EXPERIMENTAL FRAMEWORK FOR EVALUATION OF GUIDANCE AND CONTROL ALGORITHMS FOR UAVs

Sérgio Ronaldo Barros dos Santos, sronaldo@ita.br
Cairo Lúcio Nascimento Júnior, cairo@ita.br
Instituto Tecnológico de Aeronáutica, Praça Marechal Eduardo Gomes, 50, Vila das Acácias, São José dos Campos, SP.

Sidney Nascimento Givigi Junior, sidney.givigi@rmc.ca
Royal Military College of Canada, P.O. Box 17000 Station Forces, Kingston, Ontario K7K 7B4, Canada.

Adriano Bittar, bittar@ita.br
Neusa Maria Franco de Oliveira, neusa@ita.br
Instituto Tecnológico de Aeronáutica, Praça Marechal Eduardo Gomes, 50, Vila das Acácias, São José dos Campos, SP.

Abstract. *This paper describes a software-in-the-loop simulation environment for the test and validation of a hierarchical guidance and control algorithm for a small fixed-wing unmanned aerial vehicle (UAV), utilizing the commercially available X-Plane flight simulator and Matlab/Simulink. This experimental framework will allow students and researchers to develop, to evaluate and to validate multi-aircraft flight guidance and control algorithms using a realistic unmanned aircraft models in real-world modeled environments, through the X-Plane Flight Simulator. We present an algorithm framework for implementing the overall path planning and control architecture. The proposed flight guidance algorithm uses the heading angle information from the magnetic heading vector and velocity vector, obtained through the navigation system of the virtual unmanned aircraft available in X-Plane simulator. The guidance logic calculates the difference between the Line of Sight (LoS) angle and UAV heading angle and this difference is used to compute the command to be given to the designed autopilot controller. So, the flight guidance logic provides the necessary commands to the next control stage in order to accomplish the goal of the mission, which is reach a pre-defined waypoint. The complete mission to be achieved is defined by an ordered sequence of waypoints and a set of constraints to satisfy. Basically, this experimental framework employs the Matlab/Simulink running the flight guidance and autopilot controller to be tested, and another PC running the X-Plane flight simulator with the unmanned aircraft to be commanded. These resources are interconnected through data buses in order to exchange information. The interconnection between Matlab/Simulink host and the X-Plane host is made through data communication bus based on network protocol, Ethernet communication module (TCP/IP) and Uniform Datagram Protocol (UDP) available in both systems. The X-Plane has the built-in capability of transmitting flight parameters and receiving command for aircraft flight control surfaces over Ethernet using UDP. Through this platform, designed flight guidance and autopilot systems can be applied into models similar to real aircraft minimizing risks and increasing flexibility for design changes. As study case, were implemented a flight guidance consisting of trajectory generation, guidance direction and an autopilot controller for an UAV target and the results obtained from the guidance and control system are presented.*

Keywords*: Guidance, UAV, Autopilot Controllers, X-Plane Flight Simulator.*

1. INTRODUCTION

Autonomous, unmanned ground, sea, and air vehicles have become indispensable both in the civilian and military sectors. The maneuvering capabilities of small-scale aircraft make them particularly useful aerial vehicles for numerous tasks in different complex environments (Chowdhary, Ottander, Salaün, and Johson, 2009). Current military operations, in particular, depend on a diverse fleet of unmanned (primarily aerial) vehicles that provide constant and persistent monitoring, surveillance, communications, and–in some cases– even weapon delivery. This trend will continue, as new paradigms for their use are being proposed by military planners. Unmanned vehicles are also used extensively in civilian applications, such as law enforcement, humanitarian missions, natural disaster relief efforts, etc. However, controlling a vehicle in a three-dimensional space is technically and scientifically challenging. Such vehicles have complex, nonlinear dynamics, the flight conditions are often permeated with large disturbances, and the environment may be uncertain and dynamically changing (Eisenbeiss, 2004).

Both robot and control scientists have shown interest in general autonomous vehicle control problem and various techniques have been developed during the last decades. There is a clear division between more practical techniques from the robot motion planning community and control theoretic techniques like nonlinear trajectory generation. Several experimental testbeds have been developed for unmanned vehicle. Many of the developments have focused on the complex vehicle system integration problem (Garcia, and Barnes, 2010). It still requires large amounts of time and effort to build a custom experimental setup; to integrate sensors, processors, and develop the complex software to operate system. Also, these experimental facilities are usually not flexible in that hardware changes and software updates can consume significant amount of time. Finally, it is also difficult to precisely know the actual flight

conditions from the collected flight data because of un-measurable extraneous effects. Part of the methodological shortcomings is due to the lack of experience in implementing and evaluating planning and control algorithms in ways that encompass the scope of conditions faced in real world applications (Hattenberger, Alami, and Lacroix, 2006).

With this motivation we have developed an experimental setup for miniature aircraft. Our setup overcomes many of these challenges by taking advantage of recently available off-the-shelf mini helicopters and aircraft, by using a flight simulator which runs a real UAV, minimizing risk found in field test. This setup simplifies the testing of the guidance and control algorithms, and finally, it allows us to demonstrate the functionality of the algorithms in near realistic conditions while giving access to all effects into the overall performance (Johnson, Fontaine and Kahn 2001).

The process of test is known as Software-In-Loop (SIL) Simulation. Early users of the SIL were from the aerospace sector, in which reliability and high performance is demanded. Presently this technology is used in every engineering field and it is also one of the methods to test the product before the actual launch. All parts of this work has been conducted in academia, since these platforms offer an excellent avenue for students to be involved in the design and testing of sophisticated control and guidance algorithms (Meister, Frietsch, Seibold, and Trommer, 2008).

In this paper is presented a complete SIL solution for testing and validation UAV guidance and control algorithm. Each stage provides the necessary commands to the next control stage in order to accomplish the goal of the mission, specified at the top level. The execution of the entire control and guidance algorithm is demonstrated through a realistic flight simulator environment. All guidance and control algorithms are coded in Matlab/Simulink, which schedules each task efficiently. The SIL simulator consists of a PC, running the guidance and control algorithm developed using the Matlab/Simulink and another PC running the commercial X-Plane flight simulator. The two components are interfaced via Ethernet (TCP/IP) using UDP protocol. With this set-up, several algorithm of different complexity can be tested and validated with X-Plane's sophisticated flight and atmospheric models. The X-Plane transmit flight parameters and receiving command for aircraft flight control surfaces over interface communication available in both software.

The paper is organized as follows. The next sections describe the problem to be addressed, which is the platform to be developed UAV guidance and control tests. In section 3 the solution given is presented and also the softwares, the communication protocols and the signal used in communication to acomplish the task are described. Section 4 presents the guidance algorith and the control law used to validate the developed framework and section 5 presents the actual implementation of them. Section 6 presents the results obtained. The conclusion discusses the validity of the implemented framework based on the results obtained. Also, the research to be done as a sequence of this work is pointed.

2. PROBLEM STATEMENT

The main objective of this paper is to describe the applicability of SIL simulator environment to aid in the guidance and autopilot design process. This experimental framework will allow students and researchers to develop, to evaluate and to validate multi-aircraft flight guidance and control algorithms using realistic unmanned aircraft models in real-world modeled environments, through the X-Plane Flight Simulator (Gholkar, Isaacs, and Arya, 2004) and (Jung, Ratti, and Tsiotras, 2009). So, the parameters of flight as well as aircraft responses can be monitored and easily analyzed. With these experiments, we expect to improve our insight into key factors that determine performance and obtain indication about the behavior of algorithms. These features increase the flexibility to implement changes and immediately check out the results. Figure 1 show the test platform used to the experimental testing of guidance and control algorithms in realistic conditions.

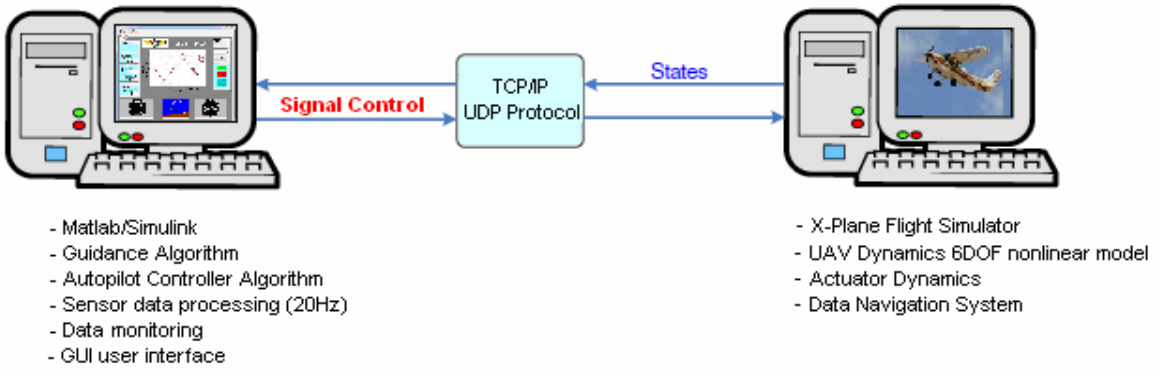


Figure 1. Environment for rapid testing of the guidance and control algorithm

To verify the functionality of the framework proposed, we present a study and implementation of the overall control architecture, including trajectory generation and guidance system. The proposed flight guidance algorithm use the magnetic heading angle and body velocity vector, obtained through the navigation system of the virtual unmanned aircraft available in X-Plane simulator.

Autonomous guidance and control for small UAVs imposes severe restrictions on control algorithm development, stemming from the limitations imposed by the on-board hardware and the requirement for real-time implementation. In order to overcome these limitations computationally efficient algorithms should be developed and the onboard computational resouces should be wisely used. A complete solution to fully automated, unsupervised, guidance and control of UAVs remains a difficult undertaking. Hierarchical structures have been successfully applied in many cases in order to deal with the issue of complexity (Lum, Rowland and Rysdyk, 2008). In such hierarchical structures the entire control problem is subdivided into a set of smaller sub-control tasks, see Figure 2. This allows for a more straightforward design of the control algorithms for each modular control task. It also leads to simple and effective implementation in practice.

In fixed-wing, the accuracy of convergence is influenced by perturbation like the crosswind, and the flight path to the next waypoint affected also by the states of approach. Considering the UAV usage and ability, the algorithm performance is significantly influenced by wind perturbation or the closeness to a waypoint. The flight guidance provides the necessary commands to the control stage in order to accomplish the goal of the mission. The mission to be achieved is defined by an ordered sequence of waypoints and a set of parameters to be satisfied (Han, Kim, Lee and Cho, 2008).

3. AIRCRAFT MODELING

The Cessna 182 is a well known aircraft with a lot of references in literature, besides, also is the default aircraft on the X-Plane Flight Simulator. Based on those natures, this aircraft was chosen to be used on this approach.

The longitudinal and lateral equations of the aircraft are obtained from the rigid body equations of motion. Using the aerodynamic stability coefficients, and geometry and mass reference for the Cessna 182, the longitudinal and lateral equations was linearized. All theses parameters are available in (Roskam, 2001).

The lateral and longitudinal model for the Cessna 182 was linearized using the same flight condition established to the test and validation of the guidance and control algorithm designed, such as; Altitude, h of 4870 (ft), Forward Speed, μ_0 of 90 (knot) and Dynamic Pressure, $\bar{q}$ of 49.6 (lbs/ft^2).

The linearized longitudinal state-space model for the Cessna 182 is showed in Eq. (1). The states of the linearized longitudinal model are forward speed (u), angle-of-attack (α), body axis pitch rate (q) and body axis pitch attitude angle (θ). Elevator deflection (δ_e) and engine deflection (δ_T) are the longitudinal controls.

$$\begin{bmatrix} \dot{u} \\ \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.0307 & 19.6083 & 0 & -32.37 \\ -0.001336 & -2.1276 & 1 & 0 \\ 0.003974 & -13.8501 & -6.8791 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0 & 0.012215 \\ -0.20596 & 0 \\ -34.727 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_e \\ \delta_T \end{bmatrix} \tag{1}$$

The linearized lateral-directional state-space model, Eq. (2), was obtained in the same manner as longitudinal model for the Cessna 182. The state of the lateral-directional model are sideslip angle (β), body axis roll rate (p), body axis yaw rate (r), and body axis roll attitude angle (ϕ). Aileron deflection (δ_a) and rudder deflection (δ_r) are the lateral-directional controls.

$$\begin{bmatrix} \dot{\beta} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} -0.18679 & -0.0029155 & -0.99168 & 0.14707 \\ -30.2497 & -12.9738 & 2.1391 & 0 \\ 9.2717 & -0.3591 & -1.2105 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ p \\ r \\ \phi \end{bmatrix} + \begin{bmatrix} 0 & 0.08889 \\ 75.0507 & 4.8177 \\ -3.4117 & -10.1879 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix} \tag{2}$$

The vertical speed is defined as the climb or descent velocity of the aircraft during the longitudinal movements. The vertical speed of the aircraft can be compute using the Eq. (3).

$$w = \alpha \cdot \mu_0 \text{ (feet/s)} \tag{3}$$

Altitude hold is an important mode in longitudinal autopilot systems. The altitude can be directly controlled by change of body axis pitch rate, produced by elevator deflection commands. The altitude transfer function found to the Cessna 182 is given in Eq. (4).

$$\frac{h(s)}{q(s)} = \frac{-1.30559s^2 - 8.9794s + 450.1868}{s^3 + 2.04546s^2} \text{ (feet)} \tag{4}$$

Typical for this type of aircraft, Cessna 182, the elevator, aileron and rudder servo transfer function can be represented as first-order lag. The used elevator, aileron and rudder servo transfer function are respectively showed in Eq. (5).

$$\frac{\delta_e(s)}{e(s)} = -\frac{10}{s+10}, \frac{\delta_a(s)}{e(s)} = \frac{10}{s+10}, \frac{\delta_r(s)}{e(s)} = -\frac{3}{s+3} \tag{5}$$

On the other hand, the engine dynamics transfer function is represented as second-order lag. Thus, the used engine dynamics transfer function is given in Eq. (6).

$$\frac{\delta_T(s)}{e(s)} = \frac{16.6}{s^2 + 11.66s + 16.6} \tag{6}$$

4. PROBLEM SOLUTION

In this section, we briefly describe test platform, which has been recently developed by the authors, and which takes into account the available resources of the Matlab/Simulink and X-Plane flight Simulator.

Figure 2 shows the overall control hierarchy. It consists of trajectory generation, guidance direction and autopilot controller implemented in Matlab/Simulink and actuator dynamics, aircraft dynamics and navigation system provided by X-Plane Flight Simulator.

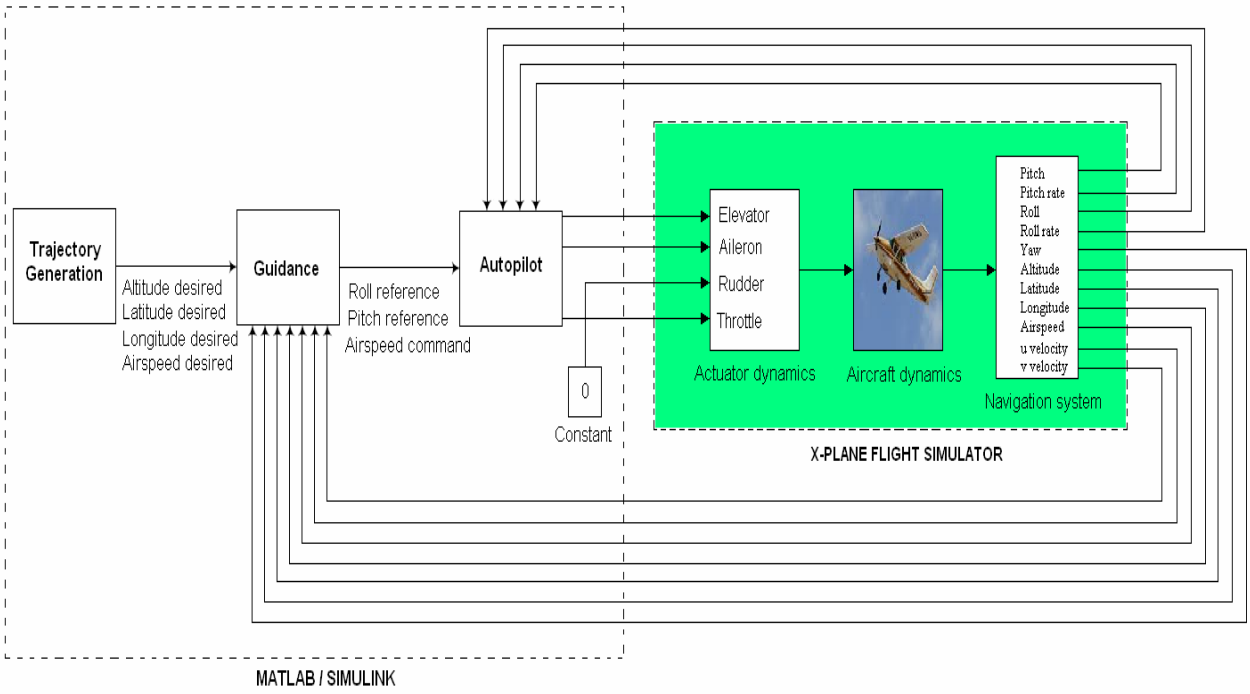


Figure 2. Test Platform block diagram

Thus, the basic principle of the test platform consists in establishing the data change between Matlab/Simulink and X-Plane simulator by use of Ethernet communication module (TCP/IP) using the Uniform Datagram Protocol (UDP) available in both systems. This experimental framework employs the Matlab/Simulink running the flight guidance and autopilot controller to be tested, and another PC running the X-Plane flight simulator contained the unmanned aircraft to be commanded. The X-Plane has the built-in capability of transmitting flight parameters and receiving command for aircraft flight control surfaces over Ethernet using UDP. Through this platform, designed flight guidance and autopilot systems can be applied into models similar to real aircraft, minimizing risks and increasing flexibility for design changes.

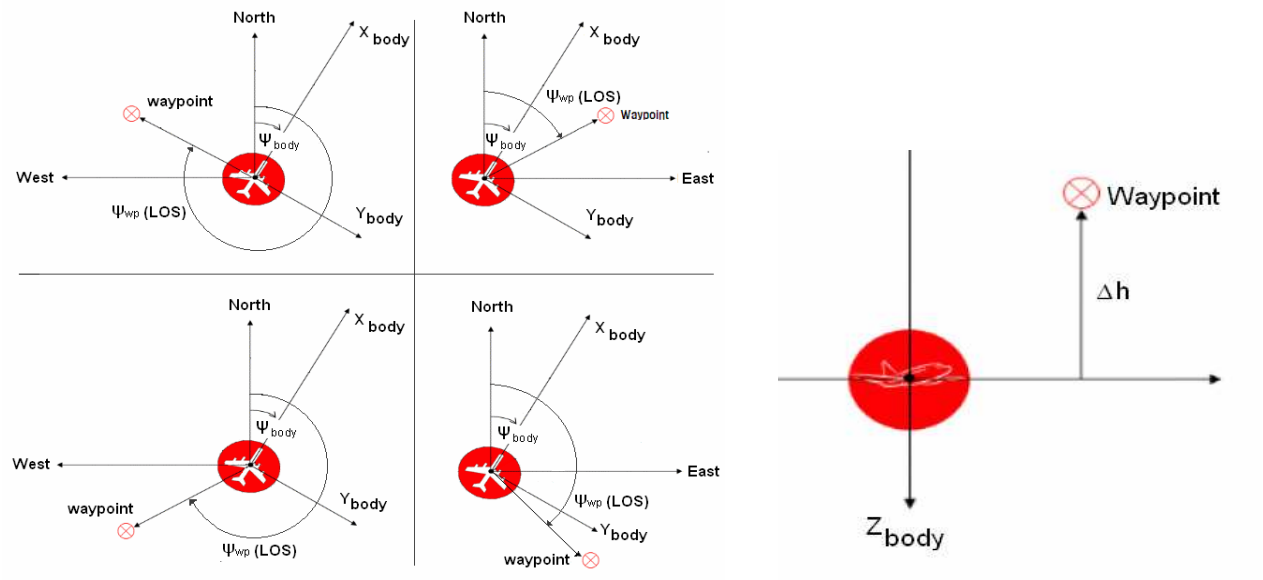
5. GUIDANCE AND CONTROL ALGORITHM

There are several techniques that can be used for determining the path that will be followed by the UAV. The algorithms available in literature focus on the following:

- Goal reaching: The UAV will only want to reach its target waypoint, without too much focus on the efficiency of the path taken.

- Path optimization: The UAV will take the shortest and most optimal path to its goal, in addition to reaching its goal.

The points to be reached by the UAV during the flight are defined by waypoints through the altitude, latitude and longitude desired. The trajectory block changes the waypoint when UAV reaches the desired position. It occurs when the UAV enter in a circle with pre-defined radius, around the waypoint. When the last waypoint is achieved, then, the aircraft start to fly around waypoint inside circle (Han, Kim, Lee and Cho, 2008). Figure 3 shows the possible geometries to be considered by the lateral and longitudinal guidance problem.



(a) (b)

Figure 3. Lateral (a) and Longitudinal (b) Guidance Law Geometry.

The magnetic heading angle provided by Attitude and Heading Reference System (AHRS) of the virtual UAV has the range of 0~360 degree, so it should be converted to -180~180 degree. Also it's important to notice in which quadrant the selected waypoint is located in order to avoid numerical problems. These kinds of problems can be solved through of atan2 that represents the 4-quadrant tangent function. Therefore the LoS can be calculated using the Eq. (7).

$$LoS = a\tan 2\left[\frac{Longitude_{waypoint} - Longitude_{aircraft}}{Latitude_{waypoint} - Latitude_{aircraft}}\right] \tag{7}$$

The guidance logic computes the difference between the LoS and the magnetic heading of the body and this value is used as the heading error in order to determine the flight command to be given (Han, Kim, Lee and Cho, 2008). The heading error can be calculated as:

$$\psi_{error} = LoS - \psi_{mag} \tag{8}$$

In the following development consider (X_{body}, Y_{body}, Z_{body}) the body coordinate frame and (u, v, w) the body velocity vector represented in this frame (Han, Kim, Lee and Cho, 2008). Using directly Eq. (7) to compute the heading error, the aircraft is more predisposed to suffer from the crosswind. Hence a new strategy is proposed, which helps to reduce the influence of the sideslip caused by crosswind. In this new strategy the angle between the lateral and longitudinal velocity is used, increasing the accuracy and the robust of the system:

$$\alpha_{velocity} = a\tan 2\left[\frac{v}{u}\right] \tag{9}$$

The heading error is now calculated as:

$$\psi_{error} = LoS - \left(\psi_{mag} + \alpha_{velocity}\right) \tag{10}$$

The controllers utilized were estimated using the aircraft dynamics model of the plane used in X-Plane flight simulator. In straight level flight, the heading error is used to lateral direction control through a PID controller (Lewis McBride, Mikkelson and Nguyen, 2005). The roll command is given by Eq. (11), in which the heading error ψ_{error} can be as in Eq.(8) or Eq.(10).

$$\phi_{cmd} = kp\left[\psi_{error}\right] + ki\int\left[\psi_{error}\right]dt + kd\frac{d\left[\psi_{error}\right]}{dt} \tag{11}$$

To altitude hold a PID is also used and the pitch's command can be calculated using:

$$\theta_{cmd} = kp\left[Altitude_{error}\right] + ki\int\left[Altitude_{error}\right]dt + kd\frac{d\left[Altitude_{error}\right]}{dt} \tag{12}$$

A PID controller is also used in airspeed control; therewith the throttle's command is given by:

$$throttle_{cmd} = kp\left[Airspeed_{error}\right] + ki\int\left[Airspeed_{error}\right]dt + kd\frac{d\left[Airspeed_{error}\right]}{dt} \tag{13}$$

The commands range used to set elevator, ailerons and throttle is given in Eq. (14), respectively.

$$-\pi/4 \le \phi_{cmd} \le \pi/4\,,\ -\pi/6 \le \theta_{cmd} \le \pi/6\,,\ 1/3 \le throttle \le 1 \tag{14}$$

The rudder command is fixed in zero in order to not disturb the control, since this surface is not used in implementation. Besides, the trim tabs of each surface were fixed in the neutral position.

6. IMPLEMENTATION

The guidance algorithm and the control law used in the proposed framework were developed in Matlab/Simulink environment.

In order to facilitate the operation and change of controller characteristics and waypoints desired, it was developed a reliable ground station software using the Graphics User Interface (GUI) tool available in Matlab. This ground station software is used to upload the altitude, latitude and longitude of each waypoint, and also define the desired UAV airspeed. Figure 4 shows the User Interface developed

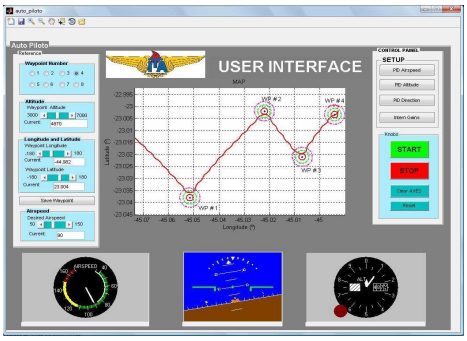


Figure 4. Ground Station Software developed

The parameters and control loop gains can be tuned or changed through of User Interface, and they are instantly modified in the guidance and control algorithm.

In Fig. 5 in the left is the lateral guidance block, responsible for generating the roll command. Notice that the roll command can be generated either by the aircraft heading, when the aircraft yaw information from X-Plane is used or by the body velocity vector, for which computation the velocities u and v from X-Plane are used. These possibilities

implements the two guidance strategies already described. The longitudinal guidance, Fig. 5 top in the right, uses a PID controller in which the input error signal is the difference between current altitude and target altitude and is responsible for generating the pitch command.

In Fig. 5 down in the right, the airspeed control uses the current airspeed from the sensor, the value of the airspeed desired and a PID controller to form the control command which is sent to the throttle servo in the aircraft.

The roll and pitch commands generated by the guidance block are not sent to X-Plane, but used by the controller block, Fig. 6, which generates the actual commands to be sent to the ailerons and elevator surface, respectively, in the aircraft running in X-Plane. The X-Plane sends back the aircraft pitch rate and roll rate to be used by the controller block in the feedback loop. Therefore, there are two stability autopilots: longitudinal control autopilot, used to hold the aircraft stable along the longitudinal axis, and lateral control autopilot, used to hold the aircraft stable along the lateral axis, or to make the aircraft return to its trim conditions with minimum disturbances in terms of motion.

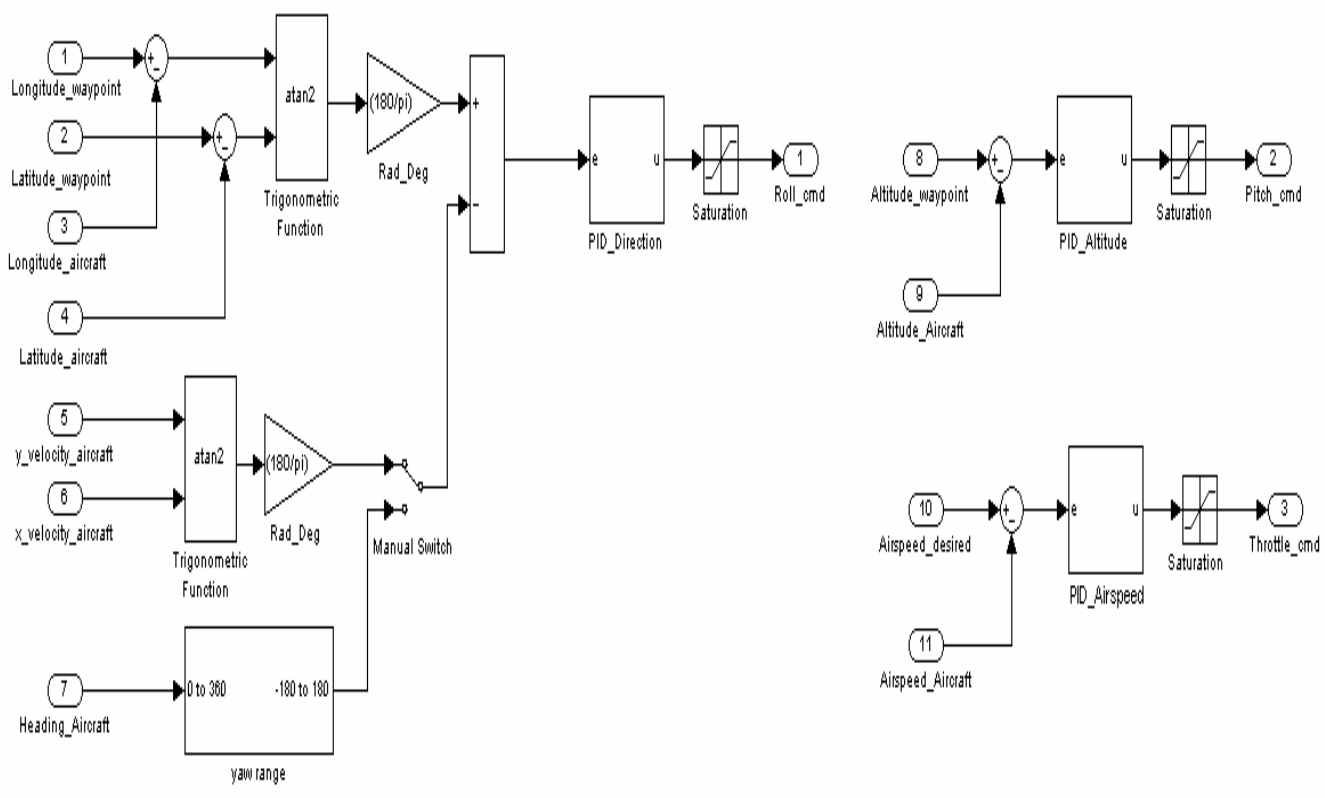


Figure 5. Block diagram of the guidance algorithm and altitude and airspeed control

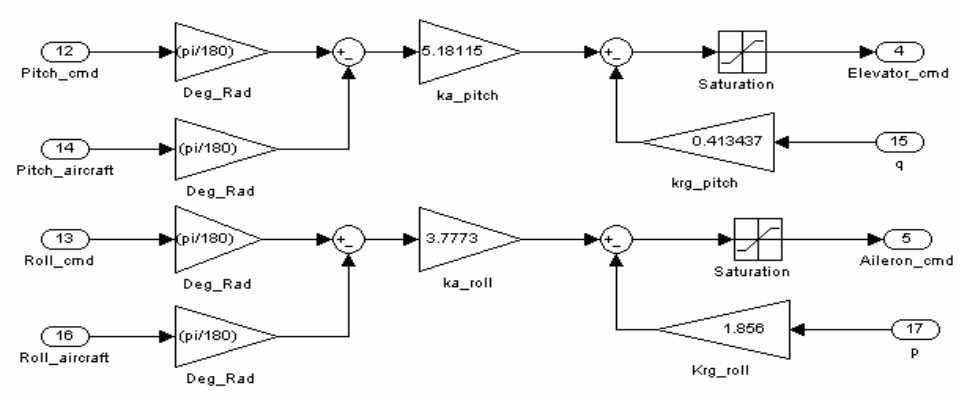


Figure 6. Block Diagram of the control law.

7. RESULTS

To utilize the Software in the Loop (SIL) configuration, the un-compiled algorithm source code, which normally runs on a dedicated desktop computer, should be compiled during the execution of flight simulator tool, allowing this algorithm to be tested in the flight simulator installed on another computer. The results, obtained using the proposed

framework in Fig.1, show the flight trajectory executed by the aircraft in XY plots visualized in Matlab. Figure 7 presents the results obtained with the guidance law using the difference between LOS angles and heading angle. The magnetic heading angle used is given by the X-Plane. The waypoints are pre-defined and the algorithm knows that the waypoint was reached when the distance from the aircraft to the waypoint is less than a chosen distance. These distances are represented in the plots as circular regions around the waypoints. Four situations are presented. In the first situation just one waypoint was defined. When the aircraft reaches the waypoint it stays in a circular trajectory around it – Fig.7 up in the left side. The second situation presents two waypoints defined. The aircraft pass through the first waypoint that means reaches the circular region defined, and goes to the next one. When the second waypoint is reached the aircraft stays flying around it – Fig. 7 up in the right side. In the other two situations, three and four waypoints were defined. In these tests the wind effects were not considered.

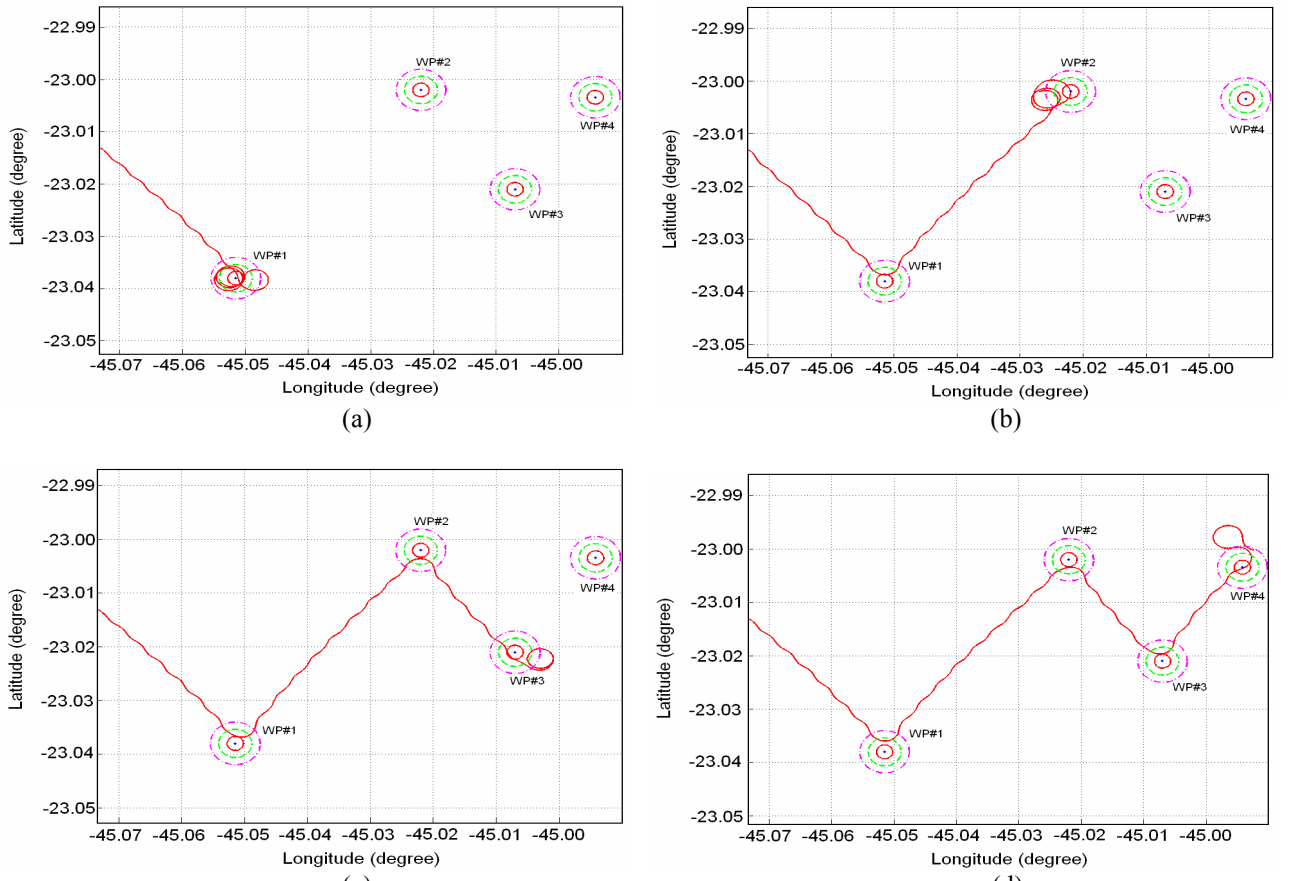
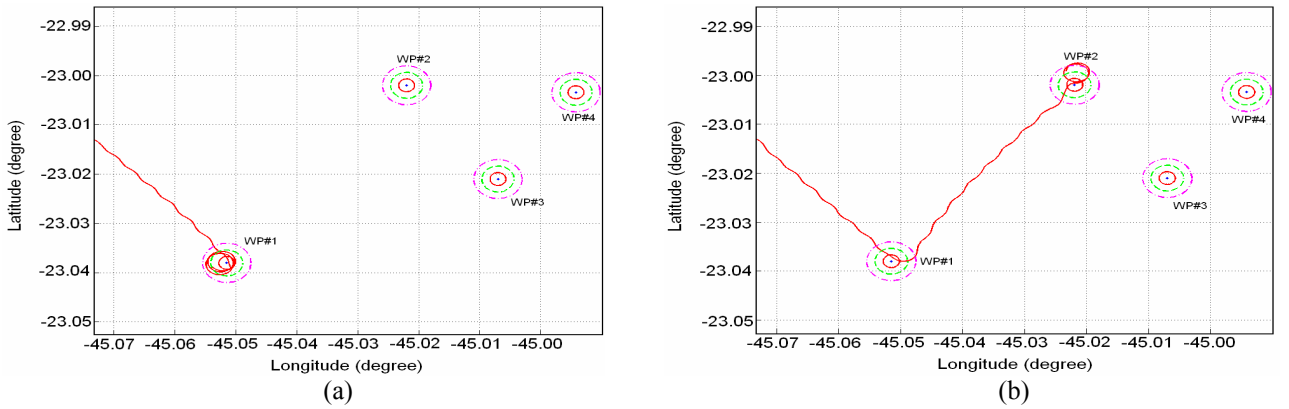
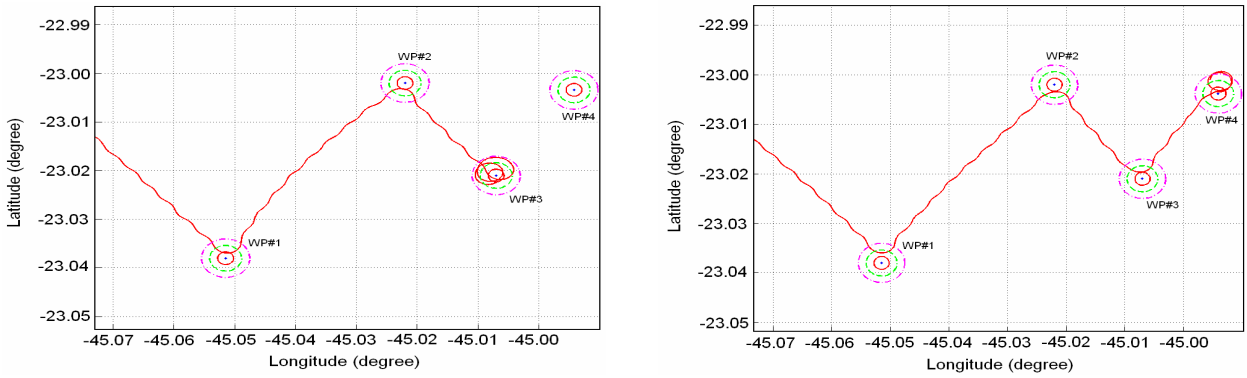


Figure 7. Autonomous flight test using the magnetic heading

Figure 8 presents the results obtained with the guidance law using the LOS angle and heading angle considering the body velocity vector. Four situations similar to those described before are presented. The AHRS of the aircraft used in the tests has 3 axis velocity components which X-Plane sends to MATLAB. With this data Matlab computes the body velocity vector using *u* and *v* velocities.





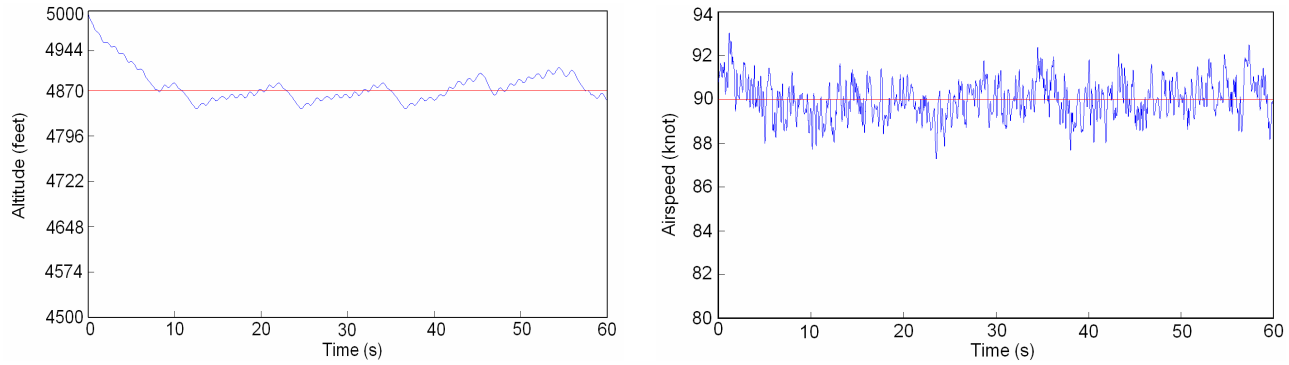
(c) (d)

Figure 8. Autonomous flight test using the magnetic heading and body velocity vector

Even though the tests results presented, Fig. 7 and Fig. 8, point to similar performance in both cases, one with the algorithm using the heading angle and the other using heading angle and velocity vector, there are some differences that should be considered. The use of the velocity vector improves the flight performance about the external perturbations in the aircraft. In other words, in a crosswind situation, the algorithm using the velocity vector will present a better flight performance than the algorithm using only heading angle.

On the other hand, the use of the body velocity vector in static (considering helicopters) or low velocity condition results in unstable flight control. In a case such as this, the use of heading angle is more appropriate.

The altitude and airspeed response for the two cases presented before, guidance algorithm using heading or velocity body information, were monitored. In both cases the aircraft response has followed the predefined values. Figure 9 (a) shows the altitude and Fig. 9 (b) shows the airspeed response when the altitude reference value, defined to all waypoints, was 4870 (ft) and the nominal forward speed was 90 (knot).



(a) (b)

Figure 9. (a) Altitude and (b) Airspeed of the aircraft during the fligh

Figure 10 shows the path flight of the aircraft when guided and controlled by the framework and algorithms discussed before. It shows the visualization of the path flight in the X-Plane environment.

(a) (b)

(c) (d)

Figure 10. Path flight visualized in the X-Plane environment.

8. CONCLUSION

An experimental framework for test and validation of guidance and control algorithms was implemented, tested and the results were presented. The simulation environment for the vehicle to be guided and controlled was the X-Plane Flight Simulator.

The proposed framework has shown to be a good set up for students and researchs to evaluate flight guidance and control algorithms to aircrafts, in particular to UAVs applications. As the framework was implemented it make possible to change the guidance algorithm or the control law without great efforts. Also, the Ground Station Software developed permits modify waypoint position, controle loop gains and the desired aircraft airspeed instantly.

The results shown the success of the guidance algorithm implemented, since the aircraft passed through the waypoints stablished. In the implemented algorithm two different informations can be used to actually guide the vehicle, nominally the heading angle and the body velocity vector. Both cases were implemented and presented good results. To explore the situations in which the use of each information in more adequate is the working in progress.

9. REFERENCES

Chowdhary, G., Ottander, J, Salaün, E. and Johson, E.N, 2009. “Low Cost Guidance, Navigation, and Control Solutions for a Miniature Air Vehicle in GPS Denied Environments”, Proceedings of 1st Symposium on Indoor Flight Issues, Mayaguez, Puerto Rico., pp. 12.

Eisenbeiss, H., 2004, “A Mini Unmanned Aerial Vehicle (UAV): System Overview and Image Acquisition”, Proceedings of International Workshop on Processing and Visualization Using High-Resolution Imagery, Pitsanulok, Thailand, pp. 10.

Garcia, R., and Barnes, L., 2010,“Real-time Implementation and Validation of a New Hierarchical Path Planning Scheme of UAVs via Hardware-in-the-Loop Simulation”. “Journal of Intelligent & Robotic Systems”. Vol. 57, Numbers 1-4, pp. 393-406.

Hattenberger, G., Alami, R. and Lacroix, L., 2006, “Planning And Control For Unmanned Air Vehicle Formation Flight”, Proceedings of International Conference on Intelligent Robots and Systems, Beijing, China, 18p.

Johnson, E.N., Fontaine, S.G., and Kahn, A.D., 2001, “Minimum Complexity Uninhabited Air Vehicle Guidance And Flight Control System”, Proceedings of 20th Digital Avionics Systems Conference, Daytona Beach, USA, Vol.1.

Meister, O., Frietsch, N., Seibold, J., and Trommer, G. F., 2008, “Software-in-the-Loop Simulation for Small Autonomous VTOL UAV with Teaming Capability”, Institute of Systems Optimization, Germany.

Gholkar, A., Isaacs, A. and Arya, H., 2004, “Hardware-In-Loop Simulator for Mini Aerial Vehicle”, Proceedings of the Sixth Real-Time Linux Workshop, Singapure, Singapure, pp. 1-9.

Jung, D., Ratti, J. and Tsiotras, P., 2009,“Real-time Implementation and Validation of a New Hierarchical Path Planning Scheme of UAVs via Hardware-in-the-Loop Simulation”. “Journal of Intelligent & Robotic Systems”. Volume 54, Numbers 1-3, pp. 163-181.

Lum,C.W., Rowland, M.L. and Rysdyk, R.T., 2008, “Human-in-the-Loop Distributed Simulation and Validation of Strategic Autonomous Algorithms”, Proceedings of the 26th Aerodynamic Measurement Technology and Ground Testing Conference, Seattle, USA, pp. 1-10.

Han, D., Kim, J., Lee, D., Cho, K. and Cho, S.,2008, “Autonomous Flight Test Using Angle Of UAV’s Velocity Vector”, Proceedings of the International Conference on Control, Automation and Systems, Seoul, Korea.

Lewis, D., McBride, J.Mikkelson,M. and Nguyen, H., 2005, “Design of an Autonomous Unmanned Aerial Reconnaissance Vehicle and Ground Control Station”, Proceedings of the 3rd Annual Student UAV Competition, Aberdeen, USA, pp. 20.

Roskam, J., 2001, “Airplane Flight Dynamics and Automatic Flight Controls” Design, Analysis and Research Corporation , Lawrence Roskam Aviation and Corporation, Part I.