BRADLEY UNIVERSITY

COOPERATIVE CONTROL OF HETEROGENEOUS MOBILE ROBOTS NETWORK

Gregory Bock, Brittany Dhall, Ryan Hendrickson, & Jared Lamkin

Project Advisors: Dr. Jing Wang & Dr. In Soo Ahn

Department of Electrical and Computer Engineering

March 1st, 2016

Outline

I. Introduction
II. E-puck Progress – Brittany & Jared
III. Kilobot Progress - Jared
IV. Qbot Progress – Ryan & Greg
V. Summary & Conclusions

I. Introduction

Problem Description

- Design cooperative control algorithms for heterogeneous groups of robots
- Implement algorithms on different robot platforms

Objectives

- Design and Experimental Validation of Cooperative Control Algorithms
  - Sensing/communication between robots
  - Implementation of local flocking control algorithms
  - Implementation of local formation control algorithms

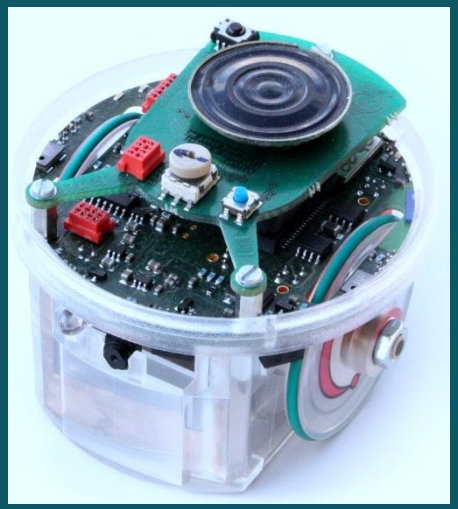
Solution

- Cooperative control algorithm design
  - Linear model
  - Non-linear model
- Deployment and validation through experimental testing
  - Modular design
  - System integration

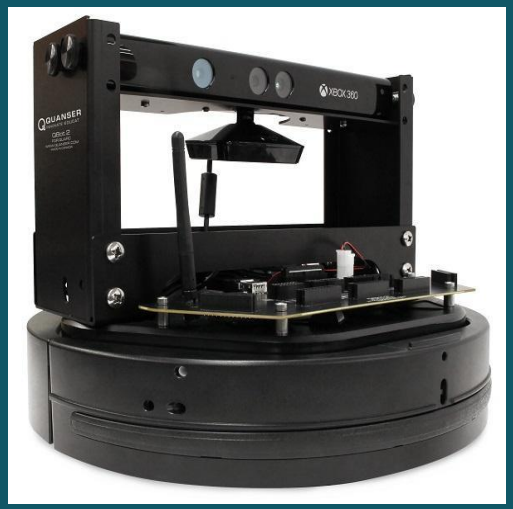
Algorithm Test Platforms

Kilobot

E-Puck



Qbot



http://www.k-team.com/mobile-robotics-products/kilobot
https://en.wikipedia.org/wiki/E-puck_mobile_robot
http://www.mathworks.com/products/connections/product_detail/product_101072.html

Division of Labor Overview

Individual Behavior	Kilobots	Jared
	Qbots	Ryan/Greg
	E-pucks	Brittany
Individual Communication	Kilobot - Kilobot	Jared
	Qbot - Qbot	Ryan/Greg
	E-puck - E-puck	Brittany
Integrated Communication	Kilobot - E-puck	Jared/Brittany
	Kilobot - Qbot	Jared/Ryan/Greg
	E-puck - Qbot	Brittany/Ryan/Greg
Algorithm Design	Linearization Based Model	Jared/Brittany/Ryan/Greg
Integrated Behavior	Formation Control Behavior	Jared/Brittany/Ryan/Greg
	Flocking Behavior	Jared/Brittany/Ryan/Greg
Testing	Software Implementation	Jared/Brittany/Ryan/Greg
	Hardware Implementation	Jared/Brittany/Ryan/Greg

II. E-puck Progress - Brittany

Recap of Last Semester

- Used IR proximity sensors for object detection
- Used IR proximity sensors for object following

https://www.cyberbotics.com/item?id=8

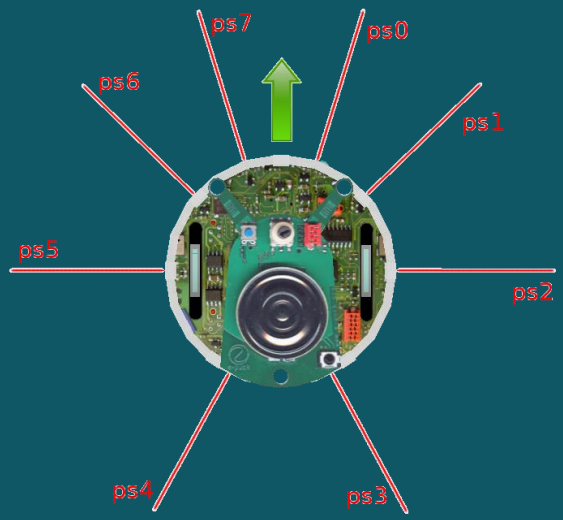
Work Accomplished

- IR proximity sensors used for communication
- Un-brick E-pucks
- Received new batteries for better life of the robot

https://www.cyberbotics.com/item?id=8

IR Proximity Sensors

- Proximity sensors can send a pulse wave
- IR receiver circuit detects signal
- Method to communicate with Kilobots
  - Sent a message (Kilobots could not see it)

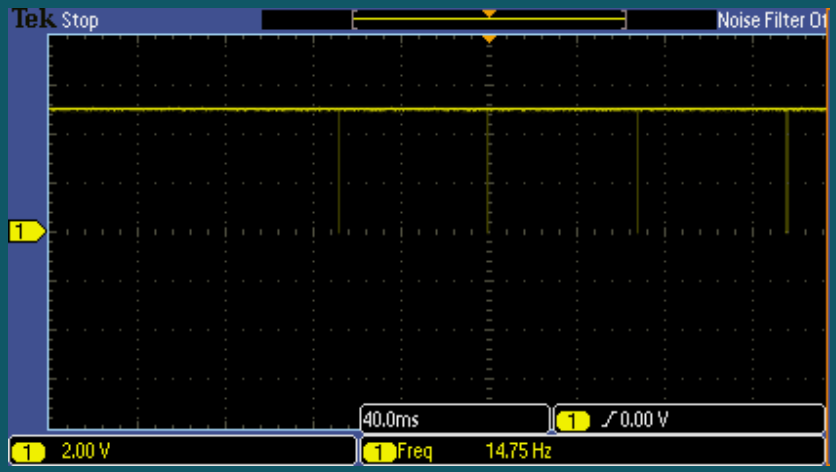


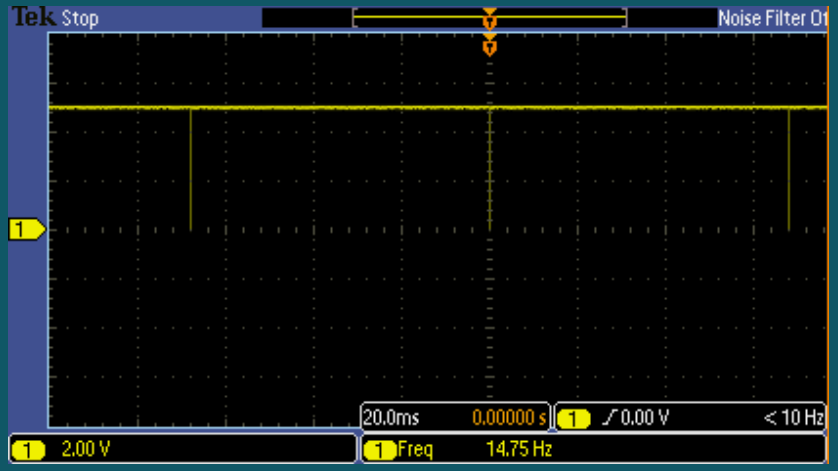
https://www.cyberbotics.com/guide/tutorial-4-more-about-controllers-20-minutes.php

Example Code for IR Proximity Sensor


```
int main() {

   long timer = 0;
        //configure LEDs pin direction
        PULSE_IR0_DIR = OUTPUT_PIN;
        LED0_DIR = OUTPUT_PIN;

   while (1) {

        PULSE_IR0 = 0;
        LED0 = 0;

        for(timer = 0; timer < DELAY; timer++);
        PULSE_IR0 = 1;
        LED0 = 1;

        for(timer = 0; timer < DELAY; timer++);

   }
```

Oscilloscope Captures





Unbricked two E-pucks

- Used the MPLAB ICD 3

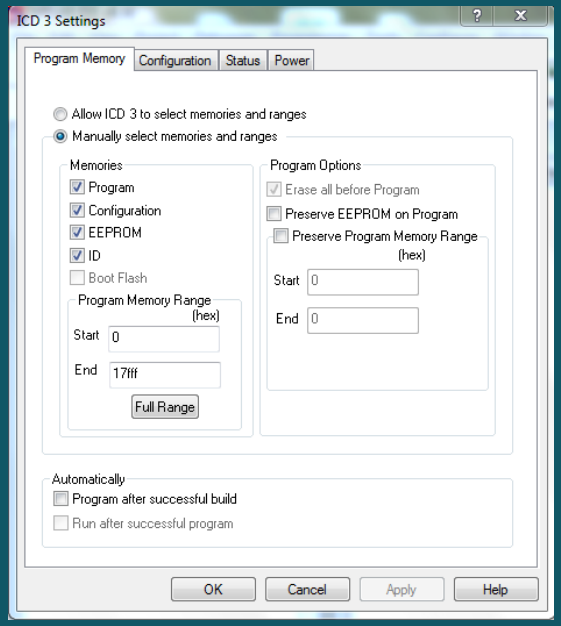
http://microchip.wikidot.com/icd3:start

Steps to Un-Brick the E-puck

- Use MPLAB IDE v8.30
- E-puck must be turned off to start
- Open MPLAB IDE v8.30
- Remove the top portion of the E-puck
- Connect the ICD3 to the computer, then to the E-puck

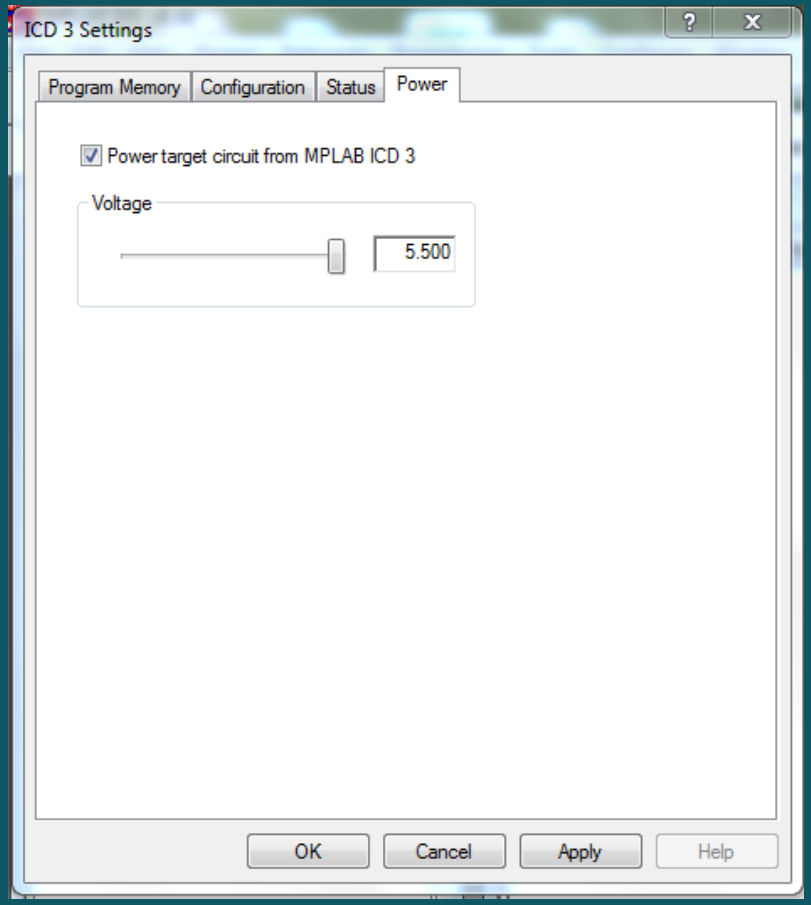
Steps to Un-Brick the E-puck

- Import a basic file into MPLAB
- Go to Programmer -> Select Programmer -> MPLAB ICD3
- Go to Programmer -> Settings -> Program Memory Tab



Steps to Un-Brick the E-puck

- ICD3 Settings Power Tab



Steps to Un-Brick the E-puck

- Select Programmer -> Erase Flash Device
- When Finished, power cycle the E-puck
- Select Programmer -> Program

Problems Encountered

- Infrared sensor communication (no add-on IR Ring)
- Communication using the proximity sensors to the Kilobots

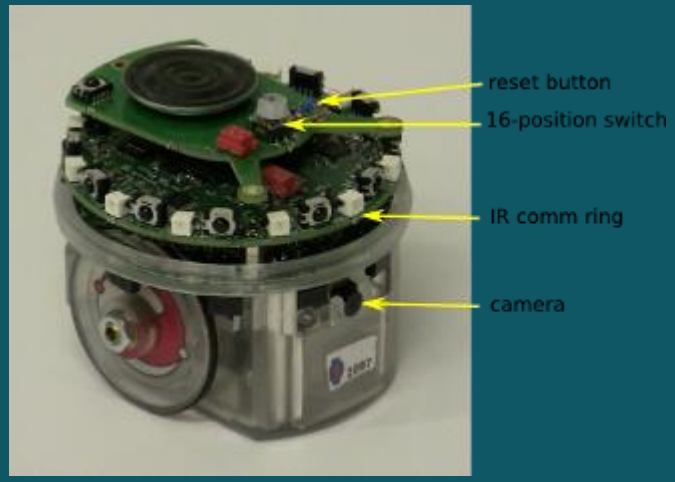


Fig. 1 – E-puck with Add-on IR ring

Fig. 2 – E-puck without Add-on IR ring

Fig. 1. http://www.eecs.harvard.edu/ssr/teaching/epucks/e-site_files/Page609.htm
Fig. 2. http://hades.mech.northwestern.edu/index.php/E-puck_Mobile_Robot

Future Work

- E-puck to E-puck communication
- Using the E-puck's camera
- Q-bot to E-puck communication

Gantt Chart – Future Work

Task Name	Group Member Responsible for Task	Finish by Date/Due Date	Dec - 16		Jan-16		Feb-16				Mar-16					Apr-16			
			1	8	19	26	2	9	16	23	1	8	15	22	29	5	12	19	26
Individual Communication																			
Research/Test E-puck - E-puck	Brittany	December 14, 2015																	
Integrated Communication																			
Test Kilobot - E-puck	Jared/Brittany	December 14, 2015																	
Test E-puck - Qbot	Brittany/Ryan/Greg	December 14, 2015																	
Algorithm Design																			
Design Linear Based Model	All	December 14, 2015																	
Integrated Behavior																			
Formation Control Behavior																			
Localization	All	January 25, 2016																	
Point Convergence	All	January 25, 2016																	
Leader Follower	All	January 25, 2016																	
Testing																			
Software Implementation	All	March 7, 2016																	
Hardware Implementation	All	March 7, 2016																	

E-puck Progress- Jared

Color and Object Detection

- The E-puck CMOS camera is capable of 640X480 resolution, in color or grayscale
- However, the image is too large to process, so instead we use a 1X120 image
- Color uses RGB565, where each pixel has 5 bits for red, 6 bits for green, and 5 bits for blue

Color and Object Detection

- First step to object detection is edge detection
- The image array is searched for two edges, from both left and right starting positions
- Individual pixels are compared to the average of the previous ten pixels
- If the difference is greater than three, that location is set as an edge
- Based on the number of edges found (0,1,2,3,4), The E-puck calculates where the center of the object is, and how wide it is.

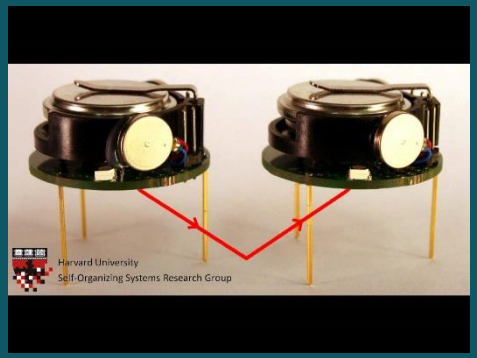
Color and Object Detection

- After Edge detection is complete, the E-puck moves on to color comparison
- The E-puck computes the average RGB value of the object
- The average is compared to the specified value within a certain tolerance
- If the comparison is acceptable, The E-puck begins maneuvering to it.

III. Kilobot Progress - Jared

How Kilobots Communicate

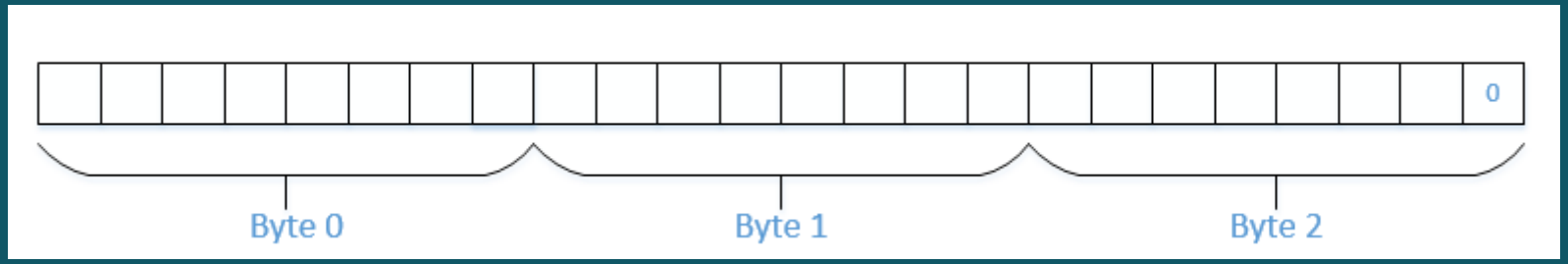
- Use infrared light
- Reflects light off the floor
- Measures light intensity to calculate distance
- Maximum rated range is 7 cm
- Messages are sent every 200 milliseconds



https://i.ytimg.com/vi/ISMwLCFwgK4/maxresdefault.jpg

Preparing Messages to send

- A message is composed of 23 bits (2 full characters, and an even character)
- A check sum is generated by taking the sum of the bits and 128
- All four bytes are operated on
- Each byte is then shifted left and incremented by 1

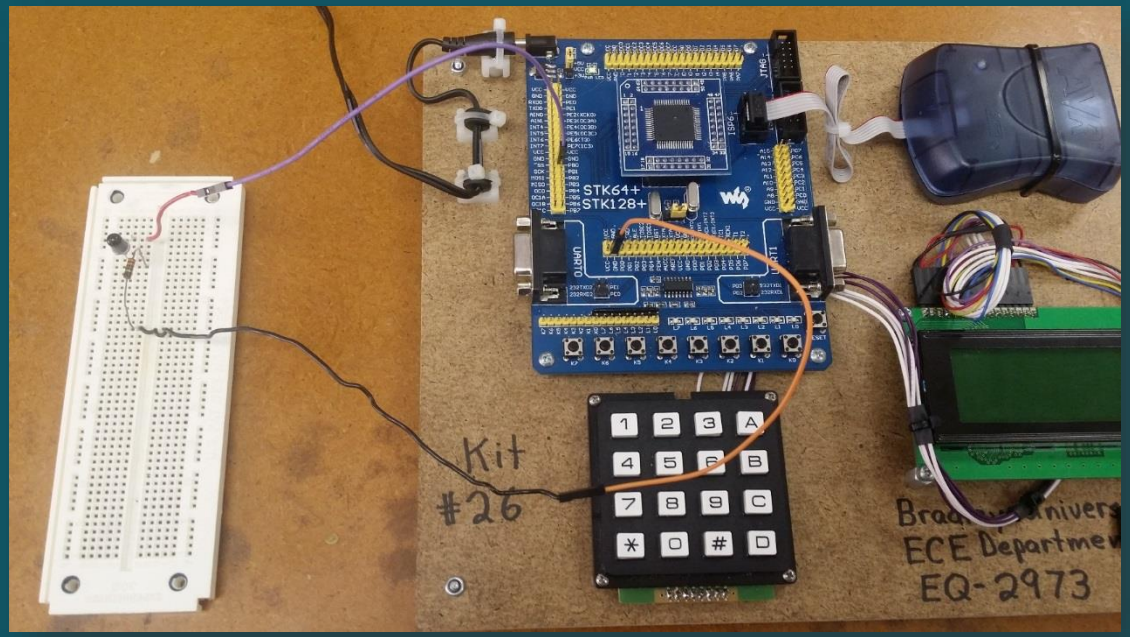


Sending messages

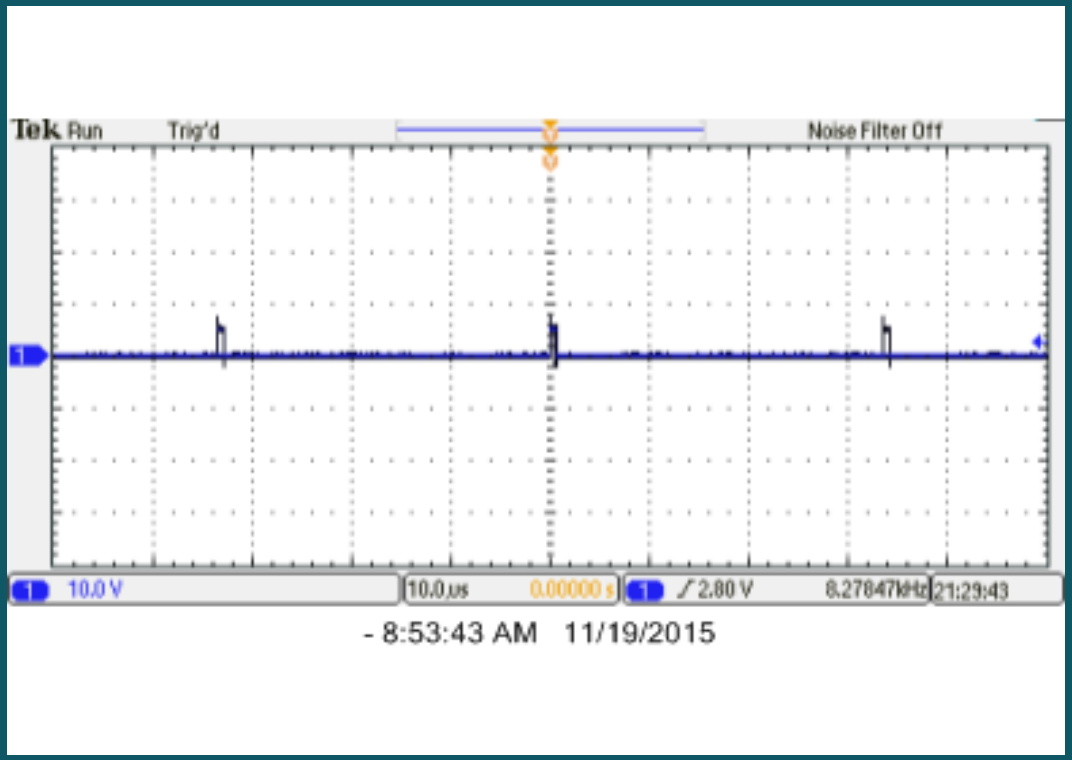
- Signal goes high briefly (0.75 microseconds)
- Signal is then set low for a duration (92.25 microseconds)
- Each bit is sent: high for 1, low for 0
- Signal is set low between each bit (13.875 microseconds per bit)
- 537 microseconds for entire message to be sent

Circuit

- Used an Atmega128 board from the lab
- PB0 in series with a 330 ohm resistor and an IR LED



Oscilloscope capture

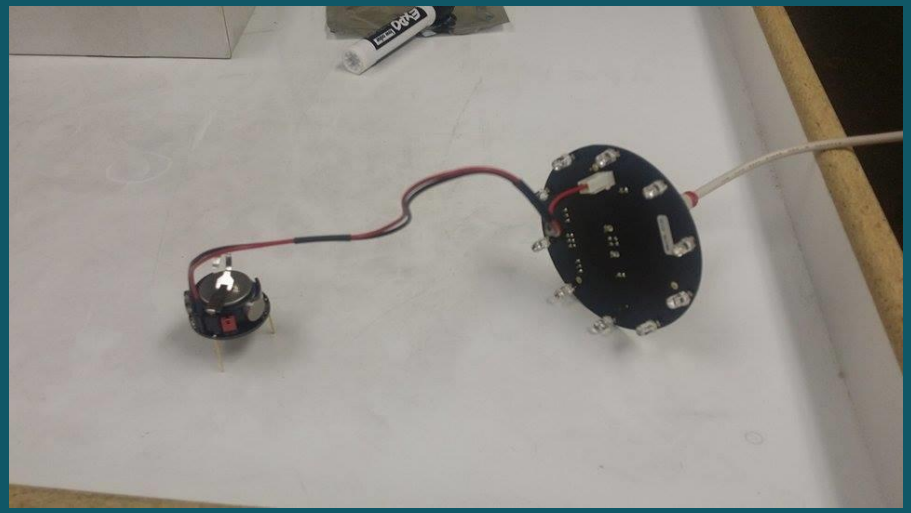


Initial Testing

- A Kilobot was programmed to turn its LED green if message was received
- The Kilobot turns it LED red if message is not received after a set amount of time
- The IR LED light was reflected off the ground next to the Kilobot

Further Testing

- Kilobot was connected to serial cable and programmed to print message values to hyper terminal
- Kilobot was placed at measured distances and compared to printed distance



Controllable Node

- A Kilobot programmed to receive orders from an outside source
- Transmits orders to follower Kilobots to mimic its behavior
- Only performs orders if at least one follower is within range

Follower Nodes

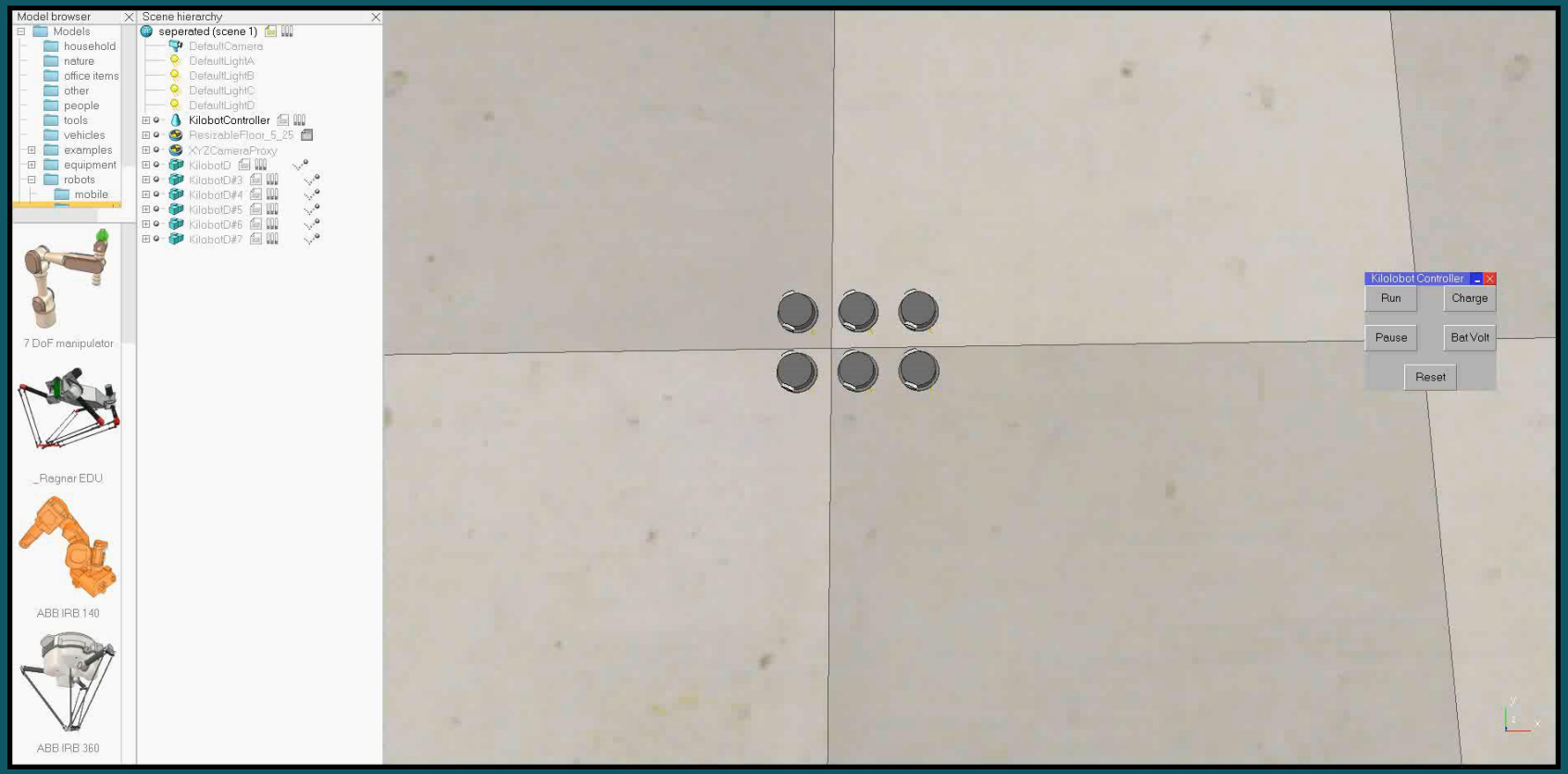
- Perform order received from controllable node
- At start up they perform gradient algorithm with other followers
- Only perform orders received from lowest value follower within range
- They only move if they are within range of the next lowest follower
- Transmit gradient value and heading

Simulation of Controllable and Follower Nodes

Separation and Attraction

- Kilobots generate a random ID at startup
- Kilobots then try to separate into two groups, evens and odds
- Opposites attempt to repulse each other, by orbiting
- Like attracts
- If an agent loses contact with others it stops
- Current issues are collision avoidance

Separation and Attraction Simulation



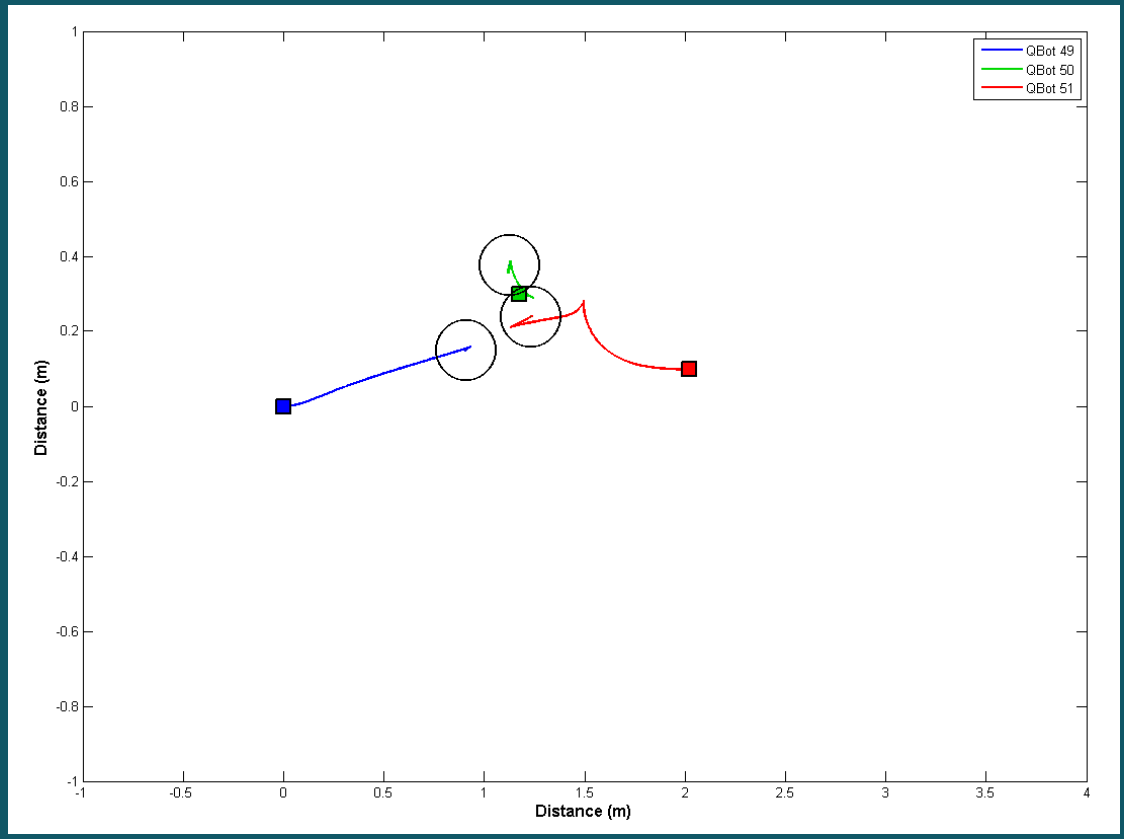
Gantt Chart – Future Work

Task Name	Group Member Responsible for Task	Finish by Date/Due Date	Dec-15		Jan-16		Feb-16				Mar-16					Apr-16			
			1	8	19	26	2	9	16	23	1	8	15	22	29	5	12	19	26
Integrated Communication																			
Test Kilobot - E-puck	Jared/Brittany	December 14, 2015																	
Test Kilobot - Qbot	Jared/Ryan/Greg	November 16, 2015																	
Test E-puck - Qbot	Brittany/Ryan/Greg	December 14, 2015																	
Algorithm Design																			
Design Linear Based Model	All	December 14, 2015																	
Integrated Behavior																			
Formation Control Behavior																			
Localization	All	January 25, 2016																	
Point Convergence	All	January 25, 2016																	
Leader Follower	All	January 25, 2016																	
Flocking Behavior																			
Neighbor Repulsion	All	February 1, 2016																	
Enpoint Attraction	All	February 1, 2016																	
Heading	All	February 1, 2016																	
Testing																			
Software Implementation	All	March 7, 2016																	
Hardware Implementation	All	March 7, 2016																	

III. QBot 2 Progress – Ryan & Greg

Recap of Last Semester

- Localization
- Point Consensus
- Formation

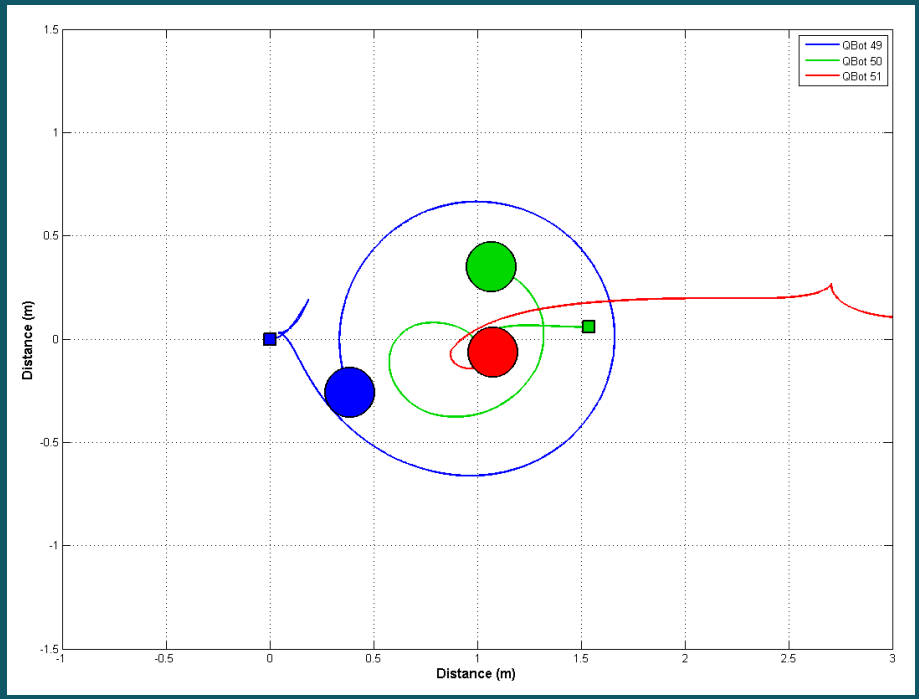


Path Following

- Sinusiodal Motion
- $\dot{x} = \omega$
- $x_d = \omega t$
- $\dot{y} = \omega \cos(x_d)$
- $y_d = \sin(x_d)$
- $v_1 = -k(x_d - z_1) + \dot{x}$
- $v_2 = -k(y_d - z_2) + \dot{y}$

Path Following

- Circular Motion
- $v_1 = -k(x_d - z_1) + \dot{x}$
- $v_2 = -k(y_d - z_2) + \dot{y}$
- $x_d = a + r\cos(t/n)$
- $y_d = b + r\sin(t/n)$
- $\dot{x} = -r\sin(t/n)/n$
- $\dot{y} = r\cos(t/n)/n$



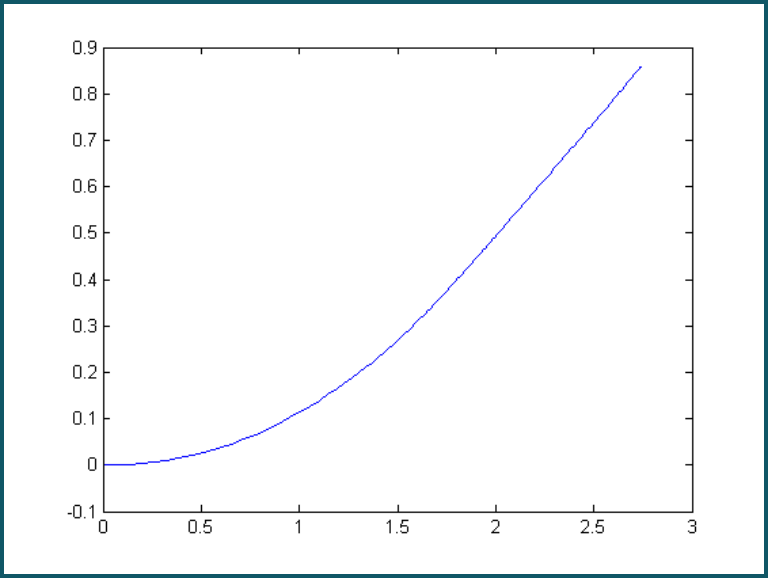
Path Following – Figure 8

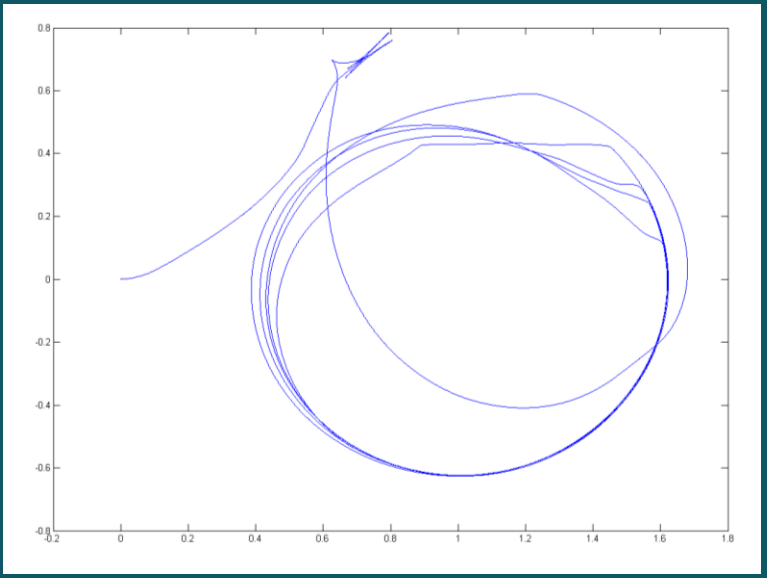
QBot Progress - Greg

Object Avoidance Using Fuzzy Logic

- Non-binary, degrees of truth
- Three inputs – left side, middle, right side
- Two outputs – left motor, right motor

Object Avoidance Using Fuzzy Logic Results





Object Avoidance

Gantt Chart – Future Work

Task Name	Group Member Responsible for Task	Finish by Date/Due Date	Dec-15		Jan-16		Feb-16				Mar-16					Apr-16			
			1	8	19	26	2	9	16	23	1	8	15	22	29	5	12	19	26
Integrated Communication																			
Test Kilobot - E-puck	Jared/Brittany	December 14, 2015																	
Test Kilobot - Qbot	Jared/Ryan/Greg	November 16, 2015																	
Test E-puck - Qbot	Brittany/Ryan/Greg	December 14, 2015																	
Algorithm Design																			
Design Linear Based Model	All	December 14, 2015																	
Integrated Behavior																			
Formation Control Behavior																			
Localization	All	January 25, 2016																	
Point Convergence	All	January 25, 2016																	
Leader Follower	All	January 25, 2016																	
Flocking Behavior																			
Neighbor Repulsion	All	February 1, 2016																	
Enpoint Attraction	All	February 1, 2016																	
Heading	All	February 1, 2016																	
Testing																			
Software Implementation	All	March 7, 2016																	
Hardware Implementation	All	March 7, 2016																	

VIII. Summary & Conclusions

Summary & Conclusions

- Design cooperative control algorithms for heterogeneous groups of robots
- Implement algorithms on different robot platforms
- Prevent collisions and implement network security
- Behind Schedule

Future Plan of Action

- Communication between platforms
- Algorithm design
- Integrated behavior

BRADLEY UNIVERSITY

COOPERATIVE CONTROL OF HETEROGENEOUS MOBILE ROBOTS NETWORK

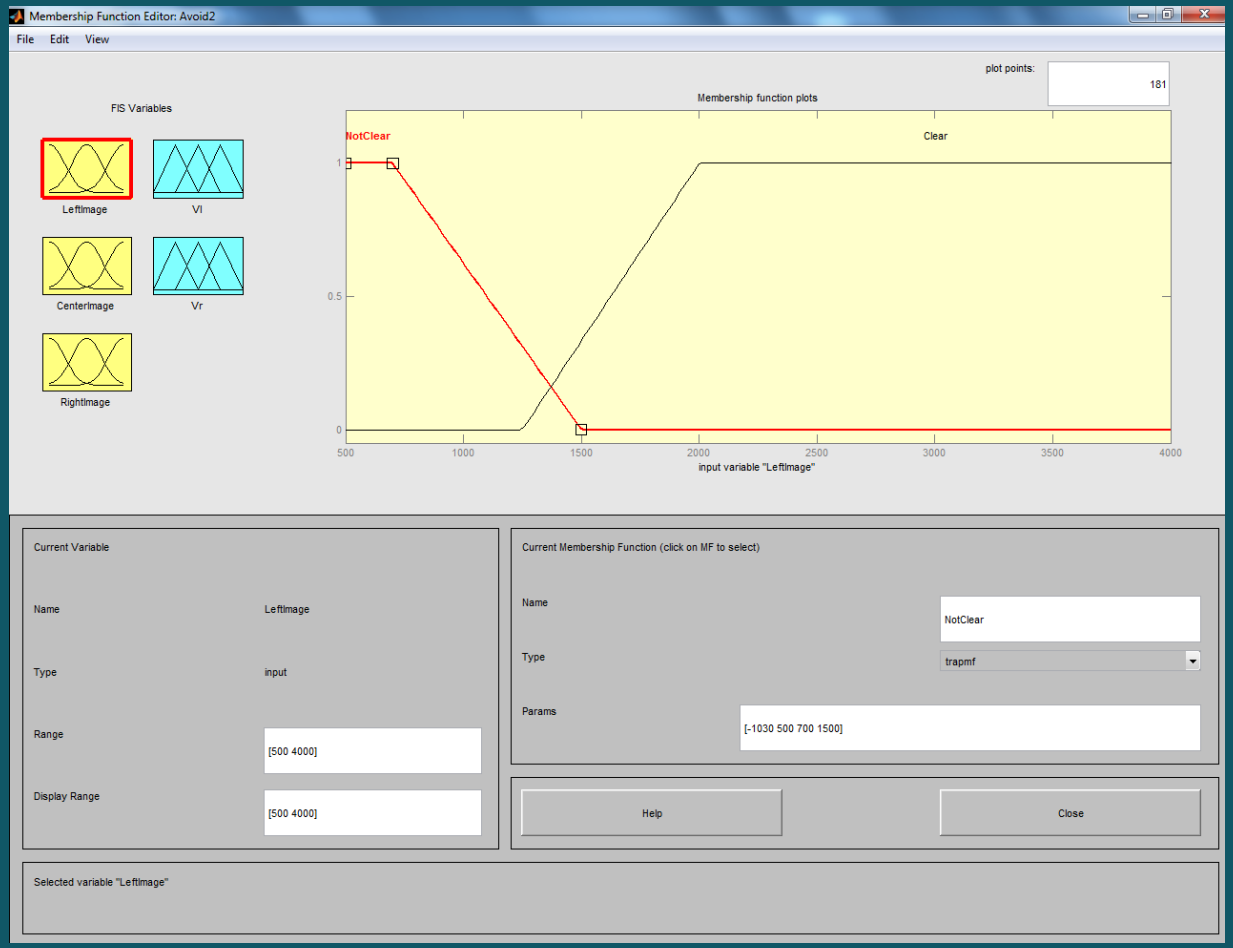
Gregory Bock, Brittany Dhall, Ryan Hendrickson, & Jared Lamkin

Project Advisors: Dr. Jing Wang & Dr. In Soo Ahn

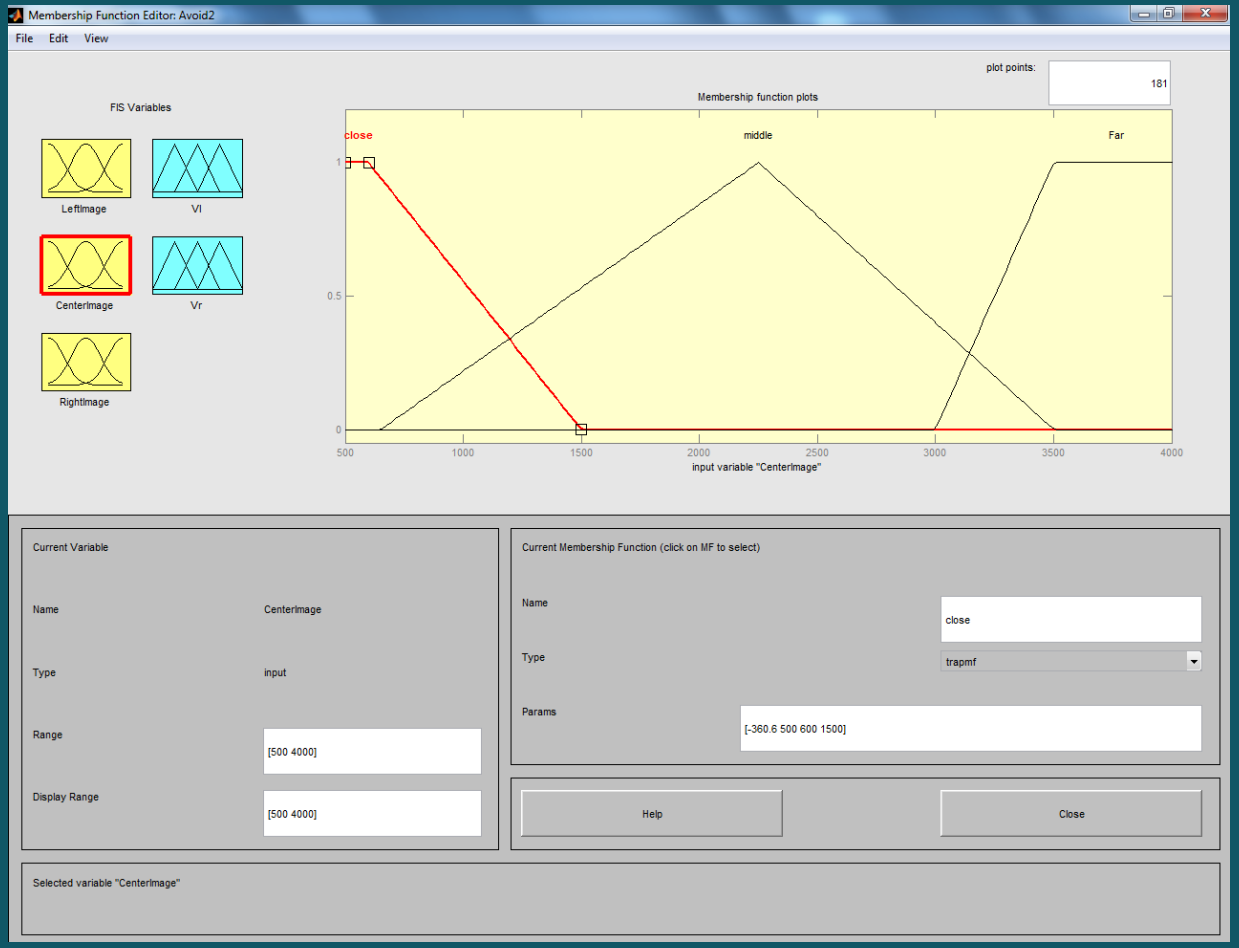
Department of Electrical and Computer Engineering

October 6th, 2015

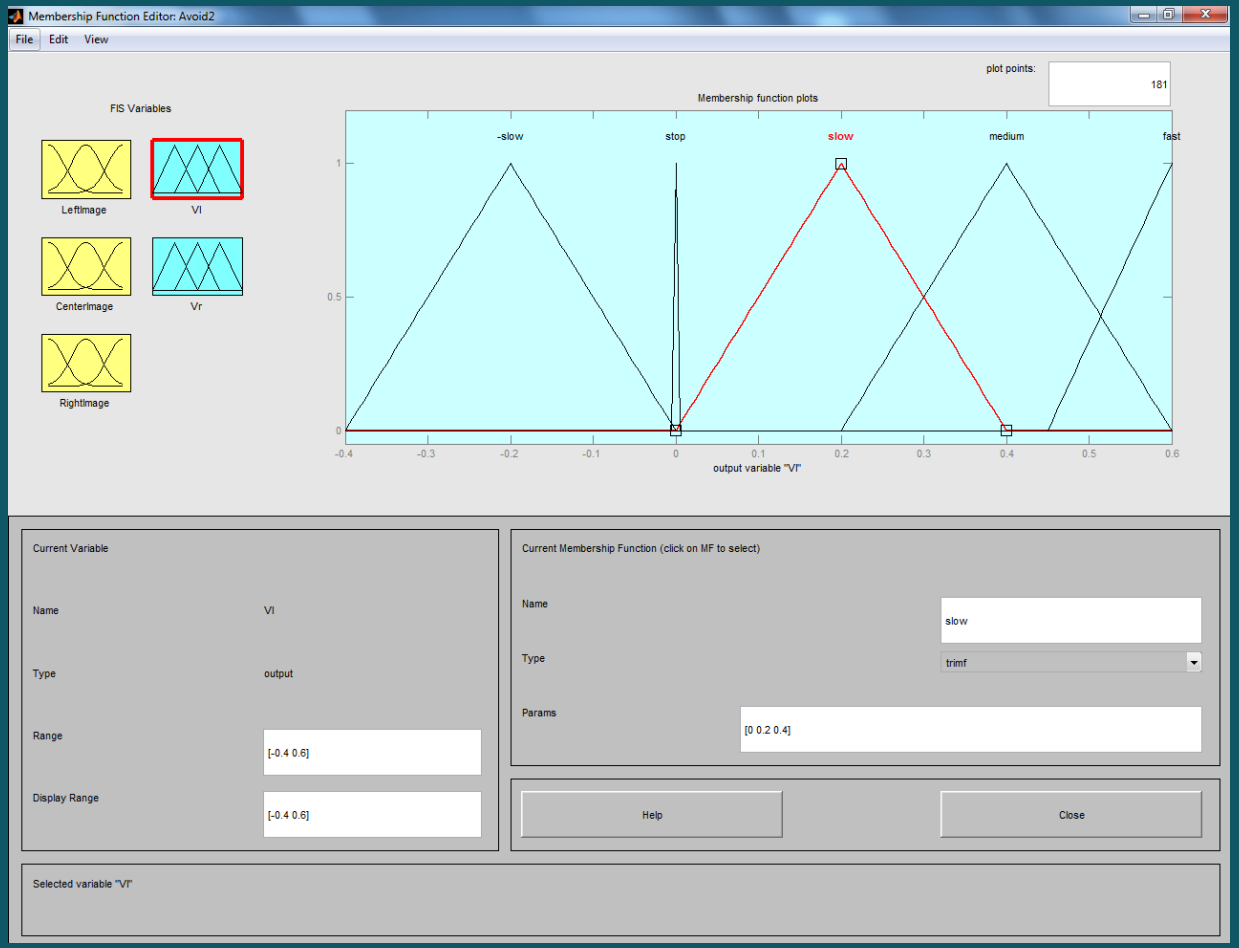
Left Side Input



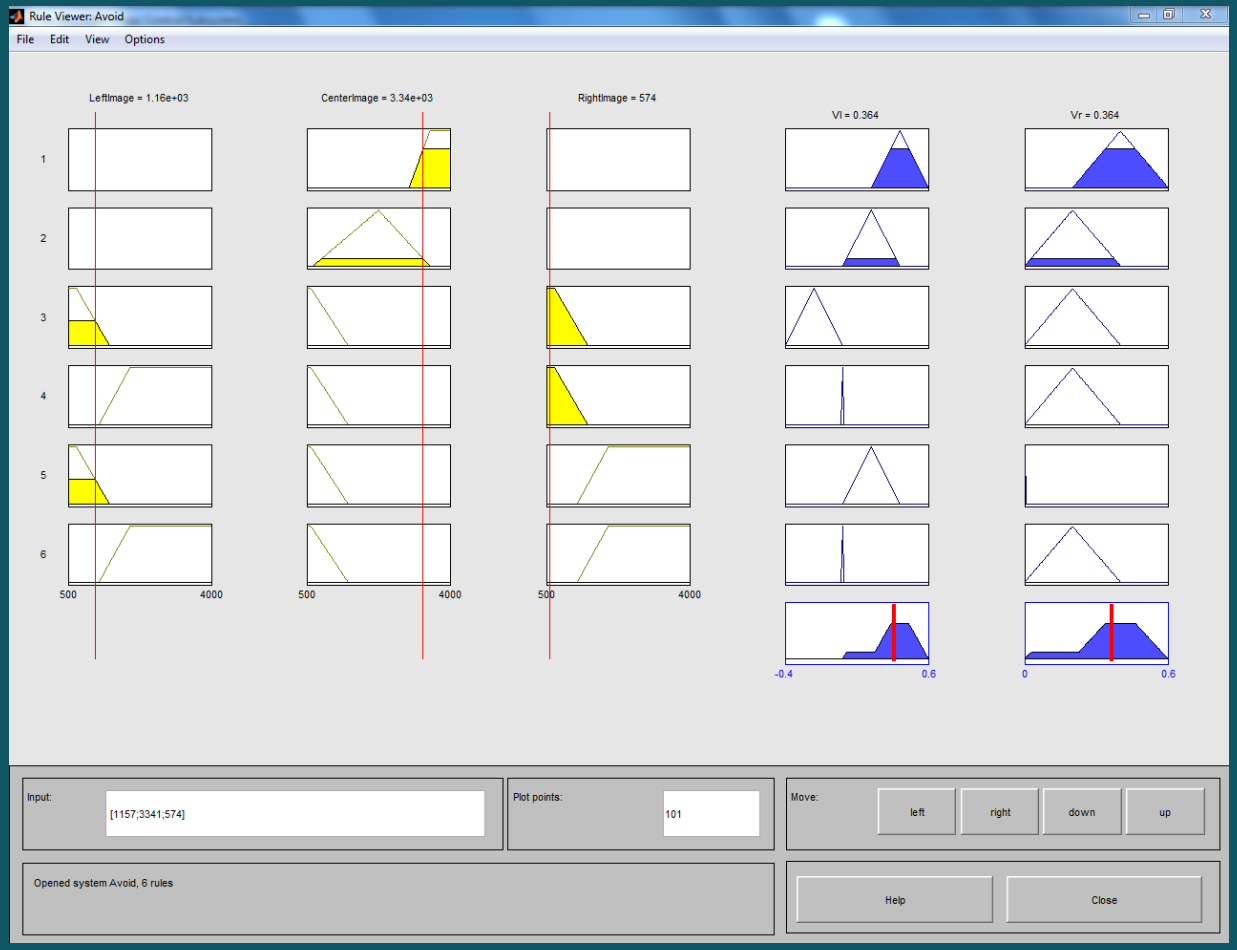
Middle Input



Left Motor Output



Simulation of Rules



Design Constraints

- Must overcome limited communication among networked robots
- Must overcome limited sensing capability of robots
- Must overcome system uncertainties

Objectives

- Mobile robot network should be applicable to different robot platforms
- Mobile robot network should be robust
- Mobile robot network should be autonomous

Solution

- Cooperative control algorithm design
  - Linear model
  - Non-linear model
- Deployment and validation through experimental testing
  - Modular design
  - System integration

Robot Model

- Linear Model
  - $\dot{x} = U_x$
  - $\dot{y} = U_y$
- Non-linear Model
  - $\dot{x} = vcos(\theta)$
  - $\dot{y} = vsin(\theta)$
  - $\dot{\theta} = \omega$

Solution Testing

- Software Implementation
  - Simulation
  - Algorithm validation
  - Algorithm implementation on platforms
- Hardware Implementation
  - Robot calibration
  - Multiple sensor fusion
- System Integration
  - Software
  - Hardware

Criteria to Determine a Successful Project

- Algorithm can be deployed on multiple robots
- Autonomous robots
- Communication amongst multiple robots

Project Funding

- Air Force Research Lab
- Air Force Proposal - "Multiagent task coordination using a distributed optimization approach"
- Grant Agreement Number – FA8780-13-1-0109

http://www.af.mil/News/Photos/tabid/129/igphoto/2000100887/Default.aspx

Expenses

- Robotic platforms (software included)
- Auxiliary components

Project Platform Costs

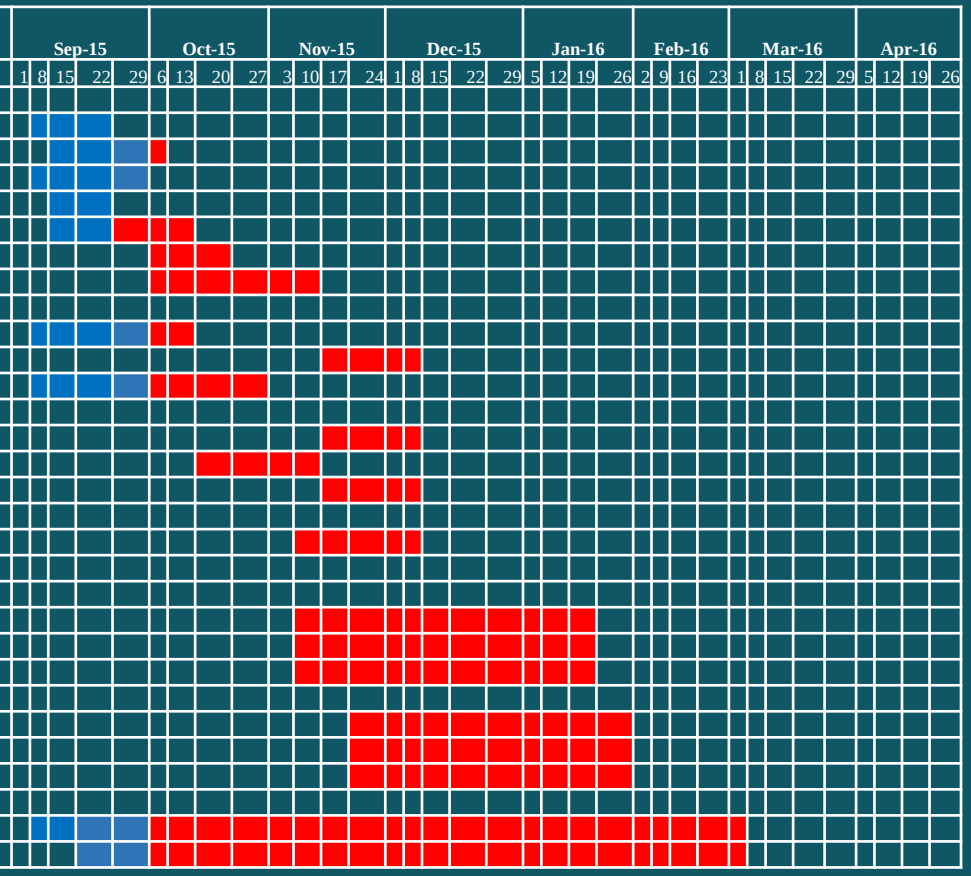
Platform	Quantity	Total Price
Qbot2	3	$9,999.00
Kilobot Kit	20	$4,583.00
Epucks	3	$5,093.00

Programming Software Costs

Software	Quantity	Total Price
Kilobot Controller IDE	1	$0.00
E-puck Programming Software	1	$0.00
MATLAB Courseware	1	$0.00

Gantt Chart

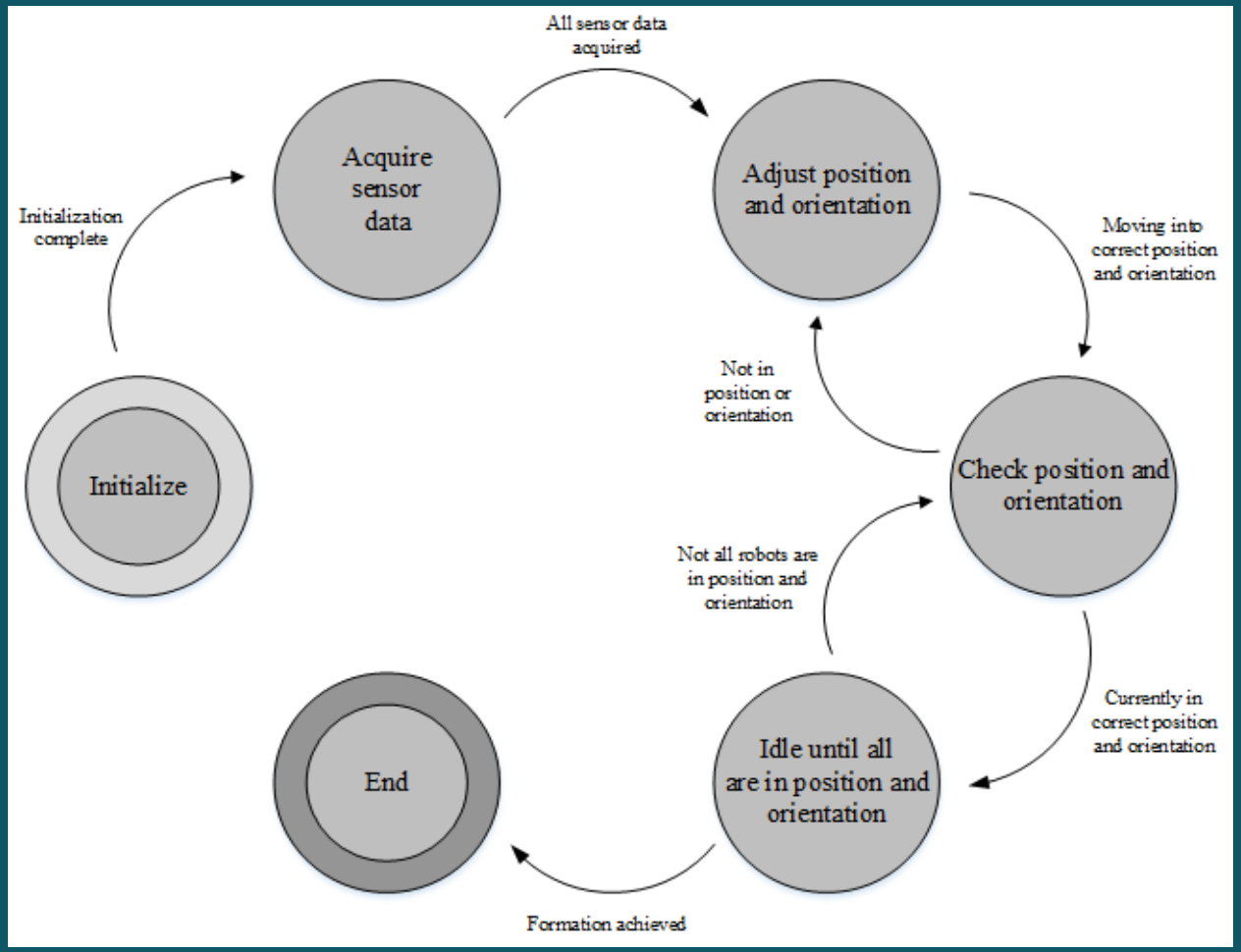
Task Name	Group Member Responsible for Task	Finish by Date
Individual Behavior		
Research Kilobot Sensors	Jared	September 28, 2015
Research Kilobot Communication Protocol	Jared	October 12, 2015
Research Q-bot Image Processing	Ryan/Greg	October 5, 2015
Research Q-bot Sensors	Ryan/Greg	September 28, 2015
Research Q-bot Communication Protocol	Ryan/Greg	October 19, 2015
Research E-puck Sensors	Brittany	October 26, 2015
Research E-puck Communication Protocol	Brittany	
Individual Communication		
Research/Test Kilobot - Kilobot	Jared	October 19, 2015
Research/Test E-puck - E-puck	Brittany	December 14, 2015
Research/Test Qbot - Qbot	Ryan/Greg	November 2, 2015
Integrated Communication		
Test Kilobot - E-puck	Jared/Brittany	December 14, 2015
Test Kilobot - Qbot	Jared/Ryan/Greg	November 16, 2015
Test E-puck - Qbot	Brittany/Ryan/Greg	December 14, 2015
Algorithm Design		
Design Linear Based Model	All	December 14, 2015
Integrated Behavior		
Formation Control Behavior		
Localization	All	January 25, 2016
Point Convergence	All	January 25, 2016
Leader Follower	All	January 25, 2016
Flocking Behavior		
Neighbor Repulsion	All	February 1, 2016
Endpoint Attraction	All	February 1, 2016
Heading	All	February 1, 2016
Testing		
Software Implementation	All	March 7, 2016
Hardware Implementation	All	March 7, 2016



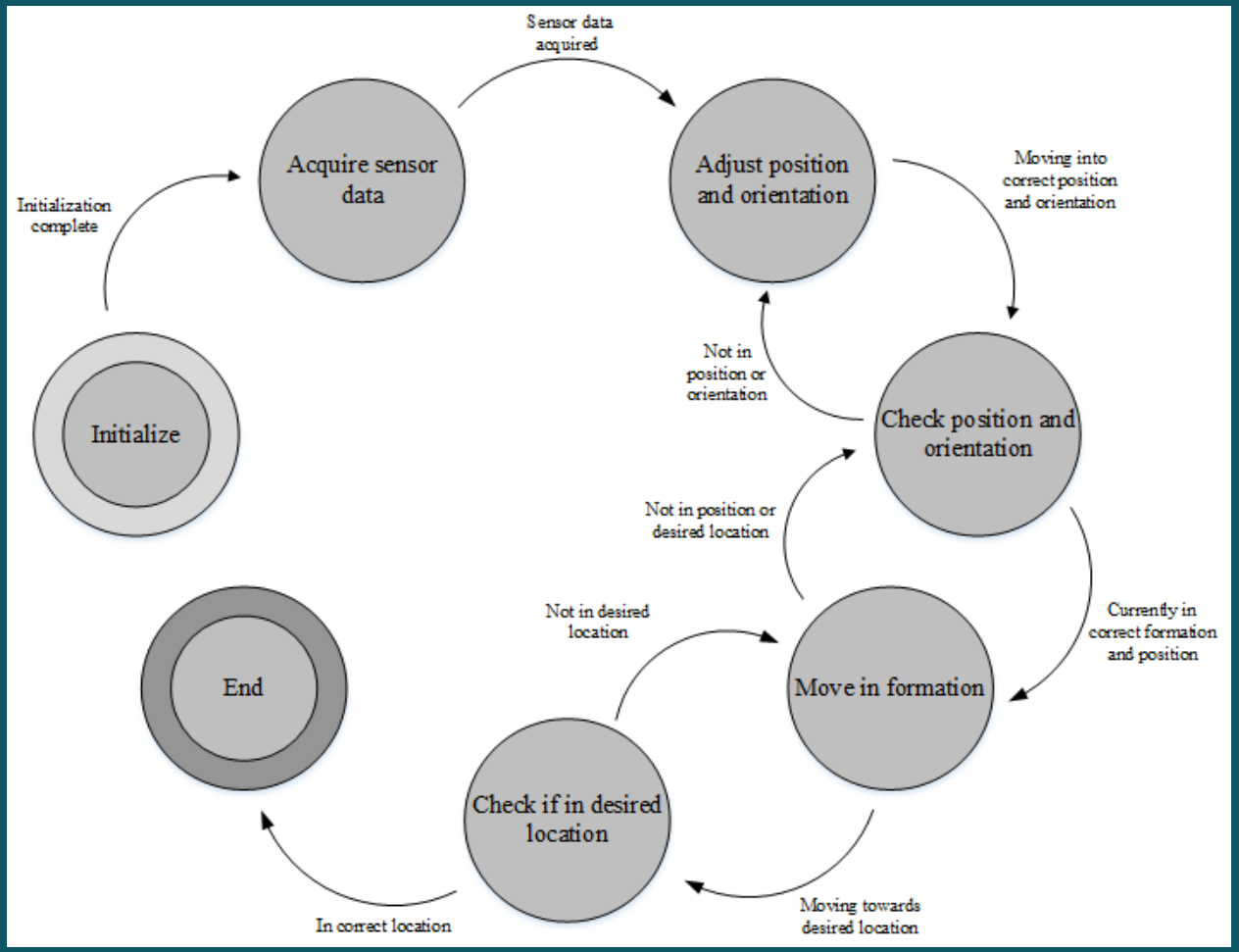
Gantt Chart – Deliverables

Task Name	Finish by Date/Due Date	Sep-15					Oct-15				Nov-15				Dec-15					Jan-16				Feb-16				Mar-16					Apr-16			
		1	8	15	22	29	6	13	20	27	3	10	17	24	1	8	15	22	29	5	12	19	26	2	9	16	23	1	8	15	22	29	5	12	19	26
Deliverables																																				
Project Proposal - Oral Presentation	October 1, 2015																																			
Project Proposal - Document	October 15, 2015																																			
Webpage Release	October 28, 2015																																			
Fall Progress Presentation	November 19, 2015																																			
Fall Performance Evaluation	November 19, 2015																																			
Fall Performance Review	December 3, 2015																																			
Spring Progress Presentation	February 18, 2016																																			
Student Expo Abstract	March 18, 2016																																			
Project Demonstration	March 24, 2016																																			
Final Presentation	April 7, 2016																																			
Student Expo Poster Printing Deadline	April 11, 2016																																			
Student Expo Poster Setup	April 12, 2016																																			
Student Expo	April 14, 2016																																			
Final Report (Draft)	April 14, 2016																																			
Final Report	April 28, 2016																																			
Final Web Page	April 28, 2016																																			
Advisory Board Poster Printing Deadline	April 28, 2016																																			
Advisory Board Poster Presentation	April 29, 2016																																			

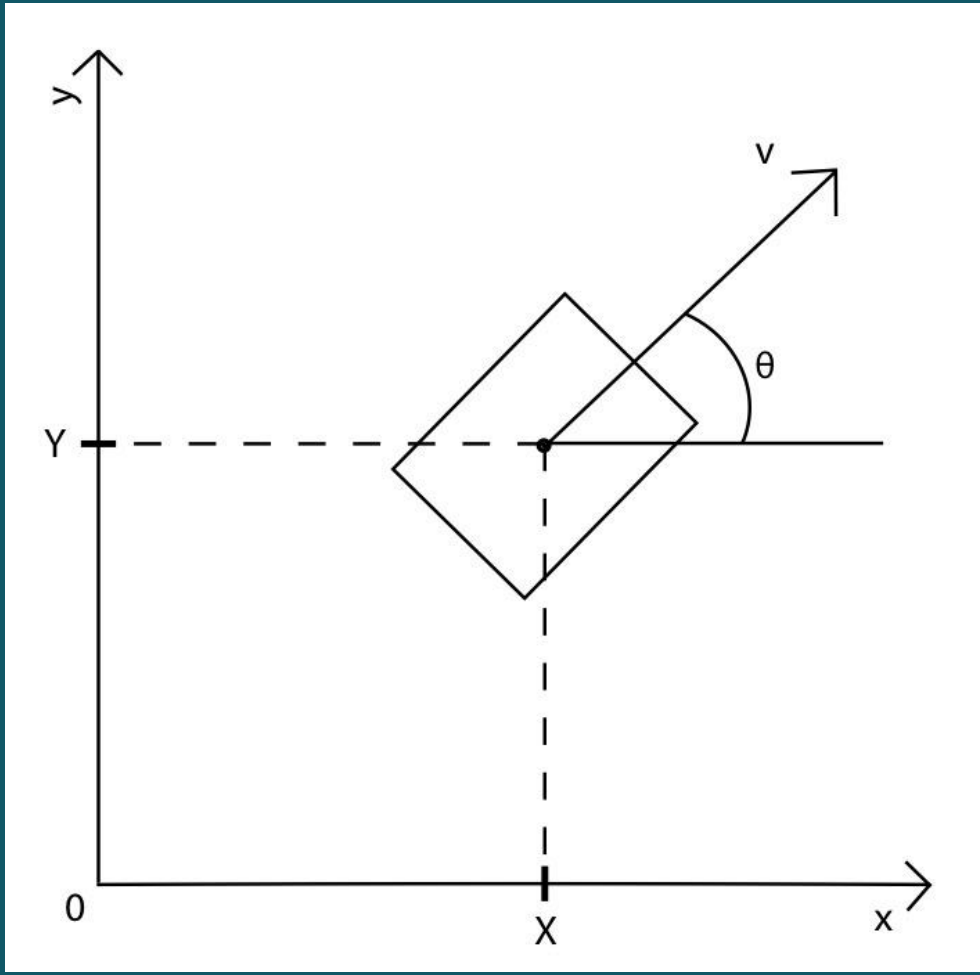
State Diagram: Formation Control Behavior



State Diagram: Flocking Formation

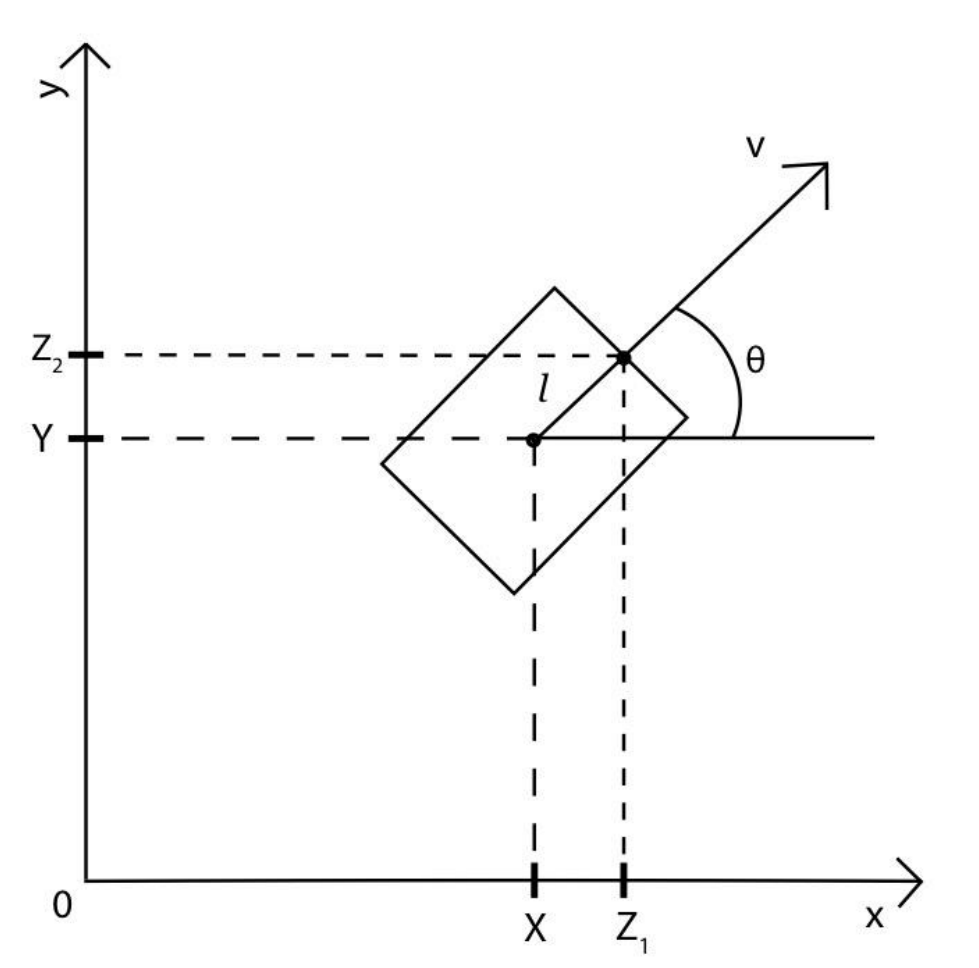


Non-linear Model



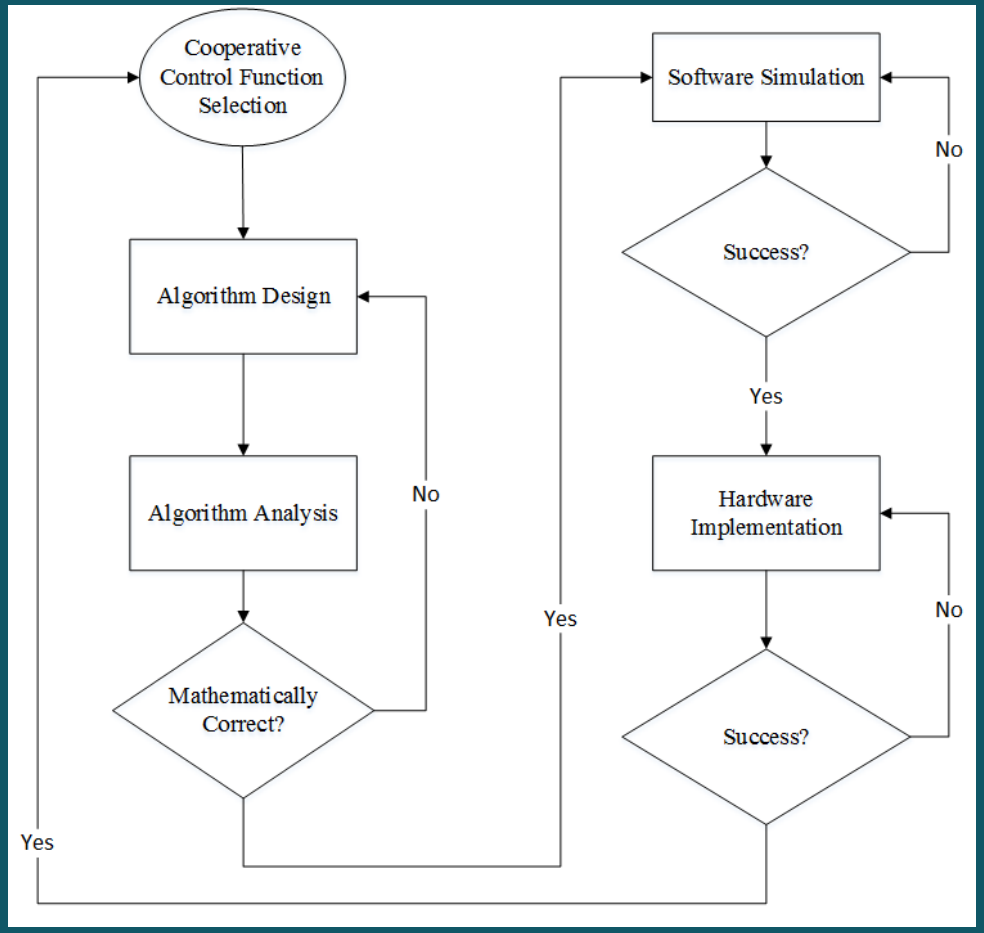
Quanser Inc. “QBOT 2 Workbook” Markham, Ontario

Linear Model



Quanser Inc. “QBOT 2 Workbook” Markham, Ontario

Solution Testing



E-puck Object Following Code


```
/*****************************************************************************/
/** Motor speed controlled depending on front proximity sensor values **/
/*****************************************************************************/

#include "p30f6014A.h"
#include "e_epuck_ports.h"
#include "e_init_port.h"
#include "e_ad_conv.h"
#include "e_prox.h"
#include "e_motors.h"

#define DELAY 50000

int main() {

    long timer = 0;

    //system initialization
    e_init_port();              // configure port pins
    e_init_ad_scan(ALL_ADC);    // configure Analog-to-Digital Converter Module

    while (1) {

        if (e_get_prox(0)> 500) {       //escape
            e_set_speed_left(0);
            e_set_speed_right(0);
        } else if (e_get_prox(0)>100) { //follow
            e_set_speed_left(400);
            e_set_speed_right(400);
        } else {                        //stop
            e_set_speed_left(0);
            e_set_speed_right(0);
        }

        //wait a little to let the robot move
        for(timer = 0; timer < DELAY; timer++);

    }
}
```

Kilobot message operations

- data_out[i]=(data_to_send[i] & (1<<0))*128 +
(data_to_send[i] & (1<<1))*32 +
(data_to_send[i] & (1<<2))*8 +
(data_to_send[i] & (1<<3))*2+
(data_to_send[i] & (1<<4))/2+
(data_to_send[i] & (1<<5))/8 +
(data_to_send[i] & (1<<6))/32 +
(data_to_send[i] & (1<<7))/128;

data_out[i]=data_out[i]<<1;
data_out[i]++;

LEDs and Buttons

- Found LED and Button addresses for reading and writing
- LEDs can be used for debugging
- Buttons can be used for synchronous start-up

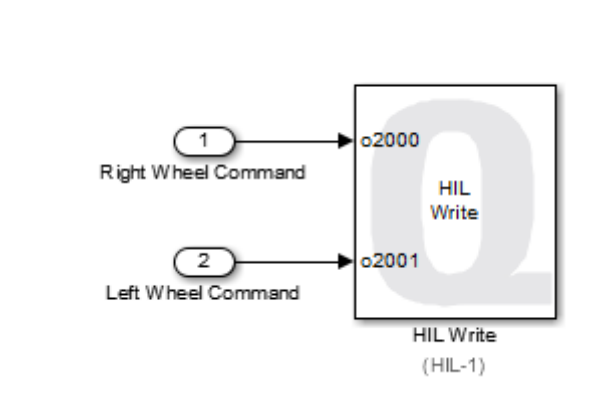
QBot Point Convergence Code

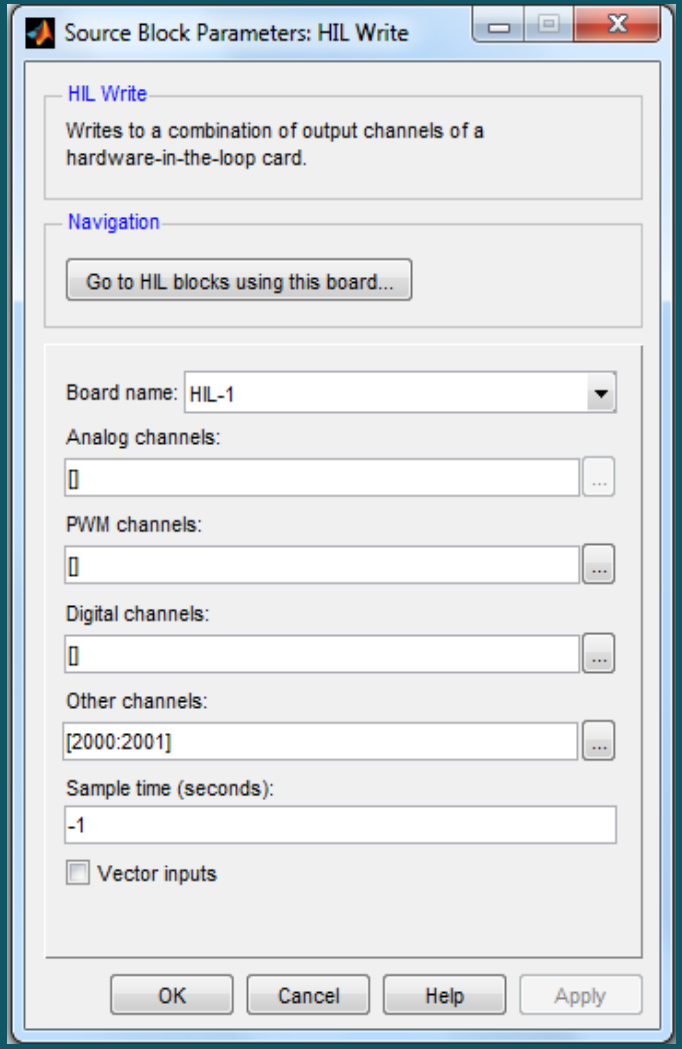

```
v11 = k1*(z21 - z11); % Calculate velocity in x direction
v12 = k2*(z22 - z12); % Calculate velocity in y direction
mat = [cos(myTheta) -d/2*sin(myTheta); sin(myTheta) d/2*cos(myTheta)];
myControl = inv(mat)*[v11;v12];

% Determine total velocity
V = myControl(1);

% Determine angular velocity
omega = myControl(2);

% Determine left and right wheel velocity
Vl = (2*V-d*omega)/2;
Vr = (2*V+d*omega)/2;
```

QBot Obtained Angle Equation

- $\alpha = (320\ - column) * (57/640) * (\pi - 180)$

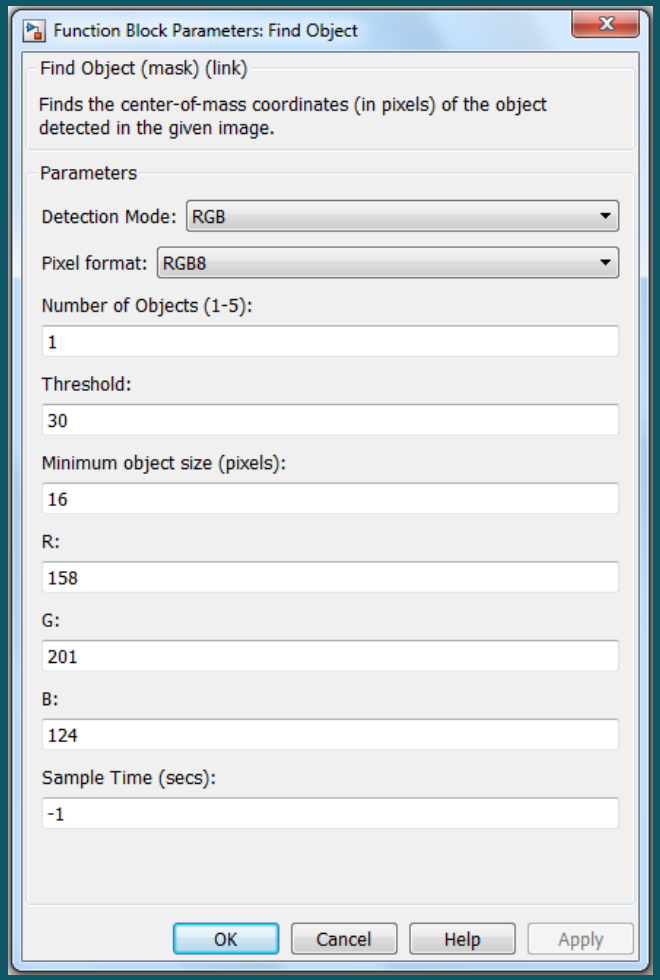
HIL Write Block



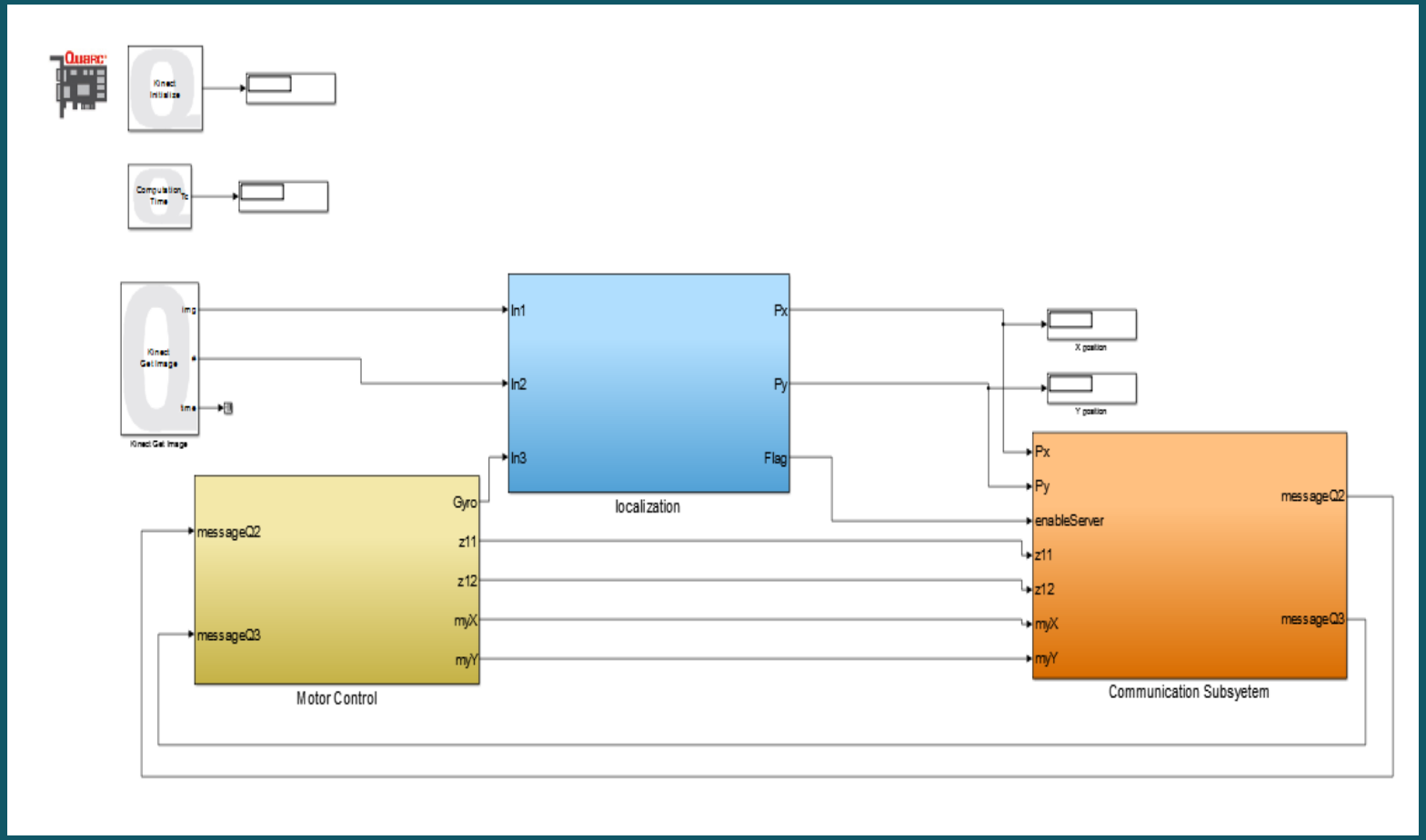


Find Object Parameters

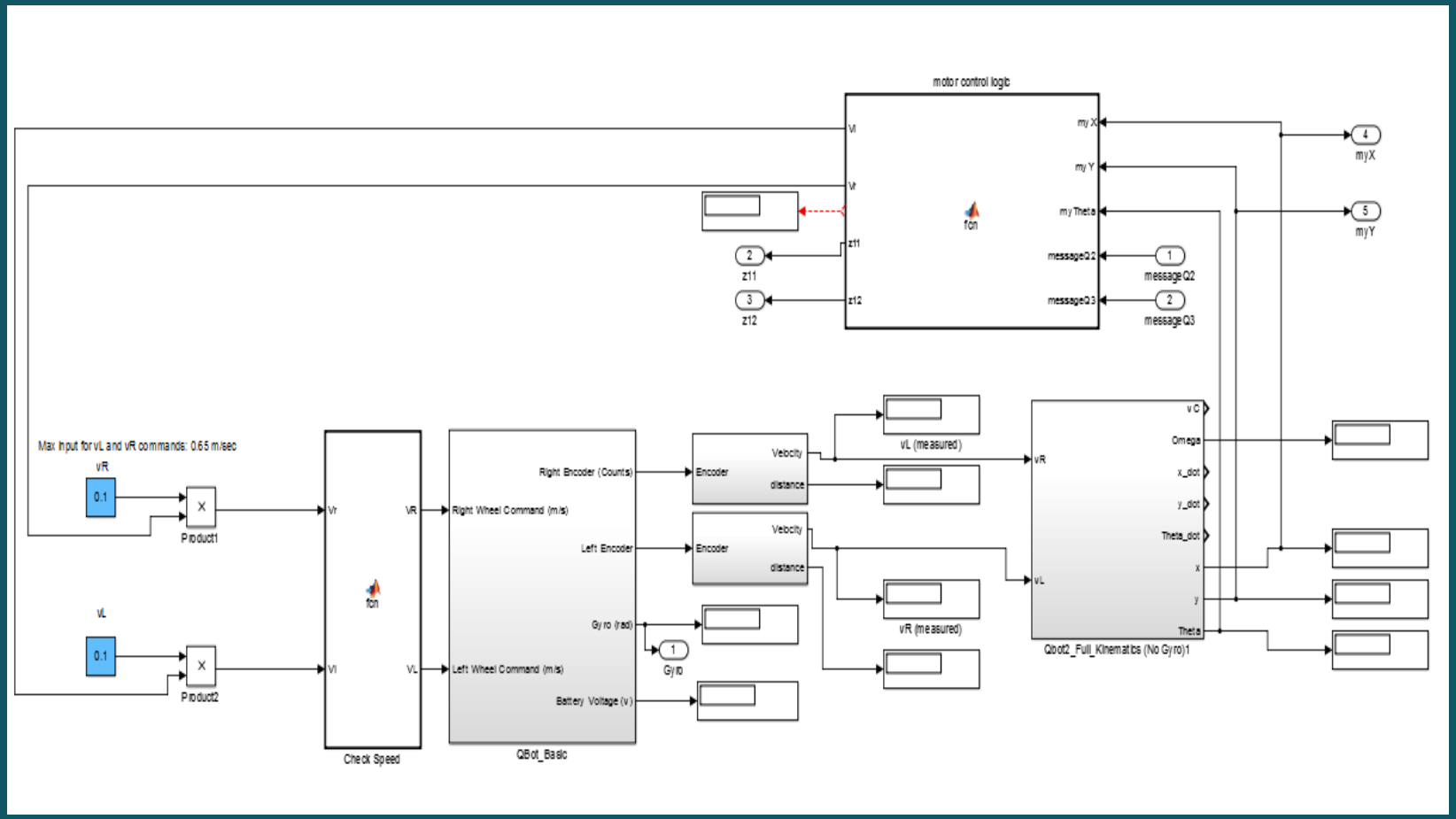
- Specify RGB values
- Value threshold
- Number of objects



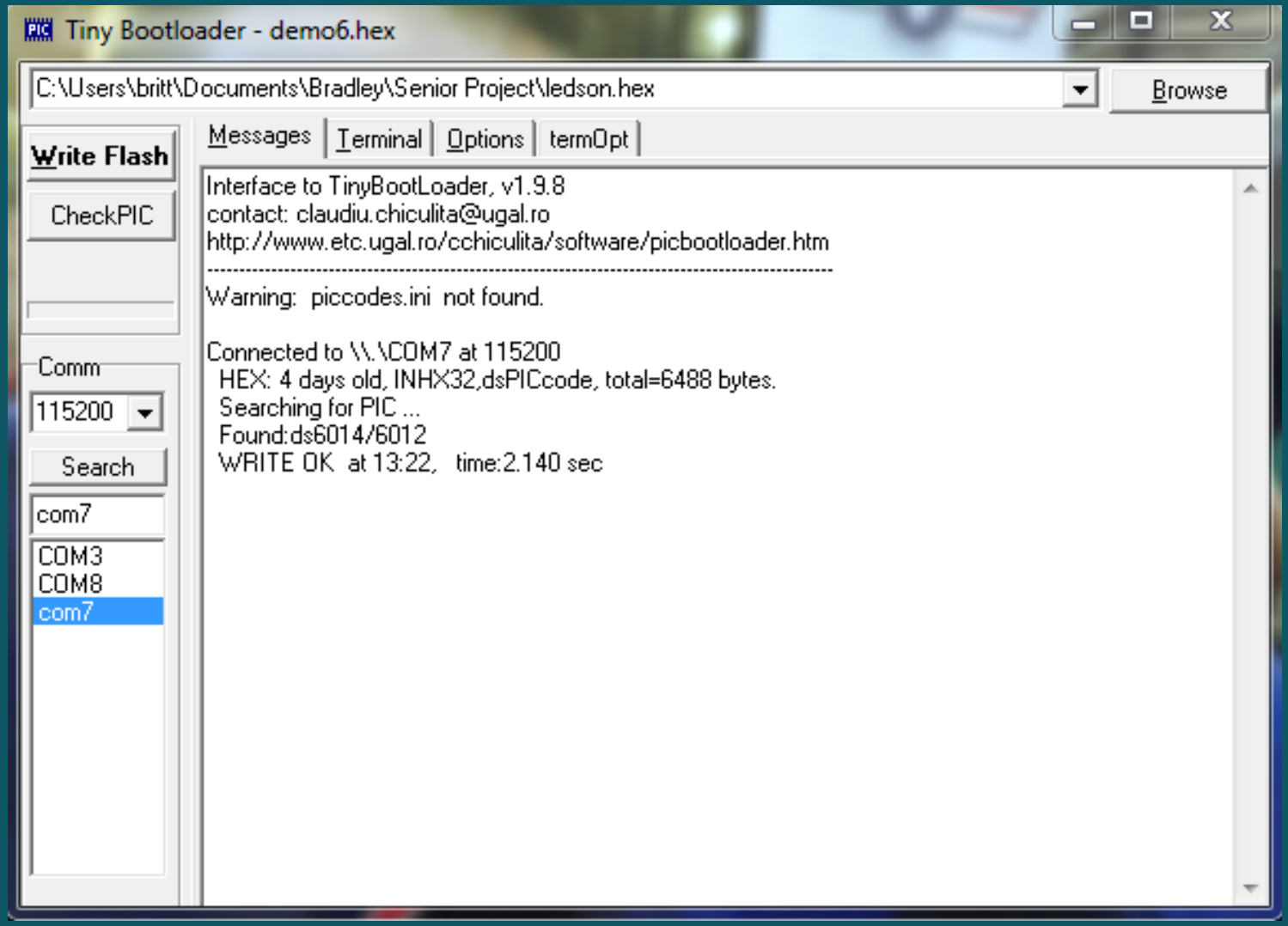
Overall Simulink Model



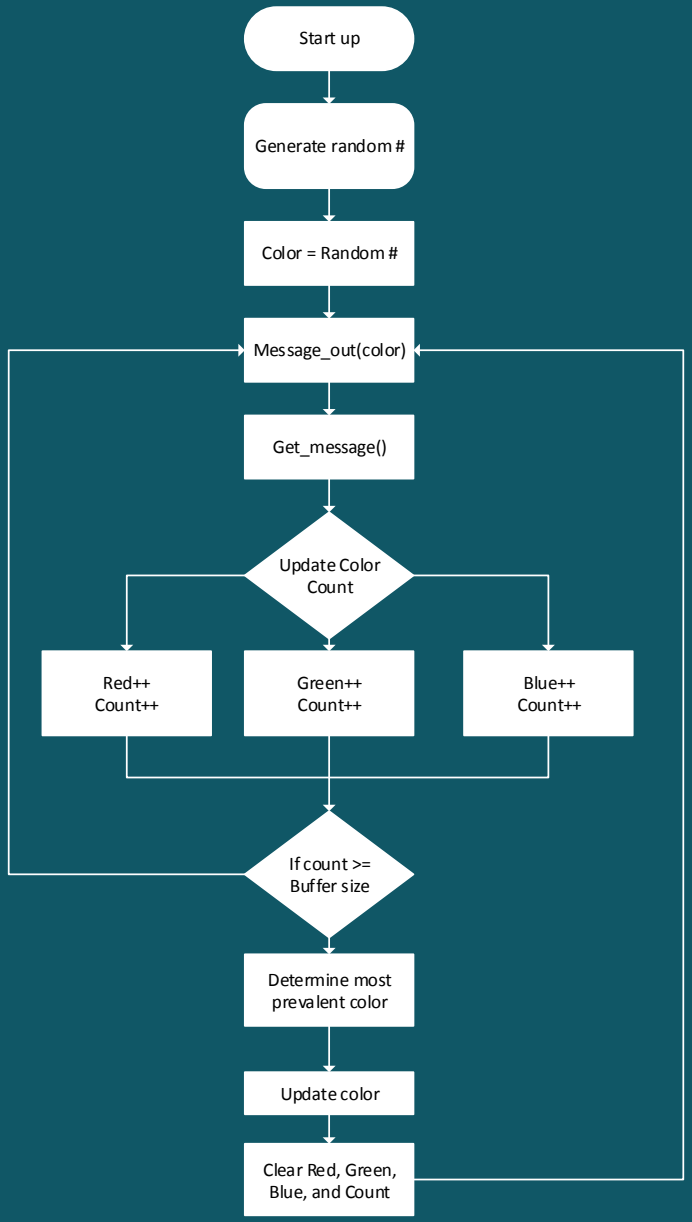
Motor Control



E-puck Tiny Bootloader



Color Consensus Flowchart



Localization Equations

- $C_i = \sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2}$
- $V_i = < \frac{x_0 - x_i}{C_i}, \frac{y_0 - y_i}{C_i} >$
- $n_i = (x_i, y_i) - D_i * V_i$
- $(x_0, y_0) = (x_0, y_0) - \frac{(x_0 - n_{ix}, y_0 - n_{iy})}{4}$