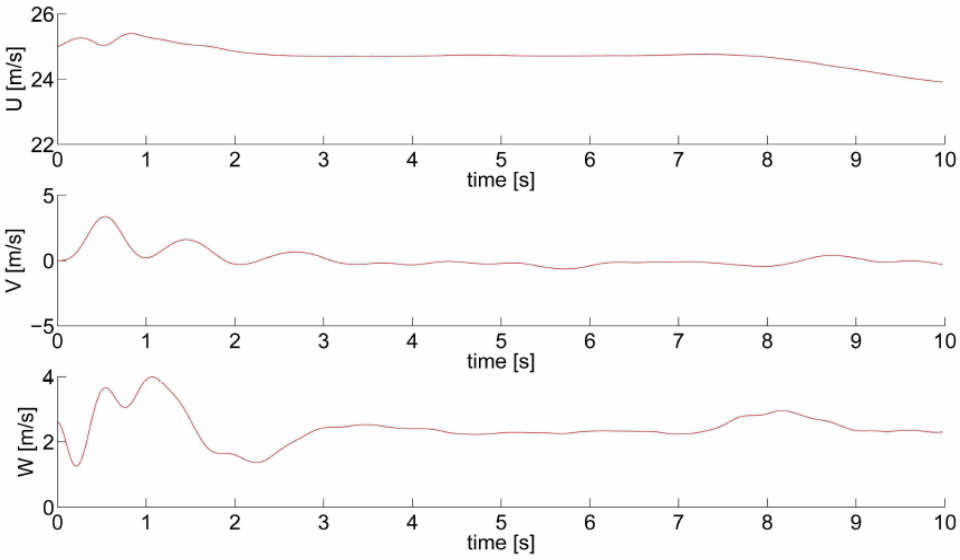


Figure 6 Comparison between predicted and real angular rate (see online version for colours)

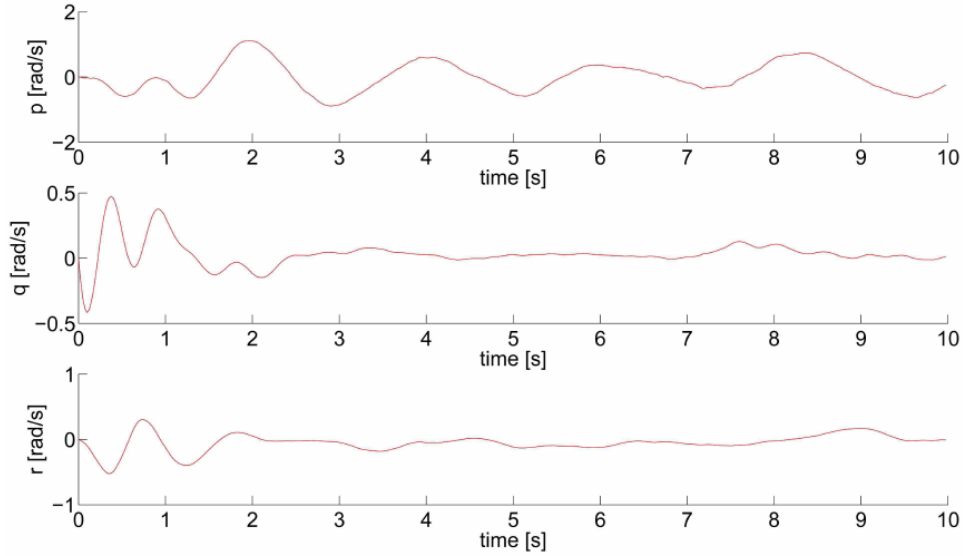
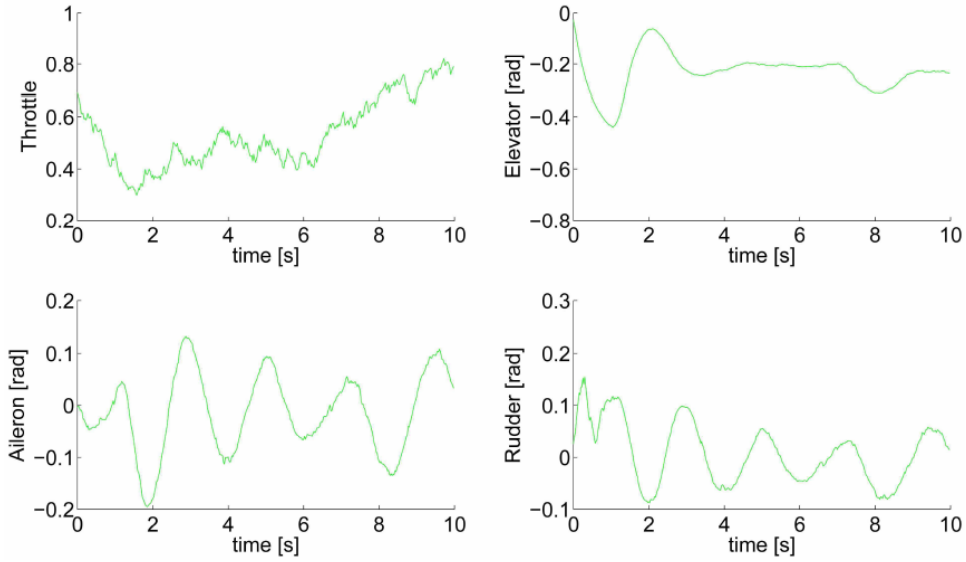


Figure 7 History of the optimal commands (see online version for colours)



Figures 5 and 6 collect time-histories for each B-axis of speeds and angular rates. Tuning cost-function gains, constraints on these state variables are imposed. Relative wind speed is compared with cruise speed introduced into the KA* model. The control system tries to keep this speed constant and equal to the reference. Coordinated turns are then imposed keeping to zero the lateral speed (*Y*-axis component) and in turn the sideslip angle. Angular rates on the other hand are bounded in order to avoid aggressive manoeuvres. However, strong turns are imposed during the first five seconds of simulation and this is due to the errors on the trim-conditions. As a matter of fact at the very beginning of the simulation the aircraft tries to track the path recovering the altitude loss. To do this, it has to perform a steep turn on the *X*-axis.

Finally, the command time-histories are shown in Figure 7. Aileron deflections are bounded and linked to the rudder one providing coordinated turns. Throttle on the other hand is decreased to lose altitude and follow the descent. Then it is kept almost constant and increased to climb in the last 4 seconds. The elevator is quickly deflected up to the limit in order to compensate the altitude tracking error.

6.2 Collision avoidance task

To test the collision avoidance task a static obstacle is introduced on the reference path (yellow marked in Figure 8). It is detected by the sensor when the simulation starts being closer than two kilometres. This is the maximum distance assumed for detection and no tracking loss is considered. The algorithm is able to follow the feature evolution and drive the aircraft to avoid the obstacle for the whole simulation without disturbances. The desired feature-attitude is 90 degrees for σ and γ. Because the camera reference system is aligned with the B system the

Z-axis is directed downward. Then $\sigma = 0$ is reached when the feature is along this axis. In this case, the obstacle is on the path and the aircraft too. Then σ already is equal to 90 degrees and the navigation system has to maintain the current value working just on γ to reach the desired attitude. Figure 8 shows that the navigation system is able to avoid the obstacle performing a fast and effective recovery manoeuvre on the lateral-directional plane.

Once the algorithm has completed the recovery manoeuvre it starts to go back to the reference path as the last part of the simulation can show. Good performances are evidenced on lateral-directional control but more tuning and investigation is required to improve longitudinal behaviour. The feature error is plotted on Figure 9; while first graph shows constant decrease of the error on γ, condition kept on σ is not sufficient. The error is small during first 5 seconds but when it start to go up the system needs too much time to compensate. Further improvements are required to solve this issue. Aircraft speed in B frame is shown in Figure 10.

Coordinated turn is required for this task too then the component along Y-axis is kept very low for the whole simulation. Figure 11 shows angular-rate time histories. Again, strong angular rates are asked when the simulation begins to recover from trim errors. Some spikes are present particularly on the Y component. This is due to visual servoing control. As a matter of fact, control signals are provided on the basis of feature evolution in the image surface that in turn depends from aircraft angular rates. Then continuous corrections to align the image feature to the reference are directly reflected on these states.

Figure 12 shows the aircraft attitude components reflecting the manoeuvre already described. Looking at roll and yaw angles the wide turn to avoid the obstacle is depicted. Pitch angle on the other hand follows altitude variations and speed fluctuations.

Figure 8 Comparison between reference and real path in longitude-latitude plane (see online version for colours)

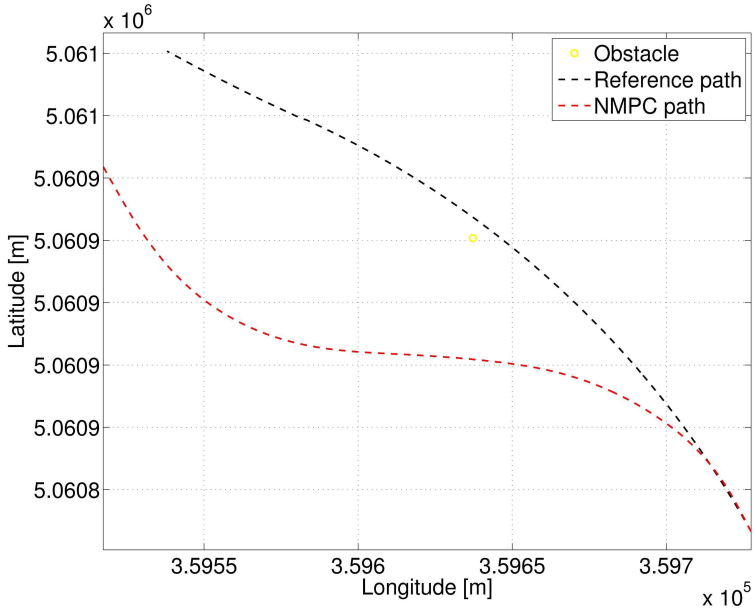


Figure 9 Error between the desired feature and the real one (see online version for colours)

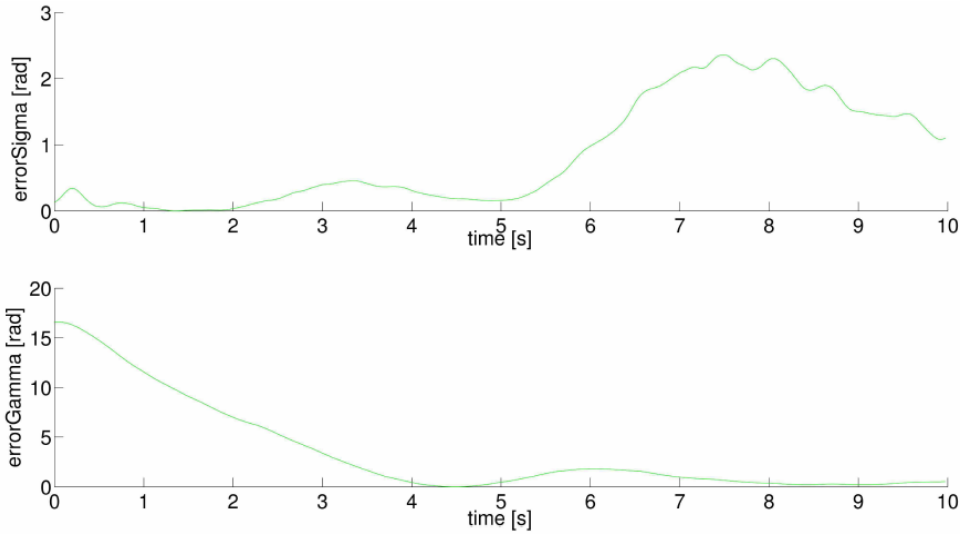


Figure 10 Comparison between predicted and real speed (see online version for colours)

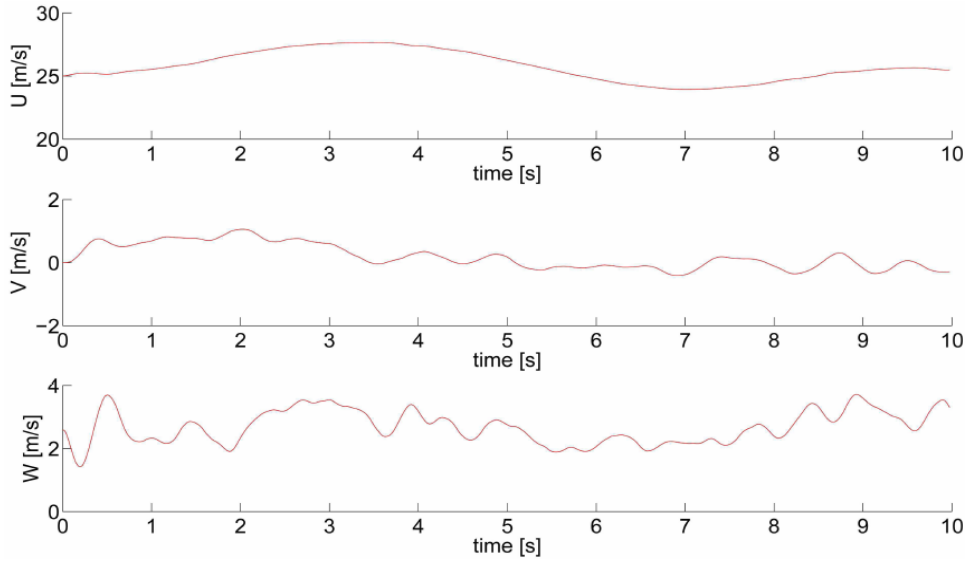


Figure 11 Comparison between predicted and real angular rate (see online version for colours)

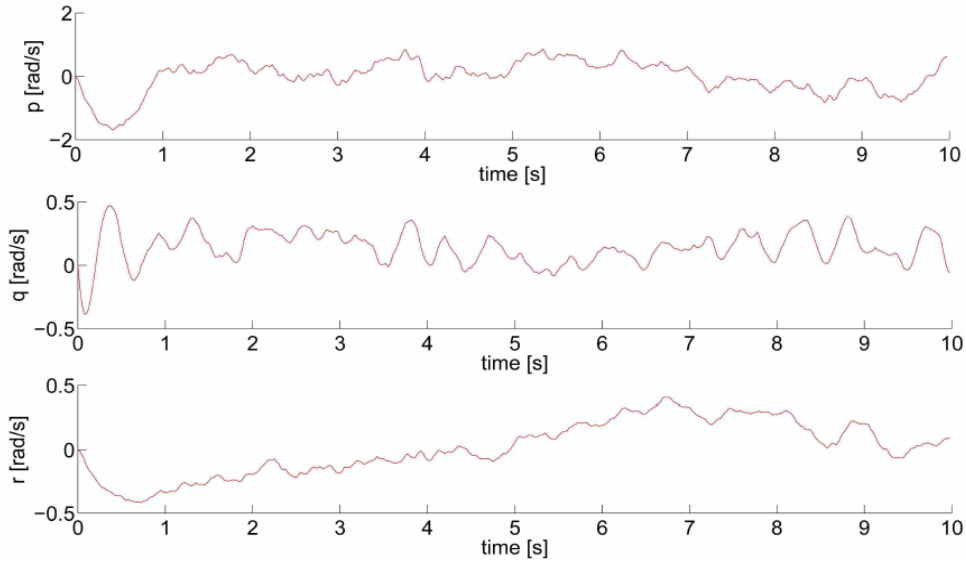


Figure 12 Comparison between predicted and real attitude (see online version for colours)

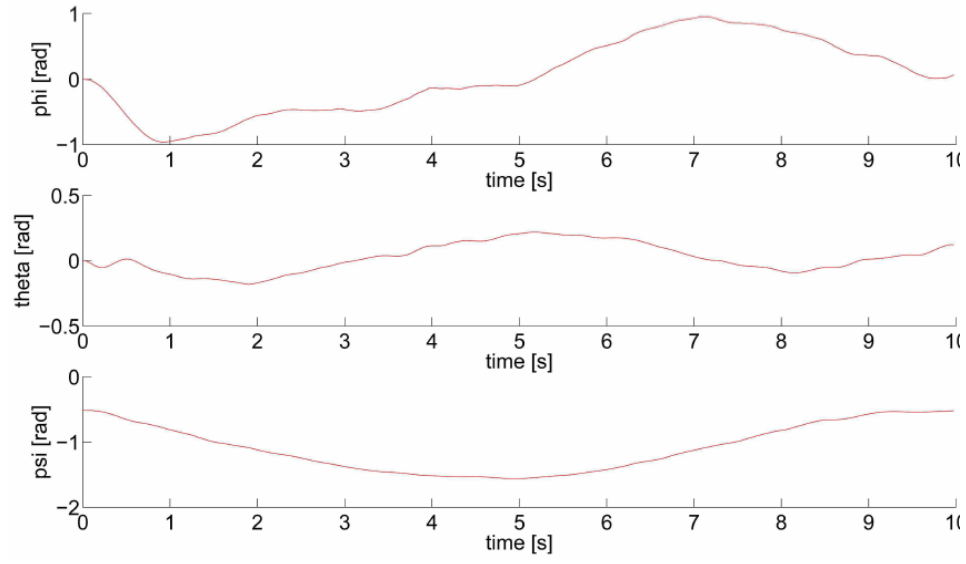
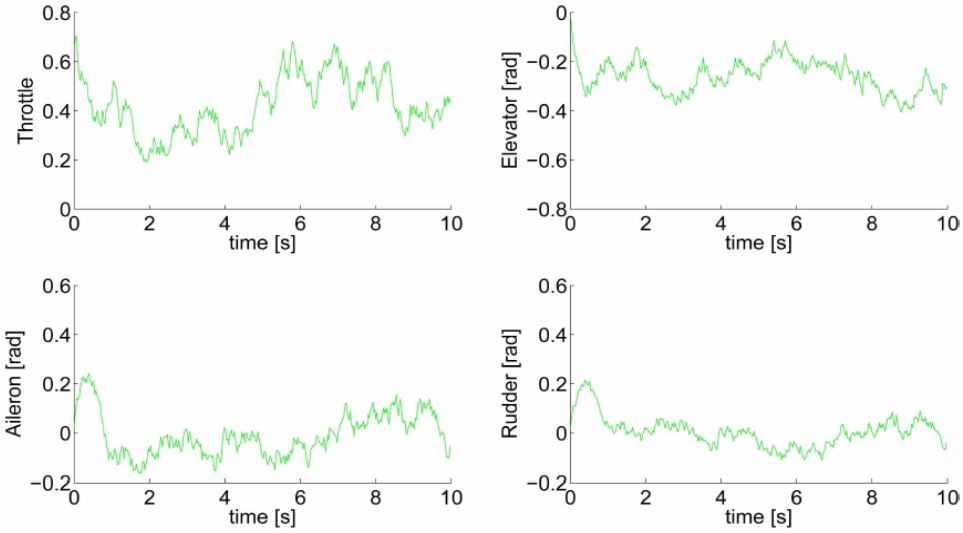


Figure 13 History of the optimal commands (see online version for colours)



Commands time-histories are reported in Figure 13. Lateral-directional commands reach high value when the simulation begins in order to start coordinated turn and avoid the obstacle. The elevator is kept to high angles trying to compensate altitude losses.

7 Conclusions

The tracking system proposed in this paper seems to reflect merits of NMPC and to accomplish with the task. As a matter of fact good tracking performance is evidenced with the results and effective control actions seem to provide smooth and safe paths. It must be stressed though that this is just the first implementation of this method and further improvements have been planned.

Particularly, the GA algorithm will be improved introducing modern genetic operators in order to optimise its convergence. On the other hand, present results are sufficient to motivate further investigations and to confirm that model prediction is a powerful and robust technique particularly useful in this field of research.

References

Allgöwer, F., Findeisen, R. and Nagy, Z.K. (2004) 'Nonlinear model predictive control: from theory to application', *J. Chin. Inst. Chem. Engrs*., Vol. 35, No. 3, pp.299–315.

Corke, P.I. (2010) 'Spherical image-based visual servo and structure estimation', *Proceedings of the IEEE International Conference on Robotics and Automation*, Anchorage, USA, 3–8 May, pp.5550–5555.

De Filippis, L. and Guglieri, G. (2012) 'Advanced graph search algorithms for path planning of flight vehicles', in Agarwal, R.K. (Ed.): *Recent Advances in Aircraft Technology*, pp.157–192, InTech – Open Access Publisher.

Deb, K. and Agarwal, R.B. (1995) 'Simulated binary crossover for continuous search space', *Complex Syst*., Vol. 9, No. 2, p.115–148.

Deb, K., Anand, A. and Joshi, D. (2002) 'A computationally efficient evolutionary algorithm for real-parameter evolution', *Evol. Comput*., Vol. 10, No. 4, pp.371–395.

Garcia, C.E., Prett, D.M. and Morari, M. (1989) 'Model predictive control: theory and practice', *Automatica*, Vol. 25, No. 3, pp.335–348.

Herrera, F., Lozano, M. and Verdegay, J.L. (1996) 'Dynamic and heuristic fuzzy connectives based crossover operators for controlling the diversity and convergence of realcoded genetic algorithms', *Int. J. Intell. Syst*., Vol. 11, No. 12, pp.1013–1040.

Holland, J.H. (1992) *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA, USA.

Kang, Y. and Hedrick, J.K. (2009) 'Linear tracking for a fixed-wing UAV using nonlinear model predictive control', *IEEE T. Contr. Syst. T*., Vol. 17, No. 5, pp.1202–1210.

Ono, I. and Kobayashi, S. (1997) 'A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover', *Proceedings of the 7th International Conference on Genetic Algorithms*, East Lansing, Michigan, USA, 19–23 July, pp.246–253.

Pehlivanoglu, Y.V., Baysal, O. and Hacioglu, A. (2007) 'Path planning for autonomous UAV via VGA', *Aircr. Eng. Aerosp. Tec*., Vol. 79, No. 4, pp.352–359.

Sprinkle, J., Eklund, J.M., Kim, H.J. and Sastry, S. (2004) 'Encoding aerial pursuit/evasion games with fixed wing aircraft into a nonlinear model predictive tracking controller', *Proceedings of the 43rd IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, 14–17 December, Vol. 3, pp.2609–2614.

Tian, J., Zheng, Y., Zhu, H. and Shen, L. (2005) 'A MPC and genetic algorithm based approach for multiple UAVs cooperative search', *Proceedings of the 2005 International Conference on Computational Intelligence and Security*, Vol. Part I, Springer-Verlag, Berlin, Heidelberg, pp.399–404.