Guidance and control algorithms for mini UAV autopilots

Elisa Capello and Giorgio Guglieri
Dipartimento di Ingegneria Meccanica ed Aerospaziale, Politecnico di Torino, Torino, Italy, and
Gianluca Ristorto
Faculty of Science and Technology, University of Bozen-Bolzano (FUB), Bozen, Italy

Abstract
Purpose – The aim of this paper is the implementation and validation of control and guidance algorithms for unmanned aerial vehicle (UAV) autopilots.
Design/methodology/approach – The path-following control of the UAV can be separated into different layers: inner loop for pitch and roll attitude control, outer loop on heading, altitude and airspeed control for the waypoints tracking and waypoint navigation. Two control laws are defined: one based on proportional integrative derivative (PID) controllers both for inner and outer loops and one based on the combination of PIDs and an adaptive controller.
Findings – Good results can be obtained in terms of trajectory tracking (based on waypoints) and of parameter variations. The adaptive control law guarantees smoothing responses and less oscillations and glitches on the control deflections.
Practical implications – The proposed controllers are easily implementable on-board and are computationally efficient.
Originality/value – The algorithm validation via hardware in the loop simulations can be used to reduce the platform set-up time and the risk of losing the prototype during the flight tests.

Keywords Autopilot algorithms for UAVs, Guidance algorithm, Robust and adaptive control laws

Paper type Research paper

Introduction

Autonomous small unmanned aerial vehicles (UAVs) have been increasingly used by military and civilian applications because of the recent advances in communications, battery technology and micro electro-mechanical systems (MEMS) electronic devices. Successful deployment of small UAVs requires powerful, intuitive and lightweight autopilots with increased autonomy level including path planning, trajectory generation and tracking algorithms.

According with the hardware features and limits, the software segment of an autopilot performs the functions of data analyzing, state estimator, path finder and controller and has to be able to guide the UAV in flight with no human assistance. Both commercial and research autopilots are designed to perform GPS-based waypoint navigation. The path-following control of the UAV can be separated into different layers:

- inner loop for pitch and roll attitude control;
- outer loop on heading, altitude and airspeed control for the waypoints tracking; and
- waypoint navigation.

The objective of this paper is focused on the development of guidance and control algorithms suitable for small UAV autopilots, which is able to perform the required mission (i.e. following waypoints) and to reconfigure the flight path in presence of disturbances, with a low computational effort. Two different controllers are analyzed:

1. one based on proportional integrative derivative (PID) controllers both for inner and outer loops (Capello *et al.*, 2012a, 2012b); and
2. one based on the combination of PIDs and an adaptive controller (Capello *et al.*, 2013a, 2013b).

For the second case, we have that the inner loop is synthesized using an L1 adaptive algorithm for pitch, roll and airspeed control, and the outer loop is based on PIDs to control the altitude and the heading angle.

The guidance segment assumes a relevant role for the accomplishment of the UAV mission, providing a feasible trajectory. Different guidance algorithms can be implemented depending on the mission the UAV has been designed for.

The MAGICC lab at Brigham Young University (Beard and Kingston, 2005) developed guidance and control algorithms divided into four distinct hierarchical layouts: path planning, trajectory smoothing and tracking and the autopilot. The trajectory smoother transforms the waypoint path into time-stamped feasible trajectory, and the trajectory tracker finds feasible control input to stabilize the system parameters.

The current issue and full text archive of this journal is available on Emerald Insight at: **www.emeraldinsight.com/1748-8842.htm**

Aircraft Engineering and Aerospace Technology: An International Journal
89/1 (2017) 133–144

[DOI 10.1108/AEAT-10-2014-0161]

Received 13 October 2014
Revised 3 August 2015
4 August 2015
Accepted 4 August 2015

Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

A limitation of this algorithm is that the trajectories are time-stamped curves which specify the desired inertial location of the UAV at a specified time.

A 3D guidance scheme, based on the sliding mode approach using nonlinear sliding manifolds, is proposed in Shah *et al.* (2014). The proposed 3D guidance scheme is implemented on a six-degrees-of-freedom (6-dof) simulation of an UAV, and simulation results are presented for different 3D trajectories with/without disturbances.

In Sujit *et al.* (2013), different path following algorithms that are easy to implement, take less implementation time and are robust to disturbances are considered and compared through simulations using the total cross-track error and control effort performance metrics. A limitation of this paper is the high computational time, which limits the application to MEMS systems.

A reconfigurable guidance algorithm is proposed by the application of methods based on the direction field theory, as in Degen *et al.* (2009) and Meenakshisundaram *et al.* (2010). The vector field approach is dependent on the aircraft perpendicular distance from the connection line of two waypoints, and lookup tables from flight data are proposed to take into account the presence of wind, as in Osborne and Rysdyk (2005). The input of the lookup table is the initial and desired course, the wind speed and direction and the aircraft airspeed. The output is a look-ahead distance, which is the distance to begin turning toward the next waypoint.

The application of these methods in real world applications is a big issue, especially when it is difficult to know the actual flight conditions from the collected flight data. In Kundak and Mettler (2007), an experimental framework is proposed for evaluating the performance of the guidance and control algorithms for a small unmanned helicopter. Therefore, the test bed construction requires large amount of time and effort to build custom experimental setup and to integrate sensors and processors.

For this reason, in Barros dos Santos *et al.* (2011a, 2011b) software in the loop simulations are performed to validate the guidance and control algorithms. The main drawback of this technique is that simulation time will be completely different than the one expected from a real-time system. Differently, hardware in the loop (HIL) simulation deals with reproducing the environment where the embedded system will run. This is usually one of the last steps in the testing procedure, probably before integration and system tests. With this motivation, we performed HIL simulations to validate the guidance and control algorithms evaluating drawbacks and performance in simulations. In the guidance part, a look-ahead distance is also considered for the nonlinear effects because of disturbances, and a *no correction* zone is proposed to avoid continuous corrections of the trajectory and glitches on the control surface variation.

The paper is organized as follows. In second section, the HIL and autopilot systems are described. The mathematical model of the small UAV is presented in third section. The control laws are described in fourth section, and the guidance algorithm is presented in fifth section. The validation of this algorithm is performed via HIL simulations in the next section. The conclusions are summarized in the last section.

Hardware in the loop and autopilot systems

Usually commercial autopilots are not reconfigurable; this means that variations of on board software are not allowed. For this reason, a custom-made autopilot, designed by the researchers of Politecnico di Torino (Capello *et al.*, 2012a, 2012b) (Figure 1) is considered in this paper. Its main characteristics comprehend an open architecture, the possibility to be reprogrammed in flight and real-time telemetry. Sensors include GPS, barometric sensor, differential pressure sensor, three-axis gyros and accelerometers and a three-axis-magnetometer. The CPU is the AtMega 1280 model with 128 Kb flash memory and 8 Kb of RAM with a CPU clock of 16 MHz. The data link transmission is accomplished with the Xbee Pro board at 2.4 GHz.

The software part consists of a multi-skilled algorithm in C language designed for the management of:

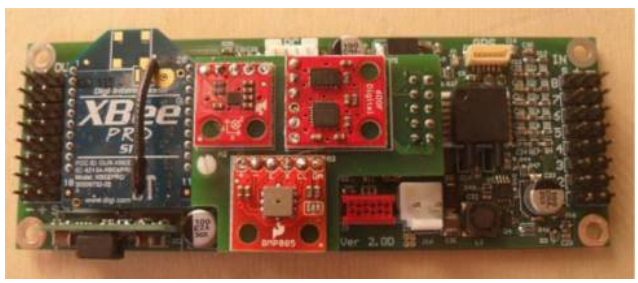
- flight data acquisition;
- navigation algorithm, based on a Kalman filter for sensor acquisition and data fusion;
- guidance algorithm, implemented to guide the UAV through waypoints with a feasible trajectory; and
- control algorithms for the inner and outer loop variables.

To validate the controller and the guidance algorithm with HIL simulations, a commercial board (the XMOS XK-1A board) with USB connection has been selected because of its characteristics and potentialities (i.e. flash memory of 128 Kb and a CPU clock of 20 MHz) similar to our microcontroller. This board must be physically connected to the laptop to effectively be part of the loop.

The XMOS XK-1A is a low-cost development board produced by XMOS Ltd (www.xmos.com), and it is characterized by the multi-core multi-thread processor XS1-L1 which is able to perform several real-time tasks. Its parallel computing ability is essential for unmanned applications where high level tasks (for instance the control logic) have to be combined with low level assignments (such as I/O) (Martins *et al.*, 2012). Moreover, the XTAG-2 debug adapter can be connected to debug the XK-1A board with a PC. The board can be powered from the USB connection using the debug adapter or by an external 5 V power supply.

One of the main advantages in using XMOS technology is the facility in programming the board. The language for the XMOS board is called XC. This language, even if not too different from C, shows some additional commands for the management of the ports and the pins.

Figure 1 Custom-made autopilot board



Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

The HIL technique is usually used in the development and test of complex real-time embedded systems such as the autopilot system for a small UAV.

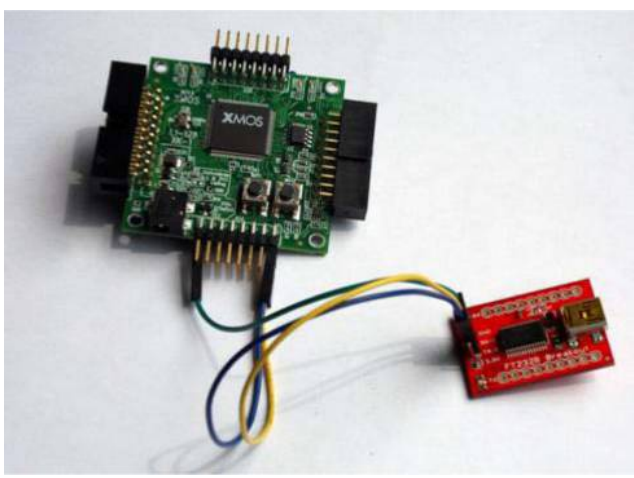
Validation of a novel asymptotic tracking controller for underactuated unmanned quadrotor is presented in Zhao *et al.* (2015). In Skulstad and Syversen (2014), the performance of a guidance controller to achieve a very precise final approach to a landing target was prior proved via HIL environment. Different control strategies of autonomous cooperative systems are addressed in Khaleghy and Song (2014) to investigate the performance of a complex system, based on both aerial and ground vehicles, in monitoring the border and crowd surveillance. HIL technique is also used to test UAV hardware equipment. In Gong *et al.* (2014), the effect of load fluctuations on the performance of the fuel-cell system of an UAV is investigated through HIL simulations using real-world flight data. Whereas a battery test bed has been developed in (Nyangweso and Bole, 2014) in order to develop and validate prognostic algorithms for predicting battery discharge time and battery failure time. Furthermore, rigorous experiments with HIL simulator reduce the risk of damaging the equipment during further flight tests (Barros dos Santos *et al.*, 2011a, 2011b).

The purpose of HIL simulations is to provide an effective platform for developing and testing the embedded system, which is modeled with a mathematical representation.

The main difference of HIL system from a software-in-the-loop simulation is the presence of a real platform in the loop. The board is placed between the guidance algorithm and the aircraft models, and it accomplishes the role of controller; this means that it receives the flight data and calculates the necessary commands for the UAV control surfaces.

Focusing on the HIL simulation, the inner and outer loop controllers are partially converted in XC language and implemented in the XMOS platform. Then the board must be physically connected to the laptop to effectively be part of the loop. To accomplish this connection, a breakout board for USB to serial conversion is placed between the XMOS board and the laptop. A detail of the HIL cables connection is represented in Figure 2.

Figure 2 Cable connection of the HIL board



MH850 small unmanned aerial vehicle

The aircraft considered for the validation of the proposed algorithms is the MH850 mini-UAV (Marguerettaz *et al.*, 2010; Capello *et al.*, 2013a, 2013b) developed by Politecnico di Torino and by the spin-off company MAVTech. The MH850 is characterized by tailless configuration, electric propulsion, tractor propeller and non-movable vertical fins at wingtips (Figure 3). The wingspan is 85 cm, the mass is about 1 kg, it is able to fly for 45 min at a cruise speed of 13.5 m/s. Aircraft control is achieved with trailing edge elevon (symmetric deflection for elevator δ_e and antisymmetric for aileron δ_a). A numerically derived database comprehensive of all aerodynamic derivatives is used to build the nonlinear aircraft model.

The MH850 aircraft is able to perform autonomous flight thanks to the on-board installation of an autopilot.

A set of 12 equations (Etkin and Reid, 1996) describing the forces, moments, angles and angular speeds which characterize the flight condition of the aircraft are considered for the nonlinear model. The trim conditions are $V_0 = 13.5$ m/s, $\vartheta_0 = 6$ deg, $h_0 = 100$ m, $\phi_0 = 0$ deg and $\psi_0 = 0$ deg.

To build the aircraft mathematical models previously described, it is necessary to estimate the aerodynamic and control derivatives of the vehicle. This is performed thanks to a software tool called aircraft configuration interface (ACI) (Marguerettaz *et al.*, 2010; Capello *et al.*, 2009). ACI is a Java-based interface that performs the aerodynamic analysis of lifting surfaces according to the extended lifting line theory. Fuselage aerodynamics is achieved by superposition of potential flow, friction drag and cross-flow effects. The overall aerodynamic configuration is obtained by adding all the separate contributions. The aerodynamic coupling of the fuselage and empennages with the wing is also considered. Aircraft control derivatives are computed with the lifting line theory; stability and damping derivatives are also evaluated. Flight tests have validated the estimated performance with a good degree of accuracy.

Control law architecture

Two different controllers are implemented:

1. one based on PIDs for both inner and outer loop variables; and
2. one based on the combination of PIDs and an L1 adaptive controller.

Figure 3 The MH850 small UAV

Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

The first suggested method follows a hybrid approach based on $H\infty$ loop shaping theory, where nominal stability is guaranteed using root *locus* method (Capello *et al.*, 2012a, 2012b), as also proposed by Ntogramatzidis and Ferrante, 2011. As detailed in Capello *et al.* (2012a, 2012b), the root *locus* technique is applied to identify the desired value of PID derivative gain and zeros (used to define the proportional and integrative gains). Furthermore, the PID gains are defined to obtain, on one hand, adequate robust stability and performance and, on the other hand, to optimize closed-loop performance in terms of step response and waypoints tracking.

In the longitudinal plane, the airspeed V is controlled by the elevator deflection δ_e, whereas altitude is controlled by the throttle variation δ_{th} (Figure 4). Three PID controllers have to be set, two feedbacks are on the airspeed loop (pitch angle ϑ and airspeed V) and one on the altitude loop (altitude h). The state of the system in this plane, in addition to ϑ, V and h, includes the angle of attack α and the pitch angular velocity q.

In the lateral-directional plane (Figure 5) ailerons δ_a control the heading angle ψ through two feedbacks on the roll and heading angle (respectively ϕ and ψ): two PID loops have to be configured. In this plane, the state of the system is defined by ϕ, ψ, the sideslip angle β, the roll angular velocity p and the yaw angular velocity r.

The root *locus* method allows plotting the position of the poles of a closed-loop transfer function at the changing of a parameter $K_{mult}K_{mult}$ is the gain of $C(s)$, the controller of the plant $G(s)$ of the system having $R(s)$ as input and $Y(s)$ as output. A dynamic compensator is required to achieve the desired performance. The closed-loop transfer function is:

$$\frac{Y(s)}{R(s)} = \frac{C(s)G(s)}{1 + C(s)G(s)} \tag{1}$$

Its poles can be found solving the characteristic equation $1 + C(s)G(s) = 0$. If $L(s)$ is a transfer function proportional to $C(s)G(s)$ through K_{mult}, the characteristic equation can be written as:

$$1 + K_{mult}\, L(s) = 0 \tag{2}$$

Figure 4 Longitudinal control scheme (PIDs)

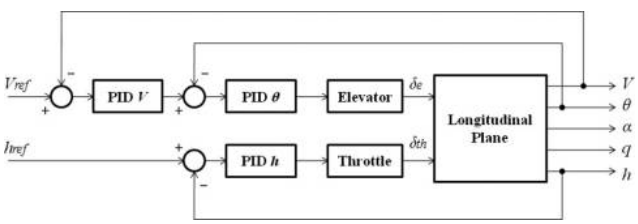
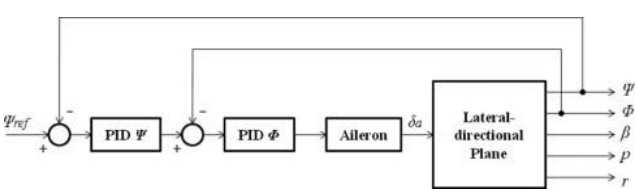


Figure 5 Lateral directional control scheme (PIDs)



The root *locus* represents the position of all the possible solutions of equation (2), as K_{mult} changes from 0 to ∞. Therefore, the root *locus* plot identifies, according to the value of K_{mult}, the position of the poles of the closed-loop transfer function and thus its stability.

Root *locus* method is applied to define PID controller gains K_P, K_I, K_D. PID zeros z_1 and z_2 can be obtained imposing equal to zero the numerator of PID controller transfer function $K_Ds^2 + K_Ps + K_I = 0$. According to Vieta's formula applied to this quadratic expression, the following link between equation zeros and equation coefficients subsists:

$$\begin{cases} z_1 + z_2 = -\dfrac{K_P}{K_D} \\ z_1 \cdot z_2 = \dfrac{K_I}{K_D} \end{cases} \tag{3}$$

from which:

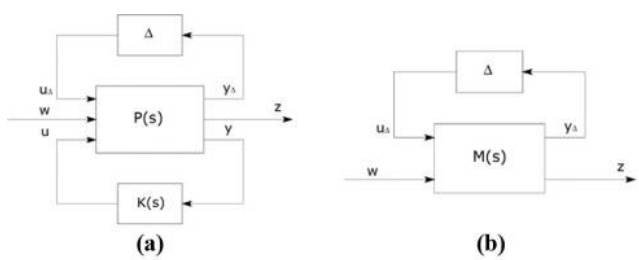
$$\begin{cases} K_P = -K_D \cdot (z_1 + z_2) \\ K_I = K_D \cdot (z_1 \cdot z_2) \end{cases} \tag{4}$$

The derivative gain K_D coincides with K_{mult}. Therefore, the individuation of PID gains requires the definition of three parameters: two PID zeros and K_{mult}. PID zeros are chosen arbitrarily with negative real part to increase system stability. Root *locus* method is applied to define a value of K_{mult} satisfying closed-loop step response requirements in terms of rise time t_r, settling time t_s and overshoot M_P.

At the same time, performance and robust characteristics of the system need to be taken into account. To assess nominal performance, robust stability and robust performance both the longitudinal and lateral-directional schemes of Figures 4 and 5 have to be reshaped into a robust control scheme. According to Zhou and Doyle (1999), the most used configuration in robustness design and analysis of uncertain feedback systems can be redrawn into a general the M-Δ framework [Figure 6(b)].

The verification of this requirement passes through the μ analysis of the frequency response of the full matrix M. In Zhou and Doyle (1999), it is shown that an uncertain system is robustly stable and satisfies (even if only partially) H_∞ performance for all $\Delta(s)$ if and only if the structured singular value μ of the corresponding interconnection model is no greater than one. A robust controller design based on μ analysis is less conservative than a classical robust H_∞ design. Its main benefit is the definition of the robust performance.

Figure 6 General interconnection for robust control problems



Notes: (a) P-K-Δ framework; and (b) M-Δ framework

Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

For the second control scheme (Figure 7), as in Capello *et al.* (2013a, 2013b), the L1 adaptive controller is chosen for the control of the inner navigation loops that need a faster response and are more prone to be affected by unmodeled uncertainties (Hovakimyan and Cao, 2010). The controlled variables include the pitch angle ϑ, the longitudinal component of the airspeed u and the roll angle ϕ. For the outer navigation loop, the altitude h and the heading angle ψ are controlled by PIDs, in which the gains are defined as in the first control methodology (Capello *et al.*, 2012a, 2012b).

The choice of the L1 adaptive controller for the aircraft control is motivated by the high level of uncertainty, which generally characterizes UAVs. The L1 adaptive controller here applied and extensively described in Hovakimyan and Cao (2010) takes into account unmatched uncertainties, which include unmodeled dynamics and state- and time-dependent nonlinearities. The adaptive law is a piecewise constant law, as explained in Cao and Hovakimyan (2009), that guarantees fast estimation. The adaptation rate can be associated with the sampling rate of the autopilot CPU. Moreover, this adaptive algorithm guarantees bounded inputs and outputs, uniform transient response and steady-state tracking. A key feature of this controller, as explained in the previous works, is that the hardware interface is executed at a lower rate (about 10-100 Hz) than the control algorithm (about 100-1,000 Hz), therefore, demanding insignificant CPU power. Thus, the computational power can be used for fast adaptation, and the control law can be computational efficient also for MEMS applications.

The L1 control structure foresees a state predictor, an adaptive law and a control law. The L1 adaptive control architecture (unmatched model) is introduced below:

- The state predictor is defined similar to the plant equations, except for the unknown variables that are replaced by their estimates:

$$\dot{\hat{x}}(t) = A_m\hat{x}(t) + B_m(u(t) + \hat{\sigma}_m(t)) + B_{um}\hat{\sigma}_{um}(t) \quad (5)$$

with $\hat{x}(0) = x_0$ and where $\hat{\sigma}_m(t)$ and $\hat{\sigma}_{um}(t)$ are the adaptive estimates.

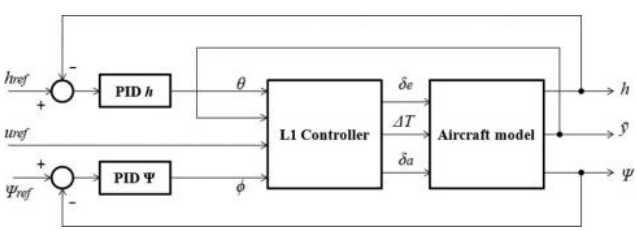
- Adaptive laws:

$$\begin{bmatrix}\hat{\sigma}_m(iT_s)\\ \hat{\sigma}_{um}(iT_s)\end{bmatrix} = -\begin{bmatrix}I_m & 0\\ 0 & I_n - I_m\end{bmatrix}B^{-1}\Phi^{-1}(T_s)\tilde{x}(iT_s)\exp^{A_mT_s} \quad (6)$$

where $\tilde{x}(t) = \hat{x}(t) - x(t)$ is the tracking error:

- Control law – the control signal is the output of the following system:

Figure 7 L1 adaptive controller and PIDs



$$\begin{cases} u(t) = -K_nD(s)\hat{\eta}(s) \\ \dot{\hat{\eta}}(t) = u(t) + \hat{\sigma}_m(t) + H_m^{-1}(s)H_{um}(s)\hat{\sigma}_{um}(t) - r_g(t) \end{cases} \quad (7)$$

with $r_g(s) = K_g(s)r(s)$, where $H_m^{-1}(s)$ and $H_{um}(s)$ are the matrices of the inversion law. These two matrices are fundamental for the creation of the unmatched control laws because they are matched with the low pass filter designed. Note that the low pass filter in the unmatched control law can be different from that for the matched one.

The tracking error dynamics can be derived from the problem formulation and the state predictor as:

$$\begin{cases} \dot{\tilde{x}}(t) = A_m\tilde{x}(t) + B_m\tilde{\sigma}_m(t) + B_{um}\tilde{\sigma}_{um}(t) \\ \tilde{\sigma}_m(t) = \hat{\sigma}_m(t) - u(t) \\ \tilde{\sigma}_{um}(t) = \hat{\sigma}_{um}(t) \end{cases} \quad (8)$$

The sampling time T_S is chosen to ensure that $\gamma_0(T_s) < \overline{\gamma}_0$, where $\overline{\gamma}_0$ is an arbitrarily small positive constant and $lim_{Ts\to 0}\gamma_0(T_s) = 0$ (Hovakimyan and Cao, 2010). If the previous sentence is true, the tracking error between the state of the system and the state predictor can be reduced by reducing the sampling time (both in transient and in steady-state). The adaptation rate is chosen to satisfy that, in this way, the tracking error between the outputs and between the inputs is uniformly bounded by a constant inverse proportional to the adaptive gain.

Two fundamental key features of this controller are the inversion law and the unmatched low pass filters in the control law. The inversion law is related to the pole placement matrix A_m.

The desired dynamics are chosen to optimize the closed-loop response in terms of rise time, settling time and overshoot.

On the longitudinal plane, two oscillating modes are defined, one with high natural frequency and damping (ω_1 = 8 rad/s and ς_1 = 0.9) representing the short period, another representing the phugoid with lower natural frequency and damping (ω_2 = 3 rad/s and ς_2 = 0.8). On the latero-directional plane, two real negative eigenvalues are chosen ($\lambda_1 = -6$ and $\lambda_2 = -10$); the desired oscillating mode representing the dutch roll is defined by ω_3 = 5 rad/s and ς_3 = 0.5. Note that all filters applied in the controller are represented by first-order transfer functions.

Guidance algorithm

The guidance algorithm considered in this work is based on some simplifying hypothesis, according to the flash memory limitation of the autopilot microcontroller. A given set of waypoints is considered, with assigned North, East and altitude coordinates. This set of waypoints includes the starting point, that is the point where the UAV finishes the climb and the autonomous flight starts. The starting point and all the waypoints are supposed to be at the same altitude; thus, a 2D path is considered. Some implementation aspects common to other algorithms available from literature are also considered:

- The trajectory smoother that makes cinematically feasible the assigned trajectory in terms of speed and turn rate constraints.

Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

Figure 8 Guidance algorithm scheme

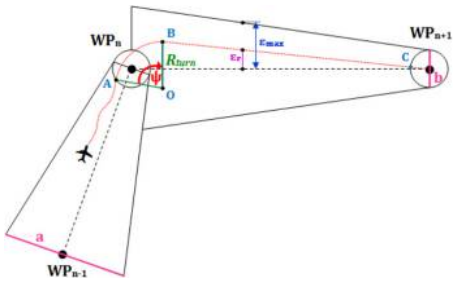
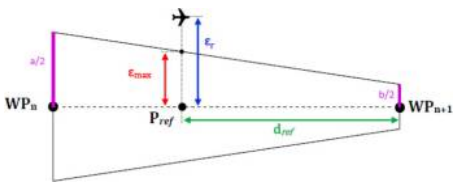


Figure 9 Cross-track error and reference distances



- The cross-track error control to monitor the UAV position with respect to the reference path. We consider the UAV real position P_{UAV} in terms of UAV East and North coordinates, respectively, E_{UAV} and N_{UAV} and the segment connecting two waypoints in terms of previous WP_n (E_n, N_n) and next waypoint WP_{n+1}(E_{n+1}, N_{n+1}). The cross-track error is then calculated as:

$$\varepsilon_r = \frac{|E_{UAV} - mN_{UAV} - (E_n - mN_n)|}{\sqrt{m^2 + 1}} \tag{9}$$

$$m = \frac{E_{n+1} - E_n}{N_{n+1} - N_n} \tag{10}$$

- Look-ahead or proximity distance to define the minimum distance of the UAV from the next waypoint to begin turning. When the distance between the aircraft and the next waypoint is less than this distance, the waypoint is reached and the aircraft can move to the next waypoint.

Considering these hypothesis and aspects, three main phases of the guidance sequence have to be analyzed.

The first phase is the waypoint approach, i.e. when the UAV gets closer to the waypoint with a theoretical straight flight and gets ready to begin turning around the waypoint. In Figure 8, this phase is identified by the red dotted line before the point A. The aircraft is flying from the waypoint $WP_{(n-1)}$ to the waypoint WP_n with an assigned speed and altitude.

It is assumed that the waypoint has been reached when the UAV flies into the imaginary circle centered in the waypoint WP_n with radius equal to the defined proximity distance, set to 20 m according to the MH850 dynamic constraints (in terms of minimum turn radius). From this moment, the UAV can begin turning around the waypoint in the direction of the next waypoint $WP_{(n+1)}$.

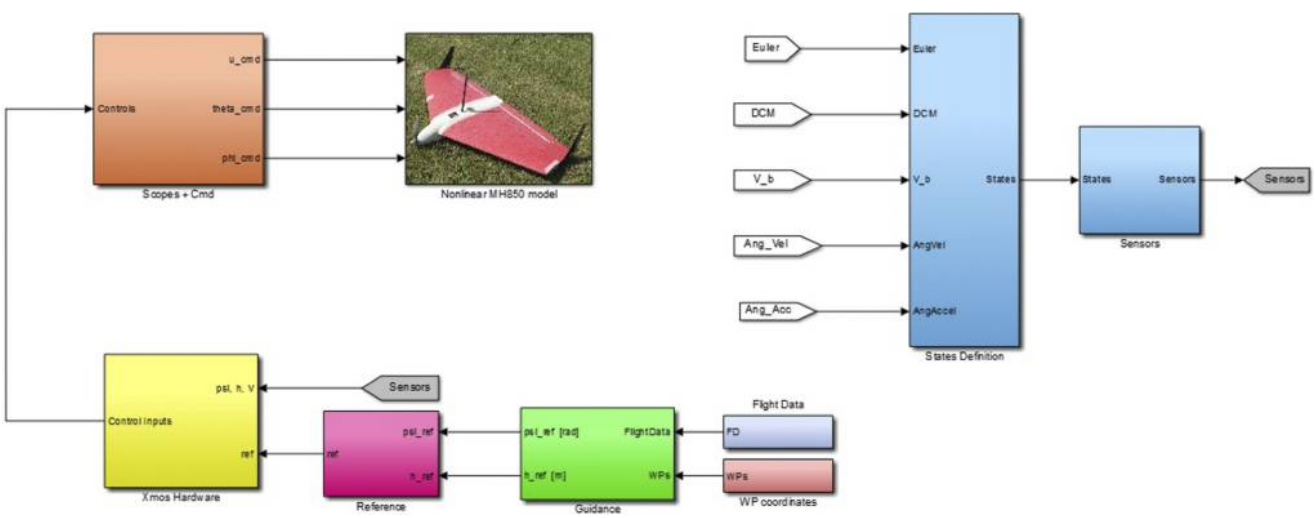
In the second phase, the aircraft turns around the waypoint. The maneuver is accomplished according with the turn rate constraints, function of the cruise speed and of the bank angle. In Figure 8, the turn segment is identified by the red dotted curve between the points A and B. It starts when the distance of the vehicle from the waypoint WP_n is minor or equal to the proximity distance (point A), and it ends when it is verified:

$$|\psi_{UAV} - (\psi_{WP})_+| \leq \Delta\psi$$

where ψ_{UAV} is the UAV heading, $(\psi_{WP})_+$ is the heading angle of the segment that connect the UAV position and the next waypoint $WP_{(n+1)}$ and $\Delta\psi$ is equal to 5°.

The third segment is related to the straight flight that starts at the end of the last turn and ends at the beginning of the next turn. In Figure 8, it has been represented by the red spotted line between the points B and C. In this phase, the cross-track error is applied (Figure 9). Moreover, during the straight flight, to avoid continuous correction of the trajectory and of the flight parameters by elevons, a no corrections region is defined as around the waypoint connecting segment. This means that the corrections on the heading angle are imposed

Figure 10 Matlab/Simulink of the HIL simulations



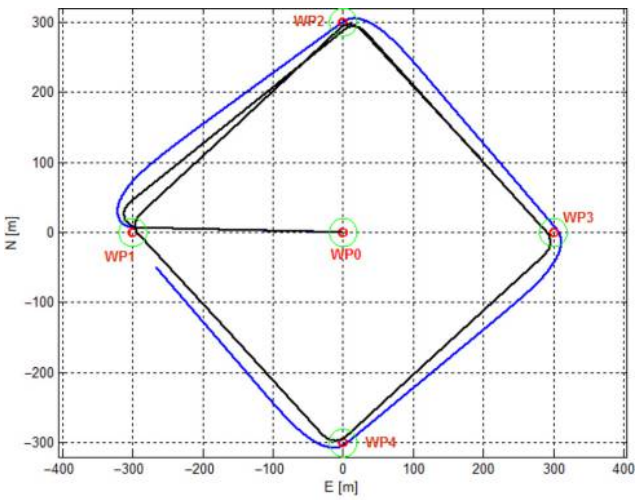
Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

only when the UAV cross-track error is larger than an assigned value (i.e. maximum acceptable cross-track error). This region is defined as an isosceles trapezoid, whose symmetry axis is defined by the segment connecting a waypoint to the next one (i.e. $\overline{WP_n, WP_{(n+1)}}$). The major base (segment a) is set equal to 70 m, considering that the turn radius is about R_{turn} = 22 m if the cruise speed V and the bank angle ϕ are imposed, respectively, equal to 13.5 m/s and 40°. The minor base (segment b) corresponds to the diameter of the circle around the next waypoint and is set equal to 40 m. For the distance definition, see Figure 9.

The maximum acceptable cross-track error ε_{max} is variable and decreases along the segment from 35 (semi-segment a/2) to 20 m (semi-segment b/2), that is:

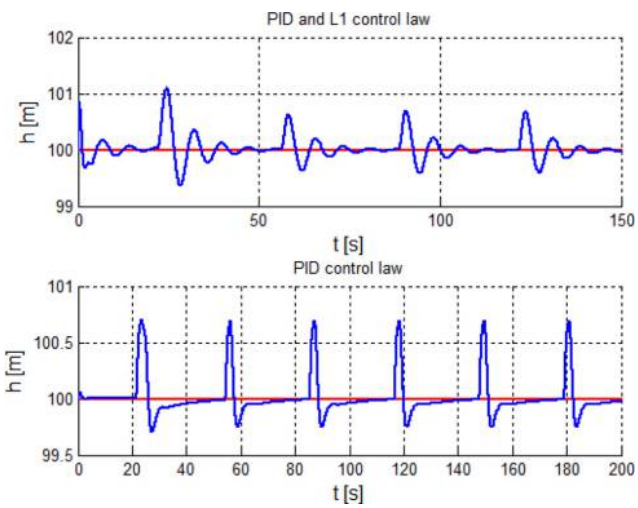
$$\varepsilon_{max} = \frac{d_{ref}\left(\frac{a}{2} - \frac{b}{2}\right)}{d_s} \quad (11)$$

Figure 11 Square trajectory and assigned waypoints



Notes: Blue line: PIDs + L1 law; Black line: PIDs + PIDs law

Figure 12 Aircraft reference (red line) and measured (blue) altitude variation for a square pattern



$$d_{ref} = \sqrt{\left(N_{WP_{(n+1)}} - N_{ref}\right)^2 + \left(E_{WP(n+1)} - E_{ref}\right)^2} \quad (12)$$

where d_{ref} is the reference distance, d_s is the length of the segment $\overline{WP_n, WP_{(n+1)}}$; $N_{WP(n+1)}$ and $E_{WP(n+1)}$ are, respectively, the North and East coordinates of the waypoint $WP_{(n+1)}$; N_{ref} and E_{ref} are, respectively, the North and East coordinates of the reference point, calculated as:

$$N_{ref} = N_{UAV} - \varepsilon_r \psi_{ref} \quad (13)$$

$$E_{ref} = E_{UAV} + \varepsilon_r \psi_{ref} \quad (14)$$

As a result, if the UAV real cross-track error ε_r is smaller than ε_{max} at a certain time during the straight flight phase, no corrections of the aircraft heading angle are carried out. On the contrary, if $\varepsilon_r \geq \varepsilon_{max}$, the new reference heading angle is the heading angle of the segment between the UAV position

Figure 13 Aircraft reference (red line) and measured (blue) speed variation for a square pattern

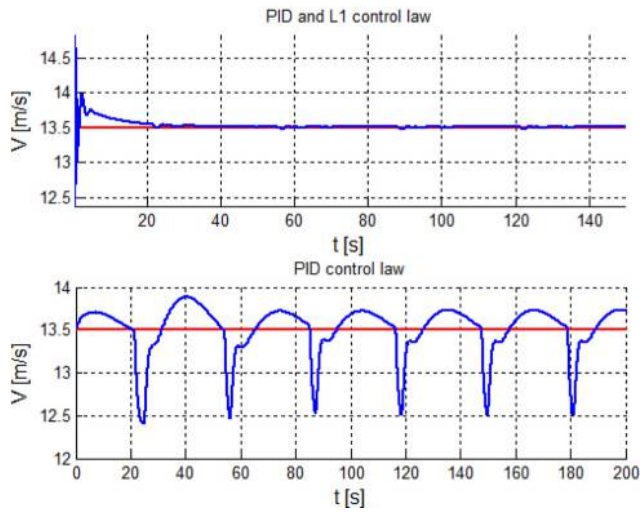
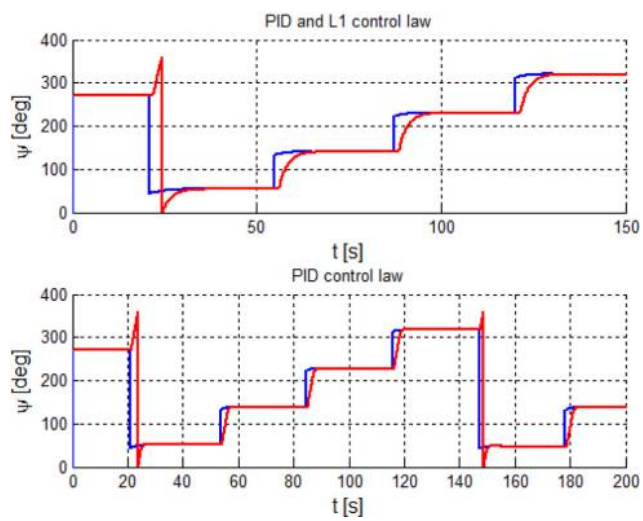


Figure 14 Aircraft reference (red line) and measured (blue) heading angle for a square pattern



Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

and the next waypoint. In this case, the heading angle error is calculated, modulated and then transmitted to the control surfaces, which will move according to the signal received.

Simulation results

The purpose of the HIL simulation is to test the guidance and control algorithms for the autopilot of the MH850; hence, the reference values of the heading angle ψ_{ref} and of the altitude h_{ref} are calculated in the guidance block.

A Matlab/Simulink model is implemented for the algorithm validation in an embedded system via HIL simulations. The main blocks of the model are (Figure 10):

- the nonlinear UAV model, in which the set of equations of motions are implemented;
- the sensor block, in which the onboard sensors and Gaussian white noises that affect the measurements are represented;
- the guidance algorithm; and
- the XMOS hardware, described in a previous section.

The guidance algorithm requires the coordinates of the given waypoint set and the filtered flight data as input data.

The waypoints coordinates are acquired at each time step with an updating frequency of 4 Hz.

In the guidance algorithm, the flight data are acquired every time step from t = 0 s to the end of flight, determined by the accomplishment of the mission (i.e. to reach the last waypoint or to accomplish a certain number of loops of the trajectory). The input flight data are the filtered data obtained with the navigation algorithm (which is based on the filtering action of the implemented Kalman filter), and they are time dependent data.

The HIL simulations are performed with a second-order Heun method with 0.01 s (100 Hz) of time step. Heun method guarantees satisfying accuracy with high computational efficiency. Flight data are received with a 20 Hz frequency to

Figure 15 Control device actions for a square pattern (PIDs and L1)

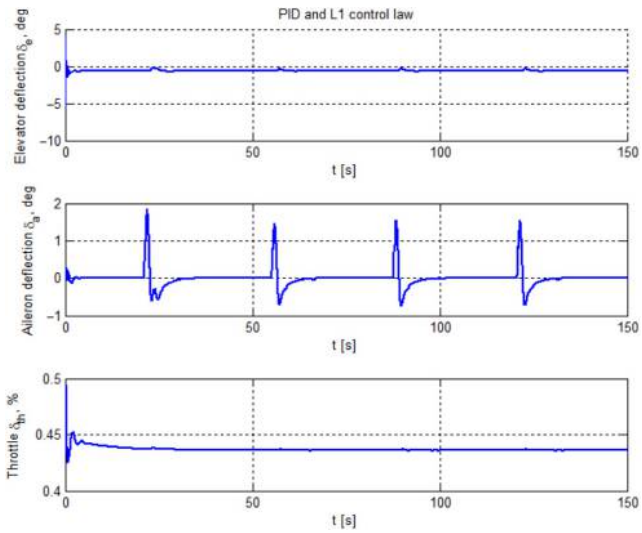


Figure 16 Control device actions for a square pattern (PIDs)

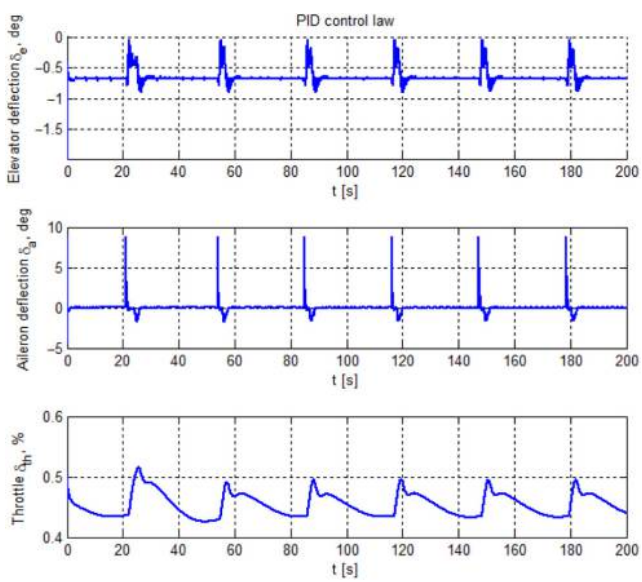
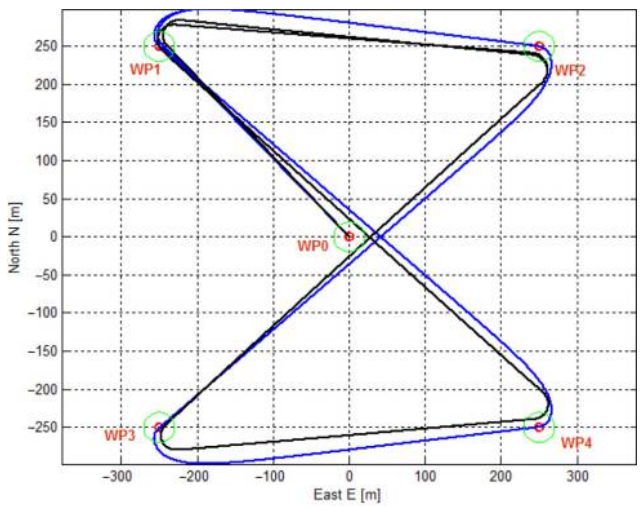
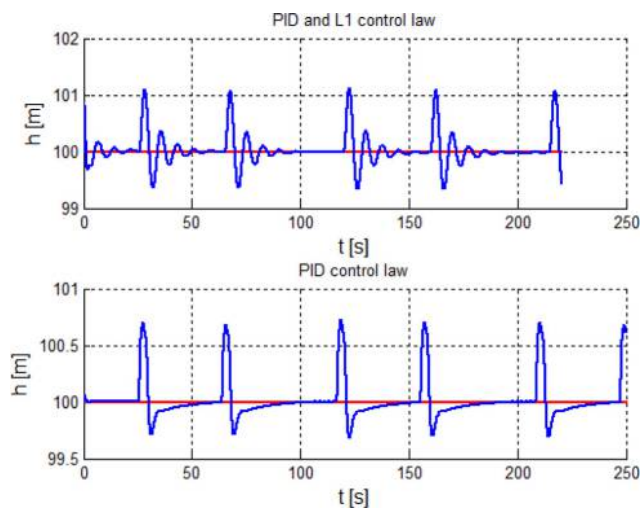


Figure 17 Butterfly trajectory and assigned waypoints



Notes: Blue line: PIDs + L1 law; Black line: PIDs + PIDs law

Figure 18 Aircraft reference (red line) and measured (blue) altitude variation for a butterfly pattern



Aircraft Engineering and Aerospace Technology: An International Journal
Volume 89 · Number 1 · 2017 · 133–144

guarantee a consistent frame rate compatible with the autopilot onboard sensors.

Three different paths are analyzed:

- a clockwise square path;
- a butterfly path; and
- a snake path.

Square pattern

Four waypoints are assigned for the square pattern (Figure 11). The starting point is at $N = 0$ m and $E = 0$ m, and the MH850 starts the flight at its trim conditions. Both control laws are considered.

Good results are obtained in terms of trajectory tracking for both controllers. The altitude (Figure 12) and the airspeed (Figure 13) are well tracked, but the L1 adaptive controller guarantees better results in terms of speed variation, even if more throttle command is required (about 45 per cent of the aircraft weight) with respect to the PID algorithm (Figure 14).

Less oscillations and glitches can be observed in the control deflections actuated by the L1 adaptive controller (Figure 15). This means that even if this control law is not easily implementable because of the controller mathematics, smoothed parameters variations can be better during flight tests. The minimum control corrections guarantee a better track of the airspeed with respect to the PID controller. Moreover, this controller is robust to uncertainties and computational efficient (Figure 16).

Butterfly pattern

For the butterfly pattern, four waypoints are assigned as in Figure 17. As for the previous case, the starting point is at $N = 0$ m and $E = 0$ m, and the MH850 starts the flight at its trim conditions.

Even in that case, good results are obtained for the trajectory tracking for both controllers. As previously

Figure 19 Aircraft reference (red line) and measured (blue) speed variation for a butterfly pattern

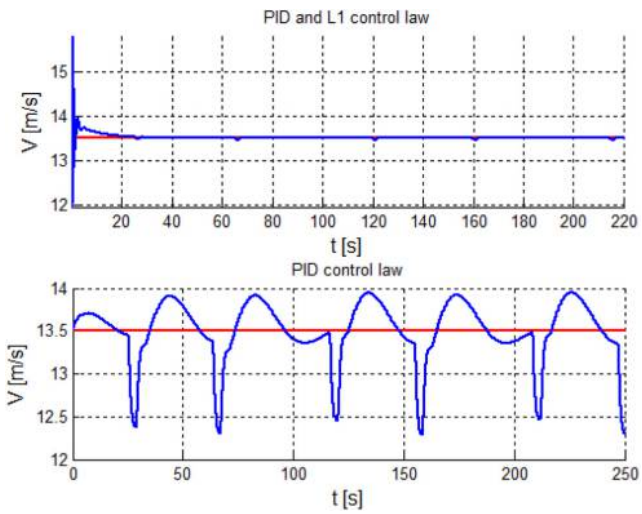


Figure 20 Aircraft reference (red line) and measured (blue) heading angle for a butterfly pattern

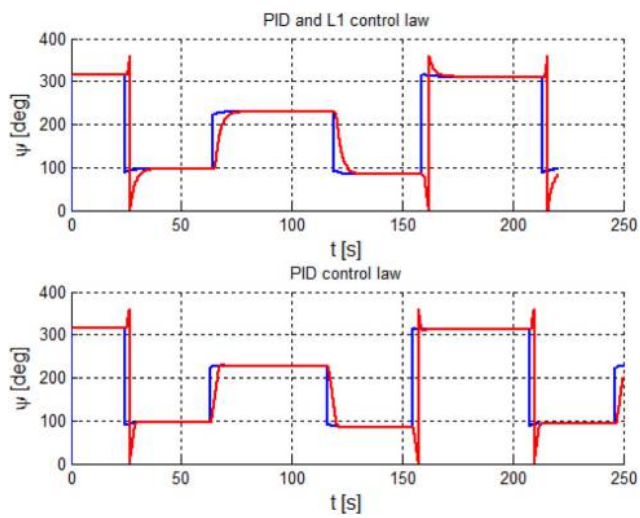


Figure 21 Control device actions for a butterfly pattern (PIDs and L1)

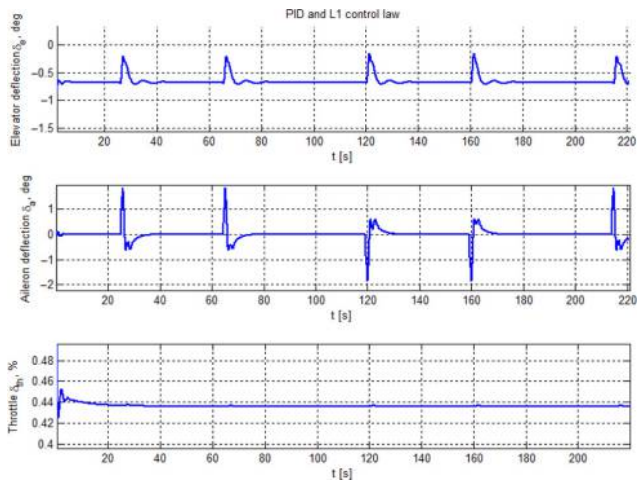


Figure 22 Control device actions for a butterfly pattern (PIDs)

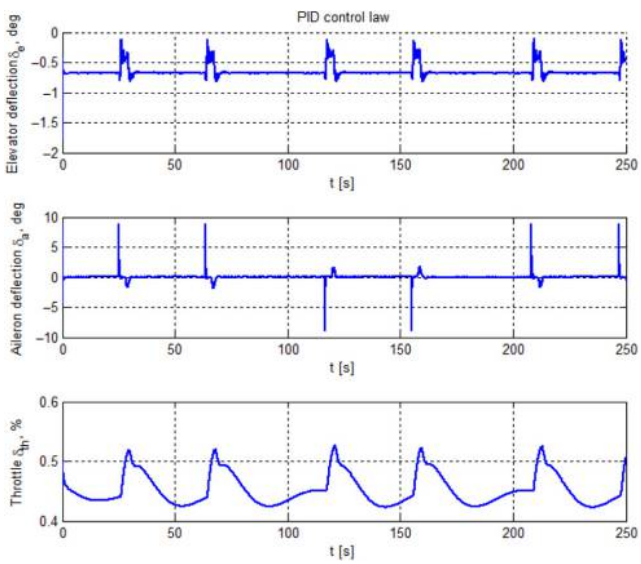
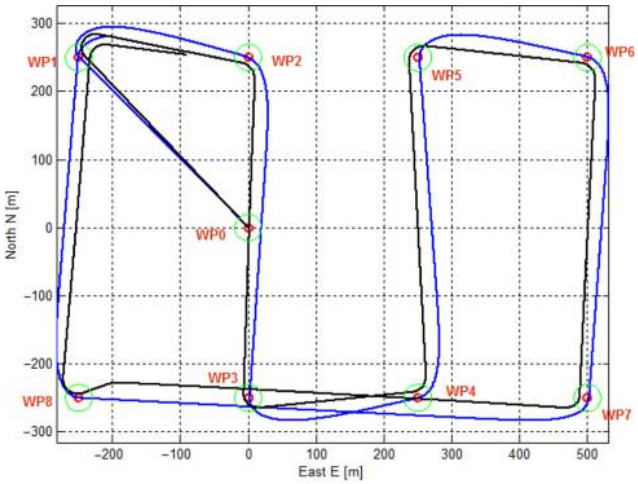


Figure 23 Snake trajectory and assigned waypoints



Notes: Blue line: PIDs + L1 law; Black line: PIDs + PIDs law

Figure 24 Cross-track error variations for the PIDs + PIDs control law

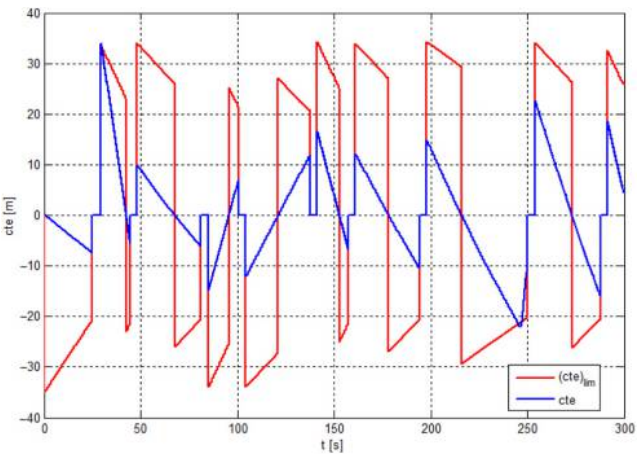
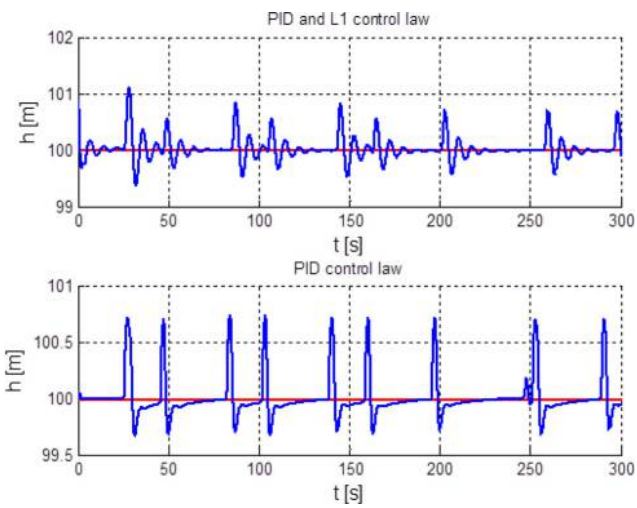


Figure 25 Aircraft reference (red line) and measured (blue) altitude variation for a snake pattern



explained, less oscillations in the control deflections (Figure 21) and smoothing responses (Figures 18 and 19) can be observed for the L1 adaptive controller. No retuning of the PID gains is required even when the waypoints are changed because of the method considered for the definition of the gains (Capello *et al.*, 2012a, 2012b) (Figures 18-22).

Snake pattern

For the clockwise snake pattern, six waypoints are assigned, as in Figure 23. The starting point at $N = 0$ m and $E = 0$ m.

In Figure 24, we can observe that the actual cross – track error cte is greater than the limit one $(\text{cte})_{\text{lim}}$ at about 250 s of simulation for the PID algorithm; this means that control corrections are required. In Figure 29 at about 250 s, we observe oscillations on aileron and throttle commands and, as a consequence, oscillations on variations of the airspeed (Figure 26), altitude (Figure 25) and heading angle (Figure 27) (Figures 24-29).

Figure 26 Aircraft reference (red line) and measured (blue) speed variation for a snake pattern

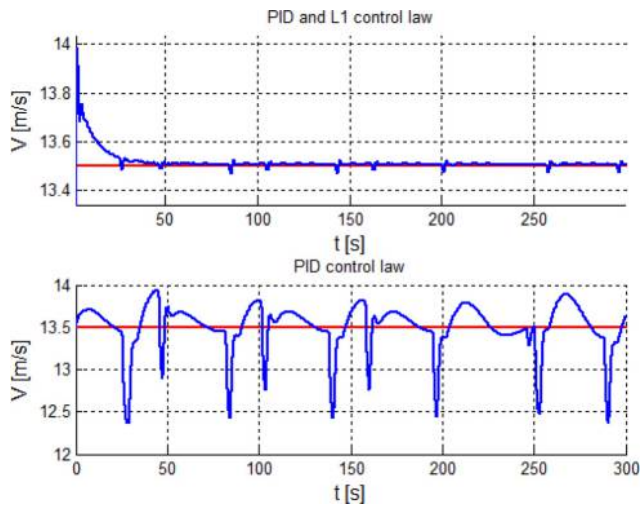


Figure 27 Aircraft reference (red line) and measured (blue) heading angle for a snake pattern

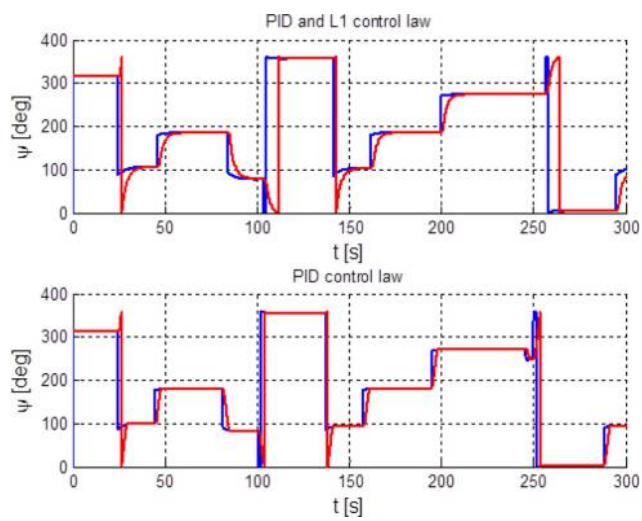


Figure 28 Control device actions for a snake pattern (PIDs and L1)

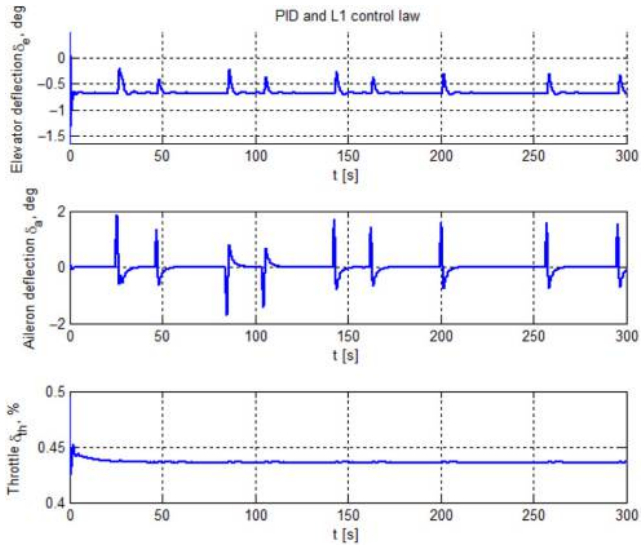
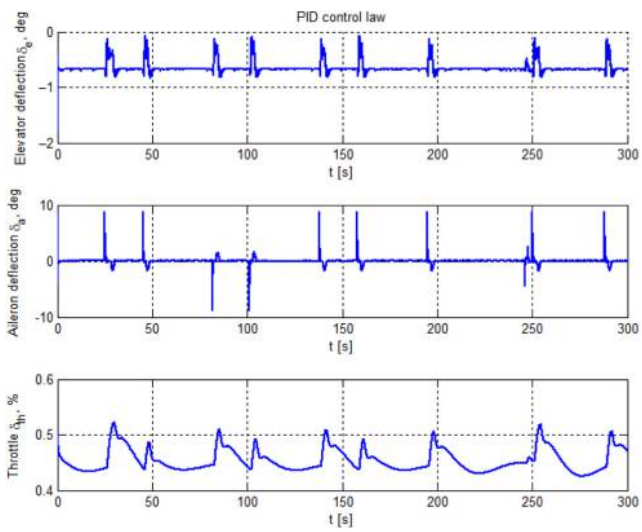


Figure 29 Control device actions for a snake pattern (PIDs)



Conclusion

This paper proposes two different control laws that could be implemented on board. A waypoint guidance algorithm for UAV autopilots has been implemented. HIL simulations are performed to validated the guidance algorithm, including the two proposed controllers. Both the controllers guaranteed good tracking of the waypoints with small variation of the flight parameters. The L1 adaptive controller can guarantee smoothing responses of the control deflections. For this reason, adaptive controllers could be a good candidate for the control of autonomous UAV flight. Moreover, the main drawback of PID autopilot is its lack of adaptation because of the changes in the UAV dynamics. This means that the PID parameters need to be retuned in the presence of wind disturbances or in case the payload characteristics are changed.

Further work

As future work simulations with flight parameter variations will be performed. Moreover, flight tests will be carried out.

References

Barros dos Santos, S., Givigi, S.N. Jr Nascimento, C.L. Jr and de Oliveira, N.M.F. (2011a), "Modeling of a hardware in the loop simulator for UAV autopilot controllers", *Proceedings of COBEM*, *Natal*.

Barros dos Santos, S., Givigi, S.N. Jr Nascimento, C.L. Jr and de Oliveira, N.M.F. (2011b), "Experimental framework for evaluation of guidance and control algorithms for UAVs", *Proceedings of COBEM*, *Natal*.

Beard, R. and Kingston, D. (2005), "Autonomous vehicle technologies for small fixed-wing UAVs", *Journal of Aerospace Computing, Information, and Communication*, Vol. 2 No. 1, pp. 92-107.

Cao, C. and Hovakimyan, N. (2009), "L1 adaptive output-feedback controller for non-strictly-positive-real reference systems: missile longitudinal autopilot design", *Journal of Guidance, Control and Dynamics*, Vol. 32 No. 3, pp. 717-726.

Capello, E., Guglieri, G. and Quagliotti, F. (2009), "UAVs and simulation: an experience on MAVs", *Aircraft Engineering and Aerospace Technology*, Vol. 81 No. 1, pp. 38-50.

Capello, E., Guglieri, G., Marguerettaz, P. and Quagliotti, F. (2013a), "Preliminary assessment of flying and handling qualities for mini UAVs", *Journal of Intelligent and Robotic Systems*, Vol. 9, pp. 109-118.

Capello, E., Guglieri, G., Quagliotti, F. and Sartori, D. (2013b), "Design and validation of an L1 adaptive controller for mini UAV autopilot", *Journal of Intelligent and Robotic Systems*, Vol. 9, pp. 43-61.

Capello, E., Sartori, D., Guglieri, G. and Quagliotti, F. (2012a), "Robust assessment for the design of multi-loop proportional integrative derivative autopilot", *IET Control Theory and Applications*, Vol. 11 No. 6, pp. 1610-1619.

Capello, E., Scola, A., Guglieri, G. and Quagliotti, F. (2012b), "A mini quadrotor UAV: design and experiment", *Journal of Aerospace Engineering*, Vol. 25 No. 4, pp. 559-573.

Degen, S., Alvarez, L., Ford, J. and Walker, R. (2009), *Tensor Field Guidance for Time-Based Waypoint Arrival of UAVs by 4D Trajectory Generation*, Institute of Electrical and Electronics Engineers (IEEE), Big Sky, MT.

Etkin, B. and Reid, L. (1996), *Dynamics of Flight: Stability and Control*, 2nd ed., *A Cura Di*, John Wiley & Sons, New York, NY.

Gong, A., Palmer, J.L., Brian, G. and Verstraete, D. (2014), "Hardware-in-the-loop simulation of a fuel-cell-based UAV propulsion system using real-world flight data", *Proceeding of the Fourth Australasian Unmanned Systems Conference*, *Melbourne*.

Hovakimyan, N. and Cao, C. (2010), *L1 Adaptive Control Theory*, Society for Industrial and Applied Mathematics, Philadelphia.

Khaleghy, A.M. and Song, Y.J. (2014), "A Hardware in-the-loop DDDAMS system for crowd surveillance via

unmanned vehicles", *Proceedings of the Winter Simulation Conference*, *Savannah*.

Kundak, N. and Mettler, B. (2007), "Experimental framework for evaluating autonomous guidance and control algorithms for agile aerial vehicles", *Proceedings of the European Control Conference*, *Kos*, pp. 293-300.

Marguerettaz, P., Sartori, D., Guglieri, G. and Quagliotti, F. (2010), "Design and development of a man-portable unmanned aerial system for alpine surveillance missions", *Proceedings of UAS International Conference*, *Paris*.

Martins, G., Moses, A., Rutherford, M. and Valavanis, K. (2012), "Enabling intelligent unmanned vehicles through XMOS technology", *Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, Vol. 9 No. 5, pp. 71-82.

Meenakshisundaram, V., Gundappa, V. and Kanth, B. (2010), "Vector field guidance for path planning of MAVs in three dimensions for variable altitude maneuvers", *International Journal of Micro Air Vehicles*, Vol. 2, pp. 255-265.

Ntogramatzidis, L. and Ferrante, A. (2011), "Exact tuning of PID controllers in control feedback design", *IET Control Theory and Application*, Vol. 5 No. 4, pp. 565-578.

Nyangweso, E. and Bole, B. (2014), "Development and implementation of a hardware in-the-loop test bed for unmanned aerial vehicles control algorithms", NASA Technical Report, Moffet Field, CA.

Osborne, J. and Rysdyk, R. (2005), "Waypoint guidance for small UAVs in wind", *AIAA@Infotech Aerospace Conference*, *Arlington, VA*.

Shah, Z.M., Ozgoren, K.M. and Samar, R. (2014), "3D guidance of unmanned aerial vehicles using sliding mode approach", *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, Vol. 8 No. 10, pp. 1675-1681.

Skulstad, R. and Syversen, C.L. (2014), "Low-cost instrumentation system for recovery of fixed-wing UAV in a net", Master thesis, Norwegian University of Science and Technology.

Sujit, P., Saripalli, S. and Sousa, J. (2013), "An evaluation of UAV path following algorithms", *Proceedings of the European Control Conference*, *Zurich*.

Zhao, B., Xian, B., Zhang, Y. and Zhang, X. (2015), "Nonlinear robust adaptive tracking control of a Quadrotor UAV via immersion and invariance methodology", *IEEE Transaction on Industrial Electronics*, Vol. 62 No. 5, pp. 2891-2902.

Zhou, K. and Doyle, J. (1999), *Essentials of Robust Control*, Prentice Hall, New York, NY, s.l.

Further reading

XMOS (s.d.), available at: www.xmos.com/

Corresponding author

Elisa Capello can be contacted at: elisa.capello@polito.it