JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 4, May 2007

Intelligent Unmanned Air Vehicle Flight Systems

Jodi A. Miller,* Paul D. Minear,* and Albert F. Niessner, Jr.†
The Pennsylvania State University, State College, Pennsylvania 16804

and

Anthony M. DeLullo,‡ Brian R. Geiger,‡ Lyle N. Long,§ and Joseph F. Horn¶
The Pennsylvania State University, University Park, Pennsylvania 16807

This paper describes an intelligent autonomous airborne flight capability that is being used as a test bed for future technology development. The unmanned air vehicles (UAVs) fly under autonomous control of both an Intelligent Controller (IC) running on an onboard computer and an autopilot. The IC provides the mission control while the autopilot controls the vehicle navigation and flight control. The autonomous airborne flight system is described in detail. An IC architecture directly applicable to the design of unmanned vehicles is also presented. The UAVs may operate independently or in cooperation with one another to carry out a specified mission. The intelligent UAV flight system is used to evaluate and study autonomous UAV control as well as multi-vehicle collaborative control.

I. Introduction

TRADITIONALLY, unmanned air vehicles (UAVs) have been used to perform "dull, dirty, and dangerous" missions. Military UAVs, such as Predator and Global Hawk, have demonstrated a reconnaissance and surveillance capability in such conflicts as Iraq, Bosnia, Kosovo, and Afghanistan. The Department of Defense described a roadmap for UAVs for the first quarter of the 21st century.[1] This roadmap cites two strong motivators for the continued development of Unmanned Air Systems: lower downside risk and higher confidence in mission success. UAV autonomy will be essential in making UAVs cost effective. Current UAVs require significant human interactions. It is important to develop systems where one person can easily supervise numerous vehicles.

Intelligent control technologies can be used to increase the autonomy of UAVs to a self-actualizing level. Some examples of approaches to intelligent systems are described in.[2–8] While there is no universal consensus on how to define or measure an intelligent system, there are several characteristic traits that an intelligent controller might have, including: adaptability, learning capability, non-linearity, autonomous symbol interpretation, and goal-oriented and knowledge-based behaviors.[9,10] As conceived for the Intelligent UAV Flight System described in this paper, an Intelligent Controller (IC) can be defined as one whose outputs are determined by employing input sensor data to build an internal representation of the external world, as opposed to using pre-established mathematical or model-based descriptions.[9]

Received 14 July 2006; revision received 12 January 2007; accepted for publication 19 February 2007.

* Research Faculty, Intelligent Control Systems, Applied Research Laboratory, P.O. Box 30, AIAA Member.
† Senior Research Associate, Emeritus, Guidance Systems Technology, P.O. Box 30.
‡ Graduate Assistant, Aerospace Engineering, 233 Hammond Building, AIAA Student Member.
§ Distinguished Professor, Aerospace Engineering, 233 Hammond Building, AIAA Fellow.
¶Associate Professor, Aerospace Engineering, 233 Hammond Building, AIAA Associate Fellow.

Long et al.[8] review several existing and historical software architectures for autonomous vehicles. These include approaches such as subsumption,[4] behavior-based,[5] three-layer,[6] and cognitive architectures.[7] While artificial intelligence (AI) has been around for more than 50 years, software architectures for autonomous systems are still in their infancy. Autonomous vehicles (air-, ground-, or sea-based) must operate in uncertain and complex environments. As Wray and Chong[11] discuss (in this same issue of JACIC), the key steps in many intelligent systems are to "perceive, reason, and act." But autonomous vehicles have a hierarchy of tasks they must perform, with varying levels of reasoning required. Some require complex reasoning (e.g. collaborative mission planning), but some are quite simple (reactions) and require little or no reasoning (e.g. obstacle avoidance). Human beings also have a complex hierarchy of control mechanisms, ranging from rapid motions to avoid a hot object to planning multi-year projects.

Future UAVs will need to build upon advances in artificial intelligence, computational intelligence, and neuroscience. Technologies such as neural networks, fuzzy logic, genetic algorithms, and symbolic processing will all have a role in the various subsystems. How these are all tied together into layers or hierarchies is one of the most challenging aspects of the problem. The optimal applications of these technologies is still to be determined; and will depend upon the vehicle mission requirements and funding available. The approach described herein uses an off-the-shelf autopilot for short-time-frame aircraft control coupled to an intelligent controller to "perceive, reason, and act" for long-time-frame mission objectives.

Examples of intelligent control applications illustrate the variety of tasks capable of being performed. Johnson et al.[13] have developed a UAV to enhance/augment human search capabilities by locating a specific building and identifying a possible opening into the building. Chalmers et al.[14] have demonstrated swarm capabilities that are robust and resilient. Valenti et al.[15] describe a multi-UAV system that uses off-the-shelf radio-control (R/C) aircraft and autopilots.

This paper describes an intelligent autonomous airborne flight capability (with operator override) that serves as a test bed for future technology development. "Autonomous" in this paper is defined as fully autonomous operations with human interactions as desired (versus remote piloting with human operations required). A commercially available autonomous autopilot is used to provide vehicle navigation and flight control stability for the unmanned air vehicle(s). Intelligent Controller software running on an onboard computer provides mission control. Communications between these two systems provides the basis for the Intelligent UAV Flight System. This architecture is representative of other existing (and probably future) systems. It is important to have a control approach that can accommodate short-time-frame and long-time-frame issues (from deliberative to reactive behaviors). In the current system this is accomplished using an autopilot coupled to an intelligent controller. The current system was initially tested in a laboratory environment using a hardware-in-the-loop (HIL) simulation. Once tested on the ground, actual flight tests in the field were performed.

II. Autonomous Airborne Flight System

This section describes the flight system for the autonomous airborne flight capability. Radio controlled model aircraft with commercially-developed autopilots, control processors, communications links, and sensors are used. The two major components of this system are the UAV and the ground base station along with their subsystems. When in use, the system is comprised of a single base station and multiple UAVs.

Each UAV flies autonomously under the control of its onboard autopilot and Intelligent Controller (IC). The autopilot controls the vehicle navigation and flight control stability. The IC runs on a single-board PC running the Windows XP operating system and provides mission control. Manual control of the airborne units from the base station is available for takeoff and landing of the aircraft as well as providing limited safe operation in the event of component failure.

A. Operator Base Station

The base station provides communications, flight control, telemetry recording, and flight visualization between a human operator and each of the airborne units. As shown in Fig. 1, the base station is comprised of five hardware items.

The Cloud Cap Technology (CCT) Base Station, Pilot Console, and Operator Interface software are supplied as part of Cloud Cap Technology's Evaluation Kit. A laptop computer runs the Operator Interface software and the IC-Ground communications software (developed in-house). The Global Communications Transceiver is part of the

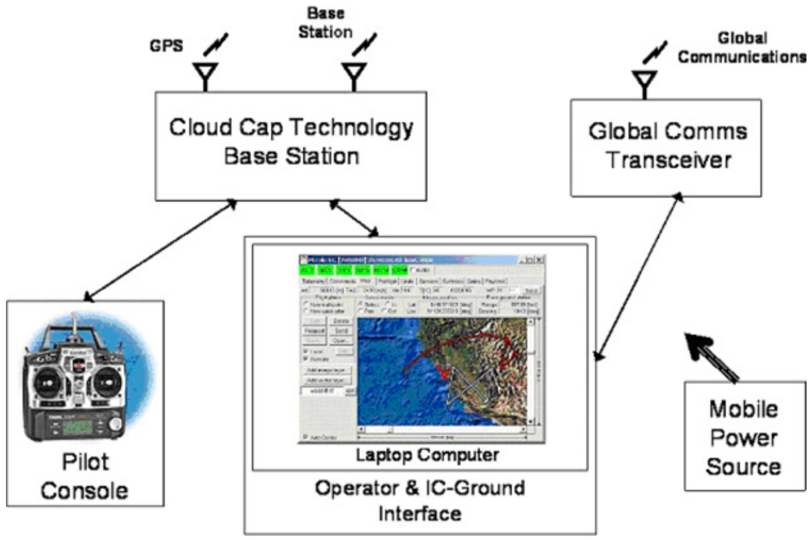

Fig. 1 Ground base station.

network that allows mission communication among the system units. The Mobile Power Source provides the power needed to run the Base Station hardware in the field.

The Operator Interface is used to send commands to the airborne units and display the flight path and sensor data from the airborne units when they are within radio range. It also records all telemetry signals between the autopilot and the base station. In addition, all communications received by the IC are recorded for later playback.

The Pilot Console, a standard Radio Control (R/C) Transmitter modified to interface with the CCT Base Station, is used to manually control any one of the aircraft. Its primary use is taking off and landing the aircraft. Control of an aircraft by the Pilot Console is asserted by first selecting the aircraft's communication channel on the operator interface, then using a switch on the pilot's console to switch between the autopilot and manual mode of flight control.

The CCT Base Station has two communications systems. The GPS communications is for receiving GPS signals to obtain an accurate location of the base station. The second communications system uses a UHF transmitter and receiver for communicating with the aircraft autopilot. It is this radio channel that allows data to be sent to and received by the aircraft autopilot. An additional communications system, an 802.11b ad-hoc network, is used to communicate with and among the ICs.

B. Airborne Platform

The airborne platform is built around an R/C trainer class model airplane that is commercially available as an Almost-Ready-to-Fly (ARF) aircraft. The model is a SIG Kadet Senior, an excellent trainer aircraft. Because of its stable flight characteristics and slow flying speed it is ideal for this testbed development effort. Figure 2 is a picture of the vehicle in a standard R/C configuration. This initial configuration was used to determine the suitability of this aircraft for the UAV application. The basic aircraft is made up of the airframe, engine, control servos, and an electrical power source. Table 1 lists the aircraft specifications.

The drive for the flight control surfaces are provided by standard Futaba servos. Electric power for the flight controls is obtained from a Nickel Metal Hydride (NiMH) battery pack. Use of the SIG Kadet model in this application required several modifications to the airframe. These modifications include the following:

1) Increased fuel capacity for extended flight times.
2) Installation of heavy duty main and nose gear.
3) Movement of the servos out of the central area of the fuselage.
4) Installation of the autopilot and control processors.
5) Installation of a pitot static tube mount.

Fig. 2 Airborne platform.

Table 1 Aircraft specifications.

Wingspan	80 inches
Wing area	1180 sq. inches
Length	64 3/4 inches
Empty weight	6.5 pounds
Gross weight	~14 pounds
Wing loading	1.7 pounds/ft^2
Engine	0.91 in^3 4-stroke

6) Installation of the GPS and communications antennas.
7) Installation of a larger than normal engine.

The required payload capacity is in excess of 5 pounds, not including an extra large fuel tank (32 oz.). Fuel weight is approximately 2 pounds. Sensors on the platform may include an infrared sensor and/or a visual sensor (camera or webcam). Due to the need to carry this additional payload, a larger-size engine, an OS Surpass 4 stroke with 0.91 cu. in. displacement, powers the aircraft. Early flight tests with weights simulating the payload brought out the

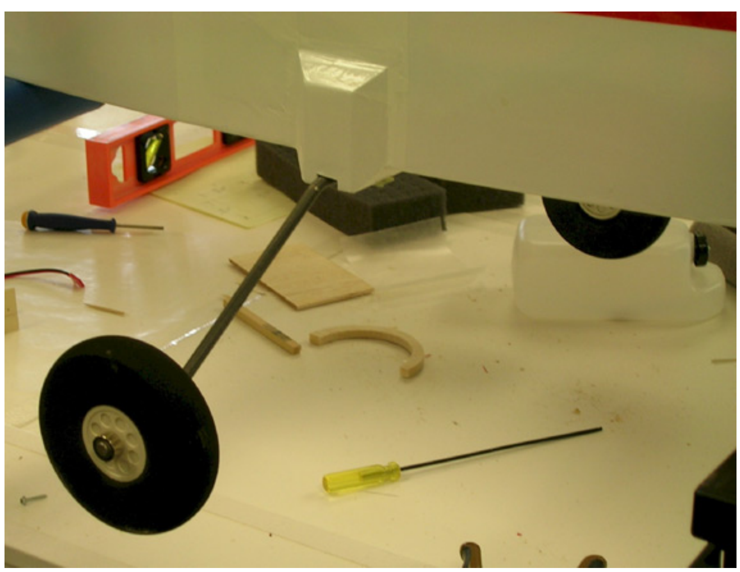

Fig. 3 Heavy duty external mount main landing gear.

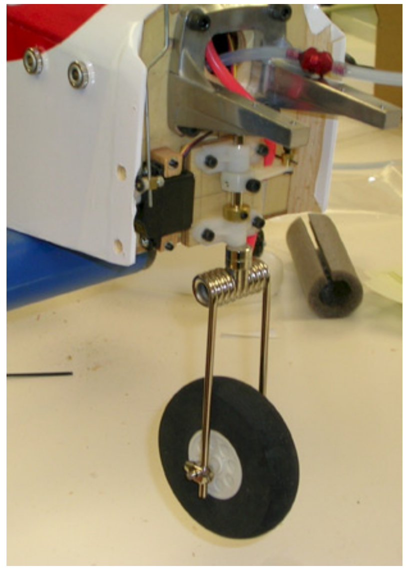

Fig. 4 Heavy duty dual strut nose gear.

need to increase the size and robustness of the landing gear. The main landing gear wire was increased to 1/4 inch diameter music wire. This heavier main gear was mounted external to the fuselage in order to maximize the interior space available for the payload. The standard nose gear was replaced with a dual strut gear. Pictures of the heavy duty landing gear and its mounting are shown in Fig. 3 and Fig. 4.

The engine servo and a heavy duty nose gear steering servo were moved to the front of the fuselage. The throttle servo can be seen mounted to the firewall in Fig. 4. The nose gear servo (which is not visible) is located directly behind the firewall. The elevator and rudder servos were moved to the rear of the fuselage as shown in Fig. 5. This method of mounting the servos is common practice on most of the larger ARF model aircraft.

The main changes in performance caused by the airframe modifications are due to the increased weight of the UAV over the trainer version. This primarily affects the maximum and minimum speeds of the aircraft. The speed range of the UAV is between 25–45 knots, while the trainer version is capable of 15–60+ knots. Structural performance issues

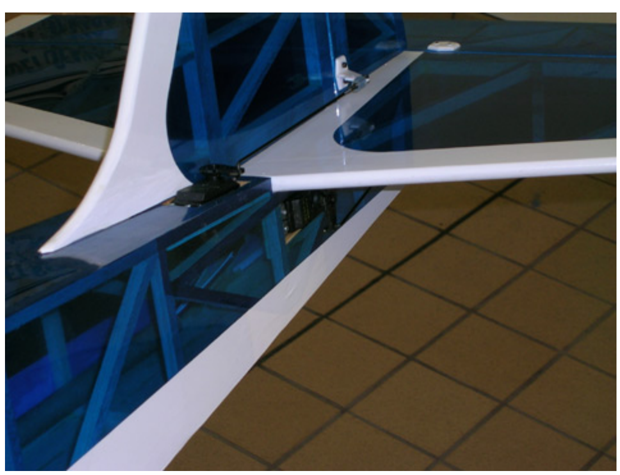

Fig. 5 Aft Elevator and Rudder Servo Mounting.

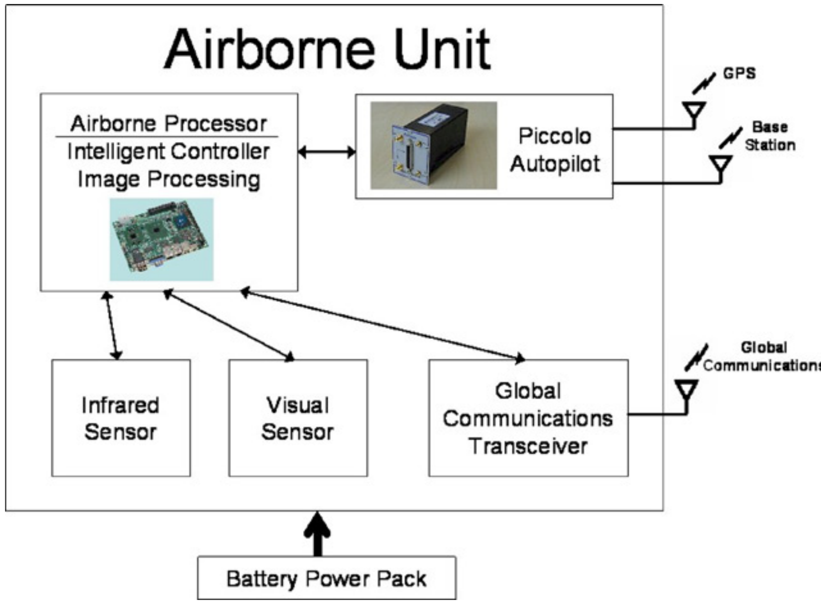

Fig. 6 Airborne platform subsystems.

due to aerobatic flight were not considered as there is no intention to maneuver the UAV aggressively. Furthermore, the wing is heavily overbuilt with a large aluminum spar.

The autopilot, Intelligent Controller, and power source work together to provide platform control as shown in Fig. 6.

1. Piccolo Plus Autopilot

A Cloud Cap Technology (CCT) Piccolo Plus autopilot is used for flight stability and autonomous control. This autopilot receives GPS input and Base Station communication via antennas located on the aircraft. The base station communications uses a 900 MHz radio link. The Piccolo autopilot is mounted in the aircraft using a CCT supplied mount that provides shock and vibration isolation. The mount is very near the center of gravity of the airplane. The autopilot is held in the mount with Velcro straps. The pictures in Fig. 7 show this installation. Fig. 8 and 9 show the GPS and telemetry antenna mounts. The GPS location under the aircraft's windshield allows a clear view of the sky

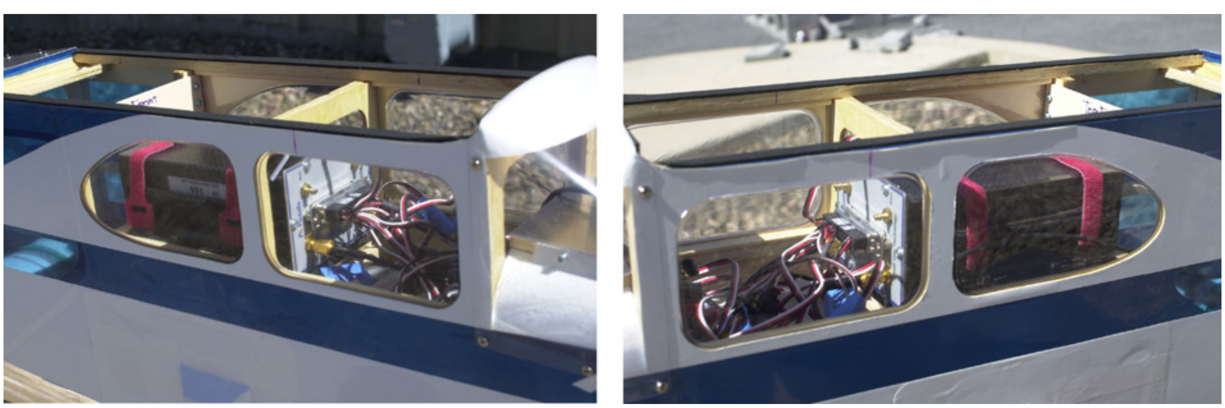

Fig. 7 Left and right views of the autopilot installation.

Fig. 8 GPS antenna mount under windshield.

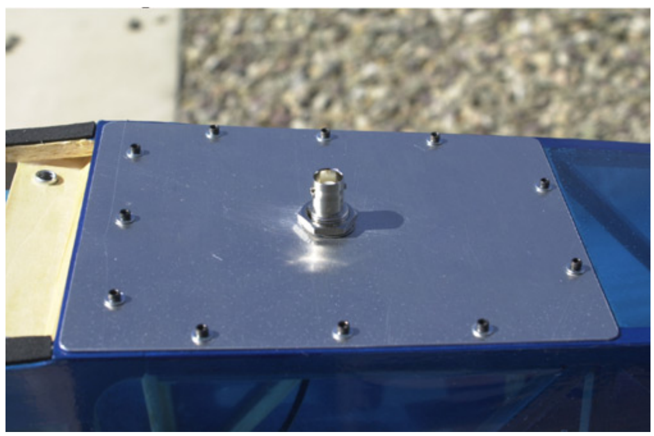

Fig. 9 Telemetry antenna mount aft of wing.

resulting in good GPS reception. The telemetry antenna location on top of the fuselage aft of the wing provides good range performance. Flights in excess of 0.6 miles have recorded full received signal strength.

2. Airborne Processor

The onboard computer or airborne processor board is an Ampro ReadyBoard 800 Single Board Computer (SBC). It is held in the aircraft on a mount developed on this project that is similar to the autopilot mount. Pictures of this unit and its approximate location along the fuselage length are shown in Fig. 10 and Fig. 11. The blue tape on the mount was used to hold the components of the mount together while it was being fit inside the fuselage. The base of the mount is fastened inside the fuselage with screws so that it can be removed for maintenance. The processor fits into slots in the mount and is held in place with nylon screws. It is easily removable.

Electromagnetic interference concerns were addressed as well. The Piccolo autopilot is enclosed in a carbon fiber box which provides EMI shielding. However, the onboard computer has no shielding which prevented the GPS system from getting a location fix and caused jittering in the tail empennage servos. The computer and GPS system operate on similar frequencies (1.4 GHz). The servo wires for the tail servos run past the computer, and servo jittering was noticed when the computer was on. Carbon fiber cloth was wrapped around a cardboard tube and mounted alongside the computer. The servo wires were passed through the tube which eliminated the servo jittering. Interference to the GPS antenna was solved in two different ways on two different UAVs. Our initial approach was to mount the GPS antenna as far away from the computer as possible, which meant mounting it on the tail. This solution eliminated

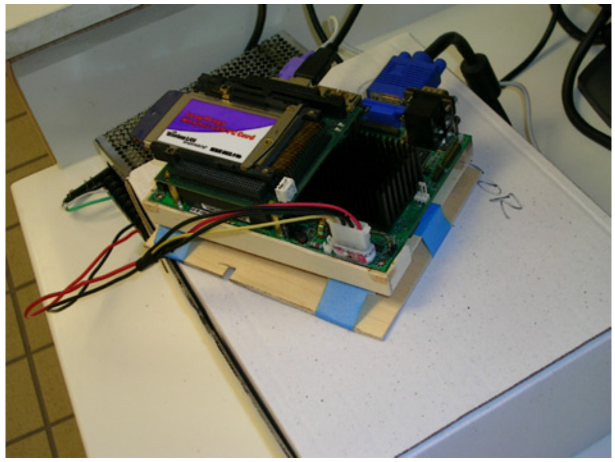

Fig. 10 Airborne processor in aircraft mount.

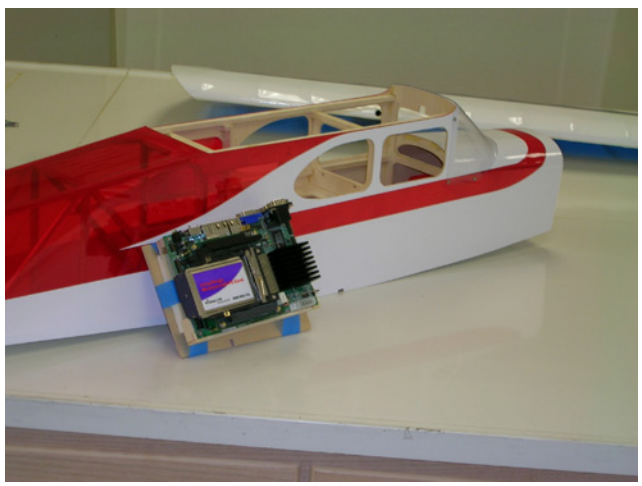

Fig. 11 Airborne processor and mount showing the approximate location in the aircraft.

all problems, however, required a bit more ballast weight in the nose to maintain a proper center of mass location. Therefore, during the build of the following UAV, the GPS antenna was kept as shown in Fig. 8, and several carbon fiber sheets were made up and placed around the fuselage experimentally to mitigate interference effects. The final solution was a carbon fiber sheet across the top of the fuselage extending from the mid point of the computer to the front of the Piccolo. This prevents interference to the GPS, and no degradation of GPS performance is observed with the computer on compared to having it off.

This unit contains a 1.4 GHz Intel Pentium M processor, with a maximum continuous power consumption of 12W at 5VDC. The Intelligent Controller software resides on this processor and provides overall mission control of the aircraft. The IC receives aircraft status information from the autopilot and onboard sensors as well as other ICs residing on partner UAVs. It uses this world view information to make an independent assessment of the mission and issues updated autopilot and sensor commands as needed. The IC also communicates with a human operator on the ground. A more detailed discussion of the IC follows in Sections III and IV.

Figure 12 shows the communications connections to and from the IC. RS232 serial communications is used between the IC and the Piccolo Plus Autopilot. Communications between ICs and between the ICs and a human operator on the ground are via an 802.11b ad-hoc network. An EnGenius 2511CD Plus wireless card is used, combined

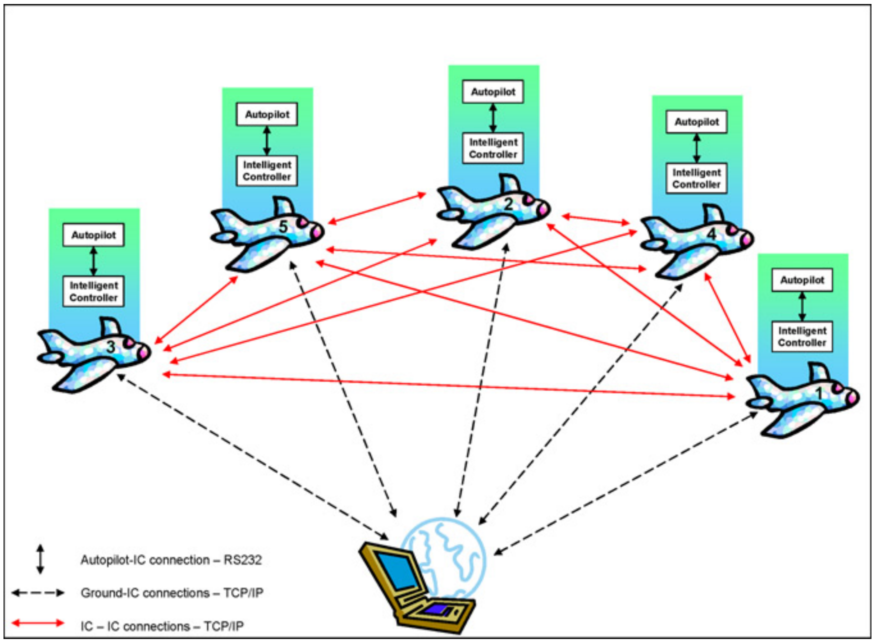

Fig. 12 Communications with the UAV IC.

Table 2 On-Board power requirements.

Power calculations			
Onboard computer		12	
Wireless card	+	3	
Total 5V power consumption	=	15	Watts
5V Regulator efficiency factor	÷	85%	
Total 5V Power required	=	17.6	Watts
Add Piccolo power (not regulated)	+	3.6	
Total power required	=	21.2	Watts
Battery energy capacity (11.1V, 4Ahr)	=	44.4	W-Hrs
Expected battery life (hrs)	=	2.1	Hrs
Actual (tested) battery life (hrs)	=	2.05	Hrs

with a 5dBi omni-directional antenna. Image processing from visual sensors (if present) is performed in a separate process on the airborne processor.

3. Power Source

A lightweight 11.1V 4Ahr Lithium Polymer battery pack provides power to both the IC and the Piccolo Plus Autopilot. Battery life calculations are listed in Table 2.

III. ARL/PSU's Intelligent Controller Architecture

ARL/PSU's Intelligent Controller architecture is the basis for the UAV Intelligent Controller.[12,16–20] Fig. 13 is a high level illustration of the controller.

The IC architecture is composed of two main modules: Perception and Response. The Perception module is where sensor data is analyzed, fused and interpreted to create an external world view. Using the world view generated in Perception for situational assessment, the Response module does mission planning and re-planning and carries out

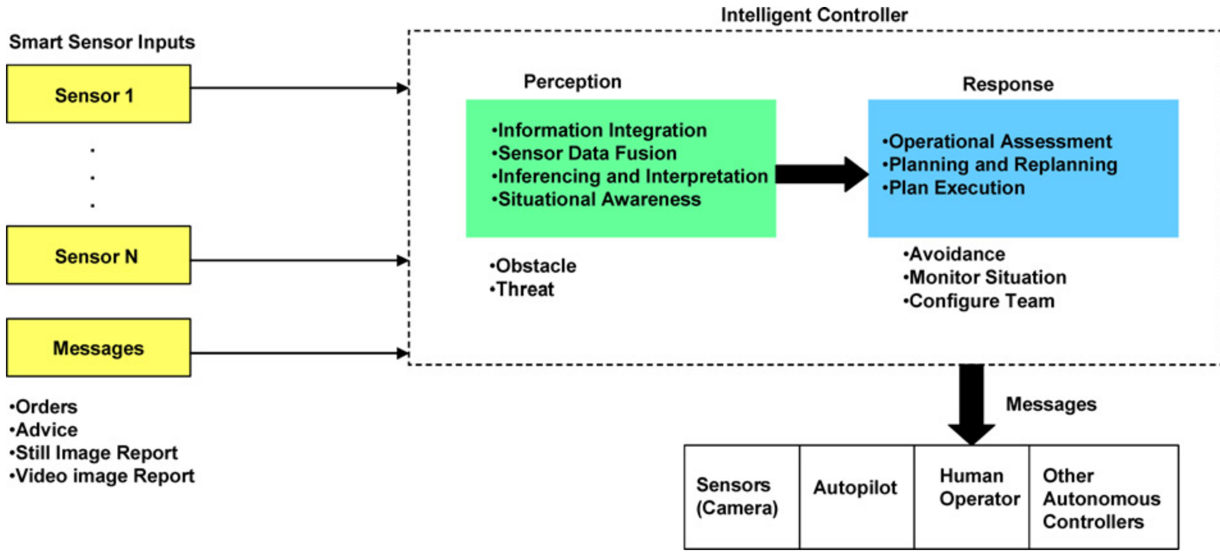

Fig. 13 ARL/PSU Intelligent Controller.

execution of the current plan. Output from Response includes commands and communications to external systems and vehicle subsystems. These modules are described in detail below.

A. Perception

The role of the Perception module is to create an internal representation of the external world relevant to the IC, using sensor data streams as inputs. A key capability of the Perception module is to make correct inferences and recognize the existence of properties in its internal models of external objects (Representational Classes) from incomplete and potentially erroneous input data. The Perception module contains data fusion algorithms and Continuous Inference Networks (CINets). The data fusion algorithms are responsible for fusing new sensor data so that the existing Representational Classes can be updated. The CINets infer properties or events, such as “obstacle”, by appropriately combining multiple pieces of information in an automatic recognition process and are described in more detail in.[16–19]

Sensor data is assumed to come from smart sensors, which is a term used to imply that some amount of sensor signal processing may have been applied before the IC receives the data. With sensing systems, for example, signal processing may involve analog-to-digital conversion, bandpass filtering, discrete Fourier transforms, beamforming, background noise level estimation, detection decisions, assembly of detection data, cross-beam data merging, and conversion of the data to an inertial coordinate system. In general, the sensing systems of an autonomous vehicle provide input to the IC along with messages from other vehicles and possibly a human in the loop.

An Input Interface module converts data streams external to the IC into forms that are usable by the Perception module. This data is accumulated in a buffer and released to Perception at discrete time intervals referred to as *processing cycles*. A processing cycle is determined by the amount of time required for an effector (e.g., a sensor controller) to complete a command and return data to the IC or possibly by a timer. In typical designs, a processing cycle is a quantum of time on the order of a second, about the human control frequency, although it can be any value required by the application and supportable by the speed of the processors involved.

B. Response

The role of the Response module is to plan and execute *in real time* a course of action to carry out a specific mission, given the situational awareness derived by the Perception module. It is constructed as a collection of autonomous operations referred to as Behaviors (agents), where each is capable of controlling the vehicle on its own, generating and executing plans in real time, monitoring progress, and adapting plan execution as appropriate. Other components of Response include the Mission Manager and Actions.

A Behavior monitors Perception for the existence of objects in its interest as indicated by certain high-level inferred properties. For example, an Investigate Behavior looks for objects classified as “of interest”. When this occurs, the Behavior will notify the Mission Manager that it is ready to take control. The Mission Manager has a definition of the current mission that enables it to determine the relative priority of a Behavior at any given time. It will select one or more Behaviors that are requesting control and turn operations over to each, as appropriate. The Mission Manager can also order Behaviors to take control when the Behavior does not depend on the existence of certain types of objects before it can function (e.g., a default Behavior such as “standby” or “loiter”).

Each Behavior has one or more Actions that generate the actual output commands and messages in reaction to the physical and inferred properties of the objects that were perceived. To control a vehicle or process, the IC sends Command messages to its effector subsystems. This is the output of the Response Module. These may include an autopilot for dynamic vehicle control or sensor controllers for sensor configuration to gather information as needed. Other output message types may be Orders, Queries, or Advisories to other ICs or humans in the team.

IV. UAV Intelligent Controller (IC)

The UAV IC provides mission control for the aircraft. This IC implementation was coded in the C++ programming language. Each UAV runs a copy of the same IC on its onboard processor. Properties of a specific UAV that the IC needs to know about (such as vehicle ID number, sensors used) are contained in a configuration file that is downloaded at startup. Examples of sensors include a digital camera and a webcam.

The capabilities of the UAV IC include autonomous operations for individual units as well as collaboration between UAVs (and possibly ground and sea autonomous vehicles). These capabilities include: Flight Path, Investigate, Standby, Coordinate Investigate, and Communicate. Each of these operations is implemented as an independent Behavior or agent that operates autonomously within its scope, and conducts real-time planning, analyzes the status of its execution, and adapts appropriately to the results of that analysis. This section describes the UAV IC in detail.

A. UAV IC Perception Module

The Perception Module is where the IC’s world view is internalized and stored in the form of Representational Classes (RCs). The UAV IC Perception Module and its relationship to the Response Module are illustrated in Fig. 14.

The UAV IC Input Interface accepts telemetry, control, and commands data from the Piccolo Plus Autopilot and formats it appropriately for use by the Perception Module, which maintains the relevant information in its own representation of the vehicle, a Self RC. This is used by the UAV IC Response Module to track the UAV’s location, attitude, and speed. Status data from partner UAVs containing similar information is received via an 802.11b ad-hoc network and maintained in Partner RCs. Orders from the ground (also via an 802.11b ad-hoc network) are stored as Standing Orders.

Visual sensor data, when present, is processed on the onboard processor and sent to the UAV IC in the form of a report. The UAV receiving the report will then broadcast it to its partner UAVs. The IC needs to determine if the object detected is an “object of interest”, i.e., an object worth investigating further. The processing structure used to determine this is the CINet.[16–18] CINets establish confidence factors (values between 0 and 1) for the set of inferred properties defined for a particular class. Since physical variables are usually continuous, classifiers (the nodes in

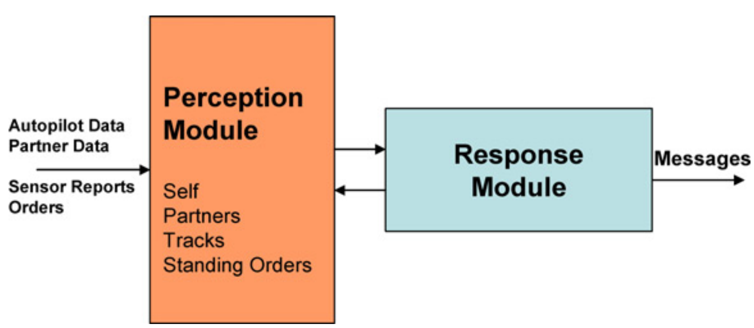

Fig. 14 UAV IC Perception and Response.

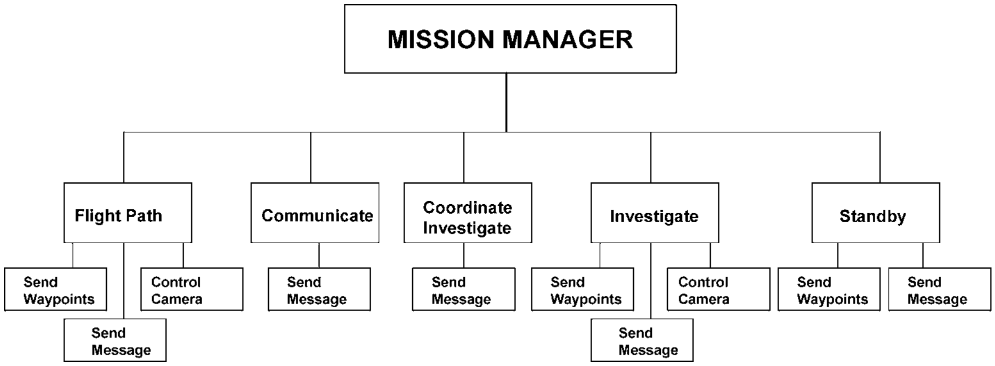

Fig. 15 UAV IC Response Module.

the CINet) need to be based on continuous logic to avoid loss of information, which precludes the use of binary (true-false) logic. Object properties and confidence factors are stored in Track RCs.

B. UAV IC Response Module

The Behaviors and Actions that comprise the UAV IC Response Module are shown in Fig. 15. Behaviors are at the middle level and Actions at the lowest level, where they generate the actual output commands and messages to vehicle subsystems and partner UAVs. The Mission Manager mediates amongst Behaviors requesting to be enabled. The Mission Manager uses a priority scheme to decide which requesting Behavior(s) should take control. The priority scheme is based on the current mission and external conditions. A mission is defined as a set of orders and is usually downloaded at startup but may be transmitted during mission execution to modify the mission's details on-the-fly. Some orders the IC understands include "transit-through-provided-waypoints" and "go-to-standby-mode". The priority scheme used by the Mission Manager is the same for all missions defined for the current flight system. The Communicate and Coordinate Investigate Behaviors are always granted control when they request it. These Behaviors involve communications only and do not control the aircraft; therefore, there is no conflict of vehicle resources and either can operate in conjunction with Behaviors that do control the vehicle. For those other Behaviors, the priority order is: Standby, Investigate, and Flight Path. The Behaviors are described below in detail.

1. Standby

Standby is the default Behavior for the UAV IC. The UAV IC starts off in the Standby Behavior until a human operator on the ground issues the command for the IC to read the mission file and send a flight plan to the autopilot. While in the Standby Behavior, the UAV is either under manual control or is flying a default flight plan under autopilot control. This default flight plan is initially loaded to the autopilot through Cloud Cap Technology's Operator Interface and is also downloaded to the IC at startup. The IC will not send any commands to the autopilot when in this behavior.

A human operator on the ground can issue a Standby Order to the UAV IC. When the Standby Behavior sees this order, it requests to be enabled. The Mission Manager will always grant Standby control when it is requested. Standby deletes all user defined flight plans, sets the aircraft speed to a "standby speed", and sends the UAV back to the default flight plan, all via messages to the autopilot. Once the UAV is back in the default flight plan, the IC will not send any commands to the autopilot while in Standby.

2. Communicate

The Communicate Behavior looks for incoming messages that require a response and sends out an appropriate reply. For example, if a Request For Status message is received from another UAV, the Communicate Behavior will reply with a Status message containing the UAV's ID number, location, altitude, speed, availability, and sensor capabilities. Because the Communicate Behavior can operate in conjunction with other Behaviors, it is always granted control by the Mission Manager when requested.

3. Flight Path

The Flight Path Behavior requests control from the Mission Manager when a Flight Path Order is the current Standing Order. When enabled, Flight Path sends a user defined flight plan to the autopilot and monitors waypoint progress. The IC remains in this Behavior until another order is sent from ground or an investigation occurs.

4. Coordinate Investigate

The Coordinate Investigate Behavior is used to coordinate partners in investigating objects of interest. It utilizes a negotiation scheme to help determine, select, and request assistance from partner UAVs. The Coordinate Investigate Behavior requests to be enabled when a UAV detects an object that is classified as an "object of interest" from its own visual sensor. This UAV will take the lead in selecting a partner to investigate the object; if there are no partner UAVs, this UAV will further investigate the object itself. Negotiations between partners to collaborate on investigating an "object of interest" involve a series of message exchanges, culminating in one (or no) UAV going to investigate the object.

5. Investigate

The Investigate Behavior looks for a track that has been marked for the UAV to investigate (as a result of investigation coordination) and requests control from the Mission Manager if such a track exists. Upon activation, it will send the autopilot an "investigate speed" and the coordinates of the object of interest. The Investigate Behavior monitors the progress of the UAV as it approaches the object. When the object is reached, the IC sends a command to activate the UAV's visual sensor (if present) and to orbit the target for a predefined amount of time. When the UAV finishes orbiting, it broadcasts a message to all its partners notifying them that the investigation is complete. If there are no other objects to investigate, the Mission Manager returns control to the previous Behavior (Flight Path), and the UAV resumes its previous flight plan and speed.

V. Intelligent Autonomous Flight

Initial testing of the UAV IC and Piccolo Plus Autopilot is done using Cloud Cap Technology's hardware-in-the-loop (HIL) simulator. The simulator allows the aircraft control laws and mission functionality to be tested without the inherent risk associated with flight test. Once the system is thoroughly tested in HIL simulations, it is evaluated and tuned in flight with minimal risk. These processes are described below.

A. HIL Simulator Testing

All system hardware, the airborne unit, and the ground base station functions as part of HIL testing. The Cloud Cap Technology supplied simulation software runs on a separate computer, the Hardware-in-the-Loop PC in Fig. 16, and communicates with the autopilot via a Controller Area Network (CAN) controller. A special cable harness replaces the flight harness. The autopilot sends servo control information and accepts external sensor data over the CAN bus. The computer running the simulator closes the loop by reading the servo commands, applying them to an aircraft dynamics model, calculating new sensor and GPS data, and sending it to the autopilot. Since the GPS signals are simulated, the GPS is not connected.

The open-source FlightGear simulation software[21] is used to visualize the state of the aircraft. FlightGear is run on the same computer as the simulation software and is used only for aircraft visualization; it is not part of the Cloud Cap Technology simulator. As such, it is configured to accept the simulation state packets instead of using its own dynamics model.

1. Configuring the Simulation Aircraft Model

Hardware-in-the-loop simulations are crucial, as they allow autopilot gains to be tuned and mission plans to be executed without endangering the aircraft. Flight dynamics simulation software is packaged with this autopilot system. Basic dimensions and weights as well as airfoil and engine performance data are provided as input to the simulation software, which then numerically estimates stability and control characteristics of the aircraft and integrates the six degree-of-freedom equations of motion. Basic dimensions, such as fuselage length and empennage area, were measured from an assembled SIG Kadet Senior aircraft. The simulator calculates moments of inertia using

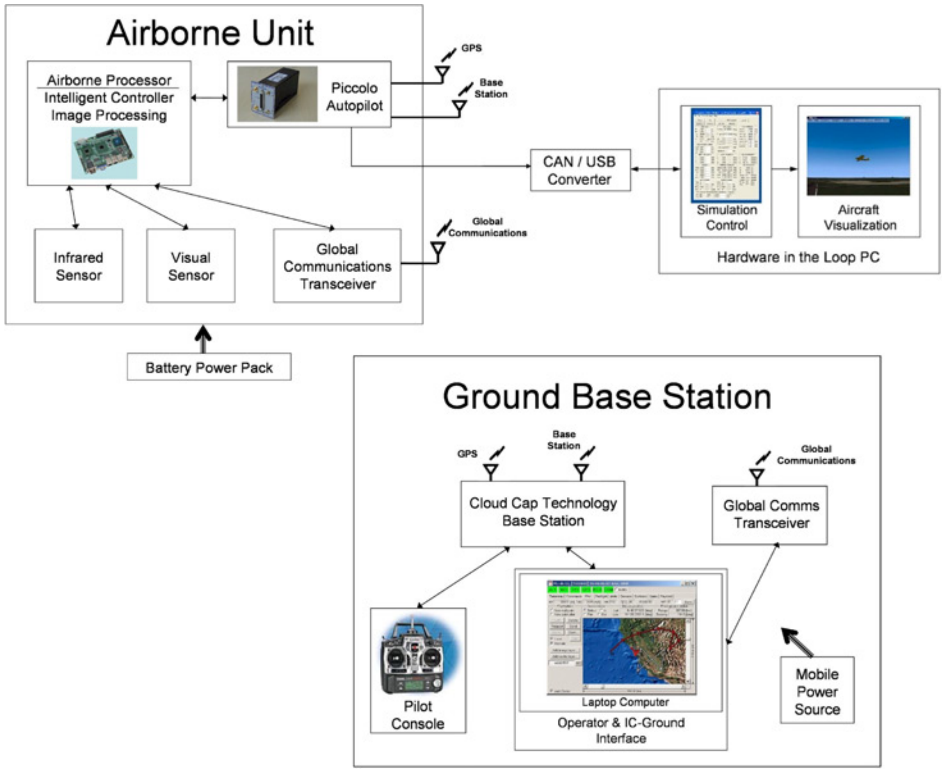

Fig. 16 Hardware-in-the-loop, HIL, configuration.

the dimensions and the weights of the various components. Dimensions such as the wing chord and span are used in the calculation of aerodynamic forces.

The SIG Kadet Senior has a Clark-Y airfoil on the main wing. XFOIL, a program created by Drela,[22] was used to analyze the airfoil performance characteristics at the expected Reynolds number. The Reynolds number is defined by the speed, U, chord, c, and kinematic viscosity, ν. ($\mathrm{Re} = \mathrm{Uc}/\nu$). Average flight conditions are expected to be 400 feet altitude and 40 miles per hour (59 feet per second). The chord of the aircraft is 1.2 ft. The Reynolds number is roughly 450,000.

Using XFOIL, the two-dimensional lift coefficient $\mathrm{C}_{\mathrm{l}\alpha}$ as a function of the angle of attack α was computed. The three-dimensional lift coefficient slope, $\mathrm{C}_{\mathrm{L}\alpha}$, was calculated from the relationship between the two-dimensional lift coefficient slope and the aspect ratio of the wing, *AR,* which is 5.5.

$$\mathrm{C}_{\mathrm{L}_\alpha} = \mathrm{C}_{\mathrm{l}_\alpha} \frac{AR}{AR + \frac{2(AR+4)}{AR+2}} \tag{1}$$

The aspect ratio correction incorporates three-dimensional effects, which reduce the lift coefficient of the wing for a given angle of attack. The zero-lift angle of attack, α_0, was retained from the two-dimensional plot of lift coefficient and angle of attack. The lift coefficient in the linear region is then defined by the following equation.

$$\mathrm{C}_{\mathrm{L}} = \mathrm{C}_{\mathrm{L}_\alpha}(\alpha - \alpha_0) \tag{2}$$

Stall effects at the higher angles of attack were modeled using the two-dimensional results that XFOIL provided. Fig. 17 shows the two-dimensional lift coefficient predicted using XFOIL and the three-dimensional lift coefficient calculated from the above equations. The three-dimensional lift coefficient was used in the Cloud Cap Technology simulator to calculate lift forces.

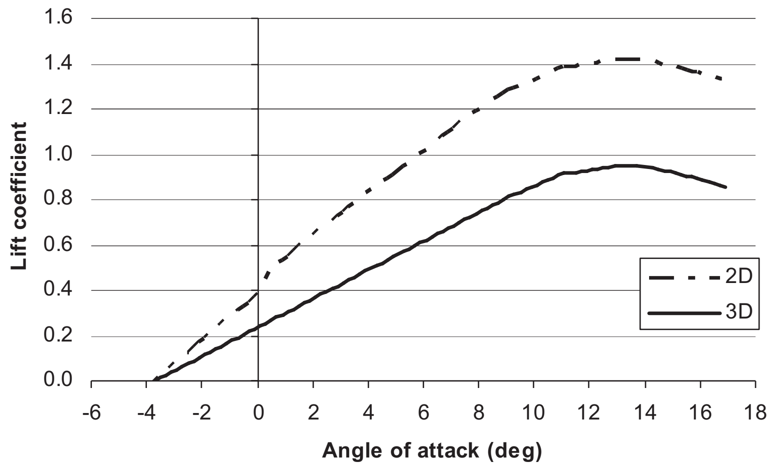

Fig. 17 Lift coefficient.

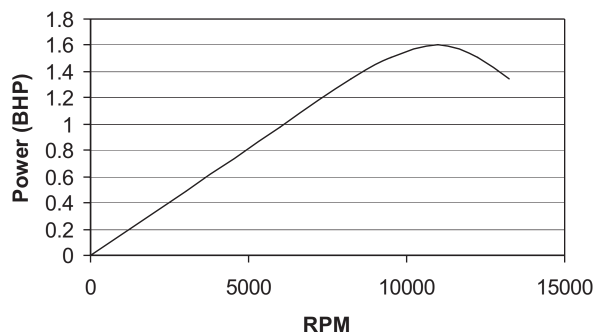

Fig. 18 Engine power.

The three-dimensional drag coefficient, C_D, of the wing are then calculated with the following relationship between the profile drag, C_d, and the induced drag. Due to the rectangular wings of the SIG Kadet Senior, a span efficiency factor, e_s, of 0.9 was used.

$$C_D = C_d + \frac{C_L^2}{\pi \cdot AR \cdot e_s} \tag{3}$$

The pitching moment coefficient does not noticeably change from two dimensions to three dimensions, and the pitching moment provided by XFOIL was retained.

The O.S. FS-91 II Surpass with Pump engine was also modeled in the simulation. The manufacturer provided only the maximum power output (1.6 Brake Horse Power) and the corresponding rotational speed (11,000 RPM). This data point was fit to the power curve (Fig. 18) on which most four-stroke engines operate.

2. Tuning Autopilot Gains

The autopilot uses eight different feedback loops with proportional, integral, derivative (PID) compensation. The longitudinal and lateral/directional modes are decoupled. Not all of the available loops were used in this application. Table 3 lists the loops used and provides a description of each.[23]

The autopilot gains are tuned one feedback loop at a time. The aircraft is trimmed and the turn rate control loop is turned on. This loop uses roll angle feedback with proportional and integral compensation to control turn rate using the ailerons. The gains are adjusted from their default values until perturbations in bank angle are corrected and turn rate commands are followed without excess oscillation.

Table 3 Summary of autopilot feedback loops.

Loop	Inputs	Outputs	Comp.	Notes
Dynamic pressure	Dynamic pressure	Elevator	PID	Maintains a commanded dynamic pressure
Altitude	Static pressure	Throttle	PID	Maintains a commanded altitude
Tracker	GPS	Turn rate command	PD	Drives the turn rate loop to achieve desired track targets
Roll	Roll angle	Aileron	PI	Turn rate control and roll angle disturbance rejection
Pitch	Pitch angle	Elevator	PD	Damps out pitch oscillations
Yaw	Yaw rate	Rudder	P	Damps out yaw oscillations

Building upon the turn rate control loop, the airspeed, pitch damper, yaw damper, and altitude loops are sequentially tuned in a similar manner starting from their default values. Finally, the tracker, which is responsible for following a path defined by waypoints, is tuned after the rest of the autopilot loops are operational.

3. UAV IC HIL Verification and Testing

Using the HIL simulator configuration provides a means for testing the communications between the UAV IC and the Piccolo Plus Autopilot on the ground along with the IC functionality. A screen capture from the Operator Interface map of a typical test is shown in Fig. 19.

Once up in the air and flying autonomously, the UAV (the small triangular object with a line coming out the top near the top center of Fig. 19) is sent a default flight plan through the Operator Interface. This is the large triangle in the figure (waypoints 10, 11, and 12). The IC is running on the airborne processor, accepting telemetry, command, and control data from the autopilot. The IC is not sending commands at this time to the autopilot and is in the Standby Behavior.

When the human operator sends the Startup Command from the IC-Ground Interface to the UAV IC, the IC will read the mission file, form a flight plan and send it to the autopilot. This is the lawnmower-like pattern shown in Fig. 19 (waypoints 25–30). The IC is now in the Flight Path Behavior.

During its flight, the UAV IC receives a simulated camera report (from a human operator on the ground, not from an actual onboard camera). The IC Perception Module determines that the object is “of interest.” The Coordinate Investigate Behavior requests, and is granted, control; because there is only one UAV, it determines that this UAV

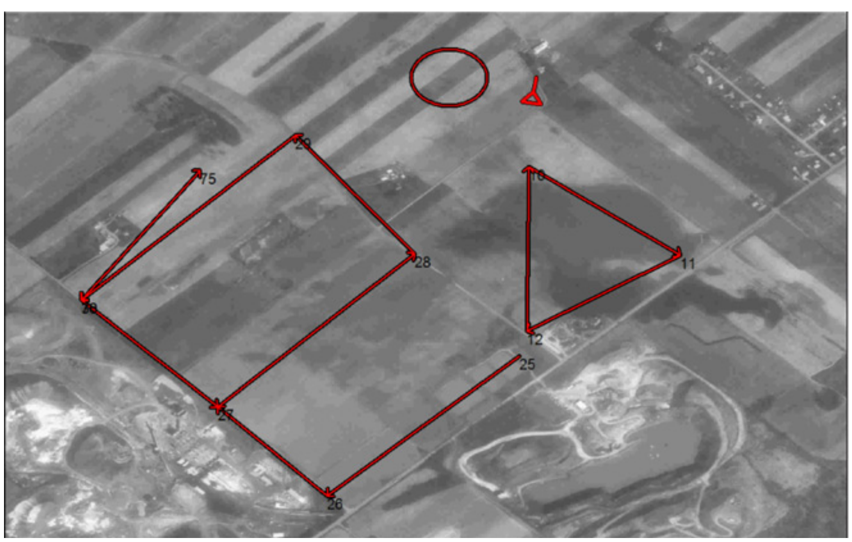

Fig. 19 Screen capture of HIL simulation demonstration.

should investigate the object. The Investigate Behavior of the IC requests, and is granted, control and instructs the autopilot to send the UAV to the location of the object and orbit. This is shown by the circle at the top of Fig. 19.

When the UAV is done orbiting, the Investigate Behavior is completed and the Mission Manager gives control back to the Flight Path Behavior. The UAV returns to the point where it deviated from its flight plan (waypoint 75) and continues to finish the plan (waypoint 30). It will remain in this flight plan (and the Flight Path Behavior) until otherwise ordered.

B. Flight Testing – UAV and Autopilot

Initial flight tests have been conducted to test the autopilot gains and validate the simulation model. For the first test, the initial flight test procedure defined in the Piccolo documentation[24] was used to verify the gains for each feedback loop. This procedure starts with only the turn rate loop enabled and builds up to full control loop by loop. At each step, perturbations from the pilot and Operator Interface commands are used to verify stability. Due to hardware-in-the-loop simulation, the initial flight was completed with no problems and the simulation derived autopilot gains only required minor changes. Of course, the simulator is not a perfect model of the actual UAV, so some gain changes are expected. Changes to gains were made to reduce any in-flight oscillation observed from the ground. For the second flight, bank angle limits were slowly increased to achieve maximum turn rate while remaining in stable flight. The Piccolo autopilot uses longitudinal control decoupled from lateral/directional control; this requires a roll angle limit to maintain stability. Table 4 gives a summary of how the gains were changed compared to the default Piccolo gains after simulator and flight testing. Table cells marked with a '-' represent no change over the default values. It is also important to note that these gains were chosen mainly based on observation of the airplane while flying and, to a lesser extent, examination of the telemetry post flight.

Fig. 20 shows a photograph of the flying field taken from the UAV, which was taken at 375 ft. altitude and 40% throttle setting. The automobiles and people are clearly visible. Fig. 21 shows the runway (red rectangle) along with our GPS-measured ground tracks. The aircraft was given four waypoints located in roughly a rectangular pattern.

The need to investigate the actual versus commanded turn rate was noted in the early autonomous flights. In order to efficiently search specified regions of the ground, it is important to know the turn-rate limits of the aircraft. During a subsequent flight, a set of left and right turns were performed. The bank angle limit was set to 30 degrees for this test. Fig. 22 shows the turn-rate data. The "Ap Cmd" line in the plot indicates when the autopilot was enabled. A value of zero indicates manual control and a value of 10 indicates autopilot control. Note that $+/-$ 20 and $+/-$ 30 deg/sec turns were commanded. The total heading change is shown in Fig. 23.

One thing to note in Fig. 22 is the bank angle limit of 30 degrees is reached in all cases, thus limiting turn rate. Table 5 summarizes the turn-rate results, which shows a maximum turn rate of approximately 18 deg/sec to the right and 15 deg/sec to the left for the airspeed and bank angle limits used in this test. Sharper turns might be achieved by increasing bank angle limits, but these tests should be performed cautiously and incrementally to ensure that autopilot stability is maintained and that the performance limits of the aircraft are not exceeded. The asymmetry in achieved turn rate is likely due to small asymmetries in the airplane from imperfect assembly and/or the effect of propeller torque.

Table 4 Summary of gain changes compared to default gains.

	Proportional	Integral	Derivative
Roll command	+50% for faster bank response	–	Set to 0.05 to add damping
Airspeed command	+55% for faster response	–	–
Altitude command	Set to zero, only seemed to cause noise in the throttle command	−60% to reduce pitch/altitude oscillation	−50% to reduce pitch/altitude oscillation
Pitch damper	–	–	–
Tracker	+100% for more aggressive turn commands	–	+100% for more aggressive turn commands

Fig. 20 Image from UAV of flying field.

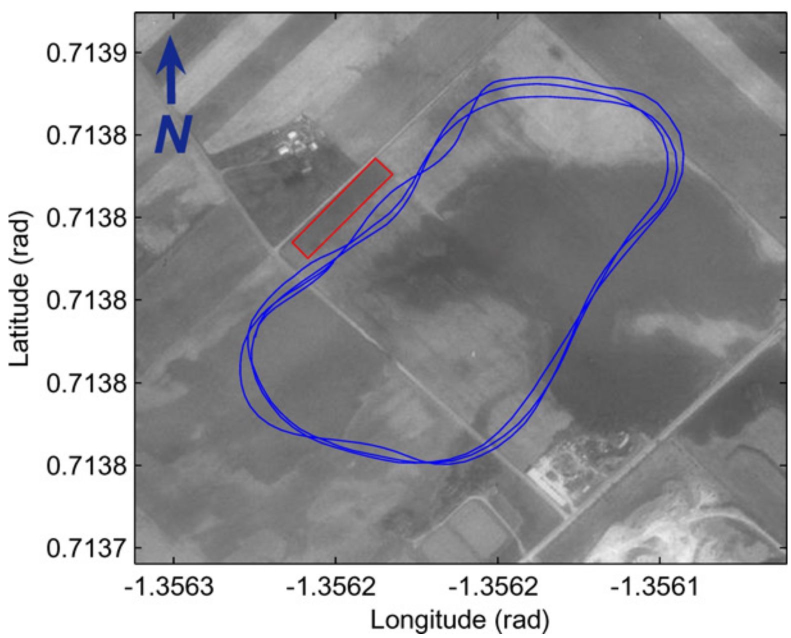

Fig. 21 Ground track of aircraft (blue line) and aircraft runway (red rectangle).

C. Flight Testing – UAV and the Intelligent Controller

Successful flight tests recreating the UAV IC HIL scenario described above have been flown. In addition, a flight test demonstrating the collaborative capabilities of the UAV IC was successfully conducted in the Fall of 2006. Two UAVs coordinated the investigation of an object through negotiations, resulting in one UAV deviating from (and ultimately returning to) its flight plan to orbit the object while the other UAV remained in its flight plan. No

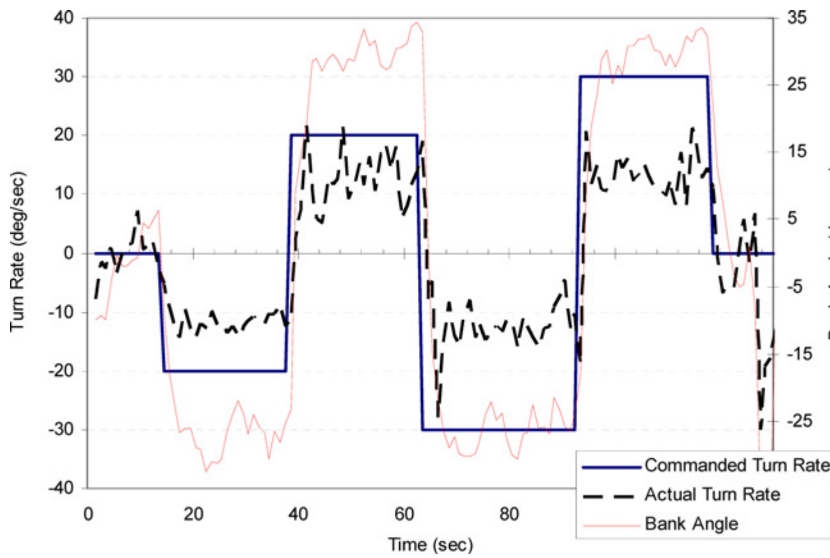

Fig. 22 Time history of turn commands and roll angle.

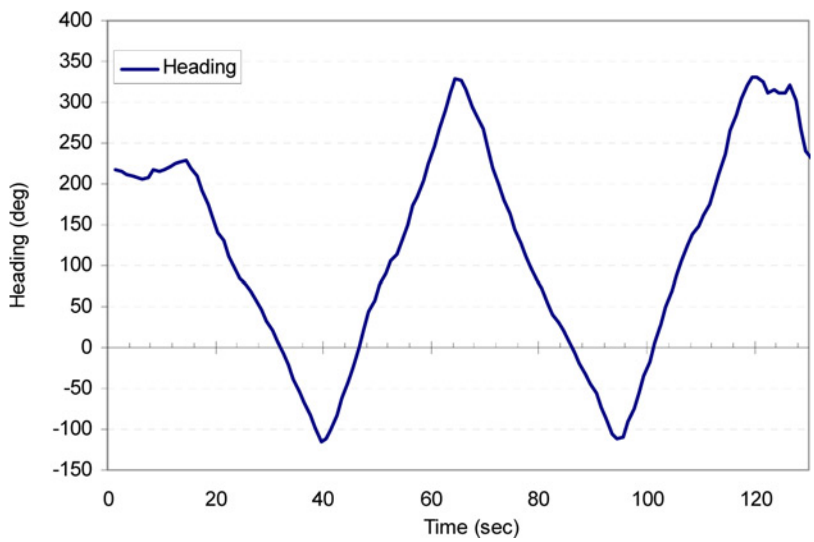

Fig. 23 Time history of heading.

Table 5 Commanded vs. Measured turn rates (degrees per second).

Commanded turn rate	−20	20	−30	30
Measured turn rate	−13.5	17.3	−14.7	18.2

onboard sensors were used for this demonstration; a camera report was simulated from the ground as in the HIL testing. Details of this flight test can be found in.[25]

VI. Conclusions

ARL/PSU's experience in unmanned vehicles contributed to the development of a robust, universal architecture for the design of intelligent autonomous vehicles. Combined with the experience of researchers in the PSU Department of Aerospace Engineering, an intelligent autonomous UAV flight system has been successfully developed and tested. Initial demonstrations of a UAV flying under flight control of a Piccolo Plus Autopilot were conducted. Hardware-in-the-loop simulations and subsequent flight tests with the Intelligent Controller providing mission control have demonstrated that the system presented in this paper is a suitable test bed for multi-vehicle collaborative control. Future flights involving multiple UAVs and coordinating ground vehicles are planned. By using the intelligent UAV flight system, autonomous UAV control can be effectively studied and evaluated.

References

[1]"Unmanned Aircraft Systems Roadmap," Office of the Secretary of Defense, Washington, D.C., August 2005. (http://www.acq.osd.mil/uas/)

[2]White, D.A. and Sofge, D.A., *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, Van Nostrand Reinhold, Sydney, 1992.

[3]Meystel, A.M. and Albus, J., *Intelligent Systems: Architecture, Design, Control,* Wiley, New York 2001.

[4]Brooks, R.A., *Cambrian Intelligence: The Early History of the New AI*, MIT Press, Cambridge, 1999.

[5]Arkin, R.C., *Behavior-Based Robotics*, MIT Press, Cambridge, 1998.

[6]Bekey, G.A., *Autonomous Robots: From Biological Inspiration to Implementation and Control,* MIT Press, Cambridge 2005.

[7]Newell, A., *Unified Theories of Cognition*. Harvard Univ. Press, Cambridge, 1990.

[8]Long, L.N., Hanford, S.D., Janrathitikarn, O., Sinsley, G.L.. and Miller, J.A., "A Review of Intelligent Systems Software for Autonomous Vehicles," presented at the IEEE Symposium Series on Computational Intelligence, Honolulu, Hawaii, April 1–5, 2007.

[9]Evans and Messina, E.R. "Performance Metrics for Intelligent Systems," Proceedings of the 2000 PerMIS Workshop, Gaithersburg, MD, Aug 14–16, 2000, pp. 101–104.

[10]AIAA Intelligent Systems Technical Committee, "Intelligence in Aerospace," *Journal of Aerospace Computing, Information, and Communication*, Vol. 1, No. 12, 2004, pp. 484–485.

[11]Wray, R.E. and Chong, R.S., "Comparing Cognitive Models and Human Behavior Representations: Computational Tools for Expressing Human Behavior," to appear *Journal of Aerospace Computing, Information, and Communication*, 2007.

[12]Stover, J.A. and Kumar, R., "A Behavior-based Architecture for the Design of Intelligent Controllers for Autonomous Systems," IEEE International Symposium on Intelligent Control/Intelligent Systems and Semiotics, Cambridge, MA, Sept. 15–17, 1999, pp. 308–313.

[13]Johnson, E.N., Proctor, A.A., Ha, J., and Tannenbaum, A., "Development and Test of Highly Autonomous Unmanned Aerial Vehicles," *Journal of Aerospace Computing, Information, and Communication*, Vol. 1, No. 12, 2004, pp. 486–501.

[14]Chalmers, R.W., Scheidt, D.H. Neighoff, T.M., Witwicki S.J., and Bamberger, R.J., "Cooperating Unmanned Vehicles," AIAA 1st Intelligent Systems Technical Conference, September 20–22, 2004, pp. 1–8.

[15]Valenti, M., Schouwenaars, T., Kuwata, Y., Feron, E., How, J., and Paunicka, J., "Implementation of a Manned Vehicle—UAV Mission System," AIAA Paper No. 2004-5142, 2004.

[16]Stover, J.A. and Gibson, R.E., "Continuous Inference Networks for Autonomous Systems," IEEE Conference on Neural Networks for Ocean Engineering, Aug. 17, 1991, pp. 177–183.

[17]Stover, J.A. and Gibson, R.E., "Modeling Confusion for Autonomous Systems," SPIE, Science Artificial Neural Networks, 1710, 1992, pp. 547–555.

[18]Stover, J.A., Hall, D.L., and Gibson, R.E., "A Fuzzy-Logic Architecture for Autonomous Multisensor Data Fusion,"IEEE Transactions on Industrial Electronics, Vol. 43, 1996, pp. 403–410.

[19]Stover, J.A., and Gibson, R.E., "Controller for Autonomous Device," US Patent #5,642,467, Issued June 1997.

[20]Weiss, L., "Intelligent Collaborative Control for UAVs," AIAA Paper No. 2005-6900, AIAA InfoTech@Aerospace Conference, Washington D.C., Sept. 26–29, 2005.

[21]http://www.flightgear.org/

[22]http://raphael.mit.edu/xfoil/

[23]Vaglienti, B., Hoag, R., and Niculescu, M., "Piccolo System User Guide," Cloud Cap Technology [online], URL: http://www.cloudcaptech.com/ [cited 7 September 2004].

[24]Vaglienti, B., Hoag, R., and Miller, T., "Piccolo Aircraft Integration Guidelines," Cloud Cap Technology [online], URL: http://www.cloudcaptech.com/ [cited 7 September 2004].

[25]Sinsley, G.A., Miller, J.A., Long, L.N., Geiger, B.R., Niessner A.F. and Horn J.F., "An Intelligent Controller for Collaborative Unmanned Air Vehicles," presented at the IEEE Symposium Series on Computational Intelligence, Honolulu, Hawaii, April 1–5, 2007.

Ella Atkins
Associate Editor