JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 1, December 2004

Development and Test of Highly Autonomous Unmanned Aerial Vehicles

E. N. Johnson,[*] A. A. Proctor,[†] J. Ha,[‡] and A. R. Tannenbaum[§]
Georgia Institute of Technology, Atlanta, Georgia 30332-0150

This paper describes the design, development, and testing of Unmanned Aerial Vehicles (UAV) with highly automated search capabilities. Here, systems are able to respond on their own in the presence of considerable uncertainty utilizing an image processor, tracker/mapper, mission manager, and trajectory generation; and are used to complete a realistic benchmark reconnaissance mission. Subsequent to the selection of the search area, all functions are automated and human operator assistance is not required. The applications of these capabilities include reduction of operator workload in operational UAV systems, new UAV or guided-munition missions conducted without the assistance or availability of human operators, or the enhancement/augmentation of human search capabilities. The resulting system was able to search the 15-building village automatically with speed comparable to a human operator searching on foot or with a conventional remotely piloted vehicle. It was successful in 6 of 7 actual flights over the McKenna Military Operations in Urban Terrain test site over two different days and a variety of lighting conditions and choice of desired building.

I. Introduction

THIS paper describes the design, development, and testing of an Unmanned Aerial Vehicle (UAV) system with highly automated search capabilities. The automated search capabilities allow the system to search a prescribed area, identify a specific building within that search area based on a small identifying sign located on one wall, and identify a candidate opening into that specified building. Subsequent to selection of the search area, all functions are automated, and do not require human operator assistance. The applications of these capabilities include reduction of operator workload in operational UAV systems, new UAV or guided-munition missions conducted without the assistance or availability of human operators, or the enhancement/augmentation of human search capabilities. This work builds upon previous development of a research UAV system and image processing algorithms, and is the first publication of the method used for automated mission management, i.e. the automation of mission-level decisions. Of particular significance is the fact that this work was carried to the flight test phase, and was tested under realistic conditions. This introduces all of the issues relating to real-time algorithms, dealing with noise/clutter/uncertainty, and logistics so important to practical automated mission management.

This paper is part of the December Intelligent Systems Special Section. Received 5 September 2004; revision received 17 November 2004; accepted for publication 18 November 2004.

[*]Lockheed Martin Assistant Professor of Avionics Integration Proctor, eric.johnson@aerospace.gatech.edu
[†]Graduate Research Assistant, alison_proctor@ae.gatech.edu
[‡]Graduate Research Assistant, gtg203c@mail.gatech.edu
[§]Professor, tannenba@bme.gatech.edu

A. Mission Definition

The baseline reference mission provides a focus for the work described herein, including specifications for an automated visual search system with real-world applications. Many other reconnaissance missions may be postulated that are similar. The mission of consideration here is the Level 2 mission of the International Aerial Robotics Competition, rules as specified for 2001-2004 (Ref. 1). This is an intercollegiate engineering competition, sponsored by Association for Unmanned Vehicle Systems International (AUVSI), in which university teams build automated UAV systems that perform difficult missions; missions that are designed to push the limits of UAV automation. At the time this article was written, Georgia Tech is one of two teams to have attempted this portion of the challenge; the approach that some of the other groups are taking or planning to take can be found in Refs. 2 and 3.

The mission specifies a search area containing any number of buildings and other obstacles. The UAV system must automatically (without human operator control or assistance):

1) Fly into the search area.
2) Locate a specific building identified by a 1-meter square sign located on one wall, Fig. 1.
3) Identify an opening into this specific building. The opening can be a window or a door, but it must be open.

Fig. 1 Symbol for building identification located on one wall, actual size is 1m diameter.

For 2003 and 2004, the competition was held at the McKenna Military Operations in Urban Terrain (MOUT) site in Fort Benning, Georgia. This location includes a village with 15 buildings, including normal obstacles such as overhead wires, towers, and trees. The pattern of building locations is irregular, and many of the buildings have unique shapes, window/door configurations, and colors. The pre-specified search area includes the entire village, and the symbol in Fig. 1 can be placed on the exterior wall of any building. An overhead view of the village is shown in Fig. 2.

Fig. 2 Overhead view of the McKenna site, a village with 15 buildings and other obstacles such as overhead wires and trees; a the white car located at the top of the image near the center provides a sense of scale.

B. Overall Approach

A research UAV with automated guidance, navigation, and flight control capabilities was augmented with a new automated visual search and mission management subsystems described in this paper. The research UAV utilized is the Georgia Tech Yamaha R-Max system, referred to as the GTMax.[4] The resulting system has been tested in simulation and flight tests, and was utilized as an entry into the Aerial Robotics Competition in 2002 and 2003 and for related flight tests.[1]

When performing realistic missions such as these, it is often necessary to make decisions based on data that is collected during the mission, such as where to fly and when. A human operator traditionally makes these decisions, based on a variety of information sources. The use of automation to assist a human operator in making these decisions is a widely employed paradigm. However, when such a mission is performed by a UAV without any human interaction, each step of the decision making process must be completely automated. To achieve this level of autonomy, data collected must be fused together to produce a quantity that is relevant to the decision to be made. An algorithm is also often required to determine the action that corresponds to the outcome of the decision, such as a desired flight path. Autonomous mission managers that can handle this type of complex scenario have been proposed by researchers for many different vehicles including underwater autonomous vehicles,[5-7] autonomous aircraft,[8] and land robots.[9]

Automating a decision requires that information from all relevant sensors be utilized to form an aggregate picture of the environment. Data fusion and data association are well-researched topics. Fusing navigation data from multiple sensors is commonplace in UAVs, and methods for correlating multiple targets in continuous imagery have been presented in many scientific disciplines.[10] However, there has been little work in the literature that covers actual field-testing of UAVs where onboard sensor data is used for real-time classification and correlation of objects detected using image processing during flight, and even less where this information is used for automated mission decisions.

Trajectory generation is also a well-researched topic. There are many methods to produce desired trajectories for autonomous vehicles from one point to another in a complex environment.[11,12] Here, the emphasis is not in the optimization of trajectory shape, but instead on the method to chose where to fly and when.

This paper describes a mission manager that uses several concepts to complete a realistic mission. This mission manager selects where the aircraft is to fly and when, configures other onboard systems, and selects system output (in this case, the selection of the opening to a building). The general approach is to simplify mission management into discrete decisions that take place at the boundaries of mission *phases*. In this way, the difficulty of reliably converting collected information into automated decisions is minimized. Furthermore, the difficulty of communicating intent to any human operators (or observers) is reduced. This approach is more complex than the one proposed for the same mission in Ref. 2, where no decisions are made in flight. In Ref. 2, an exhaustive search is flown, and images processed after landing. The benefit we anticipate from this added complexity is more probable mission completion, and a greater potential for possible missions, by allowing in flight mission decisions.

In our view, the primary significance of this paper lies in the formalization of the mission management system in a manner consistent with prior work, with a philosophy for carefully choosing a minimum number of mission decisions at the boundary of mission phases, and the actual flight testing of these algorithms in a highly realistic test.

This philosophy for selecting mission decision points evolved based on the experience of actually making the complete system function in the field.

The following section, Sec. II, describes the research UAV that was utilized, including the guidance, navigation, and control algorithms that where utilized. The methods used to locate the symbol and the opening are summarized in Sec. III. Section IV covers this mission management subsystem. Section V includes a discussion of flight test results, and Sec. VI conclusions.

II. GTMax Research UAV

The GTMax research UAV was utilized for this work, illustrated in Fig. 3, and is based on the Yamaha R-Max Industrial Helicopter airframe, which is normally utilized as a remotely piloted aircraft.[4] The basic vehicle has a rotor diameter of 10.2 ft., a 246cc 21 horsepower 2-cylinder water-cooled engine, and an empty weight of approximately 125 pounds.

Fig. 3 GTMax Research UAV

The hardware components that make up the flight avionics include general purpose processing capabilities and sensing, and add approximately 35 pounds to the basic airframe, leading to a total weight of approximately 160 pounds. The digital interface to the basic Yamaha vehicle is via a modified Yamaha Attitude Control System (YACS) interface that allows raw servo commands to be given without modification by this built-in stability augmentation system. The complete wiring diagram including configuration of RS-232, Ethernet, and power wiring is shown in Fig. 4. The research avionics configuration utilized in this work includes:

- 266 MHz Pentium II Embedded PC, 500 Mb Flash Drive
- 850 MHz Pentium III Embedded PC, 2 Gb Flash Drive
- Inertial Science ISIS-IMU Inertial Measurement Unit
- NovAtel OEM-4, differential GPS
- Honeywell HMR-2300, 3-Axis magnetometer
- Custom made ultra-sonic sonar altimeter
- Custom made optical RPM sensor
- YACS: Vehicle telemetry (RPM, voltage, low fuel) and Actuator control interface
- 11 Mbps Ethernet data link and an Ethernet hub
- FreeWave 900MHz spread spectrum serial data link
- Axis 2130R pat, tilt, and zoom network camera

The onboard software runs on the two onboard computers, referred to as the primary flight computer and the secondary computer. The Ground Control Station (GCS) software, Fig. 5, runs on the ground on one or more laptop computers, and is used primarily for system operators to configure and monitor the onboard systems.

The navigation system running on the primary flight computer is a 17-state extended Kalman filter. The states include: vehicle position, velocity, attitude (quaternion), accelerometer biases, gyro biases, and terrain height error. The system is all-attitude capable and updates at 100 Hz (Ref. 13). The flight controller is an adaptive neural network trajectory following controller with 18 neural network inputs, 5 hidden layer neurons, and 7 outputs for each of the 7 degrees of freedom (6 rigid-body degrees of freedom plus a degree of freedom for rotor RPM).[14] The

flight controller and navigation system, which coupled with the trajectory generator, is capable of automatic takeoff, landing, hover, flight speeds up to the maximum attainable by the helicopter (around 85 ft/sec) and aggressive maneuvering. The system also generates pan and tilt commands for the camera system to allow it to point at specified absolute or aircraft-relative local geographic coordinates.

The image processing and tracking/mapping components run on the second computer. During the flight, individual images come directly via Ethernet from the camera system in JPEG format at a rate of 1-2 Hz. They are then processed to identify either buildings or widows. A tracker combines these results along with navigation system data and camera state to update a single global picture of building and window tracking information. These algorithms are described in the following section.

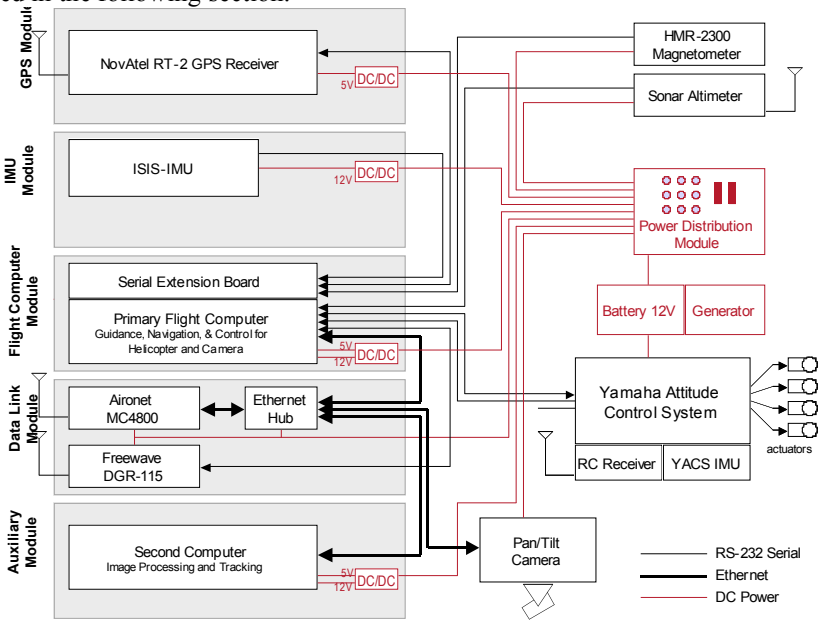

Fig. 4 GTMax wiring diagram, including computing resources, sensors, pan/tilt camera subsystem, and interface to Yamaha actuator

Fig. 5 Ground control station displays, (left) real-time video from the onboard camera and processed imagery; (right) monitoring onboard systems with a moving-map display, and other monitoring tools

III. Building and Opening Identification

Figure 6 shows a flow chart outlining the complete process for the symbol and window detection, classification and tracking. The symbol recognition system looks for line patterns in the images that are similar to the lines generated in a template of the actual symbol. Each potential symbol is filtered to determine that it only contains black and white pixels of the correct intensity. A score from the template match and color classification is then

passed to the tracker. In contrast, the window-tracking algorithm looks at the color of each pixel before the image is processed. Then it creates a black and white image that contains only the dark objects. Then the contours around the dark objects are extracted from the image and classified based on the length of the contour. Those objects whose perimeter is an appropriate length are passed to the window tracker along with information about the windows darkness and size. The tracking algorithm is similar for both the window and symbol modes. It takes the location and score of each object along with position and attitude information from the primary flight computer, and updates an array containing the position of the objects in the local geographical reference frame. Finally, it calculates an estimated probability of correct detection for each object, which is utilized for automated decisions relating to the mission execution.

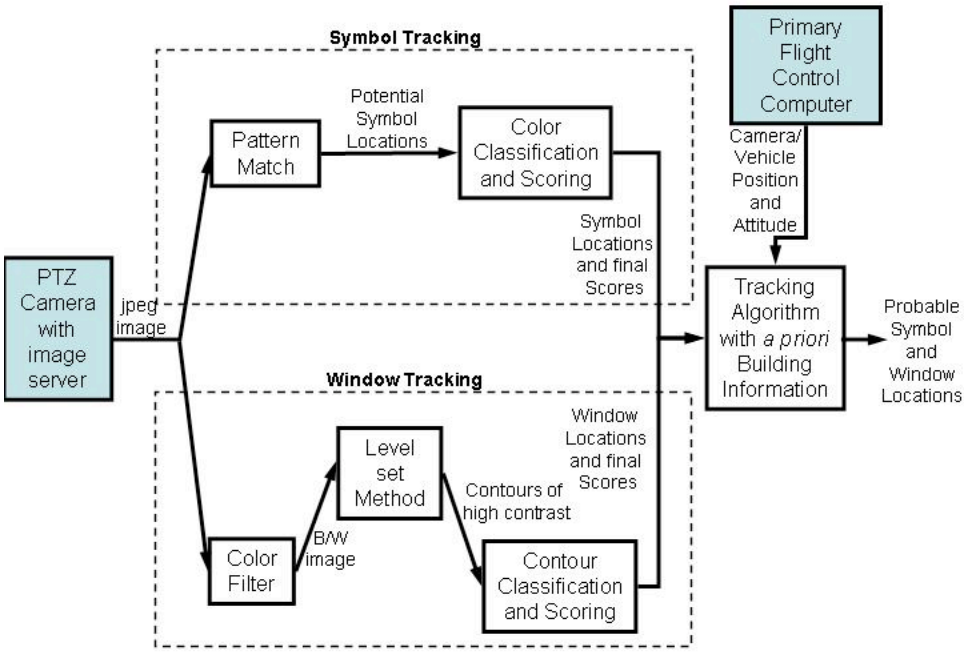

Fig. 6 Flow chart indicating the process used to identify the building by looking for the symbol and then for openings.

A. Symbol Detection

The McKenna MOUT site contains 15 buildings; for this mission, the correct building is identified by the symbol shown in Fig. 1, which is one meter in diameter. Traveling at the chosen velocity of 15 ft/sec, the symbol is in view for a maximum of 5 seconds. As a result, the symbol will only be visible in 5-10 of the 3000 images that are processing during the symbol-search portion of a mission attempt. As a result, it was important to utilize a symbol detection algorithm that minimizes the number of false rejections. The algorithm that was used to find the symbol has two stages. The first stage applies a shape-matching filter to each image and then the results are filtered by color to estimate the probability of this result being the symbol. By utilizing a shape match with a very low probability of false rejection followed by a stringent rejection filter, a highly accurate symbol detection algorithm is obtained.

The shape matching is performed using Halcon, developed by MvTec Software GmBH,[15] which is a commercially available library that provides C, C++, and Visual Basic routines for a number of image processing functions. In order to utilize the shape matching features available in Halcon, a template of the shape was generated using the shape template generation function. Here, the template was generated using a section of the interior of the symbol taken from practice images captured with the imaging system. During the template construction process it is important not to include the outer ring of the symbol, as seen in Fig. 7. The contrast between the outer ring and the building is a function of the building itself, and it is important not to include a bias for any particular building color.

The shape matching was performed on each image using a safe heuristic search for the template, where a safe search indicates that the chance of missing the template is minimized. This approach to the pattern match produces many incorrect matches, but also has a very low rate of false rejections, which allows the results to be further categorized with supplementary color filtering. This supplementary color filtering ensures that the classified object is black and white, and has appropriate ratio of black and white coloring.

The symbol identification was tested using sequential images obtained while flying sample mission trajectories. This provided imagery suitable for testing the acquisition phase of the symbol detector as well as the rejection and scoring phases.

Fig. 7 Template selection for shape matching function.

B. Symbol Tracking

The symbol tracker takes the location of any valid symbol candidates on an individual image, and transforms it into local geographical coordinates (LGC). Each symbol track is then evaluated to determine if it is a subsequent measurement of a pre-existing track or a new symbol-candidate track altogether. If it is a subsequent measurement, the new information is combined with the current information to obtain the best estimate of the actual symbol-candidate location using a Bayesian tracking technique described below. Then, once all of the buildings have been searched, the location with the highest probability is taken as the symbol location. Once the symbol location is determined in the local frame, the building closest to the symbol (normally the one overlapping the symbol track location), which satisfies heading/orientations constraints (symbol must be oriented consistent with the wall it is on), is chosen as the correct building for the second phase of the mission.

The first task of the tracker is to convert all of the relevant symbol information from the two dimensional image coordinates into LGC. This is done by first projecting the symbol image coordinates onto a pre-selected reference plane, which is parallel to the ground at the expected height of the symbol. In this case, it was decided to project the symbol onto a plane six ft. above the ground, which would lead to relatively little position error for any actual symbol height in the village (the camera was pointed approximately 45 degrees tilt below the horizon).

Once the location of a new symbol track is found, a probability that the track is the correct symbol is assigned. Each new symbol-candidate sighting is given an initial nominal probability of 0.1, and then additional comparison tests are used to modify the probability of the track by Eq. (1).

$$P(k) = P(k-1) + \left[1 - P(k-1)\right] S P_S \tag{1}$$

where k is the comparison number and S is the score between 0 and 1 which is obtained from the pattern matching algorithm, and P_S is a scaling factor to control the growth rate of the probability factor with additional comparisons. Once the final probability is determined it is used to calculate a weighted initial position variance for the symbol-candidate measurement. This variance is based on the estimated variance in position and angular measurements $\sigma_P^2, \sigma_\theta^2$ from the navigation system, and formulated as:

$$\sigma^2 = \frac{\sigma_P^2 + \sigma_\theta^2 R}{P} \tag{2}$$

where R is the estimated range to the symbol-candidate, and P is the probability found from Eq. (1).

Once this characteristic data for the location is found, it is examined to decide if it is a new object or one that is already being tracked. This decision is made based on the distance between the new symbol location measurement and all existing symbol track locations as well as the heading at which the new object and the old object were

detected. If it is decided that the proposed symbol location is a new symbol track, the position, heading and variance information are stored. If it is a new instance of a symbol that is already being tracked, the data from the new and old tracks are fused together using a weighted average based on the position variances. The variance for the newly updated track is given by:

$$\left(\sigma_t^2\right)^+ = \frac{\sigma_t^2 \sigma_{nt}^2}{\sigma_t^2 + \sigma_{nt}^2} \tag{3}$$

where subscript *t* denotes the existing track and subscript *nt* denotes the new measurement. Similarly, the position and heading information can be updated using:

$$\left(Y_t\right)^+ = \frac{Y_{nt}\sigma_t^2 + Y_t \sigma_{nt}^2}{\sigma_t^2 + \sigma_{nt}^2} \tag{4}$$

where *Y* is the variable being updated (position or heading).

C. Window Mapping

The multiple window/door opening detection utilizes the *geometric active contour* or *snake* method[16-20] to find contours around dark objects that have high contrast edges. Since the window detector is looking for naturally dark objects it is possible to use a color filter similar to the one used in the symbol detection. Then by using a fast-marching level set method the processing time for each image is decreased, which increases the number of window measurements provided at a given search flight speed.

In Refs. 18-20 an active contour model is proposed that is particularly suited for real-time tracking. The underlying mathematics is based on the Euclidean curve shortening evolution which defines the gradient direction in which a given curve is shrinking as fast as possible relative to Euclidean arc-length,[21] and on the theory of conformal metrics. The Euclidean arc-length is multiplied by a conformal factor defined by the features of interest, and then the corresponding gradient evolution equations are computed. Therefore, the features to be captured lie at the bottom of a potential well to which the initial contour will flow. Let $C = C(p,t)$ be a smooth family of closed curves where *t* parameterizes the family and p the given curve. The basic idea is to change the ordinary Euclidean arc-length function by multiplying by a conformal factor ϕ that is assumed to be a positive, differentiable function. The resulting conformal Euclidean metric is given by multiplying the ordinary Euclidean metric by the conformal factor. As in ordinary curve shortening,[21] the corresponding gradient flow for the modified length functional is:

$$L_\phi(t) = \int_0^L \phi dv \tag{5}$$

where *dv* is the ordinary Euclidean arc-length. Taking the variational derivative and integrating by parts, the gradient flow is found to be:

$$\frac{\partial C}{\partial t} = \left(\phi\kappa - \nabla\phi \cdot \mathbf{N}\right)\mathbf{N} \tag{6}$$

where $\kappa, \mathbf{N}$ denote the curvature and (inward) unit normal respectively. Regularity may be deduced from the classical curve shortening case, and consequently there will be convergence to a closed geodesic in the plane relative to the conformal Euclidean metric. Finally, if one wants to put in an area-minimizing term,[20] then the complete equation becomes:

$$\frac{\partial C}{\partial t} = \left(\phi\left(\kappa + \nu\right) - \nabla\phi \cdot \mathbf{N}\right)\mathbf{N} \tag{7}$$

In the latter equation, $\nu > 0$ is called the *inflationary parameter*, and is user-determined.

In the tracking application described in this paper, it was essential to make an algorithm fast enough to be used in real-time. Therefore, a simplified version of Eq. (7) was used, where the curvature term is neglected and ν is set to 1. This corresponds exactly to the area-minimizing flow,[20] and can be very quickly implemented using the fast-marching method as described in Refs. 22 and 23. The function $\phi(x, y)$ is constructed to select high contrast edges. Accordingly,

$$\phi(x, y) = \frac{1}{1 + \|\nabla I\|^2} \tag{8}$$

was chosen where $I(x,y)$ is the intensity of the image. To make a faster algorithm the smoothness of the evolving curve is also ignored. The curve moves with constant speed, and the evolution is stopped when the stopping term is sufficiently "small" (defined by a predefined threshold value). Therefore, it is possible to eliminate the higher order derivative requirements needed for the curve evolution. If the function $\phi(x, y)$ satisfies the stopping condition, the curve is frozen in a neighborhood around the given pixel; this preserves the edges during the evolution.

The geometric active contour method is used to get multiple contours from each image, and so potentially capturing multiple building openings. The contours are broken into individual features, and then geometric characteristics are extracted from the individual contours, which allow the tracker to determine which objects are most likely windows. The first characteristic examined is whether or not the contour is closed; open contours are rejected. The second characteristic is contour length. Valid window contours have relatively small-range of possible lengths; contours that are far longer or shorter than expected for windows and doors are ruled out. If the contour passes these two preliminary filters, the more computationally expensive geometric characteristics are calculated and used in the window tracker. Four sets of geometric data are determined: the center of mass, the interior area, the corner locations, and the darkness of the interior. The center of mass is found by averaging the pixel locations over the interior area. Once this is determined, the locations of the corners are found by finding the most distant point from the center of mass in each quadrant. Using these corners, it is possible to determine a characteristic rectangle for the contour, where the width is the average width between corners and the height is the average height between corners. Then the area of the contour can be calculated from the quadrilateral, and the darkness is determined by averaging the grayscale values of each pixel inside of the quadrilateral.

The window algorithm was tested using the same practice images as were used for the symbol. This tested the ability of the filter to eliminate noise in images that did not contain windows as well as for the entire algorithm to work on images that did contain windows. Figure 8 shows the results of the image filter with the centroid and corner results for the windows that were found using the geometric active contours. Since the initial contour is a circular region, only a portion of the image can be evaluated at one time, causing the system to miss all of the openings on the upper floor, and some on the lower floor.

D. Window Tracking

Window tracking requires a more sophisticated algorithm than the symbol tracking, for it must be 3-dimensional. Since each building has multiple stories, it is important to capture the height of the window as well as its geographical (horizontal) location. Secondly, since the windows are close to each other, their positions must be determined with far greater precision in order to ensure that measurements are not incorrectly lumped together in tracking. Also, the sizes of the openings are unknown. In this case, approximate building locations are already known based on public surveys of the McKenna site, including satellite photography. Therefore, the absolute position of the window can be determined by mapping it onto the visible wall.

Figure 9 is a diagram showing the variables used to determine the position of the window. The intersection point on the candidate wall is evaluated to see if it lies between the corners of the wall and at a height that is legitimate for the building. Since errors in the location of the building or in the position estimate of the vehicle will result in an error in height estimate for the window, the height requirements included a margin of error above and below the building of 10 ft.

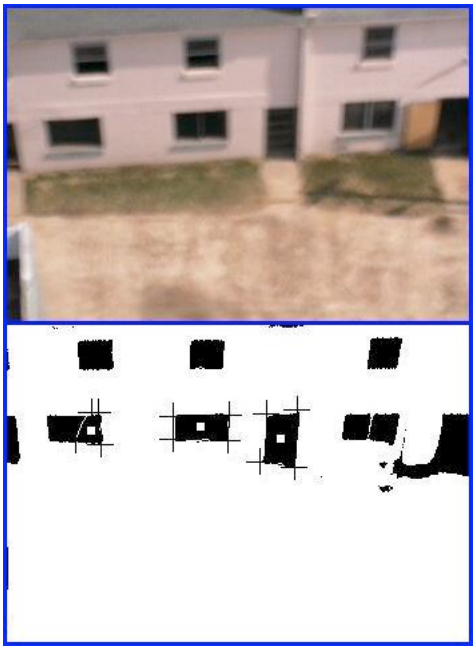

Fig. 8 Identification of the geometric characteristics of the contours.

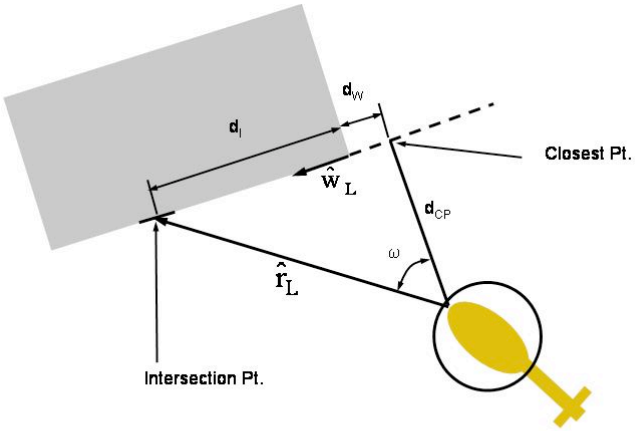

Fig. 9 Geometry for determining the window position.

IV. Mission Management

In the mission management process proposed here, the mission is broken up into phases. Each phase is characterized by setting subsystem modes, and the generation (or selection), loading, and execution of one or more flight plans, and an automated decision, as illustrated in Fig. 10. The phase is over and the next started when the automated mission decisions is made to do so. This structure allows a variety of realistic missions to be managed, including automated decisions that can alter the behavior of the system for the remainder of the flight.

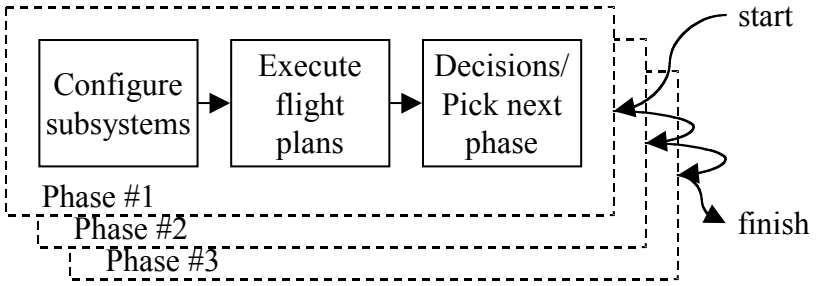

Fig. 10 Mission management process is broken up into phases. Mission decisions occur at the boundary of mission phases, including choice for next mission phase. The output of the decision is an input the following mission phase.

For the reference mission described in Sec. I, a mission management software component was developed to generate vehicle flight plans and to trigger mode changes for the image processing, tracker/mapping module, and UAV autopilot. A single command is required by the human operator to start the mission manager, and all subsequent actions were done with the human operator acting as observer. A single command by the human operator can stop or pause the mission at any time, and put the helicopter in an automatic hover mode.

The McKenna MOUT site is a well-documented area, and the geographic coordinates of each building are known, information that might be available for most any area in the world from satellite imagery. Using this *a priori* information, the approach to the mission is to pre-specify a flight plan for the vehicle that includes a camera view of all walls of each building within the search area (in this case, 15) and that remains clear of obstacles and terrain. The system processes each camera image, looking for the identifying symbol in Fig. 1, and updates a tracker. The tracker contains a list of candidate symbol locations and orientations that is continuously updated based in processed image data. After the flight path is completed, the top symbol candidate is utilized to select the "marked" building. Then, a new flight plan is automatically generated that examines this same building again in a similar manner, this time the walls of the selected building are searched, and the search is for openings rather than the symbol. The top candidate opening is then selected. These three mission phases for this reference mission are described below, and illustrated in Fig. 11.

Building/symbol Search Phase: The image processor and tracker are switched to the symbol detection and mapping mode and reset. A pre-specified search pattern for each of the 15 buildings is executed one at a time in a pre-specified order. Each new search begins once the previous one is completed. Each search consists of approximately four waypoints, one for each "corner" of the building. The helicopter is told to fly sideways, with the camera pointing out in front of it with a 45 degree down tilt angle, and with a pan angle to correct for any heading tracking error exhibited by the autopilot (such as might happen with a sufficiently strong wind to overpower the tail rotor). Following completion of the above mission phase, the tracker/mapping results are sorted to determine the most confident symbol location that was also sufficiently close to one of the 15 buildings. This represents the first major automated decision for the mission manager. If a building is selected, then the opening-search phase is initiated. If no valid symbol is found, then the mission manager transitions to a termination phase (not shown in the figure), typically an automatic landing at a pre-specified location.

Opening Search Phase: The building associated with the selected symbol track becomes the subject of an opening search. The image processor and tracker are switched to the window/door detection and mapping mode and reset. The search flight plan associated with that building is loaded and executed. Following completion of the above mission segment, the tracker/mapping results are sorted to determine the most confident opening, based on criteria discussed in Sec. III. The location and orientation of the selected opening becomes the input to the camera-pointing phase, which is initiated upon selection.

Point Camera at Opening Phase: Two things would then happen in parallel. First, a flight plan and camera control commands are generated to fly the helicopter to an appropriate spot and point the camera at the selected opening (with the opening in the center of the image). Second, a text file is created and communicated to the GCS computer giving the location of the selected building and the location and orientation of the selected opening. Both of these actions enable verification of which building and opening the system has chosen.

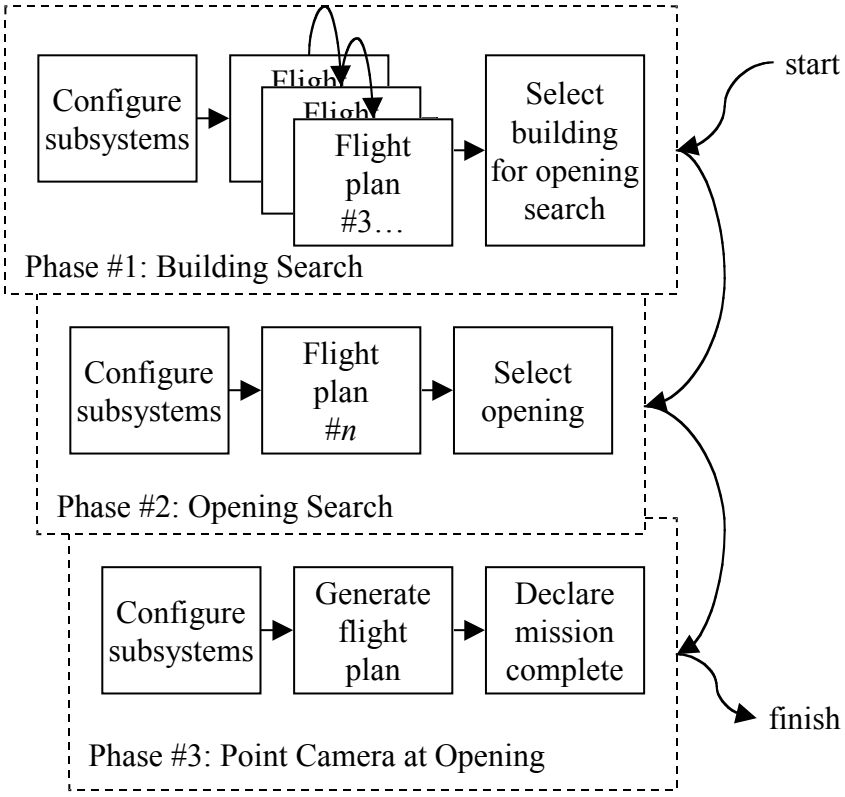

Fig. 11 For the reference mission, three mission phases are utilized, executed in sequence. The result of the building search is the input to the opening search (which building to search), and the result of the opening search is an input to the camera-pointing phase.

The mission management subsystem was tested initially in simulation, including an array of tests including all possible potential buildings that could have the specified symbol, and degenerate cases, such as those where the symbol is not found. More complex breakdowns of the mission phases where explored, such as having the system search for openings as soon as it saw a possible symbol, but it was generally true that minimizing the number of decision points (in this case, two) increased probability of mission completion by avoiding the possibility of incorrect decisions.

V. Full-System Results

The complete system described was tested in several ways. First, simulated images were presented to the image processing from a scene generator in the simulator. In these tests, the image-processing job is considerably easier, given the clean characteristics of these synthetic images. However, this is very good test of the mission management, object tracking, and flight planning components, as well as elements of the hardware. Three tests were performed utilizing the vision system hardware searching all 15 buildings of the simulated McKenna MOUT village, and the correct building was selected and a valid opening was found, illustrated in Fig. 12.

Next, tests were conducted at the actual McKenna village. The initial flight tests were done with a search area including three of the buildings. In each case, the symbol was placed on a different building. On all of these tests, the system selected the correct building among the three choices. Following this, four attempts were made to search the entire village of 15 buildings. On the first, the actual symbol was missed (missed detection); thrown out because it was considered too bright, probably due to bright sunlight directly on the symbol unlike all previous tests. The nominal distribution range for the symbol score was corrected to account for increased potential brightness. On all three subsequent tests the correct building was selected, with the symbol appearing for approximately 5 seconds over a total search flight time of approximately 15 minutes.

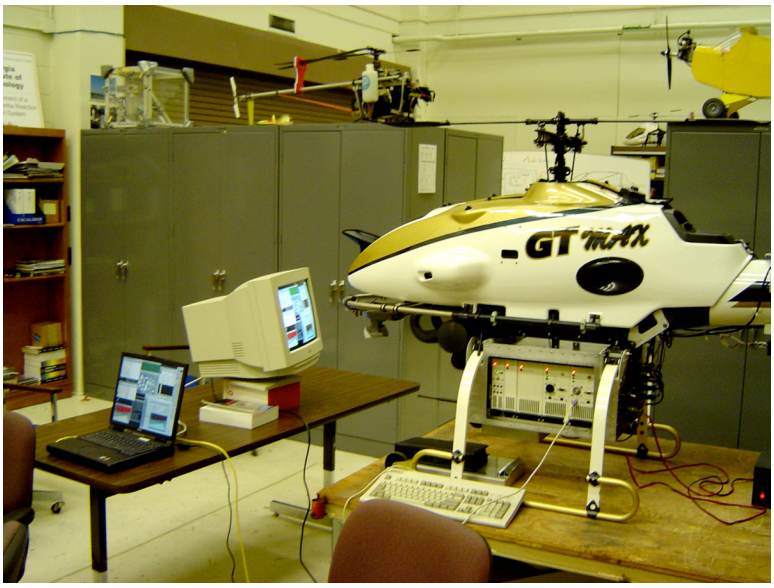

Fig. 12 Simulated and recorded-actual images presented to the image processing and mission management subsystems in laboratory testing (camera under nose of helicopter has lower right of white computer monitor in view)

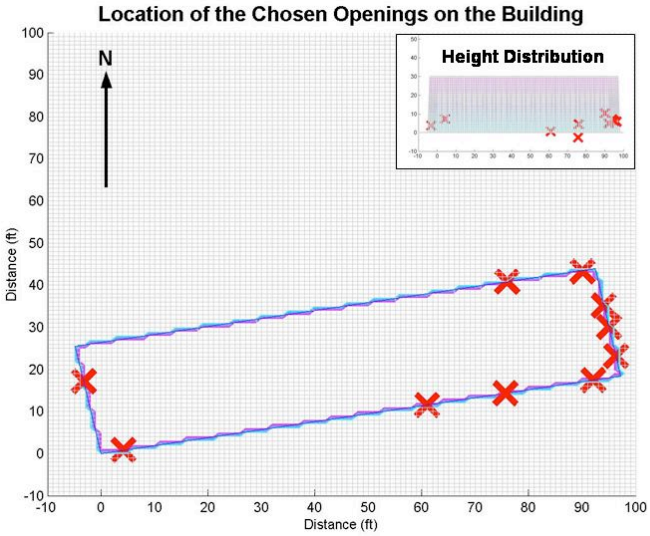

Fig. 13 Results of one flight which shows the location of the openings mapped onto the building

A typical result for the opening search on a single building is shown in Fig. 13. This plot shows a representation of the building with the ten best opening locations from the tracker superimposed. From Fig. 13, it can be seen that the openings are correctly mapped onto the walls of the building, and these positions match actual opening locations accurately. In this case, the aircraft circled the building in a clockwise manner starting at on the northern face. The majority of the openings picked are located on the eastern side; this is possibly due to favorable lighting conditions creating a better contrast and resulting score.

After searching the selected building for an opening, the UAV hovered in front of the selected opening, pointing the camera at the opening on each of the final three tests. Figure 14 shows the ground track from a flight as seen at the operator monitoring station, and the general pattern that can be seen is an orbit around each building. There are two orbits of the top center building, due to the fact that this was the building that had the symbol on it. Figure 15 shows an onboard image hovering in front of the opening at the end of the mission, in this case an open door. The

following video shows the recorded onboard images at five-times speed for one entire mission (the same one corresponding to Fig. 15): Click here to see video. The system described in this paper is the only such system to have completed this Level 2 reference mission to date (the AUVSI competition).

Fig. 14 Ground track during mission attempt, start is coming in from the top, finish at circle near center of image. The building with symbol is top center, and is circled twice.

Fig. 15 Camera placed with selected building opening in center, in this case an open door.

VI. Conclusions

This paper described the design, development, and testing of an Unmanned Aerial Vehicle (UAV) with automated capabilities: searching a prescribed area, identifying a specific building within that area based on a small sign located on one wall, and identifying an opening into that building. An approach was proposed to lump the required automated decision making to the boundary of mission phases, which were defined *a priori*. The resulting system was able to search a 15-building village and make its own make automated mission decisions with speed comparable to a human operator searching on foot, or with a conventional remotely piloted aircraft, in a highly realistic test. The lessons learned from the process of bringing a mission manager of this type though such a test program related to keeping it simple: minimize the number of automated decisions, minimize the number of mission phases, and chose automated decisions that can be based on potential processed sensor information. This study also highlights the importance of using a combination of simulation, synthetic images, recorded flight images, and flight tests to develop and evaluate these types of highly-automated UAV systems.

Acknowledgments

This work was supported in part by the DARPA Software Enabled Control (SEC) Program under contracts #33615-98-C-1341 and #33615-99-C-1500, AFOSR Multi-University Research Initiative F49620-03-1-0401, Lockheed Martin, NovAtel, Bahr Avionics, and Inflight Robotics. Other direct contributors include: Henrik Christophersen, J. Eric Corban, Joerg Dittrich, Jeong Hur, Suresh Kannan, Sumit Mishra, Wayne Pickell, Daniel Schrage, and Hungsun Son. The operations at the McKenna MOUT site required the support of McKenna personnel and AUVSI.

References

[1]Proctor, A. A., Kannan, S. K., Raabe, C., Christophersen, H. B., and Johnson, E. N., "Development of an Autonomous Aerial Reconnaissance System at Georgia Tech," *Proceedings of the Association for Unmanned Vehicle Systems International Unmanned Systems Symposium & Exhibition*, 2003.

[2]Dooley, J., Taraloba, E., Gabbur, P., and Spriggs, T., "Development of an Autonomous Aerial Reconnaissance System," *Proceedings of the Association for Unmanned Vehicle Systems International Unmanned Systems Symposium & Exhibition*, 2003.

[3]Burleson, W. L., Keller, K., Harrison, G., and Corliss, F., "Southern Polytechnic State University Autonomous Remote Reconnaissance System," *Proceedings of the Association for Unmanned Vehicle Systems International Unmanned Systems Symposium & Exhibition*, 2003.

[4]Johnson, E. N., and Schrage, D. P., "System Integration and Operation of a Research Unmanned Aerial Vehicle," *Journal of Aerospace Computing, Information, and Communication,* Vol. 1, No. 1, Jan. 2004.

[5]Yuan, X., Ganesan, K., Evett, M., and Smith, S. M., "Providing Real-Time Data Trajectory Access in Autonomous Underwater Vehicles," *IEEE/MTS Oceans Conference,* 1999.

[6]Smith, C. M., Leonard, J. J., and Feder, H. J. S., "Making Difficult Decisions Autonomously: The Impact of Integrated Mapping and Navigation," *Proceedings of the Workshop on Autonomous Underwater Vehicles*, 1998.

[7]Penn, B. S., "Goal Based Mission Planning for Remotely Piloted Air Vehicles (RPAVs)," *Aerospace and Electronics Conference*, 1989.

[8]Schaefer, P., Colgren, R. D., Abbott, R. J., Han Park Fijany, A., Fisher, F., James, M. L., Chien, S., Mackey, R., Zak, M., Johnson, T. L., and Bush, S. F., "Reliable Autonomous Control Technologies (ReACT) for Uninhabited Air Vehicles," *IEEE Aerospace Conference*, 2001.

[9]Arakawa, A., Hiyama, M., Emura, Y., and Kagami, Y., "Trajectory Generation for Wheeled Mobile Robot Based on Landmarks," *IEEE International Conference on Systems, Man and Cybernetics*, Oct. 1995.

[10]Liou, R., and Azimi-Sadjadi, M. R., "Multiple Target Detection Using Modified High Order Correlations," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No.2, Apr. 1998, pp. 553-568.

[11]Bay, J. S., "A Fully Autonomous Active Sensor-Based Exploration Concept for Shape-Sensing Robots," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 21, No. 4, July/Aug. 1991, pp. 850-860.

[12]Schreur, J. M., "B737 Flight Management Computer Flight Plan Trajectory Computation and Analysis," *Proceedings of the American Control Conference*, 21-23 June 1995.

[13]Dittrich, J. S., and Johnson, E. N., "Multi-Sensor Navigation System for an Autonomous Helicopter," *Proceedings of the 21st Digital Avionics Systems Conference*, 2002.

[14]Johnson, E. N., and Kannan, S. K., "Adaptive Flight Control for an Autonomous Unmanned Helicopter," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2002.

[15]MVTec, *Halcon/C++ Reference Manual*, MVTec Software GmbH, München, Germany, 2003, Available online at http://www.mvtec.com (cited Dec. 2004).

[16]Tannenbaum, A., "Three Snippets of Curve Evolution Theory in Computer Vision," *Mathematical and Computer Modeling*, Vol. 24, 1996, pp. 103-119.

[17]Malladi, R., Sethian, A., and Vemuri, B., "Shape Modeling with Front Propagation: A Level Set Approach," *IEEE Transactions on Pattern Analysis*, Vol. 17, 1995, pp. 158-175.

[18]Kichenassamy, S., Kumar, A., Olver, P. J., Tannenbaum, A., and Yezzi, A., "Conformal Curvature Flows: From Phase Transitions to Active Vision," *Archive for Rational Mechanics and Analysis*, Vol. 134, 1996, pp. 275-301.

[19]Caselles, V., Kimmel, R., and Sapiro, G., "Geodesic Active Contours," *International Journal of Computer Vision*, Vol. 22, 1997, pp. 61-79.

[20]Lauziere, Y., Siddiqi, K., Tannenbaum, A., and Zucker, S., "Area and Length Minimizing Flows for Segmentation," *IEEE Trans. Image Processing*, Vol. 7, 1998, pp. 433-444.

[21]Grayson, M., "The Heat Equation Shrinks Embedded Plane Curves to Round Points," *Journal of Differential Geometry*, Vol. 26, 1987, pp. 285-314.

[22]Sethian, J. A., *Level Set and Fast Marching Methods*, Cambridge Univ. Press, 1999.

[23]Osher, S., and Fedikiw, R., *Level Set Methods and Dynamic Implicit Surfaces,* Springer-Verlag, 2003.