HomeRover

Varun Kumar, Hayden Simon, and Nathan Zhu

Department of Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—A system capable of providing a simplistic method of item retrieval for individuals who have found their mobility compromised. Our design will employ a rover with a suction arm capable of lifting items and a user-side control terminal both powered by a software system running on a Raspberry Pi. It is a household friendly product that has a low cost compared to competitors and a low learning curve.**

***Index Terms*— Depth camera, kinematics, LiPo battery, motor, NMOS, object detection, printed circuit board, Raspberry Pi, rover, torque**

I. INTRODUCTION

ACROSS the United States, approximately 39 million Americans face motor impairments [1]. According to Pew Research and ACS estimates, around 7% of Americans have difficulty walking or climbing stairs, characterized as serious ambulatory difficulties. Increased likelihood of this kind of disability increases with age, with adults aged 75 and older and those aged 65 to 74 as the most impacted age groups [2]. A crucial consequence of these ambulatory difficulties is an increased susceptibility to falls as well as difficulty getting in and out of chairs, with emergency departments seeing 3 million older patients each year due to fall injuries [3]. Thus, these affected individuals face challenges to their autonomy, with their condition potentially getting exacerbated by the need to constantly bend down and pick things up.

Current solutions for object retrieval face three main shortcomings that revolve around the two types of existing solutions. Physical, handheld living aids, such as a grabber arm or claw, have a very limited range due to being capped by the user's arm length. Robotics solutions such as TidyBot can utilize vision models with large language models to automate the room cleanup process [4]. However, this solution only exists in a research capacity, being a joint venture with Google and three leading universities. Thus, these robotics solutions are not commercially available and due to the nature of the research, likely require large amounts of funding as well.

To address these limitations and serve our target audience, we propose a cost-effective, intuitive method of object retrieval for individuals with mobility challenges. Taking the form of a user-assisted autonomous robot, it features an interface for user navigation of the rover to an object's general vicinity, autonomy in operating within the vicinity to pick up the object, and the ability to return the object back to the user. By removing their need to pick things up, we hope to ease their ambulatory difficulties and improve their health and quality of life.

II. USE-CASE REQUIREMENTS

Our device is intended for individuals who find their mobility compromised. For our design to be practical, it must be effective in the home environment. From this use-case, we have determined that the rover we are designing needs to be capable of navigating in a room that is approximately 216 square feet because this is the average living room size in the United States [5]. Based on previous robotic systems that have picked items up around the house and research on user experience we have determined that the device needs to be successful 80% of the time for the best experience [6].

Furthermore, our target demographic is the older population, and we are not aware of their familiarity with modern technology. Because of this we have determined that our device needs a user control side that is very tactile and mechanical feeling. From our research and measuring keys on a keyboard we have determined that the keys on the user side need to be at least the size of the keys on a computer keyboard: approximately 0.75 inch by 0.75 inch [7]. For the best user experience, we also need a display on the user side to allow the user to navigate. We need the latency between the camera on the rover and the user display to be less than 0.1 seconds (100 milliseconds) because this is the time that User-Interface and User-Experience has determined that this is the threshold for seemingly instantaneous interaction with a device [8].

Finally, to meet the requirement of being a household device, we need the rover to be capable of navigating at safe speeds around the house and to be able to navigate on different household terrains. We have determined that hardwood, tile, and carpet flooring are the three main types of flooring that almost every house in the United States has. We have also determined that the rover needs to have an absolute maximum speed of 0.5 meters per second because this is the approximate maximum household speed of the iRobot Roomba [9]. In addition to having safe navigation speeds, we need the electronics to be protected from spills in our design and the total cost of our design to be less than $450. We determined that $450 is the ideal maximum price because this is right around the same cost as the cheapest Roomba model on the market [10].

III. ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

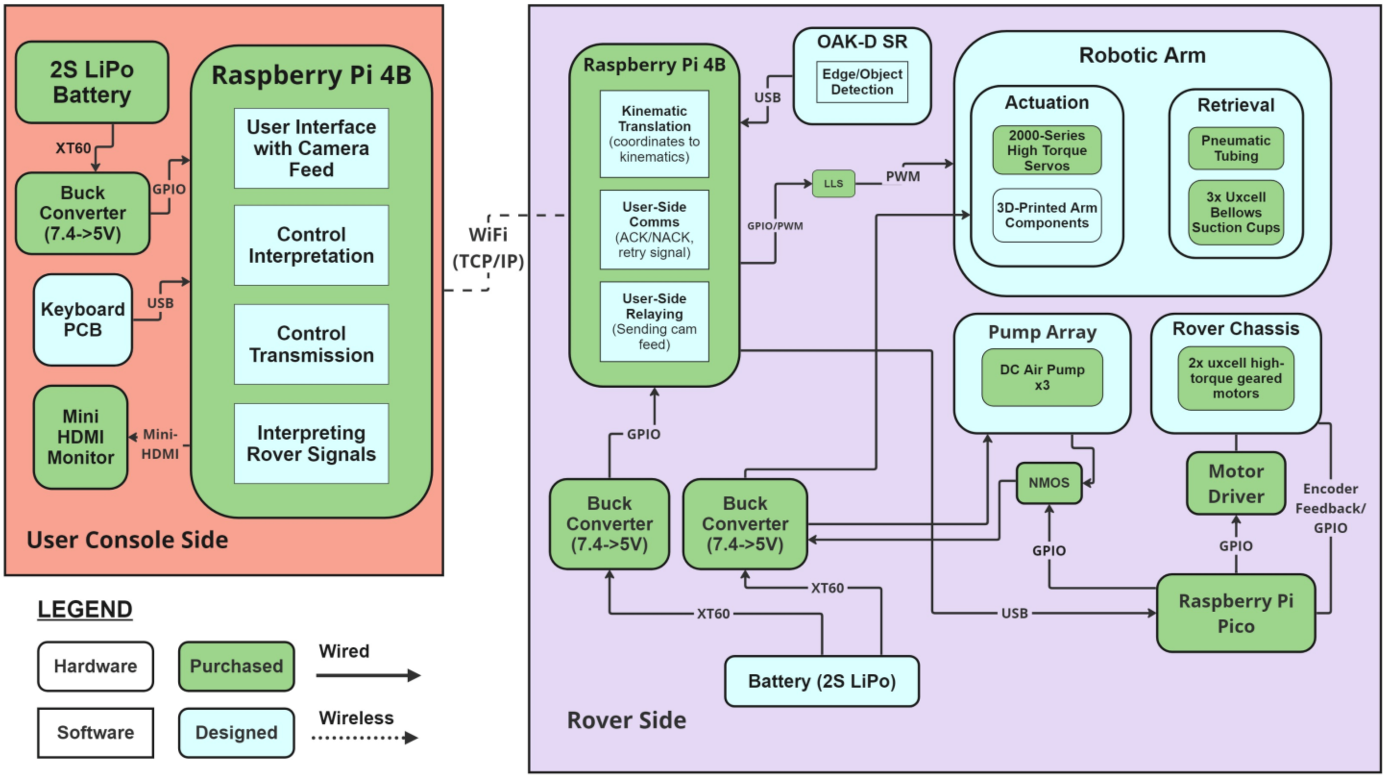


Figure 1. Overarching Block Diagram of System.

A. *Principle of Operation:*

The entire system that we design can be surmised in a single, overarching operation, which uses every subsystem on the above block diagram. When the user drives the robot towards the object they desire to pick up and begin the pickup sequence, there are quite a few steps involved.

First, the user must drive towards the object. This involves using the Controller PCB to give controls to the Raspberry Pi, and the Rover responding appropriately. The user can observe the position of the Rover relative to the object of interest, based on live camera feed from the OAK-D SR, which is displayed on the mini-monitor. Once in range (30 cm), the user presses the Pickup button on the Controller to begin the pickup sequence. Once the button is pressed, the robot gathers data from the OAK-D SR to navigate the robot arm to the correct pickup location. Once the end effector of the robot is near the object, the pumps are activated, and the object is retrieved. The user navigates the robot back to themselves, then presses the Give button, which extends the arm to the highest point possible, such that the user can pick the object up.

B. *Architecture:*

The system can be divided into two subparts, as highlighted on the Block Diagram. On the User-Console-Side, the Raspberry Pi serves as the brain of the system, and a 2S LiPo serves as power. It takes in input via a custom-designed control PCB, which is constructed to be simple for our target demographic. The PCB has an integrated Arduino Nano, whose only purpose is to translate the key presses, which are binary switches connected via GPIO, to USB, such that the Raspberry Pi will be able to interpret them in an efficient manner. Additionally present on the Console side is the mini-HDMI monitor, whose purpose is to display the camera feed that HomeRover sees as it moves to the user.

Once the signals from the Controller are interpreted, they are sent via TCP/IP to the Raspberry Pi 4 on the Rover-Side subsystem. Additionally, Rover Signals, such as RETRY and ACK and NAK are interpreted on this board.

Within the Rover-Side subsystem, there are two overarching goals: Rover Movement and Item Retrieval. To achieve Rover Movement, control signals are taken from the User-Console-Side Raspberry Pi 4 and are forwarded to the Raspberry Pi Pico, which offers the necessary opportunity of a low-latency, bare-metal implementation of a control loop to drive the Rover. Achieving the goal of Item Retrieval is a much more complex process. As determined by the user, when it is time for the Rover to attempt retrieval, the arm will move to the closest object detected in its field of view via a series of kinematic instructions. The coordinates for this movement come from the OAK-D SR and are translated to kinematic instructions on the Raspberry Pi 4 mainboard. Controlled by the Raspberry Pi Pico is the Pump Array. The control for the Array is simple- an NMOS transistor is controlled by a GPIO pin of the Pico, which, when the pumps need to be activated, will allow for the circuit to be completed. At this time, the object should indeed be retrieved and in the Rover’s grasp.

IV. DESIGN REQUIREMENTS

Our design requirements stem from our use-case requirements of accuracy, usability and being a household friendly design.

Accuracy: We need the object identification, claw positioning and arm suction system to execute with 80% accuracy. When the button is pushed to pick up an item the system needs to position itself and execute the task of picking up within 10 seconds. We arrived at these metrics from the capabilities of a similar but more advanced robot the TidyBot which was developed by a team of engineers at Princeton University [4]. We have established the item pickup range to be between 10 centimeters and 30 centimeters because of the length of the arm we have designed and the capabilities of the OAK-D SR camera. In addition to the accuracy of the picking up mechanism, we need the rover to be capable of detecting and picking up planar objects such as books, tablets and medicine boxes and keeping the suction for the entire time the user is returning the rover to them. Because of the requirement to be able to pick up a tablet we determined that the suction system needs to be capable of picking up 700 grams which is the approximate weight of an iPad with a case [11]. To maintain the suction on an object that weighs this much we have determined that we need to have a pump capable of providing 2.21 pounds per square inch of suction pressure to the suction cup and the subsequent item we are picking up. These calculations come from the area of the suction cups being 0.70 square inches or approximately 4 and a half centimeters squared.

Usability: Another important requirement is the usability of the system. We need the entire system to be user friendly and easy to understand for an older individual who does not necessarily have prior experience with modern technology such as smart phones. We have determined that driving the rover needs to be something that anyone can get comfortable with in under 10 minutes of driving. To achieve this, we have decided to go with a very tactile design with four arrow keys and two buttons for the rover to grab an item and to release the item to the user. In addition to having large buttons, we think that latency is a very important factor for a usable device. From our research on User-interface and User-experience we have found that 0.1 seconds is the time for something to be seemingly instantaneous and less than 1 second for it to feel like the user is interacting with a computer [8]. Because of this metric we have determined that we want the Raspberry Pis to communicate with each other over Wi-Fi in under 100 milliseconds. We have also determined that we need the electrical signals from the buttons being pressed to make it to the user side Raspberry Pi in under 20 milliseconds and for signals being sent out from the Rover side Raspberry Pi to propagate in under 20 milliseconds. The sum of these three paths gives us a critical path of 140 milliseconds between the user pressing the forward button and the rover reacting and moving forward. In addition to latency being important for the user experience, we have determined that we need the user to be able to see what the rover sees for the best navigation. Our design will have a Raspberry Pi screen which will display this to the user and will notify the user when they are within the previously mentioned range of 10 to 30 centimeters. We have also determined that battery life is a very important consideration for the usability of this product. We believe that 1 hour of driving between recharges is the minimum battery life we should achieve. To achieve this our design needs to use a 2S LiPo battery with a minimum of 7000 milliamp hours. We determined this by summing the power consumption of all devices on the rover and found that we needed 10,860 milliamp hours if we were to run the device at full speed, with the suction always on and maxing out the processing capabilities of the Raspberry Pi and the OAK-D SR camera; this is not something that will occur in the usage of the rover so 7000 milliamp hours or greater will suffice.

Household friendly design: Since we are targeting use in a household setting it is very important for our design to be safe while still being capable. To achieve this, we have determined that we need the rover to have an absolute maximum speed of 0.5 meters per second; we arrived at this limit based on the maximum speed of a Roomba [9]. While the maximum speed will be 0.5 meters per second, we are trying to target a slightly slower speed of approximately 0.3 meters per second because this is the speed Roomba's normally operate at within a home. We need this driving speed to be achieved on all three types of flooring we have identified: carpet, hardwood, and tile. In addition to the safety aspect of driving speed we have also determined the rover needs to be made from durable materials that will not cause damage to household objects if it collides with them. Because of this, we have decided to 3D print most of our design and only use metal for the bottom mounting plate which will be the structural basis. We also want our system to be able to withstand spills so we will be designing covers for all electrical components on the rover. Lastly, we need the total cost of our design to be less than $450; this price is close to the cheapest iRobot Roomba on the market [10].

V. DESIGN TRADE STUDIES

For our design we had to consider many factors when choosing how to solve our problem: we had to figure out the pick-up mechanism, we had to figure out the drive train and steering methodology, we had to choose an adequate type of sensing, we had to determine communication protocols and we had to select hardware powerful enough to execute the computations we want the rover to perform.

A. *Pick-up mechanism*

As previously mentioned, we need our suction pick up mechanism to be capable of lifting 700 grams. Given this metric we know the force acting down on our suction mechanism can be determined by the equation:

$$F = mg \tag{1}$$

$$F = 0.700 * 9.81 \tag{2}$$

We get a resulting force of 6.867 Newtons. This result will allow us to calculate the pressure we need to create within our suction tubes to be able to pick up an item and offset the forces of gravity. Furthermore, we know that pressure can be determined by the equation:

$$P = F/A \quad (1)$$

$$P = 6.867/0.00045 \quad (2)$$

Giving us 15260 Newtons per square meter or approximately 2.21 pounds per square inch. From this we determined that a servo powering a syringe to create suction would be capable of achieving the necessary pressure to lift this item; however, we realized that stability would be an issue if we used only one suction cup, so we decided to scale up our design to 3 suction cups in a triangular arrangement to stabilize the arm with larger items. Our design did not have room for the footprint of three large servos so we determined that our best option was to use DC power air pumps which would create the suction directly with a smaller footprint.

B. Drive Train

Safe and efficient navigation is vital to the success of our design. There were 3 main categories we discussed: steering methodology, suspension type and motor layout.

Steering Methodology:

We began by discussing different forms of drive trains for a rover; among these we considered differential steering, tank drive, omnidirectional drive, and Ackermann steering. We immediately ruled out Ackermann steering because we wanted to 3D print modules for the wheels and having a steering axel would complicate our design. We also ruled out omnidirectional drive because the kinematics would be unnecessarily complex for the design we are trying to create. This left us with differential steering and tank drive. Differential steering and tank drive are very similar, both involve different rotations per minute for each side of the rover. Our research led us to believe that tank drive, which is often referred to as skid steering, is bad for driving across carpet. Because we want to be able to drive across all types of flooring, we decided that differential steering is ideal for our setup, and we designed our drive train as such.

Suspension type:

After we determined the steering methodology, we began researching different suspension types that would allow us to navigate around a house. We discussed Rocker-Bogey suspension, independent suspension, and dependent suspension. We quickly realized that using independent suspension or Rocker Bogey suspension, while being able to help us balance the rover and offset the weight of the suction claw at the front, would be overly complex and unnecessary for our use-case requirements. We decided that a dependent suspension with 2.5-inch wheels would be able to navigate all the terrains we deem necessary for the success of the rover.

Motor Layout:

After we had settled on a suspension and a steering methodology, we were able to determine the motor layout we wanted to employ. When we were designing this part of the drive train we used to methods of analysis: calculations for the rover speed and cost analysis. To start we needed to determine the necessary rotations per minute to achieve 0.5 meters per second; from this we would be able to find a motor and a gearbox and finish designing our drive train. The equation we used to calculate the rotations per minute necessary is:

$$v = \frac{(\pi D)x}{60} \quad (1)$$

Where v is the velocity of the rover, D is the diameter of the wheels and x is the resulting rotations per minute we need to achieve our velocity. Plugging in our requirements we get:

$$0.5 = \frac{\pi 0.0635x}{60} \quad (2)$$

And when we solve for x we determined that the necessary rotations per minute we need for our design is 150. After we determined this speed, we were able to look for motors and we found the uxcell Gear Motor with Encoder DC 12V 201RPM Gear Ratio 21.3:1 which was perfect for our design. Given the cost of $20 we determined that purchasing two motors was more than sufficient and then we designed a drive train that included an inline timing belt which would keep the driving kinematics simple. We placed the motors at the back of the design going with a Rear-Wheel Drive system to offset the weight of the claw at the front.

C. Sensing

Arguably one of the most important subsystems crucial to the success of our design is the concept of sensing, particularly of camera choice. Being able to both detect an object in the general vicinity of our rover as well as being able to output useful information to dictate our arm movement is of utmost importance to achieving our mission and serving our use case.

When choosing an adequate camera, we considered numerous types of cameras before choosing the OAK-D SR camera, each with their own set of limitations that shied away from the OAK's advantages. The first main difference between the OAK and other cameras was the presence of on-device capabilities. Using the OAK's RVC2 chip, it is capable of running custom AI models, object detection, video/image encoding, 3D edge detection, and 3D feature tracking, to name a few [13]. According to the Luxonis documentation, RealSense stereo cameras do not have any of these capabilities on-device [12]. Due to the constrained timeline of the project and managing a reasonable scope, we thought the presence of these on-device features would make the software and detection more achievable while remaining a challenge in figuring how to correctly establish a pipeline and interface with these different features.

Another main difference between the cameras was the method each camera used to perceive depth. For an OAK-D, this takes the form of passive stereo depth perception, which uses "disparity matching to estimate the depth of objects and scenes" through the usage of a stereo camera pair [14]. The reason this is favorable for our application was because disparity matching is unsuitable for blank, featureless surfaces, making its highest efficacy occur in the contrast of an object and its blank surrounding, our proposed use case. On the other hand, most commercial cameras utilize lasers to determine depth, specifically through 2D LiDAR. One such example is the suite of projects made by LSLIDAR, whose 2D LiDAR "is designed to emit only a single beam onto the target object" [15]. What this means that through this single 2D sweep and resulting 2D plane, if we were to use these cameras, we are limited by the height of our object, and it could be less effective on irregular shaped objects that flit in appearance with the 2D plane. Thus, using the OAK and its stereo depth perception, we can scan more effectively for a specific object.

D. Communication Protocols

To achieve full interconnection with the various parts of our system, such as transmitting control signals from the user side to the rover side, sending ACK/NAK/RETRY signals from the rover to the user to establish fault tolerance, and sending the camera live feed from the rover to the user, we need a robust wireless communication paradigm. The specific aspects we identified as the advantages of interest are the throughput and range of the protocol.

Protocol [16]	Throughput	Range	Latency
Wi-Fi	11 Mbps	32m indoors	150ms
Bluetooth	800 Kbps	5-30m	200ms

Table I: Table showing analysis of Wi-Fi and Bluetooth Protocols

Because of our requirement of sending a live camera feed, our most emphasized category is throughput/bandwidth. As shown in the graph, the data rate for Wi-Fi is much higher than that of Bluetooth, which was a primary factor in influencing our design decision in enabling our mini-HDMI camera on the user side. In addition, the increased range in Wi-Fi is significant in that it will allow for a good signal throughout our user space of a typical living room. This has the additional effect of enabling a high data rate because with a consistent, high signal, there will be less timeout and retry periods. Additionally, in a noisy environment, being able to broadcast a strong signal will lead to increased communication strength, which is especially important in RF-heavy areas. Lastly, we also observe that Wi-Fi has lower latency than that of Bluetooth, which highlights how Wi-Fi better suits our latency metrics in our design requirements by aiming for a more responsive experience for the user with the faster message passing within our full system.

E. Hardware

When choosing hardware for the design we knew we needed a computer that could receive the camera footage processing an object detection algorithm and calculating the kinematics for the suction claw. Because of this requirement we determined we needed a single board computer rather than a smaller device such as a microcontroller. When we were determining what type of computer we would need we were choosing from a Jetson Nano, a Raspberry Pi 4 and the AMD KRIA KR260. When choosing from these computers we considered the factors of cost, performance, and power consumption.

Computer	Cost	Performance	Power Consumption
Jetson Nano	$149	Middle	+5V, 2A
Raspberry Pi 4	$55	Worst	+5V, 3A
AMD KRIA KR260	$350	Best	+12V, 3A

Table II: Table showing analysis of Single Board Computers

Overall, all these boards would have been capable of running the software interface on the rover side, but we determined that the Jetson was overkill because we do not plan on utilizing its machine learning and artificial intelligence capabilities [x]. We also determined that the power consumption of the AMD KRIA was too much for our design and that the price was significantly out of our range; we also would not be using nearly enough of the processing power of the AMD KRIA to justify the cost or the increased power consumption. This led us to the Raspberry Pi 4 which would be sufficient for our design with 4GB of ram and the Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC. After we decided on this board on the rover side of our design, we determined that putting the same board on the user side would make interfacing the design much easier. We determined that the Wi-Fi capabilities and the price-point of the Raspberry Pi 4 justified using it on both ends of our system.

On the user side, we determined the USB protocol was the easiest protocol to connect the buttons from the user input to the Raspberry Pi. From our experimentation we determined that using an Arduino on our PCB will allow us to convert an analog high or low signal to a USB signal that is meaningful for the Raspberry Pi. Due to size constraints, we will be employing an Arduino Nano on the user side PCB.

When implementing hardware on the rover side we are employing GPIO because it allows for a fast control loop. This will allow us to reach our target latency of 20 milliseconds from the Raspberry Pi to the motors. The Raspberry Pi has an operating system meaning it is too slow for the encoder feedback in our control loop. The encoder quadrature style which means it sends two square waves offset from each other; the Raspberry Pi would not be able to check its GPIO pins fast enough for the waves when the encoders are detecting if the voltage is 90° out of phase because of its GPIO sampling frequency of 40 Hertz when the encoder is sampling at 201 Hertz [17][18][19]. To solve this problem, we decided that employing a Raspberry Pi Pico will allow us to implement our control loop on bare metal which will allow us to interpret encoder feedback faster.

VI. SYSTEM IMPLEMENTATION

To better describe the HomeRover System Implementation, this section will be split, as it is in our Block Diagram, into the Rover-Side and Controller-Side subsystems.

A. User-Console-Side Implementation:

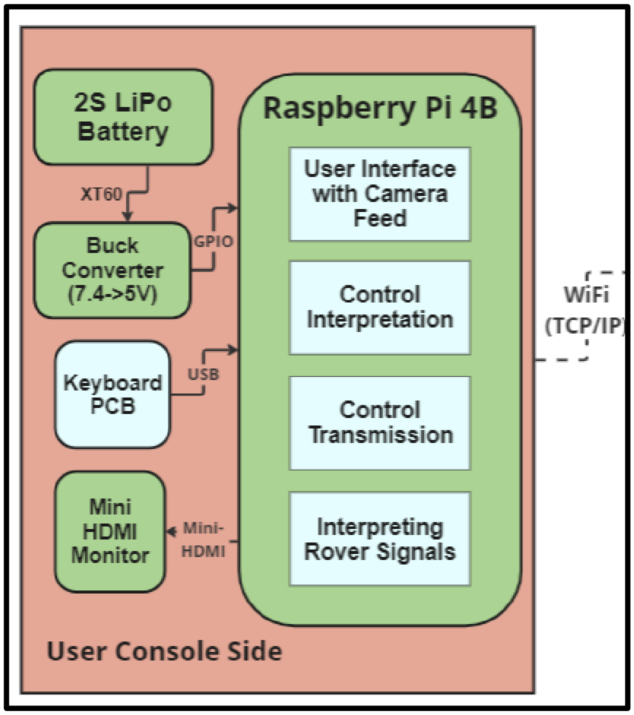


Figure 2. Cropped User-Console-Side Section of the Overarching Block Diagram.

The User-Console-Side Implementation will be powered by a 2S Lithium Polymer battery. The Raspberry Pi is best operated at 5 Volts, and the closest Lithium Polymer battery available is a 2S LiPo, which outputs a voltage of 7.4V. We will feed the output of the 2S Battery to a Buck Converter, which will step down the voltage safely and efficiently, such that the Raspberry Pi and the other components attached to it can be powered.

The input to the entire system, and the way that the user can interact with the system, is through our Controller. Aiming to achieve the most compact implementation possible, we have elected to design the PCB by ourselves, with an integrated Arduino Nano on the PCB translating the binary keypresses to USB, such that the Raspberry Pi can process the keystrokes efficiently. The Controller will be comprised of six buttons: Forward, Backward, Left, Right, and Give and Pickup. When the Forward and Backward keys are pressed, HomeRover will move forward and backward accordingly. When the left and right keys are pressed, HomeRover will turn in place. The Give button, as mentioned prior, will extend the robot arm such that the user will be able to retrieve the object from it.

A nontrivial quality-of-life aspect of the User-Console-Side is the mini monitor. The miniHDMI monitor will allow for the camera feed coming from the Rover to be displayed to the user, such that they can better navigate the Rover from farther distances, and to be able to figure out when to begin the pickup sequence.

B. Rover-Side Implementation:

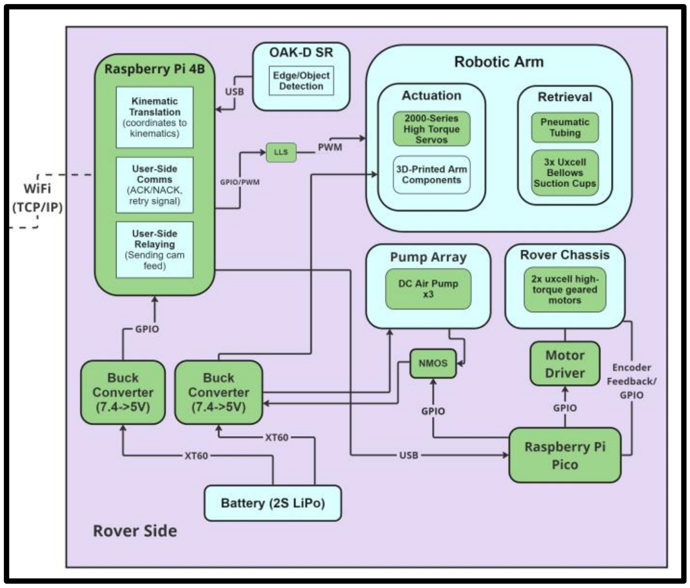


Figure 3. Cropped Rover-Side Section of the Overarching Block Diagram.

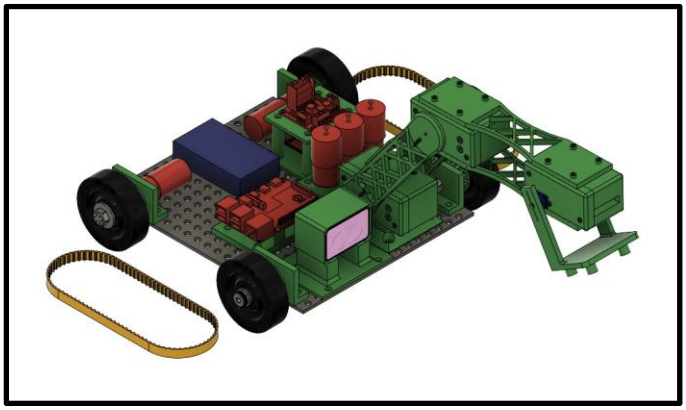


Figure 4. Isometric View of the Rover Assembly. Power elements are in Blue, 3D-Printed elements are in Green, and electronics are in Red.

The Rover is powered by a 2S Lithium Polymer battery, modeled by the blue rectangular prism near the back of the robot. The weight of the battery acts as a counterweight for the weight being picked up at the end of the robotic arm. The perforation/holes in the main chassis plate are to allow for more agile development; if components need to be repositioned, they can be, without needing to redesign and re-print objects.

As mentioned before, the overall design of the Rover can be split into two goals: Movement and Item Retrieval. The Raspberry Pi 4 Single Board Computer, pictured in the image above, acts as the onboard computer for the Rover, taking inputs from the User-Console-Side and operating the rover accordingly.

i.) Movement:

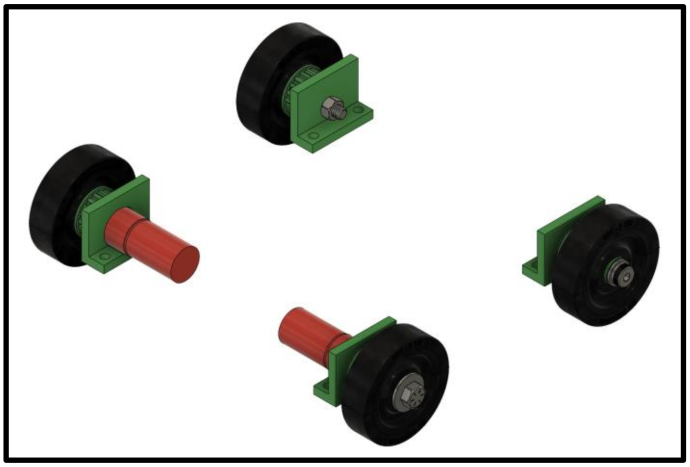


Figure 5. Isometric View of the Drive System. 3D-Printed elements are in Green, and electronics are in Red.

The Rover's drive train is designed in an effective manner to further push towards our goal. The front two motors sit on a dead shaft, where they spin directly on the bolt which holds them. This allows for the rover to drive from the back two wheels, and the front two wheels be driven by a timing belt pulley system. Regarding the arrangement of the wheels, the in-line nature allows for easier kinematic calculation.

The entire arrangement is powered by the aforementioned 2S Lithium Polymer battery. The motors can be run on 7.4V through a driver board, so we will not need a buck converter or any stepdown between the battery and the motors.

Accurate movement is quite important for the Rover. Rather than adding another Degree of Freedom to the Robot Arm, we elected to instead use the rotation of the Rover to rotate the arm, a fact that will be explained in a later section.

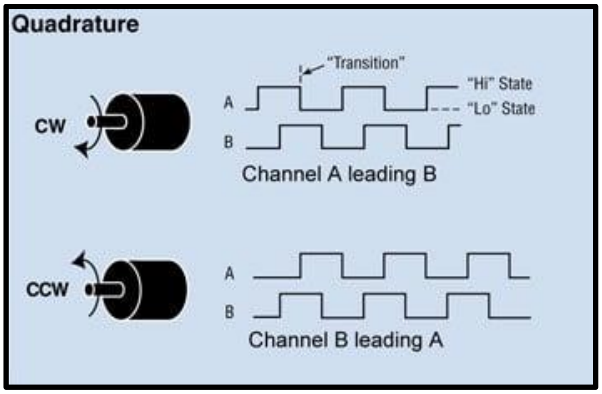


Figure 6. HomeRover's encoders use Quadrature Encoding.[20]

This necessitates encoders and a fast, high-refresh-rate control loop. HomeRover's motors use quadrature encoding, whose out-of-phase waves require that the pin to which they connect be able to be read at a high rate. Our original idea was to use the mainboard Raspberry Pi's GPIO to drive and control the motors; however, it having an operating system (Ubuntu 22.04), prevents this from occurring at a high enough rate. As such, in order to create the supporting electronics for the motor, we elected to use an embedded microcontroller, the Raspberry Pi Pico, along with an H-Bridge motor driver, the L298N (pictured above). This arrangement saves space on the chassis, and provides a centralized system where the motors can be driven.

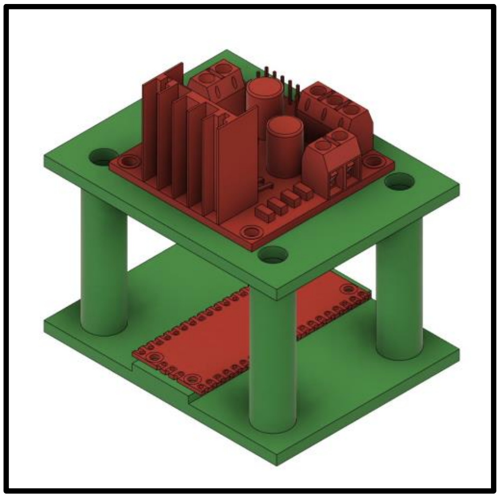


Figure 7. Isometric View of the Drive Electronics. 3D-Printed elements are in Green, and electronics are in Red.

ii.) Item Retrieval:

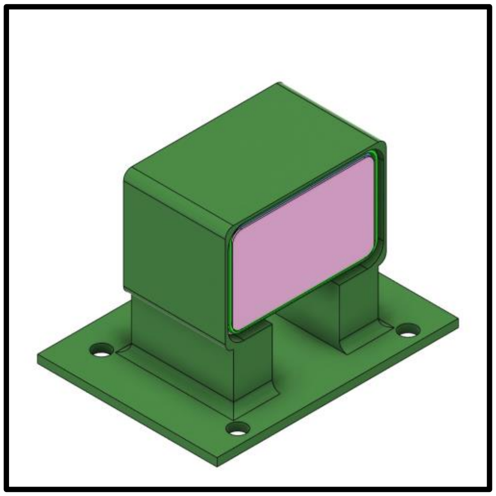
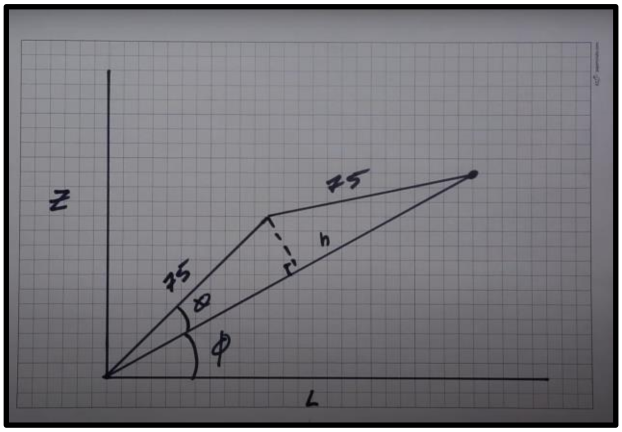


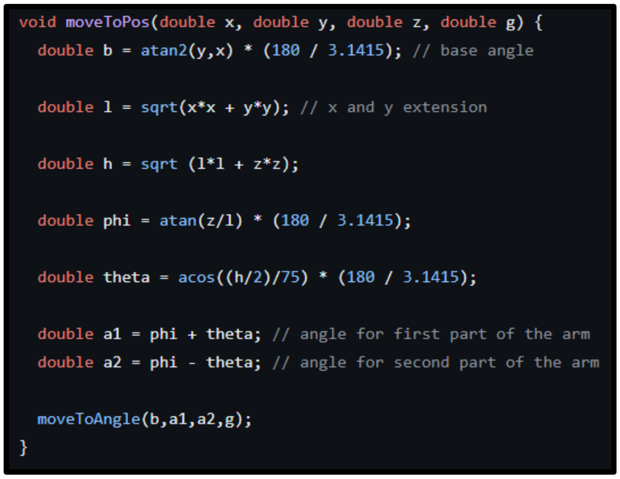
Figure 8. Isometric View of OAK-D SR Mount. 3D-Printed elements are in Green.

The first step in the process of retrieving the item is the process of detecting it, and determining where it is relative to the robot. For this, we chose the OAK-D SR camera, the mount for which is pictured above. The camera comes with the ability to do onboard ML processing, and we will be using a depth camera algorithm that it comes with to determine the distance (X, Y, and Z) from the camera. A sample output of the camera is shown below.

Figure 9. Sample Depth Camera output from OAK-D SR.

As is visible in the image above, the camera comes with the ability to run programs to identify objects; however, the scope of our project is limited to only pick up objects with one object being in the frame of the camera. Also visible is the X, Y and Z distances to the center of the object, whose numbers the Raspberry Pi 4 mainboard will process and generate the kinematic scheme for the robotic arm.





```
void moveToPos(double x, double y, double z, double g) {
  double b = atan2(y,x) * (180 / 3.1415); // base angle

  double l = sqrt(x*x + y*y); // x and y extension

  double h = sqrt (l*l + z*z);

  double phi = atan(z/l) * (180 / 3.1415);

  double theta = acos((h/2)/75) * (180 / 3.1415);

  double a1 = phi + theta; // angle for first part of the arm
  double a2 = phi - theta; // angle for second part of the arm

  moveToAngle(b,a1,a2,g);
}
```

Figures 10 and 11. Sample Diagram and Code for limited DOF kinematics.[21]

Because of the deliberately degree-of-freedom-limited nature of the robot arm (pictured below), the kinematics are rather simple, and can be calculated with the simple script shown above. A factor we will need to consider is the position of the robot arm on the rover; it not being at the center of rotation effectively adds another ‘leg’ to the arm, which may increase its complexity, but the script to do the calculation will be similar to that shown above.

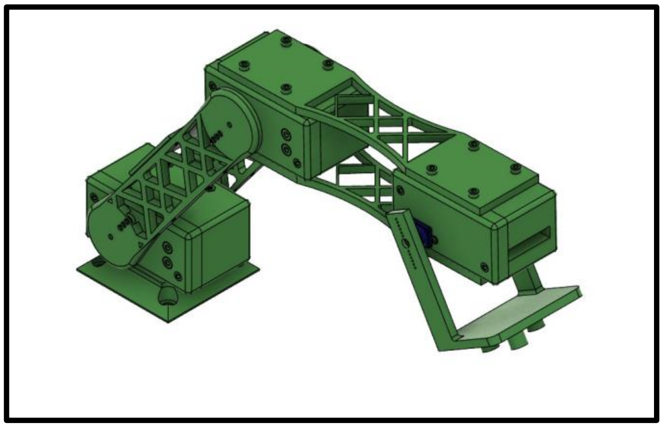


Figure 12. Robot Arm Assembly. 3D-Printed elements are in Green.

The robot arm assembly, pictured above, sits on the Rover in the configuration shown in a previous picture. Each leg of the arm is controlled by a servo, each with enough torque to carry its respective leg. The servos are controlled by Pulse-Width Modulation, a common technique seen in many different applications to use a single binary connection to send, essentially, analog signals.

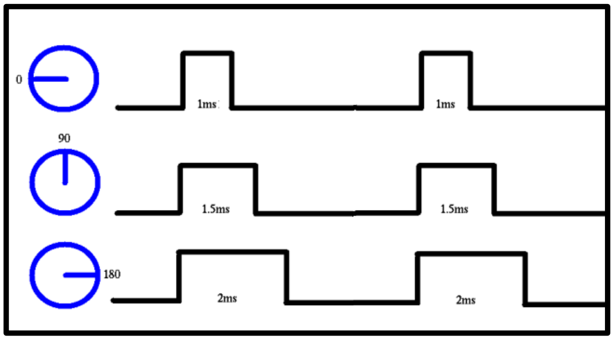


Figure 13. Diagram detailing Pulse-Width Modulation.[22]

With the ability to drive the servos to degree-accurate positions, we will be able to accurately move the end-effector to the position slightly above the object of interest.

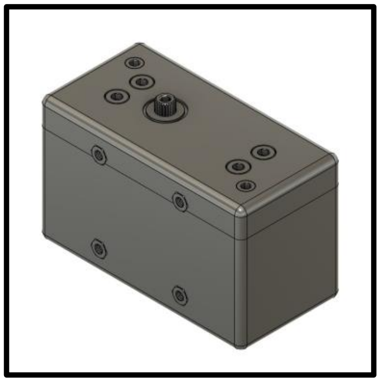


Figure 14. 3D Printed Arm Module.

Pictured above is one of the servo modules on the arm. Embedded within the 3D Printed shell is a 2000 Series Servo, whose stall torque (25.2 kg.cm) will be enough to hold our maximum weight up at the end of the arm. The range of our arm is 30 cm, and our max mass at the end of the arm is 0.7 kg, which leads to an eventual final torque of 21 kg.cm. As the calculated stall torque will be less than the servo's stall torque, the servos will be able to hold the arm at a constant orientation and navigate its end effector to the correct position.

Figure 15. Bellows Suction Cups, made of silicone.[23]

Pictured above are silicone suction cups present at the end effector of the arm, seen above in the CAD rendering. Their shape and material allows for their deformation around irregularly-shaped objects; if we decide to add irregularly-shaped objects to our scope, we will be able to accommodate them with these suction cups.

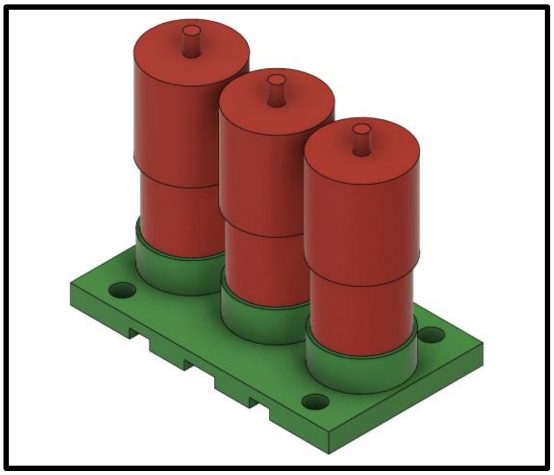


Figure 16. Pump Holder. 3D-Printed elements are in Green, and electronics are in Red.

To actually pick up objects, we elected to use DC motor pumps that will provide adequate pressure to pick up the requisite planar objects.

Control Scheme

To better describe the modus operandi of the rover, a state machine is shown below.

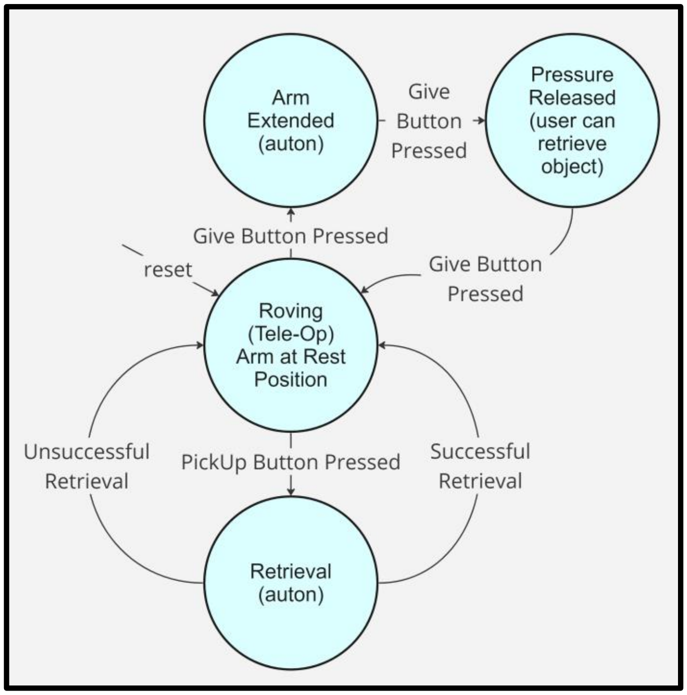


Figure 17. Simple State Machine Describing Rover Architecture

VII. TEST, VERIFICATION AND VALIDATION

A. *Tests for Accuracy*

To ensure the success of our full system in the sphere of accuracy, we need our separate subsystem components to execute with 80% accuracy, as described in the design requirements. The first sphere, object identification, requires utilizing edge/feature detection on the Oak D-SR camera. Once this pipeline is created, to achieve our desired accuracy, it must be tested and achieve greater than 95% success against a thorough and complete dataset of 50+ test pictures, which will consist of 7+ objects in various orientations and poses to mimic the variability of a real-world scenario. In addition, this dataset will have image augmentation performed on it to further increase the completeness of our testing dataset. The second and third spheres of full system accuracy are claw positioning and arm suction, which will be tested in tandem in integration tests. These tests will, following specific instructions and amounts representing camera output calculations, measure the offset and difference in distance between the observed target area and the theoretical, golden target area. This golden target area will be determined through calculations of kinematics to determine the correct landing location of the arm and end effector. Success will be measured by a maximum of 1 centimeter offset on any side of the real and theoretical overlap. To meet our other full-system accuracy requirements, such as the 10 second requirement for the system to position itself and pick up the item, the item pickup range of 10 to 30 centimeters, and the suction lifting capacity of 700 grams, it would require procedural tests where we incrementally measure a specific metric to make sure we can reach the respective threshold.

B. Tests for Accessibility

To measure the accessibility of the user-interface and user-console facing side of our design, we plan on conducting user studies and focus groups. In these tests, we plan on assessing and measuring, through a stopwatch, how long it takes for a user to get acquainted to our system with a maximum time of 10 minutes to get comfortable driving and navigating the controls of our user console.

C. Tests for Latency

Latency in our system is represented in multiple facets, those being transmission latency, control center latency, receiver to motor latency, and receiver to suction claw latency. Regarding the transmission latency, this mainly takes the form of communication between the two Raspberry Pi's. In this case, our testing method will take the form of recording the time of data transmission between the two using system timers and subtracting the time difference with a target of less than 100 milliseconds. For the control center latency, which takes the form of communication between the user-console output and Raspberry Pi retrieval, our testing plan is to record the time between pressing a button and observing its response in a terminal window on the Raspberry Pi side, with a goal of less than 20 milliseconds. To record the time, we plan on using slow motion iPhone camera video taken at 240 fps, where we will be able to clearly see the terminal response. Receiver to motor and receiver to suction claw latency are similar in that both originate through signals sent through Raspberry Pi, with the difference being which mechanical subcomponent experiences the stimulus. We plan on testing using slow motion iPhone camera video as well, with target latencies of less than 20 milliseconds.

D. Tests for Battery Life

Another sub-aspect of the user experience, battery life, will be tested using the discharge battery capacity testing method, where we charge and discharge the battery fully and time how long it takes to discharge. We will have the system turned on with periodic instructions to pick items up to model real-world current levels and real-world usage. Our testing target will be greater than 1 hour between recharges.

E. Tests for Versatility

The last main concept we highlighted in our design requirements was the ability of our system to navigate any household setting, whether it be carpet, hardwood, or tile. Success in navigation is defined as being able to reach and maintain our established 0.3 meters per second travel speed, measured through taking the time elapsed to travel between two points on any of the varied surfaces, and dividing by time to get meters per second.

VIII. PROJECT MANAGEMENT

A. Schedule

Figure 18 showcases our schedule and Gantt chart broken up in terms of individual responsibility. Without delineating between individual members, the main categories on our schedule involve design, specifically regarding the control booth, arm, and chassis, as well as fabrication of the respective subsystems. The other main spheres of our schedule involve designing and implementing the kinematics scheme for our rover as well as communication schemes between the user console and the rover. Additionally, the schedule includes tasks to tackle the software/vision part of the project, including everything from software setup, depth camera experimentation, and writing the pipeline. Towards the end of the semester, before the final presentation and final demo, we have budgeted a period of three weeks for slack time, meant for integration and full-system tuning and testing. We were conscientious of major breaks like spring break when making our schedule, and its impacts are reflected in the schedule. Lastly, major course milestones are highlighted so we can see how our progress fits into the bigger picture.

B. Team Member Responsibilities

Mimicking the design and delineation of our schedule, our division of responsibility is as follows.

Varun's main tasks will be to design both the robotic arm we will be using for our rover as well as the rover itself. This entails fleshing out all the required components, power calculations, mechanical calculations, and testing to make sure the rover is structurally and electrically sound. Additionally, he is primarily in charge of designing, building, and testing the suction mechanism. He will also play a major part in fabrication of the rover in making sure the interconnects and communication is sound.

Hayden's main tasks are similar in that he will play a major part in designing the control booth/user console as well as aspects of the rover, including all the requisite calculations and parts sourcing as stated above. Overall design and fabrication of the rover is mainly a two-person joint effort between Hayden and Varun. In addition, Hayden will primarily take ownership of the kinematics scheme in translating our camera output to rover movement.

Nathan's primary tasks revolve around the software side, particularly in the space of object recognition with the depth camera. This entails experimenting extensively with the depth camera, developing the pipeline utilizing its onboard tools, setting up the Raspberry PI and setting up the interfacing in that respect as well as enabling communication between the user side and rover side. Because it is intimately linked, I will also assist Hayden in fleshing out the kinematics.

No tasks are fundamentally disjoint and isolated from the others, and we will all likely play a part in every aspect of our system. For integration and testing, we are planning on all being involved and contributing equally.

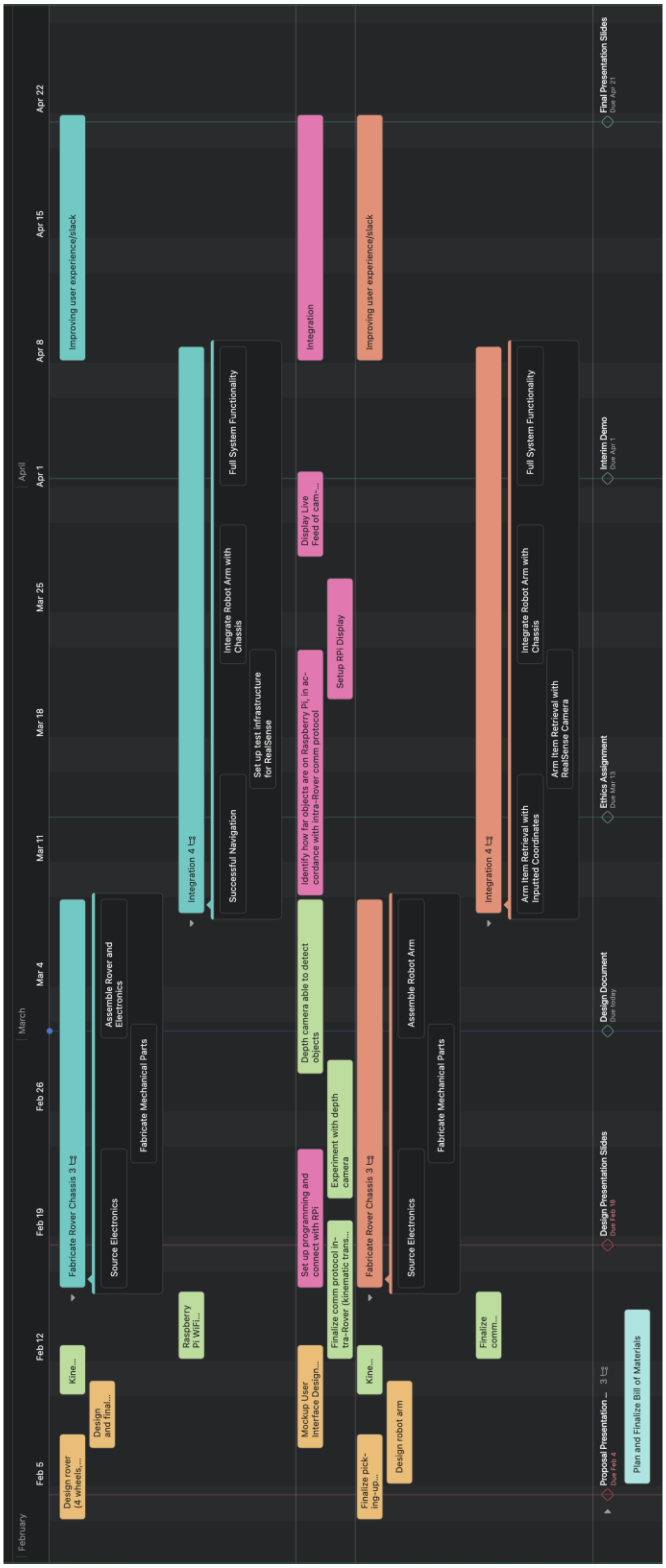


Figure 18. Schedule Gantt Chart with milestones and team responsibilities

C. Bill of Materials and Budget

Table III at the end of the report highlights our bill of materials and resulting budget. We aimed to take advantage of the ECE inventory as best we could, utilizing our own equipment and even finding communal equipment to bring costs down.

D. Risk Mitigation Plans

Regarding our most pressing risks, this will take the form of our vision system and the resulting communication to the robotic arm. The primary expertise on our team does not lie in the sphere of computer vision and object detection, so our primary risk is getting accurate depth data from the camera and successfully detecting objects. If we find that detecting objects through edge and feature detection is proving to be inaccurate and not meeting our desired metrics, a potential solution is to utilize the onboard capabilities of the camera to run machine learning algorithms and AI models to hopefully achieve a better result. If this were to happen, we would need a risk mitigation strategy that measures the new power draw of the camera from running a higher intensity workload and makes sure the battery is capable of successful function. Regarding the end effector, if we observe that our method of suction cannot fully lift the objects as stated in our design requirements, whether through pure pressure or because of irregularities, we can brainstorm and research new end-effectors and suction cups that could provide a better job adhering to irregular shaped objects. If we find that translating camera coordinates to kinematic instructions are resulting in inaccurate arm movement, we plan on adding ample integration time towards the end of the semester for calibration to work out the most accurate offsets from the camera location and coordinates to the arm's starting location.

IX. RELATED WORK

As mentioned in the introduction, there are several physical and robotics solutions to the problem described earlier. In terms of similarity to the project we are proposing to implement, the existing robotics solutions align closer than the physical solutions. The first solution of interest is TidyBot, the joint research project with Princeton and Google. The main aim of TidyBot is to leverage large language models to personalize physical assistance [4]. Personalization occurs through determining people's preferences in where to pick up objects and store them away [4]. Rather than provide mechanical assistance to the act of picking things up, as in HomeRover's mission, TidyBot aims to remove the user entirely, automating the entire process. Thus, the targeted user group varies significantly in that TidyBot is not operating with the underlying mission to assist those with ambulatory difficulties. With HomeRover's specialized mission, we can develop features that restore autonomy and empower independence in living alone.

Another solution is created by Kinova, and it is a "safe, lightweight robotic arm with a three-fingered hand" that can attach to a wheelchair [24]. The user group that is assisted through this device is "people with upper-body disabilities", and this arm enables them to "perform tasks like picking up objects and opening doors" [24]. Similar in concept to HomeRover, the ultimate use case appears to be more for a mobile application rather than from an immobile, sitting position. To increase the potential pickup radius of the arm, rather than put the arm on a rover like in HomeRover, the arm relies on user movement through its attachment to a wheelchair, thus achieving a similar result from a different approach. Major differences from our design lie in the specification of their arm, which has 9 degrees of freedom as well as custom software and a custom computing and control system. Similar to TidyBot, this project mainly exists in a research capacity, meaning it is extremely expensive to acquire, at $50,000 currently for the research version [24]. Thus, other solutions currently exist on the market with advanced proprietary technology, but in the near future, the prospect of having a device that can provide physical assistance that is cost effective and achieves a similar function is uncertain, thus providing the space that we hope HomeRover can fill.

X. SUMMARY

HomeRover is an all-encompassing solution to the challenges that those with limited mobility face. To prevent an exacerbation of injury from falls and muscle usage, our rover intends to assist the item-pickup and retrieval process in a manner that is easy to use and gives back autonomy to the user that they may have lost. Existing solutions, while offering personalization and automation of the entire process, are extremely expensive. Through our implementation, we maintain the increased pickup range of these solutions as well as keeping aspects of autonomy through our semi-autonomous system. Our primary user focuses that informed our design requirements revolve around the concepts of latency, accuracy, cost, versatility, and the user experience. With our schedule and division of labor, we believe we have a reasonable plan of action to achieve this design and make a significant positive impact on the lives of others.

GLOSSARY OF ACRONYMS

CAD – Computer-Aided Design
DOF – Degree of Freedom
FPS – Frames per second
GPIO – General Purpose In/Out
PCB – Printed Circuit Board
SoC – System on Chip

REFERENCES

[1] Martinez, C., “Disability Statistics in the US: Looking Beyond Figures for an Accessible and Inclusive Society,” *Inclusive City Maker*, Apr. 08, 2022. https://www.inclusivecitymaker.com/disability-statistics-in-the-us/#:~:text=What%20Accessibility%20Solutions%20for%20Different,around%2C%20walking%20or%20climbing%20stairs.

[2] Leppert, R., Schaeffer, K., “8 facts about Americans with disabilities,” *Pew Research Center*, Jul. 24, 2023. https://www.pewresearch.org/short-reads/2023/07/24/8-facts-about-americans-with-disabilities/

[3] Centers for Disease Control and Prevention, “Facts About Falls,” May 12, 2023. *https://www.cdc.gov/falls/facts.html*.

[4] Schwarz, J., “Engineers clean up with TidyBot,” *Princeton University*, Jul. 24, 2023. *https://engineering.princeton.edu/news/2023/07/24/engineers-clean-tidybot*

[5] homestratosphere, “What is the Average Size of Living Rooms in the USA?,” Sep. 09, 2021. https://www.homestratosphere.com/average-size-living-room/

[6] J. Wu et al., “TidyBot: Personalized Robot Assistance with Large Language Models,” arXiv (Cornell University), May 2023, doi: https://doi.org/10.1007/s10514-023-10139-z.

[7] “Mechanical Keyboard Size Guide - Which size of keyboard should I get?,” *epomaker*, Feb. 01, 2024. https://epomaker.com/blogs/guides/mechanical-keyboard-size-guide (accessed Mar. 01, 2024).

[8] J. Nielsen, “Response Time Limits: Article by Jakob Nielsen,” *Nielsen Norman Group*, 2019. https://www.nngroup.com/articles/response-times-3-important-limits/

[9] P. Strauss, “This Guy Built the World’s Fastest Roomba,” *The Awesomer*, Oct. 24, 2022. https://theawesomer.com/worlds-fastest-roomba-2022-edition/685968/ (accessed Mar. 01, 2024).

[10] “Roomba Combo$^{\text{TM}}$ i5+: The All-in-One Robot Vacuum Cleaner | iRobot,” *www.irobot.com*. https://www.irobot.com/en_US/roomba-i5plus-self-emptying-robot-vacuum/I555020.html (accessed Mar. 01, 2024).

[11] Apple, “iPad Pro,” *Apple*, 2011. https://www.apple.com/ipad-pro/specs/

[12] Luxonis, “RealSense comparison,” *Luxonis*. https://docs.luxonis.com/projects/hardware/en/latest/pages/guides/realsense_comparison/

[13] Luxonis, “OAK-D SR,” *Luxonis*. https://docs.luxonis.com/projects/hardware/en/latest/pages/OAK-D-SR/

[14] Luxonis, “Depth perception,” *Luxonis*. https://docs.luxonis.com/en/latest/pages/depth/

[15] LSLIDAR, “2D LiDAR,” *Leishen Intelligent System Co., Ltd.* https://www.lslidar.com/products/2d-lidar/

[16] Jasuja, N., “Bluetooth vs. Wi-Fi,” *Diffen*, https://www.diffen.com/difference/Bluetooth_vs_Wifi

[17] Dynapar, “Quadrature Encoders - The Ultimate Guide,” *www.dynapar.com*. https://www.dynapar.com/technology/encoder_basics/quadrature_encoder/

[18] “Buy Cheap 21.3:1 Gearmotor with Encoder DC 12V 201RPM Encoder Gear Motor 25Dx49L mm,” *www.uxcell.com*. https://www.uxcell.com/2131-gearmotor-with-encoder-12v-201rpm-encoder-gear-motor-25dx49l-p-1426432.html (accessed Mar. 01, 2024).

[19] “Benchmarking Raspberry Pi GPIO Speed,” *codeandlife.com*. https://codeandlife.com/2012/07/03/benchmarking-raspberry-pi-gpio-speed/ (accessed Mar. 01, 2024).

[20] “Quadrature Encoders - the Ultimate Guide.” Quadrature Encoders - The Ultimate Guide, Dynapar, 9 Mar. 2018, www.dynapar.com/technology/encoder_basics/quadrature_encoder/.

[21] “InverseKinematicArm.” GitHub, github.com/roTechnic/InverseKinematicArm/tree/main.

[22] “Uxcell Bellows Suction Cup,15mm Diameter 16mm Length M5 Joint ...” *Amazon*, www.amazon.com/uxcell-Diameter-Silicone-Industrial-Pneumatic/dp/B07MHGDFP4. Accessed 1 Mar. 2024.

[23] distortiondistortion 6511 silver badge1111 bronze badges, and Tony Stewart EE75Tony Stewart EE75 1. “Driving Servo Motor with PWM Signal.” Electrical Engineering Stack Exchange, 1 Nov. 2018, electronics.stackexchange.com/questions/346603/driving-servo-motor-with-pwm-signal.

[24] robotsguide.com, “Jaco”, *IEEE*, *https://robotsguide.com/robots/jaco*

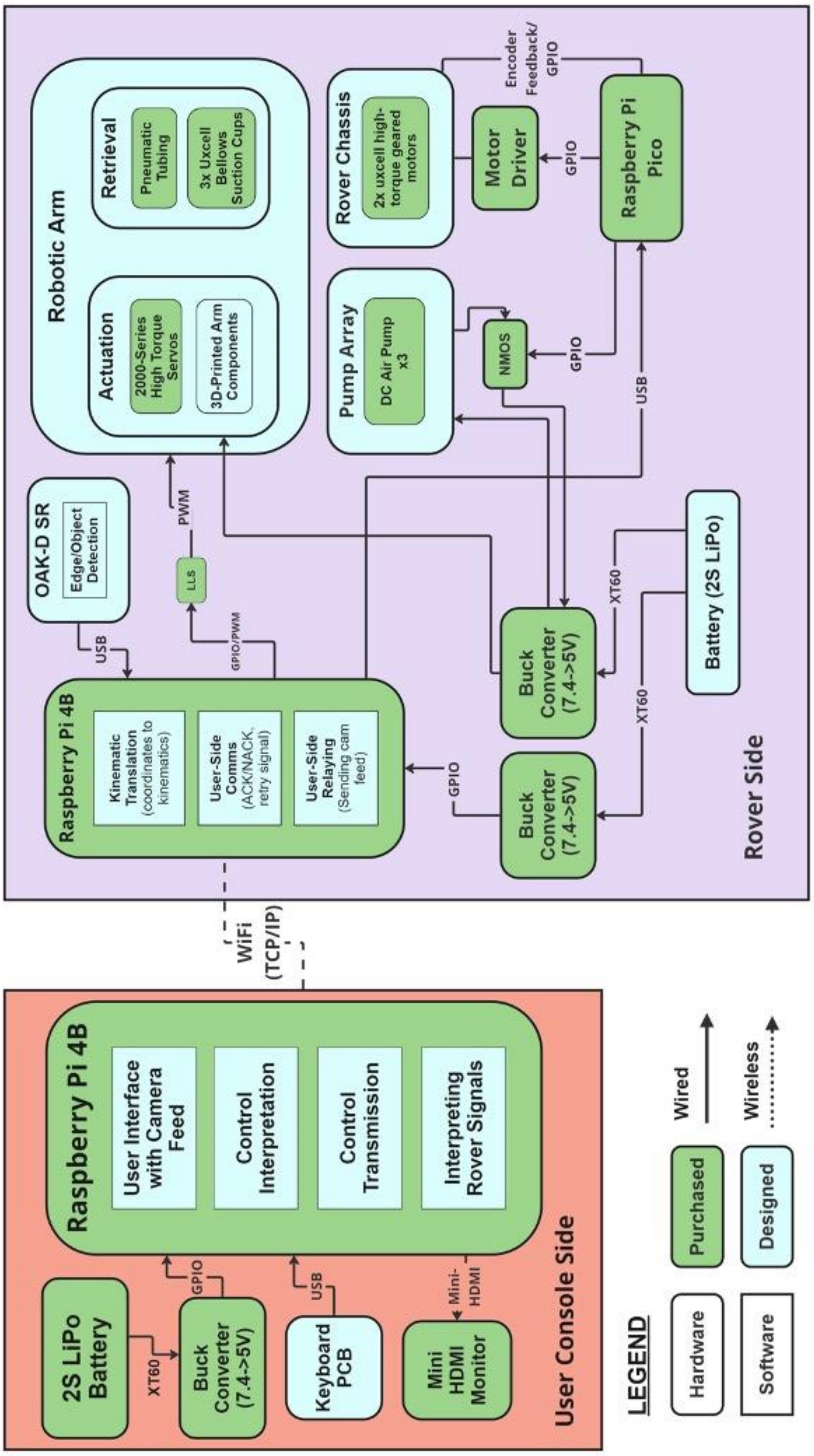

Rover Side
Robotic Arm
Actuation
2000-Series High Torque Servos
3D-Printed Arm Components
Retrieval
Pneumatic Tubing
3x Uxcell Bellows Suction Cups
OAK-D SR
Edge/Object Detection
USB
PWM
LLS
GPIO/PWM
Raspberry Pi 4B
Kinematic Translation (coordinates to kinematics)
User-Side Comms (ACK/NACK, retry signal)
User-Side Relaying (Sending cam feed)
Pump Array
DC Air Pump x3
NMOS
GPIO
Rover Chassis
2x uxcell high-torque geared motors
Motor Driver
GPIO
Encoder Feedback/ GPIO
Raspberry Pi Pico
USB
Buck Converter (7.4->5V)
Buck Converter (7.4->5V)
GPIO
XT60
XT60
Battery (2S LiPo)
WiFi (TCP/IP)
User Console Side
2S LiPo Battery
XT60
Buck Converter (7.4->5V)
GPIO
Keyboard PCB
USB
Mini HDMI Monitor
Mini-HDMI
Raspberry Pi 4B
User Interface with Camera Feed
Control Interpretation
Control Transmission
Interpreting Rover Signals
LEGEND
Hardware
Software
Purchased
Designed
Wired
Wireless

Table III: Bill Of Materials

Description	Model #	Manufacturer	Quantity	Cost	Total
Drive Train Motor	a17092900ux0541	uxcell	2	19.99	39.98
Arm Servos	B0CDWPL9QZ	FXDLSERVO	2	24.99	49.98
Arm Bearings	a19121700ux0034	uxcell	10	8.99	8.99
Suction Tubing	a13080200ux0301	uxcell	1	8.99	8.99
Suction connectors	a18092200ux0610	uxcell	10	6.99	6.99
3D printer Filament	P-PETG-Black	YOOPAI	1	12.99	12.99
Suction cups	a18110600ux0302	uxcell	4	8.49	8.49
Air Pumps	X2019050306	Hxchen	3	15.99	15.99
Arm Bearings	608ZZ	Sackorange	20	8.79	8.79
Servo Horn	CS981	ShareGoo	2	8.69	8.69
Depth Camera	Oak-D Short Range	Luxonis	1	199	199
Metal mounting plate	Aluminum plastic composite	RoboClub	1	0	0
Single Board Computer	RAS-4-4G	Raspberry Pi	2	55	110
Wheels	T81P-296BB	BaneBot	4	3.5	14
USB to GPIO	Pi Pico	Raspberry Pi	1	5	5
Timing Belt	6484k228	McMaster-Carr	2	10.85	21.7
Thread Insert for Live shaft	90742a133	McMaster-Carr	10	8.06	8.06
Shoulder Screw for Deadshaft	91259a630	McMaster-Carr	2	2.57	5.14
Washer for Deadshaft	92141a228	McMaster-Carr	50	9.88	9.88
Washer for live shaft	94051a230	McMaster-Carr	2	3.57	7.14
Screw for live shaft	92620a621	McMaster-Carr	50	16.15	16.15
Nut for Dead Shaft	94575a230	McMaster-Carr	25	13.47	13.47
Mounting bolts	800600	Everbilt	100	22.32	22.32
Mounting nuts	802582	Everbilt	100	8.98	8.98
Batteries	B0BRKG5FBH	HOOVO	2	49.99	49.99
PCB		JLPCB	1	20	20
				Total	680.71