Experimental Cooperative Control of Fixed-Wing Unmanned Aerial Vehicles

Selcuk Bayraktar, Georgios E. Fainekos, and George J. Pappas

***Abstract*— Recent years have seen rapidly growing interest in the development of networks of multiple unmanned aerial vehicles (U.A.V.s), as aerial sensor networks for the purpose of coordinated monitoring, surveillance, and rapid emergency response. This has triggered a great deal of research in higher levels of planning and control, including collaborative sensing and exploration, synchronized motion planning, and formation or cooperative control. In this paper, we describe our recently developed experimental testbed at the University of Pennsylvania, which consists of multiple, fixed-wing UAVs. We describe the system architecture, software and hardware components, and overall system integration. We then derive high-fidelity models that are validated with hardware-in-the-loop simulations and actual experiments. Our models are hybrid, capturing not only the physical dynamics of the aircraft, but also the mode switching logic that supervises lower level controllers. We conclude with a description of cooperative control experiments involving two fixed-wing UAVs.**

I. Introduction

The control systems community has historically been motivated by challenging problems in the aerospace industry. Modern aerospace applications call for increasing levels of autonomy from decreasing (in size) aerial and space vehicles, which are frequently networked.

This modern view of future aerospace vehicles, whether civilian or military, has directed a substantial amount of recent research efforts in the areas of interconnected systems, cooperative control, formation control, control with communication constraints, and the relationship between dynamic network topologies and control. The references cited in [1] and [2] can serve as a partial survey of much of the related literature.

The developments on the theoretical front have been followed with similar achievements on the experimental side. However, most experimental results have focused on single aerial vehicles. In particular, we have witnessed autonomous or aggressive control of helicopters [3], [4], [5], [6], [7], blimps [8], [9] as well as fixed wing planes [10], [11], [12], [13], [14]. However, the experimental control of multiple fixed-wing aircraft is in its infancy, and has clearly not reached the same level of theoretical or experimental maturity as control of single aerial vehicles.

In this paper, we describe our recently developed experimental testbed at the University of Pennsylvania, which consists of multiple, fixed-wing UAVs. Our testbed has been rapidly developed by integrating of-the-shelf solutions for lower levels of control. This has allowed us to bypass the typically long development of (by now mature) low level controllers, and has enabled algorithm development and experimentation at more unexplored higher levels of UAV planning and operation, which include multi-UAV planning and control. As higher levels of planning typically supervise lower level controllers, higher level UAV models will naturally exhibit hybrid dynamics, mixing continuous UAV dynamics, and mode-switching logic.

In Section II, we describe the system, software, and hardware architecture of our recently developed testbed. Section III develops a higher-level hybrid model of each UAV. Our hybrid models capture both high level abstractions of the aircraft dynamics, as well as mode switching logic that supervises the switching among lower level controllers. The system is tested on high fidelity (hardware-in-the-loop) multi-UAV simulation environment, described in Section IV, which contains accurate models of our UAVs. In Section V, we conclude this paper with actual cooperative control experiments performed in August 2003 at Fort Benning, Georgia. Due to space, we refer the reader to the technical report [16] for further details. We also refer the reader to [17] for videos of actual cooperative control experiments.

II. System Description and Architecture

Fig. 1 illustrates the Unmanned Aerial Vehicles (UAVs) recently developed at the GRASP Laboratory of the University of Pennsylvania. Each UAV consists of an airframe and engine, avionics package, onboard laptop and additional sensing payload. We briefly describe the basic components of our UAVs, as well as the overall system architecture.

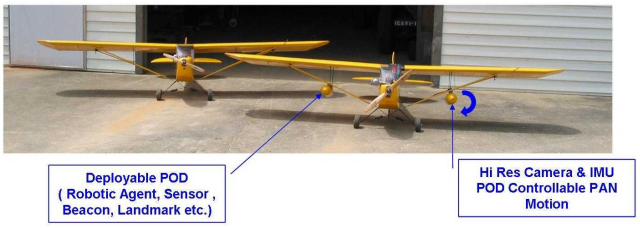

Fig. 1. PennUAVs: External Payloads (POD)

Research is partially supported by the Army Research Office MURI Grant DAAD 19-02-01-0383 and NSF ITR

Authors are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA. E-mail: {selcuk,fainekos,pappasg}@grasp.cis.upenn.edu

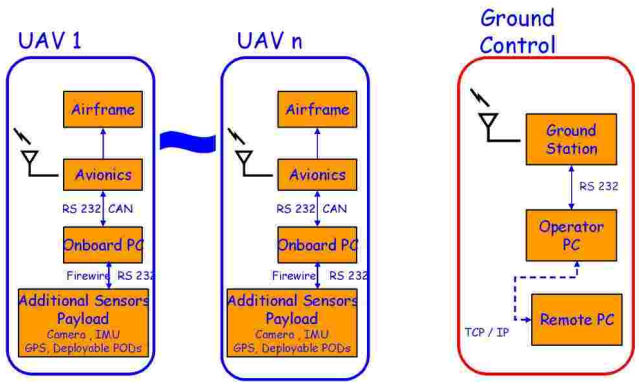

Fig. 2. Multi-UAV and Ground Station Functional Architecture

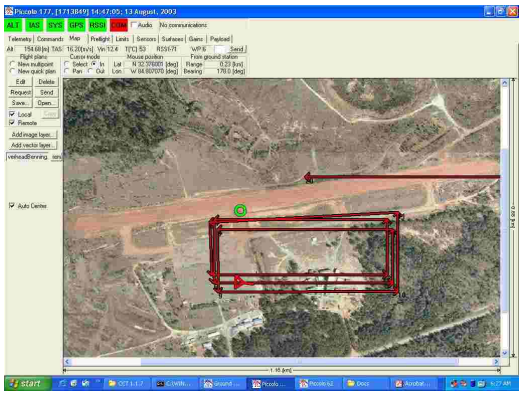

Fig. 3. Piccolo Ground Station Operator Interface showing flight plan and actual UAV position. (August 2003, Fort Benning, Georgia)

A. UAV Airframe and Payload

The airframe of each UAV is a quarter scale Piper Cub J3 model airplane with a wing span of 104 inches (~ 2.7 m) (see Fig. 1). The powerful glow fuel engine has a power rating of 3.5 HP, resulting in a maximum cruise speed of 60 knots ($\sim 30m/s$), at altitudes up to 5000 feet (~ 1500 m), and a flight duration of 15 - 20 minutes.

The airframe-engine combination enables having significant scientific payload on board. Fig. 1 shows pods that have been installed underneath each side of the wing containing high resolution cameras, IMUs, as well as deployable sensors, beacons, and landmarks. More precisely, each UAV can carry the following internal and external payloads:

- Onboard Laptop PC Dell X200
- IMU 3DM-G from MicroStrain. Includes three angular rate gyros with three orthogonal DC accelerometers, three orthogonal magnetometers, multiplexer, 12 bit A/D converter, and embedded microcontroller to provide three orientation angles (pitch, roll, yaw) in dynamic and static environments.
- External GPS Navistar GPS receiver from CMC electronics. 10 Hz raw data output.
- Camera DragonFly IEEE-1394 1024×768 at 15 frame per second (fps) from *Point Grey Research*.
- Custom designed camera-IMU Pod includes the IMU and the camera mounted on the same plate. The plate is soft mounted on four points inside the pod. Furthermore, the pan motion of the pod can be controlled through an external user PWM port on the avionics.
- Custom designed deployable Pod could be used to carry sensors, beacons, landmarks or even robotic agents. It can be deployed with a RC servo connected to external user PWM port on the avionics.

B. UAV Avionics and Ground Station

Each UAV is controlled by a highly integrated, user customizable Piccolo avionics board which is manufactured by *CloudCap Technologies* [18]. The avionics board comes equipped with the core autopilot, a sensor suite which includes GPS, Inertial Measurement Unit consisting of three gyros, three accelerometers and two pressure ports one for barometric altitude one for true airspeed. A 40 MHz embedded Motorola MPC 555 Power PC receives the state information from all sensors and runs core autopilot loops at a rate of 20 Hz commanding the elevator, ailerons, rudder and throttle actuators as well as external user payload ports.

Each UAV continuously communicates with the ground station. The communication occurs at 1 Hz and the range of the communication can reach up to 6 miles. The ground station performs differential GPS corrections, and updates the flight plan, which is a sequence of three dimensional way-points connected by straight lines. The UAVs can also be commanded in a similar way from a supervisory controller (residing on-board the UAV laptop), allowing further decentralization in the physical layer of the architecture (see Fig. 2).

The ground station can concurrently monitor up to 10 UAVs. Direct communication between UAVs can be emulated through the ground or using the local communication channel on the UAVs (80211b - wireless network card). The ground station has a friendly operator interface program (shown in Fig. 3), which allows the operator to monitor flight progress, obtain telemetry data, or dynamically change the flight plans using geo-referenced maps. Furthermore the operator interface program can act as a server and enable multiple instances of the same software to communicate over a TCP/IP connection. This allows us to monitor or command and control the experiment in real-time, remotely.

III. High-Level Hybrid UAV Model

Our goal in this section, and one of the main goals of this paper, is to derive a hybrid model of the Piccolo-controlled dynamics, coupled with the mode- (or waypoint(WP) -) switching logic. Fig. 4 shows the main components that need to be modeled, which include the aircraft dynamics (which are controlled by the Piccolo avionics board), a sensor model, as well as mode switching logic. We describe each component individually, and we model the integrated system using hybrid models.

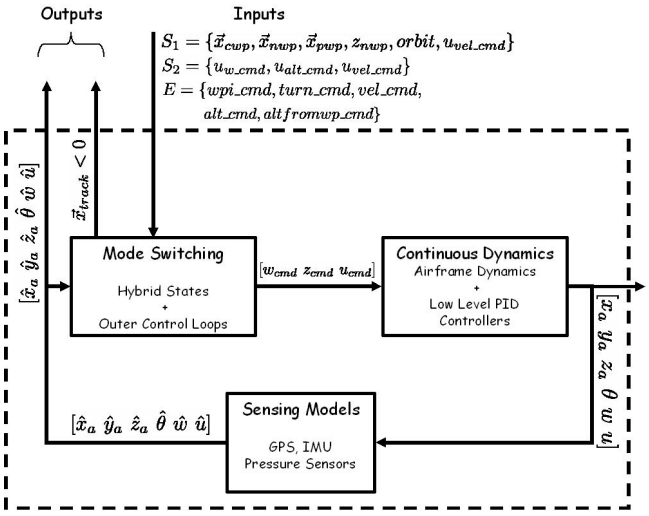

Fig. 4. Hybrid UAV Control Loop

A. Piccolo-Controlled UAV Dynamics

The autopilot in the Piccolo avionics consists of seven PID loops and a turn compensator [18]. The inner control loops regulate, among other things, airspeed, altitude, turn rate. The low-level inner control loops allows us to abstract away lower level, detailed dynamics. The longitudinal dynamics of the system, the altitude and velocity states are modeled as decoupled first order differential equations with saturation constraints. The simplified equations that model the (Piccolo-)controlled physical continuous UAV dynamics are as follows:

$$\begin{aligned} \dot{x}_a &= u\cos\theta \\ \dot{y}_a &= u\sin\theta \\ \dot{u} &= 1/\tau_u(-u + u_{cmd}),\ \underline{u} \le \dot{u} \le \overline{u} \qquad (1) \\ \dot{\theta} &= \omega \\ \dot{\omega} &= 1/\tau_\omega(-\omega + \omega_{cmd}) \end{aligned}$$

$$\dot{z}_a = 1/\tau_z(-z_a + z_{cmd}),\ \underline{z} \le \dot{z} \le \overline{z} \qquad (2)$$

where $(x_a, y_a, z_a) \in \mathbb{R}^3$ is the position of the UAV in the space with respect to some world frame, u is the UAV velocity, ω is the turn rate of the UAV, and θ is the angle defined within $-\pi \le \theta \le \pi$. External inputs include u_{cmd}, ω_{cmd}, and z_{cmd} where the subscript cmd defines controlled inputs. The time constants are τ_u, τ_ω, and τ_z respectively. Note that the inner control loops result in abstracted dynamics that decouple the planar dynamics from the altitude dynamics. As wind sensors are available on board, the above model can be easily extended to include (and compensate for) the effect of wind.

B. Sensing model

The sensors implemented in the avionics are modeled as a first order hold (with delay or not) where T_s is the sampling period, $k = \lfloor t/T_s \rfloor$ (where $\lfloor\cdot\rfloor$ is the floor function), τ_{dx}, τ_{dy}, $\tau_{d\theta}$ are the delays and

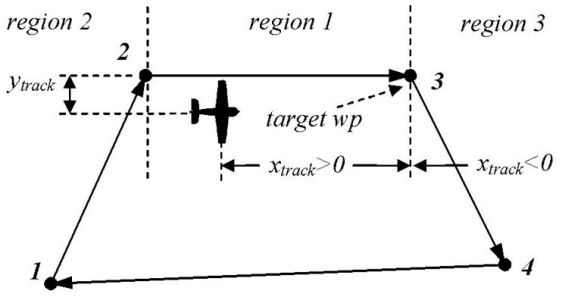

Fig. 5. Tracking a flight plan. The flight plan consists of four waypoints 1,2,3,4. The UAV is currently tracking line segment 2–3.

$$\begin{aligned} \hat{x}_a(t) &= x_a(kT_s - \tau_{dx}) \\ \hat{y}_a(t) &= y_a(kT_s - \tau_{dy}) \\ \hat{u}(t) &= u(kT_s) \qquad (3) \\ \hat{w}(t) &= w(kT_s) \\ \hat{\theta}(t) &= \theta(kT_s - \tau_{d\theta}) \end{aligned}$$

$$\hat{z}_a(t) = z_a(kT_s) \qquad (4)$$

C. Piccolo-High Level Controllers

The main objective of the high level controller is control the UAV to fly a flight plan, which is a sequence of waypoints. The avionics has the ability to store up to 100 waypoints. Each waypoint consists of latitude, longitude, and altitude of the waypoint. The flight plan can be traversed at various desired airspeeds. Note that the flight plans must be closed, that is at some point a next waypoint entry of a waypoint must point to a waypoint that it is already in the flight plan (see Fig. 5). Furthermore, the controller has the capability of circling around waypoints (holding pattern), if desired.

The high level controller completes the flight plan by using one airspeed controller, one altitude controller, and two lateral controllers, one focusing on line segment tracking, and one on circling around waypoints. These controllers are orchestrated by the waypoint switching logic, which determines which waypoint segment is active. We now describe the controllers and the waypoint switching logic.

1) Lateral Controller - Line segment tracker: The lateral control law is trying to drive to zero the angle between the actual heading of the UAV and the desired heading. The desired heading is given by the position of the aircraft $\vec{x}_a$, the line segment to be tracked $\vec{v}_{kt}$ and the track convergence parameter K. In order to determine the desired heading of the UAV, we just need to express the vector $\vec{v}_{ak}$ in terms of vectors $\vec{x}_t$ and $\vec{x}_p$ (see Fig. 6).

$$\vec{v}_{ak} = \vec{x}_t - l_{\vec{v}_{kt}} \frac{\vec{x}_t - \vec{x}_p}{\| \vec{x}_t - \vec{x}_p \|} - \vec{\hat{x}}_a \qquad (5)$$

$$l_{\vec{v}_{kt}} = \vec{v}_{at} \cdot \frac{(\vec{x}_t - \vec{x}_p)}{\| \vec{x}_t - \vec{x}_p \|} - K \qquad (6)$$

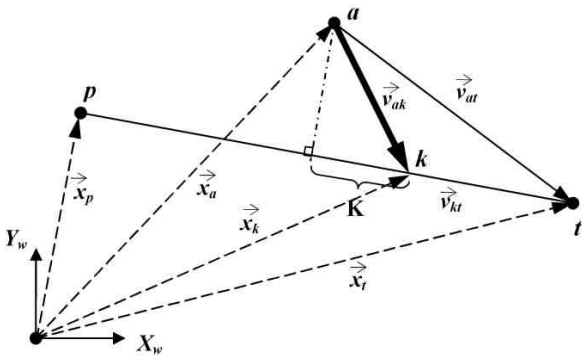

Fig. 6. Inertial frame model. X_w, Y_w is the world coordinates system

Where $\vec{x}_p$ is the position of the preceding waypoint and $\vec{x}_t$ is the position of the target waypoint. Even though the actual coordinates of each WP as well as the position of the UAV lie in $\mathbb{R}^3$, for the lateral control law we consider only the xy plane parallel to earth's plane. Hence, from now on all the vectors will be in $\mathbb{R}^2$, i.e. $\vec{x}_p, \vec{x}_t \in \mathbb{R}^2$, unless we explicitly state otherwise. Finally, K is the track convergence parameter $(K > 0)$ and it should be chosen in such a way such that it representative of the nominal vehicle turn radius [19].

The position x_{track} of the UAV along the track (see Fig. 5) is given by the following equation:

$$\vec{x}_{track} = \frac{(\vec{x}_t - \vec{x}_p)^T(\vec{x}_t - \hat{\vec{x}}_a)}{\parallel \vec{x}_t - \vec{x}_p \parallel} \tag{7}$$

The desired angle θ_{des} for the UAV is:

$$\theta_{des} = atan2(y_{ak}, x_{ak}) \tag{8}$$

where x_{ak} and y_{ak} are the components of the vector $\vec{v}_{ak}$ (5). The error to be driven to zero is:

$$\theta_e = \theta_{des} - \hat{\theta} \tag{9}$$

which is modified by the following relations in order to keep the error in the range $[0, 2\pi)$:

$$\begin{array}{cc} \text{if } \theta_e > \pi & \text{then } \theta_e = \theta_{des} - \hat{\theta} - 2\pi \\ \text{if } -\pi \leq \theta_e \leq \pi & \text{then } \theta_e = \theta_{des} - \hat{\theta} \\ \text{if } \theta_e < -\pi & \text{then } \theta_e = \theta_{des} - \hat{\theta} + 2\pi \end{array} \tag{10}$$

The following PID control is used for the command input.

$$\omega_{cmd} = K_p\theta_e + K_d\dot{\theta}_e + K_i \int \theta_e dt \tag{11}$$

Moreover the controller bounds the bank angle, which also induces bounds on the control input depending on velocity (u) by:

$$|\omega_{cmd}| \leq \frac{g \tan \phi_{limit}}{u} \tag{12}$$

where ϕ_{limit} is the bank limit and g gravitational acceleration.

2) Lateral Controller - Circle Track: When the specified waypoint is a circle waypoint the UAV circles around the waypoint at a constant radius defined by the parameter K [20] until a new waypoint index change command arrives. In this mode the outer track loop simply projects the current position of the UAV radially on to the circle and calculates the tangent to the circle at that point. Then by simply using the line track control law the UAV flies the line segment defined by the projected point and the tangent vector. The $\vec{x}_p$ and $\vec{x}_t$ gets updated periodically at T_s.

$$\vec{v}_{radial} = \frac{\hat{\vec{x}}_a - \vec{x}_{wp}}{\parallel \hat{\vec{x}}_a - \vec{x}_{wp} \parallel} \tag{13}$$

where $\vec{x}_{wp}$ is the position vector of the waypoint to be orbited.

$$\vec{x}_p = \vec{x}_{wp} + K\vec{v}_{radial} \tag{14}$$

where K is the track convergence parameter.

$$\vec{x}_t = \vec{x}_p + \begin{pmatrix} -v_y \\ v_x \end{pmatrix}_{radial} \tag{15}$$

D. Piccolo-Waypoint Switching Logic

The controllers described above are subject to supervisory switching logic. The switching logic arises in two forms. First, the system needs to decide whether the waypoint has been reached or not, and then reset the controllers with information from the next waypoint in the sequence. Second, the user can dynamically update the waypoints and UAV switches to the assigned waypoint (WP).

If the UAV is already executing a flight plan then it switches to the next waypoint when $x_{track} < 0$ (see Fig. 5). For example when the UAV tracks the line segment (2–3), then when it flies over the target WP 3, then x_{track} becomes less then 0 and the waypoint index (WPI) is updated to 4.

If the user or the supervisory control orders a new waypoint then the UAV does not always fly directly to the new waypoint but decides its flight plan according to the following logic (see Fig: 5):

1) If $0 < x_{track} < \parallel \vec{v}_{pt} \parallel$ where $\vec{v}_{pt}$ is the vector between the positions of the preceding waypoint $\vec{x}_p$ and the target waypoint $\vec{x}_t$, then the UAV tracks v_{pt} (region 1).
2) If the UAV is at a position where $x_{track} > \parallel \vec{v}_{pt} \parallel$, then it flies directly to the new waypoint using as preceding waypoint its current position (region 2).
3) If the UAV is at a position where $x_{track} < 0$, then it flies directly to the new waypoint using as preceding waypoint its current position (region 3).
4) If the new waypoint does not have a preceding waypoint, then the UAV flies directly to the new waypoint using as preceding waypoint its current position.

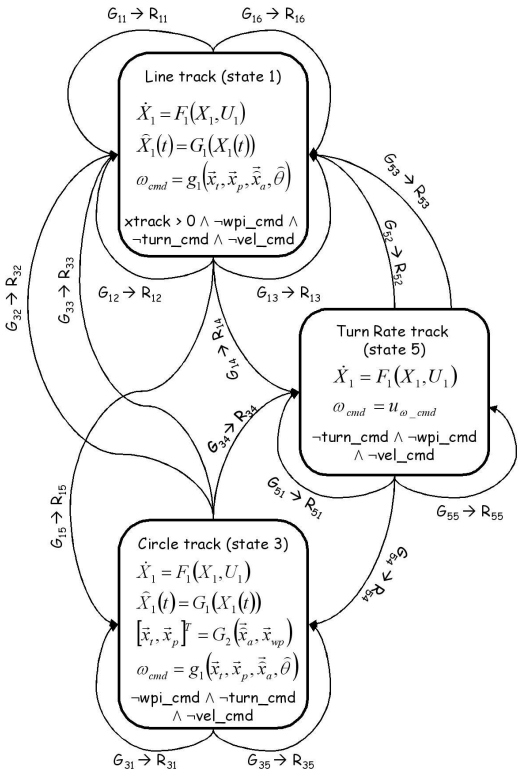

Fig. 7. Hybrid model of lateral dynamics

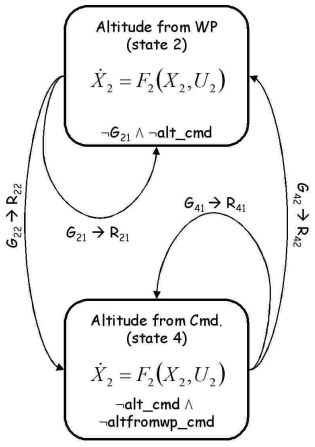

Fig. 8. Hybrid model of altitude dynamics

E. Hybrid Modeling

The hybrid model for the UAV that we present in this section captures both the continous physical dynamics of the UAV and the switching logic that supervises the lower level controllers. The hybrid model, which is presented in Fig. 7 and Fig. 8, uses the formalism developed in [21]. It consists of two concurrent but decoupled hybrid systems with inputs and outputs, one modeling the lateral and one the altitude dynamics. The lateral dynamics model consists of three (discrete) modes of operation consisting of the line and circle tracking controller described previously, as well as a turn rate controller which allows the user to directly specify a turn rate. The altitude controller has two modes of operation, one when it receives altitude setpoints from the waypoint list, and one when it directly receives them from the user.

For our model, we will use the following notation. Let $X_1 = [x_a, y_a, u, \theta, \omega]^T$ and $U_1 = [u_{cmd}, \omega_{cmd}]^T$, then $\dot{X}_1 = F_1(X_1, U_1)$ represents the set of equations (1). In the same way, let $X_2 = z_a$, $U_2 = z_{cmd}$ and $\dot{X}_2 = F_2(X_2, U_2)$ for (2). We denote the set of equations (3) by $\tilde{X}_1(t) = G_1(X_1(t))$. Let equations (5) to (12) be represented by $\omega_{cmd} = g_1(\vec{x}_t, \vec{x}_p, \vec{\hat{x}}_a, \hat{\theta})$, and equations (13) to (15) be represented by $[\vec{x}_t, \vec{x}_p]^T = G_2(\vec{\hat{x}}_a, \vec{x}_{wp})$.

There is a variety of events that trigger transitions in this hybrid model:

- wpi_cmd: command the UAV to track a new WP
- $turn\_cmd$: command the UAV to start turning with the specified turn rate u_{ω_cmd}
- alt_cmd: UAV changes its altitude to u_{alt_cmd}
- vel_cmd: UAV changes its velocity to u_{vel_cmd}
- $altfromWP\_cmd$: command the UAV to use the altitude from the current WP.

Switching from one mode of operation to another can be directly triggered using the above discrete commands. In addition, switching also occurs when some predicates (or guards) of the states and inputs are true. In particular, waypoint switching is based on a function that calculates the position x_{track}. Let the function $f_{xtrack} : \mathbb{R}^6 \rightarrow \mathbb{R}$ (Note: all the vectors are in $\mathbb{R}^2$) with:

$$f_{xtrack}(\vec{x}_a, \vec{x}_t, \vec{x}_p) = \begin{cases} \frac{(\vec{x}_t - \vec{x}_p)^T (\vec{x}_t - \vec{x}_a)}{\|\vec{x}_t - \vec{x}_p\|} & \text{if } x_p \text{ is defined} \\ -1 & \text{otherwise} \end{cases}$$

In order to determine in which region the UAV is (see Fig. 5 and section III-D), we have to know the length of the line segment to be tracked, i.e. the length of the vector $\vec{v}_{pt}$. Let the function $f_{\vec{v}_{pt}} : \mathbb{R}^4 \rightarrow \mathbb{R}^+ \cup \{-1\}$ with:

$$f_{\vec{v}_{pt}}(\vec{x}_t, \vec{x}_p) = \begin{cases} \| \vec{x}_t - \vec{x}_p \| & \text{if } x_p \text{ is defined} \\ -1 & \text{otherwise} \end{cases}$$

Note that $\vec{x}_p$ can be undefined (see case 4 in section III-D).

Due to space restrictions, the switching conditions that determine the discrete transition from one mode to another are described in full detail in [16]. In this paper, we present some sample guards for one discrete state (Line Track), as well as the reset maps that occur when the transition occurs. The remaining guards have similar form.

Sample Line Track Switching Conditions and Reset Maps:

- The UAV has reached the switching boundary of the current WP $\longrightarrow$ Switch to the next WP in the flight plan.
  Guard:

$$G_{11} = (f_{xtrack}(\vec{\hat{x}}_a, \vec{x}_t, \vec{x}_p) \leq 0) \wedge \neg orbit$$

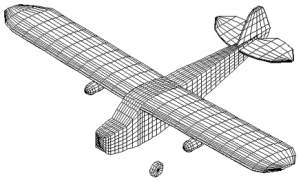

Fig. 9. The paneling for the Piper J3 cub

Fig. 10. 3D-solid CAD model of the Piper J3 cub

Reset map:

$$R_{11} = \{\vec{x}_p := \vec{x}_{cwp};\ \vec{x}_t := \vec{x}_{nwp}\}$$

- Change WP index command event (wpi_cmd) triggered (and) we are in region 1 (see Fig. 5) with respect to the new line segment $\longrightarrow$ Switch to the commanded WP.
  Guard:
  $$G_{12} = wpi\_cmd \wedge \neg orbit \wedge$$
  $$(0 < f_{xtrack}(\vec{\hat{x}}_a, \vec{x}_{cwp}, \vec{x}_{pwp}) \leq$$
  $$f_{\vec{v}_{pt}}(\vec{x}_{cwp}, \vec{x}_{pwp}))$$
  Reset map:
  $$R_{12} = \{\vec{x}_p := \vec{x}_{pwp},\ \vec{x}_t := \vec{x}_{cwp}\}$$

IV. High Fidelity Simulation

The hybrid model developed in the previous section will serve as a modeling abstraction in developing high-level algorithms involving multiple UAVs. In order to implement and verify the performance of algorithms designed using the hybrid model described in the previous section, the designed algorithms are first executed on very detailed simulation models (MATLAB simulation toolbox developed in house). Furthermore, in order to fully utilize the hardware-in-the-loop (HIL) simulator (see [22]) offered with the Piccolo developer's kit and thus attain accurate simulations, it is mandatory to develop a high fidelity dynamics model for the Piper J3 cub model airplane. This dynamics model takes into account aerodynamics, propulsion and inertia effects.

A. Aerodynamic Model

The lack of publicly available aerodynamic data for the Piper J3 cub model airplane as well as the modifications done on the airframe by the PennUAV team necessitate the estimation of the aerodynamic parameters of the model airplane. For this purpose, the vortex panel method with boundary layer analysis was employed. By using the panel method for this purpose one should expect overestimation of the lift and underestimation of the drag.

The paneling for the Piper J3 cub model airplane is presented in Fig. 9. The half model (symmetric) consists of 1167 panels and the computation time is less then $5min$ on a P4. All the tests were run at nominal speed $\sim 15m/sec$ at sea level standard (SLS) conditions (i.e. a subsonic low Reynolds number environmnet; at these conditions $Re \sim 4.2 \cdot 10^5$).

B. Inertia Model

The simple design of the Piper J3 cub made possible the rapid development of a 3D-solid CAD model (see Fig. 10). The model includes all skeletal structure of the airframe with ribs and panels and also all other significant components of the UAV namely PODs, actuators, onboard PC, avionics box and enclosure, landing gear, fuel, fuel tank, engine and propeller. From such a detailed model one can easily extract a very accurate estimate of the inertia coefficients. Furthermore, the center of gravity of the CAD model was compared with the real airframe for verification and was found to be very accurate.

C. Visualization

The results of the high-fidelity HIL simulator can be realistically visualized using *Microsoft Flight Simulator*. In addition to animating simulated trajecoties, we can also use the Microsoft Flight Simulator for playing back actual multi-UAV experiments, which is the focus of the next section.

V. Cooperative Control Experiments

A variety of formation control experiments were performed using two fixed wing UAVs. In our cooperative control experiments, two UAVs flew in formation in a decentralized manner, using a leader follower architecture. The leader UAV was given a pre-determined flight plan. The trajectory of the follower UAV was updated once per second in real time through the ground station to keep the follower at a fixed distance offset from the leader. The formation control law we used is inspired by the control law obtained in [1]. Fig. 11 shows the trajectories of the UAVs and the predetermined flight plan that was assigned to the leader. Marks on the trajectories represent the respective positions of the UAVs every 7 seconds. The winds were around 4 m/s relatively significant with respect to airspeed ($\sim 15m/s$) in the -x direction (east to west) at the time of the experiment. The tracking of the leader UAV, average deviation from course on the upwind leg is much better from the downwind leg due to the way the lateral control laws were implemented. Another detail to note is that the tracking behaviour of the system is dominated by the low sampling frequency (1Hz) of the outer lateral control loop which is responsible for generating turn rate commands. Thus, if it is desirable to achieve tighter formation, the sampling frequency of the lateral control loop has to be increased together with higher performance GPS, and IMU sensor fusion.

A variant of the previous experiment was recently conducted where the follower UAV was flying 50 meters directly above the leader, monitoring the UAV below with a high resolution camera. Videos of all experiments described in this section can be seen in [17] (also available on line at www.grasp.upenn.edu/uav).

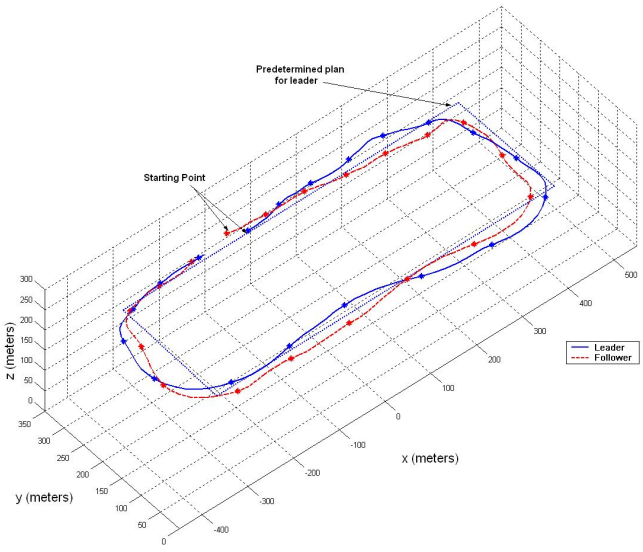

Fig. 11. Autonomous Formation Experiment 1 (Aug. 14th 2003, Fort Benning, Georgia,

Fig. 12. Autonomous Formation Experiment 1 (Aug. 14th 2003, Fort Benning, Georgia,

VI. Conclusions

In this paper, we have described the multi-UAV testbed developed at the University of Pennsylvania, developed a hybrid model that captures both abstracted UAV dynamics as well as mode-switching logic, and reported on cooperative control experiments involving two Penn UAVs. The hybrid model described in this paper, which has been validated in both hardware-in-the-loop simulations and actual experiments, will serve as modeling abstraction and will be utilized in applying hybrid control methods in order to solve challenging problems involving multiple UAVs, such as synchronous or asynchronous formations, coordinated search and rescue, air-ground integration, and rapid emergency response.

Acknowledgements: The authors would like to thank George Asteris, Ben Grocholsky, James F. Keller, Vijay Kumar and Camillo J. Taylor.

References

[1] H. Tanner, A. Jadbabaie, and G.J. Pappas, ”Stable flocking of mobile agents, Part I : Fixed Topology”, *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, December 2003

[2] H. Tanner, A. Jadbabaie, and G.J. Pappas, ”Stable flocking of mobile agents, Part II : Dynamic Topology”, *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, December 2003

[3] V. Gavrilets, A. Shterenberg, M. Dehaleh and E. Feron, ”Avionics System for a Small Unmanned Helicopter Performing Aggressive Maneuvers”, *in 19th Digital Avionics Systems Conference* Proceedings., 7-13 Oct. 2000, vol.1, pp:1E2/1 - 1E2/ .

[4] E.N. Johnson, D.P. Schrage, ”The Georgia Tech Unmanned Aerial Research Vehicle: GTMax”, *in Proceedings of the AIAA Guidance, Navigation, and Control Conference 2003*, AIAA Paper 2003-5741.

[5] S. Saripalli, D. J. Naffin, and G. S. Sukhatme, ”Autonomous Flying Vehicle Research at the University of Southern California”, *in Multi-Robot Systems: From Swarms to Intelligent Automata, Proceedings of the First International Workshop on Multi-Robot Systems*, A. Schultz and L. E. Parker, Kluwer Academic Publishers, 2002, pp. 73-82.

[6] O. Amidi, T. Kanade, and R. Miller. ”Autonomous Helicopter Research at Carnegie Mellon Robotics Institute”, *in Proceedings of Heli Japan 98*, Gifu, Japan, 1998.

[7] H.J. Kim, D.H. Shim, S. Sastry, ”Flying robots: modeling, control and decision making”, *in Proceedings of International Conference on Robotics and Automation, ICRA ’02*, 11-15 May 2002, vol.1

[8] J. Kim, J. Keller, V. Kumar, ”Design and Verification of Controllers for Airships”, *IEEE IROS*, Las Vegas, NV, Oct, 2003

[9] E. Olsen, C.W. Park, and J. How, ”3D Formation Flight using Differential Carrier-phase GPS Sensors”, *The J. of The Institute Of Navigation*, Spring, 1999, Vol. 146, No. 1, pp. 3548.

[10] J. Evans, G. Inalhan, J.S. Jang, R. Teo, C.J. Tomlin, ”DragonFly: a versatile UAV platform for the advancement of aircraft navigation and control”, *in 20th Conference Digital Avionics Systems*, 14-18 Oct. 2001, Pages:1C3/1 - 1C3/12 vol.1.

[11] J. Lee, R. Huang, A. Vaughn, X. Xiao, J.K. Hedrick, M. Zennaro, R. Sengupta, ”Strategies of Path-Planning for a UAV to Track a Ground Vehicle”, *in The Second Annual Symposium on Autonomous Intelligent Networks and Systems*, Menlo Park, CA, June 2003.

[12] Fierro, R., Belta, C., Desai, J.P. and Kumar, V., ”On Controlling Aircraft Formations”, *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, FL, Dec. 2001, vol. 2, pp. 1065-70.

[13] T. Schouwenaars, J. P. How, and E. Feron, ”Decentralized Cooperative Trajectory Planning of Multiple Aircraft with Hard Safety Guarantees”, presented at *the AIAA Guidance, Navigation and Control Conference*, Aug 2004. AIAA-2004-5141.

[14] Mark E. Campbell and Jarurat Ousingsawat, On-line Estimation and Path Planning for Multiple Vehicles in an Uncertain Environment, *Proceedings of the 2002 AIAA Conference on Guidance*, Navigation and Control, Monterey, CA, 2002

[15] D. Kingston, R. Beard, T. McLain, M. Larsen, W. Ren, ”Autonomous Vehicle Technologies for Small Fixed Wing UAVs”, *AIAA 2nd Unmanned Unlimited Systems, Technologies, and Operations–Aerospace, Land, and Sea Conference and Workshop & Exhibit*, San Diego, CA, September, 2003, Paper no. AIAA-2003-6559.

[16] S. Bayraktar, G. E. Fainekos, and G.J. Pappas, ”Hybrid Modeling and Experimental Cooperative Control of Multiple Unmanned Aerial Vehicles”, Technical Report, *Department of CIS, University of Pennsylvania*, 2004.

[17] S. Bayraktar and G.J. Pappas, ”Experimenting with Multiple Unammned Aerial Vehicles”, in *Video Proceedings of the 2004 International Conference on Robotics and Automation*, New Orleans, April 2004.

[18] B. Vaglienti, R. Hoag and M. Niculescu, ”Piccolo system user guide”, [http://www.cloudcaptech.com/downloads.htm], *Cloud Cap Technology*, Version 1.2.2, July 1, 2004

[19] M. Niculescu, ”Lateral track control law for Aerosonde UAV”, *39th AIAA Aerospace Sciences Meeting and Exhibit*, Paper 2001-0016, January 2001.

[20] B. Vaglienti, ”Lateral track control law for Piccolo”, [http://www.cloudcaptech.com/], *Cloud Cap Technology*, March 12 2003.

[21] J. Lygeros, ”Hierarchical Hybrid Control of Large Scale Systems”, Ph.D Thesis, *Department of EECS, University of California at Berkeley*, 1996.

[22] B. Vaglienti and M. Niculescu, ”Hardware in the loop simulator for the Piccolo avionics”, [http://www.cloudcaptech.com/downloads.htm], *Cloud Cap Technology*, June 18, 2004