Adrian Harminto

Custom Project: Rotatable PIR

Spring 2017

861-07-4419

Jeffrey McDaniel

Table of Contents

[**Introduction**](#_17dp8vu) **2**

[**Hardware**](#_3rdcrjn) **2**

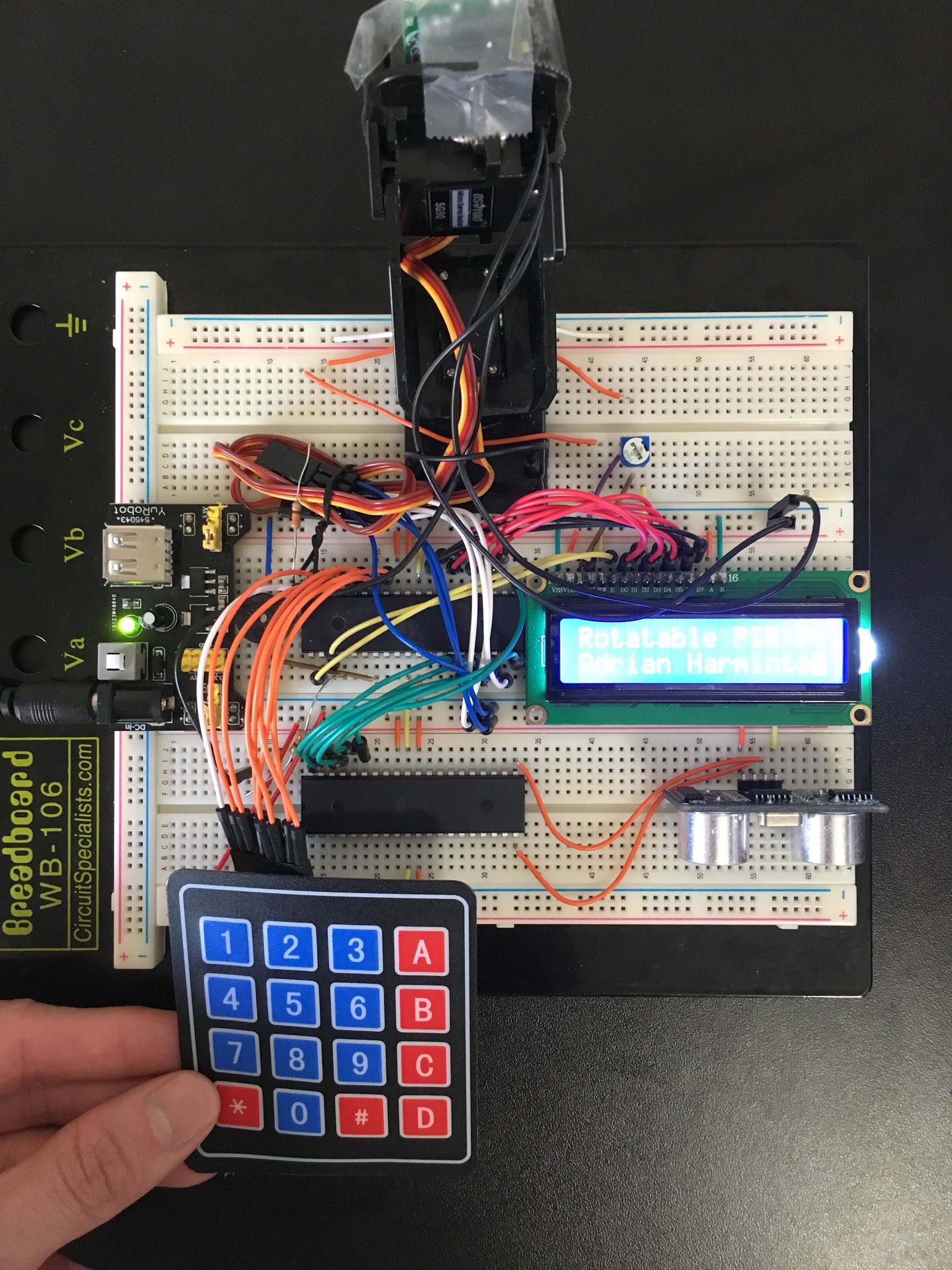
[Parts List](#_26in1rg) 2-3

[**Software**](#_35nkun2) **4-5**

**Probable** [**Complexities**](#_1ksv4uv) **5**

**Completed** [**Complexities**](#_1ksv4uv) **5**

**Link to Video Demo 5**



# Introduction

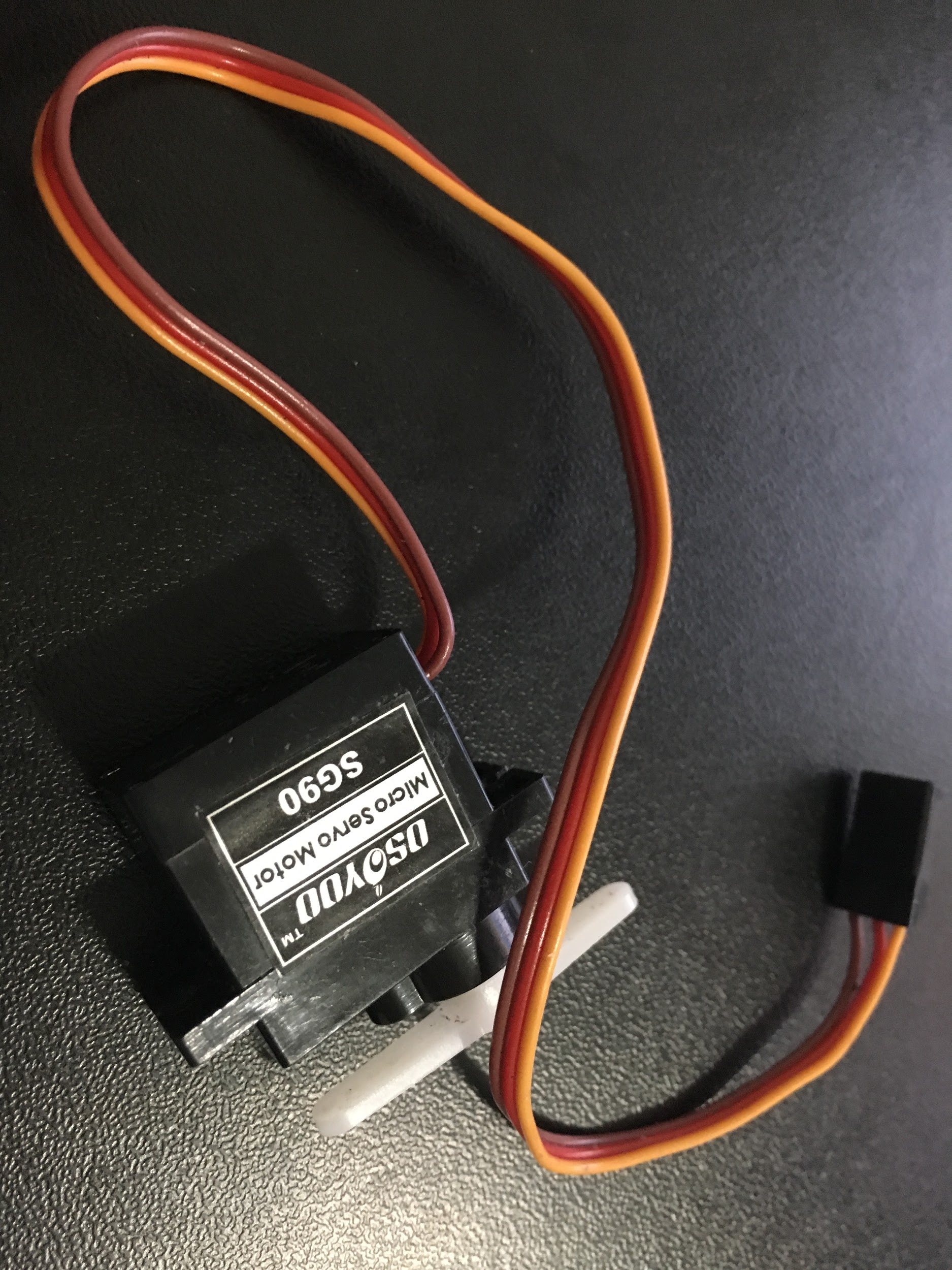
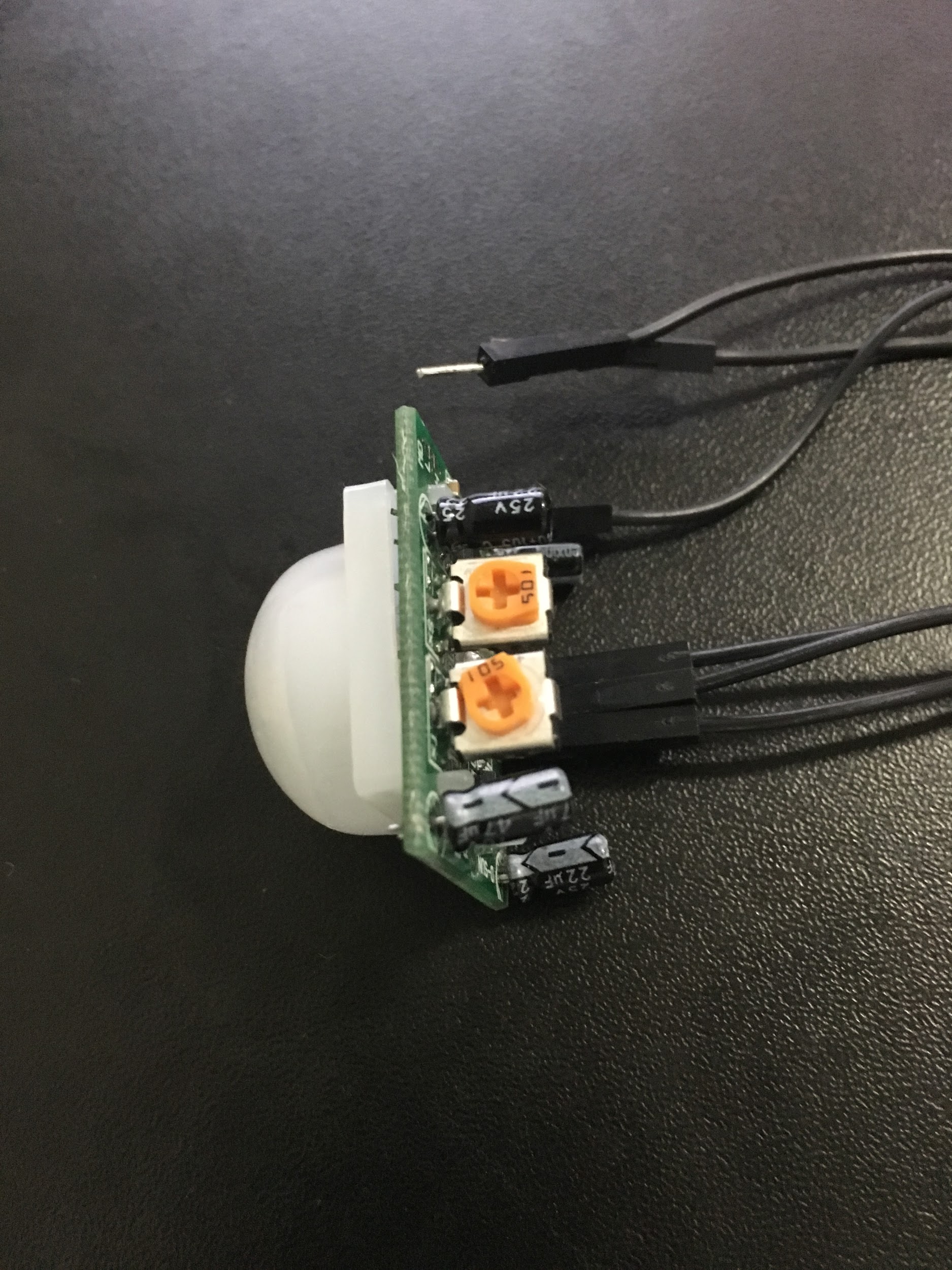
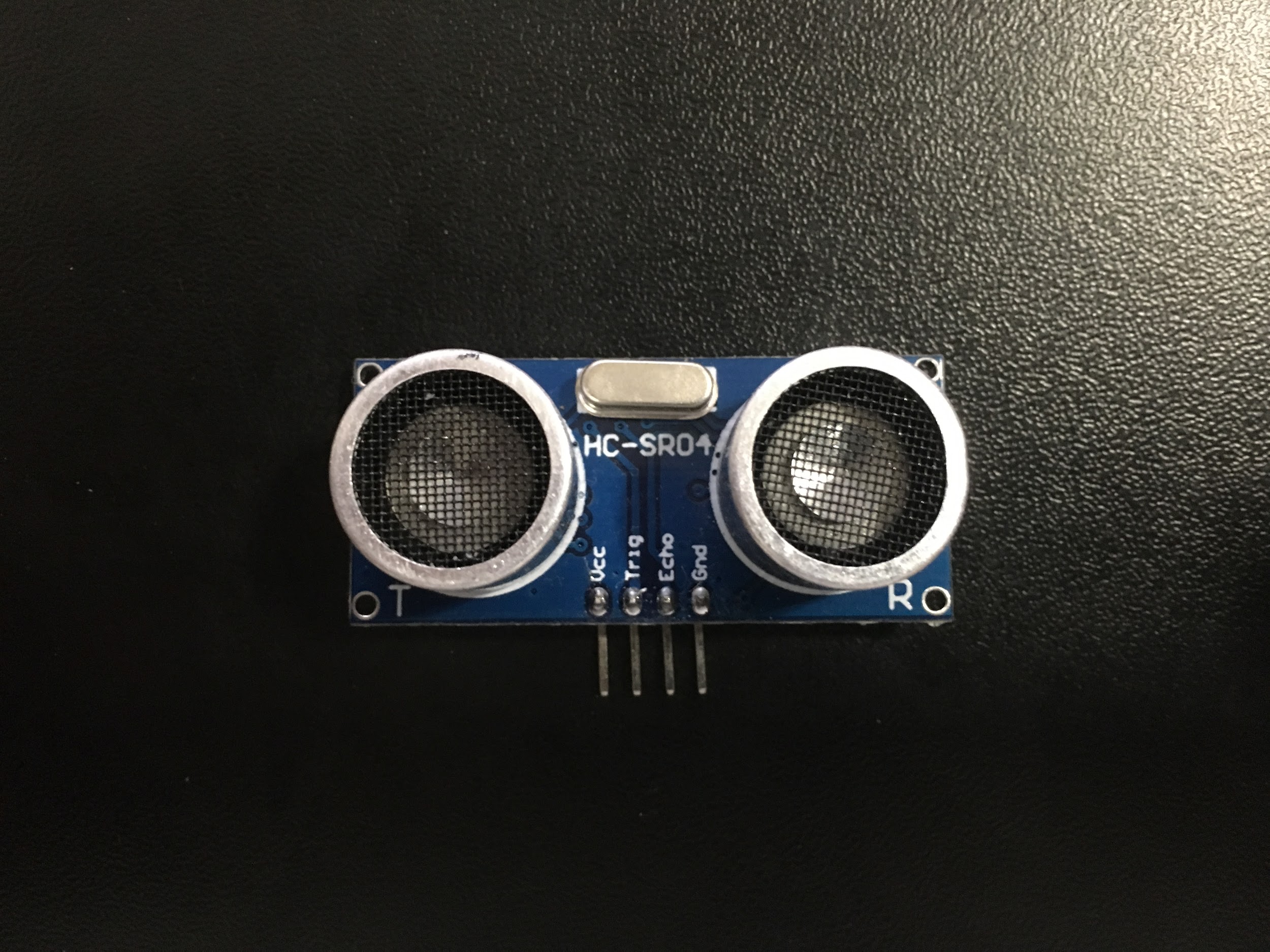
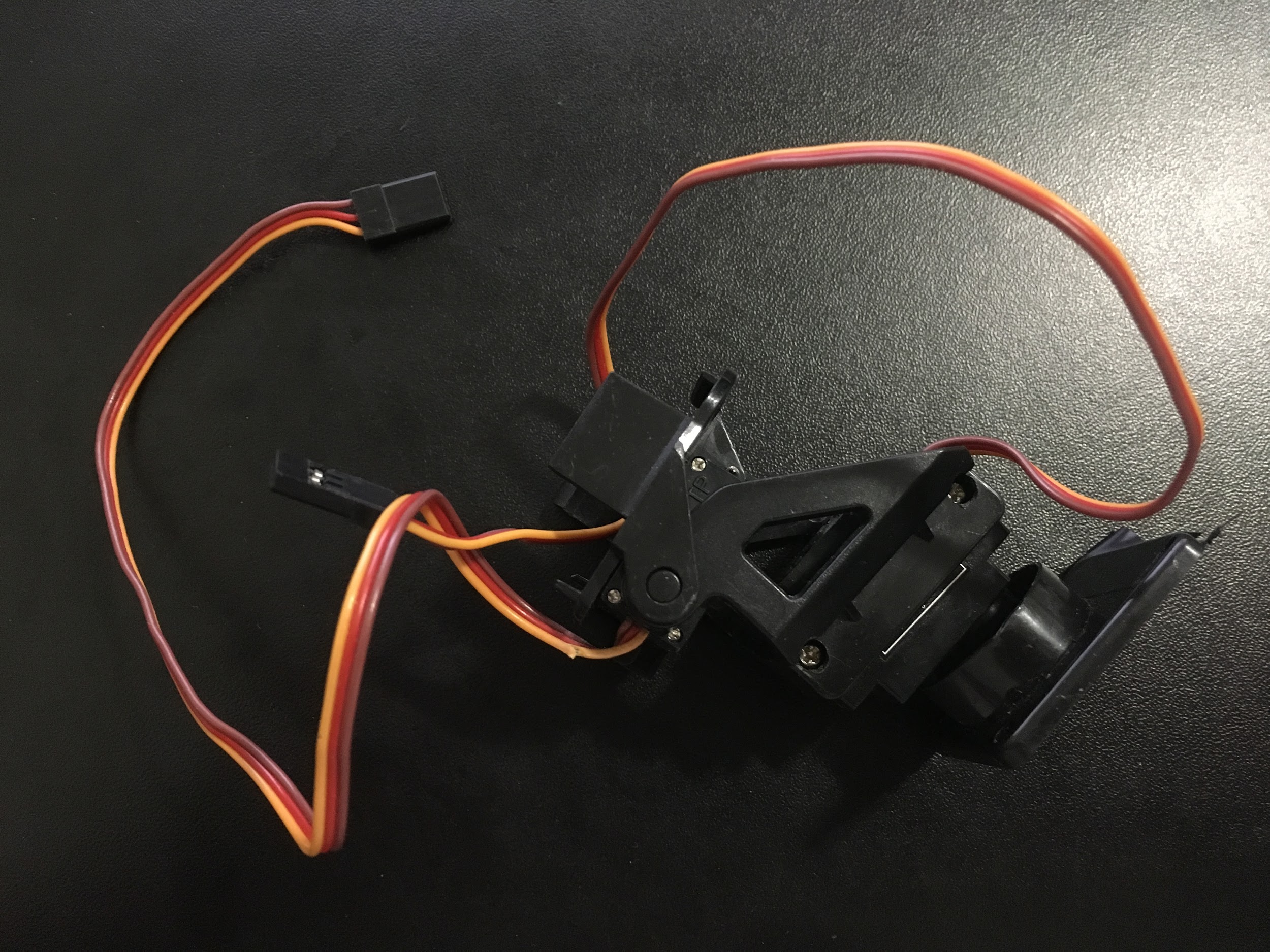
I’m building a rotatable motion sensor, controlled by multiple servo motors. This motion sensor will be rotatable to 180\* from left to right, and almost 180\* from up to down. Regarding to servomotor positioning, the user can freely rotate the servo motors either with a simple keypad buttons (up left down right middle), or with a simple hand gesture wave to the ultrasonar/distance sensor for better precision.

Regarding to "activate PIR sensor", the user will be able to either do "quick activate" for the PIR sensor to start detecting movement in its current position. The other mode will enable the user to input the timer for how long should the PIR sensor running, and prompting the user again to input the interval for how often should the servomotor rotate to scan left, middle, or right. When the timer has ended, it will display the amount of times PIR sensor was triggered by movements. (more details on Software)

# Hardware

## Parts List

The hardware that was used in this design is listed below. The equipment that was not taught in this course has been **bolded**. Text addition to most recent submission are highlighted in **RED**.

* Breadboard
* (2 pieces) ATMEGA1284
* Keypad/Numpad
* **(2 pieces) Servo Motor  
  **
* **(1 piece) Motion/heat sensor (D-SUN)  
  **
* **(1 piece) Distance sensor  
  **
* **“Skeleton part” of rotator (this picture have 2 servo motors attached to it)  
  **

# Software

Task scheduler is being used to implement the whole system, where the period is 50ms, and there will be a total of 6 tasks (and each was manually sorted according to its priority):

*[Highest priority]*

1. Keypad/Numpad [100ms]

2. PIR [100ms]

3. Alarm [50ms] (alarm needs to be 50ms bc it defines its tempo for alarm sound)

4. Ultrasonic/distance sensor [50ms] (Both this and servo motor are communicating each...)

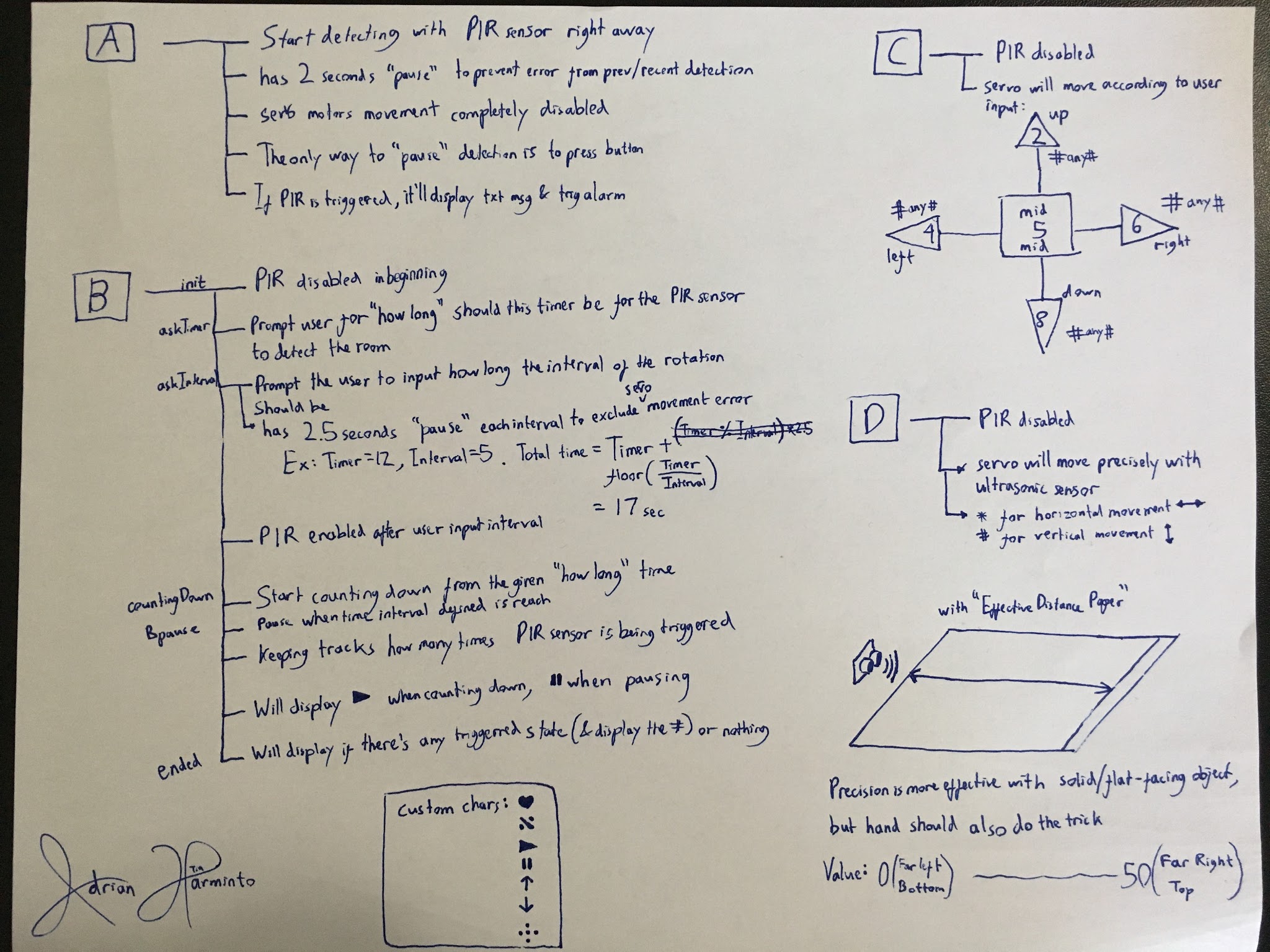
5. Servomotor [50ms] (... other in state "D", lower ms == smoother transition)

6. LCD [100ms]

*[Lowest priority]*

And as for the state machines, there are 4 major states that I define as "A", "B", "C", and "D", where each of them represents the button "A", "B", "C", "D" in the keypad.

Here is my hand written explanation of each states:



Specifically for keypad "B", it has its own state machine where it will start from: init, askTimer, askInterval, countingDown <--depending on interval--> pause, and ended. Currently, I don't define Keypad passing a character '\0' or null when it does nothing because it's causing some unnecessary problems. So to restart the B after it reached "ended" state, the user needs to go to "A" or "C" or "D", then click on B again to accomplish it.

# Probable Complexities

* To MOVE the servo motors & CALCULATE distance sensor without Arduino
* Using motion/heat sensor without using arduino
* ~~EEPROM to store the user input of time (apparently was unecessary since there is nothing to save)~~
* Algorithm to ADJUST the servo motors (in its skeleton) positioning with the distance sensor
* To print special/custom character from LCD screen for intro, requesting user for input, timer remaining, and movement detection effect

# Completed Complexities

* Have a full control of servo motors with its skeleton
* Have a full control of ultrasonic sensor with a specific range defined by CPU clock
* Using PIR motion sensor
* Custom characters for LCD: [dota logo], ♥, ↑,↓,→,←, ❚❚, ►
* Communications between 2 ATMEGA1284 (needed 1 more atmega chip for 16-bit timer)

# Link to Video Demo (click any of both)

[UCR EE/CS 120B Spring 2017 -- Adrian Harminto -- Rotatable PIR Sensor](https://www.youtube.com/watch?v=GwYmqF7stqE)

<https://www.youtube.com/watch?v=GwYmqF7stqE>