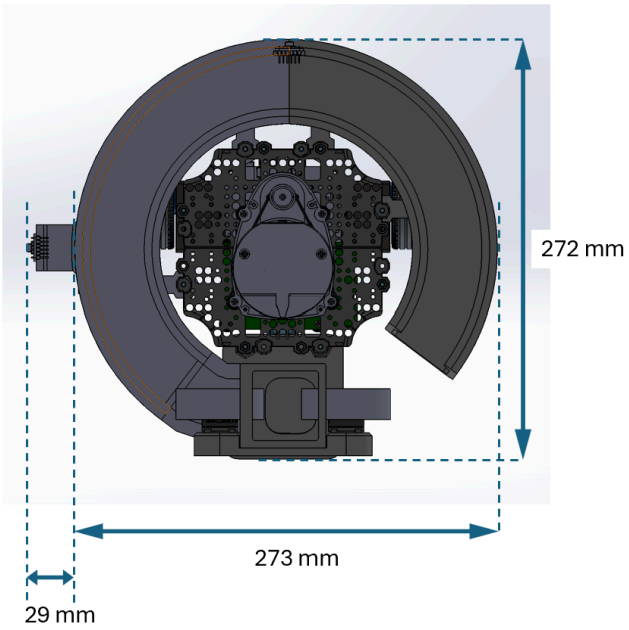
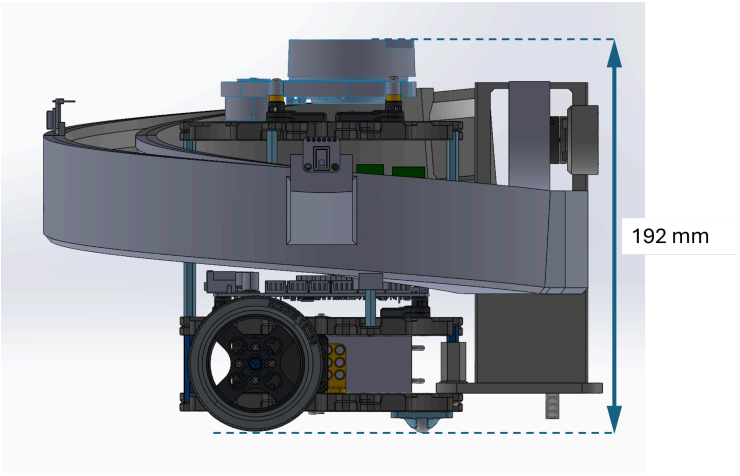
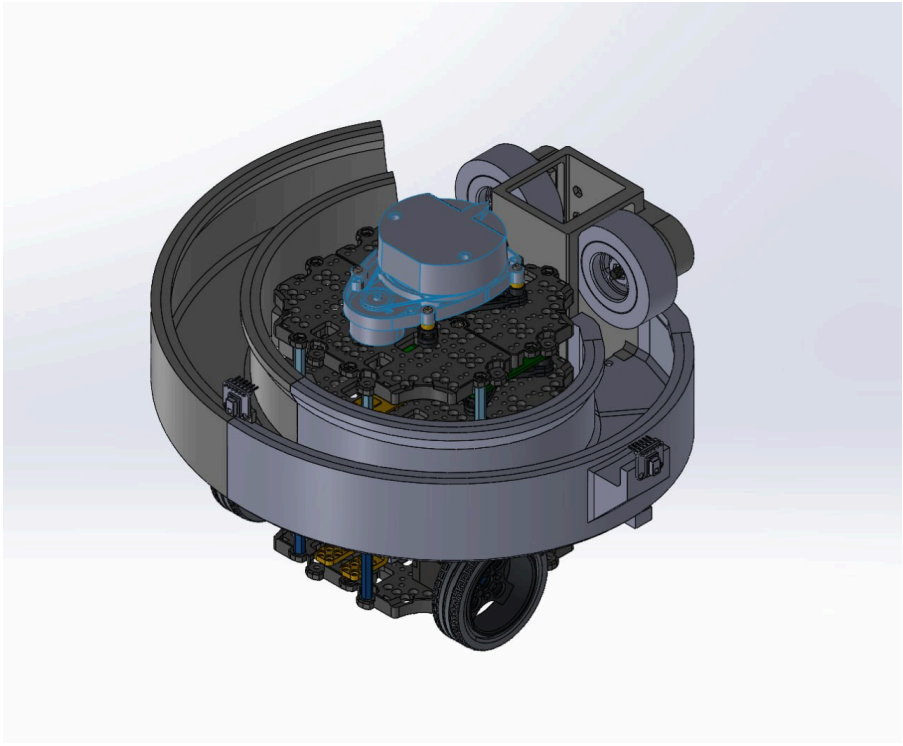
End User Documentation

Section 1 General System Description & Critical Data (Spec Sheet)

Model	TurtleBot 3
Dimensions	272 mm x 302 mm x 192 mm (L x B x H)  
Battery Capacity/ Operating Life	11.1 V, 1800 mAh (1.8 Ah) LiPo 80 minutes normal, minimum 26 minutes under heavy load

Description	An autonomous mobile robot that navigates a maze to detect heat sources using dual AMG8833 thermal sensors. Once a source is detected, the robot approaches, stops, and launches a ping pong ball using a dual flywheel launcher system.  Isometric view of the bot

Operation	1. Navigates autonomously with Nav2 stack (path planning & obstacle avoidance). 2. Detects heat sources using AMG8833 thermal sensors. 3. Shoots ping pong balls using a brushless motor-powered flywheel launcher. 4. Uses ROS2, custom Python scripts, and C++ for control.
Common Trouble-shooting	- Navigation failure: Restart Nav2 stack; check SLAM localization and RViz visualization - Heat sensor missing source: Ensure both AMG8833s are functional; - Wrong identification of source location: Ensure front of lidar and AMG8833 are the same - Launcher malfunction: Check motor ESC connections, flywheel tightness, and motor RPMs.

	- Spiral tube is misaligned, balls do not slide down: add a weight at the end to ensure all the balls slide down smoothly.
Safety and precautionary measures	- Only operate in open, obstacle-free environments. - Avoid touching flywheels while powered (risk of injury). - Monitor battery to prevent voltage drop below 9V. - Prevent exposure to liquids or moisture.

Section 2: Technical Guide

Software Setup:

Install Nav2 Stack:

- Git clone the nav2 folder into the colcon_ws: https://github.com/ROBOTIS-GIT/turtlebot3
- Git clone our code from bot_bestie branch: https://github.com/Lilyflower77/r2auto_nav_CDE2310/tree/bot_bestie
- Run “colcon build” to build all dependencies

If missing any packages, run: conda install [package name]

Deployment

Step 1: Load ping pong balls into spiral from open end
Step 2: Turn TurtleBot on
Step 3: ssh into turtlebot
Step 4: run ‘rosbu’ on bot
Step 5: run ‘slamtoolbox’ on PC
ros2 launch slam_toolbox online_async_launch.py use_sim_time:=false
Step 6: run
ros2 launch bot_bestie global_bringup.py

Expected Output: Robot will start mapping the area and start recording Hot areas. After mapping, the bot will go to hot areas and launch ping pong balls upwards.

Section 3: Acceptable Defect Log

Defect no.	Defect	Justification
1	Minor cracks on spiral/tube	If there is no hindrance to the movement of the ping pong balls, there will be no negative effect on the bot's intended operation
2	Two spiral parts not perfectly aligned	If the ping pong balls roll down the spiral successfully, there will be no negative effect on the bot's intended operation
3	Small number of screws /nuts drop off/missing	The bot is designed to function without the fasteners. There will be no severe negative impact on operation.
4	Missing/unstable stand-off	At least 3 stand-offs per layer is required for stability. One unstable/missing stand-off out of five per layer is acceptable.
5	Platform rises on startup	PWM jitter on raspberry pi startup, push platform down to lowest position before commencing

Section 4 Factory Acceptance Test

Test Description	Expected Outcome	Test Done
Run 'rteleop' test for ability to move	Bot follows all commands and move in right direction	☑
Run '*rosbu*' on startup	Flywheels will generate two different sequence of tones	☑
Run '*rosbu*' and let it map a maze	9/10 successful runs; under 8 mins per run. >95% accuracy of maze on Rviz	☑
Run '*rosbu*' and let it map an area with obstacles.	95% success rate. Area mapped with ≤1 collision in 20 runs	☑
Heat source detection (static test) Run 'rosbu' with heat in front of bot	Output: '🔥🔥🔥🔥FIRE' in terminal 100% detection within 2 m	☑
Launcher test by running *ros2 topic pub /flywheel std_msgs/msg/Int32 "{data: 50}"* (single launch message received)	Launches 3 balls successfully, ball clears 1.5 m wall height, no jams, 2 -4 -2 timing	☑

Section 5 Maintenance and Part Replacement Log

Date	Part Replaced	Issue