Robot Soccer

Concept Generation and Selection Document

Team Slash-Dash-Bang-Hash (/-!#)

Skyler Tolman, Mckay Seipel, Andrew Nuttall, Ben Simpson

Brigham Young University

**Table of Contents**

[**Introduction**](#_lqbjv1cpxxq7) **3**

[Project Overview](#_h5400irg5wof) 3

[Purpose](#_cw8nm9z32bks) 3

[**Body of Facts**](#_jiuq5l1xphd6) **4**

[Facts](#_tcen432b5mxr) 4

[Critical Assumptions](#_98nlbpy48k1h) 4

[**System Architecture**](#_b1jdybpsgr5l) **5**

[**Critical Design Areas**](#_37gw2nufw8ls) **7**

[Roles for Two-Robot Play](#_v3e7vhqn07g4) 7

[Concept Definitions](#_sovk77m2fonh) 7

[Criteria Definitions](#_to9eq18fdrqn) 7

[Decision Matrix](#_yj56qqbxnj78) 8

[Explanation of Results](#_nyhunklga976) 8

[Kicker Enhancements](#_8gdxshhazr8k) 9

[Concept Definition](#_pu6ynjiibcip) 9

[Criteria Definitions](#_hg6r3ilkaguh) 9

[Decision Matrix](#_wpjm784eya01) 10

[Explanation of Results](#_qukyobybg31c) 10

[**Summary**](#_uyzds03d05jt) **11**

# 

# 

# Introduction

### Project Overview

This is the fourth consecutive year that robot soccer has been done as a senior project at BYU. As a centerpiece of the ECEn on Display event, its primary purpose is to drum up excitement for and interest in the Electrical and Computer Engineering department. However, recent entries have often fallen short of this goal because they do not perform at a high level. Thus, the primary objective of our project is to create a product that will be entertaining to watch.

### Purpose

This document contains an analysis of our concept generation and selection process for the BYU 2017 Robot Soccer competition. Our purpose in this document is to explain the define the system architecture for our project and to select critical design areas on which we can focus to set us apart from the competition.

We will list our current understanding of facts and assumption then outline our system architecture. As part of the system architecture outline, we will identify a couple critical areas for which we wanted to brainstorm more ideas. The specific process of concept generation and concept selection for each of those areas will then be explained.

# Body of Facts

### Facts

* Final competition is tentatively planned for April 18.
* We are using a Raspberry Pi 3 as our controller on each robot.
* We will be using ROS and OpenCV for controlling the robot.
* The competition will be a socer game between two teams of two robots each.
* There will be no offsides rule.
* The soccer ball is a golf ball.
* Video data will be gathered by an overhead camera.
* Vision processing is done on a host computer and processed information is passed to the robots.
* All other computation is done on the robot.
* Most hardware on our robots and other teams’ robots will consist of hardware provided by the ECEn shop.
* Robot must fit into an 8” diameter by 10” tall cylindrical can.
* It is a violation to drop parts on the playing field.

### Critical Assumptions

* Outputs to robots are PWM commands to motors and similar commands to any additional hardware that we add.
* Collisions are illegal
* The field is designed so that the ball cannot get stuck along the walls
* Vision processing will be the most computationally intensive bottleneck
* Motion control will be comparatively difficult to implement
* This simulation has good fidelity relative to the actual operation of the robots

# 

# System Architecture

Figure 1. Block Diagram of System

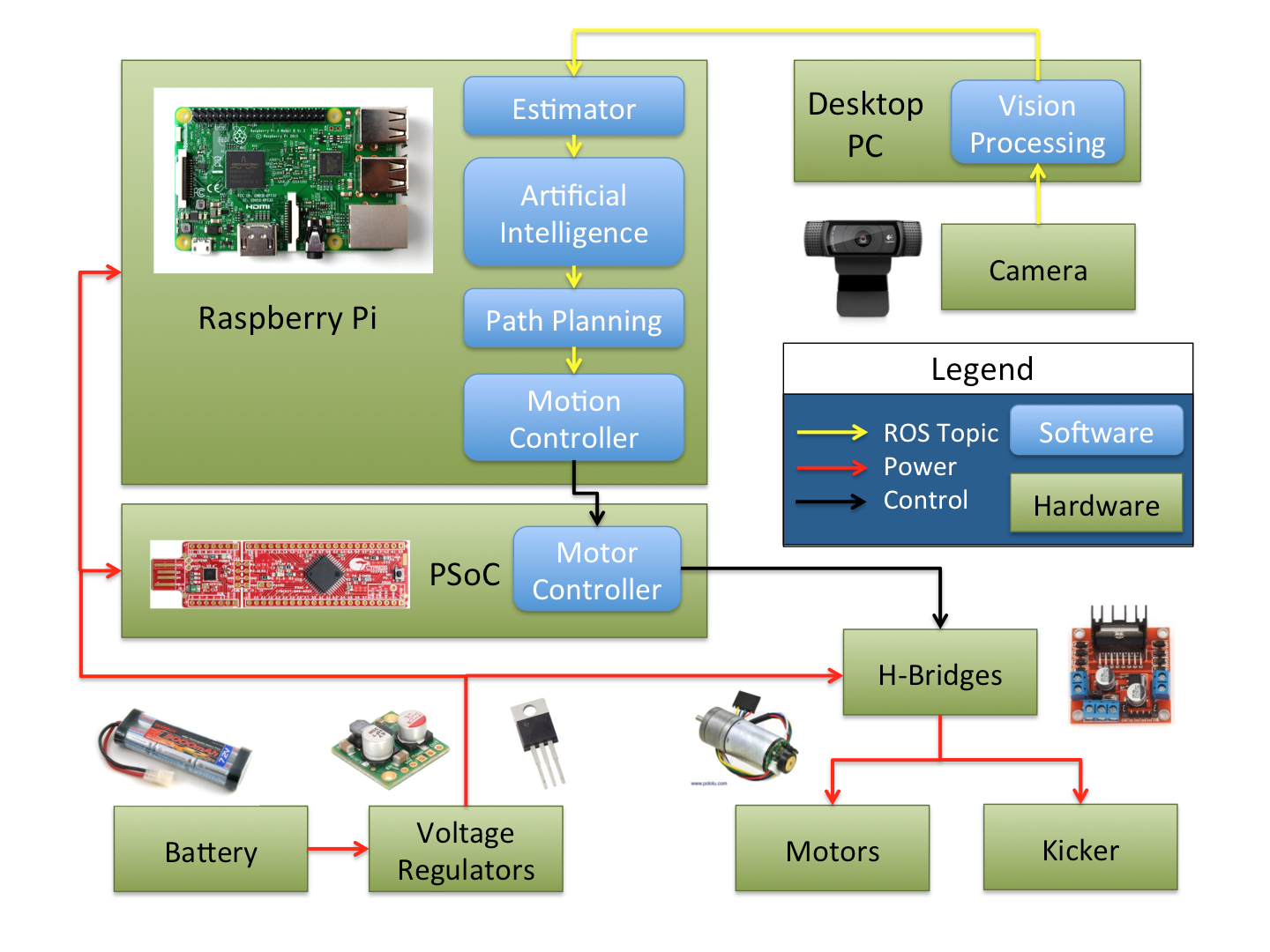


Figure 1 shows the architecture of our robot soccer system. As shown in the Legend, hardware components are represented as green blocks, software components are shown as blue blocks within the hardware components that run them, and connections are shown as arrows. The main sensory input to our system comes from the field’s overhead camera. Data from the camera are sent to a desktop computer for vision processing. Once the image data are processed, information about the current state of the game is sent to the Raspberry Pi computers controlling each robot.

The game state information is first run through an estimator, which evaluates the new information with respect to previous game states and extrapolates the data in order to form a better picture of the current and future events in the game. This estimation is used by an artificial intelligence algorithm to decide on strategies to use (offensive, defensive) and plays to make (passing, dribbling, screening, etc.). Once a play has been selected, the path planning algorithm creates waypoints for each robot to follow in order to complete it, and the motion controller translates these into desired velocities and sends the information to the PSoC.

The PSoC runs a control algorithm to break down the motion controller’s commands into commands for each individual motor and the kicker. The H-bridge provides the necessary amplification to power each output based on command signals.

This architecture uses Robot Operating System (ROS) to pass data between the camera, desktop PC, and different software components on the Raspberry Pi’s. The Raspberry Pi passes information to the PSoC through a serial connection, and the PSoC send PWM command signals to the outputs. Power for all components on the robot comes from onboard batteries.

# 

# Critical Design Areas

There were many areas that will need attention, but in seeking to identify some areas in which we could stand out from other teams, we decided to go through the full formal concept generation and decision process for two main categories, namely roles used for two robot play and kicker enhancements.

## Roles for Two-Robot Play

### Concept Definitions

* 1. Static Independent Roles
     1. This refers to an approach in which there is a designated offense robot and a designated defense robot. The two robots may attempt to guess what they think or hope the other robot is doing, but there is no formal coordination, communication, or consensus. This is the most basic approach.
  2. Dynamic Independent Roles
     1. This is similar to static independent roles, but the two robots may interchangeably assume either defense or offense.
  3. Static Coordinated Roles
     1. In this approach, the robots remain with a designated role (offense or defense), but they can communicate and coordinate with the other robot to make coordinated plays.
  4. Dynamic Coordinated Roles
     1. In this approach, both robots can assume either role and they can communicate, pass, and coordinate with each other.

### Criteria Definitions

* 1. **Play effectiveness** - this is a measure of how effective the plays can be against any given opponent using a specific approach to two robot roles. We weighted this as 40/100, because it is the most important thing to the success of our robot.
  2. **Resistance to malfunction** - this refers to how robust the approach would be to a malfunction of one of the robots. There is a possibility that one of our robots will stop working mid-game and we want our design to be able to handle that possibility as best as possible. We gave this a 10/100, because there are other ways to ensure a robust design.
  3. **Implementation simplicity** - this is a measure of the time and energy that would be required to implement a given approach and to make it work effectively. We gave this a 25/100, because we don’t want to spend too much time working on it, but we are willing to spend time to succeed on the project.
  4. **Computational overhead** - this is a measure of how much processing time and power would be consumed by a given approach. We also gave this a 25/100 because there is a threshold below which increases in overhead do not cause problems, but once that threshold is exceeded, it becomes a large problem.

### Decision Matrix

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Roles for Two-Robot Play** | | **Concepts** | | | | | | | |
| **Static Independent** | | **Dynamic**  **Independent** | | **Static**  **Coordinated** | | **Dynamic**  **Coordinated** | |
| **Criteria** | **Weight** | Rating | Weight | Rating | Weight | Rating | Weight | Rating | Weight |
| Play Effectiveness | 40 | 5 | 200 | 7 | 280 | 7 | 280 | 9 | 360 |
| Resistance to malfunction | 10 | 5 | 50 | 7 | 70 | 3 | 30 | 4 | 40 |
| Implementation Simplicity | 25 | 5 | 125 | 3 | 75 | 4 | 100 | 3 | 75 |
| Computational Overhead | 25 | 5 | 125 | 4 | 100 | 6 | 150 | 4 | 100 |
| **Totals** | **100** |  | **500** |  | **525** |  | **560** |  | **575** |

### Explanation of Results

Our baseline was static independent robots, so we gave it a 5 in every category.

The next option was dynamic independent. The robots would have more options on attack and defense, so it would be more effective. Thus, we gave it a 7 in effectiveness. That flexibility would make our robots more resistant to malfunction, because they can adapt. Thus, we gave it a 7 in resistance to malfunction as well. That flexibility adds complexity, however, so we gave it a 3 in simplicity and a 4 in computational overhead.

The next option was a static coordinated approach. Because the robots would have more extensive communication, they would be more effective, so we gave them a 7 in that category. However, because they are reliant on each other to play positions and also for a portion of the computation, it is very vulnerable to malfunction. We gave it a 3 in that category. It would be only slightly more complicated than the static independent option, because we have to make coordinated plays, so we gave it a 4 in implementation simplicity. Because the robots are not repeating work, it is a more effective use of computation, so we gave it a 6 in that category.

The main benefit of implementing our AI using this approach is the effectiveness of the plays that are made possible. It also provides quite a bit of flexibility and modularity in our coding architecture, which is why we gave it a 9 for play effectiveness. This approach also scored a 4 in resistance to malfunction, but it was not the lowest. The main reason we gave this approach a lower score than the independent role approaches is because there is a possibility that one of the robots will break during the game. In order to implement a coordinated role approach, we would need a single AI commanding both robots, but if the AI that hosts the robot malfunctions, then we would run into problems. We can mitigate this problem by having redundant nodes in each robot, but it is still more of a concern when dealing with coordinated roles. We also had lower scores in the “implementation simplicity” and “computational overhead” criteria because the approach does imply a more complicated system, but the increase in complexity was small enough compared to the play effectiveness advantage, so this option ended up being the most desirable approach. For these reasons, we gave it a 3 in simplicity and a 4 in computational overhead.

For these reasons, we decided that the dynamic coordinated roles approach would be the best option.

## Kicker Enhancements

### Concept Definition

* 1. Basic Solenoid Design - This is the baseline, represented by a single solenoid attached to a kicker.
  2. Kickers on 3 sides - This is putting a solenoid kicker on three of the six sides of our robot.
  3. Vertical Kicking - This is designing a kicker that will be able to kick the ball in the air on command. It would also require that we are able to kick the ball along the ground, as well.
  4. Double Kickers - This is having two kickers on one side of the robot in order to have more control over the direction of the kick.
  5. Putting spin on the ball - This is making a kicker that would be able to put spin on the ball so that it would be able to move around other robots.

### Criteria Definitions

* 1. Simplicity - We need to make sure that it is feasible for us to implement Before we implement any strategies using the kicker, we will need to make sure that the basic functionality of our robot is well optimized. For that reason, kicker optimization will likely be very constrained by time, so we will need something that we can implement simply. We weighted this as a 40/100, because none of us have any experience designing something like this, so we would not be able to make something complicated.
  2. Effectiveness - If the work that we put into the kicker does not help us to win games and entertain audiences, then it is not worth doing. We rated this a 35/100, because anything we do needs to be effective.
  3. Cost - We need to keep cost at a reasonable level. We gave this a 25/100, because it is less important than the other two things.

### Decision Matrix

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Kicker** | | **Concepts** | | | | | | | | | |
| **Basic Solenoid**  **Design** | | **Kickers on 3 sides** | | **Vertical Kicking** | | **Double Kickers  (left and right)** | | **Putting spin on the ball** | |
| **Criteria** | **Weight** | Rating | Weight | Rating | Weight | Rating | Weight | Rating | Weight | Rating | Weight |
| Simplicity  (Build & use) | 40 | 5 | 200 | 5 | 200 | 2 | 80 | 3 | 120 | 4 | 160 |
| Effectiveness (Shoot/Pass) | 35 | 5 | 175 | 6 | 210 | 8 | 280 | 7 | 245 | 8 | 280 |
| Cost | 25 | 5 | 125 | 2 | 50 | 4 | 100 | 3 | 75 | 4 | 100 |
| **Totals** | **100** |  | **500** |  | **460** |  | **460** |  | **440** |  | **540** |

### Explanation of Results

Because the solenoid was our baseline, we were comparing all of the other options to that one, and gave it 5’s in every category. For the kickers on three sides, we decided that, while having three kickers is more complicated from a hardware perspective, it simplifies the AI problems by allowing us to kick in more circumstances. Thus, we gave it a 5 in simplicity. However, we also determined that this approach would not be much more effective as long as our robot motion is well implemented, but it would be a little more effective. Thus, we gave it a 6 in effectiveness. It is also three times as expensive, because it takes three kickers instead of one, so we gave it a 2 for cost.

Because the vertical kicker would need to be able to kick along the ground as well, it would require a mechanism that would be prohibitively complicated. Thus, we gave it a 2 for simplicity. It would be effective, because other robots would be unable to interfere with the path of the ball, but it would also be useful in limited circumstances. For those reasons, we gave it an 8 in effectiveness. Cost-wise, it would require more hardware, but not three times as much. Thus, we gave it a 4.

The double kickers would require one extra kicker, so it is more expensive than the baseline, but less expensive than having 3 kickers, so we gave it a 3 in cost. Writing software to use it effectively would be somewhat complicated meriting a 3 in simplicity, but the control that we would get would offset some of that added complexity, which gives it a 7 in effectiveness.

The option that we decided was the best was trying to put spin on the ball. That would be extremely effective, because it would both allow us to keep control of the ball while dribbling and move the ball around obstacles when kicking. Thus, we gave it an 8 in effectiveness. Because it would require minimal extra hardware, this would be less complicated than the other options that we decided, but figuring out how to use it in software adds some complexity, so we gave it a 4 in simplicity. Because the extra hardware is minimal, the cost increase would not be very significant, so we gave it a 4 in cost.

# 

# Summary

In this document we have presented an overview of our system and identified some key aspects of the project that we feel as a team will help us to distinguish ourselves from our competitors and best meet the customer needs. We decided to generate and identify meaningful concepts for our two-player AI roles as well as kicker enhancements. We decided to use a dynamic coordinated approach to AI roles and attempt to create a kicker which can add spin on the ball. We hope that these decisions will help us to best create a system which will fulfill the needs of outperforming the other teams, provide entertainment to spectators, while still following the rules of the game.