Unmanned Aerial Vehicles: Challenges and Technologies for Improved Autonomy

GEORGE VACHTSEVANOS & BEN LUDINGTON
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0250
USA
http://icsl.marc.gatech.edu

Abstract: This paper describes recent advances in Unmanned Aerial Vehicles aimed at improving their autonomy and reliability for a diverse domain of applications from forest fires and rescue operations to military and border patrol missions. We introduce an experimental VTOL aircraft and discuss instrumentation requirements and control strategies that are essential to maintain stable operations as the vehicle executes extreme maneuvers. We review next mission scenarios that require the deployment of multiple UAVs and suggest briefly mission planning and control algorithms to achieve such challenging tasks. The control technologies are demonstrated through flight testing that illustrate the effectiveness and robustness of the planning and control strategies.

Key-Words: UAV, Mission Planning, Mode Transitioning, Fault Tolerant Control.

1 INTRODUCTION

Recent world events have highlighted the utility of unmanned aerial vehicles (UAVs) for both military and potential civilian applications. However, the reliability of these systems has been disappointing in practice. According to a recent report, nearly half of the current-generation unmanned surveillance aircraft built have been lost. This loss-rate is about ten times worse than manned combat aircraft. Clearly these numbers are driven in part by the dangerous missions these aircraft are tasked with, but there are other factors at work here. In manned aircraft, the pilot functions as the central integrator of the onboard subsystems and works to mitigate problems when they occur. Although “human error” is attributed as the most common cause of aviation accidents, human pilots are also simultaneously the most important safety-enhancing component on a manned aircraft.

To address this and other related UAV control issues, the Defense Advanced Research Projects Agency (DARPA) and the U.S. Air Force Research Laboratory (AFRL) have launched a major initiative to develop revolutionary new Software Enabled Control (SEC) systems with applications to intelligent UAVs [5].

Beyond the responsibility of responding to unexpected system faults, the SEC program is also charged with making these machines more agile, thus helping them avoid hostile actions without exceeding critical flight parameters. This has the potential to improve the loss-rate for even the most dangerous missions.

Improved performance of UAVs is expected to be achieved when such vehicles are endowed with levels of autonomy that will allow them to operate safely and robustly under external and internal disturbances, to be able to accommodate fault conditions without significant degradation of their performance, to adapt to unexpected events and to coordinate/cooperate among themselves to accomplish mission objectives. Fig. 1 depicts the expected UAV autonomy capabilities according to the U.S. DoD’s UAV autonomy roadmap [4].

Fig. 1. Autonomous control level trend

In this paper, we suggest the hardware, software and control technologies aimed to achieve such autonomy objectives.
A follow-up program currently underway is involved with tasking multiple UAVs to execute intelligence, reconnaissance and surveillance missions in an urban environment. Mission planning and target tracking methodologies constitute specific objectives of this effort.

2 MISSION INTELLIGENCE FLOW

A hierarchical control structure for mission intelligence flow is illustrated in Fig. 2. Situation Awareness is used for Mission Planning and Flight Mode Selection which constitutes the high level control elements. For inhabited aircraft the pilot and other crewmembers provide the intelligence for interpreting the data from a variety of sources to execute these functions. Much of this data is used in pre-flight or pre-mission planning and is updated on-board as the mission proceeds. As the mission segments are executed and abnormal events are encountered, Flight Mode Switching takes place which constitutes the mid level control element. On an inhabited aircraft the pilot flies the aircraft and makes necessary mode switching and control reconfiguration decisions for implementation through the use of the Flight Control System. This constitutes the low level control element and is used to execute the smooth transition between modes of flight, i.e. transition from hover or takeoff to level flight, etc., and stay within the flight envelope of the UAVs. External Abnormal Conditions cause the pilot to take corrective action, such as avoiding an obstacle or evading a target or threat. Internal Abnormal Conditions reconfiguring his/her set of controls to safely continue to fly or land the aircraft.

Without a pilot onboard the aircraft a UAV must either be controlled from the ground by a radio control ground pilot or the UAV must have its own intelligence to fly autonomously. Executing a VTOL (vertical take-off and landing) UAV mission autonomously has been demonstrated by both the Georgia Tech and Sikorsky Aircraft UAVs in the Army's Advanced Scout Rotorcraft Testbed (ASRT) Project [15]. However, both of the aircraft weren't able to use the entire flight envelope capability of the UAVs, largely limited by the control algorithms implemented. In addition, the control algorithms were very much customized for the particular vehicle's characteristics and were developed in very much of a trial and error approach. Also, the computing architecture onboard the aircraft did not provide the environment for reusability and reconfigurability, let alone for plug and play of different SEC algorithms [16].

3 MISSION PLANNING

Fig. 3 depicts the configuration of the mission planner. The high level supervisory controller receives mission commands from the command and control post and decomposes them into sub-missions which will then be assigned to connected function modules. Upon reception of start and destination points from the supervisory controller, the route planner generates the "best" route in the form of waypoints for the UAV to follow. A database of the terrain in the form of a digitized map is available to the route planner.

The configuration of the route planner is depicted in Fig. 4. The digitized map is in the form of a mesh of equal square cells, where each cell is either free or occupied by an obstacle. Having two free cells (i.e., a start and a destination) assigned by the supervisory module, the A* search engine searches the map mesh and plans a cell-based route that extends from the start cell to the destination cell and avoids stationary obstacles. The generated cell route is suboptimal in terms of distance, safety, and maneuvering (e.g., turning angles), i.e.

\[
\text{route cost} = (W_d \ast \text{distance}) + (W_h \ast \text{hazard}) + (W_m \ast \text{maneuvering}),
\]

where \(W_d\), \(W_h\), and \(W_m\) are weights for the three cost components, and are assigned by the supervisory module based on the objectives and the circumstances of the mission.

The cost elements are expressed as fuzzy membership functions reflecting the inherent uncertainty associated with the planned trajectory, the obstacles along the path and the maneuvers the vehicle is required to perform as it navigates through the terrain. A* uses heuristic knowledge about the closeness of the goal state from the current state to guide the search. The cost of every searched cell, \(n\), is composed of two components:

\[
\text{cost}(n) = kg(n) + kh(n)
\]

where \(g(n)\) is the cost of the least-cost route (found in the search so far) from the start cell to cell \(n\), \(h(n)\) is the
heuristic (i.e., estimated) cost of the minimum-cost route from cell \( n \) to the destination cell, and \( kg, kh \) are weighting factors for \( g(n) \) and \( h(n) \), respectively. Given a search state space, an initial state (start node) and final state (goal node), A* will find the optimal (least cost) path from the start node to the goal node, if such a path exists [17]. The generated cell route is further optimized and smoothed by a filtering algorithm.

![Mission Planning Configuration](image1)

**Fig. 3** The mission planning configuration.

![Route Planner Configuration](image2)

**Fig. 4.** The route planner configuration.

![Four Planned Unfiltered Routes](image3)

**Fig. 5.** Four planned unfiltered routes: (a) minimum distance; (b) minimum hazard (or maximum safety); (c) minimum maneuvering; (d) minimum (distance + hazard).
The filtered route is a series of consecutive waypoints that the UAV can navigate through.

The supervisory module reads the objectives and the status of the mission and based on that it configures the search engine and assigns weights to the route’s three cost components. Furthermore, the supervisory module chooses the start and the destination cells for the search engine depending on the current status of the UAV, i.e., whether it is stationary or already navigating towards a destination and needs to be redirected to another destination. The learning-support module acquires route cost data from the search engine at certain map landmarks and updates a cost database that is used later to provide better heuristics to guide the search engine.

Figure 6 illustrates in a flow chart the route planning implementation steps. Typical route planning results for an UAV with actual mapping data is shown in Fig. 5. The interested reader can refer to [1] for more details regarding the route planner algorithms and design.

4 FLIGHT TESTING: GTMAX RESEARCH UAV

The GTMax research UAV system, Figure 7, developed at the Georgia Institute of Technology to support SEC and other ongoing programs, utilizes a Yamaha R-Max helicopter, a modular/open avionics system, Open Control Platform, a set of baseline onboard software, and a series of simulation tools. The baseline systems enable autonomous flight of the normally remotely piloted aircraft. The R-Max configured with these systems is known as the GTMax, a highly effective UAV research vehicle that has a design based on lessons-learned from UAV research at academic institutions such as Georgia Tech, University of California at Berkeley, Massachusetts Institute of Technology, and Carnegie Mellon University for more than ten years.

Developed in Japan, the basic Yamaha R-Max helicopter has a rotor diameter of 10.2 feet, a 21 hp two-cylinder engine, and weighs 125 pounds. The weight increases to 160 pounds when configured with typical GTMax avionics. It is capable of carrying approximately 50 additional pounds of research equipment. It also has a generator, starter, and can be flown manually by a remote pilot in sight of the helicopter or by an onboard autopilot.

- Basic Yamaha R-Max Dimensions:
  - Max. Length: 3630 mm. (Rotor blade included)
  - Fuselage Length: 2750 mm.
  - Width: 720 mm.
  - Height: 1080 mm
  - Fuel Tank: 6 Liter
  - Main Rotor Diameter: 3115 mm.
  - Tail Rotor Diameter: 545 mm.
  - Max Gross Weight: 93* g N.
    - Max. Payload: 30* g N.

- Powerplant:
  - Type: Gasoline 2 cycles
  - Cylinder configuration: Horizontal opposition 2 cylinder
  - Displacement: 246cc.
  - Engine RPM: 6350 RPM (Nominal)
  - Max Power Output: 15.4 KW (21PS)
  - Max Torque: 25.5Nm
  - Cooling Type: Liquid Cooling
  - Fuel: Auto Gas

The GTMax avionics system hardware consists of a set of modules that can be added/removed as required for a flight test. All modules include electro-magnetic interference protection and their own power regulation. The modules are mounted in a vibration-isolated rack within an enclosure under the fuselage. The basic system includes a general-purpose computer, Differential Global Positioning System (D-GPS), an inertial measurement unit, an ultra-sonic altimeter, a 3-axis magnetometer, and two wireless data links. Other flight configurations used to date have also utilized a second general purpose computer, cameras, a radar

---

**Figure 6. Flow chart of the route planner**

**Figure 7. GTMax helicopter in flight, Georgia Tech's test bed research UAV.**
altimeter, and video capture/compression hardware. The basic ground equipment includes the data links, a GPS reference station, and one or more laptop computers.

The baseline onboard software [9] includes interfaces to the sensors, an integrated navigation filter, and a nominal trajectory-following autopilot. This allows the baseline system to fly a prescribed mission on its own, including the takeoff and landing. The role of the human operator can be to set the desired flight path, start the engine, monitor the flight, and to shut down the engine after landing. This enables a large number of relevant flight control scenarios to be tested, from purely manual control to autonomous operations.

Most high and middle level control components are running on the secondary computer. The implementation of these controls modules and the interface and communication with the primary computer are based on the Open Control Platform.

5 UAV ADAPTIVE MODE TRANSITION CONTROL

Control of autonomous aerial vehicles presents unique challenges not only in the design of control algorithms, but also in the strategies and methodologies used to integrate and implement those algorithms on the actual vehicles. We propose an approach to the adaptive mode transition control of UAVs. The main objective of this architecture is to improve the degree of autonomy/intelligence of the UAV and its performance under uncertain conditions, for instance when external perturbations are present. The architecture is based on concepts developed in [11-14] where the adaptive mode transition control scheme was first introduced. Here, we suggest a new approach to the adaptive mode transition control problem and we are introducing a hierarchical architecture to implement it. The algorithms have been implemented and tested using the Open Control Platform and validated by flight tests on GTMax.

Here are some definitions used in this section: a local mode is a region of the state space around an operating point in which the vehicle exhibits quasi steady state behavior. A local controller is a controller that guarantees the stability and tracking performance of the closed loop system for any feasible reference trajectory in a local mode. A transition region is a region of the state space outside any local mode that includes all the feasible trajectories between two local modes. The operating region is the region of the state space generated by the union of all the local modes and transition regions.

The proposed architecture for the control of UAV’s consists of a hierarchy of three levels as depicted in Fig. 8. At the highest level, the mission planning component stores information about the overall mission, generates a low level representation of that mission, and coordinates its execution with the middle level. The middle level includes a trajectory planning component, which receives information from the high level in terms of the next task to be executed to fulfill the mission, and generate the trajectory (set points) for the low level controller. Mode transition manager (MTM) coordinates mode selection, switching and transition automatically based on the actual state of the vehicle. At the lowest level, an adaptive mode transition controller coordinates the execution of the local controllers (one for each local mode) or the active control models (one for each transition), which stabilize the vehicle and minimize the errors between the set points generated by the middle level and the actual state of the vehicle. The adaptive mode transition control consists of the mode transition control component and the adaptation mechanism component. Fig. 9 shows the structure of the proposed Adaptive Mode Transition Control (AMTC) algorithm, where $C_i$ is the controller for local mode $i$. 

![AMTC Hierarchical Control Architecture](image1)

![Adaptive Mode Transition Control](image2)
5.1 Mode Transition Manager

The mode transition manager coordinates the transitions. Unlike the previous work [11-14] where the transitions were pre-scheduled and a Mode Selector module coordinated the transitions, the MTM coordinates the transitions automatically based on the actual state of the vehicle. In order to accomplish this task, a Mode Membership Function is defined for each local mode and the MTM determines which local mode or transition should be activated relying upon these constructs [6-8].

For local mode $i$ the Mode Membership Function is defined as

$$
\mu_i = e^{-(x-m_i)^T \Sigma_i^{-1} (x-m_i)}
$$

where $x$ is the state of the vehicle, $m_i$ is the center (operating state) of the mode, and $\Sigma_i$ is a positive semi-definite diagonal matrix whose elements represent the inverse of the deviations for each component of $x$ for that mode.

To determine which mode is active, the MTM computes the Mode Membership Functions for all local modes. If $\mu_i(x(k)) \geq 0.5$ for the actual state, then local mode $i$ will be active. Mode centers and deviations are defined so that $\mu_i(x(k)) \geq 0.5$ can be valid for only one $i$. That way the modes correspond to disjoint regions of the state space. If $\mu_i(x(k)) < 0.5$ for all $i$, then the transition corresponding to the two modes with the highest Mode Membership Function values will be active.

When a local mode is active, the corresponding local controller is used to compute the control output whereas when a transition is active, the corresponding active control model (ACM) is used to compute the control output. When a faulty mode is detected by the fault detection and identification component, the system will transition to a fault tolerant control mode, where corresponding control reconfiguration tasks are executed.

5.2 Implementation and Flight Test Results

The architecture has been implemented using the OCP. Fig. 10 shows a software-in-the-loop simulation environment used to implement the architecture. Hardware-in-the-loop simulations and flight tests have been performed to validate the control algorithms. Fig. 11 shows the flight test results of a rectangular flight.

6 FAULT TOLERANT CONTROL

UAVs are often subjected to failure modes that may lead to a catastrophic event resulting in loss of the vehicles. It is desired, therefore, to develop and implement technologies that will detect and identify in a timely manner onboard failure modes and reconfigure the available control authority so that the vehicle maintains an acceptable level of performance for the duration of the emergency [3]. The fault tolerant control architecture implemented on the GTMAX is designed to accommodate multiple fault modes without degrading the performance of the nominal system. The architecture improves reliability by integrating Fault Detection and Identification (FDI) and Reconfigurable Flight Control (RFC).

Implementation of the architecture utilizes a three-tier hierarchical control scheme implemented in the OCP. The architecture is a variation of the scheme developed by Clements [2, 3]. Each level of the hierarchy adds autonomy to the vehicle. FDI takes place...
at the highest level of the hierarchy and directs actions at each subordinate tier. After the FDI module issues a fault declaration, the fault tolerant control module issues reconfiguration commands to the controllers at the lowest tier of the hierarchy. Reconfigurable flight controllers reside with the baseline flight controller at the lowest level of the hierarchy. The low-level controllers generate the control inputs to achieve the vehicle’s desired flight path. In the event of a malfunction, reconfigurable flight controllers enable the vehicle to recover some degree of the performance from the impaired system.

The architecture implemented on the GTMAX was designed to combat faults in the flight control actuators. Malfunctions in the flight control hardware were selected for this study because they challenge both components of the fault tolerant control architecture, FDI and RFC. Specifically, a malfunction in the main rotor collective actuator was examined, but the architecture is readily expandable to accommodate additional faults. The design identifies the occurrence of a fault from a finite set of pre-determined faults. It then applies the appropriate reconfiguration to stabilize the vehicle. The fault tolerant architecture assumes the following actuator model:

\[
\delta = \max(\min(k\delta_{\text{com}} + b, s_{\text{max}}), s_{\text{min}})
\] (4)

where faults can affect the actuator gain \( k \): \( 0 \leq k \leq 1 \), bias \( b \), or saturation levels, \( s_{\text{min}} \) and \( s_{\text{max}} \). Faults, such as floating actuators, where the parameters vary constantly following the occurrence of the fault are not considered. To accommodate these cases, additional hardware on the vehicle could be employed to immobilize the actuator creating a stuck actuator condition. Flight test results demonstrate that the fault tolerant architecture can accommodate stuck actuator malfunctions with \( k=0 \).

Flight test results validate the effectiveness of the approach. The flight demonstration was initiated with the UAV in its baseline configuration. The aircraft was commanded to execute a 70-foot descent from a stationary hover. During the descent, the stuck collective fault was applied binding the collective in a typical descent position that was determined from flight data on the day of the flight test. The state-dependent neural network FDI routine detected the fault and activated system restructuring. Referencing Fig. 12, the descent is initiated at 28 seconds; the fault is applied at 30 seconds, and the fault is detected prior to 34 seconds. Without reconfiguration, the vehicle would not have been able to arrest its descent.

Fig. 11. AMTC algorithms flight test result
flying in formation in order to take advantage of their tested and implemented if such vehicles will perform challenges to the designer and the end user. They opens now avenues of research where the intelligent complementary capabilities. The UAV swarm problem of smart coordination/cooperation technologies. control community can contribute significantly in terms of modeling and control seems to provide possible solutions. The UAV community is accomplishing major milestones towards this goal but key R&D concerns remain to be addressed. More recently, researchers have been concerned with multiple and heterogeneous UAVs flying in formation in order to take advantage of their complementary capabilities. The UAV swarm problem opens new avenues of research where the intelligent control community can contribute significantly in terms of smart coordination/cooperation technologies.

7 CONCLUSION

Unmanned Aerial Vehicles present major challenges to the designer and the end user. They require new and novel technologies to be developed, tested and implemented if such vehicles will perform actual missions reliably and robustly. Autonomy stands out as the key requirement with enabling technologies to allow such vehicles to operate safely in unstructured environments within their flight envelope, to accommodate subsystem/component failure modes without major performance degradation or loss of vehicle and to perform extreme maneuvers without violating stability limits. An integrated/hierarchical approach to vehicle instrumentation, computing, modeling and control seems to provide possible solutions. The UAV community is accomplishing major milestones towards this goal but key R&D concerns remain to be addressed. More recently, researchers have been concerned with multiple and heterogeneous UAVs flying in formation in order to take advantage of their complementary capabilities. The UAV swarm problem opens new avenues of research where the intelligent control community can contribute significantly in terms of smart coordination/cooperation technologies.

References:


