GNG2101

Design Project User and Product Manual

THE SONIC SOFA

Submitted by:
Group Z15
Tristan Rodgers
Ismail Mostafa
Yahya Mostafa
Usman Khan

Sunday July 23rd, 2023
University of Ottawa

Table of Contents

List of Figures

List of Tables

List of Acronyms and Glossary

Provide a list of acronyms and associated literal translations used within the document. List the acronyms in alphabetical order using a tabular format as depicted below.

Table 1. Acronyms

Acronym	Definition
CAD	Computer-Aided Design

Provide clear and concise definitions for terms used in this document that may be unfamiliar to readers of the document. Terms are to be listed in alphabetical order.

Table 2. Glossary

Term	Acronym	Definition

1 Introduction

This product is designed to improve the lives of people living with disabilities, and it represents a significant step forward in the field of assistive technology. This user manual will provide all the information necessary to assemble and use the remote control, as well as a detailed explanation of the context for its development.

Assumptions:

Our product is designed for people living with disabilities Many assumptions were made to create a remote control that is both effective and user-friendly. First, users have some experience with remote controls and that they can use voice commands to operate the device. Next, users will have access to a lift chair that operates a wired remote control, as opposed to an integrated control panel. Finally, the user can communicate using their voice in a manner that is loud and clear enough for the voice recording module to receive data.

Structure of User Manual:

This user manual is divided into several sections, each of which provides detailed information about a specific aspect of the remote control. First, an overview of the product, including a description of its features and how it works. Next, the assembly instructions, which provide step-by-step guidance on how to put the remote control together. Also included are troubleshooting tips, in case users encounter any issues during the assembly process or use of the remote control.

Contents and Purpose:

The purpose of this user manual is to provide users with all the information they need to assemble and use the remote control. Included are detailed instructions and illustrations to help users understand how the device works, as well as tips for troubleshooting and maintenance. This manual was created to make this product as accessible as possible, so that people living with disabilities can benefit from its use.

Intended Audience:

This user manual is intended for people living with disabilities, as well as their caregivers and family members. The remote control, as well as its user and product manual, was conceived to be user-friendly and accessible.

2 Overview

Problem: Millions of people around the world suffer from a series of degenerative illnesses and disabilities such as ALS that limit their motor function. Due to this, they rely heavily on support individuals to assist them for their daily activities. Reclining chairs that function with a remote that has buttons which are pushed provide a problem since individuals with limited motor function cannot push buttons on their own. The use of new technologies such as speech recognition can help provide solutions to this problem.

User needs:

- The system must be able to operate without Wi-Fi.
- The user must be able to use the system only with their voice.

This product introduces an innovative use of speech recognition technology to assist individuals with restricted motor function. There are currently no exact products on the market that works the way this one does. Currently, there are several smart technology assistants such as Alexa and Google Home that help with similar problems. However, these smart assistants need a Wi-Fi connection to work. This product is ideal since it does not require an internet connection to work. Furthermore, this technology connects directly to a reclining chair, allowing the position of the chair to be changed purely by the user's voice.

Key Features:

- The system enables voice-controlled operation of a reclining chair.
- It includes a remote for the chair, an Arduino microcontroller, and a voice-recognition microcontroller.
- A microphone is used to capture voice commands from the user.
- Servo motors mounted on the casing push the buttons on the remote based on voice commands.
- The case consists of 3 levels. The first level houses the electronics, the second level houses the remote, and the third level houses the servo motors.

Architecture/Construction:

- The system is designed with a sturdy casing to hold all components securely.
- It consists of an Arduino microcontroller and a voice-recognition microcontroller to process user commands.
- The remote for the reclining chair is placed within the casing.
- Servo motors are mounted on the casing to interact with the remote's buttons.

User Access Mode:

- Users can control the reclining chair using voice commands.
- They simply speak clearly to the microphone to initiate actions like reclining.
- The voice-recognition microcontroller interprets the commands and triggers the appropriate servo motor actions.

Special Conditions:

- The system is designed to universally fit various remote controls, making it adaptable for a wide range of users and devices.

- The casing's insert-slots hold the remote securely, ensuring consistent button-pressing by the servo motors.
- Voice recognition should be trained or calibrated to recognize user-specific speech patterns for accurate commands.

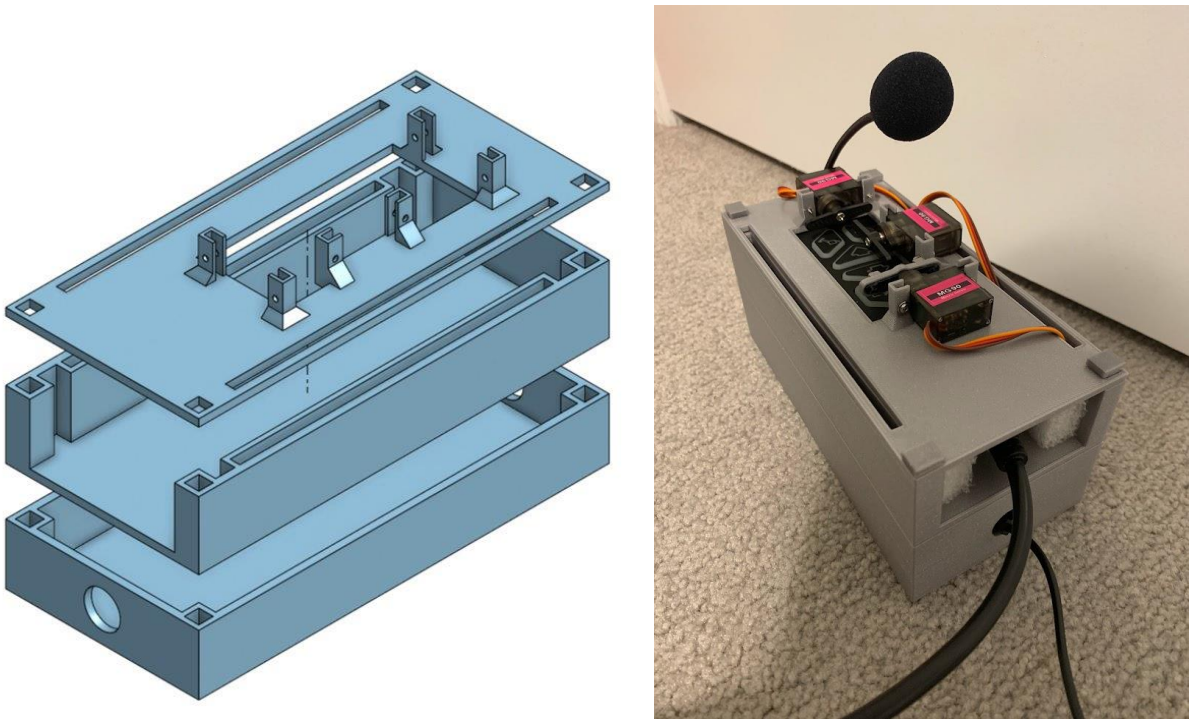

Figure 1. CAD schematic (on left) and photo (on right) of our prototype.
The bottom layer houses the electronics, the middle layer houses the remote, and the top layer houses the servo motors.

2.1 Conventions

Not applicable.

2.2 Cautions & Warnings

- User Safety: Before using the prototype, users should be made aware of the potential risks and limitations. Since the system involves motorized components and voice activation, users must be cautious while operating the lift chair. Instructions should be provided to ensure they understand how to stop or pause the movement in case of an emergency.
- Training and Familiarization: ALS patients might have varying levels of voice impairment. Adequate training and familiarization with the voice recognition system are crucial to ensure accurate command interpretation. The system should be designed to recognize and adapt to different voice patterns, tones, and accents.
- Fail-Safe Mechanisms: A fail-safe mechanism to prevent accidental activation of the lift chair is not implemented. This could include double confirmation for critical commands.

3 Getting started

In this section, we will offer a comprehensive guide of the system from setup to conclusion. The coherent arrangement of information should allow users to grasp the sequence and progression of the system and prototype usage. Images or screenshots are present to illustrate examples at each step.

3.1 Configuration Considerations

The prototype case is designed to house all the necessary components to enable voice-controlled operation of a lift chair. It consists of three levels, each serving a specific purpose:

- Level 1 - Electronics Compartment:
    - This consists of the bottom level.
    - In this level, the case contains the electronics required for the system to function, including an Arduino microcontroller and a Voice Recognition module.
    - The Arduino serves as the central control unit, while the Voice Recognition module interprets voice commands from the user.
    - Users don't need to interact directly with this level, as it's securely enclosed within the case.
- Level 2 - Remote Control Compartment:
    - This consists of the middle level.
    - In this level, the case contains a slot where the user inserts their own remote control, which operates the lift chair.
    - The slot is designed to accommodate various types of remotes, making it universally adaptable to different lift chair models.
    - The remote control interacts with the servo motors via the Arduino to control the lift chair.
- Level 3 - Servo Motor Compartment:
    - This consists of the top level.
    - In this level, the case contains one or more servo motors.
    - Each servo motor is positioned to align with specific buttons on the inserted remote control.
    - These servo motors push the corresponding buttons on the remote control based on voice commands received via the Voice Recognition module.
    - The servo motors are precisely calibrated to ensure accurate button presses.

Tools and Connections:

- Power Source: The prototype case requires a power source, which can be a standard power adapter or batteries to operate the Arduino and the Voice Recognition module.
- Voice Commands: Users can operate the lift chair by giving clear and audible voice commands to the Voice Recognition module. The module should be programmed to recognize specific phrases for activation and various lift chair actions (e.g., "Lift up," "Lift down," etc.).

- Remote Control Insertion: The user needs to insert their lift chair remote control into the designated slot in the Level 2 compartment. The insert-slots are designed to securely hold the remote in place.
- Servo Motor Connections: The servo motors are connected to the Arduino using appropriate wires or connectors to allow for precise control of their movements.
- Servo Motor Insertion: The servo motors can be inserted into the 3D printed cased design as it already contains insert-slots. Once inserted, the connection can be secured using a pair of screws.

3.2 User Access Considerations

Different users or user groups could benefit from this technology, each with their specific needs and restrictions on accessibility or use:

- ALS Patients:
    - ALS patients, who have limited mobility due to progressive muscle weakness, can use this prototype to regain some independence and control over their lift chairs.
    - Restrictions: ALS patients might have varied degrees of speech impairment or difficulty in articulating voice commands clearly. The voice recognition module should be sensitive enough to understand their speech patterns and should be customizable for individual users.

- Elderly Individuals:
    - Elderly individuals who experience mobility challenges or have difficulty operating manual lift chair controls can benefit from the convenience of voice-controlled automation.
    - Restrictions: Some elderly users may not be familiar with voice-controlled technology, so the system should be designed with simple and intuitive voice commands.

- People with Motor Disabilities:
    - People with various motor disabilities, such as muscular dystrophy or spinal cord injuries, can find this technology helpful for operating their lift chairs without relying on physical button presses.
    - Restrictions: Some users might have limited vocal capabilities, requiring the system to have alternative input methods, such as switches or assistive devices that can trigger the voice recognition module.

- Caregivers and Assistants:
    - Caregivers and assistants who provide support to individuals with mobility limitations can also benefit from the ease of use and efficiency offered by the voice-controlled lift chair.

- Restrictions: Caregivers might need to learn how to set up the system and assist users in the voice training process (if applicable) to ensure accurate recognition of commands.

- Users with Speech Impairments:
  - Individuals with speech impairments due to conditions like cerebral palsy or stroke may find the system useful if it can be adapted to recognize alternative communication methods, such as sign language translation or eye-gaze systems.
  - Restrictions: The voice recognition module should support multiple input modalities to accommodate users with different speech impairments.

3.3 Accessing/setting up the System

Setting up the physical prototype with servo motors and voice recognition technology for controlling the lift chair requires careful attention to detail. Below is a detailed step-by-step guide to ensure a successful setup:

Step 1: Unboxing and Preparation:

- Carefully unbox the prototype case, ensuring all components are present and undamaged.
- Check that the Arduino, voice-recognition microcontroller, microphone, and servo motors are securely housed and mounted in their respective compartments.

Step 2: Power Connection:

- Connect the power source (power adapter or batteries) to the Electronics Compartment (Level 1) of the prototype case.
- Ensure a stable power supply to avoid any disruptions during usage.

Step 3: Insert Remote Control:

- Slide the remote control for your lift chair into the Remote-Control Compartment (Level 2) of the case.
- Make sure the remote fits snugly and securely in the insert-slots provided.

Step 4: Voice Activation Setup:

- Initiate the voice recognition module's setup process (if applicable) following the manufacturer's instructions.
- Train the voice recognition module to recognize your specific voice commands, if needed, by providing clear and distinct voice samples.

Step 5: Mounting the Prototype:

- Place the prototype case on a stable surface near the lift chair. Ensure it is within a suitable range for the voice recognition module to function effectively.

Step 6: Voice Activation:

- The voice recognition is activated as soon as there is a supplied power source. As a result, the system will be ready to respond to voice commands.

Step 7: Issue Voice Commands:

- Clearly speak voice commands to control the lift chair's movement. For example, say "Lift up" to raise the chair or "Lift down" to lower it.

Step 8: Adjusting Features for Personal Use:

- For adjustable parameters, such as the servo motor location, servo speed movement, microphone voice sensitivity, etc., refer to section 6 of this report to manually customize any desired settings.

Step 9: Done!

3.4 System Organization & Navigation

In general terms, the organization of the system consists of the main components, accessories, and attachments, which are interconnected to achieve the overall functionality of the prototype.

Main Components:

- Prototype Case: The central component of the system is the prototype case, designed with multiple levels to house the main components securely. It provides a protective enclosure for all the internal parts and ensures their proper functioning.

- Electronics Compartment (Level 1): This level holds the main electronic components of the system, including the Arduino microcontroller, the voice-recognition microcontroller, and any necessary power source. These components act as the brain of the system, processing user commands and controlling the actions of the lift chair.

- Remote Control Compartment (Level 2): The second level contains the insert-slots where the lift chair remote control is securely placed. This is the interface through which the system interacts with the lift chair's control panel.

- Servo Motor Compartment (Level 3): At the bottom level, the casing houses the servo motors, which are mounted strategically to align with specific buttons on the inserted remote control. The servo motors' movement is controlled by the Arduino microcontroller to simulate manual button presses.

Accessories and Attachments:

- Microphone: An external microphone is integrated into the system to capture voice commands from the user. The microphone is connected to the voice-recognition microcontroller for voice processing.

- Lift Chair Remote Control: The user's existing lift chair remote control is an essential accessory for the system. It is inserted into the Remote-Control Compartment and serves as the bridge between the prototype and the lift chair.

Connections and Interactions:

- The Arduino microcontroller in the Electronics Compartment connects to various components, including the voice-recognition microcontroller and servo motors. It processes the user's voice commands and generates signals to control the servo motors accordingly.
- The voice-recognition microcontroller is linked to the microphone to capture the user's voice. Once it interprets the voice commands, it communicates with the Arduino microcontroller to trigger the corresponding servo motor actions.
- The servo motors in the Servo Motor Compartment are connected to the Arduino microcontroller, which precisely controls their movements to press the appropriate buttons on the remote control.
- The lift chair remote control, once inserted into the Remote-Control Compartment, interacts with the servo motors through the prototype casing. When the servo motors push the remote control's buttons, it sends the same signals as manual button presses, effectively controlling the lift chair.
- The Arduino microcontroller is connected to a power source to ensure everything is supplied with electricity.

The organization and connections of these main components, accessories, and attachments work harmoniously to enable voice-controlled operation of the lift chair, offering users enhanced independence and comfort in their day-to-day activities.

3.5 Exiting the System

To properly put away the system, follow these simple steps:

Step 1: Disconnect Power and Deactivate the System:

- Unplug the power source from the Electronics Compartment (Level 1) of the prototype case.
- Ensure that all power connections are properly disconnected to prevent any unintentional activation or usage.
- Please note that the system is automatically deactivated once the power source is disconnected.

Step 2: Remove Remote Control:

- Take out the lift chair remote from the Remote-Control Compartment (Level 2) of the case.
- Ensure the remote is stored safely and securely for future use.

Step 3: Securely Store the System:

- Store the entire system in a safe and dry location.

Step 4: Done!

By following these steps, the system will be properly put away and ready for future usage whenever needed. Always ensure that the system is stored in a safe and clean environment to prolong its lifespan and ensure its functionality when needed again.

4 Using the System

The voice-activated chair remote is a user-friendly control device that allows users to control the movement of their chair using voice commands. The remote features four main functions: "Head up," "Head down," "Leg up," and "Leg down." These functions enable users to adjust the position of the headrest and leg rest of the chair to achieve optimal comfort and support. The voice-activated chair remote utilizes voice recognition technology to interpret user commands and initiate the corresponding chair movements. Below is a detailed description of each function.

4.1 Function: Head Up

The "Head up" function allows users to raise the headrest of the chair using a voice command. By activating this function, the chair's headrest will gradually elevate, providing the user with additional support and promoting a more upright sitting position. To activate the "Head up" function, the user needs to clearly speak the voice command "Head up" into the chair remote's built-in microphone. The voice recognition system will then process the command and initiate the corresponding action. Upon recognizing the voice command "Head up," the system sends the signal to the servo motor which pushes the desired button on the remote. The headrest of the chair will then begin to lift by one increment. Activating the voice command again will raise the headrest by another increment.

4.2 Function: Head Down

The "Head down" function enables users to lower the headrest of the chair using voice commands. This function is helpful when the user wants to rest in a more reclined position. To initiate the "Head down" function, the user should clearly speak the voice command "Head down" into the chair remote's microphone. The voice recognition system will process the command and execute the corresponding action. Upon recognizing the voice command "Head down," the system sends the signal to the servo motor which pushes the desired button on the remote. The headrest of the chair will then begin to descend by one increment. Activating the voice command again will lower the headrest by another increment.

4.3 Function: Leg Up

The "Leg up" function allows users to elevate the leg rest of the chair using voice commands. Activating this function provides users with the ability to elevate their legs and improve circulation. To enable the "Leg up" function, the user must speak the voice command "Leg up" clearly into the chair remote's built-in microphone. The voice recognition system will interpret the command and trigger the corresponding action. Upon recognizing the voice command "Leg up," the system sends the signal to the servo motor which pushes the desired button on the remote. The leg rest of the chair will then begin to lift by one increment. Activating the voice command again will raise the leg rest by another increment.

4.4 Function: Leg Down

The "Leg down" function permits users to lower the leg rest of the chair using voice commands. This function is beneficial when the user wishes to return to a seated position with

their legs down. To activate the "Leg down" function, the user needs to clearly utter the voice command "Leg down" into the chair remote's microphone. The voice recognition system will process the command and execute the corresponding action. Upon recognizing the voice command "Leg down," the system sends the signal to the servo motor which pushes the desired button on the remote. The leg rest of the chair will then begin to descend by one increment. Activating the voice command again will lower the leg rest by another increment.

5 Troubleshooting & Support

The voice-activated reclining chair remote utilizes voice commands received through a microphone and sends signals to servo motors mounted on the remote to operate the desired buttons. While the system is designed to provide a smooth and accurate user experience, there may be certain error conditions that could arise. Below are the recovery and error correction procedures, organized into relevant sub-sections.

5.1 Error Messages or Behaviors

1. The voice recognition system may occasionally misinterpret the user's voice command, leading to an incorrect action or no response.

Corrective Action:

- Remain calm and clearly repeat the voice command: If the chair remote fails to recognize the initial voice command or executes the wrong action, the user should calmly repeat the correct voice command, ensuring clear pronunciation and enunciation.
- Ensure a suitable environment: Background noise or interference may affect the voice recognition accuracy. The user should attempt to reduce the noise level in their environment to enhance voice recognition performance.
- Use alternative voice commands: The chair remote may have multiple acceptable voice commands for the same action. The user can try using alternative phrasing for the desired function to see if the system recognizes it correctly.

2. A servo motor on the chair remote may malfunction, resulting in an inability to press the desired button effectively.

Corrective Actions:

- Check battery level: Low battery power can cause servo motor issues. The user should ensure that the chair remote has sufficient battery power or replace the batteries if necessary.
- Verify motor connections: Ensure that the servo motors are securely connected to the chair remote. If a connection is loose, reattach it properly.
- Reset the chair remote: If the servo motor continues to malfunction, the user can try resetting the chair remote by turning it off, waiting a few seconds, and then turning it back on.

3. The chair remote successfully sends signals to the servo motors, but the motors fail to push the desired button effectively.

Corrective Actions:

- Check button alignment: Verify that the servo motors are correctly aligned with the buttons on the chair remote. If misalignment is detected, adjust the motor's position accordingly.
- Clean the button area: Dust or debris on the button area may hinder the button press. The user should clean the button area on the chair remote to ensure smooth operation.

- Examine servo motor movement: Observe the servo motor movement while executing the voice command. If the motor's movement seems restricted or abnormal, there might be an obstruction or mechanical issue. Contact customer support for assistance.

4. The chair remote experiences a complete system malfunction, and no functions are responsive.

Corrective Actions:

- Check power source: Ensure that the chair remote has a sufficient power source, either through batteries or charging, and that it is turned on.
- Reset the chair remote: If the system remains unresponsive, try resetting the chair remote by turning it off, waiting for a moment, and then turning it back on.
- Contact customer support: If none of the above actions resolve the issue, the user should contact customer support for further assistance and potential repair or replacement.

5.2 Maintenance

The voice-activated reclining chair remote should undergo regular maintenance, including cleaning, battery replacement, and inspection of servo motors, to ensure proper functioning and minimize potential errors. It is crucial to follow the manufacturer's guidelines and instructions for maintenance and troubleshooting. Additionally, if there are specific error codes or indicators displayed on the chair remote, users should refer to the user manual for specific troubleshooting steps corresponding to those error codes.

5.4 Support

If the user needs emergency assistance and/or system support, they can contact Yahya Mostafa via email at “ymost020@uottawa.ca”.

6 Product Documentation

In this section, we delve into an in-depth explanation of the step-by-step construction process of the final prototype, meticulously covering design considerations that contributed to its development. This detailed breakdown is categorized to reflect the different aspects of the prototype (Physical and Software). Each category is thoroughly explored to highlight its significance in achieving the desired functionality of the prototype.

6.1 Physical Prototype

This section covers all physical aspects of the prototype, including mechanical and hardware components.

6.1.1 BOM (Bill of Materials)

Table 3. Bill of Materials

Part	Qty	Product	Cost per part	Link
Arduino Microcontroller and Cable	1	Arduino UNO Rev3	$35.20	https://store-usa.arduino.cc/collections/boards/products/arduino-uno-rev3
Voice Recognition Microcontroller & Microphone	1	Voice Recognition Module V3 Speed Recognition Compatible with Arduino	$49	https://www.elechouse.com/product/speak-recognition-voice-recognition-module-v3/
Servo motors	3	SG90 9G Micro Servo	$2.80	https://www.amazon.ca/dp/B07F7VJQL5?psc=1&ref=ppx_yo2ov_dt_b_product_details
PLA plastic	1	Plastic for 3D printing	$5	N/A
M2 screws & M2 nuts	6	Screws and nuts for servo motors	$0.75	N/A
Electrical wires	3	Short Jumper Wires	$0.05	N/A
Total	**$102.25**			

6.1.2 Equipment list

The equipment that was needed to build this subsystem are:

- A 3D printer.

- A screwdriver.

6.1.3 Instructions

This section covers how to build the prototype, from 3D printing the casing to connecting the electronics.

6.1.3.1 3D Printing the Casing

To start off, you'll need a casing to house all the different components, from the remote, to the servos to the electronics. Below are the links to the CAD files for each part of the casing.

Bottom Layer – This layer serves as the housing for the "brains" of the prototype. This layer was designed to comfortably hold the Arduino uno, the voice recognition module and the mic. It also has circular holes on its face to allow the passing of the mic and power wires. The four squared holes on each corner is the passage to allow the connection rods to securely keep all layers in place.

https://cad.onshape.com/documents/8f9e06eef86f884c6b92f1fe/w/e6af8fe8bb0a0159c235c60c/e/0ea29f1142439214617c3a5f

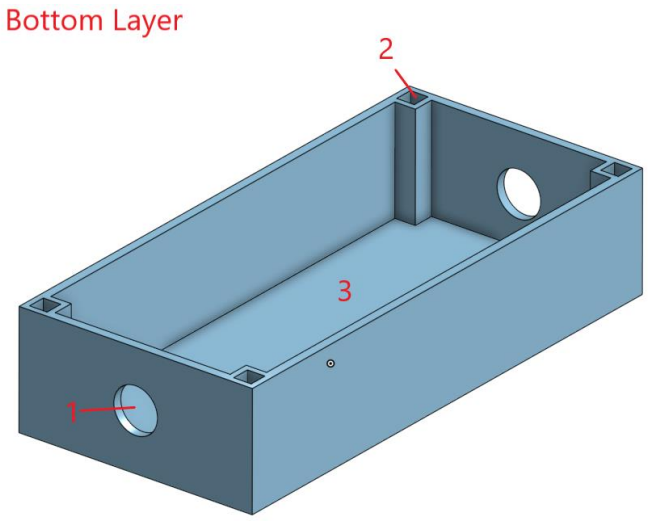

Figure 2. Bottom Layer

(1) Circular holes on the side to allow passage of mic and wires
(2) Connecting Rods cavity
(3) Electronic component housing area

Middle Layer – This layer serves as the housing for the remote. It is designed to be slightly wider than the remote to accommodate all sorts of remotes. Foam or Velcro can be added to ensure a snug fit with your specific remote. This layer also features 2 long passages on both its sides to allow wires from the servos to pass from the top layer all the way to the bottom layer without interfering with the remote.

https://cad.onshape.com/documents/8f9e06eef86f884c6b92f1fe/w/e6af8fe8bb0a0159c235c60c/e/dd73f618fb53b5470d1ca2e6

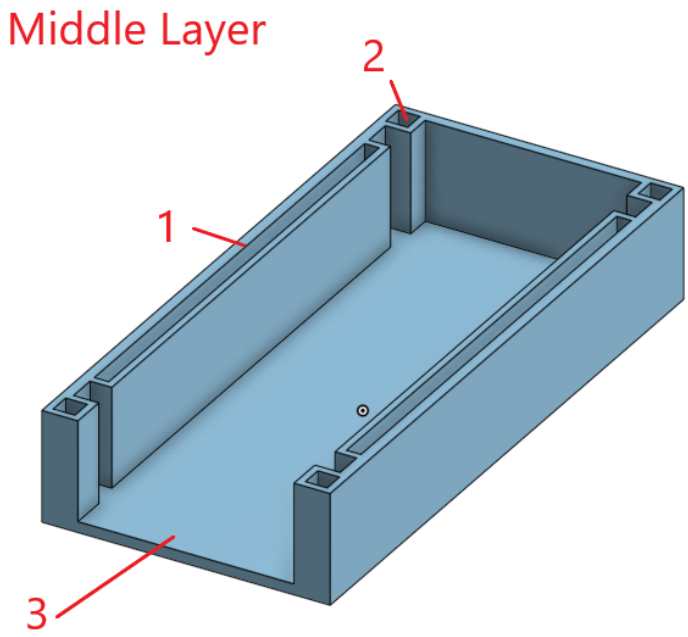

Figure 3. Middle Layer

(1) Servo wire passage
(2) Connecting Rods cavity
(3) Remote housing area

Top Layer – This layer serves as the servo mounting area. This layer is designed to mount 3 9g servos for the specific button layout of the provided lift chair remote. Little holes on the servo mounts serve as screw points to ensure the servos are securely attached to the piece. Once again, this layer also features 2 long passages on both sides to allow wires from the servo to directly attach to the electronic components in the bottom layer without interfering with anything along the way.

https://cad.onshape.com/documents/8f9e06eef86f884c6b92f1fe/w/e6af8fe8bb0a0159c235c60c/e/4f7b106e8e38775baf07ae53

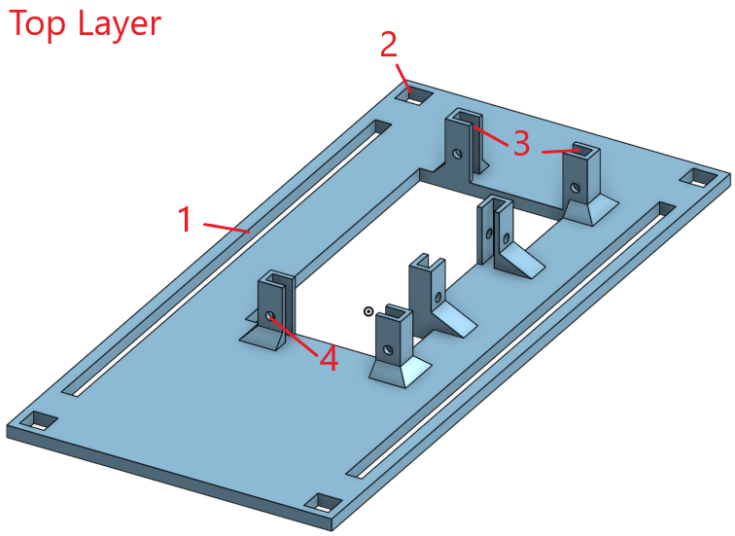

Figure 4. Top Layer

(1) Servo wire passage
(2) Connecting Rods cavity
(3) Servo Mount
(4) Screw hole to secure servos with screws

Connecting Rods – The connecting rods is what holds all the layers securely together. By placing one in each corner, they should have a snug fit and ensure that all layers stay intact with little to no wobble and rattle.

https://cad.onshape.com/documents/8f9e06eef86f884c6b92f1fe/w/e6af8fe8bb0a0159c235c60c/e/c487e35e396e1cddc052655b

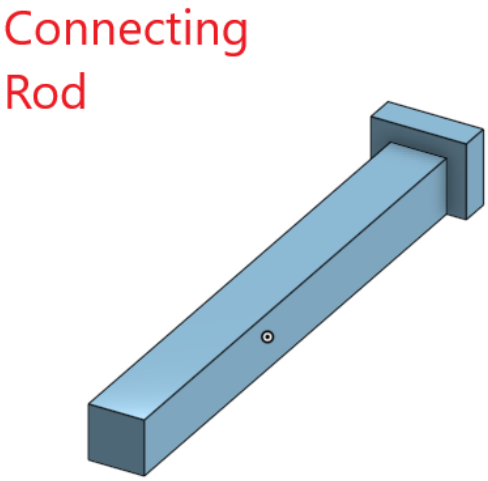

Figure 5. Connecting Rod

Additional Pieces (for customizability)

Universal Top Layer – This alternate top layer was designed to accommodate any button layout on any remote. Compared to the "Top Layer", there are no servo mounts that are fixed on this layer, instead, there are mounting holes to adjust the servo mounts to any desired location (servo mounts CAD can be found next).

https://cad.onshape.com/documents/8f9e06eef86f884c6b92f1fe/w/e6af8fe8bb0a0159c235c60c/e/e016e79eb73e1bd074a15437

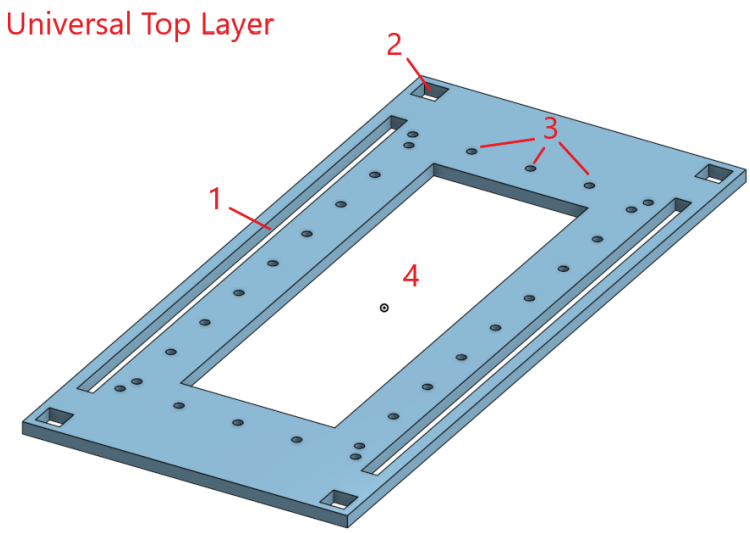

Figure 6. Universal Top Layer

(1) Servo wire passage
(2) Connecting Rods cavity
(3) Servo Mount screw holes to allow servo mounts to be attached anywhere on this layer
(4) Cavity to allow servos to spin and push on buttons

Servo Mount – These servo mounts are designed to fit 9g servos. The screw holes on the bottom allow it to be attached anywhere on the "Universal Top Layer" to adjust to any button layout on any remote. Once again, the servo mounts also feature screw holes on the face to allow a secure connection with screws.

https://cad.onshape.com/documents/8f9e06eef86f884c6b92f1fe/w/e6af8fe8bb0a0159c235c60c/e/13c67b095ac611033dc15d20

Figure 7. Servo Mount

(1) Screw hole to attach Servo Mount to Universal Top Layer
(2) Servo "nest"
(3) Screw hole to attach servo to Servo mount with screws

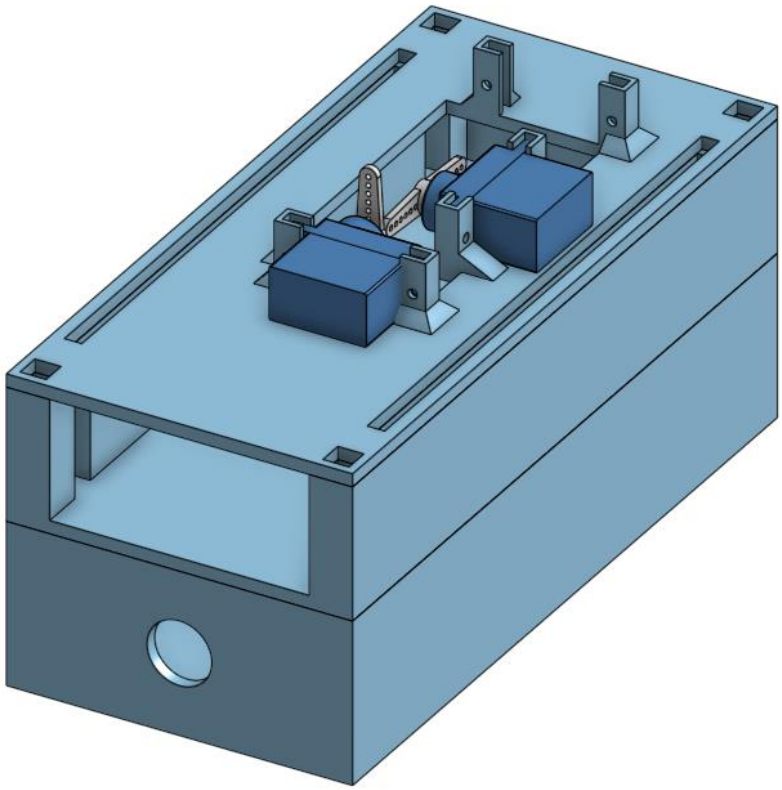

Figure 8. CAD of partially assembled design

6.1.3.2 Connecting the Electronics/Hardware

Please connect the Voice Recognition Module as well as the three Servo Motors to the Arduino Microcontroller as shown in the figure below.

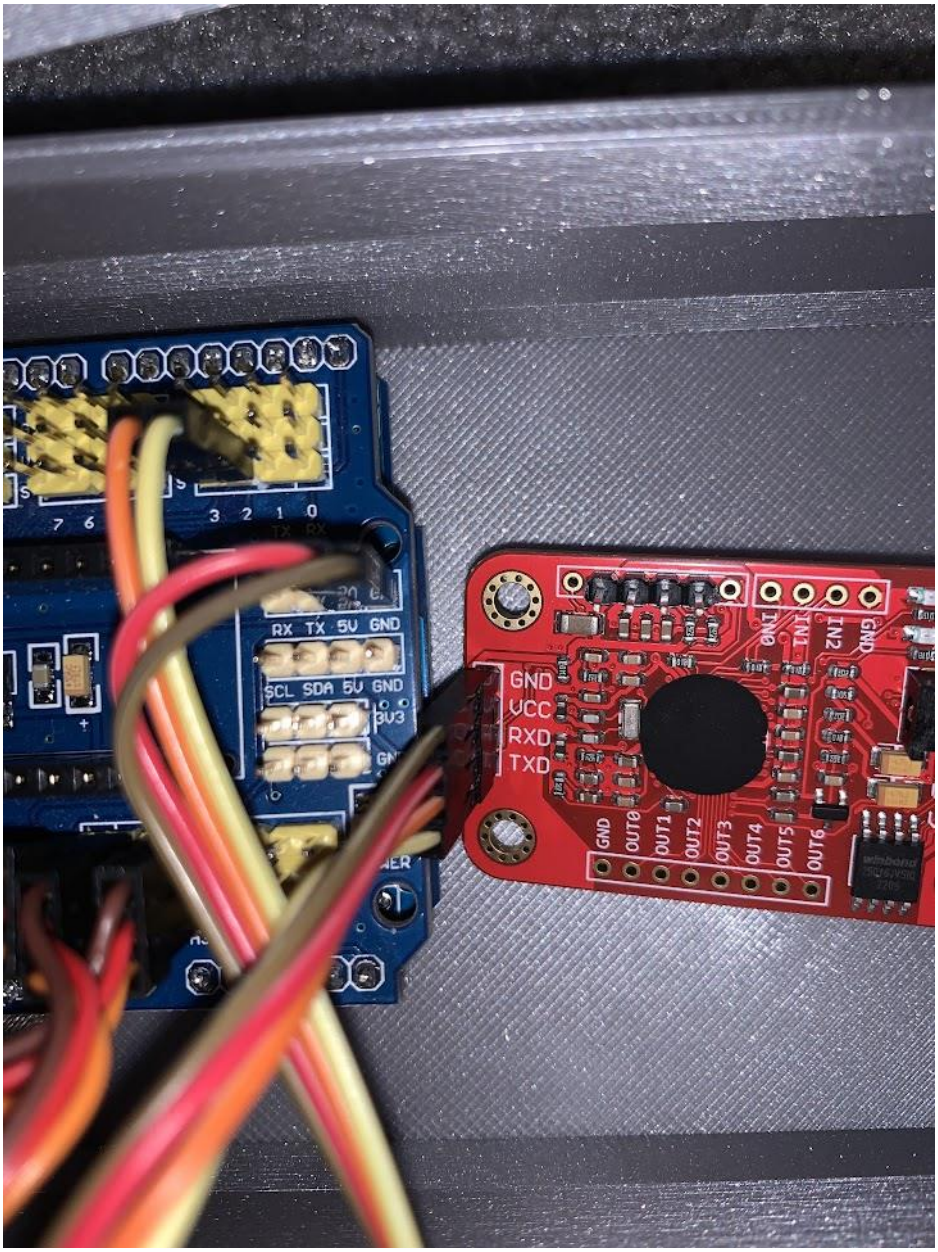

Figure 9. Photo of Voice Recognition Module connection to Arduino

Connect the Brown wire to GND and GND, the Red wire to VCC and 5V, the Orange wire to RXD and pin 3, and the Yellow wire to TXD and pin 2.

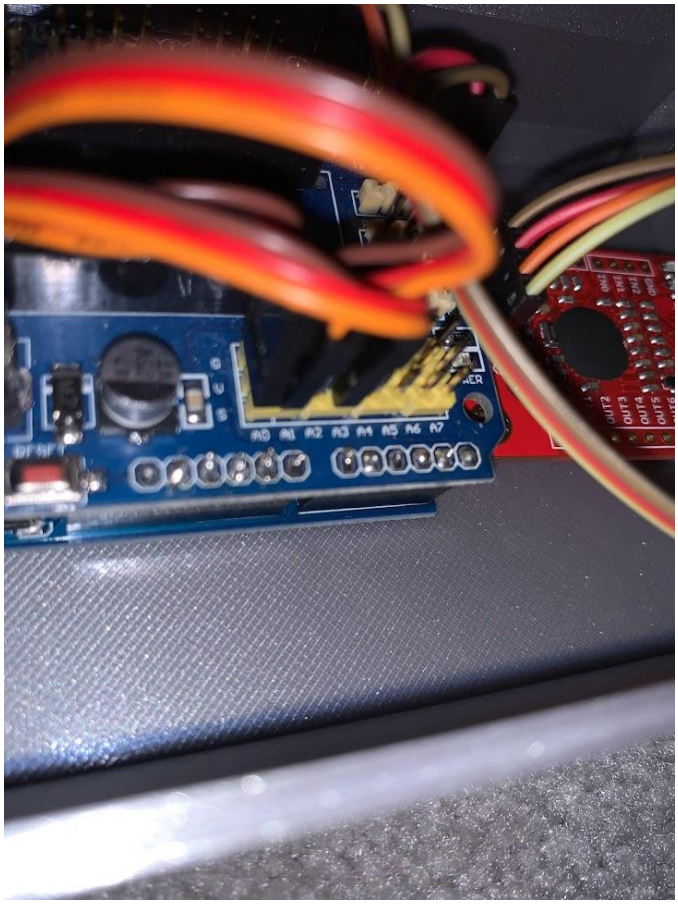

Figure 10. Servo Motors connection to Arduino

Connect the Brown wire to G (Ground), the Red wire to V (Voltage), and the Orange wire to S (pin ID).

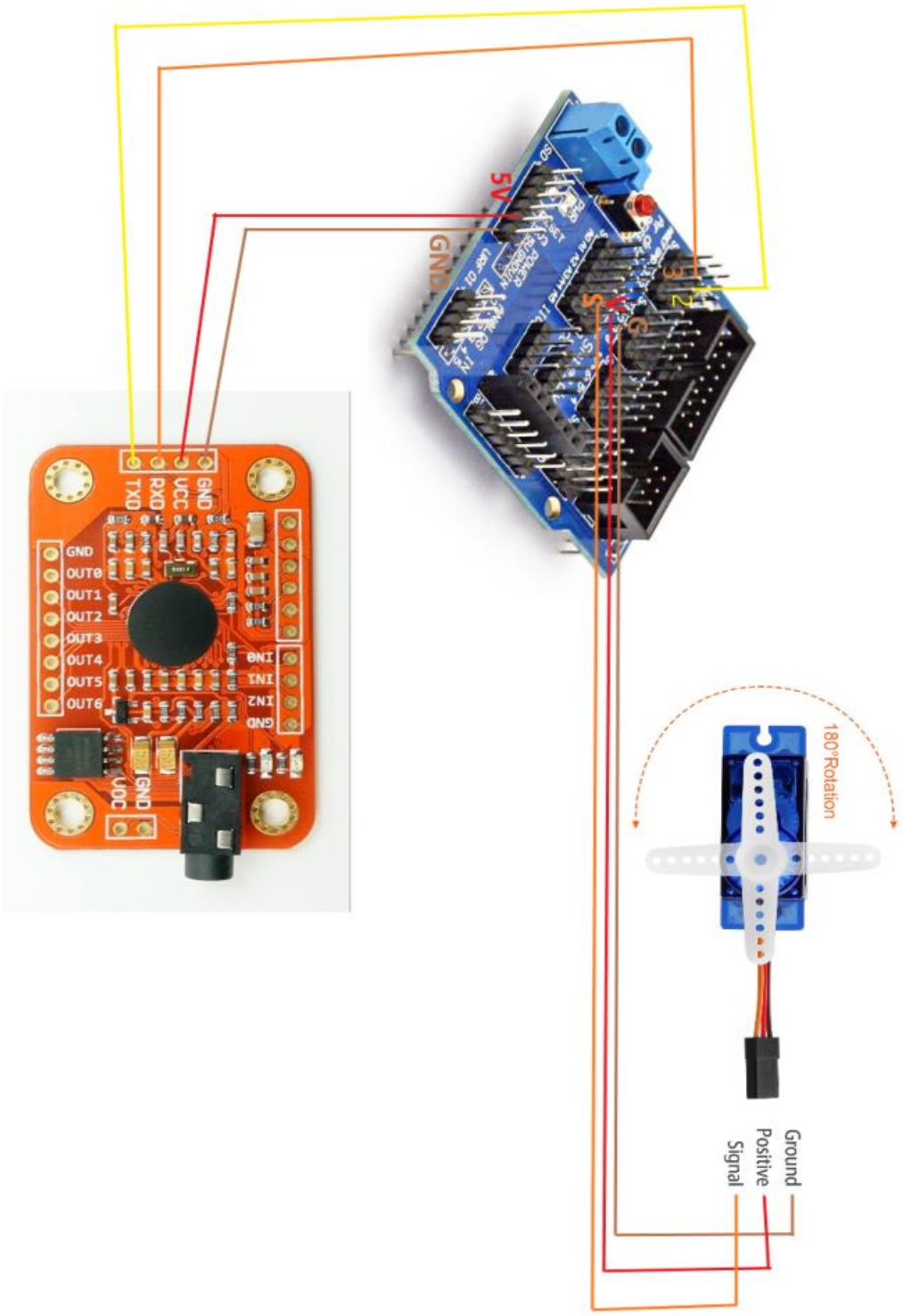

Figure 11. Complete circuit

Servo in Figure 9 shows a connection in A5, for more servo connections, connect the wires in the same order (GVS) from A1 – A5.

Servo Motor – The servo motors used in this prototype are 9g servos. They come with 3 different horns, simply attach and screw on your preferred horn for your specific application.

Figure 12. 9g Servo motor

Small screw is meant to attach the horn to the servo, 2 longer screws are to attach the servo to mounts.

6.2 Software Prototype

This section covers all software aspects of the prototype, including platforms, services, and libraries.

6.2.1 BOM (Bill of Materials)

All software platforms, services, and libraries used for this prototype are completely free to install and use. List of platforms and libraries used:

- Platform: Arduino Desktop IDE
  - Can be installed at this link: https://www.arduino.cc/en/software
- Library: Speak Recognition
  - Can be installed at this link: https://www.elechouse.com/product/speak-recognition-voice-recognition-module-v3/

6.2.2 Equipment list

The equipment that was needed to build this subsystem are:

- A laptop or computer with Arduino Desktop IDE installed.
- A USB printer cable (this cable is used to connect the Arduino microcontroller to the laptop or computer).

- All hardware components listed in the previous section of this report (*6.1 Physical Prototype*).

6.2.3 Instructions

Step by step instructions on how to build this specific subsystem:

- Platform Installation and Use: A detailed and friendly guide on how to install and use the Arduino Desktop IDE can be followed at this link:
  - https://www.arduino.cc/en/Guide
- Library Installation and Use: A detailed and friendly guide on how to install and use the Speak Recognition library can be followed at this link:
  - https://www.elechouse.com/elechouse/images/product/VR3/VR3_manual.pdf

6.2 Testing & Validation

Table 4. Physical Prototype Testing

#	Test	Purpose	Verified Product Assumption	Expected Result	Actual Result
1	Position Test	Verify if the servo accurately reaches and maintains the desired positions.	Accuracy	Positioned over 2 buttons	Success
2	Button Push/Resistance Test	Confirm if the servo successfully pushes the button / can overcome any resistance from the button mechanism.	Accuracy	Enough strength to push	Success
3	Durability Test	Assess the servo motor's durability and longevity in repeatedly pushing the button.	Accuracy	Many years of pushes	Success
4	Speed/Response Test	Evaluate the servo motor's speed and responsiveness in pushing the button.	Connectivity	Less than 1 second	Success

Table 5. Software Prototype Testing

#	Test	Purpose	Verified Product Assumption	Expected Result	Actual Result
1	Speech Recognition Test	Validate the basic functionality of speech recognition.	Accuracy of speech recognition	Functional	Success
2	Noise Tolerance Test	Evaluate the system's ability to filter out background noise.	Accuracy of speech recognition	Range of 2m	Success
3	Speaker Independence Test	Determine if the voice recognition works for different speakers.	User accessibility	Should work for any speaker	Success

4	Performance and Response Time Test	Measure the system's response time and overall performance.	User accessibility and Connectivity	Less than 1 second	Success
5	Longevity Test	Assess the program's stability and accuracy over an extended period.	Connectivity	Indefinitely	Success

7 Conclusions and Recommendations for Future Work

Throughout the process of developing this voice-activated remote control, many valuable lessons were learned. The entire project team gained a deeper understanding of the challenges faced by people living with disabilities and developed a greater appreciation for the importance of assistive technology. Also, user-centered design has never been more important, and the project team has worked hard to ensure that the product is as user-friendly as possible.

Looking to the future, we believe that there is still much work to be done in the field of assistive technology. In terms of improving the voice activated remote control, there are many improvements to make. Adding more customization options for users, such as the ability to adjust the sensitivity of the voice recognition module or to changing the language settings would greatly increase the accessibility of the product. Also, making the remote control more durable and resistant to wear and tear by using stronger materials or adding protective casing could increase its longevity.

Assistive technology has the power to transform the lives of people living with disabilities by providing them with greater independence, mobility, and access to information. As technology continues to advance, we are seeing more and more innovative solutions that are making a real difference in the lives of people with disabilities. From voice-activated remote controls to smart home systems to wearable technology, assistive devices are helping people to live more fulfilling and independent lives. As we continue to develop new technologies and improve existing ones, we can create a more inclusive and accessible world for everyone.

8 Bibliography

[1] *Getting Started with Arduino products*. (n.d.). Arduino. https://www.arduino.cc/en/Guide

[2] *Speak Recognition, Voice Recognition Module V3 - ELECHOUSE*. (n.d.). ELECHOUSE. https://www.elechouse.com/product/speak-recognition-voice-recognition-module-v3/

APPENDICES

9 APPENDIX I: Design Files

Table 6. Referenced Documents

Document Name	Document Location and/or URL	Issuance Date
Voice Recognition Module V3 – User Manual	https://www.elechouse.com/elechouse/images/product/VR3/VR3_manual.pdf	2023-07-23
Arduino – Guide	https://www.arduino.cc/en/Guide	2023-07-23
OnShape – CAD of prototype casing	https://cad.onshape.com/documents/8f9e06eef86f884c6b92f1fe/w/e6af8fe8bb0a0159c235c60c/e/0ea29f1142439214617c3a5f	2023-07-23
MakerRepo – Project Link	https://makerepo.com/usmankhan/1680.voiceactivated-reclining-chair-remote	2023-07-23

10 APPENDIX II: Other Appendices

Arduino program used (Can also be found on MakerRepo: *vr_Chair_Commands.ino*):

```
/**
  ***************************************************************************
  * @file    vr_Chair_Commands.ino
  * @author  Yahya M
  * @brief   This file provides a demostration on
              how to control three servo motors by using VoiceRecognitionModule
  ***************************************************************************
  * @note:
        voice control servo motors
  ***************************************************************************
  * @section  HISTORY

    2023/07/23    Initial version.
  */

#include <Servo.h>
#include <SoftwareSerial.h>
#include "VoiceRecognitionV3.h"

// create servo objects to control a servo
Servo servo1;
Servo servo2;
Servo servo3;

// variable for position of servo (90 degrees for horizontal)
int pos = 90;

/**
  Connection
  Arduino    VoiceRecognitionModule
   2   ------->     TX
   3   ------->     RX
*/
VR myVR(2,3);    // 2:RX 3:TX, you can choose your favourite pins.

uint8_t records[7]; // save record
uint8_t buf[64];
```

```
#define group0Record1       (1)
#define group0Record2       (2)
#define group0Record3       (3)
#define group0Record4       (4)
#define group0Record5       (5)
#define group0Record6       (6)

/**
  @brief   Print signature, if the character is invisible,
           print hexible value instead.
  @param   buf     --> command length
           len     --> number of parameters
*/
void printSignature(uint8_t *buf, int len)
{
  int i;
  for(i=0; i<len; i++){
    if(buf[i]>0x19 && buf[i]<0x7F){
      Serial.write(buf[i]);
    }
    else{
      Serial.print("[");
      Serial.print(buf[i], HEX);
      Serial.print("]");
    }
  }
}

/**
  @brief   Print signature, if the character is invisible,
           print hexible value instead.
  @param   buf  -->  VR module return value when voice is recognized.
             buf[0]  -->  Group mode(FF: None Group, 0x8n: User, 0x0n:System
             buf[1]  -->  number of record which is recognized.
             buf[2]  -->  Recognizer index(position) value of the recognized
record.
             buf[3]  -->  Signature length
             buf[4]~buf[n] --> Signature
*/
```

```
void printVR(uint8_t *buf)
{
  Serial.println("VR Index\tGroup\tRecordNum\tSignature");

  Serial.print(buf[2], DEC);
  Serial.print("\t\t");

  if(buf[0] == 0xFF){
    Serial.print("NONE");
  }
  else if(buf[0]&0x80){
    Serial.print("UG ");
    Serial.print(buf[0]&(~0x80), DEC);
  }
  else{
    Serial.print("SG ");
    Serial.print(buf[0], DEC);
  }
  Serial.print("\t");

  Serial.print(buf[1], DEC);
  Serial.print("\t\t");
  if(buf[3]>0){
    printSignature(buf+4, buf[3]);
  }
  else{
    Serial.print("NONE");
  }
  Serial.println("\r\n");
}

void setup()
{
  delay(100);
  /** initialize */
  myVR.begin(9600);

  Serial.begin(115200);
  Serial.println("Elechouse Voice Recognition V3 Module\r\nLift Chair Commands");

  //initialize and attach servos
  servo1.attach(A3);  // attaches the servo on pin A3 to the servo object
  servo2.attach(A1);  // attaches the servo on pin A1 to the servo object
  servo3.attach(A0);  // attaches the servo on pin A0 to the servo object
```

```
  servo1.write(pos);  // tell servo to go to position in variable 'pos'
  servo2.write(pos);  // tell servo to go to position in variable 'pos'
  servo3.write(pos);  // tell servo to go to position in variable 'pos'

  if(myVR.clear() == 0){
    Serial.println("Recognizer cleared.");
  }else{
    Serial.println("Not find VoiceRecognitionModule.");
    Serial.println("Please check connection and restart Arduino.");
    while(1);
  }

  records[1] = group0Record1;
  records[2] = group0Record2;
  records[3] = group0Record3;
  records[4] = group0Record4;
  records[5] = group0Record5;
  records[6] = group0Record6;
  if (myVR.load(records, 7) >= 0) {
    //printRecord(records, 7);
    Serial.println(F("loaded."));
  }

}

void loop()
{
  int ret;
  ret = myVR.recognize(buf, 1000); //listen every 1000 ms (1 second). Change this
for sensitivity
```

```
  if(ret>0){
    switch(buf[1]){
      //chair up
      case group0Record1:
        //make servo push button
        for (pos = 90; pos <= (90 + 60); pos += 1) { // goes from 90 degrees to
90+60 degrees
          // in steps of 1 degree
          servo1.write(pos); // tell servo to go to position in variable 'pos'
          delay(15);  // waits 15ms for the servo to reach the position
        }
        delay(4000);  //wait 4 seconds

        // reset servo back to horizontal position (90 degrees)
        pos = 90;
        servo1.write(pos); // tell servo to go to position in variable 'pos'
        break;

      //chair down
      case group0Record2:
        //make servo push button
        for (pos = 90; pos >= (90 - 60); pos -= 1) { // goes from 90 degrees to
90-60 degrees
          // in steps of 1 degree
          servo1.write(pos); // tell servo to go to position in variable 'pos'
          delay(15);  // waits 15ms for the servo to reach the position
        }
        delay(4000);  //wait 4 seconds

        // reset servo back to horizontal position (90 degrees)
        pos = 90;
        servo1.write(pos); // tell servo to go to position in variable 'pos'
        delay(250);  //wait .25 second
        break;
```

```
      //leg up
      case group0Record4:
        //make servo push button
        for (pos = 90; pos <= (90 + 60); pos += 1) { // goes from 90 degrees to
90+60 degrees
          // in steps of 1 degree
          servo2.write(pos); // tell servo to go to position in variable 'pos'
          delay(15);  // waits 15ms for the servo to reach the position
        }
        delay(4000); //wait 4 seconds

        // reset servo back to horizontal position (90 degrees)
        pos = 90;
        servo2.write(pos); // tell servo to go to position in variable 'pos'
        break;

      //leg down
      case group0Record3:
        //make servo push button
        for (pos = 90; pos >= (90 - 60); pos -= 1) { // goes from 90 degrees to
90-60 degrees
          // in steps of 1 degree
          servo2.write(pos); // tell servo to go to position in variable 'pos'
          delay(15);  // waits 15ms for the servo to reach the position
        }
        delay(4000);  //wait 4 seconds

        // reset servo back to horizontal position (90 degrees)
        pos = 90;
        servo2.write(pos); // tell servo to go to position in variable 'pos'
        break;
```

```
      //head up
      case group0Record6:
        //make servo push button
        for (pos = 90; pos <= (90 + 60); pos += 1) { // goes from 90 degrees to
90+60 degrees
          // in steps of 1 degree
          servo3.write(pos); // tell servo to go to position in variable 'pos'
          delay(15);  // waits 15ms for the servo to reach the position
        }
        delay(3000);  //wait 3 seconds

        // reset servo back to horizontal position (90 degrees)
        pos = 90;
        servo3.write(pos); // tell servo to go to position in variable 'pos'
        break;

      //head down
      case group0Record5:
        //make servo push button
        for (pos = 90; pos >= (90 - 60); pos -= 1) { // goes from 90 degrees to
90-60 degrees
          // in steps of 1 degree
          servo3.write(pos); // tell servo to go to position in variable 'pos'
          delay(15);  // waits 15ms for the servo to reach the position
        }
        delay(3000);  //wait 3 seconds

        // reset servo back to horizontal position (90 degrees)
        pos = 90;
        servo3.write(pos); // tell servo to go to position in variable 'pos'
        break;

      default:
        Serial.println("Record function undefined");
        break;
    }
    /** voice recognized */
    printVR(buf);
  }
  else{
    Serial.println("Nothing recognized");
  }
}
```