Stability assessment and performance comparison of VSLAM frameworks on open indoor datasets

Evgenie Lebedev[1], Denis Uchaev[1], Dmitry Uchaev[2]

[1] Laboratory of Intelligent Systems for Processing Spatial Data, Moscow State University of Geodesy and Cartography (MIIGAiK), Moscow 105064, Russia - z.lebedev.1998@yandex.ru

[2] Department of Space Monitoring and Ecology, Moscow State University of Geodesy and Cartography (MIIGAiK), Moscow 105064, Russia - d-uchaev@yandex.ru

Keywords: VSLAM, Indoor, Real-Time, ATE, Visual-Based Navigation

Abstract

This article examines the main directions of development of modern visual navigation technologies and highlights achievements and problems in this area. The article also considers popular VSLAM (Visual Simultaneous Localization and Mapping) frameworks and their main characteristics. Additionally, several indoor datasets are reviewed to highlight the importance of different testing environments when evaluating VSLAM frameworks. In the experimental part of the work, we compared three prominent VSLAM frameworks: ORB-SLAM 3, Basalt, and OpenVSLAM. For experimental study, 20 image sequences from the EuRoC and TUM-VI dataset were used. The study pays special attention to complex cases in indoor navigation, particularly those involving insufficient scene illumination, which poses significant challenges to the accuracy of VSLAM frameworks. The last part of this work provides a comparison of error estimates, including the absolute trajectory error and its variation throughout the entire estimated trajectory.

1. Introduction

The current pace of technology development in the field of unmanned vehicles is extremely high. This is due to the fact that autonomous transport systems have potential applications in many areas of human activity. There is a list of problems associated with the implementation of such technical systems in real business processes. The main one is to ensure guaranteed sufficient accuracy and speed of positioning and navigation of a robotic vehicle.

This problem has two aspects: hardware and algorithmic. From the point of view of engineering, it is necessary to develop and implement high-precision sensors and inertial positioning systems capable of autonomous and uninterrupted operation under various external operating conditions with the least systematic error.

However, organizing a continuous stream of high-quality data is not the ultimate goal, as it must be processed to extract useful information. An important part of this process is the application of sophisticated VSLAM frameworks, which allow drones to determine their location and create maps of the environment in real time. These frameworks must be resistant to dynamic changes in the environment, including meteorological ones, and not depend on the spatial features of the area. As follows from the above, localization and simultaneous mapping frameworks remain an actively researched area, due to the constant development of hardware and software. It is necessary to carry out regular systematization and evaluation of available solutions based on available data sets. Selective testing helps to identify advantages and limitations of specific VSLAM frameworks, which is important for both developers and application specialists seeking to choose the best existing solution.

One of the first publications on the implementation of VSLAM frameworks was (Davison et al., 2007), which presented Mono-SLAM (Figure 1). Mono-SLAM is an indirect framework based on the use a monocular camera and is able to estimate the camera motion using the extended Kalman filter (EKF) algorithm. The main disadvantage of Mono-SLAM is the lack of mechanisms for optimizing the global trajectory of movement.

In 2013, as shown in Figure 1, the SLAM++ system (Salas-Moreno et al., 2013) was presented, which is one of the first frameworks to leverage semantic information. Unlike Mono-SLAM, SLAM++ can work with RGB-D cameras that provide dense depth estimation. SLAM++ employs RGB-D sensor outputs and performs 3D camera pose estimation and tracking to form a pose graph. A pose graph is a graph in which nodes represent poses (positions and orientations of a camera at different time steps), and edges represent spatial constraints/measurements between the poses. The predicted poses are optimized by merging the relative 3D poses obtained from semantic objects in the scene. Thus, it becomes possible not only to detect loops, that is, to return to already known points in space, but also to minimize errors along the entire trajectory. An alternative to using RGB-D cameras is the use of stereo images obtained from two synchronized monocular cameras.

In (Forster et al., 2014), a first semi-direct visual odometry (SVO) algorithm was proposed, which uses two parallel threads: one thread is used to estimate the camera motion (motion estimation thread) and the other is used to map environment (mapping thread). This separation allows fast and constant-time tracking in one thread, while the second thread extends the map. Motion estimation thread implements the proposed semi-direct approach to relative-pose estimation, which eliminates the need of costly feature extraction and robust matching techniques for motion estimation. The SVO approach allows the use of monocular and stereo cameras. The main disadvantage of SVO is a short-term data association and the inability to perform loop closure detection and, as a result, the lack of global optimization.

In (Mur-Artal et al., 2015), ORB-SLAM was presented, which is a feature-based SLAM system, where specific ORB (Oriented FAST and Rotated BRIEF) features are used for solution of all SLAM tasks: tracking, mapping, relocalization, and loop closing. ORB features provide real-time performance without GPUs, and exhibit good invariance to changes in viewpoint and illumination.

Publisher's note: Copernicus Publications has not received any payments from Russian or Belarusian institutions for this paper.

This contribution has been peer-reviewed.
https://doi.org/10.5194/isprs-archives-XLVIII-2-W5-2024-101-2024

In 2017, ORB-SLAM 2 was released, which includes stereo mode support and loop closure tracking. The problem with ORB-SLAM 2 is the instability of the framework in the case of significantly darkened frames, texture-less frames, and frames with repeating patterns.

The most recent version to date is ORB-SLAM 3, proposed in 2021 (Campos et al., 2021). It added support for RGB-D cameras, as well as the ability to work with different types of lenses, including support for ultra-wide angle (fisheye) lenses. Moreover, ORB-SLAM 3 became to provide improved pose estimation куыгдеы.

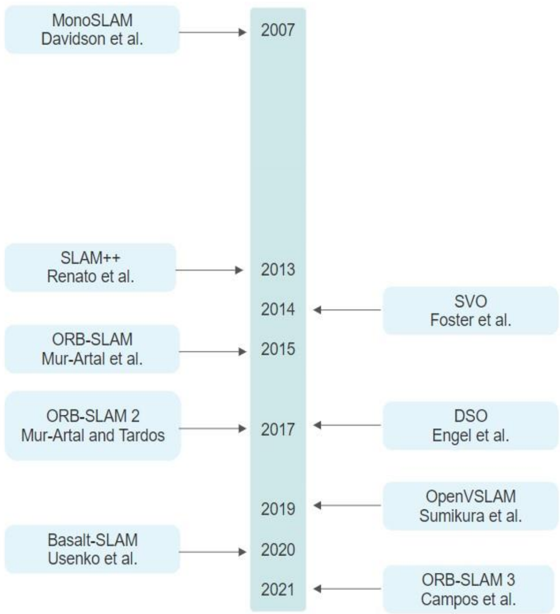

Figure 1. SLAM development history

In 2017, a Direct Sparse Odometry (DSO) framework was proposed, which uses a direct approach and sparse reconstruction to extract the highest intensity points in image blocks (Engel et al., 2018). DSO is based on continuous optimization of the photometric error over a window of recent frames, taking into account a photometrically calibrated model for image formation. Thus, it can be say that DSO can only provide perfect accuracy if photometrically calibrated cameras are used, and does not allow for high-precision results when using regular cameras.

The ORB-SLAM and DSO approaches were combined in the OpenVSLAM framework (Simikura et al., 2019) that supports different types of cameras and lenses (OpenVSLAM can accept images captured with perspective, fisheye, and equirectangular cameras). OpenVSLAM pays special attention to the elaboration of mechanisms for saving and uploading the received maps, which became a reference for other projects.

In (Usenko et al., 2020), it was proposed a novel two-layered visual-inertial mapping approach that integrates keypoint-based bundle-adjustment with inertial and short-term visual tracking through non-linear factor recovery. This approach was realized in the Basalt framework. The combination of cameras and inertial measurement units (IMU) makes the Basalt framework more accurate and robust. In particular, compared to alternative frameworks that use preintegrated IMU measurements between keyframes Basalt shows better pose estimates.

All above-mentioned frameworks are actively supported by their developers and are updated. However, ORB-SLAM 3, OpenVSLAM and Basalt are currently of the greatest interest for various research purposes, because they are the most promising and have been cited more frequently in recent year publications.

In this article, ORB-SLAM 3, OpenVSLAM and Basalt frameworks are subjected to an experimental study to identify which framework is more suitable for indoor navigation. This study pays special attention to complex cases in indoor navigation, especially those related to insufficient scene illumination, which poses significant challenges to the accuracy of VSLAM frameworks. To perform the experimental study of the frameworks and verify their claimed characteristics, the publicly available EuRoC MAV and TUM-VI indoor datasets were used, which are relevant and satisfy various shooting scenarios.

2. Data and methods

The classic formulation of the SLAM (Simultaneous Localization and Mapping) problem is to determine the position of a robotic system in the external environment based on sensor data measuring speed and direction of movement. If cameras are used as input devices, they talk about visual methods of simultaneous localization and mapping (VSLAM).

The number of cameras and their type may vary, which significantly affects the data processing methods. The following VSLAM operating modes are distinguished:

- Monocular mode (only one camera is used).
- Stereo mode (two synchronized monocular cameras are used).
- RGB-D (depth cameras are used to measure the distance to objects, as well as determine their depth).

The use of certain types of sensors significantly affects the final positioning accuracy. Thus, a monocular camera does not require the use of complex pre-calibration algorithms and allows to significantly reduce the cost of the system. However, with this camera, the absolute distance to the objects is unknown. It is necessary to estimate only the relative distance between objects using parallax (distant objects will have a smaller offset in the image than close objects). The trajectory and map obtained by the monocular VSLAM will differ from the real trajectory and map.

In stereo mode, data is obtained from two synchronised monocular cameras located at a known distance from each other, which allows (due to the stereo effect) to calculate the three-dimensional position of each pixel. However, stereo pair matching requires a lot of computing resources.

RGB-D cameras use infrared radiation to determine the distance to an object. Thus, using the physical data of the signal, you can save computing resources compared to stereo cameras. However, such technology is expensive and extremely sensitive to noise and therefore requires precise calibration for specified conditions.

In general, the main stages of data processing by VSLAM frameworks can be distinguished:

1. Receiving sensor data and their preprocessing;
2. Estimating camera movement between adjacent frames and creating an approximate local map;

3. The application of data verification procedures and error minimization. At this stage, data on the camera position at different time points and loop closing data are compared, and then the trajectory and map are calculated;
4. Loop detection. The procedure determines whether the robot has returned to its previous position in order to minimize the accumulated distortion. If a loop is detected, information is transmitted to further optimize the trajectory;
5. Reconstruction of the map of the external environment. A map is being created based on the estimated trajectory of the camera.

2.1 Basic SLAM model

The mathematical model underlying SLAM frameworks is fundamental for enabling an autonomous system to simultaneously determine its own location and map key landmarks within an unfamiliar environment based on sensor measurements. In this scenario, true values of parameters — such as the robot's position and positions of landmarks — are unknown, and measurements obtained from the environment are often subject to noise and errors.

Let's denote the position of the robot at time *t* by x_t. Thus, the sequence of counts, starting from the moment of the beginning and up to the moment *t*, can be written as:

$$X_t = \{x_0, x_1, x_2, \dots, x_t\}\,. \tag{1}$$

The movement parameters between the specified observation points can also be recorded:

$$U_t = \{u_0, u_1, u_2, \dots, u_t\}. \tag{2}$$

However, relying solely on the movement data U_t at time *t* may not be sufficient to accurately estimate the new position of the robotic system. In real-world scenarios, challenges such as wheel slippage, uneven terrain, and other environmental factors can introduce errors into the robot's photometric measurements, leading to inaccuracies in localization. These errors necessitate the continuous integration of data from multiple sensors to ensure accurate localization. The data collected from these various sensors at each time step can be represented as:

$$Z_t = \{z_0, z_1, z_2, \dots, z_t\}. \tag{3}$$

This collection of data forms a statistical dataset that can be used to estimate the state of the system, which includes both the robot's position and the map of the environment. The estimation process can be described by the probability distribution:

$$(x_t, m \,|u_t, z_t), \tag{4}$$

where *P* represents the probability of the robot's position x_t and the map *m*, given the movement data u_t and sensor measurements z_t.

The assessment requires the creation of a motion model and observations that establish a relationship between the position of the x_t autonomous system, u_t odometry and z_t measurements. The observation model and the motion model can be reduced to the form:

$$P(x_t, m \,|x_{t-1}, u_t). \tag{5}$$

On the other hand, the observation model provides information about the robot's surroundings, including the location of landmarks, the parameters of the robot's movement, and the unique identifiers of each landmark, denoted as m. The observation model can be described by the following expression:

$$P(z_t \,|x_t, m). \tag{6}$$

The observation model provides the robot with information about where the landmarks are located, what are the parameters of the robot's movement and the unique identifiers of each landmark, designated m. The observation model described by equation (6) based on statistical data estimation methods is given the form

$$N(h(x_t, m), Q_t)\,, \tag{7}$$

where *N* = normal distribution
Q_t = noise covariance matrix
h = function for estimating the measurement of the current position of the robot.

Then the motion model can be obtained by applying a normal distribution and some function to calculate the new position of the autonomous system:

$$P(x_t \,|x_{t-1}, m) = N(g(x_{t-1}, u_t), R_t), \tag{8}$$

where $g(x_{t-1}, u_t)$ = function for calculating x_t
R_t = noise covariance matrix.

This motion model, as described in equation (8), is quite flexible and does not impose specific constraints on the types of sensors used or the methods for obtaining data. It can accommodate various types of sensors and data acquisition methods, as well as different types of landmarks used to construct the map.

This generality makes the motion model adaptable to different SLAM scenarios and sensor configurations, enabling its application across a wide range of environments and conditions.

2.2 Accuracy Evaluation

The data obtained through SLAM is inherently formed within a specific local coordinate system. Concurrently, the reference coordinates of objects, which are typically obtained from sources external to the SLAM framework, are represented within a different coordinate system. This discrepancy creates a challenge in determining the appropriate transition parameters between these coordinate systems.

To accurately align these two systems and evaluate the precision of the alignment, several key parameters must be determined:

- Translation Vector. This vector describes the shift needed to align the origin of one coordinate system with the other.
- Rotation. The rotational transformation necessary to align the orientations of the two coordinate systems.
- Scaling Coefficient. A factor that adjusts the size of one coordinate system relative to the other.

To compute the rotation, the Kabsch-Umeyama algorithm is frequently employed. This algorithm is particularly useful for calculating the Absolute Trajectory Error (ATE), as it determines the optimal rotation matrix that minimizes the Root Mean Squared Deviation (RMSD) between two sets of corresponding points.

Despite its widespread use in software packages due to its convenience, this method has its limitations. Specifically, it can contribute to an underestimation of the actual error, leading to a situation where the true error might be greater than the computed value. Therefore, it's crucial to not only focus on the overall absolute magnitude of the error along the trajectory but also to assess the range of error, including the minimum and maximum deviations.

In essence, when evaluating SLAM frameworks, it is important to consider the worst-case scenario for point positioning along a trajectory. This worst-case value can serve as a critical basis for assessing the robustness and reliability of specific SLAM frameworks in practical applications.

2.3 Datasets

Despite the increase in the variety of available datasets for SLAM validation, review publications mainly compare SLAMs based on some basic datasets. This is due to the fact that these datasets are checked for compliance with the stated accuracy and metrics calculated on such datasets can be compared.

The criterion for choosing between verified datasets is the type of camera used, the availability of IMU data, and the specific shooting conditions are taken into account. Table 1 shows the most frequently used indoor datasets, which are supported by default by ORB-SRAM 3, OpenVSLAM, Basalt.

It is important to note that dataset data can be represented in different formats and have different metadata descriptions. Each dataset contains several independent shooting image sequences, which are accompanied by trajectory files with high-precision coordinate measurements. The standard term gt (Ground Truth) is used to denote such trajectories.

TUM RGB-D (Sturm et al., 2012) consists of 39 sequences of gray images and depth maps. The RGB-D camera was the Microsoft Xbox Kinect, providing 30 frames per second. The gt data was obtained using a motion capture system with eight high-speed tracking cameras. All image sequences cover the experimental room.

Dataset		TUM RGB-D	EuRoC MAV	TUM VI
Year		2012	2016	2018
Type	indoor	+	+	+
	outdoor	-	-	+
Mode	mono	-	+	+
	stereo	-	+	+
	RGB-D	+	-	-
IMU		+	+	+

Table 1. Dataset overview

The EuRoC MAV (Burri et al., 2016) consists of 11 sets of visual inertial data collected using UAVs. It consists of synchronized stereo images collected by the Aptina MT9V034 sensor at 20 frames per second, backed up by data from the ADIS16448 inertial measurement system. The true values of the labels are provided for each sample of the trajectory. Additionally, the data necessary for camera calibration is provided.

TUM VI (Schubert et al., 2018) represents the most complete set of data is the most comprehensive dataset from the point of view in terms of shooting conditions, since it is carried out both indoors and outdoors (additionally, some data were created obtained in low light conditions for stress testing of frameworks). It includes 28 trajectories captured on two synchronized IDS uEye UI-3241LE-M-GL cameras at 20 frames per second, complemented by inertial measurements with BMI160 3-axis acc/gyro.

3. Experimental Study

In our experimental study, we compared three prominent VSLAM systems: ORB-SLAM 3, Basalt, and OpenVSLAM. For the experimental study, the EuRoC MAV and TUM-VI datasets were selected. These datasets are relevant and satisfy several shooting scenarios, including:

- Navigation of unmanned systems in enclosed spaces (indoor navigation);
- Navigation of unmanned systems in low light conditions;
- Navigation of unmanned systems in the conditions of illumination of the frame.

The VSLAM frameworks were studied under the WSL shell with Ubuntu 22.04.2 LTS installed. The testing was carried out with the following computer parameters: AMD Risen 9 7900x processor, 64 GB RAM, 5200 MHz clock frequency, data was pumped from a hard disk with an interface bandwidth of 5 Gbit/s.

3.1 Metrics

To assess the quality of compared VSLAM frameworks, the ATE metric was used. ATE allows to estimate the global consistency of the estimated SLAM trajectory. To calculate ATE, the estimated and gt trajectories, which are usually specified in arbitrary coordinate frames, are first aligned. This can be achieved using the Umeyama alignment method (Umeyama, 1991), which finds a transformation matrix S. Then the absolute trajectory error matrix at time step $i \in [1, n]$ (n is the number of time steps) is defined as:

$$E_i := Q_i^{-1} S P_i\,, \tag{9}$$

where $Q_i \in SE_3$ = poses from the ground truth trajectory
$P_i \in SE_3$ = poses from the estimated trajectory.

The ATE is defined as the root mean square error from error matrices E_i (Prokhorov et al., 2019):

$$\text{ATE} = \sqrt{\tfrac{1}{n}\sum_{i=1}^{n}\left|\left|trans(E_i)\right|\right|^2}. \tag{10}$$

Thus, ATE (also denoted as ATE_{rmse}) provides the average deviation from ground truth trajectory per frame.

To estimate and visualize ATE evaluation results, EVO package was used (https://michaelgrupp.github.io/evo), which is a Python package for the evaluation of odometry and SLAM. To obtain more reliable estimates, ATE values were obtained by averaging three runs of each VSLAM algorithm.

3.2 Results

The results of the experimental study are presented in Tables 2-4. Figures 2-3 show examples of trajectories recovered by the three compared VSLAM frameworks and superimposed on gt-trajectories.

	Sequence	ORB-SLAM 3	OpenVSLAM	Basalt
EuRoC	MH01	0.035	0.043	**0.027**
	MH03	**0.031**	0.043	0.062
	MH05	0.049	**0.044**	0.145
	V101	**0.038**	0.087	0.043
	V102	**0.036**	0.065	0.045
	V103	0.098	0.081	**0.054**
	V201	0.089	0.061	**0.039**
	V202	0.055	0.061	**0.05**
	V203	0.453	0.257	**0.229**
TUM-VI	Cor 1	**0.012**	0.234	0.184
	Cor 2	**0.016**	0.287	0.369
	Cor 3	**0.010**	0.308	0.373
	Cor 4	**0.173**	0.276	0.198
	Cor 5	**0.011**	0.323	0.373
	room 1	**0.011**	0.096	0.106
	room 2	**0.011**	0.071	0.078
	room 3	**0.009**	0.088	0.128
	room 4	**0.009**	0.034	0.063
	room 5	**0.010**	0.101	0.137
	room 6	**0.005**	0.075	0.029

Table 2. ATE (m)

Table 2 displays ATE in meters, which quantifies the overall discrepancy between the estimated and actual trajectories. Table 3 presents the minimum value of ATE (ATE_{min}) in meters, highlighting the smallest deviation observed in the trajectory estimates. This value represents the best-case performance of the algorithm, where the estimated trajectory most closely aligns with the ground truth. Table 4 shows maximum values of ATE (ATE_{max}) in meters, indicating the largest deviation observed. This metric is crucial for understanding the worst-case scenario in the framework's performance, providing insights into the potential maximum error under certain conditions.

	Sequence	ORB-SLAM 3	OpenVSLAM	Basalt
EuRoC	MH01	**0.001**	**0.001**	0.002
	MH03	**0.001**	0.002	0.009
	MH05	**0.002**	**0.002**	0.052
	V101	**0.004**	0.02	0.006
	V102	**0.001**	0.015	0.01
	V103	**0.005**	**0.005**	0.013
	V201	0.009	0.01	**0.007**
	V202	0.009	0.01	**0.003**
	V203	0.043	**0.028**	0.085
TUM-VI	Cor 1	**0.001**	0.010	0.032
	Cor 2	**0.004**	0.007	0.015
	Cor 3	**0.001**	0.006	0.033
	Cor 4	0.024	**0.005**	0.022
	Cor 5	**0.001**	0.005	0.068
	room 1	**0.001**	**0.001**	0.007
	room 2	**0.001**	0.002	**0.001**
	room 3	**0.001**	**0.001**	0.002
	room 4	**0.001**	**0.001**	**0.001**
	room 5	**0.006**	**0.001**	**0.001**
	room 6	**0.001**	**0.001**	0.002

Table 3. ATE_{min} (m)

	Sequence	ORB-SLAM 3	OpenVSLAM	Basalt
EuRoC	MH01	**0.121**	0.17	0.165
	MH03	**0.096**	0.099	0.126
	MH05	0.16	**0.132**	0.252
	V101	0.106	0.162	**0.084**
	V102	0.368	0.121	**0.073**
	V103	0.374	0.187	**0.11**
	V201	0.278	0.111	**0.072**
	V202	0.498	0.171	**0.102**
	V203	1.535	**0.496**	0.613
TUM-VI	Cor 1	**0.036**	0.905	0.634
	Cor 2	**0.038**	1.143	1.280
	Cor 3	**0.026**	1.370	1.143
	Cor 4	**0.329**	1.232	0.572
	Cor 5	**0.035**	1.408	1.017
	room 1	**0.029**	0.131	0.267
	room 2	**0.027**	0.102	0.202
	room 3	**0.033**	0.156	0.284
	room 4	**0.028**	0.095	0.189
	room 5	**0.029**	0.143	0.332
	room 6	**0.020**	0.109	0.076

Table 4. ATE_{max} (m)

For each of the estimated trajectories, the lowest error estimates (best results) are highlighted with bold font in Tables 2-4.

The analysis of the obtained results allows us to draw the following conclusions:
- on the TUM-VI dataset, ORB-SLAM 3 demonstrates higher positioning accuracy compared to OpenVSLAM and Basalt;
- Basalt is the least sensitive to blur caused by fast camera movement (see sequences V103, V202, V203);
- OpenVSLAM is the least sensitive to a decrease in the scene illumination level (see sequences MH01, MH03 and MH05);
- for sequences characterized by long trajectory length (Cor 1-5 and Room 1-6), the best accuracy is demonstrated by ORB-SLAM 3, and the worst by OpenVSLAM;

	Sequence	ORB-SLAM 3	OpenVSLAM	Basalt
EuRoC	MH01	3638	3682	3682
	MH03	2631	2700	2700
	MH05	2221	2273	2273
	V101	2871	2700	2912
	V102	1671	1692	1710
	V103	2094	1692	2149
	V201	2154	2280	2280
	V202	2309	2348	2348
	V203	1690	1794	1921
TUM-VI	Cor 1	927	988	1039
	Cor 2	1282	1370	1420
	Cor 3	1095	1197	1197
	Cor 4	1045	1050	1052
	Cor 5	776	1250	1703
	room 1	2706	2730	2764
	room 2	2545	2591	2591
	room 3	2511	2511	2511
	room 4	2319	2438	2445
	room 5	2779	2834	2834
	room 6	2552	2591	2591

Table 5. The average number of keypoint pairs

- values of ATE metrics calculated using VSLAM for all sequences are in most cases correlated with the average number of pairs of keypoints (Table 5) recognized in the sequence images (see, for example, sequences Cor 5 and V203).

4. Discussion

Based on the results of the work, it should be noted that the ATE estimates obtained during experimental study were highly dependent on the VSLAM framework used. For these reasons, it weree compared not only ATE estimates, but also the variation of ATE along the entire trajectory. Taking into account the above, the following conclusions were made.

For long trajectories with many loops, as illustrated in Figures 2 and 3, and in good lighting conditions, the Basalt’s optimization method demonstrated superior performance (particularly for the trajectories labeled V101-V203). One of the standout features of Basalt in these scenarios is its minimal variation in the maximum allowable error, indicating consistent accuracy even in complex trajectories. This suggests that Basalt is particularly well-suited for environments where long loop trajectories are common, and stable, precise error margins are critical.

ORB-SRAM 3, as was found, has the best convergence rates according to the ATE_{min} criterion, when Basalt was traditionally inferior in terms of maximum allowable error. Due to this, it is achieved the best ATE results in straight-line driving conditions.

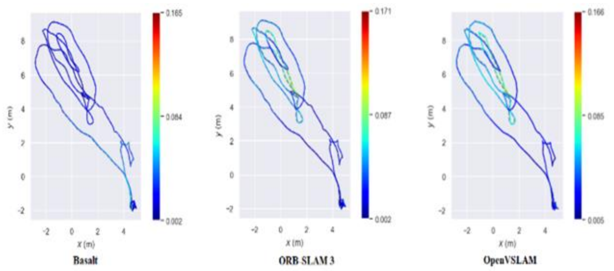

Figure 2. MH01 trajectories recovered by Basalt, ORB-SLAM 3 and OpenVSLAM (gt-trajectories are shown as dotted lines). The degree of warmth and coldness of the trajectory line color represents the ATE value.

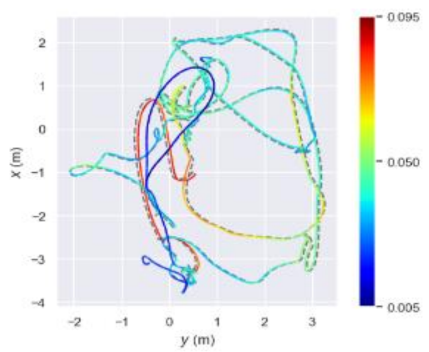

Figure 3. V202 trajectory recovered by Basalt (gt-trajectory is shown as a dotted line). The degree of warmth and coldness of the trajectory line color represents the ATE value.

The performance of OpenVSLAM generally positioned it between Basalt and ORB-SLAM 3 in terms of the ATE metric. Its minimum error values were closer to those of ORB-SLAM 3, while its maximum error tended towards the values observed with Basalt. This intermediate performance suggests that OpenVSLAM provides a balanced trade-off, offering reliable results across a variety of conditions. It may serve as a versatile option when conditions are expected to vary, and neither precision nor speed is the sole priority.

The study found that the number of keypoints detected and matched by the presented VSLAM frameworks did not differ significantly in a statistical sense. This suggests that while the frameworks may vary in terms of trajectory accuracy and error convergence, their ability to identify and match keypoints is relatively consistent. This consistency in keypoint matching indicates that differences in performance are more likely due to the frameworks' processing and optimization techniques rather than their basic feature detection capabilities.

Conclusions

In this article, ORB-SLAM 3, OpenVSLAM and Basalt frameworks were subjected to an experimental study on EuRoC MAV and TUM-VI datasets to identify which framework is more suitable for indoor navigation. As a result of the experimental study, it is established that each of the compared frameworks exhibits unique strengths, depending on the scenario and environmental conditions. ORB-SLAM 3 consistently delivers the highest accuracy, especially in straight-line driving scenarios, as reflected by its superior ATE_{min} values. On the other hand, Basalt demonstrates robustness in long looped trajectories with minimal variation in maximum error, making it particularly well-suited for complex environments. OpenVSLAM provides a balanced performance between the two, offering reliable results across varying conditions.

The study also highlights how specific conditions, such as blur from fast camera movement or changes in lighting, affect each framework differently. Basalt performs better in handling motion blur, while OpenVSLAM proves more resilient to low-light scenarios. Moreover, the correlation between ATE metrics and the average number of keypoint pairs detected suggests that the key differences in performance arise from the frameworks' optimization techniques rather than their ability to detect and match keypoints.

Overall, ORB-SLAM 3 offers the best accuracy, particularly in straight-line conditions, Basalt excels in maintaining consistent performance in complex trajectories, and OpenVSLAM serves as a versatile middle ground with a balanced trade-off between accuracy and robustness.

In summary, the testing showed that each VSLAM framework has distinct strengths, and the choice of framework should depend on the specific requirements of the task. ORB-SLAM 3 excels in accuracy, Basalt in dynamic environments, and OpenVSLAM offers a balanced solution. These findings will help developers and practitioners select the appropriate VSLAM framework based on their specific application needs.

Future testing with more diverse indoor datasets and under different environmental conditions may provide more detailed information on optimal use cases for each framework.

Acknowledgements

The analysis of the main directions of development of modern visual navigation technologies were carried out within the framework of the state assignment No FSFE-2023-0005 of the Ministry of Science and Higher Education of the Russian Federation. The comparison of prominent VSLAM frameworks was carried out within the framework of the state assignment No

FSFE-2022-0003 of the Ministry of Science and Higher Education of the Russian Federation.

References

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M., Siegwart, R., 2016: The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10), 1157-1163. doi.org/10.1177/0278364915620033.

Davison, A., Reid I., Molton, N., Stasse, O., 2007: MonoSLAM: real-time single camera SLAM. *IEEE transactions on pattern analysis and machine intelligence,* 29(6). 1052-1067. doi.org/10.1109/TPAMI.2007.1049.

Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M.M., Tardós, J.D., 2021: ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. *IEEE Transactions on Robotics*, 37(6), 1874-1890. doi.org/10.1109/TRO.2021.3075644.

Engel, J., Koltun, V., Cremers, D., 2018: Direct Sparse Odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3), 611-625. doi.org/10.1109/TPAMI.2017.2658577.

Forster, C., Pizzoli, M., Scaramuzza, D., 2014: SVO: Fast Semi-Direct Monocular Visual Odometry. Proceedings. *IEEE International Conference on Robotics and Automation*, 15-22. doi.org/10.1109/ICRA.2014.6906584.

Mur-Artal, R., Montiel, J., Tardos, J., 2015: ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5), 1147-1163. doi.org/10.1109/TRO.2015.2463671.

Prokhorov, D., Zhukov, D., Barinova, O., Anton, K., Vorontsova, A., 2019: Measuring robustness of Visual SLAM. *IEEE 16th International Conference on Machine Vision Applications (MVA),* Tokyo, Japan, 1–6. doi.org/10.23919/MVA.2019.8758020.

Salas-Moreno, R. F., Newcombe, R. A., Hauke Strasdat, Paul H.J. Kelly, Andrew J. Davison., 2013: SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. Proceedings. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 1352-1359. doi.org/10.1109/CVPR.2013.178.

Schubert, D., Goll T., Demmel, N., Usenko, V., Stückler, J., Cremers D., 2018: The TUM VI Benchmark for Evaluating Visual-Inertial Odometry. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1680-1687. doi.org/10.48550/arXiv.1804.06120.

Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D., 2012: A benchmark for the evaluation of RGB-D SLAM systems. *IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve*, Portugal, 573-580. doi.org/10.1109/IROS.2012.6385773.

Sumikura, S., Shibuya, M., Sakurada, K., 2019: OpenVSLAM: A Versatile Visual SLAM Framework. *27th ACM International Conference on Multimedia*, New York, NY, USA, 2292–2295. doi.org/10.1145/3343031.3350539.

Umeyama, S., 1991: Least-squares estimation of transformation parameters between two point patterns. *IEEE transactions on pattern analysis and machine intelligence,* 13(4), 376-380. doi.org/10.1109/34.88573.

Usenko, V., Demmel, N., Schubert, D., Stueckler J., Cremers, D., 2020: Visual-Inertial Mapping with Non-Linear Factor Recovery. *IEEE Robotics and Automation Letters*, 5, 422-429. doi.org/10.1109/LRA.2019.2961227.