Microprocessors II Final Project:
Software Preliminary Design Review

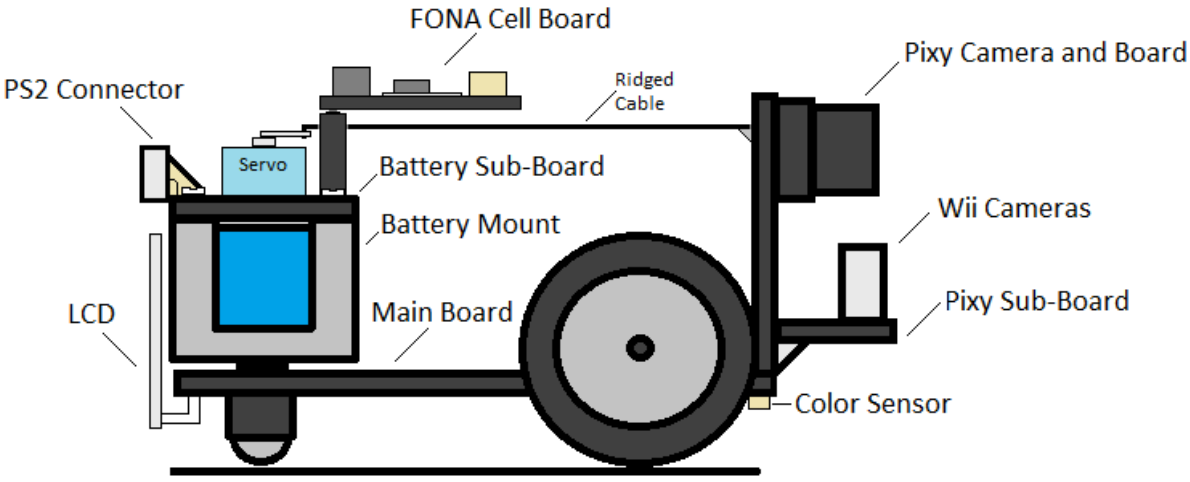
Ladder 42
Tyler Holmes & Broderick Carlin

Table of Contents

Abstract

The problem statement describes the need for a fully autonomous robot to compete head-to-head in a race to locate and extinguish a burning candle. The robot must be autonomous in the sense that it must be able to detect the beginning of a round, be able to locate a candle while avoiding obstacles, and be able to extinguish the candle within 30 seconds of it arriving at the candles location.The software to handle this will be tasked with operating on a 8-bit PIC18F and an external LPC4330 ARM processor. The two processors will both hold crucial roles in the execution of various algorithms needed to complete the task. The ARM processor will be responsible for handling our real time image processing, 3D model generation, and servo motor controlling. The PIC18F is going to be responsible for the more crucial algorithms such as drive motor control, IR camera interfacing, compass interfacing, and general logic algorithms responsible for control flow of the bot as a whole. The software for the PIC18F will be written in ANSI C, while the software for the ARM will be written in C++.

Figure 1: Board Layouts and Sensors



General layout of the major components of the Ladder 42 robot

Scope

The purpose of the software we are writing is to find and put out the candle's fire. In order to do this, it must successfully avoid obstacles, the walls, the rival robot, and recognize the candle when the robot reaches it. The robot has at its disposal a Pixy (CMUcam5) camera with an NXP LPC4330 dual core 204 MHz processor, two IR cameras from Nintendo Wii remotes (on the pixy sub-board), two compass/tilt sensors (one on the main board, the other on the pixy sub-board), a color sensor for floor color detection, and two mechanical whiskers if all else fails. Synthesizing all of this data into

a meaningful path finding algorithm is going to be the main problem. In this section, the general role of each sensor will be discussed.

Pixy

The Pixy camera's main purpose is object avoidance and, therefore, deciding where the robot can drive. This requires extensive custom algorithms and firmware for the ARM chip. Charmed Labs (the private company that released the CMUcam5 commercially through Kickstarter) luckily released all of the source code online, so we are able to make modifications. Unfortunately, the documentation is near non-existent. Despite this, we were able to implement a Prewitt filter for edge detection. Figure 1 shows the output of the sensor with the filter implemented on a simulated maze environment with a coffee cup, stapler, and colorful box in its path as well as walls on both sides.

Figure 2: Pixy Output

Output of the Pixy running the edge detection algorithm at an edge threshold of 250

From this data, we can flood fill the bottom of the screen, starting at the bottom middle, in order to determine areas we can drive, giving us an optimal path. The floor detection assumes that we always have the ground directly in front of us. This is a valid assumption because we will start facing away from any walls and obstacles will not be placed directly in front of us. From this state we can avoid obstacles and make sure not to face directly a wall through this method of object avoidance.

We can then transform the pixy's field of view into a top-down view of the ground through a mathematical transformation based on the current angle of the camera, the height of the bottom edge of the obstacle (this correlates to the distance away from the bot), and the wall edges. This array of drivable or obstructed cells of the floor then can be sent through UART to the PIC for further processing and pathfinding decisions based on all of the available sensors' inputs.

Wii IR Cameras

The Wii IR cameras will be used exclusively for locating the candle. They both communicate over I2C and send the location and intensity of the four brightest IR light sources in it's field of view. They are each on their own I2C bus because they both have the same hard-coded I2C address. These sensors are separated by a known distance, allowing us to interpolate the exact three dimensional coordinates of the candle. The cameras only look at the top half of it's field of view in order to avoid interference from any IR distance measurement coming from the rival robot.

Compass/Tilt Sensors

The compass sensor on the main bot is there to determine definitive cardinal orientation to make sure we keep driving forward and that we turn properly. This is a low priority double check sensor to make sure the other sensors, such as the cameras and the encoders, are giving us accurate data. The crucial compass sensor is on the pixy sub-board. This will measure the angle of the pixy and the Wii cameras which has a large effect on the math involved in interpolating the distance of the objects we are sensing. These sensors communicate on I2C as well, and are on separate buses.

Color Sensor

The Color Sensor on the bottom of the main board is used to determine what material is under the robot. This is only checked at as a double check to verify that the robot has actually reached a candle if it the other sensors are leading it to believe it has. The Color Sensor can be thought of as a simple one pixel camera that communicates on an I2C bus. Because there is a white circle that surrounds the candle, we can check for this circle to confirm if we have actually reached the candle or if we must keep moving/searching.

Whiskers

The mechanical whiskers are a last resort, interrupt driven, and fail safe precaution against running into objects. If the other sensors fail, we can fall back on the whiskers to make sure we don't collide with any obstacles.

Encoders

The encoders tell us how far we have traveled for path finding purposes. The encoders on the wheels are also interrupt driven. These aren't a very high priority, as we should be able to do all object avoidance and path finding without relying on these.

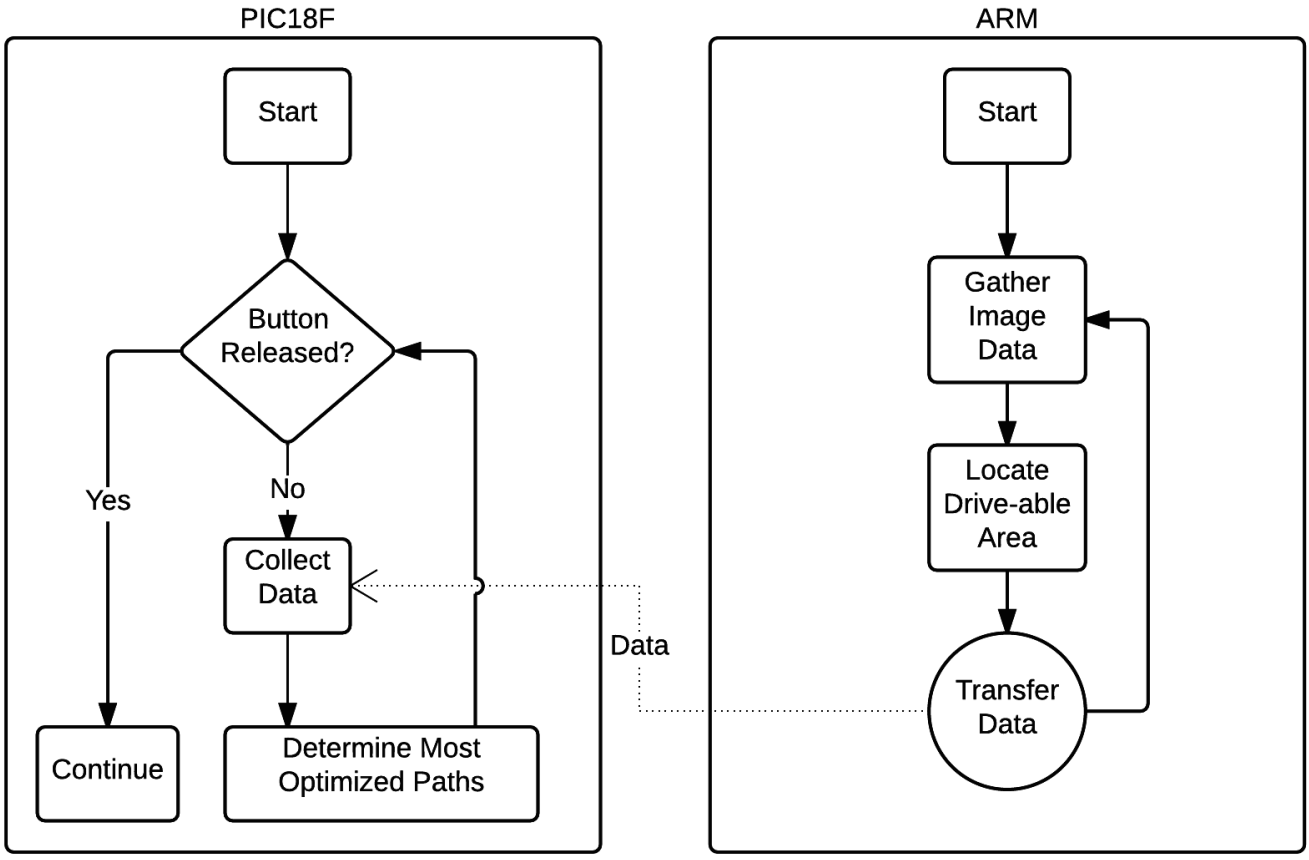
Control Flow

The software for the competition contains five unique states that determine what the software is doing during a specific time. The five states can be found listed sequentially below.

State 1

This initial state is indicative of the control flow taking place from the time the robot is powered up until the time the button has been released informing the bot of the beginning of a round. Flow Chart 1 below shows the first state flow for the PIC and the ARM processors. In this state the ARM processor is interpreting camera data to decipher a 3D model of the course seen and relaying this information to the PIC processor. This allows the PIC to begin determining possible paths to the candle before it even hears an alarm tone. By doing this the robot is able to immediately respond to the tone for faster response time.

Figure 3: State 1

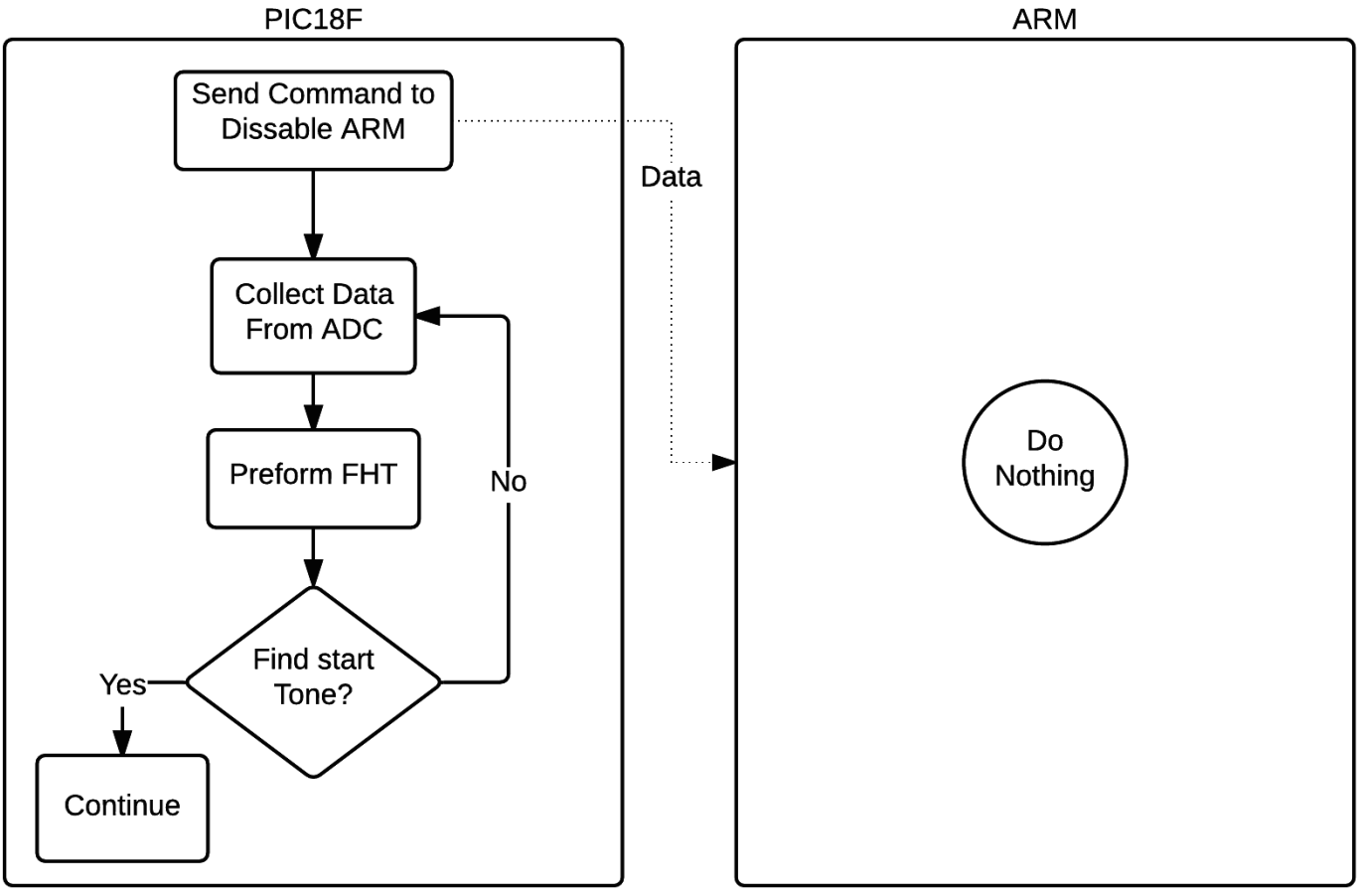


State 1 diagram, showing the initial state of the robot and the code flow

State 2

After the button has been released, the software enters its second state which processes an input from a microphone and waits for the starting tone. The incoming data is acted upon by a FHT, a derivative of the more common FFT function that has optimizations in place for working on real data. The FHT is taking place entirely on the PIC processor during this time and consumes all of the PIC processing. The results of the FHT are analyzed to determine if a start tone is detected. This is done by simply setting a threshold and seeing if the specific frequency bin exceeds this threshold for a certain length of time. During this time the ARM processor is doing nothing and simply waiting for the PIC to request more information.

Figure 4: State 2

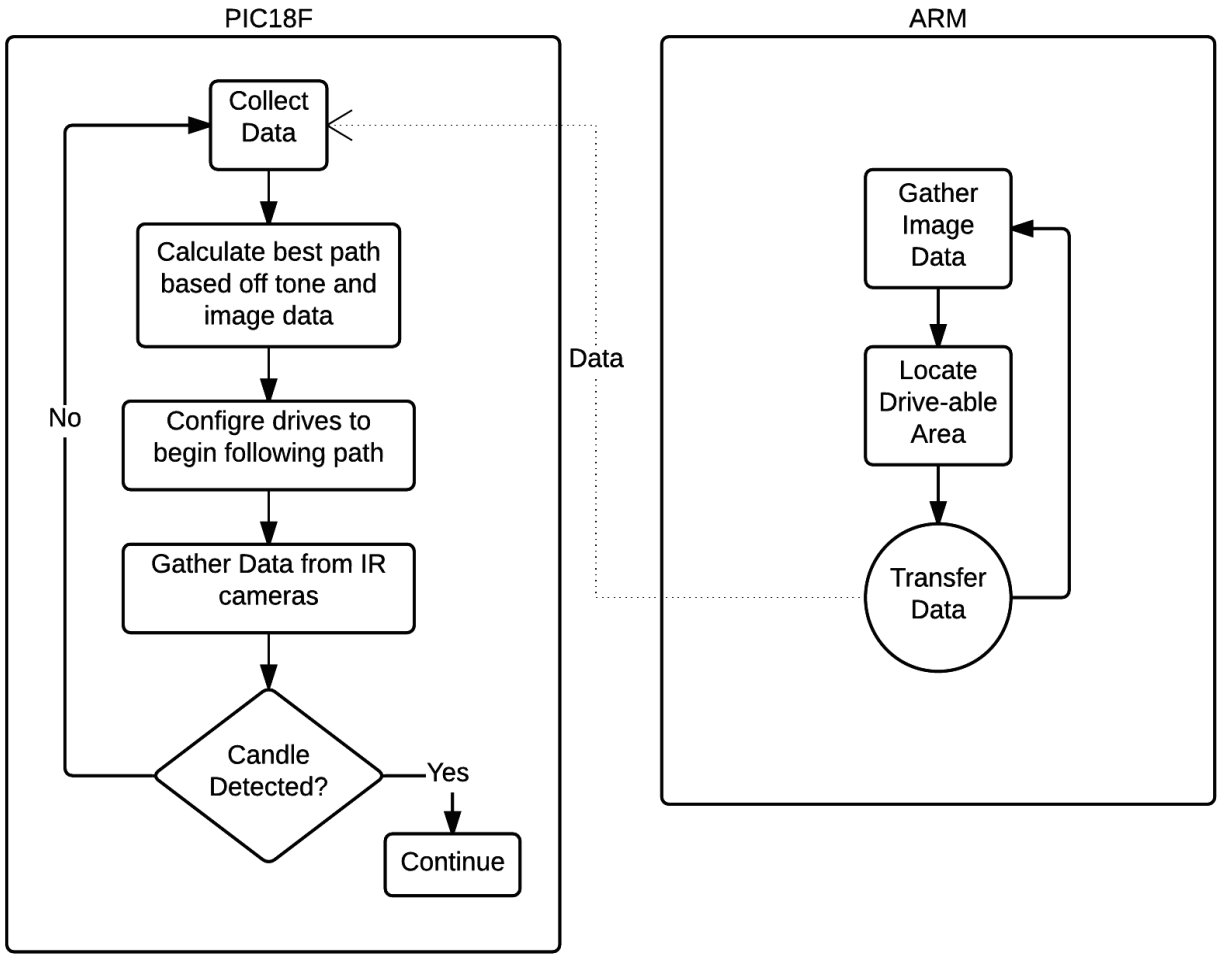


State 2 diagram, showing the frequency transform state logic

Stage 3

At this time the robot will be aware of the hemispheric location of the candle and the PIC will begin to request data from the ARM processor again. The information collected from the ARM determines the direction, speed, and target location for the robot. The PIC is responsible for actually interpreting this information and configuring the motor drives to act. At this point in time the PIC is also interpreting camera data from two IR cameras in an attempt to locate the candle. The feed from the two cameras is interpolated in order to generate a 3D position of any candle or IR source detected. Because many opponent robots are using IR distance sensors we need a way to decipher candles from IR LED's, this will be done by ignoring all IR sources in the bottom half of the camera images. The reason this solves the problem of LED vs. Candle is because the candle will be between 2-8in tall and the opponent robots have their IR distance finders mounted close to the ground.

Figure 5: State 3

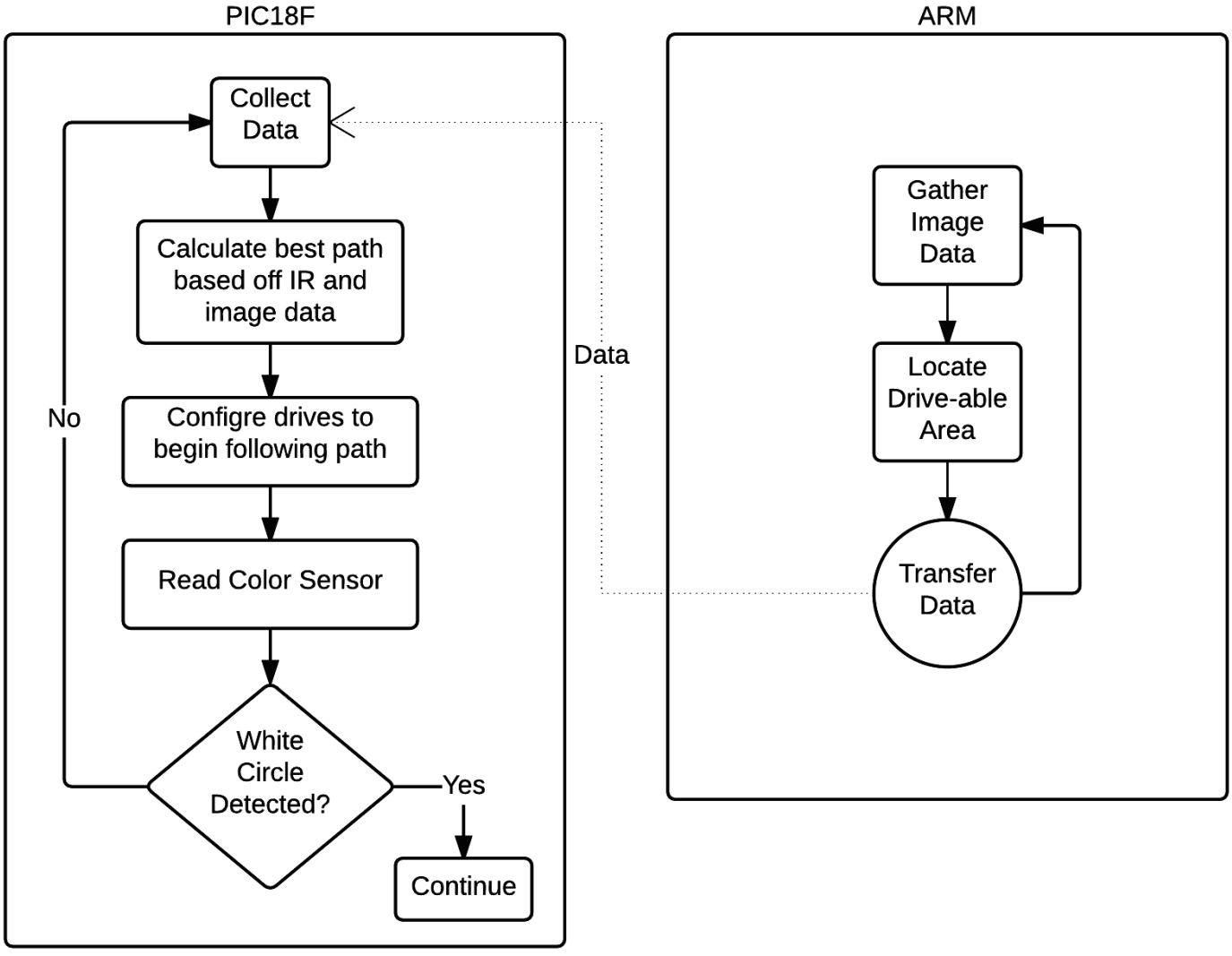


State 3 diagram, describing the path finding algorithm

Stage 4

At this point the robot believes it has found the candle and it now begins its search for the white circle that surrounds the candle. The PIC is pulling data from the ARM processor as well as the IR cameras but it also begins to ping the color sensor that is mounted on bottom side towards the front of the robot. This color sensor is in practice a single pixel camera that feeds back the information of the color it sees below the robot. This sensor is responsible for relaying information that determines if the robot has actually reached the candle or not, while the cameras are simply used for object avoidance and general direction information.

Figure 6: State 4 Diagram

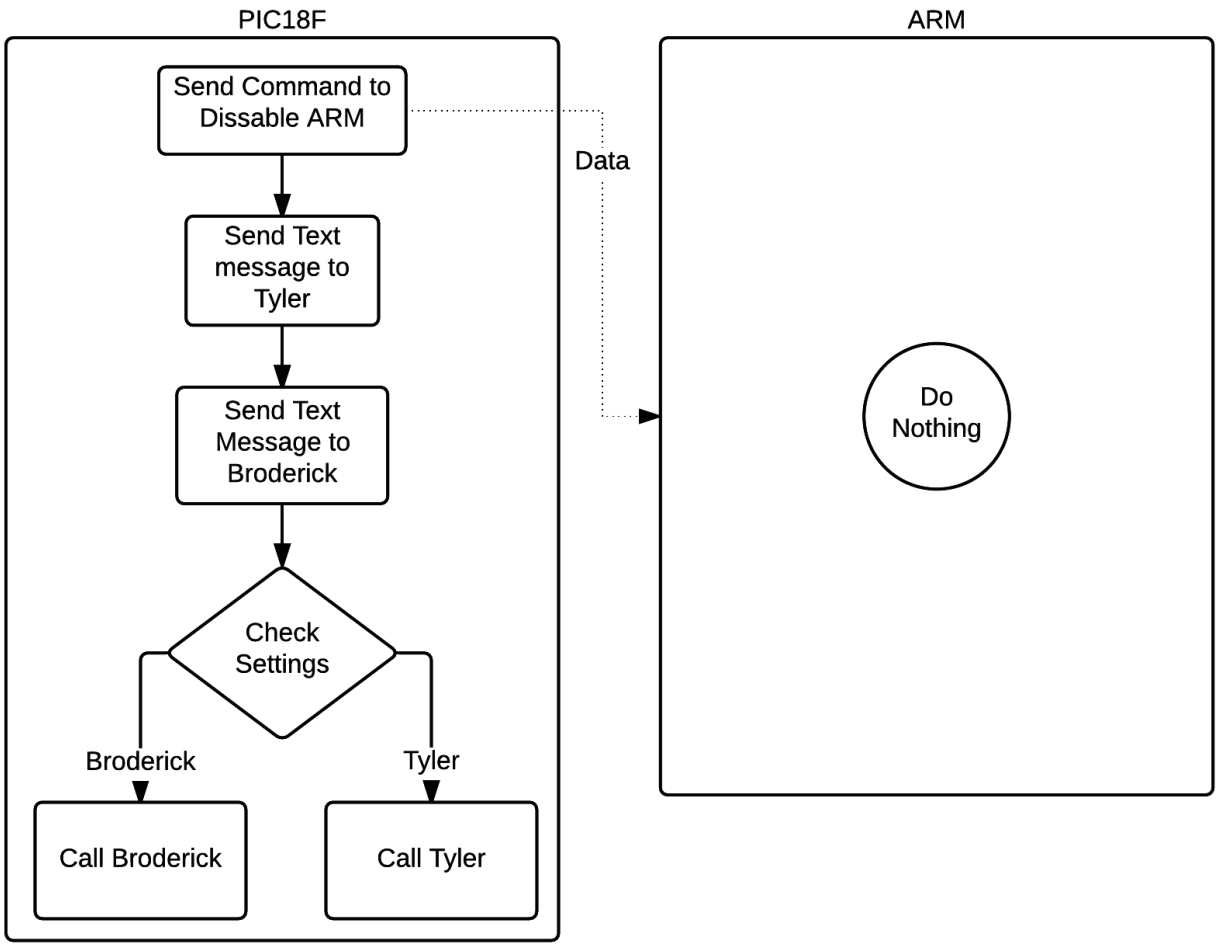


State 4 diagram, describing how the path finding logic changes when the candle is detected.

Stage 5

At this time the robot believes it has found the candle and as such it begins to try and communicate over the GSM network that it found the candle. Before making any attempt to communicate with team members, it first notifies the ARM processor that it should cease operation. The PIC then begins communication with the group members by first attempting to send a text to both team members followed by a call to one of the team members. This ensures that out of the three attempts, one of methods of communication is highly likely to go through. There is always the risk that the cell connectivity is lacking or other problems arise in regards to response timing, however we are unable to control these variables and must make the best out of what we have. Once the call is completed the robot assumes the competition was completed successfully and it simply ceases operation.

Figure 7: State 5 Diagram



State 5 diagram, showing what the robot does when the candle is found

References

Practical Algorithms for Image Analysis
By: Michael Seul & Lawrence O'Gorman

Pixy Forum:
http://www.cmucam.org/projects/cmucam5/boards/9

Pixy Technical Details:
http://charmedlabs.com/default/pixy-cmucam5/

Wii IR camera Interface:
http://www.instructables.com/id/Wii-Remote-IR-Camera-Hack/

Wii IR camera Interface:
http://cdn.instructables.com/FQL/9ZSM/FY3KH9ED/FQL9ZSMFY3KH9ED.MEDIUM.gi