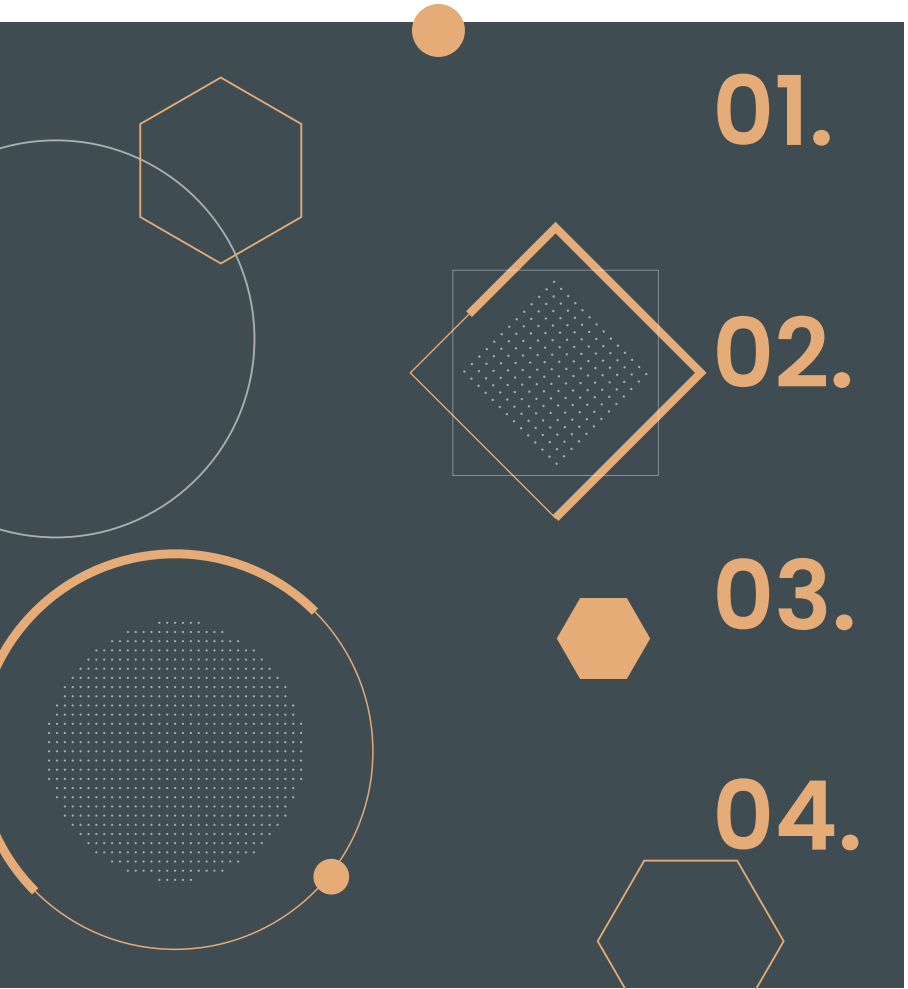
EG2310 Design Review

Group 3:	Agarwal Ananya	A0246120W
	Hang Jin Guang	A0254475X
	Loh Yin Heng	A0259282W
	Sim Yu, Jeanette	A0254736X
	Wang Bo	A0252200A



Introduction

Overview of Mission Flow

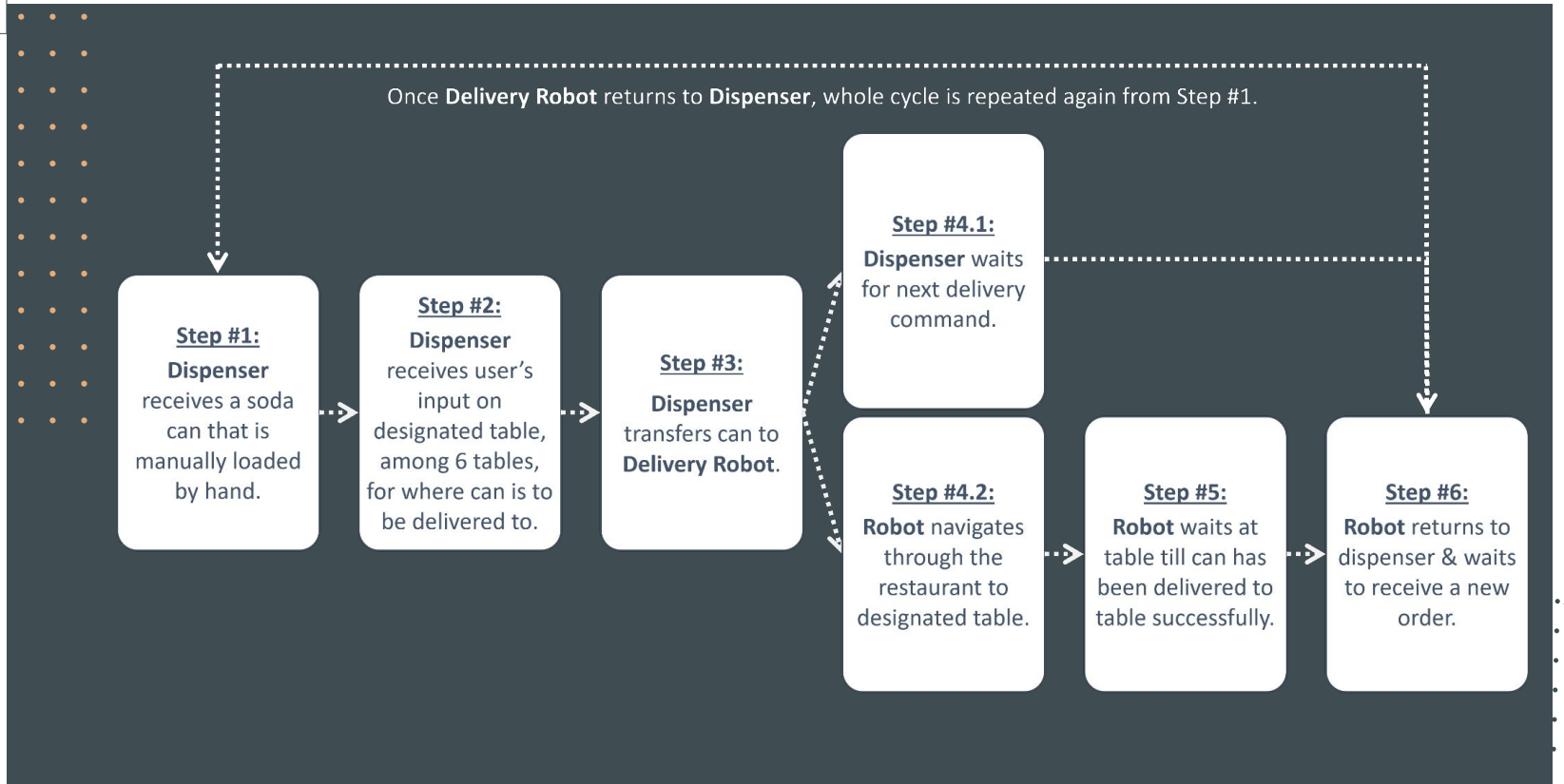
Mission Steps

Mechanical + Electrical + Software Subsystems

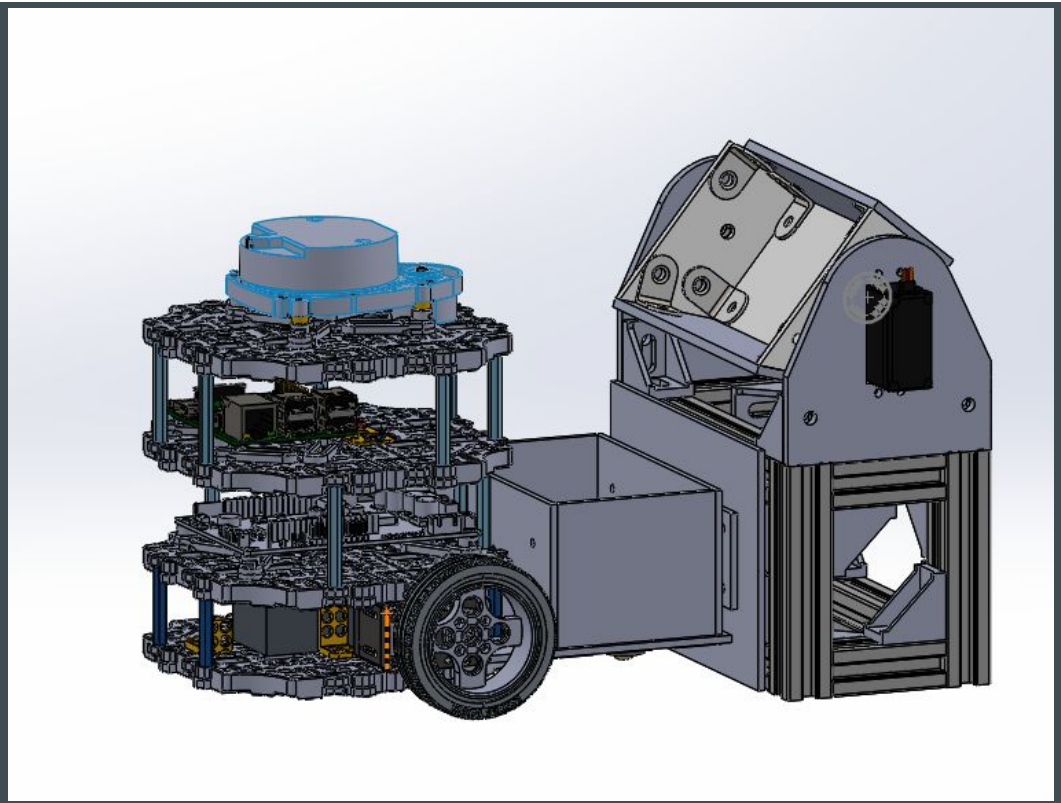
Testing Phase

BOM

Overview of Mission Flow



Integrated System



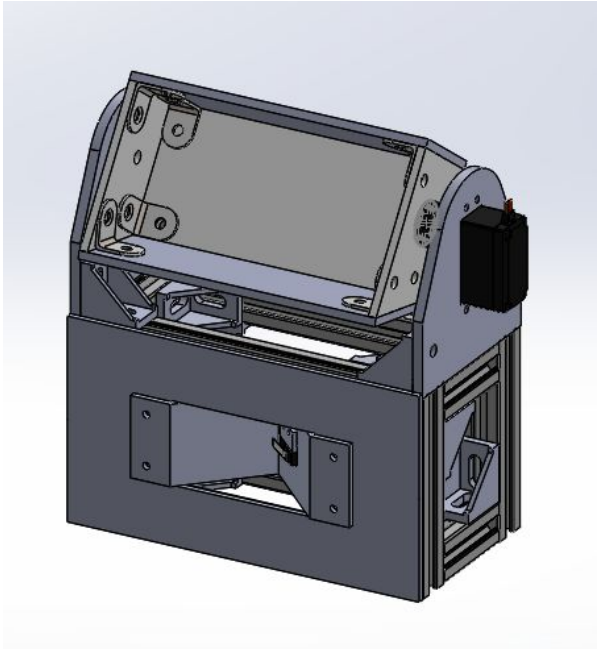
Dispenser

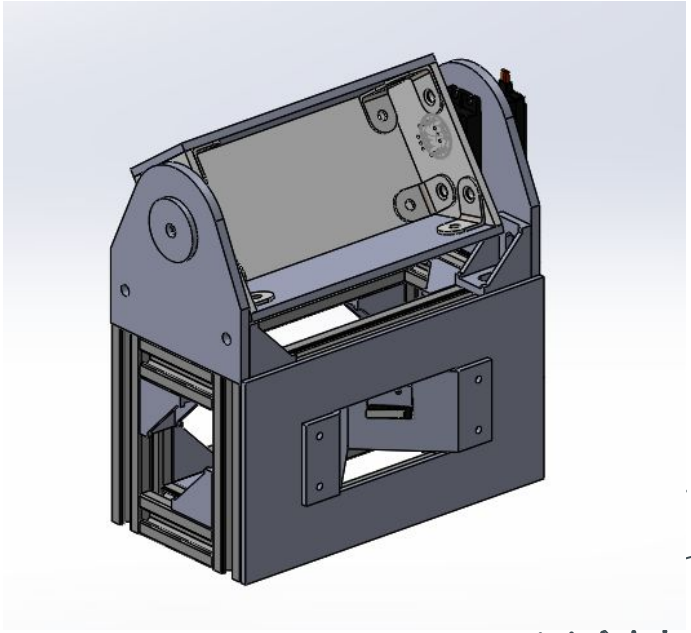
FROM GROUND UP

Working Principle of Dispenser

Assembly List:

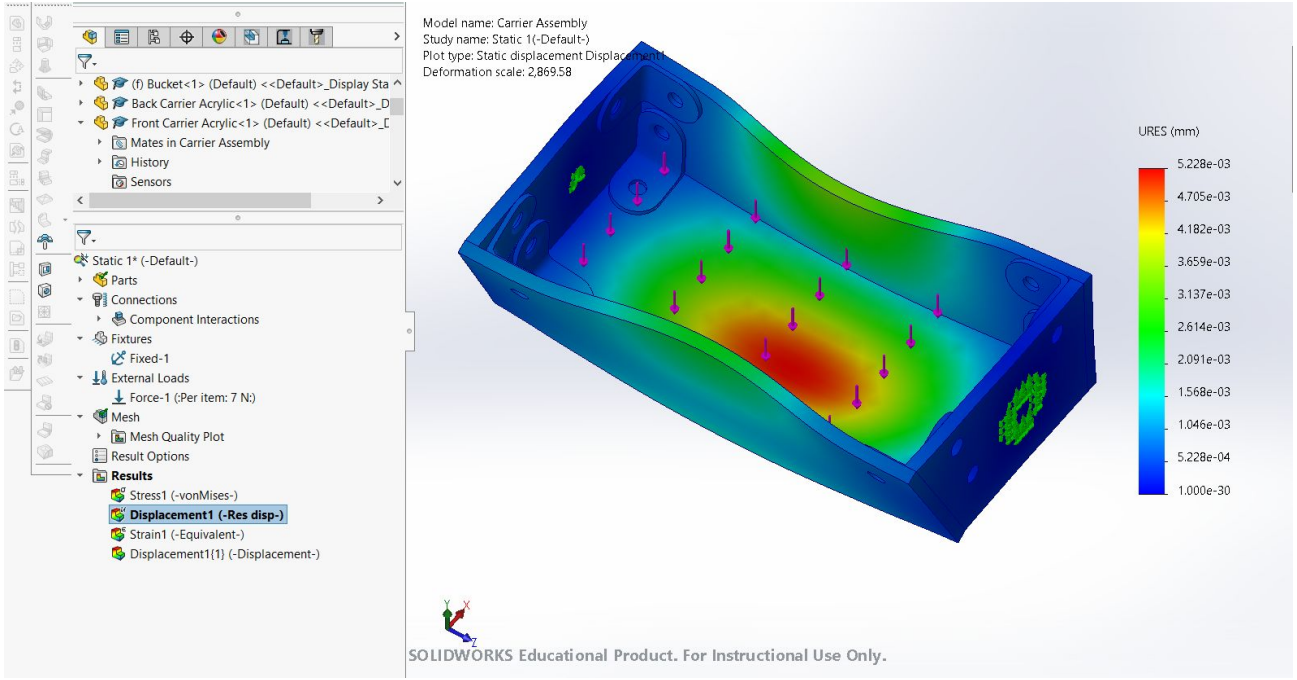
1. **Carrier**
2. **Profile Bar**
3. **Homing**
4. **Rotation Support**





Step #1: Dispenser receives can

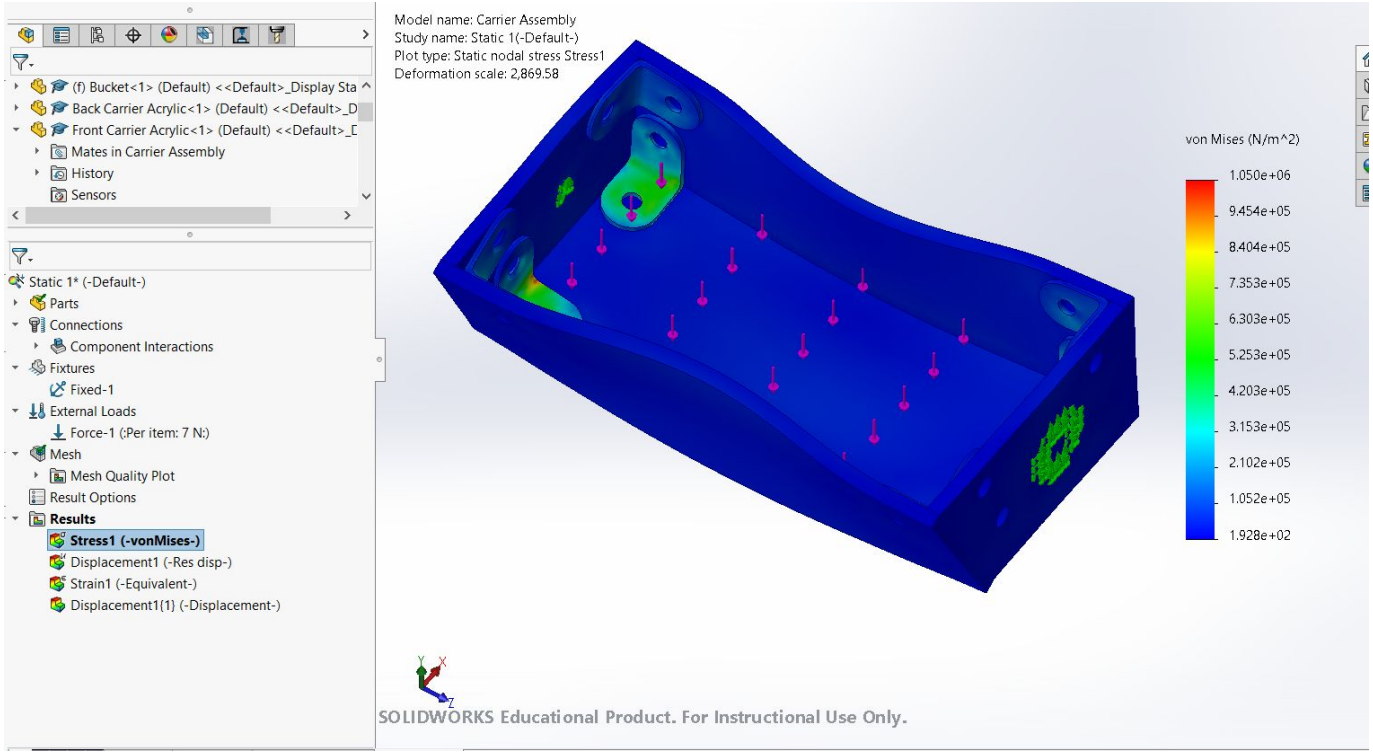
FEA for the carrier- Displacement



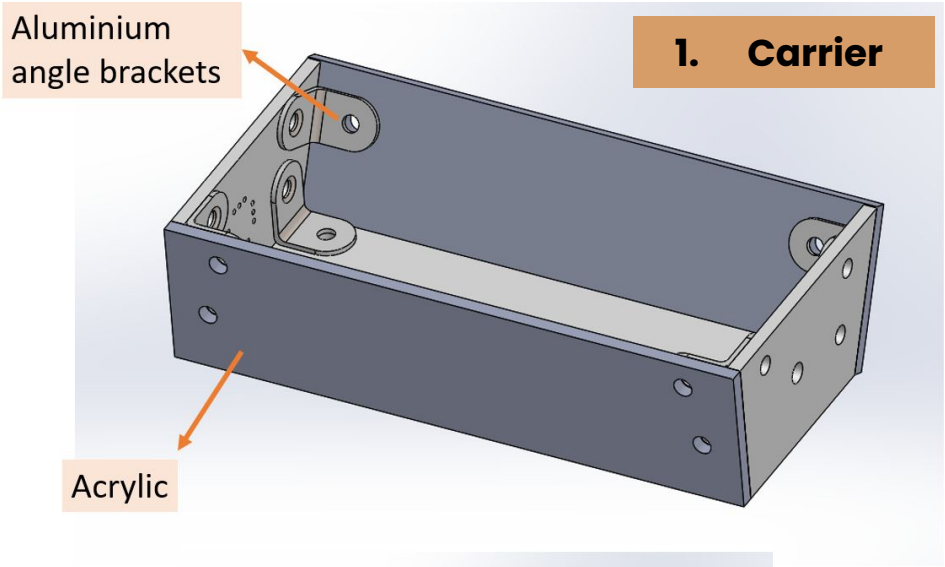
Maximum Displacement is **$5.2*10^{-3}$** mm

FEA- von Mises Stress

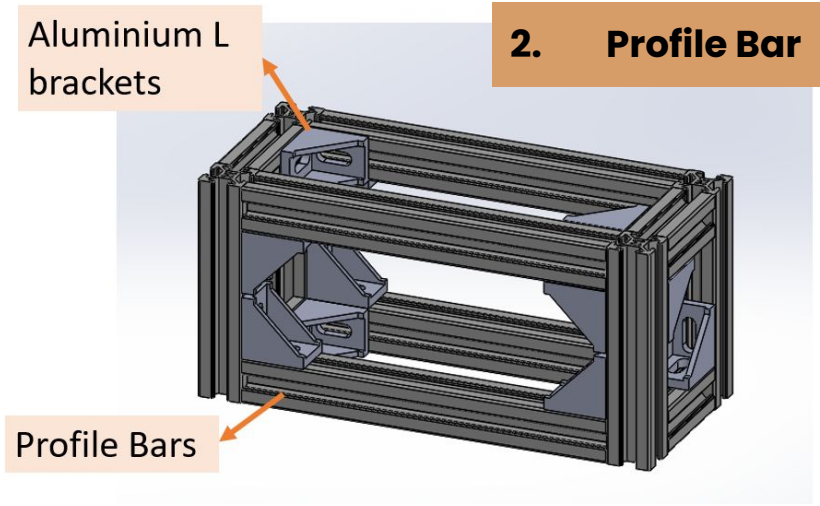
The tensile strength of the material AISI 304 is **between $4.85*10^8$ Rm N/mm^2**
For acrylic sheet it is **$7.5*10^7$ Rm N/mm^2**



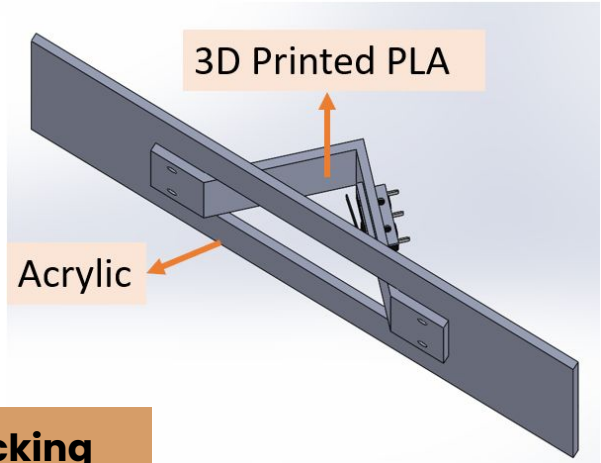
1. Carrier



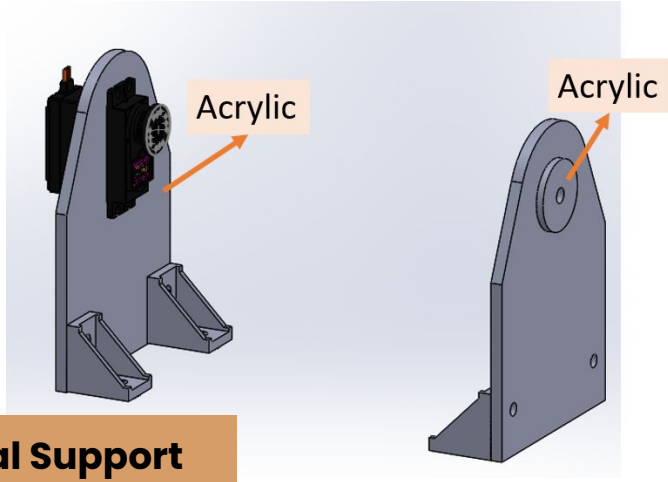
2. Profile Bar



3. Docking



4. Rotational Support



Choice of Motor and Bearing

MG996R

$$Torque = Force \times Distance$$

$$Force = 0.350\,kg \times 9.81\,m/s^2$$

$$= 3.43\,N$$

Add a safety factor of 2

$$Force = 6.86\,N$$

Torque

$$Torque = 6.86\,N \times 0.0215m$$

$$= 0.14Nm$$

Comparing to the maximum torque: 11kg-cm at 6V, the servo will be able to provide the required torque.

Specifications

626-ZZ

Attribute	Value
Inside Diameter	6mm
Outside Diameter	19mm
Ball Bearing Type	Miniature
Race Width	6mm
End Type	Shielded
Number of Rows	1
Static Load Rating	896N
Material	Steel
Race Type	Plain
Dynamic Load Rating	2.336kN
Bore Type	Parallel

Static Load from the Can is 3.26N total, that is 1.63 on each side support. This is less than static load of bearing

Dispenser

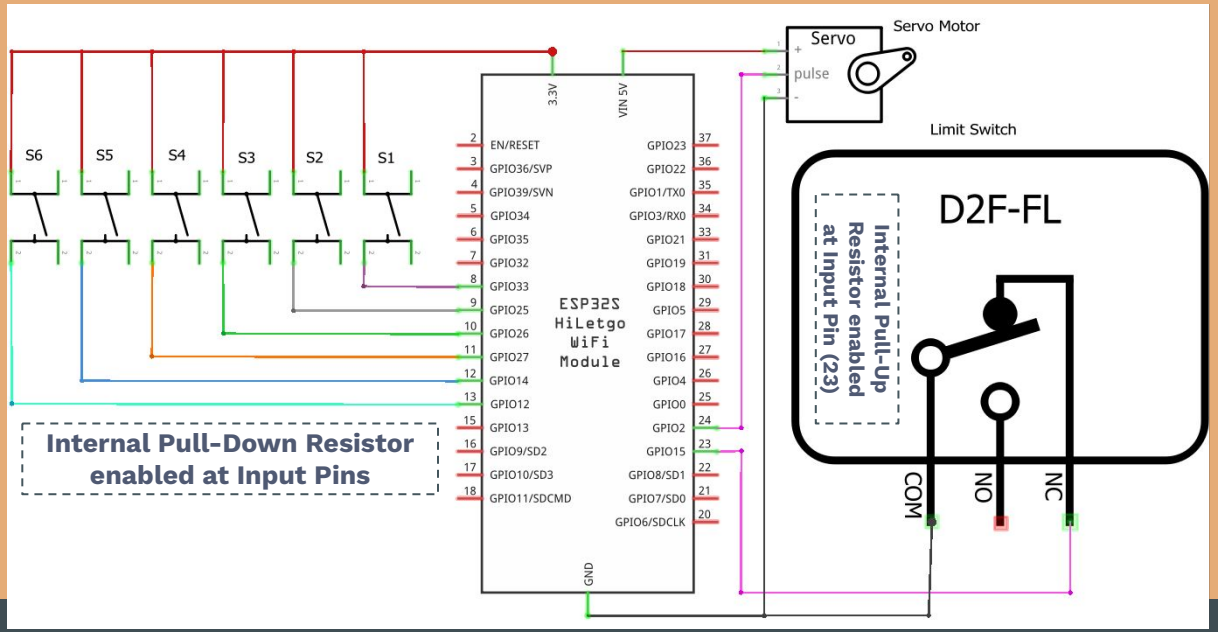
Choice of Materials and Assembly of Dispenser

Hardware Components in Dispenser

No.	Type of Component	Model Name	Quantity
1.	Microcontroller	AI-Thinker ESP32-S NodeMCU-32S	1
2.	Servo Motor	Mg996-R	1
3.	Limit Switch	Omron SS-5GL	1
4.	Push buttons	-	6
5.	Ball Bearings	626 ZZ	5
6.	Profile Bar Fasteners	20x20 L brackets & T slot nuts	1 set
7.	Angle Brackets	20x20 Angle Brackets	1 set
8.	Profile Bars	20x20 Profile Bars	2 meters

Electrical Architecture of Dispenser

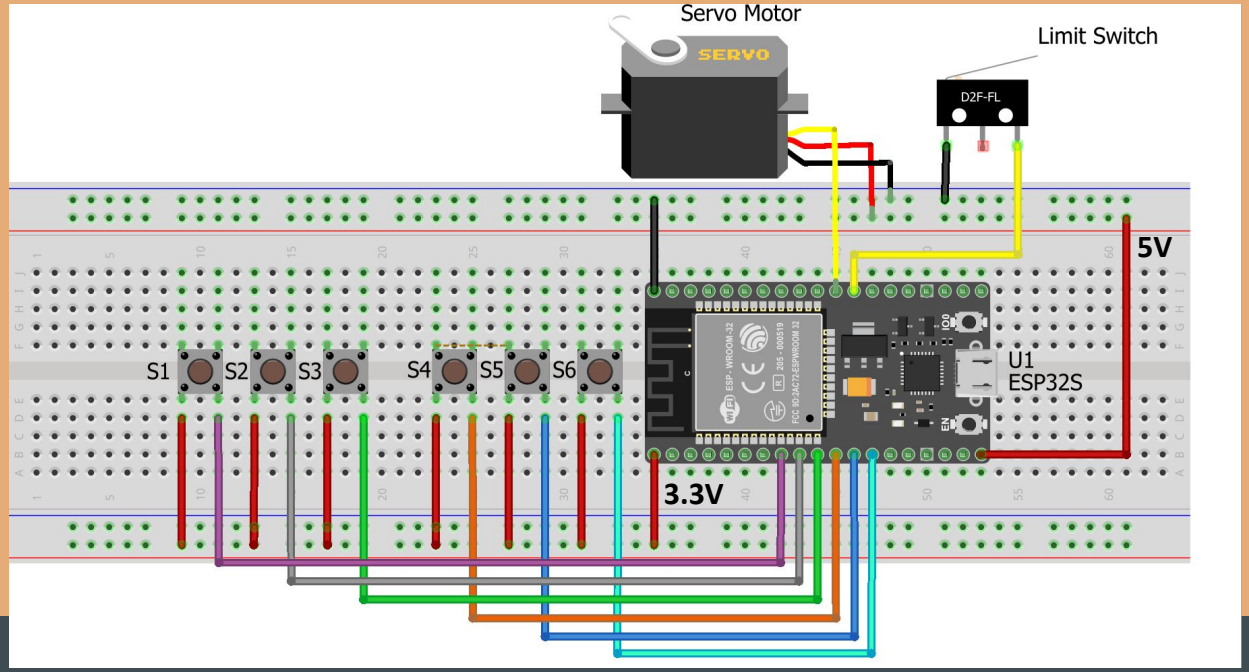
Schematics Diagram



Step #2: Dispenser receives user input

Electrical Architecture of Dispenser

Breadboard Connection



Step #2: Dispenser receives user input

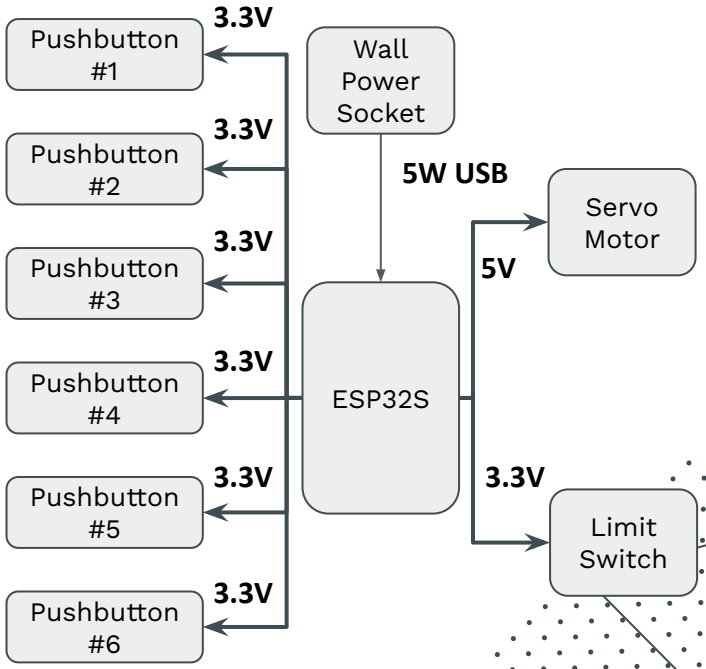
Power Calculation for Dispenser

We will be using the 5W USB adaptor with a USB to MicroUSB cable to power the ESP32 Microcontroller on our dispenser, as it will be stationary throughout the mission.

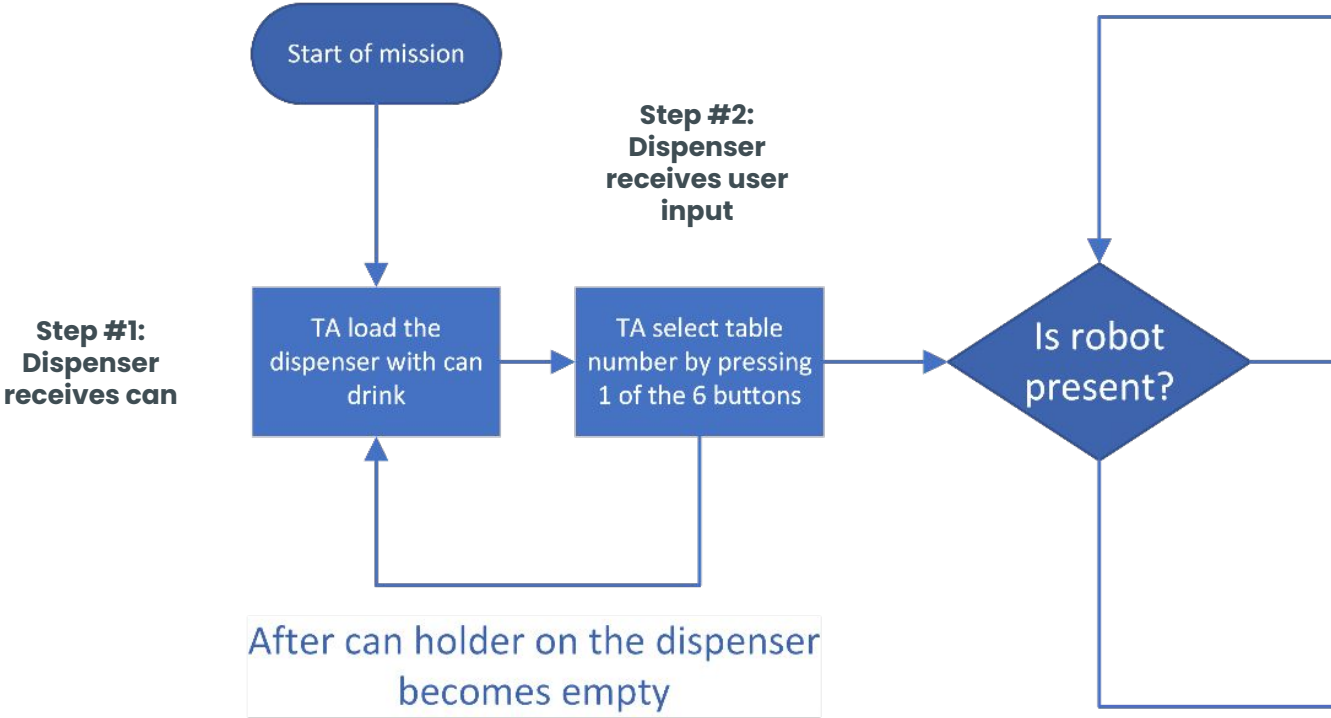
Power adaptor will output 5V voltage to the MicroUSB port of the ESP32.

Dispenser (power by microUSB)				
Components	Quantity	Current /A	Voltage /V	Power /W
ESP32S	1	0.5	5	2.5
Servo Motor (typical)	1	0.22	5	1.1
Servo Motor (surge)	1	0.65	5	3.25
Push buttons (with 10k ohm resistor*)	6	0.00033	3.3	0.006534
Limit Switch (with 10k ohm resistor*)	1	0.00033	3.3	0.001089

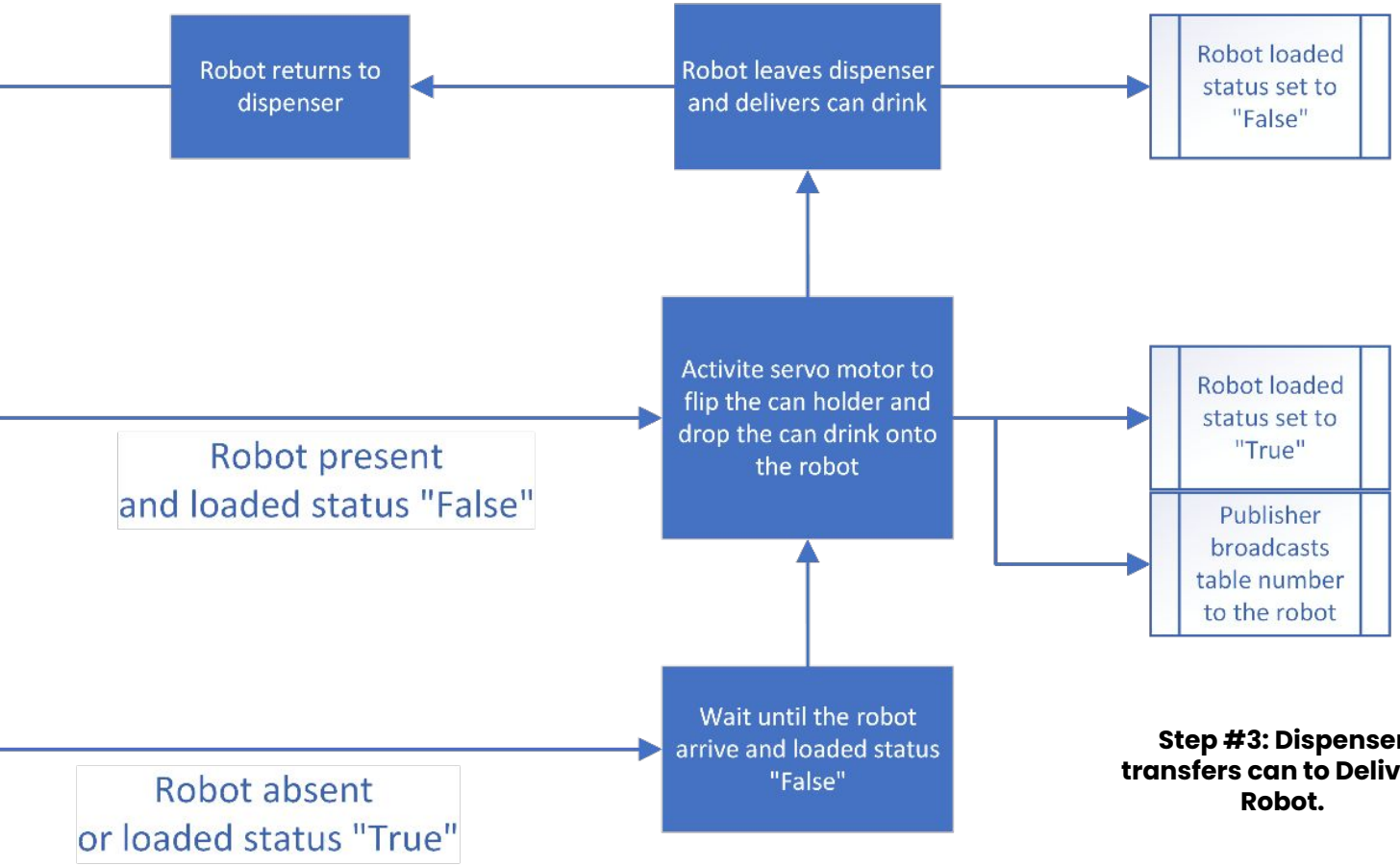
*The current for the push button and limit switch are calculated with a 10k Ohm resistor for reference, while we will be using the internal pull-up resistor whose value is unknown to us. However the difference will be negligible.



Program Flow for Dispenser



Program Flow for Dispenser



Step #3: Dispenser transfers can to Delivery Robot.

Communication (Dispenser ⇄ Robot)

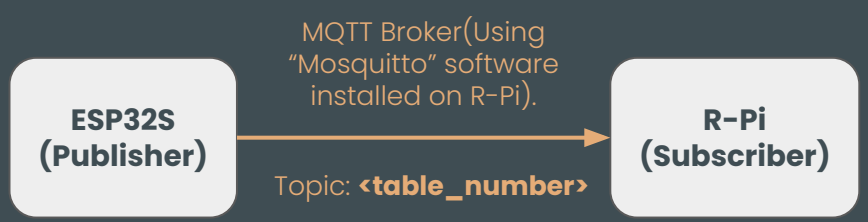
What are we using?

ESP32S:

- Communicate with RPi
  - Take in User Input (Table No. Selection)
  - Send that data to RPi on Robot (Navigates to that Table)

How are we doing it?

MQTT Protocol



1. WiFi Setup [ESP32S]

Key in same WiFI SSID & Password used for R-Pi through Arduino Code.

Communication (Dispenser ⇄ Robot)

How are we doing it?

MQTT Protocol [ESP32S (Publisher)]

4. MQTT Library → `#include <PubSubClient.h>` (enables it as a **publisher**)
5. Check if connected to broker.
6. Send input data as MQTT messages to topic **<table_number>**.

MQTT Protocol [R-Pi - Broker (Subscriber)]

1. Install Broker Software: **"Mosquitto"**
2. Set Username & Password for Publishers & Subscribers to get access to broker.
3. Start up MQTT broker (to receive data)
7. Read table no. data on topic.

Software

Why are we using them?

[Communication Protocol]

1.

ESP32S

- Wireless Communication (WiFi Module)
- Includes WiFi module & µC into 1 component → **Cheaper**
- Dual Core Processor → **Faster**
- Have knowledge about this model variation

2.

MQTT

- Familiar with Publish & Subscribe Protocol.
- More straightforward.
- Does not require large data transfer.

Communication between Dispenser & Delivery Robot

Example of what other groups wrote:

Software

What are we doing? ESP32 to

1. handle input from user,
2. turn servo motor
3. communicate with Turtlebot's RPi

How are we doing it?

MicroROS installed onto ESP32

1. PubSub model to communicate
2. rcl library to control servo motor and handle input

Software (Communication Topic)

MicroROS installed on ESP32 connected to same network -> Publish to topic and subscribe using micro-ros-agent

ESP32 Publish to <table_number>, to publish input of User

Turtlebot Subscribe to <table_number>, If isNull(nextTable), add next tableNumber to queue

ESP32 Subscribe to <ready_dispense>, use as condition for when to dispense

Turtlebot Publish to <ready_dispense>, True when NFC Detected publish, tableNumber received

Once <ready_dispense> is true, servo will move.

Software (Why are we doing this?)

MicroROS would be well integrated into the ROS topic pubsub model.

https://medium.com/@SameerT009/connect-esp32-to-ros2-foxy-5f06e0cc64df

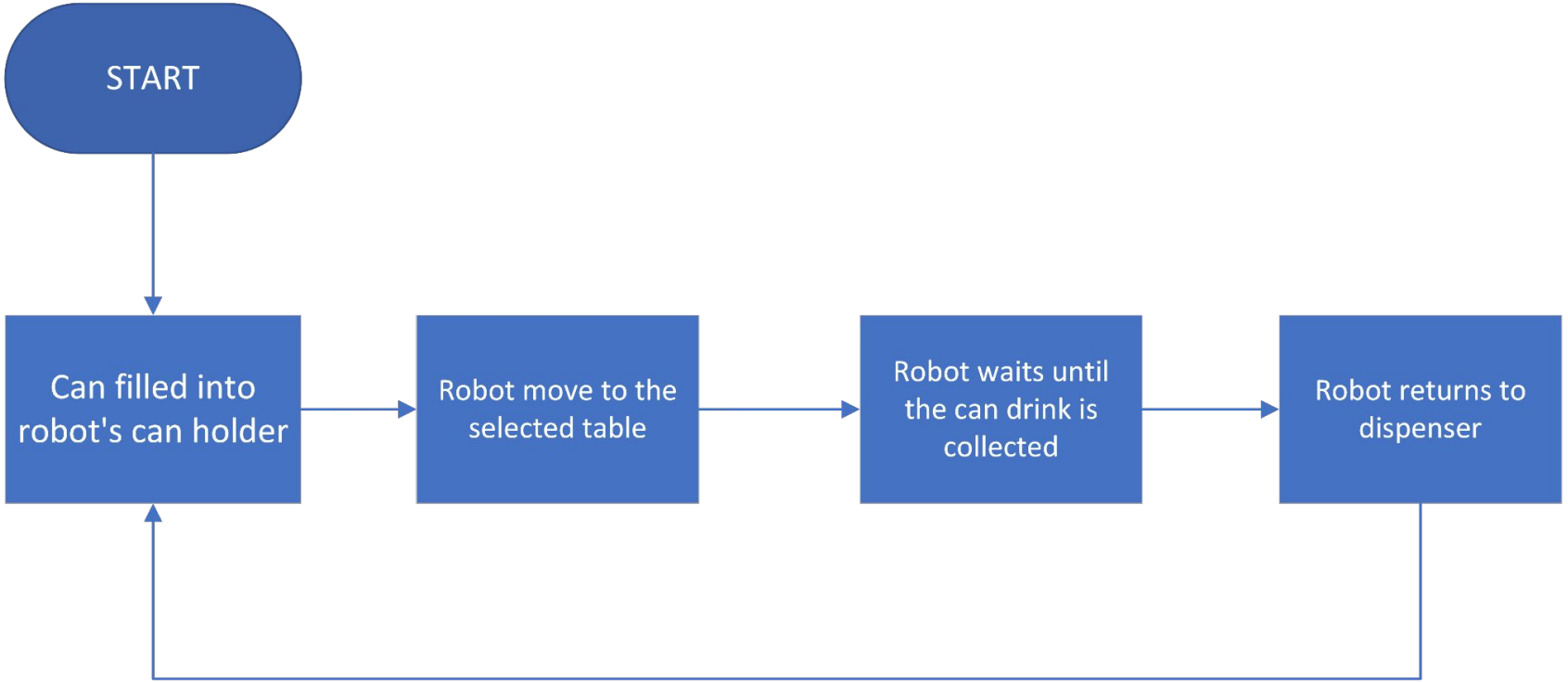
We considered MQTT to send and receive data but we would have to somehow integrate the MQTT messages into ROS topics which could be a problem.

We considered setting up a server (Flask) on the Turtlebot but we would also have to integrate the messages into ROS topics.

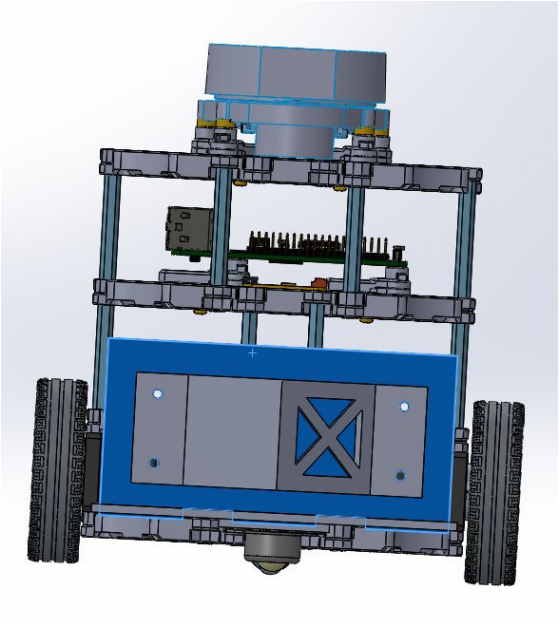
Delivery Robot

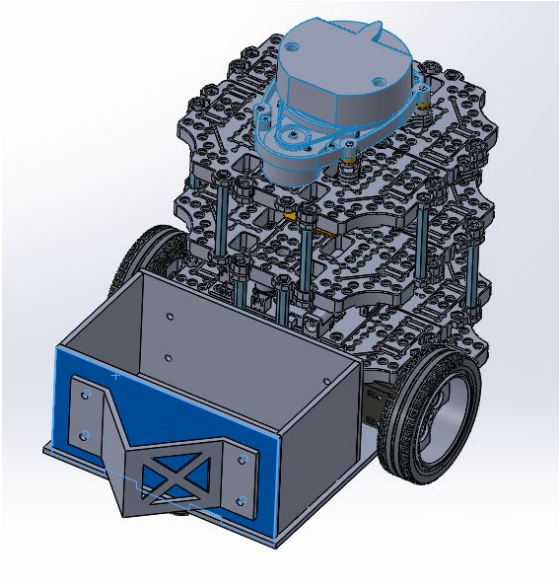
THE NIGHTMARE

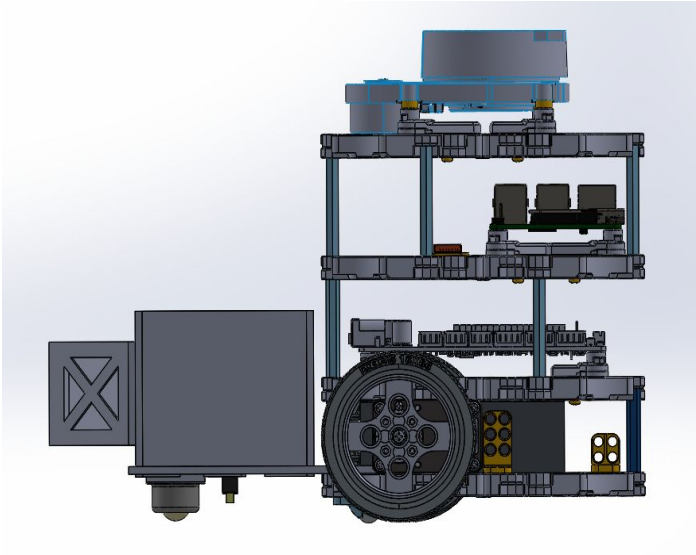
Program Flow for Delivery Robot



CAD Drawing of Delivery Robot (Functionality; Manufacturing Process; Machine Elements)







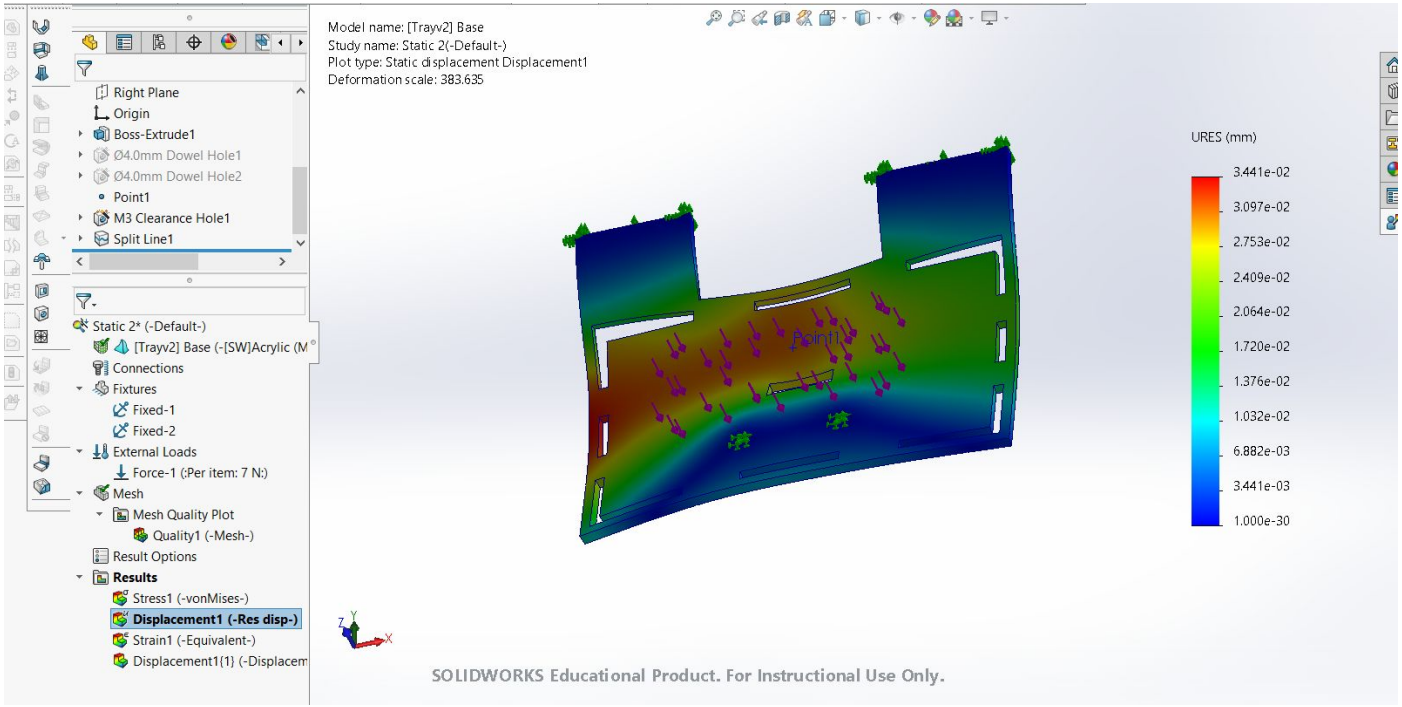
Step #3: Dispenser transfers can to Delivery Robot.

Calculations of Delivery Robot

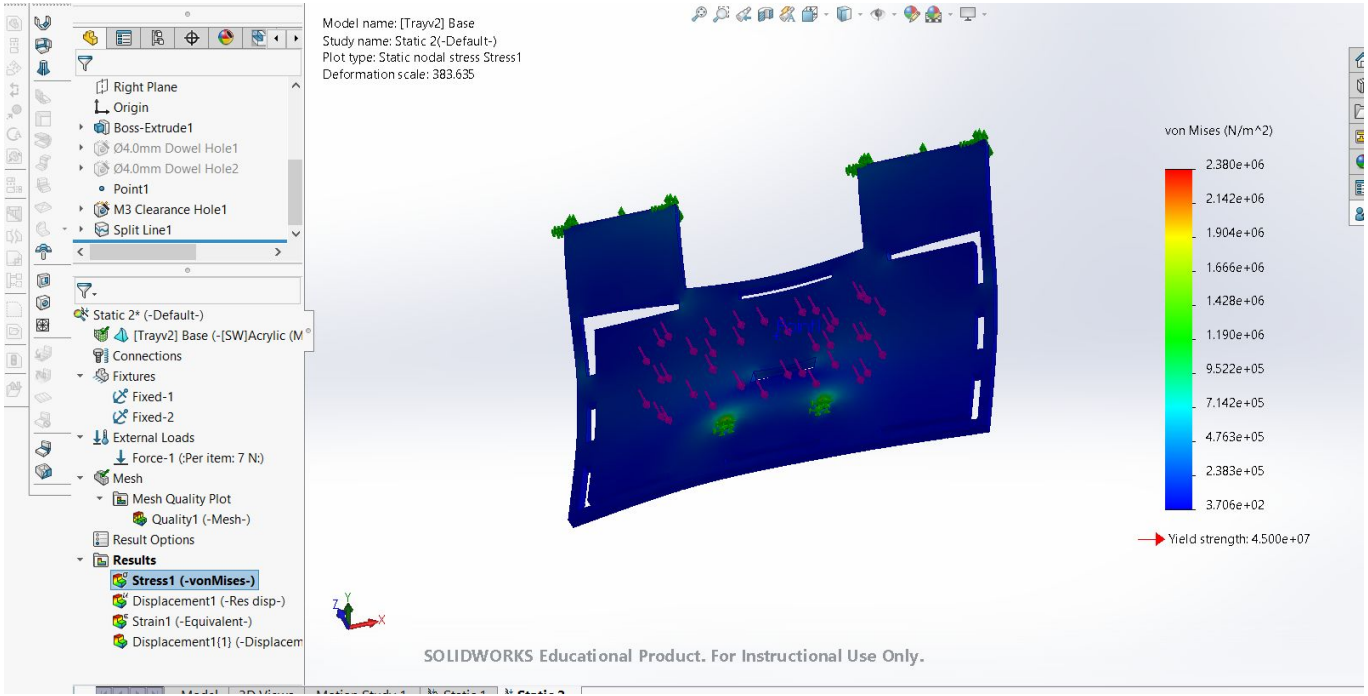
- Mass of can is **350**g.
- Ball caster mounted at the **end** of can holder, Ball caster able to sustain **10kg** load according to datasheet, thus the tray assembly is able to sustain the load of the can.
- Limit Switch Omron SS-5GL requires **0.49 N (50gf)** Maximum Operating Force to be engaged < 350g of can drink.

Step #3: Dispenser transfers can to Delivery Robot.

FEA


Model name: [Trayv2] Base
Study name: Static 2(-Default-)
Plot type: Static displacement Displacement1
Deformation scale: 383.635
Right Plane
Origin
Boss-Extrude1
Ø4.0mm Dowel Hole1
Ø4.0mm Dowel Hole2
Point1
M3 Clearance Hole1
Split Line1
Static 2* (-Default-)
[Trayv2] Base (-[SW]Acrylic (M
Connections
Fixtures
Fixed-1
Fixed-2
External Loads
Force-1 (:Per item: 7 N:)
Mesh
Mesh Quality Plot
Quality1 (-Mesh-)
Result Options
Results
Stress1 (-vonMises-)
Displacement1 (-Res disp-)
Strain1 (-Equivalent-)
Displacement1{1} (-Displacem
URES (mm)
3.441e-02
3.097e-02
2.753e-02
2.409e-02
2.064e-02
1.720e-02
1.376e-02
1.032e-02
6.882e-03
3.441e-03
1.000e-30
SOLIDWORKS Educational Product. For Instructional Use Only.

FEA

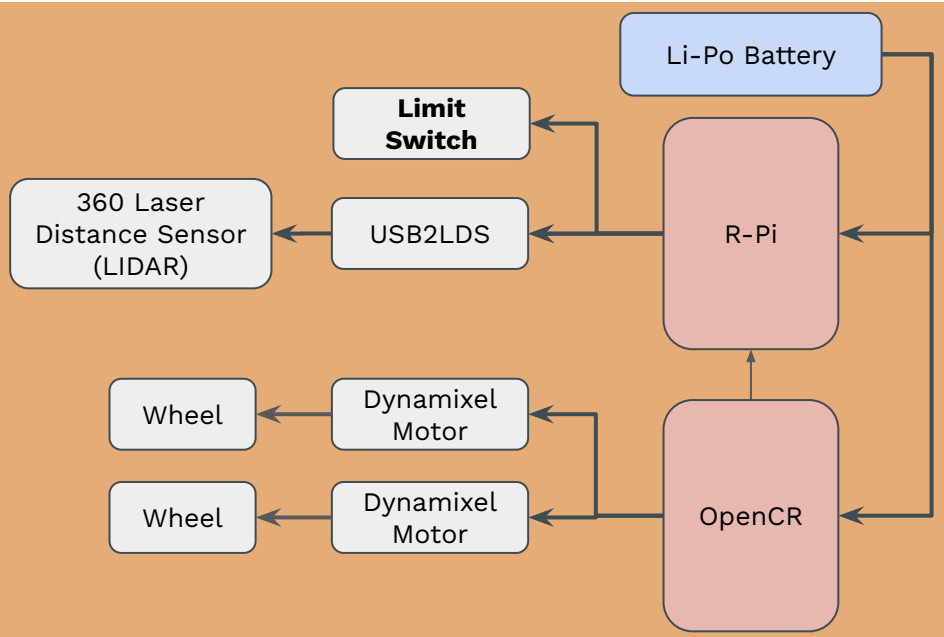

Model name: [Trayv2] Base
Study name: Static 2(-Default-)
Plot type: Static nodal stress Stress1
Deformation scale: 383.635
Right Plane
Origin
Boss-Extrude1
Ø4.0mm Dowel Hole1
Ø4.0mm Dowel Hole2
Point1
M3 Clearance Hole1
Split Line1
Static 2* (-Default-)
[Trayv2] Base (-[SW]Acrylic (M
Connections
Fixtures
Fixed-1
Fixed-2
External Loads
Force-1 (:Per item: 7 N:)
Mesh
Mesh Quality Plot
Quality1 (-Mesh-)
Result Options
Results
Stress1 (-vonMises-)
Displacement1 (-Res disp-)
Strain1 (-Equivalent-)
Displacement1{1} (-Displacem
von Mises (N/m^2)
2.380e+06
2.142e+06
1.904e+06
1.666e+06
1.428e+06
1.190e+06
9.522e+05
7.142e+05
4.763e+05
2.383e+05
3.706e+02
Yield strength: 4.500e+07
SOLIDWORKS Educational Product. For Instructional Use Only.

Hardware Components Added to Delivery Robot

No.	Type of Component	Model Name	Quantity
1.	Ball Caster	RBT-02194	1
2.	Limit Switch	Omron SS-5GL	1

Electrical Architecture of Robot

Functional Block Diagram



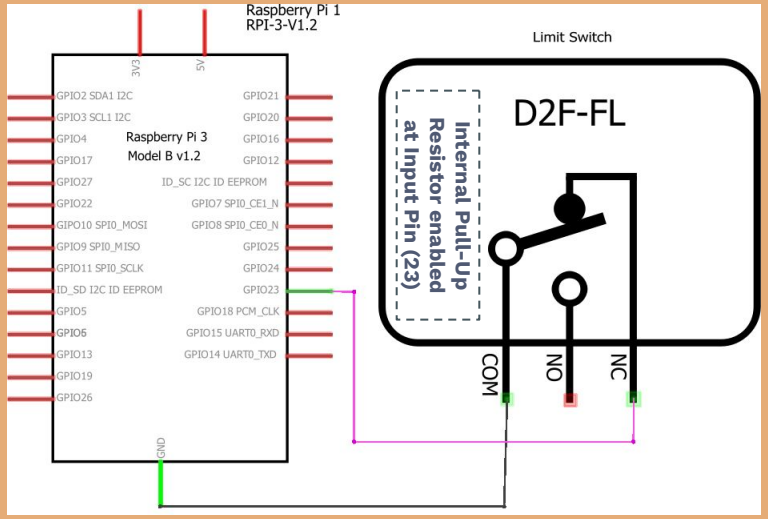
Step #3: Dispenser transfers can to Delivery Robot.

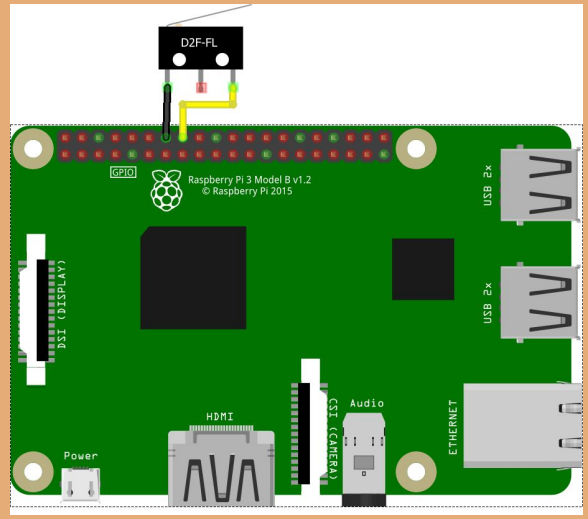
Electrical Architecture of Robot

Breadboard Connection

&

Schematics Diagram





Step #3: Dispenser transfers can to Delivery Robot.

Power Budgeting Table for Delivery Robot

Components	Quantity	Current /A	Voltage /V	Power /W	Runtime /s	Energy Required /Wh
Turtlebot (bootup)	1	0.7	11.2	7.84	180	0.392
Turtlebot (standby)	1	0.59	11.2	6.608	600	1.101333333
Turtlebot (during operation)	1	0.72	11.2	8.064	1500	3.36
Limit Switch (with 10k ohm resistor*)	1	0.00033	3.3	0.001089	1500	0.00045375

*For reference

$$Total\ Power\ Required \times Operation\ Time = Total\ Energy\ Consumed$$

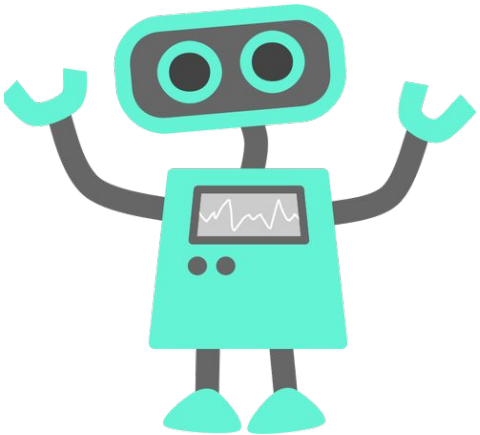
$$Energy\ required:$$
$$(7.84W \times 180s) + (6.608W \times 600s) + (8.064W \times 1500s) + (0.001089W \times 1500s)$$
$$= 4.854Wh$$

$$Energy\ provided\ by\ the\ battery:$$
$$11.1V \times 1800mAh = 19980mWh = 19.98Wh$$

Considering safety factor of 60%:

$$4.854Wh \times 1.6 = 7.7664Wh < 19.98Wh$$

After considering the safety factor, the energy required is still lower than the amount that the battery can supply, indicating that the battery is more than enough to power the turtlebot for this mission.



Software

What are we doing?

Step #3: Dispenser transfers can to robot.

Step #4: Robot navigates through the restaurant to the Table.

Step #5: Robot waits at Table for can to be picked up.

Step #6: Robot returns to Dispenser for next order.

Preparation

- Detect Can w/Limit Switch
- Receive Table Selection

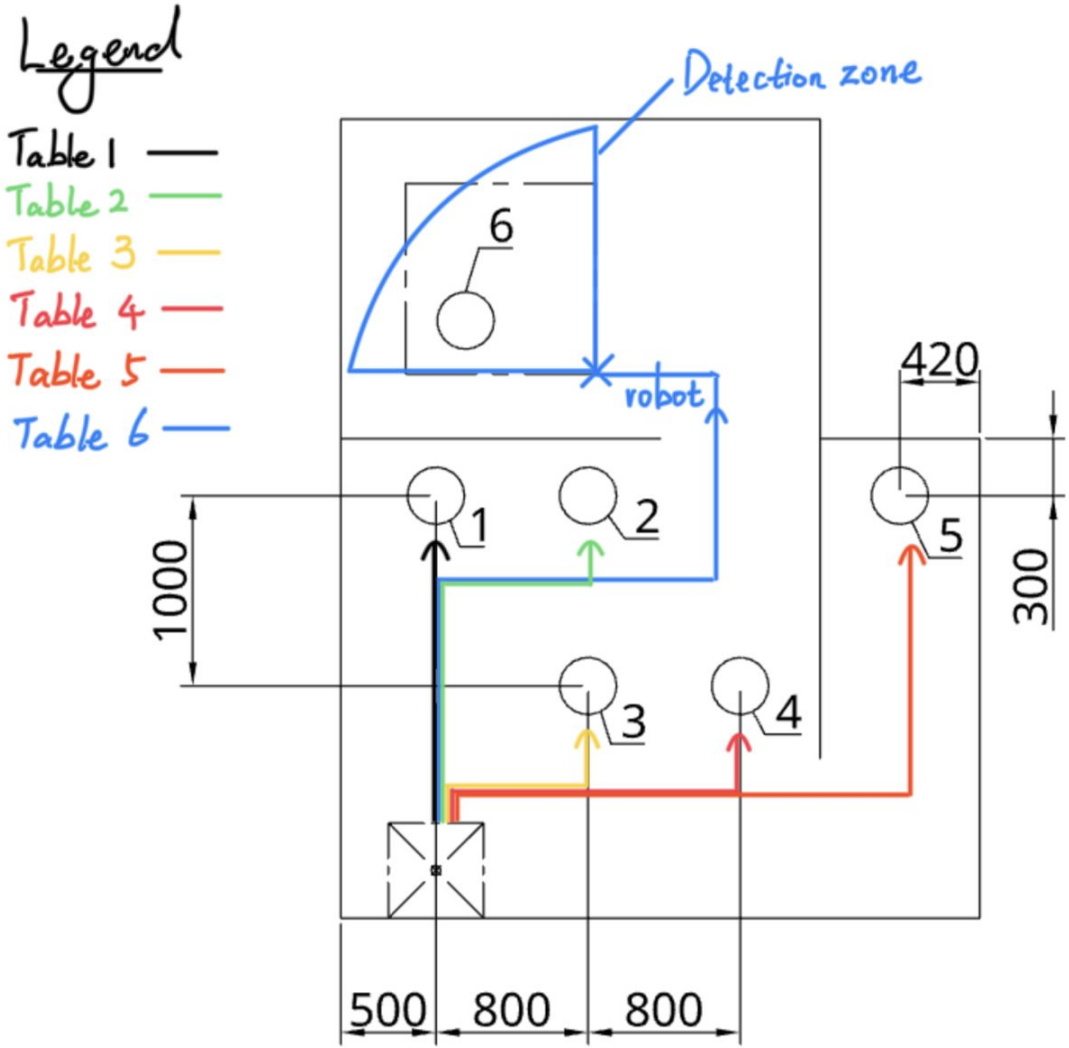
Navigation to Table

- Run pre-set script
- Stop within 15 cm before the Table

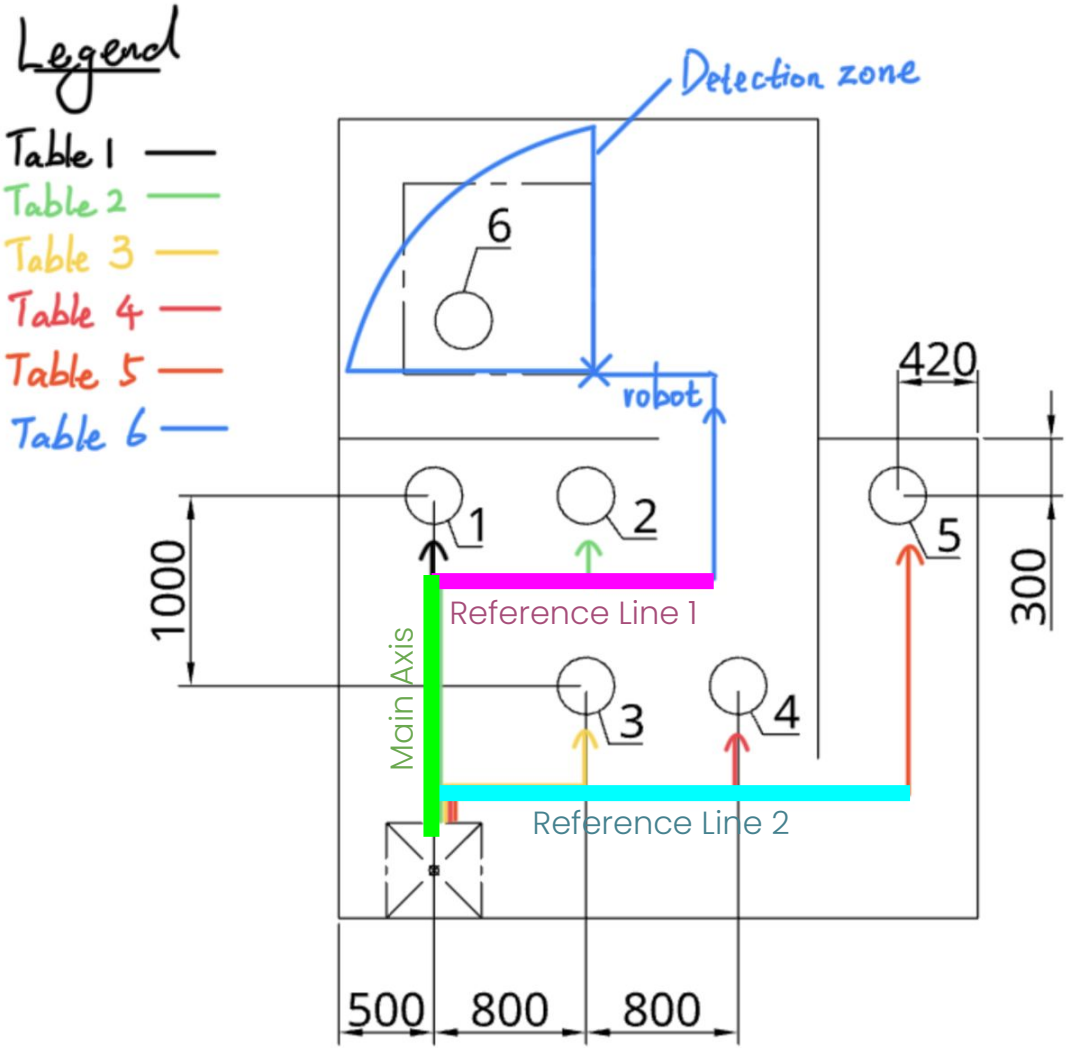
Return to Dispenser

- Run pre-set script
- Docking

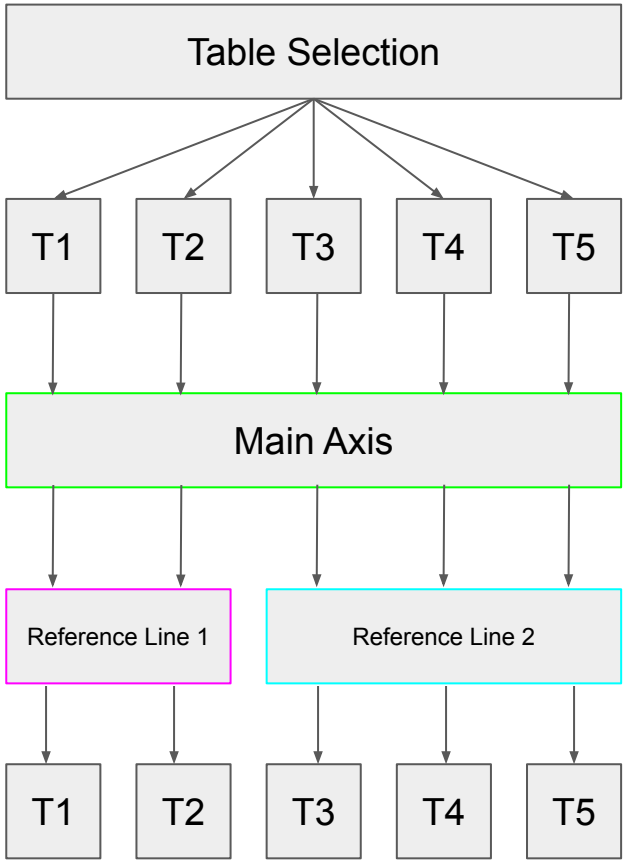
How are we doing it?

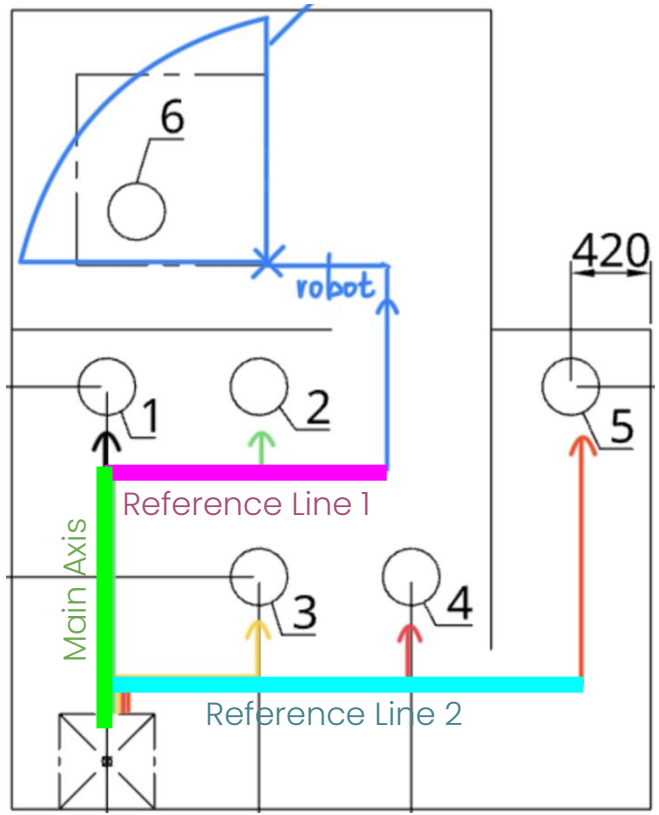


How are we doing it?



How are we doing it?

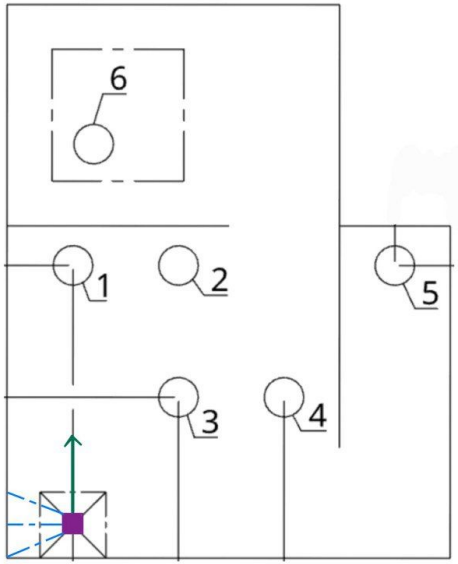




How are we doing it?

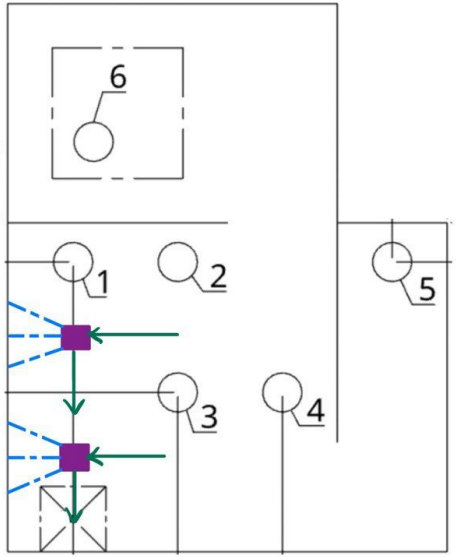
Calibration

To ensure script works



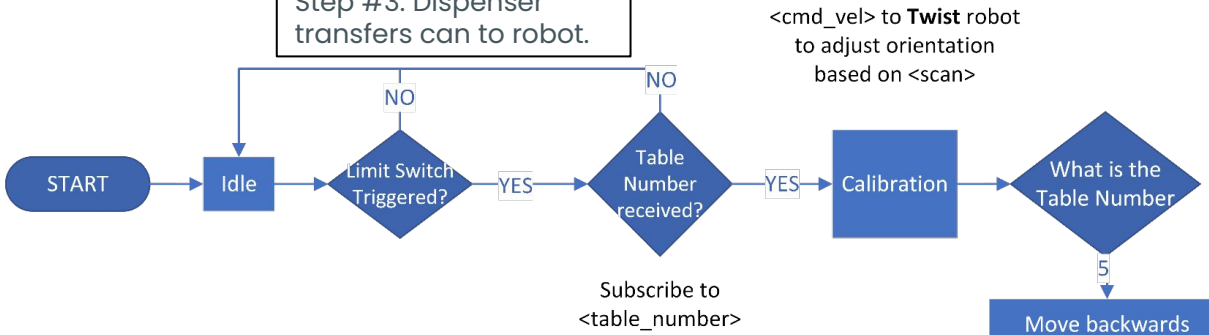
Before departing

Before docking

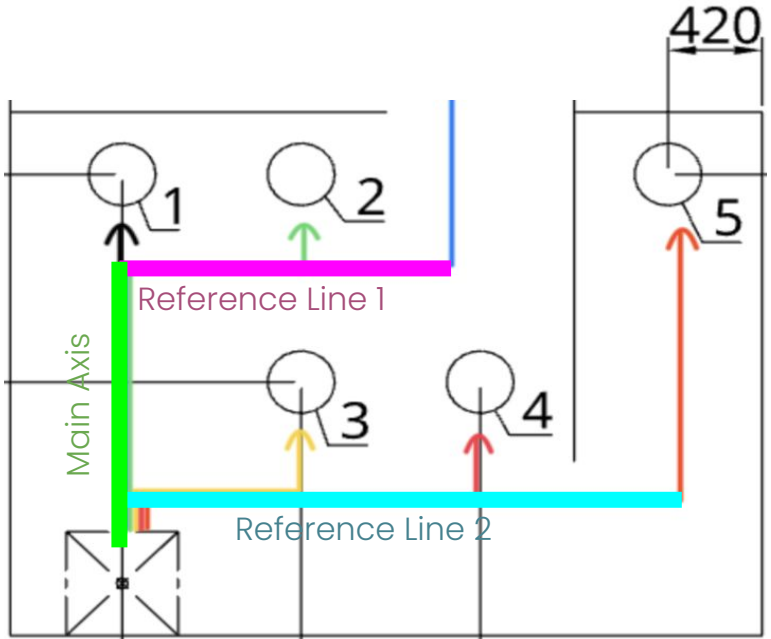


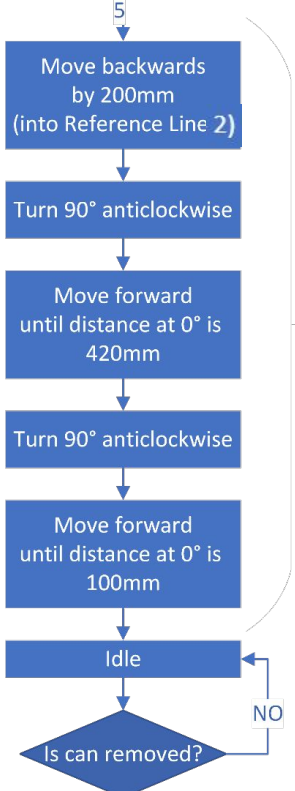
Program Flow for Table 5 (Departing Half)

Step #3: Dispenser transfers can to robot.



Step #4: Robot navigates through the restaurant to the Table.



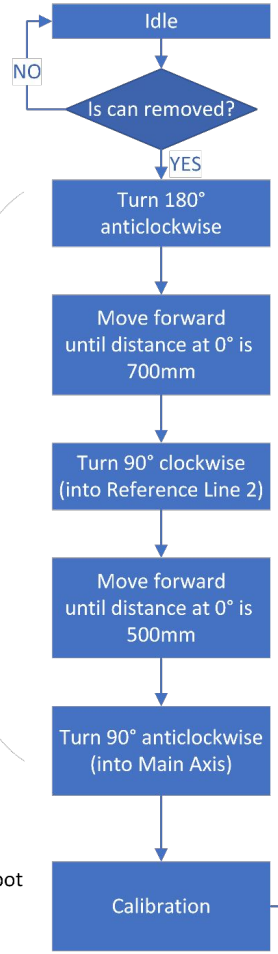


<cmd_vel> to **Twist** robot to change <u>linear velocity</u> and <u>angular velocity</u> based on parameters i.e. <scan>

Step #5: Robot waits at Table for can to be picked up.

Program Flow for Table 5 (Returning Half)

Step #5: Robot waits at Table for can to be picked up.



<cmd_vel> to **Twist** robot to change <u>linear velocity</u> and <u>angular velocity</u> based on parameters i.e. <scan>

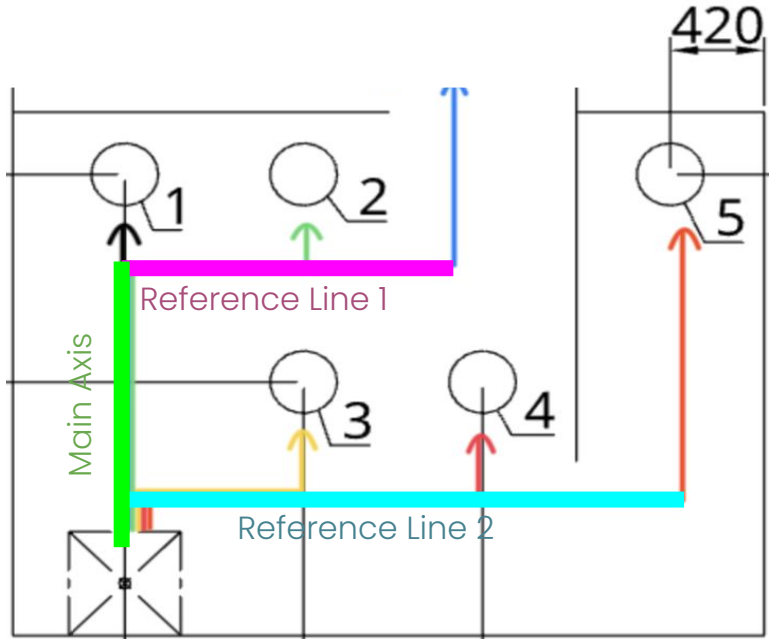
Step #6: Robot returns to Dispenser for next order.

<cmd_vel> to **Twist** robot to adjust orientation based on <scan>

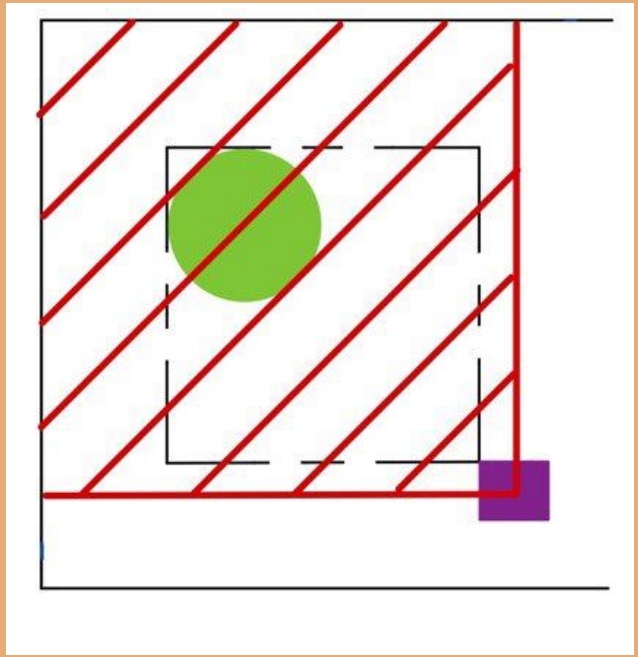
<cmd_vel> to **Twist** robot to change <u>linear velocity</u>

Subscribe to <scan>

Subscribe to <dispenser_limit_switch>



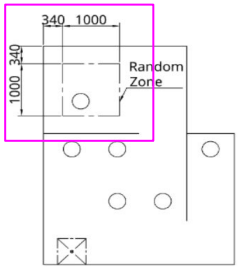
How are we handling Table 6?



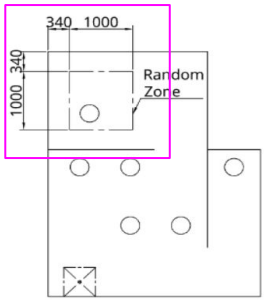
1. Arrive before bottom right corner of Random Zone

2. Scan for nearest object in top-right quadrant

3. Navigate towards it



Yin Heng



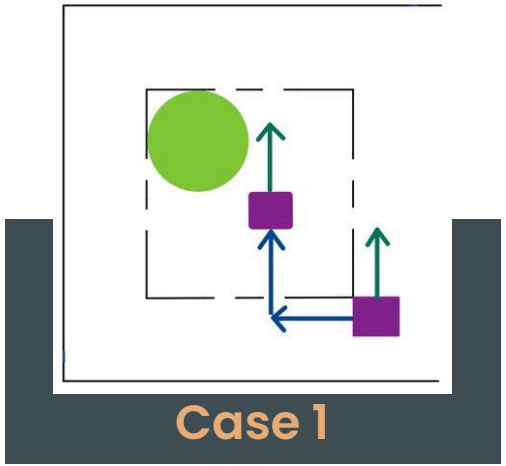
How are we Navigating to Table 6?

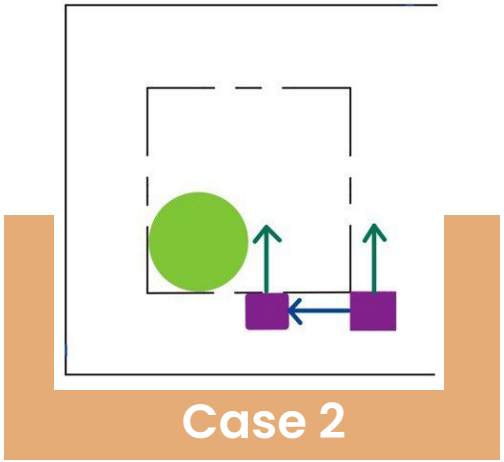
- Bot Orientation is always upright
- Right of bot is always clear

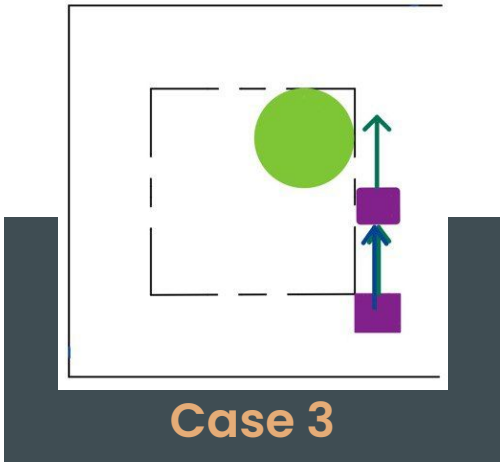
Legend

Robot Orientation (0°)

Movement

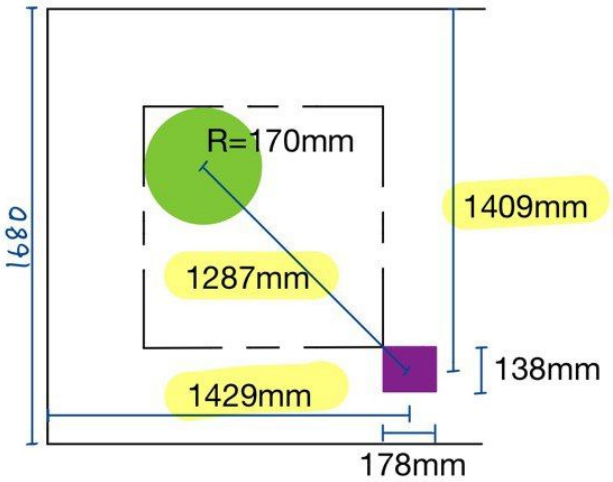


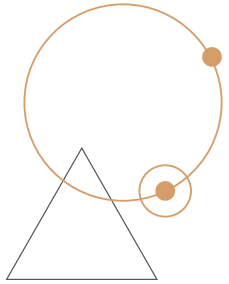


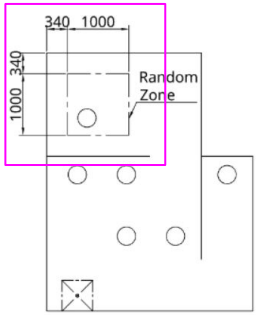


Can we mistake Wall for Table?

Calculations





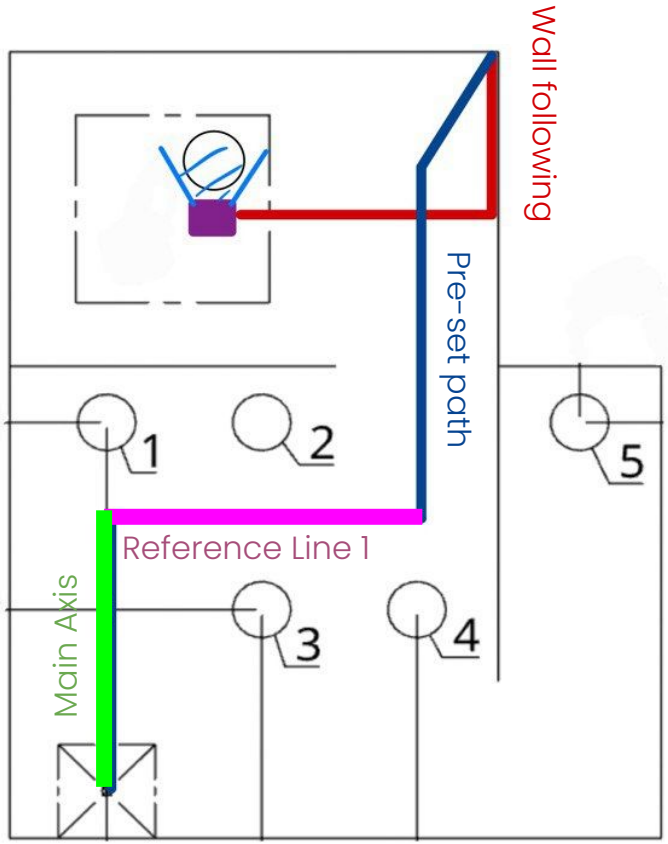


$$\sqrt{\left(\frac{138}{2}\right)^2+\left(\frac{178}{2}\right)^2}+\sqrt{2\left(1000-170\right)^2}$$

$\approx$ 1287mm (rounded up)

NO.

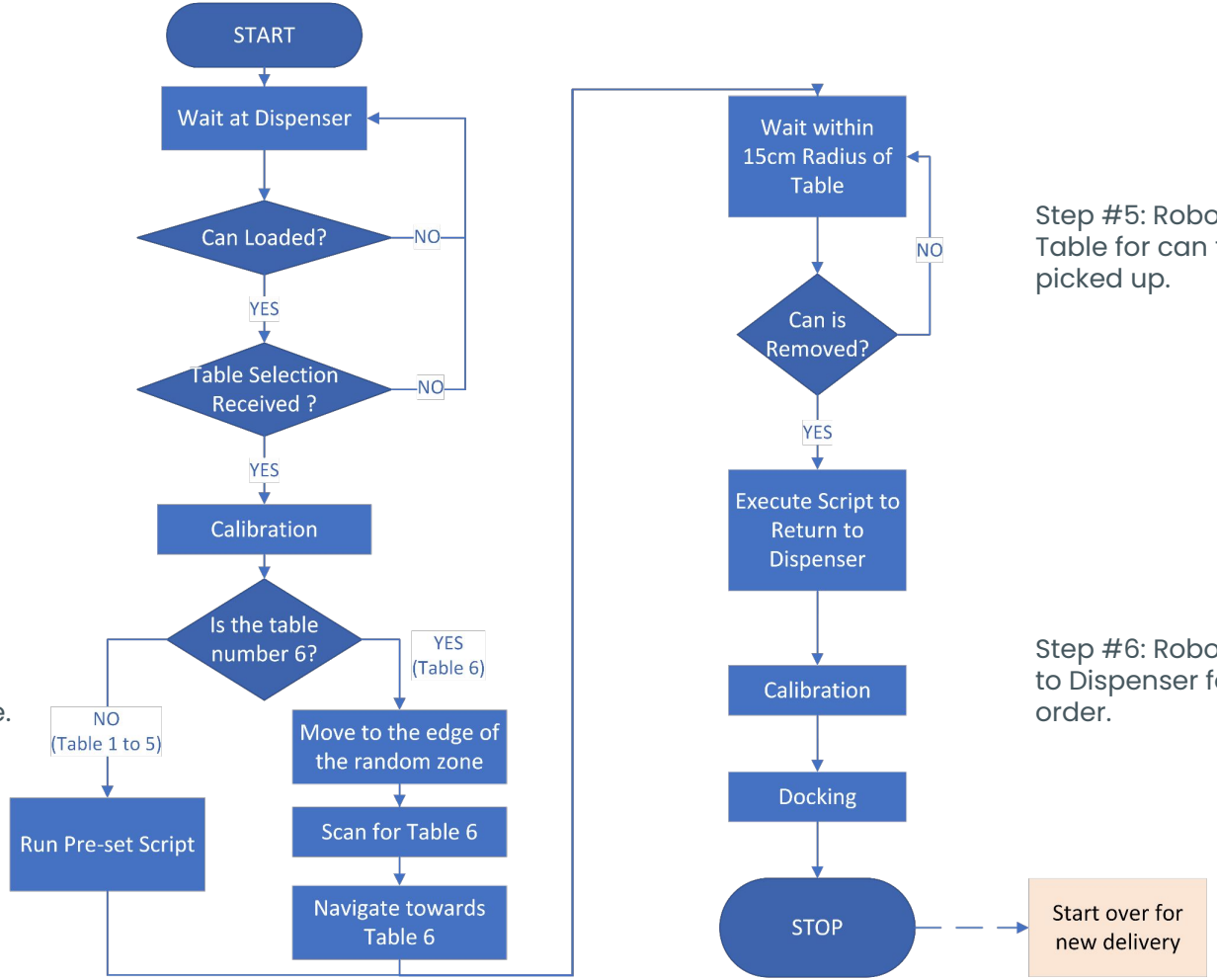
Navigating Back from Table 6



Program Flow for Delivery Robot

Step #3: Dispenser transfers can to robot.

Step #4: Robot navigates through the restaurant to the Table.



Step #5: Robot waits at Table for can to be picked up.

Step #6: Robot returns to Dispenser for next order.

Bill of Materials

No	Type of Component	Model Name	Quantity	Price (Each) /S$	Price (Total) /S$
1	Microcontroller	NodeMCU ESP-32S	1	5.51 (Electronics Lab)	5.51
2	Servo Motor	Mg996-R	1	4.88 (Electronics Lab)	4.88
3	Limit Switch	Omron SS-5GL	2	1.5	3
4	Push buttons	-	6 (2 sets)	2.75 (for 4)	5.5
5	Ball Bearings	626 ZZ	3	0.96	2.88
6	Profile Bar Fasteners	20x20 L brackets & T slot nuts	2 sets	18	36
7	Angle Brackets	20x20 Angle Brackets	2 sets	2	4
8	Profile Bars	20x20 Profile Bars	2 meters	6.47 (per 400 mm)	32.85
9	Ball Caster	RBT-02194	1	1.2	1.2
				Total Price (S$):	95.82

Validation and Verification

- ESP32– communication
- Dispenser– servo
- Push button interfacing
- Limit Switch + Can

Plan for Valid

Level 1: individual Parts– making sure the component works

- Servo works
- Esp32 works
- Limit switch works etc

Level 2: Sub system– making sure the robot and dispenser work separately

Level 3: Integrating subsystems– trying the entire sequence – like the final run

Validation and Verification
Level 1 - Individual Parts

Push Button & Limit Switch activation

- Connect them to the respective boards (RPi/ESP32) and run a script to print out the input of the component, to see if the input "High" and "Low" appears as expected
- Test if the robot is able to activate the servo motor while docking.
- Test if the can in dispenser is able to trigger the limit switch

Servo Motor

- Set up the servo motor on the dispenser and run a script to test if it is able to rotate the can holder and drop the can drink

ESP32S

- Upload a simple existing test program (eg: Blink onboard LED) - Test if the program works
- Test if MQTT communication works with R-Pi (publish & subscribe simple messages)

Validation and Verification
Level 2 - Sub-systems

Delivery robot

- Test if the robot can navigate to all 6 tables respectively, and stops within 15cm of the tables
- Make sure the robot do not crash into obstacles during the delivery

Dispenser

- Flip the can holder on the dispenser to test that the drink can fall accurately from the dispenser onto the can holder on the robot

Docking

- Drive the robot into the dispenser to test if the robot can slide into the correct position with the help of the angled homing
- Check if the dispenser can withstand the robot's push remain at its original position

Validation and Verification
Level 3 - Integrating Sub-systems

Docking of Robot

- Test that the robot is able to come back to the dispenser and stop at the correct position for can refilling, from all 6 tables

Docking Detection

- Run a script to output a result as the robot docks at the dispenser, to test if the dispenser is able to detect it with the limit switch
- Test the stability while the robot is pushing the limit switch on the dispenser while docking.

Communication

- Run scripts to check if the dispenser can broadcast message to the robot correctly and timely

THANK YOU FOR NOT KICKING US OUT <3

What can go wrong?

Dispenser not detected after running script

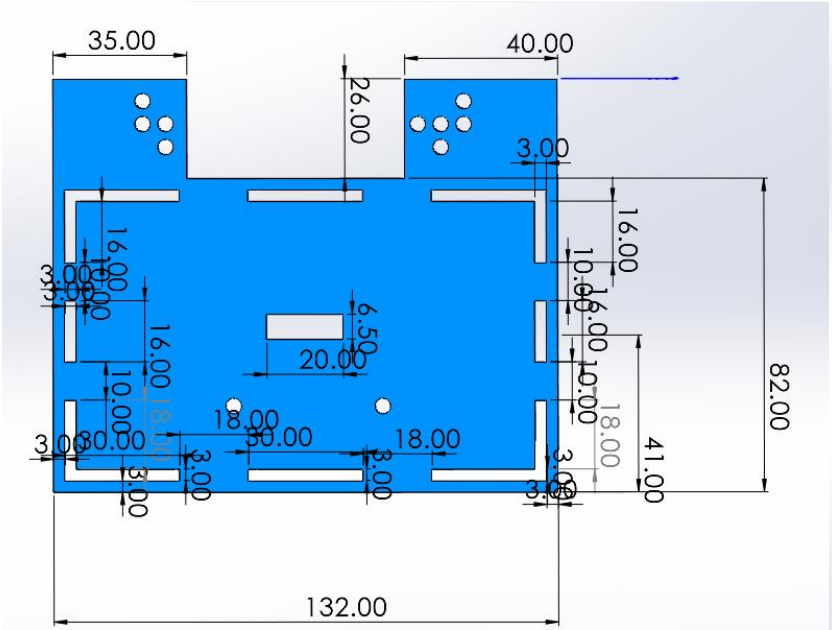
Table (1-5) not detected after running script

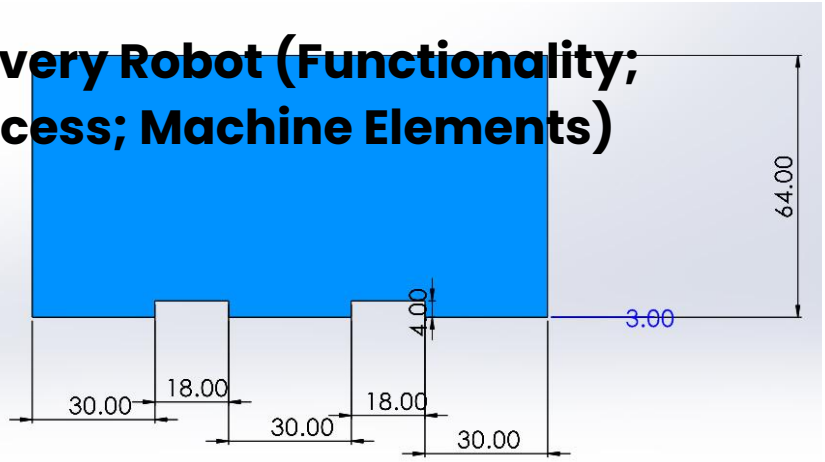
Table 6 not found

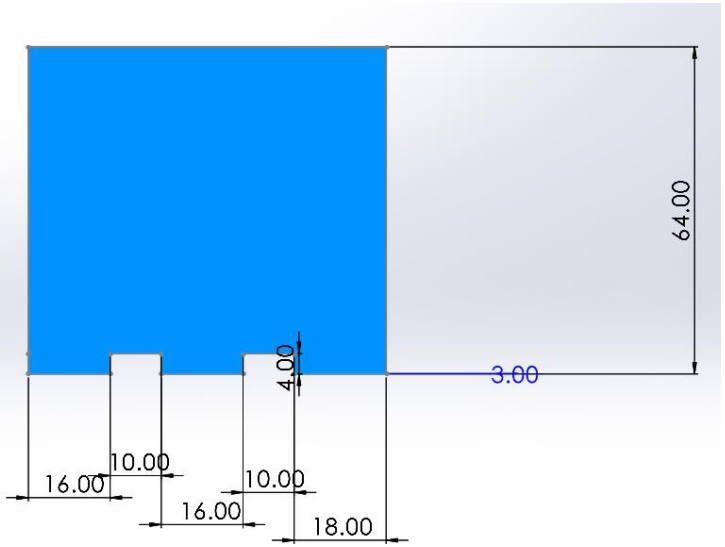
Backup Search Algorithm

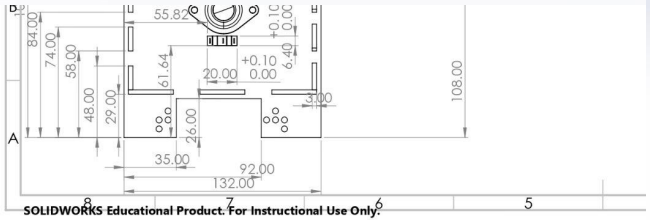
- Obstacle Avoidance

CAD Drawing of Delivery Robot (Functionality; Manufacturing Process; Machine Elements)

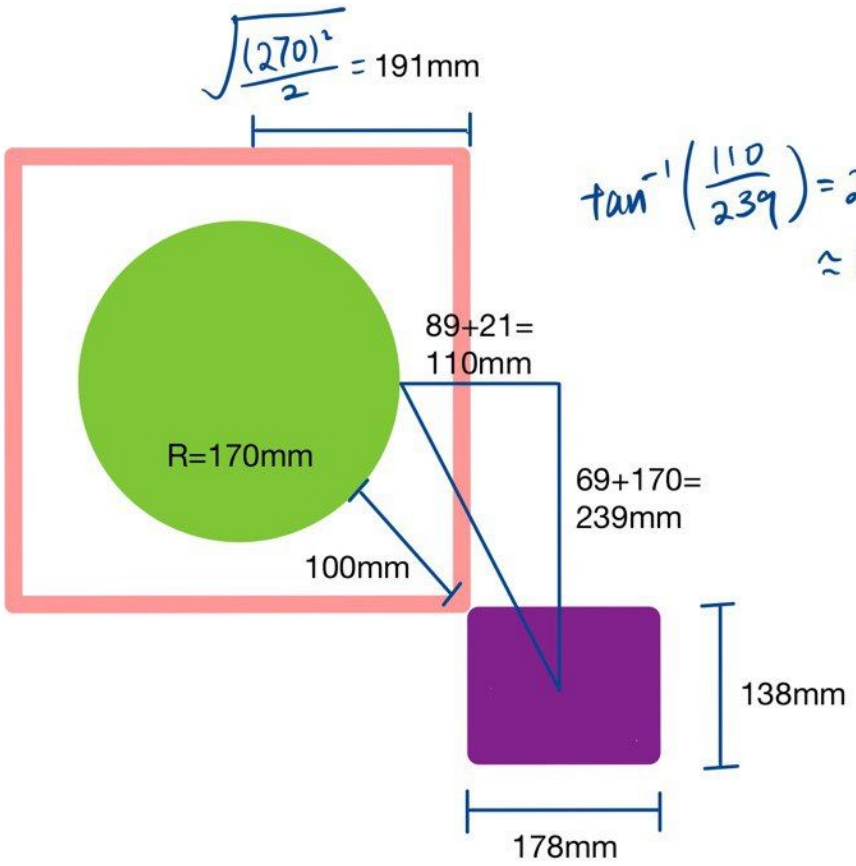


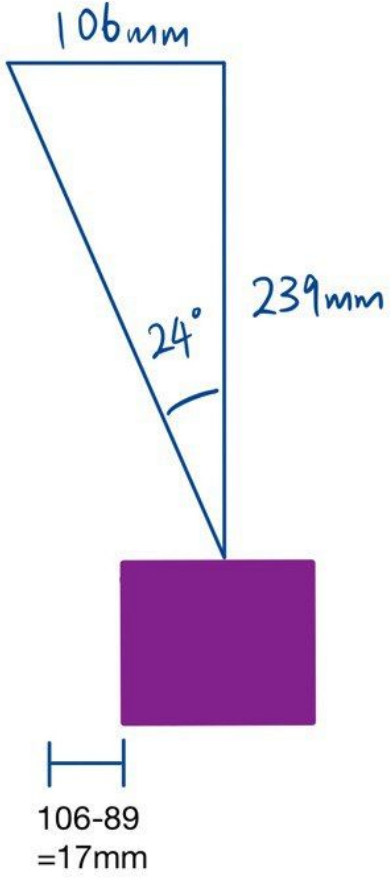




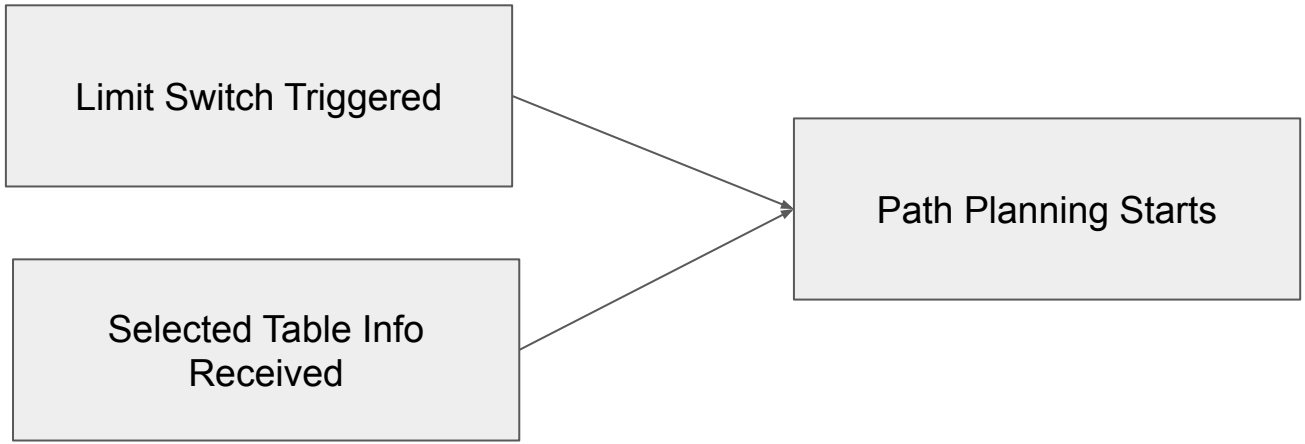


Why ±24°?


$\sqrt{\frac{(270)^2}{2}}$ = 191mm
$\tan^{-1}\left(\frac{110}{239}\right) = 24.71°$
$\simeq 24$
89+21=
110mm
R=170mm
69+170=
239mm
100mm
138mm
178mm


106mm
24°
239mm
106-89
=17mm

Program Flow for Delivery Robot



Step #3: Dispenser transfers can to Delivery Robot.

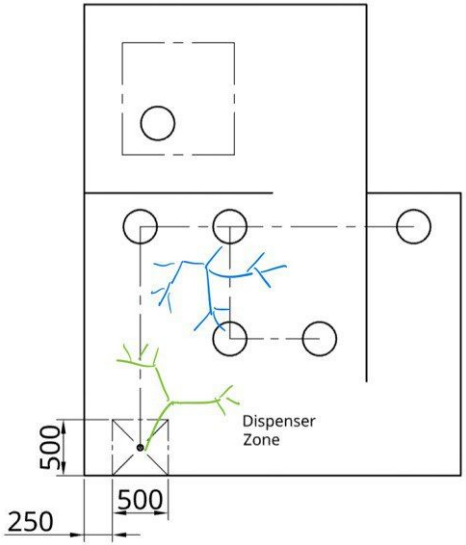
Program Flow for Delivery Robot

RRT-Connect Algorithm

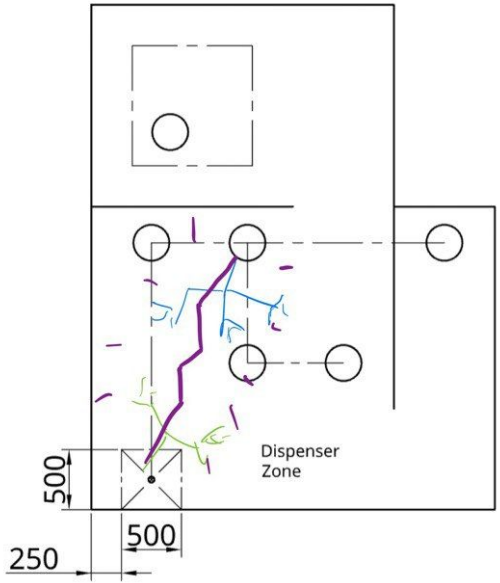
1. Table Selected

1	**2**
3	4
5	6

2. 2 Trees start growing

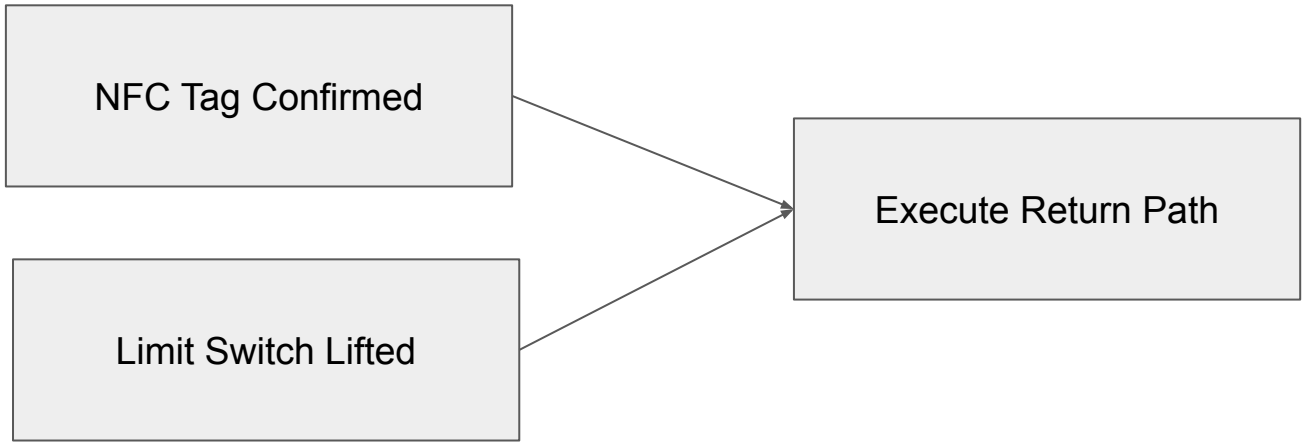


3. Working Path Established



Step #4: Delivery Robot navigates autonomously through the restaurant.

Program Flow for Delivery Robot



Step #5: Delivery Robot waits at table for can to be delivered successfully.

TABLE SIX AAAAAHHHHHH

1. Edge detection algorithm to identify edges in the map

2. Contour detection algorithm identify contours in the map

3. Use the Hough transform to detect circles in the map.

   (The Hough transform can detect circles of a specific radius)