Final Documentation

ECEN 490 Robot Soccer

Team Botnet

Yazan Halawa

Adam Hastings

Josh Powell

Brian Russell

Winter 2016

Brigham Young University

Table of Contents

Executive Summary

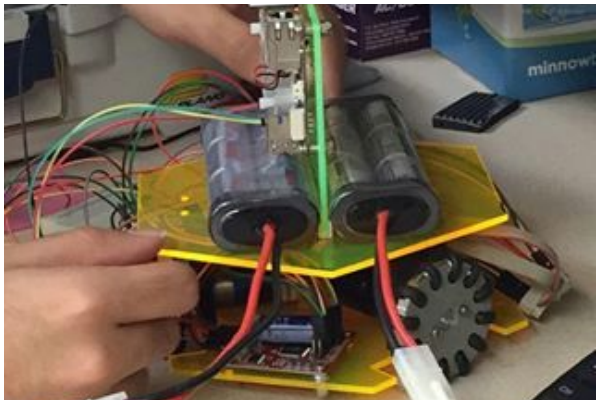
BYU Electrical Engineering Robot Soccer is a senior level team design project created to bring together many of the concepts learned throughout the ECEn undergraduate program. Students work in teams of 4 to design, build and test a fully-autonomous, soccer-playing robot.

The class is comprised of 10 teams. Using provided robotic components, each team has 3 months to prepare for the BYU 2v2 Autonomous Robotic Soccer Tournament. It is the students' responsibility to design the chassis and physical layout of their robots. Each robot must be a specific size, and must adhere to BYU competition regulations. Additionally students design a software system to detect the state of the field,and to communicate that information wirelessly to each robot. The robots must then interpret provided information to make strategic decisions. Strategic decisions are translated into observable movements using student designed control scripts.

The program involves many practice competitions, and culminates in a tournament, which is open to the public. Each student robot may function differently as intended by team strategy, however no robot can be controlled directly by a user.

Our team broke down the project into 4 major subdesign areas. These four subareas and project results will be further addressed in this document.

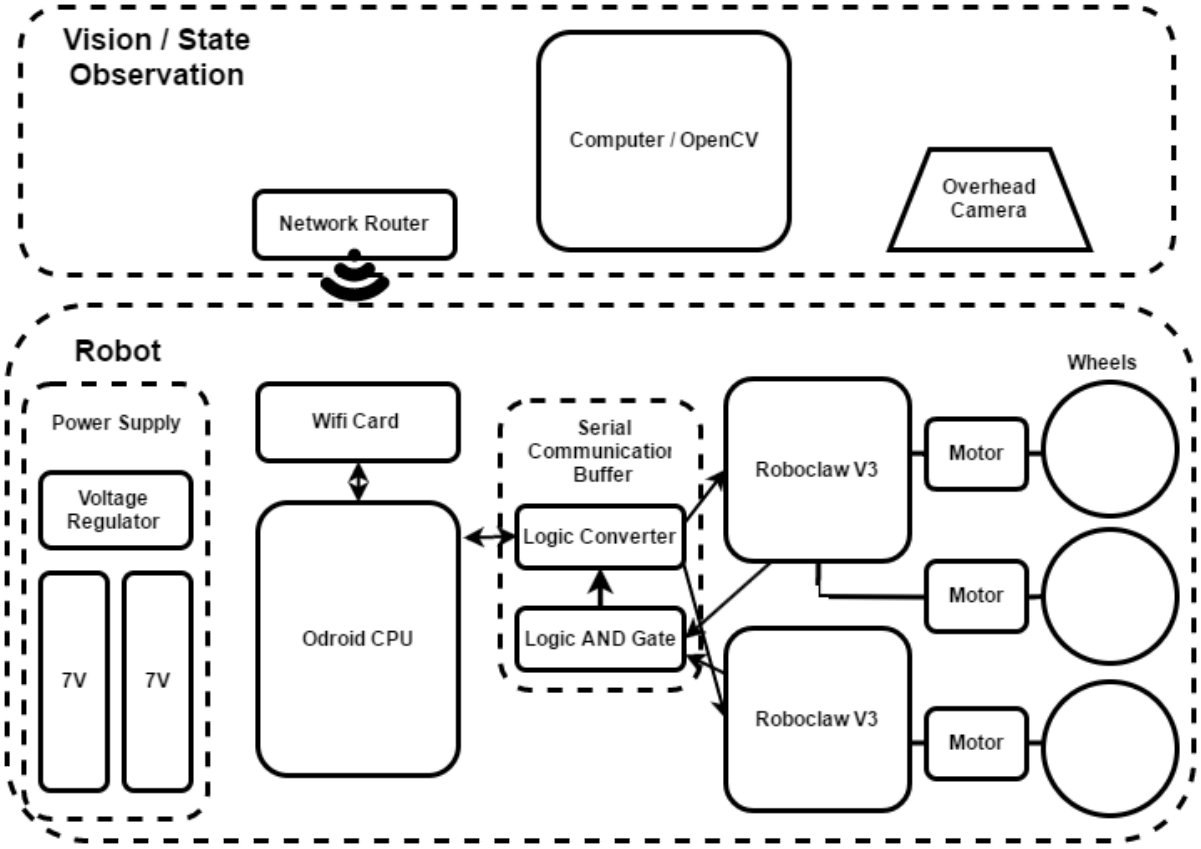
BYU Robot Soccer Competition



Example of student designed robot

Design Overview

In this project, we designed, built, and programmed a robot that plays soccer. We did accomplished this with a combination of hardware and software, which were integrated to create the complete fully-functional robot. The following chart is a high-level abstraction of the working components of our design:



Components of the Design

We decided that the best way to be competitive in this year's competition would be through the mastery of the core components of movement. Our objective was to make a robot that could move precisely where we wanted it to with speed and accuracy. We decided to focus on this goal before spending time on any additional features.

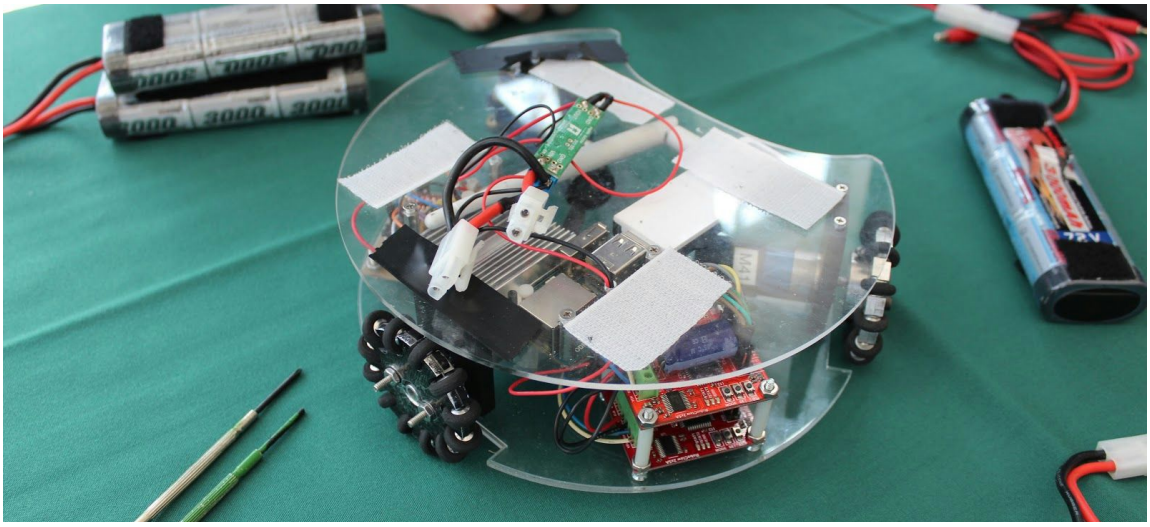
Project Requirements

At the beginning of this project, we defined a list of critical design requirements, each of which we aimed to meet by the end of the semester. The list, along with priority and parameters, can be found in the following table:

#	Customer Need	Priority	Requirement	Ideal Val
1	robot is big enough to fit all the parts, but small enough to follow the size rule	high	Diameter - Robot fits into 8 inch cylinder	8in
2	robot can move quickly in many directions	med	Time to change directions	< .8s
3	robots can respond to environment	high	Time to react to state change in game	< .5s
4	robot prevents ball from going into own goal	high	<.5 sec to respond + .5 sec to move	< 1s
5	robot able score goals on opposing goal	high	Move ball towards open goal accuracy	100 %
6	multiple robots work cooperatively	med	robots behave differently to fulfill offensive/defensive roles	
7	robots move efficiently	low	Turn and drive at the same time	
8	robots implement strategy	high	Group movements into objectives	
9	robot can kick the ball	low	Some sort of pneumatic kicker on “front” of robot	
10	battery powered	high	2x7 V batteries in series to achieve 14 V	> 20 min
11	autonomous	high	Robot requires no user control	
12	looks cool	high	Lots of robot parts showing	
13	cost	low	costs are not excessive	<$80

Major SubAssemblies

SubAssembly: Robot



Complete Robot Physical Assembly

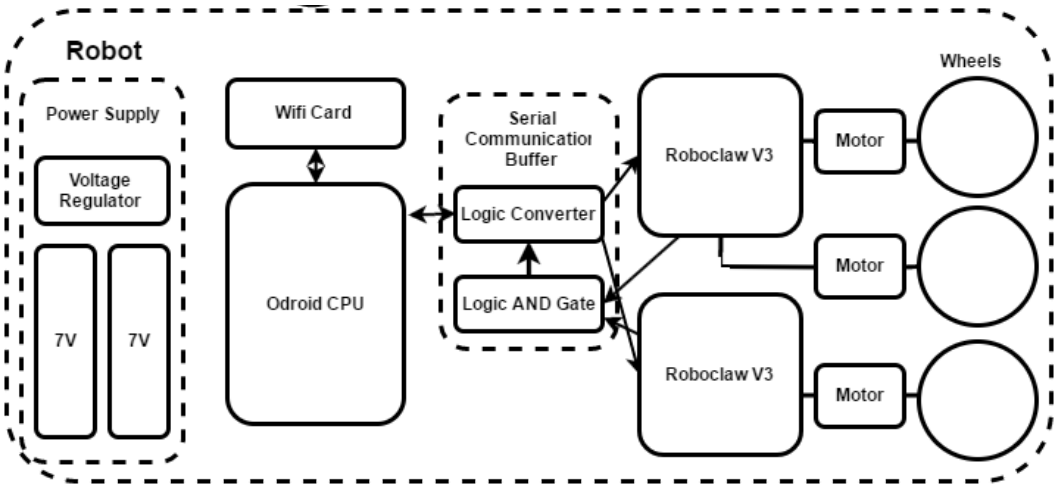
Design:

The groundwork for this project was to build a functional robot. We were given most of the hardware to make this work - namely, we used the provided Odroid, Roboclaws, motors, and wheels. While designing the robot, we had to make sure that it met certain requirements, shown below:

Requirement	Hard Specification	How Requirement Will Be Met
Cost	≤ $80	Budgeting of provided funds
Size	≤ 8 “	Adobe Illustrator + laser cutter
Battery Life	> 30 min.	Two 7.2V batteries, each at 3000mAh

Once we created initial plans for the robot, we began the assembly. This block diagram shows the connections between the different hardware components of the robot.

Connection Diagram:



The robot is powered by two 7.2V batteries in series, giving us 14.4V to work with. This voltage was given to the Roboclaws to spin the wheels. We also used the batteries to power the Odroid. However, the Odroid required 5V, so we used a voltage regulator to meet the voltage requirement for the Odroid.

The Odroid was the central processing unit of the robot. It ran an internal state machine and communicated with the vision code running on a separate machine. It also ran code that determined when and how fast to spin each wheel. It would do this by transmitting signals to the Roboclaws, which would then apply the desired voltages to the wheels. The Roboclaws would then read encoder data from the wheels, and send this data back to the Odroid to allow PID control. The Odroid and Roboclaws operated on different logic levels, so we used a 1.8V to 5V logic level converter and a logic AND gate to allow for communication.

Problems Encountered:

1. **Kicker Troubles**

One of the things we wanted to implement in our robot was a kicker. We decided to build a kicker that would be powered by compressed CO2. We actually managed to fully build a working kicker and even connect it to both the hardware and software. We we able to send a command via the Odroid, and have the kicker send the ball shooting across the field.

However, this unfortunately caused serious difficulties for the rest of the system that were difficult to find and debug. We eventually tracked down the issue to a hose within the

pressurized air system that was pressing up against one of the wheels. We believe this was causing unpredictable and erratic behavior. Once we determined that the kicker was giving us more trouble than it was worth, we decided to remove it and switch over to a new robot.

2. Roboclaws

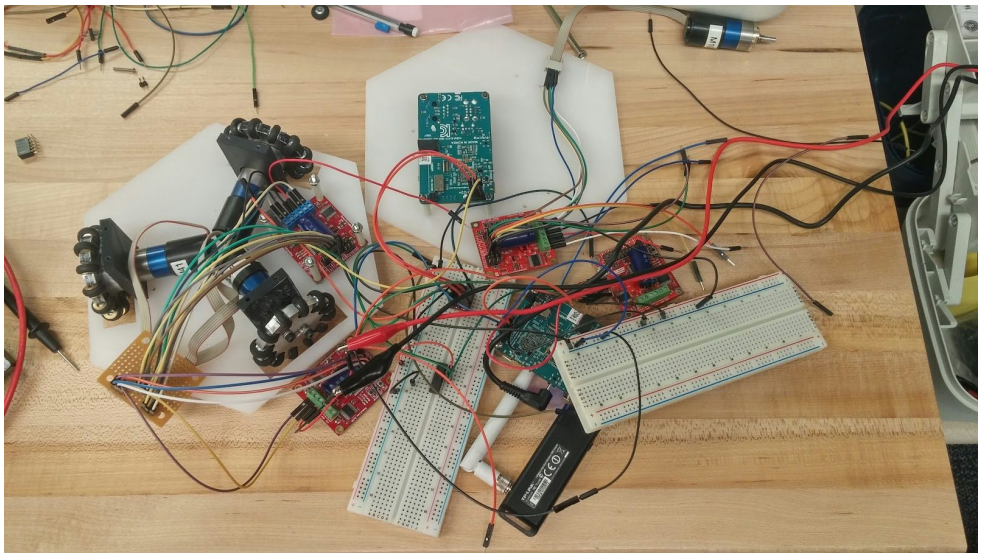
Once we decided to switch to the new robot, we thought that problems would be easy to find and fix. However, our experiences with the Roboclaws proved otherwise. We had some unreliable Roboclaws that would not give us reliable results for unknown reasons. This caused some serious bugs that were hard to track down and fix.

3. PCB

One of the things our second robot included was a printed circuit board (PCB). This board allowed for easy and sturdy connections between different parts of the robot. We found that it was causing bugs from what we determined to be bad connections. After several rounds of desoldering and resoldering, we were able to get it to work.

4. Wheels

The wheels proved to be a minor yet consistent problem throughout the semester. We had some troubles with various parts of the wheels, including keeping them attached to the motors and making repairs to the frontal plane wheels. This problem didn't cause any serious setbacks, but consistently delayed important progress.



Robot 1 during debugging session

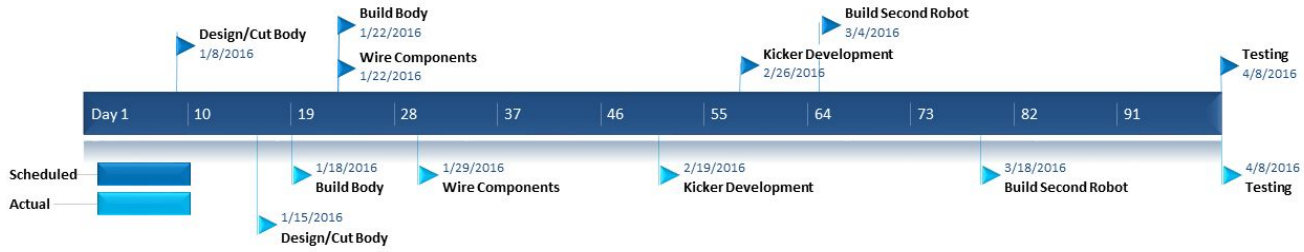
Evaluation:

The following table shows results of the mechanical system goals for our project:

Requirement as Planned	Adjustment to Original Design	Completion Status / Was Requirement Met
Cost	None	Met
Size	None	Met
Battery Life	None	Met

Requirements Planned and Executed

As the following timeline shows, we were able to meet all mechanical system goals, although some took longer than expected.



Assembly Schedule Planned and Executed

Successes and Failures:

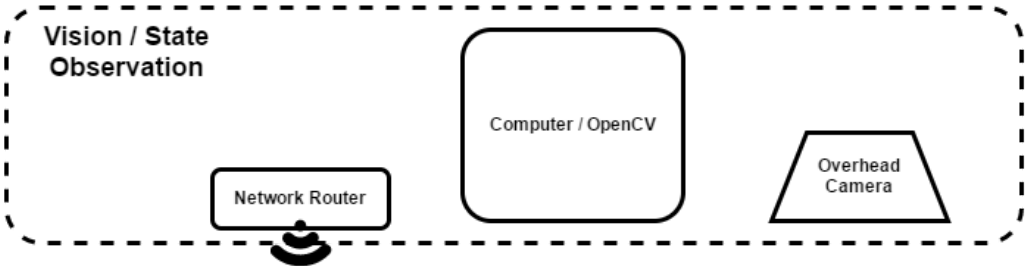
We were able to meet all requirements for cost, size, and battery life. However, other parts of the mechanical system caused problems for other parts of the project, such as motion control. Thus while we met the objectives that we set for ourselves, we had no way of knowing what kinds of other issues would come along the way throughout the semester.

Possible Future Improvements:

We ended up being satisfied with our second robot. However, given some more time, we would have liked to make certain improvements on it. First and foremost, we would have like to be able to attach a kicker to our second robot, just as we had done with the first robot. We also would have liked to include an ON/OFF switch, to make powering off the robot easier.

SubAssembly: Vision

Design:



Vision Hardware Block Diagram

An overhead camera was used to determine the state of the field in real-time. Images were served over a network address to a desktop computer. Each image was processed for needed information using the OpenCV c++ library. Useful state variables were determined and communicated to the rest of the system using ROS.

The basic implementation of the image processing software involved several steps. Images were captured from the camera, converted into a different colorspace, filtered for specific color and light ranges, and finally evaluated for play data.

Images from the camera were captured and encoded in the RGB (red green blue) color space. This is the natural state of most digital images from standard digital cameras. In general RGB images are difficult for computers to process because values change greatly with varying light. To circumvent this difficulty, each image was transformed from the RGB to the HSV (hue saturation value) color space.

The HSV colorspace offered a reliable format to extract information from the image. Hue remains constant in variable light, meaning images could be filtered for hue accurately as shadows changed and under natural or fluorescent lighting. Each image was filtered for a specific set of Hue, Saturation, and Value, that isolated elements of play, once for each robot and the ball.

The filtered images can be converted into a two dimensional binary array. The array then undergoes a stage of Erosion and Dilation. This eliminates smaller areas of pixels and reduces noise. Remaining groups of values are then evaluated based on groupings to determine robot position and direction.

Requirement	Hard Specification	How Requirement Will Be Met
Field - Center & Goals	Report Relative Positions	Measurements Based on Calibrated Center

		Position and Field Size
Robot - Position & Angle	All Reported Values Within Error % Tolerance	Contour Detection on Visual Markers
Multiple Robots	Report on both Robots and Ball at 30FPS, and Opponent at 10 FPS	Multithreading, Algorithm Efficiency

Design Constraints

Problems Encountered:

Several issues arose during the implementation phase of the project. A few of the problems dealt with Latency, Lighting, and Hue Spectrum Issues.

1. **Latency**

   Latency was an issue in two different ways. First was the issue of slow algorithms being used in the image processing phase. This was resolved by eliminating inefficient iterating loops and unnecessary displaying of images on the computer screen. Eliminating unnecessary operations was especially important because each operation was repeated over every pixel of the 480 x 840 image, at 30 FPS.

   The second aspect of latency that was never resolved was the issue of several teams accessing the same camera. Camera server access could be blocked by high traffic to the extent that the images became unusable. The closest to a resolution of this conflicting traffic was to regulate what teams could be accessing the camera during competition.
2. **Lighting**

   Images in RGB were translated into the HSV colorspace to compensate for varying lighting. Images may be thresholded in any color space, however moving shadows and lighting from fluorescents, and windows cause the image processing software to malfunction. The advantage of the HSV space is that hue does not change with shadows or other inconsistent lighting.
3. **Hue Spectrum**

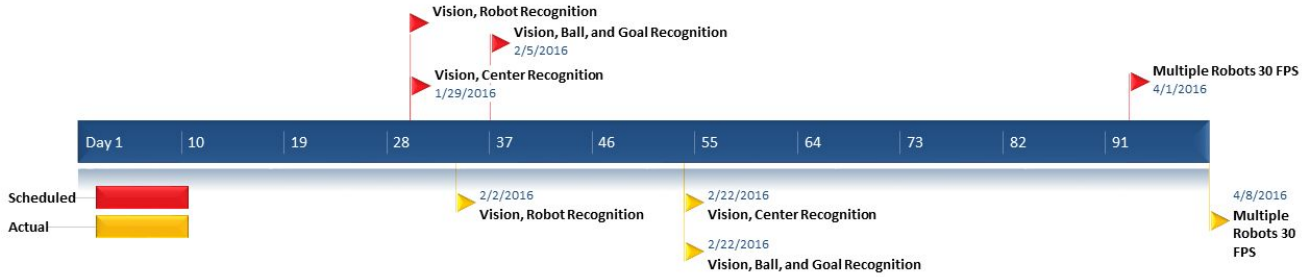
   Although the HSV colorspace proved a useful tool. It too introduced issues into the design that needed to be considered. The HSV colorspace measures each value on a range of 0 to 255. Hue is a circular colorspace, meaning both 255 and 0 are very similar shades of red. This caused erroneous behavior when filtering for red.

Evaluation:

The following were the results of our project with regards to Vision:

Requirement as Planned	Adjustment to Original Design	Completion Status / Was Requirement Met
Field - Center & Goals	No	Met
Robot - Position & Angle	Yes - Filter	Met
Multiple Robots	Yes	Adapted - Fewer Robots and Faster Speed

Requirements Planned and Executed



Vision Schedule Planned and Executed

Successes and Failures:

The main goals of the vision system were met. As shown in the timeline of predicted vs actual implementation, the actual completion of many milestones happened after the planned date. This is due to many unforeseen issues during various stages of the build process.

Possible Future Improvements:

The project could be improved by using meanshift and camshift techniques, however this requires objects to remain within view of the camera the entire time.

Additionally the project could remedy the issue of multiple users accessing the camera by changing the way the camera is accessed. Each call would pop a frame from the image stack. If several users were accessing the camera simultaneously, images could be popped from the stack before another user can access them.

SubAssembly: Gameplay

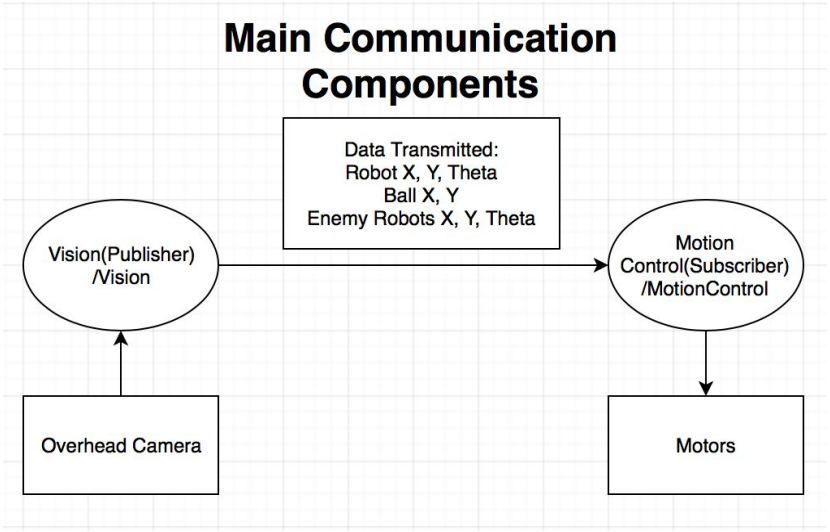
Design:

The most important aspect of Game Play is efficient communication between the various components in the system. Our plan was to design a system that would adhere to the following constraints:

Requirement	Hard Specification	How Requirement Will Be Met
Robot Can Respond To Environment	Reaction Delay < .5 s	Robust Communication Through ROS
Efficient Offense and Defense	Delay < 1s, Speed > 1 m/s	State Machine, Fast Transitions
Collision Avoidance	0 Collisions	Dijkstra, Heap Priority Queue

Design Constraints

The main tool that was used to facilitate communication between the different components of our design was the Robot Operating System (ROS). The following diagram shows how the system was set up:

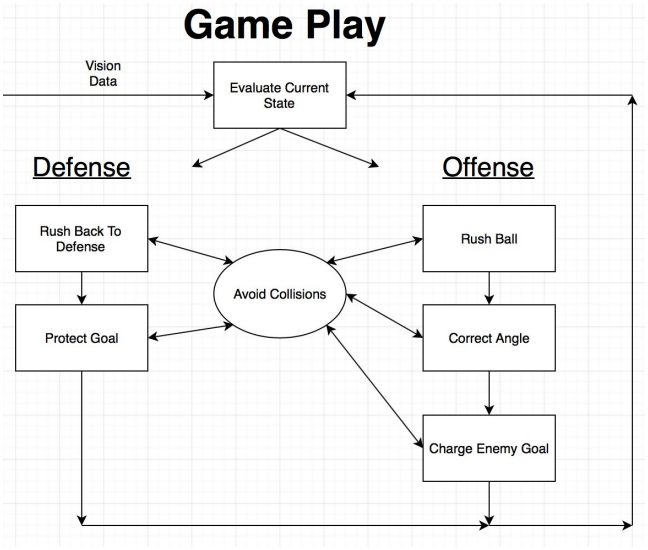


Main Communication Components

The circles in the diagram represent ROS nodes, which are ROS processes that perform some sort of computation. The way the system works is, the Camera provides information about the field, which we parse in the Vision ROS node. Next, that same node publishes the information, where the motion ROS node then subscribes to it and receives that data. Within the motion node is where all the data manipulation and AI algorithms are run, then the final motion commands are sent to the motors.

Robot Artificial Intelligence:

The actual game play of the robot was constructed using a state machine as in the diagram:



AI strategy for Game Play

Problems Encountered:

1. **Roboclaw Delay**

As part of the transition from the first robot to the second robot, we had totally new roboclaws. The mechanical setup was the same though. However we noticed that the robot would get stuck in states and would not react to the environment in time.

```
def readM2speed(addr):
        sendcommand(addr,19);
        enc = readslong();
        status = readbyte();
        crc = checksum&0x7F
#       if crc==readbyte()&0x7F:
        return enc
#       return -1;
```

Code Snippet

Through debugging, we narrowed down the problem to delay caused by reading the speed through one of the new roboclaws. For some reason, it was causing huge delay which was causing the motion ROS node to receive data at a much slower pace than that which the Vision was sending at. We realized this was caused by faulty roboclaws but we had no replacements. So to solve the problem we commented the part of the code that was checking for status while reading from the roboclaws. While this may not have solved all the issues caused by that roboclaw, it did manage to minimize the delay caused by it.

2. **Lack of Time**

With all the mechanical issues that we had to overcome, towards the end we were pressed for time. So our last tasks had to be replanned to fit the time constraint. We were planning of implementing an advanced Collision avoidance algorithm that leverages Dijkstra's algorithm to find the minimum path between two points. However, we had to simplify our plan to a basic state prediction algorithm which checks the current path and slightly alters it when a collision is expected.

3. **Camera Issues**

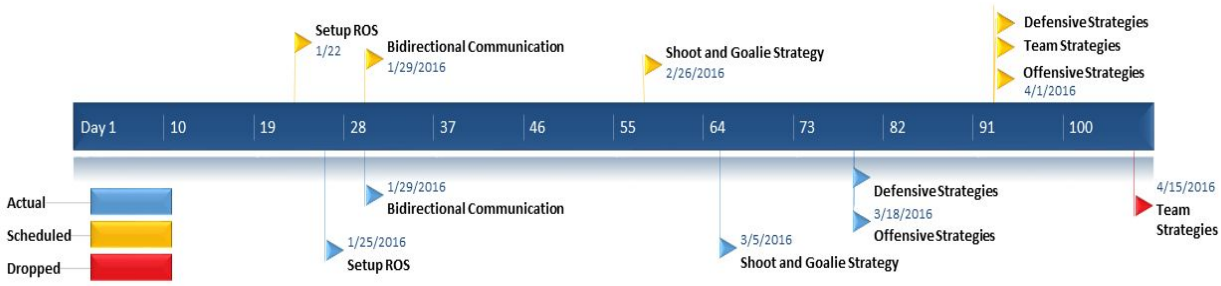
We had many instances where the camera stopped working and someone had to restart the server. Furthermore, it seemed that one camera was set up 180 degrees opposite from the other so whenever we switched to the other one it confused us and we had to flip our code. Finally, the camera would always lag whenever there are too many people connecting to it which was a vast majority of the time as all the teams worked at the same time in the lab. This set us back multiple times.

Evaluation:

The following were the results of our project with regards to Game Play:

Requirement as Planned	Adjustment to Original Design	Completion Status / Was Requirement Met
Fast Response	No	Met
Efficient Offense and Defense	No	Not Met-Speed was slow
Collision Avoidance	Yes-State Prediction	Met

Requirements Planned and Executed



Game Play Schedule Planned and Executed

Successes and Failures:

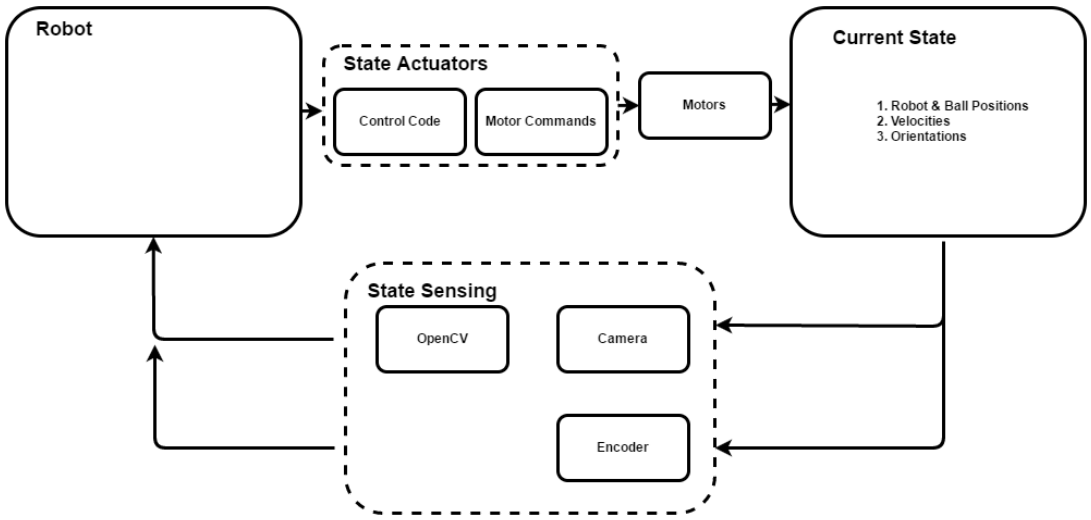
We managed to meet our requirements for time, collision avoidance, and basic game play. However, due to mechanical issues and other unforeseen circumstances, we were unable to meet the speed requirement. This lead to a huge disadvantage in the final competition. Our robot was slower than enemy robots, so we were constantly playing defense and didn't manage to take control.

Possible Future Improvements:

The one area the needed the most improvement was speed. Had we had more time, we would have made sure to optimise our code and state machine organization to focus on robust quick transitions in the game. Furthermore, we would have implemented a more advanced collision avoidance algorithm which would have minimized opponent interference to the robot.

SubAssembly: Control

Design:



Flow of Information and Commands

When designing the control of our robot, certain constraints heavily influenced our implementation. Some of these are in the following table:

Requirement	Hard Specification	How Requirement Will Be Met
Goalie Capabilities	Intercept ball path < 1 s	Appropriate Control Parameters
Precision of Movement	0.5” accuracy	Closed Loop Control

Design Constraints

The basic design for the controller was a proportional controller. The concept behind this is to correct the position and angle of the robot based on the difference between where it is and

where one would like it to be. For example if the robot were on the opposite side of the field from where it should be, one would command a fast velocity. If the opposite were true one would send a smaller velocity. As one approached the desired position, the commanded velocity slow until stopping. The same idea applies to the angle correction.

In order to know the current difference in desired and actual position and angle, the camera provides feedback which will then be used to approach the desired destination.

Problems Encountered and Solutions:

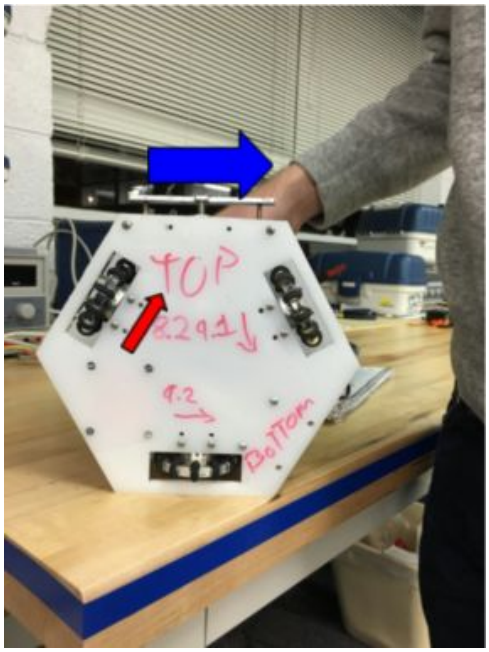


Diagram of Robot Left Movement

1. **Robot to Wheel Velocity**

As shown in the diagram above, to achieve left movement all the wheels have to spin in distinctive and coordinated directions. For each movement, calculations need to be made to know how each wheel should spin to keep desired forces and counteract undesired ones. With matrix mathematics we were able to solve for the individual velocities needed to give us our desired robot velocity.

2. **Gain Tuning**

Gains are the values that you use to control the robot's behavior based on the difference between where you are positioned and where you would like to be. Problems arose when trying to have quick response when moving towards the ball as well as fine tuning for close and precise movements (in which camera latency heavily influenced). We developed a system that used two different sets of gains depending on if we wanted quick response or fine tuning.

3. **Coupled Position and Angular Control**

Due to other problems that arose we did not have much time to solve the issue of controlling where we were facing while moving our position. A very reliable approach we found to solve this issue was to correct the angle after arriving where we desired. This allowed us to decouple the two issues and even have results when we had issues such as objects caught in the wheels.

4. **Unaccounted Forces**

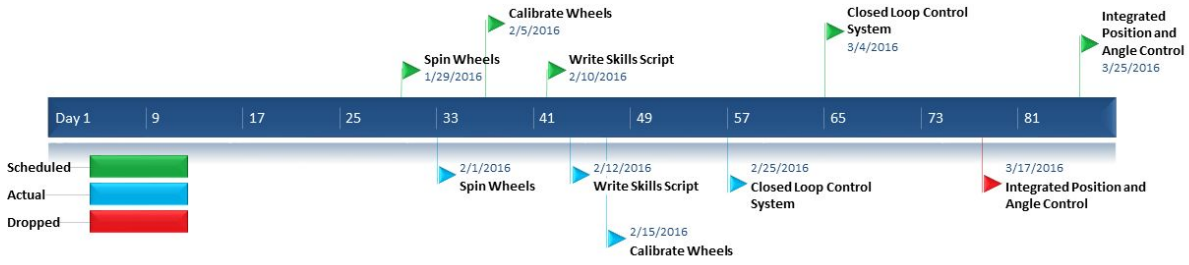
The biggest issue were unaccounted forces. From drag of the less than perfect omnidirectional wheels to objects impeding the free movement of the wheels, and all in between, there were many factors that we could not account for in our movement calculation. To solve this, we used the camera as feedback in our control model. As we moved in an undesired direction that was not expected, we were able to re-adjust ourselves to meet our needs.

Evaluation:

The following were the results of our project with regards to Control:

Requirement as Planned	Adjustment to Original Design	Completion Status / Was Requirement Met
Goalie Capabilities	Yes - Dynamic Variables	Met
Precision of Movement	No	Not Met

Requirements Planned and Executed



Control Schedule Planned and Executed

Successes and Failures:

When considering the amount of time spent debugging issues not related to controls, our robot, when tuned properly, was able to quickly move to the commanded locations. Conversely, when our robot was not tuned properly, we were still able to adequately reach within close proximity to our target.

Possible Future Improvements:

Had there been more time to work on the control, we would have first tuned our system to allow for coupled position and angular control. Next, we would have implemented an exponentially-decaying factor in our model that would not require two different sets of gains to operate correctly.

Conclusion

This project proved to be a difficult yet rewarding project. Despite the troubles that we encountered, we feel like the project has given us far more than the technical skills learned along the way. Overall, we would like to think that more went right than went wrong. We were able to accomplish a great much and we are proud of our achievements.

One of the achievements we are most proud of is that we were able to make a competitive robot in this year's tournament. We were able to meet most of the goals we set out for ourselves at the beginning of the semester, and this led to a well-designed and fully functional robot. While we had some unfortunate setbacks the day of the competition, we know that our robot was capable of much more than our placement in the tournament suggests.

Looking back, it is hard to say what we would have done differently. Many of the problems we encountered were simply obstacles that had to be overcome, or bad luck -- These weren't the kinds of problems that could have been avoided with better planning. Our initial plans and courses of action were thorough and detailed, and we are not sure how we could have better prepared for the obstacles we encountered.

In hindsight, we feel like this project has given us a "true engineering experience." It was a great privilege to work in a group with other talented engineers. We appreciate that we had the opportunity to dive into a particular area of this project and yet still learn about the other components of the project and how it all fits together. We have worked very well as a team, and have succeeded under great leadership as well. We are grateful for opportunities given to us by this project and the tremendous learning experience this has been.

Thanks for a great semester.

Yazan Halawa
Adam Hastings
Josh Powell
Brian Russel

Appendix A

Functional Specification Document

ECEN 490 Robot Soccer

Team Botnet

Yazan Halawa
Adam Hastings
Brian Russell
Josh Powell

Winter 2016

Brigham Young University

Table of Contents

Introduction

This document describes the functional specifications for The Robot Soccer Project for team Botnet. It will address the high-level implementation details of the project, the customer needs, and how they fit together. It will also analyze the level of completion expected, along with the critical assumptions and product metrics that entail this product.

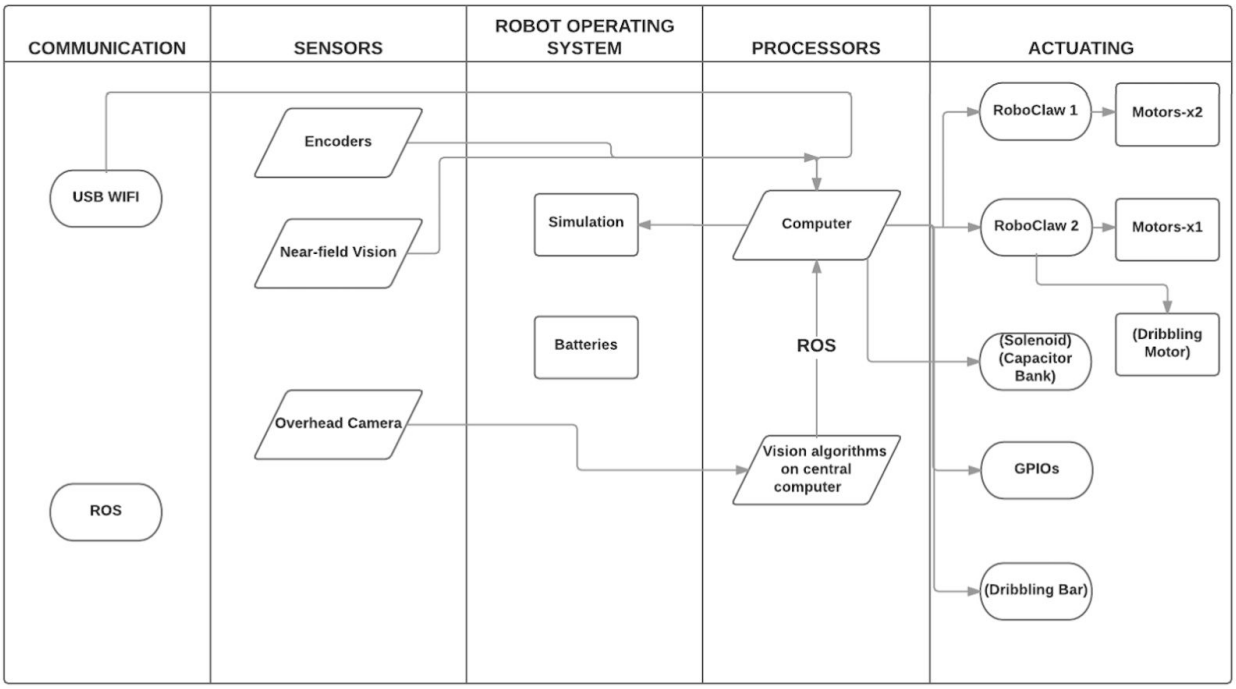
Project Description and Background

The notion of Robot Soccer started in 1997. An annual international competition “RoboCup” is held to promote robotics and AI research. Teams from all over the world participate in this competition and it has gained incredible public attention so far.

Following with the trend, Brigham Young University hosts annual Robot Soccer Competitions where teams of students participate in a two-on-two Robotic Soccer challenge. This project serves as the perfect capstone for the Electrical and Computer Engineering department where students must integrate a wireless communication system, Robotic vision, motion control, and artificial intelligence strategies in a fully autonomous system.

The system operates as follows: A single overhead camera is used to transmit crucial information to an onboard chip that processes the input and comes up with actions that the robot must perform. The onboard-chip is an ODROID running an Ubuntu linux operating system. The input to the system is the robots and ball positions. Then this information is put into an algorithm that assesses the situation using several artificial intelligence strategies and sends commands to the various motors that get the robot to move in the appropriate direction and angle.

Naturally, a lot of parts go into such a system. Here is a high-level block diagram depicting the various components that go into this system:



Project Requirements

Game Rules

- Robots must fit in a can with an 8-inch diameter and 10-inch height. All robots must be fully autonomous using on-board resources: other than start and stop directives, no information can be sent to the robots during play from human operators or from the team's base station computer.
- The ball is a standard golf ball, with the color to be determined by majority vote. We can choose from any currently available ball. (The ball will not be painted or modified.)
- The field will be five feet wide by 10 feet long. The goals will be two feet wide and specially marked with a unique color. The sides of the field will be angled so that the ball cannot get stuck against the sides or in the corner. There will be six markers at the four corners and at centerfield. The markers will be three feet high, with three one-foot color sections. The colors will be arranged so that each marker is unique.
- Each team will be assigned one of two fixed colors for each game. All players on that team must be able to wear the assigned color, and that color must be clearly visible from

all angles at player height. All players on each team will be marked in the same way, so there will be no designated goalkeeper.

- Robots must be designed and operated in such a way that they do not damage other robots, the field, or human spectators. Kickers are not allowed to shoot the ball so hard that it damages other players. Robots are to avoid collisions. Any contact with a defender while in the defense area will be deemed a violation. Outside the defense area, causing substantial contact with an opposing robot will be deemed a violation. (Contact is considered substantial if it noticeably changes a player's orientation, position, or motion.) If the responsibility for contact is not obvious play may be allowed to continue.
- The offside rule will not apply.
- It will be deemed a violation if a robot drops parts on the field.
- Robots are not allowed to fix the ball to their body, or encompass the ball in any way that prevents access by other players. 80% of the area of the ball must be outside the convex hull of the robot, when viewed from the side at a perpendicular angle.
- No robot can use adhesives such as glue or tape for purposes of controlling the ball or constructing a dribbler. It is a violation to leave residue on the ball or the field. Dribbling devices that exert backspin on the ball (to maintain contact with the robot) are permitted, provided that the spin on the ball is perpendicular to the plane of the field.

Critical Assumptions

- Additional hardware costs within budget
- All hardware will be able to communicate - WiFi, ODRIOD, camera, RoboClaw
- The battery will be able to provide the needed power/current
- Simulations will roughly approximate real life
- Processor will be able to run all of the needed software
- OpenCV will have the needed performance
- All provided hardware will function according to specs
- The motors will be able to handle the weight of the robot
- We will be able to estimate the state of the game in real time from the camera
- Controls feedback loop will run in real time

Product Requirements

Customer Needs

#	Customer Need	Priority	Requirement	Ideal Val
1	robot is big enough to fit all the parts, but small enough to follow the size rule	high	Diameter - Robot fits into 8 inch cylinder	8in

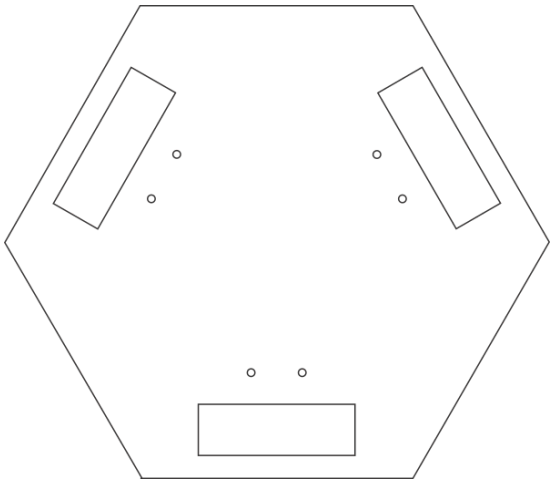
2	robot can move quickly in many directions	med	Time to change directions	< .8s
			Top Speed	> 1m/s
3	robots can respond to environment	high	Time until robot begins to react to state change of game	< .5s
4	robot prevents ball from going into own goal	high	Time to move in front of ball rolling towards goal (<.5 sec to respond + .5 sec to move)	< 1s
5	robot able score goals on opposing goal	high	Move ball towards open goal accuracy	100 %
6	multiple robots work cooperatively	med	robots behave differently to fulfill offensive/defensive roles	
7	robots move efficiently	low	Turn and drive at the same time	
8	robots implement strategy	high	Group movements into objectives	
9	robot can kick the ball	low	Some sort of pneumonic kicker on “front” of robot	
10	battery powered	high	2x7 V batteries in series to achieve 14 V	> 20 min
11	autonomous	high	Functionality determined by state of field as seen by overhead camera	
12	look cool	high	Lots of robot parts showing - Flare* LEDs Flamethrowers	
13	affordable	low	costs are not excessive	<$80

Design Specifications

Materials List

1. 3 sq. ft. acrylic plastic for body
2. 2 roboclaw boards
3. 1 ODroid board
4. 3 DC motors with encoders
5. 3 voltage dividers
6. 3 multidirectional wheels
7. 2 7.2V batteries
8. Jumper wires, solder
9. Screws

Body Diagram



Design Metrics and Need Correlation

The following table shows the linking of project requirements and specifications:

#	Metric	Min/Max	Ideal	Customer Need Addressed
1	Cost - beyond additional parts	< $80	$50	13
2	Size	<= 8”	8”	1
3	Precision of Robot Movement	< 0.5 “	0.1”	2 - 9

4	Precision of Camera detecting positions	< 1 “	0.1”	2 - 9
5	Robot movement delay	< 0.5 s	0.1 s	2
6	Camera Delay	< 0.3 s	0.05 s	2 - 9, 11
7	Response time to game state change	< 1 s	0.3 s	3 - 9, 11
8	Time to cross goal in response to ball being shot	< 0.2 s	0.1 s	4
9	Battery able to power robots for duration of time	> 20 min	30 min	10
10	Motors turn quickly with payload	> 150 RPM	200 RPM	2,4,7
11	Leaves a sparkle in Brother Clifford’s eye	> one sparkle	two sparkles	12

Metric Reasoning

1: Allowance given by program is 80 US Dollars

2: Competition rules require maximum size compliance

3: Ability to move robot to desired potion heavily influences robot performance

4: Camera provides information of ball, opposing robot, and form position feedback for team robots

5: Time between occurrence of conditions that trigger robot action and action being performed lead to game altering advantages

6: Reliability of camera information determines reliability value of feedback

7: Time between game state change and response time is affected by complexity of code and other systems leading to robot action, and need monitoring

8: Combined response and agility measurement specific to critical defensive maneuver

9: Battery duration is necessary for game play during the entire match

10: Directly determines the speed the robot may achieve

11: The only true way of measuring beauty

Summary

For this project we will have to successfully build two robots that can play soccer and win using several offensive and defensive strategies. Our main concern is to be able to meet all our customer needs in a timely and efficient manner. In addition priorities have been assigned to different needs for more efficient resource management. In order meet the customer needs, there are certain requirements that must first be reached. Some of these needs include areas such as: limitations relative to cost, space, software capabilities, and hardware compliance and other needs listed as metrics in product specifications. We are also hopeful that we will improve on the quality level of last year's robots and have a more enjoyable tournament.

Appendix B

Concept Generation and Selection Document

ECEN 490 Robot Soccer

Team Botnet

Yazan Halawa

Adam Hastings

Brian Russell

Josh Powell

Winter 2016

Brigham Young University

Table of Contents

Introduction

The goal of the Robot Soccer team project is to develop two autonomous robots that will play as a team in a robot soccer competition. This document describes the design process used by team Botnet. It will address several key design decisions and the key considerations involved in each area. It will also detail chosen design solutions.

Process Description

For each critical design area, we used our functional specifications and customer needs documentation to establish a set of criteria. We then used these criteria to evaluate various possible design specifications. By applying our criteria to each proposed design, we were able to score options and determine which would best address our needs.

Using this data, we generated a scoring matrix. By applying the matrix to a decision table, we were able to visually show which criteria should be used, what baseline it satisfies, and the pros and cons of each alternative. Using this methodology, we were able to effectively select and defend our reasons for each decision. Specific decisions and the reasons for them are detailed below.

Body of Facts

Overview of Facts and Assumptions of Competition

- Build a pair of competitive autonomous robots
- State of play will be determined by video processing from overhead camera
- Controls will run in real-time using PID loop

- Robots will not communicate directly with one another

- Soccer Field will be 10’ x 15’, with a smooth, non obstructive surface
- Robot dimensions less than 8" diameter x 10" height
- Goals 2ft wide
- Field layout similar to that of actual soccer field

- Robots will be allowed to kick the ball so that it becomes airborne.
- Control the ball without glue, tape or other adhesives.
- No human interaction is allowed after the initial setup.

- The ODROID-U3 will be able to run Linux and ROS effectively.
- All provided hardware will function according to specs
- The $50 of funding we have for the robot will be sufficient to fund the project fully.
- The power provided by the batteries is enough to run all peripheral components as well as the ODROID for the duration of an active game.

Rules:

1. Robot must fit in an 8 inch diameter can.
2. Robot must not trap the ball. Ball must be approx. 70% visible from all angles.
3. Robots may not be remote controlled.
4. Robots may not hit into or damage other robots.
5. Basic soccer gameplay will be followed

Materials (per robot):

1. 2x Nickel Cadmium Batteries 7.2V
2. 2x Roboclaw to drive the motors 5A
3. 5V voltage regulator

4. 12V voltage limiter
5. 3x motors
6. ODroid U3 Board – Ubuntu 14.04
7. 3x wheels
8. Wireless Transmitter

Basic Form:

1. 3 Wheels
2. Body of Aluminum, Plexiglass, or 3D printed
3. Must wear “jersey” for purposes of Image Processing

Movement:

1. Controlled by powerful overhead computer with low latency web cam. Image delay approx. 15ms. Position determined by overhead camera – Error introduced by angle.
2. Each wheel will have 1 motor. Two of the wheels will be controlled by the same “roboclaw”, the other wheel will have a dedicated roboclaw.
3. Use PID controller to control robot movements.
4. Using equally spaced and angled wheels robot should move in any direction .
5. Unequally spaced wheel design could make a robot move faster in one direction, but turn slower. This would lead to the robot having a front.
6. Need for a calibration script to correct for physical differences

Strategy:

1. Basic Plays – Blocking, Rushing, Matching
2. Positions – Offense / Defense

Tools:

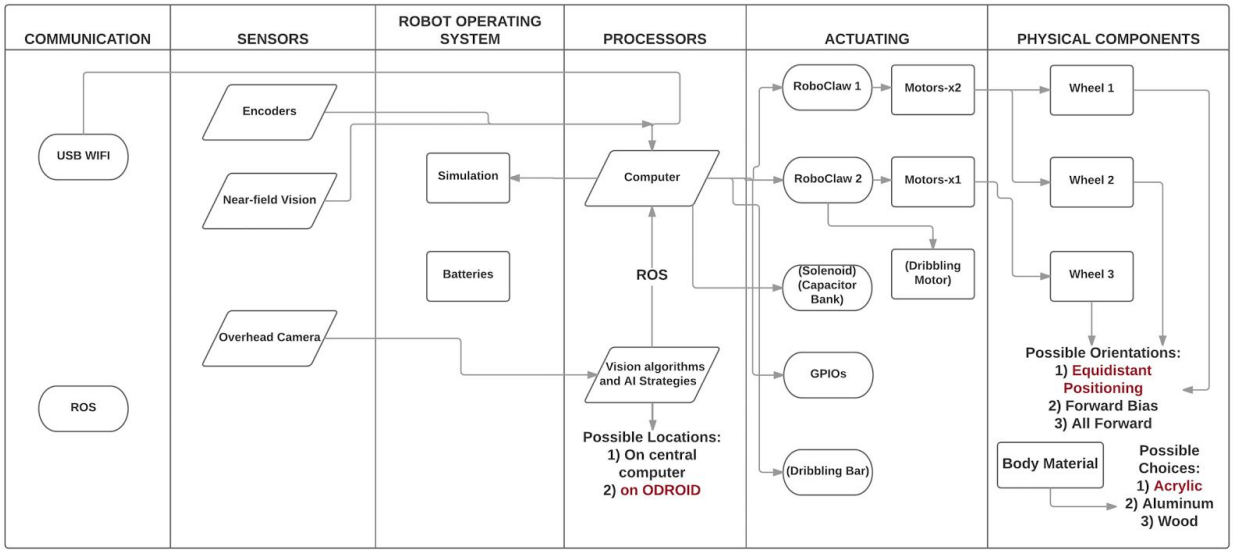
1. Simulink – Simulator
2. OpenCV
3. ROS – Robot Operating System

Necessary Assumptions to Evaluate:

Within the project requirements there are several different design implementations that must be evaluated and chosen. These elements of design will be key factors in differentiating team BotNet from other robot soccer teams. Key areas that our design will address are the following:

1. Vision Algorithm and AI Strategies Placement
2. Wheel Positioning and Alignment
3. Body Design and Material

Proposed Solutions



This block Diagram shows the various components in our system and how they interact with each other. Furthermore, it shows the various alternatives considered for the code placement, wheel orientation, and body material choice. The alternatives colored in red are the choices we settled on for our design. The analysis of those alternatives and the reasons we picked those choices is elaborated in the Critical Design Areas Section of this document.

Critical Design Areas

Vision Algorithm and AI Strategies Placement

In order to have a competitive robot, we need to have the best code organization and placement that maximizes efficiency and reduces delay.

We considered two locations where our code can be placed. Each one presents unique advantages and disadvantages:

1. We can have the code be placed on the central computer.
    - All the processing and heavy work is done on the central computer.
    - The final low level commands to the motors are communicated to the ODROID using ROS.
2. We can have the code be placed on the ODROID.
    - All the analysis and heavy work is done on the ODROID
    - The only thing that needs to be communicated from the central computer to the ODROID is the data acquired from the camera.

These are the criteria we used to judge each option:

1. Latency:

- This is the delay from the moment the camera captures the frame till the robot reacts to the current situation.
- It is weighted at 50% because this is crucial for our robot to respond to its environment and play adequately.

2. Reliability:
   - This is a measure of how our robot can handle unexpected events like the wifi signal dropping.
   - It is weighted at 30% because if such an event happens our robot has to be able to recover and not crash.
3. Organization
   - This is a measure of how efficient our code organization. The code has to be modularized with minimum number of dependencies.
   - It is weighted at 20% because while it is important for efficiency, it not as crucial to the operation of the robot as the other factors are.

Concept Scoring	Weighted Value	ODROID		Central Computer	
		Score	WT	Score	WT
Latency	50	7	350	7	350
Reliability	30	8	240	5	150
Organization	20	6	120	8	160
Total	100	710		660	

First, we scored the ODROID choice. Starting with latency, we realize there is a delay between the time the camera grabs the frame and the time we send it to the ODROID on ROS. This depends on the speed on the wifi signal and how fast we are grabbing frames from the camera. Once that is done the analysis happens really fast on

the ODROID chip and that delay is basically negligible. So we scored it at a 7. Next, we scored reliability at an 8. This is one of the huge advantages of having the analysis happen on the ODROID. If the wifi signal drops, that won't affect the robot as all the code to figure out the motor controls is on the chip itself. However, it might affect how well the robot responds to changes in its environment. Either way, it won't act unpredictably. Finally for organization, we scored it at a 6. Since the code for analysing the data from the camera and the AI algorithms are in one place, the code is not as modularized as it can be. Multiplying the scores by the individual weights and summing them up the ODROID option scored a total of 710 points.

Second, we scored the central computer choice. For latency, we scored it at a 7 as well since there is going to be delay associated with sending the data over the wifi and the speed at which the analysis happens on the computer is similar to the speed with which it would happen had it been on the ODROID, we figured they score similarly. The difference came in reliability, as we scored it at a 5. This is because we figured if all the AI strategies and analysis code happens on the central computer, then the ODROID will be blind if its connection to the computer was severed. For example, if the wifi signal falls then the ODROID has no way of recovering and it will behave unpredictably. This is why this is a bad choice in this regard. Finally, for organization, we scored it at an 8. We figured since the vision code and the AI strategies are on the computer while the motor control is on the ODROID, the code would be better modularized and the organization would be more efficient. This option scored a total of 660 points.

Finally, we compared the totals for both choices and found that the ODROID choice scored higher. The different is mostly due to the area of reliability, which is why we preferred the code be mostly done on the ODROID. We believe functionality is most important in robots and their ability to recover from failures is very crucial in a constant game of soccer. This is why we chose to go with placing the code on the ODROID.

Body Material

A critical design decision was the body material of the robot. We considered several materials for the body of the robot, including acrylic plastic, aluminum, and wood.

1. Acrylic Plastic
    - a plastic material that is reasonably strong, lightweight, and reasonably durable. Can be cut with a laser cutter
2. Aluminum
    - a metal that can be purchased in sheets. Very strong and durable. Can be cut with a metal bandsaw
3. Wood
    - Any wood-fiber based product, including hardwoods and composites. Can be cut with most saws. Lightweight and durable.

We used a concept scoring and screening matrix to determine which material to use for the body of the robot. These are the following criteria we used to evaluate the different options:

1. Strength
    - This category accounts for how well the material will hold up to the wear and tear of playing soccer.
    - It is weighted at 30% because we don't have the time or resources to constantly be repairing our robot.
2. Weight
    - This category describes the weight of the material. How much will the body shell weigh down the robot?
    - It is weighted at 40%, because the lightness will help determine the agility of our robot.

3. Workability
    - How easy is it to work with the material? Can it be cut and shaped with standard tools? How easy is it to make changes to it?
    - It is weighted at 20%, because the workability of the material will can help cut time spent constructing the body, and time is a precious resource.
4. Cost
    - How much will this material cost us?
    - It is weighted at 5%, because we have a generous but finite budget to work with.
5. Aesthetics
    - Will this material make our robot look cool? We want our robot to be visually impressive in addition to functionally impressive.
    - It is weighted at 5%, because aesthetics can improve the perception of our robot, but are also not our top priority.

Once we had identified the options and judging criteria, we created a scoring matrix:

Concept Scoring	Weighted Value	Acrylic		Aluminum		Wood	
		Score	WT	Score	WT	WT	WT
Strength	30	7	210	10	300	5	150
Weight	40	8	320	5	200	9	360
Workability	20	9	180	4	80	10	200
Cost	5	7	35	6	30	7	35
Aesthetics	5	8	40	7	35	5	25
Total	100	785		645		770	

We first scored the acrylic. Acrylic is a fairly durable plastic, so we scored its strength a 7. It is also a lightweight material, so we scored its weight an 8. Furthermore, acrylic is very easy to cut, drill, bend, and shape, and can also be laser cut. This gives it great workability, so we scored it at 9. It is cheap (~$5/sq ft), so its cost is a 7. Lastly, acrylic comes in many different colors, so its aesthetics was scored an 8. This gave acrylic a total score of 785.

Secondly we scored aluminum. Aluminum is a strong and durable metal, and will not break if stepped on, unlike acrylic or wood. We therefore determined that it was the strongest material available in our price range, so we scored it a 10. However, as a metal, it is dense and heavy (although lighter than other metals), so we scored it a 5. Furthermore, aluminum must be cut using a metal bandsaw, and can only be drilled with special drill bits, making workability low - because of this we scored it a 4. However, it is generally pretty cheap (~$3.50/sq ft) so we gave its cost a 6. Lastly, metal has an industrial aesthetics, so we scored it a 7. These scores for aluminum summed up to a final score of 645

The last material we considered was wood. Wood is fairly durable, but can degrade over time, so we scored its strength a 5. However, it is very lightweight, so its weight was given a 9. Wood is also very easy to saw, drill, and sand down, and so its workability was given the highest possible score of 10. It is fairly cheap, so its cost was given a 7. Lastly, wooden materials don’t seem to fit with the futuristic aesthetic of robotics, so its score was given a 5. This led to a combined score of 770.

The highlighted orange column identifies acrylic plastic as having the highest score, and this is the material we have chosen to construct our robot with. It is interesting to note that acrylic did not score highest for any of the most significant categories (weight > 5). In fact,The only category that acrylic won was aesthetics. Further analysis reveals that acrylic plastic scored the highest because it was the most

consistent - while it didn't win any significant field, it also didn't have any major drawbacks. These factors led us to use acrylic for the body of our robot.

Wheel Positioning

We considered three different wheel placements for our robot. trade offs of the different arrangements included ability to move in any direction, max speed in a single direction, ability to spin, and compliance with pre-existing resources.

1. Equidistant positioning
    - All three wheels are equal distances from each other. This positioning allows the most even speed in all direction, and has best spin capabilities.
2. All forward
    - All wheels are parallel to the forward direction. Allowing for the fastest speed in any single direction, but also only has the one direction of travel. Has worst spin.
3. Forward Bias
    - The wheels form an isosceles triangle shape, and the two forward wheels are closer but not yet parallel to the forward direction. This is a hybrid of the other two options.

While discussing the needs of our robot and assigning corresponding metrics, we discovered many of our top needs were directly influenced by the wheel positioning. Such needs included ability to efficiently react to the environment, prevent ball from entering goal, move quickly in a single direction, as well as in many directions. These needs were linked to metrics such as time for robot to move from one side of goal to the other and top speed. This lead to us creating a scoring and screening matrix to decide on the best positioning of our wheels.

Our matrix utilized 4 different scoring criteria. they included:

1. Maneuverability
    - The ability to move in any direction while facing any desired object.
    - We weighed it at 40%, because we determined this to be the most important criteria.
2. Max Speed
    - The fastest the robot can go in any single direction
    - It is weighed at 30%, because speed is a major factor in the game of soccer
3. Space and Mass
    - The placement of the wheels directly influences the shape and size of our robot as a whole which have their our repercussions
    - This was given 20% weight, because it is important but less important that the other categories
    - This is important but less important than other categories, so weight = 20
4. Aesthetics
    - Does our wheel positioning make our bot look sharp?
    - It is weighted at 10%, because it is the least important criteria.

Concept Scoring	Weighted Value	Equidistant Positioning		Forward Bias		All Forward	
		Score	WT	Score	WT	Score	WT
Maneuver-ability	40	10	400	5	200	2	80
Max Speed	30	4	120	8	240	10	300
Space and mass	20	6	120	7	140	5	100
Aesthetics	10	7	70	6	60	5	50

Total	100	710	640	530

In the above Matrix we had scoring concepts such as maneuverability, max speed, size implications, and aesthetics. We gave the higher values to the first two criteria.

First, we scored the equidistant positioning. We gave it a perfect 10, because it is very good at going quick in any direction. We gave it a 4 in max speed because it has the wheels spaced and angled evenly, which means for reasonable wheel posistioning it has the slowest maximum speed. It received a 6 in space because the distancing of the wheels allowed a significant amount of other components to be placed besides the wheels. Lastly we gave aesthetics a 7, because of its charming good looks. Bringing the total points to 710.

Second, we scored the forward bias positioning. We gave it a 5, because side movement and spinning slowed by close to 30% when biasing the wheels towards the forward speed positioning. We gave it a 8 in max speed because speed in the forward direction increased significantly with two wheels angled closer to being parallel with the forward direction. It received a 7 in space because the distancing of the wheels also allowed other components to be placed besides the wheels, and it gave us the option of reducing the width making it the weight less. Lastly we gave aesthetics a 6, because we liked how it would look. Bringing the total points to 640.

Third, we scored the all forward positioning. We gave it a 2, because side movement and spinning slowed by close to 45% from the equidistant positioning. We gave it a 10 in max speed because the wheels are positioned to all be parallel with the forward direction maximizing possible forward speed. It received a 5 in space because the distancing of the wheels didn't allow components to be placed besides the wheels, but it gave us the option of reducing the width making it light. Lastly we gave aesthetics a 5, because it looked okay.Bringing the total points to 530.

After completing the matrix it was apparent that the equidistant approach was the best fit. It was by far the most Maneuverable. This was important not only because it permitted easy movement to any desired position but to do so while keeping the side of the Robot with the ball kicker pointed at a desired object simultaneously. It would also have decent max speed, along with providing good spacing opportunities for the robot design and a nice look.

Other Design Areas

Robot Layout

The only constraint that we need to meet is fitting in the specified robot dimensions. As long as this metric is met, the internal organization of the robot can vary.

The design of the robot will be a “sandwich”. That is, it will be partitioned into layers that house different parts. The bottom layer will contain the motors, wheels, roboclaws, ODroid, and kicker. The top layer will house the batteries, voltage regulator, and compressed air cartridge. The layouts for the robot will be designed in Adobe Illustrator, and manufactured using a laser cutter.

Kicker Design

We have decided to use pressurized air to move our kicker. We decided that this will give us the most force per kick, and will also not drain the battery. The compressed CO_2 cartridge will be on the top layer of the robot, which will allow easy access and easy replacement of air cartridges.

Programming Languages

In choosing programming languages we gave preference to choosing a language based on the resources we had available as examples. This meant a trade off in ease of

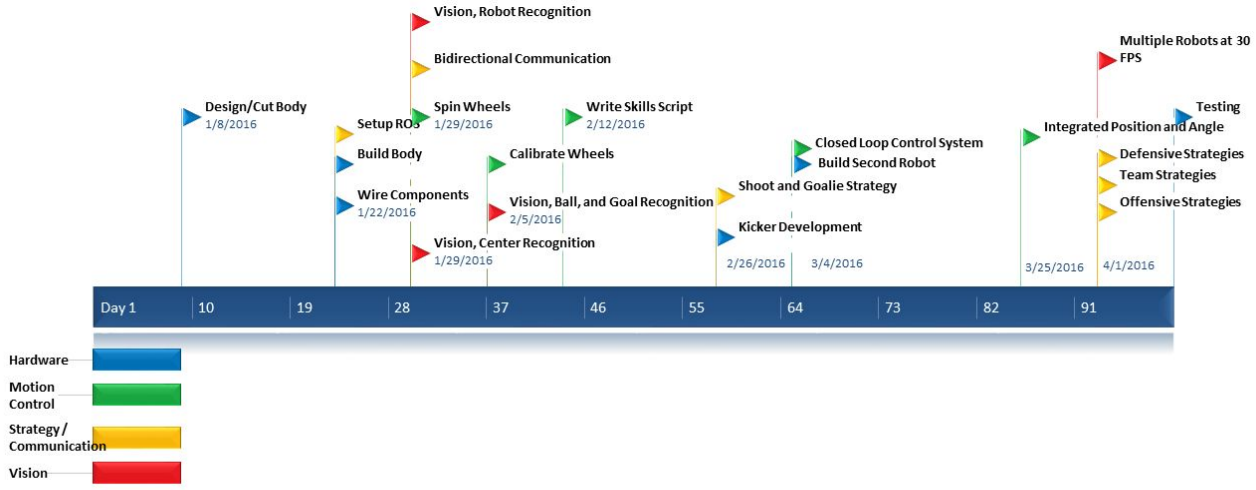
communication between different software components, and speed of code execution. As our motion control code was done in python, we had to write some of our ROS nodes in python to interface easily with the motor commands. However since our vision code was mostly done in c++, we also had to write the ROS nodes that deal with vision stuff in c++. We could have streamlined the whole code to be in one language, but that would have made it much harder to interface all the components. This is why we opted to use this hybrid model of various languages.

Conclusion

After having considered many aspects of the project , and what designs will best satisfy the customer needs, while remaining within our constraints of engineering and materials, we have made many decisions that will separate team BotNet from the competition. Our light but strong robot body, and efficient design will ensure our robots perform well under the expected play conditions. Our systematic approach to decisions regarding the design shows that we have studied the problems from many perspectives and used the gathered knowledge to create what we anticipate will be a reliable, and highly effective team of robots.

Appendix C

Project Schedule

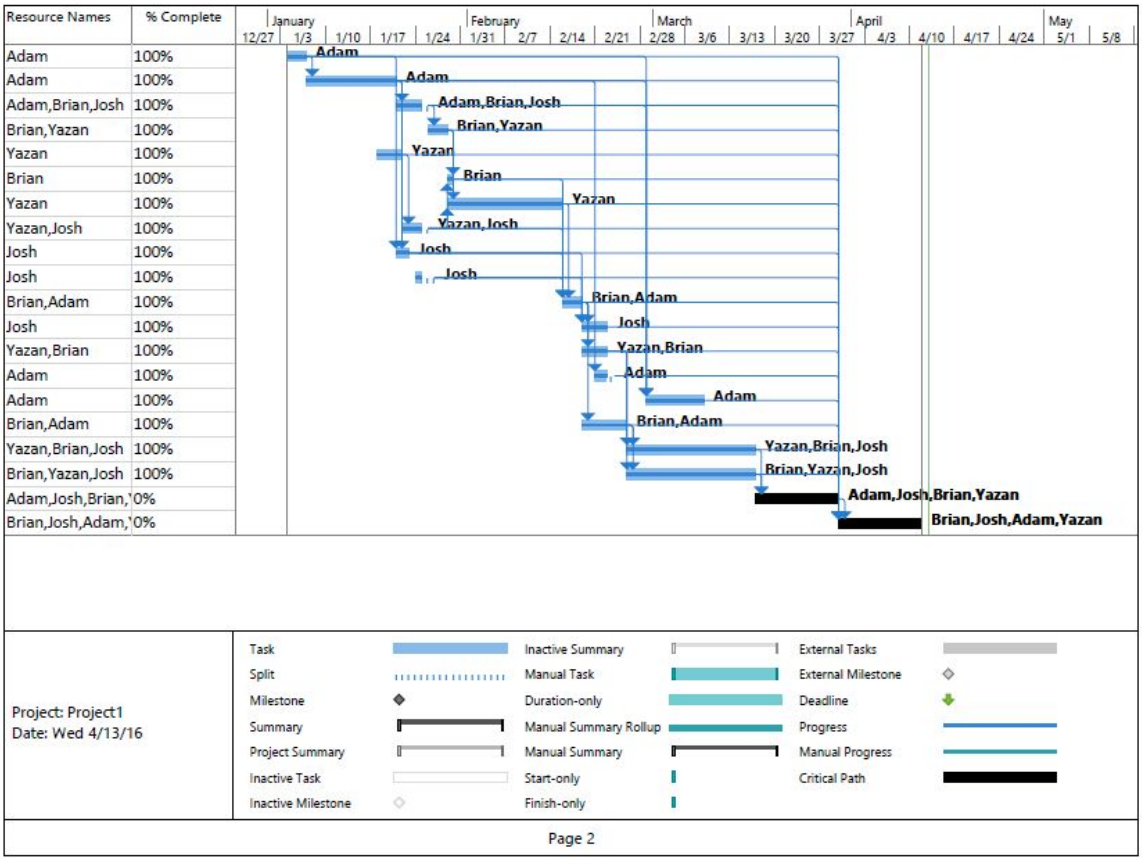


ID		Task Name	Duration	Work	Start	Finish	Predecessors
1	✓	Design/Cut Body	3 days	24 hrs	Mon 1/4/16	Wed 1/6/16	
2	✓	Build Body	10 days	80 hrs	Thu 1/7/16	Wed 1/20/16	1
3	✓	Wire Components	4 days	72 hrs	Thu 1/21/16	Mon 1/25/16	2
4	✓	Spin Wheels	3 days	48 hrs	Tue 1/26/16	Thu 1/28/16	3
5	✓	Install and Setup ROS	4 days	32 hrs	Mon 1/18/16	Thu 1/21/16	
6	✓	Calibrate Wheels	1 day	8 hrs	Fri 1/29/16	Fri 1/29/16	8,4
7	✓	Write Skills Script	12 days	96 hrs	Fri 1/29/16	Mon 2/15/16	4,8
8	✓	Bidirectional Communication	3 days	32 hrs	Fri 1/22/16	Mon 1/25/16	5
9	✓	Vision, Robot Recognition	2 days	16 hrs	Thu 1/21/16	Fri 1/22/16	2,1
10	✓	Vision, Center Recognition	3 days	8 hrs	Sun 1/24/16	Tue 1/26/16	
11	✓	Open Loop Control System	3 days	48 hrs	Tue 2/16/16	Thu 2/18/16	6,7,8,9,10
12	✓	Vision, Ball and Goal Recognition	2 days	16 hrs	Fri 2/19/16	Mon 2/22/16	9,10,11
13	✓	Shoot and Goalie Stretegy	2 days	32 hrs	Fri 2/19/16	Mon 2/22/16	11
14	✓	Kicker Development	2 days	16 hrs	Sun 2/21/16	Tue 2/23/16	2
15	✓	Build Second Robot	7 days	56 hrs	Mon 2/29/16	Tue 3/8/16	1,2,3
16	✓	Closed Loop Control System	5 days	80 hrs	Fri 2/19/16	Thu 2/25/16	11
17	✓	Defensive Strategies	14 days	336 hrs	Fri 2/26/16	Wed 3/16/16	13,16
18	✓	Offensive Strategies	14 days	336 hrs	Fri 2/26/16	Wed 3/16/16	13,16
19		Team Strategies	9 days	288 hrs	Thu 3/17/16	Tue 3/29/16	17,18
20		Testing	9 days	288 hrs	Wed 3/30/16	Mon 4/11/16	1,2,3,4,5,6,7,8,9,1

Project: Project1
Date: Wed 4/13/16

Task		Inactive Summary		External Tasks	
Split		Manual Task		External Milestone	
Milestone		Duration-only		Deadline	
Summary		Manual Summary Rollup		Progress	
Project Summary		Manual Summary		Manual Progress	
Inactive Task		Start-only		Critical Path	
Inactive Milestone		Finish-only			

Page 1



Appendix D: Vision Code

RobotSoccer/PCCode/catkin_ws/src/Vision/src/**multipleObjectTracking.cpp**

Appendix E: ROS Code

Vision Publisher

RobotSoccer/PCCode/catkin_ws/src/Vision/src/**multipleObjectTracking.cpp**

Motion Control Subscriber

RobotSoccer/ODROIDCODE/catkin_ws/src/motionControl/scripts/**skillTest1.py**

Appendix F:Control Code

Control and Strategy

RobotSoccer/ODROIDCODE/catkin_ws/src/motionControl/scripts/**skillTest1.py**

Velocity Command Functions

RobotSoccer/ODROIDCODE/catkin_ws/src/motionControl/scripts/**velchange.py**

Roboclaw Application Oriented Functions

RobotSoccer/ODROIDCODE/catkin_ws/src/motionControl/scripts/**roboclaw.py**

Frame Rotation Script

RobotSoccer/ODROIDCODE/catkin_ws/src/motionControl/scripts/**mat.py**

Resources

Documentation for Odroid

"UART Interface Using the IO-Port #1." *En:u3_ioport_uart [Odroid Wiki]*. N.p., n.d. Web. 18 Apr. 2016.

Documentation for Roboclaws

(c) 2014, 2015 Ion Motion Control. All Rights ReservedRoboClaw Series Brushed DC Motor Controllers (n.d.): n. pag. Web.

OpenCV Website

"OpenCV | OpenCV." *OpenCV | OpenCV*. N.p., n.d. Web. 18 Apr. 2016.

Class Website

"Classes - Beard." *Robot_soccer:2016:start*. N.p., n.d. Web. 18 Apr. 2016.

ROS Tutorials

"Wiki." *ROS/Tutorials/InstallingandConfiguringROSEnvironment*. N.p., n.d. Web. 18 Apr. 2016.