[2022 Summer/Fall] KAIST URP Program

Individual Research Project Final Report

[Research Participation Project]

보행 정책 기반 통행가능성 예측을 통한
사족보행 로봇의 자율주행 구현

Autonomous navigation of quadruped robot through traversability estimation based on locomotion policy

Department of Mechanical Enginnering, Minwoo Cho(20180735)

For Submission

Research Subject (Korean) : 보행 정책 기반 통행가능성 예측을 통한 사족보행 로봇의 자율주행 구현

Research Subject (English) : Autonomous navigation of quadruped robot through traversability estimation based on locomotion policy

Research Period : June 27, 2022 ~ December 16, 2022

Advisory Professor : Daehyung Park (Signature)

Teaching Assistant : Jinwoo Kim (Signature)

As a participant(s) of the KAIST URP program, I (We) have completed the above research and hereby submit the final report on the research.

December 23, 2022

Researcher 조 민 우 **(Signature)**

Contents

Abstract

In this report, we presented an autonomous navigation framework that can be applied to both simulation and real environment using quadruped robot RBQ-5. First, we set up the robot and the simulation to develop our algorithms. And using FAST-LIO2 and OctoMap representation, we made autonomous navigation system based on metric map, that plans a path with obstacle avoidance. To obtain robust navigation intelligence, traversability prediction technology was tried by making dataset through IsaacGym and training the 3D sparse network.

1. Introduction

Currently, quadruped walking robot technology has evolved from the experimental operation level in a limited environment of the laboratory to the commercialization stage of launching sales products. Due to its potential of overcoming complex environments, various researches are currently ongoing in this field. In such background, this research aims to develop a robust autonomous navigation system for quadruped robot.

First, RBQ robot from rainbow robotics is set both in simulation and real world. External sensors and PC is attached to robot, and simulation is implemented in Gazebo. And then autonomous navigation framework is suggested using SLAM algorithms such as FAST-LIO2. To perform more robust intelligence in navigation, traversability estimation technics are adapted based on 3D occupancy grid map representation.

This research is done in Robust Intelligent and Robotics Lab, RIRO Lab. Main objective of RIRO Lab is developing human-centered robot that can collaborate and assist people through communication with robots from the perspective of non-experts. For this purpose, prior research was conducted in this lab to develop a technology that combines natural language processing technology with SLAM(Simultaneous localization and mapping) and TAMP(Task and Motion planning) to understand commands and make optimal plans, using mobile robot Haetae.

2. Related work

A. Quadruped robot

Quadruped robot technology is one of the most actively researched robot fields, and commercialization of Boston Dynamics' Spot and ANYbotics' ANYmal is already taking place in various companies and is even trying to enter the personal robot market. Unlike previously studied mobile robots, quadruped robots have the potential to overcome the terrain that mobile robots cannot go due to their freedom of stride in discontinuous terrain such as stairs or gravel fields. Because of these advantages, research is currently being conducted on various aspects, ranging from services for daily life to defense, which replaces humans in factories where harmful substances exist, and performs reconnaissance.

From the review of Park [1], research field of legged robots can be divided in three parts; actuator technology, locomotion control, and intelligent walking converged with environment perception. This work focuses on ‘intelligent walking converged with environment perception’. In this research field, Hwangbo’s team presented a fully autonomous navigation framework based on learning that safely works on narrow and complex environment[2]. In this work, future base coordinate and probability of collision was combined with joint torque. Marco Hutter’s team in ETH is also a leading group of this field, and presented real-time planning framework based on traversability estimation network that can be trained from 3d occupancy map that only relies on vision sensors[3]. Traversability estimation part of this report is based on work of [3].

B. SLAM

SLAM is the abbreviation of Simultaneous Localization and Mapping, which contains two main tasks, localization and mapping. It is a significant open problem in mobile robotics: to move precisely, a mobile robot must have an accurate environment map. The first major contribution to 3D SLAM was proposed as 'LiDAR Odometry And Mapping algorithm (LOAM)' [4], which applies scan-matching techniques to estimate the odometry, and performs point cloud registration to add the current scan's points to the existing map. This algorithm generates two outputs at different rates: The robot's location on the map is estimated 10 times per second, and the map is updated every second. RTAB-Map[5] is Graph-Based SLAM approach based on an incremental appearance-based loop closure detector. Works with lidar odometry and visual odometry very versatile, accepting inputs from RGB-D cameras, stereo cameras, or LiDAR sensors. FAST-LIO2[6] combines LiDAR and IMU motion to get good real-time performance. It estimates the ego-motion fusing LiDAR odometry and IMU odometry, using an efficient tightly-coupled Kalman filter. We tried to use GMapping ROS package[7] along with RTABMap to maintain basic SLAM performance, and then used FAST-LIO2 for fast map generation.

C. Traversability estimation

Work of [3] obtained intelligent walking of quadruped robot using traversability estimation. This term traversability illustrates the difficulty of driving through a specific region. Traversability estimation is a field that has long been studied through mobile robots. Such traversability estimation can be classified as non-trainable method, conventional machine learning method, and deep learning method.

In non-trainable method, grid-based representation method is used such as Wermelinger et al. [8], which constructed the traversability map by incorporating three fundamental terrain characteristics: slope, terrain roughness and step height. Other methods uses conventional computer vision, as Thrun [9] applied obstacle detection algorithm only from monocular color vision data.

Traversability estimation using machine learning has widely progressed. [? C. S. Dima, N. Vandapel, and M. Hebert, ''Classifier fusion for outdoor obstacle detection,'' in Proc. IEEE Int. Conf. Robot. Autom., (ICRA), Apr. 2004, pp. 665–671.] used three different classifiers to detect traversability human, obstacle and terrain combining multiple sensors such as lasers and camreas. Ability to cope with uncertainty is important in this research area, therefore probabilistic approach such as [10] has been done.

Deep learning methods are utilized to overcome the limits of conventional machine learning techniques since it can create implicit relationships among data. [11], self-supervised learning such as [12][13], or unsupervised learning such as [14] are been actively researched nowadays. Deep reinforcement learning approaches such as [15] are also tried too.

This research is based on supervised learning as an input of voxel-based representation. For such network, neural network of Wang et al. [16] is used, to operate on an octree volumetric data structure for shape analysis, restraining computations on occupied voxel octants. Sparse 3D CNNs restrain computation to occupied voxels [17]. These networks have been generalized to higher dimensions by Choy et al. [18]. We adopt the sparse encoder-decoder network of Gwak et al. [19] for semantic segmentation to predict traversability. The network utilizes sparse generative convolutions and pruning with sequential up-sampling and pruning voxels in 3D.

3. Quadruped robot Setup

A. Hardware and Mechatronics Setup

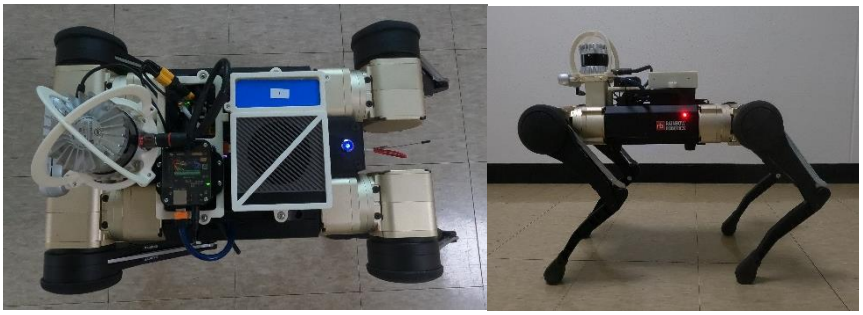

Figure 1. Image of RBQ robot. (Left : bird eye view, Right: right side view)

RBQ robot from Rainbow Robotics is used for this research. It weighs 25kg without any payload, and allows 3kg payload. Maximum movement speed is 8km/h, and can continuously walk more than 90 minutes. It is developed for surveillance or guidance purpose. Each joint of the robot leg is constitute of 3 motors, so total 12 motors are utilized for robot walking. [Figure 2] shows coordinate and pose of each joint. Internal computer of the robot performs force control in order to control and stabilize the robot from encoder data and internal IMU sensor data. Using the controller at [Figure 3], it gives velocity command to robot by TCP/IP based communications. Since we could not access to the internal computer, it was restricted to access to joint information for purpose such as locomotion policy learning, so giving velocity command was the only way to navigate the robot.

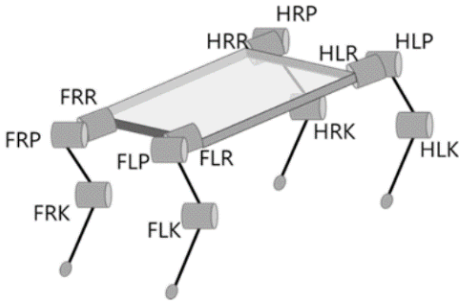

Figure 2. Joint structure of RBQ robot. First letter : F-front, H-back, Second letter : R-right, L-left, Third letter: R-x axis rotation, L-y axis rotation, K: knee rotation)

Figure 3. RBQ robot controller and emergency stop key.

Since this robot is domestically produced, it has potential of functional expansion with help of manufacturer. But it is still on development, the version we used has many limitations. For example, it cannot overcome the slopes that is steeper than 20 degree, and even for the slope with less angles than 20 degree, it must move to the rear direction. Also, it cannot climb the stairs, and has unstable gait patterns compared to other commercial quadruped robots.

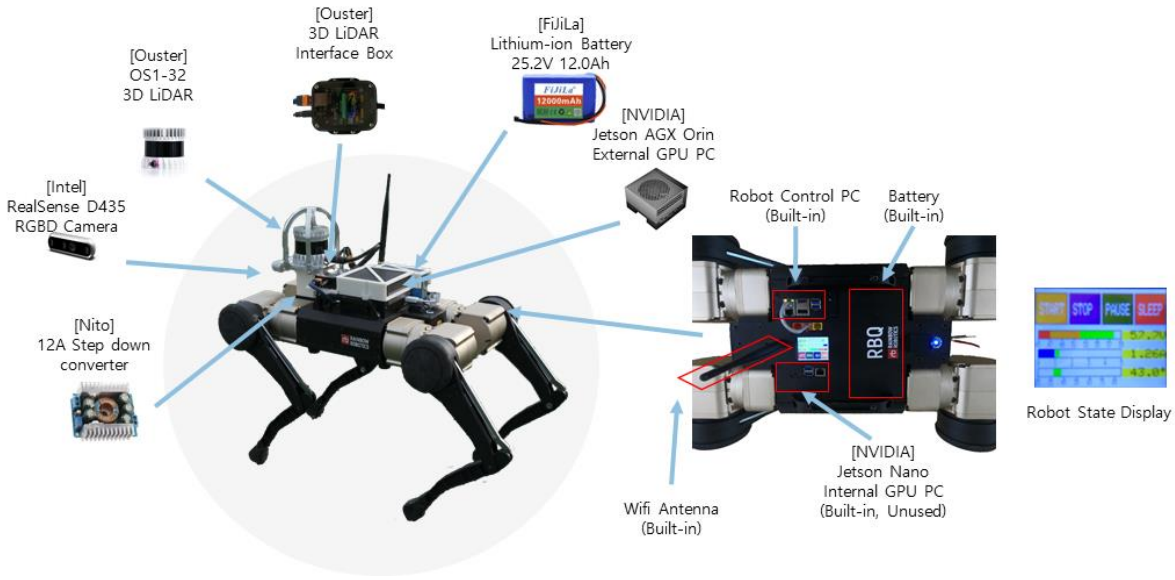

Figure 4. RBQ robot external hardware system

To perform additional tasks like autonomous navigation or object detection, additional hardware setup was required on the robot. To get environment information, LiDAR(Light detection and ranging/Laser imaging,detection, and ranging) sensor is required. 'OS1-32 3D LiDAR' is utilized for such purpose, which obtains 3D pointcloud data in 120m range with 32 channels. For image detection and to achieve advanced SLAM result, 'Realsense D435 RGBD camera' is used. It has resolution of 1280×720 and is suitable obtaining depth information of range of 0.3 ~ 3 m.

These sensor data are processed in external mini-computer of 'Jetson AGX Orin External GPU PC' that can work with low power. ROS(Robot Operating System) based programs are parallelly executed inside the PC to process the sensor data, and communicates each other based on TCPROS communication. After processing the sensor data, it delivers the command to built-in robot control PC through TCP/IP communication, where signal is propagated from extended Wi-fi antenna. All single processes are done in 1ms time, so that it doesn't make a problem in real-time demonstration.

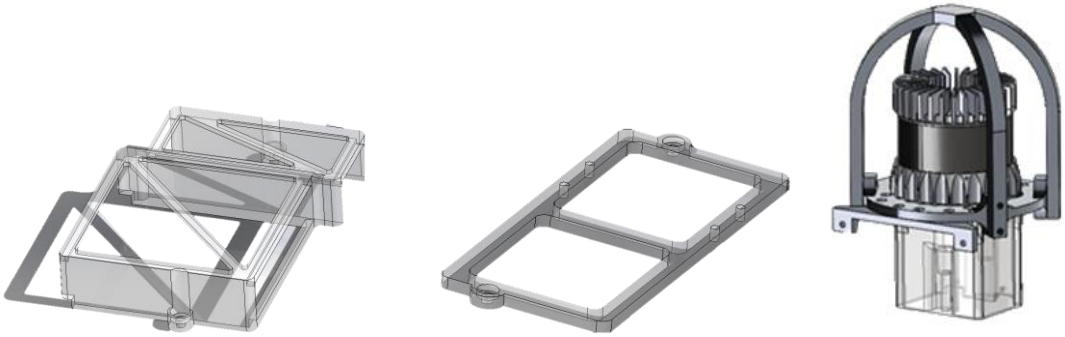

Figure 5. Mount structures for external hardware system. (Left : Mount for Battery and Jetson PC, Middle : Mount for LiDAR interface box, Right: Mount for LiDAR and converter.)

These external sensors and PC system is driven by single 12000mAh, 25.2V Lithium ion battery, along with 12A step down converter for LiDAR connection. It allows manipulation time of 40 minutes. To mount such devices, 3D printed models with ASA material is designed. Especially, AL 7075-T6 material is used for the LiDAR protection structure(Metallic arch shaped structure and it's bottom at right figure of [Figure 5]). This material has

ultimate tensile strength of 572 MPa, tensile yield strength of 503 MPa, and has density of 2.81 g/cm^3, so it is strong enough to protect the sensor and light enough to reduce the payload, at the same time.

B. Simulation Setup

Testing the programmed result directly on real-world takes a lot of effort and time, and failure can directly result robot damage. Therefore, simulation had to be set up. Simulation framework for quadruped robot, RBQ, is done at Gazebo, which supports ROS implementation so that we can test our ROS based programs on the simulation. This work used CHAMP package[20], which is a ROS package developed to promote the control of quadruped robots. It is an open-source development framework for building new quadruped robots and developing new control algorithms. It's internal locomotion planning algorithms and robot controls are based on Lee's implementation on MIT Cheetah robot[21]. This package supports simple navigation system based on ROS navigation stack of Gmapping[22] and move_base[23]. Therefore, this package is employed to demonstrate manipulation of RBQ robot in simulation world.

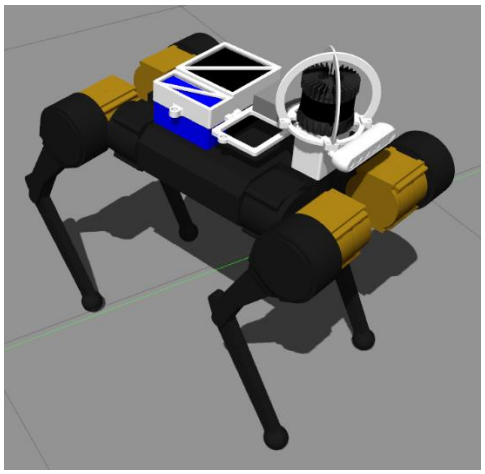

Figure 6. URDF visualization of RBQ robot in Gazebo

To develop the package for RBQ robot, first URDF model of the robot had to be created. It is done by combining mesh files of each parts, and the physical properties such as joint distance and inertial properties. The foot of the robot is designed as sphere to make continuous and soft contact with the ground. External hardware system had to be implemented along with the robot too. For the implementation of Realsense D435 camera, Intel® RealSense™ SDK 2.0 library[1] and realsense gazebo plugin[2] has been used. Implementation of Ouster 1 LiDAR is done based on Official ROS driver for Ouster sensors[3], referring Wil Selby's gazebo implementation of OS1-64[4].

Created URDF model did not immediately show successful performance in simulation. To make it work in the simulation additional gait parameters had to be tuned such as walking height and leg swing height. Those parameters are tuned and set to best mimic the motion of the actual robot. After configuring the gait parameters,

[1] https://github.com/IntelRealSense/librealsense

[2] https://github.com/pal-robotics/realsense_gazebo_plugin

[3] https://github.com/ouster-lidar/ouster-ros

[4] https://www.wilselby.com/2019/05/simulating-an-ouster-os-1-lidar-sensor-in-ros-gazebo-and-rviz/

PID gains of each joint controller had to be tuned. To tune the gain, first robot was hang above the ground and tested it's performance by tuning each leg's joint. After getting successful result without the ground contact, robot was tested on the ground and adjusted the parameters. Since internal control system of real RBQ robot and control system of CHAMP package is not identical, there still exists a domain gap between simulation and the real, but it showed performance enough to move to the goal with tele-operation in simulation.

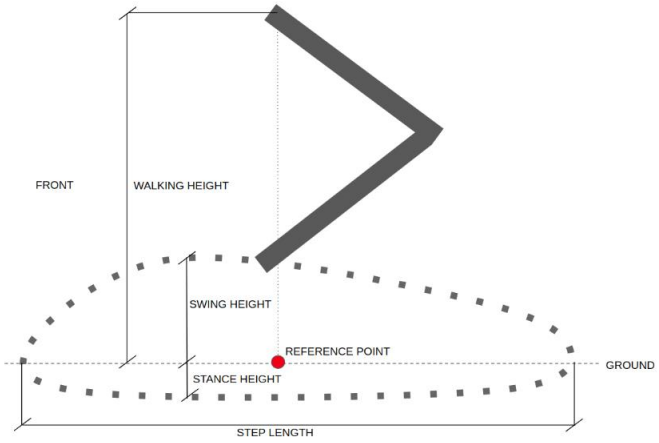

Figure 7. Gait configuration of quadruped robot system in CHAMP ROS package.

4. Autonomous navigation based on metric map

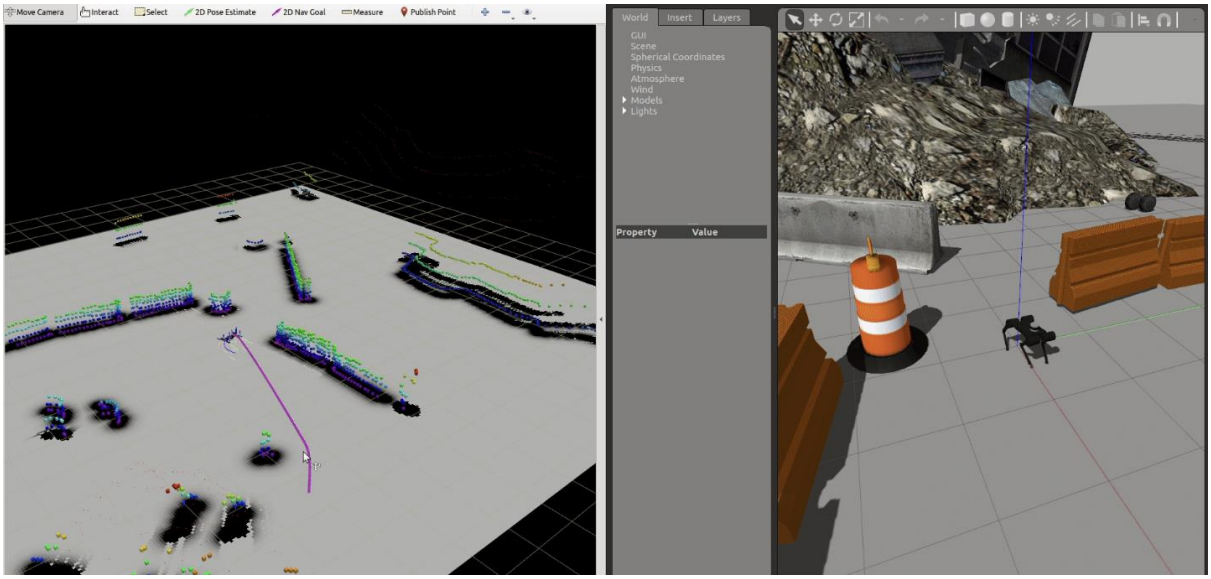

Figure 8. Navigation of RBQ robot in Gazebo Simulation using Gmapping and move_base. (Left: rviz visualization, colored boxes indicates 3D metric map and black area represents 2D occupancy-grid map. Purple line represents the planned path. Right: Gazebo simulation)

Basic navigation in simulation is supported in CHAMP package. [Figure 8] shows how such simulation works. But such algorithm has limitation in adapting to real world environment. Because it's basic performance, it is not fast enough and weak on sensor vibration. Since base of quadruped robot is not maintained horizontally while it walks, LiDAR shakes side to side, and Gmapping fails to localize at that time.

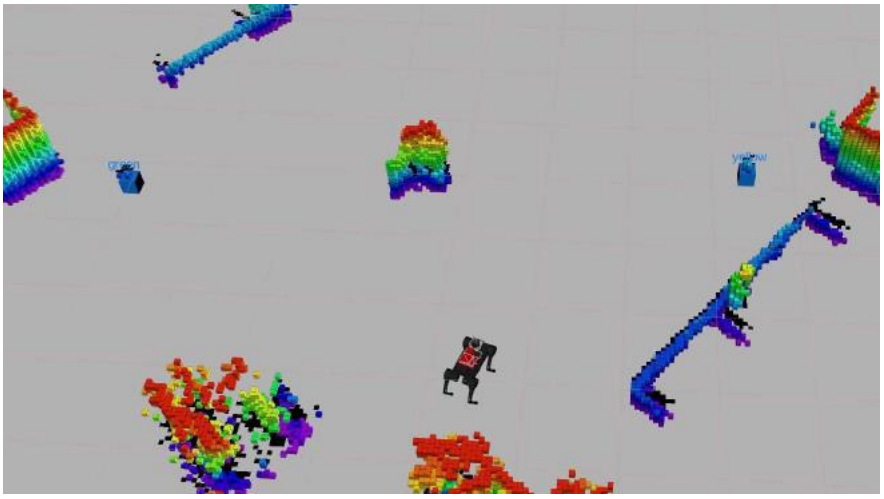

Figure 9. A screenshot of the metric map using FAST-LIO2. The colored points represent the 3D metric map. The gray area represents the 2D occupancy-grid map.

To enhance the performance, a state-of-the-art SLAM algorithm, FAST-LIO2[6] is adapted. This algorithm fuses the measurements from LiDAR and IMU sensors to estimate the robot pose relative to the odometry frame. The fast computation of FAST-LIO2 enables it to quickly process dense point clouds, resulting in a robust performance. Through FAST-LIO2, we obtain the points registered in the global map frame. Then, registered points are filtered along the z-axis coordinates to remove the ground and ceiling. Then 3D metric map is constructed by accumulating the filtered points using Octomap library [24] for efficient map management. Finally, 3D metric map is projected to get a 2D occupancy-grid map for navigation and add scene graph objects to the map.

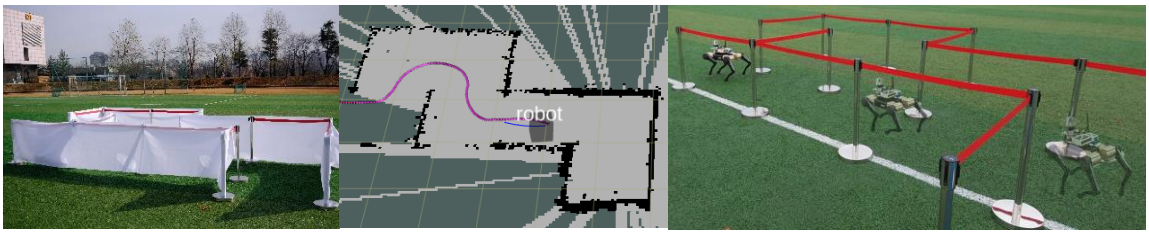

Figure 10. Navigation demonstration of RBQ robot in real world. (Left : Used environment, Middle: rviz visualization of robot during the navigation. Black points represent 2D cost map and purple line represent generated path.

To generates collision-free path based on the generated map, Dijkstra's algorithm [25] and DWA (Dynamic-Window Approach) local planner[26] is used. Based on the metric map, cost map is built showing obstacle regions by inflating the occupied points as the robot size. After generating the cost map, using Dijkstra's algorithm, the robot plans the shortest path to reach the goal without collision. The robot follows the path using the DWA local planner that determines the robot velocity command. [Figure 10] shows robot's autonomous navigation ability in complex real-world environment.

But such autonomous navigation based on simple obstacle avoidance algorithm has certain limitations. First, it

is doesn't guarantee optimal safety. Although it finds a collision-free path, it only finds a shortest path. So, such path can provide unsafe route that just a one step away from the obstacle. Also, it not suitable for environment that has complex vertical structures such as stairs or low ceilings. Since it just crops the z-axis and project the point clouds, it fails to detect multi-floor environment. And lastly, this algorithm can only adapted on plane ground, since points above the plane ground will be recognized as obstacles.

5. Autonomous Navigation based on traversability map.

To overcome the limitations of navigation by obstacle avoidance, traversability estimation techniques are used. This work is based on the work of [3]. To detect the multi-floor structures or overhanging obstacles, 3D occupancy grid map is used for the training, using 3D sparse CNN []. To train such network, data is collected by simulating the traversability in physical simulator, NVIDIA's IsaacGym. And traversability cost is defined as the success rate of reaching the desired goal pose.

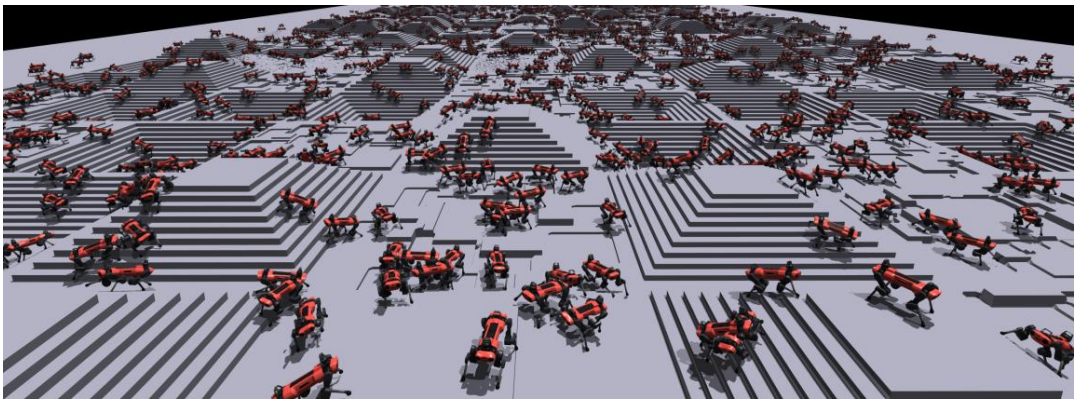

Figure 11. Training locomotion policy to walk in IsaacGym simulation. Picture from [27]

In the IsaacGym, we used simulation of ANYmal-C robot instead of real RBQ robot. This was because since we cannot control the joints in real world or neither get identical control algorithm in simulation, simulating with RBQ robot in IsaacGym was meaningless. Instead we used the model of ANYmal-C and used trained locomotion policy from [27]. (Figure 11)

A. Terrain Generation

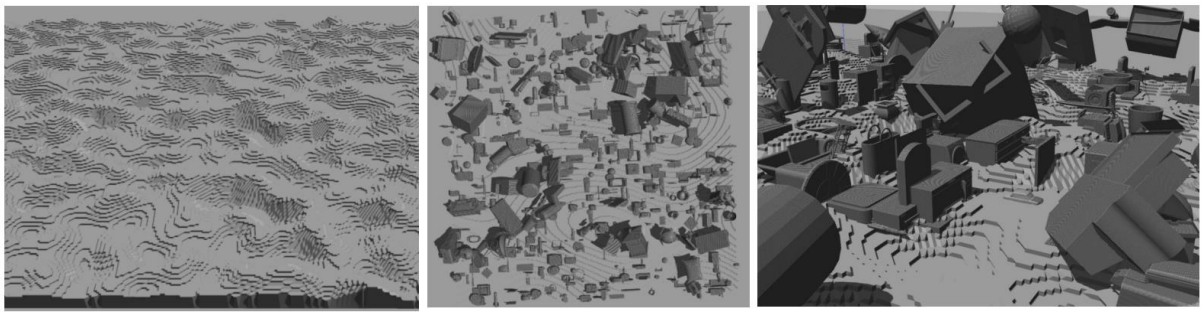

Figure 12. Terrain generation for traversability learning. (Left : Front view of ground terrain generated based on perlin noise. Middle : Bird-eye-view of randomly generated terrain. Right: Zoomed-in-view of randomly generated terrain)

For the simulation environment, the aim was to generate a large and diverse terrain with overhanging obstacles.

The ground structure of each environment was 32m*32m. The surface was randomly chosen between plane ground or Perlin noise ground, which is commonly used for terrain generation for training locomotion polices. Such Perlin noise ground is generated with random octaves, steps and frequency. 300~1000 random objects are spawned randomly above the terrain, which is based on the object meshes from ShapeNet[28] database. 90% of objects are aligned to the surface and scaled as a factor of 0.5~1.5. 10% of objects are randomly oriented and float above the surface between 0~3m high.

B. Starting pose evaluation

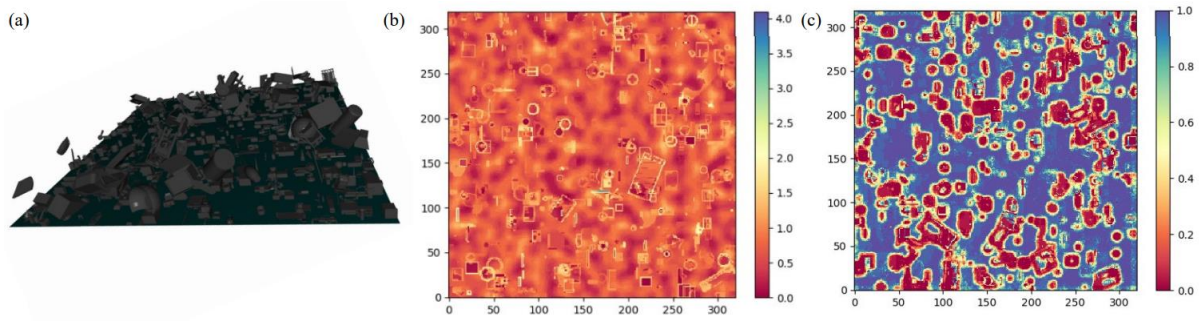

Figure 13. Starting point validation for traversability learning dataset. (Left: Voxelized terrain. Middle: Bird-eye-view of voxelized terrain. Colormap drawn based on the height. Right: Bird-eye-view of traversability evaluation. Colormap drawn based on success probability of standing(0 to 1)

Finding valid starting points are needed for accurate traversability data before traversability simulation. As a preprocessing procedure, terrain is voxelized including obstacles using binary voxelization(Figure 13.a). And then to validate the success probability of robot standing, robots are spawned for every point in a given map with precalculated heights in 0.1m resolution and 10 degree heading angle resolution. It is regarded as success if robot can sustain pose for 2 second of episode duration and considered as failure if force is exerted on base or base velocity exceeds threshold. For accurate result, it is tested multiple times for each point and heading angle.

C. Traversability data collection

After validating the safe starting pose of robot, traversability data is collected through simulation for each valid starting pose. For each starting point, 6 actions are simulated(Four translations in +x, -x, +y, -y direction. Two rotations in CW, CCW yaw direction, green arrows in middle figure of [Figure 4]). For each action, 10 repetition is tried, with adding randomness on initial base velocity and orientation, for robust learning. Traversability is calculated by total number of succeeded attempt / total attempt(60) on each starting position. Attempt is regarded as success if 1) force exerted on base and base velocity didn't exceed threshold, 2) robot succeeded to reach the goal in episode duration(4 sec). 5120 robots are simulated simultaneously in one terrain(32*32) in pararell. For one terrain, processing time 10hrs was required on average with headless run using RTX 2080 Ti as GPU.

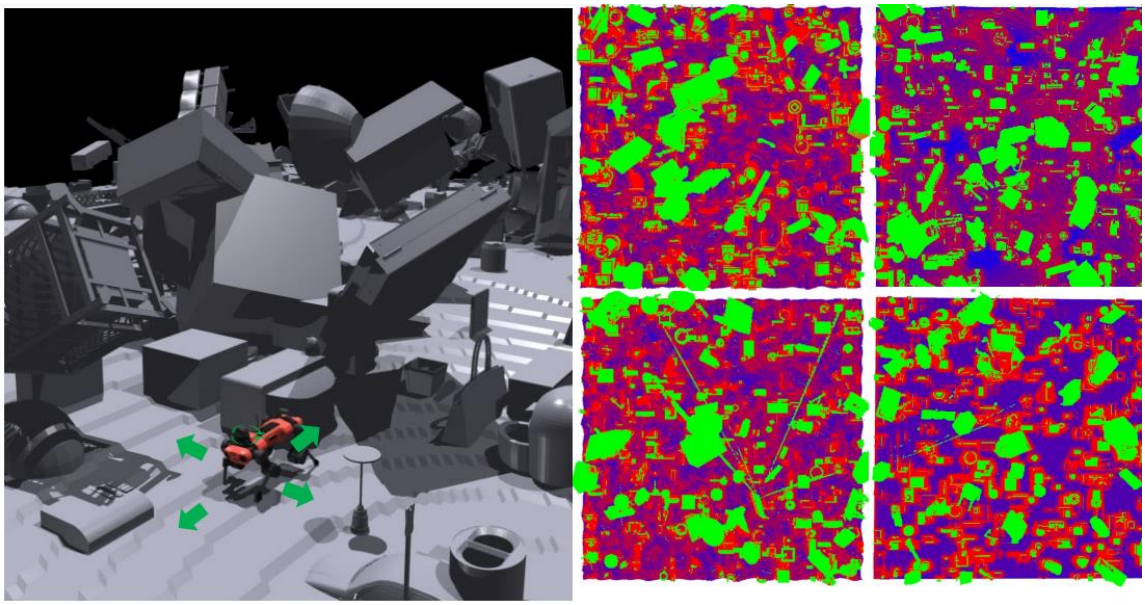

Figure 14. Traversability data collection example. Left: Traversability simulation example for single robot on randomly generated terrain with 6 actions(green arrows); Right: Example of two generated traversability maps of randomly generated terrains. Green structures are obstacles that are not regarded as possible spawn position. Red and blue colormaps are drawn based on traversability(0: red, 1: blue)

D. Traversability Prediction Network

In order to produce a traversability map using given voxel map data, Traversability Prediction Network(TPN) using generative sparse detection network is developed. [19]. It gets input as a 3D voxel-occupancy map(8m x 8m). We generated 160 maps for the input. Output of the network is traversability map which has same scale with an input. It follows an architecture of [Figure 15], U-net based semantic segmentation network with 2 skip connections, and consists of encoder-decoder structure. Especially, its decoder block has generative convolution transpose layer to upsample given feature. It uses MinkowskiEngine as a running engine for sparse tensor deep learning.[18] As an optimizer, ADAM with weight decay is used. Batch size was 8 and number of epoch was 20. Loss is measured by calculating Mean Squared Error(MSE) compared to ground truth data and adding the reconstruction loss. Accuracy at the validation set is also measured as MSE. Training was done on Google Colab.

.

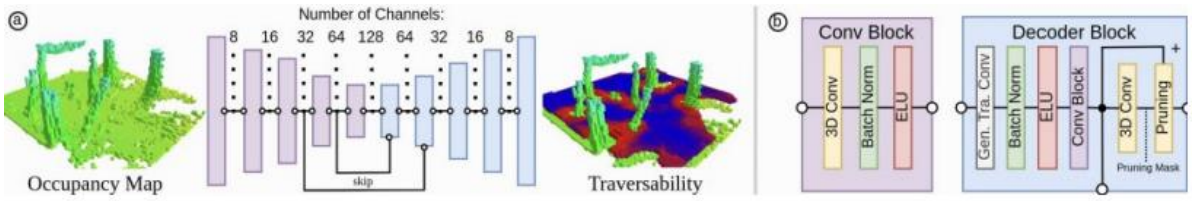

Figure 15. Left : Simplified Traversability Prediction Network Architecture. Right : Block structure of convolution block and decoder block. (Figure from [?])

[Figure 16] shows the loss graph of the training per each epochs. At each epoch, MSE to the validation data set remained under 0.02 and kept reducing. But we found that actual performance of TPN was below the expectation. [Figure 17] shows the visualization example of estimated traversability map. As the ground truth data(right figure) shows, nonzero traversability voxels should lie on the surface of the terrain. But from the network result(left), we can see undesirable nonzero traversability lies on surface of obstacles. This critical failure happened because of the generative transpose convolution block in the decode block(Right of [Figure 15.b]). Although the surface and

the obstacles are labeled separately for the inputs by labeling the obstacle voxels as zero, this convolution block unravels all the element so that the output result labels the zeros too. Therefore, nonzero traversability was labeled in undesirable places. To enhance this, we will make another mask layer that covers the obstacle surfaces, so that the 3D sparse network first segments the obstacle and the surface, and then labels the traversability on the surfaces in future works.

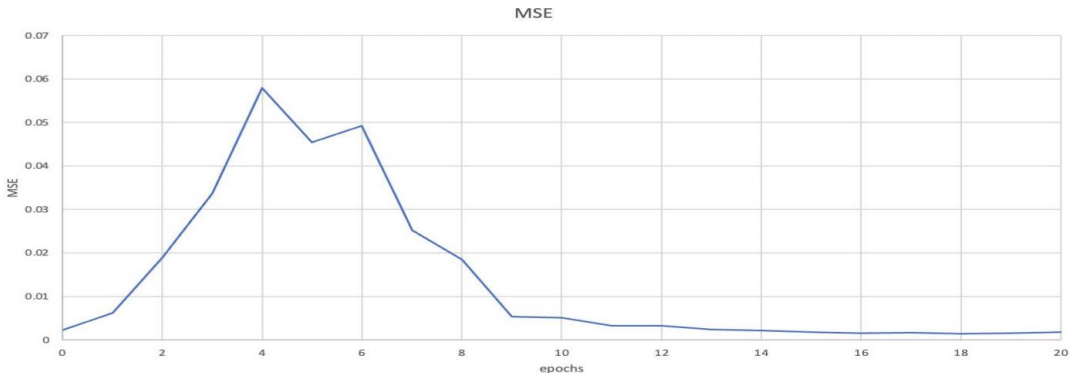

Figure 16. Loss graph of TPN. x-axis indicate epochs, and y-axis indicates MSE loss calculated from

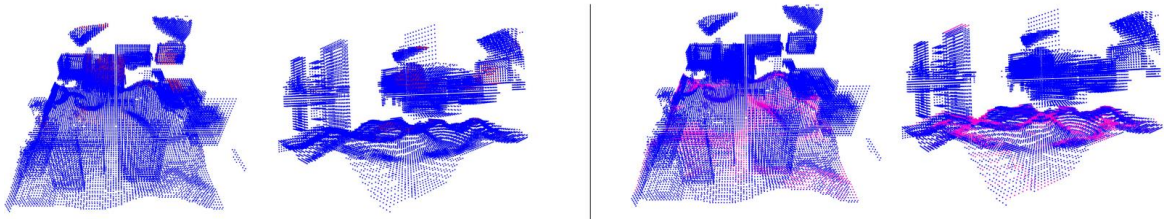

Figure 17. Figures of traversability of single 8m x 8m map in two perspectives. Left figures are results of TPN, and right figures show their GT measured from simulation. Red points indicates voxels with nonzero traversability, while blue points indicates indicate zero traversablity.

E. Path planning from traversability map

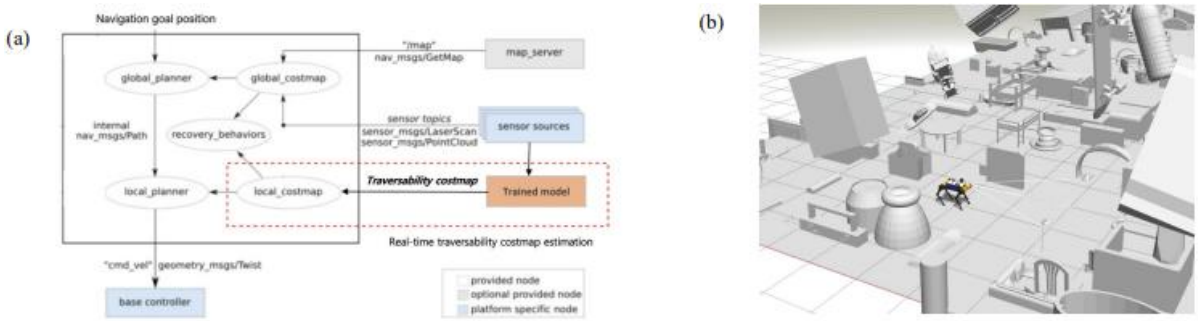

Figure 18. Global planning structure using traversability map based on ROS navaigation stack, (b) Path planning simulation from Gazebo on randomly generated terrain

Assuming the TPN has successfully achieved it's goal, global planning algorithm has been made based on ROS navigation stack package(Figure 18.a). Global planner is based on 2D grid search(A* search) and local planner is based on Dynamic Window Approach (DWA) algorithm. DWA samples robot state (Vx, Vy, Vtheta) and simulate for short range, evaluate with costs, and select highest score trajectory. For the costmap, traversability costmap is used to evaluate Generated local trajectories. Planning algorithm is first validated in Gazebo simulation environments which contains multiple objects from ShapeNet[?] models.(Figure 18.b)

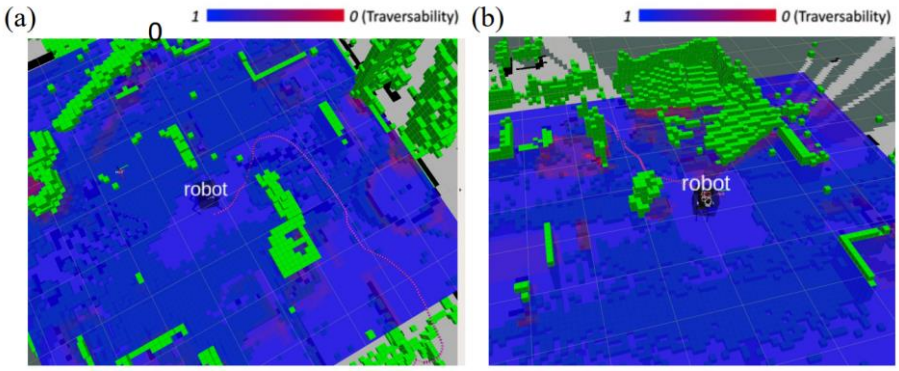

Figure 19. Path planning examplesusing traversability map on Gazebo simulation. Blue and red ground is traversability costmap projected to 2D map. Blue represents 1 traversability, and red represents 0. Purple line indicates planned path from robot. Green voxels represent object obstacles from LiDAR.

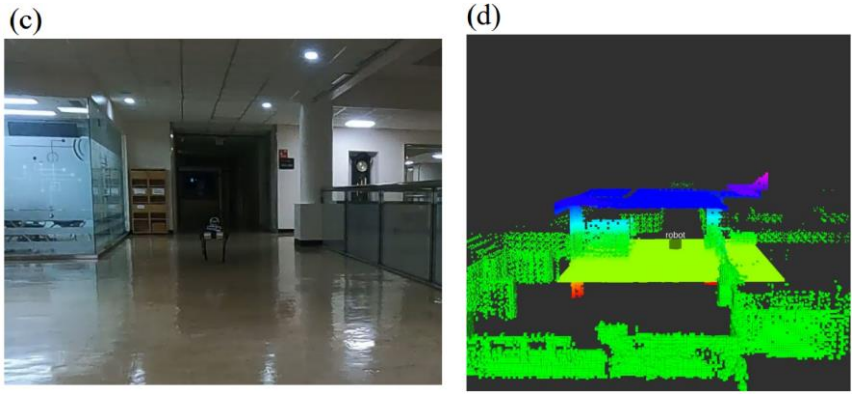

Figure 20. Local map generation for traversability estimation in real-word. Left : RBQ robot in building, : rviz visualization of situation of c).green voxels indicate the map generated from FAST_LIO[6] algorithm, and rainbow pointclouds indicate the local 3d voxel-occupancy map that can be go through TPN. Colormaps indicate the height from the ground.

Path planning was successfully done on gazebo simulation and showed that robot can traverse below over-hanging obstacles(Figure 10.a,b).10Hz real-time update was available using costmap, therefore it is expected that this planning algorithm will successfully work in real world too. Real-world implementation using RBQ robot, is shown at [Figure 20.d], which shows generation local 3d voxel-occupancy map that can be inserted into the TPN model. This map is generated from the Octomap that is generated from the previous section, by cropping 8m * 8m * 5m around the robot with point cloud of 0.1m resolution.

6. Conclusion

This work started to make a robot algorithm that can interact with environment and solve the problem that cannot be solved in conventional planning algorithms. But it is still on progress, so that work is not contained in this report. In this report, we presented an autonomous navigation framework that can be applied to both simulation and real environment. And then too obtain robust navigation intelligence, traversability prediction technology was tried by making dataset through IsaacGym and training the 3D sparse network. But it showed low performance due to network structure problem, so it will be improved in future work by adding masking layer in the convolution block.

Reference

[1] 박해원. (2019). ICRA 2019: 다족보행 로봇 연구동향. 로봇과 인간, 16(3), 12-17.

[2] Yunho Kim, Chanyoung Kim, Jemin Hwangbo. "Learning Forward Dynamics Model and Informed Trajectory Sampler for Safe Quadruped Navigation", Robotics: Science and Systems

[3] Frey, J., Hoeller, D., Khattak, S., & Hutter, M. (2022). Locomotion Policy Guided Traversability Learning using Volumetric Representations of Complex Environments. ArXiv, abs/2203.15854.

[4] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Realtime.

[5] Labb´e, M., Michaud, F.: Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. Journal of Field Robotics 36(2) (2019) 416–446

[6] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast Direct LiDAR-inertial Odometry," IEEE Transactions on Robotics, jul 2021.

[7] Grisetti, Giorgio, Cyrill Stachniss, and Wolfram Burgard. "Improved techniques for grid mapping with rao-blackwellized particle filters." IEEE transactions on Robotics 23.1 (2007): 34-46.

[8] L. Wellhausen, R. Ranftl, and M. Hutter, ''Safe robot navigation via multi-modal anomaly detection,'' IEEE Robot. Autom. Lett., vol. 5, no. 2, pp. 1326–1333, Apr. 2020.

[9] S. Thrun, ''Probabilistic robotics,'' Commun. ACM, vol. 45, no. 3, pp. 52–57, 2002

[10] M. Ollis, W. H. Huang, and M. Happold, ''A Bayesian approach to imitation learning for robot navigation,'' in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Oct. 2007, pp. 709–714

[11] Supervised learning approaches such as [? M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, ''From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots,'' in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Jun. 2017, pp. 1527–1533.

[12] D. P. Kingma and J. Ba, ''Adam: A method for stochastic optimization,'' in Proc. 3rd Int. Conf. Learn. Represent. (ICLR), 2015, pp. 1–15.

[13] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter, ''Where should i walk? Predicting terrain properties from images via self-supervised learning,'' IEEE Robot. Automat. Lett., vol. 4, no. 2, pp. 1509–1516, Apr. 2019

[14] C. Ye, ''Navigating a mobile robot by a traversability field histogram,'' IEEE Trans. Syst., Man, Cybern., B (Cybern.), vol. 37, no. 2, pp. 361–372, Apr. 2007.

[15] Z. Zhu, N. Li, R. Sun, D. Xu, and H. Zhao, ''Off-road autonomous vehicles traversability analysis and trajectory planning based on deep inverse reinforcement learning,'' in Proc. IEEE Intell. Vehicles Symp. (IV), Oct. 2020, pp. 971–977, doi

[16] P. Wang, Y. Liu, Y. Guo, C. Sun, and X. Tong, "O-CNN: octreebased convolutional neural networks for 3d shape analysis," CoRR, vol. abs/1712.01537, 2017.

[17] B. Graham, M. Engelcke, and L. van der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," CoRR, vol. abs/1711.10275, 2017.

[18] C. B. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," CoRR, vol. abs/1904.08755, 2019

[19] J. Gwak, C. B. Choy, and S. Savarese, "Generative sparse detection networks for 3d single-shot object detection," in European conference on computer vision, 2020.

[20] J. M. Jimeno, CHAMP, (2020), https://github.com/chvmp/champ

[21] Lee, J. (2013). Hierarchical controller for highly dynamic locomotion utilizing pattern modulation and impedance control: Implementation on the MIT Cheetah robot (Doctoral dissertation, Massachusetts Institute of Technology

[22] Grisetti, Giorgio, Cyrill Stachniss, and Wolfram Burgard. "Improved techniques for grid mapping with rao-blackwellized particle filters." IEEE transactions on Robotics 23.1 (2007): 34-46.

[23] da Silva Lubanco, D. L., Pichler-Scheder, M., & Schlechter, T. (2020, February). A novel frontier-based exploration algorithm for mobile robots. In 2020 6th International Conference on Mechatronics and Robotics Engineering (ICMRE) (pp. 1-5). IEEE.

[24] Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: Octomap: an efficient probabilistic 3d mapping framework based on octrees. Autonomous Robots 34, 189–206 (2013)

[25] Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271 (1959)

[26] Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE Robotics Autom. Mag. 4, 23–33 (1997)

[27] Rudin, N., Hoeller, D., Reist, P., & Hutter, M. (2021). Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning. ArXiv, abs/2109.11978.

[28] Chang, A.X., Funkhouser, T.A., Guibas, L.J., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., & Yu, F. (2015). ShapeNet: An Information-Rich 3D Model Repository. ArXiv, abs/1512.03012.