Comparison of different 2D and 3D Sensors and Algorithms for Indoor SLAM on a low-cost Robotic Platform

WEI ZHANG[1], DAVID SKUDDIS[1], PHILIPP J. SCHNEIDER[1] & NORBERT HAALA[1]

Abstract: Mobile robots are becoming a fairly important part of people's lives. Whether they are service robots that assist people in daily life, such as robot vacuums or robots in industry. Simultaneous localization and mapping (SLAM) is one of the most fundamental capabilities to perceive the surroundings and keep track of the robot's position while constructing a map incrementally. SLAM-based surveying equipment is also increasingly used for areas without GNSS availability e.g. mining or indoor cartography. For this purpose, there is a wide range of products from different manufacturers. In practice, depending on the application requirements, different sensors are deployed for this task. Furthermore, with the rapid development of this field in recent years, more new methods have emerged and pushed the boundaries of sensor performance. We noticed a lack of widespread discussion and consensus on which sensor or algorithm is more suitable for a low-cost indoor robotic platform. Therefore, this work aims to compare different low-cost environmental sensors and different advanced algorithms for each of the sensors with the ultimate goal of being able to help make decisions when it comes to choosing sensors and algorithms for a specific robot application. In order to achieve a fair comparison, third-party unbiased reference data is needed. For this purpose, we utilize a wide-angle camera mounted on the ceiling and ArUco marker to achieve a bird's view tracking of the robot's poses serving as reference data. We compare the results of different sensors and algorithms quantitatively against the reference trajectory. In addition to trajectory comparison, another product of the SLAM method is the constructed 2D and 3D maps, which are compared and analyzed qualitatively.

1 Introduction

Mobile robots are widely used in various application domains. They can replace human beings in industry, agriculture and service industries to a certain extent. In addition, they can also be employed in many dangerous environments, such as emergency rescue, space exploration, construction exploration, etc.. In the research of intelligent mobile robots, many technologies are included, such as SLAM, path planning, and navigation. Among them, SLAM technology has always been a research focus in this field.

In an unknown environment, the task of SLAM methods is to obtain an accurate map of the surroundings and localize precisely the robot's position within the environment. Measurements of different 2D and 3D sensors can be used as input for SLAM. As a low-cost robotic platform, we choose the affordable 2D Lidar and 3D depth camera for comparison. With decades of development of SLAM technology, numerous algorithms for each sensor have been developed. For the 2D Lidar, the Matlab Lidar SLAM and ICP graph SLAM methods are selected. As for visual SLAM methods, there are distinctions between keypoint-based, direct, and dense methods.

[1] University of Stuttgart, Institute for Photogrammetry, Geschwister-Scholl-Str. 24D, D-70174 Stuttgart, Germany, E-Mail: [Wei.Zhang, David.Skuddis, Philipp.Schneider, Norbert.Haala]@ifp.uni-stuttgart.de

DOI: 10.24407/KXP:1796046450

For this reason, one representative method for each paradigm is selected, which are the ORB-SLAM, the Stereo-DSO, and the DROID-SLAM methods.

For the experiment, a low-cost robotic platform is assembled, consisting of a 2D Lidar and 3D depth camera. Additionally, to provide a reference for comparison, an ArUco marker is appended on top of the platform as illustrated in Fig. 1. We employed a wide-view GoPro camera on the room's roof to keep tracking the position and orientation of the robot. The experimental results show that the recent deep learning-based DROID-SLAM method performs best with an ATE error of 2.9 cm. Nevertheless, thanks to the high precision of direct distance measurements, the 2D Lidar-based SLAM provides a more consistent 2D occupancy map. Besides, the Lidar map covers more spaces because of the greater measurement range. By contrast, the resulting 3D map of the visual system contaminates more clutter due to the insufficient accuracy of depth estimate.

2 Methods

This section first explains the low-cost robotic platform, which has a modular structure consisting of three subsystems. Then each applied sensor is introduced. After that, the SLAM algorithms of each sensor modality are elaborated in detail.

2.1 Plattform

Fig. 1 presents an overview of the used low-cost robotic platform and the module diagram consisting of three subsystems, which are the Raspberry-Pi centric robot control and 2D Lidar scanning subsystem, the 3D stereo camera subsystem with the Jetson Xavier NX board as the computing unit, and the reference data provider using a GoPro camera and ArUco marker fixed on top of the robot body. Tab. 1 presents the hardware specification of the two central processing units of 2D Lidar and 3D camera systems.

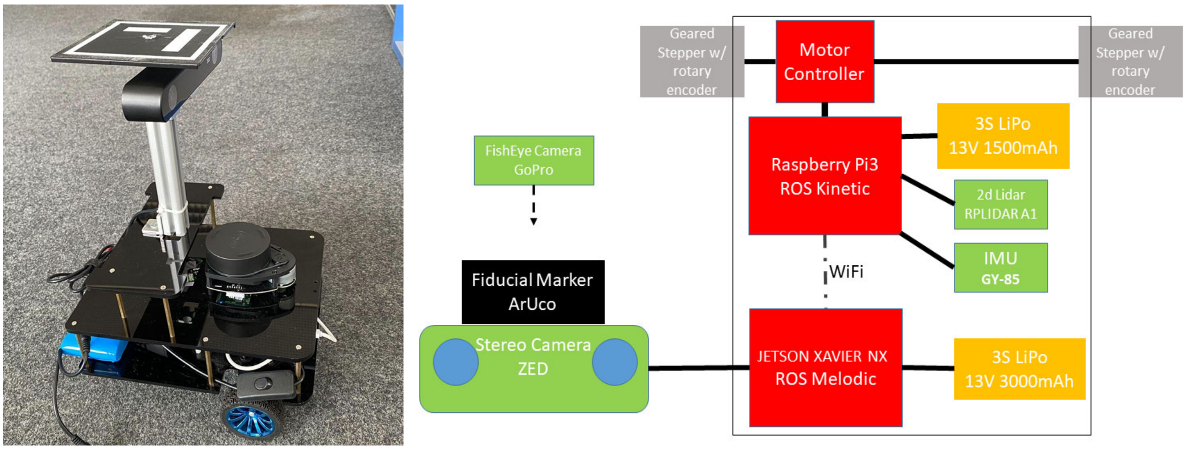

Fig. 1: The low-cost robotic platform and the module diagram

Tab. 1: Hardware specification of the two processing units on the low-cost robotic platform

	Raspberry Pi 3	**Jetson Xavier Nx**
Processor	Broadcom ARMv8 4-cores	Nvidia carmel ARMv8 6-cores
GPU	VideoCore IV	Nvidia vota 384 cores/48 tensor-cores
RAM	1GB	8GB
Sensor	RPLidar, rotary encoder	Zed2 camera
OS	Ubuntu 16.04	Ubuntu 18.04
ROS	ROS Kinetic	ROS Melodic

2.2 Sensors

This section introduces three different sensors applied in our platform, whose measurements are used to estimate the robot's trajectory.

2.2.1 Wheel Encoder

The wheel encoder is equipped on each stepper motor of the two front wheels, and it measures the angular velocities of the left and right wheels. With the available wheel radius and the baseline length between the wheels, the moving and steering speed of the robot platform can be derived, and then the 2D movements can be integrated.

2.2.2 2D Lidar

The 2D Lidar of the model *RpLidar A1* transmits laser beams to the 360-degree surrounding. With an angular resolution of 1 degree, one full spin takes exactly a point cloud of 360 points. The spin frequency is adjusted to 5 Hz so it can sample up to 3600 points per second. Two different Lidar SLAM algorithms are applied to the 2D Lidar data, and the estimated trajectories are compared against the reference.

2.2.3 Camera

The ZED2 stereo camera with a baseline length of 12 cm is applied to capture the color view and 3D information of the environment. The camera is connected to *the Jetson Xavier NX* board, on which runs the Ubuntu 18.04 operating system. Based on the ROS wrapper provided by the manufacturer, the left and right images and stereo depth estimates by ZED SDK are recorded in a rosbag file. The image resolution is set to 672x376 and recorded at 15 Hz. We choose the representative methods to verify the state-of-the-art performance of current visual SLAM methods.

2.3 SLAM Algorithms

Different SLAM algorithms are selected to produce the state-of-the-art performance of each modality. For 2D Lidar, we use Lidar SLAM implementation in the Matlab navigation toolbox and a self-implemented ICP-graph SLAM. According to (TAKETOMI et al. 2017), for visual SLAM, based on whether the image data is directly used for tracking or indirectly via feature extraction, there are two mainstream types of visual SLAM algorithms: the feature-based indirect and the photo-consistency based direct method. Furthermore, with the recent advances from the application of deep learning, a new visual SLAM paradigm emerges and is represented by the work DROID-SLAM, which we include for comparison.

Tab. 2: Different characteristics of the evaluated Lidar and visual SLAM methods

Method	Sensor	Matching	Loop closure	Map
Matlab-Lidar-SLAM	2D Lidar	Ceres optimization	Submap alignment	2D occupancy
ICP-graph-SLAM	2D Lidar	ICP	ICP close distance	2D point cloud
ORB-SLAM	Stereo	Keypoint-based	Bag of Words	Sparse point cloud
Stereo-DSO	Stereo	Photometric-based	None	Semi-dense point cloud
DROID-SLAM	Mono,RGB-D	Optical-flow-based	Visual view overlap	Dense point cloud

2.3.1 Matlab Lidar SLAM

One of the Lidar-SLAM algorithms investigated in this work is the one provided by Matlab. It is an implementation of the Google Cartographer (HESS et al. 2016). The Google Cartographer is a 2D Lidar SLAM system. The system combines local and global scan matching methods. Newly generated scans are matched locally against a submap using the Ceres solver (AGARWAL & MIERLE, 2010). Many individual submaps are created and stored as map representations. To find loop closures, submaps are compared using a branch-and-bound approach. After a loop is detected, global nonlinear optimization of the residues is performed using the Ceres solver. Since Google Cartographer is available as an open-source project, it is widely used and very popular.

2.3.2 Lidar ICP Graph SLAM

The ICP Graph SLAM is, as the name suggests, a simple self-implemented ICP-based 2D Graph SLAM algorithm. As for map representation, single downsampled scans are stored as keyframes. Newly acquired scans are first downsampled. Then they are matched with the last keyframe using the classical ICP algorithm (BESL & MCKAY 1992). Loop Closure candidates are selected based on the distance to the current keyframe. Old keyframes that are within a certain distance of the current keyframe are compared to the current keyframe using the ICP algorithm. If the residual falls below a threshold, the loop is closed. If a loop closure is found, a global pose graph optimization is performed using the g2o framework (KÜMMERLE et al. 2011). The method was created to provide a second Lidar-based option when comparing the algorithms. Two variants of the method were investigated: one with the raw Lidar data and one with the de-skewed Lidar data. De-skewing means the correction of distortion of the points induced by movement of the scanner. In our case, only rotation rates are taken into account. Velocities can be neglected due to the low vehicle speed. For this purpose the rotation rate of the robot was determined using ICP from two consecutive scans.

2.3.3 Stereo-ORB-SLAM

ORB-SLAM (CAMPOS et al. 2021) is a feature-based method, which extracts ORB features for tracking and creates a sparse point cloud as the map. It was first introduced to run in a monocular mode. The following extensions include a stereo mode, which amongst others provides an absolute scale for the resulting map. Due to the availability of a depth map from the stereo images, the system can initialize faster without the need of moving the camera for the triangulation of initial map points. Thanks to the loop closure detection module and global pose optimization (bundle adjustment), it is a complete SLAM system widely used in robotic applications.

2.3.4 Stereo-DSO

Stereo Direct Sparse Odometry (DSO) is a direct method (WANG et al. 2017), which minimizes the photometric consistency error for the pixels with sufficient intensity gradient. Within a sliding window, the selected pixels are tracked across multiple frames. While camera poses are tracked, the 3D locations of the pixels are iteratively estimated. Although there is no loop closure and global optimization, it still produces a good result with a much denser map than the ORB-SLAM result. Like ORB-SLAM, a stereo setting can provide initial depth estimation at the initial phase and thus facilitate the system initialization.

2.3.5 DROID-SLAM

DROID-SLAM (TEED & DENG 2021) is a dense optical-flow-based method, which minimizes the reprojection error with the optical flow prediction by a pre-trained neural network as the reprojection targets. The neural network extracts multi-level feature maps from color images. To predict an update of optical flow, the current flow is firstly computed based on the current pose and depth estimates, i.e. the dense projection of the pixel array between two frames with overlap view. With that, the local features at the source and reference frame can be looked up from the extracted feature maps. The local features from both frames are processed via CNN and GRU subnetwork to predict an updated optical flow. For efficiency, the keyframes are selected based on the mean motion of the optical flow, and the depth maps are down-sampled to 1/8 resolution as estimation variables. The poses and low-resolution depth maps of keyframes are jointly optimized via bundle adjustment with the Gauss-Newton algorithm. In the following experiment, the DROID-SLAM will be tested with monocular and RGB-D settings. For the former setting, only the left camera image is fed into the system. For RGB-D, the depth estimate produced by ZED SDK is used as additional depth prior to the bundle adjustment.

2.4 Ground Truth

In order to compare the different trajectories to each other a ground truth is needed. We opted to use a wide-angle camera on the ceiling to track a squared fiducial marker (ROMERO-RAMIREZ et al. 2018), mounted on top of the robot. In order to undistort the fisheye camera's images, a rectification to a rectilinear image was applied (SCARAMUZZA et al. 2006). Since all marker positions are on the same plane, a homography (HARTLEY & ZISSERMAN 2004) was applied to remove the projective distortion from the perspective linear image of this plane into a rectangular image (Fig. 2). These rectangular images are used to detect the fiducial marker's pixel coordinates and heading. By placing the rover on top of known points, a transformation from pixel coordinates to object coordinates can be computed.

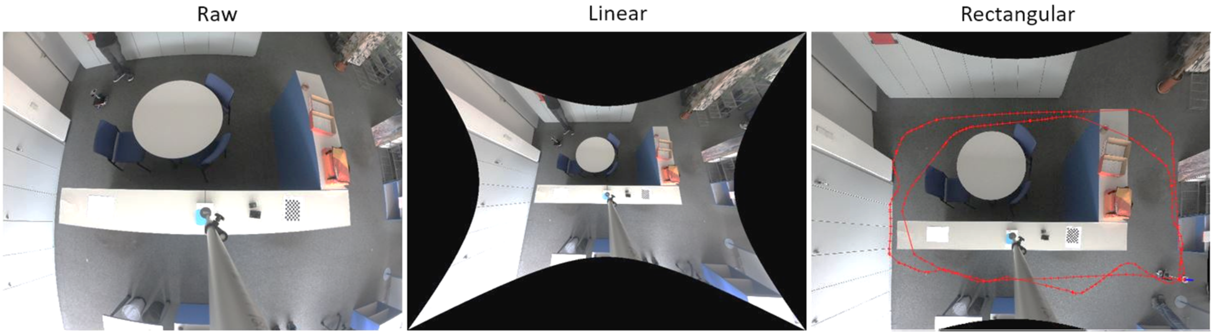

Fig.2: Workflow of reference data generation. The raw distorted images are transformed into rectilinear ones, using a fisheye camera calibration model. A homography was used to transform the marker's plane into a rectangular plane that allows measuring the rover's position in pixel coordinates, which can be transformed into object coordinates by placing the rover onto fixed points with known coordinates

2.5 Evaluation Metric

To evaluate the estimated trajectory $\mathbf{Q}_{1:n}$ by different methods, we compute the absolute trajectory error (ATE) (STURM et al. 2012) against the ground truth trajectory $\mathbf{P}_{1:n}$. Due to the difficulty of synchronizing the time of the ground truth system and sensor-specific system, the corresponding pose $\mathbf{Q}_i$ is found via the nearest neighbor search for each ground truth pose $\mathbf{P}_i$. Let *trans()* refers to the translational part of the relative pose, the error at a timestamp *i* can be computed as

$$E_i = trans(\, \mathbf{Q}_i^{-1} \cdot \mathbf{P}_i \,) \quad (1)$$

From the pose errors of all timestamps, the ATE can be derived by taking root mean square over the entire trajectory as follows:

$$ATE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \left|\left|E_i\right|\right|^2} \quad (2)$$

The ATE and the maximum of E_i (Max) are used as the metrics to indicate the average and worst performance.

3 Results

The estimated trajectories are presented in Fig. 3. For clarity, each modality is plotted in a separate subplot. The trajectory estimated by the wheel odometer has the largest deviation from the ground truth, as it suffers from the accumulated error during the movement integration over time. It should also be noted that all methods except wheel odometry and Stereo-DSO can detect loop closures by design and have detected them successfully. Since the trajectory consists of two loops, this can improve the result. For 2D Lidar-based systems, the evaluated trajectories are shown in Fig. 3b) and the evaluation results are shown in Fig. 4 and Tab. 3. Comparing the two Lidar-based SLAM approaches, it is seen that the Matlab SLAM performed marginally better than the ICP Graph SLAM.

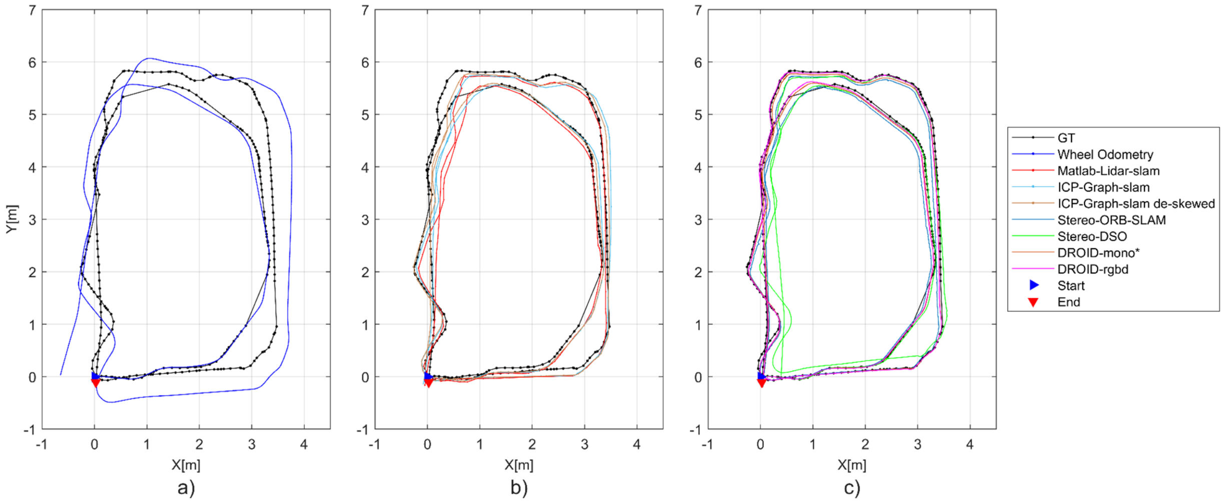

Fig. 3: Experimental results. a) shows the estimated trajectory given by the wheel odometer, b) shows the results of Lidar algorithms. c) is the estimated trajectories of visual algorithms in comparison to ground truth (GT). Note mono* method has scale ambiguity and its scale is adjusted to the GT scale.

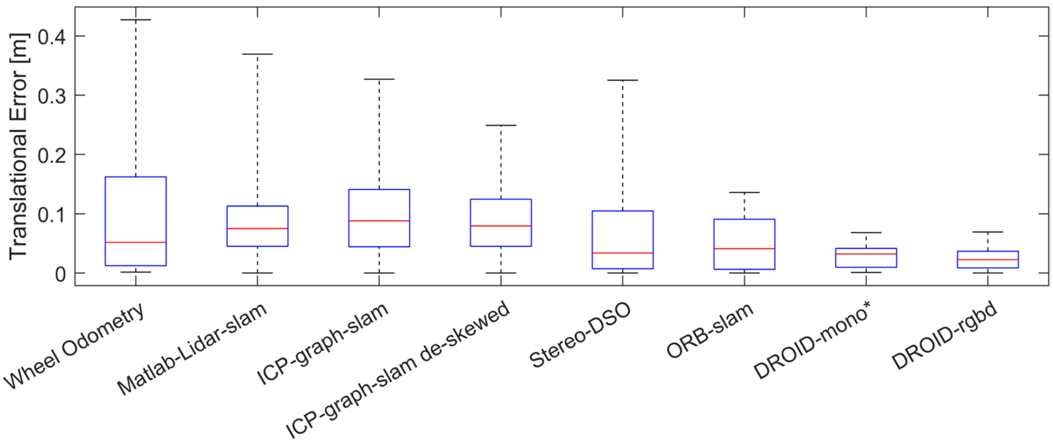

Fig. 4: Quantitative evaluation results with respect to the translational errors for different methods compared to the GT poses. The red lines mark the median in each case. The boxes mark the range between the 25th percentile and the 75th percentile.

Tab. 3: Evaluation results of different methods. Bold font indicates the best result for each metric

	Wheel	Matlab-Lidar-SLAM	ICP-graph-SLAM	ICP-graph-SLAM de-skewed	Stereo-DSO	ORB-SLAM	DROID-mono	DROID-rgbd
Data	Wheel Encoder	2D-Lidar	2D-Lidar	2D-Lidar	Stereo image	Stereo image	Mono image	Left image +depth
Loop Closure	-	+	+	+	-	+	+	+
#Pose	1663	167	125	125	2453	2554	853	853
Max Error [m]	0.567	0.369	0.322	0.249	0.325	0.136	***0.068***	0.069
ATE [m]	0.201	0.121	0.118	0.103	0.104	0.065	0.034	***0.029***

The results of the camera-based methods are shown in Fig. 3c). It can be seen that each of the camera-based methods performs better than the Lidar-based methods. For Lidar-based methods, de-skewing also significantly improved the result of the ICP-graph-slam. Furthermore, in Fig. 3 it looks like the trajectories determined by the Lidar-based methods deviate similarly in direction and magnitude from the ground truth. This may be due to a systematic measurement error of the Lidar. Within the camera-based methods, the DROID-rgbd achieves the best results. The second best results can be achieved with the DROID-mono SLAM. It should be noted that the produced result by DROID-mono is not in metric scale, which is an inherent problem of monocular SLAM. Thus the trajectory is aligned in addition by a scale factor of 1.2 to GT. The largest deviations in the camera-based methods are observed in the stereo-DSO method. The result of the stereo DSO is however remarkable in so far as it has no loop closure functionality. Thus, the errors were continuously integrated in both loops. Nevertheless, it achieves lower ATE than the Lidar-based methods as present in Tab.3. The accuracy of the ORB-SLAM is in the midfield. Nevertheless, the magnitudes of the ATE are interesting at this point. Both the ATE of the ORB-SLAM and the maximum deviation from the trajectory are twice as large as for the determined favorite DROID-rgbd.

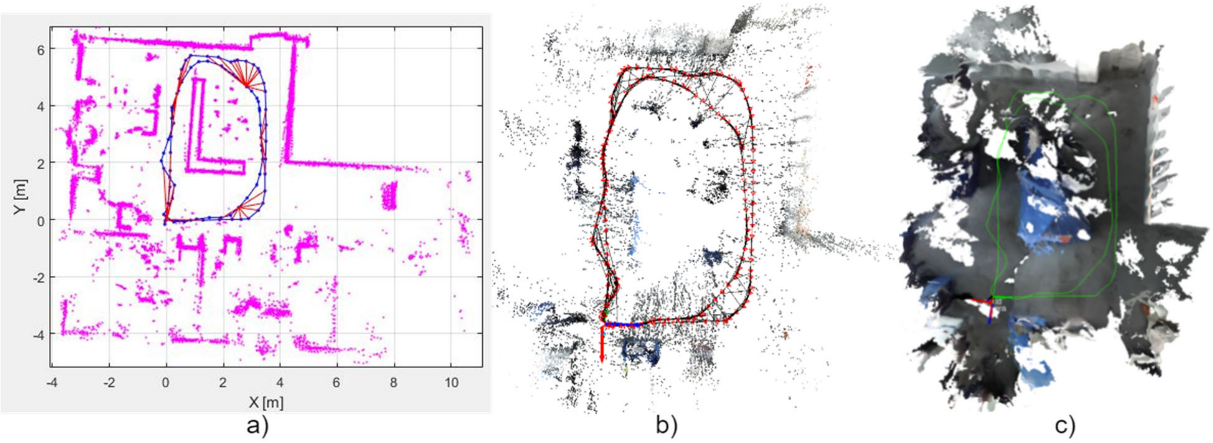

Fig. 5: Different map representations, which are 2D Lidar map, 3D point cloud map, and 3D dense volumetric map respectively

Fig. 5 shows different map representations of the applied methods. The 2D Lidar-based method produces a 2D occupancy map. As shown in Figure 5a, this 2D Lidar map is globally consistent with visually sharp edges of the room walls and furniture. By contrast, the reconstructed maps by visual SLAM methods are in 3D, whether presented as a point cloud or dense volumetric map. Compared to the Lidar map, the map reconstructed by the applied stereo camera has more clutter due to the inaccurate depth estimation. Furthermore, the space covered by the camera map is smaller than that of the Lidar map, as the camera depth range is limited by the baseline length of the stereo camera.

4 Conclusion

In our test scenario, it was shown that camera-based methods are superior to Lidar methods in the context of the used accuracy metrics. The reasons for this can be the more precise capture of the environment due to the higher resolution of the camera pixels compared to the Lidar resolution.

Nevertheless, Lidar-based methods have other advantages that do not apply in the selected scenario. These are for example the robustness in very homogeneous and featureless environments, e.g. large white walls.

It is also important to note that based on the chosen comparison scenario, no conclusions can be made about large-scale routes and the ability to detect loop closures there. Furthermore, the environment within the scenario is limited to an indoor office environment.

For future work, we believe that a more robust system can be achieved by combining the information from the 2D and 3D sensors, and also the valuable information from the inertial sensor and wheel odometer to compensate for the respective shortcomings. Moreover, towards a more intelligent robotic system, we will investigate how to derive higher-order semantic information from the constructed 2D and 3D maps.

5 Literaturverzeichnis

AGARWAL, S., MIERLE, K. & OTHERS, 2010: Ceres Solver — A Large Scale Non-linear Optimization Library. http://ceres-solver.org.

BESL, P.J. & MCKAY, N.D., 1992: A method for registration of 3-d shapes. IEEE Trans Pattern Anal Mach Intell **14**(2). 239-256, https://doi.org/10.1117/12.57955.

CAMPOS, C., ELVIRA, R., RODRIGUEZ, J.J.G., MONTIEL, J.M. & TARDOS, J., 2021: ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. IEEE Transactions on Robotics, **37**(6), 1874-1890, https://doi.org/10.1109/TRO.2021.3075644.

HARTLEY, R., ZISSERMAN, A., 2004: Projective Geometry and Transformations of 2D. Multiple View Geometry in Computer Vision, Cambridge University Press, 34-36, https://doi.org/10.1017/CBO9780511811685.005.

HESS, W., KOHLER, D., RAPP, H. & ANDOR, D., 2016: Real-Time Loop Closure in 2D LIDAR SLAM. IEEE International Conference on Robotics and Automation (ICRA), 1271-1278, https://doi.org/10.1109/ICRA.2016.7487258.

KÜMMERLE, R., GRISETTI, G., STRASDAT, H., KONOLIGE, K. & BURGARD, W., 2011: G2o: A general framework for graph optimization. IEEE International Conference on Robotics and Automation, 3607-3613, https://doi.org/10.1109/ICRA.2011.5979949.

STURM, J., ENGELHARD, N., Burgard, W. & Cremers, D., 2012: A benchmark for the evaluation of RGB-D SLAM systems. IEEE/RSJ International Conference on Intelligent Robots and Systems, 573-580, https://doi.org/10.1109/IROS.2012.6385773.

ROMERO-RAMIREZ, F.J., MUÑOZ-SALINAS, R. & MEDINA-CARNICER, R., 2018: Speeded up detection of squared fiducial markers. Image and Vision Computing, **76**, 38-47, https://doi.org/10.1016/j.imavis.2018.05.004.

SCARAMUZZA, D., MARTINELLI A. & SIEGWART R., 2006: A Toolbox for Easy Calibrating Omnidirectional Cameras. IEEE International Conference on Intelligent Robots and Systems, S.5695-5701, https://doi.org/10.1109/IROS.2006.282372.

TAKETOMI, T., UCHIYAMA, H. & IKEDA, S., 2017: Visual SLAM algorithms: a survey from 2010 to 2016. IPSJ Transaction on Computer Vision and Applications, **9**(16), https://doi.org/10.1186/s41074-017-0027-2.

TEED, Z. & DENG, J., 2021: DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. ArXiv preprint arXiv:2108.10869.

WANG, R., SCHWÖRER, M. & CREMERS, D., 2017: Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras. IEEE Inter. Conference on Computer Vision, 3903-3911, https://openaccess.thecvf.com/content_ICCV_2017/papers/Wang_Stereo_DSO_Large-Scale_ICCV_2017_paper.pdf