ARBA

Autonomous **R**obotic **B**asketball **A**gent

Rolando Desrets

August 8, 2006

University of Florida

Department of Electrical and Computer Engineering

EEL 5666

Intelligent Machines Design Laboratory

TAs:

Adam Barnett

Otto Goethals

Instructors:

Dr. A. A. Arroyo

Dr. E. M. Schwartz

Table of Contents

Abstract

The purpose of this paper is to step the reader through the inception, design, concepts, and construction of ARBA, or **A**utonomous **R**obotic **B**asketball **A**gent. ARBA will play a simple game of basketball by locating and loading balls from a hopper on a court, then finding a vertical hoop on the other end, aiming and shooting the ball through this hoop.

Executive Summary

This report will be a concise and poignant description of the design, development, construction, and implementation of a robotic Sense-React agent that has been named ARBA for **A**utonomous **R**obotic **B**asketball **A**gent. ARBA will play a simple game of basketball by locating and loading balls from a hopper on a court, then finding a vertical hoop on the other end, aiming and shooting the ball through this hoop.

In order to achieve this goal, a robotic platform was designed and constructed. This platform contains a variety of sensors: sonar range finders, infrared range finders, photo reflectors, bump switches, and a camera. Also in order to test and demonstrate the capabilities of the robot a court was constructed with a ball hopper and vertical hoop.

ARBA will start near the hoop; use a line to find its way to the ball hopper. Here it will align itself and start the picking up ping-pong balls. Each time a ball is picked up it is loaded on the shooter, the robot then takes aim and launches the ball.

Introduction

The concept for ARBA is inspired from the IEEE SoutheastCon 2007 hardware competition[1]. It is a robot that needs to "play" a simple game of basketball against an opponent. To this end I need to design a robot that can do all of the following tasks:

- It must fit in a 12" x 12" x 12" box. It can later extend in one direction an extra 6".
- It must find ping-pong balls (40mm) on a hopper and pick them up. It can also find balls on the court and pick those up.
- It must launch these balls into a horizontal hoop.
- The robot cannot interfere with its opponent.

Graduate students are not allowed to compete, therefore ARBA will not compete, and because this robot is being constructed for a grade in IMDL a great deal more emphasis will be placed on completion of the main tasks than design limitations and both with some flexibility.

The design of the robot was rather difficult and was mostly done on the fly. As each component was built, it was added to the platform. It uses sonar to find objects in front, IR sensors to find distance to object from the left side, photo reflectors to follow and find lines, and a camera to find the hoop.

The robot will run do a run through by finding the hopper and loading the balls and aiming as quickly as possible. It will turn to the right and only pick up the balls on that half of the hopper. This is done to minimize the chances of interfering with another robot during matches. A CMUcam1 is used to find the hoop once the ball is picked up, therefore there is a significant amount of effort spent interfacing this sensor.

Please visit http://plaza.ufl.edu/desrets/ for pictures and small videos of ARBA in action.

Integrated System

All the sensors are integrated through a Maveric-IIB development board[2] which uses the ATMega128 microcontroller from Atmel. All the mechanical systems were

[1] http://www.southeastcon.org/2007/.
[2] Maveric-IIB board purchased from http://www.bdmicro.com/.

designed and constructed specifically for ARBA. The servos and motors that control these mechanical systems are also controlled through the development board along with some motor driver boards[3] constructed from H-bridges. ARBA's systems are integrated with a program written in C and compiled by CodeVisionAVR[4].

Almost all the devices get power from a specially designed power board. Some devices get power from the development board. Figure 1 shows a detailed description of the integration of ARBA's systems.

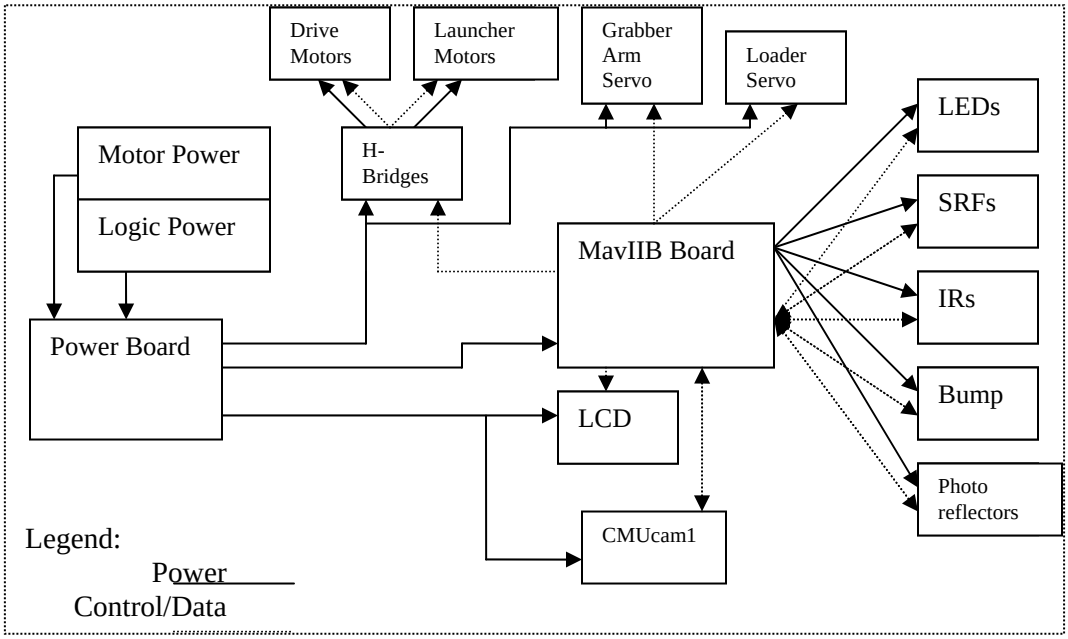

Figure 1

Mobile Platform

The main design aspect here is to be able to accommodate all the mechanical systems needed for picking up and launching a ping-pong ball, and all the sensors needed to perform the appropriate behaviors. It also has to be highly mobile within the court.

ARBA is constructed as a moderately sized mobile platform of approximately 1'x1', and the largest allowable size according to the competition rules. There are two motors with wheels that control the direction and propulsion of the robot. They are placed

[3] Motor Driver board design from http://dubel.org/motordriver/.
[4] CodeVision AVR ver.1.24.9 Evaluation ©Copyright 1998-2006 HP InfoTech.

in the near the front and pull the robot when moving forward. There is a small third caster in the rear of the robot to lift the back end of the ground and help when making turns.

The sonar sensors are placed in the front of the robot, while the IR sensors are placed on the left side. The CMUcam is mounted in the front right near the wheels. The LCD is placed on the back left. In the very middle from left to right is the arm grabber, the funnel, and the launcher/shooter.

The first iteration of the mobile platform was wholly inadequate. It was only 8”x5” and had 4” tall walls all around, this was obviously not going to work, and I moved to a flat 1’x1’ square. It became clear that if I wanted my robot to use the grabber/launcher setup I had designed, I would need a lot more space to mount these devices. The height of the ball grabber was specifically designed to reach at the height the balls are placed on the ball hopper.

Actuation

There are two gear-head 7.2VDC motors that are used for moving the robot. They are geared downed 50:1 and rated for 175rpms with a stall torque of 99oz-in and a stall current of 1.5A. These motors are plenty for my robot and can actually move it quite quickly, and they also have good response to low speeds. There are two more motors used for the ping-pong ball launcher. These are small 5VDC motors at approximately 5000rmps, and a stall current of about 1.1A. I obtained these through a surplus store and do not have more detailed information on them.

The robot has an integrated mechanical system built for the purpose of picking up ping-pong balls and loading them onto a launcher that is static on the platform. The grabber is made up of a single standard servo (42oz-in). This servo swings a curved “finger” to and away from a stationary “finger” and in this manner grabs hold onto a ball. Another servo (42oz-in) is attached as a pivot point on an arm to swing down the grabber onto the ball hopper. This is used to move the ball from the hopper onto a funnel. The funnel feeds the ball launcher. Here another servo (72oz-in) is attached to the pivot of a small loading arm that moves the ball into the spinning wheels of the launcher.

At one point there was no loading arm and the grabber/launcher system was gravity fed and able to launch the ball with no extra actuation needed. Once further

testing was preformed after the court was finalized I learned that the angle of the launcher was too steep for the ball to fall and then rise again onto the launcher. Because of this I had to build a loading arm with one of the extra servos. This solution turned out to be elegant and sleek, if not somewhat colorful (Legos where used for the arm).

Sensors

ARBA has a good sensor suite: bump, photo reflector, sonar, IR, and a camera.

Bump:

This is just a simple 3 terminal contact switch with a large arm. The switch is connected to ground and 5V. When the switch is depressed there is a shift in level. I am only using this switch as an input for the robot to start.

Photo Reflector:

These are Hamamatsu photo reflectors P5587. They are fully integrated sensors that require very little external components. Once connected correctly they provide a high voltage when reading white and a low voltage level when reading black. In order to use these sensors I designed my own board. The schematic and PCB[5] can be seen in Figure 2 and 3 respectively.

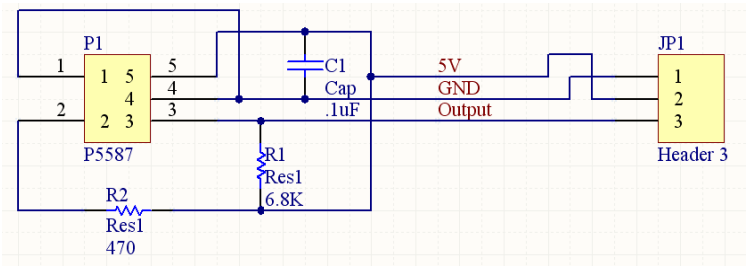

Figure 2

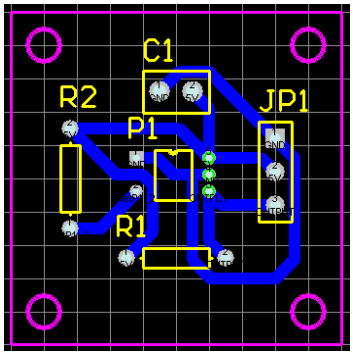

Figure 3

[5] Schematic and PCB files can be found at http://plaza.ufl.edu/desrets/resources .

Figure 3 shows the routed PCB for the schematic in Figure 2. This board is one inch by one inch. I choose to design my board this way and have multiple boards made so that I could choose to mount them in any arrangement. If I had chosen an arrangement and made the PCB to match it would have created problems as

In order for the robot to find the balls there are lines high contrast (black on white) lines painted on the court to indicate their location. A similar line is painted to lead the robot from the starting location on the court to the ball hopper. Five of these boards were constructed, with three placed in the front for following the line to the ball hopper, and two placed near the grabber for finding the lines near the balls. The arrangement of only there photo reflectors is not considered optimal for line following, however ARBA has a very limited amount of line following to do, and this arrangement is more than adequate. The two sensors used to find the ball lines are spread far apart to attempt to help straighten the robot out when looking for the balls.

This was very easy to implement in code. All that has to be done is check whether the input pins is high or low.

The photo reflectors require a very large amount of ambient light in order to work properly. It defaults to detecting black, so I am concluding that they are highly dependent on how much light white reflects. Table 1 shows some experimental results.

Photo reflector	Over Black	Over White
Output	L	H

Table 1

Sonic Range Finders:

ARBA has two sonic range finders from SRF04 and SRF05 from Devantech. These sonar sensors are very simple in premise. They use a two ultrasonic transducers and some circuitry to send out and receive an ultrasonic pulse. Using this method one can determine how far away an object is in the range of the sensor.

These sensors are used to find the distance to objects directly ahead of the robot. Because ARBA has a large berth, two sensors are used at either front corner pointed

forward. This is because the SRFs have a limited range. It is possible that with only one sensor ARBA would have a blind spot, or several.

The connection of these sensors is extremely easy. There are only 4 pins, power and ground, and output and trigger input. In order to use the sonar, a trigger has to be created on the input. The pin is held high for at least 10us then pulled low. The time between this event and when the output pin goes low is the sensor measurement.

The software algorithm to use this sensor is pretty straight forward. The trigger pin is held high for the required amount of time, and then pulled low, when this happens the timer count register value is stored. An external interrupt is enabled on falling edge connected to the output pin, whenever the output goes low, the interrupt triggers. The interrupt then reads the timer count register again and a difference is taken. This is the range detected by the sonar.

Distance /in	Left Sonar	Right Sonar
5	4640	4520
10	3470	3210
15	2390	1800

Table 2

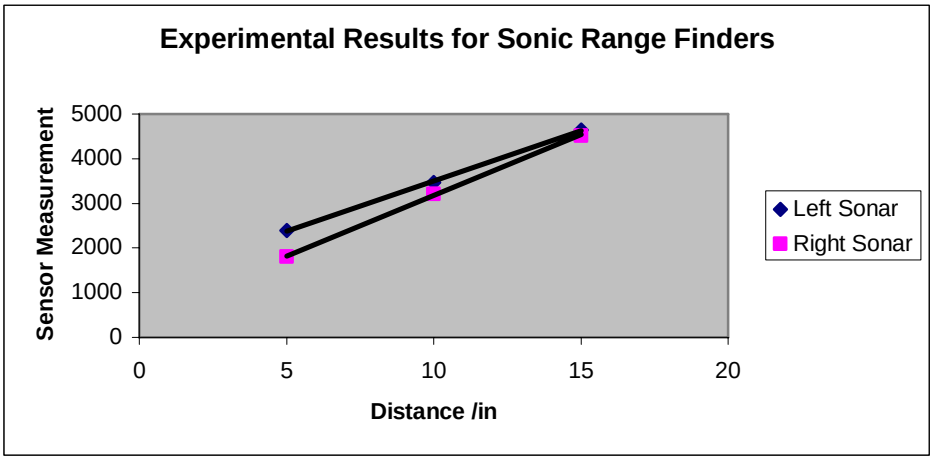

Figure 4

Table 2 and Figure 4 show some experimental results for the sonar. The values are timer count values returned by the microcontroller.

Infrared Range Finders:

ARBA uses two IR sensors GPD120 from Sharp. These sensors have all the circuitry needed along with an IR photo emitter and detector. The sensor modulates the

IR pulse and uses an internal timer to find a difference between the emitted light and the incoming light. This is then output as an analog voltage on the output pin.

These sensors are used by ARBA to follow the wall of the ball hopper. To this end, they are placed on the left side of the robot about eight inches apart. In order to maintain ARBA lined up well at a certain distance from the ball hopper the IR sensors need to be placed as far apart as possible.

Using these sensors requires only one logic pin. They have a total of three pins, power and ground of course, and an output pin which outputs an analog voltage range between about 0.3V and 2.2V. All that has to be done by the software is to use the A/D system and read the A/D value from the appropriate channel.

Table 3 shows some data for the Front and Back IR sensors. The numbers are those output from the conversions by the A/D system. Figure 5 is a graph of these data.

Distance /in	Front IR	Back IR
2	106	113
3	76	80
4	58	61
5	47	49

Table 3

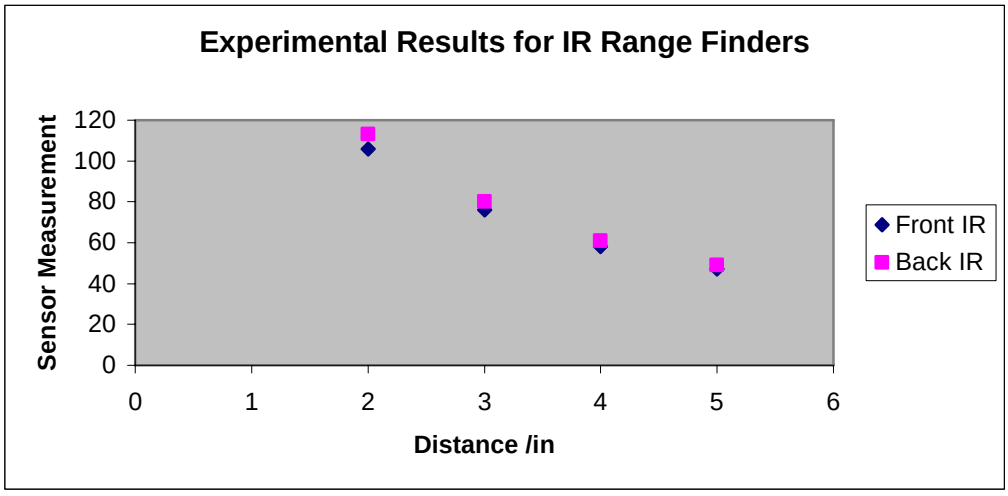

Figure 5

CMUcam1:

ARBA uses the CMUcam in order to find the vertical hoop. The camera is mounted on the front right side of the robot. The CMUcam is a highly complicated sensor

and uses a small CCD Camera with control and communication circuitry and firmware to acquire and send data.

The CMUcam uses serial communication and simple commands to: set up how images are captured, return information about what the camera sees, track a color and drive a servo, and more. The CMUcam is very simple to use, it can communicate using RS232 or non-level shifted TTL serial. The cam can be connected to the microcontroller, which can be used to send commands and receive the data. The important part of this sensor is interpreting the data, as the camera can return a great deal of data. Also this data is highly dependent on things like the ambient lighting, distance to object, and color contrast. In order to succeed using the cam the analysis of the data needs to be highly adaptive or the environment must be well constrained.

In order to use this sensor, the serial communication system has to be used on the microcontroller. The software must set up the serial communication. I created functions that send commands to the camera, and another function to receive command. ARBA is needs to know where on the horizontal axis the center of the hoop is located at any time. There for the data received from the camera is in the form of M packets which means the first number is the middle of the color mass in the horizontal axis. So the function receives the data and extracts this number. In order to have good camera tracking using TW, the robot has to find the default position for the hoop, and it does this when it is in the center of the court near the hopper, then as it moves along the hopper it finds the hoop position and determines the difference from the default position.

I preformed several experiments to find some representative data. The first experiment was very poor and the data was garbage. I tried to find the center of a 10” circle from 6 feet with no success. So, I tried to track the same circle from three feet.

The following is data received using the TW command, and moving the camera so that the circle moved around the frame.

M 73 62 44 6 80 124 143
M 0 0 0 0 0 0 0 0
M 0 0 0 0 0 0 0 0
M 6 93 2 38 12 143 104 181
M 19 77 2 1 42 143 255 188
M 25 63 2 1 67 143 255 106
M 30 59 2 1 80 143 255 68
M 26 45 2 1 80 143 208 38
M 34 57 18 1 80 143 127 29

M 47 54 35 1 80 143 94 29
M 63 62 54 3 80 143 86 47
M 72 89 2 32 80 143 81 18
M 5 111 2 71 15 143 38 77
M 6 98 2 33 20 143 119 115
M 11 80 2 1 25 143 255 163
M 11 79 2 1 26 143 255 166
M 10 80 2 1 24 143 255 16

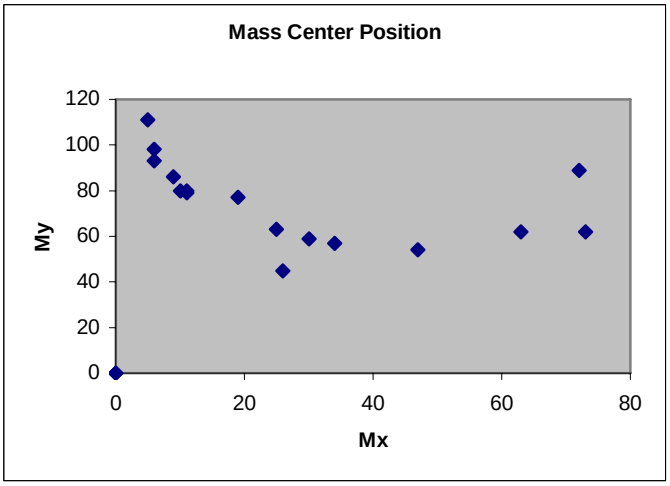

Figure 6

Figure 6 is a plot of the center position of the mass. I was trying to move the camera straight along the x-axis, I can see here that this was not perfect, and there were a couple of bad data points. Of course that is why there will be lots averaging and maybe some filtering.

From here I tired a second experiment, using the TW command; I held it close the white circle and then move it six feet away. From here I tried tracking some horizontal movement.

M 10 139 10 136 10 141 1
M 42 117 42 116 42 117 0 0
M 16 137 14 132 26 141 1 20 205
M 21 141 21 139 22 143 1 34
M 32 141 30 139 44 143 2 10
M 27 137 11 133 35 143 1
M 39 143 39 143 39 143 0 0
M 0 0 0 0 0 0 0 0
M 5 104 2 99 10 115 3 45
M 9 110 2 70 23 131 65 99 113
M 13 109 2 65 32 139 126
M 18 115 2 68 39 143 171 121
M 19 111 2 62 43 143 216 137
M 21 109 2 60 48 143 255
M 26 111 4 35 56 143 255 103
M 31 114 9 44 62 143 255 105

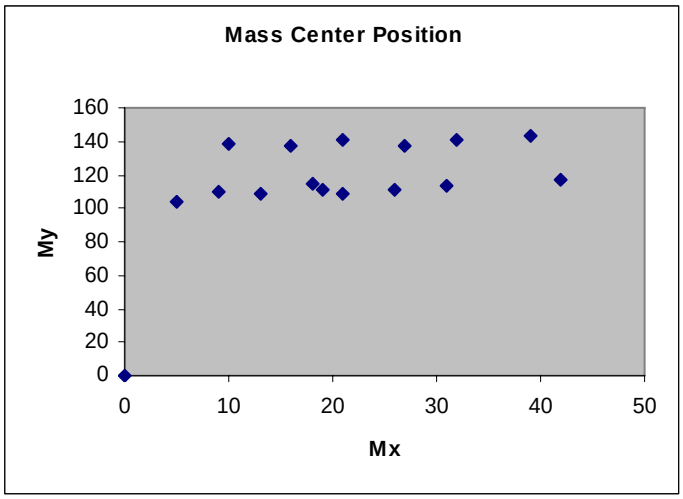

Figure 7

Figure 7 again we have a plot of the center of the mass being tracked. These results are very impressive. I did the same movement along the x-axis, panning across, while tracking the circle. The graph shows this to an outstanding degree.

All the sensors work together, and there is no conflict of any sort.

Behaviors

Line Following:

The purpose of the line following is to lead the robot to the hopper from the starting position. To this end there is high contrast black line painted on the court. Because there is so little line following needed, the software and hardware are not very complicated.

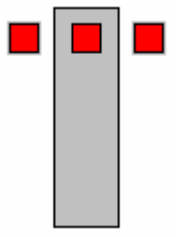

Figure 8a

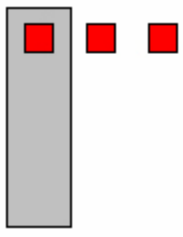

Figure 8b

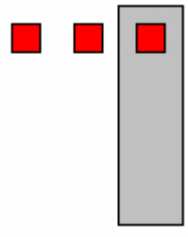

Figure 8c

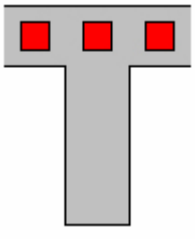

Figure 8d

The algorithm is a simple switch and case statement that adjust the movement of the robot depending on the environment reported by the photo reflectors. In the above figures the red squares are the photo reflectors and the grey rectangle is the black line. The following is a list describing the algorithm while referring to the figures for ease of comprehension.

- Figure8a – Move Forward.
- Figure8b – Turn Left.
- Figure8c – Turn Right.
- Figure8d – Turn Left (90 degrees).

There are a few, situations that are not shown but they have been programmed. If there is no line then it simply moves forward. The reason the robot turns left 90 degrees when encountering a T is because that is the direction the robot needs to travel.

Wall Following:

ARBA needs to stay within a certain distance of the wall and also move along it while remaining as parallel as possible. In order to do this the robot uses the IR sensors on the side and moves the robot forward, turning whenever there is a difference between the values of the front and back IR. The amount of turning the robot does is determined by the difference between the IR. One of the functions in the program will turn the robot while moving forward, in order to do this one motor spins a little slower than the other. This difference is obtained from the IR sensors difference scaled by a constant.

Ball Finding:

ARBA needs to be able to find the ping-pong balls accurately so that it can pick them up for launching. Again this is done with photo reflectors placed directly underneath the grabber arm. The robot moves forward using wall following, and stops whenever those sensors detect a line. Since the lines are pretty thick in width, and the grabber is not perfectly matched with the photo reflectors the robot will move slightly forward, then lower its arm and grab the ball.

Aiming:

ARBA needs to aim the launcher so that it can shoot the ball at the hoop. The camera is used to do this. First off the camera is calibrated every time the robot runs, not before hand. The robot makes it was to the center of the court near the hopper and begins tracking the hoop.

Each time a ball is picked up the horizontal coordinate of the hoop, according the tracking of the CMUcam1, is compared against the center value. This is then used to turn the robot in place. This turn in place function uses a delay to turn the robot, this delay is determined from the difference in coordinates. If the difference is positive the robot turns

left, if it is positive it turns right. The actual difference is then scaled and used as the delay. Finally, the robot is returned to its original position by turning in the opposite direction for the same amount of time.

These behaviors are integrated in a way such that the robot completes its task. The first thing the robot must make its way to the ball hopper. Line following will get it there, then the robot will turn and start wall following and finding ping-pong balls. Once a ball is found and loaded the robot aims, shoots, and continues to wall follow until the next ball is found. The process is repeated until all the balls are launched.

Experimental Layout and Results

The layout for testing my robot was provided by the rules of the IEEE hardware contest. Figure 9 and 10 shows figures taken from the rules[6] that describe the court of which I constructed half of including the hoop and ball hopper.

COURT LAYOUT – TOP VIEW

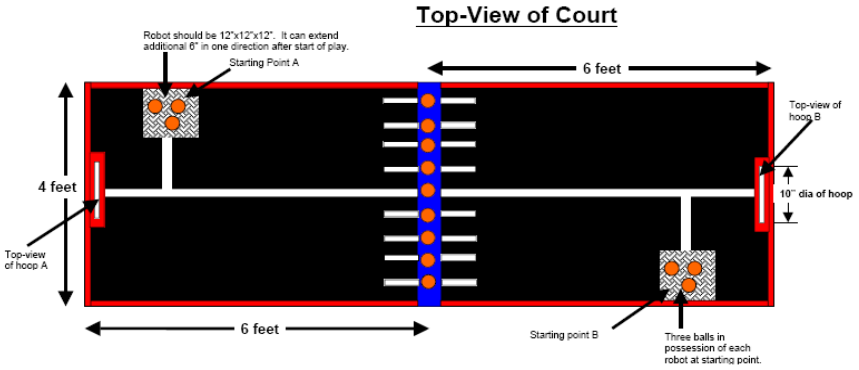

Figure A: Preliminary Top-View of Court Layout

Figure 9

COURT LAYOUT – SIDE VIEW

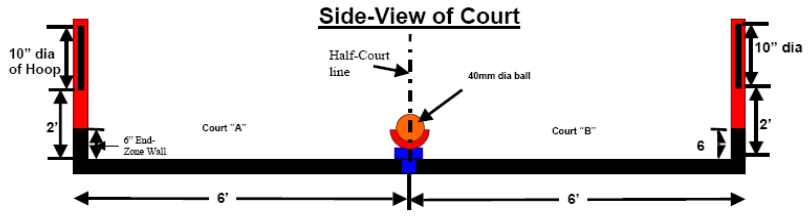

Figure B: Preliminary Side-View of Court Layout

Figure 10

[6] A copy of these rules can be found at http://plaza.ufl.edu/desrets/resources/IEEE%2520SoutheastCon%25202007%2520Preliminary%2520Hardware%2520Rules.pdf.

ARBA was placed on this court and the tested several times. After 10 trials of running the entire program and launching 4 balls every time, there was an average hit rate of 66%.

Conclusion

ARBA was taking an incredible amount of work and effort to put together, however I extremely content with the final product. After putting so much thought into something and seeing it succeed it is quite inspiring. I will endeavor to become involved in more projects of this nature. I think that ARBA is a great success as it is a rather complicated robot that successfully completed what I set it out to do. It is not perfect, and it would not harm to do some more programming work and well as platform construction/editing. All in all this a great experience and ARBA is a success.

Documentation

ATMega128 Datasheet from:

http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf.

Maveric-IIB Manual from: http://www.bdmicro.com/mavric-iib/mavric-iib.pdf.

SRF05 Datasheet can be found at:http://plaza.ufl.edu/desrets/resources/srf05tech.pdf.

Photo reflector Datasheet can be found at: http://plaza.ufl.edu/desrets/resources/P5587.pdf.

Appendices

Code can be found at http://plaza.ufl.edu/resources.