

On the Design of an Intelligent Agent for the Guidance, Navigation, and Control of Cooperative Robots

Richard D. Anthony

DeVry Institute of Technology, Calgary Campus

ABSTRACT

A robotic agent is developed from the hybrid model, where provisions are made to allow the modeling layer to use sensory information from multiple agents in the reconstruction of a shared environmental model. A hierarchical execution layer uses context-dependent blending to create a dynamic fuzzy behaviour arbitration policy taking into account sensory information and mission goals. Weight counting defuzzification is used to provide a true consensus from the command fusion process. Fuzzy supervisory control is used in the design of a generic self-tuning parametric fuzzy PID controller for low-level actuator control, and an implementation of the agent is presented for the guidance, navigation, and control of a small model helicopter.

INTRODUCTION

Since the initial mission in 1995, the difficulty level of each new mission of the AUVS International Aerial Robotics Competition (IARC) has risen exponentially. Accordingly, every year successful entries are displaying new and innovative system designs capable of increasingly advanced autonomous behaviours. By motivating new research, by guiding existing research, and by sparking an interest in the technology in various economic sectors, the IARC competition is effectively accelerating progress in the new and rapidly-evolving field of aerial robotics. Extrapolating this trend, it is easy to predict a rapid increase in demand for highly intelligent and highly versatile low-cost autonomous systems. Moreover, as the applications of the technology themselves become more advanced, attempting to meet their novel requirements with single robotic entities could easily lead to systems of insurmountable physical and computational complexity. As a result, the idea of cooperation amongst robotic systems is progressively gaining popularity. This vision has motivated the DeVry AUVS Robotics Enterprise (DARE) to invest time in the development of a new robotic agent.¹ The new agent uses a scalable network-ready architecture, giving it the flexibility to operate either as an independent entity or as part of a sophisticated multi-agent system.

This paper is by no means a complete specification of the agent, but rather a preliminary sketch focusing primarily on architectural aspects. We present an overview of the system, as well as an implementation for the guidance, navigation, and control of a small model helicopter. Wherever possible, commercial off-the-shelf (COTS) components are used in order to minimize development time and reduce implementation cost.

¹ The term *agent* is used here to mean an intelligent self-contained fully autonomous system used for the guidance, navigation, and control of a robotic or vehicular platform. An agent can communicate with other agents to perform a collective task.

It is assumed that the reader is somewhat familiar with classical control theory, and possesses a general knowledge of fuzzy controllers and fuzzy set theory. While the reader need not be intimately familiar with the problem of behaviour coordination in behaviour-based robotics, some previous exposure to the subject would be beneficial. Coverage of the subject can be found in [1].

PROBLEM STATEMENT

The 4th IARC mission (which began in 2001) requires robot contestants to “fly to a specified location from a distance of 3 kilometers and identify a particular structure. Once the structure has been identified, a sensor probe must be sent into the structure to perform reconnaissance of a particular type”[2].

Like the previous IARC mission, the 4th mission requires autonomous navigation, collision avoidance, target identification, and accurate mapping capabilities. In addition, the current mission brings four significant new challenges to the arena: (1) the imposed fifteen-minute mission time limit demands that the robot be capable of relatively rapid forward flight across the three kilometre ingress; (2) the location of the target information requires the robot to perform close-quarters manoeuvring in a non-engineered, and hence non-accommodating indoor environment; (3) the minimum dimensions (one squared meter) of a structure entry point limits the size of the robot, thus creating a conflict with the requirements of challenge (1) mentioned above, and (4) the robot entering the structure may be required to operate in close proximity to humans, and therefore must not pose a physical threats to any persons sharing its environment.

STRATEGIC APPROACH

The combination of these new and old challenges creates an engineering problem that is very difficult to solve using only a single robotic entity. The problem can be reduced by using a cooperative multi-agent approach. Following the latter scenario, a solution can easily be envisioned whereby a delivery agent capable of rapid forward flight (helicopter or plane) carries one or more probe agents (micro air or ground vehicles) as part of its payload.² Once at the target site, the area is surveyed from a safe distance to (a) locate the target structure and (b) map any unobstructed entrances to the structure. If one or more suitable entry points are located, the probe agents are assigned a target portal and dispatched. The probe agents rely on dead reckoning techniques and exteroceptive sensor navigation to quickly reach and traverse their assigned portal.³ Once inside, the probe agents make use of simple wall-following and reactive obstacle-avoidance behaviours to navigate the entire structure while transmitting audio/video signals from within. The delivery agent, still outside the structure, acts as a repeater for these audio/video signals, which are then received and analyzed at a remote monitoring station.

We have chosen to center this year’s efforts on the design of the delivery agent and the behaviours it needs to reach the target site. Our goal is then to design and implement an aerial robotic agent capable of autonomous waypoint navigation. The focus is on the agent’s scalable and network-ready architecture, as it is the backbone of the entire strategic approach.

² While the rules of the IARC 4th mission limit us to the use of a single delivery agent, the same strategy could be used with multiple cooperative delivery agents.

³ In the case that a ground vehicle is used as a probe agent, a suitable propulsion mechanism onboard the delivery agent could be used to launch the probe agent into the structure.

DESIGN

The agent described here borrows elements from biological organisms, data communication protocols, classical control theory, soft computing methodology, aircraft navigation schemes, as well as techniques used in autonomous ground vehicle navigation. This eclectic approach has resulted in a robust real-time adaptive system that is both highly scalable and highly flexible.

ARCHITECTURAL OVERVIEW

While the agent's architecture is in fact of the hybrid type, for the sake of orderliness, we have chosen to present it in the form of the traditional "Sense-Model-Plan-Act" (SMPA) paradigm. Figure 1 below (inspired from [3]) describes the general flow of information within the system from an architectural perspective. The *sensor module* collects raw data from the agent's proprioceptive, exteroceptive, and exproprioceptive sensors. The data is pre-processed, and passed on to the *modeling module*. The modeling module uses this information to reconstruct an accurate model of the agent's environment. In a multi-agent scenario, contributions from all collaborating agents are used in the reconstruction process. The model includes information such as the location of explored and unexplored points of interest, the location of any areas to be avoided, the location of surveyed structures and their entry points, the status and location of each delivery agent, as well as the status of dispatched probe agents and the location of their assigned structure and portal. Based on this information, each agent's *planning module* makes a decision as to where the agent should go and what it should do once it gets there. The planning module then updates its current *target* parameter. This parameter acts as a setpoint for the low-level *execution module*, which safely guides the agent to the target by sending appropriate commands to the platform's actuators. We use the word 'safely' because the execution module continuously monitors exteroceptive sensor data, allowing it to detect and avoid any obstacles in its flight path while seeking its target.

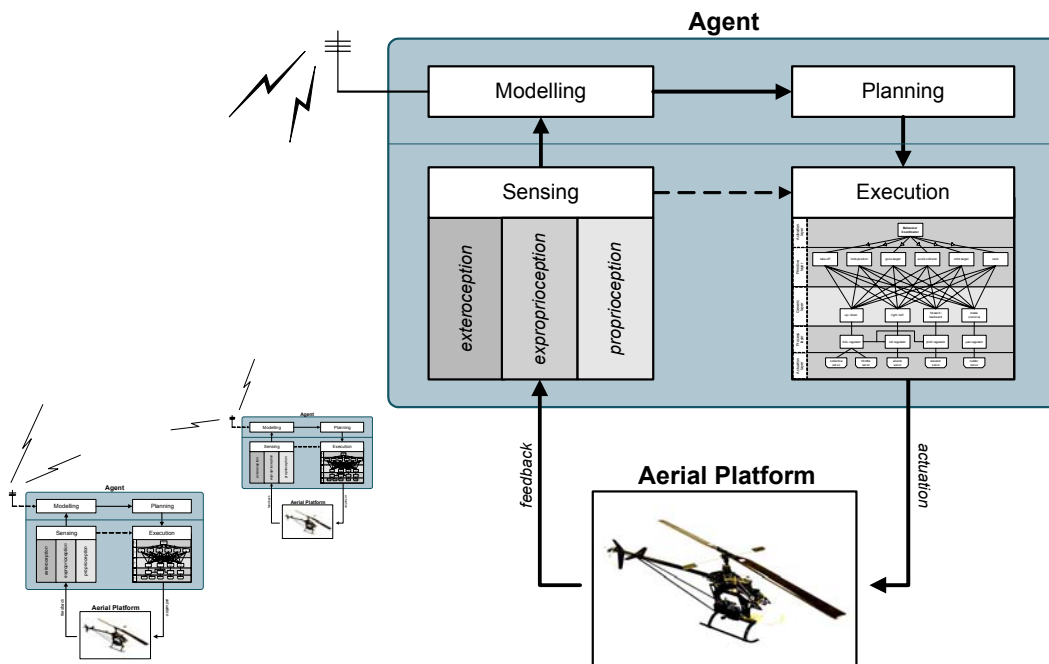


Figure 1. Hybrid architecture of the cooperative aerial robotic agent.

INSIDE THE EXECUTION MODULE

We build upon an architecture suggested by Tunstel in [4] for the goal-directed autonomous navigation of a mobile robot inside a non-engineered indoor environment. Our approach, portrayed in Figure 2, differs slightly from that of Tunstel in that, rather than having the *primitive* layer send commands directly to the robot's actuators, we have further sub-divided the execution module by inserting a *generic* and a *process* layer into the behaviour hierarchy. The process layer houses a group of specialized low-level regulatory behaviours that communicate directly with the actuators, whereas the generic layer takes care of the *command fusion* problem, thus serving as an interface between the high-level primitive layer and the low-level process layer.

The Approach

We begin by decomposing the complex task of autonomous goal-directed navigation into P simple *primitive* behaviours, each capable of achieving a specific navigation task [4]. Primitive behaviours are implemented using G Mamdani-type fuzzy controllers (less their defuzzification step), where G is the number of behaviours in the generic layer. For example, the *go-to-target* primitive behaviour contains four fuzzy controllers (one for each generic behaviour), which operate together to guide the robot to a target designated by its Cartesian coordinates (North, East, Down, or 'NED'). The first of these fuzzy controller determines whether the robot should be going up (climbing) or down (descending) based on the difference between the robot's current altitude and the altitude of the target. The second controller determines whether the robot should rotate (yaw) clockwise or counter-clockwise in order to point towards its target. The third controller determines an appropriate forward speed based on (1) the robot's distance to the target and (2) the target *type*, i.e., is the robot expected to stop and hover at the target, orbit the target, or simply fly through the target coordinates. The fourth and last controller attempts to keep the

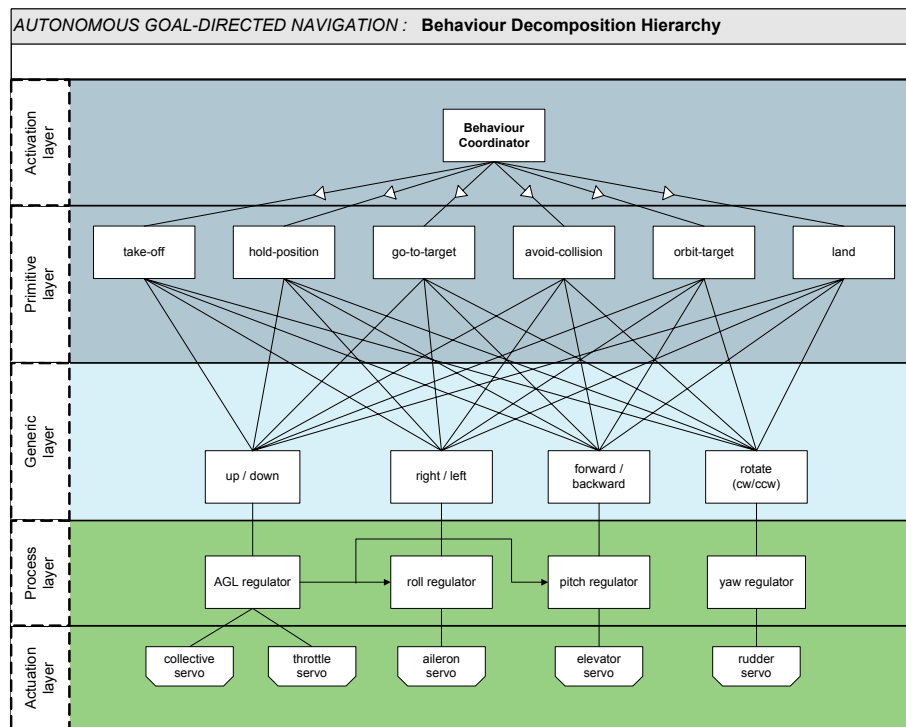


Figure 2. Behaviour decomposition hierarchy of the execution module.

robot travelling in the direction it is pointed towards by commanding right or left translational motion (bank) proportional but opposite in direction to the robot's instantaneous lateral velocity component.

During flight, the behaviour coordinator uses a dynamic fuzzy arbitration policy to *modulate* the activation level α_p of each primitive behaviour p according to its *degree of applicability* (DOA) to the current situation [4]. Each primitive behaviour then requests a particular action from each generic behaviour. This request is expressed in the form of a scaled fuzzy *preference* set $\tilde{\chi}$, where the primitive behaviour's activation level is used as the scaling factor. For example, the output preference set of the p^{th} primitive behaviour's g^{th} fuzzy controller is sent to the g^{th} generic behaviour, and is computed as:

$$\tilde{\chi}_{p_g} = \alpha_p \cdot \bigcup_{i=1}^N \tilde{u}_{i_g} \quad (1)$$

This fuzzy set expresses preference for all possible courses of action that the generic behaviour could take. Each generic behaviour uses the arithmetic sum t-conorm to combine the preference sets it receives from the primitive behaviours, and the resultant *aggregated* set is defuzzified to determine the *consensus*, or crisp output of that generic behaviour [4]. Note that for the aggregation process to be valid, the output *membership functions* of the g^{th} fuzzy controller of all primitive behaviours should be identical. Using the Center-of-Area (CoA) defuzzification method, the output of the g^{th} generic behaviour is computed as:

$$u_g^* = \frac{\sum_{i=1}^N u_i \cdot \sum_{j=1}^P \mu_{\tilde{\chi}_{j_g}}(u_i)}{\sum_{i=1}^N \sum_{j=1}^P \mu_{\tilde{\chi}_{j_g}}(u_i)} \quad (2)$$

This crisp output is bound on $[-1, 1]$, and represents a fraction of the maximum possible action the generic behaviour can take. For example, if the output of the *right / left* generic behaviour is +1, the maximum positive roll angle that the aerial platform can safely maintain is selected as the setpoint of the process layer's *roll regulator*, producing the hardest right bank possible. Similarly, a generic output of -1 would lead to the largest negative roll angle being used as the setpoint to the roll regulator, hence producing the hardest left bank possible. Any value that lies between these two extremes results in an attenuated version of the maximum bank. As can be intuitively deduced, a zero output is equivalent to a "do not bank" command.

It should be noted that we have only included generic behaviours for motion in four of the six possible degrees of freedom. We have omitted the other two degrees of freedom because our chosen aerial platform, the helicopter, is incapable of maintaining a roll or pitch angle without inducing lateral or longitudinal motion, respectively.

The Benefits

The hierarchical decomposition of the execution module achieves two important system goals: (1) the complex task of autonomous navigation is made tractable by shrinking the problem down to the design of an arbitration strategy and a few relatively simple primitive behaviours; and (2) a framework is created which allows the designer to easily scale the system according to

the requirements of a given application. Furthermore, the introduction of the generic and process layers allows the designer to reuse the same system for the guidance and navigation of a number of different vehicle platforms by simply swapping one process layer for another, without having to worry about the particulars of the command fusion step. Equally, the designer has the option of modifying the generic layer to experiment with different command fusion schemes without needing to rework any other layer. Hence, the system flexibility afforded by the generic and process layers well justifies their introduction into the behaviour hierarchy.

A Closer Look at the Process Layer

The process layer is the last layer of the execution module before reaching the physical actuators. This layer is responsible for executing the commands of the generic layer. The process layer is platform-specific, and is made up of adaptive regulatory behaviours which communicate directly with the platform's actuators. These behaviours are used to regulate the attitude (roll,pitch,yaw) and ground-level altitude of the robot.

While the function of the regulatory behaviours is highly specialised, their architecture is itself quite basic. These behaviours are made up of a low-level parametric fuzzy PID controller and a high-level fuzzy supervisory controller. The supervisor monitors the performance of the fuzzy PID controller and makes appropriate changes to the proportional, integral, and derivative gains in order to achieve a pre-determined performance criterion. The performance criterion specifies a desired value for each performance parameter. These parameters are the *rise time*, *overshoot*, *settling time*, and *steady state error* of the low-level controller. The PID gains are then tuned to match the performance criterion using the expert knowledge of an experienced process engineer. The use of fuzzy logic allows us to embed the expert knowledge into the supervisory controller in a very natural way by using a finite set of linguistic rules of the form:

$$\text{IF performance-parameter}_x \text{ is } \tilde{A}_k \text{ THEN } \Delta K_P \text{ is } \tilde{P}_k \text{ AND } \Delta K_I \text{ is } \tilde{I}_k \text{ AND } \Delta K_D \text{ is } \tilde{D}_k$$

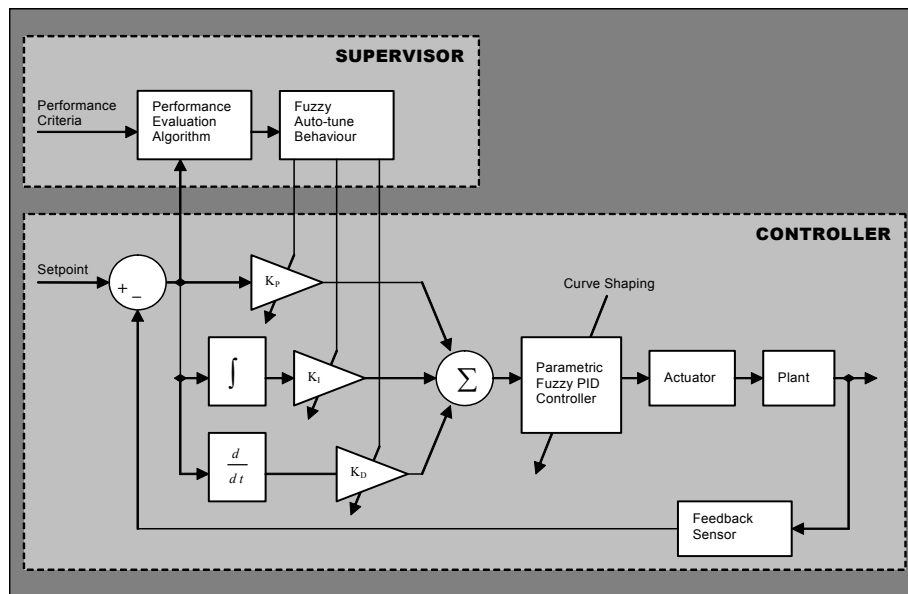


Figure 3. Schematic of the self-tuning parametric fuzzy PID controller.

The controller is called ‘parametric’ because its response to small errors can be morphed between a linear response and a sharp exponential response by varying a single parameter. This prevents a larger error from saturating the controller, while still allowing the controller to respond aggressively while the error is still small in order to prevent it from growing.

The curve-shaping parameter of the fuzzy PID controller is in fact the standard deviation of the controller’s center input membership function. The input and output membership functions of the fuzzy PID controller were arranged empirically to yield the response shown in Figure 4d, which we found to be suitable for our purposes.

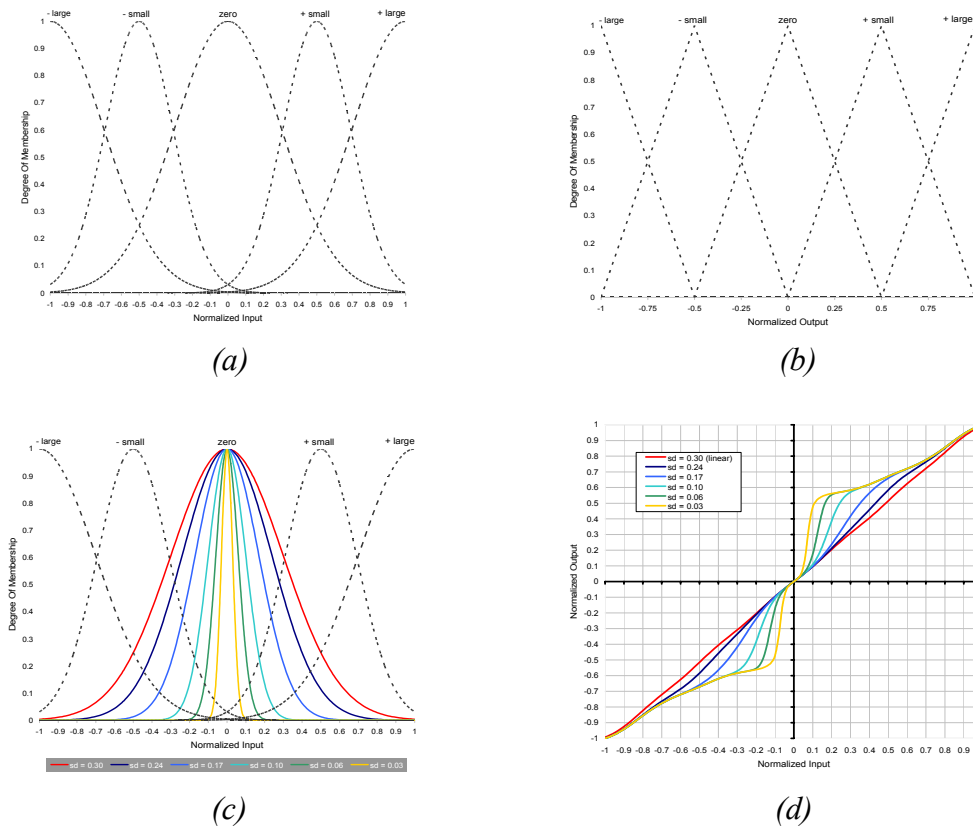


Figure 4. Characteristics of the parametric fuzzy PID controller. (a) Input membership functions; (b) Output membership functions; (c) Variations in the center input membership functions over the full parameter range; (d) Variations in the controller’s response over the full parameter range. ($K_P=1$, $K_I=0$, $K_D=0$)

IMPLEMENTATION

The aerial robotic agent is currently being implemented by a group of five full-time undergraduate students at DeVry Institute of Technology’s Calgary campus. The project is not part of any course work and no academic credit is being awarded to the students for their work. As a result, the implementation of the agent is under significant time and cost constraints. Consequently, commercial off-the-shelf (COTS) components are being exploited wherever it is possible to do so without sacrificing functionality.

HARDWARE

All of the hardware components are mounted inside a hard plastic briefcase, which makes the system portable and independent from any particular helicopter. The packaged system, shown in Figure 5, measures 28 centimetres in length, 38.5 centimetres in width, 12 centimetres in height, and weighs less than 6 kilograms. The only assumptions made of the helicopter platform are that it be equipped with five servos, have a payload capacity of at least 6 kilograms, and have some sort of landing skids. Since DARE already possesses a Bergen Industrial Twin model helicopter (see Figure 1), we have chosen to use it as the aerial platform in our implementation. The Bergen can carry a 10.5 kg payload for 30 minutes on a full tank of gas.

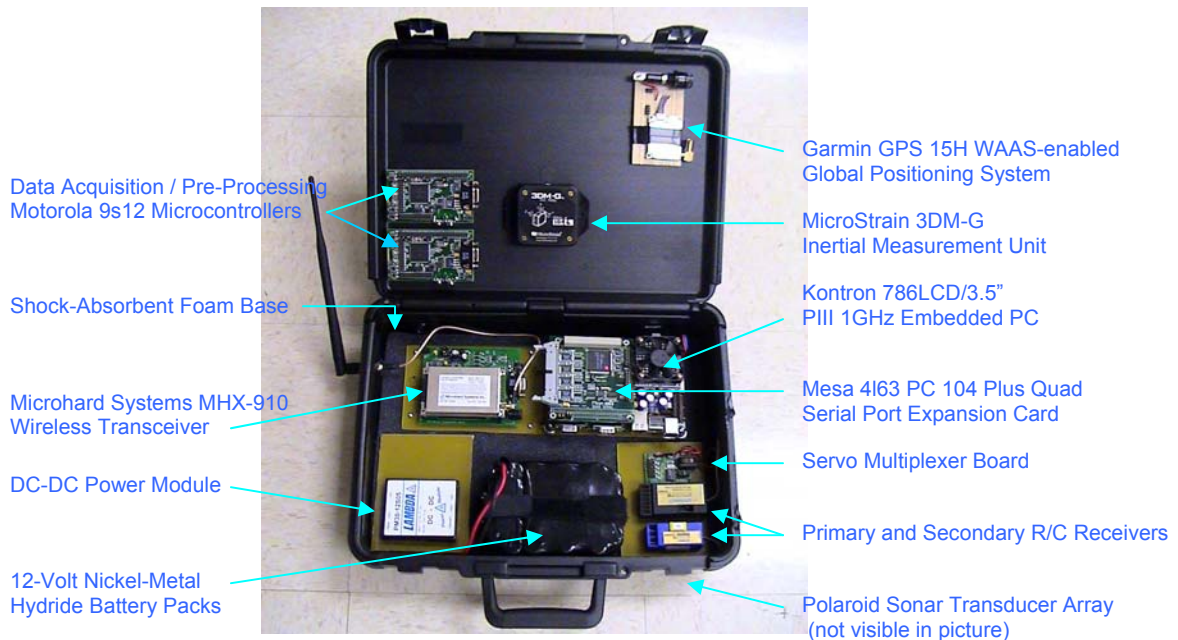


Figure 5. System packaging.

Sensors

The robot requires proprioceptive, exproprioceptive, and exteroceptive sensors to gather the necessary information for (1) the detection and mapping environmental features; (2) the self-localization of the robot in local Cartesian space (NED); and (3) the estimation of the robot's attitude.

Environmental Perception

An array of low-cost Polaroid sonar transducers and ranging modules is used to detect obstacles in the robot's flight path. This system only allows the robot to detect objects 11 meters or closer, and therefore loses its functionality in rapid forward flight. We are currently working on augmenting this system with a stereo camera, which will be used to detect obstacles, as well identifying structures and open portals.

Self-Localization

A MicroStrain 3DM-G Inertial Measurement Unit (IMU) is used as an Inertial Navigation System (INS), which estimates the position and velocity of the robot at 50 hertz. In order to limit drift in the estimates, a crude Kalman filter has been implemented to aid the INS with a WAAS-enabled Garmin GPS 15H Global Positioning System (GPS), which has a 1 hertz update rate. Unfortunately however, the altitude measurements provided by this GPS unit fluctuate erratically in the order of 30 to 40 meters, and cannot be used to effectively aid the INS in estimating the robot's altitude. The robot's altitude above ground level (AGL) is instead resolved by three downward-pointing sonar transducers. The altitude sensing is plagued with the same limitations as the obstacle detection, in that the robot loses sight of the ground if it climbs above 11 meters; this maximum altitude becomes even smaller when the robot pitches and/or rolls. This system will be augmented with a laser range finder as soon as funds become available.

Attitude Estimation

The MicroStrain 3DM-G IMU described above also serves as an orientation sensor. This inexpensive unit provides an estimation of the robot's attitude accurate to within +/- 5 degrees over the full 360 degree range for all three axes.

Processors

Two Motorola 9s12 microcontrollers are used for sensor data acquisition and pre-processing. These microcontrollers use RS-232 serial interfaces to communicate with the main processor. The main processor is a Kontron 786LCD/3.5" PIII 1GHz Embedded PC, to which a Mesa 4163 PC 104 Plus Quad Serial Port Expansion Card has been added. This configuration gives the main processor access to 6 RS-232 communication ports, 2 Parallel ports, 2 USB ports, 1 Ethernet port, and 11 digital I/O lines. The processor has 256 MB of RAM and 512 MB of Compact Flash storage space. It is used for all planning, modelling, and execution computations.

Wireless Communications

The agent uses a Microhard Systems MHX-910 wireless transceiver to communicate with the remote monitoring station and with other robots in multi-agent applications. This transceiver supports point-to-point and point-to-multipoint communications over a range of up to 37 kilometres. The MHX-910 transmits in the 902 to 928 MHz ISM band, and supports serial transfer rates up to 115200 bits per second. It supports user-selectable encryption keys, as well as 62 different user-selectable pseudo-random hopping patterns for the possibility of separately operating multiple networks [6].

Power Distribution

Two 12-volt nickel-metal hydride battery packs are used to power all components with the exception of the secondary receiver, which uses a separate 6-volt battery pack. Nickel-metal hydride batteries were chosen for their high power-to-weight ratio, low cost, and ease of maintenance. Battery pack 1 is used to provide regulated 5 volt and 3.3 volt outputs, which are used to power the main processor, the microcontrollers, the wireless transceiver, and the servo multiplexer board. Battery pack 2 provides unregulated power to the IMU and the GPS, and regulated power to the robot's actuators (servos) and the primary r/c receiver. Battery pack 3 provides power to the secondary r/c receiver which is used for the "kill switch" mechanism.

SOFTWARE

All software is written in the C computer language for maximum portability. Numerous fault management routines have been added to the agent software in order to ensure graceful degradation in the event of component failures. All system failures, including flight instability warnings are relayed to the remote monitoring station over the wireless link. The details of the remote monitoring station and telemetry software will not be discussed as they are outside the scope of this paper.

SAFETY FEATURES

A remotely-triggered “kill switch” mechanism has been implemented that renders the robot completely ballistic if a situation arises that warrants such drastic termination of a flight. For less critical emergencies, a servo multiplexer board has been designed which allows the safety pilot to remotely regain manual control of any or all of the robot’s actuators.

CONCLUDING REMARKS

The use of a behavioural hierarchy in the execution module of the hybrid system has made the agent highly flexible and highly scalable, and is therefore an ideal backbone for the design of next-generation cooperative robots. Furthermore, the introduction of the generic and process layers into the behaviour hierarchy, along with the adaptive nature of the regulatory behaviours, will allow experimentation with different aerial platforms without needing to spend time modifying the entire system.

Budgetary constraints have affected the functionality of the system by limiting the robot’s obstacle-detection capability to objects closer than 11 meters, and consequently, limiting the forward flight speed of the vehicle. Funds permitting, this problem will be overcome by developing a method of self-localization in environments with transient changes using only proprioceptive and exteroceptive sensor data, thus lessening dependence on accurate and continuous GPS measurements. This research will be focused on the use of machine vision to augment dead-reckoning techniques.

We would also like to explore ways of mathematically optimizing the command-fusion process, possibly by borrowing techniques from the field of operations research. This work will be coordinated with research on robot cooperation for multi-agent applications.

ACKNOWLEDGEMENTS

DARE would like to thank the DeVry Student Development Committee for supporting us through funding, Dean Gerald Hunt and Mr. Peter E. F. Bovell for their support with logistics, and Mr. Robert C. Michelson for giving us the opportunity to participate in an event as edifying as the International Aerial Robotics Competition.

REFERENCES

- [1] D. Driankov and A. Saffiotti, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, ed. Heidelberg; New York: Physica-Verl, 2001.
- [2] R. Michelson, “Rules for the current International Aerial Robotics Competition mission,” <http://avdil.gtri.gatech.edu/AUVS/Current/ARC/200xCollegiateRules.html> (Accessed: 21 March 2003).
- [3] A. Saffiotti, “Fuzzy logic in autonomous navigation,” in *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, D. Driankov and A. Saffiotti, ed. Heidelberg; New York: Physica-Verlag, 2001, pp. 3-24.
- [4] E. W. Tunstel Jr, “Fuzzy-behavior synthesis, coordination, and evolution in an adaptive behaviour hierarchy,” in *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, D. Driankov and A. Saffiotti, ed. Heidelberg; New York: Physica-Verlag, 2001, pp. 205-234.
- [5] A. Saffiotti, E. H. Ruspini, and K. Konolige, “Blending reactivity and goal-directedness in a fuzzy controller,” in *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 134-139, San Francisco, California, 1993. IEEE Press.
- [6] Microhard Systems Inc., “MHX-910 OEM Wireless Module,” http://www.microhardcorp.com/products_oem_910.htm (Accessed: 21 March 2003).