DEVELOPMENT AND TRIALS OF AN AUTONOMOUS FLIGHT CONTROL SYSTEM FOR UAV'S

Francis Valentinis Cees Bil Paul Riseborough

The Sir Lawrence Wackett Centre for Aerospace Design Technology
Royal Melbourne Institute of Technology
GPO Box 2476V, Melbourne, Vic 3001, AUSTRALIA

Abstract

A fault tolerant autonomous vehicle control system, which exploits the natural redundancy of a parallel processor array, has been developed as part of the Sir Lawrence Wackett Centre's Unmanned Air Vehicle (UAV) programme. The system, known as the Parallel Autonomous Control Engine for Robotics (PACER), is a direct descendant of the Jabiru AFCS, which was a proof of concept, low cost UAV controller also developed at the centre. It is programmable via a high level language based on fuzzy logic, but in which fuzzy rules are grouped into clusters called 'concepts'. The process of clustering rules is extremely advantageous, as it makes a program more readable and increases ease of maintenance. Concepts are distributed redundantly across the processor array, facilitating fault tolerant operation. The system is currently undergoing extensive bench testing and will be ready for flight trials in 1997.

1. Introduction

The ability to truly adapt to an environment, making decisions in constrained circumstances and with poor data is highly desirable for any autonomous system. This basic requirement has not however been comprehensively fulfilled in any existing intelligent controller.

A designer can never fully foresee what challenges an autonomous vehicle will confront during long unmanned missions. Environmental changes and failure of onboard systems will have significant impact on mission effectiveness. For this reason there needs to be a way for a controller to perform actions which are not defined by its initial programming. This adaptation may be restricted to simple tuning of an initial rule base provided by its developer, but it could go further to include discoveries, or in the context of emergencies, decisions based on inadequate, or missing information.

All of these forms of adaptation are desirable, however if employed without caution, they can be quite perilous. For example, in a flight control system, if adaptation of the rule base leads to an unstable flight control law, or one which causes the aircraft to stall, the results can be extremely dangerous.

Fault tolerant autonomous control systems have been available for many years now, and if conventional design techniques and classical control are used, reliability can be ensured. This cannot, however be said of the new breed of controllers, which deviate from traditional control approaches in favour of the use of so-called 'smart control' philosophies, and extensive use of techniques borrowed from the field of artificial intelligence.

Creating a framework in which such a system can be deployed reliably has been the focus of the research described herein. The Parallel Autonomous Control Engine for Robotics (PACER) was born out of a desire to add a flexible, but reliable learning capability to the Jabiru Autonomous Flight Control System (AFCS)[1]. This was a low cost, proof of concept Unmanned Air Vehicle (UAV) control system, which accepted programs written in a high level language based on fuzzy logic.

The result is a system which is quite different to its predecessor, and as a result it is also less cost effective. Regardless of this fact, a major aim of the research is to construct a scalable system which can be deployed in a range of configurations from small low cost systems, through to larger, more powerful smart controllers, albeit with an additional cost penalty.

1.1 The Multi Purpose Autonomous Flight Vehicle System

The PACER, like its predecessor, is currently being evaluated on a UAV testbed. The Global Positioning System (GPS) [17] has created new opportunities for the development and application of UAVs. It is now technically and economically feasible to replace manned aircraft and systems currently used in the roles of coastal surveillance, oil discharge policing, aerial geological surveying, weather soundings and severe weather reconnaissance with autonomous flight vehicles equipped with the appropriate sensors.

The Sir Lawrence Wackett Centre is conducting research to investigate the feasibility of a Multi Purpose Autonomous Flight Vehicle (MAFV) system concept. The objective of this project is to develop a UAV with an on-board programmable, adaptive control system that can maximise performance. The vehicle configuration is of modular architecture, ie. fuel, powerplant, payload and navigation/guidance modules will be interchangeable to enable the MAFV to be optimised for any desired mission

An area of application for the MAFV currently under development with the Division of Atmospheric Research of the Commonwealth Scientific and Industrial Research Organisation (CSIRO) in Australia is atmospheric research. The aim is to investigate the extent of down wind spread of urban pollution, in particular carbon dioxide, methane and ozone trace gasses. This application will form an optimal testbed for the PACER, as it involves a need for the vehicle to explore the spread of pollution, a task involving considerable uncertainty.

Whilst the focus of research to date has been developing a system for control of the MAFV, the results of the work are not UAV specific. The techniques developed as part of this research programme are applicable to a broad array of intelligent control applications where reliability is of importance, from power station control, and patient monitoring through to deep space probe systems.

2. The Jabiru AFCS.

The Jabiru AFCS, which was developed to control the MAFV Jabiru (see figure 1), provided complete control for every aspect of vehicle operation. The AFCS hardware was based on a Motorola 68000 series microcontroller which operated at 16 Mhz. The processor was connected to 64Kbytes of RAM and ROM, a tiny configuration by modern PC standards. The low speed and stability of the Jabiru aircraft facilitated the use of low performance hardware, which contributed in keeping cost extremely low.

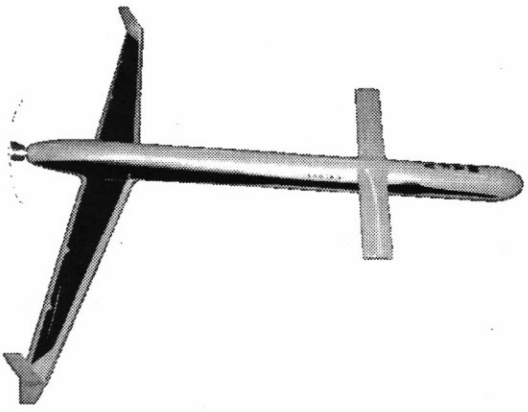

Figure 1: The MAFV Jabiru

The system was fully programmable, and allowed the user to write programs in a high level programming language, which basically provided a vehicle for the specification of fuzzy logic rules[3],[5],[6],[7],[8]. The programming language was used to define the structure of the controller at all levels of the system, from low level control of flight surfaces through to mission control.

Programs written in the programming language, known as the Jabiru Flight Control Language (JFCL), were composed on a PC using conventional text editing tools, then were checked using a PC based checking utility, and finally uploaded to the controller to take its place as a particular layer function (see figure 2). At the top level of the system was the navigator. The navigation system produced information such as required speed, heading, and altitude correction as well as the distance to the next way point. All of this information was transferred to the first fuzzy system layer.

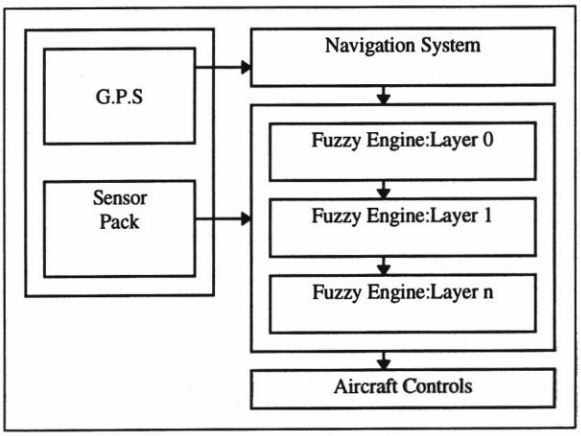

Figure 2: Jabiru AFCS software architecture.

The fuzzy engine executed each layer in sequence, and information was passed down the hierarchy. In its most primitive configuration, a two layer system was worked with. The top layer provided both high level decision making and aircraft guidance. The second layer provided more low level closed loop control of the various flight surfaces.

2.1 Observed deficiencies of the Jabiru AFCS

The design process of the PACER began with a critical review of the Jabiru AFCS, and similar autonomous system controllers. The following observations were made during these studies:

The Flight Control Language

From a software engineering perspective, a long list of fuzzy rules is not much better than "spaghetti" code in a conventional program. For this reason the concept of layering was introduced into the JFCL. Higher level decision making processes were described by their own mini controller, which performed evaluations and then commanded activity from services in the next layer in the hierarchy.

The concept of layering proved extremely powerful in developing the fuzzy logic ruleset for the Jabiru's flight control, however, once each layer reached a critical stage, it became unmanageable, as the rule base in each layer grew to an unmanageable size.

During development, a partial solution to this problem was to split a layer in two, however this was often impossible, difficult, or at least extremely cumbersome. What was needed was a means to go beyond layering, and into a scheme where related rules could be grouped and then used as individual modules within layers. This would also provide a better foundation for reuse, and make the process of developing the rule base more akin to conventional programming.

Lack of adaptation

Although adaptation may not be required in short simple missions; longer and more complex missions involve considerable uncertainty and therefore require some form of learning capability.

Lack of Fault Tolerance

The Jabiru AFCS was not fault tolerant. There was no allowance for redundancy in the system at all. This kept cost down considerably, but also limited the potential use of the system in areas where fault tolerance would be of prime importance.

3. The PACER

The PACER is a multi processor computer designed specifically for realtime control of autonomous vehicles. Like the Jabiru AFCS, it is a fully programmable system, however the technique used to program it is quite different to that used with the Jabiru AFCS.

3.1 Programming model

With the Jabiru AFCS, programs written by the user were loaded into the computer where they were interpreted and decoded. This worked well, however interpretation of the programs on board the flight computer proved to introduce a large amount of overhead. This is not an issue when programs are loaded in on the ground prior to takeoff, but becomes a major concern if programs are to be updated during flight.

To get around this difficulty it was decided to perform compilation on the ground, on a personal computer. The new compiler translates the program written by the user into a numeric stream which represents sequences of commands which drive a state machine on board the PACER. This state machine constructs the appropriate rule network in the memory of the system. By using this technique the performance of the system during reprogramming was improved by approximately ten times.

This process also dramatically reduced the amount of memory required on board to store the programs. Raw source code takes up a considerable amount of memory, so converting programs into a low level format proved to be extremely advantageous. Early observations indicate that space savings in excess of ten times can be achieved with code containing few comments.

One of the biggest innovations of the Jabiru AFCS was the JFCL, a flexible, easy to use programming language for defining the UAVs control. While the language, and rule based systems generally, were very easy to use for systems with low rule counts per layer, development and maintenance of larger systems was difficult.

In order to ease the process of creating and maintaining systems with larger rule counts per layer, a new variation of the original FCL was created, called the Concept Oriented Robotics Engineering Language (COREL). COREL is a major extension of the original Jabiru FCL, however while much of its syntactic structure is identical to that of the original FCL, it is not fully backward compatible.

COREL minimises program complexity by grouping fuzzy logic rules into clusters, called 'concepts'. On a logical level, concepts are hierarchical, and can contain other concepts, allowing for a divide and conquer approach in design, an absolutely essential tool in the development of large software systems. Concepts can also be stored in libraries, as can functions or objects in conventional programming, permitting reuse. They are permitted to inherit their basic structure from predefined concepts, as in object oriented programming. Libraries could be created covering particular application domains, and allowing an end user to simply link pre-specified building blocks together to construct the UAV internal fuzzy rule base.

Concepts should not be confused with behaviours [14],[15],[16], which have been used as fundamental building blocks of autonomous robotics languages in the past. Concepts are lower level entities, simply representing fuzzy logic rule clusters. Behaviour based systems can be

developed using the COREL language, using concepts to construct behaviours.
Concepts are not just high level entities which exist at the programming language level. Their role in clustering rules together is a very important one. When a COREL program is loaded into the PACER, it must be split into blocks and distributed across the parallel processor array. The subdivision of a program into concepts is perfect for this purpose. The system simply distributes the different concepts across the various processors.

Although a simple concept distribution strategy exploits the performance advantages of moving to a parallel processor array, it does not give the user a fault tolerant system. Rather than distributing individual concepts around the array therefore, redundant instances of each concept are created and distributed, along with a single 'checker' process. The organisation is depicted in figure 3.

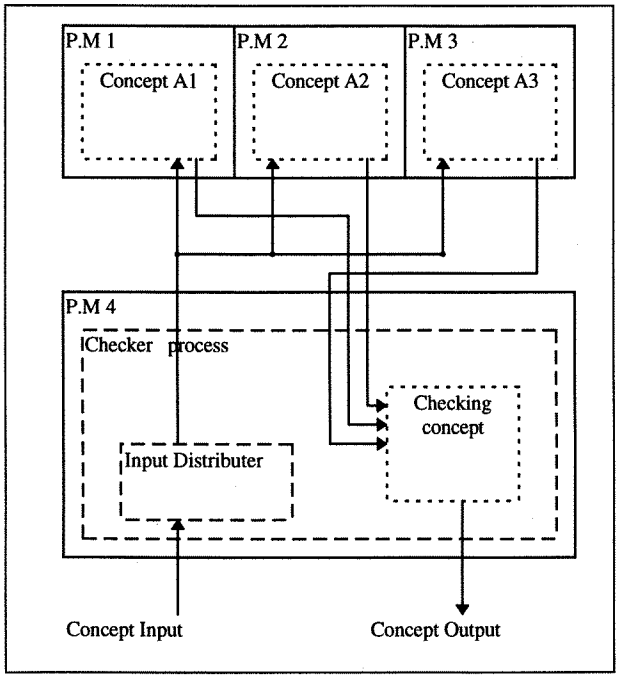

Figure 3: Concept distribution strategy.

The creation of redundant instances of concepts can take place via two different techniques. Different programmers could develop different concepts, each of which perform the same function. This would be analogous to conventional techniques for fault tolerant software development. Likewise, a programmer could recognise that there are different ways of achieving the same task, and could develop the redundant concepts him/her self.

Alternatively, if a learning algorithm were to be employed, the system would distribute identical copies of the previously engineered concept across the array. A tunable learning algorithm could then be invoked to generate different rules in the different concepts. Such a learning algorithm is currently under development.
Once the redundant instances of the concept are generated, by whatever means, they generate their own output independently. Input to each redundant instance (denoted as Concepts A1,2,3 in the diagram), comes directly from the checker process, which is also responsible for output collection.

The checker process acts as a type of place holding concept, ie all input directed to the concept goes to it, and all output directed from the concept is directed from it. When a concept is created, a checking concept may be defined by the user which would execute a user specified checking procedure. For example, for a flight control law, the rules in the checker concept may check for any indication that an output may lead to a dangerous manoeuvre.

Alternatively, rather than relying on a user defined checker, the system may look at the value of a 'certainty' measure which is generated by each alternative concept. From this information, a particular alternative concept is chosen as a 'winner' for a particular round, and its outputs are driven to the required output destinations.

In order for this scheme to function, concepts must be fully reactive, ie output should never be produced in anticipation that the next output will be of a particular value, or that a particular output took place in the past. This is important since at any time the output of a another redundant concept might be chosen.

If any processor fails, the system will have active alternatives, which can continue with processing. If any of the alternative concepts contains incorrect fuzzy logic rules due to programmer error, or learning, it can be picked up by the checker concept, and output taken from an alternative concept, potentially designed by another programmer. If the PM housing the checker process fails, the individual concepts will time out, then cooperatively create a new checker process, which takes over the role of the old checker. This guarantees recovery.

3.2 Systems software

At the core of the PACERs software systems sits the RTEMS (Real Time Executive for Military Systems), a real time executive which was written for the US army missile command by the On-Line Research Corporation, and is freely available via the internet[12].

The kernel provides the following services:

- Multitasking.
- Support for multiprocessor systems.
- Event driven, priority based scheduling.

- Intertask communications and synchronisation.
- Priority Inheritance.
- Responsive Interrupt management.
- Dynamic memory allocation.

RTEMS provides all low level operating system level support for the software of the system. The execution engine on board the PACER is very different to that on the Jabiru AFCS, and has been re written from scratch in the Ada 95 programming language using the GNAT Ada compiler[13], which is also freely available.

Many of the low level tasks which were previously built into the software of the Jabiru AFCS have therefore been taken up by RTEMS, or its device drivers, leaving only the new execution engine and user interface to be written in Ada.

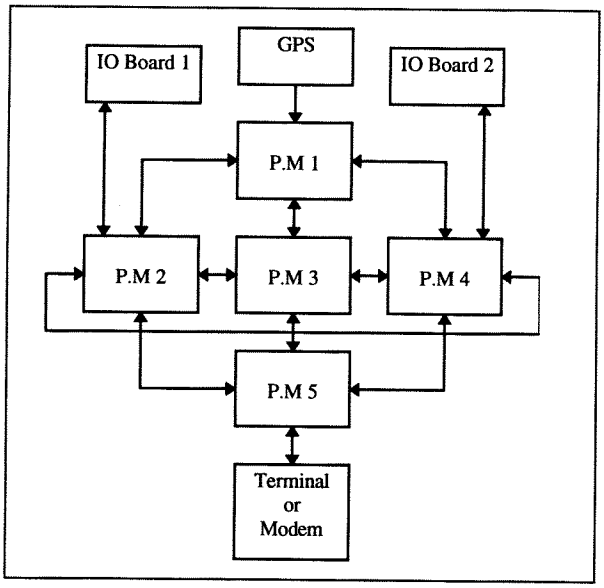

Figure 4: PACER module layout.

3.3 Hardware

The hardware design of the PACER varies dramatically from that of the Jabiru AFCS. In developing the new hardware, two key objectives were at the forefront of all decisions:

1. Fault tolerance at the hardware level had to be ensured.

2. In order to facilitate the adequate operation of the significantly expanded software, the computational power of the system had to be significantly increased.

As a result of these two important considerations, it was decided to make the system a parallel processor array, the organisation of which is depicted in figure 4.

The system consists of a series of processor modules (PMs), each of which is housed on its own printed circuit board. These processor modules each have four 2Mbps communications ports. Three of these ports are dedicated intermodule communications ports (IMCP) , used to link the various modules together to form the parallel computer. The fourth port is configurable and can be used either as a general purpose serial port, or as a fourth IMCP.

In its present configuration, the PACER consists of five processor modules, connected as depicted in figure 4. Three of the five modules are fully connected to their neighbours (ie all four IMCPs are used), whilst the other two use the fourth communications port as a general purpose port. One of the two general purpose ports is used to link up to a SONY Pyxis GPS receiver, the other is used to connect the system to either a dumb terminal or a cellular modem for in flight communications.

The configuration outlined in figure 4 is basically is a small one, suitable for evaluating the system. The system is not, however restricted to that configuration. The number of PMs is limited only by space, power, and cost . Likewise the connection strategy is fully configurable. This organisation makes the system as a whole highly versatile. For example, very low cost systems can be produced with small numbers of PMs, or alternatively very large powerful systems could be marketed, all employing the same basic system components. Their design is outlined in figure 5.

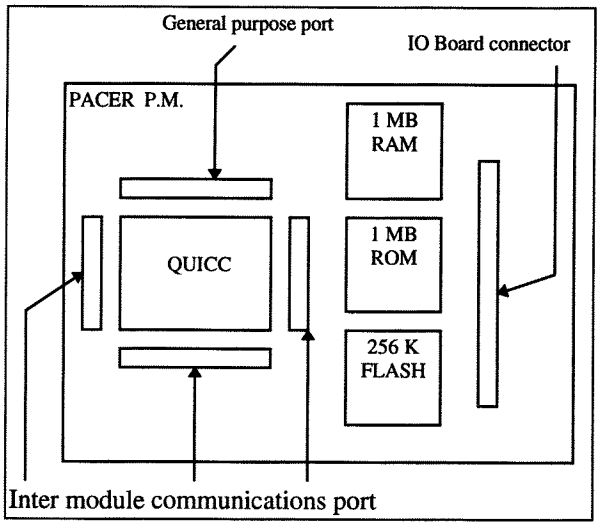

Figure 5: Processor module organisation

At the core of the system is the Motorola 25Mhz 68360, also known as the Quad Integrated Communications Controller (QUICC). This is a high performance microcontroller, with 4 2Mbps serial channels, DMA, and memory interface units built in on chip. It is a very highly integrated processor which has allowed the

designer to keep the overall board space of the PMs minimal.

Connected to the QUICC is 1MB of ROM, for systems software storage, a SIMM socket, initially supporting 1MB of dynamic RAM, and 256K of FLASH memory, which is used to store programs written by the user in the COREL programming language.

Also on each module is a general purpose I/O port. This port allows certain PMs to be connected to I/O devices such as a data acquisition and motor control systems. Given that an IO port is provided on each PM, the number of IO boards is limited only by the number of PMs. For fully fault tolerant operation, more than one of each board can be connected.

4. System trials

Whilst autonomous systems controllers can be tested on the bench to a certain degree (see below), true evaluation of autonomous operation can only take place in a real autonomous vehicle under the very uncertain but real pressures of a real mission.

A simulation of a mission can only contain challenges that the developer, or other engineers have thought of. The beauty of testing a system such as the PACER in a real environment is that it will come up against challenges which a development team would never have thought of putting in a simulation, and it is in such operating conditions that the effectiveness of a system such as the PACER can be correctly evaluated.

The trial application and vehicles which will be described in the proceeding sections will form the testbed for the PACER, and will be used as a means to test the systems ability to cope with a real mission involving considerable uncertainty. It should be emphasised however, that the PACER is a highly generic system, designed to be able to interface with a myriad of vehicles and systems. Once the evaluation process using the MAFV is complete, it is anticipated that trials using different systems will be undertaken.

4.1 The Trial Application.

The application in which the PACER will initially be deployed will be in the investigation of the extent of down wind spread of urban pollution, in particular carbon dioxide, methane and ozone trace gasses.

The concentration of carbon dioxide, for example, has risen by more than 25% over the past 200 years. Ozone depletion in the upper atmosphere is the result of human-produced chlorofluorocarbons and halons. Depletion is especially severe over Antarctica in spring, causing the so-called ozone hole. Results from the analysis of air samples aid in the determination of the main sources of greenhouse and ozone-depleting gases and how they spread once they get into the atmosphere.

The Global Atmospheric Sampling Laboratory (GASLAB) [19] at CSIRO has been involved in making long-term measurements of the atmospheric abundance of carbon dioxide and its stable isotopes. A fixed Baseline Air Pollution Station, located on the north coast of Tasmania, is used for continuous measurements in addition to information collected from a global network of sampling sites. Since 1991, a light aircraft has been used once per month to collect air samples to an altitude of 8 km. When combined, these measurements provide information about the spatial gradients of CO_2 in the atmosphere. Such gradients can then be interpreted with models of atmospheric transport to yield information about fluxes of CO_2 to or from the atmosphere. The ability to exploit this approach to the estimation of CO_2 fluxes is however, currently data limited.

In order to understand the extent of the pollution, systematic measurements on a regular grid in elevation, longitudinal and lateral direction is needed. Unfortunately, the high cost of utilising manned aircraft for this purpose is prohibitive. RMIT and CSIRO have identified a common interest in teaming up in the development of the MAFV as a potential low-cost platform for conducting large scale atmospheric CO_2 measurements. In addition, this project paves the way for other proposed payloads such as: ozone, other atmospheric species via direct measurement and/or flask sampling, infrared radiation, atmospheric turbidity cloud absorption and aerosols.

4.2 The Trial Platform.

The vehicle will be designed to accommodate the airborne continuous CO_2 analyser which currently under development at GASLAB at CSIRO. The mission requirements dictate a high level of autonomous operation for a long endurance mission. This requirement places a challenge on the design of the airframe, in terms of light weight construction, high aerodynamic efficiency, low fuel consumption and reliable engine operation.

The design concept initially investigated was a canard configuration, similar to the Boeing Brave 200 and the Tractel AMT-RPV1, referred to as the MAFV Jabiru [2] (see figure 1). This configuration has potential advantages in cg-travel tolerance, high aerodynamic efficiency in cruise and a simple, modular architecture. A half scale model was built, but it proved to be very sensitive in pitch control. In the interest of project continuity, a twin-boom, pusher configuration was selected for the development of

the full-scale MAFV. This design, referred to as the MAFV Sarus, is shown in figure 6:

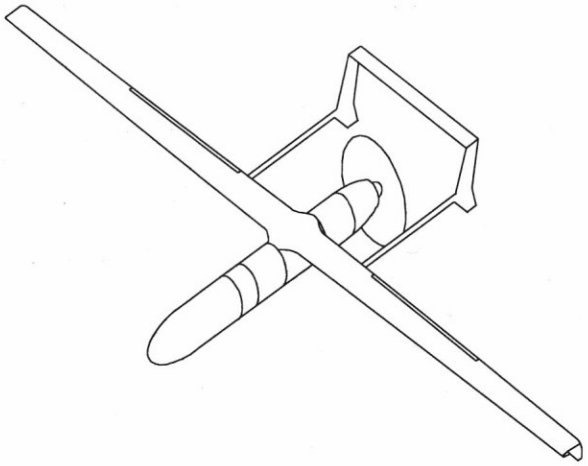

Figure 6: The MAFV Sarus.

This configuration has been widely adopted for military UAVs and has proven to be easily controllable under adverse conditions, such as gust, wind and high angles of attack. The payload is mounted either in a nose cone or under the fuselage, forward of the wing. The fuselage fuel tank consists of two compartments and different fuel tank sizes can be fitted, depending on the mission endurance requirement. Launch and recovery modes currently being investigated, include fixed undercarriage, catapult launch, fall-away undercarriage and net recovery. Special attention must be given to all areas of design to ensure that the vehicle meets its endurance target, this includes considering model aircraft building techniques, different engine and fuel types, etc.

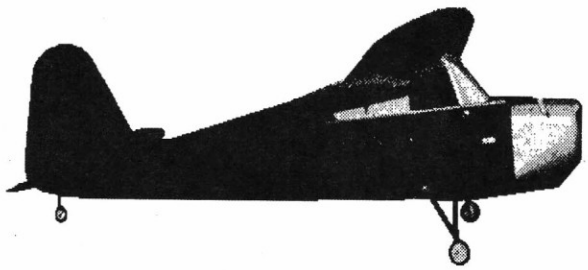

Figure 7: The Precedent T240 model aircraft.

Development of avionics for the MAFV Sarus is proceeding concurrently with development of the flight vehicle itself. In the absence of an airframe, a Telemaster Precedent T240 model aircraft (see figure 7) will be used as an initial test platform. The Precedent is a trainer model, and is consequently extremely stable and easy to control. The model also has a large fuselage section which can be used to carry the electronics. The T240 will be only be used until the Sarus is constructed, at which time long range trials will commence.

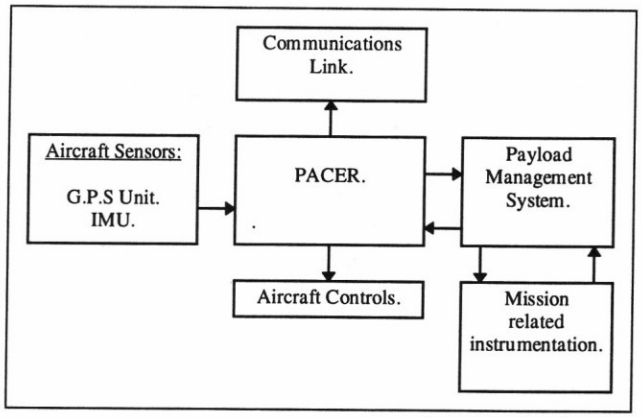

Figure 8: Avionics systems overview

4.3 Flight vehicle avionics subsystems

With the exception of the control system, which has gradually evolved; the avionics systems deployed in the Wackett Centre UAV programme have not changed. Figure 8 presents an overview of the various systems. The modularity of the design of all systems has ensured that the controller, or any other component can easily be replaced with superior, or more cost effective components.

The intelligent control system is connected to both the aircraft flight controls and a series of sensors via a specialised IO interface. The interface supports up to 8 analog channels, and 8 servo motors. Additional interface boards can be deployed however, which will allow for potentially more inputs and outputs if required. A port is provided to directly control a instrumentation manager, which will be initially developed by Australia's Commonwealth Scientific and Industrial Research Organisation (CSIRO), but will be application specific.

It is anticipated that communications with a ground station will initially be provided by a cellular modem connected directly up to a RS-232 port on the controller. Currently this port is used to connect the system to a "dumb" terminal for computer in loop testing.

Flight Instrumentation

The aircraft instruments deployed on board the Wackett Centre UAVs have been designed for the RC model hobby market. They have proven to be reliable and inexpensive.

The Inertial Measurement Unit (IMU) consists of three NEJ1000 Piezoelectric rate gyros from the Japan Radio Co, and three Setra Systems model 141 accelerometers. The rate gyroscopes are designed for use on remotely

controlled model helicopters. Their drift characteristics under constant temperature is surprisingly good, however they drift considerably under temperature variation. This has been counteracted via the use of a temperature sensor mounted under the gyros, and an algorithm which approximately compensates for the drift based on temperature developed as part of the navigation system. The entire IMU is mounted on a single block (strapdown configuration), close to the aircraft centre of gravity.

A SONY Pyxis GPS is used as the system's GPS receiver. This unit is not capable of differential operation, but this is not a problem, given that the GPS position accuracy provided by the GPS SA mode is sufficient for the requirements of the MAFV Sarus. The GPS position is augmented by inertial information to derive a position estimate more suitable for use with a UAV system (see section 4.4, below).

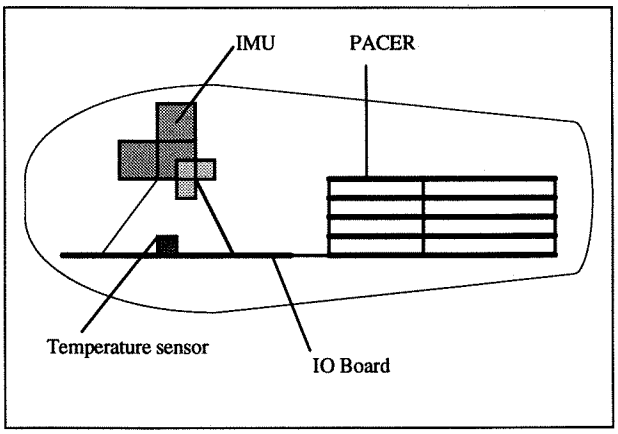

Figure 9: Pod internal layout.

Mounting

All MAFV Avionics are mounted in a 400x150 mm fibreglass pod. The pod provides mounting for all of the printed circuit boards and instruments. It slides into the UAV fuselage, but can also be removed and easily transported, or quickly loaded into the MAFV once a bench test cycle is complete. Alternatively, the pod can be mounted underneath a smaller UAV. The internal layout of the pod is depicted in figure 9.

4.4 Navigation

One of the primary factors which has made UAV technology affordable by the civilian community has been the widespread availability of low cost navigation systems, in the form of GPS receivers. Whilst the GPS is an invaluable tool in UAV navigation, alone it is not sufficient.

Differential GPS can help to correct one of its deficiencies, namely the errors introduced by the Selective Availability (SA) signal, however in many missions envisaged for a UAV differential GPS will not be feasible. For example, if the UAV were required to perform long range meteorological data acquisition over sea, or perhaps at the antarctic, it would be practically impossible to guarantee that a differential base station would be available throughout the entire journey. Selecting a GPS receiver capable of differential operation also raises the cost of the unit. In designing a UAV navigation system, it can not be assumed that differential GPS would be available.

The accuracy of GPS SA mode will generally not be a problem for many of the applications this research is aimed at, however the behaviour of the SA signal can cause considerable problems, and it is for this reason that corrective measures must be taken to compensate for these GPS SA errors as much as possible.

The primary problem with the SA mode in UAV navigation does not come from the fact that there is a large error term, but rather from the fact that this error term can change abruptly. For example, imagine a UAV flying in close proximity to a waypoint, at say 30 m/s. The navigator indicates that the point is 50 metres away from the aircraft. The aircraft therefore continues in the current direction. Suddenly, the next navigation reading indicates that the waypoint is 50m behind the aircraft, an impossible scenario, yet if this type of error is not compensated for the UAV would be forced to make a hard turn, putting it temporarily off course and wasting fuel.

If SA mode GPS is to be used, the following precautions must be taken:

- Some form of position interpolation must take place, to ensure that the flight control system has an updated position in between the GPS update points. This smooths out steering, and must be done to avoid jerky turns being made at the GPS update times.

- Ridiculous position updates must be intelligently filtered out. For example, in the scenario above, rather than turning back the aircraft should simply keep flying forward to the position where it previously determined that the waypoint would be.

It was originally intended that the Jabiru AFCS use a combination of fuzzy logic rules to achieve the above. The rules would have carefully updated the change in the required heading, ensuring that the navigator was not making inappropriate requests of the controller.

This approach would have worked, but in order to improve the estimate of aircraft position between GPS

updates, more and more rules had to be added, looking at things like past trends in direction. As rule count increases, so does overhead. In addition, the updates in position between the GPS updates would be strictly artificial estimates based only on GPS information and a little analysis.

A superior solution was to rely on a Strapdown Inertial Navigation System (INS) for navigation, using the GPS receiver for INS alignment only. The INS provides very good short term information, and its algorithms can be invoked at a rapid rate, leading to much more frequent position updates. The GPS information is used to correct the inertial information, which tends to degrade over time due to effects such as sensor drift.

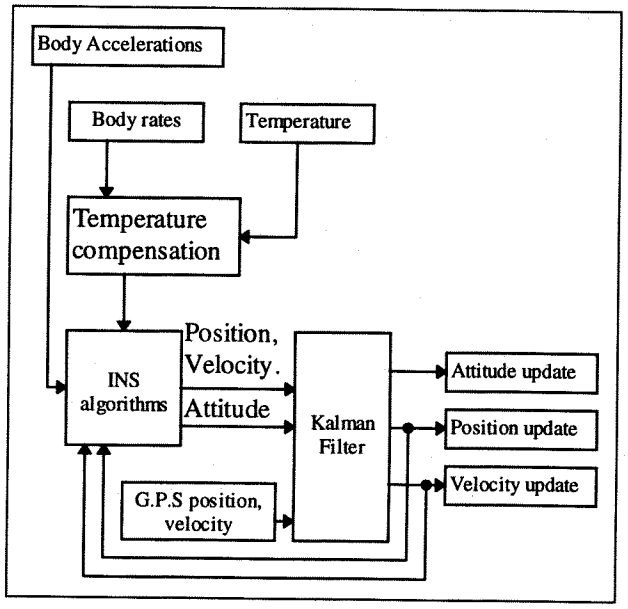

Figure 10: Navigation system overview.

The resultant navigation system can be visualised as shown in figure 10. The body rates are biased to compensate for temperature dependant drift, and are then passed, along with the body accelerations to the conventional strapdown INS algorithms[18], which derive both attitude and position information. GPS position and velocity information is used to update the INS estimates via a Kalman filtering algorithm. The revised state is then back propagated to realign the INS.

4.5 Ground Station

Although the MAFV has been developed for fully autonomous operation, there will be times when it will be necessary to communicate with users on the ground. This would include initial programming, and communication of mission status. This is accomplished via a workstation using a dedicated software package. A screen shot of the ground station software is provided in figure 11, it provides mission planning facilities, utilities to aid fuzzy programming, as well as a data acquisition subsystem.

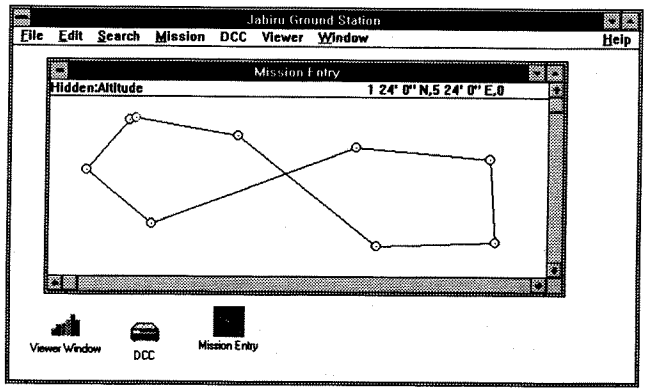

Figure 11: Ground station software.

Editing facilities are available for authoring the programs that describe the flight control aspects of the system. From the editor a COREL compiler can be launched, which will both check and compile programs.

Mission planning facilities include a map drawing program which allows the user to enter waypoints on a map surface with a mouse. Commands can be entered at waypoints instructing the payload management system to execute programs. Mission profiles may be sent to the aircraft at any time, even during flight if the user decides to change the mission.

Finally, the software will provide a display giving all key flight parameters including position, roll rate, yaw, pitch heading etc. This will give the user a good indication of exactly what the aircraft is doing at any time.

4.6 Bench Testing Strategy

Trials of the PACER to date have consisted of simulation based testing. These trials have been of critical importance, as they have picked up problems which otherwise would have caused considerable problems during flight trials. The development process unfolded as follows:

The testing of the INS was an important step which by necessity came before all others. The INS tells the UAV where it is (both its attitude and position). This information is indispensable during trials, and therefore must be verified to be correct. To fully test the INS, an IMU simulator was constructed which derived a set of typical gyro and accelerometer readings which were fed to the INS. The kinematic model used to derive the accelerometer and rate gyro readings was also used to produce position and attitude estimates, therefore testing the INS.

The next step was to integrate a model of the aircraft dynamics[9]. Once this was done, the simulator had been fully constructed, and concept development could commence. Concept development began with the creation of concepts controlling UAV guidance. Once this was completed, autopilot programming and evaluation took place.

5. Future directions

A very solid base has been constructed for the implementation of a highly adaptive intelligent controller. The next step is to develop and integrate the learning algorithms. Many different learning systems have been developed and verified for use with a variety of autonomous mobile robotic systems, eg[11],[20],[21] however none of these fit the requirements of the PACER well.

Although the system presented in this paper makes implementation of adaptive systems for UAVs far more reliable, it cannot control the adaptation process itself, only prohibit poor results of adaptation from being used to drive outputs. It is clear that in a UAV control system, different systems may benefit from different levels of adaptation. for example, autopilot facilities require little to no learning. Indeed, if the concept base representing the autopilot was allowed to change, results could be quite perilous. On the other hand, very high level concepts such as those responsible for exploration, require extensive exploitation of learning. This may go as far as permitting a trial and error approach on the part of the system. The differences in requirements are stark.

It is clear, therefore that it is highly desirable for the concept developer to be able to specify a permissible level of learning. This level of permissible learning should start from no learning through to permitting minor tuning of existing rules, intelligent creation of rules and finally, for very high level processes, guessing. The development of an integrated learning algorithm which can permit definition and control of its level of learning is therefore the current focus of research.

As well as the addition of learning facilities, it is clear that as experience is accumulated in designing concepts, new ideas will be generated as to how the flight control language may be improved. Language development will therefore also continue.

Finally, when a learning facility exhibiting the properties outlined above is integrated into the system, a new challenge will arise. Very few programmers are accustomed to having their programs change over time once they are written. This adds an interesting twist to existing software engineering life cycles, and both techniques and tools will have to be developed to assist designers in coping with this change.

6. Conclusion.

Through a process of critical evaluation, a low cost, fuzzy logic based UAV controller, the Jabiru AFCS, has been transformed into the PACER, a scalable, and powerful autonomous system controller capable of use in both low cost and high performance applications.

This has been accomplished through the deployment of a parallel processor array. Individual processor modules can be linked together to produce faster and more capable UAV control systems, or low cost configurations can be obtained by using only a handful of modules.

The PACER facilitates the construction of UAV control systems which are both hardware and software fault tolerant, and provides a systems organisation which is capable of reliably supporting adaptation of on line programs. The system has been bench tested through the development of simulators which mimic the behaviour of the UAV, and is now ready for in flight trials.

Future challenges include adding a learning capability to the system which is capable of modifying user written programs to an extent prespecified by the user. This ability to restrict the learning is essential if a reliable system is to be constructed.

7. Acknowledgment

The authors acknowledge the support of the Australian Defence Science and Technology Organisation for part of this work under TSS contract 332037

8. References

1. VALENTINIS,F., IKIN,T., KNEEN,J., THOMPSON,L "A Low Cost Fuzzy Logic Based Flight Control System for the MAFV Jabiru", Proceedings PICAST2-AAC6, Melbourne,Australia, March 1995

2. THOMPSON, LA.,BIL,C "The Design and Flight Trials of a Multi Purpose Autonomous Flight Vehicle System".Proceedings of ICAS 94, Los Angeles, 1995.

3. BOURMISTROV,AV., RISEBROUGH,P., MCPEEE,J.,"Fuzzy Logic Based Guidance and Control System for Autonomous Flight Vehicle", Proceedings of Control '95 , Melbourne Australia, October 1995.

4. KNEEN,J., THOMPSON,L.,VALENTINIS,F., "Avionics for an Autonomous Flight Vehicle",International Conference and Exhibition on Electronic Measurements and Instrumentation, Shanghi China, October 1995,

5. ZIMMERMAN,H., Fuzzy set theory and its applications,2nd Ed,Kluwer-Nijhoff 1990.

6. CHIU, S.,CHAND,S.,MOORE,D.,Chaudhary,A., "Fuzzy logic for control of roll and moment for a flexible wing aircraft", IEEE control systems, June 1991.

7. SABHARWAL, W., RATTAN, K., "Design of Rule based Fuzzy Controller for the Pitch axis of an Unmanned research vehicle",Proceedings of the 1992 National Aerospace and Electronics conference vol 2, 1992.

8. TARN JH., HSU FY., "Fuzzy Control of Wing Rock for Slender Delta Wings".Proceedings of IEEE Region 10 Conference TAECON,vol2,1992.

9. SOFYAN,E., BIL,C., DANAHER,RA "UAV Model flight test for parameter identification" Proceedings, IASE2-ISASTI'96 Jakarta, June 1996.

10. FAGG,AH., LEWIS,M., MONTGOMERY,JF. BECKY,GA.,"The USC Autonomous flying vehicle: An experiment in real time behaviour based control" Proceedings of the 1993 IEEE/RSJ iternational conference on intelligent robotic systems, pp 1173-80

11 WHARINGTON, JM.,HERSZBERG,I.,"A Behavioural Conditioning approach to Autonomous Flight Control Design",Proceedings of the first Australian Congress on Applied Mechanics",Melbourne, IEAust.

12. RTEMS Home page: http://lancelot.gcs.redstone.army.mil/rtems.html

13. COMAR,C.,GASPERONI,F.,SCHONBERG,E., "The GNAT Project: A GNU Ada-9X compiler" Proceedings of TriAda '94, Baltimore Dec 1994.

14. BROOKS,RA.,"A robust layered control system for a mobile robot",IEEE Journal of Robotics and Automation,RA-2 April 1986.

15. BROOKS,RA., "The behaviour language users guide" MIT AI lab Memo 1227.

16. COLOMBETTI,M.,DORIGO, M.,"Behaviour analyis and training - a methodology for behaviour engineering", IEEE Transactions on Systems Man and Cybernetics, Vol 26 1996

17. PARKINSON,BW.,SPIKER,JJGlobal Positioning System:Theory and Applications, Vol 1&2, Pogress in Aeronautics and Astronautics Vol 163,AIAA,1996.

18. STEVENS,BL.,LEWIS,FL,Aircrat Control and Simulation,John Wiley & sons, 1992.

19. FRANCEY RJ,STEELE LP,LANGENFELDS RL,LUCARELLI MP,ALLISON CE,BEARDSMORE DJ,CORAM SA,DEREK N,DESILVA FR,ETHERIDGE PJ,FRASER PJ,HENRY RJ,TURNER B,WELCH ED,SPENCER DA,COOPER NL, "Global Atmospheric Sampling Laboratory (GASLAB):Supporting and extending the Cape Grim trace programs", Baseline Atmospheric Program (Australia),193,pages 11-32, February 1996.

20. BONARINI A.,"Evolutionary learning of general fuzzy rules with biased evaluation functions: competition and cooperation",Proceedings of ICEC '94, IEEE.

21. BONARINI A.,"Learning to coordinate fuzzy behaviours for autonomous agents",Proceedings EUFIT '94, ELITE foundation, Aachen Germany.