T06 - Rover

Abdi Genemo
Dmitrii Fotin
Sead Mohammed
Taylor Hoyt

Contents

Abstract

The goal of the project was to build a research rover with an arm and a claw to pick up objects and a camera and lenses to examine them. The rover was controlled via bluetooth with an app. The final prototype failed to function as expected due to miscalculated weight load on the components of the rover, which could be resolved by using lighter materials and components or more powerful motors. The total cost of the project was approximately $30-40.

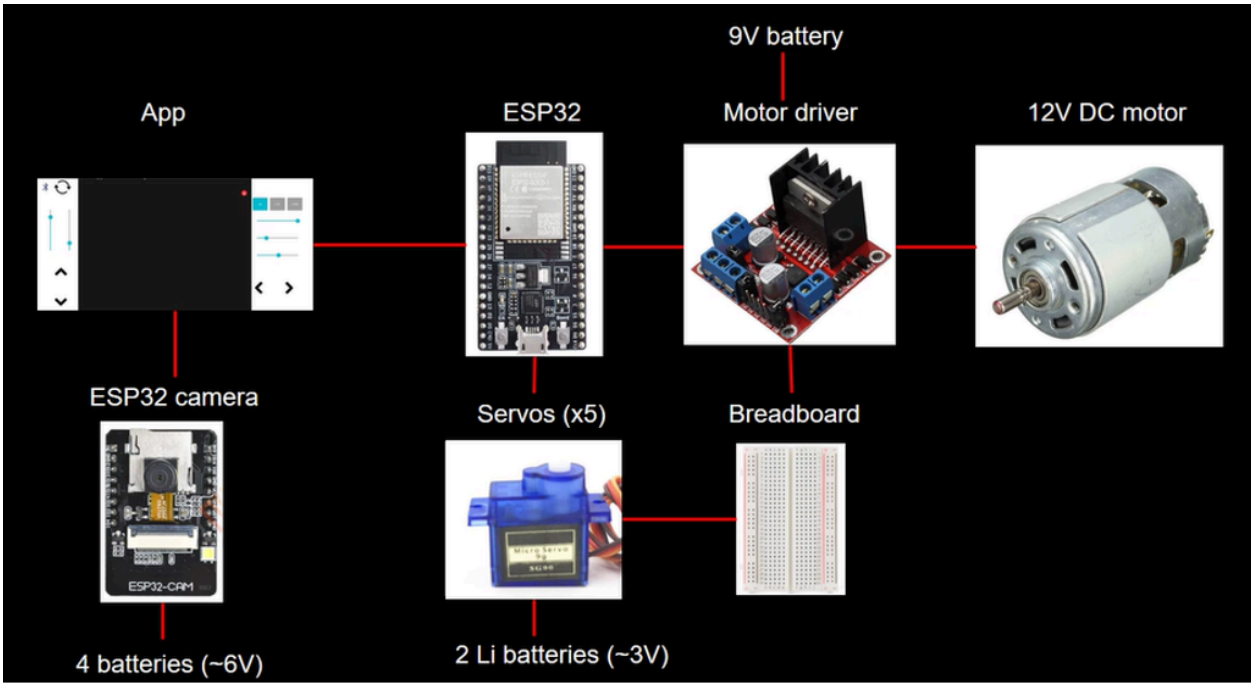
Introduction

The goal of the final project was to build a simplified rover with a scientific purpose to examine and collect specimens. The number of components and tasks involved in building something of the kind was an exciting challenge for the team. The rover included multiple electrical and mechanical elements: an RC car with a mechanical arm, a camera, a claw and a magnifying mechanism - all controlled with an app via bluetooth. The car would have a turning mechanism controlled with a servo motor and a 12V motor to drive it. The arm would rotate in three points: the base with a 360 degree servo motor, and the shoulder and the elbow with a 180 degree servo motor each. It would be attached to the car and would carry an ESP32 camera, a servo motor controlled claw and a servo motor controlled magnifying mechanism with two lenses. This rover would have full driving capabilities and full range of motion with the arm. All of this controlled inside an Android app.

Design Process

Each team member was tasked with a different component to design: Abdi Genemo was in charge of the car, Sead Mohammed was working on the arm design, Taylor Hoyt was designing the claw, the camera housing, and the lens mechanism, and Dmitrii Fotin connected the components with the code and the app. This approach allowed each team member to research, design and prototype a unique component of the overall project. A simple map of all components used in this project is listed in Figure 1.

Figure 1: Overall design of the rover

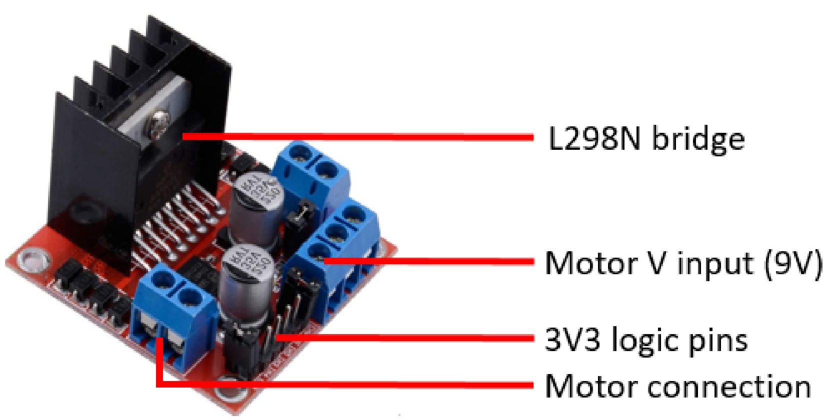


Technical Description

Car and Motor Driver

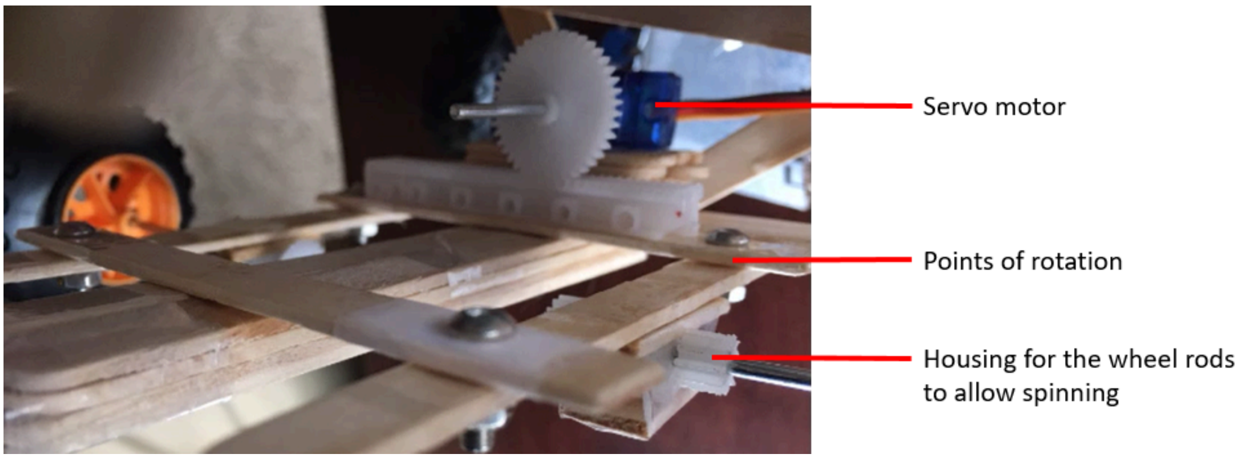
It was decided the car would need at least a 12V motor to drive. The motor was controlled with a L298N motor driver, which accepts 3V3 logical inputs from the microcontroller used for this project.

Figure 2: L298N Motor Driver



The turning mechanism was built based on a design from Instructables.com with minor adjustments. It was made by connecting four craftsticks with bolts to allow change of position, and the wheels were attached to the sides by gluing housing for the wheel rods to the turning mechanism and attaching immovable elements on either side to fixate the position.[1]

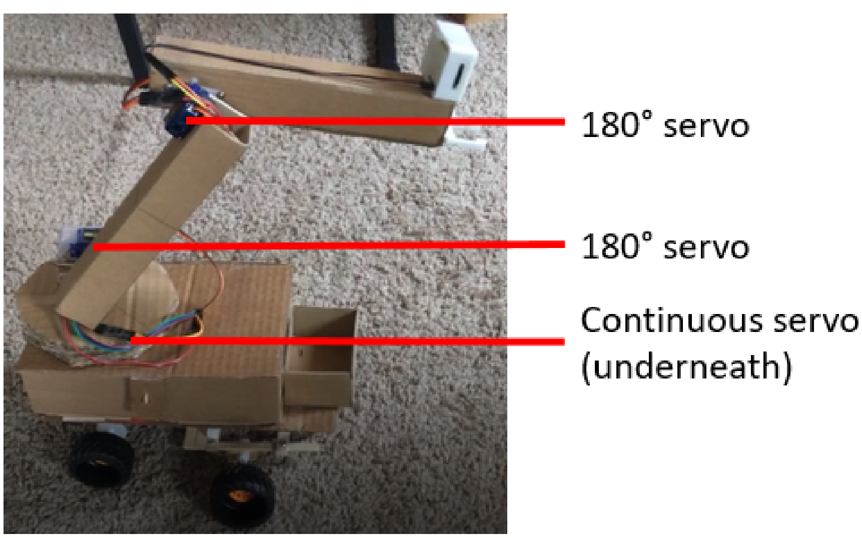
Figure 3: Car turning mechanism



Arm

The arm was made of cardboard and designed to include three servo motors: a continuous servo motor for the base and a 180 degree motor for the shoulder and the elbow.

Figure 4: Arm structure

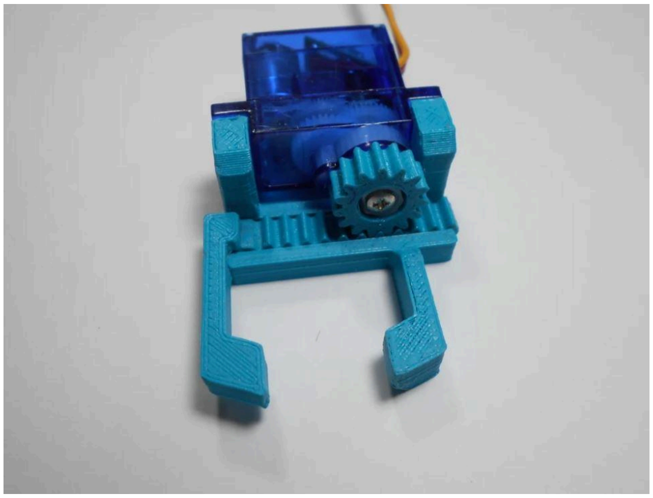


[1] https://www.instructables.com/RC-Car-4/

Claw, lenses and camera housing

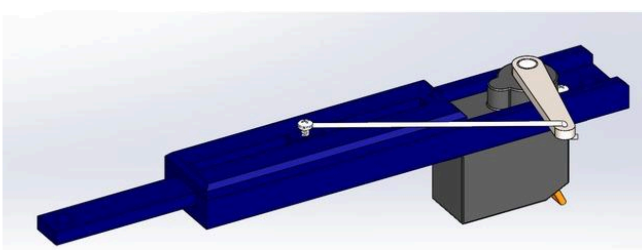
Tinkercad.com was used for the designing of the magnifying lens actuator arm, claw, and camera housing.[2,3] <reference>

Figure 5: Claw design



The claw is a simple three part design all printed on a 3D printer. The linear design keeps the claw lightweight and slide easy. This mechanism was placed at the end of the arm to grasp an object. It is compatible with a 9g servo motor. Printing time: 2 hours 30 minutes.

Figure 6: Lens mechanism

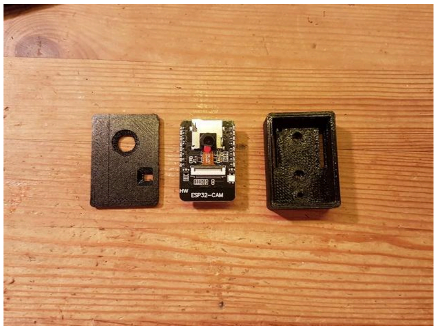


The magnifying actuator arm was a design found on Thingiverse.com, but modified for this project. The only modified part of the design was the actuator arm. It was designed to be a bit longer and include two holes for lenses to fit into. This was to be mounted adjacent to the camera. When activated, it would move the actuator arm in front of the

[2] https://www.intorobotics.com/robot-claw/
[3] https://www.sparkfun.com/tutorials/258

camera to magnify an object. Designed for the same 9g servo motor as the claw. Printing time: 2 hours.[4][5]

Figure 7: Camera housing



The camera housing was a design from Thingiverse.com. A simple two piece housing for the ESP32 CAM. This design made it easy for the actuator arm to rest and slide in front of the camera without any extra part movement. Printing time: 3 hours 20 minutes.

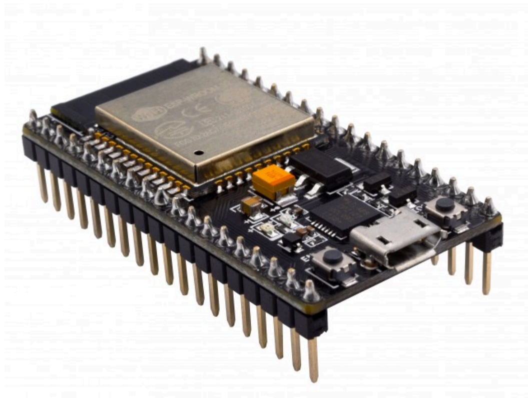
Camera

The camera used in this project is ESP32 CAM. This module has a capability to connect via wifi and transmit the video stream to a local IP address. The camera operated independently of the microcontroller with a separate power source. The code for the camera was included in the ESP32 Arduino IDE examples. The only alterations to the code were to specify the camera model and enter the WiFi credentials.
Figure 8: ESP32 CAM

[4] https://www.thingiverse.com/thing:1875930
[5] https://www.thingiverse.com/thing:2038205

Microcontroller

The rover was operated with the ESP32 microcontroller, which has the capability to connect via bluetooth and is compatible with the Arduino IDE. The six servo motors used in this project as well as the L298N motor driver were connected to ESP32.[6]
Figure 9: ESP32 microcontroller



Power Sources

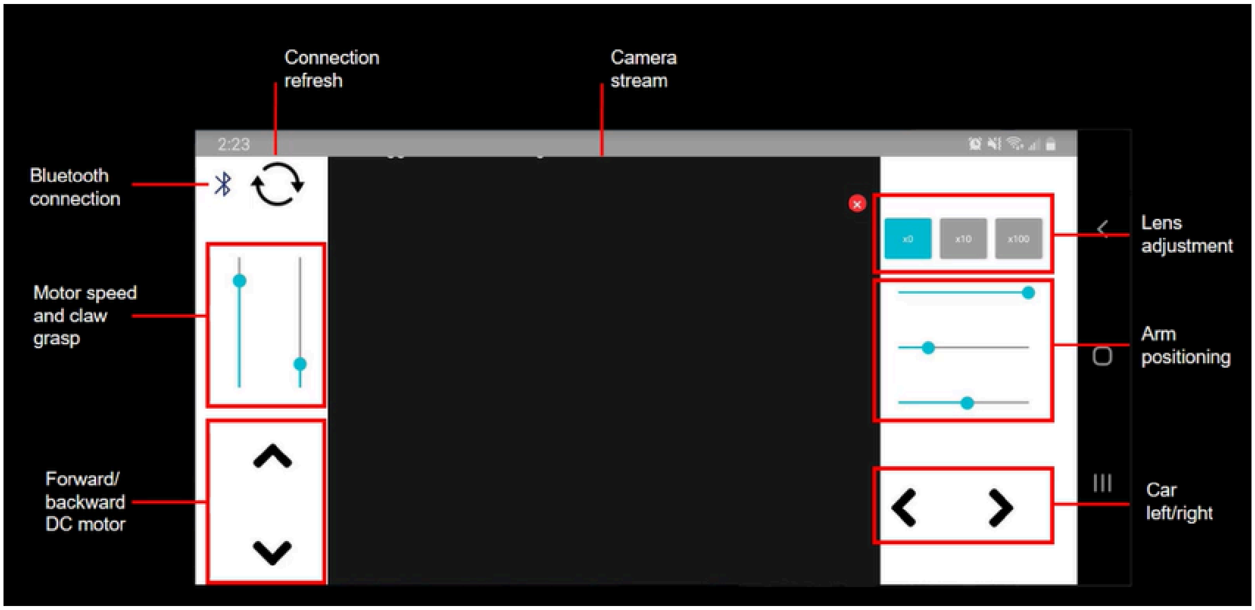
There were three independent power sources used in this project: two Li batteries (2.6V) for the ESP32 and the servos, four Alkaline batteries for the camera (6V), one 9V alkaline battery for the motor.

App

The app was built with Kodular, which is a more robust version of MIT App Inventor. The app was sending text strings via bluetooth to the ESP32 microcontroller. The app included the browser window for the local IP video stream and the controls to connect to a paired bluetooth device (ESP32), to adjust motor speed, claw grasp, lens position, and arm position as well as the driving direction of the car. The app .AIA file can be downloaded [here]. This file can be imported to Kodular to view the layout and the code blocks of the app in detail.

[6] https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

Figure 10: App layout



Code

The code for the ESP32 was written with the background knowledge of operating servo motors, as well as relying on the Arduino Starter Kit examples[7] and online sources for analog write commands with ESP32[8] and bluetooth connection setup.[9]<reference> The code was receiving characters from the app and concatenating them into a string up to a line delimiter. Depending on the string value, the code was then sending commands to the DC motor and the servo motors. In the case of the arm, the claw and the motor speed, the code determined which component value was updated based on the letter identifiers and then parsed out the numeric values and sent them to the respective servos. The full code can be viewed [here](#).

Project Management

Trello was the main application used for keeping track of the progress. It was a great way to visualize what is being currently worked on and what is finished. The team met twice a week to discuss progress and align on the bigger picture. Unfortunately, due to the pandemic the team was unable to collaborate in person, which resulted in mechanical flaws in the final design. Assembling the rover face-to-face would have allowed for faster and more effective debugging and prototyping. Due to these circumstances, the final version was assembled by only two of the teammates.

---

[7] https://programminginarduino.wordpress.com/2016/03/01/project-05/
[8] https://randomnerdtutorials.com/esp32-pwm-arduino-ide/
[9] https://randomnerdtutorials.com/esp32-bluetooth-classic-arduino-ide/

Discussion

The final version of the rover was not fully operational as the pandemic prevented the team from prototyping and debugging quickly and effectively in person. The servo motors were unable to move the weight of the arm with the claw and the camera attached to it and the lens mechanism was not implemented due to time constraints. The whole rover was also too heavy for the 12V DC motor to drive. Reinforcing cardboard and independent power sources with multiple batteries resulted in component weights that could not be operated by the motors used. The strain on the motors could be reduced in a few ways:

- 3D printing the car and the arm components;
- Using voltage dividers to minimize the number of batteries;
- Using more powerful servo motors and DC motor;
- The shoulder and elbow dimensions could have been calculated more accurately with the motor limitations in mind.

The inability to collaborate in person prevented the team from implementing these solutions earlier in the design process.

The total cost of the materials for this project was approximately $25-30, which included 3D printing materials, ESP32, ESP32 CAM, L298N motor driver, mechanical gears and rods, glue and tape.

References

[1] instructables.com, “RC Car”, [Online], Available: https://www.instructables.com/RC-Car-4/. [Accessed 12/10/2020].

[2] intorobotics.com, “Making a Robot Claw”, [Online]. Available: https://www.intorobotics.com/robot-claw/. [Accessed 12 10 2020].

[3] sparkfun.com, “Robot Claw Tutorial”, [Online], Available: https://www.sparkfun.com/tutorials/258. [Accessed 12 10 2020].

[4] MakerBot Thingverse, Servo Linear Actuator (9g) + adjustable backlash Version”, [Online], Available: https://www.thingiverse.com/thing:1875930. {Accessed 12 10 2020].

[5] MakerBot Thingverse, “Servo Linear Actuator (95-137mm) v1”, [Online], Available: https://www.thingiverse.com/thing:2038205. [Accessed 12 10 2020].

[6] Random Nerd Tutorials, “ESP32 Pinout Reference: Which GPIO pins should you use?”, [Online], Available: https://randomnerdtutorials.com/esp32-pinout-reference-gpios/. [Accessed: 12 10 2020].

[7] Programming in Arduino, “Project 05: Mood Cue”, [Online], Available: https://programminginarduino.wordpress.com/2016/03/01/project-05/. [Accessed 12 10 2020].

[8] Random Nerd Tutorials, “ESP32 PWM with Arduino IDE (Analog Output)”, [Online], Available: https://randomnerdtutorials.com/esp32-pwm-arduino-ide/. [Accessed: 12 10 2020].

[9] Random Nerd Tutorials, “ESP32 Bluetooth Classic with Arduino IDE – Getting Started”, [Online], Available: https://randomnerdtutorials.com/esp32-bluetooth-classic-arduino-ide/. [Accessed: 12 10 2020].