|  |
| --- |
|  |
| Pathfinding Robot |
|  |
|  |
| **Victoria Zhong** |
| **5/12/2014** |

|  |
| --- |
|  |

# Table of Contents

Component List 3

Goals 4

# Components

4 - 940nm IR LEDs

2 - TSOP 38238 IR Reciever

4 - 220Ω Resistors or equivalent

2 - NEMA 17- size 12V stepper motors 350mA

2 - Large Kinect Wheel

1- Medium Kinect Wheel

1 - Protoboard

1 - Arduino UNO R3

1 - Adafruit Motorshield v2

1 - 12V alkaline battery pack

1 - 3V alkaline battery pack

# Goals

## Initial Design

The original design of the robot was suppose to be a small robot driven by the Arduino UNO that could traverse a maze using IR to detect obstacles. The robot would drive using differential steering with steppers, and IRs blinking to check if the path is clear. It would find its way out using depth-first search.

## Final Design

Unfortunately with the UNO, and Adafruit's Motorshield for steppers, small was not possible. The shield would not drive small steppers and the robot itself ended up the size of a robotic vacuum. Because of its size, there would be limited areas that would allow for the robot to even turn, so in the end, the lights did not have to constantly blink, but instead only do so when it has travelled approximately its body length. This also made it so that the Tree that mapped the maze also did not need to store steps, saving some memory. Because the UNO still only has 2kb of RAM, however, this robot will only work on very small mazes as nodes eventually take up a lot of memory.

The code for the bot uses the standard library, the library for the motorshield[[1]](#footnote-2), as well as a library I created for a "tertiary tree," because of the backtracking aspect of depth-first search, it was more of a graph. It creates a tree object that links to three tree objects and the route back to its parent object. When it is determined there is a path to be taken, if the path was not taken ie not visited, then the nodes representing each available path: left, right, and/or forward will be created.

The IRs are primarily driven by the Arduino using digital pins 2-5. Using the tone() function, the Arduino sends a square wave of 38kHz, the required carrier frequency to be picked up by the IR receiver.[[2]](#endnote-2)[[3]](#footnote-3) Because calculating microseconds takes time, and light travels faster than the Arduino can react, instead of calculating the microseconds it takes to return, the bot uses analogRead which is slower, but instead will calculate how strong the signal received was ie the farther it went before it bounces back, the weaker the signal. The problem with IRs is that due to noise from the previous blinks, the receiver may not be accurate and sometimes gives wildreadings as to the strength of the signal found. I have had problems where my hand was in front of the diode and the receiver read a full 1023 which is 5V, which is why the bot makes some weird turns sometimes. In retrospect, ultrasonic rangefinders would have been a better choice.

Both the IR diodes and the stepper motors are powered by the 12V battery pack powering the Arduino, while the receiver is powered by the 3V. The steppers will not run on anything less than 6 without shorting out and restarting the Arduino, which makes updating the code a pain as the bot is rolling around while it uploads.

1. https://github.com/adafruit/Adafruit\_Motor\_Shield\_V2\_Library/ [↑](#footnote-ref-2)
2. [↑](#endnote-ref-2)
3. https://www.sparkfun.com/datasheets/Sensors/Infrared/tsop382.pdf [↑](#footnote-ref-3)