An integrated framework for cooperative ground and aerial vehicle missions utilizing Matlab and X-Plane

Adriano Bittar, Nikolaos I. Vitzilaios, Matthew J. Rutherford, Kimon P. Valavanis
University of Denver Unmanned Systems Research Institute (DU2SRI)
Denver, CO, 80208, USA
Email: firstname.lastname@du.edu
www.du2sri.com

***Abstract*—This paper presents an integrated control framework for the simulation and visualization of cooperative missions for unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs). The X-Plane simulator is utilized to simulate vehicle dynamics and visualize experiments in realistic environments, whereas the control algorithms are executed and validated in Matlab\Simulink. A novel approach to integrate ground vehicles in X-Plane is presented and an overall open source framework is developed to facilitate the interaction and usability of the two software programs used. The framework facilitates research in cooperative vehicle control, path planning, formation control, and centralized control topologies through easy and cost effective system simulation, visualization and evaluation.**

I. Introduction

Unmanned Ground Vehicles (UGVs) and Unmanned Aerial Vehicles (UAVs) have been individually utilized to accomplish complex tasks in a variety of unmanned systems applications. Lately there has been an increasing interest in the cooperative use of both platforms in order to be used in even more complex applications [1], [2]. For example, a UGV can be used as a landing platform and\or as a refueling\recharging station for a UAV, increasing the range of action of the aerial vehicle [3]. On the other hand, a UAV can provide aerial vision to assist the UGV to perform its tasks more accurately [4].

Research in unmanned systems is characterized by high development cost, time restrictions and increased risk of failures during the first set of experiments. To expedite development and have a first quick assessment\evaluation of the designed system, various simulation tools are used and if the results are promising experimental tests follow. In the literature several frameworks can be found that provide tools for simulating a single vehicle [5]–[8], however little work has been done in simulating teams of unmanned vehicles of the same type [9], [10] and even less work in simulating cooperative missions of different types of unmanned vehicles [4]. The aim of this work is to develop a framework that allows the simulation of cooperative missions that use aerial and ground vehicles.

This paper presents a complete Software-in-the-Loop (SIL) simulation environment for UAVs and UGVs cooperative missions. For this purpose several tools have been developed for the simulation and evaluation of guidance algorithms, trajectory generation, path planning and formation structures, centralized or non-centralized control topologies focused in unmanned aerial and ground systems. The library provides Matlab\X-Plane communications, controls, guidance, trajectory generation, and mission planner blocks for UGVs, fixed-wing aircraft and rotorcraft. The aim is to simulate cooperative missions but each vehicle may be simulated independently too.

Another contribution of this architecture is the integration to the system of the X-Plane flight simulator, which is able to precisely reproduce aircraft dynamics applying the blade element theory, that provides accurate models for several types of vehicles [10]–[13] without the need of developing complex theoretical models for the aircraft [4]. X-Plane can also provide tools to realistically simulate weather conditions such as wind gusts and other forms of turbulence. X-plane has been successfully used for simulation of airplanes [12], [14], helicopters [10] and quadrotors [11], [15]. The novelty of this paper is the development and simulation of a ground vehicle in X-plane. By the reasonable assumption that a fixed-wing aircraft taxiing at very low speed behaves as a ground vehicle, a theoretical model is created and compared to the X-Plane model in order to validate the latter.

The paper is organized as follows: Section II presents the proposed architecture. The X-Plane to NEDs coordinate system and the vehicles model developed is presented in Section III. Section IV introduces the control, guidance and mission plan blocks. Results for a chemical leak response and a cellphone tower maintenance application are presented on Section V. Finally conclusions and future work are discussed in Section VI.

II. Architecture

The system consists of a skid steering differential ground vehicle and a small electric helicopter modeled and visualized by the X-Plane flight simulator. The UGV and UAV guidance and control blocks, and the mission planner block that manages both vehicles during a mission are implemented in Matlab, as presented in Fig.1. The design of the system has a modular approach where each block can be replaced by another more complex block provided that it has the same input/output ports.

Communication blocks exchange data with X-Plane and provide a fast plug-and-play integration. Using these blocks, the number of UGVs or UAVs can be increased (or decreased) during simulation up to the X-Plane limitation of 20 vehicles simulated in real time on a network based communication [14]. A cluster is created in which each machine provides the

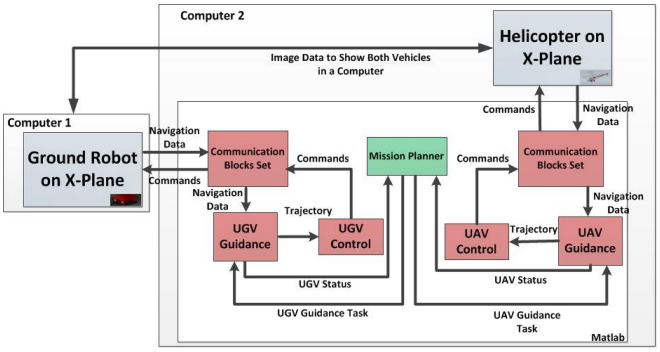

Fig. 1. The modularized Testbed architecture schematics

dynamics, kinematics and visualization of each vehicle. Also, the system allows simultaneous visualization of all the vehicles in only one monitor by exchanging visual data between the computers.

The system is developed in a way that hides the low-level network communication from the control engineer, which is to the concept of the Open Controller Platform (OCP) [16]. Therefore, if the same network configuration is kept, the SIL simulation can effortlessly be upgraded to a Hardware-in-the-Loop (HIL) simulation by implementing the Matlab blocks in a specific hardware, or by substituting the machines with sensors and actuators.

III. Model Development

A. Coordinate Frames

The scenery of X-Plane is based on local maps allowing the user to choose which maps suit their needs during the installation process. Due to this feature, there are two coordinate systems, world and local. The world coordinate system provides coordinates of the vehicle using earths latitude, longitude and elevation (mean sea level). Although world coordinates can be sufficiently used for fixed wing UAVs, the system does not present the necessary resolution for some applications as waypoint navigation for small unmanned rotorcraft. Consequently, for these applications, local coordinates are preferred. The local coordinate system provides the vehicles position based on the local map that the simulation takes place. The positions in X-Plane are given in OpenGL coordinates. OpenGL coordinates are extensively used in games and graphic simulation environments. On X-Plane this system follows the East-Up-South (EUS) convention. This means that from a reference point, the positive x-axis points east, the y-axis points up, and the z-axis points south. Nevertheless in aviation the North-East-Down (NED) convention is mostly used as the navigation frame, therefore coordinate transformations are required to present X-Plane data on the NED system. The transformation sequence from the X-Plane EUS system to the NED system, is presented in Fig.2. The following equations (1-4) provide the rotation matrix from X-Plane to NED system (where $c_\alpha = cos(\alpha)$ and $s_\alpha = sin(\alpha)$).

$$R_{NED}^{X-Plane(EUS)} = R_{x,\phi} \times R_{y,\phi} \tag{1}$$

$$R_{NED}^{X-Plane(EUS)} = \begin{bmatrix} c_\theta & s_\theta s_\phi & s_\theta c_\phi \\ 0 & c_\phi & -s_\phi \\ s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{2}$$

Since the position and velocities are the main concern on this paper, a coordinate frame transformation of $\theta = 90^\circ$ and $\phi = 90^\circ$ results in:

$$\begin{bmatrix} X^{X-Plane} \\ Y^{X-Plane} \\ Z^{X-Plane} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ -1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} X^{NED} \\ Y^{NED} \\ Z^{NED} \end{bmatrix} \tag{3}$$

$$\begin{bmatrix} X^{X-Plane} \\ Y^{X-Plane} \\ Z^{X-Plane} \end{bmatrix} = \begin{bmatrix} Y^{NED} \\ -Z^{NED} \\ X^{NED} \end{bmatrix} \tag{4}$$

Following the same approach to transform coordinates from the X-Plane system to the NED system, we get the following equation:

$$\begin{bmatrix} X^{NED} \\ Y^{NED} \\ Z^{NED} \end{bmatrix} = \begin{bmatrix} -Z^{X-Plane} \\ X^{X-Plane} \\ -Y^{X-Plane} \end{bmatrix} \tag{5}$$

As expected equations 4 and 5 are equivalent. These equations should substitute the equations derived in [13] since they implement a more accurate local frame.

B. Ground Vehicle Design

The ground vehicle developed in X-Plane (Fig.3), is based on a mobile landing platform developed by the DU2SRI team [3], that has the physical specifications presented in Table I. However, because of the limitations of the flight simulator, it is not possible to design an engine compatible to this ground vehicle. Therefore the platform is simulated using four jet engines with 35 N of thrust that allows a maximum velocity of 5 m/s.

Although X-Plane does understand the robot as an aircraft, the UGV does not have enough speed or surface to create lift and is treated as a fixed wing airplane taxiing on the runway (which is faithfully simulated by X-Plane [17]).

TABLE I
UGV PARAMETERS

Parameters	Value
Mass	12.247 kg
Distance between lateral Wheel (c)	44 cm
Distance between longitudinal Wheel (a)	49 cm
Radius of the Wheel (r)	7.3152 cm
Drag Coefficient (μ)	0.5

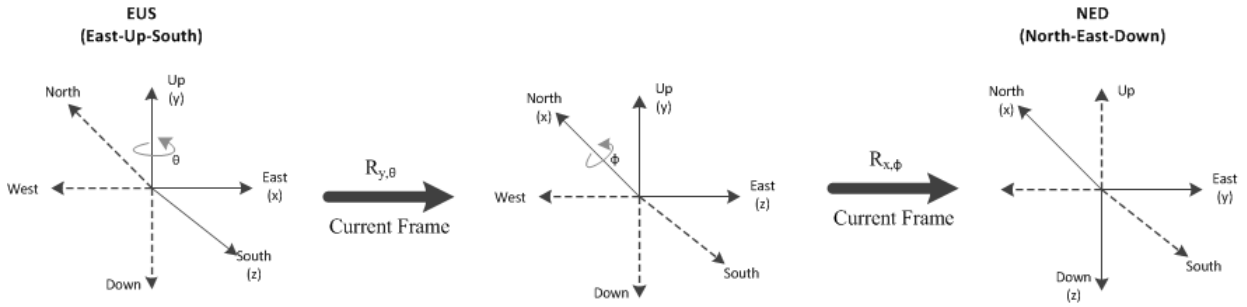

Fig. 2. X-Plane to NED Coordinate System

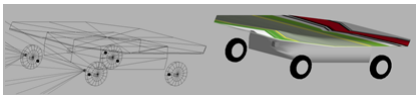

Fig. 3. Ground Vehicle with a landing Platform integrated

C. Ground Vehicle Model

A theoretical model is built and compared to the X-Plane model, in order to validate the behavior of the latter and guarantee that the simulator can provide a good estimation of the UGV states. The theoretical model was developed in the NED system in order to have the same frame as the UAV.

This also allows the model to be easily compared with X-Planes response. The equations were implemented in Matlab/Simulink, and modularized in three blocks: (1) the kinematics, which outputs the position of the platform on the NED system based on the inputted wheel velocities, (2) the dynamics, that calculates the angular wheel velocities accordingly with the forces generated by each engine, and (3) the approximation for X-plane engine model, as presented by Fig. 4.

Omitting the thickness of the wheel, the longitudinal and lateral slip, velocities are introduced as follows:

$$V_L = Vx_{fl} = Vx_{bl} = v_x^{body} + c\dot{\psi} \tag{6}$$

$$V_R = Vx_{fr} = Vx_{br} = v_x^{body} - c\dot{\psi} \tag{7}$$

$$V_F = V_B = Vy_{fr} = Vy_{fl} = Vy_{br} = Vy_{bl} \tag{8}$$

Where V_L is the resultant velocity of the left wheels, V_R is the resultant velocity of the right wheels, and V_F and V_B are the lateral velocities of the front and back wheels respectively.

Rewriting $V_i = r \times W_i$, being r the radius of the wheel, and solving (6) and (7) for Vx^{body} and $\dot{\psi}$:

$$v_x^{body} = \frac{r}{2}(W_L + W_R) \tag{9}$$

$$\dot{\psi} = \frac{r}{2c}(W_L - W_R) \tag{10}$$

Next, translating the body frame to the NED frame, the following velocities are obtained:

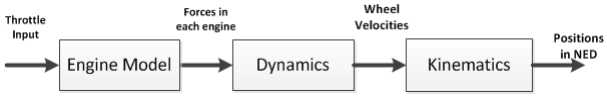

Fig. 4. UGV Block Model

$$v_x^{NED} = v_x^{body} \times \cos\psi \qquad v_y^{NED} = v_x^{body} \times \sin\psi \tag{11}$$

Applying (9) and (10), and integrating the velocities we arrive at the final form of the kinematics with the NED positions as functions of the angular velocity of the wheels, as shown by Fig. 5.

$$P_x^{NED} = \int v_x^{NED} dt = \int \frac{r}{2}(W_L + W_R)\cos\psi dt \tag{12}$$

$$P_y^{NED} = \int v_y^{NED} dt = \int \frac{r}{2}(W_L + W_R)\sin\psi dt \tag{13}$$

$$\psi = \int \dot{\psi} dt = \int \frac{r}{2}(W_L - W_R) dt \tag{14}$$

The dynamic forces acting on the wheel are shown in Fig. 2. The longitudinal force being F_i, and the lateral force being F_{li}. F_{si} is the drag force as a result of the movement.

Due to the symmetry in X and Y, related to the CoM (Center of Mass) of the UGV, the normal force in each individual wheel is equal to a quarter of the gravitational force ($Ni = mg/4$). The lateral force is zero, because the side slip is not being considered. Since that the platform is designed to only drive forward, the drag force can be approximated by [18]:

$$F_{si} = \mu N_i \times sig(v_{xi}^{body}) = \mu N_i = \mu \frac{mg}{4} \tag{15}$$

where μ is the dynamic drag coefficient between the tire and the airport pavement. Analyzing only the planar motion, the

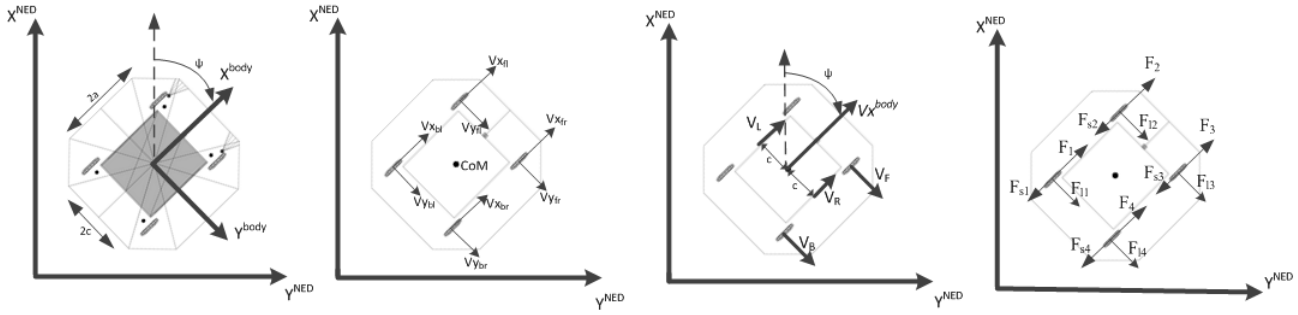

Fig. 5. Landing Platform Forces and Velocities

potential energy of the UGV is assumed to be zero, deriving the following Lagrangian of the system composed by the kinetics energy:

$$L(q,\dot{q}) = \frac{1}{2}m[(v_x^{NED})^2 + (v_y^{NED})^2] + \frac{1}{2}I\dot{\psi}^2 \tag{16}$$

Where $q = \begin{bmatrix} P_x^{NED} \\ P_y^{NED} \\ \psi \end{bmatrix}$, $\dot{q} = \begin{bmatrix} v_x^{NED} \\ v_y^{NED} \\ \dot{\psi} \end{bmatrix}$, $\ddot{q} = \begin{bmatrix} a_x^{NED} \\ a_y^{NED} \\ \ddot{\psi} \end{bmatrix}$.

First calculating the partial derivative and then its time-derivative, we arrive to the inertial forces as stated by 17:

$$\frac{d}{dt}\left(\frac{\partial E}{\partial \dot{q}}(q,\dot{q})\right) = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix}\ddot{q} = \mathbf{M}(\mathbf{q})\ddot{q} \tag{17}$$

Again omitting the longitudinal slip, the momentum caused by the drag forces is introduced as:

$$M_r(\dot{q}) = c\left[\sum_{i=1,2} F_{si}(v_x^{body} - \sum_{i=3,4} F_{si}(v_x^{body})\right] \tag{18}$$

Hence the drag induced vector can be expressed as:

$$R(\dot{q}) = \begin{bmatrix} \cos\phi \sum_{i=1}^{4} F_{si}(v_x^{body}) \\ \sin\phi \sum_{i=1}^{4} F_{si}(v_x^{body}) \\ M_r(\dot{q}) \end{bmatrix} \tag{19}$$

The forward forces generated by the robot engines in the NED frame are:

$$F_x = \cos\phi \sum_{i=1,2} F_i \qquad F_y = \sin\phi \sum_{i=1,2} F_i \tag{20}$$

And the resulting torque:

$$M = c(-F_1 - F_2 + F_3 + F_4) \tag{21}$$

The active vector is given by:

$$F = \begin{bmatrix} F_x & F_Y & M \end{bmatrix}^T \tag{22}$$

Imposing the forces produced by the engine as inputs of the dynamic model, using (17 - refForce) we get the following equation:

$$\mathbf{M}(\mathbf{q})\ddot{q} + R(\dot{q}) = F(q) \tag{23}$$

However, in order to have the dynamic model as function of the v_x^{body} and the heading rate, we introduce the following change of variables:

$$\dot{q} = S(q)\eta \tag{24}$$

where $S(q) = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ 0 & 1 \end{bmatrix}$ and $\eta = \begin{bmatrix} v_x^{body} \\ \dot{\psi} \end{bmatrix}$.

With respective derivative:

$$\ddot{q} = \dot{S}(q)^T eta + S(q)\dot{\eta} \tag{25}$$

Multiplying (28) by $S(q)T$ and using (24) and (25) the final dynamic system is as follows:

$$\bar{\mathbf{M}}\dot{\eta} + \bar{(}C)\eta + \bar{R} = S(q)^T F \tag{26}$$

where

$$\bar{M} = S(q)^T\mathbf{M}(\mathbf{q})S(q) = \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \tag{27}$$

$$\bar{C} = S(q)^T\mathbf{M}(\mathbf{q})S(q) = \begin{bmatrix} 0 & \dot{\psi} \\ -\dot{\psi} & 0 \end{bmatrix} \tag{28}$$

$$\bar{R} = S(q)^T R = \begin{bmatrix} \sum_{(} i = 1,2)F_{si}(v_x^{body} \\ M_r \end{bmatrix} \tag{29}$$

Finally to provide the angular velocity of the wheels as output of the dynamic block as presented by Figure 5, we use the angular form of equation (6) and (7):

$$W_L = \frac{1}{r}(v_x^{body} + c\psi) \qquad W_R = \frac{1}{r}(v_x^{body} - c\psi) \tag{30}$$

D. UGV Engine

X-Plane does not support engines compatible with a ground robot, because of that low bypass jet engines were used. Also the simulator only accepts throttle commands as input, instead of engine forces as the theoretical model derived. Therefore there is a need to model the jet engine used on the UGV. This was done by applying a sequence of throttle command to X-Plane and recording the thrust produced by the engine. Applying a quadratic fitting on the raw data results on the following equation:

$$F_{engine} = 5.5u^2 + 30u + 1.52 \tag{31}$$

where u is the normalized throttle input. Note that the performance of jet engines is dependent upon the altitude above sea level, therefore this equation is valid only for an altitude around of 1,5641,731 m (Denver, Colorado) [19].

E. UGV Validation

With the engine modeled, we have a theoretical model which accepts the same input as the X-Plane model and outputs the same states. Validation experiments have been carried out to compare the models using the same controller described in section IV. Fig. 6 shows the response of both vehicles under the use of the same position controller and waypoint track. The use of a controller instead of only using the same input was already applied in quadrotor [11], [13] and fixed wing [14]. This is done because controlling the model in a trajectory in X-Plane helps to reject environment disturbances. The same physical parameters presented in Table I were entered in both models, and a dynamic drag coefficient of 0.5 was set for X-Plane airport's pavement. With a maximum standard deviation of 0.29, it can be concluded that that X-Plane provides an accurate enough model for our application.

F. AERIAL VECHILE MODEL

The UAV is a Raptor 30 model RC helicopter comes with the X-Plane package. Since helicopters have already been successfully used in X-Plane [5], [10], [20], and the raptor model has been used for controller validation [21], this paper considers that X-plane is a representation of the true behavior of this UAV. However some modifications on the model were a critical for controller development and accomplishing the missions proposed.

- Set to zero all artificial stability provided by the simulator (equivalent to remove the gyros). The controller is responsible for stabilizing the helicopter.
- Increase the servo delay response, which is set to zero as an X-Plane default. It was used the same delay presented by the Futaba S3001 [22] .
- Night Vision Camera added.

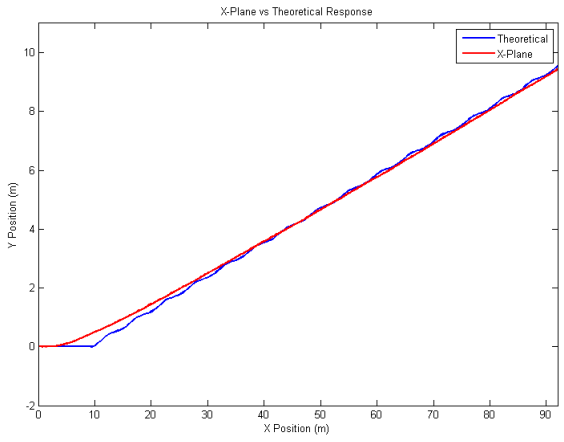

Fig. 6. X-Plane and Theoretical Model Responses

IV. CONTROL, GUIDANCE AND MISSION PLAN

The helicopter controller, inspired by the model free adaptive switching control scheme proposed by [23], was developed specially for this testbed. The X and Y position control, presented on Fig. 7, is composed of an outside position control loop in order to generate the body's velocities reference in X and Y, an intermediate velocity loop which generates the attitude control and an inner attitude loop which generates the elevator and ailerons commands. The altitude and heading control are based on the same methodology.

The control position of the landing platform also uses switching control to improve performance, see Fig. 8. However in this case since it is a planar control motion, the position control generates the reference for the forward velocity control, which is the resultant of the X and Y NED velocities. Using the line of sight algorithm, the set point for the heading control is generated [14].

The guidance block developed supports both rotary wings aircrafts and ground vehicles. It is able to generate track position for diverse types of trajectory including circle, waypoint using Linear segment with parabolic blend trajectory (LSPB), helix, landing, takeoff (the last three only applying to UAVs). This block receives a message from the mission plan with all the parameters of trajectory, such as radius of the circle, number of revolutions, climb rate, etc. Besides generating the trajectory, this block also monitors the execution, provides real time feedback to the mission plan, manages the vehicle when the trajectory is completed, and warns the mission plan that the objective has been completed. The block, implemented in Matlab/Simulink, was designed to work with events, similarly to a thread in a processor. This is done so that the code may be easily ported to a hardware, in case of upgrading the simulation to an HILS.

The mission plan block works as a manager of the entire mission. Its main functions are monitor and control the behavior of all the vehicles being simulated, besides taking the necessary action to fulfill the mission. This paper uses two mission plans. The first implemented was developed for a mission in where the helicopter provides aerial support for the UGV allowing an eye-in-the-sky perspective. While on the second implementation, the mission plan provides all the mechanisms necessary to complete a mission where the UGV gives landing support for the UAV, and the UAV is used to provide images of cellphone towers. However the entire simulation package developed is available free of charge for research and education purposes at:http://www.mathworks.com/matlabcentral/fileexchange/47516-x-plane-library-zip

V. APPLICATIONS

The main expectation for developing a simulation environment is to evaluate and validate the autonomous algorithms developed and to perform initial verification of the system integration before the final system is released to the industrial or academic partner [8]. Henceforward, two industry orientated applications are presented in this section.

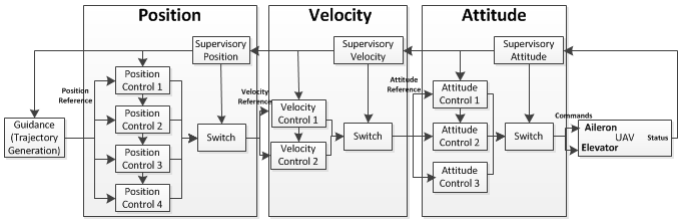

Fig. 7. Helicopter Control Block

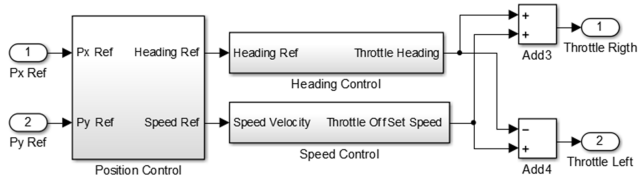

Fig. 8. Helicopter Control Block

A. *Factory Chemical Leak Response*

This application, inspired on one of the scenario presented by [24], evinces a situation that avoids danger to human life. A ground control station deploys, at a safe distance, a ground robot vehicle at location where a chemical leak has occurred. The ground robot must collect samples from chemical product at determined points (specified by the waypoints). A small helicopter follows the UGV, with 3 m of altitude (AGL) to provide support and on line aerial images of the region for use at the base station. This mission is executed prior a human intervention to evaluate the number of people and the necessary equipment.

Fig. 9 presents the trajectory performed by both vehicles, where the UGV collect samples in nine different locations and the UAV follows it. The mission is completely supervised by the mission plan algorithm which is used to manage the waypoint. It also ensures that the helicopter follows the UGV providing aerial image of affected region. A movie of this simulation can be found on: https://www.youtube.com/watch?v=ar6bBLnF7b8&feature=youtu.be

For this application warehouses and obstacles were included on DU2SRI test Airport in order to make the virtual environment more similar to a real factory.

B. *Cellphone tower Maintenance*

According to [25], maintenance of cellphone tower is also a very dangerous task. Only in 2013 13 workers were killed due to accidents which occurred during cellphone tower maintenance. If UAVs could be used instead of works for any component of this activity costs and risks could be greatly reduced. Based on this theory, the testbed presented is used to simulate a mission where the UAV streams real time video of the tower under inspection. As small electric UAVs have a limited time of operation due to short battery life, the UGV developed provides support, allowing the helicopter to land on the platform and recharge its batteries during the mission.

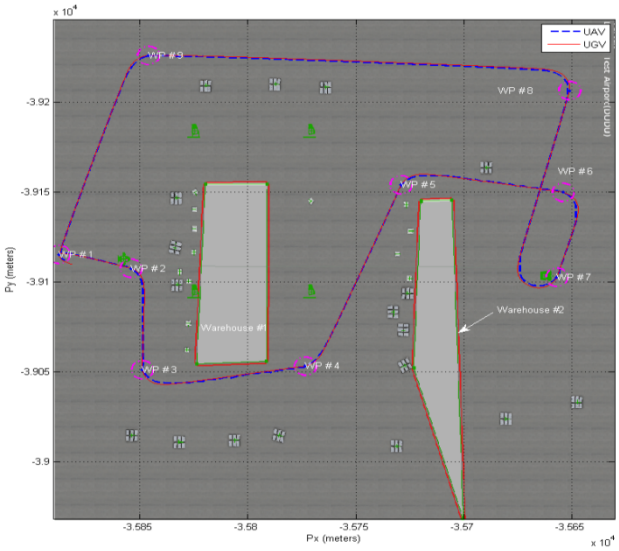

Fig. 9. UGV is followed by the UAV in a Chemical Leak Response Mission

The mission, executed completely autonomously, is managed by a different mission plan from the previous application. In this case the Helicopter takes off from the landing platform (UGV), and analyses the first cellphone tower, while the mission plan ensures the landing platform reaches a determined point between two towers. When the helicopter finishes the inspection of the first tower, it receives the platform's coordinates landing and heads to the location, landing when authorized by the mission plan. After the batteries are recharged the helicopter takes off heading to the second tower and the UGV steers to the initial point and waits for the helicopter to finish the inspection. The mission is finalized when the helicopter lands on the UGV. Fig. 9 presents the complete mission in a 2D aspect, while Fig. 11 and 12 presents the X-Plane environment with the trajectories performed. For

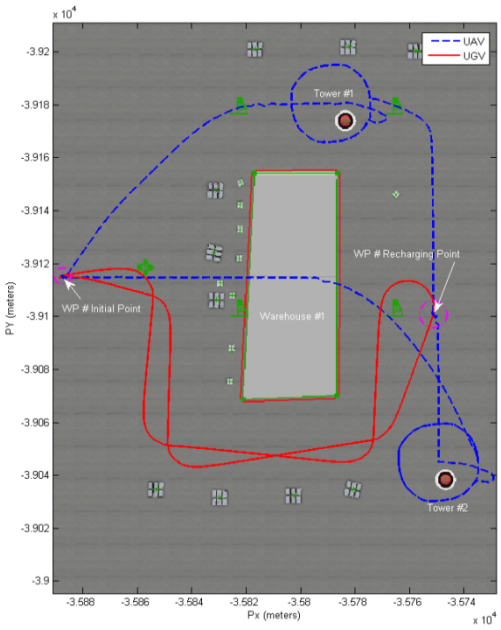

Fig. 10. Cellphone Tower Mission

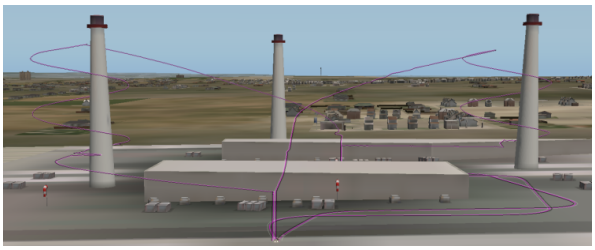

Fig. 11. X-Plane Print Screen

this application two tower were added to Airport.

Fig. 13 show the on-board camera of the helicopter during a mission performed in day light and one performed at night and presents a takeoff procedure.

VI. Conclusion

Within the capabilities of the testbed research in vehicle control, path planning, formation structures, and centralized control topologies to be easily and cost effectively explored. Being this environment classified as a SILS, algorithm can be evaluated before any attempt to use the system in the field. These tests help finding early failures, preventing crashes, and thereby minimizing costs.

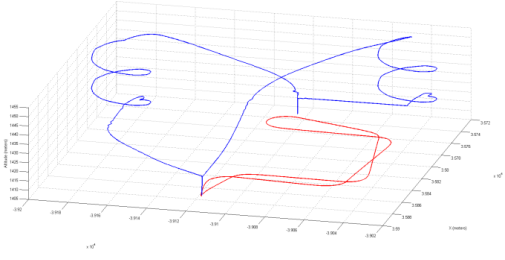

Fig. 12. 3D trajectory

Fig. 13. Take off Procedure

However the platform has some limitations. If the mission depends on a specific sensor that X-Plane can not provide information about it, the mission may not be completely simulated using the environment. As a example, since X-Plane does not provides sensors for collision and avoidance, such algorithm cannot be tested and when developing guidance blocks, prior information of the environment, as obstacles and terrain must be known.

Nevertheless, instead of developing a very detailed non-linear model, vehicle parameters can be imported directly to X-Plane. Therefore a fast development of algorithms for unmanned system is allowed using the tools presented to integrate X-Plane and Simulink and entire mission can be simulated.

Future work comprises on expanding the capabilities of the platform to use of quadrotor and fixed wing aircrafts working cooperatively in the simulation environment. Also the development of a linear quadratic tracking control for landing the UAV with the UVG in movement is being analyzed.

Acknowledgment

This research was partially supported by U.S. National Science Foundation (NFS) Grant CNS-1229236 and Brazilian National Research Council (CNPq), Process 245548/2012-2.

References

[1] C. Phan and H. H. Liu, "A cooperative UAV/UGV platform for wildfire detection and fighting," *2008 Asia Simul. Conf. - 7th Int. Conf. Syst. Simul. Sci. Comput.* , pp. 494–498, Ieee, Oct. 2008.

[2] T. L. P. M. S. Blumenthal, D. Holz and H. Surmann, "Teleoperated Visual Inspection and Surveillance with Unmanned Ground and Aerial Vehicles," *International Conference on Remote Engineering and Virtual Instrumentation.* , pp. 1–10, Nov. 2013.

[3] S. Conyers, "xxxxx," *xxxx* **xx**, p. xxx, Mar. 2014.

[4] D. E. Brech and R. C. Hoover, "Development of a Virtual Reality Simulation Testbed For Collaborative UGV And UAV Research Using Matlab," pp. 1–10, ASME 2013 International Mechanical Engineering Congress and Exposition, (San Diego, California, USA), 2013.

[5] R. D. Garcia and K. P. Valavanis, "The Implementation of an Autonomous Helicopter Testbed," *J. Intell. Robot. Syst.* **vol. 54 no. 14**, pp. 423–454, Aug. 2008.

[6] H. Shin, D. You, and D. H. Shim, "Autonomous Shipboard Landing Algorithm for Unmanned Helicopters in Crosswind," *J. Intell. Robot. Syst.* **vol. 74, no. 12**, pp. 347–361, Oct. 2013.

[7] C. M. Korpela, T. W. Danko, and P. Y. Oh, "MM-UAV: Mobile Manipulating Unmanned Aerial Vehicle," *J. Intell. Robot. Syst.* **vol. 65, no. 14**, pp. 93–101, Nov. 2011.

[8] Z. Wang, D. Song, J. Qi, J. Han, Y. Miao, and L. Meng, "A Full-functional Simulation and Test Platform for Rotorcraft Unmanned Aerial Vehicle Autonomous Control," *Robot Intelligence Technology and Applications* **vol. 208**, pp. 537–547, 2013.

[9] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP Multiple Micro-UAV Testbed," *IEEE Robot. Autom. Mag.* **17**, pp. 56–65, Sept. 2010.

[10] R. Garcia and L. Barnes, "Multi-UAV Simulator Utilizing X-Plane," *J. Intell. Robot. Syst.* **vol. 57, no. 14**, pp. 393–406, Oct. 2009.

[11] S. R. B. dos Santos, S. N. Givigi, and C. L. Nascimento, "Nonlinear tracking and aggressive maneuver controllers for quad-rotor robots using Learning Automata," *2012 IEEE Int. Syst. Conf. SysCon 2012* , pp. 1–8, Ieee, Mar. 2012.

[12] N. M. F. de Santos, Sergio R.B; Givigi Junior Sidney Nascimento; Nascimento Júnior, Cairo Lúcio; Bittar, Adriano; Oliveira, "Experimental Framework For Evaluation of Guidance," *21st Brazilian Congress of Mechanical Engineering, COBEM,* , Oct. 2011.

[13] H. Figueiredo, A. Bittar, and O. Saotome, "Platform for Quadrirotors: Analysis and applications," International Conference on Unmanned Aircraft Systems, May 2014.

[14] A. Bittar, H. V. Figuereido, P. A. Guimaraes, and A. C. Mendes, "Guidance Software-in-The-Loop Simulation Using X-Plane and Simulink for UAVs," International Conference on Unmanned Aircraft Systems, May 2014.

[15] M. Iskandarani, S. N. Givigi, C. A. Rabbath, and A. Beaulieu, "Linear Model Predictive Control for the encirclement of a target using a quadrotor aircraft," *21st Mediterr. Conf. Control Autom.* , pp. 1550–1556, June 2013.

[16] J. Paunicka, B. Mendel, and D. Corman, "The OCP - an open middleware solution for embedded systems," *Proc. 2001 Am. Control Conf. (Cat. No.01CH37148)* **5**, pp. 3445–3450, 2001.

[17] "FAA-Certified X-Plane," *Available: http://www. x-plane.com/pro/certified. Acessed on 10/12/2014* , Oct. 2014.

[18] K. R. Kozowski and D. A. Pazderski, "Modeling and Control af a 4-Wheel Skid-Steering Mobile Robot," *Int. J. Appl. Math. Compu. Sci* **vol.14, no. 4**, pp. 477–496, 2004.

[19] T. Benson, "Propulsion System Analysis," *Available: http://www.grc.nasa.gov/WWW/k-12/airplane/bgp.html. Acessed on 10/12/2014.* , Oct. 2014.

[20] D. Ernst, K. Valavanis, R. Garcia, and J. Craighead, "Unmanned Vehicle Controller Design, Evaluation and Implementation: From MATLAB to Printed Circuit Board," *J. Intell. Robot. Syst.* **49**, pp. 85–108, Mar. 2007.

[21] A. N. Khizer, D. Yaping, S. Amjad Ali, and X. Xiangyang, "Stable Hovering Flight for a Small Unmanned Helicopter Using Fuzzy Control," *Math. Probl. Eng.* **vol. 2014**, pp. 1–17, 2014.

[22] "S3003 Futaba Servo," *Available: http://www.futaba-rc.com/servos/analog.html. Acessed on 10/12/2014* , Oct. 2014.

[23] G. Battistelli, J. a. P. Hespanha, E. Mosca, L. Fellow, and P. Tesi, "Model-Free Adaptive Switching Control of Time-Varying Plants," *IEEE Transactions on Automatic Control* **vol. 58, no. 5**, pp. 1208–1220, 2013.

[24] C. Brune, T. Dityam, J. Girwar-Nath, K. Kanistras, G. Martins, A. Moses, I. Samonas, J. L. St. Amour, M. J. Rutherford, and K. P. Valavanis, "Enabling civilian applications of unmanned teams through collaboration, cooperation, and sensing," *Unmanned Systems Technology XIV,* **vol. 8387**, pp. 83870D–83870D–9, May 2012.

[25] L. H. Newman, "13 Cell Tower Maintenance Workers Died on the Job in 2013,"