System Design for a Small Autonomous Helicopter

Jonathan Roberts, Pavan Sikka & Leslie Overs
CSIRO Manufacturing Science & Technology
PO Box 883, Kenmore, Qld 4069, Australia
http://www.cat.csiro.au/cmst/automation

Abstract

The detailed system design of a small experimental autonomous helicopter is described. The system requires no ground-to-helicopter communications and hence all automation hardware is on-board the helicopter. All elements of the system are described including the control computer, the flight computer (the helicopter-to-control-computer interface), the sensors and the software. A number of critical implementation issues are also discussed.

1 Introduction

Small autonomous helicopters offer a useful platform for a number of aerial applications such as aerial mapping and photography, surveillance (both military and civilian) and power-line inspection. A number of research groups have developed fully autonomous or partially-autonomous helicopters over the past decade [Amidi *et al.*, 1998][Shim *et al.*, 1998][Miller and Amidi, 1998][Vaughan *et al.*, 2000].

In December 1999, our group began to develop a small autonomous helicopter. The main aim of the project was to demonstrate fully autonomous flight (including take-off, landing, hover and forward-flight) within one year and starting from scratch. The ultimate aim of this work is to develop a relatively low-cost and useful platform for applications requiring a low-speed or hovering autonomous flight capability. However, there were two other important initial aims of this project. Firstly, we wished to enhance our existing skills in: control, computer vision and electronics packaging. Secondly, we wished this project to act as a catalyst for collaboration with other research groups around Australia. So far, we have three students from two Universities working on the project and we have hosted the First Australian Workshop on Autonomous Helicopters which was attended by researchers from five Australian research organisations.

1.1 Paper outline

The remainder of this paper is structured as follows. Section 2 describes the overall system design and lists the major automation components. Section 3 describes the so-called 'flight-computer' which acts as an interface between the helicopter and control-computer. Section 4 describes the control computer, which calculates the control demands for the helicopter. Section 4.1 describes the system software and Section 5 discusses some practical problems/issues encountered when developing the system. Finally, Section 6 lists some conclusions.

2 Overall System

2.1 The helicopter

Figure 1 shows the CMST Experimental Autonomous Helicopter. This helicopter is a commercially available 60 size radio-controlled helicopter fitted with a number of custom-made automation system components.

Figure 1: The CMST Experimental Autonomous Helicopter.

2.2 Design constraints

The goal of this project is to achieve complete autonomy of the helicopter. i.e. the helicopter will take-off, hover, achieve forward flight and land autonomously without any input from a human pilot. The human will simply supply a *mission* which the helicopter must complete. This level of autonomy imposes two key design constraints that affected how we have designed and built the system:

1. All processing must be performed on-board the helicopter.
2. There will be no communications between the helicopter and the ground other than the standard remote control transmitter which can be used to control the helicopter in the case of an emergency.

A further design constraint was that of size and weight of the automation hardware. The payload capacity of our helicopter is approximately 3.5kg.

A final design issue was the appearance of the package. This issue is normally ignored by other researchers and their helicopters end up looking very untidy. Our experience with other projects has shown that neatness and good design leads to improved reliability.

2.3 System design

The helicopter is controlled using five Pulse Width Modulation (PWM) servo motors. Normally, during human piloted flight, the pilot's radio transmitter sends the PWM signals to a radio receiver on-board the helicopter. The receiver then sends the signals to the five servo motors. Three extra PWM signals are also sent. One is to set the gain of the on-board yaw stabilisation controller (a standard feature of modern radio-controlled helicopters) and the other two are uncommitted.

[Conway, 1995] designed and built an autonomous helicopter that used a dual HC11 micro-controller based board to allow the helicopter to be controlled via a serial link. The HC11 board converted the demand (serial) into the PWM signals which were then sent to the servos. In his case, a computer on the ground controlled the helicopter and sent control demands via a radio modem. We have borrowed this idea and have designed and built a dual HC12 micro-controller board to perform a similar function. We call this element of the system the *flight computer*. In our case, rather than the *control computer* being on the ground, it is located the helicopter and hence the serial link is implemented using wire and not radio. The overall system design is shown in Figure 2.

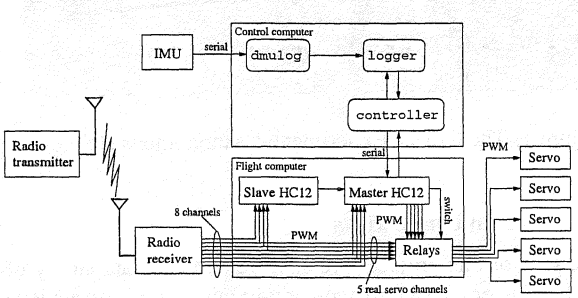

Figure 2: The overall system setup.

There are seven main elements of the system:

- the radio transmitter,
- the radio receiver,
- the five servos,
- the yaw stabilisation system,
- the flight computer based on dual HC12 micro-controllers,
- the control computer based on PC104-Plus and
- the sensors (including IMU).

Figure 3 shows where the elements are located on the helicopter.

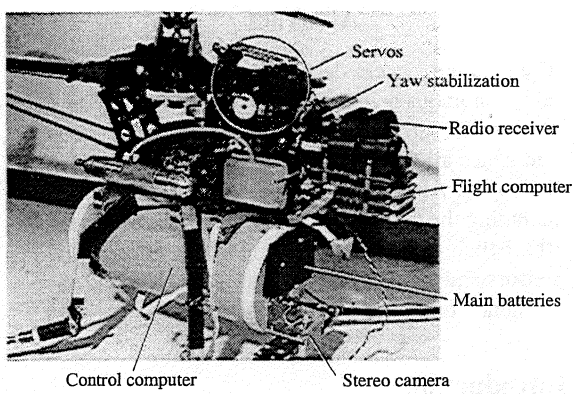

Figure 3: The automation hardware.

2.4 Sensors

We have attempted to configure the helicopter with as many sensors as possible while keeping the weight of the payload to a minimum. We can configure the helicopter with any or all of the following sensors simultaneously:

1. human pilot radio commands,
2. rotor RPM counter,
3. ultra-sound height sensor,
4. 8 landing-gear strain gauge pairs,
5. battery voltage sensing,
6. Crossbow 6 dof inertial platform (DMU-VG),
7. colour camera & gray-scale camera OR Stereo gray-scale camera and
8. GPS.

Sensors 1 to 5 above are sensed by the HC12 flight computer and their data is then passed to the control computer via the serial link. Items 6 to 8 are sensed by the control computer directly.

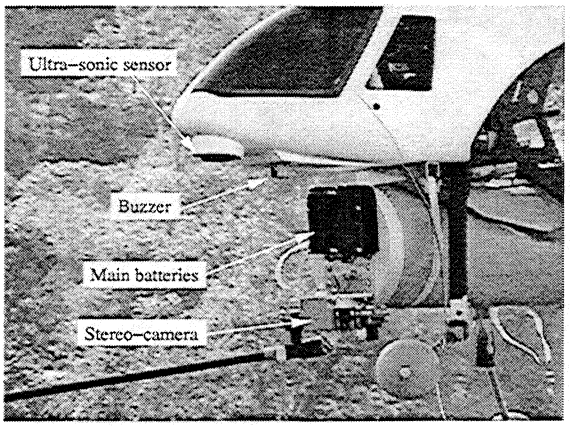

Figure 4: Location of external components.

3 The Flight Computer

The nose of the helicopter houses the flight computer. The flight computer acts as the interface to the helicopter. The flight computer handles all PWM signals and also handles AtoD, DtoA and digital IO tasks. This brings all the helicopter specific wiring to a central point and hence makes interfacing with the control computer simple. The control computer and flight computer communicate via a standard RS232 serial link.

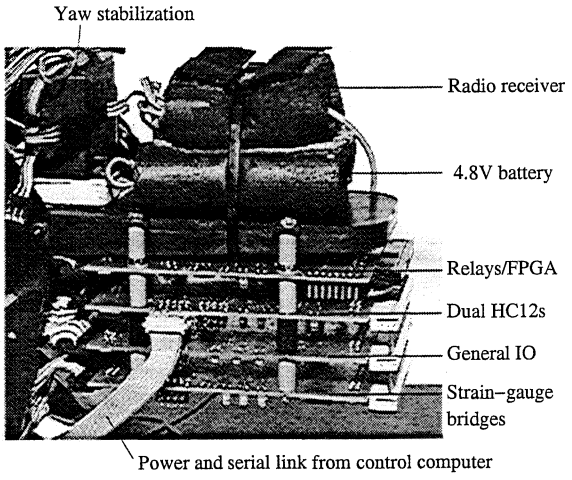

Figure 5: The flight computer.

The flight computer has the following functionality:

- It samples the PWM radio receiver channels.
- It can generate PWM signals for all servos. The servo values are sent from the control computer via the serial link.
- It acts as a servo channel switch, allowing individual channels to be controlled by the control computer or by the human pilot (via the normal radio transmitter).
- It contains 12 ADCs, 2 DACs and 8 DIO lines.
- It monitors the battery level and sounds an alarm when the battery is low.
- It monitors the serial link and switches all servos back to human pilot control if the link is broken. An alarm is sounded.

The flight computer is located underneath the electronics tray in the nose of the helicopter (Figure 5). The flight computer actually consists of a stack of four separate PCBs:

Relay card contains five relays for the five servo channels. (Section 3.1).

HC12 card contains dual HC12 micro-processors that sample the radio receiver channels and generate signals for all servos. The control computer communicates with this card via an RS232 serial link.

General IO card contains two DACs, four 0 to 12VDC AtoD channels, ten 0 to 5VDC AtoD channels, and eight digital IO lines.

Strain-gauge bridge card contains eight strain-gauge Whestone bridges for the landing gear strain gauge sensors.

3.1 Safety philosophy

It is important that in the event of an emergency, or if the computer control of the helicopter is unsatisfactory, that a human pilot can take back control quickly and easily (Figure 6). The relay card in the flight computer provides this capability. This card contains five solid-state relays for the five servo channels and a Field Programmable Gate Array (FPGA) containing some simple safety logic. Figure 7 shows how the safety system works for a single servo channel.

Figure 6: A backup pilot is always ready to take control during autonomous flight.

The card has been designed so that when in an un-powered state, the relays default to a 'straight-through' configuration, i.e. the signals from the receiver are sent straight to the servos. The state of the relays can be changed by the FPGA when the switch-over line is set high by the HC12 card. However, for the switch-over to actually happen the HC12 card must be

'tickling' a watchdog in the FPGA at a pre-defined rate. The panic line must also be low. If at any time the watchdog is not tickled at the desired rate, or if the panic line is set high by the HC12, the FPGA will reset the relays back to the straight-through state.

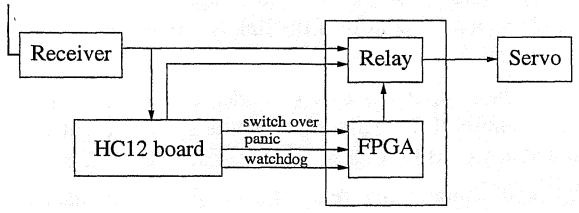

Figure 7: Safety system (single channel) implemented using an FPGA and a relay.

A 12VDC buzzer is located underneath the nose of the helicopter (Figure 4) which emits a high-pitch noise when the relays switch back to their default state. It is then up to the human backup pilot to save the day.

4 The Control Computer

The control computer is housed in a tube mounted underneath the helicopter. This computer is responsible for actually flying the helicopter. It is here that the sensor data is processed and the demand data calculated. The control computer is based on PC104 and PC104-Plus cards (Figure 8) and consists of:

- a PC104-Plus Profive-CPU-P5 motherboard with a Mobile AMD K6 -2-CPU 300MHz processor
- a PC104-Plus Profive ethernet board
- a PC104-Plus Profive VGA card
- a solid-state flash disk
- a Tri-M PC104 power supply
- a Sensoray PC104-Plus frame grabber
- a PC104 GPS card

The computer stack is housed inside a protective tube. This tube is constructed from a PVC pipe which has been drilled with holes to reduce its weight by 60%. The PVC pipe is then surrounded by a layer of brass shim to contain RF within the tube. A final weather/fuel resistant layer covers the shim. The tube is 150mm longer than the standard computer stack allowing room for the stack to grow and providing space for the inertial unit.

4.1 The Control Software

The software system is based on the LynxOS real-time operating system (LynxOS) from Lynuxworks, Inc. LynxOS is based on UNIX System V and also supports some BSD APIs. LynxOS is fully compliant with POSIX.1, POSIX.1b (POSIX REALTIME) and POSIX.1c (PTHREADS) standards and

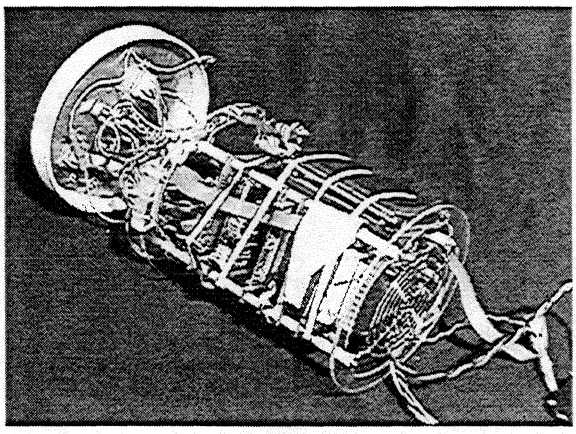

Figure 8: The control computer PC104-Plus stack

that makes it very attractive for developing portable real-time applications.

Software development within the group is motivated by the need to rapidly develop and test applications varying from data logging to control. The applications need to deal with a variety of sensors and actuators depending on the project. Our group has developed several generic applications and tools to meet these requirements. Of these, the logger has been the most relevant for the helicopter project.

The Logger

The logger was developed by our group to fulfill two main tasks:

1. act as a clearing-house for data by accepting data from other applications and also by providing data to other applications, and
2. act as a data logger by continuously logging selected data to disk.

The logger data is specified by a limited subset of 'C'-style data declarations. The logger supports the primitive data-types *boolean*, *integer* and *real*. The logger also supports arrays and one-level deep 'C'-like structures.

At startup, the logger reads its data definitions and parameters from a human-readable configuration file. This file can be easily changed to adapt to different applications.

Applications developed using the logger are client-server with the logger being the server. Once the logger is running, it waits for clients to establish connections using the TCP/UDP/IP protocols. A client can read data from the logger and also write data back to the logger. The logger can support multiple clients. The clients can be run on different machines that are networked together, or on the same machine.

Figure 2 shows an example of a logger-based application in the context of the autonomous helicopter.

Discussion

The logger has been under continuous development by the group since 1996. The first version of the logger was deployed on the Dragline automation project [Roberts *et al.*, 1999]. The Dragline is a 3500 tonne "arm" used in the open-cut mining industry. This experience led to a further refinement and a second version of the logger was developed in late 1998. The current (second) version of the logger has proven to be very useful and allowed us to develop several applications in a very short frame of time. This version was used on the LHD automation project [Roberts *et al.*, 2000] and for several smaller projects within the group. The LHD is a 30 tonne vehicle used to move rock in under-ground mines. This version of the logger is currently being used for the Helicopter automation project. The helicopter weighs 7kg.

The logger implementation is based on the Remote Procedure Call (RPC) library developed by Sun Microsystems in 1988. This library handles data in an architecture-neutral way and therefore makes it easy to have clients running on different computer platforms. Furthermore, the library presents a simple API for inter-process communication. The disadvantage is the overhead involved in providing these facilities. If the entire application runs on the one machine then this overhead can become a significant issue (as is the case in the helicopter).

We are currently developing a new version of these facilities that attempt to provide the best of both worlds (no overheads for applications running on the same machine, and little overhead for applications on multiple machines).

5 Implementation Issues

This section discusses a number of important issues that have arisen during development.

5.1 Radio frequency interference

Because the radio transmitter/receiver combination of a standard remote controlled model helicopter is still being used in the system (as a backup) we must be very sure that the additional electronics added to the helicopter do not create an RF problem and cause the receiver to receive erroneous commands. This could obviously be catastrophic if any of the servos are being controlled manually. Erratic behaviour was observed during some early experimental flights which were attributed to RF interference.

This problem has been addressed in two ways:

- We have shielded all significant sources of RF.
- We have replaced the original piece of wire that acted as the antenna of the receiver with mini-coax and wire. The mini-coax runs down to the end of the landing gear where the wire antenna then begins. This physical separation of RF producers and antenna helps significantly.

A serious cause for concern is the switch mode power supply in the control computer stack. It produces large amounts of RF right around the carrier frequency of the radio transmitter. It is for this reason that the computer stack is shielded using brass shim.

The camera box (mounted on the front of the computer tube) is also shielded with brass shim.

5.2 Disk drive

Initially, we used a laptop style 2.5inch IDE hard disk drive in the control computer. We have found that these disk drives are very susceptible to high frequency vibration. The specification on our drive states that they will not work if subjected to vibrations in excess of 500Hz. We have confirmed this experimentally using accelerometers.

We now use two solid-state flash disks. A small 40MB disk contains the operating system and control/logging binaries, while data are logged to a larger 192MB disk.

5.3 Landing gear

The landing gear supplied with model helicopters is not adequate for an autonomous system. The landing gear must be forgiving of harsh landings and must be able to cope with significant non-vertical landing. Landings may occur with significant roll or pitch of the helicopter and also with significant horizontal motion across the ground. The landing gear must also cope with the extra mass of the helicopter due to the automation hardware. Finally, it must be capable of surviving landings with significant vertical speed.

The design of landing gear to meet these challenging demands proved difficult. The current design is our fifth attempt! However, this design seems to work very well and has already saved the helicopter during a crash landing.

The main sections of landing gear are constructed out of 20mm square aluminium tube. Two pieces were constructed as shown in Figure 9. Each of the two pieces was constructed from a single length of tube. Bends were then cut on three sides, bent and then welded. The two pieces were then notched and welded together. Fibreglass poles extend from the ends of the legs with a large polythene ball attached at the end of each. These balls allow the helicopter to skid along the ground on landing.

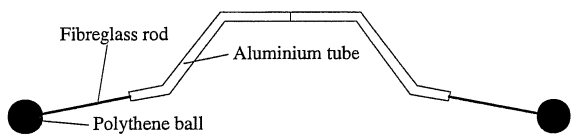

Figure 9: The undercarriage (one half).

5.4 Power

The control computer, flight computer and all sensors are powered by two Sony style 7.2VDC camcorder batteries. These batteries are located on the front end of the control

computer tube allowing for easy battery swapping. The radio receiver and servos are powered by a standard model helicopter 4.8V battery because the high current draw of the servos upsets the computer power supply. The camcorder batteries give a flying time of approximately 40 minutes, which translates to about two full tanks of fuel.

6 Conclusion

The detailed system design of a small experimental autonomous helicopter has been described. Due to our desire to achieve full autonomy with no ground-to-helicopter communications, all automation hardware is on-board the helicopter. The 3.5kg payload limit of the helicopter has constrained the design and has dictated how we engineered a solution. A generic flight computer has been designed and constructed which acts as the interface between the control computer and the helicopter itself. This flight computer may be used in all types of model aircraft that use PWM servos. The flight computer contains safety logic implemented in an FPGA to allow a backup pilot to take control in the event of an emergency. A control computer based on PC104 cards has been design and built. A large number of sensors are located on the helicopter including, a stereo camera, IMU, strain-gauges, rotor RPM sensor, ultra-sonic height sensor, battery sensing, GPS and radio transmitter sensor (to read a human pilot's commands). The general software architecture has been described along with a number of critical implementation issues that have arisen during the project.

Acknowledgments

The authors would like to thank Peter Corke, Graeme Winstanley, Stuart Wolfe, Reece McCasker and Stuart Addinell who all contributed to the development of our experimental autonomous helicopter.

References

[Amidi *et al.*, 1998] Omead Amidi, Takeo Kanade, and James Ryan Miller. Autonomous helicopter research at carnegie mellon robotics institute. In *Proceedings of Heli Japan '98*, April 1998.

[Conway, 1995] Andrew R Conway. *Autonomous Control of an Unstable Model Helicopter using Carrier Phase GPS only*. PhD thesis, Stanford University, March 1995.

[Miller and Amidi, 1998] Ryan Miller and Omead Amidi. 3-d site mapping with the cmu autonomous helicopter. In *In Proceedings of the 5th International Conference on Intelligent Autonomous Systems*, June 1998.

[Roberts *et al.*, 1999] Jonathan Roberts, Peter Corke, and Graeme Winstanley. Development of a 3,500 tonne field robot. *The International Journal of Robotics Research*, June 1999.

[Roberts *et al.*, 2000] Jonathan M. Roberts, , Elliot Duff, Peter I. Corke, Pavan Sikka, Graeme J. Winstanley, and Jock Cunningham. Autonomous control of underground mining vehicles using reactive navigation. In *Proc. IEEE Conf. Robotics and Automation*, San Francisco, USA, 2000.

[Shim *et al.*, 1998] H. Shim, T. J. Koo, F. Hoffmann, and S. Sastry. A comprehensive study on control design of autonomous helicopter. In *In Proceedings of IEEE Conference on Decision and Control*, Florida, USA, December 1998.

[Vaughan *et al.*, 2000] Richard T. Vaughan, Gaurav S. Sukhatme, Francisco J. Mesa-Martinez, and James F. Montgomery. Fly spy: lightweight localization and target tracking for cooperating air and ground robots. In *Proceedings of the International Symposium on Distributed Autonomous Robot Systems*, 2000.