Trimester 2 Project

Thursday, December 5, 2019 8:25 AM

12/5/19 - Brainstorming Ideas and Recognizing the Problems - Step 1 Understand

- Problem - Can't take photos with everyone in the photo because someone needs to align the camera and not always somewhere to prop the camera
- Solution - Automatic photographer
  - Uses facial recognition to align
  - Multiple modes
    - On
    - Off
    - Timer(Infinitely many times you can set)
      - Have a button to raise or lower time
        - OLED board to display delay
  - Items needed most likely
    - Arduino (Rev3?)
    - OLED Board
    - Breadboard
    - Wires
    - Servos/Micro servos
    - Camera/webcam
      - Preferably capable of taking images as well
    - If time allows for it
      - Add a microphone option, if decibel threshold is reached = snap photo

1/5/20 Researching Past Solutions – Step 2 Explore

- Some cameras come with auto centring capabilities
  - All very expensive
    - Over $300
    - Some not very effective
    - Large mount often necessary
      - Not as portable

1/17/20 - Researching the Pan Tilt – Step 2 Explore

- 2 servos necessary
  - Attached to some sort of base/rack
  - One facing towards the ground
  - One attached to the base of the first servo going perpendicular
- Micro servos most likely possible
  - Advantages of using micro servo
    - Cheaper
      - Good for budget
    - Smaller
      - Could be advantageous for portability
  - Disadvantages of using micro vs normal size servo
    - Not as strong
      - Predicting it will be strong enough for a camera, but there's a chance a micro servo won't be
    - Shorter servo

Previously made pan tilt using 2 servos

1/23/20 - Researching different Arduinos – Step 2 Explore

Arduino Rev 3

- Operating Voltage: 5v
- Input Voltage recommended: 7-12V
- Input Voltage limit: 6-20V
- Digital I/O pins: 14
- PWM Digital I/O pins: 6
- Analog input pins: 6
- DC Current per I/O pin: 20mA
- DC Current for 3.3V pin: 50mA
- Flash memory: 32kb

Arduino Mega 2560 Rev 3

- Operating Voltage: 5v
- Input Voltage recommended: 7-12V
- Input Voltage limit: 6-20V
- Digital I/O pins: 54
- PWM Digital I/O pins: 0
- Analog input pins: 16
- DC Current per I/O pin: 20mA
- DC Current for 3.3V pin: 50mA
- Flash memory: 256kb

Pros of Mega vs Normal Rev 3

- More pins
  - More pins most likely won't be necessary
- More memory

- Could be useful depending on size of facial recognition program

Cons of Mega vs Normal Rev 3
- Much more expensive
  - Normal rev 3 $18 vs $32.75 Mega rev 3
  - Takes up over half of the budget which isn't very realistic to buy all the other components with the remaining $17.25

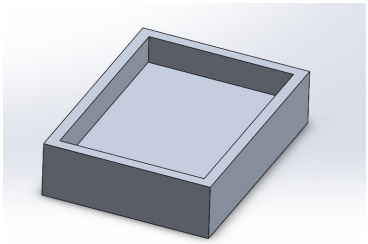
1/24/20 - Researching Measurements – Step 3 Define

Components	Length	Width	Weight
Arduino Rev 3	68.6 mm	53.4 mm	25 g
Possible Camera	n/a	n/a	n/a
Servo	29mm	n/a	n/a
Micro servo	38mm	n/a	n/a

Possible Cameras:
1. https://www.amazon.com/Cimkiz-Webcam-Built-Computer-Black/dp/B07GVF4M25/ref=sr_1_37?qid=1579889972&refinements=p_36%3A1253503011&s=pc&sr=1-37
2. https://www.amazon.com/docooler-Megapixel-Camera-Desktop-Computer/dp/B00OB88D2Y/ref=pd_cp_147_4/145-8739310-6714348?_encoding=UTF8&pd_rd_i=B00OB88D2Y&pd_rd_r=5e655b80-2485-43a1-a1a9-6cb1d3925ca8&pd_rd_w=z4fKC&pd_rd_wg=5Ff5x&pf_rd_p=592dc715-8438-4207-b7fa-4c7afdeb6112&pf_rd_r=ABCV7NZVH8NS86ETJKQG&psc=1&refRID=ABCV7NZVH8NS86ETJKQG

-Comes with built in microphone as well

1/28/20 Necessary Prototype of 3d Printed Component – Step 4 Ideate
- Problems
  - Not enough work finished to determine the dimensions
    - Specific shape to hold components once all together
    - Position of components are unknown currently
- Current design
  - A simple 80mm by 60mm rectangle that has 4mm thick offset walls that is large enough to hold an Arduino rev 3

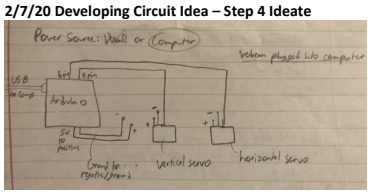


80mmx60mmx10mm Rectangle with 4mmx60mmx10mm walls

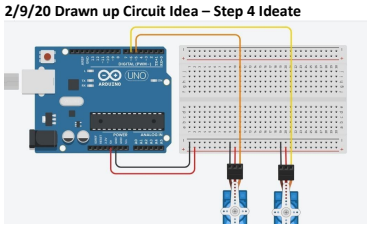
2/3/20 Components and Budget – Step 3 Define

Part	Purpose	Price
Arduino	Main computer controlling the servos	$18.00
2 MG90s'	Servos to rotate the camera upon a pan tilt	$6.00
Wires	Necessary for the circuit	N/A
Breadboard	Connect components (preferred to not have a somewhat permanent project because I will be reusing the parts for other projects)	Unknown Currently
USB cable type A/B	Connects the Arduino to the computer/laptop which is running the facial recognition code	$3.95
Webcam	Necessary to capture images/video for the code to process and track a face (Deciding on camera still)	N/A
USB Hub	Necessary to plug in a USB webcam and the USB type A/B cable from the Arduino to the laptop at the same time(My laptop only has 1 USB port which is a limiting factor)	N/A
Current Known Total	All Known prices currently to help estimate how much I have left to spend on necessary components	$27.95
Rough Total Estimate	Must be less or equal to $50	~$45

2/7/20 Developing Circuit Idea – Step 4 Ideate

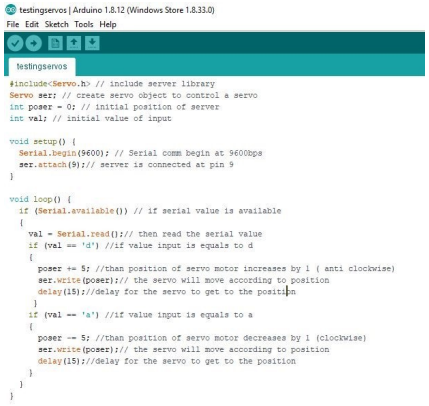


2/9/20 Drawn up Circuit Idea – Step 4 Ideate

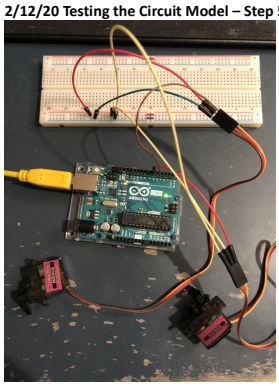


2/10/20 Testing Servos – Step 5 Prototype
Tutorial Used for Project: https://create.arduino.cc/projecthub/WolfxPac/face-tracking-using-arduino-b35b6b
Testing to see if the servos work properly

- Should be capable of 180 degree rotation
- Made simple code with Arduino communicating with Tera Term
  - Through Serial
  - Code is in "testingservos" folder
  - Tera Term is a third part program allowing Serial communication with an Arduino and keyboard interactions
- Singular Servo plugged into 5 ping, 5 volt, and grounded
- Tested each servo individually and they both worked



2/12/20 Testing the Circuit Model – Step 5 Prototype



2/15/20 Planning Facial Recognition and Tracking – Step 4 Ideate

- Planning on using OpenCV – Python for facial recognition
  - Code that is capable of identifying a face based off of embeddings, cascades, and nodes
    - Haar Cascade Created by Rainer Lienhart
      - Facial detection
- Make python code to make a display/window that shows the webcam input and the coordinates being output to the Arduino
  - Also include a centerpoint to check for alignment of a face once tracking
  - Send coordinates to serial to send to Arduino
    - Arduino code then take the coordinates from Serial and move the servos in the pan tilt accordingly to align the detected face in the middle of the video capture

2/17/20 Starting Code – Step 6 Choose

- Using OpenCV and Haar Cascade approach
- First downloaded OpenCV, PySerial, Numpy, and Haar Cascade
- Pip installed OpenCV, Numpy, and Serial through cmd
- Started coding python display to display Haar Cascade output
- Made the basic display window with a center point and + sign to split the screen into 4 quadrants
  - 500 by 500 pixel window

2/22/20 Finished Python Code for Display and Arduino Code– Step 7 Refine and Step 9 Implement

- Finished the Python code by adding constant feed of what the Haar Cascade is outputting and sending to the Arduino via serial
- Made all of the Arduino code today
  - Servos will start at 90 degrees
  - React to whatever coordinates are sent from the Python code through serial
  - Max and min degrees set
    - Min: 70 degrees for horizontal servo
    - Min: 95 degrees for vertical servo

    - Max: 179 degrees for vertical and horizontal servos
- Facial recognition works and outputs values fine, but servos won't move
    - Checked servos are still working with "testingservos.ino" and they both worked fine

servo.ino

```
#include<Servo.h>

Servo servoVer; //Vertical Servo
Servo servoHor; //Horizontal Servo

int x;
int y;

int prevX;
int prevY;

void setup()
{
  Serial.begin(9600);
  servoVer.attach(5); //Vertical Servo to Pin 5
  servoHor.attach(6); //Horizontal Servo to Pin 6
  servoVer.write(90);
  servoHor.write(90);
}

void Pos()
{
  if(prevX != x || prevY != y)
  {
    int servoX = map(x, 600, 0, 70, 179);
    int servoY = map(y, 450, 0, 179, 95);

    servoX = min(servoX, 179);
    servoX = max(servoX, 70);
    servoY = min(servoY, 179);
    servoY = max(servoY, 95);

    servoHor.write(servoX);
    servoVer.write(servoY);
  }
}

void loop()
{
  if(Serial.available() > 0)
  {
    if(Serial.read() == 'X')
    {
      x = Serial.parseInt();
      if(Serial.read() == 'Y')
      {
        y = Serial.parseInt();
        Pos();
      }
    }
    while(Serial.available() > 0)
    {
      Serial.read();
    }
  }
}
```

2/25/20 Problems with Servos – Step 7 Refine

- data = "X{0:d}Y{1:d}Z".format(xx, yy)
    - Have to change d representing a double type to f for a float, code originally worked in Python 2.7.4, but isn't working in 3.8.2
- Arduino.write(data.encode())
    - Both lines are causing a problem with communicating with Arduino
    - This line of code is preventing any signal through serial to get to the Arduino resulting in the servos not working
        - Code had previously worked on an older version of Python and OpenCV on my desktop (version 2.7.4 of Python)

2/26 Debugging Code – Step 7 Refine

- First going to attempt downloading an older version of Python and OpenCV to test current code with
- Downloaded Python 2.7 and OpenCV 2.1 on laptop
    - Pip cut off support for Python 2.7 if you don't already have the old pip installed
        - Must adapt the current code, can't use the old version
- Ran basic logic code to check if there is any signal going between the serial and reaching the Arduino
    - Current code there is a flashing light on the Arduino as if it is receiving information

2/27/20 Debugging and Pan Tilt Kit – Step 7 Refine

- Ran several steps of logic testing
    - Tested if python code was outputting values
        - Passed
    - Tested if circuit was getting power
        - Passed
    - Tested servos to see if they work independently
        - Passed
    - Tested Pyserial connection to Arduino
        - Passed

Root of problem still unknown, but ordered kit for pan tilt. Decided to buy a pan tilt kit after reconsidering the amount of time remaining and little progress on finding the root of the error in the code.

2/30/20 Finished Debugging and Testing – Step 10 Test

Error was a very simple fix, but took a long time to pin point. The Arduino code was attempting to parse an int while the output of the python code being sent through pyserial was a float so it was not receiving an integer it could interpret.

```
void loop()
{
  if(Serial.available() > 0)
  {
    if(Serial.read() == 'X')
    {
      x = Serial.parseInt();

      if(Serial.read() == 'Y')
      {
        y = Serial.parseInt();
        Pos();
      }
    }
    while(Serial.available() > 0)
    {
      Serial.read();
    }
  }
}
```

2/31/20 Found a box to fit the project in – Step 6 Choose
Planning on using an old iPhone box
Length: About 6 ¾ Inches
Width: About 3 ¾ inches
Capable of holding the Arduino and the bread board

3/1/20 Pan Tilt Arrived and Assembled – Step 9 Implement
Pan Tilt Assembly

- Took out all of the parts
- No instructions were included and the servo pins did not fit...
- I cut the servo pins to fit accordingly
- Used a hot glue gun to attach the horizontal servo to the base
- Cut the top servo's pin to fit into the notch because the pin did not fit and I had no pin that could fit
- Through trial and error, I found the correct screws provided to fasten the remaining parts together
- Filed down the camera on the bottom to get a flat surface to sit on the pan tilt

3/3/20 Finalizing Bill of Materials – Step 11 Reflect
Note: bought some sticky 3M squares to attach some parts

Part	Purpose	Price
Arduino	Main computer controlling the servos	$18.00
2 MG90s'	Servos to rotate the camera upon a pan tilt	$6.00
Wires	Necessary for the circuit	$0.49
Breadboard	Connect components (preferred to not have a somewhat permanent project because I will be reusing the parts for other projects)	$2
USB cable type A/B	Connects the Arduino to the computer/laptop which is running the facial recognition code	$3.95
Webcam HP KQ245AA Web Cam	Necessary to capture images/video for the code to process and track a face	$10.00
Lightning to USB adapter	Necessary due to my laptop only having 1 USB port when I need access to two	$4.00
Mounting Squares	Also referred to as sticky squares, needed to attach multiple parts on my widget	$1.14
Hot Glue stick	Needed to assemble pan tilt due to servo pins not fitting	$.18
Rough Total Estimate	Must be less or equal to $50	$45.76

There are screenshots of the prices in google drive

3/4/20 Finishing Project – Step 10 Test and Step 11 Reflect
Assembly

- Used 4 of the squares to stick the Arduino to the bottom of the iPhone box
- Then peeled off the back of the sticky paper on the breadboard and stuck it to the iPhone case next to the Arduino
- Then used an exacto knife to cut out a notch in the iPhone case for the USB type A/B to plug into the Arduino through
- Used two more of the sticky squares to attach the pan tilt to the top of the iPhone case
- Cut another hole on the lid of the iPhone box to feed the servo wires through to connect to the circuit
- Used another sticky square to attach the webcam to the pan tilt (7 used in total)
- Rewired the circuit and tested out the widget

Testing

- Camera was originally very twitchy so I added a sleep time on the update feed
- Added a sleep timer of 0.1 of a second on the python code so it wouldn't update as fast
- I had to recode some of the adjustments and min and max angles of the camera according to the field of vision of the camera
- The webcam has a FOV of about 30 degrees so adjusted the max, min, and adjustments to center accordingly

What I would do in the future

- I would spend more time tweaking the facial recognition to make it more accurate and not lock onto random objects falsely being identified as a face
- I would also make the camera capable of identifying multiple faces

and having a priority feature so it would prioritize the closest face or possibly have a mod to center between all of the faces

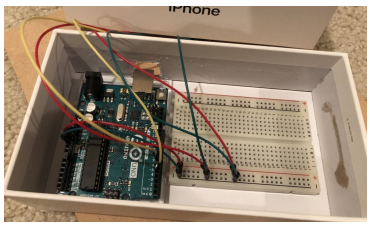


Photo of the inside and circuity of the widget

Completed Widget from the outside