CSE321 Term Project Report

<Video controller with Arduino and Ultrasonic sensor>

(Daly, Sankara, LangXian Xu)
SUNY at Buffalo: CSE 321, University at Buffalo
sankarad@buffalo.edu;

LangXian Xu Sankara Daly

Hand Gesture Recognition Control

1. Executive Summary

Real-time embedded system that allows the user to control the laptop and video control. Using the arduino device we were able to formulate a device which sense specific gestures that assist in navigating/ controlling a laptop.
Members of the team undertook this project after meeting at an earlier date and discussing what would be the most applicable project to create with arduino. We then formulated the project after viewing a video on youtube and showing interest in the ability to control a the video on a laptop without physically touching the keys of the laptop. Advantages to the the program is that it because, a hands free video playing device allowing the user the ability to control his laptop/ device playing the video from a farther distance. The sensor has the ability to enhanced with other features. However, the major benefit that the Hand recognition gesture control system offers is that it uniquely controls videos, unlike any other device currently in circulation.
This application will help individuals whom are seeking new technology to control, newly popularized streaming technology such as Netflix, Hulu, HBO go, YouTube among other similar video streaming services. The Hand Gesture Recognition Control System (HGRC) will be implemented on Youtube for testing purposes however will be attached to the device (i.e. laptop/ television). We plan to design the HGCS using an arduino uno and implement them using sensors attached to the device and modified to recognise the gestures made by the users. The users of the system can be any individual who is able to make the set gestures. Including children and adults in the users of the system and the ability of the system to pause videos moving in Real-time streaming or videos in real-time satisfies the realtime/embedded system aspect of the project. Additionally, the ability to instantly receive and interpret gestures satisfies this aspect of the project.

2. Project objectives

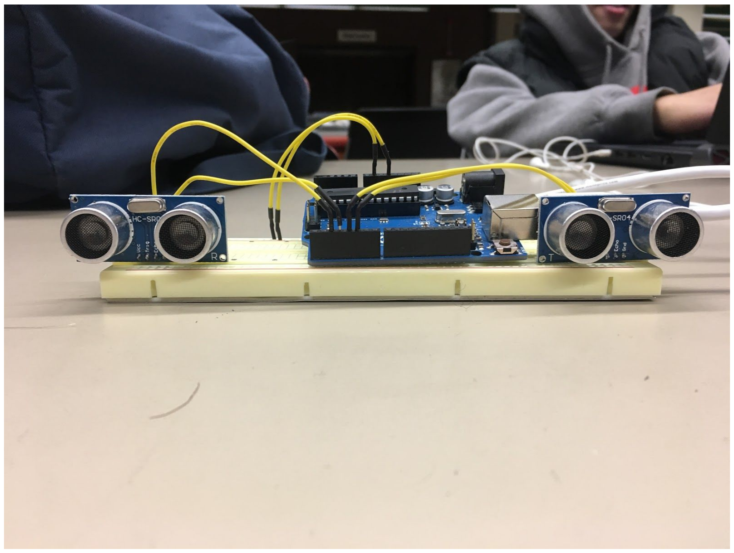
The objective of this project is to provide users a more convenient and intuitive way to control video playing options while watching a video on a laptop without physically touching the keyboard. The motivation behind this project was that when people watching videos on a laptop from a far distance, it's rather annoying to get up from the couch and walk all the way to the laptop to just pause or increase the volume or whatever. With a bit of researching, we decided to create a project using arduino and ultrasonic sensor to help solve the problem by detecting the distance of users' hands.
The goal of the project is to provide a multiple of necessary features/functions to give users an enhanced video watching experience with hands-free commands. Features of the project include but not limited to increase and decrease volume, mute, rewind, forward, pause/unpause, full screen, and skipping to next video.There features are all controlled by detecting the distance of users' gesture with the help of ultrasonic sensors.

3. Project Approach

The HGRC system will be developed in 3 phases; software development, hardware creation, modification stage. The prototype for the project will first be developed using a singular approach focusing on the ability to sense singular gestures with a single hand. This stage in testing the singular hand encompasses software development and designing of the code that will be replicated for an addition hand. Shortly after creation of the software, the hardware creation will be implemented due to the time restraint in receiving the parts after software there will be a delay between these stages. Hardware creation will encompass the testing of connections and creating the connection between the devices.
Finally, modification stage will occur when both platforms will be tested together to ensure that they work together. The project will be written and designed as a gesture control system calculating the distance to determine the gestures. Modeled after the use of radars to sense movement and the use of programming language for the communication between the arduino, sensors and the laptop/ device itself. The programming language will be used to store the desired gestures that will generate particular responses from the system. Making gestures not in the system will result in the program being unresponsive/ not functioning or completing the desired task . The overall approach to this project was follow the aforementioned steps to develop software then further develop the hardware components and finally modify and create a more complicated system encompassing more functionality.

4. Project Description

The Hand Gesture Recognition Control system is displayed in the diagrams below.

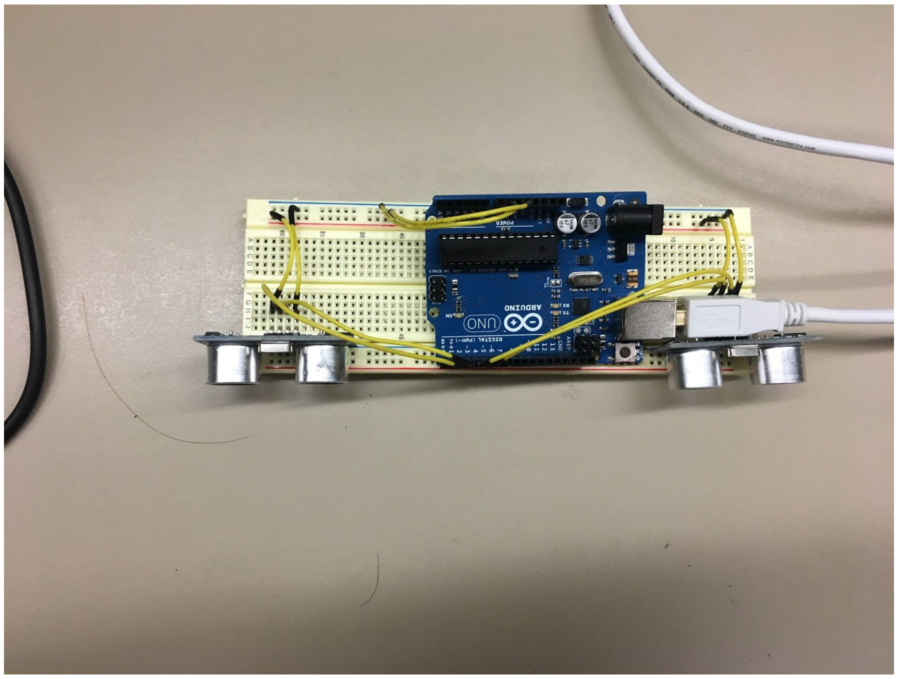
Figure 1:


Hands are human organs which are used to manipulate physical objects. Using this reasoning hands are used most frequently by human beings to communicate and interact with machines. Mouse and Keyboards are the basic I/O to computers and the use of both devices require the use of hands. Computers of this age provide humans with the impeccable able to type up to 60 words a minute among other feats. Hand gestures on contrast support us in our daily communication to convey our messages clearly. Hands are most important for the deaf and mute who depend on their gestures to communicate . If the computer had the ability to understand hand gestures is a huge leap in the field of human computer interaction. Every gesture has distinct feature which differentiates it from another command.
The real life applications of the Hand Gesture Recognition control system is endless from interacting with virtual object, in controlling robots to the ability for the disabled to communicate as well if the technology is modified. In this lead to the system which incorporates a real time communication between a user and their laptop to effectively control the settings and manipulate video play on a laptop on multiple websites, programs and applications on the computer. The project utilizes a programming language namely python and the arduino development environment to communicate within the system. The project as previously described in its introduction recognizes the distance between the user's hands and returns data giving the program a value. It is then our responsibility as the creators to manipulate the data returned to determine what actions are taken on the laptop. The project was successful in completing its original tasks namely, start/pause, fast forward, rewind, and volume control. Modifications were made to further increase functionality to incorporate the ability to mute and full screen allowing the device to effectively allow the full removal of contact with the laptop's keyboard by the user.

5. Design Details

The Hand Gesture Recognition Control system uses has multiple components as seen in the previously shown figure. The components of the manufactured device that allowed our system to flourish must be understood. The list of tools and technologies used in developing the project are:

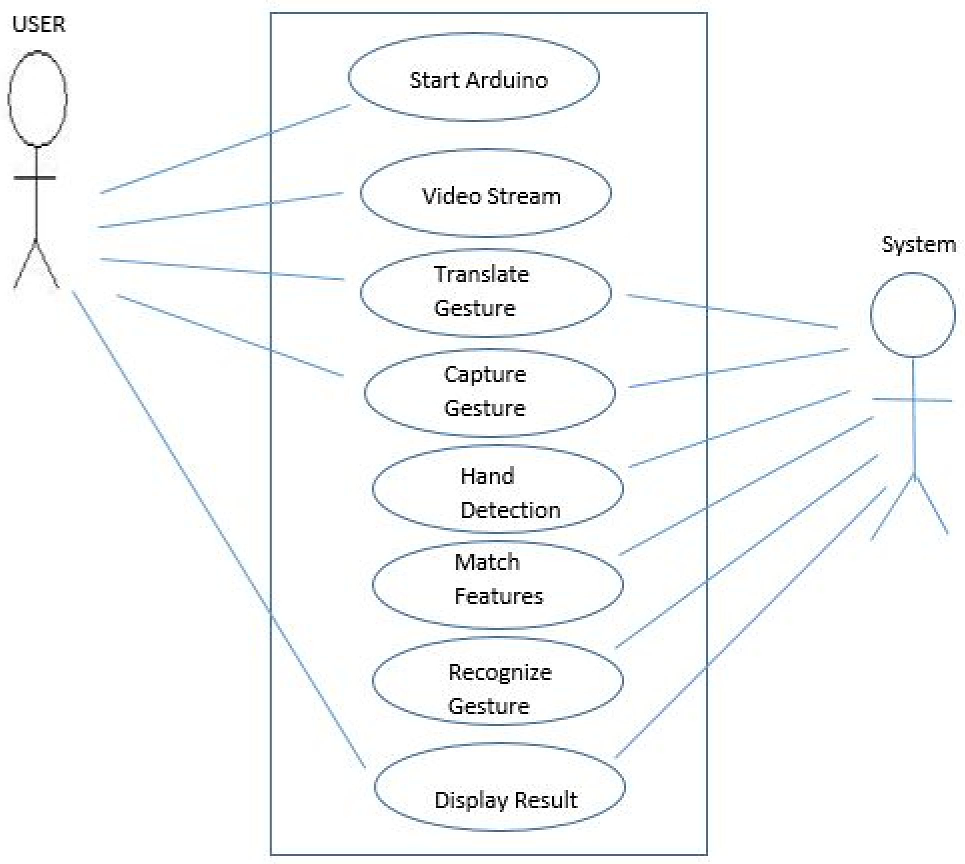
- Bread Board Circuit
- Python 2.7.9
- Arduino Uno
- Two Ultrasonic Sensors
- Arduino IDE
- 10 Independent wire connections



Further details of the Design Details is completed on the following page.

In further describing the design detail of the project additional figures of the Use Case Diagrams is below to visually detail the tools and technologies used.

Use Case Diagram for Video controller with Arduino and Ultrasonic sensor



6. User's Manual

In operating the device we have created, we will break down total usage into 4 steps:

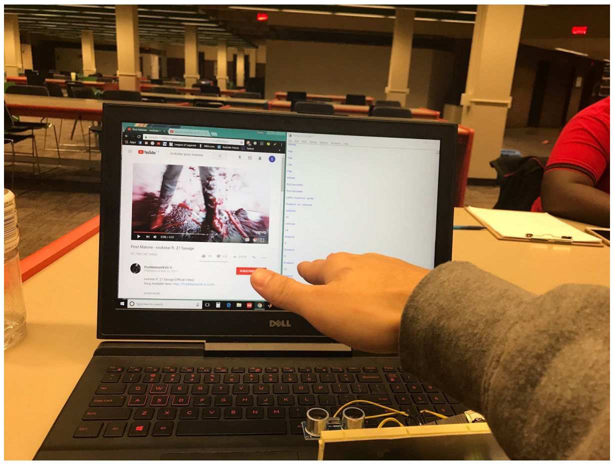
Step 1:Place Arduino device to the ceiling ensuring that there are no moving objects within 5 feet of your operating space. Open python and Arduino IDE to ensure that the device has been Connected.

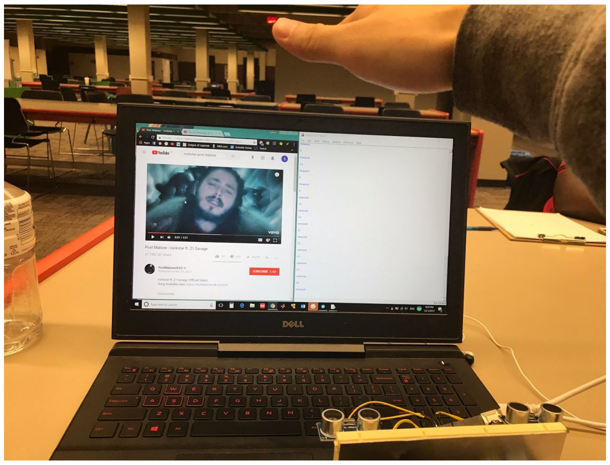
Step 2:After connecting and verifying the correct program . Wait for the start message " Starting"
To display within Python 2.7.9

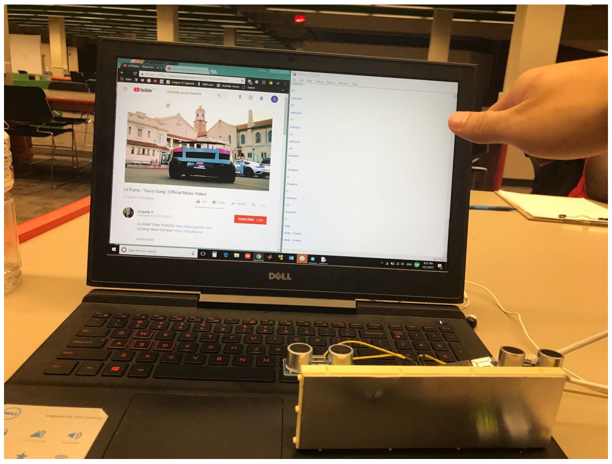
Step 3:The device will delay for five secs before beginning to read data through the sensors. The data displayed in the form of distances.

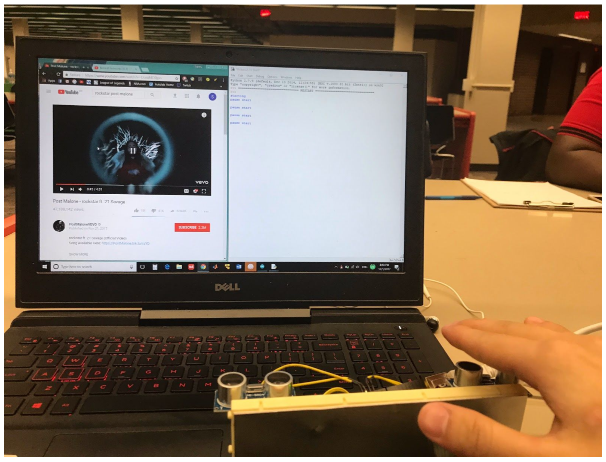
Step 4: Place both hands in front of the ultrasonic sensors to begin the video.

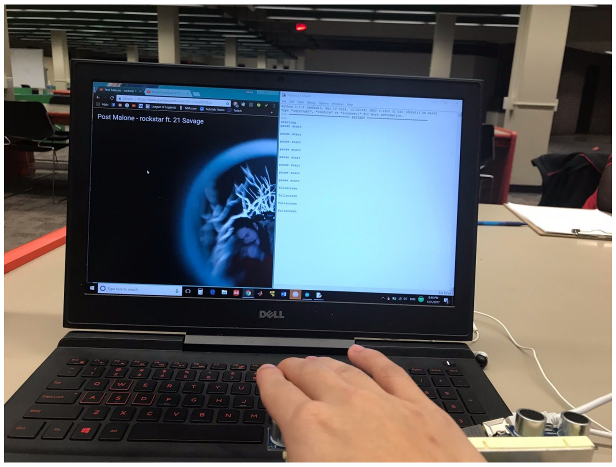
For further interactions within the videos below are a list of hand placements to consider for controlling the relevant operations while using the video streaming/ player.

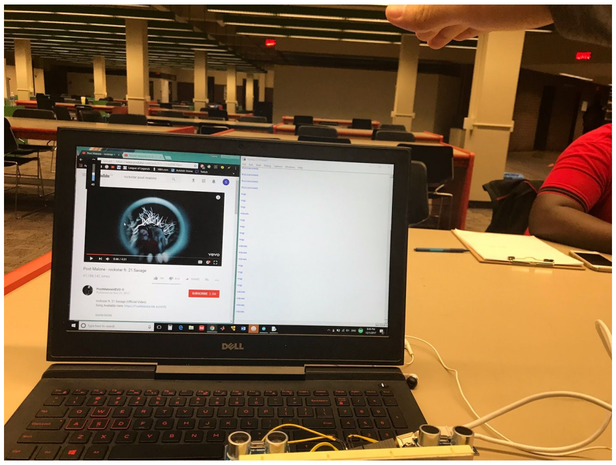

This figure shows fast forward.


This figure shows rewind.


This figure shows volume up.


This figure shows pause/unpause.


This figure shows full screen.


This figure shows volume down.

Mute: Place both hand within 10-15 cm of both sensor to mute or unmute volume.

Volume Up/Volume Down: Place right hand between 10-15 cm away from the right sensor to begin increasing the volume and 16-20 cm away to begin decreasing the volume . The instruction will display before execution.

Fast Forward/Rewind: Place left hand 15-20 cm away from the left sensor. Simultaneously, using the left hand. Wait for the message " Left- hand mode" to display. At this point begin moving hand closer to the Sensor to fast forward the video and pull left hand further away to rewind the video

Pause/Stop/Start: Place right hand, 0-5 cm away from the sensor to activate pause or stop the video. Repeat process to return the video to its initial start.

Full-Screen: Place left hand, roughly 0-5 cm away from the sensor to activate full-screen. Repeat process to return the video to its original size.

Next Video: Place one hand on the left sensor about 10-15 cm away, and swiping over right sensor to play next video.

During operating of the device each instruction is displayed in the module of python before activating. In order of list, the order of relevant instructions are : left-hand mode, right-hand mode, vup, vdown, forward, rewind and pausestart.
The list of prescribed can be adjusted for functionality, If the user deems that they would prefer it to be operational from greater distances. However, the operations listed above all require that there is no moving object at in the background while operating the devices.

7. Programmer's Manual

In order to run the program, one has to install latest edition of arduino IDE also python 2.7.9. After the python is installed, it would required to install a couple more libraries. The first library that's going to be installed is call Pyserial, this library would allow python to interact with the arduino. The second library that needs to be installed is call pyautogui, this library would allow the users to actually use the commands for the laptop. One can install pyautogui by typing this following command in windows prompt

```
python -m pip install pyautogui
```

After the libraries are correctly installed, one should be running the code with no problem.

Arduino:

1. The first part of code would be a function that's going to calculate the distance between the user and the ultrasonic sensor. The algorithm behind this code is to trigger the sensor to send a signal with a pulse, and then the sensor would return a signal after it detects an obstacle. The algorithm would then calculate distance by using the formula

$$\text{Distance} = \text{time} * \text{speed} / 2$$

Where time is the time it took for the signal to detect an obstacle and return back, and the speed is always 0.034. The code for the distance function is following:

```
void calculate_distance(int trigger, int echo)
{
  digitalWrite(trigger, LOW);
  delayMicroseconds(2);
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger, LOW);

  time_taken = pulseIn(echo, HIGH);
  dist= time_taken*0.034/2;
  if (dist>40){
    dist = 40;
  }
}
```

2. The second part of the code is an infinite loop which it keeps detecting signals from the user. Then it would performs whatever task the user wants depends on the command. One of the feature that we has is call control mode, which allows the user to use rewind or forward on a video. The logic behind this function is that after user hold its gesture at a certain distance for more than a second, the program would enter control mode and allow use to forward if the distance is close, and rewind if the distance is far. User can simply exit the mode by pulling his hand out. The code is following:

```
if(distL>=25 && distL<40){

   delay(800);
   calculate_distance(trigger1, echo1);
   distL = dist;
   if(distL>=30 && distL<40){
     Serial.println("left-control mode");
     delay(500);
       Serial.println("foward or rewind");
       delay(500);
       while(distL<40){
         calculate_distance(trigger1, echo1);
         distL = dist;

         if(distL<=12){
           Serial.println("foward");
           Serial.println(distL);
           delay(500);
         }
         else if(distL>12 && distL<40){
           Serial.println("rewind");
           Serial.println(distL);
           delay(500);
         }
       }

   }
  }
```

Python:

1. In order to interact python with arduino, the program would first need to detect where arduino comes from. It is important to know which port the arduino is connected to with the computer. The code for connecting arduino is following:

```
ArduinoSerial = serial.Serial('com3', 9600)
```

2. The second part of the code is an infinite while loop to detect message send by arduino and perform the task according to the message. For whatever message it receives, it would trigger the computer to perform such task by using the pyautogui library. The code is following:

```
while 1:

    command = str (ArduinoSerial.readline())
    print command

    if 'vup' in command:
        pyautogui.press('volumeup')
```

Complete Code for Arduino:

```
const int trigger1 = 2;
const int echo1 = 3;
const int trigger2 = 4;
const int echo2 = 5;

long time_taken;
int dist, distL, distR;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(trigger1, OUTPUT);
  pinMode(echo1, INPUT);
  pinMode(trigger2, OUTPUT);
  pinMode(echo2, INPUT);

}
void calculate_distance(int trigger, int echo)
{
  digitalWrite(trigger, LOW);
  delayMicroseconds(2);
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger, LOW);

  time_taken = pulseIn(echo, HIGH);
  dist= time_taken*0.034/2;
  if (dist>40){
     dist = 40;
  }
}

void loop() {
 // put your main code here, to run repeatedly:

    calculate_distance(trigger1, echo1);
    distL = dist;

    calculate_distance(trigger2, echo2);
    distR = dist;

    if(distR > 8 && distR <= 15 && distL > 8 && distL <= 15){
      Serial.println("mute");
      delay(300);
    }
    else if(distL>=25 && distL<40){

      delay(800);
      calculate_distance(trigger1, echo1);
      distL = dist;
      if(distL>=30 && distL<40){
        Serial.println("left-control mode");
        delay(500);
          Serial.println("foward or rewind");
          delay(500);
          while(distL<40){
            calculate_distance(trigger1, echo1);
            distL = dist;

            if(distL<=12){
              Serial.println("foward");
              Serial.println(distL);
              delay(500);
            }
            else if(distL>12 && distL<40){
              Serial.println("rewind");
              Serial.println(distL);
              delay(500);
            }
          }

      }
    }

    else if(distR <= 5 ){
      Serial.println("pause start");
      delay(500);
    }
    else if(distR > 20 && distR < 30){
       Serial.println("vup");
       delay(500);
    }
     else if(distR > 30 && distR < 40){
       Serial.println("vdown");
       delay(500);
    }
    else if(distL <= 5){
      Serial.println("fullscreen");
      delay(500);
    }
    else if(distL > 15 && distL<25){
      delay(800);
      calculate_distance(trigger2, echo2);
      distR = dist;
      if(distR > 15 && distR <25){
        Serial.println("next video");
        delay(2000);
      }
    }
}
```

Complete Code for Python:

```
import serial
import time
import pyautogui

ArduinoSerial = serial.Serial('com3', 9600)

time.sleep(2)

print ("starting")

while 1:
```

```
command = str (ArduinoSerial.readline())
print command

if 'vup' in command:
    pyautogui.press('volumeup')

if 'vdown' in command:
    pyautogui.press('volumedown')

if 'pause start' in command:
    pyautogui.press('space')

if 'fullscreen' in command:
    pyautogui.press('f')

if 'foward' in command:
    pyautogui.press('right')

if 'rewind' in command:
    pyautogui.press('left')

if 'mute' in command:
    pyautogui.press('volumemute')

if 'next video' in command:
    pyautogui.hotkey('shift','n')
```

8.References:

Youtube:

https://www.youtube.com/watch?v=dSIEuxMHVSI

Ultrasonic Sensors for beginners:
https://www.youtube.com/watch?v=AMDb45oBub4

https://circuitdigest.com/microcontroller-projects/arduino-ultrasonic-sensor-based-distance-measurement

Circuit Digest:

https://circuitdigest.com/microcontroller-projects/interfacing-arduino-with-vpython-creating-graphics

https://circuitdigest.com/arduino-projects

https://circuitdigest.com/microcontroller-projects/control-your-computer-with-hand-gestures