Faculty of Engineering

Implementation of Software System on Robotic Platform for Autonomous Navigation

Guelph, Ontario

Prepared by:
Shilpan Shah
Student ID: 20603389
2B Mechatronics Engineering
Monday May 8th, 2017

46 Shawbrooke Rise SW
Calgary, Alberta
T2Y 3A3

May 8, 2017

William W. Melek, Director of Mechatronics Engineering
University of Waterloo
Waterloo, ON, N2L 3G1

Dear Professor Melek,

This report, entitled "Implementation of Software System on Robotic Platform for Autonomous Navigation" is the first work report that I have written for my 2A work term. The information presented in the report is based upon my recently completed work term with Linamar Corporation.

Linamar Corporation is Canada's second largest automobile parts manufacturer. Manufactures and supplies automotive parts to numerous automotive companies worldwide. I was situated in the BOLD Innovations team, which is responsible for projects involving system integration, improving process flow, and prototyping new products. During my time at Linamar Corporation, I was a part of the prototyping engineering team. My responsibilities included designing, prototyping, and testing a robot capable of autonomous navigation within their plant.

The following report shows a detailed software and hardware components for a prototype robot capable of moving autonomously. The need for a robot was realized after the company needed to streamline their workflow and increase efficiency by reducing time and resources. My scope of work included software and hardware configurations and writing code. With project management and integration of software, I had to decide on different designs and testing methods.

I would like to thank my manager, Leigh Copp, who defined the purpose of the project and assisting me in hardware configurations for the robot. I hereby confirm that I received no help, other than what is mentioned above, in writing this report. I also confirm that this report was written entirely by me and has not received any previous academic credit at this or any other institution.

Sincerely,

Shilpan Shah
Student ID: 20603389

Table of Contents

List of Figures

List of Tables

Summary

The following report aims to implement and develop robot with software capabilities for autonomous navigation. Linamar Corporation manufactures high precision automobile parts that is now interested in autonomous mobilization in manufacturing sector. The purpose of this project included prototyping a robot that can handle material flow autonomously which is affordable and increases efficiency on the floor. In a short duration, an open source software platform was needed for mapping and general navigation with obstacle avoidance system.

After researching different robotic platforms, 3 robot base's were narrowed and analyzed. The three robots were judged based on software development and programmability. Other criteria included data communication, expandability, durability and safety of the robot. Mechanical and software compatibility had to match with the chosen robot for developing a prototype that can carry payload in fast speeds. The result was iRobot Create 2 robot that was chosen for implementation through engineering analysis. The implementation process involved using Simultaneous Localization and Mapping (SLAM) technique with the navigation stack available with Robot Operating System(ROS). ROS was also compatible with LiDAR laser and Raspberry Pi microcontroller to update position of robotic base as it navigated around the warehouse.

It was concluded that Create 2 robot was the right solution with ROS to perform autonomous navigation within plant. This prototype was tested and showed 200% production efficient while being $900 under budget. Future development includes creating internal application to track robot movement and send notifications as goods move within plant.

1.0 Introduction

Autonomous navigation in society has caused a huge shift in mindset in what the possibility of transportation might hold for the future. This has led to increase investment by automotive industry and software companies to collaborate in creating technology that can create the future car. Autonomous navigation has been applied to manufacturing which has led to creating start-ups that focus on providing autonomous mobility. Automated transportation has changed how goods and objects move internally, providing efficiency and increasing volume.

Linamar is a global manufacturing company that operates over 58 facilities around the world. The company makes high precision parts, assemblies, and castings for the auto industry. Manufacturing components include engines, transmissions, drivelines, steering, suspension components, and brake components. The company is second largest automobiles parts manufacturer in Canada, providing support and advance systems parts for light vehicle, commercial truck, off-highway, energy and industrial OEM markets with powertrain system solutions [1]. The company also owns a Skyjack brand, which manufactures aerial work platforms to construction business. The company is diversifying into new markets, providing innovative solutions backed with a strong economic model for sustainability.

Linamar is now looking to use technology within their existing plants to improve process flow and efficiency. Linamar Corporation practices lean manufacturing and uses 5S principles for streamlining processes and increasing production. To be a participant of using technology to solve manufacturing problems, Linamar has decided to focus on solving internal movement of power tools within their plant. It has been noted that machine operators and other labours waste

lots of time in moving parts around inspecting for quality control. It has become Linamar's interest to transform this by investing in autonomous navigation and focussing on material flow to resolve bottlenecks to material handling.

1.1 Background

This section will go over how autonomous navigation is used by other companies in manufacturing environments and how autonomous navigation is used in real life applications. This is important to know when analyzing different mobile robots that have capable software for implementation.

Autonomous navigation has become very popular recently due to the impact it is having in manufacturing and assembly lines. Autonomous navigation goes beyond a factory level but into the real world, affecting real lives and saving time for millions of people. Companies such as Tesla Motors, and Uber have created a disruption in the transportation industry that is transforming the way people move. Due to this massive surge in demand and push in the technology industry, autonomous navigation is the one force that will help everyone, regardless of disability or physical movement challenges. Large corporations along with start-up companies are making an impact in their community and helping to thrive the economy. Such company includes Seegrid and Clearpath Robotics.

Seegrid has created autonomous forklifts and vision guided vehicles that is used by local manufacturing companies to do auto navigation within plant. This company uses vision guidance from different cameras and sensors to localize and move accordingly. Through their internal supervisor-software product, they can use web applications installed as virtual machines

which can communicate with multiple Vision Guided Vehicle's (VGV) over Wi-Fi. This will allow for dynamic notifications, customized operations, vehicle/work tracking and dispatching systems [2].

Another example of a company that has developed a robot platform to carry parcels and other objects within warehouse is Clearpath Robotics. With their recent launch of product OTTO Motors, they can use inbuilt safety LiDAR's and other sensors to carry 1500 kg of payload with top speed of 2 m/s [3]. This robot is durable, made with metal construction that is capable of self-driving with mapping and auto localization features. Through intelligent path planning and without the need to modify existing infrastructure, a network of mobile robots can be used simultaneously to perform different tasks [3].

The need for self-driving vehicles have become vital to achieving the company's futurist goals and production lines. For robotics to create that vision of factory of the future, Linamar and the innovation team must find a solution to meet growing demands of manufacturing precision auto parts and create automatic mobile platform that can achieve complex tasks.

1.2 Objective

The objective of the report is to implement a software system (ROS) on a robotic mobile platform that is capable of handling autonomous navigation within warehouse. This project must be completed within short time frame with the most cost effective solution. Robot Operating System (ROS) has been identified as the open source software implementation choice since it will allow for mapping and navigation in a relative short time frame, through inbuilt functionalities and open source algorithms.

1.3 Robot Operating System (ROS)

“ROS is an open-source, meta-operating system for your robot. It provides services including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management [4].” ROS contains libraries, and other data management tools that allow for communication between multiple computers and devices. This means code can be written and run across multiples devices when connected with ROS.

All ROS software is organized into packages. A ROS package is a coherent collection of files, generally including both executables and supporting files, that serves a specific purpose. For all the files to work and package to execute properly, a collection of small, mostly independent programs called nodes must run at the same time. Nodes must communicate with each other for the package to be executed. ROS master facilitates this communication.

“A running instance of a ROS program is called a node. A node is a process that performs computation. [5[.” Example of nodes include, laser finder sensor, moving robotic arm independently of other external attachments, performing localization, path planning, etc.

“Topics are named buses over which nodes exchange messages. Topics have anonymous publish/subscribe semantics, which decouples the production of information from its consumption [6].” Nodes subscribe to topic which are relevant to its operation, and generate data to publish to the. There can be many publishers and subscribers to a topic. Certain nodes even activate a topic to send data to and from.

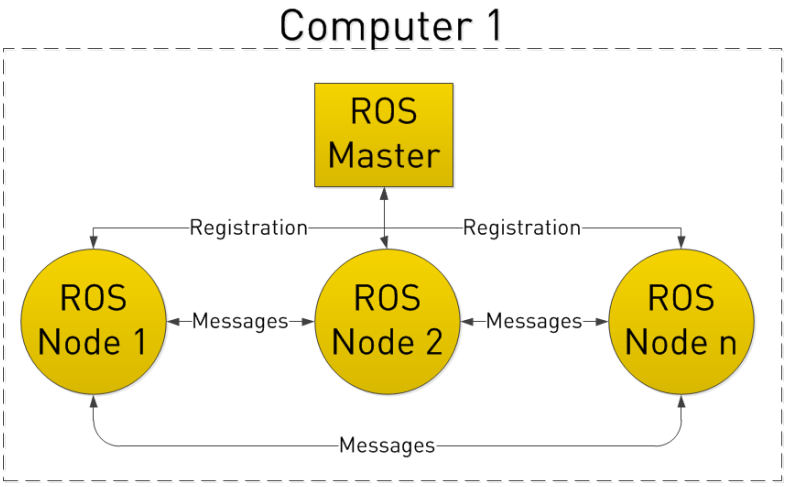

Figure 1: ROS Configurations and Workings [7].

Tf (transformation message) is a package designed to update multiple coordinate frames over time and let user keep trach of the robot's movement. "Tf maintains the relationship between coordinate frames in a tree structure buffered in time, and lets the user transform points, vectors between any two coordinate frames at any desired point in time [8]." **Figure 1** refers to the idea of sending data and messages between different nodes and updating topics.

2.0 Engineering Judgement and Analysis

This section will analyze 3 different robots to check if the criteria and constraints are met. This will be done with a help of a decision matrix to choose the best robot base to be implemented. LiDAR and Raspberry Pi were the other external hardware that is used with the robot to implement mapping and autonomous navigation.

2.1 Criteria

The robot base must have the following features with weighted criteria in percentage:

- **Software Development (35%)**: Open source framework (ROS) or equivalent that has built in functions and controls that is capable of mapping and navigation, given the limited resources and short duration of the project. The robot must be fully programmable and controlled using a software platform that will allow for customization and path planning. This accounts for a significant portion of the project, hence 35%.
- **Handshake/Instant Feedback (25%)**: There must be two-way communication between the robot and computer. This is critical for updating the maps based on the robot's odometery and other sensor data, hence 25%.
- **Expandability (15%)**: There should be room to connect external sensors with the robot and work simultaneously for building applications. To do this, microcontrollers are required for information processing and communication with external hardware. Microcontrollers act as the main device that transmits, receives and interprets data for proper functionality and operations. This is typically achieved using circuitry and connecting with I/O ports, which becomes important to check when deciding a robot to purchase, hence 15%.

- **Durability (15%)**: This is the ability to withstand pressure, strength, wear and tear. This was based on the materials used to build the robot and its physical capability for travelling long distances without part failure. In the initial prototype phase, this is not a significant factor, hence only 15% weightage.
- **Maximize Safety (10%)**: Need to make sure that the robot would not be a potential hazard has its moving in warehouse. Furthermore, there should be inbuilt sensors that can spot the obstacles and perform obstacle avoidance. Safety is important in the plant, however in initial prototype phase this is not the most challenging part to achieve, hence 10% weightage.

2.2 Constraints

Linamar Innovation team is a newly form division of Linamar Corporation with limited resources to work with:

- **Cost**: The project was allocated $2000 amount of money to work with and buy sensors and other equipment need for the prototype.
- **Payload**: Since the goal is for carry materials and parts around the warehouse, having a robot base that can carry at least 1.5 kg weight will allow for quick mobility within warehouse.
- **Battery Life**: Initial testing should last robot for minimum of 1 hour in full operation.
- **Time Frame**: Implementation of software within 4 months of CO-OP (mapping and basic autonomous navigation)

2.3 Potential Robotic Solutions

The equipment that was required to successfully make a prototype included a prebuilt robot base, sensors and microprocessor required for computing. To complete this, the company had to

compare and order parts from different suppliers around the world that had met specifications in terms of design and engineering for development of the project.

A direct comparison is made of the three robots (**Figure 2, Figure 3, Figure 4**) by evaluating the built-in features and comparing with the requirements of the project. **Table 1, 2, 3** shows the advantages and disadvantages for each robot, using an analytical approach.

Figure 2: Nexus Robot [9].

Table 1: Evaluation of Nexus Robot's Features

Advantages	**Disadvantages**
Handle large payloads (10 kg)	Not compatible with ROS framework
Can attach with any microcontroller boards	No mechanism to control motor and speed of robot via controller.
Obstacle avoidance system	Expensive (>$1000)
Visible in manufacturing plant environment	Max speed of 0.5 m/s

Figure 3: Hangfa Robot [10].

Table 2: Evaluation of Hangfa Robot's Features

Advantages	**Disadvantages**
Handle very large payloads (30 kg)	Not compatible with ROS framework
Easily visible in manufacturing plant and customizable	Programmable with only certain C language microcontrollers

Long battery life (>8 hr)	Low Speed (0.65 m/s)
Safety sensors built in	Very expensive (>$1500)

Figure 4: iRobot Create 2 Robot [11].

Table 3: Evaluation of iRobot Create 2 Robot's Features

Advantages	**Disadvantages**
Low cost(<$500)	Handle low payloads (2 kg)
Inbuilt sensors for obstacle avoidance	Low battery life (~3 hrs)
High Speed (2m/s) with load	Hard to spot in manufacturing plant due to low noise of motors and size
Works with most microcontrollers	
Able to have 2 way data communication to allow controlling motors and speed parameters	
Compatible with ROS framework	

2.4 Decision Matrix

Table 4: Decision Matrix

Criteria	**Weighting**	**Nexus Robot**	**Hangfa Robot**	**iRobot Create 2 Robot**
Software Development	0.35	3	3	5
Handshake/ Instant Feedback	0.25	3	3	4
Expandability	0.15	3	3	3
Durability	0.15	5	4	3
Safety	0.10	4	4	3
Total	1	3.40	3.25	3.95

Since the Create 2 Robot included open source ROS framework which allowed for navigation and path planning, it was given a weighting of 5, while the Hangfa and Nexus were given 3, due to manually coming up with code and algorithms that would otherwise be written from scratch and would not be able to finish within work term.

The Create 2 robot was given a weight of 4 since it was possible for data to send from robot to computer and send command back to robot. Hangfa and Nexus were not able to set parameters of the motors and control the robot from the computer. Hangfa and Nexus can only receive data but cannot change different parameters for motors and other sensors from code. Thus, they received score of 3 each.

All three robots can be attached to microprocessors, which then can be used for external hardware attachments. Hence, they were all given a weight of 3.

Nexus and Hangfa were given a weight of 4 and 5 since they were made of metal and wooden parts, making them more durable and increasing their strength so it can be used for longer period. The Create 2 Robot was rated 3 as it was made of primarily plastic materials.

The Nexus and Hangfa were weighted 4 since they are easily visible within warehouse and large enough to be spotted. As well, these robots have safety sensors built in to prevent major incidents and stop motors. Create 2 was weighted 3 due to small size and low noise it would make as its travelling. Thus, it was determined Create 2 robot is less prone to accidents and injuries.

Based on the decision matrix, it was concluded to purchase iRobot Create 2 mobile base that is fully programmable and compatible with Robot Operating System (ROS).

3.0 Implementation

The solution that was implemented was based on using Robot Operating System as software platform for development. The decision matrix showed that iRobot Create 2 was the only programmable robot that was supported on ROS and programmable for further development. This was a crucial decision based on given criteria's and constraints. iRobot Create 2 met the specifications for the project and allowed for advanced software development. Furthermore, ROS can be installed on Raspberry Pi microprocessor which can be mounted on the robot itself and not interfere with navigation. LIDAR can be attached to the Raspberry Pi which will scan the area and create 2D map. Furthermore, using different algorithms for finding routes and shortest distance will be crucial for autonomous navigation.

3.1 SLAM- Simultaneous Localization and Mapping

Simultaneous localization and mapping, or SLAM for short, is the process of creating a map using a robot or unmanned vehicle that navigates that environment while using the map it generates. While trying to plot out the area, the robot must figure out where it is located on the map. This is the first step in autonomous navigation inside warehouse area. "The process of SLAM uses a complex array of computations, algorithms and sensory inputs to navigate around a previously unknown environment or to revise a map of a previously known environment [12]." The complexity of SLAM is mapping out an area and figuring out where it is at in relation to the map. For this to work, knowing the robots position is a first before trying to plot new area. "Simultaneous localization and mapping, developed by Hugh Durrant-Whyte and John L. Leonard, is a way of solving this problem using specialized equipment and techniques [12]." SLAM consists of multiple parts; Landmark extraction, data association, state estimation, state

update and landmark update. The goal of these several steps is to constantly update position of the robot as its moving around. To do this, the robot odometry data is used for localization. Odometry is the measure of how well the robot can estimate its own position. “This is normally calculated by the robot using the position of its wheels [12].” However, there is normally small margin of error with the robot’s odometry data, laser scans are used to correct the position of the robot.

“One requirement of SLAM is a range measurement device, the method for observing the environment around the robot [12].” LiDAR was used in the project to collect laser scan data. Another key process in SLAM is acquiring data about robot’s surroundings. The robot will use landmarks for localization using its sensors and laser scans. These landmarks must be clearly visible, stationary and unique from other landmarks. This is the only way the robot will be able to correctly position itself. “Once a robot has sensed a landmark, it can then determine its own location by extracting the sensory input and identify different landmarks [12].” SLAM is the process of continuously using data from sensors and laser scans to update its position on the map. Through this process, SLAM is seen to be an advanced tool and open source software development project on its own.

ROS uses Adaptive Monte Carlo Localization(AMCL) algorithm to track the position of the robot. The pose of the robot is shown as a particle. These particles, represented as red arrows move according to relative movement of robot using odometry and laser scan to fit the map given the position of the particle. It transforms the map frame to odom frame to correct the odometry results.

As the robot is moving, a file type .bag gets recorded which stores /odom, /scan/ and /tf information while its driving. A particle filter is applied to track the robot's trajectories. The map is then saved from .bag file and is represented as an image showing the blueprint of its environment. The map gets saved as a configuration file (.yaml) that gives information about the map (origin, size of a pixel, etc).

3.2 Navigation Stack

For this project, using the navigation stack was essential for moving the base of the robot, based on different sensor information that was internally and externally mounted on the robot. Using the different nodes within navigation stack allowed for controlling the iRobot autonomously and use the information to write a program for autonomous navigation. The navigation stack will be used to autonomously control the mobile base by sending the required command for controlling the robot's velocity in the form of: x velocity, y velocity, theta velocity[13]. The laser scan will also play a critical role in navigation for self-localization and figuring out orientation and direction.

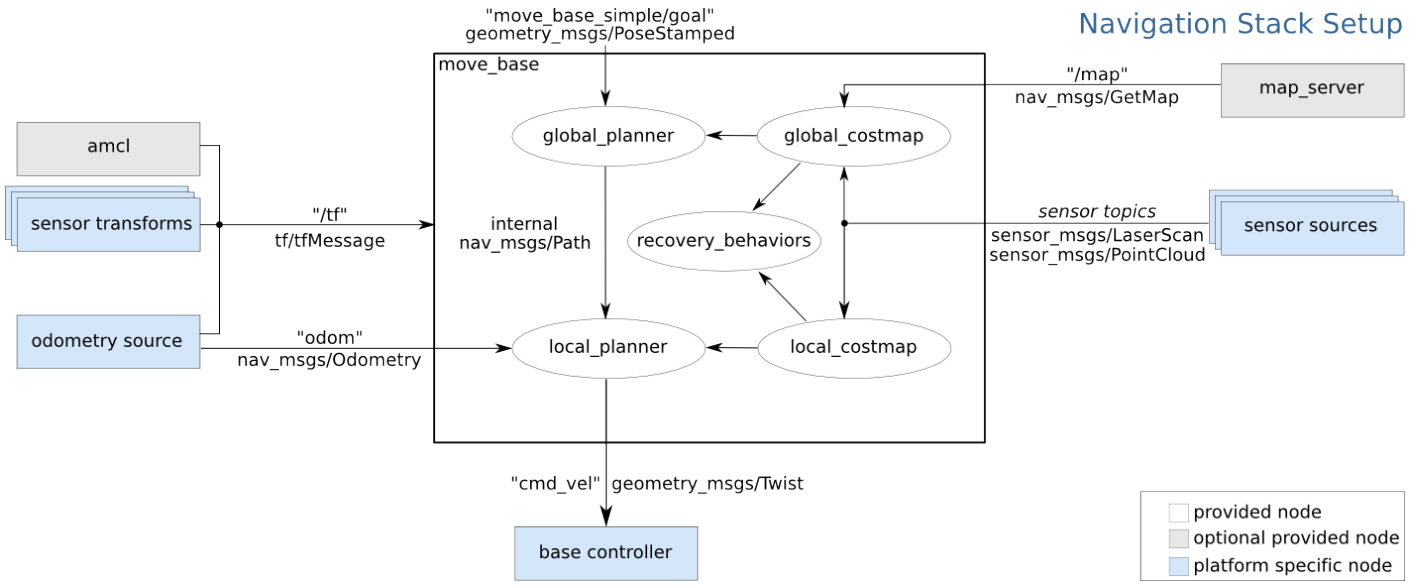

Figure 5: Navigation Stack Flowchart [13].

Figure 5 shows the configuration of the robot and it relation to the navigation stack. For the robot to run effectively, several data must be received by the navigation stack for proper path planning and execution. Such sensor data include information from external hardware; including the iRobot Create 2's odometry data and the LIDAR laser scan data. Using the data from these sources, the robot is properly able to align with the prebuilt map and localize.

Map_server and amcl are local nodes that need to be separately run but works in conjunction with the navigation stack.

To store information about obstacles that the robot faces, global and local costmaps are used. Global costmap stores information for global planning, for navigation planning local costmap is used to do obstacle avoidance. Essentially, these map layers are formed on top of the prebuilt

map and displayed on RVIZ (output screen). Costmaps are essential in creating an environment for robot to move on the map and plan its path without hitting any obstacles.

Global path planner and local path planner are plugins that take in information from global/local costmaps and other sensor information to create the shortest path between two selected points and use it to command with the base of the robot for physical robot movement (base controller). The main nodes that were used in this project included:

- move_base: moving the base of robot. Sending cmd_vel (velocity) commands to base.
- laser_scan_matcher: uses laser scans to match with map
- waypoint: having multiple destination points on the map
- planner: navigating where the robot should go.
- map: sending saved map to RVIZ screen.
- robot(ca_driver): data communication between robot and computer
- laser: LiDAR scan information.

The transform configuration was critical to maintaining connections among the different sensors and hardware components. Tf is what allows different nodes to talk to each other and use that info to execute different commands by other nodes.

4.0 Results

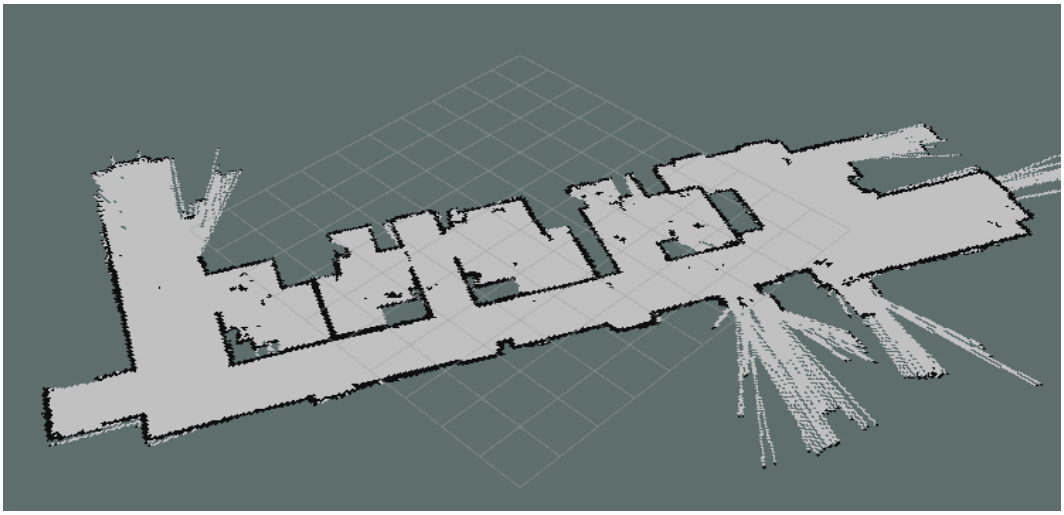

Figure 6: Map of Office Area in Ariss Facility created using SLAM.

Figure 6 shows the map of the office that was created from the moving of the robot. This was done using SLAM methods and LiDAR to scan and map as the robot moved around the office.

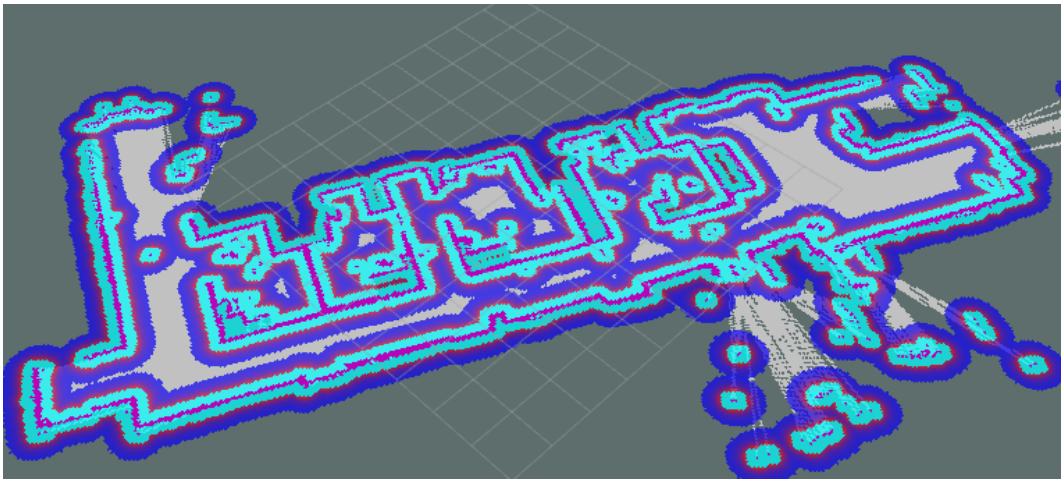

Figure 7: Costmap of Office Area in Ariss Facility.

Figure 7 shows the map of office with costmap layer. Costmap is a map that considers obstacles and adds the radius of the robot to the obstacles.

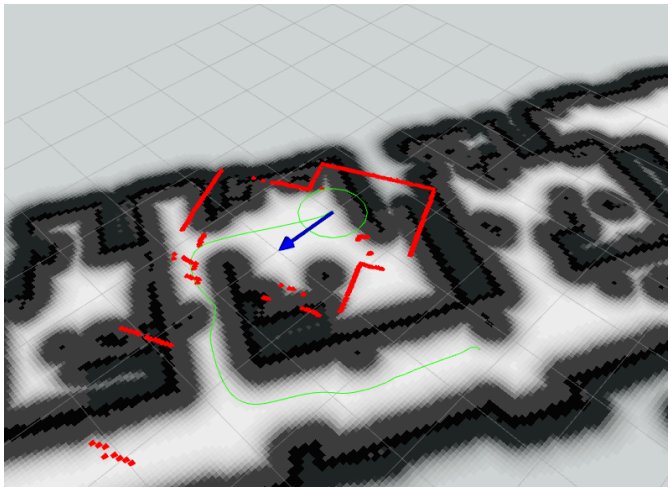

Figure 8: Auto Navigation of Robot in Ariss Facility.

Figure 9: Robot with LiDAR and Raspberry Pi.

Figure 8 and **Figure 9** shows LiDAR laser scan using laser match node to match the scan with the immediate environment within the map for localization. The blue arrow shows the direction of the odometry, and the green circle represents the base of the robot (polygon) on the map. The green line that wraps around this cubicle is the path the robot needs to take to reach its destination. **Figure 9** has attached LiDAR and Raspberry Pi microcontroller moving autonomously. In autonomous navigation, it is crucial for odometry, map, and laser scan data to continuously work together for localization and maintain its path on the green line.

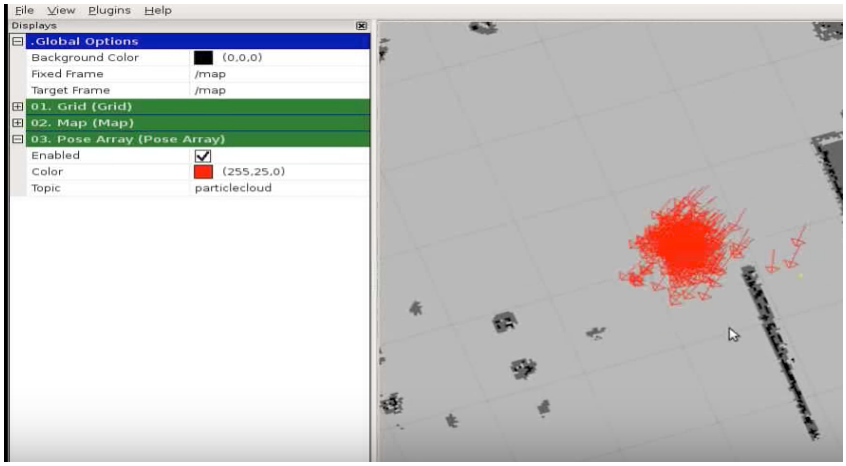

Figure 10: Particle Filter for Robot.

The cluster of red arrows represents the initial estimate by ROS software system when travelling to destination. As shown in **Figure 10**, the cluster of arrows become smaller as the robot continues its path and localization becomes more accurate. It uses laser scan data to estimate where it might relatively be on the map with the robot polygon base. This cluster is constantly updated as odometry and laser data changes when the robot moves autonomously.

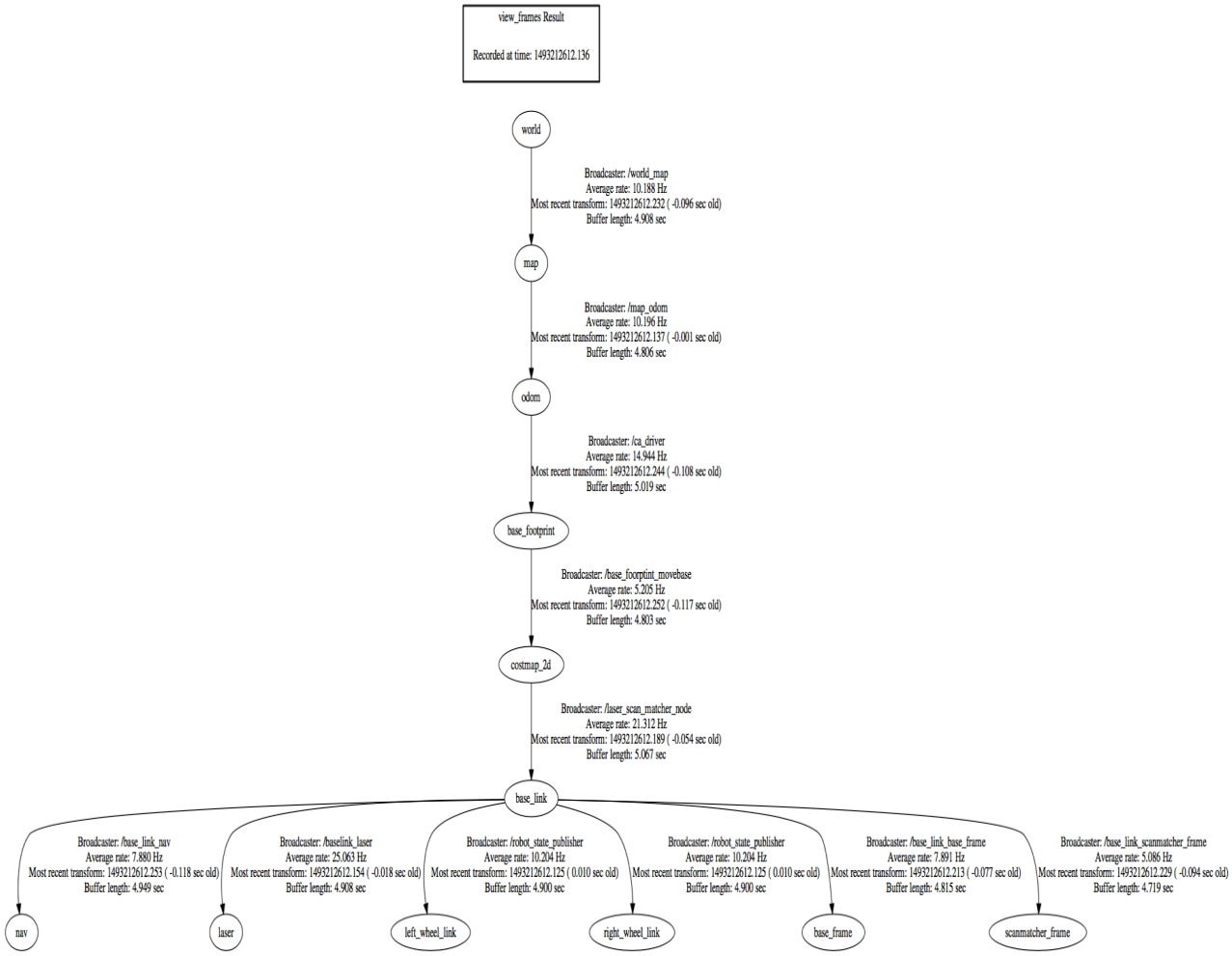

Figure 11: tf tree.

Figure 11 is the tf tree, which shows all the topics that are linked using transform configuration. As mentioned earlier, tf is used to connect several topics, and feed data from one sensor to another from different processes to run and update as robot moves autonomously.

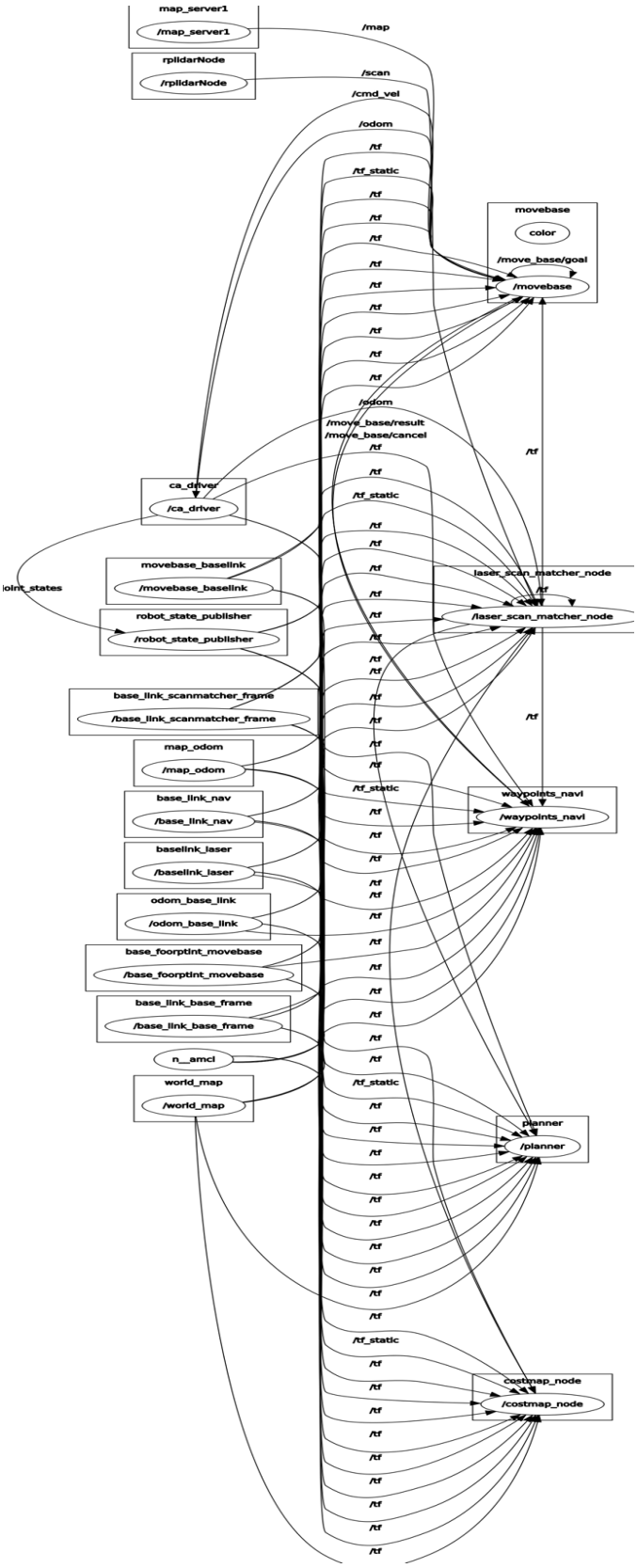

Figure 12: rqt_graph.

Figure 12 is the rqt_graph, which shows the connection of different nodes on the tree when the robot is moving and is being updated. This graph was important when trying to debug different error and tune parameters in code files.

Table 5: Total Cost for the Project

Item	Price(CAD)
iRobot Create 2	$400
Raspberry Pi Kit	$100
LiDAR sensor	$600
Total	$1100

This project resulted in a cost-effective solution to autonomous navigation for Linamar's application, as shown in **Table 5**. This initiation proved that innovation does not need to cost a fortune, but engineering design and process will help company achieve its goals. This project was delivered on time and ended up saving the company $900, given the initial $2000 budget. Furthermore, the average battery life was proved to be over 3 hours exceeding the constraint values. The robot was properly implemented in the work term and met the criteria and constraints initially described.

Table 6 shows timings between manual moving vs robot moving different parts. This experiment was conducted moving 100 meters with the speed of the robot set to 2 m/s and carrying up to 2 kg of weight. The test run was executed at Ariss Facility (Linamar plant).

Table 6: Initial Test Results

Trial	Manual Moving Time (sec)	Robot Moving Times (sec)
1	80	60
2	78	58
3	75	55
4	82	56
5	81	59
Average	79.2	57.6

Through this trial, the robot had successfully traveled without object falling off and performing obstacle avoidance. The robot moves on average 21.6 seconds faster than their average employee. As seen throughout the report the robotic method was the favoured solution and was then one being tested at the facility.

4.1 Production Day Testing

Production day testing involved real life simulation by aiding the workers moving parts from shipping area to work site. The cost of running the entire shift in the shipping area is $1200/day. This is based on averaging 10-people working 10 hours a day, with minimum wage of $12/hr given at Linamar. For production testing day, the labour force was cut in half (5 people) and with the robot, ended up transporting the materials within their 10-hour shift. If the company runs one shift in the receiving department, the company will save $600/day ($1200- $(12x10x5)). However, more testing is needed to come up with reliable results and data to calculate return of investment (ROI) for the robot.

While putting the robot into the facility, the company concluded that production efficiency also increased while labour had decreased. The production efficiency for that specific day was 200% and has potential to increase even more. Linamar's definition for production efficiency, as seen from equation 1 is to take the total working hours of all employees for one shift and divide it by the total amount of time worked by all employees to complete the task. In production day testing, standard labour hours are 10 people x 10 hours = 100, and amount of time worked is 5 people x 10 hours= 50. Thus, 100/50 x 100% yields 200% production efficiency. Using the robot to move parts internally has increased cost savings and overall production in the warehouse [14].

$$\text{Production Efficiency} = \frac{\text{Total Manhours}}{\text{Manhours Needed to Complete Task}} \times 100\% \qquad (1)$$

The robot proved to be potential to the company, serving its function in different applications. The innovations group was delighted by the fact that the project executed as expected and will benefit the company in future.

5.0 Conclusion

Autonomous navigation is creating the future of tomorrow and helping people move by transforming the transportation industry. Technology companies and software companies are partnering up with automotive manufactures, to create self-driving vehicles that can increase safety for passengers and reduce traffic. The streets would be safer because of autonomous vehicles, providing faster means to move and increasing productivity of workers and other persons.

Linamar's target is to increase productivity by transforming its workplace through automation and robotics. Through this project, they could prove how technology can transform workplace. Robotics and software is a combination that will change Linamar future businesses approaches and their applications in building future cars. The future concepts of factories will be led by Linamar's approach to new technology and adaptability to changes.

Linabot (Create 2 Robot), a latest addition to Linamar tech products, is a disruption in the manufacturing business. Robot Operating System(ROS) provided the right solution to creating a prototype and shaping its future possibilities. Open source software and external sensors are the future of integration and creating innovation. It has been tested that Linabot was able to carry up to 2 kg of weight, and proves to be a safer solution rather than manual object moving. The robot proved to be 200% production efficient in warehouse and testing showed its stability and strength.

When building robots or creating a tech product, it is very important to consider different hardware and software aspects during engineering analysis. Considering the expandability and software compatibility of the robot is crucial to development. Using microprocessor and attaching different sensors, camera and other I/O ports are needed to future integration with warehouse. Careful analysis of what is available and resources allocated to this project will have a direct impact on its success. Based on the success of this project, Linamar and the management team has decided to create more robots that will work together and integrate within their manufacturing plants. Having multiple robots work in conjunction using neural network and artificial intelligence will continuously improve the robot's performance and learn through trial and error process.

6.0 Recommendations

Moving forward, the next steps would include building an app that connects the robot to a mobile device and send notifications about robot's position. The company also plans on implementing the concept in drones.

The application will need to be designed to allow supervisors, setup techs, maintenance, and other designated employees to easily access the drone and or land robot logistics system to request tooling, parts, and other job necessities be delivered to their location. This application must also could request items such as first off reports, QA paper work, ODS, tooling and other packages be picked up from their location.

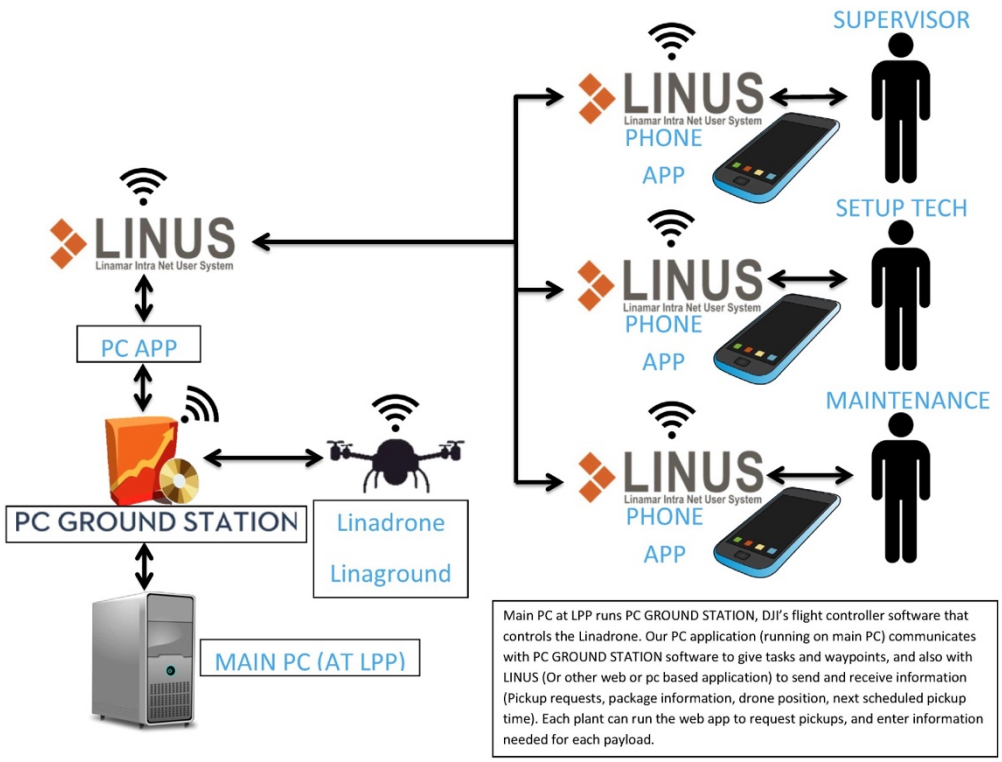

Figure 13: Proposed Future Software Development Plan.

Figure 13 is a general layout of the network needed for the application. A main PC would run the software to control the drone/land vehicle. This software would communicate and exchange information (such as drop off/pickup locations) with the PC APP. This APP could use a pre-existing network (such as Linus) to communicate with phone APPs employees would have on their phones.

The phone application needs to be simple and easy to use. As shown in **Figure 14** below, a login should be required so employees who are not authorized to request drones can't do so. After login is confirmed the user will have the choice between pickup or drop off. Sub groups and a search menu for drop off could help speed up the process.

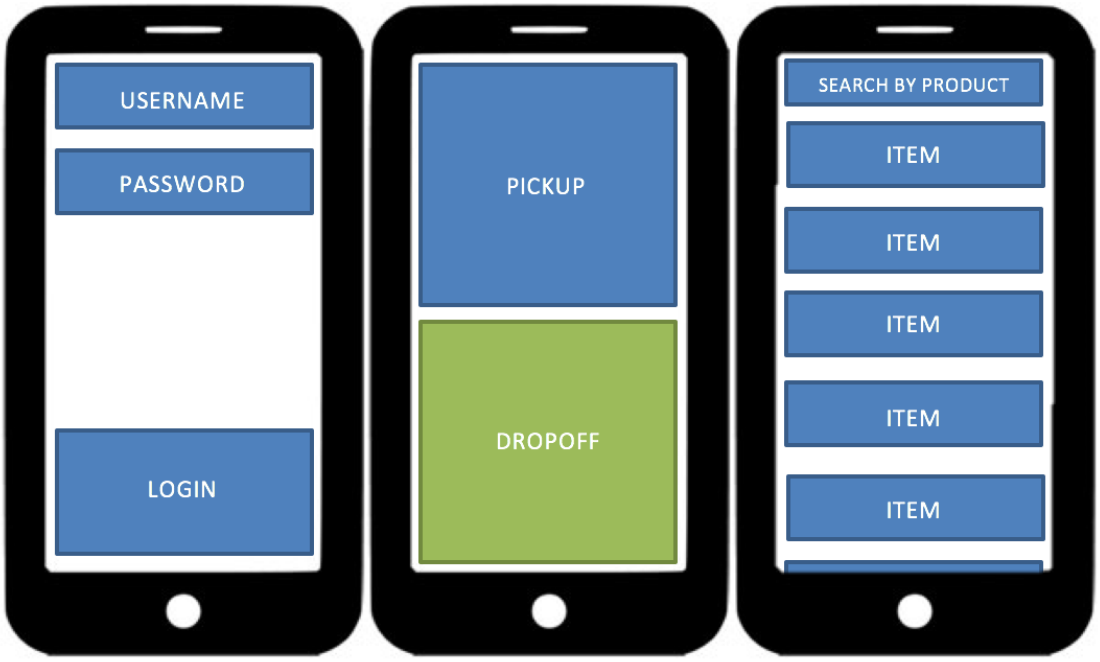

Figure 14: Proposed Future User Interface Design.

Proof of concept application would be a simple application that selects waypoints and the robot travels there completely autonomous, without hitting any obstacles or people. The robot should route the most logical route to the waypoint and know where it is and where it is heading always.

The vehicle should be able to consistently identify and avoid obstacles as thick as a power cord under normal factory lighting. Once achieving its destination, it would send notification to appropriate supervisor or nearby factory worker.

7.0 References

[1] Linamar Corporation, "Overview, Power to Perform", 2017. [Online]. Available: http://www.linamar.com/products. [Accessed: 17- Feb- 2017].

[2] Seegrid Corporation, "Increase Throughput Productivity", 2017. [Online]. Available: https://seegrid.com/products/. [Accessed: 18- Feb- 2017].

[3] Otto Motors, "AGV vs. SDV", 2017. [Online]. Available: https://www.ottomotors.com/resources/info/agv-vs-sdv. [Accessed: 18- Feb- 2017].

[4] Robot Operating System, "What is ROS?", 2015. [Online]. Available: http://wiki.ros.org/ROS/Introduction. [Accessed: 18- Feb- 2017].

[5] Robot Operating System, "Nodes", 2015. [Online]. Available: http://wiki.ros.org/Nodes. [Accessed: 18- Feb- 2017].

[6] Robot Operating System, "Topics", 2015. [Online]. Available: http://wiki.ros.org/Topics. [Accessed: 19- Feb- 2017].

[7] Clearpath Robotics, "Image", 2017. [Online]. Available: https://www.clearpathrobotics.com/assets/guides/ros/_images/ros101one.png. [Accessed: 19- Feb- 2017].

[8] Robot Operating System,"tf", 2015. [Online]. Available: http://wiki.ros.org/tf. [Accessed: 18- Feb- 2017].

[9] Nexus Robot, "Image", 2017. [Online]. Available: http://nexusrobot.net/category9a93.html?id_category=1. [Accessed: 19- Feb- 2017].

[10] Robot Shop, "Image", 2017. [Online]. Available: http://www.robotshop.com/ca/en/hangfa-discovery-c2-robot-platform.html.[Accessed: 19- Feb- 2017].

[11] iRobot Corporation, "Image", irobot.com, 2017. [Online]. Available: http://www.irobot.com/About-iRobot/STEM/Create-2.aspx. [Accessed: 19- Feb- 2017].

[12] R. Maxwell, (2013, January 15), "Robotic Mapping: Simultaneous Localization and Mapping (SLAM)". [Online]. Available: https://www.gislounge.com/robotic-mapping-simultaneous-localization-and-mapping-slam/. [Accessed: 18- March- 2017].

[13] Robot Operating System, "navigation", 2015. [Online]. Available: http://wiki.ros.org/navigation. [Accessed: 28- Feb- 2017].

[14] Linamar Corporation, "Calculating_Production_Efficiency", 2017. Unpublished.