Alternatives for programming a low-cost UAV in robotics undergraduate course

Mario Castro de Lera(1) Vicente Matellán(2)

(1) Vega Space GmbH. GSOC - Oberpfaffenhofen (Germany)

(2) Grupo de Robótica - Escuela de Ingenierías Industrial e Informática

Campus de Vegazana. Universidad de León. 24071 León (Spain)

e-mail: mario.castrodelera@vegaspace.com, vicente.matellan@unileon.es

Abstract

The aim of this paper are to present different alternatives to program a commercially available flying platform, and to discuss what we consider more convenient to be used by our under-graduate students in a regular robotics course to teach them basic vision based control. Due to the background of our students, we will focus more on the description of the software application interface, than on the hardware, although we will describe also both. We also will describe

Keywords: Teaching, SDK, ARDrone, ROS, URBI

1 INTRODUCTION

Attracting students to computer and electronics related studies is becoming a problem in developed countries. Robotics has been identified as a good tool for increasing the enrolling in technical studies [1, 2]. Our group has been using different platforms [3, 4] in the last years, but these classic wheeled platforms are becoming old-fashioned. Fortunately, the miniaturization of inertial sensors chipsets, accelerometer and gyroscopes, have created a wide variety of very light IMUs (Inertial Measurements Units) which are letting manufacturers create very cheap UAVs, that we can use to teach the basic of robotic control.

The quick development of these low-cost flying machines is allowing the introduction of these platforms in universities as teaching tools. Some researching groups have built their own platform themselves, ENAC (Toulouse, France) [1] and ETH (Zurich, Switzerland)[2]. We are not focused on building our own hardware, so we decided to use commercially available ones.

[1] http://www.enac.fr

[2] http://www.eth.ch

In particular, we have chosen the ARDrone from Parrot[3]. It is a commercial quadroptor helicopter which is sold for around 300 euros. The manufacturer allows its use as a programmable platform by providing an open Software Development Kit (SDK). Using this SDK an its specifications, third-party manufacturers have also create alternative development environments, as for instance Gostai (offering Urbi support), or Willow Garage (providing ROS support). In the next sections we will analyze this alternatives.

The rest of the paper is organized as follows. In the next section we summarize the hardware of the system, and the basic software provided by the manufacturer. In the following two sections we are describing third-parties alternatives for programming the drone, so in third section we analyze Urbi and in fourth section the ROS-based approach. Fifth section will be the comparison, and in the last one we discuss the alternatives a justified our decision.

2 ARDrone Hardware and software

The platform ARDrone has been developed to provide a flying toy device controlled from a remote device, like a smartphone or a computer, using a WiFi "ad-hoc" communication. Although it has been designed to be controlled off-board, it has an on-board Linux system running on an ARM processor. The main mission of the embedded systems is processing data from IMU, propulsion engines, and managing communications. The processor is also in charge to control USB, I2C and WiFi comunication.

2.1 Hardware

The hardware of the ARDrone can be described as a traditional robotic platform made of sensors, control, and actuators. We briefly describe these elements in this section.

As sensors ARDrone has integrated a IMU composed of three accelerators and three gyroscopes, one of them is a yaw precision gyrometer. This IMU provided raw accelerations and angular rotation data. From the frontal camera a stream of images is sent to be processed to the off-board processor. The bottom camera release a stremming of black and white of images which by using visual computation are used to provided horizontal speed, they can also be sent to the off-board computer system, but only one video stream can be transmitted, so a switching command is provided. Finally, a ultrasound sensor give the altitude information with respect to the floor.

The control system computes the IMU, the bottom camera, and the ultrasound sensor information to generate stabilization on board commands to the propulsion engines.

The quadroptor is propelled by four engines dedicated to stabilize the platform.

[3]http://www.parrot.com

2.2 Software

The ARDrone embedded operating system is based on a Linux kernel. The embedded software is in charge of providing services to the platform. WiFi ad-hoc connection is created during the boot of the system allowing communication between the off-board and on-board computers. The on-board DHCP server gives the IP address to the connected device. Telnet port is available to accept connections with the ARDrone. It also uses UDP ports to send telemetry and receive commands.

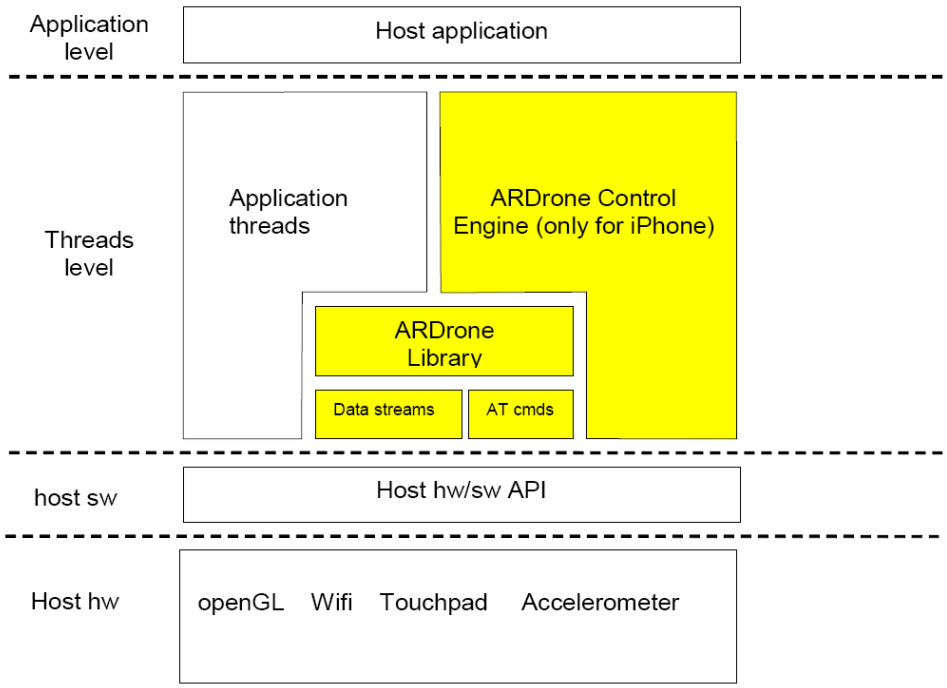

Figure 1: Description of ARDrone Architecture, courtesy of Parrot

3 Control Client Application

The ARDrone was designed to implement a human control through the UDP connection. A Control Client Application (CCA) is an application developed to receive, decode, and analyze the navigation data and the pipeline video information and to send back control commands. In the commercial version, the CCA simply shows the data to a human operator using a mobile phone or a computer, and sends back the commands issued by the operator.

As we have previously stated, the goal of our courses is training students to design autonomous control systems, so using this platform, the goal will be to design a CCA running in a PC. This article analyze three programming environments that currently allow to create Control Client Applications.

3.1 SDK by Parrot

The ARDrone SDK provides the tools and resources needed to Create Control applications for computers, smartphones, and in general, for any hardware device supporting the WiFi ad-hoc mode. The ARDrone SDK allows customize and extend the functionalities of the reference platform.

The ARDrone library has been developed in C++. The Library is currently provided as an open-source library with high level APIs to access to the drone. It provide all the hardware raw values issue from the IMU components. It manages the communications and the video pipeline.

Este apartado necesita un poco más de descripción, o al menos la captura del interfaz de control básico

3.2 Urbi

Urbi is a open source software application for robotics developed by Gostai. Gostai Standard Robotics API allows to developer creating Urbi Engines for standard devices and components implemented as UObject and attibuting methods/attributes/events to access them. The developement of UObject with C++ allows to access to low-level hardware details.

Urbi includes a runtime enviroment, called *Gostai Runtime*. Its development environment includes a set of graphical development tools called *Gostai Suite*, a toolsuite containing *Gostai Studio*, an IDE to observe robotic behavior using final state machines, and *Gostai Lab* to easily create Graphical User Interfaces (GUIs).

Urbi is designed to create components and drivers, called `UObjects`, which can run on top of *Gostai Runtime* and manage the hardware devices of the robotic platform. It provides interfaces with other platforms, notably with ROS and Matlab. *Urbiscript* is the programming language used to manage the components of the robot. It is a script language object-oriented. Its main feature is the parallel and event-driven for process orchestration.

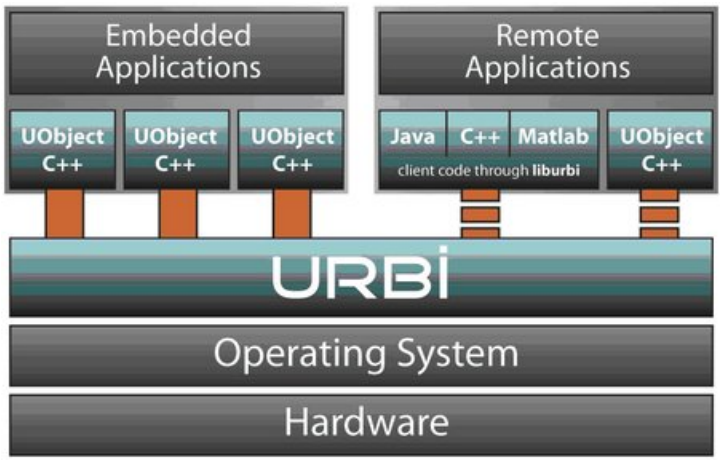

Figure 2: URBI Architecture by Gostai (TM), courtesy of Gostai

URBI proposes *Urbiscript* as programming language. It is a easy language allowing parallel task

execution. As script language it isn't needed to be compiling with each new modification.

The program in *Urbiscript* can analyze navigation data and perform visual computing to generate the commands.

The module developed for URBI has been created in C++ to give easy access to navigation data and video pipeline and create friendly commands.

esta frase necesita más explicación. También un poco del ejemplo

3.3 ROS

ROS (Robot Operating System)[6] provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS was originally developed in 2007 under the name switchyard by the Stanford Artificial Intelligence Laboratory in support of the Stanford AI Robot (STAIR[**?**])

The programming languages used in ROS are C++ and Python. The University of Brown has developed a specific stack for ARDrone to have access to navigation data and pipeline video from the frontal camera. This driver provide also the commands to be sent from the CCA to control the ARDrone.

Esta sección necesita algo más. También algún ejemplo

4 Software comparative

The three different control programming systems proposes in this paper allow to develop CCAs.

The SDK by Parrot provides a very high access to all the hardware of the ARDrone. However programming using the SDK is reserve to advanced developers with a high C++ skills. The libraries release with the SDK can be used as source code to obtain drivers for other software environments.

ROS is an Open source platform that mantains thanks to a big community which has created an abundant amount of libraries. ROS is an extended platform for robots programming, and also for learning basic robotic skills, so using it is an advantage for robotics students.

On the other hand, URBI is the easiest programming enviornment when we use *Urbiscript* to control the ARDrone. We can produce a control application with a MMI with a low programming skill. Unfortunately there is not, at this moment, many libraries to develop control application but the bridge to use ROS module open de possibility to perform advanced control application by using both platform. For example, in *Urbiscript* with 22 code lines we can controlled the following tracking a red ball by visual computing.

5 Conclusion

We propose to use ARDrone as an accessible, cheap, flying platform for robot programming due to the low cost and easy programming.

In this article three software programming solution are proposed to provided different level of hardware access, programming complexity and visualization as tool for robot teaching at the University. From the point of view of teaching drivers development and advanced hardware access, we consider the original SDK as the best solution. This is also the recommended software environment when we want to teach flying stabilization control techniques.

From the point of view of robotic control teaching, *Urbiscript* is a programming language accessible for beginner. The extensive variety of libraries, the graphic visualization and 3D track of attitude data allows to consider ROS also a very friendly platform to teach.

6 Acknowledge

We thank Fablab Artilect and IRIT from Toulouse for hosting and lending the ARDrone equipement during our stay in Toulouse.

REFERENCES

[1] *Retention 101: Where Robots Go ... Students Follow.* Journal of Engineering Education, pp. 85-89, (2003).

[2] Sünderhauf, N., Krause, T. & Protzel, P. (2005). *RoboKing - Bringing Robotics closer to Pupils.* Proc. of IEEE International Conference on Robotics and Automation ICRA05, Barcelona, Spain, pp. 4265-4270.

[3] *Different robotic platforms for different teaching needs.* Vicente Matellán y José María Cañas Plaza. Seminario **Hispabot 2003**. Alcalá de Henares (Madrid), 29 y 30 de Abril de 2003. ISBN 84-607-7238-1.

[4] *Programming commercial robots.* José María Cañas, Vicente Matellán Bruce MacDonald and Geoffrey Biggs. Software Engineering for Experimental Robotics, Series: Springer Tracts in Advanced Robotics , Vol. 30, pp. 125-132. Brugali, Davide (Ed.). ISBN: 978-3-540-68949-2,

[5] *STAIR: Hardware and Software Architecture.* Morgan Quigley, Eric Berger, Andrew Y. Ng (2007), AAAI 2007 Robotics Workshop.

[6] *ROS: an open-source Robot Operating System.* Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, Andrew Ng. International Conference on Robotics and Automation (ICRA 2009).