Journal of Multidisciplinary Engineering Science and Technology (JMEST)
ISSN: 2458-9403
Vol. 6 Issue 9, September - 2019

Verification of SLAM Methods Implemented in ROS

[1] **František Duchoň, Jakub Hažík, Jozef Rodina, Michal Tölgyessy, Martin Dekan, Adam Sojka**
Department of Robotics, Faculty of Electrical Engineering and Information Technology
Slovak University of Technology, Bratislava, Slovakia
Bratislava, Slovakia
[1]frantisek.duchon@stuba.sk

***Abstract*—Simultaneous Localization and Mapping (SLAM) is well-known problematics and there are plenty of suitable or less suitable approaches for various sensors and robots. This article is focused on comparing some approaches, which are available as the packages in Robot Operating System (ROS). In the beginning of this article, we introduce the SLAM problematics in general. In the next section, available SLAMs in ROS are compared: Gmapping, Hector SLAM, Karto SLAM, Lago SLAM, Core SLAM, and Critical Rays Scan Match (CRSM) SLAM. In the main body, three selected SLAM techniques (Gmapping, Hector SLAM and Karto SLAM) are compared. The results showed that Gmapping had the best results. Therefore, more complex solution was developed by the probabilistic model of used laser rangefinder (Hokuyo URG-04) implemented into Gmapping package. Finally, the results of Gmapping with and without this model are compared. Results showed that with well-identified measurement model of laser rangefinder, the quality of mapping was also improved.**

Keywords—SLAM, ROS, Gmapping, Hector SLAM, Karto SLAM, Hokuyo URG-04

I. INTRODUCTION

Algorithms solving the problem of robotic orientation in an environment are collectively called SLAM - Simultaneous Localization and Mapping. In simple terms, it is possible to find a functional relationship between the robot environment mapping and localization tools. These two processes, i.e. mapping and localization have to run in parallel because the robot creates an environment map based on the knowledge of its current location.

Historically, several approaches have been developed to address SLAM problematics. The main ones are Particle Filter SLAM, Extended Kalman Filter (EKF) SLAM and Graph SLAM. They all have one common attribute; the given task is solved by a statistical approach.

The Kalman filter (KF) [1] is one of the most popular implementations of Bayes' theorem. The KF has two distinct phases: predict and update. In the prediction phase, it estimates the status based on the previous state and control. In the update phase, the estimated state is combined with data from the sensors. Therefore, the resulting estimation is as close as possible to the measurement. The measured data are evaluated by the least square's method, meaning that the resulting estimation is determined from the multiple values of the measured parameter based on the smallest measurement difference and the resulting value. Thereby the measurement noise is very well eliminated. The Kalman filter has one problem indeed - solving the nonlinear tasks is quite problematic. However, this problem is solved by EKF using the Jacobi matrix for the system linearization.

The Particle filter Method [2], also known as the sequencing Monte Carlo or FastSLAM, uses a representation of probable positions of the robot, each representing a hypothesis. Probable estimation of robot position is determined based on the motion and environmental sensing. Algorithm initially assumes that the robot is likely to be located anywhere. Once the robot scans an environment, probable positions are recalculated by Bayes theorem. Based on this calculation, positions with the smallest match are eliminated, and afterwards new probable positions, closer to the most likely ones, are generated. Consequently, the positions are converging, and the resulting estimation of position is evaluated.

Graph SLAM [3] is the most advanced and innovative SLAM approach that addresses weaknesses in the Particle filter (FastSLAM) and EKF SLAM. The solution involves creating a diagram whose nodes represent robot positions or reference points. The edges between the nodes represent encoded measurement information. It represents a restriction between the nodes. After creating this diagram, it is necessary to find the node configuration that will have the smallest error with respect to the measured data. This approach as particular allows the supplementary map modification after creating the individual parts. However, this can cause higher computational demands. It is also possible to use this type of algorithms for SLAM in three-dimensional space. Diagram-based SLAM algorithms are typically more effective than other approaches during the long-term map maintenance and as well as during the large-scale surroundings mapping.

II. SLAM MODULES IN ROS

Gmapping [4] [15] is the most widely used SLAM package available in ROS. Implementing SLAM is based on Rao-Blackwellized particle filters that represents Particle filter with optimization adjustments. The input data are odometry and data from the laser rangefinder. The output of the described node is a map

in the raster grid displaying the scanned obstacles and the free space. Furthermore, it displays also identified position and robot's orientation in the given map [7].

The advantage of this package is sufficient documentation on developer's websites. The package is easy to configure and use. With enough accurate odometry, it may be an advantage to use it by an algorithm. This fact also helps the algorithm to work with the lower category of lidar. Odometry is for the operation of this algorithm inevitable. Therefore, the use of this package is limited only to cases where the robot has data from odometry.

Hector SLAM [5] [16] is also one of the well-known and frequently used packages in ROS. It is based on the comparison of the scanned data, called scan matching with already created map or other scanned data. This data comparison is solved by the Gauss-Newton Algorithm. It attempts to find optimal alignment of laser scan endpoints with created map. During the comparison of scans, the algorithm takes into account solely significant points. The great advantage of this implementation is that robot odometry is not necessary. Accordingly, this enables easy application of package into flying robots or field robots, as well. On the other hand, the disadvantage of this implementation is that in the case of a combination of the slow rangefinder, the absence of odometry and the rapid movement, a high inaccuracy occurs during the data comparison. This inaccuracy is also reflected in the resulting map.

Karto SLAM [6] [17] is an algorithm based on the Graph-SLAM method developed by Karto Robotics with ROS extension. They achieved successful algorithm optimization to such an extent that the computational complexity is comparable to other SLAM algorithms. In this case, each node represents the position of the robot with respect to its trajectory and the set of sensory data. Nodal links represent a movement between positions immediately following each other. For each new node, the map is calculated by finding a spatial optimal node configuration. In some cases, there can be problem during the implementation caused by the insufficient package documentation, the small number of adjustable parameters and the necessity of odometry application.

Less known modules can be found, as well. They are not much popular because of their out of date algorithms, demanding deployment, or any other reason that contributes to their low popularity.

Lago SLAM [8] is based on Graph SLAM. Its distinction from other diagram-based SLAMs is that the optimization process for finding the right graph configuration does not require any initial estimation. This fact, however, causes an increase in the computational complexity of the algorithm. The module unfortunately has no information posted on the official ROS web site. Consequently, it is relatively difficult to find it.

Core SLAM [9] uses a simple Particle filter to match the scanned data. Another problem is that the module has almost no documentation. Therefore, its deployment is extremely problematic.

CRSM SLAM [10] is built on so-called scan matching process that is performed through a randomly updated climbing algorithm. Map updating is performed with additional dynamic intensity information. This information includes the relevance of the searched area. However, the module is used not often and therefore it loses support in the newer ROS versions [11].

III. Testing the Main SLAM Packages in ROS

A space with rectangular wall structure was created for testing the implemented methods in ROS (Fig. 1). All sensor data from the laser rangefinder and odometry were uploaded and afterwards executed for each SLAM algorithm. Accordingly, maps were created under the same conditions [14]. Hector SLAM requires a laser rangefinder with an extremely short measurement cycle. However, the Hokuyo URG-04LX-UG01 rangefinder does not meet this condition. In order to suppress this restriction, the speed of the robot's movement was limited.

Fig. 1. Space for testing SLAM modules; in the forefront is the KUKA YouBot robot that was tested.

The map created by the *gmapping* package best describes the real environment (Fig. 2). The rendered environment walls have the least noise and the map surface is not in any way deformed. This kind of a map is fully sufficient for the accurate localization and navigation.

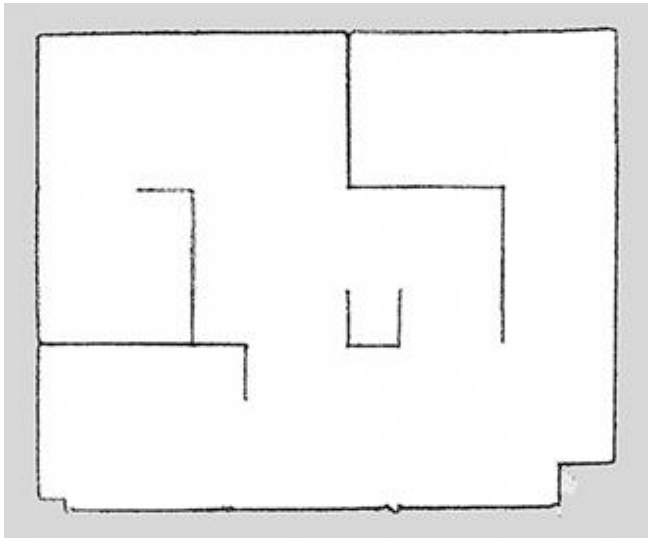

Fig. 2. Environment map created by the *gmapping* package.

The map created by the *hector_slam* package contains an area that could not be mapped (Fig. 3). The reason is the lack of significant points on a long straight wall. The walls of the environment obviously contain noise and the areas are slightly deformed. Measured data has a relatively low repeatability and the algorithm has a clear problem with is. Therefore, Hector SLAM is only suitable for use with a higher class of laser rangefinder with high repeatability and short measurement cycle.

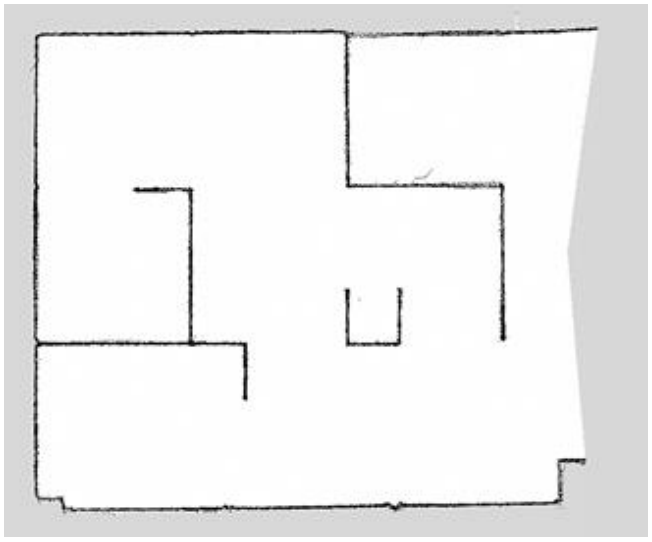

Fig. 3. Created environment map using the *hector_slam* package.

Abnormal deformation and noise of walls is visible in the case of a map created by the *karto_slam* package (Fig. 4). Thanks to the diagram-based algorithm, particularly this package is able to solve inaccuracies in mapping a closed space, called loop closing.

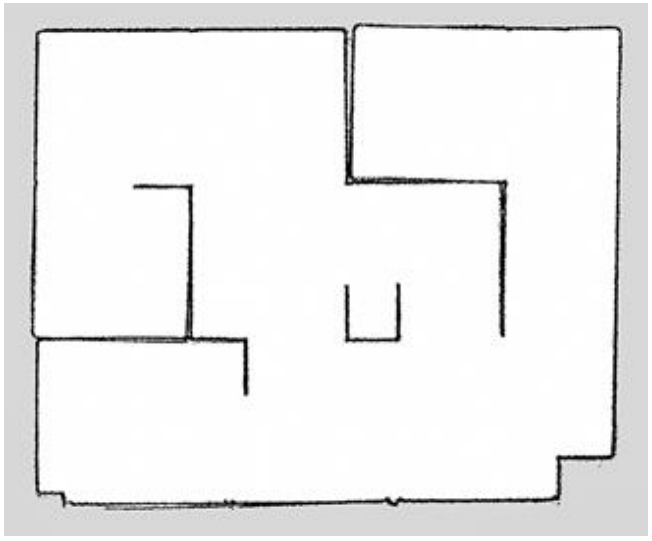

Fig. 4. Environment map created using the *karto_slam* package.

The *gmapping* package achieved the best match with the real environment. Therefore, it was selected to implement KUKA YouBot mobile handler control. *Gmapping* uses the Particle filter method. This method requires stochastic sensor and odometry models. The preset values of these models do not accurately describe the sensor's behavior on the robot. Therefore, it was necessary to create a model of used sensors to better the results of the mapping process.

IV. Scanning Laser Rangefinder Hokuyo URG-04 Stochastic Model

During more complex development of solutions, especially in the area of mobile robotics, where the robot acquires data from multiple sensors, it is advisable to have knowledge of the data relevance from individual sensors [12]. Theoretically, if such knowledge exists, weights can be assigned to sensors, what makes it possible to obtain much more accurate results. Creating a stochastic sensor model is one of the ways how to determine the relevance of the sensor data. The model can only be created based on predefined parameters. Only one parameter – measurement repeatability - will be listed in the stochastic model [13].

50 measurements were made to evaluate the measurement repeatability of laser rangefinder. Each measurement was 500 times repeated in a short sequence consecutively and under the same conditions. Measurements were made on various types of materials: varnished wood, cardboard, office paper, black metal plate, white fabric. These objects were placed in front of the laser rangefinder up to 1 m, 2 m, 3 m, 3.5 m, 4 m distances at a 90° and 45 ° angle. Each of these objects has specific features. The bright, glossy and smooth surfaces reflect the light very well. The dark, matte, and rough surfaces, on the other hand, do not reflect it, but rather absorb or disperse it. This stands for as well as for the conventional light and as for the laser rangefinder. From the measured data, the standard deviation was presented. It represents the sought repeatability of the measurement. The Gaussian function was used to graphically represent the probability.

The highest repeatability rate was achieved in the repeatability measurement of the vertical measurement at 3 m distance. The wooden surface shows the best results, the white paper with fabric shows identical results and the black metal plate shows the worst results (Fig. 5).

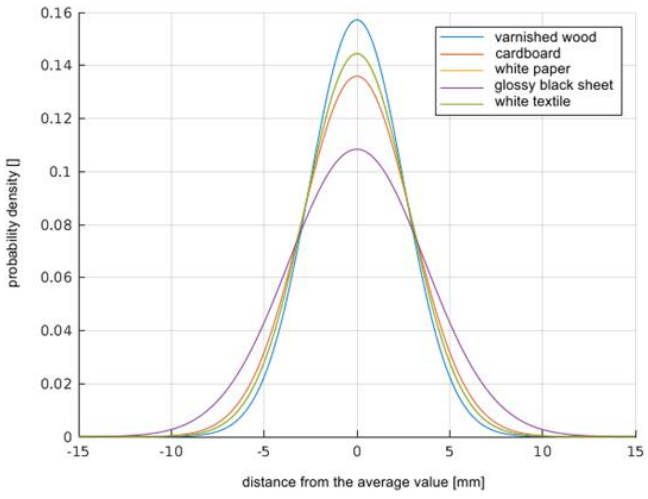

Fig. 5. Distribution of probability for laser rangefinder at 3 m distance.

The measurement results for determining the standard deviation for individual surfaces are shown in Tab. 1. The resulting sigma is expressed as the mean standard deviation of the measurements at the given distances.

TABLE I. MEASURED STANDARD DEVIATION FOR INDIVIDUAL MATERIALS AND TYPES OF MEASUREMENTS.

Surface Type	σ [mm] - 90° Measurement	σ [mm] - 45° Measurement
Varnished wood	4.3872	5.5417
White textile	4.5669	5.7235
White paper	5.3563	7.4447
Cardboard	6.0059	8.5790

Measurements, depending on the object's distance, can be seen in the figures (Fig. 6, Fig. 7). The approximate 3rd order functions were defined for the measured points. These functions represent the dependence of the standard deviation from the measured distance.

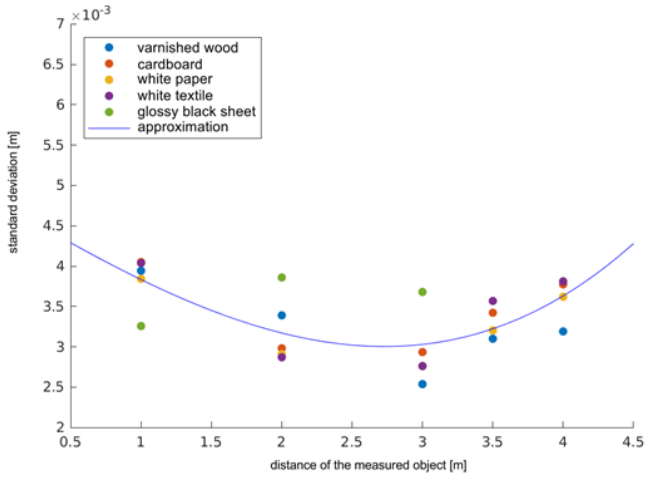

Fig. 6. Figure 6. Standard deviation approximation of the measurement repeatability of the laser rangefinder at the 90 ° angle with the 3rd order multi-nominal: $\sigma(x) = 0.000037032x^3 + 0.000037966x^2 - 0.001034868x + 0.00479271$

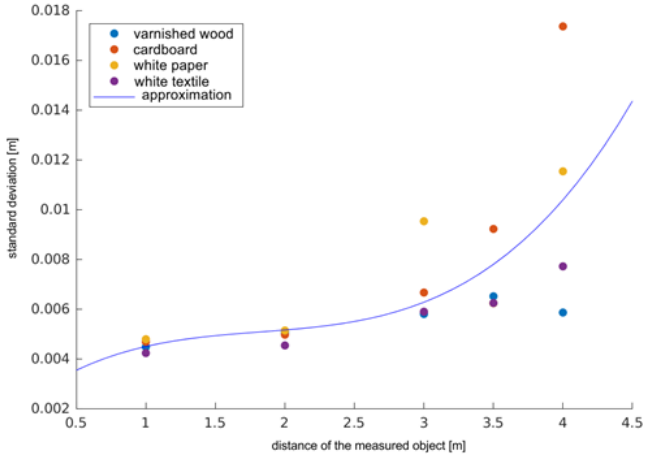

Fig. 7. Figure 7. Standard deviation approximation of the measurement repeatability of the laser rangefinder at the 45 ° angle with the 3rd order multi-nominal: $\sigma(x) = 0.000424014x^3 - 0.002317497x^2 + 0.004646225x + 0.001744066$

The repeatability of the measurement at 45 ° angle is significantly worse. One reason is that the transmitted beam is not reflected directly to the laser rangefinder. This problem arises primarily for lighter objects that reflect the light better and behave in a similar way as a mirror at this time. Another reason for the repeatability deterioration is that the uncertainty of the beam routing enters the measurement. Therefore, the beam slightly deviates to the sides, whether to the right or to the left. Accordingly, the illuminated point on the sloping surface appears either closer or further. Further 3rd order approximation function of the 3rd order was expressed in order for the measurement repeatability changes with respect to the distance and slope of the measured object to be expressed also analytically. This function represents the average of the two previous functions:

$$\sigma(x) = 0.0002500x^3 - 0.0012220x^2 + 0.0018264x + 0.0032760 \quad (1)$$

For the *gmapping* application in ROS, one specific value is required to define. This value will represent the stochastic model and it will be represented the input argument. This value was determined for a distance of 2.5 m, because in terms of multiplicity, this value was the highest in the given environment. The reference deviation for this distance is 0.0041 m.

V. STOCHASTIC MODEL VERIFICATION

The SLAM *gmapping* module with and without the application of the stochastic model was used to verify and evaluate the created sensor model. The *gmapping* module enters not only laser rangefinder data, but also odometry. This fact does not allow the stochastic model to be fully expressed, but only partially. Experiments have also been adapted to this.

In the first experiment, the mapping quality was evaluated assuming that the robot was static. In this way, the impact of odometry on the resulting map was limited. The map shows a wall in front of which is a beverage can of circular shape. The arched shape of the can is somewhat more evident on the map with the used model (Fig. 8 on the left) than in the case of a map without using the model (Fig. 8 on the right).

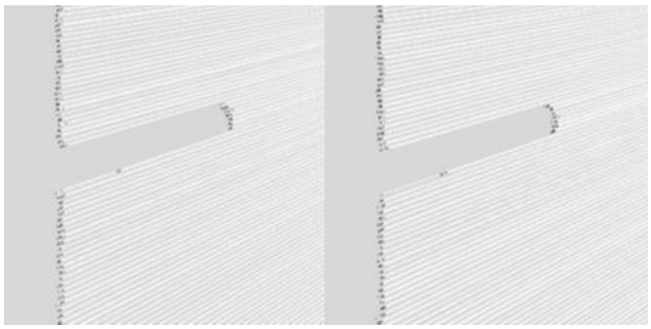

Fig. 8. Figure 8. Static robot mapping: left - without the use of created model, on the right - with the use of created model.

Further experiments have already been performed with mobile robot application, with obstacles taken from different positions. It is clear from the Fig. 9 that the use of the model eliminates noise, which ultimately results in a smoother and narrower display of obstacles.

Fig. 9. Figure 9. Mapping a broken wall with a mobile robot: on the left - without the use of model, on the right – with the use of model.

Another experiment (Fig. 10) consisted of mapping the wall with a rectangular rebound. The rectangularity of the rebound from the wall is better captured in the map with the application of model. Noise has similar characteristics in the case of application of the stochastic model compared to the application without the stochastic model.

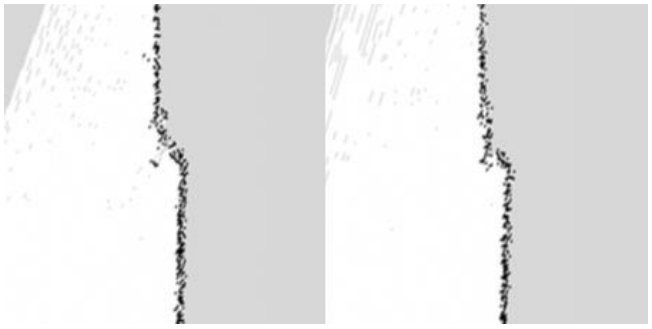

Fig. 10. Figure 10. Mapping a broken wall with rectangular rebound: on the left - without the use of model, on the right – with the use of model.

One of the *gmapping* output variable nodes is entropy. This value is an indicator of the uncertainty of the robot position estimate in the generating map. This means that the lower the value, the robot is surer in its position. This value changes and converges to a lower value each time the map is reset. After multiple mapping tests, it has been shown that during the mapping with the use of stochastic sensor model, the entropy rate is always lower. In this experiment the robot was static in order to suppress the effects of odometry. Such a measurement was performed ten times with and ten times without using a stochastic model. Without the model, the average entropy was 3.40119738166, with the model 3.40119717885. The difference is therefore equal to only 0.00000020300. Although there is only minimal improvement in the result, the main reason is the static state of the robot.

Another way to verify a stochastic model is to measure the measurement repeatability under the different conditions where the stochastic model was created, meaning different measured distance, light conditions, scanned material, and inclination of the scanned material. The laser rangefinder placed on the robot was placed against the artificial wall at random distances from 0 ° to 30 ° angle. Measurements were performed at a different time comparing to the measurements for the model creation. Consequently, the light conditions have changed as well. The robustness of the stochastic model has been verified using this technique.

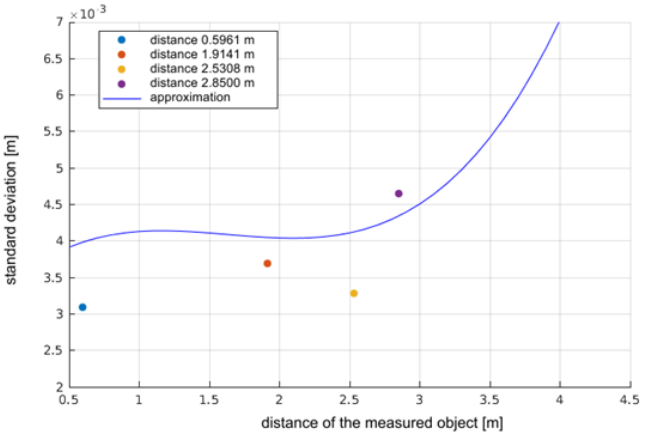

Fig. 11. Comparison of approximate repeatability with actual repeatability of the measurement.

It is clear from the figure that the measured points lie below or in the vicinity of the approximate function. The stochastic model and the 0.0041 mm selected repeatability value are robust enough to achieve better mapping over the entire range of measured distances in SLAM algorithms.

VI. Conclusion

Based on the achieved results it can be said that the *gmapping* package has achieved the best results. Therefore, it has been applied for further control needs and improved by creating a stochastic module for the Hokuyo URG-04LX-UG01 Scanning Laser Rangefinder. Stochastic model was created based on one parameter, specifically the measurement repeatability. One specific value representing the stochastic model was used - σ = 0.0041 mm. This value has proven to be both sufficient and robust based on measurements and verification. However, there exist still occasional mistakes in mapping and localization. These errors were originated due to the inaccuracies of odometry. Odometry moreover enters the SLAM algorithms and greatly affects the resulting map or localization. Therefore, our next work will focus on creating a stochastic model for the chassis' behavior, accordingly the KUKA YouBot robot odometry.

Acknowledgment

This work was supported by VEGA 1/0752/17, APVV-16-0006 and VEGA VEGA 1/0754/19.

References

[1] F. Duchoň a A. Babinec. Localization of Mobile Robots. Bratislava, 2015. ISBN: 978-80-227-4461-4.

[2] S. Thrun, W. Burgard, and D. Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press, 2005. ISBN: 0262201623.

[3] J.L. Fernández-Madrigal, J.A. a Blanco. Simultaneous Localization and Mapping for Mobile Robots: Introduction and Methods. IGI Global, 2012. ISBN: 978-1466621046.

[4] W. Burgard G. Grisetti, C. Stachniss. Documentation ROS package gmapping [online]. Avaliable online: http://openslam.org/gmapping.html. Cited 22.11.2017.

[5] Documentation ROS package Hector_SLAM [online]. Available online: http://wiki.ros.org/hector_slam. Cited 25.4.2018.

[6] Documentation ROS package Karto_SLAM [online]. Available online: http://wiki.ros.org/slam_karto. Cited 25.4.2018.

[7] Ď. František, Ü. Róbert. SEF roboter and ROS: From 3D model to real-time robot control. Novus Scientia 2017. ISBN: 978-80-553-3080-8.

[8] Documentation ROS package LagoSLAM [online]. Available online: https://github.com/rrg-polito/rrg-polito-ros-pkg. Cited 25.4.2018.

[9] Documentation ROS package CoreSLAM [online]. Available online: http://wiki.ros.org/coreslam. Cited 25.4.2018.

[10] Documenatation ROS package CRSM SLAM [online]. Available online: http://wiki.ros.org/crsm_slam. Cited 25.4.2018.

[11] J. M. Santos, D. Portugal, and R. P. Rocha. An evaluation of 2d slam techniques available in robot operating system. In 2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pp. 1-6, Oct 2013. ISSN: 2374-3247.

[12] P. Kocmanova, L. Zalud & A. Chromy (2013, August). 3D proximity laser scanner calibration. 2013 18th International Conference on Methods & Models in Automation & Robotics (MMAR).

[13] G. C. Anousaki, & K. J. Kyriakopoulos (2007). Simultaneous localization and map building of skid-steered robots. IEEE Robotics & Automation Magazine, 14(1), pp. 79-89.

[14] A. Filatov, K. Krinkin, B. Chen, & D. Molodan (2017, November). 2d slam quality evaluation methods. IEEE 2017 21st Conference of Open Innovations Association (FRUCT), pp. 120-126.

[15] B. M. da Silva, R. S. Xavier, T. P. do Nascimento, & L. M. Gonsalves (2017, November). Experimental evaluation of ROS compatible SLAM algorithms for RGB-D sensors. IEEE 2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR), pp. 1-6.

[16] Y. Abdelrasoul, A. B. S. H. Saman, & P. Sebastian (2016, September). A quantitative study of tuning ROS gmapping parameters and their effect on performing indoor 2D SLAM. IEEE 2016 2nd IEEE International Symposium on Robotics and Manufacturing Automation (ROMA), pp. 1-6.

[17] L. Fang, A. Fisher, S. Kiss, J. Kennedy, C. Nagahawatte, R. Clothier, & J. Palmer, (2016). Comparative evaluation of time-of-flight depth-imaging sensors for mapping and slam applications. Proceedings of the Australian Robotics and Automation Association, Brisbane, Australia, pp. 5-7.