CLAWAR 2020: 23rd International Conference on
Climbing and Walking Robots and the Support
Technologies for Mobile Machines,
Moscow, Russian Federation, 24-26 August 2020.
https://doi.org/10.13180/clawar.2020.24-26.08.12

Comparative analysis of ROS-based centralized methods for conducting collaborative monocular visual SLAM using a pair of UAVs

BULAT ABBYASOV, ROMAN LAVRENOV, AUFAR ZAKIEV, TATYANA TSOY and EVGENI MAGID

Laboratory of Intelligent Robotic Systems, Intelligent Robotics Department, Higher Institute for Information Technology and Intelligent Systems (ITIS), Kazan Federal University, 35 Kremlyovskaya street, Kazan, 420008, Russian Federation
E-mail: BuRAbbyasov@kpfu.ru, lavrenov@it.kfu.ru, zaufar@it.kfu.ru, tt@it.kfu.ru, magid@it.kfu.ru
kpfu.ru/robolab.html

MIKHAIL SVININ

Information Science and Engineering Department, College of Information Science and Engineering, Ritsumeikan University, 1-1-1 Noji-higashi, Kusatsu, Shiga 525-8577, Japan
E-mail: svinin@fc.ritsumei.ac.jp

EDGAR A. MARTÍNEZ-GARCÍA

Institute of Engineering and Technology, Department of Industrial Engineering and Manufacturing, Autonomous University of Ciudad Juarez, Manuel Díaz H. No. 518-B Zona Pronaf Condominio 32315 Cd Juárez, Mexico
E-mail: edmartin@uacj.mx
robo-server.uacj.mx

Unmanned Aerial Vehicle (UAV) is a flying robot that acts without a constant human pilot involvement. UAVs are applied in military and civilian areas, in search and rescue operations, 3D mapping, simultaneous localization and mapping (SLAM) and other tasks. SLAM approaches are based on various sensors usage including lidars and cameras. Visual SLAM approaches rely on visual sensing systems and successfully operate within GPS-denied environments. Further, applying several UAVs allows for complex tasks that cannot be handled by a single robot, minimizes exploration time and adds a security level for a case of a single robot failure. This paper presents a comparison of two most applicable vision-based collaborative monocular SLAM methods in Robot operating system, CORB-SLAM and CCM-SLAM, that run on a pair of UAVs. The evaluation is performed on preassembled datasets that correspond to a virtual environment in the Gazebo simulator. The error estimation in virtual experiments demonstrated that CCM-SLAM has a higher global localization accuracy than CORB-SLAM.

Keywords: UAV; visual SLAM; ROS; Gazebo simulation; algorithm comparison.

1. Introduction

Intelligent unmanned aerial vehicles (UAVs) nowadays play a significant role in security monitoring, locating victims after earthquake or landslides disasters, remote area surveillance, hazardous environment monitoring and small package delivery.[1,2] Since most of these environments typically suffer from a GPS signal jitter and interference, an alternative localization system is required.[3] One of the possible solutions in GPS-denied environments is combining an inertial measurement unit (IMU) that could determine a UAV position in space with an onboard monocular camera for building a map and localizing the UAV.[4] Such tasks, referred as simultaneous localization and mapping (SLAM), are among the most interesting challenges for robotics community research.[5,6] The goal of a Visual SLAM (vSLAM) method is to map an unknown environment and to localize a robot (or a sensor) within this map with a focus on real-time operation while using a visual sensor or several visual sensors.[7] A monocular camera is a good selection of a visual sensor for a UAV-based vSLAM due to its reasonable price, light weight and ability to provide reach data about an

underlying terrain.

One of the disadvantages of a monocular camera based vSLAM is that missing depth information (e.g., on the contrary with RGB-D camera or a laser range finder) complicates system initialization while creating an initial map. While original vSLAM algorithms were constructed for single robots, recent achievements in collaborative vSLAM usage[8] allowed a significant speedup in mapping and environment exploration time.[9]

2. Related Work

Most vSLAM systems could be divided into two broad categories: indirect (feature-based) and direct methods.[10] While indirect methods use only specific image features, e.g., object contour corners or particular image patterns, direct approaches operate directly with raw pixel intensities. The main drawback of using a direct method in collaborative scenarios is an increasing communication network load as agents need to exchange with raw images. An architecture of a vSLAM method could be centralized or decentralized (distributed).

Centralized architecture is a wide-spread SLAM system that consists of a server (a ground station) and participants (clients). The server performs resource-intensive tasks of executing computationally expensive algorithms, e.g., Bundle Adjustment[11] or Place Recognition.[12] Such server allows to equip UAVs with a low-cost central processing unit (CPU) and a monocular camera, and thus limited resources of a UAV are used to execute only simple and critical tasks, e.g., a real-time visual odometry.

In 2012 Zou and Tan[13] presented CoSLAM algorithm - a collaborative vSLAM for dynamic environments that simultaneously employs multiple cameras. Cameras were clustered into groups according to their view overlapping and worked together to reconstruct a global map. The approach requires the onboard cameras of different UAVs to capture an identical picture for synchronizing at cameras initialization stage and Nvidia graphics card at the server for resource-intensive calculations.

In 2017 Li et al.[14] proposed CORB-SLAM, which is a collaborative ORB-SLAM2[15] based method. It is a centralized multi-robot vSLAM that provides map fusion and map sharing capabilities. Each agent performs local mapping and then transmits data to a central server for further processing.

In 2019 Schmuck and Chli[16] developed a centralized collaborative monocular SLAM framework CCM-SLAM with a communication strategy that allows to operate with a limited bandwidth. Individual UAVs use their own limited resource for executing real-time visual odometry, while a server receives each client's experiences and performs a resource-intensive task of merging individual maps into a single map.

In a **decentralized approach**, UAVs exchange experiences directly with each other and perform all tasks onboard. The biggest challenges in decentralized systems are data overlap detection and efficient information sharing between team members under network delays and communication failures.

Cieslewski et.al.[17] presented a data-efficient decentralized vSLAM system that is focused on effective data exchange between team members. Chen et al.[18] proposed a distributed multi-agent SLAM system based on a novel collaborative relative pose estimating and map merging method. Moreover, the approach doesn't need initial relative poses of robots.

Egodagamage and Tuceryan[19] developed a collaborative augmented reality framework based on LSD-SLAM solution with freely moving cameras without knowledge of their initial relative poses. The framework detects map overlaps of agents using appearance-based method, ORB-keypoint detectors[20] and BRISK-descriptors combination.[21] Yet, decentralized SLAM systems require resource-intensive computational equipment and efficient data exchange protocols.

3. Description of the selected SLAM algorithms

For evaluation, we selected two most popular feature-based centralized collaborative vSLAM methods with an open-source implementation: CORB-SLAM and CCM-SLAM. Both algorithms are based on the original ORB-SLAM2 algorithm[15] and extend it for multiple cameras use. ORB-SLAM2 is a versatile feature-based SLAM solution for monocular, stereo and RGB-D cameras, which utilizes the ORB (Oriented FAST and Rotated BRIEF) fast feature detector for tracking, mapping and place recognition. These features are robust to rotation and scale and have a good invariance to camera auto-gain and auto-exposure, and illumination changes. Moreover, the features are fast to extract and match allowing for real-time operation and show good precision-recall performance in bag-of-word place recognition.[22] The system has the embedded place recognition module based on DBoW2[12] for relocalization or for reinitialization in an already mapped scene, and for a loop detection.

CORB-SLAM[14] is an extended version of ORB-SLAM2 in which clients transmit local map information to a server, and the server creates a single integrated map. Next, the server sends each client updated (at the global map generation procedure) data about a changed part. The clients only refer to the updated global map and do not reflect it in their local onboard maps.

In **CCM-SLAM**[16] each agent runs only visual odometry with a limited number of key frames. Data, including detected features, their descriptors, and 3D positions, are constantly sent to a server. The server constructs local maps from those key frames, trying to close a loop and merge the local maps. Pose optimization and bundle adjustment are also applied by the server to further refine those maps. Agents constantly download updated key frames from the server to augment their own map for a better pose estimation.

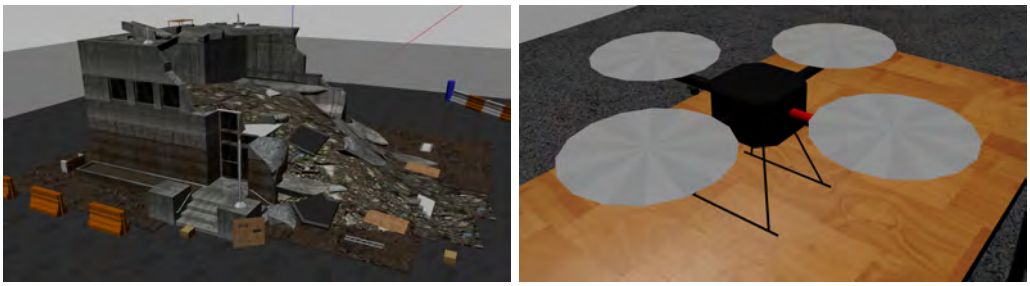
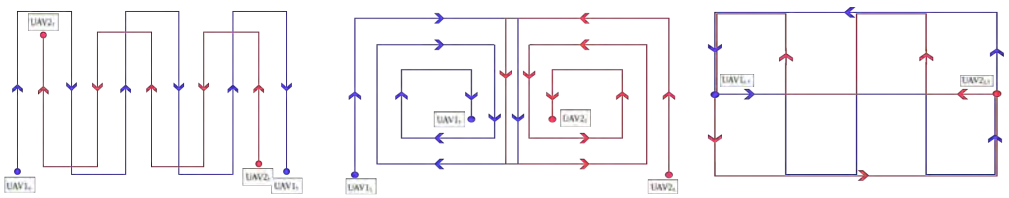


Fig. 1: A virtual world in the Gazebo simulator. The collapsed building (left) is patrolled by a pair of Hector_Quadrotor UAVs (right).



(a) Bountrophedon[23] paths. (b) Spiral paths. (c) Closed F-shape paths.

Fig. 2: Area coverage paths: the red and blue lines are UAVs trajectories. S and T denote start and target positions respectively.

4. Virtual Experiments

4.1. *Experimental configuration*

We used the Hector Quadrotor UAV[24] of Technische Universität Darmstadt, which is integrated into the Robot operating system (ROS) framework as *hector_quadrotor* ROS package for its modelling and control in the Gazebo simulator.[25] The package provides a 3D-model of the UAV (Fig. 1, right), onboard cameras, laser rangefinders and plugins for control. The default monocular camera of the *hector_quadrotor* package is a forward-looking camera. Such configuration is not suitable for our virtual experiments setup, because UAVs perform a vSLAM task for an underlying terrain. We used the Unified Robot Description Format (URDF) to modify the camera into a perspective downward-looking onboard camera.

For vSLAM performance evaluation we constructed a virtual Gazebo world[26,27] that contains the modified *collapsed police station*[28] model, which was extended with additional objects, including pieces of walls, trusses and furniture (Fig. 1, left). All vSLAM calculations, both on the server and the client sides, were carried out on the same personal computer with Intel Core i5-3210M processor, 6 GB RAM, and Ubuntu 16 (x64) operating system.

4.2. *Surface coverage and area reconstruction*

UAVs' movements should cover an entire area of interest involving the Coverage Path Planning (CPP)[29] mechanism, which has a broad variety of practical applications.[30–33] The aerial CPP is a path planning where a UAV covers all accessible parts of an exploration area. In our virtual experiments two UAVs used three different coverage paths (Fig. 2) in order to explore the entire underlying area (Fig. 1). A process of a surface reconstruction is a restoration of a digital representation of a scanned physical form. In our case, scanning provides a 3D point cloud as a result of a monocular vSLAM algorithm execution. Two UAVs explore the collapsed building area within Gazebo virtual experiments and obtain 3D point clouds while running CCM-SLAM (Fig. 3) and CORB-SLAM (Fig. 4) algorithms.

Fig. 3: CCM-SLAM: the 3D point cloud of the collapsed building.

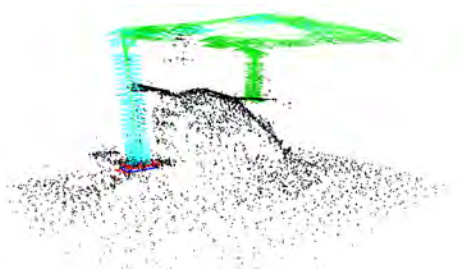


Fig. 4: CORB-SLAM: 3D view of the world; the blue/green rectangles are key frames.

4.3. *Visual odometry*

Localization accuracy is an important feature of a navigation task, and visual odometry mechanisms allow a camera position estimation and a UAV localization within a map (Fig. 5). A visual odometry component receives image snapshots and for each frame calculates movement of a camera by tracking scene landmarks.[34]

5. Comparison of SLAM methods

We compared the two vSLAM methods, CORB-SLAM and CCM-SLAM, by considering their localization accuracy and quality of a resulting 3D point cloud. Table 1 shows results

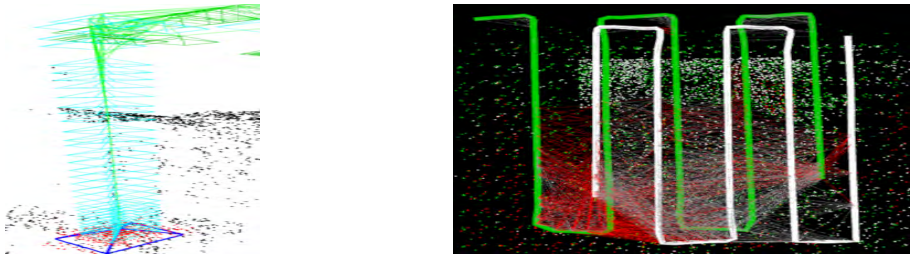
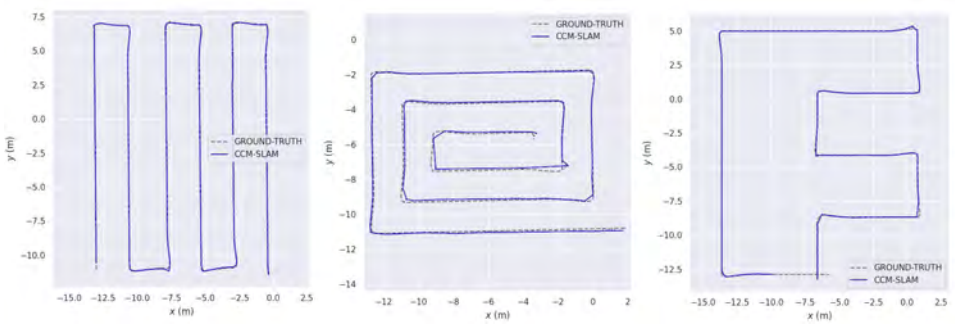
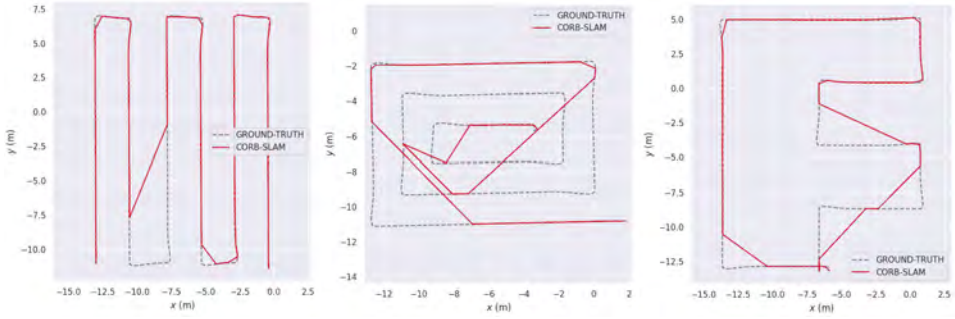


Fig. 5: Visual Odometry: the estimated key frames (the blue rectangles) while the UAV was landing using CORB-SLAM (left) and visualization of CCM-SLAM odometry-based computed trajectories in RViZ (right).



(a) CCM-SLAM estimated trajectories

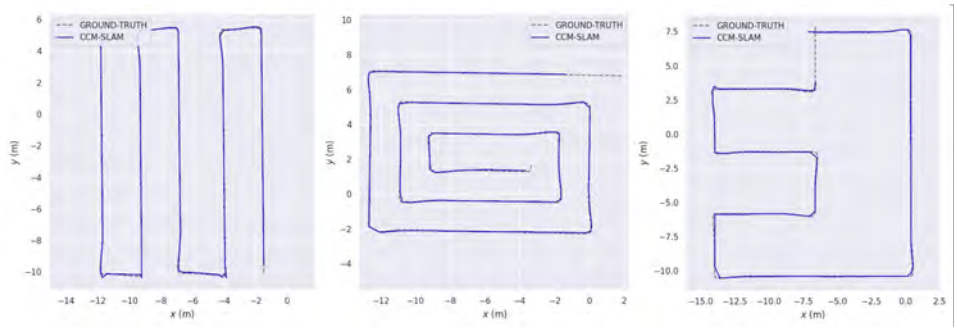


(b) CORB-SLAM estimated trajectories

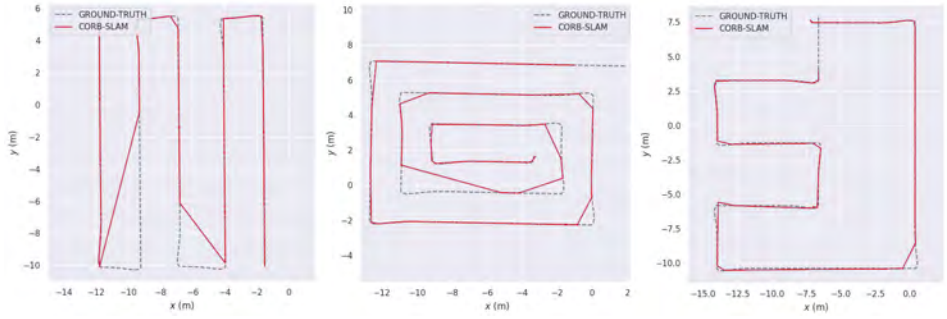
Fig. 6: Comparison of the trajectory output of the **left-sided** UAV (in meters) computed by monocular CCM-SLAM (top) and CORB-SLAM (bottom) methods.

of error estimation for CORB-SLAM and CCM-SLAM with regard to ground truth data while the pair of UAVs explored the environment using one of the three predefined trajectory types - a bountrophedon, a spiral or a closed F-shape paths. The third column specifies a UAV position at the start of the exploration: the $left$ ($right$) UAV always starts at the left (right) bottom corner of a map for all trajectory types. In Fig. 2 the paths of the $left$ and $right$ UAVs are depicted with blue and red colors respectively.

We used Absolute Trajectory Error (ATE) for measuring difference between points of the ground truth UAV trajectory and the estimated by each SLAM algorithm UAV trajectory. To represent the ATE evaluation function we used maximum and minimum ATE values, root mean square error (RMSE), median and standard deviation,[35] which appear in columns 4-8 respectively, all measured in metres. To calculate these values from the SLAM algorithms' odometry and the ground truth data, the EVO package[36] was used. The table shows that CCM-SLAM ATE values significantly exceed the ones of CORB-SLAM, e.g., the average



(a) CCM-SLAM estimated trajectories



(b) CORB-SLAM estimated trajectories

Fig. 7: Comparison of the trajectory output of the **right-sided** UAV (in meters) computed by monocular CCM-SLAM (top) and CORB-SLAM (bottom) methods.

median error of CCM-SLAM exceeds the median error of CORB-SLAM in 2.56 to 3.22 times. However, this happens only locally and does not work at a global scale due to the key frame optimization procedure, applied by CORB-SLAM, which is explained in some more details in the next paragraph. Thus, as Table 1 shows, CORB-SLAM is more precise than CCM-SLAM while localizing at a small-scale of local patches.

At the same time, the algorithms' behaviors at a global scale differ from the local patch results. The trajectories estimated by visual odometry of CORB and CCM odometry modules while following the three predefined trajectories are presented in Fig. 6 and Fig. 7. Even a rough estimation of these global trajectories clearly demonstrates that CCM-SLAM was significantly more efficient in localization than CORB-SLAM. The CCM-SLAM odometry localized more accurately at a vertical distance (along Z-axis) from an object's surface, while the CORB-SLAM odometry attempted to optimize key frames by merging similar frames, thus loosing a significant amount of valuable intermediate data of the in-between frames. For example, in Fig. 7 b, left sub-figure, all frames between a key frame at (-9.2,-0.8) and a key frame at (-12,-10) were lost. In such cases the CORB-SLAM operated UAV trajectory demonstrated significant deviations from the ground truth trajectory.

As for the mapping, both algorithms constructed sparse 3D point clouds, and Figures 8, 9 demonstrate maps that were generated by CCM-SLAM and CORB-SLAM respectively. The maps were constructed by the ORB-SLAM2 mapping system module for both CCM-SLAM and CORB-SLAM, and had the same high quality.

6. Conclusions

This paper presented a comparison of two centralized vision-based collaborative monocular SLAM methods, CORB-SLAM and CCM-SLAM. The proposed vSLAM methods were evaluated and analyzed at preassembled datasets within virtual experiments in Gazebo

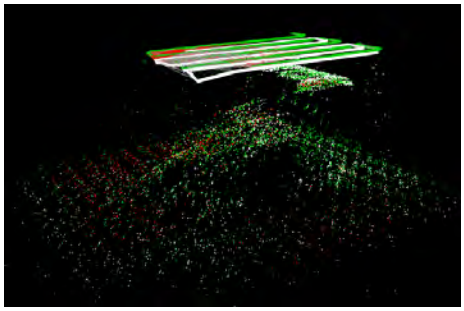


Fig. 8: The 3D point cloud of CCM-SLAM. The green and white lines show the trajectories of the right and the left UAVs respectively.

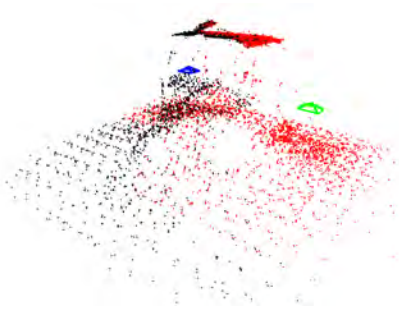


Fig. 9: The 3D point cloud of CORB-SLAM. The blue and green rectangles indicate the positions of the right and the left UAV's camera respectively.

Table 1: Absolute Trajectory Error for all trajectories

Trajectory	SLAM method	UAV position	Max (m)	Min (m)	RMSE (m)	Median (m)	Standart Deviation (m)
Bountrophedon	CORB-SLAM	LEFT	0.043540	0.000224	0.005823	0.001949	0.004983
		RIGHT	0.008748	0.000215	0.002226	0.000987	0.001654
	CCM-SLAM	LEFT	0.056326	0.000160	0.009041	0.004295	0.006303
		RIGHT	0.041613	0.000716	0.007012	0.003210	0.005277
Spiral	CORB-SLAM	LEFT	0.016961	0.001380	0.006423	0.002834	0.004127
		RIGHT	0.040958	0.000436	0.007334	0.003275	0.005649
	CCM-SLAM	LEFT	0.048540	0.001804	0.010874	0.008390	0.005518
		RIGHT	0.069150	0.001084	0.020647	0.011273	0.013048
F-shape	CORB-SLAM	LEFT	0.028148	0.000231	0.004053	0.001566	0.003402
		RIGHT	0.119337	0.002851	0.021766	0.008247	0.017544
	CCM-SLAM	LEFT	0.064889	0.000479	0.011147	0.002747	0.009578
		RIGHT	0.145269	0.003860	0.035081	0.023757	0.020233

simulation. Two UAVs performed CORB-SLAM and CCM-SLAM exploration of a virtual environment using three predefined trajectories of a bountrophedon, a spiral or a closed F-shape trajectories. The error estimation demonstrated that CORB-SLAM had a higher localization accuracy at a local scale. At the same time, at a global scale CCM-SLAM significantly outperformed CORB-SLAM; the later demonstrated unstable behavior with regard to a ground truth trajectory due to its key frames optimization approach.

Acknowledgment

This work was supported by the Russian Foundation for Basic Research (RFBR), Project No. 19-58-70002, and by the Japan Science and Technology Agency, the JST Strategic International Collaborative Research Program, Project No. 18065977, within a joint eAsia project framework. This work was partially supported by the research grant of Kazan Federal University.

References

1. E. Magid, A. Pashkin, N. Simakov, B. Abbyasov, J. Suthakorn, M. Svinin and F. Matsuno, Artificial intelligence based framework for robotic search and rescue operations conducted jointly by international teams, in *Smart Innovation, Systems and Technologies*, pp. 15–26, 2020.
2. J. Li, Y. Bi, M. Lan, H. Qin, M. Shan, F. Lin and B. M. Chen, Real-time simultaneous localization and mapping for uav: a survey, in *Proc. of International micro air vehicle competition and conference*, 2016.

3. A. Ronzhin, A. Saveliev, O. Basov and S. Solyonyj, Conceptual model of cyberphysical environment based on collaborative work of distributed means and mobile robots, in *International Conference on Interactive Collaborative Robotics*, 2016.
4. S. Weiss, D. Scaramuzza and R. Siegwart, *Journal of Field Robotics* **28**, 854 (2011).
5. E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser and P. Sayd, Monocular vision based slam for mobile robots, in *18th International Conference on Pattern Recognition (ICPR'06)*, 2006.
6. R. Lavrenov and E. Magid, Towards heterogeneous robot team path planning: acquisition of multiple routes with a modified spline-based algorithm, in *MATEC Web of Conferences*, 2017.
7. R. Safin, R. Lavrenov, T. Tsoy, M. Svinin and E. Magid, Real-time video server implementation for a mobile robot, in *Int. Conf. on Developments in eSystems Engineering (DeSE)*, 2018.
8. N. Chebrolu, D. Marquez-Gamez and P. Martinet, *PPNIV* **2**, p. 4 (2015).
9. T. Yang, P. Li, H. Zhang, J. Li and Z. Li, *Electronics* **7**, p. 73 (2018).
10. J. Zijlmans, *Improving Monocular SLAM: using Depth Estimating CNN* 2018.
11. B. Triggs, P. F. McLauchlan, R. I. Hartley and A. W. Fitzgibbon, Bundle adjustment—a modern synthesis, in *International workshop on vision algorithms*, 1999.
12. D. Gálvez-López and J. D. Tardos, *IEEE Transactions on Robotics* **28**, 1188 (2012).
13. D. Zou and P. Tan, *Transactions on pattern analysis and machine intelligence* **35**, 354 (2012).
14. F. Li, S. Yang and X. Yi, Corb-slam: a collaborative visual slam system for multiple robots, in *Int. Conf. on Collaborative Computing: Networking, Applications and Worksharing*, 2017.
15. R. Mur-Artal and J. D. Tardós, *IEEE Transactions on Robotics* **33**, 1255 (2017).
16. P. Schmuck and M. Chli, *Journal of Field Robotics* **36**, 763 (2019).
17. T. Cieslewski, S. Choudhary and D. Scaramuzza, Data-efficient decentralized visual slam, in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
18. X. Chen, H. Lu, J. Xiao and H. Zhang, Distributed monocular multi-robot slam, in *Int. Conf. on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2018.
19. R. Egodagamage and M. Tuceryan, *Computers & Graphics* **71**, 113 (2018).
20. E. Rublee, V. Rabaud, K. Konolige and G. R. Bradski, Orb: An efficient alternative to sift or surf., in *ICCV*, (1)2011.
21. S. Leutenegger, M. Chli and R. Siegwart, Brisk: Binary robust invariant scalable keypoints, in *2011 IEEE international conference on computer vision (ICCV)*, 2011.
22. R. Mur-Artal and J. D. Tardós, Fast relocalisation and loop closing in keyframe-based slam, in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
23. H. Choset and P. Pignon, Coverage path planning: The boustrophedon cellular decomposition, in *Field and service robotics*, 1998.
24. J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf and O. Von Stryk, Comprehensive simulation of quadrotor uavs using ros and gazebo, in *Int. conf. on simulation, modeling, and programming for autonomous robots*, 2012.
25. N. Koenig and A. Howard, Design and use paradigms for gazebo, an open-source multi-robot simulator, in *International Conference on Intelligent Robots and Systems (IROS)*, 2004.
26. B. Abbyasov, R. Lavrenov, A. Zakiev, K. Yakovlev, M. Svinin and E. Magid, Automatic tool for gazebo world construction: from a grayscale image to a 3d solid model, in *International Conference on Robotics and Automation (ICRA)*, pp. 7226-7232, 2020.
27. R. Lavrenov and A. Zakiev, Tool for 3d gazebo map construction from arbitrary images and laser scans, in *Int. Conf. on Developments in eSystems Engineering (DeSE)*, 2017.
28. Gazebo Model Database. `https://github.com/osrf/gazebo_models`.
29. H. Choset, *Annals of mathematics and artificial intelligence* **31**, 113 (2001).
30. R. N. De Carvalho, H. Vidal, P. Vieira and M. Ribeiro, Complete coverage path planning and guidance for cleaning robots, in *International Symposium on Industrial Electronics*, 1997.
31. A. Zelinsky, R. A. Jarvis, J. Byrne and S. Yuta, Planning paths of complete coverage of an unstructured environment by a mobile robot, in *Int. conf. on advanced robotics*, 1993.
32. B. J. Englot and F. S. Hover, Sampling-based coverage path planning for inspection of complex structures, in *International Conference on Automated Planning and Scheduling (ICAPS)*, 2012.
33. R. Lavrenov, E. Magid, F. Matsuno and M. Svinin, Development and implementation of spline-based path planning algorithm in ros/gazebo environment, in *Trudy SPIIRAN*, pp. 57-84, 2019.
34. D. Scaramuzza and F. Fraundorfer, *IEEE robotics & automation magazine* **18**, 80 (2011).
35. J. Sturm, N. Engelhard, F. Endres, W. Burgard and D. Cremers, A benchmark for the evaluation of rgb-d slam systems, in *Int. Conf. on Intelligent Robots and Systems*, 2012.
36. M. Grupp, evo: Python package for the evaluation of odometry and slam. `https://github.com/MichaelGrupp/evo`, (2017).