Overcoming the Limitations of Standard LIDAR in Autonomous Robots using Ultrasonic Sensors

Ryan Hurnen

***Abstract*— Autonomous robots, with potential applications in agriculture, healthcare, and repetitive tasks like cleaning, rely on accurate sensor data for optimal performance. LIDAR, a prevalent sensor in robotics, struggles with detecting objects that are transparent or absorb infrared emissions. To address this, we developed a system that integrates a 360-degree infrared LIDAR with eight 40kHz ultrasonic sensors, uniformly spaced in a 3.8cm radius circle. This configuration ensures that non-overlapping areas are within the ultrasonic sensor's 15 cm minimum detection distance. All sensors are housed on a 3D-printed mount, positioned to prevent interference from the robot's echoes.**

We carried out extensive software development, establishing two Wi-Fi-connected Robot Operating System (ROS2) workspaces for both onboard robots and external computers. These workspaces feature a custom plugin for the ultrasonics representation in the ROS2 navigation stack (NAV2) and specific nodes to relay ultrasonic sensor data and transformation information. This integration allows for the incorporation of ultrasonic data in A* path planning calculations. Ultrasonic sensors and robots' ability to navigate, were rigorously tested to ascertain their accuracy. Ultrasonics showed discrepancies within 1cm, allowing NAV2 adjustments for accurate navigation.

The robot's ability to navigate to a position given its bulky nature was shown to be at worst 80% as accurate as existing solutions. When comparing ultrasonic to non-ultrasonic systems, we noted a 3-11-fold increase in navigation time for ultrasonic systems. However, the combined system adeptly navigated challenging terrains and detected objects typically invisible to LIDAR. This combined approach, as exemplified in Clutterbot's system, can potentially reduce onboard processing demands, leading to cost savings, extended operational lifetimes, and an expanded task repertoire for robots. By increasing the efficiency of such robots, we can stimulate more robotic utilisation in the world, allowing a wide range of tasks to be offloaded, increasing the productivity and autonomy of societies thus progressing towards the United Nations' Sustainable Development Goals.

I. INTRODUCTION & PURPOSE

IN the specialized domain of autonomous robotics, the accuracy and reliability of sensory input data are paramount [1]. Robots envisioned to revolutionize diverse industries, rely heavily on their navigational capabilities [2]. LIDAR (Laser Imaging Detection and Ranging) has emerged as an essential tool in this context, offering detailed environmental mapping [3]. However, its limitation in detecting transparent or infrared-absorbing (TIA) objects poses significant challenges in autonomous navigation systems [4] including cleaning robots, as identified by companies like ClutterBot, a company seeking to develop cleaning robotics (as identified in private conversation) [5].

This project was supervised by Supervisor Hamish Colenso, and Supervisor Christopher Hollitt.

The United Nations' Sustainable Development Goals (SDGs) aim to tackle global challenges [8]. Numerous experts are exploring solutions within these seventeen goals [9]. Researchers highlight six transformative paths for their realization, with the pivotal sixth anchored in the Fourth Industrial Revolution's technological leaps, such as artificial intelligence and autonomous robotics [10][11].

Integrating automation in industries offers improved productivity, cost savings, and sustainability, emphasizing a circular economy. Such advancements are crucial for New Zealand, given its recent labour challenges affecting productivity [12][13].

Overall research goal

This research focuses on enhancing autonomous navigation, particularly in scenarios with ITA objects, leveraging ultrasonic sensors and LIDAR technologies. The system under development will adopt an open-source framework, aligning with standards set by the Open Robotics Robot Operating System [13.1]. The aim is not only to create a more cost-effective robotic solution but also to contribute to the realization of the Sustainable Development Goals (SDGs) and support the United Nations Sustainable Development Goals (SDGs).

Current studies indicate that only 16% of robots in the agricultural sector employ LIDAR for navigation [13.2]. The potential of compact robots, capable of incorporating 3D-printed components, is particularly evident in rural areas. These robots offer advantages such as on-site replacement of parts and increased operational efficiency in farming environments [13.2].

From a regulatory perspective, another research paper underscores the importance of human oversight in robotic operations [13.3]. Demonstrable performance metrics, such as the ability to operate without collisions, may influence regulatory stances. It's worth noting that New Zealand's adoption rate of autonomous robotics is relatively low, a situation exacerbated by labour shortages during the COVID-19 pandemic. An interesting observation from the study is the emerging consensus that smaller, coordinated robot groups may offer greater agricultural efficiency compared to their larger counterparts [13.3].

This paper introduces an approach to the integration of high-frequency LIDAR with low-frequency ultrasound sensors. This

synergistic approach aims to harness the collective strengths of both systems, ensuring holistic object detection. The robotic system constructed is built upon a robust platform developed by our engineering department depicted in Figure 1, taken from [7], the coloured boxes depict different existing aspects of the robot. In Purple is the Raspberry Pi, yellow indicates the motors, red indicates the motor driver's "robot claw", orange indicates the battery's typical position, and blue indicates the voltage regulation involved with supplying 5V power for control circuitry. The green indicates a previous sensor used for student learning purposes. Physical improvements to this system include a mounted 360º field of view (FoV) LIDAR and an added array of ultrasonics distance sensors.

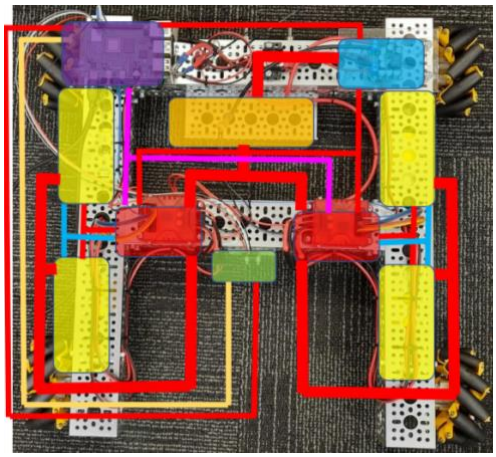

Figure 1, The omnidirectional "EEEN325 strafing robot platform" taken from [7].

The system's software backbone is ROS2, a state-of-the-art open-source robot operating system [14], complemented by developed C++ and Python elements, relevant libraries such as navigation stack 2 [15] (NAV2) and slam_toolbox [16]. The efficacy of this solution will be evaluated based on its ability to navigate, complex environments and discern TIA objects, showing the advantages without ultrasonic sensors. The successful implementation of this solution could mark a significant milestone in the realm of autonomous robotics, offering enhanced efficiency and safety for robotics to be used in more settings. Furthermore, the environmental implications of such advancements, particularly in reducing carbon emissions through automation of carbon-intensive tasks underscore broader societal benefits. Subsequent sections of this paper will provide a detailed exploration of the related work, requirements, and technical intricacies with included challenges encountered.

Contents:

A. Tools

An understanding of the basic functioning of the robotic operating system and a thorough understanding of all stages of robotic design relative to this project is required for this report:

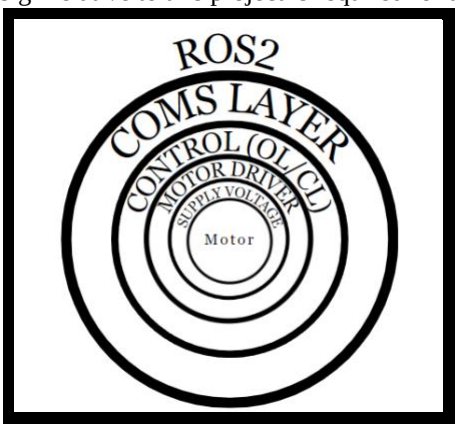

Figure 2, the onion of dependencies for generic ROS2 robotics projects.

The Figure 2 diagram shows the generic hierarchy system involved in a typical ROS2 robotics project. The centre of the onion shows how motors are controlled via different independent layers that provide abstraction and exportation of compute power for ease of use and flexibility.

In the next layer of the onion, the supply voltage provides the voltage needed to control the motors. This project uses a sealed lead acid battery (SLA). The SLA is nominally around 12V, however, needs to be charged before reaching 11.5V to increase the lifespan of the battery. A DC-to-DC converter 5V is produced to supply the Roboclaw motor driver logic circuit, Raspberry Pi, and LIDAR and ultrasonic sensors when attached.

The next stage depicted in Figure 2, is the motor driver. A motor driver is a device that regulates the speed and direction of a motor by taking a low voltage, low current signal from a controller and amplifying it to provide the necessary power to drive the motor. The control signal is typically in the form of a Pulse Width Modulation (PWM) signal, where the ratio of the duty cycle determines the motor's speed. A motor controller calculates the appropriate signal to send to the driver to achieve a target speed or position. It can be combined with the driver or motor itself.

An open-loop controller maps input signals to output signals. However, this approach can be affected by factors such as load and battery charge. To achieve robust automation, the actual speed of the motor must be measured and fed back into the controller for adjustment. These methods are called open-loop and closed-loop control, for our system the signal and required calculations to give to the motor are abstracted away by the Roboclaw motor driver. In our system, the Roboclaw uses PID closed-loop feedback to ensure the correct speed or position. Separating the motor control from the Raspberry Pi allows more flexibility in our design.

The Communications layer involves communication between physical circuitry, from the onboard odometry to the motor driver, between the motor driver and Raspberry Pi and all other sensor connections involved in our robot.

Finally, ROS2. ROS2 is an updated version of the Robot Operating System (ROS), which consists of a collection of software libraries and tools that aid in the creation of robot applications. It offers a structure for the development of robot software and comes equipped with drivers, algorithms, and tools for developers [14]. ROS2 provides communication between defined nodes which handle specific tasks [14]. Tasks could involve specific sensors or routines that combine information or send out outputs to external services like the motor driver. Communication between nodes can also be set up so tasks that require heavy computing like navigation can be done externally from another computer [14].

In ROS2, nodes can communicate with each other using three main methods; Topics, Services and Actions [14] shown in Figure 3,4.

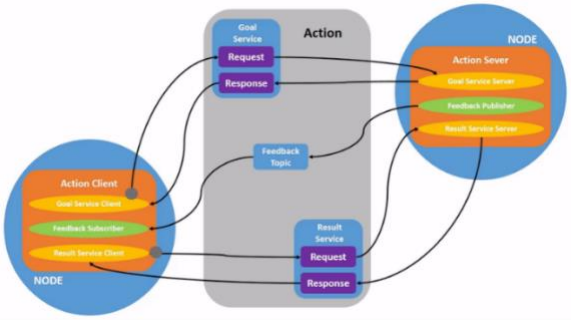

Figure 3, a diagram showing an example of action-based communication between nodes [18].

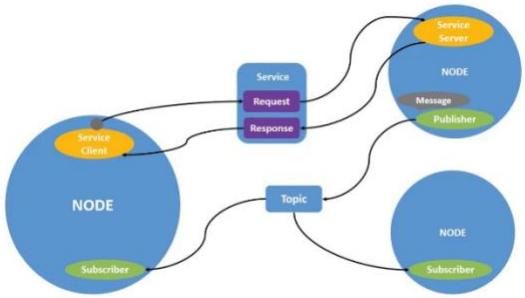

Figure 4, is a diagram showing the differences between topic and service communication between nodes [19].

Topics are used for continuous data streams such as sensor data or robot state [20]. For example, transmitting data to the motors, or receiving information from the sensor suit. They can also be used to update the status of a longer-running process contained in an action.

Nodes facilitate communication by publishing and subscribing to specific topics, offering services for request-response interactions, and supporting composition. This compositional approach, akin to a nodelet and termed a "Component", simplifies the integration of common functionalities into existing code [21] [22].

Services They are used to initiate the action, and receive the processed response once the action is complete. [20].

Actions are used for long-running tasks that have a clear beginning and end. An Action Client node sends a task to an Action Server node, which provides feedback on its progress towards achieving the goal and sends a result message when the goal is completed [20].

SLAM_toolbox

SLAM_toolbox, a notable ROS package, is used to conduct simultaneous localisation and mapping in ROS2. It tracks a representation of the robot's pose (position and orientation) and environmental sensors over time. The pose and sensor information are used to generate a map which represents the environment the robot is navigating. SLAM toolbox offers advanced capabilities, including the ability to utilise the ROS2 transform library (tf2), where it can form a representation through a transformation of the robot's poses relative to other "frames" of reference (positions and orientations) over time [23]. This pose graph is crucial in Simultaneous Localization and Mapping (SLAM) as it helps in understanding the structure of the environment for navigation [16][23]. This global map is a representation of the environment that the robot has explored and mapped [16]. The toolbox can save and serialize the data and pose-graph of this SLAM map, allowing it to be reloaded later [16]. This can be useful for continuing mapping, localizing, or merging with other maps.

SLAM Toolbox operation

SLAM_toolbox works by taking data from the laser scan and odometry topics. Using this input, it processes the data to construct an occupancy grid map of the environment and determine the robot's position within that map. The primary outputs of the SLAM_toolbox are the map-to-odometry transform and the generated map of the environment.

The "map to odom" transform describes the relationship between the map frame (a fixed reference frame representing the map) and the odom frame (a frame representing the robot's

odometry). In the context of SLAM (Simultaneous Localization and Mapping), this transform is crucial for reconciling the robot's internal odometry with the global map that SLAM is building or updating. If at any point the odometry is inconsistent with measured sensor data, the relationship between this fixed frame "map" is adjusted, to compensate. The reconciliation is known as closing the loop.

The Navigation stack

The Navigation Stack (NAV2) is the professionally supported successor of the ROS Navigation Stack [15]. It incorporates advanced technologies used in Autonomous Vehicles and optimizes them for mobile and surface robotics [15]. **NAV2 is a comprehensive navigation tool**, it is designed to handle complex tasks, allowing a mobile robot to autonomously navigate through various environments using a range of robot kinematic equations [15]. It can manage tasks beyond just moving from Point A to Point B, such as following objects and navigating through intermediary poses. NAV2 is a **production-grade** and high-quality navigation framework that is trusted by over **50 companies worldwide** [15]. NAV2 uses behaviour trees to create customized and intelligent navigation behaviour by orchestrating many independent modular servers. This modular approach allows for the creation of unique tasks and behaviours for different robots [15].

NAV2 operation

Behaviour Trees: NAV2 uses behaviour trees to orchestrate various independent modular servers. These servers can handle tasks like computing a path, controlling the robot, recovery, and other navigation-related tasks. They communicate with the behaviour tree over a ROS interface, such as an action server or service.

Plugins: NAV2 supports multiple plugins for controllers, planners, and recoveries. These plugins can be used to create contextual navigation behaviours tailored to specific needs. In this report, I will discuss the creation of a customizable NAV2 plugin to tailor the ultrasonic data stream to an accurate description of its detection.

Inputs: The expected inputs to NAV2 include tf2 transformations, a map source (if using the Static Costmap Layer), and relevant sensor data sources such as the ultrasonic associated range sensor message and the "scan" laser scan topic for the LIDAR.

Outputs: NAV2 provides valid velocity commands for the motors of various robot types, including holonomic, differential-drive, legged, and Ackermann (car-like) base types. It supports both circular and arbitrarily shaped robots for SE2 collision checking.

NAV2 offers a range of tools such as the Map Server for map management, AMCL for localization, NAV2 Planner for path planning, NAV2 Controller for robot control, and NAV2 Smoother for path smoothing [15]. One tool of focus for this project is the use of a plugin to alter the NAV2 2D costmap, this tool allows a globalized and localized costmap representation to guide the controller [24]. Cost in the local costmap is used to offset the A* algorithm, regions of high cost are avoided entirely, and regions of low cost offset the path planner to avoid the area if possible. Zero cost indicates regions in which the A* algorithm is unaffected, so it will take the most direct and optimal path [25][26].

ROS 3D Robot Visualizer (RIVZ2)

RVIZ2 is the premier visualization tool for ROS2, designed to provide real-time 3D or 2D visual insights into robotic operations [27]. It facilitates the display of diverse data, from sensor readings to robot trajectories, enhancing the debugging and understanding of robotic algorithms [27]. Its integration with the ROS ecosystem, coupled with its user-friendly interface, positions RVIZ2 as an indispensable asset for roboticists. Figures 5 and 6 give an example of visualising different aspects when the robot is fully functional in its completed state. Figure 5 gives a reference point "Map" frame to adjust for failures in odometry. The red dots within this image represent the individual LIDAR measurements, the black pixels represent detected geometry on the global map created by SLAM. The white represents areas of certain emptiness.

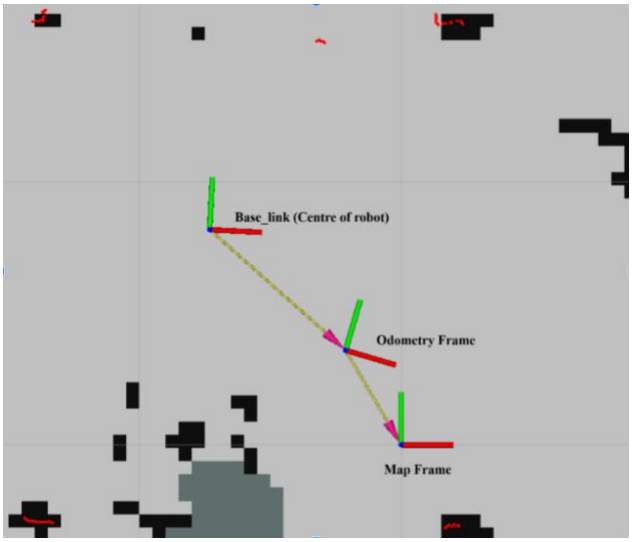

Figure 5, a visualisation of the robot (position and orientation) frame, relative to the odometry adjustment frame, and the global map frame.

Adding onto Figure 5, layering on the cost map, LIDAR, ultrasonic orientation and position relative to the centre of the robot we come to Figure 6.

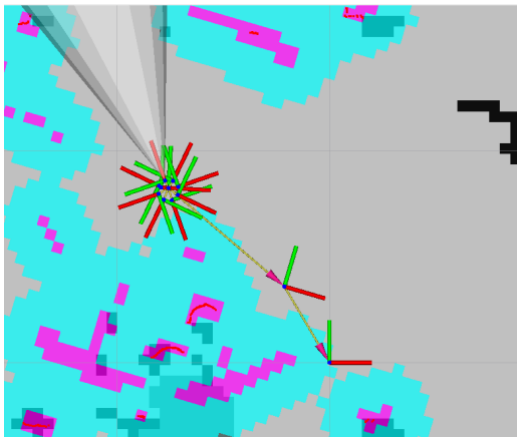

Figure 6 is an extension of Figure 5 with a costmap and nine added axes positioned to represent the sensors' actual layout in space.

Figure 6 is a coloured pixel array, the purple represents a "LETHAL" cost of 100 or above [28] of which it is so costly the robot is not able to navigate through it using the chosen A* navigation algorithm. The blue represents lesser costs of 60 to discourage the robot from moving into the regions around the detected objects created by the inflation layer [28]. Note the cone extended outwards off-screen indicates an active use of the ultrasonics adding to the cost map.

B. Related work

The challenge of accurate navigation in autonomous robotics, especially in environments with ITA objects, is not new [29][30][31]. The reliance on LIDAR, while beneficial in many scenarios, has its limitations, particularly when it comes to detecting ITA objects [29][30][31]. This section reviews some of the notable works in the domain, focusing on the integration of different sensors and methodologies to overcome such limitations.

Multi-sensor Fusion Glass Detection for Robot Navigation and Mapping by Hao Wei et al. [29] highlights the challenges faced by robots in modern buildings with transparent objects like glass panels. The paper proposes a method to detect glass panels based on sensor fusion of ultrasound and laser LIDAR data. This approach is especially pertinent given our project's focus on integrating high-frequency LIDAR and low-frequency ultrasound sensors.

Comments on the study:

- This study was not able to merge the glass position onto the generated map, which was done by hand. In this project, this will be tackled using NAV2's local cost map and the additional aim of implementing the previously mentioned triangulation-based method so that it may be appended to the global cost map.
- This study mentions that they have selected laser LIDAR data in the range of the beam angle of the ultrasonic sensors they used, which has a beam angle of 30º. The real data is said to be the same as the ultrasonic sensor data. However, there's no explicit mention of uncertainty associated with the arc that ultrasonic produces, therefore it must have been omitted.
- This study does not provide explicit information about the speed at which the robot moves, or the number of samples required to produce the map. Since, there are only three ultrasonic sensors fixed in the front of the robot, and another one is directly behind. This provides sparser information than my planned setup.

Project requirement implications:

The project to show improvement on existing systems must be able to manage navigation, localisation, and detection of ITA objects simultaneously.

Robust Mobile Robot Localisation from Sparse and Noisy Proximity Readings using Hough Transform and Probability Grids by Axel Großmann and Riccardo Poli [30] introduces a position-tracking method for a mobile robot using sonar sensors. The method employs the Hough transform and probability grids, focusing on handling sparse sensors and noisy data for real-world applications. This paper refers to seven ultrasonic sensors placed at the front of the robot and as well as a ring of sixteen ultrasonic sensors as "*sparse*" information which doesn't "*work reliably*", it's worth noting that if a setup involving less than sixteen ultrasonics is achieved it would greatly reduce the cost.

Comments on the study:

- The method used in this project is not solely reliant on ultrasonic for localization and only needs simplistic data for local navigation and basic geometry for mapping purposes. Therefore "*sparse*" levels of sensing data can be justified as we have comparatively dense levels of information provided by the LIDAR sensor.

Project requirement implications:

The project must function reliably in navigation compared to existing solutions and allow the detection of objects using a "*sparse*" array of ultrasonics to decrease cost, increasing the financial incentive for projects to use robotics to help solve different tasks.

SLAM in Large Indoor Environments with Low-Cost, Noisy, and Sparse Sonars [31] discusses the SLAM problem using low-cost, noisy, and sparse sonar sensors in large indoor environments. The paper presents an approach to SLAM with sonar sensors, applying particle filtering and a line-segment-based map representation. These works underscore the ongoing efforts in the robotics community to address the challenges posed by environments with transparent objects. The integration of sonar and the fusion of their data appear to be promising avenues for enhancing robot navigation in such scenarios. Our project builds upon these foundational works, aiming to provide a cost-effective and efficient solution for robots navigating environments with transparent obstacles.

Comments on the study:

- This paper focuses on using sixteen ultrasonic sensors to achieve localisation and mapping, whereas as previously mentioned will be using fewer ultrasonic sensors and enabling a fusion of sensors for local navigation purposes.

- The maps in this paper mention taking between 5-30 minutes of scanning to create this project using LIDAR will be able to construct a map within milliseconds of SLAM operation.

Project requirement implications:
The generation of local and global cost maps must be done so that the robot can generate a map and navigate within seconds to a few minutes at most.

The company Clutterbot
Clutterbot is a company dedicated to developing robotic cleaning solutions [5]. Currently, they are challenged with cost reduction and ITA object detection to avoid inefficient navigation and property damage. This will be a major driving factor for upcoming requirements, to meet the needs of this company and for contributing to the open-source robotics community.

C. Requirements

To meet the needs of companies like Clutterbot, and SDG, and contribute to the open-source robotics community this project must be able to fill requirements laid out in this section such as cost, power and processing considerations under computation, accuracy and precision measurements of linear motion and ultrasonic measurements, timing measurements for obstacle completion, path planning analysis to show the distance from the object is kept at a margin greater than stopping distance. The following requirements with associated four-letter code are established:

Cost efficiency requirement (COST):
Cost-effectiveness is paramount for Clutterbot and other robotic platforms to adopt such a scheme, the solution should be or show the capacity to be expanded to be cost-effective, ensuring that the integration of high-frequency LIDAR and low-frequency ultrasound sensors does not significantly increase the overall cost of the robotic platform as required by Clutter Bot.

Specifications:

Industrial grade LIDARs are often onwards of $~27.70-11,000 NZD [32] to bring down potential costs this must be minimised and use only what the project necessitates. Using ultrasonics will also likely lower requirements for accuracy therefore in our project, we must use a consumer-grade LIDAR with an estimated cost of $100 [33]. Ultrasonic cost also must be kept to a minimum, in an article discussing Tesla's installation the total cost was estimated to be $114 USD [34] for 12 sensors, which amounts to $16 NZD per sensor, therefore, our system must be a cheaper solution.
Fulfilled:

- The selected RPLIDAR A1M8 sensor, priced at NZD 99, offers a 10Hz scan frequency and 1-2.5% range accuracy [35][36]. This sensor represents an upgrade from the previously used Hoyuko 270-degree LIDAR [37], as it can detect rear objects. Alternatively, a more affordable option is the NZ$26.70 LD06 LIDAR portable 360º FoV direct time of flight (DTOF) Laser Sensor Scanner Kit from Aliexpress, offering a 5-13Hz scanning frequency and +-45mm accuracy [38].
- The URM13 ultrasonic sensor breakout board by DFRobot, costing NZD 15, was chosen primarily for its quick availability [39]. However, other cost-effective options, like the 3V-5.5V SR04P Ultrasonic Ranging Module from Cytron.io priced at USD 1.23, exist [40]. It's worth noting that using the latter option would necessitate additional hardware for signal interpretation, introducing unnecessary development challenges.
- The URM13 ultrasonic sensor breakout board from DFRobot, priced at NZD 15 [39], was chosen for its quick availability. While more affordable options like the 3V-5.5V SR04P Ultrasonic Ranging Module from Cytron.io at USD 1.23 exist [40], using them would necessitate additional hardware for signal interpretation, introducing unnecessary development complexities.

Computational reduction requirement (COMP):
Clutterbot need to reduce costs, one of the biggest costs in their current project is the single board computer (SBC) Nvidia Jetson AGX Xavier [6] which currently provides the brains of the robot. A paper published by one of the members of clutterbot [6] Anuj Rathore indicates that the camera detection system cannot operate with the Jetson hardware and computation must be exported to the cloud for operation, making the original justification behind using the Jetson redundant and indicating the need for a fast local navigation system without the latency involved in cloud computing paramount.

Specification:

The chosen microcontroller should demonstrate a reduction in benchmarks, specifications, and cost.
Fulfilled:
[41] Using this provided data it appears that the Nvidia Jetson is 1.77x times faster on average using all the selected benchmarks than Raspberry Pi. Specs comparison using [42][43]:
CPU: Jetson AGX Xavier: 8x ARM v8.2 @ 2.26GHz, Raspberry Pi 4B: 4x ARM Cortex A72 @ 1.5GHz, 3x reduction in throughput.
GPU: Jetson AGX Xavier: 512x Volta GPU with 64 Tensor Cores Raspberry Pi 4B: Broadcom VideoCore VI @ 500 MHz
Memory: Jetson AGX Xavier: 32GB LPDDR4 (137GB/s) Raspberry Pi 4B: Options of 2GB, 4GB, or 8GB LPDDR4 (4x reduction)
Storage: Jetson AGX Xavier: 32GB eMMC Raspberry Pi 4B: MicroSD card.
Vision: Jetson AGX Xavier: 7-way VLIW Vision Accelerator Raspberry Pi 4B: OpenGL ES 3.0 Graphics
Price: Jetson AGX Xavier: $699 Raspberry Pi 4B (8GB RAM): $75.00 (9x reduction)

Accuracy and precision requirement (ACRY):
Assumption of a flat surface in an uncluttered and well-traversable environment
Decreasing accuracy might impact navigation negatively as the path planner may not be able to accurately control position to

avoid objects. To avoid this outcome, we will set an additional aim of the project to try and get accuracy as close to existing cleaning robotics solutions, so we can guarantee successful navigation. 94% floor coverage appears to be what cleaning robots can achieve [44], therefore if we move 1 metre in the x and y direction, we expect to see a 6 cm deviation. This means that if we test only in one direction, we should expect all results to be within 3% of the mean for all tests. If my robot can achieve this then it will be able to clean the surfaces to the same degree as existing methods using the EEEN325 robot which is likely more difficult to control in terms of momentum and drift than cleaning robots in [44]. In this project we are utilising sensors that detect sub 6 cm, therefore the robot should be able to position and perform with greater accuracy, allowing us to complete this objective and allowing this solution to be more appealing to adopt for other open-source robotics projects.
Specifications:
A movement test in one direction must be undertaken to show their final navigation location vs the expected location. If the tests are within a range of 3% of the target distance from the target distance, this bodes well for overall controllability and helps to achieve requirements DOUT and DTRAN so that it both detects and accurately away from detected objects.

Detection and improvement requirements:
To offer improvement over existing systems and to prove the viability of generic navigation with this setup, the additions of the ultrasonics sensors must be able to detect the following objects and add them to the cost map:

- Detection of Transparent objects (**DTRAN**) using ultrasonics.
- Objects out of view **(DOUT),** of the LIDAR and within the cone of ultrasonics.

The ultrasonic addition must not adversely affect the planning by more than specified by **ACRY** unless new sensor information reveals a collision would occur, as would happen when **DTRAN** or **ACRY** occurs. Doing so definitely allows robotics to operate in new areas enabling a wider range of robotics in society.

Specifications:
When an ITA object is present within the test the ultrasonics must detect it. Tests must show the robot's ability to stop in front of and the ability to navigate around both detected objects by a radius of the stopping distance determined by SAFE.

Safety Protocols requirement (SAFE):
Ensure that the control of the robot does not act to collide with ***detected objects***, potentially damaging both itself and the object or uncontrollably ***(soft requirement)*** where the robot takes a conservative sensible path around objects to compensate for potentially unreliable data. If the robot runs into objects this could cause damage to itself and the collided object, the tested stopping distance of the robot is within 2-4cm (average of 2.81 cm at set $0.2m.s^{-1}$) when moving in one direction. Tested using by applying constant speed and at set point giving changing the command velocity, video analysis allowed accurate data collection.

Specifications:
The robot must always keep a 4cm buffer distance from objects to show accurate control and prevent accidental collisions due to stopping distance.

Open-source requirement (OPEN):
This project is designed to help the robotics community and company Clutterbot. It must remain open source and available for use.
Specifications:
Access to this project must be accessible and downloadable from online sources.

Fulfilled:

- GitHub Link to project [45][46]

II. Design and Implementation

To achieve the above requirements, the project conceptually was split into hardware acquisition, physical planning, and software development stages.

A. System Architecture:

The URM13 ultrasonic sensor was procured from DFRobot, a reputable platform known for its timely shipments within New Zealand. Within the allocated budget, we were able to acquire eight of these sensors. The URM13 model is characterised by its ultra-low power consumption and an effective range of 15-900cm.

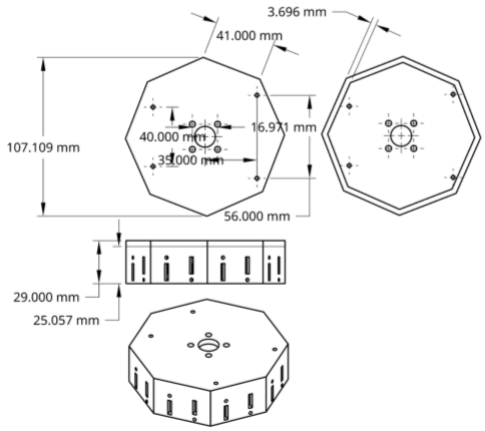

Figure X shows the design of the octagonal housing for the ultrasonic sensors array and LIDAR. Each outer face of the octagon contains one ultrasonic sensor mounting point. The flat top has mounting points for the LIDAR.

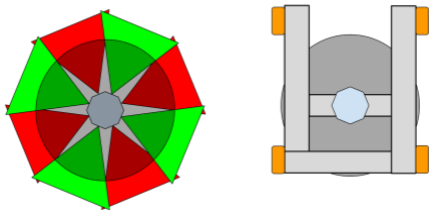

Figure 8 is a demonstration of the cone layout of the ultrasonic sensors surrounding the octagon sensor mount in the 2D plane.

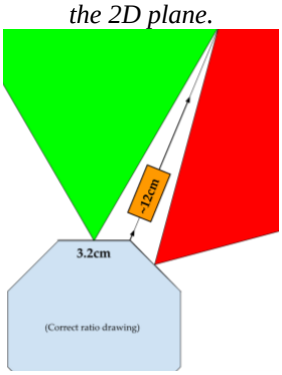

Figure 9, a scaled-down-to-scale of two ultrasonic sensors overlapping in the 2D plane, the distance to the start of the overlap is roughly 12cm from the sensor mount.

It operates within a power supply range of 3.0 - 5.5V and is seamlessly compatible with 3.3V or 5V devices, including the Raspberry Pi, via an Inter-Integrated Circuit (I^2C) bus. This compatibility facilitates the connection of multiple sensors to the Raspberry Pi's 5V pin and I2C ports, enabling direct distance communication and reducing overhead. Notably, this sensor incorporates an open-source library to process incoming data, meeting the ACRY, DIR-TO, and RNOR requirements. The central sensor control approach as mentioned previously using seniors [40] is cost-effective and holds the potential for future expansion to accommodate COST requirements.

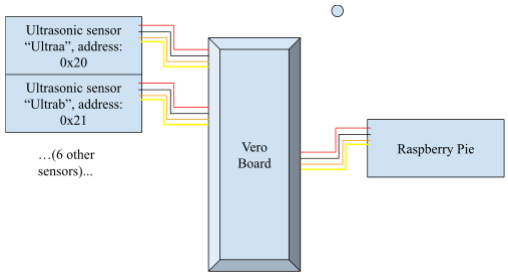

Figure 9.1, physical wiring connections from Raspberry Pi to sensors.

The RPLIDAR A1M8 LIDAR was selected to fill the RNOR requirement, given its compatibility with the turtlebot4 open-source software, which significantly reduces development time. The A1M8 represents an upgrade from the Hoyuko 270º FoV LIDAR used in the EEEN325 robot previously [38], which is not able to detect rear objects. It features a 360º FoV laser scanner and is a cost-effective 2D LIDAR solution by the RoboPeak Team, capable of scanning up to a 12-metre radius [36]. There's potential to meet the COST requirement by replacing this sensor with a cheaper equivalent sensor, the NZ$27.22 LD06 LIDAR, a portable 360º FoV DTOF Laser Sensor Scanner Kit from Aliexpress, boasting a 12m range, 5-13Hz scan rate, and an accuracy of ±45mm [37].

The designed robot is running on a Raspberry Pi and thus cannot effectively run the computationally expensive NAV2 and Slam toolbox. Therefore, it is necessary to use a hotspot linking the ROS2 system to an external desktop computer running a Linux virtual machine. This enables me to run ROS2 nodes on the desktop which take in connected sensor data from the robot platform via Wi-Fi. An additional design benefit of this is being able to remotely control the robot using RVIZ2 and separate the navigation software from the onboard robotics software. This mimics the intended Clutterbot design, where computation is offloaded to a cloud service provider to keep the cost of individual robots low and allow the aggregation of computer vision training data.

The current setup of my software side is simplified and shown in Figure 10. NAV2 intricacies aspects of which involve the following:

- The Behaviour trees (BTs): These are a graphical way to plan tasks in NAV2. The main behaviour tree used in NAV2 replans the global path periodically and has recovery actions. BTs are primarily defined in XML and can be broken down into smaller subtrees for easier understanding.
- The Path following: This refers to the robot's ability to follow a computed path. If the path following fails, contextual recoveries will be attempted.
- The Planning servers: These handle the computation of paths. The default behaviour tree in NAV2 plans at a frequency of 1 Hz to prevent flooding the planning server with too many requests. Lifecycle managers: Not explicitly mentioned in the provided content.
- The Goal poses: These are the destinations the robot aims to reach. The behaviour tree checks if a new goal has been received and can react quickly to new goals.
- The Controller server: This is related to the robot's movement. The local costmap is relevant in the context of the controller.
- The Local and global costmap also is a simplification and involve aspects such as:
- The High-level interface, used to compute kinematics and communicate the state of the robot to NAV2 and Slam toolbox.
- The Low-level interface, used to communicate to the Roboclaws and give velocity commands to drive the robot through its associated Motor class.
- Command velocity topics sent from NAV2 and the Xbox controller when configured allowed the robot to be controlled to a set velocity.
- The Twist mux node, combines twist messages on two streams into a single stream, allowing for priority messages from the Xbox controller.
- The Global/local costmap node: This refers to the representation of the environment around the robot. NAV2 uses a module called "NAV2 Costmap 2D" to convert sensor data into a costmap representation of the world. This costmap provides a layered 2D grid map that the robot uses for navigation purposes, such as path planning and obstacle avoidance.
- The Published footprint topic: The footprint represents the robot's physical shape or outline in the costmap. It can be defined in different forms, such as a polygon or a circle. For instance, in the configuration of "sam_bot", a

polygonal footprint is used for the local costmap, while a circular representation (defined by "robot_radius") is used for other purposes. Additionally, footprints are published for both global and local nodes, ensuring that they are actively running and being utilised in the navigation process.

- The Published raw topic: The provided search results do not offer a direct explanation or description of the "Published raw topic" in the context of NAV2 or ROS2. However, in ROS (Robot Operating System), topics are channels where nodes publish and subscribe to messages. The term "raw" typically indicates unprocessed or primary data, which might be directly from sensors or initial computations.

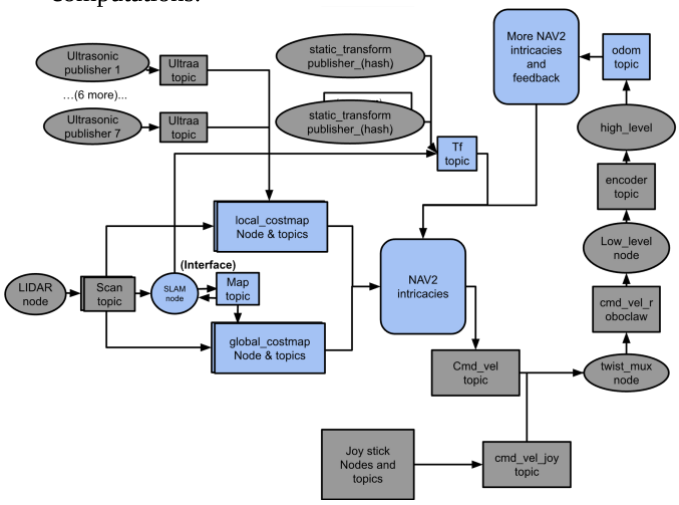

Figure 10, shows the simplified link between nodes and topics for ROS2 shared across the robot (marked in grey), and external computer (marked in blue).

B. Physical planning:

LIDAR and ultrasonic Placement: The ultrasonics height is set at 29.6cm to prevent any interference with the robot's chassis, particularly its backboard. For robots resembling a Roomba, which are closer to the ground, the 3D detection can be approximated to a 2D plane shown in Figure 8, the dark circle region on both images indicates the 15cm region in which the ultrasonics cannot detect objects, encapsulating all the non-overlapped areas. Furthermore, the 15 cm non-detection zone sits within the robot platform, shown in the right-hand diagram, effectively reducing it to a negligible area.

As the robot is omnidirectional, detection in all directions is needed, thus the ultrasonics will be arranged outwards in an *octagonal* pattern. The sensors are mounted on a raised platform shown in Figure, evenly spaced, to minimize signal overlap. This design not only simplifies the implementation but also ensures optimal sensor performance. When arranged in such a way ultrasonics have a non-detection zone, where if an object were placed the objects would not encounter the emitted sound. However, when the sensors are arranged in a circle of approximately 3.8cm radius, simplifying to a 2D plane the arrangement is calculated to be less than the minimum quoted ultrasonic detection distance of 15cm. As demonstrated by Figures 8 and 9. Figure 9.1 shows how each sensor name and address are associated. Additionally, it shows how all wires 5V (red), GND (black), Tx (yellow) and Rx (orange) are inputted into the Veroboard. Note, the one 5V connection going from the Raspberry Pi to Veroboard splits into 8 sets of parallel connections going to each sensor. The same is true for other wired connections.

Given this, objects within a range of 15-900 cm in this plane are detectable by at least one ultrasonic sensor. However, if required to detect objects substantially lower than the LIDAR, then ultrasonic sensors may provide this additional benefit. The placement of the ultrasonic sensors relatively high compared to the ground hopefully highlights this benefit.

By placing the LIDAR on top of the sensor mount depicted in Figure 7, 11 it can achieve a full 360º FoV detection around the robot without obstructions from the chassis or pillar. Thus, the planned arrangement of sensors acts to fill requirements DIR-TO and DIR-OV.

Connectivity: The sensors interface with a Raspberry Pi through separate connections to one Veroboard, using custom-made female header connectors, saving on cost as needed for COST requirements. The design also incorporates a wiring loom and dedicated passage in the aluminium pillar and chosen sensor mount for wired connections to the ultrasonic sensor breakout board. The Raspberry Pi connects to the LIDAR through a PCB, via a USB type A to C connection.

Material Utilisation: The sensor platform will be made from 3D printed ABS for rapid prototyping, its shape will be an octagon to provide 8 flat surfaces separated by 45º.
The sensor platform is affixed using an aluminium pillar, which not only provides a robust structure but also promotes the reuse of materials from the EEEN325 course, thus promoting sustainability goals.

C. Sustainability aspects of the project

Economic maintenance of my project:

- Motors: The longevity of motors varies based on the type, usage, and maintenance. A detailed analysis out of the scope of this report is required to determine the longevity of the motors [47].
- Battery (Lead Acid): Lead-acid batteries have been a staple in the EEE325 course due to their cost-effectiveness. A typical deep-cycle lead-acid battery delivers between 100-200 cycles before a gradual decline begins, necessitating replacement when the capacity drops to 70-80% [48]. Notably, over 99% of lead-acid batteries are recycled, underscoring their environmental friendliness [49].
- Electronics: The lifespan of electronics has seen a significant reduction over the years. Historically, electronics were expected to last around 40 years. However, by the 1990s, this expectancy was halved. In the present day, most electronics have a lifespan ranging from 1.5 to 13 years, with an average of 4.5 years [50]. Factors such as rapid technological advancements and consumer preferences play a role in this reduced lifespan. Finally

open-source projects, “accelerates the transition” to sustainable solutions [50.1]

Note about this project’s maintenance and durability Concerns: The projected maintenance costs are influenced by the inherent durability of the components and external factors. One significant external threat to the project's longevity is potential damage by future students. As such, protective measures and user guidelines are recommended to mitigate this risk.

D. Software design:

The development of the software for this project was systematically approached in three distinct stages: **Configuration of the robot side, ROS2 workspace, Sensor Integration, and External Desktop ROS2 development.**
Software related to the robot is stored in [46]. Software related to the external computer, which holds all navigation and control aspects stored in [47]. Below is a detailed breakdown of the methodology employed in each stage:

1. Configuration of the ROS2 Workspace

The configuration of the ROS2 workspace was a pivotal stage and was executed at various intervals throughout the project. The methodology for this stage encompassed:

- Arrangement of Packages: Existing packages were organized in a structured manner to ensure compatibility and ease of access.
- Launch Files Functionality: The launch files were tested with the new configuration to ascertain their proper functioning.
- Creation of TF2 static publishers: These transform the ultrasonic and lidar frame of reference to the robot “base_link” frame of reference.

Debugging Nodes: Nodes that were launched and visible, but faced communication issues, were debugged. These nodes, once launched, became immutable, necessitating rigorous debugging.

2. Sensor Integration

The integration of ultrasonic sensors was a crucial component of the project. The methodology for this stage was:

- Configuration and Testing: The ultrasonic sensors were configured and tested using the official DFRobot GitHub repository [51]. The settings adjusted included:
    - Polling Rate: Set at 10Hz for optimal data collection.
    - Address Configuration: Sensors were labelled from “Ultraa” (Address: 0x20) to “Ultrah” (Address: 0x27) for clarity.
    - External Temperature Estimation: The temperature was estimated at 20.2°C, as measured in the lab.
    - Mode Selection: The short-range mode was chosen to optimize regions around the robot.
    - Sensitivity: Set at the maximum value of 10 (0x1-10) for heightened responsiveness.

Publisher Node Development: A unique publisher node was developed that takes in the required address and automatically assigns a name based on the address. This approach circumvented ROS2's aversion to numbering systems while maintaining a consistent and understandable naming convention. The decision to use individual publishers, as opposed to a single publisher, ensured a clear distinction between publishers and the sensor data they disseminate. Additionally, this node was equipped with a warning system to indicate the frequency of sensor unavailability, facilitating timely diagnosis of connection issues.

Note on Issues Encountered:
Ultrasonic Sensor Mode: Despite the physical bit changing and the wiki's indication of "Save when powered off, take effect after restarting", the modes of operation did not change. It was necessary to delve into the code to rectify this issue. It was observed that the author might not have utilised the short range, leading to this oversight.

3. External Desktop development

The virtual Linux environment was continually adjusted to cater to the evolving needs of the project. Alongside this, navigation was enhanced with the integration of various accessories. The methodology for this stage was iterative, with regular feedback loops ensuring that the environment was optimized for the project's requirements.

The LIDAR has an associated open-source GitHub from the SLAMTEC manufacturers [35] allowing for laser scan messages to be published from their created node onto the “scan” topic directly from data coming in from the port. This has helped towards the RNOR requirement.

SLAM Toolbox Integration: After downloading the SLAM toolbox, it was run in online asynchronous mode. This ensures that it can operate in real-time, efficiently despite potential delays in sensor readings. This toolbox generates a ROS node that subscribes to laser scan and odometry topics and subsequently publishes from map to odometry frame.

Utilisation of External Costmap Plugin, the range sensor plugin for NAV2 involves initialisation of a node that takes in a range message that comes with associated settings such as angular size and max distance possible and outputs a “LETHAL” cost onto the costmap to ensure the robot cannot navigate into it. This plugin is taken and heavily altered to ensure fine-grain control over the expected cone. This project intends to be able to use this plugin to reduce noise in posted measurements and to create known geometry in both global and local cost maps.

Configuration Challenges:
The process in which to create this plugin is sparsely documented and nuanced which required trial and error to implement both code and create the configuration yaml file. The inflation layer, which expands the local costmap, presented some challenges. It was observed that the order of configuration settings was pivotal for its proper functioning. The NAV2 documentation did not provide clear guidelines on this, making the configuration process a bit challenging.

The final robot for comparison to Figure 1 is shown in Figure 11.

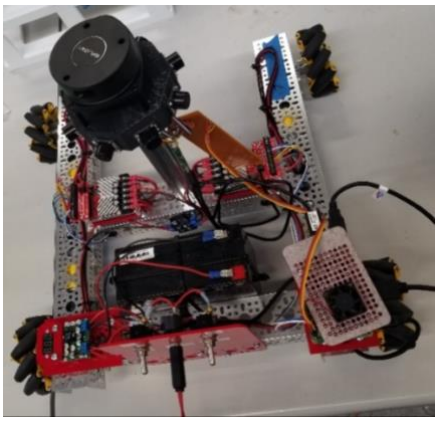

Figure 11, the final robot hardware, in this image you can see the ultrasonic mount bolted to the aluminium pillar. It has an array of eight ultrasonics surrounding it, with the RP A1M8 LIDAR screwed in at the top. Next to the pillar are the motor drivers and Veroboard for connecting all eight sensors to the 5V, GND, Tx and Rx pins on the Raspberry Pi. The robot has a pull E stop backboard with separate switches controlling power to the motors and labelled main power.

III. Evaluation

To evaluate the created robot platform, sensor evaluation of the ultrasonics, and three forms of navigation and detection tests have been devised, linear motion accuracy, nominal object avoidance and complex environment navigation. All three tests are demonstrated in Figure 12.

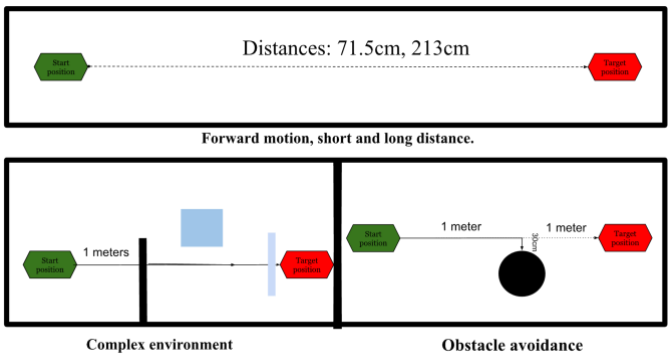

Figure 12, all three planned tests, tasking the robot in simulation to stop at prescribed distances. The obstacle avoidance to understand ultrasonics effect on normal navigation and finally, the complex environment where the robot is tasked with navigating around an opaque wall, partially transparent objects and stopping in front of an ITA object where the target position will be on top of this object.

A. Ultrasonic characterisation:

The ultrasonics have been noted in the literature as noisy and prone to error [30][31] as a result, a characterisation of such a sensor is required to ensure it can enable the robot to meet its ACRY requirement. Thus, under 3% range accuracy is needed for objects close to their minimum detection distance.

A 3D cut acrylic structure as shown in Figure 13 was developed to measure set distances away from the ultrasonic. The assumption was made that there would be no reflections back from the structure, ensuring measurement integrity. Testing this structure once assembled proved that this assumption was not congruent with reality, and thus alterations to the structure setup were made to place instead in the vertical plane, allowing the slots designed for the structure to reach out further than gravity would allow in the horizontal plane as slots were pulled down into place, thus ensuring measurement integrity from interference from the structure.

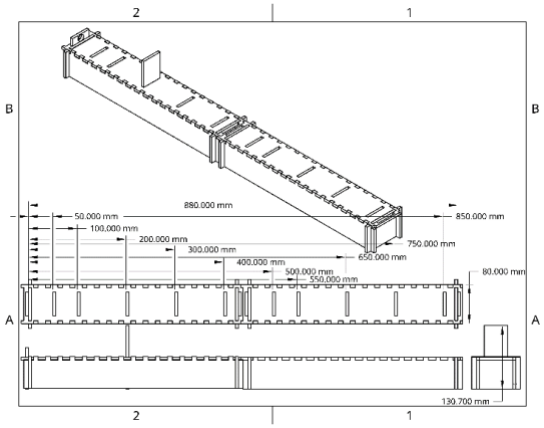

Figure 13, an assembled drawing of acrylic structures used to characterise the URM13. The slots within the structure allow multiple distances to be tested as accurately as the laser cutter can create, which is typically mm levels of accuracy.

Results:

Figure 14 shows every result within the expected range for the ultrasonics is within 1cm of the actual distance, meeting 3% accuracy requirements easily.

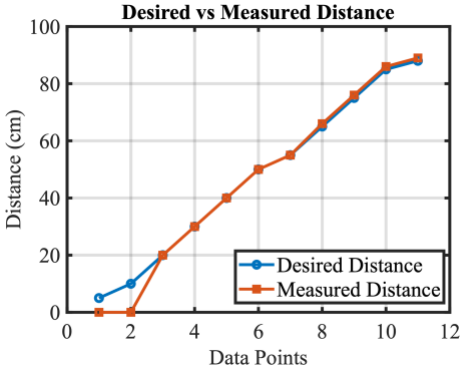

Figure 14, Distance to object vs measured distance on ultrasonics using the vertical setup from ranges 5-88cm.

The implication of this is we will be able to measure cones around the robot with 1cm accuracy every 0.01 seconds, this will help to fill requirements ACRY, DTRAN.

B. Linear motion accuracy test:

To fill ACRY requirements the robot must in linear motion display sub means for every 1 metre of movement we would expect to see 3cm deviation in one direction. To test this physical datums were marked on the floor using a straight wooden structure shown in Figure 15, measurements at 71.5cm, and 213cm were marked. The robot's front square edge was positioned using another straight wooden object against the datum orthogonal edge of the tape shown in Figure 16. The robot was tasked to navigate to the prescribed location, within the RVIZ2 and results relative to the datum object were measured.

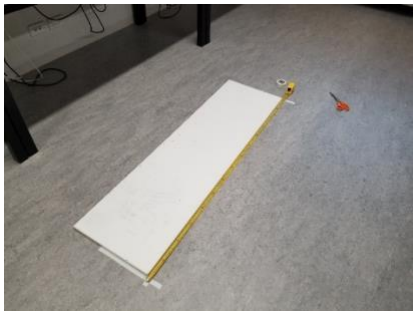

Figure 15, the straight wooden structure used to measure out the required distance needing to be traversed by the robot.

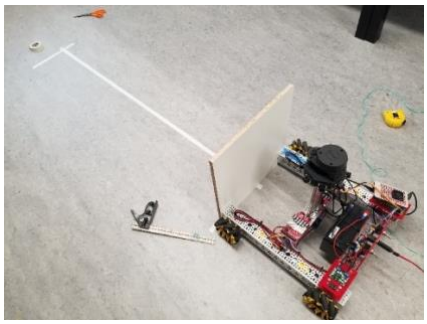

Figure 16, the alignment of the robot to datum points using another wooden structure.

Results:

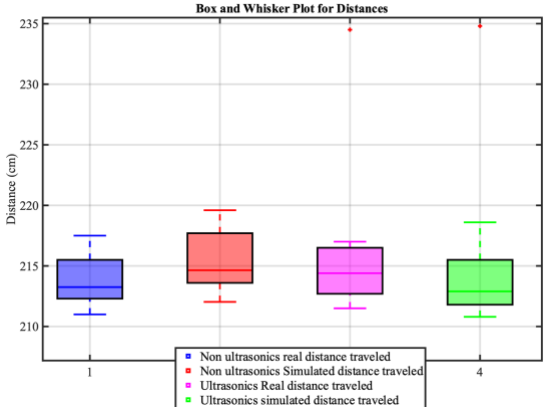

Figure 17 is a box and whisker graph of the measured distance travelled in simulation and reality when ultrasonics are enabled and disabled when the robot is tasked with moving 213cm.

Distance away from 213cm (2 S.F), shown in Figure 17:

Non-Ultrasonics: mean 0.7cm, variance 4.33, all data points are within 3% of the required distance.

In the simulation, Non-Ultrasonics: mean 2.37 cm, variance 6.58, 90% are within 3% of the required distance.

Ultrasonics mean 3.31 cm, variance 7.25, 90% are within 3% of the required distance.

In the simulation, Ultrasonics: mean 2.42 cm, variance 51.7, 90% are within 3% of the required distance.

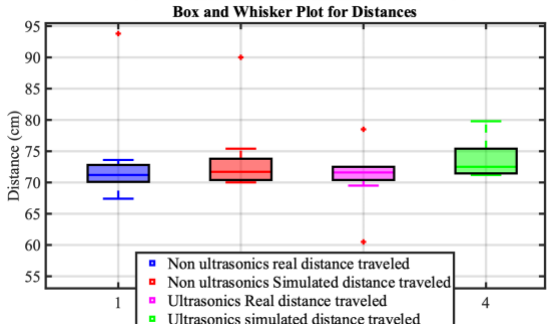

Figure 18 is a box and whisker graph of the measured distance travelled in simulation and reality when ultrasonics are enabled and disabled when the robot is tasked with moving 71.5cm.

Distance away from 71.5cm (2 S.F), shown in Figure 18:
Ultrasonics mean -0.460 cm, variance 19.5, 80% are within 3% of required distance.

Non-Ultrasonics: mean 1.72 cm, variance 55.4, 70% are within 3% of required distance.

In the simulation, Ultrasonics: mean 2.12 cm, variance 7.80, 80% are within 3% of the required distance.

In the simulation, Non-Ultrasonics: mean 2.25 cm, variance 51.7, 60% are within 3% of the required distance.

Discussion:
During testing the controller introduced bias so that when tasked with navigating to any position it would always be off in simulation and reality by 22cm. Accounting for this by adding 22cm to all my desired distances appeared to correct this.

About the ACRY requirements, the vast majority of data points are within the 3% margin required aside for the 71.5cm simulation measurements with no ultrasonics. One interesting finding is that all results in the simulation show a mean of around 2cm away from the desired target, this could be due to the 22cm bias implemented being incorrect when calibrating initially. Accounting for this bias would shift the non-ultrasonics, and simulated distance into the desired 3% range and allow for 80% of measurements taken to be within the required 3% range. This however can likely be improved by stopping distance and increasing the frequency of controller NAV2 updates which is typically [find reference]. Thus, this should not discourage open-source robotics projects and Clutterbot from adopting this scheme.

C. Nominal object avoidance test.
The object avoidance test is used to check the time it takes to navigate around an ITA object, this is useful to ensure that ultrasonics are not a harmful addition due to their inherent noise. The test will be conducted by placing the outermost edge of an object 13cm away from the centre line and tasking the robot to navigate to the end of the line. The time taken and path used will be analysed to characterise how ultrasonics affect navigation.

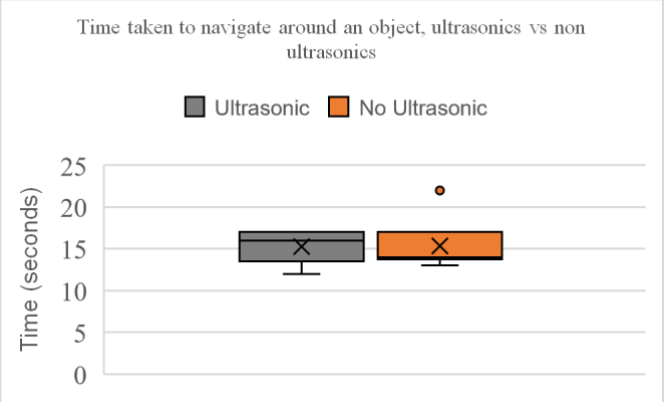

Figure 19 is a box and whisker graph of the time taken to navigate around an object when ultrasonics are enabled and disabled.

Time taken to navigate around an object, as shown in Figure 19:
Ultrasonics enabled mean 15.2 seconds, std 1.86
No Ultrasonics enabled, 15.3 seconds, std 2.75

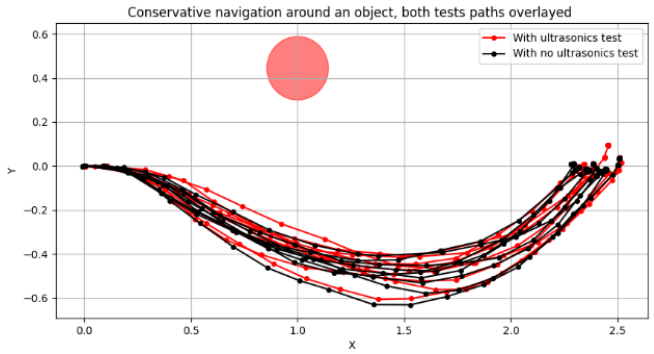

Figure 20, plotted paths taken in metres on the x and y axis. The red dot indicates the actual position and size of the object.

Discussion:
The results shown in Figure 19 and Figure 20 clearly show that based on the time and path there is no significant difference between navigation in ultrasonics vs no ultrasonics enabled for basic obstacle avoidance when the inflated region is increased. This means that there is no downside for simplistic navigation tasks in my setup, if able to detect ITA objects, then it makes for a compelling case to add to existing or in development-systems using LIDAR only.

D. Complex navigation test:

To thoroughly test the improvements of the ultrasonics about requirements DOUT and DTRAN, we must navigate the robot with ultrasonics enabled and disabled to show the relative speed of the robot, per cent completion of the course and ability to detect objects accurately in the simulation. The tests involve placing a wall with a gap, a partially transparent object and an ITA object in the robot's way and tasking it to navigate in a straight line into and past all objects.

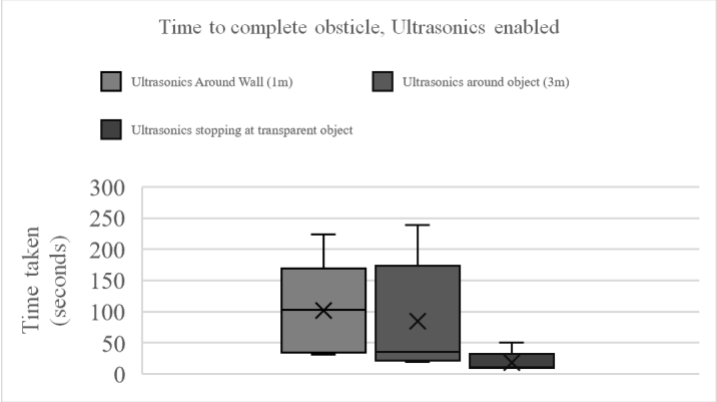

Figure 21 is a box and whisker graph of the time to complete each part of the complex environment test.

Time taken to navigate around each object, as depicted in Figure 21:
Navigation around the wall, mean 102.08 seconds, std 77.7.
Navigation around the object mean, 84.92 seconds, std 93.07.
Navigation up to and stopping at ITA object, 18.92 seconds, std 17.4.

One hundred per cent completion of the course by the designed robot for all runs, Figure 22 gives an image of one of the paths taken visualised in RVIZ2. Figure 23 gives an example of all paths taken with ultrasonics enabled.

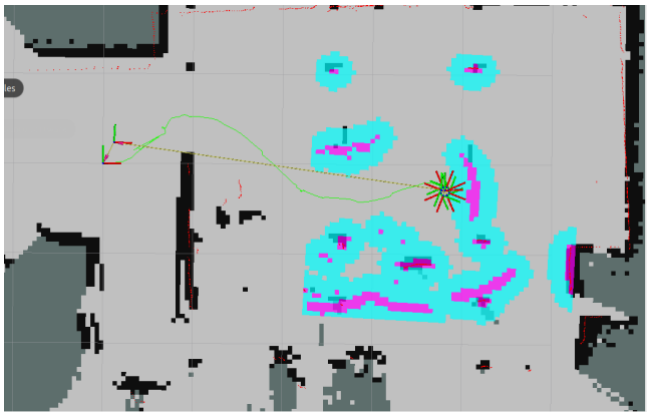

Figure 22, course complex navigation, ultrasonics enabled. The path is indicated by the green line. The robot is stopped in front of a detected transparent object.

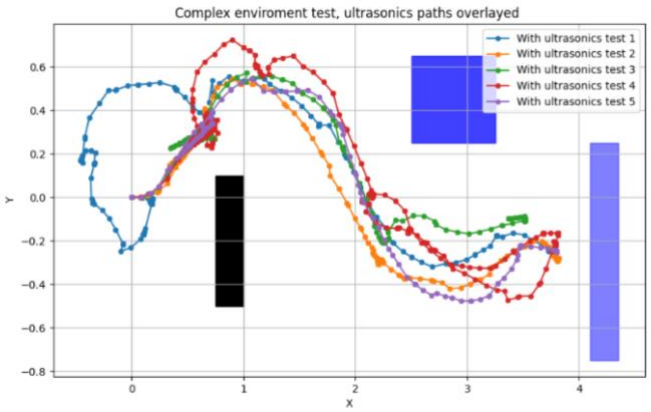

Figure 23, this graph shows a visualisation of all paths taken during tests with ultrasonics enabled. The X and Y axes are labelled in metres.

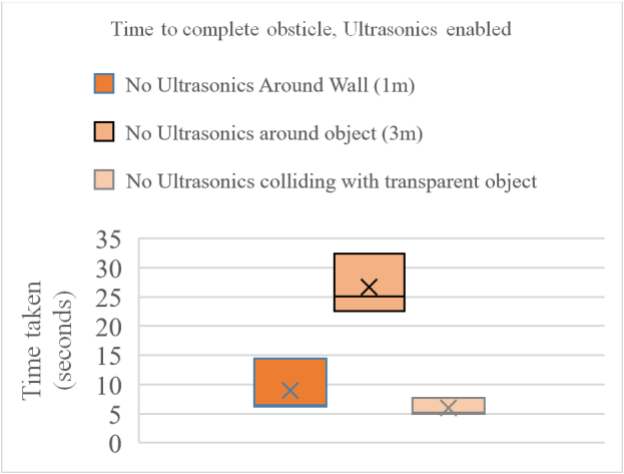

Figure 24, the time taken for the robot to navigate around each obstacle with ultrasonics disabled.

Time taken to navigate around each object, as depicted in Figure 24:
Navigation around wall, no ultrasonics, mean 9, std 4.67 it is a factor of 11x faster than with ultrasonic enabled.
Navigation around object, no ultrasonics, mean 26.7 seconds, std 5.1, it is a factor 3x faster than with ultrasonic enabled.
Collision with ITA object, mean 5.96, std 1.5, factor 3x faster than with ultrasonic enabled.
Two out of three objects were completed of course by the designed robot for all runs, before the failure criteria of being within 5 cm of the ITA object.

Discussion:
In summary, while the ultrasonics did introduce a time overhead, they significantly enhanced the robot's object detection accuracy and course completion rate, underscoring their importance in complex navigation scenarios. The detection and navigation around objects shown in Figure 23 was successful, thus requirements ACRY, DOUT, and DTRAN were shown to be filled. This navigation slowdown when undertaking tests is due to the controller's inability to cope with added information, having to take considerable time to recalculate the path needed if even the slightest bit of added information coincides with the path. This recalculation is further exacerbated by the arc added into the cost map to show where the object could be, which takes time to disappear as the probability of the object being there decreases due to lack of sensor detection. This time delay could likely be reduced if more sensors are added or additional methods to utilise the existing sensors are implemented.

IV. Conclusion

In the rapidly evolving field of autonomous robotics, the precision and reliability of sensory data play a pivotal role in determining the robot's effectiveness. While LIDAR has become an indispensable tool for environmental mapping, its inherent limitations, such as the inability to detect transparent or infrared-absorbing objects, have posed challenges.

Addressing this, the research introduced a groundbreaking system that synergistically combines high-frequency LIDAR with low-frequency ultrasound sensors. This integrated system, underpinned by the Robot Operating System (ROS2) and other advanced software components, has showcased no negative effects in basic navigation and linear and basic navigation tests, shown marked however has shown marked improvements in course per centage completion especially in environments with previously undetectable objects as shown in the complex environment test.

This new improved system however came at a cost of increased navigation time when introducing sensor data, which as mentioned can be improved with adjustments in either the method of navigation, global path planning or increased sensor density. Through testing successes and the chosen design of the robot, all requirements have been justifiably completed.

On a broader scale, improving robot efficiency, especially in routine tasks, can lead to better productivity and resource use. As technology continues to evolve, innovations like autonomous robotics have the potential to offer more efficient solutions for global challenges. This aligns with the global push towards sustainable practices, as highlighted by the United Nations' Sustainable Development Goals (SDGs). The Fourth Industrial Revolution, characterised by innovations like autonomous robotics, stands as a testament to the potential of technology in driving sustainable and efficient solutions for global challenges.

V. FUTURE WORK

This project may be extended in many facets, one of which includes storing ultrasonic data and robot position so that an algorithm may be applied in an attempt to reduce the error associated with large distances from the ultrasonic sensor during detection. This algorithm may be based on the simplistic triangular-based fusion [52] or more advanced recent works. Another factor could be adding behaviour trees to help move the robot rotationally helping to add information before the path planner in NAV2 activates, thus removing delays associated with path recalculation after new information is added.

REFERENCES

[1] "Autonomous Robots for Services—State of the Art, Challenges, and Research Areas," Sensors, [Online]. Available: https://www.mdpi.com/1424-8220/23/10/4962

[2] "Path Planning for Autonomous Mobile Robots: A Review," Sensors, [Online]. Available: https://www.mdpi.com/1424-8220/21/23/7898

[3] S. Bi, "A Survey of Low-Cost 3D Laser Scanning Technology," MDPI, vol. 11, no. 9, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/9/3938.

[4] Y. Li, " Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems" in IEEE, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9127855/.

[5] ClutterBot, [Online]. Available: https://www.clutterbot.com/

[6] A. Ghosh, S. Iyengar, S. Lee, A. Rathore, and V. N. Padmanabhan, "Streaming Video Analytics On The Edge With Asynchronous Cloud Support," arXiv preprint arXiv:2210.01402, 4 Oct. 2022.

[7] EEEN325 Robotic Engineering. (2022). Motor Control Session, Laboratory Session 1: Introduction. Victoria University, Wellington, New Zealand.

[8] "THE 17 GOALS," Sustainable Development, [Online]. Available: https://sdgs.un.org/goals

[9] M. Trane, L. Marelli, A. Siragusa, R. Pollo, and P. Lombardi, "Progress by Research to Achieve the Sustainable Development Goals in the EU: A Systematic Literature Review," Sustainability, vol. 15, no. 9, p. 7055, Apr. 2023, doi: 10.3390/su15097055.

[10] J. D. Sachs, G. Schmidt-Traub, M. Mazzucato, D. Messner, N. Nakicenovic, and J. Rockström, "Six Transformations to achieve the Sustainable Development Goals," Nature Sustainability, vol. 2, no. 9, pp. 805–814, 2019. [Online]. Available: https://doi.org/10.1038/s41893-019-0352-9

[11] R. M. Oosthuizen, "The Fourth Industrial Revolution – Smart Technology, Artificial Intelligence, Robotics and Algorithms: Industrial Psychologists in Future Workplaces," Frontiers, 2022. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frai.2022.913168/full

[12] "Economic Surveys: New Zealand 2017," OECD iLibrary, 2017. [Online]. Available: https://www.oecd-ilibrary.org/sites/eco_surveys-nzl-2017-6-en/index.html?itemId=/content/component/eco_surveys-nzl-2017-6-en.

[13] A. Barker, "Improving productivity in New Zealand's economy," OECD iLibrary, 2017. [Online]. Available: https://www.oecd-ilibrary.org/sites/eco_surveys-nzl-2017-6-en/index.html?itemId=/content/component/eco_surveys-nzl-2017-6-en.

[13.1] "Open Robotics," [Online]. Available: https://www.openrobotics.org/. [Accessed: 15-Oct-2023].

[13.2] "MDPI Article on LIDAR in Agriculture," [Online]. Available: https://www.mdpi.com/2624-8921/4/3/47/pdf. [Accessed: 15-Oct-2023].

[13.3] "Online Library Article on Robot Regulation," [Online]. Available: https://onlinelibrary.wiley.com/doi/pdf/10.1002/aepp.13177. [Accessed: 15-Oct-2023].

[14] [Online]. "ROS Documentation," ROS.org, Available: https://docs.ros.org/en/rolling/index.html.

[15] [Online]. "ROS Navigation Documentation," ROS.org, Available: https://navigation.ros.org/.

[16] Author(s), "Title of the repository," GitHub, Year. [Online]. Available: URL. [Accessed: 15-Oct-2023]..

[17] ROS 2 Documentation: Foxy. (2020). [Action-SingleActionClient.gif].

[18] BotBuilder. (2023, 5). ROS2 Basics #4 - Understanding ROS2 Executables and Nodes [Video]. YouTube. https://www.youtube.com/watch?v=aeOS9xqblrg https://docs.ros.org/en/foxy/_images/Action-SingleActionClient.gif

[19] "How-To-Guides: Topics, Services, Actions," ROS 2 Foxy documentation, [Online]. Available: https://docs.ros.org/en/foxy/How-To-Guides/Topics-Services-Actions.html. [Accessed: 15-Oct-2023].

[20] "Topics vs. Services vs. Actions in ROS2-Based Projects," Automatic Addison, [Online]. Available: https://automaticaddison.com/topics-vs-services-vs-actions-in-ROS2-based-projects/. [Accessed: 15-Oct-2023].

[21] "Creating custom msg and srv files," ROS 2 Documentation, Foxy. [Online]. Available: https://docs.ros.org/en/foxy/Tutorials/Beginner-Client-Libraries/Custom-ROS2-Interfaces.html.

[22] "About Composition," ROS 2 Documentation, Foxy. [Online]. Available: https://docs.ros.org/en/foxy/Concepts/About-Composition.html.

[23] "tf2 - ROS Wiki," ROS Wiki. [Online]. Available: http://wiki.ros.org/tf2.

[24] "Configuring Costmaps," ROS Navigation. [Online]. Available: https://navigation.ros.org/configuration/packages/configuring-costmaps.html. [Accessed: 15-Oct-2023].

[25] "costmap_2d," ROS Wiki. [Online]. Available: http://wiki.ros.org/costmap_2d. [Accessed: 15-Oct-2023].

[26] "Setting Up Navigation Plugins — Nav2 1.0.0 documentation," ROS. [Online]. Available: https://navigation.ros.org/setup_guides/algorithm/select_algorithm.html?highlight=uses. [Accessed: 15-Oct-2023].

[27] "Rviz2 · User Manual," Turtlebot4. [Online]. Available: https://turtlebot.github.io/turtlebot4-user-manual/software/rviz.html. [Accessed: 15-Oct-2023].

[28] "costmap_2d," ROS Wiki. [Online]. Available: http://wiki.ros.org/costmap_2d. [Accessed: 15-Oct-2023].

[29] H. Wei et al., "Multi-sensor Fusion Glass Detection for Robot Navigation and Mapping," 2018 WRC Symposium on Advanced Robotics and Automation (WRC SARA), Aug. 2018. DOI: 10.1109/WRC-SARA.2018.8584213.

[30] A.-T. Nguyen and C.-T. Vu, "A study and design of localization system for mobile robot based on ROS," arXiv preprint arXiv:2109.05551, 2021. arXiv:2109.05551 [cs.RO].

[31] T. N. Yap and C. R. Shelton, "SLAM in large indoor environments with low-cost, noisy, and sparse sonars," in 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 2009, pp. 1395-1401. DOI: 10.1109/ROBOT.2009.5152192.
[32] "Wholesale 2D LIDAR on AliExpress," [Online]. Available: https://www.aliexpress.com/w/wholesale-2D-LIDAR.html. [Accessed: 15-Oct-2023].
[33] E. Dans, "The Incredible Shrinking LiDAR," Forbes, 11 Sept. 2020. [Online]. Available: https://www.forbes.com/sites/enriquedans/2020/09/11/the-incredible-shrinking-lidar/. [Accessed: 15-Oct-2023].
[34] "Tesla saves an estimated $114 per car by removing USS," Teslarati, [Online]. Available: https://www.teslarati.com/tesla-114-removing-uss/. [Accessed: 15-Oct-2023].
[35] Slamtec, "RPLIDAR ROS package," [Online]. Available: https://github.com/Slamtec/rplidar_ros. [Accessed: 15-Oct-2023].
[36] "The RPLidar A1M8 - 360 Degree Laser Scanner," [Online]. Available: https://www.robotshop.com/products/rplidar-a1m8-360-degree-laser-scanner-development-kit. [Accessed: 15-Oct-2023].
[37] "UST-05LN :: Sentek Hokuyo," Hokuyo-USA. [Online]. Available: https://hokuyo-usa.com/products/lidar-obstacle-detection/ust-05ln. [Accessed: 15-Oct-2023].
[38] "LD06 Lidar portable 360 degree DTOF Laser Sensor Scanner Kit with 12m range for ROS robot," AliExpress, [Online]. Available: https://www.aliexpress.com/item/1005003540394166.html. [Accessed: 15-Oct-2023].
[39] DFRobot, "Fermion: URM13 High Sensitivity Ultrasonic Distance Sensor Breakout (15~900cm, I2C / UART / PULSE)" DFRobot. [Online]. Available: https://www.dfrobot.com/product-2161.html. [Accessed: 15-Oct-2023].
[40] Cytron Technologies, "3V-5.5V SR04P Ultrasonic Ranging Module," Cytron.io. [Online]. Available: https://www.cytron.io/p-3v-5.5v-ultrasonic-ranging-module. [Accessed: 15-Oct-2023].
[41] A. K. Das, "Nvidia Jetson AGX Xavier vs Raspberry Pi 4 Benchmark," Arnab Kumar Das. [Online]. Available: https://www.arnabkumardas.com/topics/benchmark/nvidia-Jetson-agx-xavier-vs-raspberry-pi/. [Accessed: 15-Oct-2023]
[42] NVIDIA, "Jetson AGX Xavier Series," NVIDIA. [Online]. Available: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/Jetson-xavier-series [Accessed: 15-Oct-2023].
[43] Raspberry Pi, "Raspberry Pi 4 Model B specifications," Raspberry Pi. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/. [Accessed: 15-Oct-2023].
[44] K. Zheng, G. Chen, G. Cui, Y. Chen, F. Wu, and X. Chen, "Performance Metrics for Coverage of Cleaning Robots with MoCap System," in Lecture Notes in Computer Science, vol. 10464, 2017. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-65298-6_25. [Accessed: 15-Oct-2023].
[45] R. Hurnen, "ENGR489_Navigation," GitHub repository, Year. [Online]. Available: https://github.com/riththurn/ENGR489_Navigation. [Accessed: 15-Oct-2023].
[46] R. Hurnen, "ENGR489_UltrasonicNavigation," GitHub repository, Year. [Online]. Available: https://github.com/riththurn/ENGR489_UltrasonicNavigation. [Accessed: 15-Oct-2023].
[47] "Fisher & Paykel Technologies. 'How long does a motor last for? How do you make a motor last longer?'. Fisher & Paykel Technologies Knowledge Hub, [Online]. Available: https://www.fisherpaykeltechnologies.com/knowledge-hub/how-long-does-a-motor-last-for-how-do-you-make-a-motor-last-longer. [Accessed: 15-Oct-2023]."
[48] BU. 'BU-804: How to Prolong Lead-acid Batteries'. Battery University, [Online]. Available: https://batteryuniversity.com/article/bu-804-how-to-prolong-lead-acid-batteries. [Accessed: 15-Oct-2023].
[49] BCI. 'New Study Confirms Lead Batteries Maintain 99% Recycling Rate'. Battery Council International, [Online]. Available: https://batterycouncil.org/new-study-confirms-lead-batteries-maintain-remarkable-99-recycling-rate/. [Accessed: 15-Oct-2023].
[50] "What's the Average Lifespan of Your Electronics?," Quantum Lifecycle. [Online]. Available: https://quantumlifecycle.com/en_CA/blog/whats-the-average-lifespan-of-your-electronics/. [Accessed: 15- Oct- 2023].
[50.1] [1] Open Source, "Open source sustainability," OpenSource.com, 2023. [Online]. Available: https://opensource.com/article/23/1/open-source-sustainability
[51] DFRobot, "DFRobot_URM13," GitHub, 2023. [Online]. Available: https://github.com/DFRobot/DFRobot_URM13. [Accessed: 15- Oct- 2023].
[52] 0. Wijk and H. I. Christensen, "Triangulation-Based Fusion of Sonar Data with Application in Robot Tracking," IEEE Trans. on Robotics and Automation, vol. 16, no. 6, pp. 740-752, 2000.