Unmanned aerial vehicles (UAVs) are becoming vital warfare and homeland-security platforms because they significantly reduce costs and the risk to human life while amplifying warfighter and first-responder capabilities. These vehicles have been used in Iraq and during Hurricane Katrina rescue efforts with some success, but there remains a formidable barrier to achieving the vision of multiple UAVs operating cooperatively. Numerous researchers are investigating systems that use multiple autonomous agents to cooperatively execute these missions [1]–[4]. However, little has been said to date about how to perform multiday autonomous system operations. Autonomous mission systems must balance vehicle capability, reliability, and robustness issues with task and mission goals when creating an effective strategy. In addition, these systems have the added responsibility of interacting with numerous human operators while managing both high-level mission goals and individual tasks.

To investigate and develop unmanned vehicle systems technologies for autonomous multiagent mission platforms, we are using an indoor multivehicle testbed called Real-time indoor Autonomous Vehicle test ENvironment (RAVEN) to study long-duration multivehicle missions in a controlled environment. Normally, demonstrations of multivehicle coordination and control technologies require that multiple human operators simultaneously manage flight hardware, navigation, control, and vehicle tasking. However, RAVEN simplifies all of these issues to allow researchers to focus, if desired, on the algorithms associated with high-level tasks. Alternatively, RAVEN provides a facility for testing low-level control algorithms on both fixed- and rotary-wing aerial platforms. RAVEN is also...
being used to analyze and implement techniques for embedding the fleet and vehicle health state (for instance, vehicle failures, refueling, and maintenance) into UAV mission planning. These characteristics facilitate the rapid prototyping of new vehicle configurations and algorithms without requiring a redesign of the vehicle hardware. This article describes the main components and architecture of RAVEN and presents recent flight test results illustrating the applications discussed above.

BACKGROUND
Various research platforms have been developed to study UAV concepts [5]–[23]. The BErkeley AeRobot (BEAR) project features a fleet of commercially available rotary-wing and fixed-wing UAVs retrofitted with special electronics. These vehicles have been used in applications such as autonomous exploration in unknown urban environments and probabilistic pursuit-evasion games [20], [22]. In the Aerospace Controls Laboratory at the Massachusetts Institute of Technology (MIT), an outdoor testbed consisting of a fleet of eight fixed-wing autonomous UAVs provides a platform for evaluating coordination and control algorithms [12], [17]. Similarly, researchers in the Multiple AGent Intelligent Coordination and Control (MAGICC) Lab at Brigham Young University have built and flown small fixed-wing UAVs to perform multivehicle experiments outdoors [19]. These planes are launched by hand and track waypoints autonomously.

Likewise, the DragonFly project at Stanford University’s Hybrid Systems Laboratory uses a heavily modified fixed-wing model aircraft [7], [21]. The objective of this platform is to provide an inexpensive capability for conducting UAV experiments, ranging from low-level flight control to high-level multiple aircraft coordination. Similarly, to demonstrate new concepts in multiagent control on a real-world platform, the Hybrid Systems Laboratory developed the Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC). STARMAC is a multivehicle testbed consisting of two quadrotor UAVs that autonomously track a given waypoint trajectory [10]. Quadrotors are used in STARMAC due to their convenient handling characteristics, low cost, and simplicity.

In addition, several indoor multivehicle testbeds have been constructed to study multiagent activities. For example, the HOvercraft Testbed for DEcentralized Control (HOTDEC) Platform at the University of Illinois Urbana-Champaign is a ground-vehicle testbed used for multivehicle control and networking research [23]. At Vanderbilt University, researchers have built the Vanderbilt Embedded Computing Platform for Autonomous Vehicles, which supports two flight vehicles in an indoor environment [18]. Also, the UltraSwarm Project at the University of Essex is designed to use indoor aerial vehicles to examine questions related to flocking and wireless cluster computing [11]. Likewise, researchers at Oklahoma State University use the COMET (COoperative Multivehicle Testbed) ground-vehicle testbed to implement and test cooperative control techniques both indoors and outdoors [6].

Some of these testbeds have limitations that inhibit their utility for investigating health management questions related to multiday, multiagent mission operations. For example, most outdoor UAV test platforms can be flown safely only during daylight hours. In addition, many outdoor UAV test platforms must be flown during good weather conditions to avoid damage to vehicle hardware. These outdoor UAVs also typically require a large support team, which makes long-term testing logistically difficult and expensive.

In contrast, RAVEN is designed to examine a wide variety of multivehicle missions using both autonomous ground and air vehicles. The facility uses small electric helicopters and airplanes and can accommodate up to ten air vehicles at the same time. At the heart of the testbed is a global metrology motion-capture system, which yields accurate, high-bandwidth position and attitude data for all of the vehicles.

Although RAVEN’s indoor global metrology system is expensive, there are significant advantages to operating indoors. Since the position markers are lightweight, the motion-capture system can sense vehicle position and attitude without adding substantial payload. Thus, RAVEN does not require significant modifications to off-the-shelf, low-cost, radio-controlled (R/C) vehicle hardware. This setup allows researchers to conduct experiments with large teams (ten units) of inexpensive flying vehicles (≈US$1000 each). In addition, one operator can set up and flight test multiple UAVs in under 20 min. RAVEN thus facilitates rapid prototyping of multivehicle mission-level algorithms at a fraction of the cost needed to support multiday, multivehicle external flight demonstrations.

The motion-capture metrology system can be used to obtain navigation and sensing data for many independent vehicles both in ideal and degraded form to simulate GPS obstructions. The motion-capture metrology data can then be augmented with onboard sensing to extend the scope of the flight experiments being performed.
This system architecture is designed to separate the mission- and task-related components of the architecture from RAVEN's vehicle infrastructure. Therefore, adjustments in the mission, task, and trajectory planning components of the system do not require changes to the test-bed's core software infrastructure and can be made in real time.
The primary flight test range, which is located on the MIT campus, is approximately 10 m by 8 m by 4 m. The environmental conditions of the indoor facility can be carefully controlled for flight testing. Conditions can range from ideal to wind induced, for example, by blowers or fans. The controlled environment is available 24/7 since it is not dictated by weather, day/night cycles, or other external factors such as poor visibility. In addition, flight experiments using small-scale, electric-powered vehicles in an indoor range can be carefully monitored and set up in a safe way by establishing flight procedures and installing safety features such as nets.

SYSTEM ARCHITECTURE AND COMPONENTS

One objective of RAVEN is to enable researchers to test a variety of algorithms and technologies applicable to multi-UAV mission problems in real time. Therefore, the architecture must facilitate adding and removing hardware and software components as needed. Another design objective is to provide a single operator with the capability to simultaneously command several autonomous vehicles. Consequently, the architecture must include components that provide the mission operator with sufficient situational awareness to verify or issue changes to a vehicle’s plan in real time.

To meet these requirements, RAVEN’s architecture has the hierarchical design shown in Figure 1(a). This architecture separates the mission and task components from the testbed’s vehicle infrastructure. Thus, changes in the mission and task components of the system can be made in real time without changing the testbed’s core software infrastructure. This approach builds on the system architecture used in the DARPA-sponsored Software Enabled Control capstone flight demonstration by the MIT team [24], [25].

As shown in Figure 1(b), the planning part of the system architecture has four major components: 1) a mission-planning level that sets the system goals and monitors system progress, 2) a task-assignment level that assigns tasks to each vehicle in support of the overall mission goals, 3) a trajectory-design level that directs each vehicle on how to best perform the tasks issued by the task-processing level, and 4) a control-processing level that carries out the activities set by higher levels in the system. In addition, health information about each system component is used to make informed decisions on the capabilities of each subsystem in the architecture. As a result, system components are designed to support and promote strategies that are in the system’s best interests for each task.

The architecture used in RAVEN allows researchers to rapidly interchange mission-system components for the purpose of testing various algorithms in a real-time environment.
example, to allow users to rapidly prototype mission, task, and other vehicle-planning algorithms with RAVEN's vehicle hardware, each vehicle must be able to accept command inputs, such as waypoints, from a high-level planning system. Although low-level commands such as "fly at a set speed" can be issued to any vehicle in the platform, a waypoint interface to the vehicles allows users to substitute mission-, task-, and path-planning components into the architecture without changing the vehicle’s base capabilities. This interface allows users to add, remove, and test algorithms and other related components as needed. In fact, the waypoint interface to the vehicles allows users to develop and implement code on the platform in real time to test centralized and distributed planning algorithms using computers from other locations on campus. As a result, researchers can implement and test control, navigation, vehicle-tasking, health, and mission-management algorithms on the system [26]–[28]. Likewise, users can incorporate additional vehicles into the testbed architecture since vehicle controllers and base capabilities can be added, removed, and tested in the architecture without affecting the remaining system components.

MAIN TESTBED HARDWARE

Figure 1(c) shows the system setup, with perception, planning, and control processing performed in linked ground computers. This system configuration can emulate distributed planning (using network models of the environment to govern vehicle-to-vehicle communication) as if the planning were being done on board. Note that the motion-capture system, which provides vehicle position and attitude data, is the only centralized component; however, each vehicle’s position and attitude data are transmitted to that vehicle’s ground-based computing resources. Therefore, each vehicle is unaware of the other UAVs during an operation unless a vehicle-to-vehicle communication link is established.

Currently, the control processing and command data for each vehicle is processed by a dedicated ground computer and sent over a USB connection to the trainer port interface on the vehicle’s R/C transmitter. Each dedicated ground computer has two AMD 64-bit Opteron processors and 2 Gb of memory and runs Gentoo Linux.

A Vicon MX motion-capture system provides position and attitude data for each vehicle in the testbed [29]. By attaching lightweight reflective markers to the vehicle’s structure, the motion-capture system tracks and computes each vehicle’s position and attitude at 100 Hz. The accuracy of the motion-capture system’s position and attitude estimates is difficult to confirm during flight operations. To assess the static accuracy, Figure 2 shows a scatter plot of the measured \((x, y)\) position (in meters) of a quadrotor sitting on the floor at position \((0, 0)\). With the rotors not turning, the maximum \(x\)-position error measured by the system is 0.325 mm, while the maximum \(y\)-position error measured by the system is 0.199 mm. Tracking multiple reflectors in a unique orientation on each vehicle enables the motion-capture system to calculate the position of the center of mass and the attitude of each air/ground vehicle within range. For example, an 18-camera configuration can track five air vehicles and multiple ground vehicles in an 8-m-by-5-m-by-3-m flight volume.

Currently, RAVEN is comprised of various rotary-wing, fixed-wing, and ground-based R/C vehicle types, as shown in figures 3, 4, and 5. However, most testbed flight experiments are performed using the Draganflyer V Ti Pro quadrotor [30]. This quadrotor is small (\(\approx 0.7\) m from blade tip to blade tip) and lightweight (under 500 g), with a payload capacity of about 100 g and the ability to fly between 13–17 min on one
battery charge (using a 2000-mAh lithium-ion battery from Thunder Power [31]) while carrying a small camera. The four-propeller design simplifies the dynamics and control, and the vehicle’s airframe is robust and easy to repair in the event of a crash. The rotor blades are designed to fracture or bend when they hit a solid object. The Draganflyer V Ti Pro quadrotor is thus durable and safe for indoor flight.

A separate landing and ground-maintenance system are used to support the quadrotor vehicle hardware in an around-the-clock environment. More specifically, the landing hardware and its associated real-time processing aid the vehicle’s guidance and control logic during takeoff and landing. In addition, a maintenance module evaluates whether the vehicles are due for maintenance and monitors the recharging of the batteries prior to flight [32], [33].

Likewise, modified R/C trucks made by DuraTrax are used as ground vehicles in the platform [34]. The modifications consist of replacing the stock onboard motor controller and R/C receiver with a custom motor driver circuit, Robostix microcontroller [35], and RF communication module with 802.15.4 wireless connectivity. These modifications improve the vehicle’s precision-driving capabilities while making it possible to autonomously command and control multiple ground vehicles by means of one ground computer in mission scenarios, such as airborne search and track missions, search and recovery operations, networking experiments, and air and ground cooperative mission scenarios.

**Task Processing and Operator Interface Components**

The control system for each vehicle in the testbed can process and implement tasks defined by a system component or user. For example, each vehicle has a vehicle manager module that is executed on the dedicated ground computer and designed to handle the task processing, trajectory generation, and control processing. This module

---

**FIGURE 4** Airplane model in (a) autonomous hover and the airplane axis setup for (b) hover experiments. The lightweight reflective spheres glued to the vehicle’s structure shown on the airplane in (a) and (b) are used by the motion-capture system to track and compute the airplane’s position and attitude.

**FIGURE 5** Fully autonomous flight test with (a) five UAVs and (b) a closeup of five UAVs in flight. In this flight, the autonomous tasking system commands all five vehicles to take off and hover 0.5 m above the ground for two minutes. In (b) the five vehicles are shown as they hover during the test. In (a) the five transmitters for the vehicles are shown. The transmitter for each vehicle is connected directly to a dedicated ground computer, which monitors and commands the vehicle during flight.
is designed to allow an external system or user to communicate with the vehicle using task-level commands, such as "fly to waypoint A," "hover/loiter over area B," "search region C," "classify/assess object D," "track/follow object E," and "take off/land at location F." These commands can be sent to the vehicle at any time during vehicle operations. Each agent's vehicle manager processes these tasks as they arrive and responds to the sender acknowledging the task request.

RAVEN is also designed with an automated task manager that manages the testbed's air and ground vehicles using task-level commands. As a result, multivehicle mission scenarios, such as search, persistent surveillance, and area denial, can be organized and implemented by the task manager without human intervention. Figure 5 shows a mission in which the automated task manager controls a group of five UAVs.

Although coordinated multivehicle flight tasks can also be managed autonomously by RAVEN, the system has an operator interface with vehicle tasking capability. The task manager system is designed to allow an operator to issue a command to any vehicle at any time. Currently, the operator interface includes a three-dimensional representation of the objects in the testing area (as shown in Figure 3) and a command and control user interface, which displays vehicle health and state data, task information, and other mission-relevant data.

Each vehicle's trajectory is specified as a sequence of waypoints consisting of a location $\bar{x}_i = (x, y, z)$, vehicle heading $\psi_i$, and speed $v_i$. Given these waypoints, several options are available for selecting the vehicle's reference inputs. Perhaps the simplest path is to follow a linear interpolation of the points defined by

$$x_{\text{ref}}(t) = \frac{1}{|x_i+1 - x_i|}(x_{i+1} - x_i)v_it,$$  

where the choice of $v_i$ can be used to move between waypoints at varying speeds. This approach is used to automate the takeoff and landing procedure for the quadrotor vehicles. As a result, the quadrotor vehicles are fully autonomous from takeoff to landing during all flight operations.

**QUADROTOR CONTROL DESIGN MODEL**

A model of the quadrotor vehicle dynamics is needed to design a hover controller. Figure 6 defines an inertial frame $(x_E, y_E, z_E)$, which is used to specify the vehicle's position, and the roll, pitch, and yaw Euler angles $\phi$, $\theta$, and $\psi$, which are used to specify the vehicle's orientation. The body frame is specified by the $x_B$, $y_B$, and $z_B$-axes; the body moments of inertia are $I_x$, $I_y$, and $I_z$; and $p$, $q$, and $r$ denote the body angular rates in the body frame. Additional parameters include the distance $L$ from the quadrotor's center of mass to its motors, the mass $m$ of the vehicle, the moment of inertia $J_R$ of a rotor blade, and a disturbance $d$ generated by differences in rotor speed. The inputs to the system are $\delta_{\text{collective}}$, $\delta_{\text{roll}}$, $\delta_{\text{pitch}}$, and $\delta_{\text{yaw}}$, which are the collective, roll, pitch, and yaw input commands, respectively. Starting with the flat Earth, body-axis six-degree-of-freedom (6DOF) equations [36], the kinematic and moment equations for the nonlinear quadrotor model can be written as

$$\dot{p} = qr \left( \frac{I_y - I_z}{I_x} \right) - \frac{J_R}{I_x} qd + \frac{L}{I_x} \delta_{\text{roll}},$$  

$$\dot{q} = pr \left( \frac{I_z - I_x}{I_y} \right) + \frac{J_R}{I_y} rd + \frac{L}{I_y} \delta_{\text{pitch}},$$  

$$\dot{r} = pq \left( \frac{I_x - I_y}{I_z} \right) + \frac{1}{I_z} \delta_{\text{yaw}},$$  

$$\dot{\phi} = p + \tan \theta (q \sin \phi + r \cos \phi),$$  

$$\dot{\theta} = q \cos \phi - r \sin \phi,$$  

$$\dot{\psi} = (q \sin \phi + r \cos \phi) \sec \theta,$$

where $-(I_R/I_x)rd$ and $-(I_R/I_y)qd$ represent the disturbances caused by changing rotor speeds (as discussed in [37]), although the gyroscopic effects due to these disturbances for the Draganflyer [30] model are small. Also, the cross-product moments of inertia are omitted since $I_{xx}$ is considerably smaller than $I_x$ and $I_z$ due to the shape of the quadrotor, while $I_{xy}$ and $I_{yz}$ are zero due to the vehicle symmetry. Since the force applied to the vehicle as a result of the collective command can be represented as a thrust vector along the negative $z_B$-body axis, the nonlinear navigation equations for the quadrotor in the reference frame defined in Figure 6 are given by
Several mission-management, task-allocation, and path-planning algorithms have been successfully implemented and tested on RAVEN.

FIGURE 7  Single-vehicle hover experiment. In this test flight, a quadrotor UAV is commanded to hover at \((x, y, z) = (0, 0, 0.7)\) m for 10 min. Part (a) shows the \(x\)–\(y\) plot of vehicle position, (b)–(d) show histograms with percentage of time at location for \(x, y,\) and \(z\) positions, and (e)–(g) show histograms with percentage of time at each flight condition for \(x, y,\) and \(z\) velocities. These results demonstrate that the vehicle can hold its position during flight. In this test, the vehicle remains inside the 20-cm box throughout the 10-min hover test flight.
where \( u = \delta_{\text{collective}} \). After linearizing this model around the hover flight condition with \( \psi \approx 0 \), setting \( \delta_{\text{collective}} = mg \delta_{\text{collective}} \), and dropping small terms in the \( x_E \) and \( y_E \) dynamics, yields

\[
\begin{align*}
\dot{x}_E &= \phi, \\
\dot{y}_E &= -\theta, \\
\dot{z}_E &= \frac{1}{m} \delta_{\text{collective}}, \\
\phi &= p, \\
\theta &= q, \\
\psi &= r, \\
\dot{p} &= \frac{L}{I_x} \delta_{\text{roll}}, \\
\dot{q} &= \frac{L}{I_y} \delta_{\text{pitch}}, \\
\dot{r} &= \frac{L}{I_z} \delta_{\text{yaw}}.
\end{align*}
\]

The values \( I_x, I_y, I_z, L, m, \) and \( J_k \) are measured or determined experimentally for the Draganflyer models used in RAVEN.

An integrator of the position and heading error is included in each loop so that the controller can remove steady-state position and heading errors in hover. Since the motion-capture system accurately and directly measures the system’s state, four linear-quadratic-regulator (LQR) controllers using four combinations of vehicle states, namely \( \phi, \theta, \) and \( \psi \), are used to stabilize and control the quadrotor. The regulators are designed to optimize the vehicle’s capabilities in hover, while ensuring that the vehicle can respond quickly to position errors.

The vehicle’s controllers are optimized for surveillance experiments. In these experiments, a camera is mounted on the quadrotor vehicles facing toward the ground. Thus, large changes in pitch, roll, and yaw affect the vehicle’s ability to focus on a ground target. In addition, an anti-windup bumpless transfer scheme (similar to those described in [38]) with adjustable saturation bounds is used to prevent the integrators from winding up while the vehicle is on the ground before, during, and after takeoff and landing.

**HOVERING AIRPLANE CONTROL DESIGN MODEL**

In addition to quadrotors, a foam R/C aircraft [shown in Figure 4(a)] is used to explore the properties of an aircraft flying in a prop-hang (that is, nose up) orientation for the purpose of landing vertically and performing other complex maneuvers, such as perching. To avoid Euler-angle singularities, the airplane’s body-fixed reference frame is defined such that the positive \( x \)-axis points down from the airplane’s undercarriage, the positive \( y \)-axis points out along the right wing, and the positive \( z \)-axis points out the tail of the aircraft as shown in Figure 4(b). Using this reference frame, the vehicle’s nominal attitude reference in hover corresponds to \( \phi = \theta = \psi = 0^\circ \).

Assuming that the Earth is an inertial reference frame and the aircraft is a rigid body, the aircraft equations of motion [36], [39] can be written as (5)–(7), (8)–(10) with \( u = \delta_{\text{throttle}} \), and

\[
\begin{align*}
\dot{p} &= pr \left( \frac{I_y - I_z}{I_x} \right) + \frac{A_{\text{rudder}}}{I_x} \frac{L}{y} \delta_{\text{rudder}}, \\
\dot{q} &= qr \left( \frac{I_z - I_x}{I_y} \right) + \frac{A_{\text{elevator}}}{I_y} \delta_{\text{elevator}}, \\
\dot{r} &= pq \left( \frac{I_x - I_y}{I_z} \right) + \frac{\omega_{\text{prop}} L}{I_z} d_{\text{prop}}^3 + \frac{C_{l_{\text{prop}}}}{2\pi} \dot{\omega}_{\text{prop}}^2 \frac{d_{\text{prop}}^3}{I_z} + \frac{A_{\text{aileron}}}{I_z} \dot{\delta}_{\text{aileron}}.
\end{align*}
\]

where \( d_{\text{prop}} \) is the diameter of the propeller.
Near hover, the influence of external forces and moments on the aircraft, other than gravity, are negligible, and velocities and rotational velocities are small. Using small angle approximations, the equations of motion (5)–(7), (8)–(10), and (20)–(22) can be considerably simplified. Since \( I_{\text{prop}}/I_z \approx 0.04 \ll 1 \), the torque due to a change in the rotational speed of the motor can also be disregarded. Next, define \( \delta_{\text{throttle}} = m g + \dot{\delta}_{\text{throttle}} \) and

\[
\delta_{\text{aileron}} = -C_{1, \text{prop}} \left( \frac{\omega_{\text{prop}, 0}}{2\pi} \right)^2 \frac{L_{\text{prop}}}{I_z} + \dot{\delta}_{\text{aileron}},
\]

where \( \omega_{\text{prop}, 0} \) is the average rotational speed of the propeller to keep the aircraft in hover [40]. Next, since the vehicle’s reflective markers are mounted on top surface of both wings and the fuselage, they are only visible to cameras facing the top of the vehicle when the airplane is in hover. Therefore, \( \psi_{\text{reference}} = -(\pi/2) \) to ensure that there are at least three or more cameras facing the reflective markers. Linearizing the equations of motion for the airplane in hover yield

\[
\begin{align*}
\dot{x}_E &= g \dot{\theta}, \\
\dot{y}_E &= g \dot{\phi}, \\
\dot{z}_E &= \frac{1}{m} \dot{\theta}_{\text{throttle}}, \\
\dot{\theta} &= \dot{q}, \\
\dot{\phi} &= \dot{p}, \\
\dot{\psi} &= r, \\
\dot{p} &= \frac{A_{\text{rudder}}}{L_x} \dot{\theta}_{\text{rudder}}, \\
\dot{q} &= \frac{A_{\text{elevator}}}{L_y} \dot{\theta}_{\text{elevator}}, \\
r &= \frac{A_{\text{aileron}}}{L_z} \dot{\theta}_{\text{aileron}}.
\end{align*}
\]

Here, \( I_x, I_y, \) and \( I_z \) correspond to the body moment of inertia terms, while \( A_{\text{elevator}}, A_{\text{rudder}}, \) and \( A_{\text{aileron}} \) correspond to the deflected area of each control surface that is subject to propeller flow while the airplane is in hover. In addition, \( L_{\text{elevator}}, L_{\text{rudder}}, \) and \( L_{\text{aileron}} \) are the lengths of the control surface moment arms. These terms are measured or determined experimentally for this model.

Four control schemes are applied to combinations of vehicle states \( \phi \) and \( x_E, \theta \) and \( y_E, \psi, \) and \( z_E \) to stabilize and control the airplane in hover. Each loop uses proportional plus derivative (PD) and proportional plus integral plus derivative (PID) controllers to maintain the vehicle’s position during the hover condition. In particular, the controllers use large gains on the state derivative errors to prevent the vehicle from moving too quickly when trying to maintain its position.

Several issues arise in trying to control an airplane in hover. For example, propeller drag torque changes with motor speed, causing the vehicle to rotate about its \( z_B \)-axis shown in Figure 4(b). This rotation is mitigated by adding an aileron deflection proportional to motor speed error around the equilibrium speed. The varying

![FIGURE 9 Multivehicle coordinated flight experiment. In this test, two vehicles are commanded to fly at a constant speed in a circular pattern with changes in altitude. The vehicles are commanded by the system’s task advisor to take off and fly a circular trajectory maintaining a constant speed of 0.25 m/s and 180 deg of angular separation. In particular, the vehicles are flying in a circle (as projected in the \( x-y \) coordinate frame shown in (a) and (b)), while they are changing altitude (flying from an altitude of 0.5 m to 1.5 m) as they move around the flight pattern as shown in (c). This test is repeated multiple times, and the vehicles fly similar flight paths in five consecutive tests as shown in (b). Notice that the vehicle trajectories in the lower right corner of the plot appear to be more noisy. This disruption is partially caused by the quadrotors flying through the rotor downwash from another vehicle. These results show that the tasking system can command and coordinate the actions of multiple vehicles.](image-url)
speed of the propeller also affects the speed of airflow over control surfaces, hence affecting control surface actuation. This issue is most prominent in roll control. To overcome this effect, the ailerons are deflected additionally at low throttle settings. Unfortunately, as aileron deflection is increased, the ailerons block a portion of the propeller wash that would otherwise reach the elevator. To resolve this issue, a gain proportional to aileron deflection is added in the elevator control path to compensate for reduced airflow over the elevator and improve the vehicle’s pitch response when the ailerons are deflected.

Finally, fast control surface motions can cause the vehicle’s airframe to deform during flight. This deformation causes the reflective markers used by the motion-capture system to shift in position, thereby giving an incorrect estimate of the airplane’s momentary location. Consequently, the controller gains are sized to minimize rapid changes in control surface position. Flight results for the airplane in hover are given in the next section.

**RESULTS**

An overview of RAVEN testbed operations is provided in “Testbed History.”

**Testbed History**

Various multivehicle tests and mission scenarios have been flown using RAVEN. In particular, since January 2006, more than 2500 vehicle experiments have been performed, including approximately 60 flight demonstrations during a 16-h period at the Boeing Technology Exposition at Hanscom Air Force Base near Lexington, Massachusetts, on May 3–4, 2006. Each test performed at the event involved two vehicles. One test involved two air vehicles flying a three-dimensional coordinated pattern (see Figure 9), while another involved an air vehicle following a ground vehicle. These demonstrations show that the platform can perform multiple UAV missions repeatedly, on demand, and with minimal setup.
This article describes an indoor testbed developed at MIT for studying long-duration multivehicle missions in a controlled environment.

**Quadrotor**

Typical results from a 10-min hover test are shown in Figure 7. In this test a single quadrotor is commanded to hold its position at \((x, y, z) = (0, 0, 0.7)\) m for a 10-min period. Figure 7 shows four plots, including a plot of the vehicle’s \(x\) and \(y\) positions. The dashed red box in the picture is ±10 cm from the center point. As shown in Figure 7, the vehicle maintains its position inside this 20-cm box during the entire flight. The remaining plots in the figure are the histograms of the vehicle’s \(x\), \(y\), and \(z\) positions during these tests. This test shows that a quadrotor can maintain both its altitude (staying between 0.65–0.75 m) and position (staying mostly within 0.05 m from center) in hover over the full charge cycle of a battery.

The results of a single-vehicle waypoint tracking experiment are shown in Figure 8. In this test the vehicle is commanded to hold its altitude at 1 m while flying at a velocity of 0.05 m/s between each of the following waypoints: \((-1.5, 0, 1)\) m, \((-1.5, -0.5, 1)\) m, \((1.5, -0.5, 1)\) m, \((1.5, 5.5, 1)\) m, and \((-1.5, 5.5, 1)\) m, and then back to \((-1.5, 0, 1)\) m. In addition, the vehicle hovered at each waypoint for 10 s before flying to the next waypoint. This test demonstrates that the vehicle can follow a set trajectory around the indoor laboratory flight space. The plots in Figure 8 show that the vehicle follows the trajectory around a 3-m-by-6-m rectangular box as specified. The cross-track error is less than 15 cm from the specified trajectory at any given time during the flight.

In addition to these single-vehicle experiments, several multivehicle experiments and test scenarios have been conducted, as shown in Figure 5. These tests include, but are not limited to, formation flight tests, coordinated vehicle tests involving three air vehicles, and multivehicle search and track scenarios. In particular, Figure 9 shows the results from a two-vehicle coordinated flight experiment. In this experiment, the vehicles are commanded by the system’s task advisor to take off and fly a circular trajectory maintaining a constant speed of 0.25 m/s and 180° of angular separation. In particular, the vehicles are flying in a circle (as projected in the \(x – y\) coordinate frame) while they change altitude (flying from an altitude of 0.5 m to 1.5 m) as they move around the flight pattern. Figure 9(a) shows the \(x – y\) projection of one of the five circle test flights completed as part of this experiment. Notice that the vehicle trajectories in Figure 9(a) appear to be more noisy. This disruption is partially caused by the quadrotors flying through the rotor downwash from another vehicle. Flight testing shows that the downwash from these quadrotor vehicles is substantial, thus making it difficult to fly one quadrotor below a second quadrotor without significant altitude separation. Figure 9(c) shows a three-dimensional view of the trajectory, making it clear that the vehicles are also changing altitude during the flight. Figure 9(b) shows the results of five consecutive two-vehicle circle test flights performed over a 20-min time span. These results demonstrate that experiments run on RAVEN are repeatable and that the platform can be used to perform multiple test flights over a short period of time.
Hovering Airplane

Just as with the quadrotor, hover tests are performed with the foam airplane shown in Figure 4. In Figure 10 the vehicle is commanded to hold its position at $(x_E, y_E, z_E) = (0, 0, 0.7)$ m for 5 min. Figure 10 shows four plots, including a plot of the vehicle $x - y$ location while it maintains its position and attitude. The dashed red box in Figure 10 is $\pm 0.5$ m from the center point. As shown in Figure 10, the vehicle maintains its position inside this $1$-m box for most of the 5-min test period. The remaining plots give histograms of the vehicle’s $x$, $y$, and $z$ positions. These plots confirm that the vehicle is within a 50-cm box around the target point more than 63% of the time and stays within 0.8 m of the desired location throughout the entire test. These plots also show that the vehicle maintains its altitude by staying between 0.65–0.75 m during the entire hover test. A video of the aircraft in the hover flight condition can be found online at [41].

Figure 11 shows two waypoint tracking tests for the airplane starting from hover and then moving at a horizontal rate of 0.3 m/s between a set of waypoints while remaining nose-up. In (a) the vehicle starts from (1.5, 5.5, 1) m, while the vehicle starts from (−1.5, 0, 1) m in (b). In both tests, the airplane flies along the desired flight path as it hovers between each waypoint despite the fact that the vehicle has less control authority in its yaw axis, making it difficult to maintain the vehicle’s heading during flight. The reduced control authority of the vehicle’s yaw axis in hover is due to the fact that propeller wash covers less than 10% of the ailerons in the hover flight condition, thus reducing the vehicle’s ability to counteract disturbances in propeller torque while maintaining vehicle heading. As a result, external disturbances cause the vehicle to deviate from a straight flight path between waypoints. However, the vehicle stays within 0.5 m of the intended flight path throughout most of the tests. “Rapid Prototyping” discusses the short timelines for completing these aircraft flight results, which validates RAVEN’s rapid prototyping capabilities.

CONCLUSIONS

This article describes an indoor testbed developed at MIT for studying long-duration multivehicle missions in a controlled environment. While other testbeds have been developed to investigate the command and control of multiple UAVs, RAVEN is designed to explore long-duration autonomous air operations using multiple UAVs with virtually no restrictions on flight operations. Using the motion-capture system capability, RAVEN enables the rapid prototyping of coordination and control algorithms for different types of UAVs, such as fixed- and rotary-wing vehicles. Furthermore, RAVEN enables one operator to manage up to ten UAVs simultaneously for multivehicle missions, which meets the goal of reducing the cost and logistic support needed to operate these systems.

Finally, several mission-management, task-allocation, and path-planning algorithms have been successfully implemented and tested on RAVEN [32], [33]. For example, the UAVs use software monitors that determine when they must return to base for recharging. These monitors, in conjunction with other system health-management information, enable the strategic- and tactical-level mission planning modules to make informed decisions about the best way to allocate resources given the impact of likely failures. Future work designed to develop and integrate alternative health-management and multi-UAV mission technologies into the platform for real-time testing is ongoing.

ACKNOWLEDGMENTS

The authors would like to thank James McGrew and Spencer Ahrens for their assistance in the project. This research has been supported by the Boeing Company, Phantom Works, Seattle, and AFOSR grant FA9550-04-1-0458. Brett Bethke would like to thank the Hertz Foundation and the American Society for Engineering Education for their support.

REFERENCES


AUTHOR INFORMATION
Jonathan P. How (jhow@mit.edu) is a professor in the Department of Aeronautics and Astronautics at the Massachusetts Institute of Technology (MIT), where he leads the Aerospace Controls Laboratory. He graduated from the University of Toronto with a B.Sc. in aerospace engineering and received the M.Sc. and Ph.D. degrees in aeronautics and astronautics from MIT in 1989 and 1993, respectively. He is a Senior Member of the IEEE. He can be contacted at Massachusetts Institute of Technology, Aerospace Controls Laboratory, 77 Massachusetts Ave., Rm 33-326, Cambridge, MA 02139 USA.

Brett Bethke is a research assistant in the Aerospace Controls Laboratory at MIT. He has S.B. degrees in physics and aerospace engineering, an S.M. in aeronautics and astronautics, and is pursuing his Ph.D. degree in the Aeronautics and Astronautics Department at MIT. He currently has a fellowship from the Hertz Foundation and the American Society for Engineering Education.

Adrian Frank is a visiting student in the Aerospace Controls Laboratory at MIT. He is currently pursuing his M.Sc. degree in aeronautical engineering at the Royal Institute of Technology in Stockholm, Sweden.

Daniel Dale received an M.Eng. degree from the Electrical Engineering and Computer Science Department at MIT in 2007. He has S.B. degrees in physics and electrical engineering from MIT.

John Vian is a technical fellow with The Boeing Company, where he is responsible for leading Phantom Works’ self-aware adaptive systems strategic technology area. He holds a B.Sc. in mechanical engineering from Purdue University and an M.Sc. in aeronautical engineering and Ph.D. in electrical engineering, both from Wichita State University.