AN INNOVATIVE MISSION MANAGEMENT SYSTEM FOR FIXED-WING UAVS

L. De Filippis*, G. Guglieri*, F. Quagliotti*
***Politecnico di Torino**
luca.defilippis@polito.it; giorgio.guglieri@polito.it; fulvia.quagliotti@polito.it

Keywords: *Path Planning, Collision Avoidance, Kinematic A*, Model Predictive Control, Genetic Algorithms*

Abstract

This paper presents two innovative units linked together to build the main frame of a UAV Mission Management System. The first unit is a Path Planner for small UAVs able to generate optimal paths in a tridimensional environment, generating flyable and safe paths with the lowest computational effort. The second unit is the Flight Management System based on Nonlinear Model Predictive Control, that tracks the reference path and exploits a spherical camera model to avoid unpredicted obstacles along the path. The control system solves on-line (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 minimizes the tracking error with respect to the reference path, driving the aircraft far from sensed obstacles and towards the desired trajectory.

1 Introduction

Mission Management Systems (MMSs) of unmanned aircraft run the vehicle in order to accomplish with the mission. Platform and mission characteristics, payload and reference scenario drive the MMS complexity which normally encloses ground and in-flight segments, made up of different subsystems interacting together. In the present discussion the main onboard-MMS units of an unmanned fixed-wing vehicle are identified in the Path Planner (PP), the Navigation System (NS), the Guidance System (GS), the Flight Control System (FCS) and the Sense&Avoid (S&A) unit. GS and FCS are linked to compose the Flight Management System (FMS) as shown in Figure 1. This architecture strongly simplifies the real MMS complexity, neglecting many other subsytems such as the Health Monitoring and Failure Management units, the Ground Segment and the Payload Management System that are basical components of any MMS for UAVs.

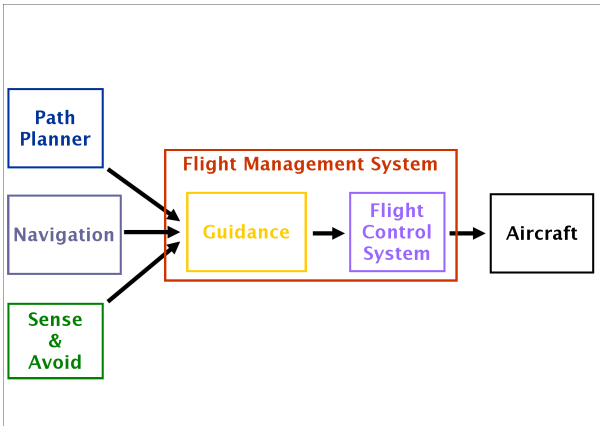

Fig. 1 Basic Mission Management System architecture

The best path to perform the mission is planned with the PP and it is uploaded on the Flight Management System (FMS) providing the references for the aircraft navigation. The GS merges the optimum-waypoint sequence with information coming from the NS and the S&A unit in order to generate the right guidance for the FCS. Tracking the path and avoiding unpredicted

obstacles in flight is the way to link kinematic references related to the desired aircraft positions with its dynamic behaviors, to generate the right command sequence. Different PP and GS have been developed, that exploit wide range of techniques providing encouraging results. However UAV dynamics is nonlinear and control systems able to optimize aircraft performance are desirable in order to plan complex trajectories and in turn face challenging mission tasks.

Path planning is the capability to generate a real-time trajectory to the target, avoiding known obstacles or collisions (assuming reference flight-conditions and providing maps of the environment) and optimizing a given functional under kinematic and/or dynamic constraints. Several solutions were developed matching different planning requirements: performances optimization, collision avoidance, real-time planning or risk minimization, etc. Graph search algorithms were developed for computer science to find the shortest path between two nodes of connected graphs. The A* algorithm was developed between 50s and 70s, explicitly oriented to motion-robotics [8]. It improved the graph search logic of the previous methods adding a heuristic component to the cost function. However to avoid obstacles on the map fixed position of obstacles was assumed. This is a logic assumption for many planning problems, but represents a limit when robots move in unknown environments. This problem excited research on algorithms able to face map modifications during path execution. Dynamic re-planning with graph search algorithms was introduced. D* (Dynamic A*) was published in 1993 [18] and it represents the evolution of A* for re-planning. D* focused was the evolution of D*, published by the same authors and developed to improve its characteristics [19]. Then, research on dynamic re-planning brought to development of Lifelong Planning A* (LPA*) and D* Lite [12], [13]. Onother important drawback of A* and the entire dynamic algorithms resides on heading constraints connected with graph structure. Different approaches were developed to cope with this problem, based on post-processing algorithms or on improvements of the graph-search logic. Very important examples are Field D* [6] and Theta*[14].

In the last decades wide research has been done on Receding Horizon Control (RHC) techniques [7] to cope with: a) intrinsically nonlinear dynamic systems, b) high quality requirements, c) growing use of robotic systems in any working division. Sprinkle et al. [17] presented a NMPC system applied to trajectory tracking for persuit/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 et al. [11] implemented an interesting NMPC system to cope with trajectory tracking problems. They designed a high-level tracking controller for a small fixed-wing UAV and they studied close-loop stability extracting some performance properties of the control strategy adding an outer loop to the inner control loop. Whether theoretical and mathematical support is fundamental to convert a simple problem solving approach in a deeper investigation able to provide general concepts on this control technique, on the other hand this approach to the problem requires simplifications that could mismatch with a real implementation problem.

This paper describes a PP unit, a S&A unit and a FMS working together to steer the aircraft over the mission path. In more details the proposed PP architecture exploits a novel graph search algorithm (named Kinematic A*), developed to face an important drawback of classic graph search algorithms applied to path planning. As a matter of fact any aircraft kinematics is included into the problem formulation in order to constraint path search on classic graph search methods. Kinematic A* implements the graph search logics to generate feasible paths, introducing basic kinematic constraints to drive the search. On the other hand the proposed FMS exploits Nonlinear Model Predictive Control (NMPC) together with a spherical camera

model to accomplish with collision avoidance and trajectory tracking tasks. The FMS solves in real time an optimal control problem with two concurring objectives: tracking the optimal path provided with KA* and avoid unpredicted obstacles detected with the S&A sytem that exploits a spherical camera and visual servoing techniques. Further simplifications are needed to test these subsystems. The NS is assumed unaffected by noise and sensor errors. A simple aircraft plant has been implemented, particularly the FMS has been linked to the Aerosonde UAV Simulink model to perform the tests.

2 Reference path

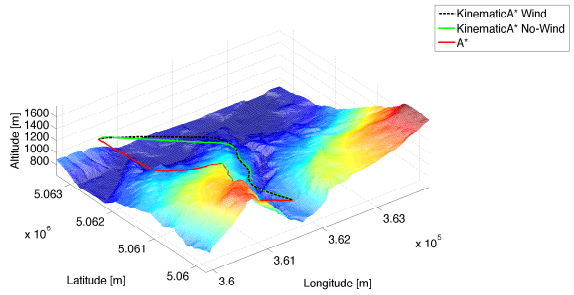

Fig. 2 Valley way out test (3D view)

The higher-level PP system that provides the path the aircraft must follow exploits Kinetamic A* algorithm [5]. 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 2). Kinematic A* includes a simple kinematic model of the vehicle to evaluate the moving cost between waypoints in a tridimensional environment. Movements are constrained with minimum turn radius and maximum rate of climb.

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

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

The model is a set of four differential equations describing the aircraft motion in Ground reference frame (G frame). (X,Y,Z) is the position vector, the two angles (χ and γ) are respectively the angle between the X-axis and the projection of the speed vector (V, constant in magnitude) on the $X-Y$ plane and the angle between the speed vector and its projection on the $X-Z$ plane. Command variable w $(-1 \leq w \leq 1)$ modules the climb angle between its minimum and maximum values coincident with γ_{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.

KinematicA* being derived from classic graph search algorithms exploits the same search logics. The best path is obtained iteratively, evaluating a cost function J from the starting state up to reach the target one. Each iteration minimizes functional F_{ij} of each state composing the path and the global functional J is obtained summing up minimum F_{ij}s at each iteration. Choosing the smaller value of F_{ij} the algorithm selects a new state that reduces distance from the target minimizing the commands. Then the optimum path minimizes F_{ij} at each step and global optimality of J is not evaluated. This is the typical "greedy" approach that characterizes graph search algorithms. F_{ij} is made of two terms related respectively with states and commands. At each step (j) the algorithm generates the set of movements (i) from the current state. Then it evaluates F_{ij} for each new state and chooses the one with the smaller value, so that J is defined as:

$$J = \sum_{j=S_0}^{j=S_t} \min(Fij) = \sum_{j=S_0}^{j=S_t} \min\left(\bar{H}_{ij}^T \cdot \alpha \cdot \bar{H}_{ij} + \bar{G}_{ij}^T \cdot \beta \cdot \bar{G}_{ij}\right) \quad (2)$$

H represents cost to go then it is the distance between the new state (X_i,Y_i,Z_i) and the target (X_t,Y_t,Z_t). On the other hand G substitutes cost to come of classic graph search algorithms with the amount of command needed to reach the new state. Gain matrices α and β are diagonal matrices, used to weight state variables and commands

tuning their influence on F:

$$\bar{H}_i = \begin{bmatrix} X_t - X_i \\ Y_t - Y_i \\ Z_t - Z_i \end{bmatrix} = \begin{bmatrix} \Delta X_i \\ \Delta Y_i \\ \Delta Z_i \end{bmatrix} \qquad \bar{G}_i = \begin{bmatrix} u_i \\ w_i \end{bmatrix} \tag{3}$$

$$\bar{\alpha} = \begin{bmatrix} \alpha_1 & 0 & 0 \\ 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_3 \end{bmatrix} \qquad \bar{\beta} = \begin{bmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{bmatrix} \tag{4}$$

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. When an unpredicted obstacle is detected, the GS steers the aircraft to avoid collision (as it will be described in the following sections) going back to the reference path when the obstacle is avoided.

3 Sense&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.

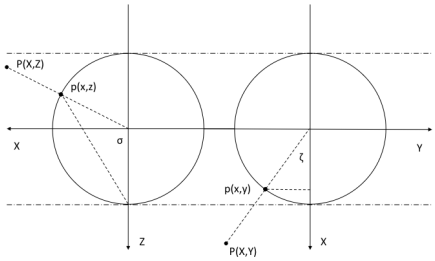

Fig. 3 Feature representation in the image surface

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 maneuver the aircraft has to sense the intruder when it is some kilometers 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 maneuver.

Figure (3) 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 [2]:

$$\begin{Bmatrix} \dot{\sigma} \\ \dot{\zeta} \end{Bmatrix} = [J_{V_c}] \cdot \begin{Bmatrix} V_x^c \\ V_y^c \\ V_z^c \end{Bmatrix} + [J_{\omega_c}] \cdot \begin{Bmatrix} \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{Bmatrix} \quad (5)$$

with:

$$\bar{V}_c = \begin{Bmatrix} V_x^c \\ V_y^c \\ V_z^c \end{Bmatrix} \quad (6)$$

$$\bar{\omega}_c = \begin{Bmatrix} \omega_x^c \\ \omega_y^c \\ \omega_z^c \end{Bmatrix} \quad (7)$$

$$[J_{V_c}] = \begin{bmatrix} -\frac{\cos\sigma \cdot \cos\zeta}{R} & -\frac{\cos\sigma \cdot \sin\zeta}{R} & \frac{\sin\sigma}{R} \\ \frac{\sin\zeta}{R \cdot \sin\sigma} & -\frac{\cos\zeta}{R \cdot \sin\sigma} & 0 \end{bmatrix} \quad (8)$$

$$[J_{\omega_c}] = \begin{bmatrix} \sin\zeta & -\cos\zeta & 0 \\ \frac{\cos\sigma \cdot \cos\zeta}{\sin\sigma} & \frac{\cos\sigma \cdot \sin\zeta}{\sin\sigma} & -1 \end{bmatrix} \quad (9)$$

where $\bar{V}_c$ is the camera linear speed vector, $\bar{\omega}_c$ is the camera angular speed vector and R is range. Equation (5) 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 behavior on image surface according with aircraft dynamic evolution. In other words the sensor model is exploited to predict future aircraft behaviors such that predicted feature would reach a prescribed position on image surface.

A reference feature position is needed to trigger the avoidance manouver, minimizing 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 maneuver. Figure (4) shows a recovery maneuver where an obstacle is detected on the right side of the aircraft and slightly above it. The reference feature-vector assigned to perform the maneuver is $f(90,90)$. Then the aircraft is forced to increase the altitude up to drive σ to ninety degrees and turn on the left so that γ reaches the same value.

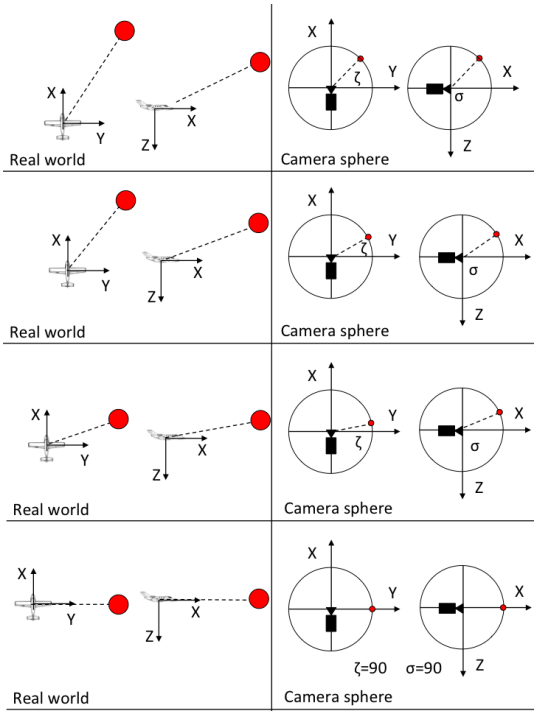

Fig. 4 Recovery maneuver

4 NMPC formulation

NMPC acts solving a finite horizon open-loop optimal control problem in real-time. The cost function is the function minimized with the optimal commands. This function characterizes the problem containing variables that represent the optimization task. A classical quadratic function has been selected here made of two terms:

- **State error**: this term evaluates the error between the reference states and the predicted one and it is the term where predicted positions are compared with the de-

sired one provided by KA*.

- **Command**: this term evaluates the amount of command needed to perform the predicted maneuver.

The cost function depends from 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:

$$\min_{\bar{U}^*(\cdot)} J(\bar{X}_0, \bar{U}^*(\cdot)) = \int_t^{t+T_p} \left(\hat{\bar{X}}^*(\tau)^T \cdot Q \cdot \hat{\bar{X}}^*(\tau) + \bar{U}^*(\tau)^T \cdot R \cdot \bar{U}^*(\tau) d\tau \right) \quad (10)$$

with:

$$\hat{\bar{X}}^*(\tau) = \bar{X}^*(\tau) - \bar{X}_{ref}(\tau) \quad (11)$$

where $\bar{X}_0 = \bar{X}(t) \in \Re^n$ is the initial-state vector and $\bar{U}^*(\cdot) \in \Re^m$ is the predicted-command vector. Then $\bar{X}^*_\tau \in \Re^n$ is the predicted-state vector and $\bar{X}_{ref}(\tau) \in \Re^n$ is the reference-state vector. Finally Q and 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 optimization loop and it commonly differs from T_p. The command strategy is then the other fundamental element to formulate NMPC. The classic command vector of an aircraft contains:

$$\bar{U} = \begin{cases} \delta_e, & \delta_{e_{min}} \leq \delta_e \leq \delta_{e_{max}} \\ \delta_a, & \delta_{a_{min}} \leq \delta_a \leq \delta_{a_{max}} \\ \delta_r, & \delta_{r_{min}} \leq \delta_r \leq \delta_{r_{max}} \\ Th, & 0 \leq Th \leq 1 \end{cases} \quad (12)$$

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 commands variation has been chosen here over the command horizon. Particularly a peacewise linear function is built over the prediction horizon based on functions:

$$\bar{U}^*_i(\tau) = \bar{U}^*_{i_0} + \bar{A}_i * \tau \quad 1 \leq i \leq n_c \quad (13)$$

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

$$\begin{array}{ll} \bar{U}^*_1(\tau) = \bar{U}_0 + \bar{A}_1 * \tau & t-1 \leq \tau \leq T_c/2 \\ \bar{U}^*_2(\tau) = \bar{U}_{T_c/2} + \bar{A}_2 * \tau & T_c/2 \leq \tau \leq T_c \end{array} \quad (14)$$

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

$$\bar{U}^*(\tau) = \bar{U}^*_{n_c}(T_c) \quad T_c \leq \tau \leq T_p - T_c \quad (15)$$

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 (13) are bounded with disequalities in (12) but $\bar{A}_i$ vector must be bounded too. Maximum command variation over unitary time-step is chosen such that:

$$\Delta\bar{U}^*_{min} \leq \bar{A}_i \leq \Delta\bar{U}^*_{max} \quad (16)$$

$$\Delta\bar{U} = \begin{cases} \Delta\delta_{e_{min}} \leq \Delta\delta_e \leq \Delta\delta_{e_{max}} \\ \Delta\delta_{a_{min}} \leq \Delta\delta_a \leq \Delta\delta_{r_{max}} \\ \Delta\delta_{r_{min}} \leq \Delta\delta_r \leq \Delta\delta_{r_{max}} \\ \Delta Th_{min} \leq \Delta Th \leq \Delta Th_{max} \end{cases} \quad (17)$$

Solving online the optimization problem the linear-function slopes in (13) are chosen in order to minimize the cost function.

Defining the state vector as:

$$\bar{X} = \{u\ v\ w\ p\ q\ r\ \phi\ \theta\ \psi\ x_V\ y_V\ z_V\ \omega_p\} \quad (18)$$

the complete optimization problem prescribes to find:

$$\bar{U}^*_{opt} = f(\tau, \bar{U}_0, \bar{A}_{opt}, T_c, T_p, n_c) \quad (19)$$

so that equation (10) is satisfied according with system of equations (12) and (17) but also with the aircraft nonlinear equations of motion:

$$\dot{\bar{X}} = f(\bar{X}, \bar{U}) \quad (20)$$

5 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.

Genetic algorithm 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.

5.1 Tracking task

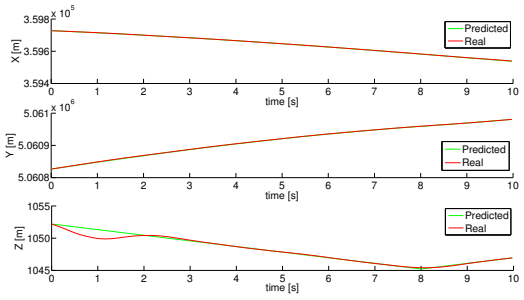

Fig. 5 Comparison between reference and real path on each Ground axis

Figure 5 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 meters and it is quickly reduced by the control system.

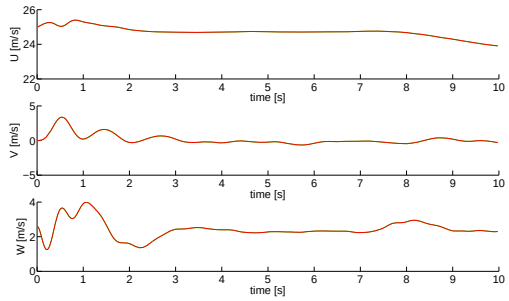

Fig. 6 Comparison between predicted and real speed

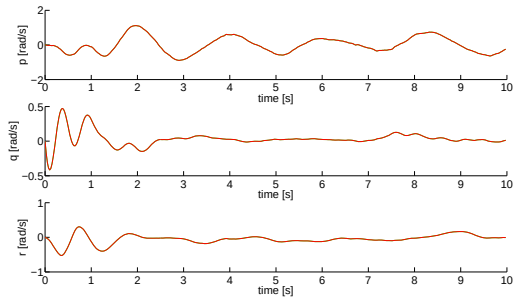

Fig. 7 Comparison between predicted and real angular rate

Figures 6 and 7 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 maneuvers. 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 tryes 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 8. 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 de-

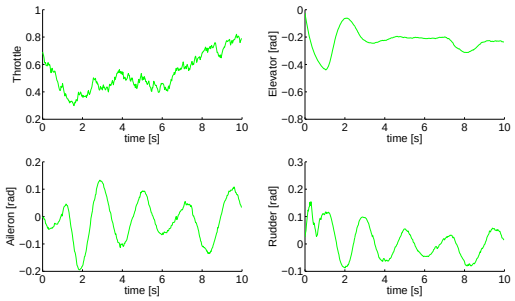

Fig. 8 History of the optimal commands

scent. 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.

5.2 Collision avoidance task

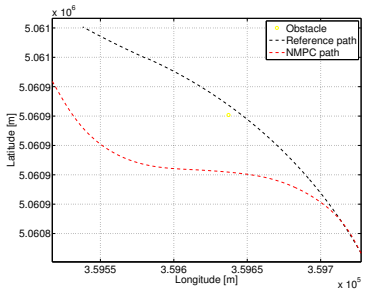

Fig. 9 Comparison between reference and real path in Longitude-Latitude plane

To test the collision avoidance task a static obstacle is introduced on the reference path (yellow marked in Figure 9). It is detected by the sensor when the simulation starts being closer then two kilometers. 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 9 shows that the navigation system is able to avoid the obstacle performing a fast and effective recovery maneuver on the lateral-directional plane.

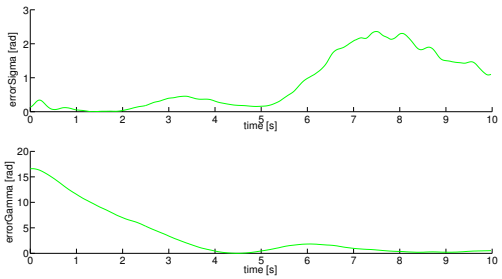

Fig. 10 Error between the desired feature and the real one

Once the algorithm has completed the recovery maneuver 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 behavior. The feature error is plotted on Figure 10; 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 Figures (11).

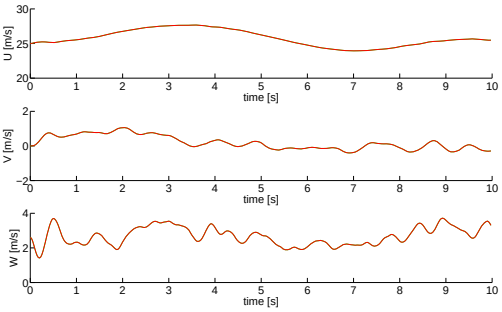

Fig. 11 Comparison between predicted and real speed

Coordinated turn is required for this task too

then the component along Y-axis is kept very low for the whole simulation. Figure (12) 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.

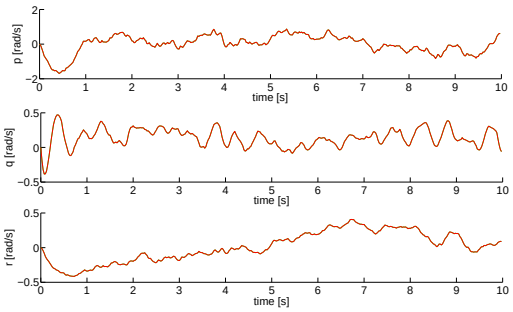

Fig. 12 Comparison between predicted and real angular rate

Figure (13) shows the aircraft attitude components reflecting the maneuver 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.

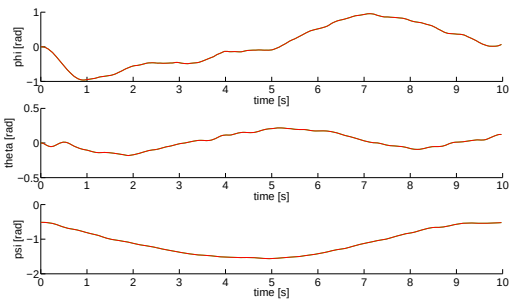

Fig. 13 Comparison between predicted and real attitude

Commands time-histories are reported in Figure (14). 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.

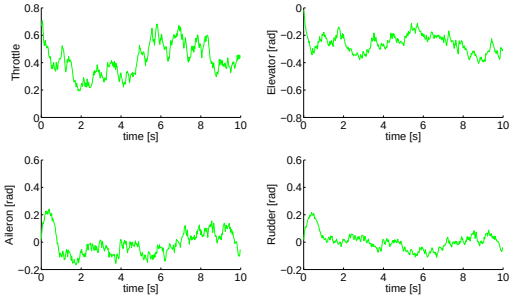

Fig. 14 History of the optimal commands

6 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 optimize 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 these field of research.

References

[1] F. AllgŽwer, R. Findeisen and Z.K. Nagy. Nonlinear Model Predictive Control : From Theory to Application. *J Chin Inst Chem Engrs*, 3: 299-315, 2004.

[2] P.I. Corke. Spherical image-based visual servo and structure estimation. In proceedings of 2010 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Anchorage, Alaska, pp. 5550-5555, (2010).

[3] K. Deb and R.B. Agarwal. Simulated binary crossover for continuous search space. *Complex Syst.*, 9:115-148, 1995.

[4] K. Deb, A. Anand and D. Joshi. A computationally efficient evolutionary algorithm for real-parameter evolution. *Evolutionary Computation Journal*, 10 (4), 371-395, 2002.

[5] L. De Filippis and G. Guglieri. Advanced Graph Search Algorithms for Path Planning of Flight Vehicles. *Recent Advances in Aircraft Technology*, InTech - Open Access Publisher, 1:1-36, 2012.

[6] D. Ferguson and A. Stentz. Using interpolation to improve path planning: The Field D* algorithm. Journal of Field Robotics, No. 23, Vol. 2, pp. 79-101, (2006).

[7] C.E. Garcia, D.M. Prett and M. Morari. Model predictive control: theory and practice. *Automatica*, 25 (3):335-348,1989.

[8] P. Hart, N. Nilsson and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics, SCC-4(2), pp. 100-107, (1968).

[9] F. Herrera, M. Lozano and J.L. Verdegay. Dynamic and heuristic fuzzy connectives based crossover operators for controlling the diversity and convergence of real-coded genetic algorithms. *International Journal of Intelligent System*, 2, 1013-1041, 1999.

[10] J.H Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[11] Y. Kang and J.K Hedrick. Linear Tracking for a Fixed-Wing UAV Using Nonlinear Model Predictive Control. *Control Systems Technology, IEEE Transactions on*, 17(5):1202-1210, 2009.

[12] S. Koenig and M. Likhachev. Incremental A*. Proceedings of the Natural Information Processing Systems, (2001).

[13] S. Koenig and M. Likhachev. D* Lite. Proceedings of the AAAI Conference on Artificial Intelligence, pp. 476-483, (2002).

[14] A. Nash, K. Daniel, S. Koenig and A. Felner. Theta*: Any-angle path planning on grids. Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1177-1183, (2007).

[15] I. Ono and S. Kobayashi. A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover. *in T. Back (Ed.), Proceedings of the Seventh International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, pp. 246-253, 1997.

[16] Y.V. Pehlivanoglu, O. Baysal and A. Hacioglu. Path planning for autonomous UAV via VGA. *Aircraft Engineering and Aerospace Technology: An International Journal*, 79 (4), 352-359, 2007.

[17] J. Sprinkle, J.M. Eklund, H.J. Kim and S. Sastry. Encoding aerial pursuit/evasion games with fixed wing aircraft into a nonlinear model predictive tracking controller. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, Vol. 3, pages 2609-2614, Atlantis, Paradise Island, Bahamas, 2004.

[18] A. Stentz. Optimal and efficient path planning for unknown and dynamic environments. Carnegie Mellon Robotics Institute Technical Report, CMU-RI-TR-93-20, (1993).

[19] A. Stentz. The focussed D* algorithm for real-time replanning. Proceedings of the International Joint Conference on Artificial Intelligence, pp.1652-1659, (1995).

[20] J. Tian, Y. Zheng, H. Zhu and L. Shen. A MPC and Genetic Algorithm Based Approach for Multiple UAVs Cooperative Search", In *Proc. CIS (1) '05*, pages 399-404, Xi'an, China, 2005.

6.1 Copyright Statement