(2018) 1–10

Depth camera based navigation algorithms for indoor mobile robot

Tran Van Lien[1], Nguyen Viet Thang[1], Quach Cong Hoang[2], Phan Xuan Hieu[1], Pham Minh Trien[2]

[1]*Faculty of Information Technology, VNU University of Engineering and Technology, No. 144 Xuan Thuy Street, Dich Vong Ward, Cau Giay District, Hanoi, Vietnam*

[2]*Faculty of Electronics & Telecommunications, VNU University of Engineering and Technology, No. 144 Xuan Thuy Street, Dich Vong Ward, Cau Giay District, Hanoi, Vietnam*

Abstract

How can we efficiently search an object in a room? This report introduces a method for a single indoor mobile robot to find a hidden item based on states of the room when the robot is moving. A 2D distribution, called cognitive map, is built during robot movements to boost the exploring time. It is known that in the filed of exploring algorithms, A^* usually takes more time to reach the target than recent invented algorithms such as rapidly-exploring random trees (RRT) and probabilistic roadmap (PRM). However, by adapting the cognitive map as a cost map, the A^* algorithm is significantly improved and surpasses the two algorithms in Scannet 3D dataset. We also introduce application of depth sensors and SLAM solvers on reconstructing the room and updating cognitive map. By running a virtual robot in Gazebo simulator, it is proved that our method can work well on synthetic environment and hence, is very promising to be worked on real-life environment.

Keywords: Indoor mobile robot, Navigation, Cognitive map.

1. Introduction

Motivation

In real life, autonomous robot has a wide range of applications. Several practical examples can be listed: robotics arms in factory, people recuse after disasters, autonomous transport robot, etc. A traditional approach for self-governing robot is to divide it into sub-problems: data acquisition, localization, transfer acquired data to a higher level of understanding, decision making (path planning), motion control, etc.

Among these tasks; we chose to handle the navigation task for indoor mobile robot, which is a combination of localization and path planning. There are two main reasons of our decision: robot sensors and computer have signification growths nowadays; navigation is the final step before hardware control. For the first reason, Janai et al. [1] listed a lot of recently released data capture devices, which are ready for autonomous vehicles. Higher data quality would lead to more accurate understanding and smarter decisions. He

also mentioned big step of computer vision in processing these data within the help of deep learning. For the second reason, it is known that robot can be simulated and run virtually without real hardware control. Meyer [2] built an autonomous UAVs system that demonstrates the ability of building a complete self-driving robot without hardware. It means with careful simulations, a study on virtual is enough to prove the feasibility of work on soft parts. Additionally, we focus on indoor mobile robots since it is much cheaper to build compared to outdoor robots. The drawback of the indoor mobile robot is low nature of data. However, this environment can be manual refined and all the data can be retrieved and stored.

A survey is taken on some well-known navigation approaches: step by step, end to end, fuzzy logic and reinforcement learning [3, 4, 5]. It is figured out that the end-to-end and reinforcement learning approaches require an enormous amount of data to make good decisions. Fuzzy logic gives hidden and unpredictable decisions. The step-by-step approach divides the task into sub-tasks. Error would increase after each step, but the system is manageable and less depends on data. Having said in the previous paragraph, indoor data is not natural. And indoor mobile robot is low-cost hardware; hence, we chose a step-by-step approach to handle the navigation task for indoor mobile robot.

This report focuses on developing navigation algorithms. Based on cognitive map, a map created from previous step of navigation step, we design a modified A* algorithm to decide robot trajectory. To prove the feasibility of the idea, three types of experiment: on Scannet dataset, on Gazebo simulator and on synthetic maps are performed and presented in Chapter 4.

Problem statement

The problem can be formulated as below: let the robot to find an object O without its exact position. The robot has the input of positions of each object in the room, appearance distribution of object O (on each known object), RGB and depth images acquired by robot's cameras. Our main goal is to derive a trajectory for the robot in order to find the object O as soon as possible.

Specifically, Figure 1 illustrates the map with known fixed objects, an input of problem. In this figure, each pixel is labeled with a color identifying its object type.

2. Background

2.1. Autonomous robots

An autonomous robot is a robot which is designed to work on a specific environment in a long period of time without human

Figure 1. An example of fixed objects with their label.

intervention. There are many kinds of autonomous robot with their notable candidates that can be listed: Stanley, a famous autonomous vehicle [6]; NVIDIA self-driving cars and drones [3]; Bigdog, the robot that can move like a dog, created by Boston Dynamics; ASIMO, a human-like robot that is designed by Honda [7]; etc. Autonomous robots also have a variety in working environment. Some works in buildings like MIT RACECAR [8] while many other autonomous robots work outdoor. For example, most of NASA's robot in the moon are fully autonomous [9]. Yuh and Junku presented a number of studies for autonomous underwater robots [10] in 2000. The researches on making underground and mining robot automation are also noticeable as described in Bakambu et al. survey [11]. In this report, only indoor robots are being analyzed as they are affordable but still have a number of applications.

Indoor robots are designed for many purposes. Freund at el. mentioned some common scenarios such as: dangerous areas, rescue after disasters, transportation, health care, etc. Many robot systems can be applied to achieve a specific purpose. Some common autonomous systems are listed by Arai et al in their editorial [12]: single robot only, single robot with outside supplements, multi-robot with pair-wise communications, human-interactive robots, etc. For simplicity, this report focused on a single indoor robot with surroundings supplements such as observing system and workstations.

There are many approaches for building an autonomous robot. A common way is to derive the whole process into sub-tasks such as data acquisition, data processing, map reconstruction, path planning, robot control, etc. Each task can be implemented in the robot or outside device. Usually, data processing, path planning, robot controls are directly implemented in the robot. Data processing and map reconstruction can be handled by a strong computation device, and the results are forwarded to the robot. Recently, as the growth of hardware and software in this field, a strong single robot can handle all the task and learn how to give decisions without explicit rules. This approach is well-known as the end to end system. A notable research in this approach is NVIDIA "End to end learning for self-driving cars" [3]. For more understandings on the two approaches, brief introductions about mentioned tasks are presented in later parts in this section.

2.2. Data acquisition

2.2.1. RGB sensor

Color camera is one of the most common sensors in robot. The sensor takes light waves as input and after the quantization, the wave strength is converted into some numbers. Usually, they are represented as three values: red, green and blue. There are many reasons behind the decision of representing as R, G and B. Further details and analysis on color representing are extensively studied by Mukai at el. in their publication [13];

2.2.2. Depth sensor

There are two approaches to get depth information from the environment. The first is passive acquisition where the camera only receives data and infers the distance. A notable method in this approach is the stereo

camera where many cameras are used to reconstruct the 3D-like scene. The second is active acquisition. In this approach, sensors take some action in the environment and infer depth based on the responses. Microsoft Kinect v2 [1] is a famous example. This camera projects a bunch of light rays into the environment and calculates their time-of-flight. Based on the difference between the shooting and receiving time of each ray, the distance is calculated with a very high accuracy.

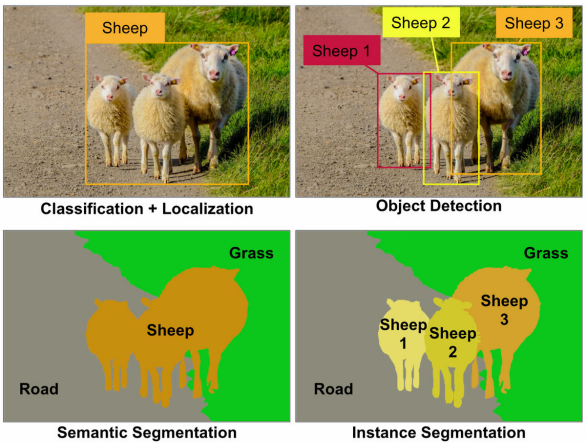

Figure 2. Some well-known computer visions tasks on RGB image

2.3. Data processing

RGB information

RGB image was used in the early steps of autonomous robots. Many features within the environment can be derived from the color image capture by robot camera. In this part, we present two examples of using RGB information in developing an autonomous robot.

In figure 2, many semantic meanings of the objects are retrieved. This kind of understanding can be used to reconstruct the map or derive a movement for the robot. There are some well-known studies on this step such as: Yolo9000, a fast object detection network by Redmon et at. [25]; ENet, a deep learning architecture for real-time semantic segmentation by Paszke et al. [26]; Scannet, a segmentation network for 3D scene by Dai et al. [27]

There is also a trend of using the methods on embedded Kit. In kit Jetson TX1, ENet performs 3.8 fps on a 1280×720 image and 21.1 fps on a 480×320 image. In kit Jetson TX2, Yolo9000 have a speed of 16-17 fps in a 1280×720 image.

Depth information

Depth information processing is a recent trend on autonomous robots. Despite a large number of studies on this kind of data, there still not a dominant representation of 3D data. Hartley mentioned some commonly used representations: voxel, polygon faces, octo-map and point cloud in his famous book "Multiple view geometry" [19]. Voxel is a three dimensions array which has the same idea as two dimensions array of 2D image. Polygon face is a common representation of objects in computer graphic. Octo map is the way of representing the 3D image as a tree. Quality of the image increase with the depth of the tree but also the memory consumption. Some recent applications of octo map in 3D image processing can be found in [28] and [24]. Point cloud is a sample of the original scene with a number of 3D points. This representation allows point cloud to have a

[1] https://en.wikipedia.org/wiki/Kinect

small memory consumption compared to other methods. However, the neighborhoods in point cloud representations are not explicit. In 2017, Qi et al. proposed PointNet, an efficient 3D image classification network using point cloud. The paper presented a method to make the use of 3D information and reduce the disadvantages of point cloud.

Apart from the representation, the work on 3D tasks such as reconstruction, navigation, semantic segmentation also has an increasing attention. In 2013, Gupta et al. published a work on RGB-D data perceptual organization and recognition [16]. In 2017, Dai et al. proposed a Scannet, method to reconstruct and get semantic meaning of objects in real environments with the help of depth information[27].

3. Method

Modifications on a-star algorithm

With mentioned geometric information (by RGB-D camera) and semantic meaning (data processing step) of objects in the environments, the robot can derive a belief about local objects. As the robot may not know the exact position of the goal, knowledge about current objects in its view and their relations with the goal plays an important role in navigation the robot. We formulated this belief as a cognitive map, a local probabilistic map p where each pixel in the camera view is assigned to a value in $(0, +\infty)$. The usage of the cognitive map in our method is presented first, then the details of cognitive map calculation.

As discussed in the previous chapter, A^* algorithm is a best-first search algorithm with the heuristic function:

$$f(n_i) = g(n_i) + h(n_i)\ \forall n_i \in Neighbors(c) \quad (1)$$

Where c is current position, $Neighbors$ is a set of positions that the robot can reach, $g(n_i)$ is path length taken by the robot from the source to n_i and $h(n_i)$ is an estimation of path length from n_i to the goal. The estimation h should guarantees that $h(x) \leq$ length of true shortest path from x to the target. Usually, $h(x)$ is measured as Manhattan distance for four neighbors, Chebyshev for eight neighbors and Euclidean for any-angle neighbors. In this study, we add a weight w to the of which value is a positive scalar:

$$f(n_i) = g(n_i) + w(n_i) * h(n_i)\ \forall n_i \in Neighbor(c) \quad (2)$$

Actually, the weight w is based on belief map p about the local environment. In our problem, the belief involves only two factors: geometric distance and semantic meaning. Values of p range from 0^+ to 1^- and w is calculated with formula 3

$$w(c) = \frac{1 - p(c)}{p(c)} \quad (3)$$

In equation 3, values of p in $(0, 1)$ are mapped into $(0, +\infty)$. It is easy to aware that when all positions have the same belief of $0.5, w(c) = 1\ \forall c$; hence, our algorithm is same as traditional A^* algorithm. As the algorithm mainly used the cognitive map, from now on, we will use cognitive map-based a-star algorithm (CMA) as the name of our method. It is noted that we set $w(c) = INF$, where INF is a big constant number, when $w(c)$ goes to infinity.

Cognitive map

In this report, we derive a simple version of the cognitive map based on objects' semantic meaning only. Assume we are finding the book. The probability $p(c)$ is defined as the distribution of appearance of the book in the whole map. Assuming there are $count_c$ separated objects of type c in the map and the book is put in that object N_c times throughout N surveys. The probability of containing the book in each instance of object c is:

$$p(c) = \frac{1}{N \times N_c} \sum count_c$$

Algorithm details

Actually, the cognitive map is divided into a set of objects that each object is a connected component of pixels with the same label. A specific object is assigned to a value in $(0, 1)$ where sum of all objects' probability is equal to 1. While exploring the map, the robot marks pixels in its field of view as visited. If an object is partly discovered; in other words, only some pixels of the object is visited, the its probability remains the same. However, when the robot fully discovered an object but could not find the target, that object is eliminated and its probability is redistributed to other non-visited objects.

Beside the prior knowledge of appearance distribution, the robot also take the advantage of the global map and RGB-D camera to retrieve distance to objects; hence, can calculate the exact position of each object in its field of view. As we are consider a grid map where robot can view and move to a position in its view, the distance is measured as Euclidean distance. Specifically, for a position *cur*, the estimation distance $h(cur)$ is evaluated by the formula:

$$h(cur) = ||cur - goal||$$

This estimation guarantees that the $h(cur)$ is always less or equal the length of true shortest path from *cur* to current *goal*.

With $p()$ and $h()$ are calculated as above explanation, we can evaluate the estimation $w(n_i) * h(n_i)$ for all neighbors of current position. From a set of non-visited neighbors, we choose the one with lowest value of $w()*h()$ to reach.

4. Experiment and evaluation

4.1. Experiment setting

The algorithm is experimented on Scannet dataset and customized grid maps. Figure 3 illustrates the scene 001 of Scannet dataset containing 1513 labeled scenes. Scene 001 has 23 objects: table, bed, chair, salon, etc. In this scene, we let the robot to find the a book with appearance distribution $p(x)$ where $p(table) = 0.2, p(bed) = 0.11, p(chair) = 0.04$, etc. means that on the average, the book appears on the table with a probability of 0.2, on the bed with a probability of 0.11 and so on.

To verify the feasibility of the algorithm, we deployed our idea on Gazebo simulator and real-life. Figure 4 presents Xbot, our real-life robot.

Figure 5 illustrates and overview of our virtual robot on simulator.

To verify the performance of our method, the cognitive map-based a-star algorithm (CMA) is compared two well-known algorithms. The first is greedy maximum

Figure 3. Scannet dataset scene 001

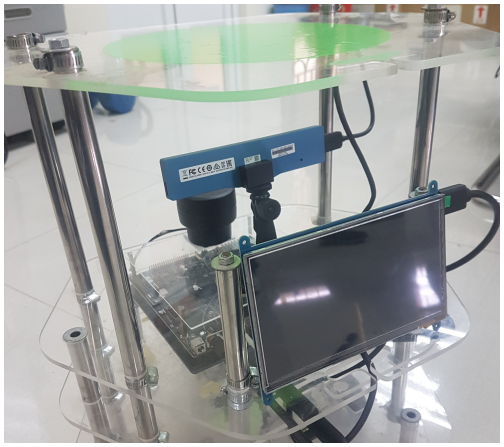

Figure 4. Xbot, the robot in real-life

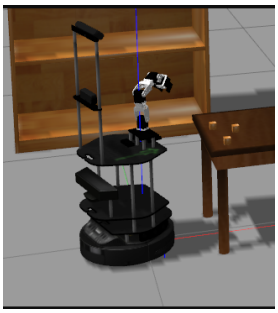

Figure 5. The robot on Gazebo simulator

probability (GMP). In GMP method, the robot always reach to the object with largest probability of containing the target. The second is rapidly-exploring random tree (RRT). In RRT, the robot set some milestones in the global map and tries to explore the map by traveling throughout the milestones as a tree.

4.2. Result

We recorded the results of 10000 queries for each algorithm in each map. Figure 6 demonstrates the expected path length of the algorithms on some notable maps. Figure 7 presents average run time of each algorithm on mentioned maps. Figure 8 describes the runtime of each process of our robot on Gazebo simulator. As can be seen from the figures, our CMA algorithm has slightly shorter path compared to other methods. To achieve the better paths, our method needs to spend larger computations; hence, longer runtime. However, Figure 8 shows that the path planning algorithms has much smaller runtime compared to image acquisition and object detection parts. That means the robot still works smooth with our method and the trade off is worth doing.

5. Conclusion

In conclusion, this report has studied a simple cognitive map-based A* (CMA) algorithm. This study has shown advantages of CMA over traditional A* and RRT algorithms where the semantic meanings of environment are retrieved. The method is experimented on Gazebo simulator, Scannet 3D dataset and one thousands custom grid map scenarios. It is turned out that our algorithm has shorter

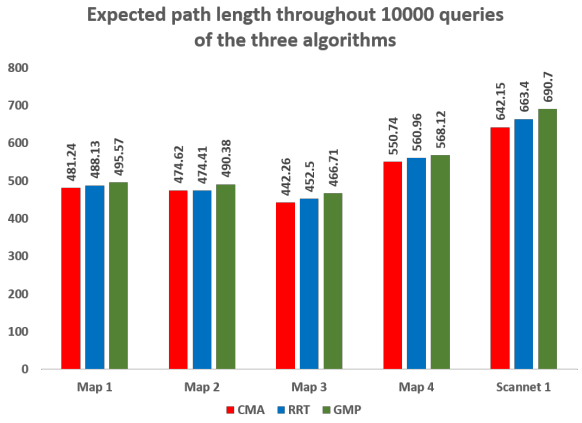

Figure 6. Expected path length of three algorithm

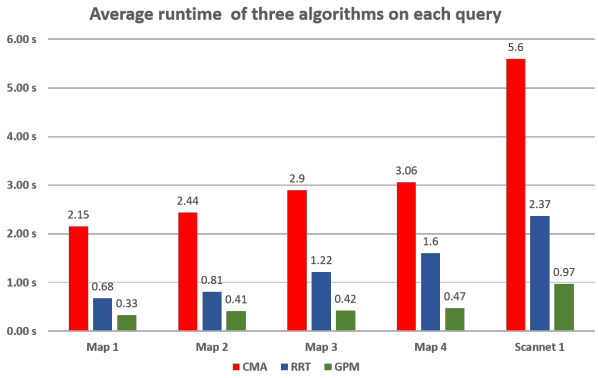

Figure 7. Average run time of three algorithms

exploration time compared to A* and RRT algorithms and can be run on embedded kit NVIDIA Jetson TX2.

Future work

For future direction, we are planning to compare the performance of the method with most-recent approaches and run the algorithm on the real-life robot. Here are the main follow-ups are considered to be taken care of right after the report.

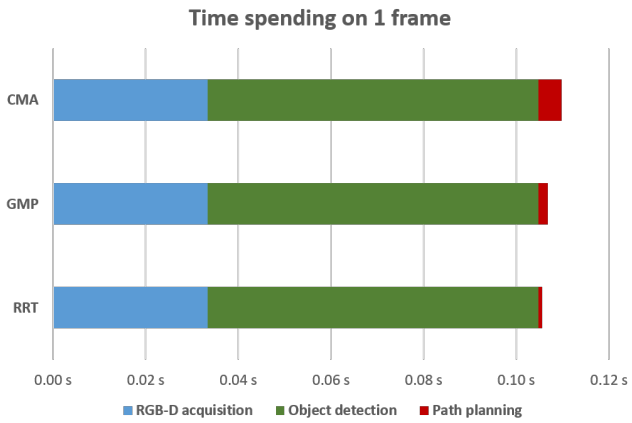

Figure 8. Runtime of each process or robot on Gazebo simulator

Comparisons with state-of-the-art methods

Some recent deep learning-based methods are mentioned in this report. Nonetheless, they are not appeared in the experiment because of their huge resource consumption. For example, cognitive mapping and planning method by Gupta et al. [17] requires at least 500GB of disk to store a scene (out of five) in the dataset including pre-trained models, raw XYZ point cloud of the whole room, annotations of objects. This is due to the methods are currently designed for end-to-end system where strong computational devices are always available.

It is a luck that all the datasets above are opened for all people to access and most of the benchmarked scenes in the paper are labeled. With current workstation, we hope that a deep learning-based method will be performed to take a comparison between an end-to-end and a step-by-step approach.

Lanching the real-robot

Currently, Xbot can run some simple autonomous tasks such as lane marker detection, people detection, etc, run in some custom lanes. However, it is impossible to launch a sequence of autonomous processes from acquiring the data to giving the decision. This is due to our incomplete work on 3D SLAM task. Currently, we are focusing on this task and when the whole room is reconstruct, the robot can be launched as the one in synthetic environments.

Acknowledgements

We would like to thank Dr. Do-Van Nguyen for reading the manuscript and giving us useful comments.

References

[1] J. Janai, F. Güney, A. Behl, A. Geiger, Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art, CoRR abs/1704.05519. arXiv:1704.05519.
URL http://arxiv.org/abs/1704.05519

[2] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, O. Von Stryk, Comprehensive simulation of quadrotor uavs using ros and gazebo, in: International Conference on Simulation, Modeling, and Programming for Autonomous Robots, Springer, 2012, pp. 400–411.

[3] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al., End to end learning for self-driving cars, arXiv preprint arXiv:1604.07316.

[4] H. R. Beom, H. S. Cho, A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning, IEEE transactions on Systems, Man, and Cybernetics 25 (3) (1995) 464–477.

[5] J. Dixon, O. Henlich, Mobile robot navigation, Information Systems Engineering Year, Imperial College 2 (1997) 1–10.

[6] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, et al., Stanley: The robot that won the darpa grand challenge, Journal of field Robotics 23 (9) (2006) 661–692.

[7] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, K. Fujimura, The intelligent asimo: System overview and integration, in: Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on, Vol. 3, IEEE, 2002, pp. 2478–2483.

[8] R. Shin, S. Karaman, A. Ander, M. T. Boulet, J. Connor, K. L. Gregson, W. Guerra, O. R. Guldner, M. Mubarik, B. Plancher, et al., Project based, collaborative, algorithmic robotics for high school students: Programming self driving race cars at mit, Tech. rep., MIT Lincoln Laboratory Lexington United States (2017).

[9] B. H. Wilcox, T. Litwin, J. Biesiadecki, J. Matthews, M. Heverly, J. Morrison, J. Townsend, N. Ahmad, A. Sirota, B. Cooper, Athlete: A cargo handling and manipulation robot for the moon, Journal of Field Robotics 24 (5) (2007) 421–434.

[10] J. Yuh, Design and control of autonomous underwater robots: A survey, Autonomous Robots 8 (1) (2000) 7–24.

[11] J. N. Bakambu, V. Polotski, Autonomous system for navigation and surveying in underground mines, Journal of Field Robotics 24 (10) (2007) 829–847.

[12] T. Arai, E. Pagello, L. E. Parker, Advances in multi-robot systems, IEEE Transactions on robotics and automation 18 (5) (2002) 655–661.

[13] T. Mukai, M. Yamada, S. Nakamura, Characteristics of ingan-based uv/blue/green/amber/red light-emitting diodes, Japanese Journal of Applied Physics 38 (7R) (1999) 3976.

[14] I. P. Howard, Perceiving in depth, volume 1: basic mechanisms, Oxford University Press, 2012.

[15] N. Silberman, R. Fergus, Indoor scene segmentation using a structured light sensor, in: Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on, IEEE, 2011, pp. 601–608.

[16] S. Gupta, P. Arbelaez, J. Malik, Perceptual organization and recognition of indoor scenes from rgb-d images, in: Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, IEEE, 2013, pp. 564–571.

[17] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, J. Malik, Cognitive mapping and planning for visual navigation, arXiv preprint arXiv:1702.03920 3.

[18] K. Khoshelham, S. O. Elberink, Accuracy and resolution of kinect depth data for indoor mapping applications, Sensors 12 (2) (2012) 1437–1454.

[19] R. Hartley, A. Zisserman, Multiple view geometry in computer vision, Cambridge university press, 2003.

[20] R. B. Rusu, S. Cousins, 3d is here: Point cloud library (pcl), in: Robotics and automation (ICRA), 2011 IEEE International Conference on, IEEE, 2011, pp. 1–4.

[21] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, Proc. Computer Vision and Pattern Recognition (CVPR), IEEE 1 (2) (2017) 4.

[22] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: Advances in Neural Information Processing Systems, 2017, pp. 5105–5114.

[23] J. Biswas, M. Veloso, Depth camera based indoor mobile robot localization and navigation, in: Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, 2012, pp. 1697–1702.

[24] G. Riegler, A. O. Ulusoy, A. Geiger, Octnet: Learning deep 3d representations at high resolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vol. 3, 2017.

[25] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, arXiv preprint 1612.

[26] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, Enet: A deep neural network architecture for real-time semantic segmentation, arXiv preprint arXiv:1606.02147.

[27] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Nießner, Scannet: Richly-annotated 3d reconstructions of indoor scenes, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2017.

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