Simulation of a UAV ground control station

Alain Ajami, Thibault Maillot, Nicolas Boizot, Jean-François Balmat, Jean-Paul Gauthier

9^{th} *International Conference of Modeling, Optimization and Simulation - MOSIM'12*
June 6-8, 2012 - Bordeaux - France
"Performance, interoperability and safety for sustainable development"

SIMULATION OF A UAV GROUND CONTROL STATION

A. AJAMI, T. MAILLOT, N. BOIZOT, J-F. BALMAT

Université du Sud Toulon Var
LSIS, UMR CNRS 7296
B.P 20132
83957 LA GARDE CEDEX - FRANCE
{ajami,maillot,boizot,balmat}@univ-tln.fr

J-P. GAUTHIER

Université du Sud Toulon Var
LSIS, UMR CNRS 7296
and
INRIA TEAM "GECO"
gauthier@univ-tln.fr

ABSTRACT: *In this article we present the development of a UAV ground control station simulator. We propose a module based description of the architecture of this simulator. We recall the nonlinear model of a fixed-wing aircraft. Finally, we outline ideas for improved path planning tasks. The approach is made clear through several diagrams, figures of the resulting station are displayed.*

KEYWORDS: *path planning, simulation, UAV, optimal control, inverse optimal control problem*

1 INTRODUCTION

Today's interest in drone technology led to an increasing number of dedicated projects (Tisdale, Kim & Hedric 2009, Kim, Shim & Sastry 2002, Fabiani, Fuertes, Piquereau, Mampey & Teichteil-Königsbuch 2007, Ippolito, Yeh & Campbell 2009). Those projects tackle problems such as the autonomy improvement, the reduction of drone crashes due to poor availability of information, the organization of drone swarms, or the calculation of secure and optimal flight paths. LSIS laboratory joins this research effort in the framework of project SHARE. Supported by a consortium of companies and research laboratories[1], the research aims to develop a universal and interoperable ground control station for fixed and rotary-wing UAVs for a reduced number of operators. Specifically, LSIS focuses on the connections between the UAV trajectory and its sensors. As a consequence, improved path planning algorithms that take into account payload requirements, optimal costs and obstacles (or no flight zones) avoidance are needed.

In order to test and demonstrate our techniques a ground control station simulator is required. The use of engineering and simulation softwares allows shorter development time and still maintain portability of the code. Since the development is done in close contact with our research partners, modularity is a key factor. Indeed, we want to be able to easily add, remove, or modify parts of the simulator. Another challenge is to determine the automation level of the station. (Cummings, Platts & Sulmistras 2006, Sheridan, Verplank & Brooks 1978) propose insights on automation strategies that are useful to describe the simulator. On the Sheridan-Verplank scale, the station ranks no more than 3 and, according to (Cummings et al. 2006), it has a level of interoperability of 4 with respect to the STANAG 4586 classification. This latter level can be described as follows: *the ground station allows the control and monitoring of the UAV at the exception of launch and recovery situations.* The simulation of such a device starts with the simulation of the aircraft's trajectory. Then follows the emulation of all the data exchanged between the operator and the UAV through the station, and finally of the control algorithms.

Classic tools for the implementation of the aircraft's dynamics, the controller part and the man machine interface are Matlab and Matlab/Simulink. We prefered the use of S-functions to a systematic block based implementation in order to ease portability in case of discard of Matlab's automatic code generation features. The simulation of the virtual environment and of the video flow is done by means of by a flight simulator (Craighead, Murphy, Burke & Goldiez 2007). Flightgear which is open source and interfaces nicely with Matlab/Simulink is used (Yang, Qi & Shan 2009, Sorton & Hammaker 2005).

The rest of this article is organised as follows. The structure of the simulator is presented in section 2. It is broken into several modules which functionalities are explained. The nonlinear aircraft dynamics are recalled in section 3. Although project SHARE aims at a ground control station that addresses both fixed and

[1] Opéra Ergonomie, ONERA, Thales Alénia Space, Eurocopter, Thales Systèmes Aéroportés.

rotative wing aircrafts, we focus on fixed-wing ones. Finally section 4 contains both low and high level descriptions of the control module (cf. figure 4). In particular, several notions regarding high level path planning and learning algorithms are sketched in subsection 4.2.

2 SIMULATOR

A ground control station simulation instance has two main parts: the initialization phase and the scenario realization. The initialization phase allows the definition of the parameters that are specific to a given simulation instance, this step is detailed in subsection 2.1. During the simulation several different missions can be realized accordingly to the operator's scenario. The architecture of this scenario realization part is detailed in subsection 2.2.

2.1 Simulator's Initialization Phase

We divide simulation parameters into two categories: setup parameters and core parameters. Core parameters contribute to the simulation inner mechanisms: model parameters (subsections 3.3.1, 3.3.2 and 3.4) and control parameters (section 4). As such the user cannot manipulate them directly: they are either loaded through a *setup assistant* utility or managed through a *modify/create assistant*.

The initialization procedure is schematized in figure 1 below. The user selects a *UAV type* between fixed and rotary-wing aircrafts which determines the equations of motion that are solved. The *UAV model* parameter allows the user to specify which particular aircraft is simulated. The aerodynamic model (subsection 3.3.2), the corresponding aerodynamic parameters and the level one controller coefficients are loaded accordingly. The controller coefficients can be further modified by the *level one controls efficiency* parameter (i.e. regular or degraded mode).

Several *payloads* are embedded on the UAV. They are selected out of a list. In particular camera parameters are specified (subsection 3.4). Noise levels and bias of the sensors are also defined.

Constraints are meant to define the type of high level path planning algorithm the simulator uses. They can be specified in the form of minimum/maximum time to next waypoint, energy consumption related constraints, stealth mode, line of sight preservation, or more technically, weighting parameters for the algorithms cost functions. This functionality is to be added during the final implementation stage.

Finally the set of *maps* of the area where the scenario takes place and the *initial position of the aircraft*[2] are selected. At the end of the initialization phase a *map synchronization utility* ensures the connection between the 3D virtual environment and the 2D maps of the area of interest.

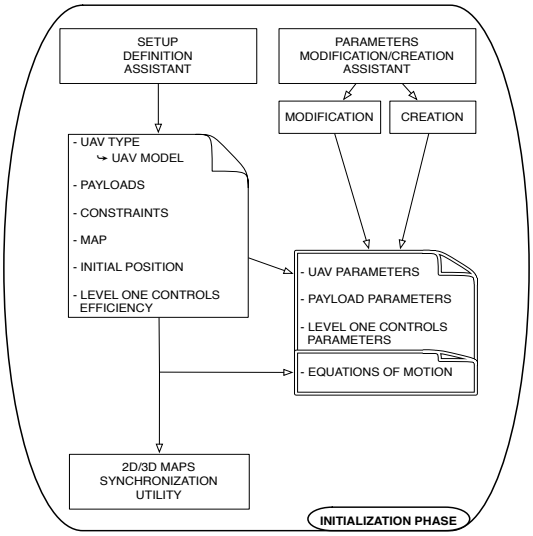

Figure 1: Functional diagram of the initialization phase

2.2 The Scenario

In this subsection we propose a module based presentation of the simulator. This approach is schematized in figure 2.

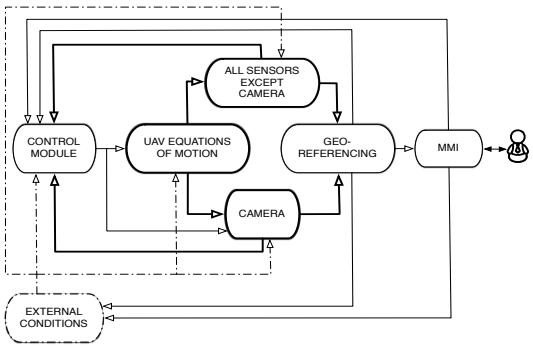

Figure 2: Module based architecture diagram

Let us start with the resolution of the *UAV equations of motion*. As said before, they are completely determined by the choices made in the initialization phase. Input and output vectors are defined in subsection 3.1 for a fixed-wing aircraft. Let us focus on the output of this module, namely the state vector. It is used to generate the sensors outputs which are lumped into two categories: *all sensors except the camera* and *the camera*. The state vector is actually fully available from the several sensors a UAV carries. However, we want to be able to simulate desynchronized sensors, noise levels and bias. The position which is available in the (x, y, z) coordinate system is transformed

[2] i.e. the origin of the world frame defined in subsection 3.2

into the WGS84 system for geo-localization purposes (Grewal, Weill & Andrews 2007). The UAV's attitude transforms from the unit quaternion representation to the Roll-Pitch-Yaw angles one. The camera has a special treatment since first, a set of differential equations has to be solved and second, the video flow is not generated in this module. We actually only take care of the line of sight orientation calculation and the evolution of extra camera parameters when they are used (subsection 3.4).

The outputs of the two sensors related modules are sent to the *geo-referencing module*. This module ensures the synchronization between the 2D map, the 3D environment and the radar representation. The 2D synchronization generates the data needed to display the trajectory of the aircraft on a map, to represent the video cone and the targets whose positions are known to the ground station. The radar shows the relative positions of possible targets. Finally the 3D synchronization is the computation of the actual video flow (i.e. Flightgear). Those data are sent to the *man-machine interface* (MMI) module for display (figure 3). This module is detailed further below in the present subsection.

All the modules described before feed the *control module* which in return sends its data to the *equations of motion* and *camera* modules. The control module is the object of section 4.

The *exterior conditions* module is the last part of this diagram, it encapsulates all the other ones since perturbations can be introduced in all the other blocks. At this stage of the development we only use it to manage weather conditions, more particularly wind. A Von Karman model is used to generate the wind velocity vector required by the resolution of the equations of motion. In our opinion, at least two extra modules shall be considered. We didn't add them in the diagram for clarity purposes and since they are at early development stages. Those are the *extra payloads management* (i.e. sensors specific to a mission, packages needed to be dropped, etc.) and the *targets management unit*.

We end this subsection with a few words on the *man-machine interface* module. Figure 3 gives an account of how data flows from the operator to the UAV and back. Whereas the information sent back to the operator talk for themselves, it is not the case of all the features managed by the operator. *UAV control modes* and *payloads control modes* are given in the next section. *Simulation management features* are: 1) start, 2) pause, 3) stop and 4) visualization options. As for the rest, it is pretty obvious except for the *mission type management* part. This task is performed by the MMI modules which triggers the adequate module functionalities accordingly to the operator's will. Let us focus on possible missions. The UAV has

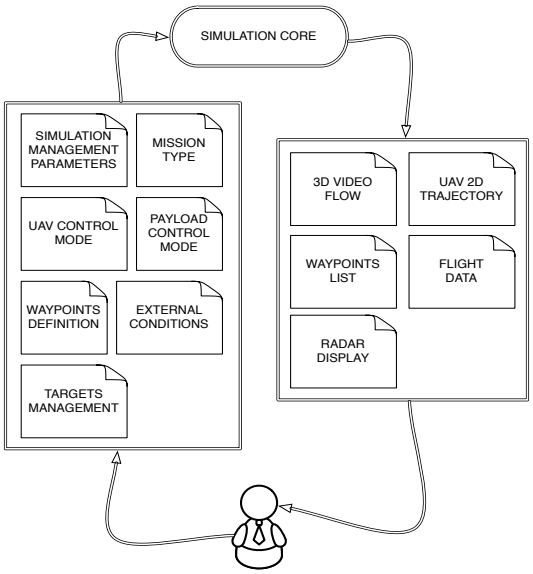

Figure 3: Man-machine interface data flow diagram

to perform several types of mission (NATO 2008) of which we implemented the following ones.

Follow flight plan This is the basic mission. A flight plan is a series of waypoints the UAV has to visit. The flight plan is either defined before the scenario or made by the operator as the scenario undergoes.

Patterns This mission consists in following operator parametrized patterns: circle, racetrack, and figure eight (table 1) in order to detect any activity in a specific area and/or collect information. The trajectory that leads to a given pattern is the subject of more or less sophisticated algorithms (section 4). Indeed the UAV has to be tangent when it enters the pattern. The approach trajectory to the pattern is constrained by the geographical configuration, UAV physical characteristics, UAV initial orientation, and the traveling direction of the pattern.

Tracking The objective is to follow a friend or enemy point of interest. In the case of an ally, an expanded surveillance zone is considered. Multiple situations can be encountered: the target is in a no-fly zone, target occlusion occurs, the target is moving, etc. For this mission camera controls are either automated or manually activated by the operator. In the case of an enemy target, it is of utmost importance to be able to maintain it in the camera field of view.

Replanification Replanification happens when the UAV needs a waypoints list that is better generated by appropriate algorithms – i.e. when a mission has to be changed because of unexpected

pattern	parameters list
common to all	center point rotation direction
circle	radius
racetrack / hippodrome	small radius large radius orientation
figure eight	pseudo radius orientation

Table 1: Pattern parametrization

events. No-fly zone avoidance, search of an admissible path and optimization of an optimality criterion are the three aspects that drive this mission type (subsection 4.2).

3 MODELING

3.1 Notations and Definitions

$v^{(f)}$	vector v projected in frame (f)
Id_n	$(n \times n)$ dimensional identity matrix
$X^{(f)}$	aircraft's position vector
$V^{(f)}$	aircraft's speed vector
$V_{ae}^{(f)}$	aerodynamic speed vector
q	unit attitude quaternion
$\left[(\tilde{q})^{\times}\right]$	matrix associated to $v \mapsto \tilde{q} \times v$
$\phi^{(w)}$ (resp. ϕ_c)	aircraft's (resp. camera's) roll angle
$\theta^{(w)}$ (resp. θ_c)	aircraft's (resp. camera's) pitch angle
$\psi^{(w)}$ (resp. ψ_c)	aircraft's (resp. camera's) yaw angle
$\omega^{(f)}$	aircraft's angular velocity vector
$(x_c, y_c, z_c)^{(w)}$	camera's position
α	aerodynamic angle of attack
β	aerodynamic sideslip angle
$F_{ae}^{(b)}$	aerodynamic force
$F_{pr}^{(b)}$	propulsive force
$M_{ae,g}^{(b)}$	aerodynamic moment at point g
$M_{pr,g}^{(b)}$	propulsive moment at point g
$G^{(f)}$	gravity vector (i.e. $G^{(w)} = (0, 0, g)$)
I	aircraft's inertia matrix
δ_l	roll control
δ_m	pitch control
δ_n	yaw control
δ_x	thrust control
T	overall thrust
δ_ϕ	$\phi_c^{(b)}$ set point
δ_θ	$\theta_c^{(b)}$ set point
δ_ψ	$\psi_c^{(b)}$ set point
$u(t)$ (resp. $u_c(t)$)	UAV (resp. camera) control vectors
ρ	atmospheric density
ρ_0	atmospheric density at sea level
S	aircraft's surface of reference
l	aircraft's length of reference
m	mass of the aircraft

The controls are

$$u(t) = (\delta_l(t), \delta_m(t), \delta_n(t), T(t))^T$$

and

$$u_c(t) = (\delta_\phi(t), \delta_\theta(t), \delta_\psi(t))^T$$

The state vector is

$$\begin{cases} X^{(w)} &= \left(X_x^{(w)}, X_y^{(w)}, X_z^{(w)}\right)^T \\ V^{(b)} &= \left(V_x^{(b)}, V_y^{(b)}, V_z^{(b)}\right)^T \\ q &= (q_0, \tilde{q})^T = (q_0, q_1, q_2, q_3)^T \\ \omega^{(b)} &= \left(\omega_1^{(b)}, \omega_2^{(b)}, \omega_3^{(b)}\right)^T \end{cases}$$

The matrices $\left[(\tilde{q})^{\times}\right]$ and I are defined as

$$\left[(\tilde{q})^{\times}\right] = \begin{pmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{pmatrix}$$

and $I = I^T = \left(I_{(i,j)\in\{1,2,3\}^2}\right)$

3.2 Hypotheses and Frames

The nonlinear model for aircraft dynamics proposed here is inspired by (Boiffier 1998, Wanner 1984, Junkins & Schaub 2002). The modeling hypotheses are[3]

1. earth is flat and motionless with respect to the simulation duration,
2. gravity is constant (i.e. the center of mass is the same as the center of gravity),
3. atmospheric pressure and temperature are constant (i.e. only density variation with respect to height is considered),
4. there is a uniform wind velocity field,
5. the aircraft is a 6 DoF solid which fuselage has a symmetry plane,
6. the propulsion system is on the fuselage axis,
7. the mass and the inertia matrix of the aircraft are constant parameters.

The first hypothesis makes the world frame inertial, thus we only need three frames to derive the equations.

World frame (w) This frame is attached to a reference point O, the Ox, Oy, Oz axes are oriented in the north-east-down directions. Although the

[3]See (Boiffier 1998) for a comprehensive treatment of the hypotheses.

aircraft's position expressed through the equations of motion is given in the (x, y, z) system, we need to express it in the WGS84 system in order to perform geo-referencing (Grewal et al. 2007).

Body frame (b) This frame is attached to the aircraft's center of mass (G). The $Gx^{(b)}$ axis goes along the fuselage axis. The $Gz^{(b)}$ axis is taken in the plane of symmetry of the aircraft and points downward, and $Oy^{(b)} = Oz^{(b)} \times Ox^{(b)}$
The composition of a translation and a rotation transforms frame (w) into frame (b). The rotation defines the attitude of the aircraft. We use the Euler angles *Roll-Pitch-Yaw* parametrization for its physical meaning, and the unit quaternion parametrization to solve differential equations (Junkins & Schaub 2002, Diebel 2006).

Aerodynamic frame (a) This frame shares his origin with frame (b). The $Gx^{(a)}$ axis is carried by the aerodynamic velocity vector. The rotation that transforms $Gx^{(b)}$ into $Gx^{(a)}$ is parametrized by the angle of attack and the sideslip angle. The other axes are obtained through this rotation (subsection 3.3.2).

3.3 Aircraft Dynamics

3.3.1 Equations of Motion

The dynamic resultant theorem and the dynamic moment theorem write as

$$m\frac{d}{dt}\dot{V}^{(b)} = F_{ae}^{(b)} + F_{pr}^{(b)} + mG^{(b)} - \omega^{(b)} \times mV^{(b)} \tag{1}$$

$$I\dot{\omega}^{(b)} = M_{ae,g}^{(b)} + M_{pr,g}^{(b)} - \omega^{(b)} \times I\omega^{(b)} \tag{2}$$

where

$$F_{ae}^{(b)} = \frac{\rho S V_{ae}^2}{2}\begin{pmatrix} -C_A^{(b)} \\ C_Y^{(b)} \\ -C_N^{(b)} \end{pmatrix} \quad F_{pr}^{(b)} = \begin{pmatrix} F_{pr,x}^{(b)} \\ F_{pr,y}^{(b)} \\ F_{pr,z}^{(b)} \end{pmatrix} \tag{3}$$

$$M_{ae}^{(b)} = \frac{\rho S l V_{ae}^2}{2}\begin{pmatrix} C_l \\ C_m \\ C_n \end{pmatrix} \quad M_{pr}^{(b)} = \begin{pmatrix} M_x^{(b)} \\ M_y^{(b)} \\ M_z^{(b)} \end{pmatrix} \tag{4}$$

The rotation matrix from the world frame to the body frame expressed in terms of q, and the dynamics of q are (Junkins & Schaub 2002)

$$R(q) = \left(q_0^2 - \tilde{q}\tilde{q}^T\right) Id_3 + 2\left(\tilde{q}^T\tilde{q} - q_0\left[\tilde{q}^\times\right]\right) \tag{5}$$

and

$$\dot{q} = \frac{1}{2}\begin{pmatrix} 0 & -\omega^{(b)} \\ \omega^{(b)} & -\left[\left(\omega^{(b)}\right)^\times\right] \end{pmatrix} q \tag{6}$$

Therefore we have

$$mG^{(b)} = R(q)\,(0, 0, mg)^T \tag{7}$$

$$\dot{X}^{(w)} = V^{(w)} = R^T(q)V^{(b)} \tag{8}$$

The equations of motion of the aircraft with state $\left(X^{(w)}, V^{(b)}, q, \omega^{(b)}\right)$ are equations (8), (1), (6) and (2). In order to actually use them, we still need to define a few elements.

- The atmospheric density model is

$$\rho = \rho_0 e^{-1.1210^{-4}h} = \rho_0 e^{1.1210^{-4}z}$$

- From hypothesis (5), elements $I_{1,2}$ and $I_{2,3}$ of the inertial matrix are null (the symmetry of the inertial matrix implies $I_{2,1} = I_{3,2} = 0$).

- From hypothesis (6), the propulsion force vector components $F_{pr,y}^{(b)}$ and $F_{pr,z}^{(b)}$ are null. The propulsion moment vanishes – see (4.3.4) in (Boiffier 1998), with $\alpha_m = \beta_m = 0$.
An engine efficiency model can be used for the modulus of the thrust (i.e. $F_{pr,x}^{(b)}$ in our case). For example, in (Boiffier 1998) $T = k_m \rho V_{ae}^{\lambda} \delta_x$ is proposed. We simply used $T = k_m \delta_x$, with $0 < k_m \leq 1$.

- The wind is characterized by its velocity vector $V_w^{(b)}$. The aerodynamic speed vector is then defined as $V_{ae}^{(b)} = V_w^{(b)} - V^{(b)}$, and $V_{ae}^2 = \left(V_{ae}^{(b)}\right)^T V_{ae}^{(b)}$

3.3.2 Aerodynamic Model

We call aerodynamic model the equations used to fully express aerodynamic forces and moments (equation 4). Force coefficients are first defined in frame (a) (i.e. from wind tunnel experiments), then rotated into frame (b). As such the angle of attack and the sideslip angle are needed. Starting from the aerodynamic speed vector we derive the equations

$$\begin{aligned} V_{ae,x}^{(b)} &= V_{ae}\cos(\alpha)\cos(\beta) \\ V_{ae,y}^{(b)} &= V_{ae}\sin(\beta) \\ V_{ae,z}^{(b)} &= V_{ae}\sin(\alpha)\cos(\beta) \end{aligned}$$

and we write

$$\begin{pmatrix} C_A^{(b)} \\ C_Y^{(b)} \\ C_N^{(b)} \end{pmatrix} = \begin{pmatrix} \mathbf{C_x^{(a)}}\cos(\alpha)\cos(\beta) + \mathbf{C_y^{(a)}}\sin(\beta)\cos(\alpha) - \mathbf{C_z^{(a)}}\sin(\alpha) \\ \mathbf{C_y^{(a)}}\cos(\beta) - \mathbf{C_x^{(a)}}\sin(\beta) \\ \mathbf{C_x^{(a)}}\sin(\alpha)\cos(\beta) + \mathbf{C_y^{(a)}}\sin(\beta)\sin(\alpha) + \mathbf{C_z^{(a)}}\cos(\alpha) \end{pmatrix}$$

There are many possible models for $C_x^{(a)}$, $C_y^{(a)}$, $C_z^{(a)}$, C_l, C_m and C_n (Wanner 1984, Schmidt 1998). We cite the simple model

$$\begin{aligned}
C_x^{(a)} &= C_{x,0} + k_i C_z^2 \\
C_y^{(a)} &= C_{y,\beta}\beta + C_{y,\delta_n}\delta_n \\
C_z^{(a)} &= C_{z,\alpha}\alpha + C_{z,\delta_m}\delta_m \\
C_l &= C_{l,\beta}\sin(\beta) + l\left(C_{l,\omega_1}\frac{\omega_1}{V_{ae}} + C_{l,\omega_3}\frac{\omega_3}{V_{ae}}\right) \\
&\quad + C_{l,\delta_l}\delta_l + C_{l,\delta_n}\delta_l \\
C_m &= C_{m,\alpha} + C_{m,\delta_m}\delta_m + C_{m,\omega_2}\frac{l\omega_2}{V_{ae}} \\
C_n &= C_{n,\beta} + C_{n,\delta_n}\delta_n + C_{n,\omega_3}\frac{\omega_3}{V_{ae}}
\end{aligned}$$

3.4 Camera Dynamics

This model tracks the orientation of the line of sight of the camera. This latter is assumed to be solidly attached to the center of mass of the aircraft:

$$\begin{aligned}
x_c^w &= X_x^{(w)} \\
y_c^w &= X_y^{(w)} \\
z_c^w &= X_z^{(w)}
\end{aligned}$$

Following camera datasheet specifications (wescam 2011), the dynamics can be approximated as a first order response to an attitude set point. This attitude set point is given in the (b) frame (i.e. $\dot{\phi}_c^{(b)} = (-\phi_c^{(b)} + \phi_{set}^{(b)})/\tau$). The time constant τ defines the speed of the camera steering system. Constraints on the coverage can be expressed in terms of constraints on the inputs rather than on the output (a significant asset both for implementation and optimal control purposes).

$$\left\{\begin{array}{rcl}
\dot{\phi}_c^{(w)} &=& \omega_1^{(b)} - \frac{1}{\tau}\left(\phi_c^{(w)} - \phi^{(w)}\right) + \frac{1}{\tau}\delta_\phi \\
\dot{\theta}_c^{(w)} &=& \omega_2^{(b)} - \frac{1}{\tau}\left(\theta_c^{(w)} - \theta^{(w)}\right) + \frac{1}{\tau}\delta_\theta \\
\dot{\psi}_c^{(w)} &=& \omega_3^{(b)} - \frac{1}{\tau}\left(\psi_c^{(w)} - \psi^{(w)}\right) + \frac{1}{\tau}\delta_\psi
\end{array}\right. \quad (9)$$

Extra parameters that can be used to model the camera system are described in the following table (wescam 2011).

parameter	indicative value
optical eye field of view	36 to 1 (°)
IR fields of view	30, 7 and 1.8 (°)
frequency	24 (FPS)
turret coverage	
azimuth	0 to 360 (°)
elevation	90 to -120(°)
roll	na
turret steering speed	0 to 60 (°/sec)
turret stabilization quality	abt 3.10^{-3} (°)
autofocus time constant	0
	(i.e. instantaneous)

4 CONTROL MODULE

As explained in (Cummings et al. 2006) both the level decomposition and strategies of automation are crucial elements in order to achieve efficient, accurate, and safe UAV operations. First, we give the global picture of the control module in subsection 4.1, and then focus on two high-level aspects in subsection 4.2. Those two important topics are path planning and learning techniques.

4.1 General Considerations

Three levels are considered for this module (figure 4). Level zero corresponds to a direct access to the control surfaces of the UAV. Although unrealistic, we kept the possibility to directly manipulate control surfaces as far as the simulator only is concerned.
The level one controller input signal is of the form *heading, speed, altitude*. Output is a $(\delta_l, \delta_m, \delta_n, \delta_x)$ vector (section 3). This controller consists in a stability augmentation system (yaw, pitch, and phugoid dampers) in series with a basic PID-like autopilot system (heading, airspeed and altitude holders). Note that the complexity of this controller grows with the complexity of the aerodynamic model (recall that the controller parameters are loaded accordingly to the UAV model, cf. section 2).
The level two controller consist of encapsulates all the algorithms that generate *heading, speed, altitude* set points.

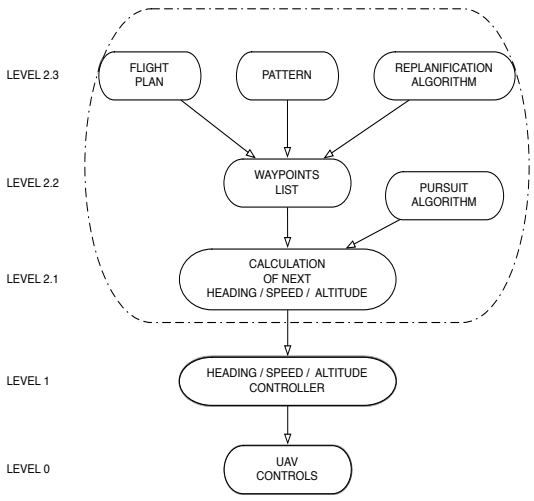

Figure 4: Level based decomposition of the controller module

The operator has the choice between several different piloting modes (figure 3) for both the UAV and the camera turret.

Manual mode Through a joystick, the *heading, speed, altitude* setpoint is directly controlled. As mentioned before, a very special option, only

available in simulation, grants access to the control surfaces of the UAV.

Semi-manual mode At least one of the three components of the *heading, speed, altitude* setpoint is managed by the system – e.g. altitude is kept fixed.

Automatic mode The level one set point is defined by a level two algorithm which depends on the active mission.

The control modes of the camera are simpler to handle as only manual and automatic modes are proposed. Knowing the lastly visited, the targeted and the next on the list waypoints, the calculation of the new *heading, speed, altitude* set point is easily obtained from geometry. Note that a waypoint is defined by its WGS84 coordinates and an optional timestamp. When minimum/maximum time between waypoints constraints are activated, timestamps are used to adjust the set speed.

The next three parts to explain are the *pattern, replanification and pursuit* algorithms. As said before a pattern is defined by a set of parameters (table 1) as such a waypoints list describes it. What remains to be done is to propose a trajectory that allows the UAV to reach the pattern with the correct configuration (i.e. tangentially), starting from any initial point and any initial orientation. This is one of the jobs done by the *replanification* algorithm. The second one is to generate optimal trajectory to travel through an area without having a pre-established flight plan. A strategy in order to use this algorithm is to solve a replanification problem, then propose the solution trajectory as a list of waypoints and finally refresh the list through a new calculation each time a waypoint is reached. The *pursuit algorithm* output can be seen as a *heading, altitude, speed* vector that is recomputed whenever necessary.

A few insights on replanification algorithms are proposed in the next subsection. We even make one step farther by presenting experiment based learning of optimal cost functions.

4.2 planning and Learning

4.2.1 Introduction

In the context of the project SHARE we have also to achieve two extra tasks:

1. Fill in the modules called *replanification algorithm* and *pursuit algorithm* in figure 4. There is a lot of bibliography on this topic (Betts 2001, Bullo & Lewis 2004, Fliess, Lévine, Martin & Rouchon 1995, Kim et al. 2002, Laumond 1998, LaValle 2006, Park, Deyst & How 2004, Van-Nieuwstadt & Murray 1998) , non exhaustively. These classical algorithms are based upon very different approaches such as geometric control, optimal control, flatness. In fact we are also developing our own methods that we present briefly below (subsection 4.2.2).

2. An idea to develop planification/replanification methods is to learn from the behavior of experimented pilots. We do this here for HALE (High Altitude Long Endurance) drones. In subsection 4.2.3 we present briefly our ideas, that are inspired from the beautiful work of Jean and al. in the papers (Chitour, Jean & Mason 2012, Chittaro, Jean & Mason to appear). In these papers, they attack the problem of identification of the cost minimized in human locomotion. Our problem is very similar since HALE drones behave kinematically more or less as a human being moving on a plane (constant altitude, constant speed). There is a lot of other methods dedicated to this human locomotion problem, see for instance (Li, Todorov & Liu 2011, Berret, Darlot, Jean, Pozzo, Papaxanthis & Gauthier 2008).

4.2.2 Planification/Replanification

In general the mission of a drone is planed in advance and specified by a certain number of checkpoints and a certain number of patterns (line, circle, figure eight and hippodrome). There is the need of on-line replanification methods when the mission is interrupted and the drone has to:

- Join a fixed target and turn around following a certain pattern,
- Join a moving target and follow it. This is not an obvious problem when the minimum speed of the drone is higher than the target speed.

Regarding the case (not yet treated above) of HALE rotorcraft-based drones, they behave kinematically more or less as the classical "simple car" model (Laumond 1998). Hence one can formulate an optimal control problem which looks like a left-invariant subriemannian problem over the group of motions of the plane. On this topic there is the beautiful complete mathematical work of Yuri Sachkov (Moiseev & Sachkov 2010, Sachkov 2010, Sachkov 2011) that does the job, if no obstacles are taken into account. If obstacles occur, the Sachkov method can be coupled to a method for finding first non admissible trajectories avoiding obstacles and approximating them by pieces from the Sachkov synthesis. Several methods are available to find such non admissible trajectories,

see (LaValle 2006) for an extensive overview of such techniques.

The case of fixed-wing drones is different. If we consider a simple model of Dubins type (Laumond 1998), the problem can be stated as a repetition of minimum-time problems for the Dubins car, where the final target is a (non-oriented or oriented) pattern. This optimal control problem can be solved easily, but it leads to a non-smooth and eventually non-continuous optimal synthesis. In fact, a nice smooth optimal synthesis can be found, leading to trajectories shown in figure 5. Details on this optimal synthesis and proof of the stability and convergence, which relies on Lyapunov theory methods and the LaSalle invariance principle, will be given in a forthcoming paper.

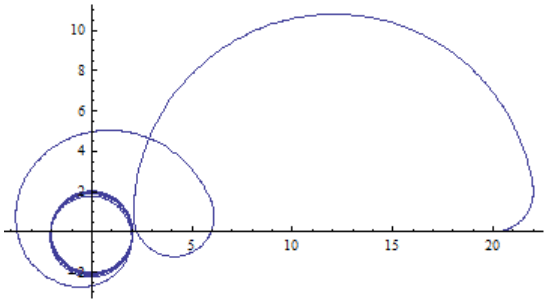

Figure 5: Trajectory resulting of smooth optimal synthesis starting at the point (20;0) with a direction of 0 radian and arriving tangent to a pattern, here the circle centering at (0;0)

4.2.3 Learning from Experimented Pilots

When the planification problems are specified in terms of optimal control problems, the choice of the cost to be minimized can be made from the experience of experimented pilots. For our fixed wings HALE drones, as we said, the basic kinematic model is the classical Dubins one. It has been shown by Jean and al. (Chitour et al. 2012, Chittaro et al. to appear) that the cost to be minimized for human locomotion is an integral length-curvature compromise. We completed their work by developing a program allowing to identify this cost. Moreover, we proved the following general theoretical result: for a generic model of motion, the integral cost can be recovered from three experiments, provided that these experiments have one common value of the control. However, in the case of the Dubins car model, witch is kinetically equivalent to a fixed-wing UAV flying at constant altitude and constant speed, only two such experiments are needed. These developments will be the purpose of another paper. On the figure 6, we show the reconstruction of the cost as a function of length and curvature from two experiments (i.e. in the case of a Dubins car model of motion).

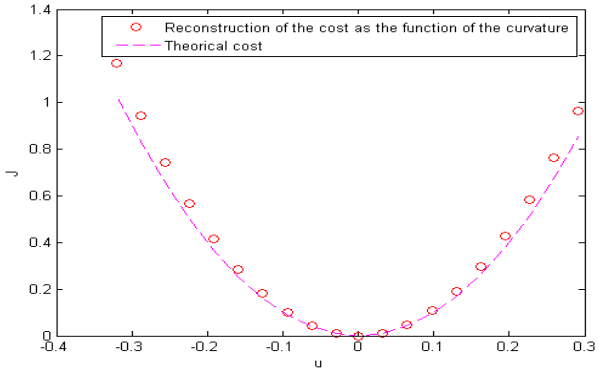

Figure 6: Reconstruction of the cost as the function of the curvature and the theorical cost

5 CONCLUSION

In this article we presented a ground control station simulator for fixed-wing aircrafts that can easily be

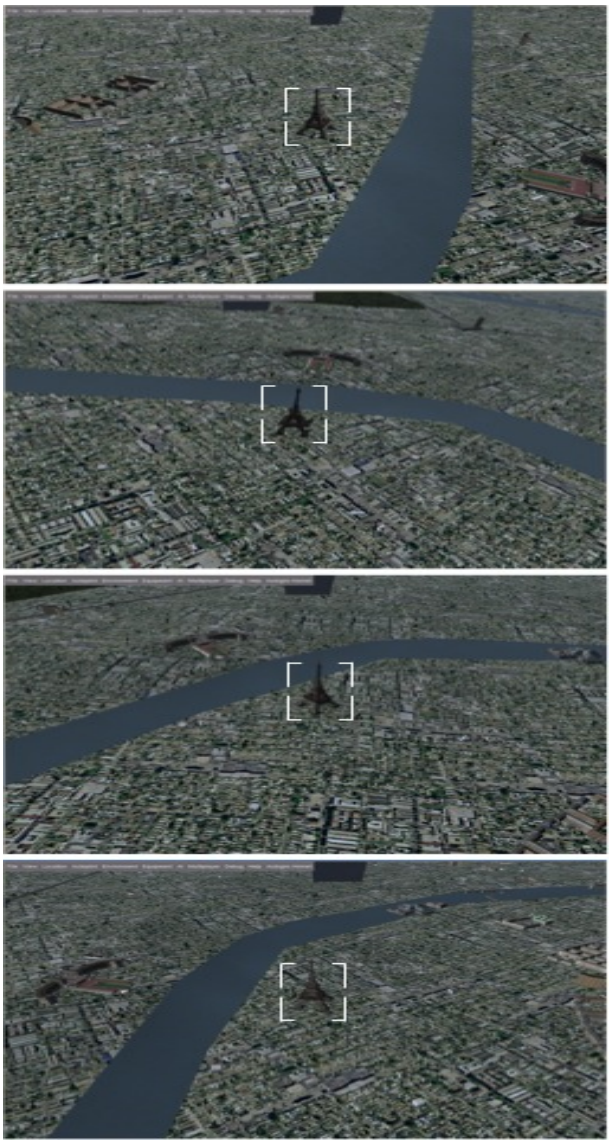

Figure 7: Watching the tower while traveling on a circle

Figure 8: Large view of the simulator

extended to rotary-wing ones. This description has been done in the form of a module based decomposition. The control module was given special attention as it deals with several issues important in order to improve the UAV autonomy. We sketched ideas to tackle the planification/replanication task which can be used to deal with pattern based navigation and tracking problems. We also presented an approach to improve the cost functions that are used in optimal control synthesis based on trajectories obtained from piloted aircrafts.

To conclude this work we display in figure 8 a view of the ground control station simulator. Snapshots of a monitoring mission of a famous building while flying in circles are shown in figure 7 (white squares have been added to ease visualization).

ACKNOWLEDGMENTS

This research was supported by the FUI AAP9 project: SHARE, and by the ANR Project GCM, program "blanche", project number NT09-504490.

REFERENCES

References

Berret, B., Darlot, C., Jean, F., Pozzo, T., Papaxanthis, C. & Gauthier, J.-P. (2008). The inactivation principle: Mathematical solutions minimizing the absolute work and biological implications for the planning of arm movements, *PLoS Computational Biology* **4**(10): 25.

Betts, J. T. (2001). *Practical Methods for Optimal Control using Nonlinear Programming*, number 3 in *Advances in design and control*, SIAM.

Boiffier, J.-L. (1998). *The dynamics of flight, the equations*, Wiley and sons.

Bullo, F. & Lewis, A. D. (2004). *Geometric Control of Mechanical Systems*, Springer-Verlag, Berlin.

Chitour, Y., Jean, F. & Mason, P. (2012). Optimal control models of the goal-oriented human locomotion, *SIAM J. Control and Optim.* **50**: 147–170.

Chittaro, F., Jean, F. & Mason, P. (to appear). On the inverse optimal control problems of the human locomotion: stability and robustness of the minimizers, *Journal of Mathematical Sciences* .

Craighead, J., Murphy, R., Burke, J. & Goldiez, B. (2007). A survey of commercial and open source unmanned vehicle simulators, *IEEE International Conference on Robotics and Automation*, pp. 852 – 857.

Cummings, M. L., Platts, J. T. & Sulmistras, A. (2006). Human performance considerations in the development of interoperability standards for uav interfaces, *Moving Autonomy Forward Conference*.

Diebel, J. (2006). Representing attitude: Euler angles, unit quaternions, and rotation vectors, *Technical report*, Stanford University.

Fabiani, P., Fuertes, V., Piquereau, A., Mampey, R. & Teichteil-Königsbuch, F. (2007). Autonomous flight and navigation of vtol uavs: from autonomy demonstrations to out-of-sight flights, *Aerospace Science and Technology* **11**(2-3): 183–193.

Fliess, M., Lévine, J., Martin, P. & Rouchon, P. (1995). Flatness and defect of non-linear sys-

tems: Introductory theory and examples, *International Journal of Control* **61**(6): 1327–1361.

Grewal, M. S., Weill, L. R. & Andrews, A. P. (2007). *Global Positioning Systems, Inertial Navigation, and Integration*, Wiley-Interscience.

Ippolito, C., Yeh, Y. & Campbell, C. (2009). A trajectory generation aproach for payload directed flight, 47^{th} *AIAA Aerospace Science Meeting*, Vol. AIAA-2009-1351.

Junkins, J. L. & Schaub, H. (2002). *Analytical Mechanics of Aerospace Systems*, aiaa edn.

Kim, H. J., Shim, D. H. & Sastry, S. (2002). Flying robots: modeling, control and decision making, *Proceedings 2002 IEEE International Conference on Robotics and Automation Cat No02CH37292* **1**(1): 66–71.

Laumond, J.-P. (1998). *Robot Motion Planning and Control*, Springer-Verlag, Berlin. Available online at http://www.laas.fr/∼jpl/book.html.

LaValle, S. M. (2006). *Planning Algorithms*, Cambridge University Press, Cambridge, U.K. Available at http://planning.cs.uiuc.edu/.

Li, W., Todorov, E. & Liu, D. (2011). Inverse optimality design for biological movement systems, *Proceedings of teh* 18^{th} *IFAC World Congress*, Vol. 18 Part 1.

Moiseev, I. & Sachkov, Y. L. (2010). Maxwell strata in sub-riemannian problem on the group of motions of a plane, *ESAIM: Control, Optimisation and Calculus of Variations* **16**: 380–399.

NATO (2008). *STANAG 4586-B-80*.

Park, S., Deyst, J. & How, J. P. (2004). A new nonlinear guidance logic for trajectory tracking, *Proceedings of the AIAA Guidance Navigation and Control Conference*, AIAA, pp. 1–16.

Sachkov, Y. (2011). Cut locus and optimal synthesis in the sub-riemannian problem on the group of motions of a plane, *ESAIM: Control, Optimisation and Calculus of Variations* **17**: 293–321.

Sachkov, Y. L. (2010). Conjugate and cut time in the sub-riemannian problem on the group of motions of a plane, *ESAIM: Control, Optimisation and Calculus of Variations* **16**: 1018–1039.

Schmidt, L. V. (1998). *Introduction to Aircraft Flight Dynamics*, Vol. 6, AIAA education series.

Sheridan, T., Verplank, W. & Brooks, T. (1978). Human and computer control of undersea teleoperators, *The 14th Annual Conference on Manual Control* pp. 343–357.

Sorton, E. F. & Hammaker, S. (2005). Simulated flight testing of an autonomous unmanned aerial vehicle using flightgear, *Infotech@aerospace Online Proceedings*.

Tisdale, J., Kim, Z. & Hedric, K. (2009). Autonomous uav path planning and estimation, *Robotics and Automation Magazine, IEEE* **16**(2): 35–42.

Van-Nieuwstadt, M. J. & Murray, R. M. (1998). Real-time trajectory generation for differentially flat systems, *International Journal of Robust and Nonlinear Control* **8**(11): 995–1020.

Wanner, J.-C. (1984). *Dynamique du vol et pilotage des avions*, onéra - supaéro edn.

wescam (2011). *www.wescam.com*.

Yang, Z. J., Qi, X. H. & Shan, G. L. (2009). Simulation of flight control laws design using model predictive controllers, *Proceedings of the IEEE International Conference on Mechatronics and Automation, Changchun, China*, pp. 4213 – 4218.