AN INTELLIGENT CONTROL ARCHITECTURE FOR UNMANNED AERIAL SYSTEMS (UAS) IN THE NATIONAL AIRSPACE SYSTEM (NAS)

Pritesh Narayan*, Paul Wu, Duncan Campbell, Rodney Walker

* *p.narayan@qut.edu.au, PhD research student, Australian Research Centre for Aerospace Automation (ARCAA), Queensland University of Technology (QUT), Brisbane, Australia*

Abstract

In recent times, Unmanned Aerial Systems (UAS) have been employed in an increasingly diverse range of applications. Numerous UAS market forecasts portray a burgeoning future, with many applications in both the military and civilian domains. Within the civilian realm, UAS are expected to be useful in performing a wide range of missions such as disaster monitoring (e.g. wildfires, earth-quakes, tsunamis and cyclones), search and support, and atmospheric observation.

However, to realise these civilian applications, seamless operation of UAS within the National Air Space (NAS) will be required. Increasing the levels of onboard autonomy will help to address this requirement. Additionally, increased autonomy also reduces the impact of onboard failures, potentially lower operational costs, and decrease operator workload.

Numerous intelligent control architectures do exist in the literature for mobile robots, space based robots and for UAS. These include: the WITAS project, Open Control Platform, Remote Agent and TRAC/ReACT. However, none of these are specifically targeted at providing the required support for a wide range of civilian UAS missions. Operation of UAS in the NAS for civil applications require robust methods for dealing with emergency scenarios such as performing forced landings and collision avoidance to preserve the safety of people and property.

This paper presents a new multi layered intelligent control architecture. The highest layer provides deliberative reasoning and includes situational awareness and mission planning subsystems. The middle layers deals with navigational aspects (such as path planning and manoeuvre generation). Finally, there is a functional control layer which comprises sensor and actuator subsystems and provides reactive functionality to enable forced landings and collision avoidance. Collision avoidance and forced landing technologies are currently under development at the Australian Research Centre for Aerospace Automation (ARCAA).

Biography

Pritesh Narayan received a Bachelors degree in Aerospace Avionics Engineering with first class honours from QUT in 2005. He is currently completing a PhD in UAV navigation with ARCAA at QUT. The PhD research is focused on reactive UAS flight management in uncertain urban environments.

Paul Wu completed both his Bachelors degree in Electrical and Computer Engineering and his Masters degree in Computer and Communications Engineering from QUT in 2005. He is currently working towards a PhD degree with ARCAA at QUT. His research interests include multi-objective path planning, fuzzy logic and embedded systems.

A/Prof Duncan Campbell is with the School of Engineering Systems at QUT and a member of ARCAA. He has research expertise in real-time computational intelligence, intelligent control, embedded systems and decision support systems. A/Prof Campbell has commercial research linkages in agricultural, food and beverage, and information and communication technologies (ICT) industries. He has a career total of 60 internationally refereed scholarly papers and book chapters, and international patents on an intelligent control application for beverage production.

A/Prof Rod Walker is Director of ARCAA, which is a $12M facility supported by QUT, CSIRO and the Queensland State Government. Rod has a diverse aerospace research background that has involved electromagnetic propagation modelling, GPS multipath modelling, aerospace image processing, aerospace risk assessment and fault tolerant control. He has consulted widely on aerospace automation issues with clients such as: the mining industry, Airservices Australia, Boeing Phantom Works, DSTO and the Federal Government. He is the author of over 70 scientific papers and book chapters and has appeared in over 20 media articles.

Introduction

In recent times, UAS have been employed in an increasingly diverse range of applications. Numerous UAS market forecasts portray a burgeoning future, including predictions of a USD10.6 billion market by 2013 [1]. Within the civilian realm, UAS are expected to be useful in performing a wide range of airborne missions such as disaster monitoring, search and support, and atmospheric observation [2].

However, to realise these civilian applications, seamless operation of UAS within the NAS will be required; this is a difficult problem. Most literature [3, 4] indicate that an equivalent level of safety (ELOS) to that of a human pilot will be one of the requirements for integration of UAS into the NAS. The ELOS requirement, indicates that the system must be capable of replicating some of the capabilities of a human pilot; this leads to the need for a higher degree of onboard autonomy.

A higher degree of onboard autonomy includes the ability to respond automatically to hardware failures and respond to changes in the environment through onboard replanning and execution. These tasks are routinely performed by human pilots; automating these tasks results in a more robust UAS that is not as susceptible to onboard failures. Furthermore, it reduces the human operator's workload and therefore potentially allows a single human to operate multiple unmanned aircraft instead of many human operators controlling one aircraft. As well, it allows the operator to focus on the mission rather than piloting aspects. Such autonomy could potentially lead to a decrease in operational costs.

However, taking the human pilot out of an aircraft removes much sensory and decision making capability. To demonstrate that a UAS still has an equivalent level of safety to a human piloted aircraft, this capability must be automated. For this to occur, UAS will need to possess greater "intelligence" than they do today, aspiring to acquire the traits of the human pilot. The UAS will need to acquire the capacity to monitor the vehicle's internal systems and the outside world, and to detect any changes that affect the mission safety and mission outcome. With this information, the UAS must then make rational decisions and take the necessary actions to preserve safety and achieve mission objectives. This capability can be implemented through the use of an intelligent control architecture.

Defining Intelligent Control

Intelligent control is a multi disciplinary field (***Figure 1***) that involves the use of techniques from the fields of Artificial Intelligence and Control within the context of the Operational Requirements of the task. Intelligent control systems are generally structured in a hierarchical manner. High level (Complex and abstract) tasks are decomposed into a series of time critical low level tasks (data rich and precise). This obeys the so called "principle of increasing precision with decreasing intelligence" [5].

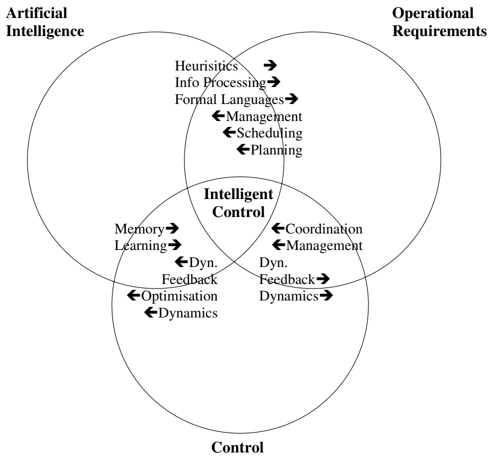

Figure 1 - Definition of Intelligent Control Discipline [6]

Intelligent Control and the Human Pilot

Replicating the capabilities of a human pilot is not a trivial task. For example, during a routine manned flight in civilian airspace, the pilot uses available data (e.g. terrain maps), sensor readings and instructions from air traffic management (ATM) to fly the aircraft safely to its destination. The pilot is capable of dealing with varying situations including and not limited to: turbulence, onboard failures (e.g. actuator, sensor, engine), performing a forced landing and avoiding potential collisions with terrain and other aircraft.

To encapsulate the qualities of a human pilot within UAS, the intelligent control architecture must accurately model a pilot's decision making process. An example of aircraft pilots cognitive process [7] during routine flight is shown in ***Figure 2***. The cognitive model is relatively complex but the reader should note that human pilots have their own sensors (e.g. vision, touch) and actuators (e.g. hands, feet). Pilots use their own perception (e.g. recognition of obstacles) in conjunction with memory (prior experiences) to take appropriate actions in a broad range of scenarios.

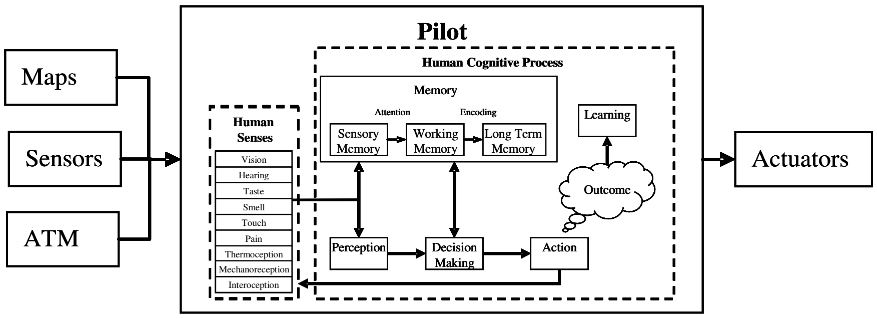

Figure 2 - Cognitive Model of an Aircraft Pilot's Decision Making Process during flight

The purpose of this paper is to combine the principles of the human cognitive process and the field of intelligent control to encapsulate the qualities of a human pilot. In order to achieve this, a review of existing architectures is presented in the next section.

Review of Intelligent Control Architectures

An overview of existing architectures in robotics, spacecraft and UAS is presented in this section to identify relevant architecture design methodologies and the benefits and shortcomings of different architectures.

Robotics Architectures

Traditionally, many architectures in the field of robotics have made use of the state-action model. The state action model is based on the idea that the system can be described as a set of states [8]. The agent (e.g. a ground based robot) transitions from one state to another through actions. This is under the assumption that the environment remains constant unless acted upon by the robot.

Bonasso [9] is the pioneer of a three tiered (3T) intelligent control architecture which has been successfully implemented on a variety of robotics platforms (***Figure 3***). The deliberation layer evaluates goals, resources and timing constraints and outputs a partial list of ordered tasks called a Reactive Action Package (RAP). The sequencing layer decomposes a selected RAP, into a sequence of skill sets (basic agent commands e.g. move left) which are activated and deactivated to accomplish tasks. This architecture does not provide any way of enforcing hard real time limits on these specific skill sets. Furthermore, since all replanning (mission level and reactive) is performed by the deliberation layer, it is difficult to calculate the time required to generate a RAP as deliberative planners are generally symbolic in nature. This may not pose a problem for slow moving robots, but is a critical problem in UAS operations (e.g. reactive sense and avoid).

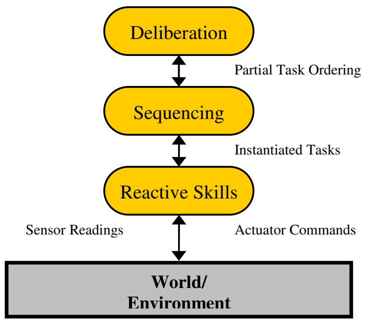

Figure 3 - 3T Intelligent Control Architecture [9]

The ATLANTIS architecture by Gat [8] is very similar to Bonasso's 3T architecture [9]. ATLANTIS also includes planning and reactive skills to allow the robot to operate in dynamic environments. The main difference is that ATLANTIS leaves the overall control of the system to the sequencing layer. Deliberation is treated as an activity that is scheduled by the sequencing layer. In situations where the computational urgency of the reactive component is greater (e.g. obstacle avoidance), the sequencer can temporarily suspend other ongoing deliberative activities. However, ATLANTIS also lacks a method for enforcing real time constraints.

Noreils [10] developed a three layer architecture with the aim of improving overall system reactivity. The highest two layers (planning and control) correspond approximately to the top two layers of Bonasso's 3T architecture [9] (Deliberation and Sequencing layers). However, the Functional (reactive) layer of Noreils' architecture includes sub-modules (***Figure 4***) which can independently trigger the appropriate actions (e.g. obstacle avoidance, target tracking) if the required command is not provided by higher layers in time. As a result, the use of independent sub-modules increases the extensibility of the architecture. Specialised modules are very beneficial for performing specific tasks which must meet real time deadlines. However, using specialised modules also results in added complexity as adding additional functionality requires the addition of new sub-module

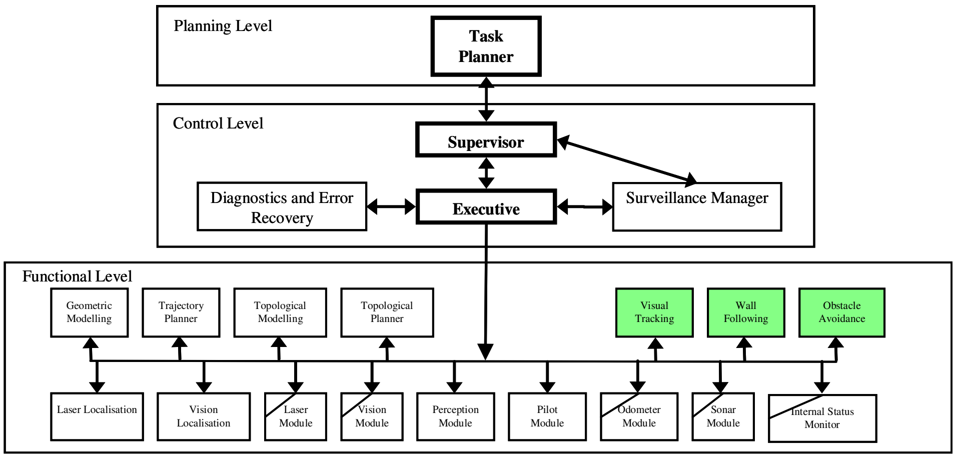

Figure 4 - Noreil's Architecture [10]

Arkin [11] presents an alternate architecture to the 3T architecture [9], entitled AURA. The deliberative layer here consists of a mission planner, navigator and pilot sub-system. The navigator performs mission level path planning; the pilot then constructs a linear path through this free space by calling upon available sensors and motor schemas (motor schemas comprise of low level navigational tasks e.g. trajectory tracking; similar to sub-modules found within Noreils' Functional layer [10]). The use of motor schemas limits the functionality of the agent to navigation only. No mention has been made about the inclusion of onboard payload activity (e.g. onboard camera control) scheduling; an important component for civil UAS operations (e.g. surveillance, disaster monitoring).

Brooks [12] presents another architecture known as Subsumption, which decomposes the control system problem into multiple modules (a module is an independent subsystem which is focused on completing a specific task), also referred to as behaviours. The simplest module is implemented first, and subsequently more complex modules are then implemented above it, providing more functionality (***Figure 5***). As more functionality is added to the robot, the system can quickly become very complicated. Furthermore; this architecture lacks flexibility, where once a complete system is implemented, it becomes very difficult to change the system functionality as each module is very task specific.

In addition, there are no provisions for fault detection and accommodation (FDA) in the robotics architectures reviewed here. This may not an issue for operations in controlled environments, but is a critical issue for UAS operations over populated environments (e.g. urban terrain).

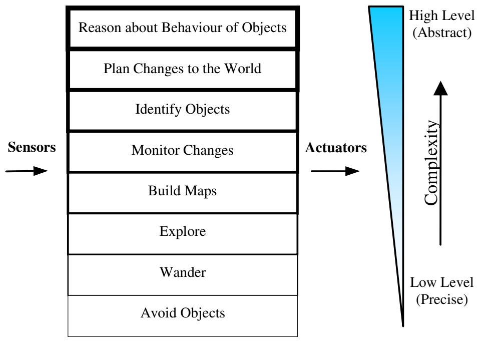

Figure 5 – Brooks' Subsumption Architecture [12]

The architectures reviewed in this section represent the most common architectures used in robotics. A review of the architectures used in space based robotics is presented in the following section.

Space Based Architectures

A wide variety of architectures for onboard intelligence have been developed for space based systems. The operation of spacecraft bears many similarities to that of UAS; both deal with a remote semi-autonomous system that operate in the natural environment and must therefore deal with dynamic changes and uncertainty. The need for robustness in space-based applications is important, due to the significant financial cost of failure. Furthermore, both are constrained by finite resources (such as fuel and battery energy) and must meet stringent real time computational requirements. Both UAS and spacecraft are currently heavily reliant on human operation and receive commands for low level control (such as manoeuvre control) from manned ground stations. Consequently, both fields can benefit greatly from increased onboard autonomy. A brief overview of several key projects in space-based automation is presented here.

NASA's Autonomous Science-craft Experiment (ASE) [13] demonstrated automated scheduling and planning routines on the EO-1 Satellite launched in late 2000. This was the first time a space based mission was conducted autonomously using an intelligent control architecture (***Figure 6***) implemented onboard the spacecraft.

The cornerstone of the ASE architecture is the *Continuous Activity Scheduling Planning Execution and Replanning* (CASPER) [13] module which employs a repair based technique to: create a plan (which resolves conflicts that violate spacecraft constraints); propose a set of resolutions for a chosen conflict using a genetic algorithm; and choose the desired solution using heuristics. This process occurs iteratively until no more conflicts remain. The Spacecraft Command Language (SCL) uses rule based checking to convert this high level plan into low level commands. Therefore, even though the general concept is useful, the architecture itself is not focused on path planning and is instead concerned with scheduling of activities; which are critical aspects of UAS operations.

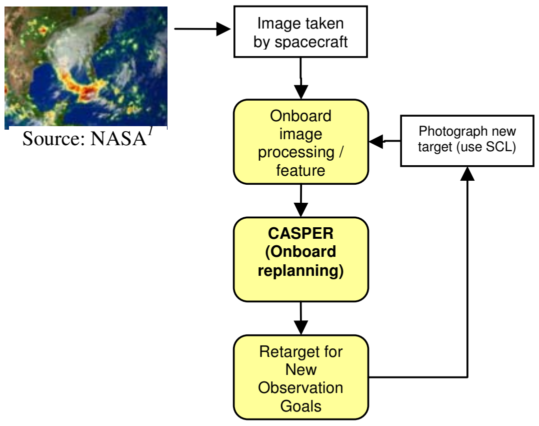
Source: NASA[1]
Image taken by spacecraft
Onboard image processing / feature
Photograph new target (use SCL)
CASPER (Onboard replanning)
Retarget for New Observation Goals

Figure 6 - NASA ASE architecture [13]

Another NASA based architecture is the Automated Planning/Scheduling Environment (ASPEN) [14] system which is essentially a software based application framework. It is an object oriented framework based on C++ that can intelligently schedule activities onboard the spacecraft. Activities are represented using state action models with the actual scheduling decisions performed using a parameter dependency network. This is similar in concept to a temporal constraint network [15] but extends its capabilities to include physical resources. A temporal constraint network is a graph based method for scheduling where nodes represent instances in time and edges represent time delays. ASPEN has been tested in numerous scenarios including onboard the New Millennium Earth Observing One (NM EO-1) satellite and Navy UHF Follow On One (UFO-1) satellite. Again, this architecture is targeted at the scheduling of activities rather than the path planning and execution problems, so important for UAS.

NASA's Remote Agent [16] is another intelligent control architecture designed for use onboard satellites. It employs a three tiered hierarchy similar to that presented by Bonasso [9]. The remote agent contains a set of decision making and scheduling tools to synthesise responses to unexpected situations which may arise during the mission (***Figure 7***). The Mission manager determines achievable goals both in the long term and current short term. The Planner/Scheduler takes these goals and uses a heuristic guided backtracking search to create an execution schedule of activities. Like ASPEN, plans are generated using temporal constraint network related methods. The Smart Executive plays a similar role to the sequencing layer in Bonasso's model and also makes use of Reactive Action Packages to implement these activities. Additionally, there is a Mode Identification and Reconfiguration subsystem that provides dynamic information on the status of the spacecraft. This provides an added layer of robustness as the execution of plans is dynamically modified by the perceived health of the spacecraft. Unlike the ASE and ASPEN scheduling systems, this architecture is more comprehensive as it includes methods for handling changes in the spacecraft's internal state as well as the external state. However, again, it is more focused on activity scheduling rather than path planning.

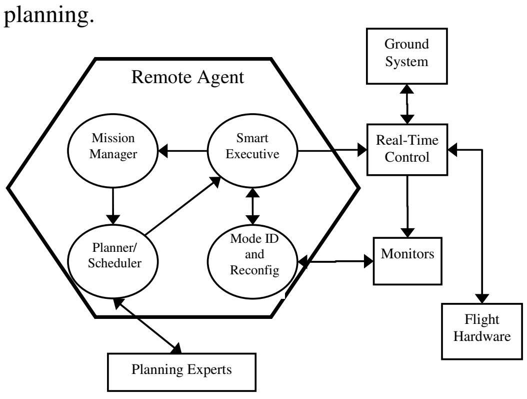

Figure 7 - NASA's Remote Agent Architecture [16]

The architectures reviewed in this section represent the most common architectures used in spaced based operations. A review of the architectures used within UAS is presented in the following section.

[1] *http://www.nasa.gov/mission_pages/hurricanes/multimedia/index.html*

UAS Architectures

Intelligent control architectures implemented onboard UAS are generally extensions of architectures found in robotics. However, many robotics architectures cannot be directly implemented in UAS. Firstly, UAS operate in highly dynamic environments where atmospheric changes can occur almost instantaneously; therefore, the agent's response must meet real time constraints. This is further compounded by the fact that aircraft typically travel at much greater velocities than ground based robots. Secondly, UAS dynamics can be highly non-linear and thus require careful consideration in the controller design and how this will interface to other subsystems onboard the aircraft. Finally, failures onboard UAS can be catastrophic and result: in loss of the UAS; property damage; and in the worst case, loss of human life as these robots are exposed to the general public.

A UAS can be thought of as a special type of robot that takes directives asking it to move from one location to another within a certain timeframe. Generally, there are two types of UAS: rotary/helicopter UAS that have the ability to brake and hover, and fixed wing UAS. Rotary UAS generally have shorter flight times while fixed wing UAS often have greater endurance but must always maintain some minimum (greater than stall) velocity.

Various architectures have been proposed that are specifically targeted at UAS. Schaefer [17] for example, presents a multi-layered UAS decision making architecture known as "Technologies for Reliable Autonomous Control (TRAC)". This is a variation of the 3T architecture pioneered by Bonasso [9] that has been augmented with another layer known as the Meta-Executive layer. The meta-executive layer is used to coordinate and synchronise interactions between the deliberative (which is goal driven e.g. performing a set of tasks based on accomplishing a particular goal) and execution (which is event driven e.g. performing a set of tasks based on a schedule) layers.

The TRAC architecture (***Figure 8***) revolves around a central data communications and storage module named the Active State Cache. The topmost, deliberative layer is called Closed Loop Execution and Recovery (CLEaR). This is responsible for high level mission management and task sequencing. Plans created by CLEaR are acted upon by the Autonomous Command Executive (ACE) which oversees the execution of mission plan elements. Beacon-based Exception Analysis (BEAM) and Spacecraft Health Inference Engine (SHINE) are subsystems that monitor the health of the unmanned vehicle in real time. The TRAC architecture is an extension of that developed in the NASA Remote Agent project. Significant emphasis has been placed on the importance of low level fault detection and Identification (FDI) through the inclusion SHINE and BEAM subsystems. The ACE subsystem can deal with some reactive situations, but this is limited to terrain avoidance and stability corrections during wind gusts [18]. There is no specific subsystem onboard to deal with reactive collision avoidance, a necessity for flight operations within the NAS.

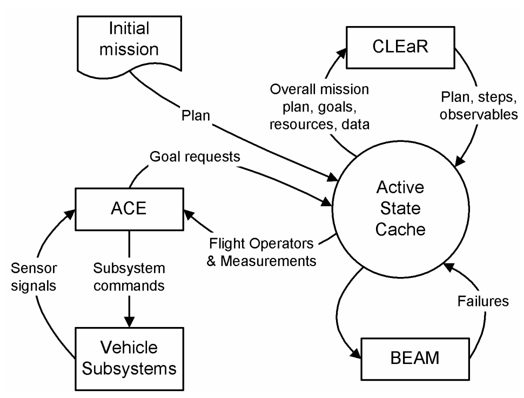

Figure 8 - TRAC System Structure [19]

The NASA APEX software robotics Architecture [19] is also based on Bonasso's 3T architecture [9]. This architecture has been successfully applied to a range of applications, notably that of NASA's Autonomous Rotorcraft Project. The upper two layers of APEX are collectively referred to as Reasoning and Control Services (RCS). This architecture emphasises reusability through modularity and thus separates RCS procedures (which are the most reusable) from lower layer modules which are less reusable.

Boskovic [20] presents a UAS architecture which is optimized for navigation, in similar fashion to the upper layers of AURA [11]. The layers within this architecture are defined with respect to specific UAS functionalities rather than generic functions in robotics. The highest layer (***Figure 9***) in this hierarchal four layered model is the Decision Making layer. This layer uses *a priori* information in conjunction with information obtained from sensors to make appropriate decisions to achieve mission goals. The next level is the Path Planning Layer which generates mission waypoints. If an obstacle is detected that was not known *a priori*, then the waypoints are recomputed online. There is no communications subsystem to give the operator any decision making capability throughout the mission. Again, there is no specific subsystem to deal with reactive collision avoidance.

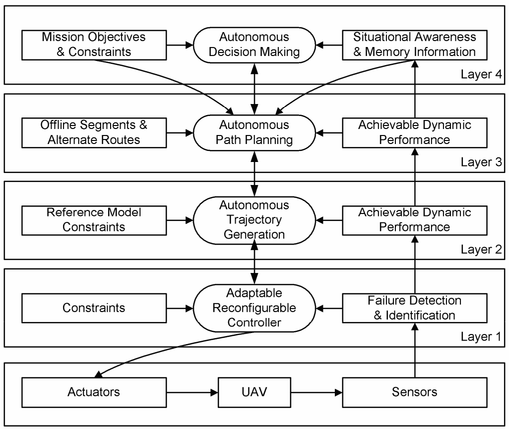
Mission Objectives & Constraints
Autonomous Decision Making
Situational Awareness & Memory Information
Layer 4
Offline Segments & Alternate Routes
Autonomous Path Planning
Achievable Dynamic Performance
Layer 3
Reference Model Constraints
Autonomous Trajectory Generation
Achievable Dynamic Performance
Layer 2
Constraints
Adaptable Reconfigurable Controller
Failure Detection & Identification
Layer 1
Actuators
UAV
Sensors

Figure 9 - Boskovic's UAS Decision Making Architecture [20]

The WITAS UAS project [21] presents yet another intelligent control architecture which has deliberative, reactive (like sequencing) and control layer components. It is best to view this as a reactive concentric architecture as the individual processes at the various levels of abstraction are executed concurrently at different latencies (high level path planner runs at higher latency than low level controllers). The deliberative layer here contains a collection of path and trajectory planners, predictors and recognition packagers. A set of flight control modes such as hovering, dynamic path following and take-off and landing, are automated using sets of Task Procedures (similar to a RAP in 3T). However, in order to switch between autonomous flight modes, it is necessary for the UAS to brake and hover before executing the next Task Procedure. Doing so decreases the operational efficiency the UAS; furthermore, this strategy is infeasible for fixed wing UAS. Finally, there is no explicit provision for handling collision avoidance.

The open control platform (OCP) [22] is similar to APEX in that it too is a software robotics architecture based on Bonasso's 3T architecture [9], which can potentially be applied to UAS operations. The top layer here comprises of supervisory tasks such as: data management; event detection and situation awareness. The middle layer (reconfigurable control) performs mode transitioning stability control (e.g. use of an adaptive control algorithm during transition between approach and landing phases) whilst the lower level is dedicated to trajectory tracking (low level controller implantation). All internal communications makes use of middleware (CORBA) with custom extensions implemented to ensure hard real time execution of commands. This also allows the architecture to operate as a distributed network (similar to the WITAS Project) and different components can be written in different languages. Similar to other UAS architectures, there are no specific subsystems to deal with external communications.

Summary of Findings

From the literature review, it was found that the vast majority of architectures were hierarchical. This approach was often used to separate slower, deliberative planning processes from faster, time-critical hardware control systems [9]. Additionally, it allows for abstraction of complexity from one layer to the next; this is useful not only in reducing subsystem complexity, but also helps in software reusability [19]. The vast majority of architectures employed some variation of Bonasso's 3T hierarchy [9] which had separate layers for deliberation, sequencing of actions and control execution.

Ideally, a human operator should only need to interact with the high level deliberative layer. In this scenario, the operator performs high level tasks such as specifying mission goals and the schedule associated with these goals. In these instances, there is a need for a communications subsystem that provides the link between the remote agent and the ground station. Such a communications module is incorporated into the ASE, APEX and Remote agent architectures [13, 16, 19].

It was found that in many UAS and spacecraft based architectures that an important capability was a method for monitoring the agent's internal state (i.e. the health of the vehicle) and its impact on vehicle performance. This was implemented as a form of Fault Detection and Accommodation (FDA) in TRAC, Remote Agent and OCP and in Boskovic's architecture [16, 17, 20, 22].

At the same time, it is important to have accurate knowledge of the external environment in which the agent is situated. It was found that even though all architectures made provisions for a sensing mechanism, very few explicitly explored the computational complexities involved in processing sensor information for use in higher level planning algorithms. Obviously, very little processing is required for low level control systems as raw data, such as position and velocity can feed directly into an actuator control module. However, avoiding dynamic obstacles when generating manoeuvres requires predictions of the current and future state of the dynamic obstacle. Therefore, analysis of sensor data is required to transform it into a form usable by the higher level algorithms through sensor fusion. There is currently no UAS architecture which explicitly separates the sensor data requirements between lower (raw sensor data) and higher layers (accurate state information calculated through sensor fusion methods).

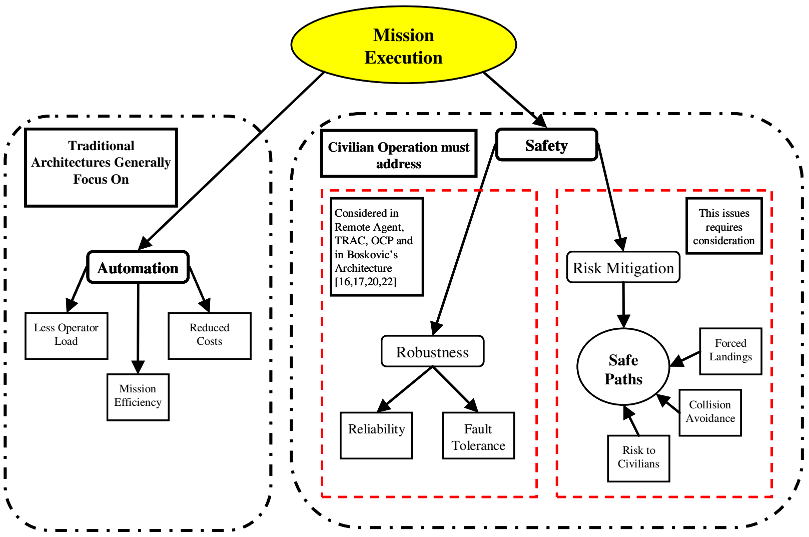

Figure 10 - Summary of the functional requirements for an architecture during civilian UAS operations

As the majority of UAS operations require positioning the aircraft in the right place at the right time, it can be seen that there is a need for a robust architecture that provides this path planning and execution functionality. It was found that the majority of architectures focus on only the mission execution component of UAS operations and do not explicitly provide for a method of ensuring the safety of the aircraft and the minimisation of risk to other aircraft and people on the ground. There has been some work in the areas of fault tolerance and reliability, but risk mitigation (actions to reduce the impact of a risk) has not fully been addressed (***Figure 10***).

Furthermore, the vast majority of architectures do not provide a complete end to end architecture from goal deliberation to planning to action execution (with the exception of Boskovic [20]). However, Boskovic's architecture does not address the problem of managing risks in during UAS operations. There is no communications subsystem to give the operator any decision making capability throughout the mission. Also, there is no specific subsystem to deal with collision avoidance. A proposed architecture is presented in the following section addressing these critical issues.

Proposed Architecture

We propose an architecture for civil UAS operations based on Boskovic's [20] architecture. This architecture not only accommodates path planning and FDA, but also includes provisions for intelligent execution of activities not explicitly involved in path planning. As well, it also encompasses modules dedicated to ensuring the safety of the aircraft. At this point in time, all aspects of high level decision making however, are left to the responsibility of the Human Operator (e.g. choosing which goals to pursue). In terms of efficiency, this architecture provides the potential to reverse the current Civilian UAS trend from many operators monitoring a single UAS, to a single operator monitoring multiple UAS.

To allow the human operator interaction with the onboard high level deliberative layer, a communications subsystem has been included to allow real time interaction between a human operator and UAS activities (e.g. uploading new mission goals) during the flight operation.

In the previous section, it was concluded that, none of the current UAS architectures explicitly separate the sensor data requirements between lower and higher layers. In the proposed architecture, raw sensor data is forwarded to lower layers (which have real time requirements) in an effort to minimise any lag which may occur with processed data. Raw sensor data is also forwarded to a sensor fusion subsystem which generates an accurate approximation of the aircraft state. This data is stored within the Integrated Shared State Memory (ISSM – similar to Schaefer's Active State Cache [17]) which can be accessed by the relevant layer.

To incorporate risk mitigation strategies within the proposed architecture, specific modules (similar to Noreil's Architecture [10]) to deal with sense and avoid and forced landing situations were used. The sense and avoid module performs detection of obstacles within the immediate vicinity of the aircraft. This data is used by the manoeuvre generation layer to perform the appropriate collision avoidance manoeuvres. Likewise, the forced landing site classifier module is used to detect potential landing sites during critical onboard failures.

Currently, Boskovic's architecture does not include any functionality for scheduling and control of payloads (e.g. camera, lights) as it is focussed purely on the navigational aspects of UAS operations. The proposed architecture includes an activity scheduler and controller. The activity scheduler creates a schedule of payload activities by synchronising start and finish times using mission time and aircraft state. The activity controller activates and deactivates the relevant payload. This feature allows the UAV to perform a range of missions including and not limited to surveillance, disaster monitoring and search and support. A detailed representation of the Proposed UAS Architecture is presented in ***Figure 11***

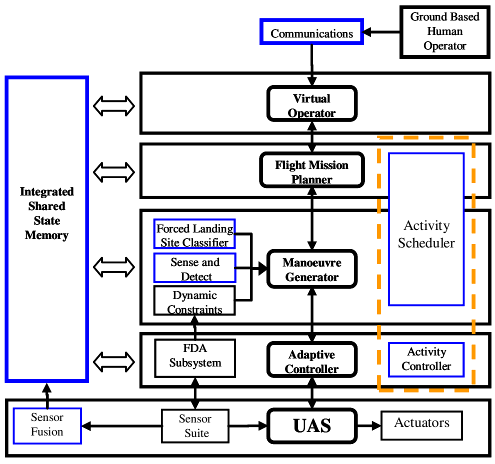

Figure 11 - Proposed Civilian UAS Architecture

Virtual Operator

The Virtual Operator (VO) is concerned with providing mission goals to the lower layers and simultaneously monitoring the state of the UAS in assessing whether these goals have been achieved or not. When communicating with the mission flight planner, the VO provides a set of prioritised goals which may be defined in terms of spatial position, velocity, aircraft orientation and time. The mission flight planner calculates a path based on these goals and returns the costs (in terms of fuel and time for example). With this information, the VO can then make a decision as to whether to continue the mission using the current plan or to reject this plan and create a new one by changing one or more mission goals. A similar process is necessary for interaction with the Activity Scheduler. When communicating with the Activity Scheduler, the goals the VO provides are instead activities that need to be performed when the UAS reaches certain states. For example, an activity could be to turn on the camera and begin capturing images when the aircraft is at a specified location.
In the case that the VO is the human operator, the operator has the full authority (and accountability) to choose the order of operations. Once the operator has decided on an operations schedule, the schedule can be uploaded to the unmanned aircraft via the communications channel and stored in onboard memory. The VO has the authority to update the operations schedule and the maps used for planning (e.g. terrain maps) throughout the mission. In the absence of certifiable decision making techniques that can meet the requirements of such a VO, it is envisaged that a semi-autonomous VO module, coupled to a Human Machine Interface (HMI) and human operator would constitute the VO shown in this architecture.

Mission Flight Planner

The Mission Flight Planner is in essence a multi-objective path planner that evaluates multiple criteria in determining an optimal path for the aircraft. It receives goals, which may comprise multiple prioritised waypoints from the VO. As well, it obtains information about the environment through multi-resolution maps stored in the memory module. These maps could include terrain data, risk data (risk associated with overflying certain areas) and predictions of dynamic obstacles (such as other aircraft).

Additionally, the planner also obtains from this dynamic memory the current aircraft position, fuel load and other related constraints (such as the need to maintain within line of sight of the operator). The VO is informed of the costs involved in reaching each waypoint and of any unreachable waypoints. At the same time, the mission flight planner passes to the Manoeuvre Generation subsystem a path which effectively consists of a series of intermediate goals (or waypoints). When there are changes to the environment (which is reflected in the data obtained from dynamic memory), or significant deviation of the aircraft from the planned route as reported by the Manoeuvre Generation subsystem, the mission flight planner replans a new flight path. If any of the intermediate goals are unachievable, the Manoeuvre Generation subsystem modifies the maps in memory and marks these unachievable regions as no-go.

Manoeuvre Generator

The Manoeuvre Generation Layer generates a feasible local path between a set of intermediate waypoints given by the Flight Mission Planner. A feasible path is one that is collision free and which satisfies aircraft dynamic and kinematic constraints.

Basic Manoeuvres (e.g. flying straight and level, pitching, rolling and yawing motion) can be combined together to create more complex manoeuvres. Representing aircraft motion as a set of manoeuvres is essential for flight in civilian airspace. For example, consider a scenario whereby the UAS is instructed to reach a higher altitude to avoid other aircraft, but is not physically able to do so due to the constraints of restricted airspace and the limitations of the aircraft's manoeuvrability (insufficient rate of climb). A candidate solution in this situation is to perform a spiral manoeuvre.

A sense and detect capability within UAS is essential for flight in segregated airspace. The sense and detect subsystem uses onboard sensors (e.g. vision) for detection of obstacles (static and moving). The manoeuvre generation algorithm then uses the data provided by the sense and detect unit to generate collision free path segments, and to perform emergency collision avoidance manoeuvres if necessary. Carnie [23] is currently investigating the use of machine vision to provide sense and detect capabilities onboard UAS at the Australian Research Centre for Aerospace Automation (ARCAA).
The forced landing classifier is used to detect potential landing sites during flight, in case the UAS needs to perform an emergency landing due to onboard failures which cannot be accommodated (e.g. engine failure). This information is input to the Manoeuvre Generation algorithm, which provides the adaptive controller with a suitable landing trajectory for tracking. Fitzgerald [24] has conducted research into detection and classification of potential forced landing sites at ARCAA.

Adaptive Controller

The low level controllers are designed to ensure aircraft stability at all times. A broad range of techniques are available to create the desired response including: Proportional, Integral and Differential (PID); State Space; Fuzzy; Optimal; to mention a few [6].

Small scaled UAS generally, do not have the available onboard payload capacity to include sensor redundancy. If sensor failure occurs without detection, this can lead to critical failure as the stability controllers will receive incorrect or no state information. Critical Failure can also occur if an actuator becomes inoperable.

Fault Detection and Accommodation algorithms (FDA) are used to detect if a particular sensor is providing erroneous data, and allowing continuation of operations by disable the erroneous sensors operation. This however leads to reduction in aircraft performance, as fewer sensors are now available to provide an estimation of the UAS state. This information is conveyed to the Dynamic Constraints Subsystem for recalculation of new UAS dynamic and kinematic constraints. Cork [23] is currently investigating FDA techniques to reduce the effects of erroneous sensors at ARCAA.

Concluding Remarks

It is apparent that the operation of UAS in civilian applications requires an equivalent level of safety to that of manned aircraft. Achieving this level of safety requires, in addition to system robustness, an intelligent system that is capable of both tactical and strategic planning to minimise the risk involved when undertaking a mission. At the same time, the system must also be able to execute emergency procedures in the event of hardware failures.

Through an investigation of existing architectures in unmanned aircraft, space based systems and robotics, it was found that few offered a framework that catered for the path planning and manoeuvre generation aspects of onboard intelligence in light of the needs of sensor integration. Many have considered mission scheduling and fault detection and accommodation, but few have integrated this with the aforementioned path planning and execution elements with a focus on emergency scenarios; including and not limited to collision avoidance and forced landings. Furthermore, even fewer have considered the multiple criteria, in terms of airspace regulations, mission objectives and mission safety that must be considered in civil UAS operations.

To address these deficiencies, an intelligent control architecture for UAS was devised that addresses the requirements of intelligent planning, execution and handling of emergency scenarios. This architecture encompasses many subsystems that are currently being developed at ARCAA. It is envisaged that the integration of the various components in this architecture would help increase the level of intelligence onboard unmanned aircraft in terms of mission efficiency and increased safety. This is not only paramount to the acceptance of UAS in the NAS, but will also allow for decreased operator workload and thus reduce operational cost.

Acknowledgements

The authors wish to acknowledge the support of the Queensland University of Technology (QUT), the Australian Research Centre for Aerospace Automation (ARCAA) and the Commonwealth Scientific and Industrial Research Organisation (CSIRO).

Bibliography

1. Cox, T., Civil UAV Capability Assessment (Draft Version). 2004, NASA. p. 103.

2. Wegener, S., UAV Autonomous Operations for airborne Science Missions. 2004, American Institute of Aeronautics and Astronautics.

3. Australia, C.A.A.o., Advisory Circular - UAV Operations, Design Specification, Maintenance and Training of Human Resources. 2002.

4. EUROCONTROL, Specifications for the use Of Military Unmanned Aerial Vehicles as Operational Air Traffic Outside Segregated Airspace. 2006.

5. Tzafestas, S., ed. Methods and Applications of Intelligent Control. 1997, Kluwer Academic: Boston.

6. Gupta, M.S., N, ed. Intelligent Control Systems - Theory and Applications. 1996, IEEE Press: New York. 820.

7. Roberts, P., et al., En-Route Human Pilot Automation Considerations for future UAV Systems. 2005, Queensland University of Technology: Brisbane.

8. Gat, E. Integrating planning and reacting in a heterogenous asynchronous architecture for controlling real-world mobile robots. in National Conference for Artificial Intelligence (AAAI). 1992.

9. Bonasso, R.P., et al. Experiences with an Architecture for Intelligent, Reactive Agents. in International Joint Conference on Artificial Intelligence. 1995.

10. Noreils, F.R. and R.G. Chatila, Plan execution monitoring and control architecture for mobile robots. Robotics and Automation, IEEE Transactions on, 1995. 11(2): p. 255-266.

11. Arkin, R. Motor schema based navigation for a mobile robot: An approach to programming by behavior. in IEEE International Conference on Robotics and Automation. Proceedings. 1987.

12. Brooks, R., A robust layered control system for a mobile robot. Robotics and Automation, IEEE Journal of [legacy, pre - 1988], 1986. 2(1): p. 14-23.

13. Rabideau, G. Mission Operations with Autonomy: A Preliminary Report for Earth Observing-1. in 4th International Workshop on Planning and Scheduling for Space. 2004. Darmstadt, Germany.

14. Fukunaga, A., et al. Towards an application framework for automated planning and scheduling. in IEEE Aerospace Conference. 1997.

15. Schwalb, E. and R. Dechter, Processing disjunctions in temporal constraint networks. Artificial Intelligence, 1997. 93(1-2): p. 29-61.

16. Muscettola, N., et al., Remote Agent: to boldly go where no AI system has gone before. Artificial Intelligence, 1998. 103(1-2): p. 5-47.

17. Schaefer, P., et al. Technologies for reliable autonomous control (TRAC) of UAVs. in Digital Avionics Systems Conferences. 2000.

18. Johnson, T.L., et al. The TRAC mission manager autonomous control executive. in IEEE Aerospace Conference. 2001.

19. Freed, M., et al., An Architecture for Intelligent Management of Aerial Observation Missions. AIAA, 2005. 2005-6938.

20. Boskovic, J.D., R. Prasanth, and R.K. Mehra. A multilayer control architecture for unmanned aerial vehicles. in American Control Conference. 2002.

21. Doherty, P., et al. A Distributed Architecture for Autonomous Unmanned Aerial Vehicle Experimentation. in 7th International Symposium on Distributed Autonomous Systems. 2004. Toulouse, France.

22. Wills, L., et al., An open platform for reconfigurable control. Control Systems Magazine, IEEE, 2001. 21(3): p. 49-64.

23. Cork, L., R. Walker, and S. Dunn. Fault Detection, Identification and Accommodation Techniques for Unmanned Airborne Vehicles. in Australian International Aerospace Congress. 2005. Melbourne.

24. Fitzgerald, D., R. Walker, and D. Campbell. A Vision Based Forced Landing Site Selection System for an Autonomous UAV. in Intelligent Sensors, Sensor Networks and Information Processing Conference. 2005.