CS Capstone Report: PiBot

Andriy Rudyk & Tim Sizemore

12 December 2013

CONTENTS

ACRONYMS AND ABBREVIATIONS

AI	Artificial Intelligence
API	Application Programming Interface
GUI	Graphical User Interface
SSH	Secure Shell
TSR	Terminate and Stay Resident
UML	Unified Modeling Language

1

OVERVIEW

In order to demonstrate the skills obtained from individual classes within the Computer Science program, a capstone project must be undertaken by each individual so as to be deemed satisfactory in the completion of their education at the University. This project will incorporate techniques acquired throughout course work in previous classes to develop a robust piece of software that employs all the necessary attributes of a software designer.

Undergoing an independent project is an ambitious endeavor that allows the student to express their creativity and display ingenuity. It also gives them the opportunity to use lesser known resources to accomplish a self-set task. These tasks generally include web development, software engineering, networking, and third party code integration. All of which are used in industry careers depending on the field pursued. The purpose of this final report is to describe the successes and failures of just such a project. The following will discuss our particular problem as well as some background information about robotics and eventually the specific implementations we used during development.

Our initial goals and timeline: In an attempt to understand how crucial AI is, we will try to create a robot centered around a goal-based agent that uses a knowledge base to infer its partially observable environment. In order to give the project structure, we have created general deadlines for production. The measure of completion will occur at the end of each month. For September, the hardware components for the robot should be installed and a basic website is running that allows users to login and request controller accounts. For October, the website should have a more polished look and the robot will be able to accept movement commands from the web application. For November, we will finalize key features of the web application and begin AI programming. The specifics of this aspect are unclear and as of now the task environment is about the only concrete knowledge we have. And finally December, the robot and the website should be completely functional with limited components of AI potentially working.

1.1 Problem Statement

The purpose of this project is to create a robot capable of substituting itself in place of a human for a virtual tour of a designated space. The robot will be remotely controlled with the ability to transmit video to an external source, all while avoiding potential dangers while in operation.

1.2 Background

An iRobot Roomba is a product very similar to the project we undertook. That is, a robot capable of navigating a room with limited knowledge of the environment. Unlike the Roomba though, our robot does not incorporate an AI feature this semester. Research from iRobot in the field has been revolutionary for robotic and human interactions. Their R&D department also focuses on advancing robotic platforms, collaborative systems, semi-autonomous operations; the last being particularly interesting to us. The largest projects under this heading include Ava, AwareHead, and Stringray which are all being developed by iRobot. The robot "Ava is advancing self-navigating mobile robotic platforms while supporting the development of practical applications and open platform and tablet-device interfaces" (iRobot).

iRobot is known mainly for their Roomba cleaning device but it is practical products like these that provide the funds for ground-breaking projects. And with the company's purported 500 top professional developers, enough money for funding is essential. On top of Ava, iRobot is actively researching Model Transition Control, Dynamic Stability Control, Unmanned Ground Vehicles, and Dynamic Transitional Learning algorithms to name a few key areas.

1.3 Design Implementation & Testing

The testing of this project can be broken into two parts. Firstly, the robot and its various mechanics followed by software debugging. Following the definitive testing of both hardware and software components, the web application's functionality was the next step. Most testing occurred when functionality was added.

The robot is controlled using a Raspberry Pi module and can move along a four point axis meaning left, right forwards, and backwards. To test this aspect the robot was placed in a sanctioned area with minimal obstacles and then navigated the obstacles using raspberry-gpio to abstract hardware component from the written software. The robot also has a camera used to display its forward surroundings and it was tested as a separate component before adding it to the robot. This particular test involved streaming raw data to a port and then reading that raw feed and building it on the device in order to save processing time without compressing. This concludes most of the hardware and some software testing for the robot.

As for the web application, most functionality was tested as new features were added. Using the WSGI Minimalist Framework of Flask, we managed to create a sleek minimalistic website that allowed for login through OpenID and registration for a virtual tour that corresponds to an add to a MySQL database. To test all database functionality, we created a Python environment and imported the models from the database and called the appropriate add, drop, and create calls. The website was more trial and error and as with most websites, Google was our best friend. As for supported browsers, the testing portion of all code occurred on Chrome and FireFox. Other browsers should be supported without much difference in appearance as we conformed to W3C markup validation standards.

To test the GUI mock users were registered and attempt to navigate select features as well as register for a PiBot controller account and attempt to move the robot. Since a lot of the aspects had to be completed before the project could be fully tested, this procedure encompassed three major components; user registrations, basic site navigation, and remote access to the robot. While this was not idyllic, individual component issues only developed once code was introduced to the whole system. But by limiting the number of major components involved in the test, it still allowed for easier debugging. And this concludes the testing of features that occurred for this semester.

For next semester, we are hoping to delve into AI. We want to add some semi-autonomous features that will use object detection to identify potential hazards and avoid them. We also want to solidify the design of the website and implement some form of multiple connects to a single PiBot. This would allow multiple users to watch a stream while another user controls the robot. If we could restart this semester, we would do everything the same. We actually redesigned in the middle of this semester when we switched from Django to Flask and we are thrilled with what has come from the change.

2

SCHEDULE OF COMPLETION

10th September 2013

For this meeting, we had the robot completely assembled and accepting basic commands through SSH command line. These commands included forward, backward, left, and right. We were still missing a portable power source and the camera.

24th September 2013

At this point, we had the robot with a camera and a portable power source that allowed us to submit video feed and run the robot entirely through a router. We had the framework for a Django website started. We ran into a problem at this point in the project with video streaming though. The hardware on the pi limited our ability to transmit video without 2 to 3 seconds of lag.

8th October 2013

For this two week period, we focused 100% on finding a substitute for the video feed. We managed to work through several third party programs before giving up and pushing video transmission back along the timeline.

22nd October 2013

We finally managed to find a suitable video transmission program called GStreamer. It was the saving grace of a crucial aspect. Right after video, we switched to pushing key features to the website. After switching frameworks, we had to work to clean up some extra code.

5th November 2013

We managed to finish login, registration, and profile pages within the website. We integrated the MySQL database in the production server rather than using the more portable SQLite3.

19th November 2013

Revamped the look and feel of the website and added the most important page, tour registration.

3rd December 2013

After tour registration, we had to expose the features that allowed it to be used. For this, we created a tours page that generated all the tours a user was registered for and allowed a user to click the tour ID and be redirected to a page in which you could control the robot.