Prototype Robot for Computer Vision and Control Systems Applications

Augusto Masetti

Supervisor: Lucas Terissi

Laboratorio de Sistemas Dinámicos y Procesamiento de la Información
Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Universidad Nacional de Rosario

augmas15@gmail.com
lterissi@fceia.unr.edu.ar

Abstract. This paper describes a robot designed and developed by a student in the context of an Electronic Engineering degree course. This robot is composed by three wheels, two of them can be controlled independently and the third one is used for stability. The robot also includes a webcam provided with pan and tilt control. This work was focused on the implementation of a prototype useful for academic research in the areas of Computer Vision and Control Systems Dynamics. In this document, the main characteristics of this robot are described.

1 Introduction

The development of autonomous and teleoperated vehicles, including wheeled, legged, undersea and aerial vehicles, for robotics applications has become an important research topic during the past few decades. Mobile robotics has applications in various fields such as military, medical, space, entertainment and domestic appliances fields. In those applications, mobile robots are expected to perform complicated tasks that require navigation in complex and dynamic indoor and outdoor environments with the minimum (or without any) human interaction. There exist a huge variety of applications described in the literature, for instance, for site reconstruction [12, 4] and inspection [7, 5], object recognition [13, 9] and modeling [6, 11], surveillance [10, 1], tracking [2] and search [14], as well as for robotic manipulation and assembly, localization and mapping [16], path planning [3, 17], navigation [8] and exploration. In most of these applications, control strategies for moving the robot appropriately in the scenario, and computer vision techniques for interpreting the surrounding environment, are crucial tasks.

In this document, the description of a simple wheeled robot, developed in the context of an Electronic Engineering degree course, is presented. This robot is composed by three wheels, where two of them can be controlled independently and the third one is used for stability. The robot also includes a webcam provided with pan and tilt control. The main focus of this work is the implementation of

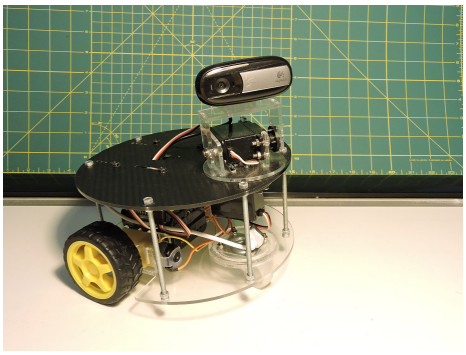

Fig. 1. Picture of the robot.

a prototype that could be used for academic research in the areas of Computer Vision and Control Systems Dynamics.

The rest of this paper is organized as follows. In section 2, the physical description and the electrical components of the robot are presented. The electrical connections and communication protocols between the components are described in section 3. Section 4 presents some examples of computer vision algorithms already implemented on the robot. Finally, some concluding remarks are made in section 5.

2 Robot Description

The robot, shown in Fig. 1, is composed of a cylindrical chassis build up with two acrylic disc-like parts. These discs were designed using CAD software and joined together by six threaded metal rods with their respective hex nuts. The chassis houses the electronics and has two brackets on top for mounting and moving the camera. The robot has three wheels, two on the back that can be controlled independently, and a fly wheel on the front included for stability reasons. The two wheels are driven by two DC Motors with a gearbox and the brackets by two S3003 Servo Motors for panning and tilting the camera.

Regarding the electronic devices, the robot includes a Raspberry Pi (model B+) computer, an Arduino (Pro Mini 5v @16MHz) board, a Logitech USB WebCam and a TP-Link WiFi Dongle. The Raspberry Pi controls all the characteristics of the robot, such as wheels and camera movements, WiFi connection and image acquisition. It also provides the interface to control the robot via secure shell service (SSH). The Arduino is used to generate and send the Pulse-Width Modulation (PWM) signals to control the DC and Servo motors. The Arduino board includes a FTDI driver for programming it. The robot also includes an USB power bank, a 9V Battery Pack, and a custom made Printed Circuit Board (PCB) with 5v regulator and DC motor driver. The approximate cost of the robot is US$ 120.

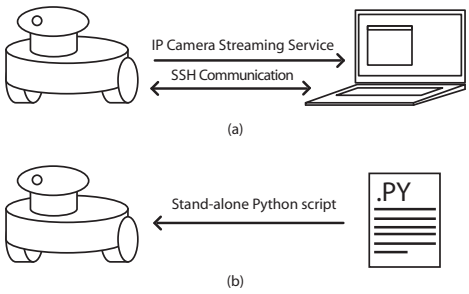

Fig. 2. Schematic representation of the possible modes of operation of the robot. (a) User-robot interactive mode. (b) Stand-alone operation.

From the user side, the robot can be accessed via a WiFi connection. As it is schematically depicted in Fig. 2(a), the robot has an image streaming service for accessing in real time to the images acquired by the camera, and a SSH communication service to controlled it. The robot can be employed to perform on-board processing, for example, following a predefined trajectory and recognizing or locating different objects in the scene. In this case, as schematically depicted in Fig. 2(b), the corresponding processing routine can be loaded and executed on the Raspberry Pi computer. In addition, the robot can controlled remotely, for instance for computer vision applications with high computational load, such as automatic navigation based on images. In this case, the image processing routine can be executed in a remote computer, receiving the images from the robot, and sending back the corresponding commands to control the wheels or the camera position. The autonomy is around 8 hours.

3 Diagram of Connections

3.1 Electrical Connections

For the sake of keeping the electrical connections tidy a custom PCB was developed using the Eagle CAD Software. This PCB, shown in Fig. 3, contains the Arduino, the 5v Regulator and the DC Motor Driver. It connects with the Raspberry Pi as a “Shield” or “Hat” through the GPIO, acting as a voltage adapter between them.

The Arduino has logic levels of 5v whereas the Raspberry Pi has logic levels of 3.3v, hence a direct connection between them is not possible without some sort of mediation. For that reason, the PCB contains two resistors acting as a voltage divider that transforms the 5v signals from the Arduino into the appropriate 3.3v signals for the Raspberry Pi. The Arduino is connected to the Motor Driver via the PCB in order to send the PWM signals to control the DC and Servo motors. The Arduino board is included in the robot due to the fact that the Raspberry Pi is quite unstable when producing PWM signals. Thus, the Arduino is exclusively used to generate four PWM signals, two to control the wheels and two to control

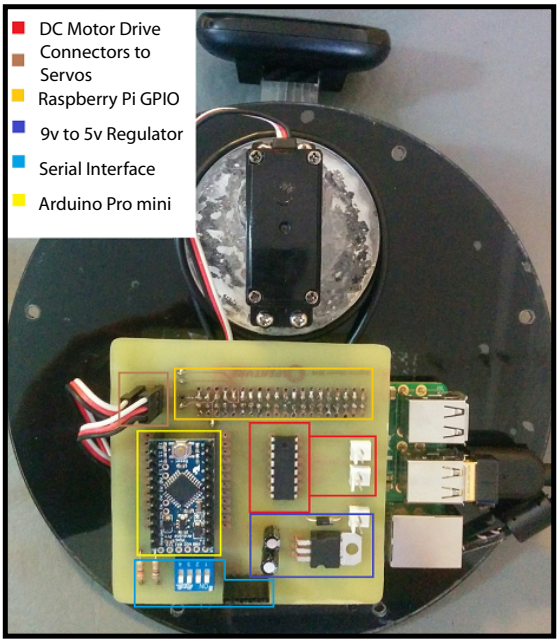

Fig. 3. Picture of the PCB.

the Servo Motors. The Arduino board has 6 digital pins to generate 8-bit PWM signals. In most Arduino boards, when using the standard Servo libraries some pins are disabled, in this case pins 9 and 10. For that matter the remaining four PWM pins (3, 5, 6 and 11) were used.

The Raspberry Pi is connected to the TP-Link WiFi Dongle and the Logitech WebCam through its on-board USB ports. The 9v Battery Pack is directly connected to the PCB for later regulation to 5v, for powering the Arduino, the DC Motor Driver and the Servos. Only the DC Motors are powered directly by 9v Battery Pack.

3.2 Communication

The Robot is able to work using on-board processing in the Raspberry Pi computer, but it was designed to allow processing in a remote computer. The main reason for this is that the speed of the Raspberry Pi to process images is very low. Thus, there will be latency for the actuation that can not be solved with traditional embedded systems. One way of solving this issue is to process the data outside the robot and communicate it the necessary actuation parameters, for the robot to react to that data. Therefore, the robot has a WiFi Dongle and a daemon program that allows devices to connect to it via Secure Shell as well as connecting to a known network for remote control over the internet. Also the Robot has an IP Camera service that can be loaded through its terminal.

The WiFi router daemon starts at the booting stage of the Raspberry Pi. This WiFi Access Point has a range of operation of about 20 meters without walls in between connected devices. Through it the robot connects to other devices via SSH and the communication is secured by WPA2 protocol, without hardcoded usernames or passwords.

The webcam can be accessed via the IP Camera Service that can be launched, as mentioned before, after the booting sequence by a connected device through the terminal. This service is a particular compiled version of the Mjpeg-Streamer Service for the Raspberry Pi, that was originally intended for other Unix running devices. This allows a low latency image acquisition at 60 fps with a resolution of 640x480 pixels, up to a resolution of 1080x720 pixels with a latency of 300 msec. In this robot, the camera is fixed to 640x480 pixels resolution.

Once a connection is established, the robot can be controlled through shell commands and the camera can be access via a web browser or pulled out directly from a script in the controlling device.

The Raspberry Pi and the Arduino communicates through a serial protocol based on fixed length data packages. The Raspberry Pi transmits 8-byte length messages, encoded as illustrated in Fig. 4, to the Arduino in order control the movement of the wheels and camera. The message starts with the ASCII character 'S', and ends with character '*'. The next two bytes, referred in Fig. 4 as P and T, indicate the Pan and Tilt angles for the corresponding Servo brackets. The following two bytes in the message are used to indicate the speed and rotation direction of the left wheel. The first byte (LS) represents the speed in a range of 0 to 255, indicating the duty cycle of the PWM wave, being 255 its maximum for a continuous wave. The duty cycle of the PWM determines the speed of each individual motor via switching on or off the transistors inside the Motor Driver, which is essentially an H-Bridge. The second byte (LD) indicates if the left wheel must move forward (LD = 1), backward (LD = 2) or do not move at all (LD = 0). The final two bytes of the message (RS and RD) indicate the speed and direction of the right wheel, similarly to LS and LD for the left wheel, respectively. Any message with less or more than 8 bytes will be automatically discarded by the Arduino. For instance, sending a message with values [S,80,150,255,1,255,1,*] will drive the Robot forward at maximum speed with the camera facing forward until the next command is received, or a timeout of one second is reached, after which the robot will seize all movement maintaining the actual Servo positions.

In Fig. 5, a schematic representation summarizing the connection and communication between the different electronic devices of the robot is shown.

4 Implementations

In this section, some examples of applications already implemented on the robot are described.

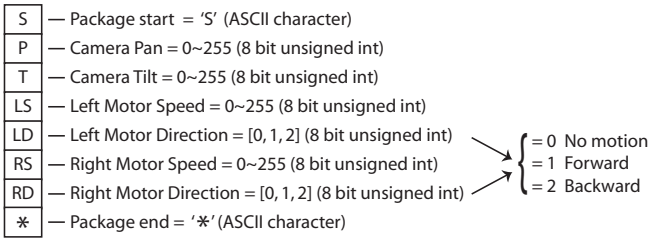

Fig. 4. Raspberry Pi and Arduino communication protocol. Description of the data contained in the transmitted packages.

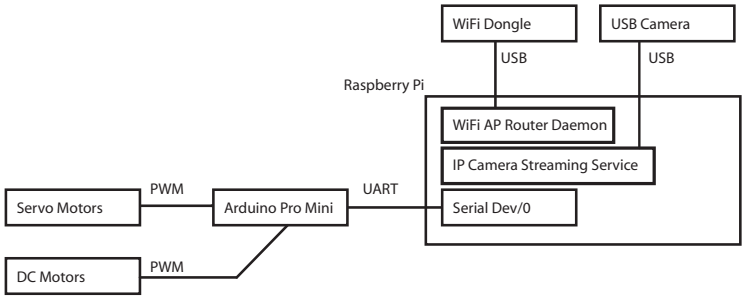

Fig. 5. Schematic representation of the communication between the devices present in the robot.

4.1 Joystick Control

One of the most typical uses for mobile robots is in telepresence applications, in which the user can see a remote place with relatively low latency and move around it in real time. For this purpose, a Python script to be executed in a remote computer was developed. This script contains two threads, one for reading the events from an USB Joystick (see Fig. 6), and the other for sending the control information to the robot and for visualizing the images from the camera. A library called *paramiko* was employed to create an SSH tunnel from the PC to the robot. The images are retrieved via custom made IP Camera Streaming service. Then the images are displayed in real time using *MatplotLib*. The joystick's lateral movements control the speed relation between the two DC Motors, allowing steering the robot in the desired direction. The speed of the robot is controlled by the vertical movements of the joystick. This allows to move the robot forward and backward. Additionally, another on-off button of the joystick is used to perform a safe stop of the robot. If this button is not pressed, the vertical and horizontal control information of the joystick are bypassed, and thus the robot do not move at all. In order to control the pan and tilt angles of the camera, a D-Pad included in the joystick is used. A video showing how the robot is controlled using this joystick can be seen in the following link (https://tinyurl.com/k6czycq).

Fig. 6. Picture of the joystick used to control the robot.

4.2 Face Tracking

Another Python script was developed to track the movement of a person's face with the camera, by controlling the Servo brackets appropriately. This application is executed in a remote computer, and it also makes use of *paramiko* for communicating the robot, and *MatplotLib* library for displaying the images from the camera. For this implementation *OpenCV* library was used to process the images from the camera, in order to detect and to estimate the positions of faces on the images. For this task, the well-known Viola-Jones face detection algorithm [15], based on Haar-like features was employed. As depicted in Fig. 7, this method retrieves a bounding box for every face detected in the image. The idea is to move the camera in order to center the face in the image.

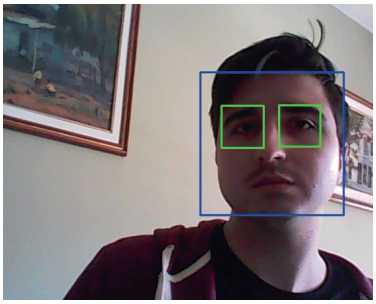

Fig. 7. Face detection example.

For every image from the camera, the algorithm computes the relative position from the face to the center of the image and moves the Servos accordingly. The motions of the Servos to pan and tilt the camera are proportional to the distance from the face to the center of the image, and to the size of the face in the image. If the face to be tracked is away from the camera, its size on the image will be rather small in comparison to the case of being closer to the camera. Therefore, the panning and tilting angles will be different, even though the distance between the center of the face and image will be the same for each case. Additionally, in order to avoid constant oscillations around the face, four boundaries around the center of the image are defined. If the center of the face is inside those boundaries, the robot will stop moving the camera until the face is detected outside it. If more than one faces are detected in the image, the algorithm will track the one that is closer to the camera, i.e., the one that is represented with a bigger bounding box. A video showing how the robot tracks a face can be seen in the following link (https://tinyurl.com/k6czycq).

4.3 Android Phone Control

Another way of controlling the robot is via an Android phone with a custom made application. In this case the java application handles the Secure Shell connection to the robot and allows controlling it without the need of a PC or a joystick. The graphical user interface, depicted in Fig. 8, consists of buttons for establishing the connection and enabling the IP Camera Service, and a D-Pad for moving the robot.

4.4 QR Code Detection

A Python script to detect and read QR Codes was also implemented on the robot. The algorithm is able to read and report back the information, included in the QR code, via the terminal or writing it in streamed images from the robot, as depicted in Fig. 9. This algorithm also is able to determine the location of it in the image. This can potentially allow others more sophisticated tasks, such as automatic inventory in a warehouse or a factory without the need of a human operator. The information included in the QR code could be also used to indicate the robot to perform different tasks, for example to move to a certain position, or to analyze a particular object in the scene marked with a QR Code.

5 Conclusions

In this paper, a robot designed aiming to build-up a prototype useful for academic research in the areas of Computer Vision and Dynamic System Control was described. The robot fully designed and developed by student in the context of an Electronic Engineer degree course. This robot is composed by three wheels, two of them can be controlled independently and the third one is used for stability. The robot also includes a webcam provided with pan and tilt control. In this

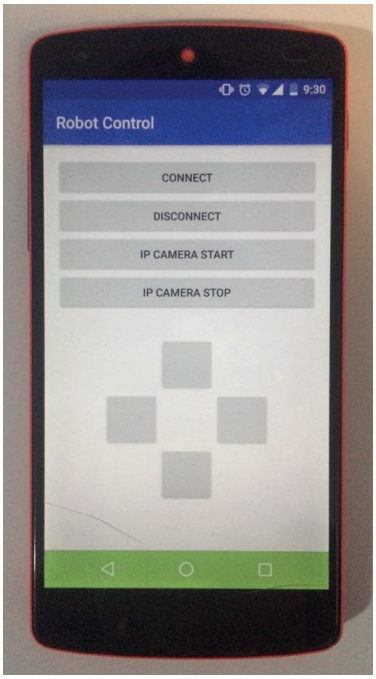

Fig. 8. Android application example for controlling the robot.

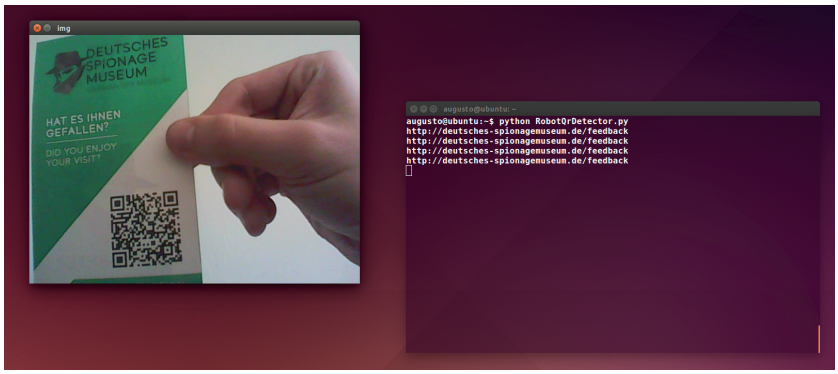

Fig. 9. QR detection and reading example.

document the main characteristics of the robot were described. The robot can operate in stand-alone mode or it can be controlled remotely. Some applications already implemented on the robot were presented.

References

1. A, A.B., Laurentini, A., Rosano, L.: A new lower bound for evaluating the performances of sensor location algorithms. Pattern Recognition Letters 30, 1175–1180 (2009)
2. Barreto, J., Perdigoto, L., Caseiro, R., Araujo, H.: Active stereo tracking of n ¡= 3 targets using line scan cameras. IEEE Transactions on Robotics 26, 442–457 (2010)
3. Baumann, M., Leonard, S., Croft, E., Little, J.: Path planning for improved visibility using a probabilistic road map. IEEE Transactions on Robotics 26, 195–200 (2008)
4. Blaer, P., Allen, P.: View planning and automated data acquisition for three-dimensional modeling of complex sites. Journal of Field Robotics 26, 865–891 (2009)
5. Chen, S., Li, Y.: Automatic sensor placement for model-based robot vision. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 34, 393–408 (2004)
6. Chen, S., Li, Y.: Vision sensor planning for 3-D model acquisition. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 35, 894–904 (2005)
7. Dunn, E., Olague, G., Lutton, E.: Parisian camera placement for vision metrology. Pattern Recognition Letters 27, 1209–1219 (2006)
8. English, A., Ross, P., Ball, D., Corke, P.: Vision based guidance for robot navigation in agriculture. IEEE International Conference on Robotics and Automation (ICRA) pp. 1693–1698 (2014)
9. Farshidi, F., Sirouspour, S., Kirubarajan, T.: Robust sequential view planning for object recognition using multiple cameras. Image and Vision Computing 27, 1072–1082 (2009)
10. G. Sivaram, M.K., Ramakrishnan, K.: Design of multimedia surveillance systems. ACM Transactions on Multimedia Computing, Communications and Applications 35(23) (2009)
11. Li, Y., Liu, Z.: Information entropy-based viewpoint planning for 3-D object reconstruction. IEEE Transactions on Robotics 21, 324–337 (2005)
12. Reed, M., Allen, P.: Constraint-based sensor planning for scene modeling. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 1460–1467 (2000)
13. Ruiter, H.D., Mackay, M., Benhabib, B.: Autonomous three-dimensional tracking for reconfigurable active-vision based object recognition. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 224(B3), 343–360 (2010)
14. Shubina, K., Tsotsos, J.: Visual search for an object in a 3D environment using a mobile robot. Computer Vision and Image Understanding 114, 535–547 (2010)
15. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1, 511–518 (2001)
16. Yousif, K., Bab-Hadiashar, A., Hoseinnezhad, R.: An overview to visual odometry and visual slam: Applications to mobile robotics. Intelligent Industrial Systems 1(4), 289–311 (2015)
17. Zhang, G., Ferrari, S., Qian, M.: An information roadmap method for robotic sensor path planning. Journal of Intelligent and Robotic Systems 56, 69–98 (2009)