ELP 305: Design and Systems Laboratory **Semester II 2020-2021**

Laboratory Report, Group H: Mule Bot

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*

Members of the group H			
Sl	**Position**	**Entry Number**	**Name**
1	**Lead Coordinator**	2018EE10593	Arunava Das
2	Vision Coordinator	2018MT60787	Prateek Singh
3	Vision Sub-Coordinator	2018EE30610	Harsh Wardhan
4	Electrical Coordinator	2018EE10603	Dedeepyo Ray
5	Electrical Sub-Coordinator	2018MT60795	Snehil Grandhi
6	Design Coordinator	2018MT60791	Rishu Raj
7	Documentation Coordinator	2018MT10770	Subhalingam D
8	Member	2018EE10459	Chathur Gudesa
9	Member	2018EE30569	Yerukula Sravanasai
10	Member	2018MT10756	Ishant Bhaskar
11	Member	2018MT60798	Zuhaib ul Zamann
12	Member	2018EE30538	Dharmendra Seervi
13	Member	2018EE30543	Himanshu Gaud
14	Member	2018MT60793	Satendra Singhparashar
15	Member	2018EE10494	Rohit Agarwal
16	Member	2018MT10737	Akshat Rao
17	Member	2018MT60790	Ravi Pushkar
18	Member	2018MT60788	Ramneek Singhgambhir
19	Member	2018MT60776	Aakash Garg
20	Member	2018EE30534	Darpan Kumaryadav
21	Member	2018EE10499	Shashank Goyal
22	Member	2018MT60786	Pranaav
23	Member	2018MT60779	Bhupender Dhaka
24	Member	2018EE10491	Ranajay Medya
25	Member	2018EE10189	Aditya Bansal
26	Member	2018MT60777	Adwaith H Sivam
27	Member	2018MT10759	M Santosh
28	Member	2018EE10483	Penumudinagavenkata Saiabhinay
29	Member	2018MT10740	Anirudha Akela
30	Member	2018EE30598	Bharat Runwal
31	Member	2018EE30558	Reddy Cihir
32	Member	2018MT60778	Ashwini Kumar
33	Member	2018MT10745	Aryan Gupta
34	Member	2018MT60244	Shaurya Goyal
35	Member	2018MT10763	Punit Shyamsukha
36	Member	2018MT10760	Mukul Kumar
37	Member	2018MT60797	Vishal Meena
38	Member	2018EE10514	Vikas Kumar
39	Member	2018MT60199	Anshul Tak
40	Member	2018EE10456	Biruduraju Harahima dhruthi

Contents

1 Requirements

1.1 Electrical Requirements

The device has the following electrical requirements:

- A rechargeable battery that can take a 220V-50Hz AC supply input and provide enough output that could last full day or at least 2-3 shopping sessions of (about 135 minutes each)
- An alarm that can let the user know if the bot get overloaded or get stuck somewhere
- A vehicle speed sensor (VSS)
- A weight measuring device which would tell us the weight of the load and warn us if it is beyond the capacity
- A Gyroscope and an Accelerometer
- An inbuilt software system for automatically driving and steering mechanism
- Energy efficiency
- Fast reaction speed of all subsystems
- Proper driving and steering mechanisms

1.2 Mechanical Requirements

The device needs to have the following:

- A suitable shape such that it should not get stuck but should also such that maximum surface area of platform can be utilised
- An aesthetic look
- Dimensions such that it does not topple easily. Height must be enough such that user does not need to bend down, but center of mass must be low to avoid toppling
- A suitable base area taking toppling and item holding capacity into account
- A strong surface that can provide sufficient friction
- A railing or lateral boundary so that item do not
- A shock absorber so that bot does not take damage from impact
- 40 kg capacity with some overhead
- A strong and light encasing material, such that it can handle the collisions that may occur
- An on - off button for Automatic mode
- A provision that allows user to control the bot manually if need be
- A removable container so that the user can carry item themselves if need be and it is easy to transfer items between bots
- A 4 wheeled system to provide stability and making the device omni-directional (e.g., using mecanum wheels)
- Minimum possible weight - to save energy and capacity
- Waterproofing of sensitive areas to prevent damage from leaks (e.g., the juice box may leak)
- Organisation of elements such that its easy to clean, repair and there shouldn't be any heating issues
- Elements such that it does not make noise
- The ability to overcome small obstacles and inclines (approx order 1cm, and 15 degrees) [*climbing staircases not required*]
- Speed should be greater than the human walking speed (i.e., greater than 1.46m/s) so that it can overcome some obstacles and still be able to catch the user

1.3 Sensing/Vision Requirements

After researching about previous work done in this area [1], we have the following as the requirements related to the vision or sensing of the robot:

- The robot has to detect the target person accurately by camera or other sensors
- Needs to stay between 0.5 and 2 m of the user
- Bot should remain connected to the user
- Should be able to differentiate between different people and the background
- Able to reject signal noise
- Wide field of view
- The overall robot system must have a high reaction-speed/frame-rate
- Should work for a wide range of environment conditions (i.e., different lightening conditions - bright/dark)
- Should be able to detect distance and direction relative to it
- Should be able to find its way to the user (i.e., should not lose the user at turns)
- The robot must be able to do obstacle avoidance
- What if it loses its calibration? An automatic or easy procedure to recalibrate/connect

1.4 Miscellaneous Requirements

- If the bot is owned by the mall then an economic model is needed that makes sense for the user and the mall
- A charging solution will also be needed in this case, that is convenient to the mall and ensures that all the shoppers can get a bot with full battery
- An aesthetic look that matches the mall environment
- Should start and stop slowly so that item don't fall due to inertia
- Safety standards - speed, obstacle detection, procedure to follow in case of accident, etc.
- The time required for developing and marketing the device needs to be appropriate and not too long in order to compete with other such offerings in the market.

2 Specifications

2.1 Electrical Specifications

2.1.1 Battery

1 kwh,18650 Li-ion 2600mAh 4s52p Battery. [2, 3]

Figure 1: 18650 Li ion cell

Nickel stripes are used to put together the lithium cells using spot welding.

Battery type	Li-ion 18650
Number of Battery cell	208(52×4)
Nominal capacity of Battery cell	2600mAh-0.5C
Nominal output voltage	14.8 V
Voltage	3.7 V
Dimensions of cell	18.4(D) * 65.2(H) mm
Weight of single cell	46.5g

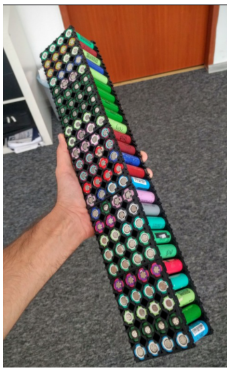


Figure 2: Sample Battery Arrangement

2.1.2 Current Limiter

MAX17613C 4.5V to 60V, 3A Current-Limiter.[4]

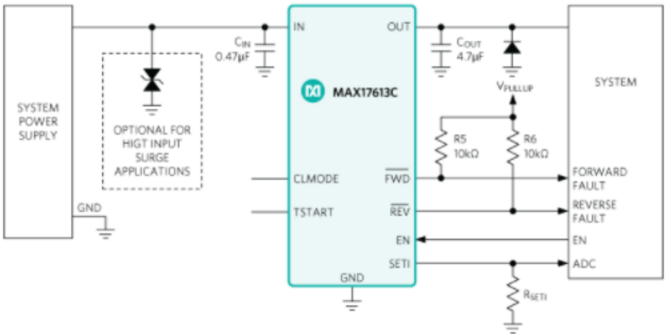


Figure 3: Current Limiter

V(Min)	4.5 V
V(Max)	60 V
Current Limit(Min)	0.15 A
Current Limit(Max)	3 A

2.1.3 Battery Management System

4S 40A 14.8V 16.8V BMS for safe charging of battery.

Model	BM085-4S40A-Li-ion
Charging Voltage range	16.8-18.1 V
Continuous Charge current(upper limit)	20 A
Dimensions	3*1.7*1.4 cm
Weight	9.07 g

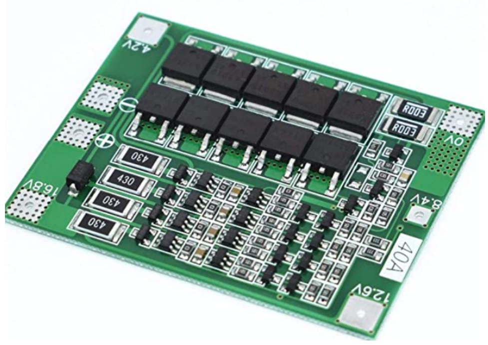


Figure 4: Battery Management System[5]

2.1.4 Step Down Power Charger

DC to DC 6-24V to 5V USB Output Step Down Power Charger with Adjustable Buck Converter [6].

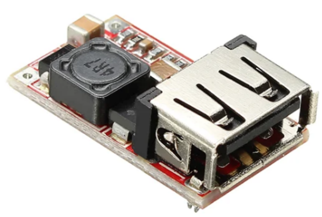


Figure 5: Step Down power Charger

Input Voltage	6-24 V
Output Voltage	5V
Maximum Output Current	3 A
Conversion efficiency	up to 97.5 %
Output ripple	around 10 mv at 20M BW
Dimensions	26.4 x 15 x 7.4
Weight	9 g

2.1.5 Speed Sensor

LM393 Speed Sensor Module

This speed sensor is a type of tachometer that is used to measure the speed of rotating devices (e.g motors, wheels). This is an infrared light based sensor which is integrated with a Voltage Cooperator IC. This consists of Infrared LED with a NPN Photo Transistor which helps in filtering out infrared light from external sources in the surrounding.[7]

Specifications of the sensor are as follows [8]:

- Dimensions: 32 x 14 x 7 mm
- The sensor reading slot has a width of 5 mm.
- Working voltage : 3.3 - 5 V
- Two outputs, one Digital and one Analog.
- LED power indicator.
- LED indicator of the output pulses of pin D0.

Figure 6: LM393 Speed Sensor Module

2.1.6 Driving and Steering Mechanism

Our bot is a non-holonomic drive which means that the robot must either rotate to the desired orientation before moving forward or rotate as it moves to change it's direction of orientation. Rotation of the bot can be done by rotating one side wheel in opposite direction to the other side ones. So, for rotation and movement in straight line or with some angle, we just have to change the rotational speed of wheel according to the requirement.[9]

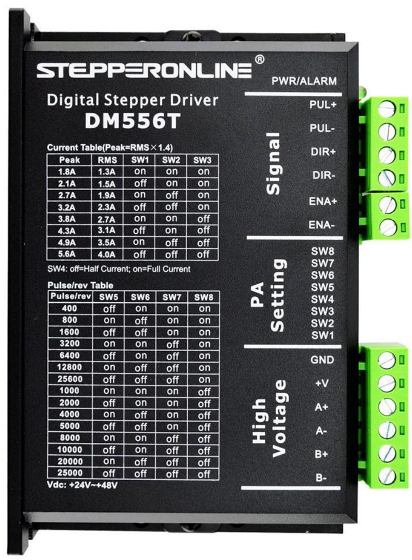


Figure 7: DM556T Stepper Drive

This is done using a **DM556T** which is a digital stepper driver and is able to control 2 and 4 phase stepper motors smoothly with low motor heating and noise. [10]

- Dimensions : 16 x 11.68 x 6.6 cm
- Peak Output Current : 1.8 - 5.6 A
- DC Supply Voltage : 20 - 50 V
- Logic signal current : 7 - 16 mA
- Pulse input frequency : 0 - 200 kHz

- Minimal pulse width : 2.5 μS
- Minimal direction setup : 5.0 μS
- Isolation resistance : 500 MΩ
- Weight : 300 gm

2.1.7 Gyroscopic Sensor

MPU6050 Gyroscopic Sensor Module

This rotational sensor combines 3-axis accelerometer, 3-axis gyroscope with Micro Electro Mechanical Sensing Technology and Digital Motion Processor in a single package. It communicates with microcontrollers via I2C bus controller.[11]

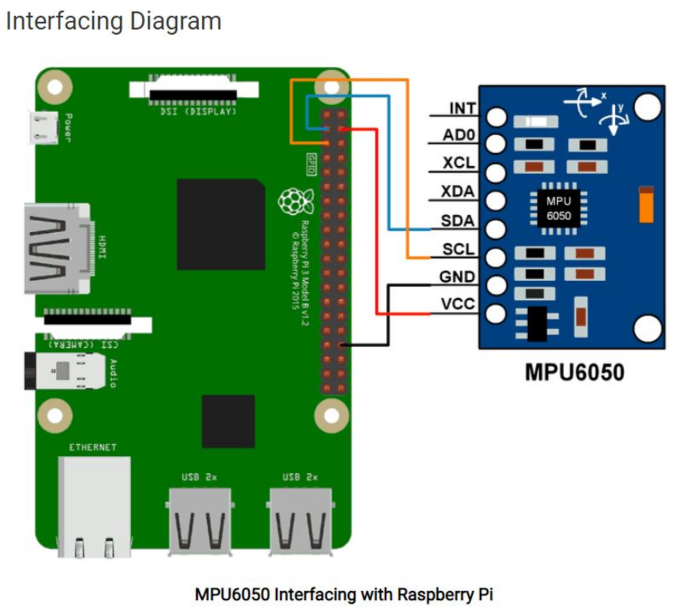


Figure 8:

Specifications of GY-521 MPU6050 [12, 13] :

- Dimensions : 21.2 x 16.4 x 3.3 mm [14]
- Power supply : 4.3 - 9 V
- On-board low Dropout Regulator KB33
- In-built 16-bit Analog to Digital Converter, 16-bit data output
- Weight : 2.1g
- Gyroscope range : +/-250, +/- 500, +/-1000, +/-2000 degrees/s

2.1.8 Weight Measuring Sensor

HX711 Weight Measuring Sensor

A transducer converts force to electrical output which is then amplified and converted to digital bits. [15]

- HX711 chip : 24-bit A/D Converter
- Load Cell : Transducer
- 16 x 2 LCD : LM016L

Load Cell Specifications [16]:

- Model : YZC-97
- Excitation Voltage : 5 - 12 V DC
- Rated Load : 40 kg
- Dimensions : 97 x 24 x 21.5 mm

HX711 ratings [17, 18] :

- Supply : Regulated 2.7 - 5.5 V
- Dimensions : 9.9 x 3.69 x 1.4 mm

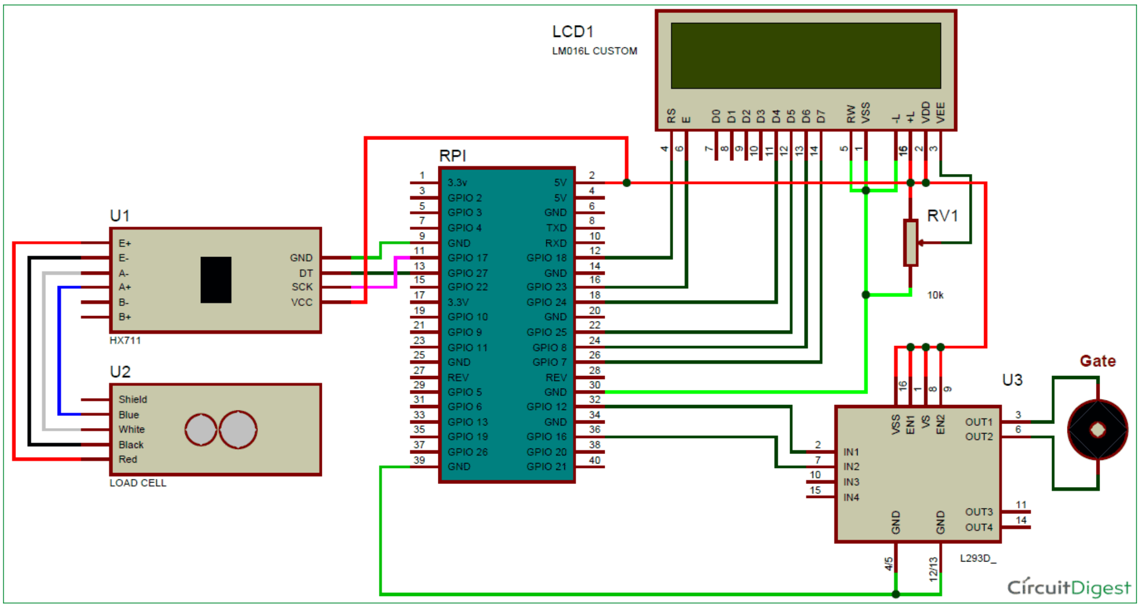


Figure 9: HX711 and YZC-97 interfacing with Raspberry Pi

2.1.9 Raspberry Pi

All the above components are connected to the Raspberry Pi via connecting wires to the appropriate pins. The micro-controller decides on the digital input from each of these components and acts accordingly. [19]

Digital Input/Output Pins are added to the Raspberry Pi hardware using MCP23017 which requires a supply voltage of 1.8 - 5.5 V. [20] [21]

2.1.10 Alarm System

This is designed to notify the user of his goods' weight exceeding the maximum allowed weight of the trolley. For this a Bluetooth-enabled mobile phone is required. This involves connecting the Raspberry Pi to the mobile phone via Bluetooth. [22]

When the weight is exceeded, the Raspberry Pi sends a notification (in .json format) to the mobile phone of the user and she/he can act accordingly.

2.2 Mechanical Specifications

2.2.1 Shape

Cuboid shape with rounded edges

2.2.2 Surface material

RS PRO Black Rubber Sheet [23]

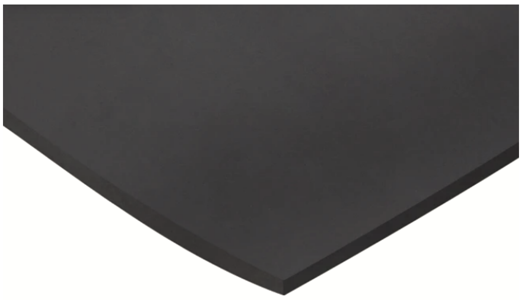


Figure 10: RS PRO Black Rubber Sheet

This acts as an absorber of sudden collisions, and avoids damage to the bot. This is attached to the edges of the bot. The Specifications are

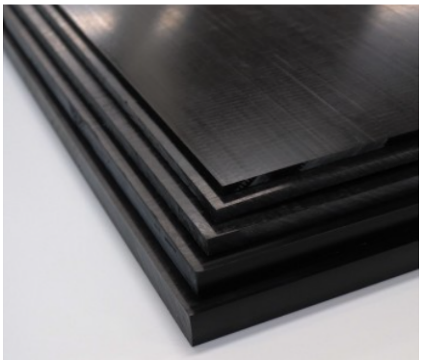
1. Colour : Black
2. Width : 1.5mm
3. Density : 1.5g/cm^3
4. Temperature range : -20 deg C - 70 deg C

2.2.3 Shock absorber

65mm Metal Front/Rear Shock Absorber for RC Car (Pack of 4) at 4 corners under the base [24]

Specifications are

1. Material: Metal and plastic
2. Length (mm): 65
3. Width (mm): 14 (Body Diameter)
4. Weight (gm): 8 (each)
5. Shipment Weight: 0.3 kg
6. Shipment Dimensions: 8 × 10 × 2 cm
7. Suspension: 1500LBs in rear shock
8. Spring diameter: 8mm



2.2.4 Material for the body of the bot

Acetal (high strength low friction plastic) [25]

Material: Homopolymer Acetal.

The sheet has the following specifications:

1. Dimensions: 48 in x 120 in
2. Thickness:0.4 in
3. Material Properties - [26]

2.2.5 Alternate Motor Driver/Adapter for motor

L298N DC Stepper Motor Driving Module 5V 2A [27]

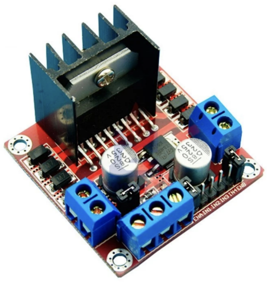


Figure 11: L298N DC Stepper Motor

1. Double H bridge unit
2. Chip: L298N (new ST)
3. Logic voltage: 5 V
4. Unit voltage: 5 V-35 V
5. Logic: 0mA-36mA
6. Transmission current: 2 A (Maximum single bridge)
7. Maximum power: 25 W Size: 43x43x26mm

2.2.6 Motors to run wheels

NEMA34 87 kg-cm Hybrid Stepper Motor [28]

Figure 12: NEMA34

1. Step Angle Step Angle: 1.8 °
2. Current: 5 A/Phase
3. Holding Torque: 870 N-cm
4. Lead Wires: 4
5. Shaft diameter: 14 mm

2.2.7 Wheels

130mm Wheel Set with Coupler [29]

Figure 13: 130mm Wheel

1. Diameter: 130mm
2. Width: 60mm
3. Interior six-angle side to side for 12mm (for Hex Coupler)
4. Weight: 158G

2.2.8 Coupler

6mm to 14mm - 6mm Hex coupling for Robot Smart Car Wheel 18mm Length [30]

Figure 14: Coupler

1. Type: Hexagonal coupling.
2. Material: brass.
3. Inner diameter: 6 mm.
4. Total length: 18 mm.
5. Size of Hex Socket(Long Diagonal): 14mm
6. Weight: 35gm

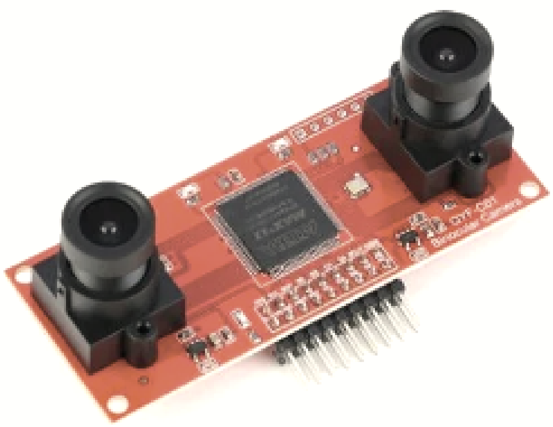
2.3 Sensing/Vision Specifications

Approach: The mule bot detects the position with the help of a UWB(ultra-wideband) sensors based indoor positioning system. The mule bot is fitted with 3 UWB sensors in a 2-d triangular formation. The customer will be given a bracelet which interacts with the sensors to determine the relative location of the customer from the mule bot. The bracelet will emit a short-duration signal, that is received by all 3 UWB sensors.

Due to difference in distance from the bracelet, the signal is received by all in different times. We then utilize this time difference to understand the direction and distance of the person from the bot. The mule bot then processes this information and moves towards the customer while avoiding objects that it detects with the help of ultrasonic sensors fitted on it. Additionally, a stereo camera is attached to the bot so that computer vision techniques can further help detect obstacles and find a suitable path towards the customer.

After researching about previous work done in this area [1], we have the following as the requirements related to the vision or sensing of the robot:

1. Stereo-vision camera mounted At the highest point
2. Python (Library : OpenCV) Reason : Most widely used, documentation available and overall ease of use. Lot of in-built ML/CV libraries
3. Frame rate for the camera :- 20FPS
4. **Camera :- OV2640 Binocular Camera Module CMOS STM32 Driver 3.3v 1600*1200 for 3D Measurement with SCCB Interface**[31]



The camera module containing OV5647 is connected directly to the CSI connector of the Raspberry Pi module (instead of the GPIO pins).

Specifications:

(a) Pixel : 1600 x 1200

   (b) Voltage : 3.3V
   (c) Lens aperture : F2.0
   (d) Lens angle of view : 78°
   (e) Current Consmption : 40 mA
   (f) Lens focal length : 3.6mm
   (g) Data format : 8 bit
   (h) Shipment Dimensions : 10 × 6 × 2 cm

5. **UWB Sensor :- DecaWave's DW1000 Single Chip**[32]

Figure 15: DW100 Chip

   (a) Supports 110kbit/s, 850kbit/s and 6.8Mbit/s data rates
   (b) 6 frequency bands supported with center frequencies from 3.5GHz to 6.5GHz
   (c) Transmit Power -14dBm or -10dBm
   (d) Preamble Length 64s to 4ms
   (e) Modulation: BPSK with BPM
   (f) Standard SPI interface to host (18Mb/s max)
   (g) Single Supply Voltage 2.8V to 3.6V
   (h) Receive mode from 64mA
   (i) 2A watchdog timer mode
   (j) 100nA deep sleep mode
   (k) Supports both Two Way Ranging and One Way Ranging, using Time of Flight (TOF) and Time Difference of Arrival (TDOA)methods
   (l) Industrial Temperature Rang e - 40 C to +85 C
   (m) 6 x 6mm 48 pin QFN package

6. Using OpenCV, we combine the feed from two adjacent cameras to understand the depth of different objects in front of the robot. CNNs are used to help us detect obstacles that the mule bot may face.

7. Ultrasonic sensors (we use ultrasonic sensors over Infrared because they are almost the same price but more reliable)[33]

8. 4 sensors : 3 in front (one facing front and other 2 at an tilt looking towards left and right), 1 sensor on the back (if we need to reverse at some point)

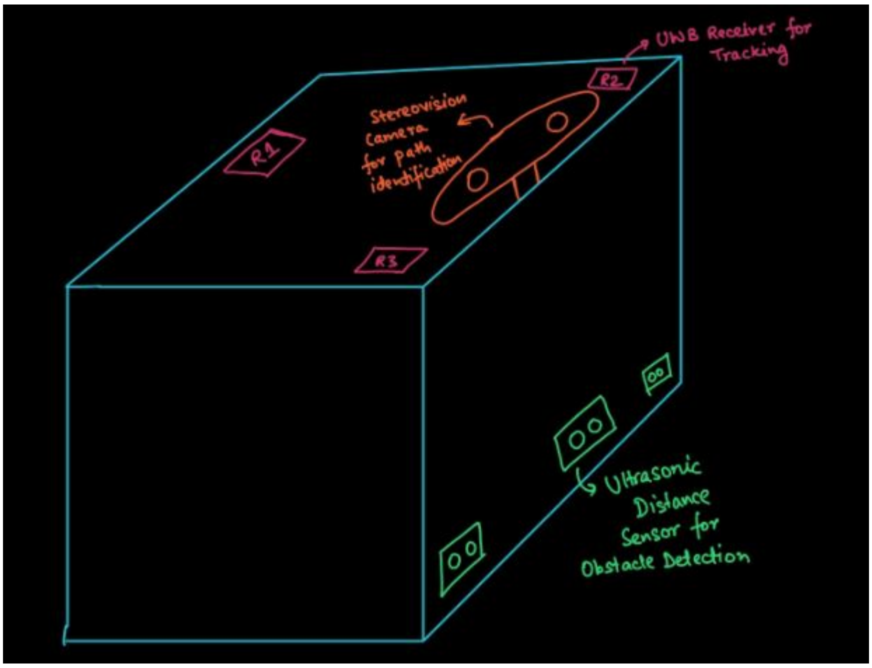


Figure 16: Mockup for Vision components

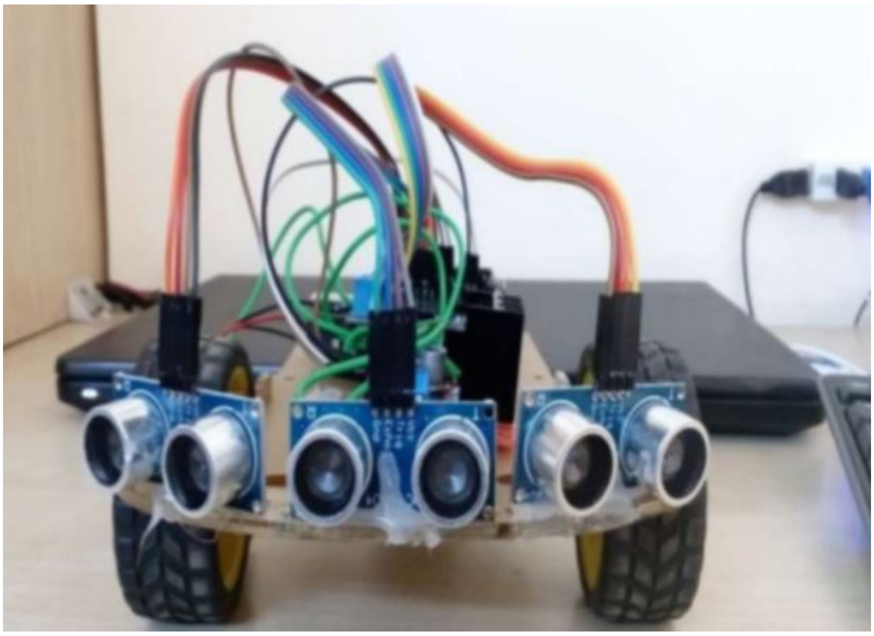


Figure 17: Model Design for the bot

Following are the dataset requirements as the bot needs to be able to recognise obstacles, and hence must know how to detect everyday objects and feet of people:-

- The people following CNN[34], which allows us to train our model to recognize people who might be in front of it. It consists of stereo images of 9 indoor and 2 outdoor sequences where each sequence has been 2000 and 12000 frames.
- ImageNet[35] to train our model to detect common everyday objects. The dataset consists of more than 14 million objects organized into 20000 categories.

3 Design

3.1 Electrical Connections

Figure 18 shows the Circuit Diagram of the mule bot which we have designed.

It consists of the following parts (as labelled in Figure 18):

1. A battery charging chip with USB input.
2. A battery pack at the top denoted by cell arrangement as shown.
3. a current limiter to prevent overdrawing of current.
4. MAX17613C IC to hold the voltage output steady.
5. A raspberry pi module with 1 IC, 1 port chip and a display.
6. A motor driver attached to the raspberry pi module.
7. A speed sensor attached to the raspberry pi module.
8. A weight sensor attached to the same module with a driver.
9. A ultrasonic sensor connected to the same module.
10. A UWB sensor attached to the same module.
11. A camera feeding digital input into the raspberry pi circuit.
12. Gyroscope attached to the same module.

Note: The device functions fine even if there is a malfunction in the battery. The charging circuit is directly connected to the input of MAX17613C along with the battery box, hence, it can directly supply power from mains.

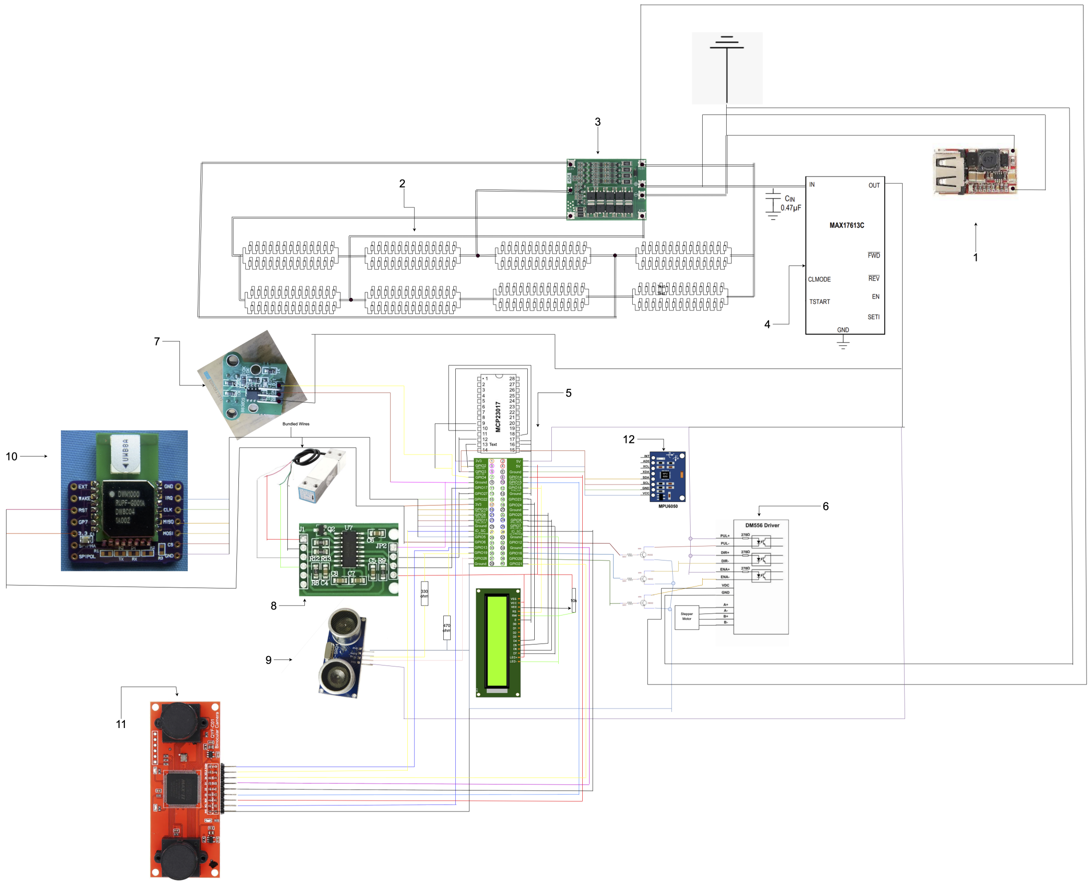


Figure 18: Circuit Design

If you are using an electronic version of this document, you might be interested in using the "Zoom In" feature to see finer details, if required, as the image is of a very high resolution.

3.2 Sensing/Vision Component

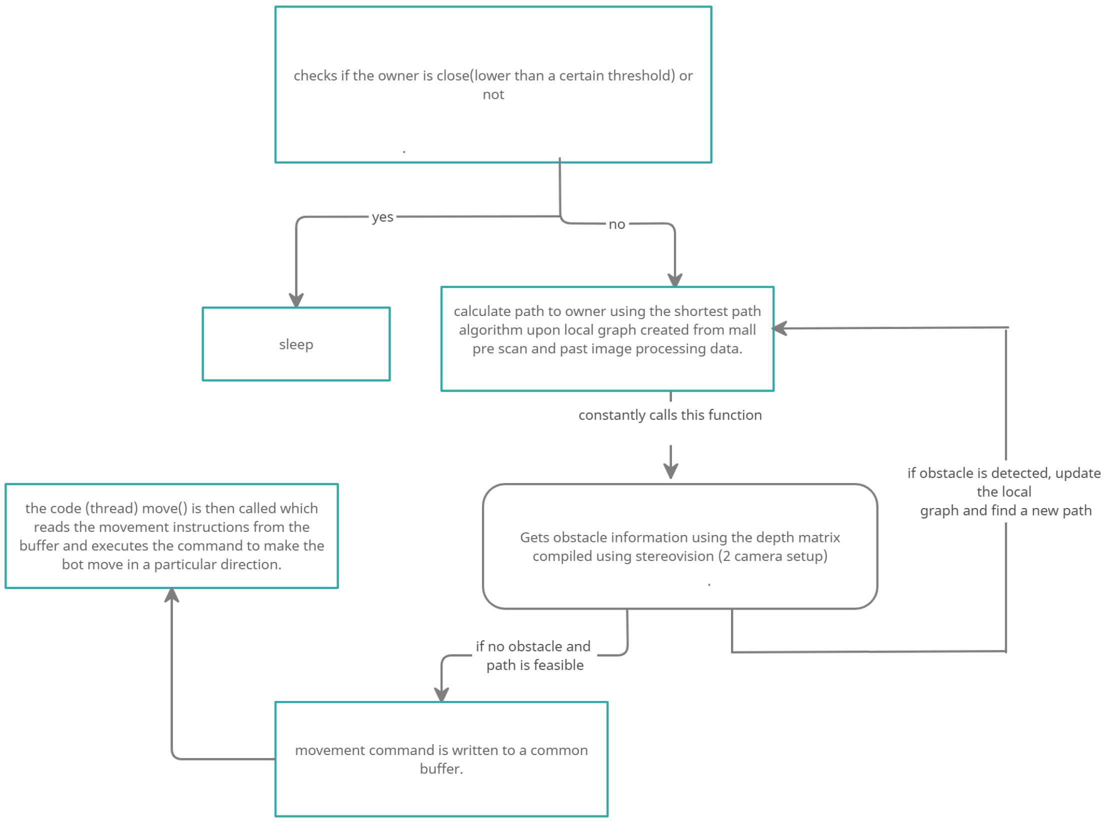


Figure 19: Flow Chart depicting the overall workflow of our Mule Bot

Figure 19 shows the workflow of our Mule Bot. First the bot checks if it is in the proximity of the owner. If yes, then it sleeps. If not, the bot undergoes a series of steps in an algorithm which helps it to get near the owner. The shortest path to the owner is first calculated using a local graph generated from mall data. Obstacles are avoided using a 2 camera setup for depth perception using stereo-vision. Figure 20 shows the workflow for our ultrasonic component, which looks for immediate obstructions and stops the bot. The relevant codes for our design can be found in Appendix A.

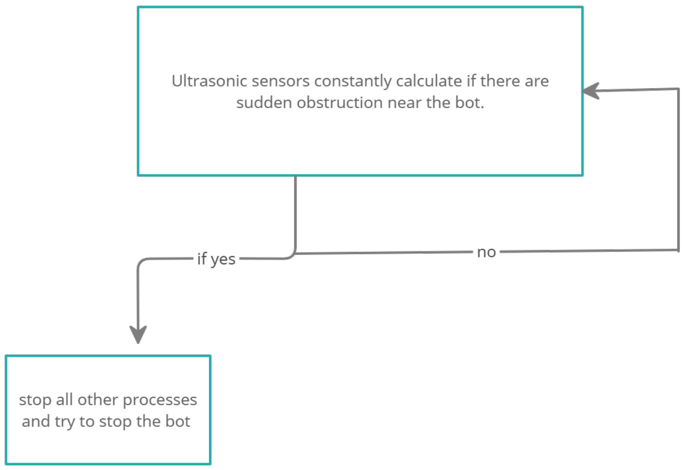


Figure 20: Flow Chart for Ultrasonic Sensor Component

3.3 Enclosure Components

Below are the specifications of various components of enclosure of our bot.

3.3.1 Cover A

This is a plastic cover that is used to enclose the electrical circuit parts and protect them from exposure to foreign materials.

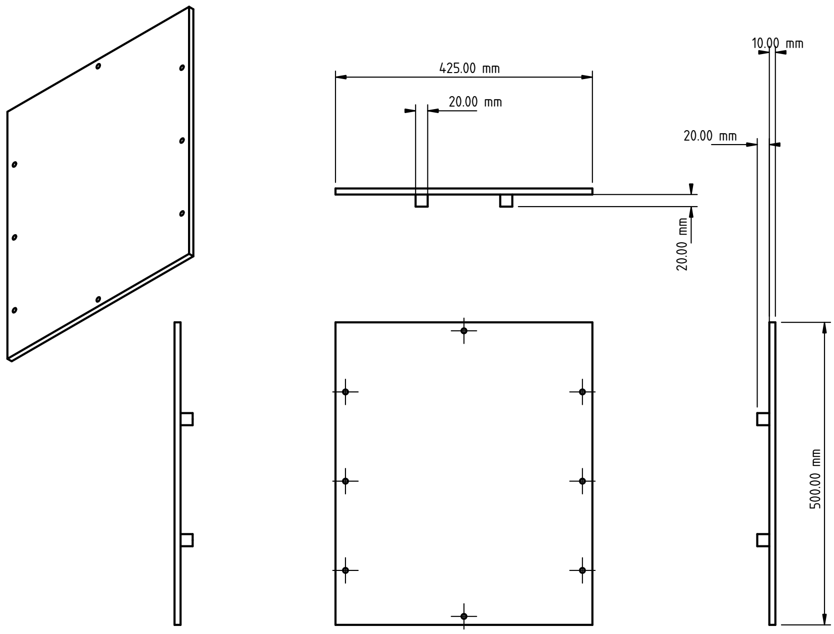


Figure 21: Cover A

3.3.2 Cover B

This covers the battery and separates the electrical components from the shopping items.

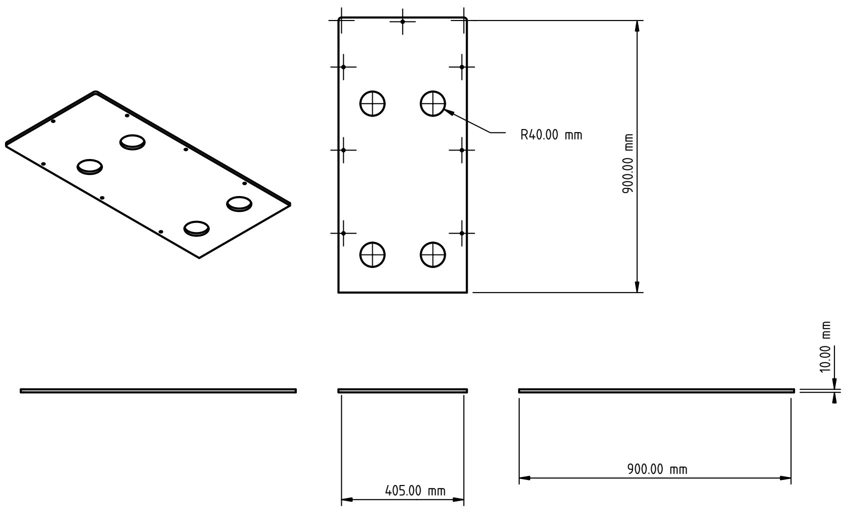


Figure 22: Cover B

3.3.3 Support

This component takes the weight of batteries and distributes it over the wheels, and is made up of aluminium, a strong but light metal.

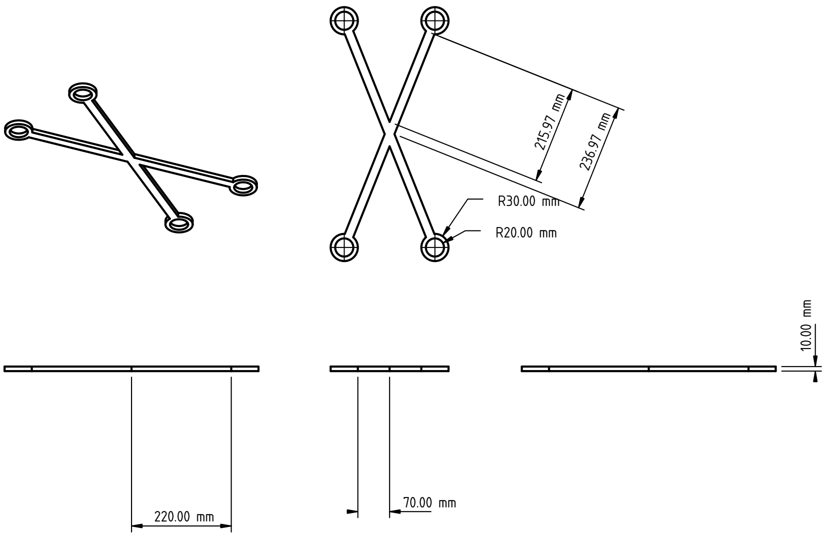


Figure 23: Support for our Mule Bot

3.3.4 Wheel

Wheels have been mentioned in the specifications (Section 2.2.7). They have to be purchased separately. The design only reflects the dimensions of the wheels.

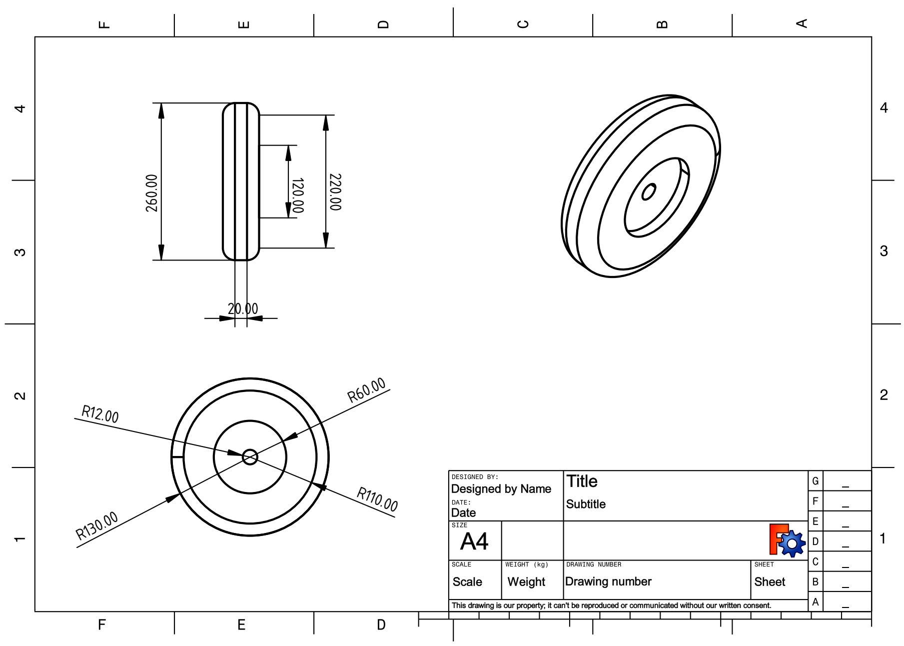


Figure 24: Dimensions of wheel used in Mule Bot

3.3.5 Body

This is the outer casing with dimensions specified. The thickness of casing is 10mm. The casing is made of strong and light plastic named acetal.

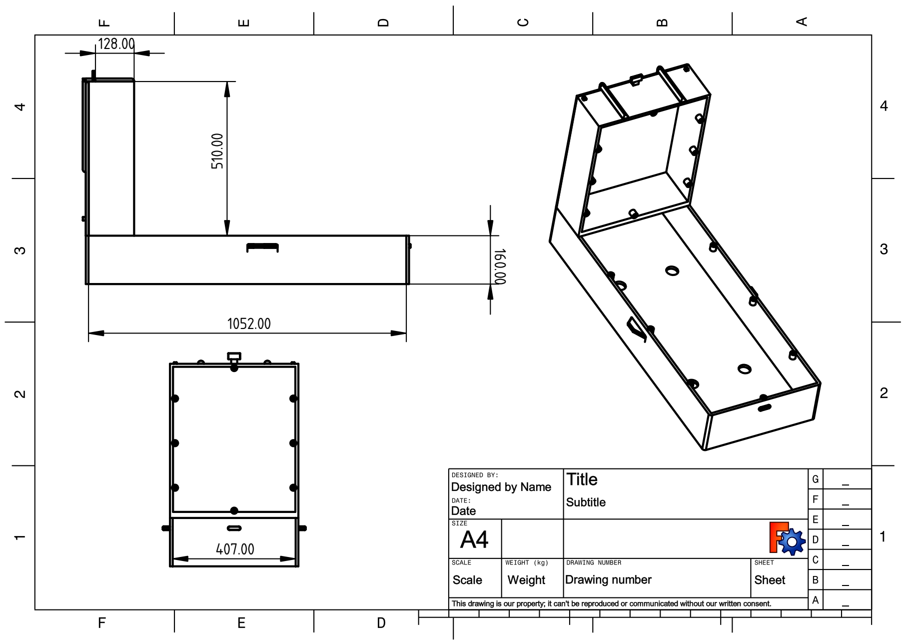


Figure 25: Body Dimensions for Mule Bot

3.4 3-Dimensional Views of Trolley

3.4.1 Front View

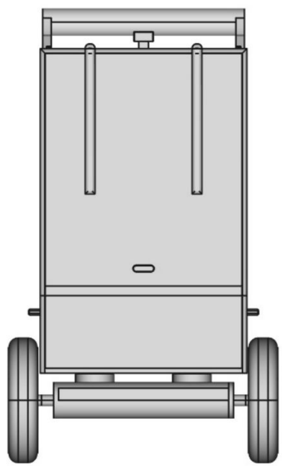


Figure 26: Front View of trolley

3.4.2 Side View

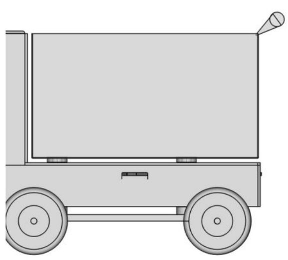


Figure 27: Side View of trolley

3.4.3 Back View

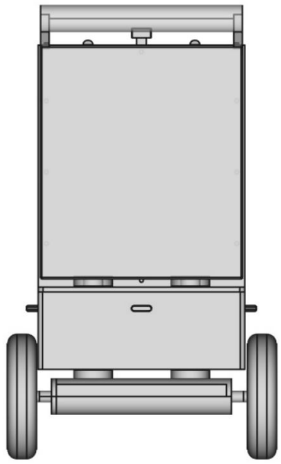


Figure 28: Back View of trolley

3.4.4 Bottom View (Bin)

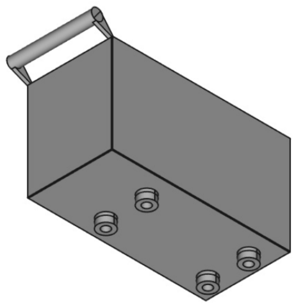


Figure 29: Bottom View of trolley bin

3.4.5 Isometric View

The covering through which we can see the base rods (where the handle is attached) is made of strong wire mesh to hold the items in the basket. The slots we can see on the top and the sides of the bot are the places where the camera and sensors are mounted. A rough diagram of the mounting is given below.

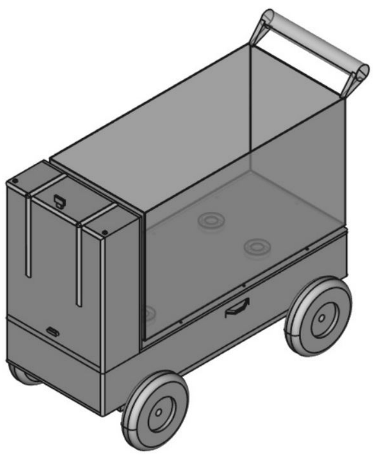


Figure 30: Isometric View of trolley

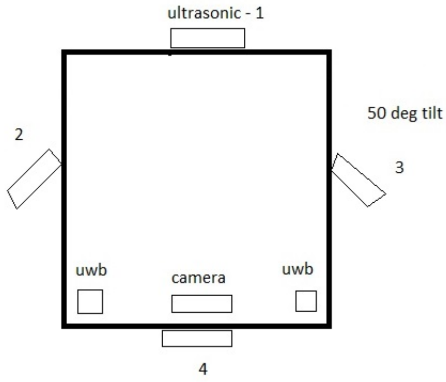


Figure 31: Sensor placement

In Figure 31, the labels 1, 2, 3 and 4 correspond to ultrasonic sensors and the rest of the sensors are at positions as mentioned in the diagram.

3.4.6 Transparent Body View

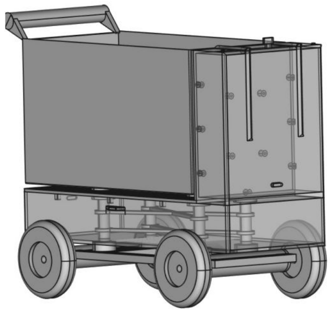


Figure 32: Transparent Body View of trolley

From the above diagram we can see that there are four rods which essentially hold the weight of the items kept in the basket. Thus these rods are to be of solid iron and the layer above them have to be of steel (3 mm) layered with plastic. This will carry the weight of the entire bot as well as the battery and will directly transfer it to the wheels while keeping the center of gravity low to avoid toppling.

4 Conclusion

4.1 Summary of the design

1. The battery management system manages the rechargeable battery by protecting the battery by keeping it within its safe operating area.

2. A charging system is connected to the power supply. The output of the charger is fed to a current limiter MAX17613C to provide overdrive of current, which fed power to a battery management system to protect the battery from sudden fluctuations in supply.

3. Two stepper motors were connected to the two wheels to enable changing the direction of movement of the bot. The stepper motor was fed by the DM556T driver to which the micro-controller fed maneuvering instructions for rotating the two wheels at different angular speeds to turn right or left.

4. The MPU6050, integrated with 3-axis accelerometer and gyroscope, fed the orientation and rotation parameters of the bot to the micro-controller which were later used for steering.

5. The load cell YZC-97 attached to base of the platform on which the user keeps one's stuff, sends a weight-calibrated electrical signal to the load cell HX711 (weight measuring sensor) which, in turn, informs the

micro-controller of the current weight of the items stored in the container. When the user exceeds the weight limit the Alarm system will kick into action and warn the user.

6. DecaWave DW1000 Single Chips (a UWB sensor) are attached to the mule bot to perceive the depth of various objects in front of the mule.
7. A speed sensor module containing LM393 was placed with a wheel (or the axle) in a slot between the two narrow cuboidal blocks, which shine Infrared light to track the number of rotations per minute (much like a tachometer).
8. The bot receives data from the user about the location it has to reach through the receiver, and this data is processed into the circuit.
9. Using the ultrasonic sensor and the stereo-vision camera, the circuit calculates the path to the user from the above data in form of a depth matrix.
10. The bot moves in this path. If any unidentified obstacle is detected by the ultrasonic sensor, the bot stops and the path is calculated again. Otherwise, the path is calculated in such intervals so that the bot maintains a 1m distance from the user.

4.2 Customer's Viewpoint

1. The body design of the bot is made such that it is able to take weights up to 35 Kgs from the user without causing any mechanical damage.
2. The bot maintains a 1m distance from the user while movement, but if called upon, can reach the user to his exact location.
3. The model of the mule bot is very similar to that of the trolley that is used nowadays in shopping malls. So it doesn't feel altogether like a very different product. This will be convenient for the user. Hence, the user would require minimal prior training to use the product.
4. There will be an app that the user can download from the store, to assist him in operating the bot. The user needs to identify a bot using his app. Then, he can call the bot, or set a distance for it to follow him, etc. The bot will only send the data of the gyroscope of the mobile to our bot, so there is no privacy concerns (No use of GPS data etc)
5. If anyone feels to drive the mule bot themselves without automation, can do that as well.
6. One can forget about where the trolley is and focus on shopping without any worries. They can call the bot anytime.

4.3 Battery Life and Charging Methods

If many of these bots are operating in a mall, where multiple users visit everyday, then there needs to be a management system (kind of like a future self-driving-taxi stand). This can have the following features.

1. We will supply 200V to 5 V 3 A chargers to charge the bots.
2. There will be multiple automatic charging sections throughout the mall, scattered around.
3. When the bots have a low battery charge, they will themselves navigate to the nearing charging stations. Depending on the Mall, one can incorporate these locations into the bot by default, and even attach an inductive charging mechanism to charge the bots in a wireless manner.
4. Users can be encouraged to leave the bots at the nearest charging station, by providing some monetary benefits (discount/coupons).
5. At the end of the day, the bots can be programmed to automatically return to an assigned spot in one of the charging stations at the mall closing time (say 9 PM)

The net capacity of the battery is 540.8 Ah, so it should take **around 10 Hrs to fully charge** at 5V 3A input. The max power requirement of the Motors is 100 W (at full load), which implies the bot has a max work-time of **1.5 Hrs at continuous full load**. But, in our case, we do not have a continuous full load (as it would need to stop multiple times while the user is browsing through stuff at the mall, get multiple opportunities to recharge at intervals etc). So, if we take the average power requirement as 30 W, the bot will run for around **5 Hours at full battery**. Any user generally shops at a mall for max 3-4 Hrs, so this working time is sufficient. Also, charging the bot in intervals will increase this time further. So, the user does not face any inconvenience in this regard.

4.4 Perks of our model

1. The support for battery is in such a way that they work as a casing for the battery so that no pressure from the cart can potentially damage our battery.
2. The front of the trolley contains two vertical translucent plastic strips, under these strips we can put led lights so that when the weight is under 40 then the light is green and in case of overweight the strip goes red. This would provide an interactive way to manage overloading.
3. The Bin is directly mounted on the iron rods coming from the wheel chassis (along with a suspension in between), so that whatever weight is put on the wheel would go directly down on the wheels without much affecting our plastic case or components inside.
4. The bin also includes a handle which can be used to move around the trolley in case the battery runs out.

4.5 Future Scope of Improvement

1. We can add RFID sensors for authentication in the future for a faster user experience.
2. Secure communication protocols can be developed between the user and the bot. As of now, a password which is private between the user's mobile app and the bot is used for security. This has limitations.
3. Battery with better Ah to weight ratio can be used, making the bot lighter and faster.
4. We can add bar-code scanners, which automatically scans an item and shows it's store price for the convenience of the user.
5. An alarm system can be added in the mall which goes off once someone tries to take any bot outside it's designated area , i.e. the mall boundary. This can work as a good anti-theft practice.
6. Communication among different bots can be added, leading to a more smooth user experience.
7. A lid can be attached on the top, which can be locked and unlocked by the user's personal app. This will keep the shopping items secure even if the user is busy in some other activity.
8. The basket can be made detachable to make exchange of items easier between bots. (We have not implemented this as this would reduce both the security as well as the strength of the bot, decreasing performance)

In future, we can search for design improvements that includes these features without compromising the strength and security of the bot.

5 Budgeting

5.1 Time Budgeting

For the designing of our bot, our team of 40 was broken down into 4 teams, 3 teams of 11 each and one documentation team of 7 members. They worked for a week and half on the design part of this project.

1. The work was planned for 10 days.
2. The first day, an overall meeting was called to give the overall idea.
3. the following 3 days constituted of team-level meetings, under coordinator of each teams.
4. The next 5 days were the main "work days" where all the code formulation and designing took place.
5. Finally, we wrapped up with a overall meeting to discuss how much we attained our goals.

The division of work into different teams helped individuals with more experience in specific areas like assembly, raspberry Pi coding, etc. led to a better and robust design with a bundle of small details that will make enhance the user experience. The previous sections provided a deeper insight into our model and the overall user experience.

5.2 Manufacturing time and cost

Many of the parts in our bot are purchased. The things that are customised are the body (made of acetal plastic) and the wiring of the bot. We also have to incorporate coding of the bot and assembly into consideration.

1. The body can be made using standard procedures (CNC machining)
2. Assembly, wiring and coding incur labor costs.

So, in factory scale, the estimated time of production can be around 4 hours given all the materials we ask in BOM are available. Assembly will take one more hour. So, the factory can manufacture 50 bots parallel in 5 hours at industrial scale.

Manufacturing cost includes the total cost in BOM 5.4, the CNC cost, and the assembly cost. This will depend on the manufacturing technology.

5.3 ELPU (Euro-Pound-Peso-Units) Man Costing

The basic idea is modelled below

1. Let us consider the Team coordinator gets a remuneration of 100 ELPU per project.
2. Then the other coordinators working full time will get a remuneration of 95 ELPU. (Company Policy)
3. Each person working on the project will get a base pay of 50 ELPU and depending on the time they gave to the project, a bonus of 40-43 ELPU.
4. One day extra work will require paying everyone a 10 ELPU fee + bonus.
5. Working for half the time will need a base pay of 30 ELPU, which is the minimum wage.
6. Here, there are 40 members, among which 6 worked as coordinators and 1 lead coordinator.
7. 5 members worked half time.
8. Among the other 28 members, 17 of them worked hard and brought their bonus upto 40 ELPU.
9. Rest 7 of the others had devoted plenty of time to gain a 20 ELPU bonus.
10. The rest of the people got a 5 ELPU bonus.
11. Thus the total payable amount comes out around (100+95*6+90*17+70*7+55*4+30*5) ELPU = **3060 ELPU** for this week of the project.

5.4 Bill of Materials

Bill of Materials			
Name	**Quantity**	**Rate *(INR)***	**Amount *(INR)***
Batteries	1	25760	25760
CentIoT - 4S 40A 14.8V 16.8V BMS PCM PCB for 4S Li LicoO2 Limn2O4 18650 Battery - with Balancer	1	300	300
Step down power charger	1	125	125
MAX17613A	1	179	179
Stepper motor drive	1	649	649
GY-521	1	220	220
Hx711	1	120	120
Rasberry PI 3	1	2691	2691
RS PRO Black	1	1406	1406
Shockers	1	701	701
Acetal sheets	1	1449	1449
L298N	1	105	105
Motors	4	6000	24000
6mm hex coupelling	1	78	78
Wheels	2	600	1200
US sensor	4	59	236
Camera	1	3450	3450
		Total (Approx):	**62,600**

References

[1] *Person Tracking Mobile Robot.* URL: https://core.ac.uk/download/pdf/48537093.pdf.

[2] *BatteryPack Calculator.* URL: https://www.dnkpower.com/18650-battery-pack-calculator/.

[3] *18650 Li Cell Datasheet.* URL: https://www.ineltro.ch/media/downloads/SAAItem/45/45958/36e3e7f3-2049-4adb-a2a7-79c654d92915.pdf.

[4] *Current Limiter.* URL: https://datasheets.maximintegrated.com/en/ds/MAX17613A-MAX17613C.pdf.

[5] *Battery Management System.* URL: https://www.amazon.in/Battery-Charging-Protection-lithium-Limn2O4/dp/B07H2FYGDY.

[6] *Step Down Power Charger.* URL: https://robu.in/product/dc-to-dc-6-24v-to-5v-usb-output-step-down-power-charger-with-adjustable-buck-converter/?gclid=CjwKCAjw6fCCBhBNEiwAem5SOwK9mLN8j8N1sqeuXr2lFJJ7WAinwn7KyiOv-WeQtcJaZaYBvEHYKBoC5dcQAvBwE.

[7] *LM393 Speed Sensor.* URL: https://www.electronicshub.org/interfacing-lm393-speed-sensor-with-arduino/.

[8] *LM393 Properties.* URL: https://5.imimg.com/data5/VQ/DC/MY-1833510/lm393-motor-speed-measuring-sensor-module-for-arduino.pdf.

[9] *Robot Steering.* URL: http://faculty.salina.k-state.edu/tim/robotics_sg/Algorithms/steering.html.

[10] *DM556T User Manual.* URL: https://www.omc-stepperonline.com/download/DM556T.pdf.

[11] *MPU6050 - Gyroscope Interfacing.* URL: https://www.electronicwings.com/raspberry-pi/mpu6050-accelerometergyroscope-interfacing-with-raspberry-pi#:~:text=It%20can%20accept%20inputs%20from,module%20using%20I2C%20communication%20protocol.

[12] *MPU6050 product.* URL: https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-6050/.

[13] *MPU6050 datasheet.* URL: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf.

[14] *MPU6050 dimensions.* URL: https://www.addicore.com/GY-521-MPU6050-p/170.htm#:~:text=Dimensions%20(excluding%20pins)%3A%2021.2,%3A%202.1g%20(0.07oz).

[15] *Weight sensing.* URL: https://circuitdigest.com/microcontroller-projects/raspberry-pi-weight-sensing-automatic-gate.

[16] *Load Cell.* URL: https://www.amazon.com/gp/product/B00R1J9LEY/ref=as_li_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=B00R1J9LEY&linkCode=as2&tag=shophow2elect-20&linkId=2eccd3f5c08001f40d5740360c1beea7#descriptionAndDetails.

[17] *HX711 Datasheet.* URL: https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf.

[18] *HX711 Product.* URL: https://robu.in/product/hx711-weighing-sensor-dual-channel-24-bit-precision-ad-weight-pressure-sensor/.

[19] *Raspberry Pi Documentation.* URL: https://www.raspberrypi.org/documentation/hardware/raspberrypi/power/README.md#:~:text=The%20power%20supply%20requirements%20differ,uses%20a%20USB%2DC%20connector.

[20] *MCP23017 Datasheet.* URL: https://ww1.microchip.com/downloads/en/devicedoc/20001952c.pdf.

[21] *MCP23017 Input Pin Expansion.* URL: https://www.mathworks.com/help/supportpkg/raspberrypiio/ref/add-digital-i-o-pins-to-raspberry-pi-hardware-using-mcp23017.html#:~:text=Raspberry%20Pi%20hardware%20has%20a,as%20digital%20inputs%20or%20outputs.

[22] *Mobile - Raspberry Pi Connection.* URL: https://videos.cctvcamerapros.com/digital-io-alarm-in-out/send-push-notifications-from-raspberry-pi.htm.

[23] *506-3078 RS PRO | RS PRO Black Rubber Sheet, 1m x 600mm x 1.5mm | 506-3078 | RS Components.* URL: https://in.rsdelivers.com/product/rs-pro/506-3078/rs-pro-black-rubber-sheet-1m-x-600mm-x-15mm/5063078?cm_mmc=IN-PLA-DS3A-_-google-_-PLA_IN_EN_Engineering_Materials_%26_Industrial_Hardware_Whoop-_-(IN:Whoop!)+Rubber+Sheets-_-PRODUCT_GROUP&matchtype=&pla-326961060940&s_kwcid=AL!7457!3!474344388154!!!g!326961060940!&gclid=CjwKCAjwxuuCBhATEiwAIIIz0ezCfl_wXmQXQRFvdVsmRaPdQyZg5XpyDplhjW-BoN1WyvedfRQ9sBoCHoMQAvD_BwE&gclsrc=aw.ds (visited on 03/24/2021).

[24] *65mm Metal Front/Rear Shock Absorber for RC Car (Pack of 4).* URL: https://robu.in/product/65mm-metal-front-rear-shock-absorber-for-rc-car-pack-of-4/?gclid=Cj0KCQjwjPaCBhDkARIs Qip-ilyEm4qQHP7xYyPcDN0QqhuRrsPZickaAq4EEALw_wcB (visited on 03/24/2021).

[25] *Acetal Plastic | Low Friction, Wear Properties | Curbell Plastics.* URL: https://www.curbellplastics.com/Research-Solutions/Materials/Acetal (visited on 03/24/2021).

[26] *Acetal Datasheet.* URL: https://www.curbellplastics.com/Research-Solutions/Technical-Resources/Technical-Resources/Acetal-Data-Sheet.

[27] *L298N DC Stepper Motor Driving Module 5V 2A For Arduino | ThinkRobotics.in.* URL: https://thinkrobotics.in/products/l298n-dc-stepper-motor-driving-module-5v-2a-for-arduino?pr_prod_strat=copurchase&pr_rec_pid=1665095958600&pr_ref_pid=1621190082632&pr_seq=uniform (visited on 03/24/2021).

[28] MACFOS. *Buy NEMA34 Hybrid Stepper Motor 87 Kg-cm Online at the Best Price.* Robu.in | Indian Online Store | RC Hobby | Robotics. URL: https://robu.in/product/nema-34-87-kg-cm-hybrid-bipolar-stepper-motor-86sth118-5004a14/ (visited on 03/24/2021).

[29] *130mm Wheel Set with Couplers | ThinkRobotics — ThinkRobotics.in.* URL: https://thinkrobotics.in/products/130mm-wheel-set-with-mounting-brackets?variant=15984448110664 (visited on 03/24/2021).

[30] *Buy 6mm hex brass coupling 18mm length for robot car wheel online.* URL: https://robu.in/product/6mm-hex-coupling-robot-smart-car-wheel-18mm-length/?gclid=CjwKCAjwxuuCBhATEiw BwE (visited on 03/24/2021).

[31] *OV2640 Binocular Camera Module CMOS.* URL: https://robu.in/product/ov2640-binocular-camera-module-cmos-stm32-driver-3-3v-16001200-for-3d-measurement-with-sccb-interface/?gclid=CjwKCAjwxuuCBhATEiwAIIIz0QSm1D77EUUW0-XgF4ozAasVtSpW51_X-tbmiKE2KrBV8e25vrjyPRoCjSkQAvD_BwE.

[32] *Single Chip UWB Wireless Transceiver.* URL: https://www.semiconductorstore.com/cart/pc/viewPrd.asp?idproduct=50015.

[33] *Ultrasonic Sensor Chip.* URL: https://www.electronicscomp.com/ultrasonic-distance-sensor-module-hc-sr04-india?gclid=CjwKCAjwxuuCBhATEiwAIIIz0cv46JX1W7C_Qy6NwOodl35OZOn BwE.

[34] *Person Following CNN.* URL: http://jtl.lassonde.yorku.ca/2017/05/person-following-cnn/.

[35] *ImageNet.* URL: http://www.image-net.org/.

A Code Snippets

A.1 Workflow

The following is our code for Workflow of our bot:

```
thread move() {
  if(the buffer is not empty()) {
    // stores which direction our robot is currently facing (N, S, E or W)
    current_orientation = get_current_orientation();
    // a sequence of N, S, E, W commands
    path = read_path_from_buffer();
    // number of commands in our path
    size = path.size();
    while path is not empty {
      // if the path given to us "NNNWSSENS", then the largest same prefix is "NNN"
      string prefix = largest_same_prefix(path);
      path.erase(prefix);
      char target_direction = prefix.front();
```

```
      // we must take cnt steps in the direction target_direction
      int cnt = prefix.size();
      move_wheels(current_orientation, target_direction, cnt);
    }
}
// the map is viewed as square tiles of BLOCK_SIZE * BLOCK_SIZE m^2 dimensions
const double BLOCK_SIZE = get_block_size();
// set the maximum rpm of wheels
const double MAX_RPM = get_max_rpm();
// set the radius of the wheel
const double WHEEL_RADIUS = get_wheel_radius();


function move_wheels(current_orientation, target_direction, cnt) {

    // move the robot wheels in such a way that we take cnt steps in direction
    // target_direction, if initially we are facing direction current_orientation

    // turn the robot so we are facing direction target_direction
    make_turn(current orientation, target_direction);

    // time taken for our robot to move cnt steps in the required direction
    int movement_timer = 60 * BLOCK_SIZE * cnt / (2 * PI * MAX_RPM * WHEEL_DIAMETER);

    // we use threads to move both the left wheels and right wheels move simultaneously
    Thread t1 = get_thread_object(turn_left_wheels(MAX_RPM, forward, movement_timer));
    Thread t2 = get_thread_object(turn_right_wheels(MAX_RPM, forward, movement_timer));
    t1.start();
    t2.start();
    // update the current orientation of the robot
    set_current_orientation(target_direction);
}

thread turn_left_wheels(rpm, direction, timer) {
  // tell the controller to move the left wheels at speed rpm in direction
  // (forward/backward) for a time timer. The controller also updates current_orientation.
  pass_to_controller(rpm, direction, timer, left_wheels);
}

thread turn_right_wheels(rpm, direction, timer) {
    // tell the controller to move the right wheels at speed rpm in
    // direction (forward/backward) for a time timer. The controller also
    // updates current_orientation.
    pass_to_controller(rpm, direction, timer, right_wheels);
}

function make_turn(current_direction, target_direction) {
  // if the robot is currently facing direction current_direction, then we want
  to turn the robot such that it is facing direction target_direction
  while(current_direction != target_direction) {
    // our robot uses differential drive steering, so in order to turn the
    robot we move the left wheels and right wheels in opposite direction

    // this would make the robot rotate about its own axis
    Thread t1 = get_thread_object(turn_left_wheels(MAX_RPM/2,forward,0.1 sec));
    Thread t2 = get_thread_object(turn_right_wheels(MAX_RPM/2,backward,0.1 sec));
    t1.start();
    t2.start();
  }
}
```

Listing 1: Workflow

A.2 Shortest Path Calculation

The following code is to calculate the shortest path to the user.

```

//  mall_pre_scan[i][j][k] = 1 if i,j th space(descretized) has an isle on k th floor
// i.e. inaccessible to robots.


boolean mall_pre_scan [][][];

// discreteized in blocks of x*x;
Global time;
```

```

vector owner; // location of owner from beginning point
vector mule_pos; // location of mule from beginning point
owner_id = NULL;
synchronized buffer = [] // buffer for commands to be converted into motor output;
thread main(){
  // accounts for obstacle other than what is seen by mall objects
  // (shopper their carts etc) with time stamp at which they were
  // noticed to automatically invalidate obstacles after a threshold of time passed

  mall_pre_scan_dirty = mall_pre_scan;

  while(owner_id!=NULL){
    if(dist(owner,mule_pos)<threshold1 and
    bfs_distance_in_mall_pre_scan_dirty(mule_pos , owner) < threshold2){
      sleep(1);
      continue;
    }
    new_path = bfs_distance_in_mall_pre_scan_dirty;
    while(true){
      vision_o = vision_api // depth map of surroundings
      // this is an array of {distance, angle}
      ultrasonic_o = ultrasonic_api;
      for(auto p : ultrasonic) {
        distance = p.first; theta = p.second;
        distance_X = distance*cos(theta);
        distance_Y = distance*sin(theta);
        obstacle = {mule_pos.first + distance_X/x, mule_pos.second + distance_Y/x};
        //define that 2D array element as an obstacle
        mall_pre_scan_dirty(obstacle) = {true,time.current};
      }
      if(new_path is not blocked){
      // write to buffer
        buffer.push(new_path,top());
        new_path.pop();
      }
      else break;

    }
  }
}
thread send(){
    while(owner_id!=NULL){
        if(buffer,size()>0){
            // send input to motor threads
            buffer.pop();
        }
    }
}
```

Listing 2: Shortest Path Calculation

A.3 Stereo-vision For Depth Perception

The following is code for depth perception for 2 camera setup to avoid obstacles:

```
import standard lib
import opnecv as cv %cv2

#have 2 cameras

vidl = cv2.videocapture(leftcamera)
vidr = cv2.videocapture(rightcamera)

stereo = cv2.StereoBM.create()

while(True):
  retl, framel = vidl.read()
  reltr, framer = vidr.read()

  if retl and retr:

    # turn into grayscale
    framel = cv2.cvtcolor(framel, cv2.BGR2GRAY)
    framer = cv2.cvtcolor(framer, cv2.BGR2GRAY)

```

```
    #stereo rectification is important specially in epipolar geometry problems

    left_rectification = cv2.remap(framel,
          Left_Stereo_Map_x,
          Left_Stereo_Map_y,
          cv2.INTER_LANCZOS4,
          cv2.BORDER_CONSTANT,
          0)

    right_rectification = cv2.remap(framer,
          Left_Stereo_Map_x,
          Left_Stereo_Map_y,
          cv2.INTER_LANCZOS4,
          cv2.BORDER_CONSTANT,
          0)


  stereo.setNumDisparities(numDisparities)
  stereo.setBlockSize(blockSize)
  stereo.setPreFilterType(preFilterType)
  stereo.setPreFilterSize(preFilterSize)
  stereo.setPreFilterCap(preFilterCap)
  stereo.setTextureThreshold(textureThreshold)
  stereo.setUniquenessRatio(uniquenessRatio)
  stereo.setSpeckleRange(speckleRange)
  stereo.setSpeckleWindowSize(speckleWindowSize)
  stereo.setDisp12MaxDiff(disp12MaxDiff)
  stereo.setMinDisparity(minDisparity)

  # Calculating disparity using the StereoBM algorithm
  disparity = stereo.compute(Left_nice,Right_nice)

  ret, sol = cv2.solve(coeff,z,flags=cv2.DECOMP_QR)

  if ret:
    depth_matrix_code(so)l
```

Listing 3: Stereo-vision For Depth Perception

A.4 Ultrasound Component

The following is code for our ultrasound component.

```


def mover_ultra(){
  while(mule.location != target.location) {
    distance = UltraSonicSensor.Scan().distance

    if(distance < CRITC_DISTANCE2) {
        brake()
        while(UltraSonicSensor.Scan().distance < CRITC_DISTANCE2) {
            sleep(0.1) // sleep for 0.1 sec
        }
        accelerate(0.1); // accelerate for 0.1 sec
    }
    else if(distance < CRITC_DISTANCE1) {
        if(mule.speedSensor.speed > 0 ) {
            deaccelerate(0.1) // deaccelerate for 0.1 sec
    }
    else if(distance > SAFE_DISTANCE) {
        if(mule.speedSensor.speed < MAX_SPEED) {
            accelerate(0.1) // accelerate for 0.1 sec
        }
    }
    // if CRITC_DIS1 < distance < SAFE_DISTANCE, keep moving at same speed

    if(getDistance(mule.location, target.location) < CRITC_DISTANCE1) {
        if(mule.speedSensor.speed > SAFE_SPEED) {
            deaccelerate(0.1);
        }
    }
  }

  brake()
```

```
}
```

Listing 4: Ultrasound Component

B Document Statistics

The `.tex` file was converted to `.txt` format using CloudConvert[1] to obtain the document statistics from Online-Utility[2]. The names of the team members, *Table of Contents*, *References* and the sections in *Appendix* were excluded while getting the statistics.

Number of characters (without spaces):	24,433.00
Number of words:	5,406.00
Number of sentences:	456.00
Lexical Density:	63.54
Average number of characters per word:	4.52
Average number of syllables per word:	1.56
Average number of words per sentence:	11.86
Gunning Fog index:	9.69
Coleman Liau index:	8.29
Flesch Kincaid Grade level:	7.41
ARI (Automated Readability Index):	5.78
SMOG:	10.30
Flesch Reading Ease:	63.08

[1] https://cloudconvert.com/tex-to-txt
[2] https://www.online-utility.org/english/readability_test_and_improve.jsp