Chapter 30
Evaluation of Visual SLAM Methods in USAR Applications Using ROS/Gazebo Simulation

Ramil Safin, Roman Lavrenov, and Edgar Alonso Martínez-García

Abstract The problem of determining the position of a robot and at the same time building the map of the environment is referred to as SLAM. A SLAM system generally outputs the estimated trajectory (a sequence of poses) and the map. In practice, it is hard to obtain ground-truth for the map; hence, only trajectory ground-truth is considered. There are various works that provide datasets to evaluate SLAM algorithms in different scenarios including sensor configurations, robots, and environments. Dataset collection in a real-world environment is a complicated task, which requires an elaborate sensor and robot configuration. Different SLAM systems demand various sensors resulting in the problem of finding an appropriate dataset for their evaluation. Thus, in this paper, a solution that is based on ROS/Gazebo simulations is proposed. Two indoor environments with flat and uneven terrain to evaluate laser range and visual SLAM systems are created. Changing the sensor configuration and the environment does not require an elaborate setup. The results of the evaluation for two popular SLAM methods—ORB-SLAM2 and RTAB-Map—are presented.

R. Safin · R. Lavrenov (✉)
Laboratory of Intelligent Robotic Systems (LIRS), Intelligent Robotics Department,
Higher Institute for Information Technology and Intelligent Systems,
Kazan Federal University, Kazan, Russian Federation
e-mail: lavrenov@it.kfu.ru
URL: https://kpfu.ru/erobotics

R. Safin
e-mail: safin.ramil@it.kfu.ru
URL: https://kpfu.ru/erobotics

E. A. Martínez-García
Laboratorio de Robotica, Institute of Engineering and Technology,
Universidad Autonoma de Ciudad Juarez, Ciudad Juarez 32310, Mexico
e-mail: edmartin@uacj.mx

A. Ronzhin and V. Shishlakov (eds.), *Proceedings of 15th International Conference on Electromechanics and Robotics "Zavalishin's Readings"*, Smart Innovation, Systems and Technologies 187, https://doi.org/10.1007/978-981-15-5580-0_30

30.1 Introduction

Robotic systems are employed in a variety of applications, such as urban search and rescue (USAR), space exploration, military, medicine, and education [14]. In all of the applications, robots are required to perform particular tasks interacting with an environment. In some cases, environments are complex and non-deterministic which makes the intended tasks even more complex for a robot to accomplish [5].

There are applications where a robot is required to determine its position in the environment for navigation and space exploration. If the map of the environment is provided with certain accuracy, one can localize itself on the map and accomplish the intended task. However, in most cases, the map of the environment is partially available or not available which leads to the necessity of the map construction. Hence, the task becomes even more complicated, to simultaneously perform localization and build a map of the environment. This problem is referred to as simultaneous localization and mapping (SLAM).

One of the widely used sensors in SLAM is cameras. They provide extensive information about the environment, and they are not vulnerable to slippage as wheel encoders. SLAM using cameras is referred to as Visual SLAM (VSLAM). VSLAM finds its applications in autonomous driving [2], 3D-scene reconstruction, augmented reality (AR), etc.

One of the problems of SLAM methods evaluation is that there are various datasets with different sensors configurations. For example, stereo SLAM methods require a pair of monocular cameras [18]. Generally, most of the datasets provide laser range and stereo image data. However, in the case of multi-camera (two or more cameras) SLAM systems, those datasets cannot be employed due to the lack of necessary sensor data. Thus, in this paper, we propose to use ROS/Gazebo simulation for collecting datasets using different sensors configuration for the multisensor SLAM systems development and evaluation. We show the usability of the solution by evaluating Hector SLAM, ORB-SLAM2, and RTAB-Map methods on the collected datasets in the USAR scenario with flat and uneven terrains.

30.2 SLAM Overview

In SLAM, a robot traverses a particular environment constructing its map and estimating the state of the robot. The map is a representation of the landmarks, obstacles, and objects in the surroundings. It helps a robot to navigate through the environment being the reference for localization [15]. Maps can be represented in several ways: metric, topological, or hybrid. The state that is estimated can be defined as a 2D position and orientation or a 6D pose (Cartesian coordinates and orientation).

SLAM approaches can be classified into filtering and smoothing methods. Filtering approaches are based on Bayesian theory; here, the position of landmarks and robot poses are computed over time by incorporating new measurements into the probabilistic model, i.e., estimating the posterior state distribution. There are two major steps in this framework: prediction of the state based on the control commands and update of the state by incorporating new sensor measurements. Considering the probability distribution representation, filter-based methods can be classified into unimodal and multimodal ones. The former is represented by the Kalman filter family methods (e.g., EKF, UKF, SIKF) while the latter—particle filters (e.g., Rao-Blackwellized [16]). The main disadvantages of unimodal filtering methods are the quadratic growth of time complexity with the growth of the map size (number of features) and poor accuracy in case of nonlinear motion. As for the multimodal approaches, the accuracy depends on the number of particles which directly correlates with time complexity.

Smoothing methods take into account a sequence of states—full trajectory or some part of it—in order to estimate the new state using nonlinear least-squares optimization techniques (e.g., Gauss–Newton, Levenberg–Marquardt). Optimization approaches can be divided into bundle adjustment (BA) and graph classes. The key idea behind BA is to jointly optimize a 3D structure (map) and camera poses. This is accomplished by minimizing the re-projection error. Full BA over the whole trajectory is a computationally expensive task; hence, it might be used in case of loop closure as a part of the global optimization process. On the contrary, local BA over a sliding window of poses is able to mitigate the accumulated drift providing real-time performance even on mobile devices. Graph approaches are modeled using a graph representation of SLAM. The graph optimization is solved by minimizing a particular cost function using the same optimization techniques as for the BA.

Generally, the architecture of VSLAM contains two main modules: back-end and front-end (Fig. 30.1). The front-end processes sensor measurements and builds the model of the state. Back-end solves for the state trying to estimate the best fit for the SLAM solution. Here, we have BA and graph optimization.

Visual odometry (VO) is the main building block of VSLAM. It is the process of estimating the relative motion between consequent frames using cameras [17]. It is less affected by the drift issue compared to the wheel odometry.

The pipeline of VO consists of the following blocks: feature detection, motion estimation, data association, and local BA. VO methods can be separated into two

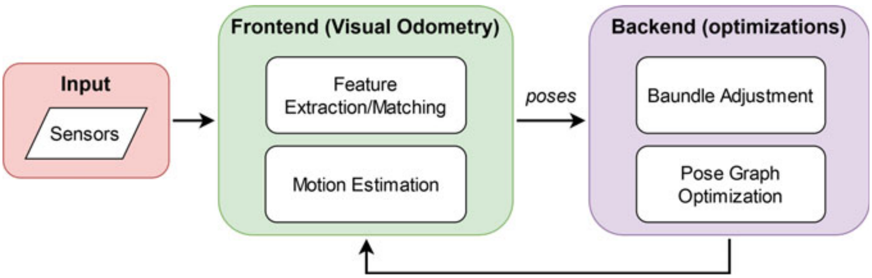

Fig. 30.1 Main components of VSLAM algorithm

groups: direct and indirect. Direct methods enable to build a dense map of the environment directly working with image pixels. On the other hand, indirect methods detect features in the images which increase the speed of computations. In feature-based VSLAM, the so-called feature detectors and descriptors are used to find salient points in the images and describe them into some representation (e.g., SIFT, ORB, SURF, etc.). This mechanism enables one to perform data association (feature matching) on them to recognize the same 3D points from different images. In the case of a direct approach, there is no feature detection routine in the pipeline.

Motion estimation can be divided into two categories: feature-based and optical flow methods. In the first case, the motion can be computed using one of the 3D-3D (Euclidean distance), 3D-2D (re-projection error) or 2D-2D techniques where the corresponding error is minimized to obtain the pose relative to the previous one. In the case of direct VO, generally, the photometric error is minimized. Local BA over the last keyframes can be performed to mitigate the drift errors.

VSLAM pipeline additionally may include global optimization and loop closure routines. Loop closure is the final step in SLAM algorithms. It optimizes the estimated SLAM solution incorporating additional constraints into the system. Here, feature matching techniques, a bag of words, etc., can be used for loop detection where features are matched across a set of keyframes.

30.3 VSLAM Evaluation

A SLAM system generally outputs the estimated camera trajectory (a sequence of poses) and the map. In practice, it is impossible to obtain the ground-truth map; hence, only trajectory ground-truth is obtained. Ideally, the sequences of SLAM output and ground-truth should be time-synchronized and evenly sampled with the same sequence length. The trajectory is a sequence of poses with respect to some reference frame which is obtained during the calibration. The reference frame is not required to be the same for the ground-truth and estimated trajectory.

The global consistency of SLAM can be evaluated by comparing the absolute distances between the estimated and ground-truth trajectories—absolute pose error (APE). The alignment is necessary in case if trajectories are in different reference frames [21]. Given the reference trajectory $GT_{1:n}$ and estimated trajectory $T_{1:n}$, APE at time step i can be computed as follows:

$$APE_i = GT_i^{-1} \cdot S \cdot T_i, \tag{30.1}$$

where S—is a least-squares rigid-body transformation that maps estimated trajectory $T_{1:n}$ onto the ground-truth $GT_{1:n}$. For the APE, we can compute translation RMSE using the following formula:

$$RMSE(APE_{1:n}) = \frac{1}{n}\left(\sum_{i=1}^{n} ||transl(APE_i)||^2\right)^{\frac{1}{2}}. \tag{30.2}$$

The relative pose error (RPE) measures the accuracy of the trajectory over a fixed period, which is drift. RPE considers both relative and translation errors while APE takes into account only translation error. Nevertheless, as the translation part is correlated to the rotation error, APE and RPE are correlated.

There are various works that provide with datasets to evaluate SLAM algorithms in different scenarios including sensor configurations and environments. One of the first datasets is provided in [4] with outdoor and indoor information of the ground-truth, GPS, IMU, LiDAR, and image data.

KITTI datasets [6] are obtained from a vehicle in an urban environment. The vehicle is equipped with a stereo camera and LiDAR. Translation and rotation error can be measured comparing the real data with ground-truth.

Amey K. provides a comparison of two SLAM methods—RGBD-SLAM and RTAB-Map [10]. Here, TUM RGBD dataset [20] is used for SLAM methods evaluation. The dataset provides with data from Microsoft Kinect RGBD sensor and time-synchronized ground-truth camera poses from a motion capture system. The evaluation metrics are absolute trajectory error (ATE or APE) and relative pose error (RPE). The analysis involves the RMSE computation for the two metrics above and processing time comparison.

In the other paper by Maksim et al. [9], SLAM systems in an indoor environment are compared for a mobile robot. Real-world experiments on their dataset were collected using LiDAR. They evaluate and compare LiDAR, monocular, and stereo algorithms: GMapping, Hector SLAM, ORB-SLAM, SVO, S-PTAM, etc. The robot was teleoperated through the environment to follow a closed-loop trajectory along a rectangular area. Mostly, the environment is consisted of walls. The rotation was accomplished in a radius of around 1m. Data from sensors was collected into ROS bag files. In the evaluation, the authors used ATE taking Hector SLAM algorithm as a ground-truth. The trajectory of the Hector SLAM was compared to the marked line on the floor.

In the work done by Arthur et al. [8], they evaluate different SLAM algorithm on TUM RGBD datasets. There are several metrics to evaluate SLAM systems: localization accuracy (RMSE), processing time (CPU usage), memory consumption, robustness (produce the same results for multiple runs), specific for visual SLAM: camera frame processing time (should be near FPS), map specific for occupancy grid maps: quality of the map (e.g., GMapping).

In the paper by Riccardo et al. [7], authors conduct a comparison of ROS compatible stereo SLAM methods on NVidia Jetson TX2 platform. The following metrics are evaluated: CPU and memory usage, trajectory estimation accuracy, and loop closure capability. The ground-truth is obtained utilizing LiDAR (reference). ZED camera is used as the main sensor. The rover is controlled remotely over a circular path while acquiring a stereo video stream and LiDAR. Ground-truth is provided by

LiDAR SLAM, Hector SLAM algorithm. Images are captured at 15 Hz. Both sensors are stored for further usage. ATE and RTE errors are computed for the computed trajectories.

In SlamBench2, the authors propose a framework for SLAM evaluation [1]. Currently, several algorithms are supported for evaluation (eight algorithms): ORB-SLAM2, OKVIS, etc. Its purpose is to unify the interface of benchmarking SLAM algorithms and provide reproducibility of results for different datasets. It allows to measure different metrics: computation speed (time per frame, FPS), ATE and RPE with online and offline alignment, power consumption (PAPI), memory usage. In their up-to-date paper SLAMBench3, they move toward scene understanding [3].

In the paper by David S. et al., TUM benchmark for evaluation of VO systems is presented [20]. It provides with camera images taken at 20 Hz synchronized with the IMU, HDR, and photometric calibration. For trajectory evaluation, they provide with a motion capture system at the start and the end of the sequences. It is focused RGBD odometry evaluation. There are lots of datasets available for VO evaluation: (a) KIITI odometry (stereo @10 Hz, IMU, software sync, GT: GPS); (b) Malaga Urban (stereo @20 Hz, IMU, software sync, GT: GPS); (c) EuRoC MAV (stereo @20, IMU, hardware sync, GT: motion capture); (d) PennCOSYVIO (stereo @20hz, fisheye @30 Hz, IMU, GT: Fiducial markers [22]). The main idea of TUM is to compute the drift using the start and the end positions of the trajectory.

An interesting solution is to move toward simulation environments where one can perform repetitive controlled SLAM methods evaluation. However, in the review literature, there is no solution which supports for multi-camera configurations. For instance, InteriorNet [13] provides with datasets for CV algorithms evaluation. There are multiple types of camera models and scenes which are supported. However, there is no possibility for the moment of writing this paper to incorporate three or more cameras into the simulation.

In a lot of works, Hector SLAM [11] is utilized to obtain ground-truth trajectories to evaluate against. However, generally in those works, only a flat terrain is considered, whereas in USAR applications, uneven terrains are frequently met. On the other hand, Hector SLAM provides with tools for six DoF that pose estimation mostly based on IMU data which is arguably not the most reliable source for ground-truth pose estimation. In more complicated environments, motion capture systems are employed, e.g., Vicon. However, those systems are expensive and require elaborate setup.

30.4 Proposed Solution

30.4.1 Environment

Simulators for robotic systems have been extensively used in research for repetitive evaluation of algorithms in different scenarios, including flat and uneven terrains,

Fig. 30.2 View of the Gazebo indoor environment (flat terrain)

various weather conditions, etc. Gazebo is one of the most popular simulators. It is widely used in conjunction with robot operating system (ROS) framework—a set of libraries and tools for the development of robotic systems. Gazebo provides with means for modeling various robotic systems equipped with different sensors, such as monocular and stereo cameras, IMU, and LiDAR [12].

In our solution, we create two environments for SLAM algorithms evaluation, i.e., an indoor environment of a building with flat and uneven terrain (Fig. 30.2). The building construction is 10 by 10 meters in size. The model of the environment was made in Blender 2.8.

Both environments have similar objects on the scene: windows, shelves, damaged wall, and truss. The flat environment's floor is a plane without any height changes throughout the scene. On the contrary, uneven environment's floor represents a surface with changes in height of up to 1.3 m relative to the ground. Illumination is even all over the environment without shadows.

30.4.2 Robot Model and Sensors

In this work, we use Husky mobile robot provided by the *husky_description* ROS package and modify it to include necessary sensors. The model of the robot is represented in the unified robot description format (URDF). The model consists of links connected by joints with their visual and collision meshes. The simplified transformation tree and robot model are presented in Fig. 30.3. The robot is equipped with the following sensors: (a) stereo view camera (resolution: 640 × 480, focal length: 320 pixels, RGB); (b) laser range finder (field of view 270°); (c) inertial measurement unit (IMU).

Gazebo sensors API enables to incorporate distortions into the sensory input. For example, attached cameras can have pixel intensity noise (e.g., Gaussian) as well as radial and tangential lens distortions. For stereo cameras, we employed a mutli-camera sensor API to provide synchronized global shutter images from both cameras [19].

30.4.3 Dataset Collection

We collected four datasets for each type of terrain for SLAM algorithms evaluation using ROS/Gazebo by remotely controlling the robot (in teleoperation mode) in a looped trajectory using *teleop_twist_keyboard* ROS package. The average linear velocity is around 0.15 m/s and angular—0.15 rad/s. Final trajectories do not have sharp turns to avoid drastic changes in the observed environment.

In our datasets collection, we employ a system with the following properties: (a) Intel Core i7-8750H @2.2 GHz CPU; (b) 16 GB LDDR4 RAM; (c) 1TB M.2 PCI-E SSD; (d) Ubuntu 16.04 LTS OS (with ROS Kinetic installed); The ground-truth 6D pose trajectory is obtained using the P3D Gazebo plugin. Poses are provided for the *base_footprint* frame w.r.t the *world* fixed frame (Fig. 30.3). Datasets were recorded into ROS bag files, they contain the following topics:

(a) */clock*—simulation timestamp;
(b) */cmd_vel*—control commands;
(c) */odometry/filtered*—IMU and wheel odometry EKF fusion;
(d) */ground_truth*—6D pose of the robot w.r.t world;
(e) */tf_static*—transformations between robot links;
(f) */scan*—LiDAR scans data;
(g) */camera/left/image_raw* and */camera/left/camera_info*—left stereo camera topics with image data and meta information;

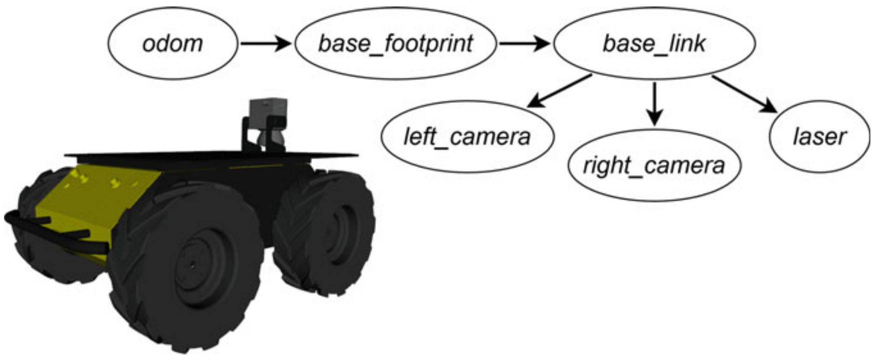

Fig. 30.3 Husky robot URDF model and its transformations tree (left—robot model visualization, right—robot model links hierarchy)

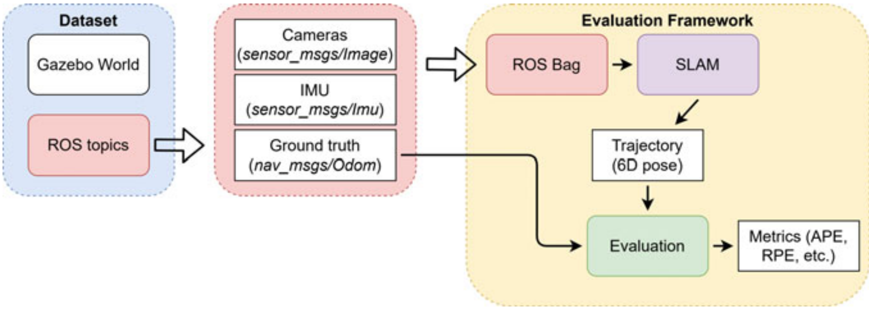

Fig. 30.4 SLAM evaluation components. Datasets contain the simulated environment and sensor data

(h) */camera/right/image_raw* and */camera/right/camera_info*—right stereo camera topics with image data and meta information;

We propose to record datasets and distribute them with the simulated environment (Fig. 30.4). The dataset contains the Gazebo world with all the meshes and textures included along with recorded ROS topics with sensor data. Datasets are stored in a ROS bag file which can be further issued to different SLAM algorithms for consequent evaluation.

30.4.4 Experiments

In our experiments, we evaluated the Hector SLAM, stereo RTAB-Map, and stereo ORB-SLAM2 algorithms on our recorded datasets of flat and uneven terrain environments. In the case of the flat terrain, Hector SLAM estimated trajectory error

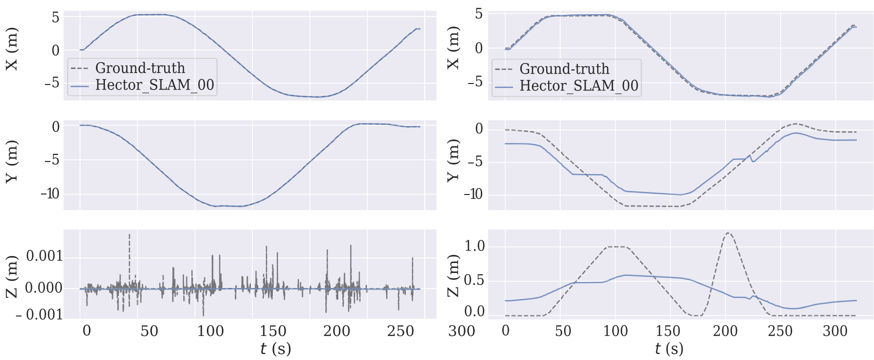

Fig. 30.5 Hector SLAM absolute pose error (APE) over time for flat and uneven terrains (sequence No. 0)

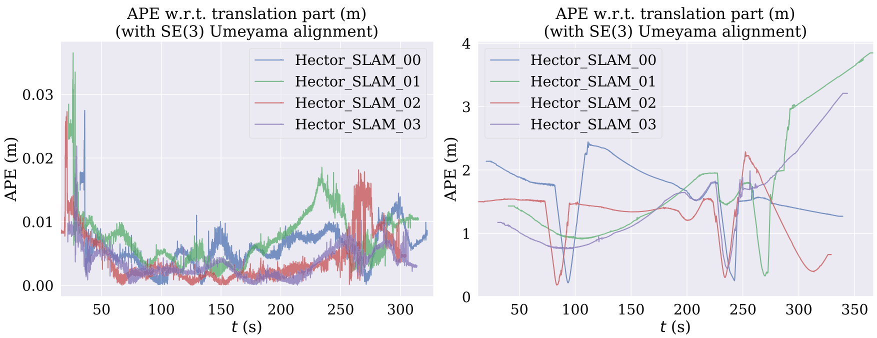

Fig. 30.6 Hector SLAM absolute pose error (APE) over time for flat (left) and uneven (right) terrains (all sequences)

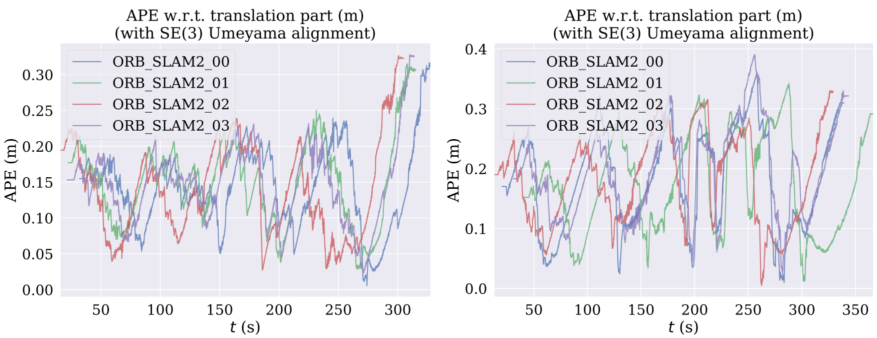

Fig. 30.7 ORB-SLAM2 absolute pose error (APE) over time for flat (left) and uneven (right) terrains (all sequences)

w.r.t the ground-truth is relatively small (Fig. 30.5). On the other hand in case of the uneven terrain, Hector SLAM diverges from the ground-truth trajectory in Y and Z axes. Hector SLAM APEs for all sequences are shown in Fig. 30.6.

As for the ORB-SLAM2, it demonstrates good results for both flat and uneven terrains resulting in translation RMSE of 0.15 and 0.19 m, respectively (Fig. 30.7). RTAB-Map showed relatively better accuracy compared to ORB-SLAM2 resulting in translation RMSE of 0.019 and 0.03 m for flat and uneven terrain, respectively.

30.5 Results

In this paper, we presented SLAM methods evaluation in ROS/Gazebo simulation considering both flat and uneven terrains, the later of which is frequently met in USAR applications. The evaluation process consisted of world creation, datasets collection, and computation of pose error metrics. The key advantages of simulations are the ease of configuration changes (using a different set of sensors and robot models) and repetitiveness of simulations.

We evaluated two popular Visual SLAM methods, ORB-SLAM2 and RTAB-Map, in both flat and uneven terrains using collected datasets, and Hector SLAM. The results showed that Hector SLAM in its simple form cannot be used in uneven terrains. However, in the case of flat terrains, due to its accuracy, it can be considered as a ground-truth for other SLAM methods evaluation. On the contrary, stereo ORB-SLAM2 and RTAB-Map estimated trajectories are fair in both cases, though in the case of flat terrains Hector SLAM pose error is less in an order of magnitude. In our future work, we plan to investigate other Visual SLAM algorithms for USAR applications.

Acknowledgements This research was funded by the Russian Foundation for Basic Research (RFBR), project ID 18-58-45017. This work was partially supported by the research grant of Kazan Federal University.

References

1. Bodin, B., Wagstaff, H., Saecdi, S., Nardi, L., Vespa, E., Mawer, J., Nisbet, A., Luján, M., Furber, S., Davison, A.J., et al.: Slambench2: multi-objective head-to-head benchmarking for visual slam. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–8. IEEE, New York (2018)
2. Bresson, G., Alsayed, Z., Yu, L., Glaser, S.: Simultaneous localization and mapping: a survey of current trends in autonomous driving. IEEE Trans. Intell. Veh. **2**(3), 194–220 (2017)
3. Bujanca, M., Gafton, P., Saeedi, S., Nisbet, A., Bodin, B., O'Boyle Michael, F., Davison, A.J., Riley, G., Lennox, B., Luján, M., et al.: Slambench 3.0: systematic automated reproducible evaluation of slam systems for robot vision challenges and scene understanding. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 6351–6358. IEEE, New York (2019)
4. Ceriani, S., Fontana, G., Giusti, A., Marzorati, D., Matteucci, M., Migliore, D., Rizzi, D., Sorrenti, D.G., Taddei, P.: Rawseeds ground truth collection systems for indoor self-localization and mapping. Autonom. Rob. **27**(4), 353 (2009)
5. Denisov, E., Sagitov, A., Lavrenov, R., Su, K.L., Svinin, M., Magid, E.: Dcegen: dense clutter environment generation tool for autonomous 3d exploration and coverage algorithms testing. In: International Conference on Interactive Collaborative Robotics, pp. 216–225. Springer, Berlin (2019)
6. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The Kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3354–3361. IEEE, Berlin (2012)
7. Giubilato, R., Chiodini, S., Pertile, M., Debei, S.: An experimental comparison of ros-compatible stereo visual slam methods for planetary rovers. In: 2018 5th IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace), pp. 386–391. IEEE, New York (2018)
8. Huletski, A., Kartashov, D., Krinkin, K.: Evaluation of the modern visual slam methods. In: 2015 Artificial Intelligence and Natural Language and Information Extraction, Social Media and Web Search FRUCT Conference (AINL-ISMW FRUCT), pp. 19–25. IEEE, New York (2015)

9. Ibragimov, I.Z., Afanasyev, I.M.: Comparison of ros-based visual slam methods in homogeneous indoor environment. In: 2017 14th Workshop on Positioning, Navigation and Communications (WPNC), pp. 1–6. IEEE, New York (2017)
10. Kasar, A.: Benchmarking and comparing popular visual slam algorithms. arXiv preprint arXiv:1811.09895 (2018)
11. Kohlbrecher, S., Meyer, J., Graber, T., Petersen, K., Klingauf, U., von Stryk, O.: Hector open source modules for autonomous mapping and navigation with rescue robots. In: Robot Soccer World Cup, pp. 624–631. Springer, Berlin (2013)
12. Lavrenov, R., Zakiev, A., Magid, E.: Automatic mapping and filtering tool: From a sensor-based occupancy grid to a 3d gazebo octomap. In: 2017 International Conference on Mechanical, System and Control Engineering (ICMSC), pp. 190–195. IEEE, New York (2017)
13. Li, W., Saeedi, S., McCormac, J., Clark, R., Tzoumanikas, D., Ye, Q., Huang, Y., Tang, R., Leutenegger, S.: Interiornet: mega-scale multi-sensor photo-realistic indoor scenes dataset. arXiv preprint arXiv:1809.00716 (2018)
14. Magid, E., Pashkin, A., Simakov, N., Abbyasov, B., Suthakorn, J., Svinin, M., Matsuno, F.: Artificial intelligence based framework for robotic search and rescue operations conducted jointly by international teams. In: Proceedings of 14th International Conference on Electromechanics and Robotics “Zavalishin’s Readings”, pp. 15–26. Springer, Berlin (2020)
15. Magid, E., Tsubouchi, T.: Static balance for rescue robot navigation: discretizing rotational motion within random step environment. In: International Conference on Simulation, Modeling, and Programming for Autonomous Robots, pp. 423–435. Springer, Berlin (2010)
16. Montemerlo, M., Thrun, S.: FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics, vol. 27. Springer, Berlin (2007)
17. Nistér, D., Naroditsky, O., Bergen, J.: Visual odometry. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004, vol. 1, pp. I–I. IEEE, New York (2004)
18. Safin, R., Lavrenov, R., Saha, S.K., Magid, E.: Experiments on mobile robot stereo vision system calibration under hardware imperfection. In: MATEC Web of Conferences, vol. 161, p. 03020. EDP Sciences (2018)
19. Safin, R., Lavrenov, R., Tsoy, T., Svinin, M., Magid, E.: Real-time video server implementation for a mobile robot. In: 2018 11th International Conference on Developments in eSystems Engineering (DeSE), pp. 180–185. IEEE, New York (2018)
20. Schubert, D., Goll, T., Demmel, N., Usenko, V., Stückler, J., Cremers, D.: The TUM VI benchmark for evaluating visual-inertial odometry. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1680–1687. IEEE, New York (2018)
21. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGB-D slam systems. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 573–580. IEEE, New York (2012)
22. Zakiev, A., Shabalina, K., Tsoy, T., Magid, E.: Pilot virtual experiments on ARUco and ARTag systems comparison for fiducial marker rotation resistance. In: Proceedings of 14th International Conference on Electromechanics and Robotics “Zavalishin’s Readings”, pp. 455–464. Springer, Berlin (2020)