Jurnal Kejuruteraan SI 3(1) 2020: 59-64
https://doi.org/10.17576/jkukm-2020-si3(1)-09

Development of Floor Mapping Mobile Robot Algorithm Using Enhanced Artificial Neuro-Based SLAM (ANBS)

Khairul Anuar Juhari[a]*, Rizauddin Ramli[b], Sallehuddin Mohamed Haris[b], Zunaidi Ibrahim[c] & Abdullah Zawawi Mohamed[d]

[a]*Department of Industrial Electronics, German Malaysian Institute (GMI), Malaysia*
[b]*Department of Mechanical and Manufacturing Engineering, Faculty of Engineering & Built Environment, Universiti Kebangsaan Malaysia, Malaysia*
[c]*Sustainable Advanced Manufacturing, Faculty of Technology, University Sunderland, United Kingdom.*
[d]*Petronas Research Sdn Bhd, Kawasan Institusi Bangi, Malaysia*

**Corresponding author: khairul_anuar@gmi.edu.my*

Received 10 October 2019, Received in revised form 20 May 2020
Accepted 05 June 2020, Available online 30 September 2020

ABSTRACT

A complex and expensive system in floor mapping mobile robot platforms are the challenges in this age of technology revolution. Sensors that are equipped with the robot could be different, the complexity of the algorithm and the robot performance itself are not adequate. In this paper, we present an efficient way with an economically cost-saving mobile robot floor mapping system based on simultaneous localization and mapping (SLAM). The paper will highlight implementing a Rplidar sensor with a floor mapping mobile robot platform with the enhanced error corrections based on the Artificial Neuro-Based SLAM (ANBS) algorithm. The proposed system runs on Robot Operating Systems (ROS) and Tensor Flow programming. The experimental results showed how the different controllers can be improved by adding the ANBS algorithm which intelligently filtering the unnecessary error and produce the precise output on the map. The different controllers also can be used with this algorithm. For this research, the ANBS are tested on Hector SLAM and Gmapping SLAM where the output produced by each SLAM method is fed into the ANBS algorithm. At the end of the experiment, the ANBS improves the output result by 14.67% for Hector SLAM and 17.36% for the Gmapping SLAM and produces a precise map than ever before. In the future, there will be more SLAM method can be embedded with this ANBS algorithm.

Keywords: Mobile Robot; Tensor Flow; SLAM; Gmapping; ANBS algorithm

INTRODUCTION

A different model of environment and controller approach can be very difficult to achieve when dealing with an autonomous robot. The autonomous capability of a mobile robot to map in real words within its environment can be very difficult to achieve due to many reasons (Dissanayake et al. 2011; Rodin & Stajduhar 2017). An autonomous robot consumes the high-speed microprocessor to do the onboard computational algorithm based on the different controllers when doing simultaneous localization and mapping (SLAM). Furthermore, the investment in this technology might not cheap due to the high-end sensors, actuators, and computers that are needed for the robot to do SLAM. The mapping is a process where the robot creates a graphic model or its surroundings. This graphic model can be any dimensional presentation as such proposed in (Kamarudin et al. 2014; Mur-Artal et al. 2015) the mapping can be done using laser scanners and since the robot operation system (ROS) has been introduced, this system helps many researchers to complete the robot without investing a lot in hardware (Kohlbrecher et al. 2012). Thus, it makes the research gap shorter than usual. With ROS, the mathematical model, simulations and hardware integration can be done with minimum cost (Achmad et al. 2016). Hence, the ROSbot used in paper runs on ROS, thus represent a module that can be used to enhance the output of the floor mapping bile robot based on a different controller. An artificial neuro-based SLAM (ANBS) has been used to improve the output provided by a different controller. A convolution in between two functions of the robot could produce the third function which represents the output (Dewi, et al. 2017). In this case, is the 2D map. In (Tian, et al. 2019) the RGB-D SLAM methods have been used to match the environment and the robot maps. The input is taken from the Asus Xtion Pro RGB-D sensor, while Ran (2017), used a vision-based method to produce the map output.

Basically, at the end of every SLAM process, the output of the robot position P is represented by equation 1.0 where the and represent the coordinate of the robot frame position while is the angle of the robot heading based on a global map position. The measurable distance relative to the robot is calculated by a different controller. In this paper, we have tested the ANBS with the Hector SLAM and the Gmapping SLAM.

$$P = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{1}$$

Equation 1 has been used by most of the floor mapping mobile robots (Cadena, et al. 2016; G. 2011). For the robot to recognize the distance of objects, the prediction of the robot position is a must. From the robot position, the point at which the sensor detects as an object can be determined by measuring the distance. The method of continuous scanning and matching of the post estimation by the distance measuring sensor which will return and update the transform could be challenging. The working environment of the robot could be very difficult for it to determine its position.

Many methods of SLAM have been introduced by previous researchers. Rik Claessens (2013), has introduced the methods called Graph-based SLAM using the turtlebot platform. The robot used the odometry data with other sensor data and based on the scan matching method, the landmark could be obtained. This method matches two measurements over time, therefore it requires high resolutions of laser scanner such as Hokuyo laser scanner and a powerful computer to generate the map in real-time. Furthermore, there are a bunch of methods and techniques that have been developed recently by previous researchers for SLAM (Ran et al. 2017; G. 2011; Aravindan, et al. 2016; Kamburugamuve et al. 2016). In mobile robotic applications, most of the methods proposed have different approaches and designs. The system also based only on a certain part of the mechanical structure of the mobile robot and is lacks control robustness inadequate a new and different platform.

Thus, if any changes are made to its mechanical parts, the controller has to be tuned and reprogram to suit with the robot (Abdelrasoul, et al. 2017). For example, Cadena (2016) described the concurrent SLAM to commit future directions and the needs of the industry involving the SLAM robotics. The robustness and long term scalability, especially in SLAM for a mobile robot, has played the main role for the researchers to find the research gap.

Extended Kalman Filtering, Gmapping SLAM, Hector SLAM, Graph-Based SLAM, and Direct monocular SLAM are the most SLAM method that is used in mobile robot today. Furthermore, these methods allowed for certain sensors and types of a robot and some of it doesn't allow the output position and mapping to be easily caught by the open-source software. As mentioned by Kohlbrecher (2014), a standard module in developing a mobile robot for searched and rescue requires hardware dependency and software localization system. With support from the community, the module became an open-source so other developers could benefit from the system.

METHODOLOGY

TECHNICAL SPECIFICATIONS

The changing scenario of the landmarks and field makes the robot suffer to do mapping its environment. To simplify the system, the proposed robot is equipped with 2 wheels without encoder and laser range sensor as shown in Figure 1.

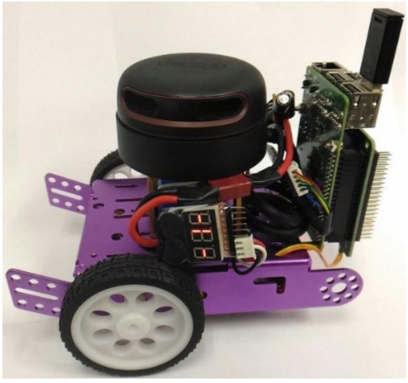

FIGURE 1. The ROSbot

The robot uses a battery and runs on single board computer Raspberry Pi 3 Model B. The system also runs on raspbian stretch, ROS kinetic and Tensor Flow software on board.

TABLE 1. Standard operating step for Running ROSbot.

	Steps for running ROSbot
1.	Bring up the ROSbot platform
2.	Bring up the navigation control for ROSbot
3.	Start Hector SLAM or Gmapping
4.	Obtain the map and save
5.	Repeat steps 5 until 7 with different SLAM method

In this paper, we emphasize on the improvement of the Hector SLAM and Gmapping using the proposed method. To build the map using the robot, there are few steps needed as shown in Table 1.

HARDWARE OPERATIONS

The localization system used for ROSbot is based on Artificial Neuro Based SLAM (ANBS). Hence the robot will navigate in an unknown environment, therefore Hector SLAM and Gmapping will be used in comparison for generating the map based on the ANBS algorithm.

The experiment environment setup was build up at the first stage using the simulation Gazebo from ROS. Since Gazebo is a 3D robot simulator and has a bunch model of robot kinematics in its library, therefore it is suitable to test the robot algorithm and doing experiment. The actual setup of the field is shown in Figure 2.

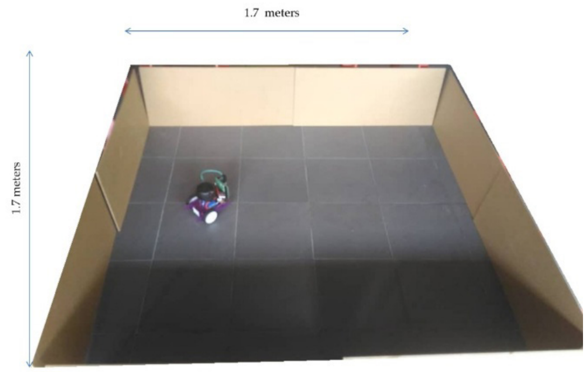

FIGURE 2. Actual robot set up in a square size arena

The size of the experiment arena is limited to 1.7meter x 1.7meter in the square since the size of the robot is small and to minimize the standard deviation. There is only a solid wall surrounding the arena.

This setup will then be simulated in Gazebo and the simulation results can is used as a pre-testing rather than using the actual robot. The map output which is produced will be display using the RViz. RViz is the 3D visualizer for displaying sensor data and state information from ROS. Before starts, there are few steps to follow as shown in Table 2.

These steps are a standard operating procedure to operate the robot so that the steps are always the same.

TABLE 2. Steps to running ROS and mapping environment in simulation

Steps for running mapping environment in simulation	
1.	Bring up the roscore in a new terminal
2.	Start anbs_hector or anbs_gmapping in a new terminal
3.	Start rosbot_teleop for control the robot
4.	Move the robot with a joystick
5.	Record the map with Rviz and save the map

RESULTS AND DISCUSSION

MATCHING THE ENVIRONMENT BY USING HECTOR SLAM

The environment at each map of the robot matching in Hector SLAM is performed based on the Gaussian-Newton minimization method (Eliwa, et al. 2017). This method measures the optimum distance of the laser distance relatives to the map build. By getting the transformation, the coordinate of each point of the laser can be represented as:

$$\tau = (P_x, P_y, \Upsilon)^T \quad (2)$$

Where τ is the transformation, (P_x, P_y, Υ) are the coordinate of the laser endpoint. From (2), the error is expressed in (3) where M represents the Map, s_i which is the laser endpoint.

$$\tau = \sum_{i=1}^{n} [1 - M(s_i(\tau + \Delta\tau))^2 \rightarrow 0 \quad (3)$$

Due to the accuracy of the LIDAR sensor and its high update rate, the algorithm which wasn't a closed loop can be satisfied to be used with the Hector SLAM (Kamarudin, et al. 2014). However, when dealing with a dynamic environment, the performance of the algorithm is lower down as it closes its loops in the real world scenario.

This has been declared by the previous researchers (Kamarudin, et al. 2014). By applying the first-order Taylor expansion $M(s_i(\tau + \Delta\tau))$, the partial derivative concerning $\Delta\tau$ is equal to the Gauss-Newton equation minimization problem as in Equation 4 and the value of H is represented by Equation 5.

$$\Delta\tau = H^{-1} \sum_{i=1}^{n} [\nabla M(s_i(\tau)) \frac{\partial s_i(\tau)}{\partial \tau}]^T [1 - M(s_i(\tau))] \quad (4)$$

$$\left[\nabla M(s_i(\tau)) \frac{\partial s_i(\tau)}{\partial \tau} \right]^T \left[\nabla M(s_i(\tau)) \frac{\partial s_i(\tau)}{\partial \tau} \right] \quad (5)$$

By considering Equation 4 as an input to the feedforward for the convolution neural networks input, thus the precise coordinate of the laser end can be estimated precisely based on the convolution of neural networks as proposed Ran (2017).

MATCHING THE ENVIRONMENT USING GMAPPING SLAM

Different from the Gmapping, it used the scan matching method with a grid map based algorithm post estimation. However Gmapping requires odometry data but in (Esenkanova, et al. 2013) have proposed a Gmapping algorithm without using the odometer data. By default, the parameter values for linear update and the angular are 1.0 and 0.5 respectively. Changing the angular update with another value can change the results as shown in Figure 3.

In Figure 3, it shows two different maps with different sets of the angular update with the updated value is 0.7 while keeping the linear update value unchanged. From the results, the map update rate is fast and the Gmapping can correct the map when the encoder has misalignment. However, when the encoder fails to be read by the controller, the updated map is messy. This can be avoided when using the Hector SLAM.

The output from Hector and Gmapping SLAM is fed into the ANBS algorithm. The input form is based on equation 1. The ANBS algorithm used the convolution matrices method where it calculates the percentages of scan matching images and compare them with the actual sources. When the robot use hector slam or Gmapping SLAM, the output results are justified according to the output graph based on the map accuracy.

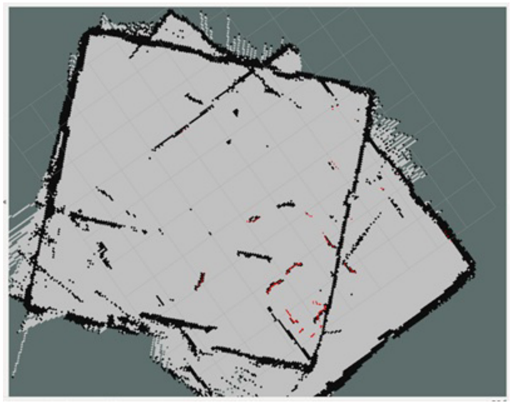

FIGURE 3. Gmapping map results with new parameters

As shown in Figure 4(b), the graph represents the distance measured over the number of sampling with the standard deviation, σ which is 8.544211. The % of the difference λ is 22.7579% between the measured values and the Hector values. Different from the results shown in Figure 5(b). The standard deviation of the results σ is 11.323996. The % of the difference λ is 25.5564% between the measured values and the Gmapping values. This indicates that without tuning the Hector SLAM, it still produced better results than the Gmapping.

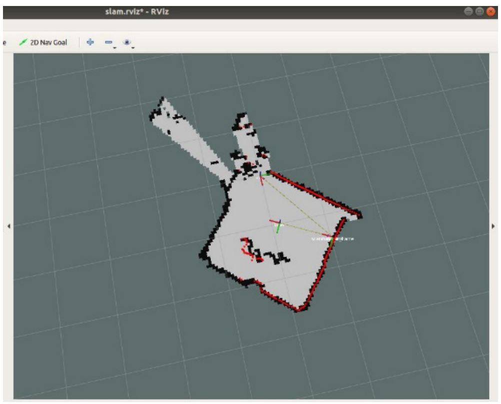

FIGURE 4(A). Actual map based on Hector SLAM without ANBS

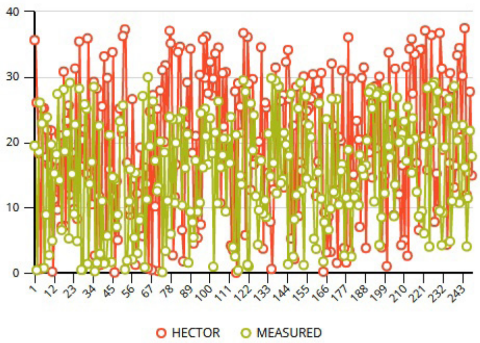

FIGURE 4(B). Results Hector SLAM without ANBS (Distance VS sampling number)

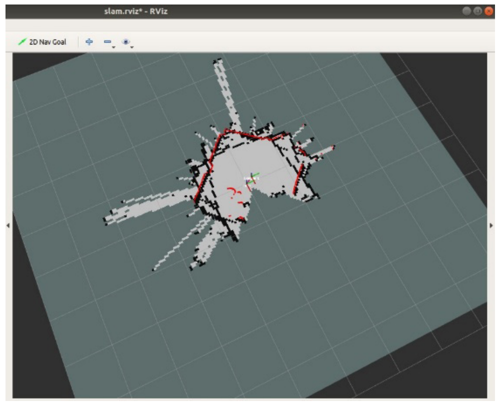

FIGURE 5(A). Actual map based on Gmapping SLAM without ANBS

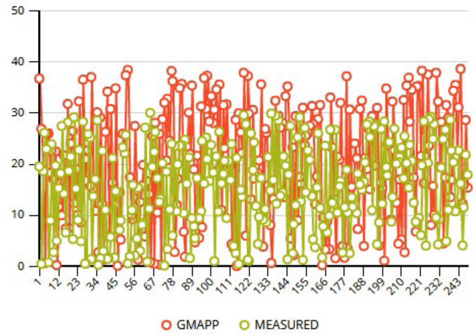

FIGURE 5(B). Results Gmapping SLAM without ANBS (Distance VS sampling number)

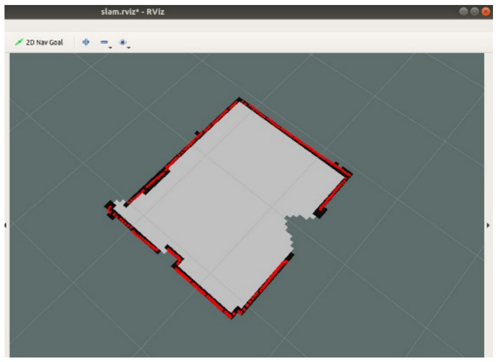

FIGURE 6(A). Actual map based on Hector SLAM with ANBS

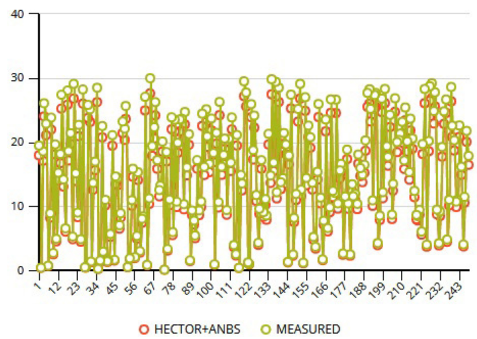

FIGURE 6(B). Results of Hector SLAM with ANBS (Distance VS sampling number)

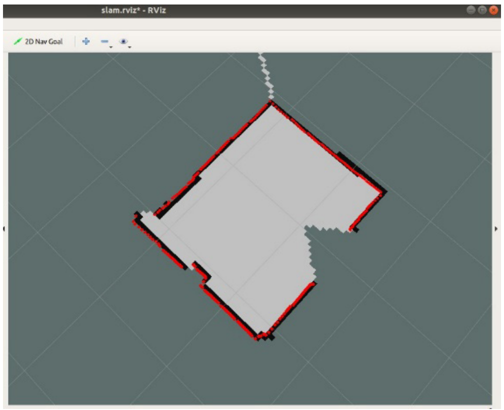

FIGURE 7(A). Actual map based on Gmapping SLAM with ANBS

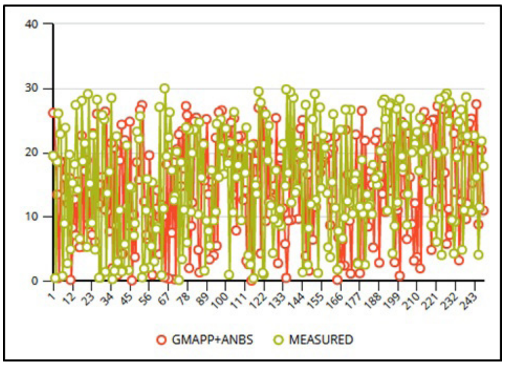

FIGURE 7(B). Results of Gmapping SLAM with ANBS (Distance VS sampling number)

For the Hector SLAM with the ANBS algorithm, the result is represented by Figure 5(b). The standard deviation σ is 8.00316 with 8.08407% of the difference (λ).

Finally in Figure 7(b), after 498 samples taken, the final results produce the standard deviation, σ value is 8.11575 and the % of the difference between the measured values and the output, λ is 8.19237% which is higher than the results in Figure 5(b).

Comparing with both Hector+ANBS with the Gmapp+ANBS method, the percentages of difference for both methods, λ is 1.3307% which can be considered as nearly the same.

CONCLUSION

The ANBS algorithm is proposed to address the complexity of the SLAM algorithm for a mobile robot to produce the most accurate results by addressing the drawbacks in different methods of SLAM. The ANBS algorithm is based on the convolution of the neural network that formulated to eliminates the time consuming for the calibration process. The original output from the Hector SLAM and Gmapping SLAM are then fed into the ANBS algorithm to produce the result. By implementing the convolution matrices method as the main controller for the ANBS algorithm, the output result improves by 14.67% for Hector SLAM and 17.36% for the Gmapping SLAM. As a result, the disadvantages of each SLAM method can be minimized thus can produce a precise map.

DECLARATION OF COMPETING INTEREST

None.

ACKNOWLEDGEMENTS

This work was supported by the fundamental research grant scheme (FRGS). FRGS Grant No. [FRGS/1/2017/TK03/UKM/02/4], National University of Malaysia (UKM).

REFERENCES

Abdelrasoul, Y., Saman, A.B.S.H. and Sebastian, P., 2016, September. A quantitative study of tuning ROS gmapping parameters and their effect on performing indoor 2D SLAM. In *2016 2nd IEEE international symposium on robotics and manufacturing automation (ROMA)*, IEEE.

Achmad, H., Daud, M.R., Razali, S., Pebrianti D. 2016. A ROS-based human-robot interaction for indoor exploration and mapping. *International Journal of Advanced and Applied Sciences* 3 (4): 88-92.

Aravindan, A., Zaheer, S. and Gulrez, T., 2016, September. An integrated approach for path planning and control for autonomous mobile robots. *2016 International Conference on Next Generation Intelligent Systems (ICNGIS),* IEEE, 1-6.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I. and Leonard, J.J., 2016. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics* 32(6):1309-1332.

Claessens, Rick, Yannick Muller, & Benjamin Schnieders. 2013. Graph-based Simultaneous Localisation and Mapping on the Turtlebot platform. 16.

Dewi, T., Risma, P., Oktarina, Y. and Roseno, M.T., 2017. Neural network controller design for a mobile robot navigation; A case study. In *2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, IEEE, 1-5.

Dissanayake, G., Huang, S., Wang, Z. and Ranasinghe, R., 2011. A review of recent developments in simultaneous localization and mapping. *2011 6th International Conference on Industrial and Information Systems*, IEEE, 477-482.

Eliwa, M., Adham, A., Sami, I. and Eldeeb, M., 2017. A critical comparison between Fast and Hector SLAM algorithms. *REST Journal on Emerging trends in Modelling and Manufacturing* 3(2): 44-49.

Esenkanova, J., İlhan, H.O. and Yavuz, S., 2013. Pre-Mapping system with single laser sensor based on Gmapping algorithm. *IJOEE: Int. Journal of Electrical Energy* 1(2): 97-101.

Dissanayake, G., Huang, S., Wang, Z. and Ranasinghe, R., 2011. A review of recent developments in simultaneous localization and mapping. In *2011 6th International Conference on Industrial and Information Systems*, IEEE 477-482.

Kamarudin, K., Mamduh, S.M., Shakaff, A.Y.M. and Zakaria, A., 2014. Performance analysis of the microsoft kinect sensor for 2D simultaneous localization and mapping (SLAM) techniques. *Sensors* 14(12): 23365-23387.

Kamburugamuve, S., He, H., Fox, G. and Crandall, D., 2016, March. Cloud-based parallel implementation of slam for mobile robots. *Proceedings of the International Conference on Internet of things and Cloud Computing*, 1-7.

Kohlbrecher, S., Meyer, J., Graber, T., Petersen, K., Klingauf, U., & Von Stryk, O. 2014. Hector open source modules for autonomous mapping and navigation with rescue robots. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8371 LNAI: 624-631.

Kohlbrecher, S., Petersen, K., Steinbauer, G., Maurer, J., Lepej, P., Uran, S., Ventura, R., Dornhege, C., Hertle, A., Sheh, R. and Pellenz, J. 2012. Community-driven development of standard software modules for search and rescue robots. *2012 IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2012* 2-3.

Mur-Artal, R., Montiel, J.M.M. and Tardos, J.D., 2015. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics* 31(5): 1147-1163.

Ran, L., Zhang, Y., Zhang, Q. and Yang, T., 2017. Convolutional neural network-based robot navigation using uncalibrated spherical images. *Sensors* 17(6): 1341.

Rodin, R. and Štajduhar, I., 2017. The challenge of measuring distance to obstacles for the purpose of generating a 2-D indoor map using an autonomous robot equipped with an ultrasonic sensor. *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, IEEE, 1021-1026.

Tian, G., Liu, L., Ri, J., Liu, Y. and Sun, Y., 2019. ObjectFusion: An object detection and segmentation framework with RGB-D SLAM and convolutional neural networks. *Neurocomputing* 345: 3-14.