Comparative Analysis of ROS-Based Monocular SLAM Methods for Indoor Navigation

Alexander Buyval[a], Ilya Afanasyev[b], Evgeni Magid[c]
[a]Bryansk State Technical University, Russia
[b]Innopolis University, Russia
[c]Kazan Federal University, Russia

ABSTRACT

This paper presents a comparison of four most recent ROS-based monocular SLAM-related methods: ORB-SLAM, REMODE, LSD-SLAM, and DPPTAM, and analyzes their feasibility for a mobile robot application in indoor environment. We tested these methods using video data that was recorded from a conventional wide-angle full HD webcam with a rolling shutter. The camera was mounted on a human-operated prototype of an unmanned ground vehicle, which followed a closed-loop trajectory. Both feature-based methods (ORB-SLAM, REMODE) and direct SLAM-related algorithms (LSD-SLAM, DPPTAM) demonstrated reasonably good results in detection of volumetric objects, corners, obstacles and other local features. However, we met difficulties with recovering typical for offices homogeneously colored walls, since all of these methods created empty spaces in a reconstructed sparse 3D scene. This may cause collisions of an autonomously guided robot with unfeatured walls and thus limits applicability of maps, which are obtained by the considered monocular SLAM-related methods for indoor robot navigation.

Keywords: Monocular SLAM, localization and mapping, ROS, indoor navigation.

1. INTRODUCTION

Visual SLAM algorithms are used for simultaneous building of a 3D global map of environment while a robot is running toward a goal and tracking camera location and orientation. The 3D structure of a scene is calculated from multiple images of an estimated camera motion, taking into account a point of observation and intrinsic camera parameters. This 3D structure of environment can be dense or sparse, depending on implemented SLAM algorithms, which can create point clouds of different density. Visual SLAM methods can be classified into direct methods or feature-based methods. The dense direct methods use all image pixels to execute direct image-to-image alignment, whereas the feature-based methods apply a sparse image representation for reducing the scene to a set of observed feature points. Executing SLAM procedures, direct methods minimize photometric errors, whereas feature-based methods reduce feature reprojection errors. As far as tasks of autonomous navigation and mapping require fast, robust and simultaneously precise real-time SLAM applications, the dense direct methods implemented on Time of Flight (ToF) camera or Kinect depth data, could be difficult for implementation. That is why we observe rapid development and widespread use of both feature-based algorithms and semi-dense monocular methods such as a semi-dense monocular LSD-SLAM [1]. LSD-SLAM locally tracks the camera motion using direct image alignment and continuously builds a semi-dense depth map. Moreover, nowadays vision-based algorithms allow achieving real-time SLAM with a single RGB camera, thus substituting LIDAR, odometry and inertial measurement unit (IMU) sensors. About ten years ago, A. Davison [2] demonstrated one of the first successful camera-only real-time SLAM, named MonoSLAM. Over the last decade a number of monocular SLAM-related algorithms have appeared: Parallel Tracking and Mapping (PTAM,[3]), Dense Tracking and Mapping (DTAM, [4]), Large-Scale Direct Monocular SLAM (LSD-SLAM,[1]), REgularized MOnocular Depth Estimation (REMODE, [5]), ORB-SLAM [6], Dense Piecewise Planar Tracking and Mapping (DPPTAM,[7]), etc. Most of these methods have demonstrated sound results in tracking, mapping, and camera localization for outdoor applications, but there are many uncertainties with robust feasibility of these methods for mobile robot navigation in typical office-style indoor environment. Some critical analysis of these methods have been presented by different research groups and communities, e.g., TUM Computer Vision group[1], T. Malisiewicz[2], the ICCV

[1]LSD-SLAM: Large-Scale Direct Monocular SLAM, TUM Computer Vision group, 2015, http://vision.in.tum.de/research/vslam/lsdslam
[2]The Future of Real-Time SLAM and "Deep Learning vs SLAM", Tombone's Computer Vision Blog, January 13, 2016, www.computervisionblog.com/2016/01/why-slam-matters-future-of-real-time.html

Ninth International Conference on Machine Vision (ICMV 2016), edited by Antanas Verikas, Petia Radeva, Dmitry P. Nikolaev, Wei Zhang, Jianhong Zhou, Proc. of SPIE Vol. 10341, 103411K
 · doi: 10.1117/12.2268809

workshop on the Future of Real-Time SLAM[3], etc. Although all of the above mentioned methods have implementations in Robot Operating System (ROS)[4], the paper focuses on four ROS-based Monocular SLAM algorithms - ORB-SLAM, REMODE, LSD-SLAM, and DPPTAM - since the authors believe that these newest algorithms had absorbed most successful achievements of the previously proposed approaches. Let us note that ORB-SLAM and REMODE are feature-based algorithms, whereas LSD-SLAM and DPPTAM are semi-dense direct methods. ORB-SLAM calculates a camera trajectory and recovers a sparse 3D scene, using the same features for tracking, mapping, re-localization and loop closing. REMODE uses a depth map computation approach, which combines Bayesian estimation and convex optimization for image processing, executing a camera pose estimation by Semi-Direct Monocular Visual Odometry (SVO) [5]. LSD-SLAM creates a real-time global, semi-dense map in a fully direct mode (performing image-to-image alignment) without using keypoints, corners or any other local features (in opposite to feature-based methods like ORB-SLAM and REMODE) [1]. DPPTAM estimates a dense reconstruction of a scene in real-time, carrying out a search for planar areas from the information of a superpixel segmentation and a semi-dense map from highly textured areas [7].

This paper compares and analyzes different monocular SLAM-related algorithms that are realized in ROS using our test video data. The video data is acquired from a camera that is mounted on an indoor vehicle, which moves forward and back in a closed loop. We detect the features, reconstruct 3D scene and recover a camera trajectory to conclude about their quality and feasibility for tasks of indoor autonomous navigation. We test the algorithms in off-line mode.

2. SYSTEM SETUP

To analyze ROS-based Monocular SLAM algorithms we recorded raw video data streams from a wide-angle full HD webcam mounted on a human-operated prototype of Unmanned Ground Vehicle (UGV). The UGV computational platform is based on the Intel Core™ i3 processor and GeForce GT 740M graphics card, which supports CUDA technology and produces on-board parallel computation. To track the UGV localization and build a map, we use Genius WideCam F100[5], which is a USB webcam with a wide-angle lens up to 120 degrees. The camera can record Full HD video by software with up to 30fps, and has a support with ROS usb_cam webcam driver, allowing to use it for ROS-based Monocular SLAM application. Table 1 contains main parameters of our Vision System. The camera was calibrated with a chessboard before moving the UGV prototype. The UGV prototype travelled in an indoor environment forward and backward under the control of an operator, forming a closed-loop trajectory. The webcam was initially oriented toward the UGV main moving direction, then into side direction. Therefore, we recorded two raw video data streams (direct and side video data) of indoor environment (Fig. 1 and 2).

Table 1. The Vision System Configuration and Genious WideCam F100 Camera Characteristics

Vision System Configuration of UGV prototype		**Genious WideCam F100 Camera**	
Parameter	**Configuration**	**Parameter**	**Configuration**
Processor	Intel Core™ i3-4160 CPU @ 3.60GHz x 4	Image Sensor	1080p Full HD pixel CMOS
GPU	GeForce GT 740M	Video resolution	VGA/720P HD/1080p FHD
RAM	8 Gb	Interface	USB 2.0
Camera	Genius WideCam F100	Image Resolution	12MP, 1920x1080, 1280x720, 640x480
OS	Linux	Frame rates	up to 30fps
ROS	Jade Turtle	Lens	120 degrees
Driver	ROS usb_cam webcam	Shutter	Rolling shutter

[3]The Future of Real-Time SLAM: Dec-18 2015, ICCV Workshop, http://wp.doc.ic.ac.uk/thefutureofslam/programme/
[4]Robot Operating System (ROS) is a set of software libraries and tools to build robot applications, www.ros.org
[5]WideCam F100 is the 1080 Full HD webcam produced by Genius GmbH company, www.genius-europe.com/en/

Figure 1. Snapshots of a forward looking camera of the UGV (video 1).

Figure 2. Snapshots of side looking camera of on the UGV (video 2).

3. TESTS AND COMPARISON OF ROS-BASED MONOCULAR SLAM METHODS

3.1 Results of Monocular SLAM tests

During our tests we moved the UGV prototype forward and backward in indoor environment, recording the monocular video (Fig. 1 and Fig. 2). Then we sequentially used the four ROS-based Monocular SLAM-related methods to process the recorded video off-line, detect the features (Fig. 3) and recover point clouds (Fig. 4-7). Finally, LSD and ORB SLAM methods computed and visualized a closed-loop trajectories (Fig. 8) by built-in tools, whereas REMODE and DPPTAM did not allowed to do it. Note that visual odometry outcomes for the same closed-loop UGV trajectory were recovered incorrectly by LSD-SLAM and correctly by ORB-SLAM (Fig. 8). The main result for all of the considered SLAM-related methods is that they showed reasonably good outcomes in detecting volumetric objects, corners, and other local features. However, these methods poorly detect office-style wall surfaces colored in light paints, leaving empty space in a recovered sparse 3D scene. It can limit a feasibility of SLAM maps obtained by these Monocular SLAM-related algorithms for indoor robot navigation. Next, we present brief overview of the Monocular SLAM tests' peculiarities for each method separately.

Figure 3. From left to right: Feature detection visualization by ORB-SLAM, REMODE, LSD-SLAM, and DPPTAM methods.

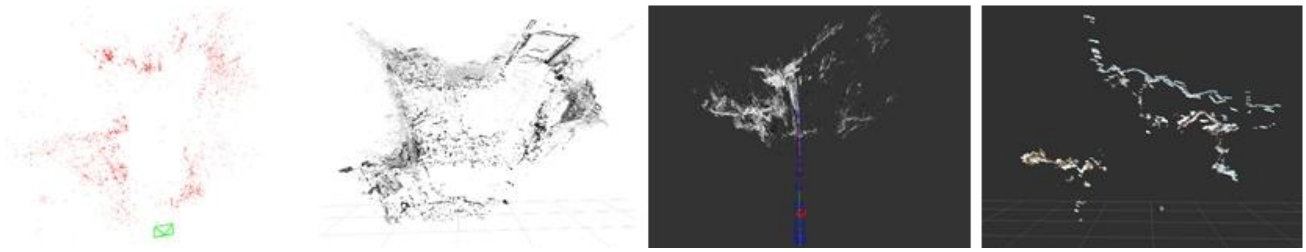

Figure 4. Point clouds (top view), which were recovered from video 1 by ORB-SLAM (left), REMODE (middle left), LSD-SLAM (middle right), and DPPTAM (right).

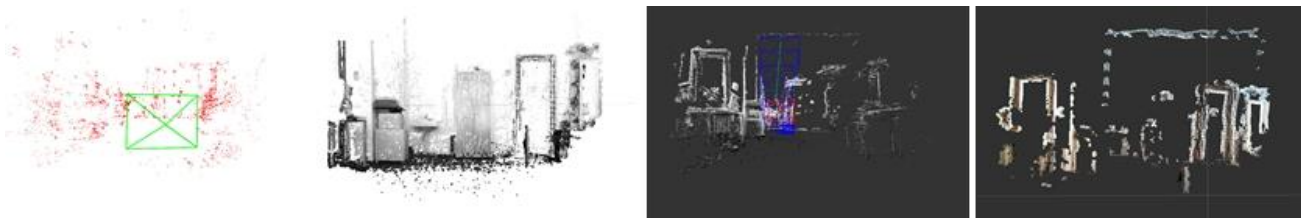

Figure 5. Point clouds (forward view), which were recovered from video 1 by ORB-SLAM (left), REMODE (middle left), LSD-SLAM (middle right), and DPPTAM (right).

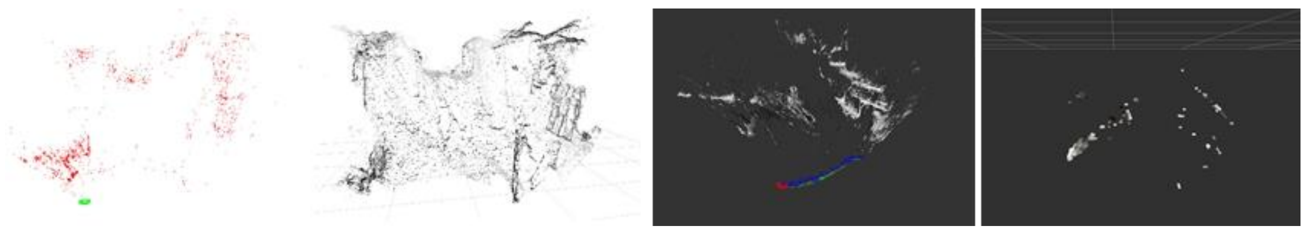

Figure 6. Point clouds (top view), which were recovered from video 2 by ORB-SLAM (left), REMODE (middle left), LSD-SLAM (middle right), and DPPTAM (right).

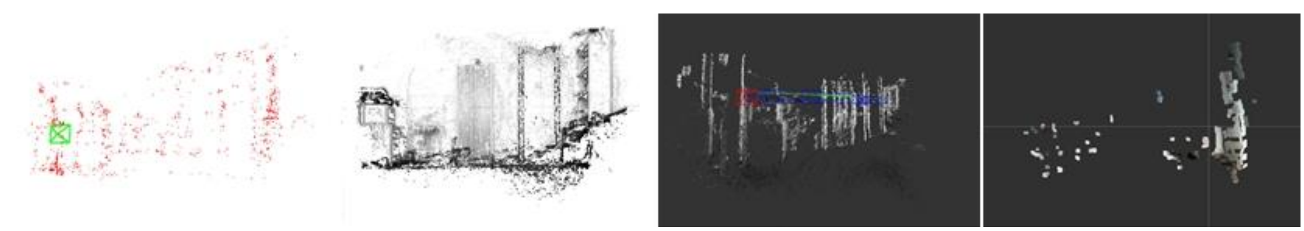

Figure 7. Point clouds (forward view), which were recovered from video 2 by ORB-SLAM (left), REMODE (middle left), LSD-SLAM (middle right), and DPPTAM (right).

3.1.1 ORB-SLAM tests.

The method produces only 3D cloud of observed features - therefore this cloud is too sparse. However, if compared with REMODE method, ORB-SLAM creates significantly fewer false points (outliers).

3.1.2 REMODE tests.

This method is optimized for plane-parallel motion of a camera relatively to the observed surface that can be typical, for example, for a camera, which is mounted on an unmanned aerial vehicle (UAV) and is oriented downwards, toward the ground. For other cases of vehicle motion, REMODE method may lose features that are detected in images and stop creating new ones. In our experiments, in order to work with REMODE method we set max_fts parameter (maximum number of features) to 300 features. Analyzing the results of REMODE tests, we paid attention that it creates a very similar to experimental environment 3D point cloud. Moreover, what is more important for all types of robot applications, REMODE depth estimation is reasonably good in forming the ground plane. However, we should notice that REMODE creates false points inside empty space that can result in a false obstacle detection by autonomous navigation algorithms.

3.1.3 LSD-SLAM tests.

This method is optimized for the default image resolution of 640x480. Other image resolutions should correspond to a multiplication factor of 16. Built-in visual odometry algorithm generates sufficiently large errors that decrease the overall quality of mapping. For example, while processing the test data of our side-pointed camera (video 2), LSD-SLAM visual odometry erroneously recovered the UGV closed-loop trajectory with a shift of about 1 m (Fig. 8, left). This method also demonstrates sensitivity to illumination changes between video frames, which limits its application for environment with blinking lighting conditions.

3.1.4 DPPTAM tests.

This method is quite close to LSD-SLAM, but in addition, it tries to estimate planar areas, which belong to homogeneously colored regions (e.g., walls and a ground plane). However, in practice, DPPTAM has demonstrated the worst 3D reconstruction quality for our test data (especially for video 2), although the low noise level (a quantity of noise points) was not significant.

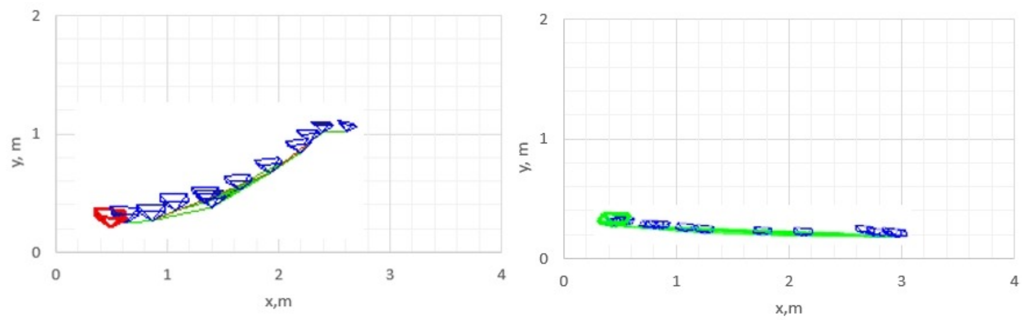

Figure 8. The visual odometry outcomes for the same closed-loop trajectory of the UGV prototype: incorrect by LSD-SLAM (left) and correct recovery by ORB-SLAM (right).

3.2 Comparative analysis of Monocular SLAM algorithms

Table 2 presents our comparative analysis of Monocular SLAM-related algorithms: ORB-SLAM, REMODE, LSD-SLAM and DPPTAM.

Table 2. The comparison of Monocular SLAM-related methods

Parameter	**ORB-SLAM**	**REMODE**	**LSD-SLAM**	**DPPTAM**
Type of method	Feature-based	Feature-based	Direct	Direct
CUDA-enabled	No	Yes	No	No
Separate odometry module	No (Built-in)	Yes (SVO)	No (Built-in)	No (Built-in)
Camera trajectory module	Yes	No	Yes	No
Visualization	Built-in	ROS/RViz	Built-in	ROS/RViz
Noise level	Low	Middle	High	Low
Visual odometry quality	Good	Good	Poor	Poor

We have considered two feature-based methods (ORB-SLAM and REMODE), and two Direct SLAM algorithms (LSD-SLAM and DPPTAM). Only REMODE method has CUDA-based implementation and works with the separate odometry module (SVO). Only ORB and LSD SLAM methods visualize camera trajectories with built-in tools, although for our test data the closed-loop UGV trajectory was incorrectly reconstructed by LSD and correctly by ORB SLAM. REMODE and DPPTAM apply ROS/RViz tool for 3D scene and map visualization, whereas ORB and LSD SLAM use their own graphical user interfaces. Although, it is difficult to compute noise level of these SLAM-related algorithms (i.e. approximate amount of false points in a recovered 3D scene), we visually estimated 3D scene reconstruction quality relatively to each other, ranking the noise level of ORB-SLAM and DPPTAM as the lowest, REMODE as middle, and LSD-SLAM as the highest. However, as far as DPPTAM has demonstrated the worst 3D reconstruction quality for our test data (especially for video 2), the "low noise level" is quite approximate. Processing our video, feature-based methods have shown better visual odometry quality than direct methods. Nevertheless, for all of these SLAM-related methods we met difficulties with recovering typical for offices homogeneously colored walls, instead of which they obtained empty spaces in a reconstructed 3D scene.

4. CONCLUSIONS AND DISCUSSION

We have compared the four most recent challenging ROS-based Monocular SLAM-related algorithms: ORB-SLAM, REMODE, LSD-SLAM, and DPPTAM, which absorbed successful achievements of earlier SLAM methods. During our tests, we moved the human-operated UGV prototype forward and backward in indoor environment of a typical office with monotonous off-white walls. To record a test video we used a conventional wide-angle Full HD webcam with rolling shutter and a frame rate of 30 fps. Then we processed the recorded video off-line by the above-mentioned Monocular SLAM-related methods that are implemented in ROS, reconstructing 3D scene, a general map and a camera trajectory. The camera trajectory visualization option is available only for ORB and LSD SLAM methods, but for our test data, the closed-loop UGV trajectory was correctly reconstructed only by ORB-SLAM. Although we used both feature-based and direct SLAM methods, the main results for indoor mobile application were approximately the same. Almost all of considered SLAM-related methods demonstrated reasonably good outcomes in detecting volumetric objects, corners, and other local features. However, these methods poorly detected office-style wall surfaces colored in light paints, leaving gaps in point clouds of a sparse 3D scene, where the surface of walls should be. It may limit applicability of maps, which are obtained by these Monocular SLAM-related methods for indoor robot navigation because of a collision risk of an UGV with walls. Therefore, the question of robust feasibility of considered ROS-based Monocular SLAM-related methods for autonomous indoor navigation stays open.

ACKNOWLEDGEMENTS

This research has been carried out in the framework of Russian-Israeli cooperative scientific research grant ID 15-57-06010 “Localization, mapping and path planning for an unmanned ground vehicle (UGV) with a help of a group of unmanned aerial vehicles (UAVs) using active collaborative vision and multi-robot belief space planning” funded by Russian Foundation of Basic Research and Ministry of Science Technology & Space state of Israel (2015-2017).

REFERENCES

[1] J. Engel, “LSD-SLAM: Large-Scale Direct Monocular SLAM”, In: 13th European Conf. on Computer Vision, Zurich, Switzerland, 834-849, (2014)

[2] A.J. Davison, I.D. Reid and N.D. Molton, O. Stasse, “MonoSLAM: Real-Time Single Camera SLAM”, IEEE Trans. on Pattern Analysis and Machine Intelligence, **29**(6), 1052-1067, (2007)

[3] G. Klein and D. Murray, “Parallel Tracking and Mapping for Small AR Workspaces”, In: 6th IEEE & ACM Int. Symposium on Mixed and Augmented Reality, Nara, Japan, 1-10, (2007)

[4] R.A. Newcombe, S.J. Lovegrove and A.J. Davison, “DTAM: Dense Tracking and Mapping in Real-time”, In: IEEE Int. Conf. on Computer Vision, 2320-2327, (2011)

[5] M. Pizzoli, C. Forster and D. Scaramuzza, “REMODE: Probabilistic, monocular dense reconstruction in real time”, In: IEEE Int. Conf. on Robotics and Automation, 2609-2616, (2014)

[6] R. Mur-Artal, J.M.M. Montiel and J.D. Tardos “ORB-SLAM: A Versatile and Accurate Monocular SLAM System”, In: IEEE Trans. on Robotics, **31**(5), 1147-1163, (2015)

[7] A. Concha and J. Civera, “DPPTAM: Dense Piecewise Planar Tracking and Mapping from a Monocular Sequence”, In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 5686-5693, (2015)