Embedded Control System Architecture applied to an Unmanned Aerial Vehicle

Jaime del-Cerro, Antonio Barrientos, Jorge Artieda, Enrique Lillo, Pedro Gutierrez, Rodrigo San Martín
Departamento de Ingeniería y sistemas de Automática
ETSI Industriales
Universidad Politécnica de Madrid.
José Gutiérrez Abascal, Madrid
Spain
barrientos@etsii.upm.es

***Abstract* – This work describes the hardware and software architectures used for controlling an Unmanned Aerial Vehicle (UAV). It has successfully operated for more than a hundred flight hours. The main aim of this work is to allow quick development and test of control strategies on a complex dynamic system. The software has been designed using parallel modules. Application to the control of a Radio Control (RC) helicopter is briefly described.**

I INTRODUCTION

Number of UAV's used in civil applications is constantly raising specially due to aerial photography and filming fields growth, but to reach a robust and reliable autonomous control for a small RC helicopter is not an easy task. Several Universities and research institutes around the world are working on it.

Helicopters dynamic and different control techniques as classical PID, fuzzy or Neural Networks based controllers applied on it are widely studied in bibliography [1]. [2]. [3]. [4]. [5]. [6]. Nevertheless the implementation of this theory in real applications is not easy to carry out, due to hardware problems that happen when the computers or sensors work on a vibrating machine as a helicopter or trying to execute complex algorithms on a power-limited machine required in light avionics systems.

Another universities or research centers have been developed avionics system for autonomous helicopter [15]. [16]. but not completely based in generic commercial systems.

This work describes the hardware and software design used to built a reliable UAV (Fig.1) using a commercial RC helicopter, (Benzin Trainer by Vario) as vehicle detailing the architecture that facilitates the implementation of attitude and positioning controllers.

A Motivation

Unmanned Aerial Vehicles emerged in the beginning of Robotics. Original UAV's applications were essentially military functions, but recently unmanned aircrafts are used in new fields, such as crops supervision, forest fire monitoring or emergency rescue. For these missions aircrafts can be equipped with cameras, sensors, communication systems or other equipment.

Nowadays some research centers are using the elevate point of view of a UAV for working in cooperating air-ground vehicles for complex missions performance.

Figure 1
Unmanned Aerial Vehicle

UAV degree of autonomy measures how much an UAV depends on a human pilot. Most recent approaches are focused on aerial vehicles with a high degree of autonomy. Compared to the manufacturing of UAV flight hardware, the market for autonomy technology is still immature and undeveloped. Because of this, autonomy has been and may continue to be the bottleneck for future UAV developments, and the overall value and rate of expansion of the future UAV market could be largely driven by advances to be made in the field of autonomy.

B Requirements

Usually, the hardest requirements for an UAV are:

- Adaptability for working in different situations as wind speed, different payload (carrying objects or not) or temperature.
- Flexibility to introduce information from new sensors in the high level control algorithms during complex task as performing visual tracking, or delimiting areas with a high level of contamination, etc,
- Robustness. Safety is the most important aspect to consider when the machine is working into a populated environment. The system must be fail tolerant, relying on some kind of redundancy in sensors, hardware of control and communications systems. Actually, safety elements as parachutes are required by insurance companies for normal operation of UAV's in civil applications.

1-4244-9713-4/06/$20.00 ©2006 IEEE

Nevertheless, the high cost of the avionics and extra sensors as infrared cameras demand high level of reliability.

So as to fulfill these requirements, it is necessary to rely on flexible systems that allow performing fast control test using an extensive variety of techniques. This system usually is a multi-frequency system due to the different kind of sensors and control levels (attitude, velocity, position, and mission). The ability of the system to log the state and control commands and their transmission to the ground to be analyzed in real-time or post situations is also considered as a basic for fast design and results analysis although the most relevant fact is that the software must provide a real-time actuation during the operation.

II HARDWARE DESCRIPTION

The hardware implementation was designed for its use in an small unmanned helicopter. The helicopter has a maximum payload of 5 kg. Usually commercial UAV's based on helicopters have a bigger payload but are slightly expensive air-frames comparing with a commercial RC model.
All avionics and control equipment have to fit in a space of 0,3x0,15x0,4 m. In addition to these restrictions the requirement of autonomy implies to reduce to the maximum the electrical consumption. Fig.2 shows the aluminum rack that support the avionics. The rack has been metal designed so as to reduce the interferences generated in converters and computers for the radio signals.

A Computer

The chosen computer is based on the PC104+ form factor with military specification. This platform was chosen by its small size, consumption and its robustness in high temperature and humidity environments. This is one of the most critical part and the most fragile. Vibrations, heat and electrical interferences had strong adverse effects to these boards so they must be carefully chosen. Usually modern Pentium IV systems require different working voltages with a high consumption that mean heavy converters on board and a small efficiency of the batteries.
So as to insolate the computer from the vibrations and torsion of the avionics box a semi-rigid polyurethane foam as Figure 2 shows. This system has been used after testing different isolators architecture without good results. This system allow to flight for more than one hundred hours without any hardware problem.

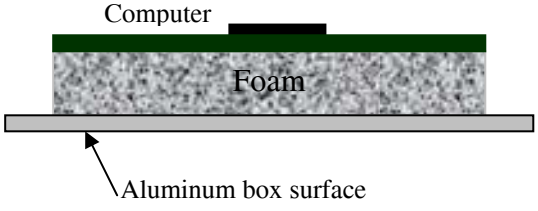

Figure 2.

Montage scheme

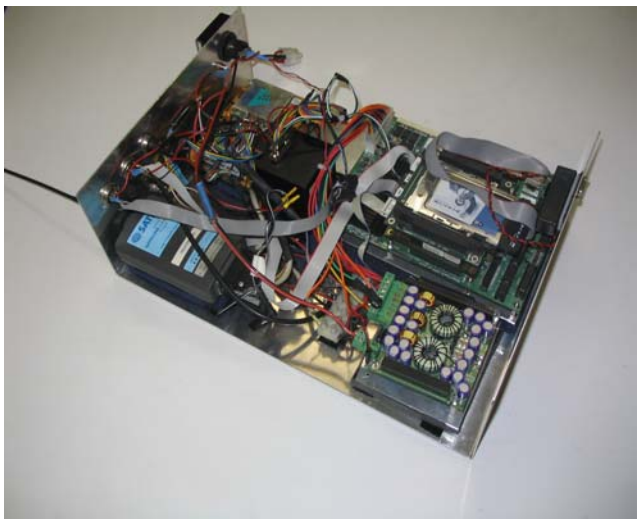

Figure 3
Avionics and control equipment

B Sensors

The helicopter relies on a DGPS receiver Novatel RT2), a 3 axis inertial gyroscope (Microinfinity A3350M) and a magnetic gyroscope (Honeywell HMR3000) to evaluate its state (position and attitude). The measurements of these sensors are collected via RS-232 in an 8-port extension

board connected to the PC-104 bus. Table II details their main characteristics.

TABLE I
SENSOR INFORMATION FREQUENCIES

Sensor	Precision	Frequency	Variables
DGPS	<2cm	20 Hz	Position and Velocity
Inertial Gyros	<0.1º	100 Hz	Angular Rates
Magnetic Gyros	<1º	1 Hz	Heading

C Communications

Communication between the helicopter and the base station is attained using wireless IEEE 802.11b link. This connection is provided by a PCMCIA Wi-Fi card connected to a card-bus slot. This slot is connected to the

processor through the PC104+ bus.

We endow the system with an external antenna so as to improve the range that is about 1 km.
A serial RS-232-C radio modem has been added for receiving the differential corrections from the base station of the DGPS.

D Actuators

Commands calculated by the control algorithms are sent to the helicopter using the trainer port facility of the RC transmitter as Fig.3 shows. In this way, if the control algorithm generated a bad result, an expert pilot would

take the manual control of the helicopter immediately.
For using this trainer signal, it is necessary to generate a very strong time requirements signal with a resolution of some microseconds. For this reason we use an external microprocessor that receives information from the control computer thought the parallel port and generates the corresponding signal using the valor of reference for each stick of the transmitter (throttle, roll, pitch and elevation of the swash plate of the rotor and angular rate for tail rotor gyroscope). The RC Transmitter software is in charge of generating the orders to the different servos that control the helicopter movements. The main point of this solution is that the orders are independent of how the helicopter is constructed in terms of number of servos for the main swash plate, or mechanical adjustments.

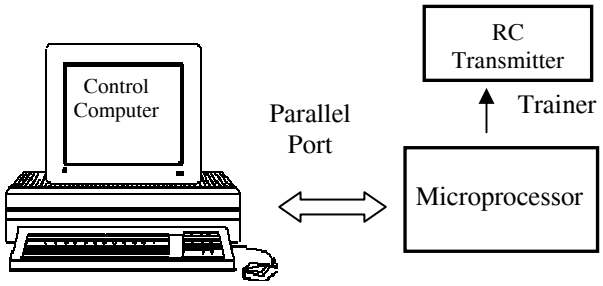

Figure 4
Controlling thought the RC Transmitter

E Power supply

When the payload is a strong restriction, the use of appropriated batteries became essential due to light batteries means long autonomy. A Lithium-polymer battery contributes with a high relation capacity/ weigh.

III SOFTWARE DESCRIPTION

A Parallel Architecture

The mentioned system is based in a parallel processing scheme working on a linux Red Hat 8.0 system [7][8]. This scheme allows monitoring and controlling the sensors and actuators at different frequencies. Every sensor or data channel is read by a thread or process. Processes communicate each other through shared memory or a TCP-IP socket. This communication is synchronized by POSIX semaphores.
1) Processes: Parallel processing is implemented with different processes or threads. Different processes can be started with the instruction *fork* or can be different executables started separately, as in the case of real time modules or client threads on remote machines.
In this way, software running onboard the helicopter is composed of sensors attention software, filtering information software EKF (Extended Kalman filter) and communications servers as fig.4 desribes.
Every sensor has his own thread that receives the information from the sensor and put it on the adequate place in a shared memory space controlled by semaphores so as to allow the filters recovering all the information and generating desired state information. Client is in charge of distributing the information to ground computer using Ethernet wireless link. Ground computer runs software in charge of receiving information from the helicopter and perform the control algorithms as well as to write in the parallel port the resulting data.

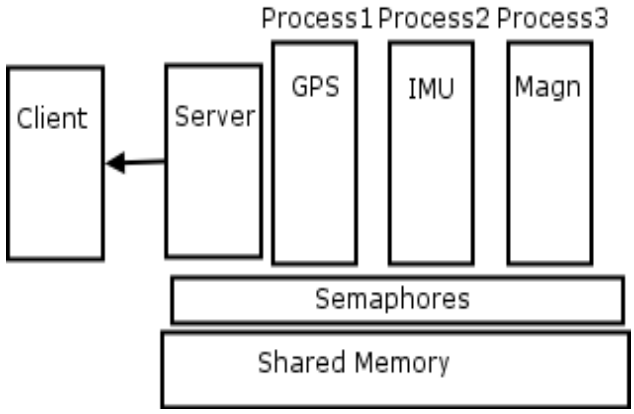

Figure 5
Software architecture

In our case, due to the communication between ground station and air station is performed using TCP-IP sockets, ground software can be boarding any time without any changes in software, although working on a ground computer turns to be more convenience because there is no strong limitations for computer resources like weight or consumption and bigger capacity of monitoring while working.

2) Inter-process communication: Al processes and threads need to communicate each other. This is attained by shared memory blocks or through TCP-IP sockets. The first are preferred with real time modules and with high volume of information while the latter are preferred when communicating processes running in different machines.

3) Synchronization: Different processes need to synchronize between them in order to use different algorithms and to interact with real time modules. Synchronization is also important when using shared memory to avoid conflict when two processes write or read the same area. Synchronism is attained using POSIX semaphores.

B Control architecture

Control architecture is based on a generic approach of state control. This allows a flexibility implementation of controllers. The tools provided by the Object Oriented programming allow building a frame for a generic controller.
The structure is based on a series of objects corresponding with the different control level implemented (mission, position, velocity, attitude) which provides a generic control method. This method takes as an input a description of the state of the system and a reference to be followed by the system.

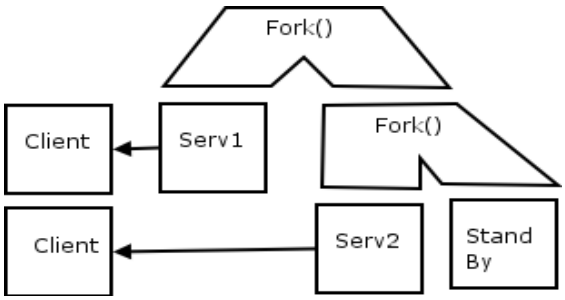

Figure 6
Client-Server Architecture

C Client-Server approach

To fulfill the requirements of transmitting system telemetry for monitoring or identification tasks and to allow to de control modules to run in remote machines at the same time maintaining a reliable behavior a Client-server approach have been adopted.

The software requirements force to allow the reconnection of the client, i.e. if the network fails. In this case, the system will close the server but permit to open a new without crashing.
To solve this problem the server is divided in two processes, the first one attend the request of the client while the other stands by. In case of crashing of the first process the second will attend the reconnection of the client. The second process will be divided again in case of another connection. Another advantage of this approach is the possibility of having multiple clients requesting the data from the server.

D Real-Time modules

The control of real systems usually needs to attend events on real time and to be provided with a precise time base for the execution of the algorithms. To solve this situation the architecture used must be compatible with real time extensions for the operative system.
In this application an approach base on RT-Linux is considered. The structure of the program allows a process be replaced by a kernel module executed in real time space. The communication with the other threads will be made through the real time driver for shared memory provided within RT-Linux.

E Fast Controller Design

Accomplishment of the proposed requirements makes necessary to have a tool that allows the fast design of controllers in order to test them on the real platform. So as to achieve this point, a Simulink implementation of a simulator of the dynamic of a helicopter is used. [13][14]. This dynamic model turns to be useful for testing different flight modes.
Controllers can be designed and linked to the model block so as to test their performance before real flight test. Using the Matlab toolbox "Real Time embedded coder" a "c++" code is automatically generated and ready to be used on the control computer. The proposed architecture provides a frame to compile and execute the automatic generated code. This result in a powerful way of design and test different control strategies.

F Supervision system.

Robust systems have to be fault-tolerant in different situations. In order to simulate sensor failure, a state flow machine using Matlab has been created. This system allow to change the behavior of the helicopter simulating lost of sensors information or transitions between manual or automatic control so as to analyze the controller performance in these situations which are basic for a safety operation.

IV APLICATION

The architecture proposed in this paper has been developed for its use in the control of an unmanned autonomous helicopter. This application requirements of robustness, flexibility, traceability, real-time functionality, different sample frequencies.

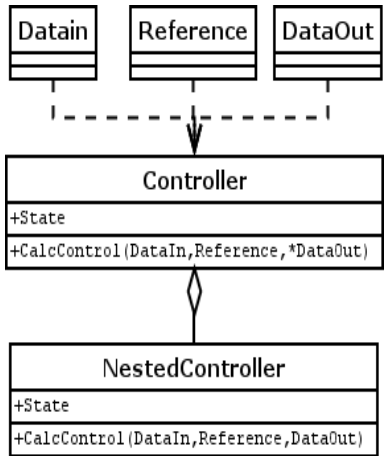

Figure 7
Architecture Software

A Nested Control

As a demonstrator of the ability of design any kind of controllers, the Fig 7 shows a classical PD control structure. A complete work using fuzzy controllers for attitude control can be found in [6].
The outer loop is implemented in a class which has the inner loop implemented as another class declared as member.

B State chart control

The reason of using an architecture based on object oriented programming is show when implementing an algorithm based on a state chart machine. The fig. 6 shows the implementation

C Advanced controller
The design of more complex controller is usually done with analysis tools which can generate C code for the algorithms simulated. Using this architecture and the Embedded Real-Time Toolbox several controllers has been

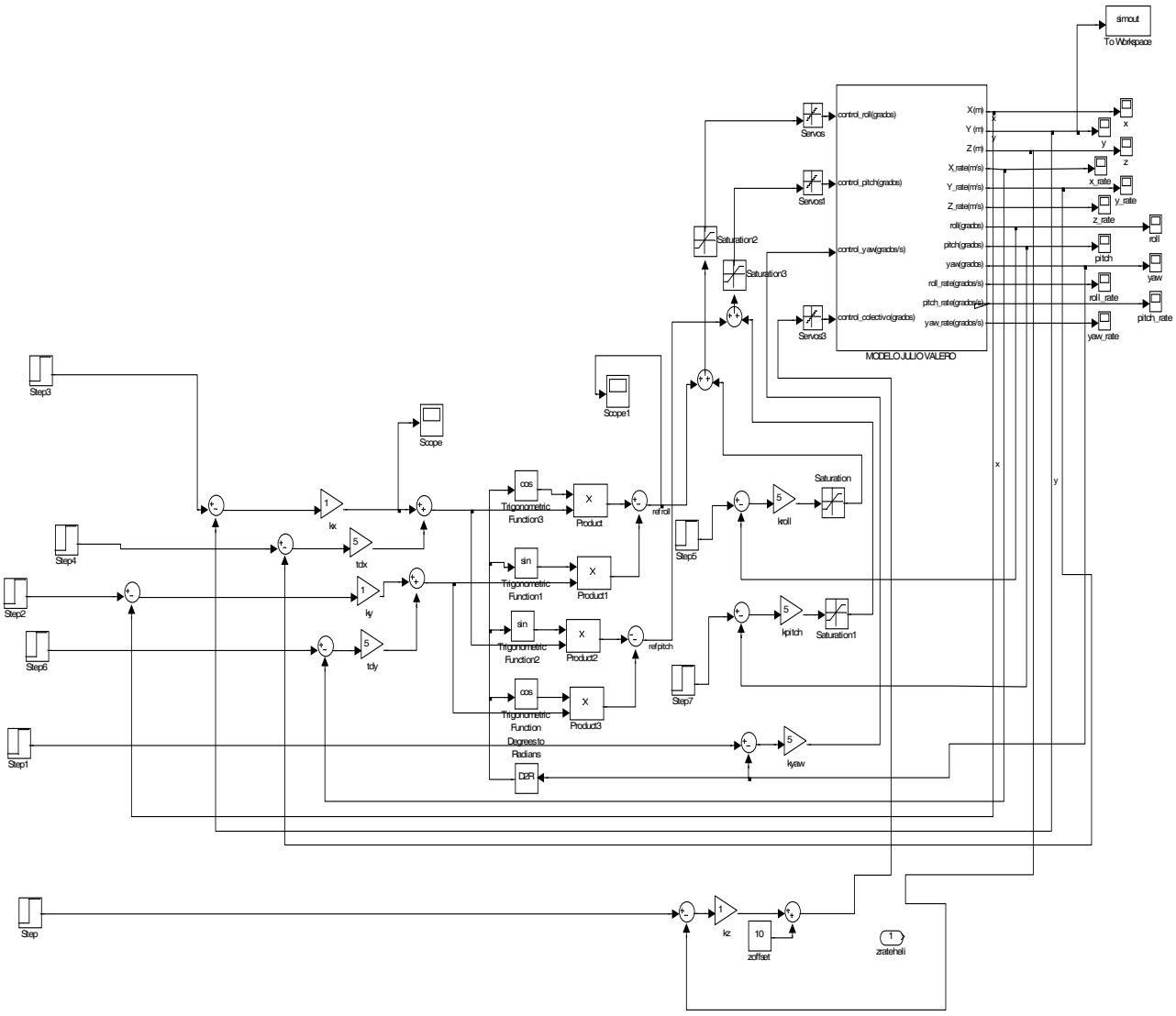

Figure 8
Nested PD Control

implemented. In case of using fuzzy controllers, the inference engine of Matlab has been used after compiling under linux so as to execute the ".fis" file generated by Simulink.

IV RESULTS

The proposed architecture has been applied successfully to the control of an autonomous helicopter. The objective was to attain hovering control around a given position.

The implemented control consisted on a nested control. The inner loop controlled the attitude variables and the Ç outer loop controlled the position variables. The controllers were designed as two PD controllers using the simulator. The system provided the state of the helicopter to the controller class and retrieved the orders to the actuators. The loops were closed according to de table II.

TABLE III

PROCESS REFRESH FREQUENCIES

Process	Frequency
Controller	100Hz
IMU	100Hz
GPS	10Hz
Compass	10Hz

V CONCLUSION

The paper has proposed a robust full architecture for testing control strategies on a complex system under hardware and software point of view detailing each part. The main objective of the paper is not to show the results of the controller's performance, but the ability of the system to allow a fast design of it in an application of a helicopter based UAV. Better performance controllers are the main aim in future works.

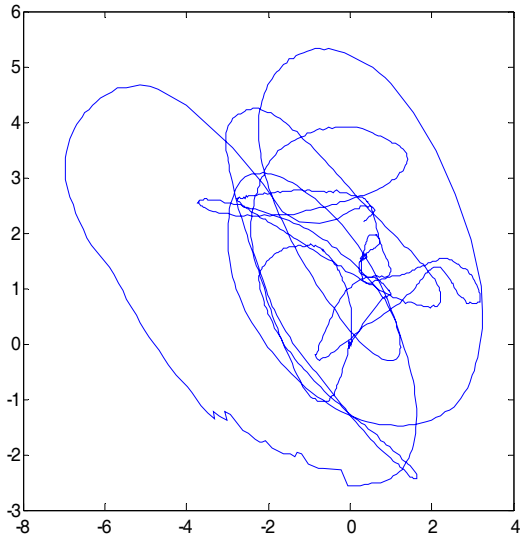

Figure 9
Result for hovering control around (0,0) during 5 minutes.

V REFERENCES

[1] Buskey G, Roberts J. "Autonomous Helicopter Hover Usinga an Artificial Neural Network" *Proceedings of the 2001 IEEE Internatinal Conference on Robotics & Automation*. P 1635-642

[2] Nguyen. "Fuzzy Modeling and Control. Selected Works of M. Sugeno". 1999 P.13

[3] Avila-Vilchis, J. C, B. Brogliato, A Dzul and R. Lozano. "Non linear modeling and control of helicopters". *Automática* 39, 1583-1596

[4] Frazzoli- E, Munther A, Feron-E, "Robust Hybrid Control for Autonomous Vehicle Motion Planning.". *lids-p-2468* December 1999

[5] Aguirre, Iñaki. "Identificación y control de un helicóptero en vuelo estacionario." *Tésis Doctoral. ETSIIM UPM, 2001*

[6] León, Jorge. "Control de un minihelicóptero usando técnicas de control fuzzy". Master degree Tesis. *ETSIIM UPM. 2003*

[7] Yaghmour, Karim. *Building Embedded Linux Systems*. Editorial O'Reilly. ISBN: 0-596-00222-x

[8] Abbott, Doug. *Linux for Embedded and Real-Time applications.* Editorial Newness. 0-7506-7546-2

[9] Ogata, Katsuhiko. *Ingeniería de Control Moderna.* Editorial Prentice-Hall, Inc. ISBN 970-17-0048-1

[10] McGuire, Nicholas, *Minidoc.* Universitae Wien. Inst. F. Materialphysic.

[11] McGuire, Nicholas, *Minirtl FAQ.* Universitae Wien. Inst. F. Materialphysik.

[12] Divakaran, Dinil. *RTLinux How-To*. 2002-08-29 FSMLabs Inc.

[13] Del-Cerro at al. "Modeling and Identification of a small unmaneed helicopter Model." *World Automation Conference (WAC 2004)*

[14] Del-Cerro at al. "Identification of a Small Unmanned Helicopter Model using Genetic Algorithms." *Proceedings on International Conference on Intelligent Robot and System IROS 2005.*

[15] http://pdv.cs.tu-berlin.de/MARVIN/