Team-based Project Design of an Autonomous Robot

Thomas J. Norrie John S. Seng
Dept. of Computer Science
California Polytechnic State University
San Luis Obispo, CA 93407
{tnorrie, jseng}@calpoly.edu

Abstract

In this paper, we discuss the design and engineering of the C-P30, a custom robot design at Cal Poly State University, San Luis Obispo. This robot is designed by undergraduate computer engineering students at the university. This robot project is intended to be continually developed as students enter and leave the project. In addition, working on the project allows the students to fulfill a university senior design experience requirement. As this project is a continual work-in-progress, this paper outlines the current state of the hardware and software design.

This paper covers the technical aspects of the robot design, as well as the educational objectives that are achieved.

1. Introduction

Robots serve as effective educational tools for teaching engineering in the university. Because robot design requires the interfacing of software, computer hardware, and mechanical systems, robot design projects often challenge students by presenting problems beyond what they typically encounter in the classroom. The multidisciplinary nature of the field of robotics often simulates the complexity of projects students will encounter when they leave school and enter the workforce.

In addition to the experience the students obtain in areas outside of computer hardware and software engineering, team projects also mimic the development model of industry. It is important for undergraduate students to obtain real experience in managing team dynamics, defining hardware/software interfaces, and developing teamwork and communication skills.

As a means to providing students with an opportunity to apply their engineering education in a practical setting, we developed the C-P30 project at Cal Poly. The overarching goal for the project is to develop a robot system that is capable of navigating areas of our campus. The project is developed and maintained by a team of undergraduate computer engineering students. Creating such an opportunity allows these engineering students to work on a large project that involves mechanical, electrical, and software systems. Each student is assigned a particular subsystem of the robot and is required to develop the subsystem and define the interfaces necessary to connect it with the subsystems of the other students.

This paper outlines the design, construction, and management of this student-built robot at Cal Poly State University, San Luis Obispo. This project was started as an on-going project for the computer engineering program. At Cal Poly, seniors at the university are required to have a senior project that lasts two quarters (six months). The purpose of the robot project is two-fold: to provide students with an additional means of fulfilling the senior project course requirement and to provide students with a project which they find motivating and challenging.

In this paper, we document the hardware and software technologies we utilize. As this is a continual work-in-progress, all systems are described as implemented at the time of this writing.

Because Cal Poly State University is primarily a teaching-focused institution, it is difficult for us to assemble the manpower and financial budget to compete in larger competitions such as the DARPA Grand Challenge [1]. We view this endeavor as a challenging intermediate project that provides a similar education experience for the students but at a much lower cost. The challenges in building the robot include interfacing with low-level electronics, vision algorithms, control algorithms, and higher-level planning and decision making.

This paper is organized as follows: Section 2 outlines work that is related to this project. Section 3 outlines the hardware components used in the construction of the robot. Section 4 describes the software architecture and tools used for this project. Section 5 outlines the budget for this project. Section 6 discusses the project management tools used in our development. Section 7 covers the technical and teamwork learning experience. Section 8 describes future work for this project. Section 9 concludes the paper.

2. Related Work

This section briefly describes projects which are related to this work. Related works cover a wide spectrum of designs that range from individual student projects to larger robots such as contestants in the most recent DARPA Grand Challenge [1].

In our project, we focus on providing this project as an experimental platform for undergraduate projects and research.

Our robot is similar in size to the robots provided by the DARPA-funded Learning Applied to Ground Robots [2] program. In this grant program, the robots are provided by the funding agency and used in periodic competitions to test vision and control algorithms when driving over rural terrain. The focus of the program is to develop robots which can function on unstructured off-road terrain using only passive sensors (i.e. no LIDAR or sonar).

The robot described by Ulrich et al. [3] is a motorized wheelchair platform with a single camera mounted on-board. The wheelchair navigates using a monocular color vision algorithm. We have experimented thus far using similar algorithms based on the work described by Ulrich et al.

3. Hardware Architecture

This section outlines the hardware components used in the construction of the robot. We cover what hardware systems are currently present.

Figure 1 provides an overview of the hardware systems present on the robot. The hardware subsystems are designed to be as modular as possible to allow for ease of part replacement and also to ease the task of subdividing projects among students.

For this project, we choose to use a frame and drive system from a motorized wheelchair. Selecting a stable and pre-built drive system advances the project so students can work on subsystems much earlier. A stock motorized wheelchair comes with a frame, motors, batteries, and a motor controller. For this project, we opted to use the frame, motors, and batteries, while substituting the standard motor controller with one that we built ourselves. We keep the original connector in place so that we can reinstall the original motor controller when necessary.

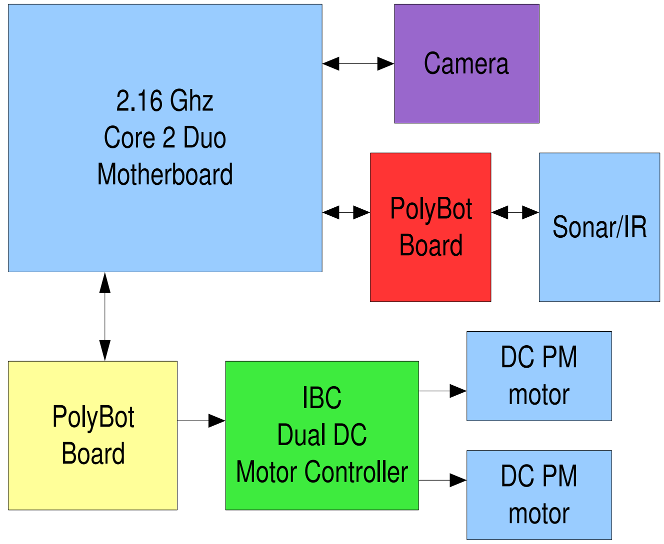

Figure 1. This is a block diagram of the hardware architecture. The primary controller is based on an Intel Core 2 Duo motherboard. The sensor and actuator interface consists of 2 PolyBot boards along with an IBC dual motor controller which controls two DC permanent magnet motors.

The chassis for the robot is a motorized wheelchair (Pride Mobility Jet 1). This wheelchair is powered with two 12V 55AH lead acid batteries wired in series to provide 24V, and give the wheelchair a range of 25 miles. The frame of the chassis is constructed from tubular steel which is very strong. The entire chassis, including the batteries, weighs approximately 100 lbs. A picture of the wheelchair can be seen in Figure 3.

The motors are bi-directional DC permanent magnet motors with an integrated disk brake. The actuation of the brake is performed using a 24V solenoid. We use a p-type MOSFET controlled by one of I/O pins on the micro-controller to activate and deactivate the brake.

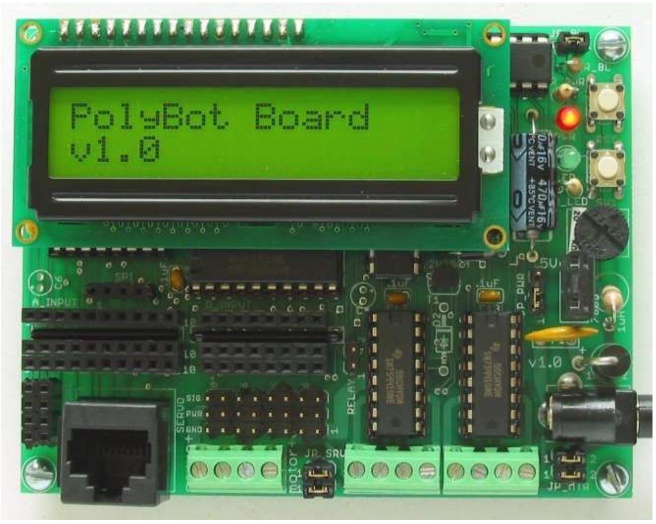

Figure 2. There are two PolyBot boards present on the current robot. One is used to manage the task of reading the sonar and infrared sensors. The other is used to generate control signal for the motor controller.

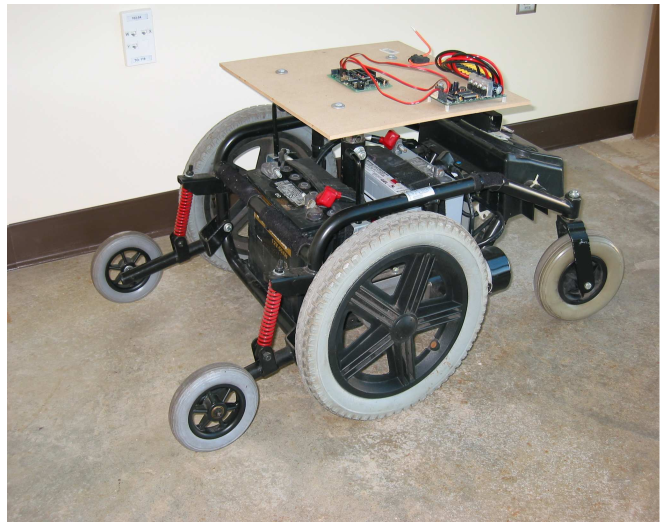

Figure 3. This is a picture of the robot chassis. The housing has been removed and the base mounting plate is shown.

The connection from the original motor controller to the wheelchair consists of a 9-pin Beau plug. We interface an IBC dual motor controller [4] through this plug. This motor controller was originally designed for BattleBot-type robots. This motor controller is designed to control two permanent magnet DC motors. The motor controller was designed to interface with the receiver from an R/C radio setup. Fortunately, the PolyBot board provides servo control outputs and three of these outputs on the PolyBot board are connected to the motor controller.

We use a 12V automotive-style relay to turn on power to the entire system. This relay is energized when the main power switch is toggled.

The main controller board is supplied by Intel Corp. and is a reference laptop motherboard design. This board features an Intel Core 2 Duo (T7400) running at 2.16 GHz. This controller board is used as the core computational unit of the robot.

Two PolyBot boards are used as secondary controller boards [5]. One of the boards is used to read sensor information. The other board is used to perform motor control.

One primary goal in the arrangement of our hardware was to have all sensor input and motor output run through a single USB connector. Our PolyBot boards use a USB-serial interface which emulates the RS-232 interface over the USB bus. The GPS receiver we use is USB-based, as is our camera. By running all of these devices into a single hub, we can easily switch out laptops for debugging.

For our vision system, we use a single color web cam. In addition to using vision, we have a number of active ranging devices. Currently we are using both sonar sensors and infrared sensors for ranging.

4. Software Architecture

This section describes the approach we take in developing a software architecture for the robot. In addition, we describe the technical details of the software design and discuss the trade-offs we consider.

The philosophy behind our software design is to implement the higher-level ‘glue’ code using Python and to implement the functionality requiring high performance in C. The Python code is used to connect a number of system-level modules in the system, such as the vision module, the PolyBot board interfaces, and the GPS module. Additionally, Python is used to provide a GUI for debugging purposes.

Writing some of the code in Python provides the advantage of simplifying the implementation due to the high-level constructs available in the language. This allows for faster modifications to the software

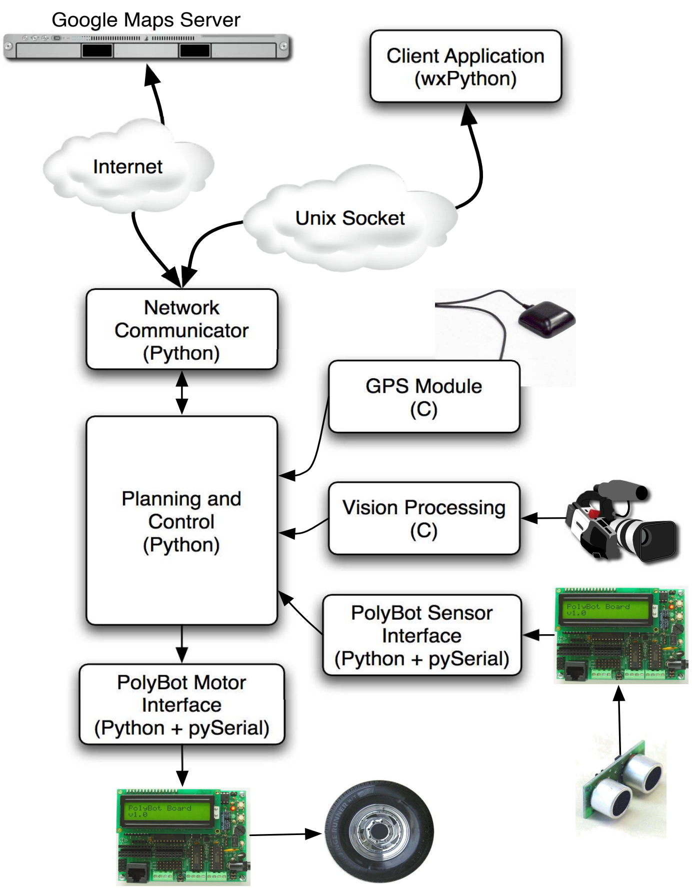

Figure 4. A block diagram of the software architecture along with the language used to implement each module. The vision code is written in C and uses the OpenCV library. Lower-level sensor interface code is also in C, while higher-level code is written in Python.

architecture without the need to worry about things such as memory management. Additionally, using multiple languages (Python and C) gives students exposure to each language and the details of interfacing cross-language code. The performance penalty of using Python versus C is minimal because most of the Python code will be interfacing with IO-bound systems and will not be computationally intensive. In comparison with other high-level scripting-type languages, Python is a natural fit due to its large system library (allowing easy access to serial interfaces and such) and its ability to interface cleanly with C.

In order to improve performance, we include the Psyco module [6]. This module dynamically compiles Python bytecode into native code. For our current implementation, the Psyco module does not seem to provide much added performance.

The vision code is written in C and uses the OpenCV library extensively [7]. We have found this library to be highly robust and practical for our purposes. We are currently testing a ground detection algorithm with our monocular vision system. Thus far, we are using algorithms similar to the work by Ulrich et al. [3] and Dahlkamp et al. [8].

The high-level control of the robot is designed to be modular to allow for the testing of a variety of path-planning and control algorithms. Currently, we are implementing a system that will perform path-planning based on GPS data until an obstacle is detected. At that point, the system will enter a reactive mode to safely navigate around the obstacle. Our current low-level control algorithm is a PID controller.

The code running on the PolyBot board is written in C and uses the PolyBot board library extensively. One PolyBot board is used to read the sonar and infrared ranging sensors. The sonar modules are connected over an I^2C bus. The IR ranging sensors are connected through an ADC. The other PolyBot board is used as an interface with our motor controller and also monitors on-board voltages.

The GUI code is written in wxPython [9]. WxPython provides cross-platform bindings to the wxWidgets [10] toolkit to Python. The GUI provides a visual display of the sensor inputs and robot outputs.

The underlying operating system that we are using on the primary controller board is Linux. Our code is cross-platform because it is written in standard C and Python.

5. Budget

In this section we discuss the budget for the project. Cost was a major concern for the project because the project was not supported by a grant. The project thus far has been funded by the university as well as corporate sponsors.

In order to minimize costs, the wheelchair was purchased used; the remainder of the parts for the robot were purchased new. In addition, we found that using components that have a built-in USB interface not only simplifies interconnection, but also reduces cost. The GPS module and 802.11b module both connect to the USB bus.

Item	Cost
Chassis	$700
Motherboard/CPU	$1000
PolyBot boards	$120
Motor controller	$250
Camera	$50
Sonar	$40
IR	$50
GPS	$75
Wireless module	$50
Total	$2335

6. Project Management

Software development is managed using the open-source program Trac [11]. Trac provides an excellent front-end to our Subversion source code repository. By using Trac, we are able to view source code changes in a graphical manner which greatly simplifies tracking modifications. In addition, Trac provides a wiki environment which is very useful for documentation of the project.

7. Experience

This project was initiated by a faculty adviser and currently involves six undergraduate students. The undergraduate students are all computer engineering majors with senior-level experience. Each student is responsible for a particular subsystem of the robot: two students work on the robot chassis, two students work on the vision system and high-level control, one student is working on the sensor data acquisition, and one student is working on a web interface to the robot.

From a technical standpoint, the project has greatly benefited the students. There are a number of computer engineering topics which the students encounter when working on the robot. In terms of software experience and exposure, the students have gained experience with developing embedded software, control algorithms, vision algorithms, and multi-language interfacing. In terms of hardware, the project has required circuit analysis, power analysis, mechanical design, and modular design.

In addition to the development of technical skills, the students also were exposed to issues faced by industrial engineering teams. The students were required to define their own software and hardware interfaces as well as document those interfaces.

Using the Trac program has greatly alleviated the challenges to managing a large software project with multiple developers. Through the wiki-based program, the students are able to share information regarding the subsystem interfaces, as well as share information via comments in the source code. This kind of flexibility and ease of information sharing has greatly benefited the project.

As this is the initial group of students working on the project, it is unknown as to how long it will take the next group of students who participate in the project to understanding the previous group's work. Previous work will be documented in the Trac program and in the source code. This project will be constantly evolving as students enter and leave the project team.

8. Future Work

There are a number of areas where our current implementation can be improved. Because this project is on-going, we intend to continually add improvements.

One current limitation of our robot is that it uses a single camera for vision. Monocular vision has its advantages such as lower computational requirements, but stereo vision can provide much more information. In the future, we would like to add stereo vision to the robot. Adding a second camera will provide the potential for disparity analysis.

The motherboard we are currently using has a dual-core CPU, but our code is not currently optimized for multiple CPUs. Currently, we plan on executing the vision algorithms on one core while executing control algorithms on the other core. In the future, it would be beneficial to properly distribute the workload across the cores after analyzing the computation requirements of each thread of execution.

One area we have not explored is the human-robot interface. Eventually, we would like the robot to interact with people and thus the robot would need to have a simple, user-friendly interface. This interface could be as simple as some LEDs and some push-buttons or as complex as voice recognition and speech generation.

9. Conclusion

This paper describes the hardware and software design of a robot along with the pedagogical experience obtained through the design project. The project was developed by six undergraduate students. The students have found the project simultaneously challenging and motivating. A strength of this project is its flexibility in providing multiple generations of students with a wide range of engineering projects.

The project goal is to develop a robot that can autonomously navigate the Cal Poly campus. The robot will do this via sonar, infrared, and vision sensors. The primary controller for the robot is an Intel Core 2 Duo motherboard which is connected to PolyBot boards. The PolyBot boards perform low-level sensor interfacing and motor control interfacing.

10. Acknowledgments

We would like to thank Intel for donating the primary controller board and Raytheon for funding the motor controller. We would also like to thank the Computer Engineering program at Cal Poly for funding the purchase of the motorized wheelchair.

References

[1] "DARPA Grand Challenge." http://www.darpa.mil/grandchallenge.

[2] "Learning Applied to Ground Robots (LAGR)." http://www.darpa.mil/IPTO/programs/lagr/index.htm.

[3] I. Ulrich and I. Nourbakhsh, "Appearance-Based Obstacle Detection with Monocular Color Vision," in *The AAAI National Conference on Artificial Intelligence*, July 2000.

[4] "IBC Dual Motor Controller Manual." http://robotcombat.com/marketplace_robowars.html.

[5] J. Seng, "PolyBot Board: a Controller Board for Robotics Applications and Education," in *The 2006 International Conference on Frontiers in Education: Computer Science and Computer Engineering*, June 2006.

[6] A. Rigo, "Representation-Based Just-In-Time Specialization and the Prototype for Python," in *Symposium on Partial Evaluation and Program Manipulation*, Aug. 2004.

[7] "Open Source Computer Vision Library." http://www.intel.com/technology/computing/opencv/index.htm.

[8] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski, "Self-supervised Monocular Road Detection in Desert Terrain," in *Symposium on Partial Evaluation and Program Manipulation*, Aug. 2004.

[9] "wxPython." http://www.wxpython.org.

[10] "wxWidgets - Cross-Platform GUI Library." http://www.wxwidgets.org.

[11] "The Trac Project." http://trac.edgewall.org/.