# Real Autobots and Decepticons

# Project Plan

# Version 1.0

Prepared by

**Carter Chase**

**Trevor Sietz**

**Ben Smith**

**Jeff Ross**

Prepared for

# The Analogical Constructivism and Reasoning Lab

# Table of Contents

Introduction 4

Development Phases 4

Subphases 5

Project Development Summary 6

Preliminary Phase 6

*Project Proposal* 6

*Familiarity with Simulator* 6

Feasibility Phase 6

*Choose Robot Prototype* 6

*Research Arduino-Computer Communication* 7

*Research Simulator-Computer Communication* 7

*Develop Preliminary Robot Model* 8

Communication Phase 9

*Implement Arduino-Computer Communication* 9

*Implement Simulator-Computer Communication* 9

*Update Simulator Model* 9

*Test Sensor Reliability* 9

Movement Phase 9

*Assemble Arena* 9

*Develop Basic Robot Internals Library* 10

*Move Robots from Controller* 10

*Finalize Sensors* 10

*Interpret Sensor Information in Controller* 10

*Develop World Model* 10

*Finalize AI to Controller API* 10

AI Phase 10

*Research AI Planning Algorithms* 10

*Develop AI Planning Algorithm* 10

*Enhance Robot Internals Library* 10

Enhancement Phase 10

*Enhance Algorithm* 10

*Expand Formation Number* 10

*Enhance Robot Sensors* 10

Publicity Phase 10

*Team Poster at Research Expo* 11

*Department Poster* 11

*Demonstrations* 11

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Changes** | **Authors** | **Date** |
| 1 | Rough Draft | Carter | 09/27/2016 |
| 2 | Updated feasibility phase | Carter | 10/04/2016 |
| 3 | Updated Communication Phase | Carter | 10/25/2016 |

# Project Document

# Introduction

The goal of our project is to achieve the unaided assembly of modular robots into a predetermined shape from random starting points. In simple terms, we want to be able to toss a bunch of modular robots onto a clean floor, have the robots find each other, and have them arrange into a chosen shape (like a plus or a square).

This document outlines the project plan for the self-reconfiguring robot research project conducted by senior students from the IPFW Department of Computer Science. The purpose of this document is to:

* Identify the phases of the project development
* Define the purpose of each phase
* Detail results of each phase
* Document any roadblocks and solutions

## Development Phases

The project was completed in 7 nonconsecutive phases.

* **Preliminary Phase**: Become familiar with the problem and gather requirements.
* **Feasibility Phase**: Determine what is and is not possible.
* **Communication Phase**: Implement the program layer for communicating with the robots.
* **Movement Phase**: Implement the program layer for moving the robots.
* **AI Phase**: Implement the AI.
* **Enhancement Phase**: Enhance the features implemented in the previous phases.
* **Publicity Phase**: Publicize results.

Figure 1. Development Phases

## Subphases

Each phase consists of several subphases whose timeline is detailed below.

Figure 2. Subphases

# Project Development Summary

Each of the 7 phases has a specific purpose, with an end result of achieving our project goal. This section gives detailed descriptions of each phase and documents any major discoveries from the phases.

TEAM: FOLLOW THIS TEMPLATE

1. Focus on what we’ve already done
2. Why we did this phase
3. What we did
4. What we learned
5. What troubles we encountered
6. What was the end result

## Preliminary Phase

The goal of the preliminary phase was to determine the basic requirements of the project and to familiarize ourselves with the available technology. It consisted of two subphases that are detailed below.

### *Project Proposal*

Completed by: Carter, Ben, Trevor, Jeff

The project proposal was a document required by the computer science department. The proposal that was submitted was an extension of the original proposal created by Dr. Licato. The proposal was extended by gathering new requirements and narrowing the project scope. A project goal was developed and agreed upon by the team. The phase finished with a presentation of the completed proposal to a group of faculty and fellow CS students.

### *Familiarity with Simulator*

Completed by: Ben, Jeff

Access was gained to a powerful 3D simulator called Webots. This simulator could be used to interact with virtual models of the robots. Time was spent familiarizing the team with the software’s capabilities. The phase was completed by successfully importing the robot’s model and interacting with it.

## Feasibility Phase

The goal of the feasibility phase was to research and test what is and isn’t possible. The team emerged from this phase with functional prototypes and a reasonable understanding about how approaches to future phases. This phase was divided into four subphases which are described below.

### *Choose Robot Prototype*

Completed by: Carter, Ben, Trevor, Jeff

Before beginning any implementation, a robot type needed to be determined. By the end of the summer it was believed that the robot had already been finalized. The team was under the impression that the engineers were going to complete a prototype called the SMORES robot. Unfortunately, the engineers believed that the team had already agreed on their “block bots.” After several meetings with the engineering team, the miscommunication was recognized. A team meeting was held and a robot type was determined. The team proposed two robots, a “dream” robot and a “backup” robot.

* Dream: The engineers finish the SMORES robots.
* Backup: Roombas are purchased and configured by the team.

A SMORES prototype was promised by mid-October. The prototype would be evaluated and future prototypes built. Trust in the engineering team was not very strong, and a final prototype will be needed by January. It was decided to keep options open in case they didn’t come through, but development would continue assuming that the robots are completed on time.

### *Research Arduino-Computer Communication*

Completed by: Trevor

The first step in communicating with the Arduinos was to determine what type of connections it used. It was expected that the connection would be over Bluetooth using the Pybluez library to manage the Bluetooth connection. However, the Arduinos communicate using a subset of Bluetooth called BLE. From the program’s perspective, BLE behaves like a regular serial communication port connection. As a result, the project required a serial com library and not a Bluetooth library at all. Several serial communication ports libraries were investigated, however PySerial was the only one that was actively updated and supported. With that information, the project was limited to using PySerial to communicate with the Arduinos.

Originally, it was decided that multithreading would be the best way to communicate with the robots. This would allow for multiple comport to be read concurrently. There was one problem with multithreading serial communication ports, PySerial was not thread safe. According to its documentation, PySerial is currently under development to be thread safe and multithreading is actively discouraged. There was an alternative to multithreading: multiprocessing. Multiprocessing allows for multiple processes to be ran concurrently the only issue is the processes can share memory space making the program potentially unstable. The stability risk makes it advised to switch to switch to multithreading once PySerial becomes thread safe.

To allow communication between processes, it was determined that the python queue data structure contains the necessary concurrency protections. As a result, all communication between processes will be accomplished using the python queue. The final communication architecture can be seen in figure 3.

### *Research Simulator-Computer Communication*

Completed by: Ben

Originally, it was determined that communication with the simulator should be *identical* to communication with a physical Arduino. Communication with the Arduino involved serial communication ports, therefore research was conducted into how to use serial communication ports with the simulator

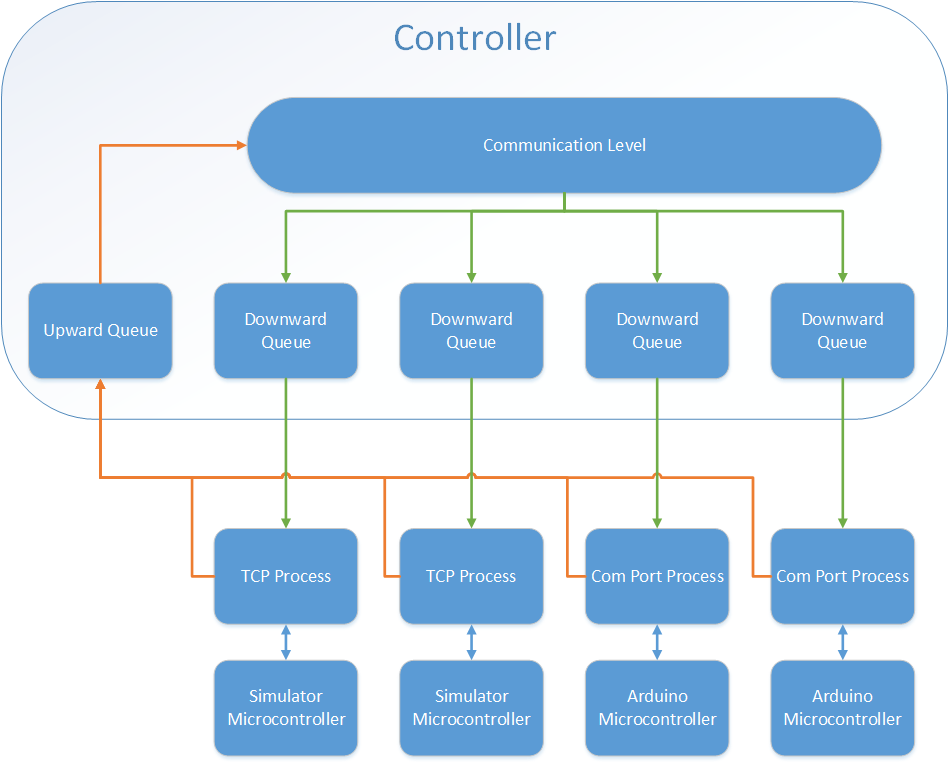
Continuous difficulties arose when using comports with the simulator because of some limitations of the simulator. In order to use serial communication ports in python it was necessary to import the PySerial module. Unfortunately, the simulator can’t import modules that are not already a part of the programming languages’ default libraries. The end result is a program the runs just fine outside of Webots, but crashes once loaded into a virtual robot.

Figure . Architecture for communication between the controller and the simulator/robots

After continued discussion it was agreed that another attempt should be made to use serial communication ports, this time in the C language. Even that proved futile, research revealed that recent updates in Windows 10 broke much of the support for serial communication ports. After this revelation, it was decided to abandon serial communication ports for the simulator altogether. A new approach was proposed using TCP ports in place of the serial communication ports. The architecture is shown in Figure 3.

### *Develop Preliminary Robot Model*

Completed By: Jeff

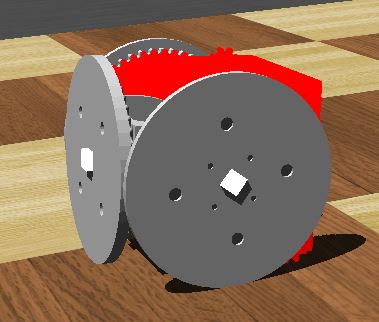
The engineering team provided a primitive model of the SMORES robot. Their model was accurate regarding the look of the robot and included some basic functionality such as wheel motors. Unfortunately, the model lacked several important devices, such as sensors and connectors, that would be needed for the simulation.

Figure . Preliminary simulator SMORES model

After replacing the missing components, an additional problem was apparent. The geometry of the model (which was also used for collision detection) dragged as the robot moved. This caused the robot to catch on the ground and veer off course. The simple solution was the creation of a simpler collision box.

The preliminary robot model can be seen in Figure 4.

## Communication Phase

The goal of the communication phase was to implement the network level of the architecture produced in the previous phase. We hoped to emerge from this phase with functional code and a standardized API between the network level and the movement level.

### *Implement Arduino-Computer Communication*

### *Implement Simulator-Computer Communication*

### *Update Simulator Model*

### *Test Sensor Reliability*

## Movement Phase

### *Assemble Arena*

### *Develop Basic Robot Internals Library*

### *Move Robots from Controller*

### *Finalize Sensors*

### *Interpret Sensor Information in Controller*

### *Develop World Model*

### *Finalize AI to Controller API*

## AI Phase

### *Research AI Planning Algorithms*

### *Develop AI Planning Algorithm*

### *Enhance Robot Internals Library*

## Enhancement Phase

### *Enhance Algorithm*

### *Expand Formation Number*

### *Enhance Robot Sensors*

## Publicity Phase

### *Team Poster at Research Expo*

### *Department Poster*

### *Demonstrations*