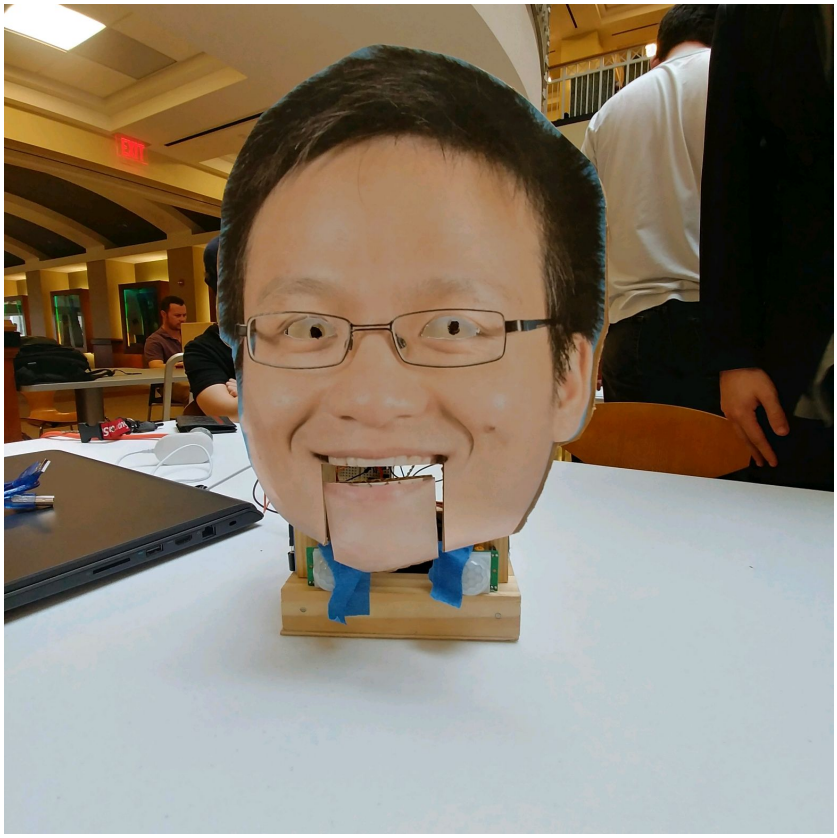
Animatronic Face: Professor Gao



Brandon Cox
Dept. of Mechanical Engineering
San Antonio, TX, USA 78249
yqy154@my.utsa.edu

Andrew Kim
Dept. of Mechanical Engineering
San Antonio, TX, USA 78249
sfr468@my.utsa.edu

Kyle Kinsey
Dept. of Mechanical Engineering
San Antonio, TX, USA 78249
yhj962@utsa.edu

Submitted as part of project for ME 4543 Mechatronics, Fall 2018

TABLE OF CONTENTS

ABSTRACT

The purpose of this project is to demonstrate the tools and concepts learned in the mechatronics lab in a fun application to ascertain and to solidify these concepts. This is ment to demonstrate the core concepts of mechatronics and the application of constructing a simple animatronic head. The requirements for this head are to have two different types of sensors record two different kinds of interactions and be programmed to have a response to each of these interactions. Additionally the project requires audio of the professor in some form in response to a stimulus and the professors likeness designed into the device. After several weeks of work all requirements were met for this project.

Section 1: Literature review

Face Design 1:

The animatronic face studied consists of a Halloween themed talking (moving jaw) skull and pumpkin face as shown in Figure 1. In the design, the creator opted to utilize RC Servos for the functionality. While the Servos controller circuitry is powered via USB, the functionality/movements from the controller is powered by a 5 volt external battery. The controller utilized for the design is the Parallax USB Servo Controller which serves as a interface between the servo motors and computer. The audio is recorded and edited utilizing Audacity. Afterwards, Visual Show Auto, which is a primary feature for its display, is used to synchronize/choreograph the servo movements with the sound. The creator opted to use a styrofoam base with hollow insides for its facial feature/base. Both faces follow the concepts stated above.

Figure 1. Halloween Themed Animatronic Face

Face Design 2:

For this face it was decided to look for something that was minimalistic to better analyze and interpret the design of the face. For this head only the eyes, eyebrows and mouth are moved to give a illusion of talking and emotion. The eyebrows are the simplest in that they are just two seperate motors that rotate to a set angle for a given phrase or emotion. By tilting above the 90 degree or 0 degree x-axis. Emotion in the eyes in conveyed by using four servo motors. There are two in each eye. One motor controls each axis. This allows the motor to move to a set x-y coordinate given by the controller. Lastly, the mouth is controlled by a motor that raises and lowers the mouth. Overall, this design depends heavily on having multiple motors run

each component of the face with a total of 7 motors. This has the downside of needing a lot of power to use, however the design is simple.

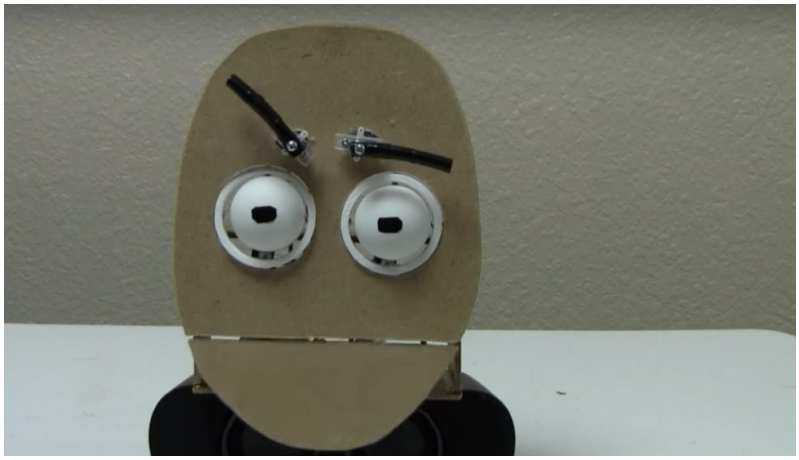


Figure 2: Pyroelectro animatronic head

Face Design 3:

The third animatronic face studied is the Fritzbot from Kerkits. It is a simplistic design but with a large range of movements. This design provides good inspiration for each of the possible degrees of freedom we would pursue, for example the model can be studied to learn how they solved some of the motion problems that Geoffrey Toombs ran into. This model is built from a set of interlocking laser cut pieces of plywood with space to mount each of the servos and both the arduino and power supply. Each of its movements is powered by a mini servo except for the tilt and turning of the head which are full sized servos. The animatronic is programmed by a custom made C-Sharp program that writes to an arduino. The power required is supplied by an AC adapter because the usb connection cannot provide enough to power all of the servos. The use of a custom program makes this example less useful to learn from a programing standpoint but the large number of movements and relatively simple actuation of each of those makes this a good mechanical example.

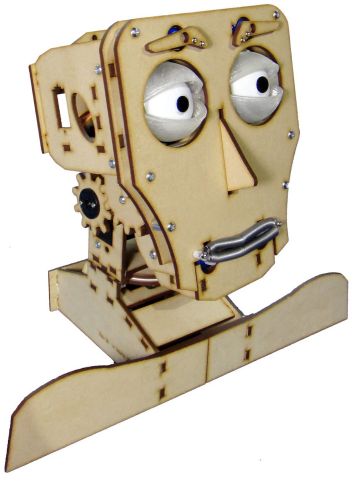


Figure 3: Pyroelectro animatronic head

Section 2: Brainstorming (initial planning)

This head will be able to say different things in response to positioning and orientation of the head. This will be done by using a two set infrared positioning sensors, and a gyroscope/ accelerometer. The two infrared sensors will indicate if there is a object or person in each of the zones that the sensors read. Based on which one is activated, a servo will rotate the eyes in the required direction. Eyes will be only orientated on the x axis for simplicity. Additionally a verbal response will be given if both sensors receive a reading. If no response both eyes will wonder randomly from side to side. A tilt sensor will indicate if the head is flipped and the head will make verbal responses based on its orientation. The voice will be controlled by a mp3 shield which will be given a indicated signal from the arduino to play a set mp3 file. Mouth will move in response using another servo. Materials used will be a wooden chassis, with a clear plastic frame to protect the enclosure. The head itself will be made of cardboard with the image of professor Gao adhered to the surface. The chassis will be made from scrap wood and the face will be placed on cardboard. Arduino will be powered by a power cord connected to a wall. No computer will be needed to be plugged into the device. We intend to have professor gao say “Welcome to utsa department of mechanical engineering.” and “Ahhh, please put me back down.” These will be recorded and uploaded to the mp3 shield.

Section 3: Supporting structure

The face and supporting structure is to be a combination of cardboard and wood using screws and wires to hold the pieces together. The eyes will only operate on a x axis and will be controlled using a servo that moves a wire left and right. This will angle both eyes in the same direction specified by the servo as seen in figure 4.

Figure 4: Chassis with initial servos and eye mechanism

The wooden frame is supported by 3 supports to give a operating space to insert internal components such as the board and the mp3 shield.The face will attached to cardboard since it is a cheap material that can be easily altered or replace in the event of errors. Additionally it is easy to attach print outs of professors faces. A wire attached to back servo will hold the mouth aloft and when the servo is activated the mouth will bounce up and down. This movement should be fast since the wire is giving the servo motor a large radius which increases the linear velocity of the output. Lastly the speaker is placed in the center of the base and hot glued to cardboard which is attached to the base. This holds the speaker securely in place. The mechanical components and electrical will all be attached to the frame using hot glue. This also includes the face and the picture.

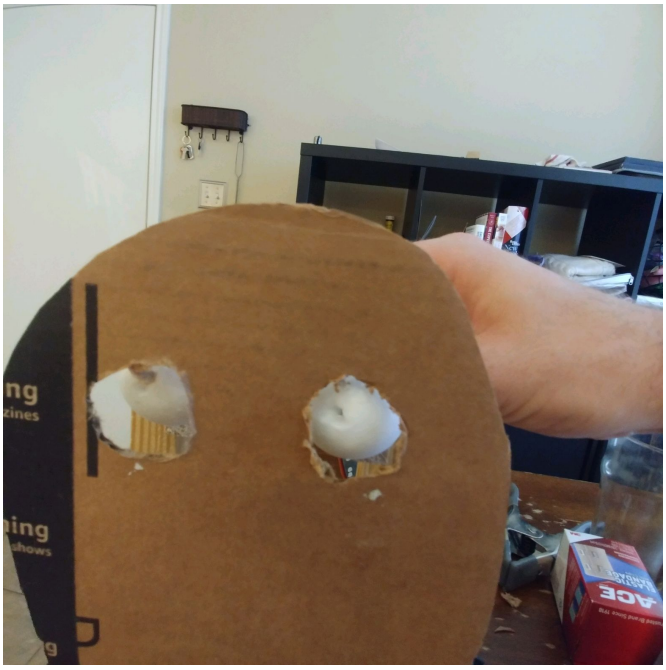


Figure 5: Front of face structure

Figure 6: Face with chassis

In this initial phase since, most mechanical and electrical components are not attached the frame is held together with clamps in some areas in the event that the pieces need to be adjusted at later steps.

Section 4: Joints and motors

This device will only use two servo motors to power and control all the actions which will be both eyes and the mouth. Metal wiring will connect one servo to both eyes and the other two the mouth. The metal wire will connect both eyes and connect to the motor in a T shape and the middle of the T is held by a pivot. When the servo moves left the top of the T shape of the metal wire moves the eyes right. If the servo moves right the eyes move left. The mouth is held up by a single metal wire that connects to a servo in back. The length of the wire which is approximately 15 cm amplifies the rotational motion of the servo which creates fast movements for the mouth. This action is demonstrated by the equation $\omega r = v$. By rotating the servo from a upper position to a lower the mouth moves up and down. See figure 7 to see that the servo which is attached to back has the metal wire extending forward to the front of the device. Additionally figure 4 shows how the eyes are wired to the upper servo.

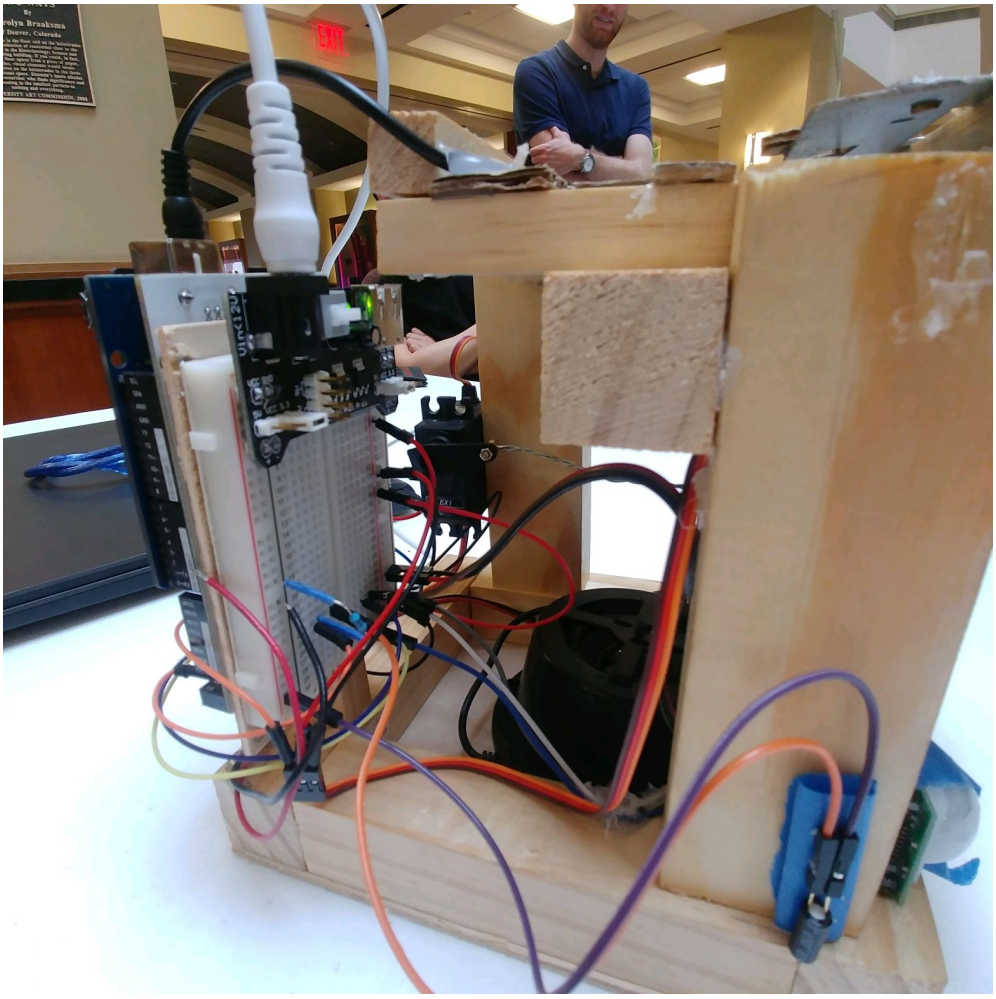


Figure 7: Side view of animatronic with mouth wire

Section 5: Sensors

There are a total of three sensors used in this device, two of which are infrared sensors and the last one being a tilt switch. The infrared sensors are placed under the face on the left and right side of the chassis. To give the sensors a blind spot so that they can distinguish direction, tape is attached to the sensors to limit their sensitivity. This can be seen in figure 8. Also the tilt switch is placed upside down and hot glued to the bottom left side of the face near the left infrared sensor. Two infrared sensors were chosen for this device to simplify both the coding and the devices interactions. If one infrared sensor triggers the back servo turns the eyes in the direction of the triggered sensor. Same for the opposite sensor. However, if both are triggered, the eyes will move to the center and the mouth will move up and down. The mp3 shield will then play the recording of Professor Gao saying, “Welcome to UTSA department of Mechanical Engineering.” Also if the sensors do not sense anyone, the device will start moving its eyes left and right as if to search for someone.

The tilt switch services to allow the device to know its spatial orientation. If the device is lifted and flipped upside down, the mouth servo will trigger and the device recording will then play, “Ahhh, please put me back down.” These sensors were chosen due to there simplicity. They are meant to only give a yes or no response to the arduino. This makes calibrating the device non-existent sense the device only cares if it is receiving a response, and not what the reading actually is. This has the added benefit of making the device quick to set up, and simple to code.

The operation of the infrared sensor is that it reads a 30 foot cone in front of it and searches for any infrared light. This is usually generated by people. If there is then the sensor creates a signal. The tilt switch works by completing the connection when gravity moves the connection back into place creating a completed current and thus a signal. All wiring of the sensors and servos connects to the arduino mega which is attached to the mp3 shield along with its own support pillar.

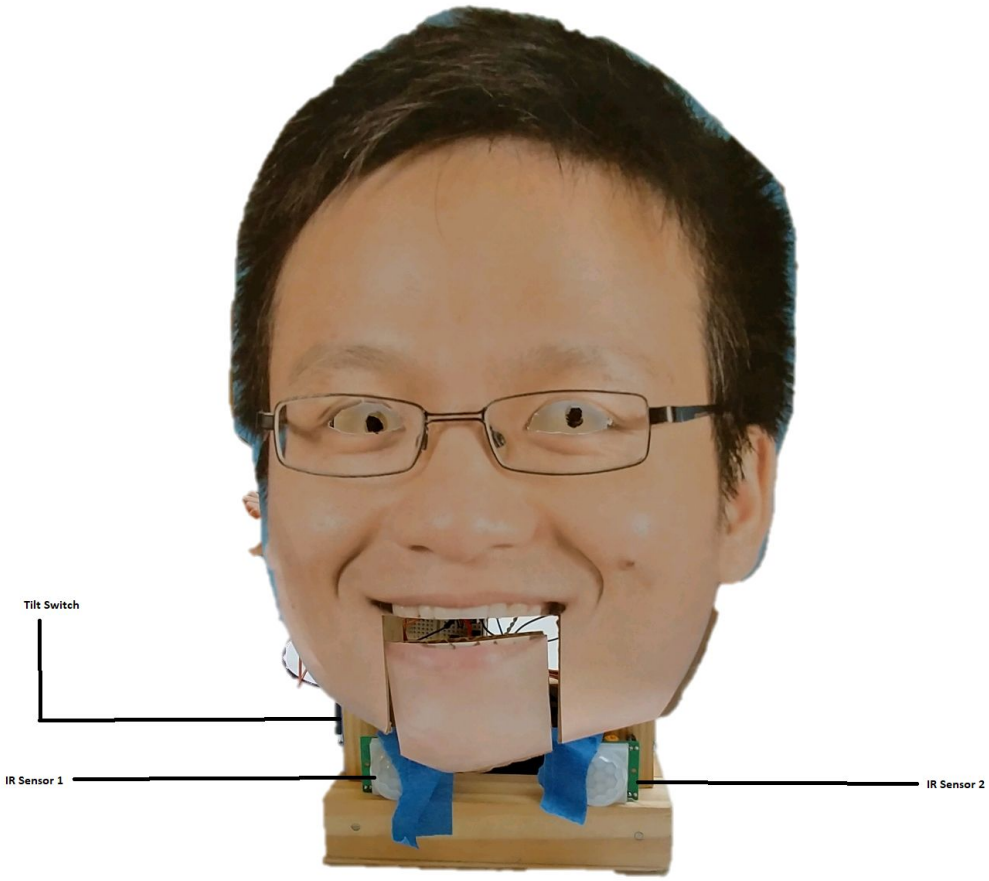


Figure 8: Front with IR sensors

Section 6: Programming for interaction

The programing for this device has been set up preemptively to be simple for the sensors. The logic is set up with if statements in the event that a device receives signal. The more complicated part of the code is specifying at which instances the servos should move and to which degree. This required primitive testing to find the best servo angles for the device and how many oscillations were required for the mouth movements to be believable as the source of the noise. The code for this device can be seen in the appendexies.

The code first initializes the pins and sets up the libraries to be used with this device. Then the pins for the shield are defined. The two servos are then distinguished as eye servo and mouth servo. Next the loop is set up, with the mp3 player being initialized followed by the

mp3 beeping tones which will be used if there is a error. A second loop is then created to direct the servos, and sensors. The loop first defines the three states for the eyes which are to look left, right, and center. Depending on if the inputs from the infrared sensors is high or low. If both are high then the welcome to utsa text is played and the bottom servo moves with the words. If both eyes are low then the eye servo is told to move randomly. Lastly, if the tilt switch is triggered by flipping the head, the mp3 shield player the second track (put me down). Again the mouth servo moves with the words in this if statement. The code then ends, and then the statement loops. The operation of this device and be seen in the link shown in the references. [5]

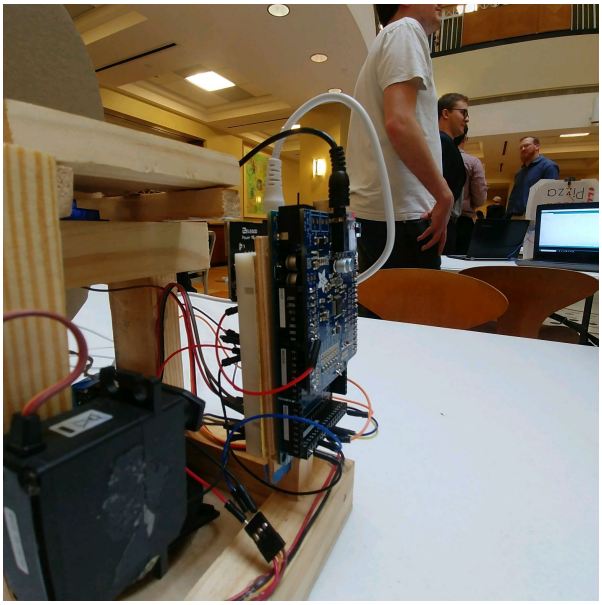


Figure 9: Back Wiring of device

Section 7: Lessons learnt and suggestions (1 page)

1. The most difficult part of this project was getting the mp3 shield to work which required, the unexpected step of soldering 6 pairing connections on the board together. This is the first time a device had this requirement in order to get it operating when soldering. Also within the mp3 shield code.
2. The shield would enter into a infinite loop of error due to the initial code. This was corrected by writing a override to skip over this looping error.
3. Finding the right command the correct folder that was written on the micro-sd card that was inserted into the shield. This was based off a folder naming convention.
4. When soldering a technique of applying the solder directly to the base of the pin was mastered after some difficulty and is recommended to be learned to any who wish to replicate this project. This is needed since no pins can or should have a branching connection across the board unless specified by the manufacturer.
5. The second difficulty with this project was finding the correct angle and timing needed for the mouth servo to be believable as the source of the audio. This timing was done

through trial and error with the use of lip syncing concepts. It is recommended that you use a lip syncing guide.

6. The moving the tilt switch didn't always trigger the connection, to fix it tapping the device would move the switch into place. It is recommended that you use a more expensive tilt switch so that this does not happen.
7. The audio would sometimes play both tracks simultaneously when demonstrating the device. This would cause the shield to beep loudly and crash. It is recommended that a response is programed if all sensors activate.
8. The infrared sensors during demonstration would continuously trigger due to the large number of people present. This "error" (technically it is supposed to do this) could have been avoided if motion sensors were used instead and would be triggered if a value was read that was below a certain threshold.
9. Sometimes the connections between the sensors, would come apart sense they were all wired directly into the breadboard. It is recommended that you find a disposable breadboard and hard solder the wires to it.
10. Due the device continuously talking during demonstration, a switch was grabbed from spare parts and added to the device to mute it when it was demmond desirable for the speaker to be going off. It is recommended that this aspect be included into your design.
11. The metal wire that connected the cardboard mouth to the servo would sometimes bounce out of place and get stuck behind the face. It is recommended that the enclosed region be designed around the mouth wire so that it blocks the wire from bouncing out of expected bounds jamming the device.
12. It was noted from observation from other projects that they could not get loud enough audio from there device. This was due to there devices all having a single power source and our device having a speaker with a battery inside.. It is recommended that the speaker be powered by a external source instead of the powered by the arduino.

Section 8: Personnel and bill of materials
(a) Personnel

Task	Main Personnel	Secondary personnel
Chassis Assembly	Kyle Kinsey	Brandon Cox
Electrical Assembly	Brandon Cox	Kyle Kinsey
Part acquisition and blueprinting	Andrew Kim	Kyle Kinsey
Coding/Programing	Kyle Kinsey	Brandon Cox
Jointing, motor interface	Brandon Cox	Kyle Kinsey
Audio Recording and editing	Kyle Kinsey	Andrew Kim

(b) Bill of materials

No.	Description	Website/comment	Qty.	Unit $	Total $
1	Arduino MEGA 2560	Provided	1	$90	$90
2	Small stepper motor	https://www.sparkfun.com/products/10551	1	6.95	6.95
3	Wood	Scrap	1	NA	NA
4	Micro servos	https://www.amazon.com/gp/product/B072V529YD/ref=oh_aui_detailpage_o01_s00?ie=UTF8&psc=1	1	$17.99	$17.99
5	Mp3 Shield	https://www.amazon.com/Adafruit-Music-Maker-Shield-Arduino/dp/B00SDTE380	1	$39.87	$39.87
6	Dollar store speaker	Dollar store	1	$1	$1
7	RC car servo	Scrap	1	NA	NA
8	Wires	Scrap	NA	NA	NA

9	Metal Wire	Scrap	3	NA	NA
10	Power Cord	Scrap	1	NA	NA
11	Tilt Sensor	Left over from measurements and instruments lab	1	$2.25	$2.25
12	Foam balls	Dollar store	2	$1	$1

Acknowledgements

We thank Professor Gao for allowing us to use his voice and image for this project.

We also thank Professor Bhounsule for giving us the opportunity to work on this fun project.

References:

[1] Animatronics Face of UTSA's President, Dr. Ricardo Romo, https://youtu.be/xkze1_hnam0

[2] ROMOBOT - ANIMATRONIC FACE ROBOT
https://www.instructables.com/id/RomoBOT-Animatronic-Face-Robot/
https://www.instructables.com/id/Halloween-Animatronics/

[3] Pyroelectro.com animatronic mouths, eyes, and eyebrows
http://www.pyroelectro.com/tutorials/animatronic_mouths/
https://www.youtube.com/watch?v=KgCbQvBw-Bc
http://www.pyroelectro.com/tutorials/animatronic_eyes/

[4] FRITZ - Robotic Head Kit
http://www.keenbots.com/Fritz/index.php
https://youtu.be/1A_sELjnYPs

[5] Dr. Gao mechatronics UTSA
https://www.youtube.com/watch?v=1mfvOX6Nu6k&feature=youtu.be

Appendix A: Code

```
#include <SPI.h>
#include <Adafruit_VS1053.h>
#include <SD.h>
#include <Servo.h>


// These are the pins used for the breakout example
#define BREAKOUT_RESET  9      // VS1053 reset pin (output)
#define BREAKOUT_CS     10     // VS1053 chip select pin (output)
#define BREAKOUT_DCS    8      // VS1053 Data/command select pin (output)
// These are the pins used for the music maker shield
#define SHIELD_RESET  -1      // VS1053 reset pin (unused!)
#define SHIELD_CS     7      // VS1053 chip select pin (output)
#define SHIELD_DCS    6      // VS1053 Data/command select pin (output)
// These are common pins between breakout and shield
#define CARDCS 4     // Card chip select pin
// DREQ should be an Int pin, see http://arduino.cc/en/Reference/attachInterrupt
#define DREQ 3       // VS1053 Data request, ideally an Interrupt pin

Adafruit_VS1053_FilePlayer musicPlayer =
  Adafruit_VS1053_FilePlayer(SHIELD_RESET, SHIELD_CS, SHIELD_DCS, DREQ, CARDCS);

int inputPinL = 40;
int pirStateL = LOW;
int valL = LOW;

int inputPinR = 38;
int pirStateR = LOW;
int valR = LOW;

int tiltpin = 24;
int valT = LOW;

Servo eyeservo;
Servo mouth;

////

void setup() {
  Serial.begin(9600);
 // initialise the music player
  if (! musicPlayer.begin()) { // initialise the music player
     Serial.println(F("Couldn't find VS1053, do you have the right pins defined?"));
     while (1);
  }
  Serial.println(F("VS1053 found"));

  musicPlayer.sineTest(0x44, 500);    // Make a tone to indicate VS1053 is working

  if (!SD.begin(CARDCS)) {
    Serial.println(F("SD failed, or not present"));
    //while (1);  // don't do anything more
  }
```

```
  Serial.println("SD OK!");

  // Set volume for left, right channels. lower numbers == louder volume!
  musicPlayer.setVolume(10,10);

  if (! musicPlayer.useInterrupt(VS1053_FILEPLAYER_PIN_INT))
    Serial.println(F("DREQ pin is not an interrupt pin"));

  pinMode(inputPinL, INPUT);
  pinMode(inputPinR, INPUT);
  pinMode(tiltpin, INPUT);

  eyeservo.attach(44);
  mouth.attach(48);
  //mouth.write(90);

}

void loop() {
  valL = digitalRead(inputPinL);
  valR = digitalRead(inputPinR);
  valT = digitalRead(tiltpin);

     if (valL == HIGH && valR == LOW) {
       eyeservo.write(25);
       delay(500);
    }

     if (valL == LOW && valR == HIGH) {
       eyeservo.write(55);
       delay(500);
    }

     if (valL == HIGH && valR == HIGH) {

       musicPlayer.playFullFile("track001.ogg");

         if (! musicPlayer.startPlayingFile("/track001.mp3")) {
            Serial.println("Could not open file track001.mp3");
            while (1);
         }
      Serial.println(F("Started playing"));

    while (musicPlayer.playingMusic) {
       eyeservo.write(45);
       mouth.write(90);
       delay(1150);
    //Hello
       mouth.write(85);
       delay(100);
       mouth.write(90);
       delay(100);
       mouth.write(85);
       delay(200);
       mouth.write(90);
    //Welcome
       delay(400);
```

```
    mouth.write(85);
    delay(200);
    mouth.write(90);
    delay(100);
    mouth.write(85);
    delay(200);
    mouth.write(90);
  //To
    delay(25);
    mouth.write(85);
    delay(300);
    mouth.write(90);
  //UTSA
    delay(100);
    mouth.write(85);
    delay(150);
    mouth.write(90);
    delay(50);
    mouth.write(85);
    delay(300);
    mouth.write(90);
    delay(50);
    mouth.write(85);
    delay(400);
    mouth.write(90);
    delay(50);
    mouth.write(85);
    delay(160);
    mouth.write(90);
  //Mechanical
    delay(50);
    mouth.write(85);
    delay(200);
    mouth.write(90);
    delay(50);
    mouth.write(85);
    delay(200);
    mouth.write(90);
    delay(50);
    mouth.write(85);
    delay(150);
    mouth.write(90);
    delay(50);
    mouth.write(85);
    delay(200);
    mouth.write(90);
  //Engineering
    delay(50);
    mouth.write(85);
    delay(200);
    mouth.write(90);
    delay(30);
    mouth.write(85);
    delay(100);
    mouth.write(90);
    delay(40);
    mouth.write(85);
```

```
    delay(140);
    mouth.write(90);
    delay(25);
    mouth.write(85);
    delay(240);
    mouth.write(90);
    delay(300);

  delay(3000);
}
  }

   if (valL == LOW && valR == LOW) {
     eyeservo.write(random(25, 55));
     delay(1000);
}


  if ( valT == HIGH) {

   musicPlayer.playFullFile("track002.ogg");

      if (! musicPlayer.startPlayingFile("/track002.mp3")) {
         Serial.println("Could not open file track001.mp3");
         while (1);
      }
    Serial.println(F("Started playing"));

  while (musicPlayer.playingMusic) {
    eyeservo.write(45);

  //Ahh!
    delay(530);
    mouth.write(85);
    delay(950);
    mouth.write(90);
  //Please
    delay(400);
    mouth.write(85);
    delay(200);
    mouth.write(90);
 //Put
    delay(100);
    mouth.write(85);
    delay(100);
    mouth.write(90);
 //Me
    delay(25);
    mouth.write(85);
    delay(200);
    mouth.write(90);
 //Back
    delay(25);
    mouth.write(85);
    delay(200);
    mouth.write(90);
 //Down
```

```
      delay(25);
      mouth.write(85);
      delay(500);
      mouth.write(90);
      delay(1500);

  }
    }
}
```