Robotic Arm with Mobile Platform

Mingyu Pan
Electrical and Computer Engineering
Stevens Institute of Technology
Hoboken, USA
mpan5@stevens.edu

Ilana Zane
Electrical and Computer Engineering
Stevens Institute of Technology
Hoboken, USA
izane@stevens.edu

Shucheng Yu
Electrical and Computer Engineering
Stevens Institute of Technology
Hoboken, USA
syu19@stevens.edu

***Abstract*—The work done during the Spring 2022 semester is a continuation of research results from previous semesters. For the spring semester, our main objectives are: First, design and build a platform with mobility capability that can support the weight of the robotic arm and all other necessary components. Second, refine the voice recognition algorithm that had already been integrated into the Jetson Nano system to achieve command words recognition. By using components that were purchased but not used last semester, we will also install a more advanced 2D laser scanner on the platform for automated world map generation. In the end we designed and constructed a sturdy platform with mobile capability, improved the ASR functions and installed the spinning LIDAR on the platform and integrated it into the Jetson Nano system. NOTE: The members of the team will be referred by their initials. IZ for Ilana Zane and MP for Mingyu Pan.**

***Index Terms*—control, robotic platform, voice recognition**

I. Introduction

ROS nodes are widely used to enable proper communication and control between different components in our robotic system. Currently, the Interbotix Reactor X150 robotic arm and the Raspberry Pi Camera Module V2 are able to work together using custom written ROS nodes. The USB microphone was used for speech recognition and was integrated into the system last semester. Nodes for communication between components were written for the object detection module to detect and output ids as well as name of all detected objects on a monitor. During this semester's research, most challenges were solved while other new challenges emerged due to the introduction of new components and algorithms.

In our proposal at the beginning of the semester and in the mid-stage report, we stated that the current Jetson system was able to achieve automatic speech recognition through a plug-in microphone and was able to perform accurate single-word and short-sentence recognition. Later this semester, we planned to integrated a keyword recognition function provided by jetson-voice. By modifying the objection detection script, we enabled the system to output the names, identification numbers and number of detected objects in our Ubuntu terminal through ROS. We also stated that the old platform was not stable enough, so a new mobile platform was designed and constructed. Unfortunately, We were informed by the ProofLab that all large 3D printers would not be available for several months due to maintenance, so we had to discard our original custom platform design. Ultimately, all the components to build the platform were purchased from gobilda.com and constructed by us. The new platform was proven to be very modular and rigid enough to support the arm and all other sub-functions while remaining stable. The mecanum wheels also provided better maneuverability compared to previous traditional rubber wheels. The new, more advanced LIDAR system also enabled the automated generation of a world map. The generated map was displayed in RVIZ and would be useful in future dynamic path planning researches. We divided our work as follows: MP worked on designing and constructing a new platform as well as integrating the LIDAR to the Jetson to enable automated world map generation. IZ worked on improving the existing software functions of the Jetson Nano system to ensure smoother communication between the voice detection algorithm and various ROS nodes in the system.

Our paper is structured as follows. In the second section, we discussed various problems encountered during our research and the updated components list. In the third section, we presented our solutions to those problems by providing our ideas and methods. Detailed descriptions and algorithms were provided to explain our research process. In the fourth sections, we presented our analysis and results of the solutions using our models constructed both in Fusion and in real-life. Finally, in section five, we conclude our paper by summarizing the research done in this semester and proposed several other research directions for future studies.

II. Problem Statement

At the beginning of the semester, we set our first goal to enable the Jetson Nano system to recognize recognize specific words such as ”forward”, ”backwards”, ”stop”, ”go”, etc, using the automatic speech recognition framework already implemented in the system. The voice-recognition algorithm was still rudimentary and can not yet communicate with other components in the system. Enabling proper communication was one of the goals. An original goal of ours was to refine the object detection algorithm in order to reduce the number of object it detects. We decided that although, it would be more complicated, it would be better to take a step back and begin incorporate a voice recognition module to the ROS system. Our second goal was to research on designing and constructing the new platform. In the previous semesters, the development of the platform was put on-hold since the research was focused on improving the software of the system. We had concluded that the previous platform was too unstable for our system. As a result, we decided to design and construct our own.

The purpose of the chassis is to allow for the robotic arm to have mobility. A goal that we worked on this semester was to allow the robotic arm to communicate with our microphone via the Jetson. On a high level, we aim to a fully functioning node that can receive voice commands from the user via microphone, then store these commands to a database that will eventually connect to our camera module, as well as the module that controls the motion of the chassis. Finally, the LIDAR would be used to plan a path towards this object while avoiding obstacles.

The original plan was to custom design a platform using Fusion360 to 3D print parts. However, after our design was submitted, we were informed that the parts could not be printed out in time due to the machines being repaired in the ProofLab and we did not find a suitable third-party printer. As a result, we decided to purchase platform frame parts from gobilda.com. We did not want to purchase a pre-made platform since platforms with lower prices were not suitable for modular design and could not fit components such as the Jetson Nano, a power supply, and other necessary hardware inside. The platform we constructed using various components was proven to be modular and stable enough for our research purposes. We also upgraded our traditional rubber wheels to mecanum wheels. The designing and construction process of the platform will be detailed in the next section. In conclusion, several challenges needed to be addressed in this paper such as:

1) Improve the speech recognition module by redesigning relevant ROS nodes and custom scripts to extract keywords. Ensure algorithms and hardware can be integrated with the rest of the system. 2) Research on the construction of the vehicle and other essential parts such as wheels and LIDAR, as well as other software algorithms that can integrate separate components to ROS using custom nodes or other methods.

The components we used in our research are listed here. New components were used and installed on the platform during our research and we decided to include an updated components list here:

1) Trossen Robotics ReactorX150 Robotic Arm
2) NVIDIA Jetson Nano Computer
3) Monitor for Head Mode operation and setup of Jetson Nano
4) RPLIDAR-A2 Laser Range Scanner
5) Raspberry Pi Camera V2 Module
6) Kinobo USB 2.0 Microphone
7) Various aluminum frames and components for platform construction(50 pieces)
8) goBuilda 96mm Mecanum Wheels and supporting hubs for connections
9) DZS Elec DC-DC Buck Converter Module 5.3-32V 24v to 1.2-32V 5v 9v 12A 160W Large Power Adjustable Step Down Voltage Regulator CC CV Power Supply with LCD Display
10) Alitove 12V 15A 180W Power Supply Transformer Switch AC 110V/220V to DC 12V 15amp Switching Power Adapter Converter LED Driver for LED Strip Light CCTV Camera Security System
11) 4 X Cytron 10A Dual Channel Bi-Directional DC Motor Driver, 5-25V, 30A Peak
12) Milapeak 12pcs 8 Positions Dual Row 600V 15A Screw Terminal Strip Blocks with Cover + 400V 15A 8 Positions Pre-Insulated Terminals Barrier Strip

In the next section, we detail our research ideas and results by explaining our process through constructed models.

III. Ideas and Design Process

For our mobile platform, we needed to research on designing our own since it would give us the most freedom to be creative. It was proven that a pre-made chassis sold on websites like Amazon could not allow us to install components purchased from various websites and sources, generally due to the lack of screw holes on the surface. Also, almost all cheaper chassis's sold were built to serve one or two purposes only, with simple motor control using traditional, non-turning wheels. Those platforms also did not account for extra weight that needed to be put on the surface since all components were usually put inside the chassis.

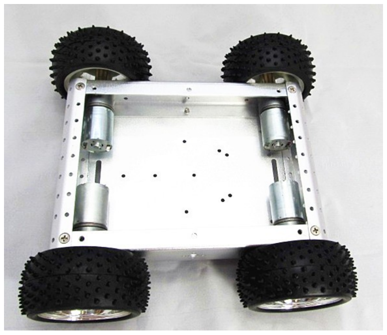


Fig. 1. The previously used chassis. Note the lack of screw holes, thin interior space and bulky rubber wheels.

We also had to give up the idea of 3D printing our own chassis since we could not get a definitive timeline on when the platform could be printed. We decided to purchase our parts from gobilda.com. The website has great reputation on providing high-quality parts for both small and large projects. The parts provided on the website were also all very modular, which gave us the freedom to modify our design freely without needing to drill extra holes on the platform.

Next, we detail our construction process for the platform. Two U-channels with length of 432 millimeters were used as mounting points for the 4 mecanum wheels and motors. Since the arm has a max reach around 550 millimeters, to ensured that the arm can still reach an object no matter where the arm was installed on the surface, we chose the longer U-channel out of a set of options, taking into consideration the shift of center of mass when the arm was in operation to maintain stability if the arm was at its max span. Two 264 millimeter U-channels were then used to connect the longer U-channels to form a rectangular platform. Quad block mounts and screws were used to connected all U-channels together. There are sufficient amount of screw holes on the platform for us to move the shorter U-channels along the longer U-channel so the platform can accommodate wider components such as the Jetson and power supply.

Next, we needed to design the output shafts for the chassis. Since we were using U-channels instead of design of our own, the motors could not be connected directly to the mecanum wheels due to the width constraint of the channels. As a result, we used several 8 millimeter REX bore miter gear to transfer the power of the motors to the wheels. This enabled us to install the motors inside the long U-channel to save space on the platform. Four 80 millimeter REX shafts were used to connect the miter gears to the U-channels. Ball bearings, thrust washers and flanged bearing were used to secure the connections between the shafts, miter gears and the U-channels to create as little friction as possible when the shaft was spinning. Next, 8mm REX bore Hyper Hubs were used to connect the mecanum wheels to the REX shafts.

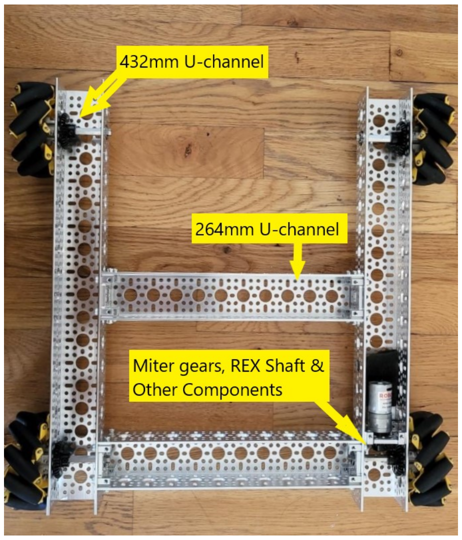


Fig. 2. Platform frame construction. The position of the 264mm U-channel can be changed to accommodate for various component sizes. (Only motors and the wheels were installed in this picture).

Next, we needed to install the drive motors inside the long U-channels. This was the hardest part in our assembly process due to several reasons. First, the motors we were using had neither a hexagon shaped shaft, nor were the shafts wide enough to fit inside the miter gears. Also, there were no screw holes on the outside of the motors, which meant we could not secure the motors to the U-channels. Due to this, we decided to design an element using Fusion360 that can secure the motors to pattern mounts and allow force transfer from the motor shaft to the connected miter gear. The designed part was small, so we could 3D print it on our own (Fig.3 & 4).

The motor was secured to a quad block pattern mount using pattern spacer and screws. The miter gear was installed on

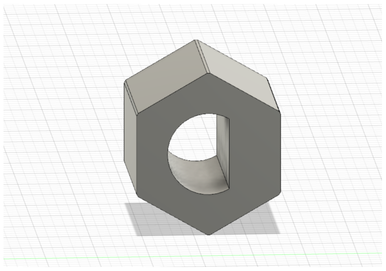


Fig. 3. Designed connection between motor and miter gear.

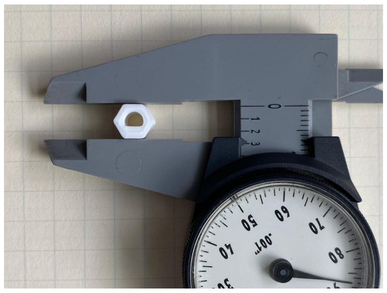


Fig. 4. Design after being printed.

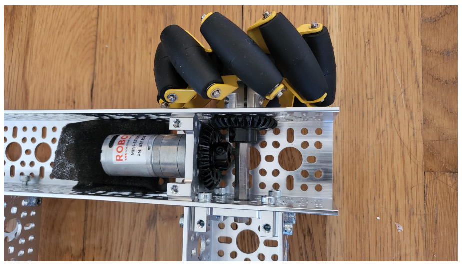


Fig. 5. Close up of the connection between the motor and the mecanum wheel using miter gears and custom designed parts.

the modified shaft , with a thrust bearing placed between the spacer and miter gear to reduce friction when the motor was spinning. The output shaft gear and drive motor gear were placed at an 90 degree angle, enabling force transfer from the motor to the wheels. The wheels were also able to function independently.

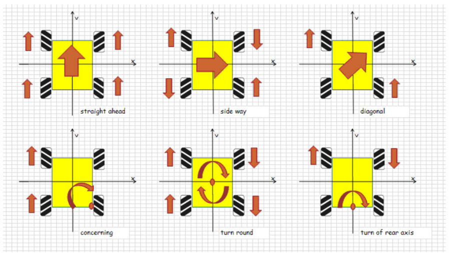


Fig. 6. Mecanum wheels functions. Sourced from: https://www.roboteq.com/applications/all-blogs/5-driving-mecanum-wheels-omnidirectional-robots

Here, we will justify our decision of using mecanum wheels instead of regular wheels.

Fig. 7. Mecanum wheels purchased from gobilda.com

We determined early in the semester that better wheels needed to be used to increase the stability and performance of the entire platform. We measured the new platform to be 16.95 lbs based on our measurements when all components were put on the platform. The solid construction provided excellent rigidity even when the platform was at full load, which provided significant improvement compared to the old design. The Mecanum wheels also provided much better maneuverability [5], since they enabled the platform to move horizontally, diagonally, and rotate around a single axis. We also have individual control of each motor and no front or back axles were needed, so the directional motion of the platform would also needed to be planned.

Based on our research, Mecanum wheels would also better suit future path-finding algorithms such as A-star search [3]. In some algorithms, the robot was treated as a point mass. This gave the Mecanum wheels an advantage since future researcher may not need to code an additional turning algorithm if the platform is traversing in a tight environment, since independent control of the motors can enable the platform to perform a single-axis turn.

In this semester, we also acquired a more advanced laser scanner. The previous LIDAR used was a Garmin Lite V3. It was able to detect the distance between the receiver and an obstacle in front of it. However, it could not publish more comprehensive information about the environment the platform was in. As a result, the RpLIDAR was used. The RpLIDAR a2 was a spinning LIDAR that could turn counterclockwise and perform a 360 degrees laser scan of the surrounding. It had a sample rate of 10Hz, 8000 points and could detect a maximum range of 16 meters. The detection rate would be sufficient to generate an area map automatically while the platform was moving.

When operating independently, the RpLIDAR was able to generate an area map, where the red dots indicated an obstacle detected. We also included pictures taken from the real life environment to verify the accurate of the generated map. Several landmarks were indicated in the picture such as the west curved wall, divider 1, divider 2, and the position of several doors (Fig. 8 & 9).

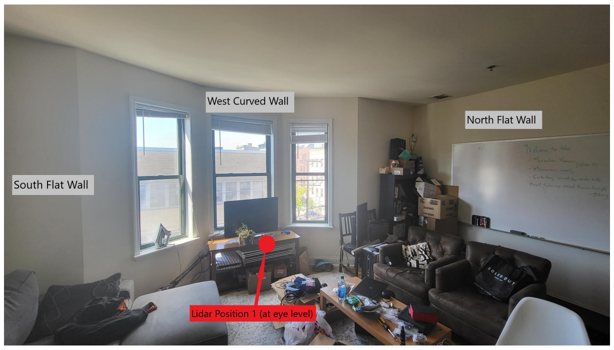


Fig. 8. Testing environment - West, Facing Living Room.

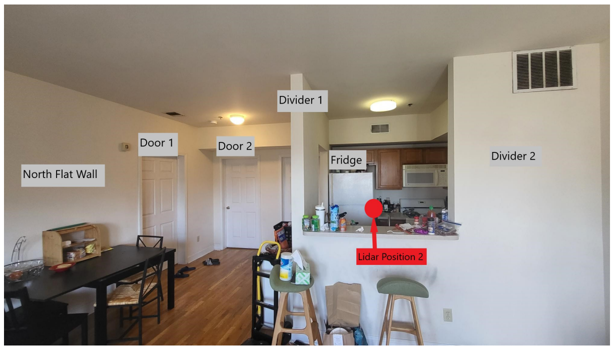
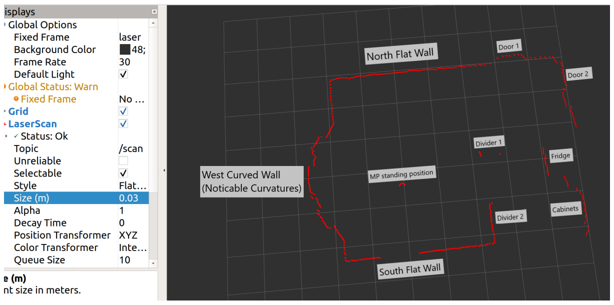


Fig. 9. Testing Environment - East, Facing Kitchen & Doors.

The area maps generated using the scan are included in Figure 10 and Figure 11. The scan was performed at a stationary position, 180cm above ground, to avoid small obstacles blocking the laser. In Figure 11, the position of the LIDAR was changed to in front of Door 1, at the same height, to scan the kitchen. Note that the previous map of the living room became inaccurate due to increased scanning distance and blockage of laser due to Divider 1.

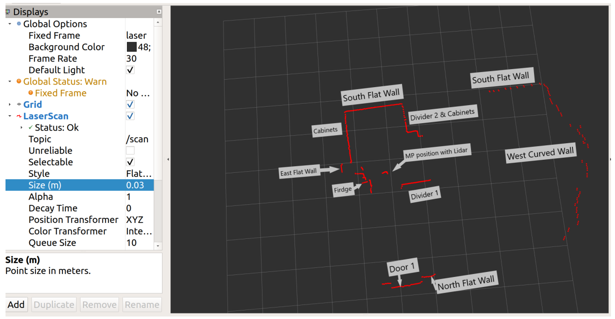


```
isplays
Global Options
  Fixed Frame            laser
  Background Color       48;
  Frame Rate             30
  Default Light          ✓
Global Status: Warn
  Fixed Frame            No ...
Grid                     ✓
LaserScan                ✓
  Status: Ok
  Topic                  /scan
  Unreliable
  Selectable             ✓
  Style                  Flat...
  Size (m)               0.03
  Alpha                  1
  Decay Time             0
  Position Transformer   XYZ
  Color Transformer      Inte...
  Queue Size             10
e (m)
nt size in meters.
```

Fig. 10. Scanned Result - Living Room - LIDAR Position 1.

We noticed that in LIDAR position 2, due to the small space, most of the South and North wall was not scanned due to divider 1 blocking the laser. Door 2 was also not shown due to the fridge. In conclusion, however, the scanning results were very accurate.

To expand on its functionality, we wanted the LIDAR to create an persistent area map, which would lay some basic



Displays

Global Options	
Fixed Frame	laser
Background Color	⬛48;
Frame Rate	30
Default Light	✓
Global Status: Warn	
Fixed Frame	No …
Grid	✓
LaserScan	✓
Status: Ok	
Topic	/scan
Unreliable	
Selectable	✓
Style	Flat…
Size (m)	0.03
Alpha	1
Decay Time	0
Position Transformer	XYZ
Color Transformer	Inte…
Queue Size	10

Size (m)
Point size in meters.

Fig. 11. Scanned Result - Kitchen - LIDAR Position 2.

ground work for the path planning objective. The RpLIDAR was suitable for simultaneous localization and mapping, or SLAM, which enabled the platform to explore a previously unknown environment while keeping track of the platform's location. This method is widely used in autonomous mobile robots. Based on our research, we decided to use ROS Hector Mapping and integrate it to the RpLIDAR to enable automatic closed area map generation.

We will now detail our operation method in this following section.

First, we cloned the ROS packages from github.com/Slamtec. Slamtec is the company that designed and manufactured the RpLIDAR. The designed ROS nodes and testing applications were cloned to the Jetson Nano catkin workspace folder to enable proper communication between the ROS nodes of the RPLIDAR with our ROS Melodic system. No errors were produced when compiling the workspace. We first ran a test trial to determine if the LIDAR was working properly or not. The output of the LIDAR was outputed in RVIZ. RVIZ is a graphical interface for ROS that visualizes various information outputted by the connected component. The output was included in Figure 10 and 11.

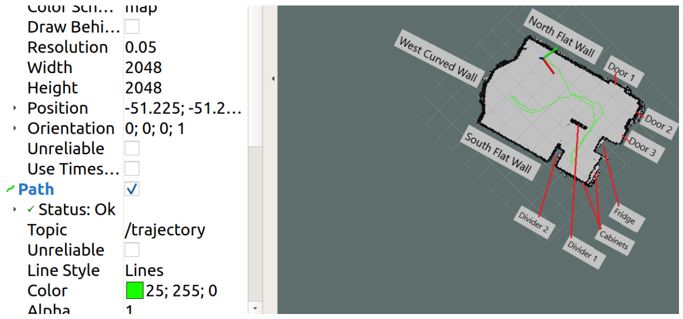
It was noted that in the dotted map above, an image outsine of the obstacle could persist on RVIZ only if the LIDAR was actively scanning it. If the LIDAR was to change its position, say for example, going in to another room, the knowledge of the previous room would be lost.

We think it was important to generate a persistent area map in our research since a map will make it possible to determine a starting point and a goal for the platform while it is in motion. Based on our research, we determined that Hector SLAM would work well in our system and environment [9]. There are many types of 2D simultaneous localization and mapping, including but not limited to: HectorSLAM, Gmapping, KartoSLAM, Core SLAM, LagoSLAM. HectorSLAM utilizes robust scan matching to generate an accurate area in real-time while tracking its own position [4]. The high-scanning frequency of the RPLIDAR meant it fit well with the HectorSlam algorithm, and the scanning range of the LIDAR meant an accurate image of the obstacle outline could be produced. HectorSLAM also did not require a huge amount of processing power when running continuously. High CPU load had been proven to be a huge problem in our research since the system needed multiple programs to run at once and lagging was always an issue. A low resource consumption algorithm could make our experiments more efficient.

Gmapping is another widely used mapping algorithm, and in theory, should work better than HectorSlam due to its extensive application. However, during our research, it was not the case for the RPLIDAR since HectorSLAM performed much better than Gmapping. Many researchers also determined HectorSLAM outperformed Gmapping in several experiments [1] [2] due to several angle and scan range requirements by Gmapping. As a result, HectorSLAM remained as our primary choice of algorithm.

IV. Results and Analysis

In HectorSLAM, several main nodes include hector_mapping that was the SLAM node, hector_geotiff that saved the map and robot's trajectory, and hector_trajectory_server that saved tf-based trajectories [6]. To setup HectorSLAM properly with the LIDAR, we modified several values. For example, base_link was used instead of base_footprint to include pitch and roll angles [7], and included the information broadcast by the LIDAR scanner along with base_link. After catkin_make, the errors were solved and the entire system compiled correctly. After giving proper permission to the USB port, we ran the example using rosrun rpLIDAR_ros rpLIDAR.launch and roslaunch hector_slam_launch tutorial.launch to produce the following area map below:



Color Sch…	map
Draw Behi…	
Resolution	0.05
Width	2048
Height	2048
Position	-51.225; -51.2…
Orientation	0; 0; 0; 1
Unreliable	
Use Times…	
Path	✓
Status: Ok	
Topic	/trajectory
Unreliable	
Line Style	Lines
Color	🟩25; 255; 0
Alpha	1

Fig. 12. Scanning result using HectorSlam.

Based on the results produced, we determined that the RpLIDAR and HectorSLAM were able to produce an very accurate area map of the environment that included exact positions of all obstacles detected and the position of the robot. The green line on the map also recorded the route taken to scan the entire room. The map was exported using the map_server ROS node. A .yaml file was created inside the catkin_ws folder where the map could be exported into another RVIZ environment.

We also decided to include the ASR classification from the jetson-voice repository. The ASR classification algorithm is trained using the QuartzNet model from NVIDIA. The QuartzNet model is a deep neural network made up of building blocks of a specified value. In comparison with the normal automatic speech recognition, the classification algorithm required lesser resources from the processor since only chunks of audio input were needed to enable classification. The MatchboxNet [11] classification algorithm (based on the QuartzNet architecture) is a model that was trained to recognize 12 keywords such as ”left”, ”right”, ”stop”, ”go”, which would be essential to control the motion of the platform when integrated to the system. Figure 13 shows a flowchart representing the layers of the QuartzNet NN. The model is composed of multiple blocks with residual connections between them. Each block consists of one or more modules with convolutional layers, batch normalization, and ReLU layers [10]. One of the toughest issues that we currently face in our project is the computing power of the Jetson and accompanying hardware. Using a smaller model like MatchboxNet for classification provides advantages such as it being faster to train and optimize upon running it with other modules. Therefore, it is much more feasible to deploy this model on hardware, like our Jetson, that has limited computing power [10]. Since the network was tested on a very small subset of data from the Google Speech Commands dataset, there is room for error. The model did not always recognize our speech patterns and was sensitive to noise. This is a possible issue that we may have to worry about later when connecting our voice module to the robot.

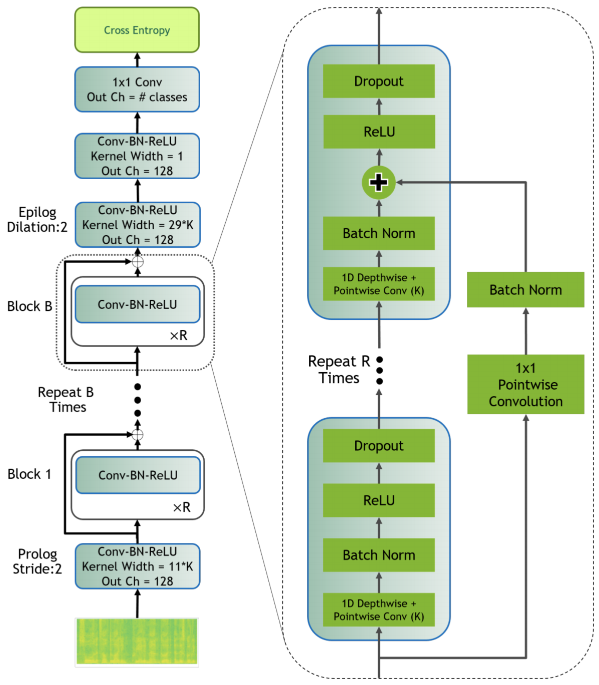
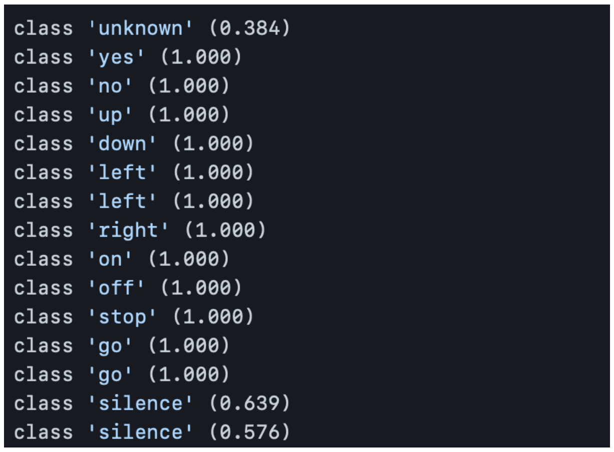

Cross Entropy
1x1 Conv Out Ch = # classes
Conv-BN-ReLU Kernel Width = 1 Out Ch = 128
Epilog Dilation:2
Conv-BN-ReLU Kernel Width = 29*K Out Ch = 128
Block B
Conv-BN-ReLU
×R
Repeat B Times
Block 1
Conv-BN-ReLU
×R
Prolog Stride:2
Conv-BN-ReLU Kernel Width = 11*K Out Ch = 128
Dropout
ReLU
Batch Norm
1D Depthwise + Pointwise Conv (K)
Repeat R Times
Batch Norm
1x1 Pointwise Convolution
Dropout
ReLU
Batch Norm
1D Depthwise + Pointwise Conv (K)

Fig. 13. CNN Model cited from Nvidia. https docs.nvidia.comdeeplearning (Accessed May. 1, 2022)

Figure 14 shows the results from testing the MatchboxNet module. In order to deploy the ASR classification module, we had to run it as a part of a Docker container, which contained



```
class 'unknown' (0.384)
class 'yes' (1.000)
class 'no' (1.000)
class 'up' (1.000)
class 'down' (1.000)
class 'left' (1.000)
class 'left' (1.000)
class 'right' (1.000)
class 'on' (1.000)
class 'off' (1.000)
class 'stop' (1.000)
class 'go' (1.000)
class 'go' (1.000)
class 'silence' (0.639)
class 'silence' (0.576)
```

Fig. 14. MatchboxNet Outputs

all of the code and dependencies that were required. Through the container and a shell script we were able to detect whether the microphone was properly connected to the Jetson and as an audio input. Then, through a python script we were able to launch our file, *asr.py*.

For each word that is spoken into the microphone, the model assigns the word to its respective class with a number associated to the probability of the spoken word being correctly assigned to that class. The silence class at the end of the output sequence represents that the user has stopped speaking.

V. Conclusion

We conclude our paper by presenting several goals we achieved during this semester’s research. We designed and constructed a platform with mobile ability, implemented a more advanced LIDAR for dynamic map generation. We also believe that dynamic path planning is possible with further research. Successful results had already been produced in another simulated environment, where a waffle type turtlebot3 with a laser scanner was able to produce an area map and avoid dynamically moving obstacles through A*, a dynamic obstacle avoidance algorithm. Since the result was also produced in RIVZ, we believe an automatically moving platform with obstacle avoidance ability can be made possible. We have also incorporated and semi-implemented a way to direct commands to the Jetson that will be crucial when mobility of the chassis is finalized. The use of the MatchboxNet module has significantly reduced the computational toll of classification on the Jetson, which is important as we will soon be running multiple modules in parallel that connect our different pieces of hardware.

References

[1] A. Francescon, ”GMapping and RPLIDAR”, geduino.org. http://www.geduino.org/site/archives/35 (accessed Apr. 15, 2022).

[2] A. Francescon, ”Navigation stack test: GMapping vs Hector Slam”, geduino.org. http://www.geduino.org/site/archives/36 (accessed Apr. 15, 2022).

[3] A. Stentz, ”Optimal and efficient path planning for partially-known environments,” *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, 1994, pp. 3310-3317 vol.4, doi: 10.1109/ROBOT.1994.351061.

[4] J. M. Santos, D. Portugal and R. P. Rocha, ”An evaluation of 2D SLAM techniques available in Robot Operating System,” *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 1-6, 2013, doi: 10.1109/SSRR.2013.6719348.

[5] M. O. Tătar, C. Popovici, D. Mândru, I. Ardelean and A. Pleşa, ”Design and development of an autonomous omni-directional mobile robot with Mecanum wheels,” *2014 IEEE International Conference on Automation, Quality and Testing, Robotics*, 2014, pp. 1-6, doi: 10.1109/AQTR.2014.6857869.

[6] S. Kohlbrecher and J. Meyer, ”hector_slam”, wiki.ros.org, http://wiki.ros.org/hector_slam (accessed May. 1, 2022).

[7] S. Kohlbrecher and J. Meyer, ”How to set up hector_slam for your robot”, wiki.ros.org, http://wiki.ros.org/hector_slam/Tutorials/SettingUpForYourRobot (accessed May. 1, 2022).

[8] S. Rosa, A. Russo, A. Saglinbeni and G. Toscana, ”Vocal interaction with a 7-DOF robotic arm for object detection, learning and grasping,” 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2016, pp. 505-506, doi: 10.1109/HRI.2016.7451828.

[9] S. Saat, W. A. Rashid, M. Tumari and M. Saealal, ”HECTORSLAM 2D MAPPING FOR SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM),” *Journal of Physics Conference Series*, vol. 1529, no. 4, April 2020.

[10] S. Kriman et al, ”QUARTZNET: DEEP AUTOMATIC SPEECH RECOGNITION WITH 1D TIME-CHANNEL SEPARABLE CONVOLUTIONS,” October 2019, doi: 10.48550/ARXIV.1910.10261, 22.

[11] S. Majumdar and B. Ginsburg, ”MatchboxNet: 1D Time-Channel Separable Convolutional Neural Network Architecture for Speech Commands Recognition,” October 2020, doi: 10.21437/interspeech.2020-1058.