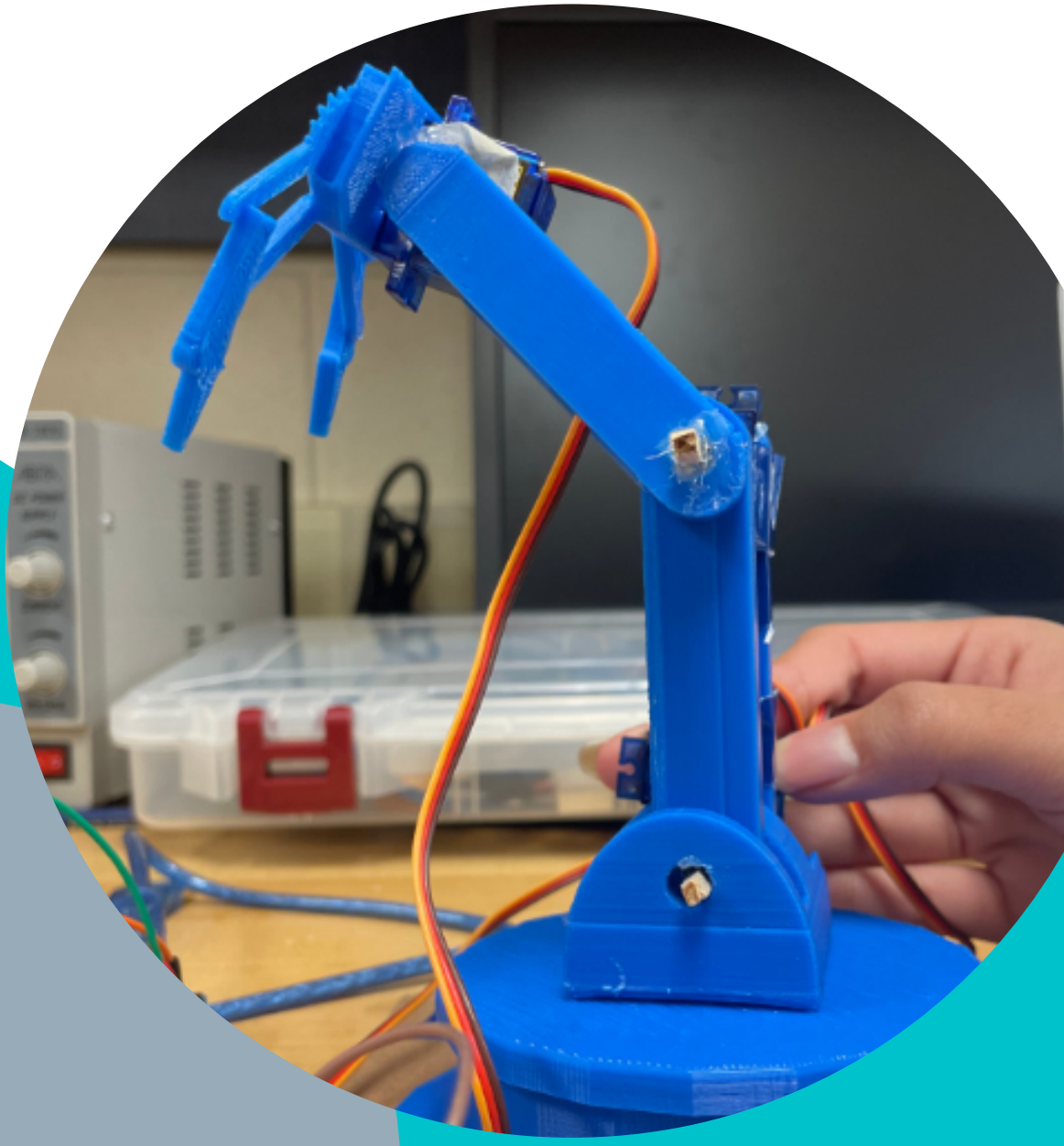
Bluetooth Controlled Robotic Arm



TEJ4M1 | Ms.Gigg | Momina & Elaine | Jan.2023

Table of Contents

01 Title Page

02 Table of Contents

03 Device Overview

05 Safety Information

06 User Instructions

07 Research + Process

10 Parts Needed

12 Circuit Diagram

13 Breakdown of Code

16 Application + code

18 Reflection

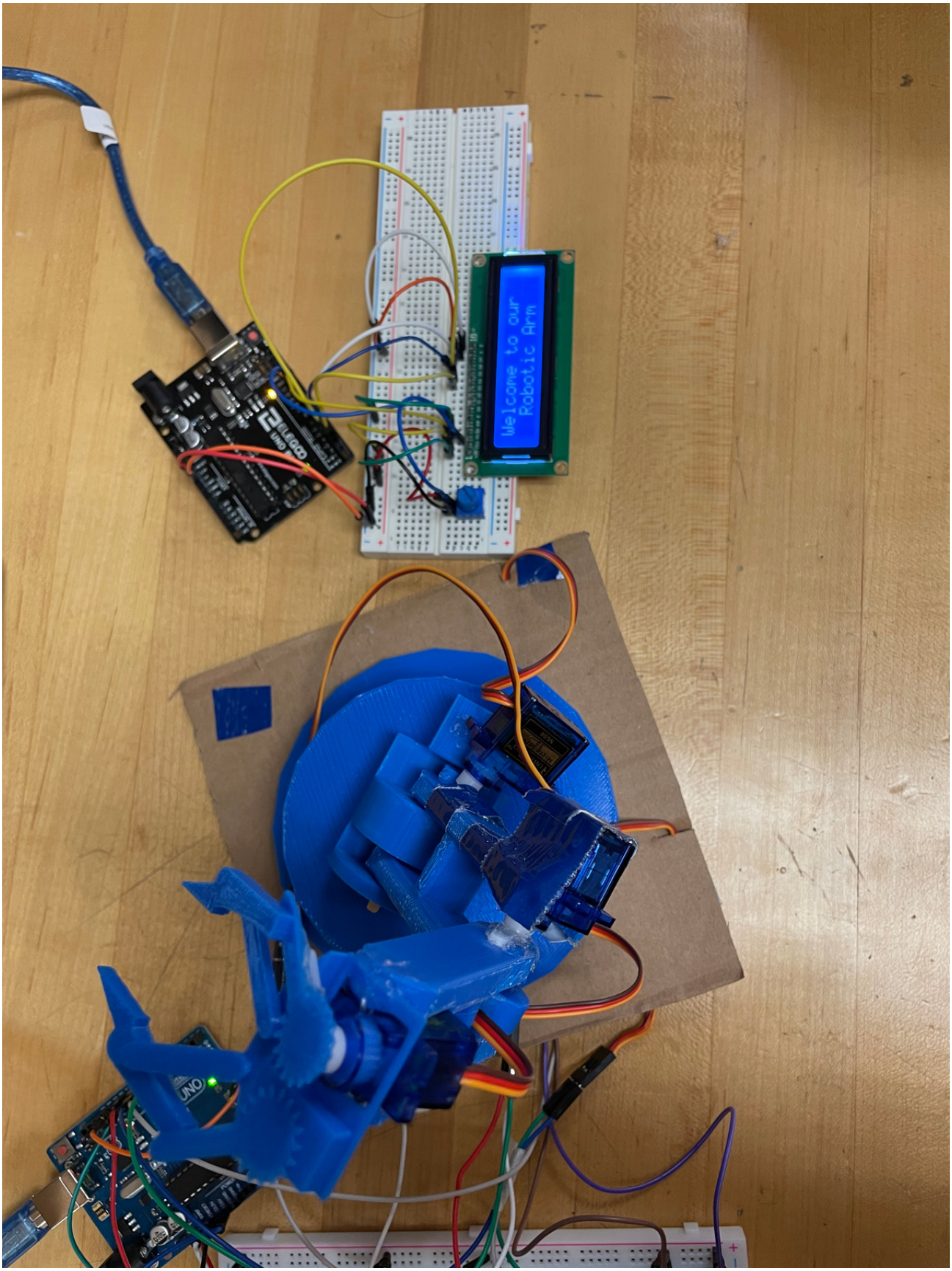
19 References

Device Overview

Our seminar presentation focused on the Da Vinci Robot, which mimics and simulates a surgeon's hand movements to perform surgeries. We learnt that where human capabilities are limited, the potential of robotic systems opens a lot of doors filled with innovative opportunities. They can perform tasks over and over again with the same speed, consistency, and efficiency. This can be seen in many industrial, manufacturing, and healthcare industries which is why our robotic arm serves as an interactive prototype for these revolutionary machines that support our society as a whole and in our daily lives.

Users can control and steer our 4-jointed robotic arm however they wish; rotating it and picking up objects solely through the mobile software we developed. The networking component is through a Bluetooth module we purchased, allowing the robot to be controlled wirelessly over a long distance. Our robot utilizes 4 servo motors, a Bluetooth module, and an Android device to control the flexible movement of the arm parts we designed and 3D printed. Additionally, we also created an interactive LCD display that introduces the user to our device.

Overall, our goal was to experiment and combine the characteristics of robotics with Arduino and networking to create a successful Bluetooth-controlled device.


Welcome to our
Robotic Arm

Safety Information

The robotic arm is connected to four servo motors that operate at a maximum voltage of 5V, which is low-voltage electricity and not considered harmful to human health. However, because the robotic arm still has electric components and moving elements, there are some safety measures that need to be taken into account when using this device. Please take note of the safety instructions and information below:

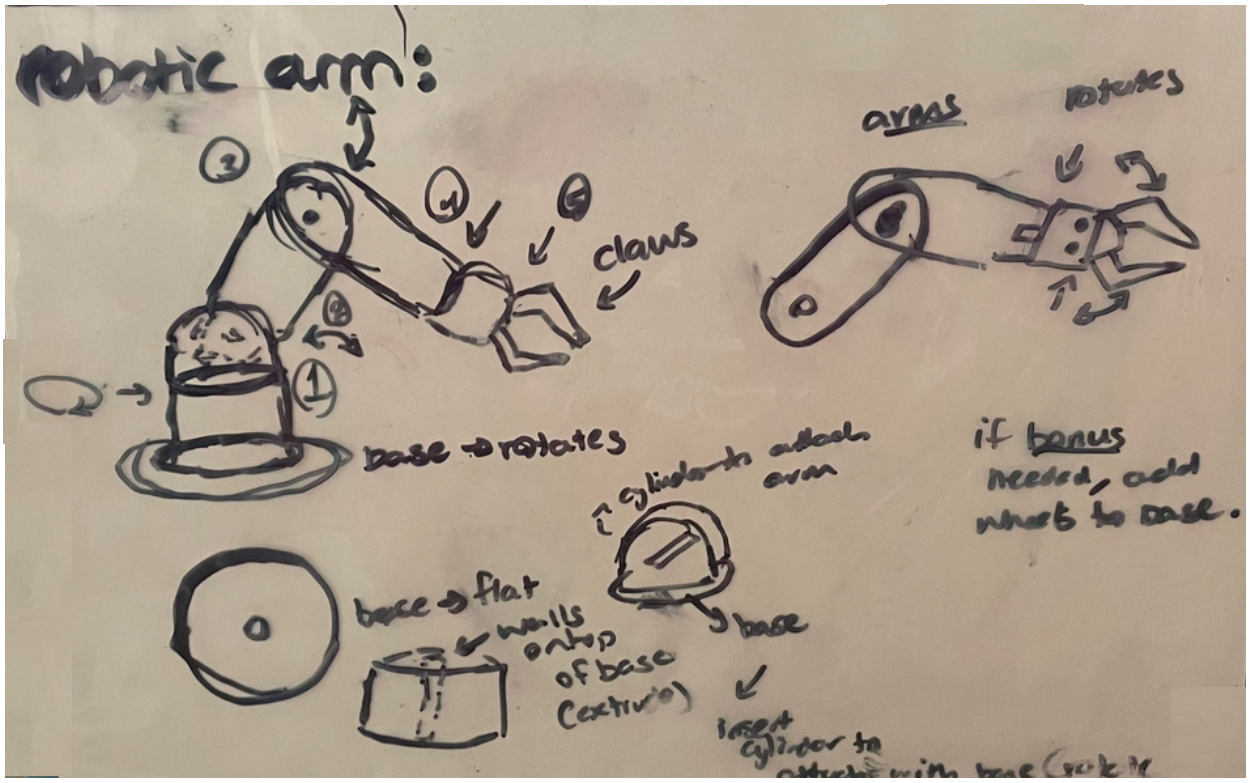
- Avoid touching equipment or cables that have come in contact with liquids.
- Please keep your hands away from the exposed wires that connect the Robotic Arm to the Arduino to avoid damaging or pulling them. If using the cords, hold the connections firmly instead of pulling them.
- Keep unnecessary materials away from the robot operation range, including coats, backpacks, food, or drinks.
- Always maintain stability with the robotic arm to ensure it doesn't lose balance.
- Please handle the mobile phone (used for remote control) carefully.
- Please ensure that the Bluetooth module is connected to 3.3V instead of 5V to avoid any damage to the component.

User Instructions

This project requires you to have two Arduino Uno boards and breadboards. Firstly, you need to assemble the circuits (please refer to the diagram on page 12). There is one breadboard for the LCD component and one for connecting the servo motors and Bluetooth application to control the movements of the robotic arm. Once the circuits are assembled, connect two USB 2.0 cables from the Arduino boards to your computer. After connecting, you must upload the supplied code (please refer to the code on page 13) to the Arduino boards via USB using the Arduino software. Since there are two circuits, you have to upload separate codes for each of them. For building the physical model of the robotic arm, we have provided examples of our original design process of the 3D-printed parts (page 8) for which we used TinkerCAD.

After the assembly, the first step is to connect the Bluetooth. Please ensure that our app is downloaded on an Android phone and that the Bluetooth option is turned on and paired to HC-06 from phone settings. The blinking orange light on the Bluetooth module indicates that it is not connected. To connect, click the scan button on the app, which will take you to another window with all the Bluetooth options. Click on HC-06 and wait for the light on the module to stop blinking, which shows a successful connection. We also have a text label on the app which shows the Bluetooth connection status. Additionally, you can turn the potentiometer on our second Arduino to adjust the contrast of the text and see an introductory welcome message for display. Once you have completed all steps properly and applied the correct settings, you can now control the robotic arm with the 4 horizontal sliders on the phone.

Research + Process



Researching and Brainstorming

We started with brainstorming and researching to find information on our questions. We considered our biggest concern, which was how to make the robot work. We found many similar projects to assist us and decided to incorporate a networking component with Bluetooth. We then had to decide what Bluetooth module to use and how to create the phone application.

Components

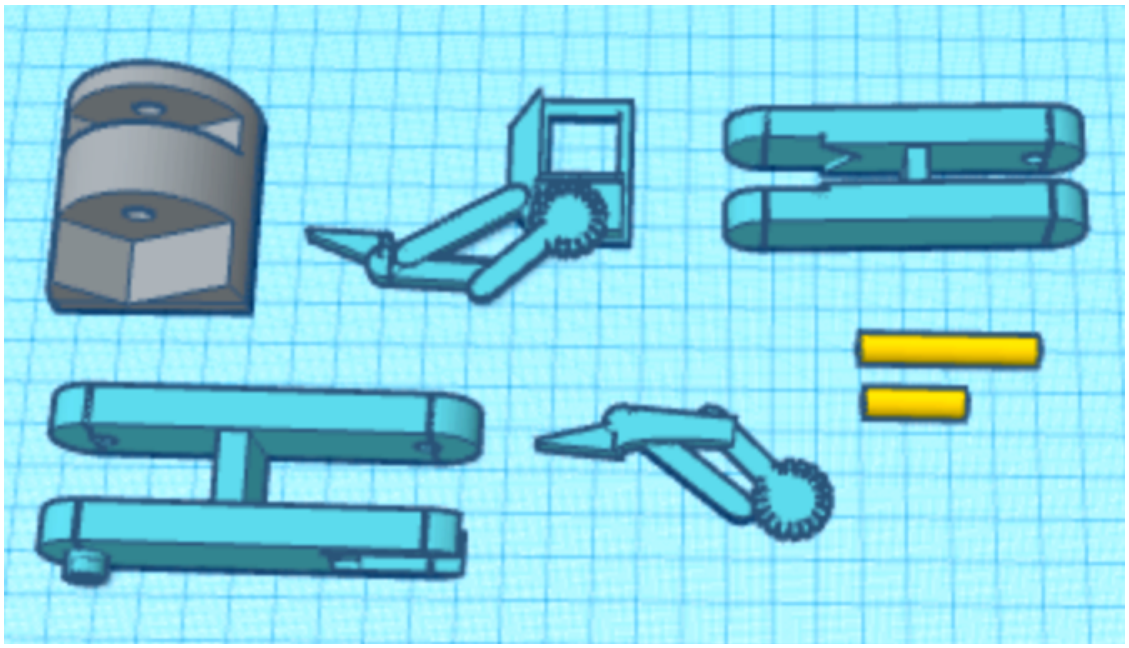
1. Buying the HC-06 Bluetooth module
2. Collecting materials (servo motors, LCD)
3. Finding an Android device to connect Bluetooth.

Build & Testing

1. Building the application through MIT developer.
2. Establishing a Bluetooth connection from phone to Arduino
3. Testing 1 servo motor at a time through the app.

3D Printing

1. Designing and scaling all parts on TinkerCAD
2. Finding errors and reprinting parts.
3. Improvising and parts replacement (Wooden dowels)

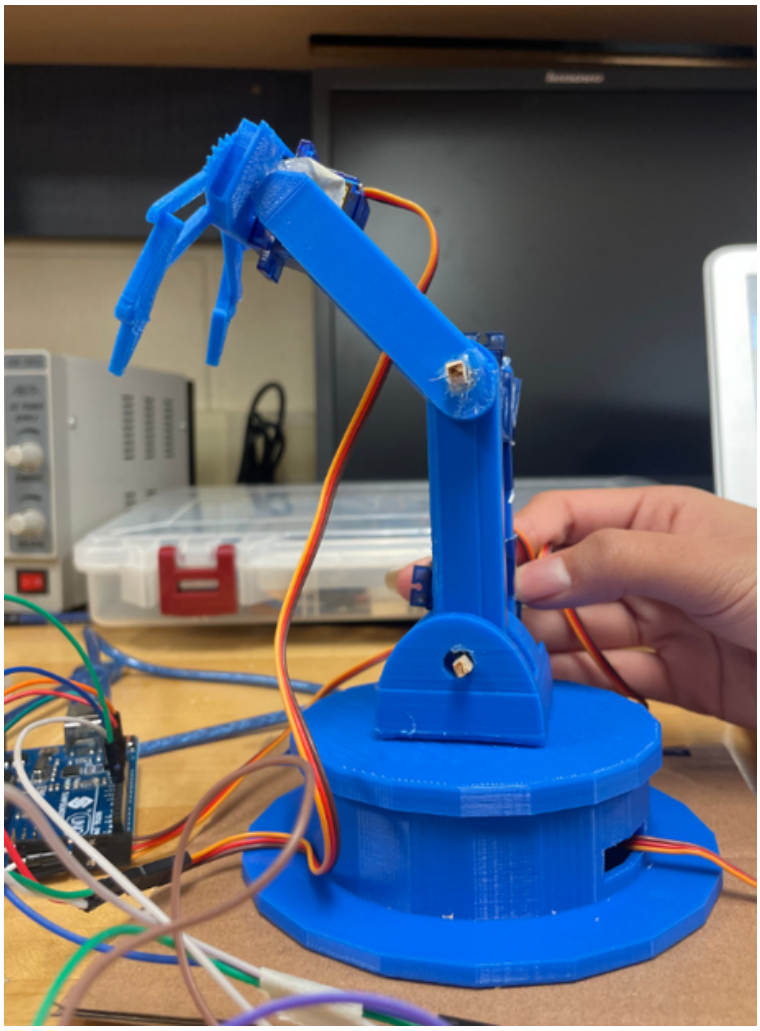


Code

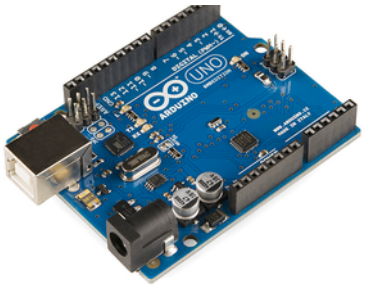
Modifying code: Adding pin numbers, changing Bluetooth settings, assigning personalized variables, simplifying and organizing code for each of the servo motors.

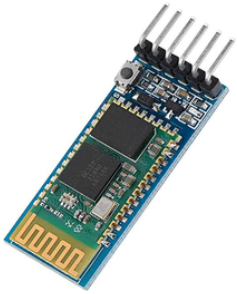
Assembly

1. Putting all pieces together
2. Problem-Solving along the way (Glue gun, tape)
3. Organizing all wiring.

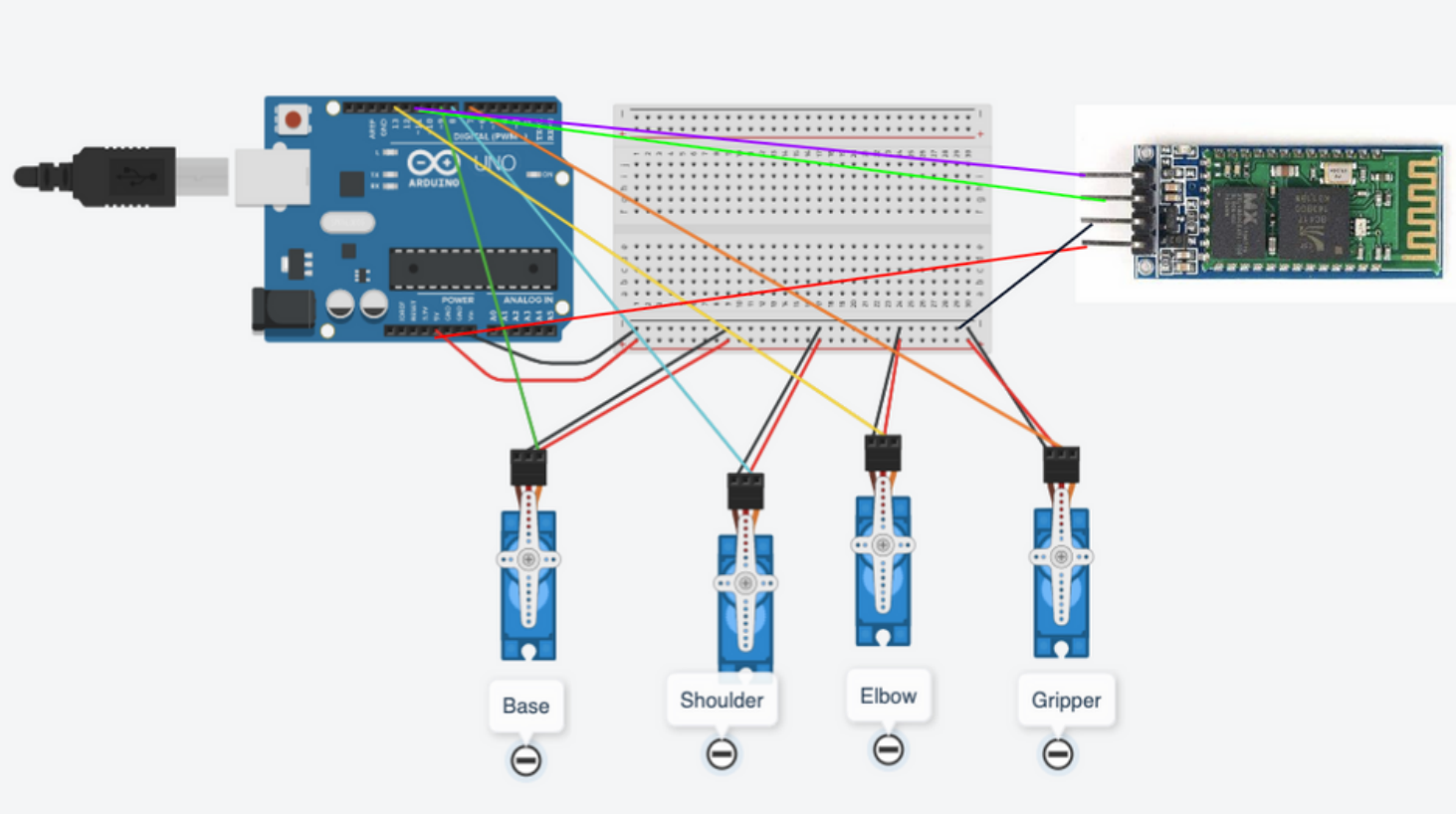


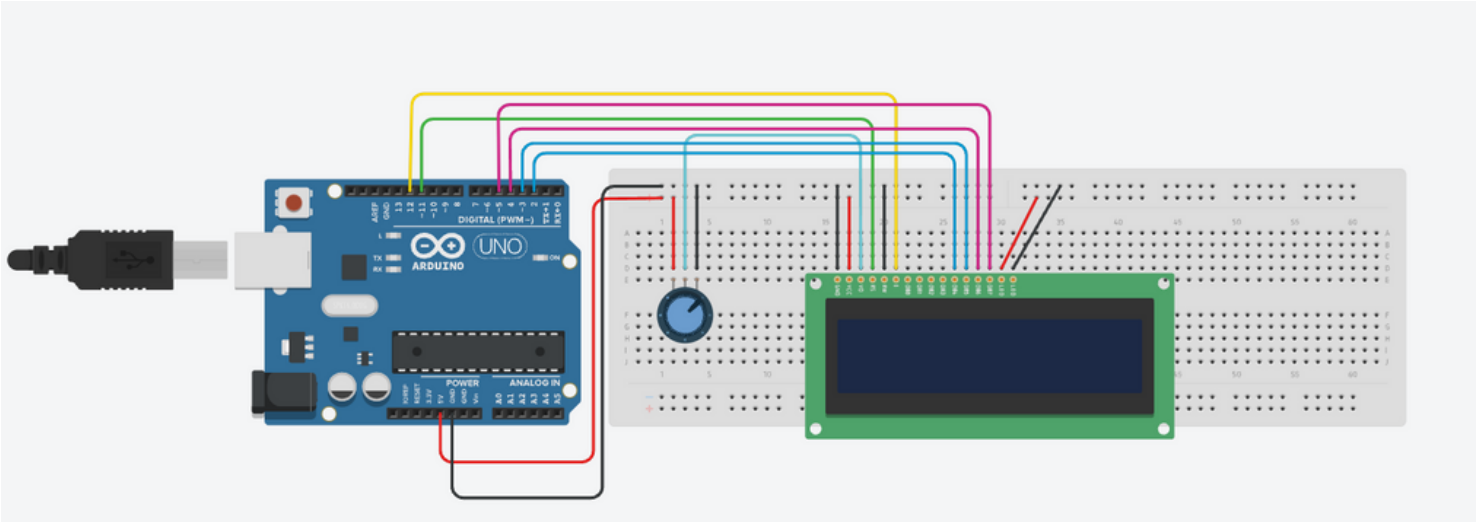
Parts Needed

Part	# of parts needed	Photo
Arduino Uno	2	
USB 2.0 Cable	1	
Breadboard	2	
Resistor（220Ω）	3	
Potentiometer	1	

Part	# of parts needed	Photo
Jumper Wires	50	
Servo Motor(SG90)	4	
Bluetooth Module	1	
16 * 2 LCD Character Display	1	

Circuit Diagrams





Breakdown of Code

Code	Explanation
`#include <SoftwareSerial.h>` `#include <Servo.h>`	Adding the Rx Tx Software library. Adding the server library.

`int bluetoothTx = 10;`
`int bluetoothRx = 11;`
 	
`SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);`
	Assigning the Bluetooth Tx to pin 10 and Rx to pin 11.
	Defining Bluetooth to create communication with Arduino Board over the pins Rx and Tx which serve as input and output

`motor1.attach(7);`
`motor2.attach(13);`
`motor3.attach(8);`
`motor4.attach(9);`
	Attaching the servo motor wires to the corresponding pin numbers.

`Serial.begin(9600);` `bluetooth.begin(9600);`	Connecting USB serial to Arduino and Bluetooth module to android device
`if(bluetooth.available()>= 2 )` `{`	Starting a nested If-Else statement (condition) that runs the remaining blocks of code only if there are at least two bytes of data on the bluetooth connection (sent when bluetooth is connected to the android phone)

unsigned int servopos = bluetooth.read();
unsigned int servopos1 = bluetooth.read();
unsigned int realservo = (servopos1 *256) + servopos;
Serial.println(realservo);
	This section allows for the communication from the android phone to the servo motors. It reads the data from the bluetooth, assigning them to "servopos" and "servopos1". Then it takes its combination result and assigns it to "realservo", which is the servo position from the android app.

if (realservo >= 1000 && realservo <1180) {
int gripper = realservo;
gripper = map(gripper, 1000, 1180, 0, 180);
motor1.write(gripper);
Serial.println("Gripper ON");
delay(10);
}
if (realservo >= 2000 && realservo <2180) {
int elbow = realservo;
servo2 = map(elbow, 2000, 2180, 0, 180);
motor2.write(elbow);
Serial.println("Elbow ON");
delay(10);
}
if (realservo >= 3000 && realservo <3180) {
int shoulder = realservo;
shoulder = map(shoulder, 3000, 3180, 0, 180);
motor3.write(shoulder);
Serial.println("Shoulder ON");
delay(10);
}
if (realservo >= 4000 && realservo <4180) {
int base = realservo;
servo4 = map(base, 4000, 4180, 0, 180);
motor4.write(base);
Serial.println("Base ON");
delay(10);
}
	There are 4 nested statements in this section. Each section controls one of the four servo motors in our robot. Taking a look at the first motor; if the "*realservo*" value from last explanation is between 1000 and 1180, then the value will be assigned to *gripper*. Then, the map function rewrites the given value (from 1000 -> 1180) to the servo1 motor range which is from 0 to 180 (in degrees). For example, if I have a value of 1010 on my phone application, then the *gripper* position value will be 10 degrees. The delay function adds a delay of 10 milliseconds for the program to execute. This process is repeated for all 4 servo motors with their own defined range.

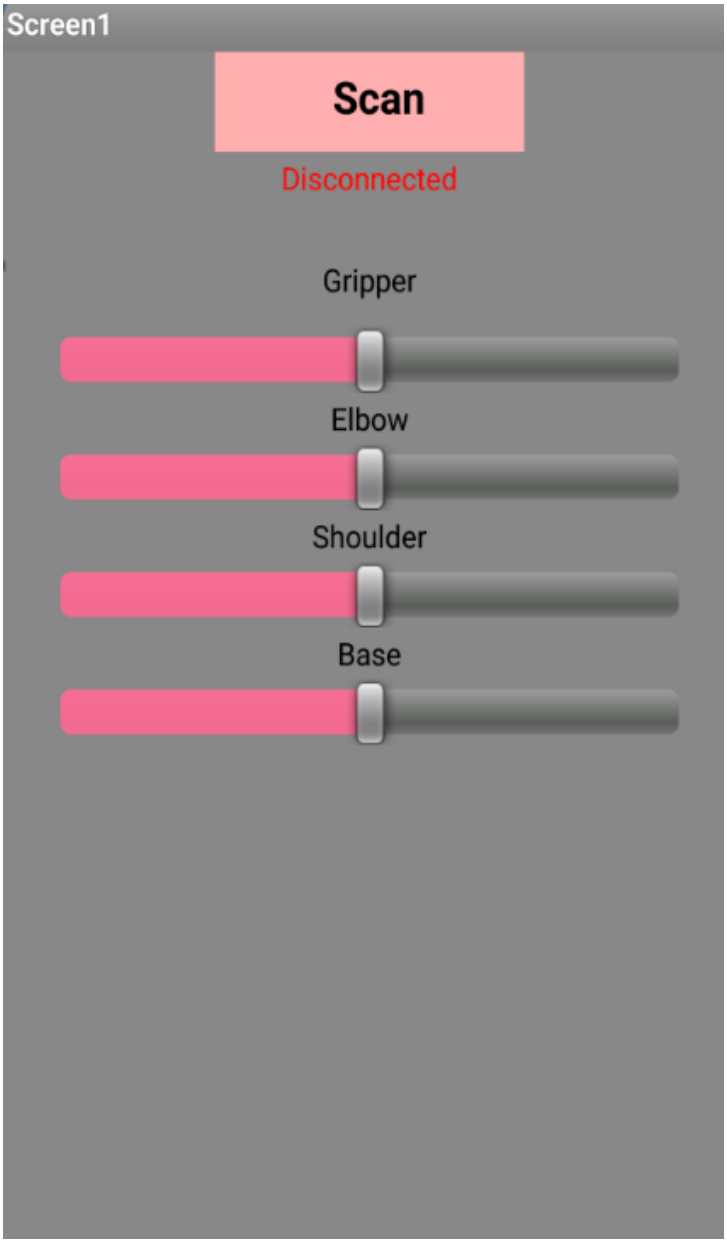
Code Breakdown - LCD Message

Code	Explanation
#include <**LiquidCrystal**.h>	Adds the library for the LCD
const int RS = 11, EN = 12, D4 = 2, D5 = 3, D6 = 4, D7 = 5; LiquidCrystal lcd(RS, EN, D4, D5, D6, D7);	Creates an instance from library to introduce LCD. Defines the variable for the LCD pins and assigns them to their own pin on the Arduino.

lcd.begin(16, 2);
lcd.setCursor(0, 0);
lcd.print("Welcome to our");
lcd.setCursor(2, 1);
lcd.print("Robotic Arm);
	Sets the number of columns and rows for the LCD and moves the curser to coordinate position of (0,0)
	Then, it prints the first message "Welcome to our" on this coordinate.
	The curser position is then changed to (2,1) where it completes our message by displaying "Robotic Arm"

Android Application

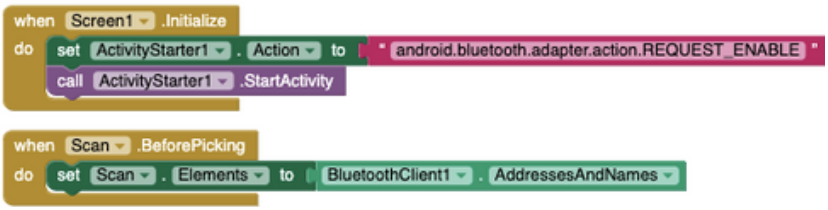
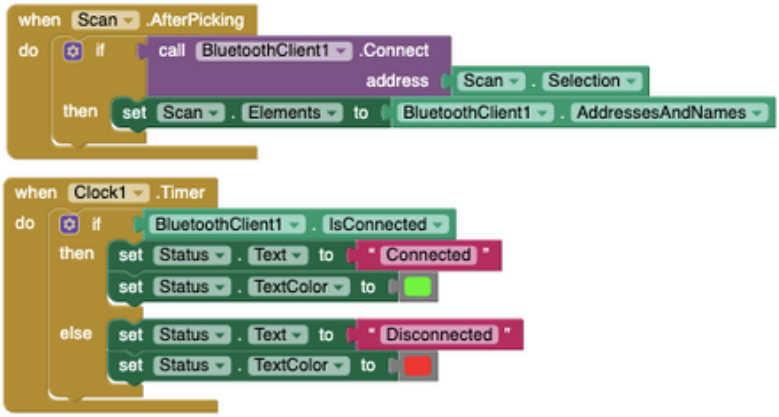
We built our app using the MIT app inventor. The first thing we did was add all the components we needed which were scan lists, horizontal sliders, labels, and a bluetooth client. Then we customized all the features and added a specific range for each slider.



Instructions

- Connect Arduino to power
- Open the App and click scan
- Choose HC-06 from the list
- Screen will say if Bluetooth is connected or disconnected

Breakdown of App

Blocks	Explanation
	Initializes the main screen and enables android device to give permissions to the app for connecting bluetooth. Adds a scanner that provides a list with all Bluetooth clients.
	If the bluetooth client is chosen, it finds the bluetooth address and connects it. If the client connects, the status text label returns "connected" in green or else "disconnected" in red.
	There are 4 different blocks for each of the servo motors. It sets the sliders thumb position to a specific range (1st servo is 1000 - 1180, 2nd servo is 2000 - 2180, etc.). If the sliders position is changed, it calls the bluetooth client (our bluetooth module) and sends a 2 byte number to it, which allows the codes condition to run. This is repeated for all 4 servo motors.

Reflection

This performance task served as a great learning opportunity for us. It was our first time building a robotic device through Bluetooth, so speaking truthfully, it was quite challenging. Our primary strategy for solving Arduino and code-related problems was trial and error. If a problem would arise, we used the divide and conquer strategy to efficiently manage our time, with one of us troubleshooting the code while the other fixed the wiring. If the problem was more serious, we broke it down into smaller bits to determine where we went wrong.

Throughout the process, our biggest challenge was 3D printing the components. We spent several hours designing various parts and scaling them to the proper size. We ultimately had to scale our entire design to 90% of the dimensions due to its large dimensions. This meant that the servo motors we had originally intended to put inside the components did not fit. Despite being able to reprint all the components, we wanted to be innovative and resourceful by making use of what we already had. In order to improvise, we had to use tape and hot glue. For one of the motors, we had to hold it down ourselves as we ran short of time for problem-solving. We used parts replacement by using wooden dowels as the arm joints.

Despite the obstacles we encountered, developing this device was an excellent opportunity for us because there was a lot to take away from the research and building process. As a result, this experience was truly exciting and rewarding as we got to watch our ideas come to life.

References

Arduino - LCD | Arduino Tutorial. (n.d.). Arduino Getting Started. Retrieved January 21, 2023, from https://arduinogetstarted.com/tutorials/arduino-lcd

Topić, T. (2020, August 25). *Robot arm*. YouTube. Retrieved January 21, 2023, from https://grabcad.com/library/robot-arm-142/details?folder_id=8869535

Bluetooth Controlled Servo. (2018, November 30). Arduino Project Hub. Retrieved January 23, 2023, from https://projecthub.arduino.cc/JANAK13/5e6688a1-794c-4997-98e4-161692bef6db

DIY Arduino Robot Arm with Smartphone Control. (n.d.). How To Mechatronics. Retrieved January 23, 2023, from https://howtomechatronics.com/tutorials/arduino/diy-arduino-robot-arm-with-smartphone-control/

Home. (n.d.). YouTube. Retrieved January 23, 2023, from https://create.arduino.cc/editor/mertarduinotech/3e101020-29e0-4ce1-bb12-5afb2fddb645/preview