Original Article
International Journal of Fuzzy Logic and Intelligent Systems
Vol. 19, No. 2, June 2019, pp. 119-128
http://doi.org/10.5391/IJFIS.2019.19.2.119

ISSN(Print) 1598-2645
ISSN(Online) 2093-744X

An Open-Source Benchmark for Scale-Aware Visual Odometry Algorithms

Hyukdoo Choi
Department of Electronics and Information Engineering, Soonchunhyang University, Asan, Korea

Abstract

This paper introduces an open-source benchmark for scale-aware visual odometry (SAVO) algorithms, such as stereo visual odometry (VO) and visual-inertial odometry (VIO). The latest open-source VO algorithms are collected and evaluated with the EuRoC MAV and TUM VI datasets. Although there have been a number of benchmarks for VO, we were the first to make the evaluation system containing algorithm sources publicly available. The algorithms are ORB SLAM2 with stereo inputs, ROVIOLI, VINS-fusion, and SVO2, and the latter two algorithms have variations with different sensor configurations. The evaluation results suggest that ORB-SLAM2 makes the best tracking performance with smooth motion, ROVIOLI is robust to highly dynamic motions, and VINS-fusion and SVO2 have the merits of short processing time. Our benchmark system is available at: https://github.com/goodgodgd/docker-vo-bench.

Keywords: Visual odometry, Visual inertial odometry, Open-source, VO benchmark

1. Introduction

Visual odometry (VO) is a key technology for localization systems, which is utilized by various applications. Navigation systems of robots, drones, and vehicles, as well as augmented and virtual reality, depend on visual odometry. Since MonoSLAM [1] and PTAM [2] opened the deep potential of visual odometry and visual simultaneous localization and mapping (vSLAM), huge progress has been made in this field, with advances in both accuracy and frame rate [3]. As a number of novel algorithms and their modifications have been proposed, we need a thorough benchmark to compare the state-of-the-art algorithms. A benchmark is useful, since it presents a starting point or a target for subsequent researches. This paper introduces a benchmark system, and presents the evaluation results of scale-aware visual odometry (SAVO) algorithms. The main feature is that we provide the complete benchmark system as an open-source for anyone to be able to easily compare algorithms by themselves. Therefore, a new algorithm can be compared with the existing ones in our benchmark. The following subsections will briefly review the visual odometry algorithms and previous benchmarks, and explain the difference between our benchmark and the previous ones.

1.1 Visual Odometry Algorithms

VO algorithms have been developed independently, or as a module of vSLAM systems. Since vSLAM is roughly said to be VO with loop closure, we have to review vSLAM, as well as VO. The initial successful vSLAMs, such as ORB-SLAM [4] and LSD-SLAM [5] used only

Received: May 21, 2019
Revised : Jun. 8, 2019
Accepted: Jun. 24, 2019

Correspondence to: Hyukdoo Choi
(hyukdoo.choi@sch.ac.kr)

monocular camera. This simple sensor configuration was soon diverged to stereo [6–9], RGB-D [6], and visual-inertial composition [10–13]. The complex sensor configuration enabled algorithms to be aware of physical scale, and achieve higher accuracy. Monocular VO algorithms look simple and fancy, but due to their scale ignorance and relative scale drifts, their industrial applications are highly limited. In addition, the evaluation method for monocular VOs seems weird, as the ground truth poses are used to rescale estimated poses. This is the reason that in this paper, we only compare scale-aware algorithms. There are various sensor configurations that are aware of absolute scale, as follow:

1) Stereo camera: A stereo camera triangulates a 3-D point in a single frame using a fixed baseline. Various algorithms has been proposed in this configuration [6–9], and they outperformed monocular approaches.
2) RGB-D camera: An RGB-D camera directly outputs the physical scale of all pixels. Most of the RGB-D approaches have focused on 3-D reconstruction to fully utilize plentiful depth information [14, 15]. Since our benchmark focuses on localization accuracy and frame rate, and the VO datasets normally do not provide depth maps, RGB-D approaches are not compared in our benchmark.
3) Monocular visual-inertial odometry (MVIO): A monocular camera with an IMU is the latest trend in VO. It is more robust to fast rotation and textureless environment than camera-only approaches [11–13, 16, 17].
4) Stereo visual-inertial odometry (SVIO): Some of the MVIO algorithms also support stereo cameras [13, 16]. We found the benefits of the additional camera in the experimental results.

1.2 Previous Benchmarks

Evaluating localization algorithms is difficult, since it requires an exact ground-truth of 6-DoF poses of a moving camera or body. An ideal benchmark has to provide sensor data with the ground-truth poses of all frames and evaluation tools. Only after the KITTI odometry dataset [18] and RGB-D SLAM dataset [19] were made public in 2012 did it become possible to evaluate VO algorithms exactly and objectively. ORB-SLAM [4] and many other studies relied on these datasets to objectively evaluate their algorithms.

As VO algorithms diverged to various sensor configurations and challenges to harder conditions, more datasets emerged. Monocular VO dataset [20] does not contain ground-truth poses, but the sequences always finish at the starting position, allowing the evaluation of accumulated trajectory errors. The most popularly used dataset for the recent VO works is the EuRoC MAV dataset [21]. This dataset includes stereo images, IMU data, and the full ground-truth trajectory from the Vicon motion capture system. It was the only dataset that could evaluate visual-initial odometry (VIO) algorithms before the latest TUM Visual-Inertial (VI) dataset [22]. The TUM VI dataset provides more and longer sequences in various environments. Our benchmark evaluated algorithms with the EuRoC MAV and TUM VI datasets. Section 3 describes the details of the datasets.

Comparison of the performances of the VO algorithms was tried in each algorithm's paper, but most of them compared their algorithm only with ORB-SLAM or ORB-SLAM2, which comparison therefore did not reflect the latest progress [13, 16, 17]. This paper tries to evaluate the latest algorithms with the latest datasets, and show the results with comprehensive plots. The most similar work to our benchmark is the VIO benchmark [23]. The latest VIO algorithms at that time and the composition of VIO and global optimization algorithms were evaluated with the EuRoC MAV dataset. Since the main target of the benchmark was a flying robot, not only trajectory accuracy but also processing time and CPU and memory load were evaluated in various single-board computers. Although the benchmark provided extensive evaluation results, the evaluation system is not publicly available. Our benchmark has several differences, and is advanced from [23], which are described in the next subsection.

1.3 Contributions

Our SAVO benchmark aims at an open-source benchmark, where the state-of-the-art algorithms are evaluated with the latest datasets. Our contributions are five-fold:

1) Open-source: whenever we look at decent evaluation results in papers, we are curious about the detailed configuration and implementation, not reported in the papers, to reproduce the same results. Some of them have made their implementations public, but collecting each one of them and making them run properly in a single framework is hard work. We provide a public repository that contains implementations of the latest algorithms, and a docker image to create a clean and isolated environment to build and run them. With the docker image and a number of utility scripts, algorithms can readily be built and evaluated. We mainly use open-source algorithms, but the source codes of SVO2 are not publicly

available, and should be obtained by request of the authors.

2) Latest algorithms: The algorithms evaluated in the VIO benchmark were recently upgraded. ROVIO [12] was upgraded to ROVIOLI [17], and VINS-mono [13] to VINS-fusion. We evaluate only the latest algorithms published after 2017.
3) Latest datasets: Our evaluation is based on both the popular EuRoC MAV, and the latest TUM VI dataset. The TUM VI dataset is superior to the EuRoC MAV in both quality and quantity, and is more challenging, due to its highly dynamic motion. Though the TUM VI dataset paper provided evaluation results of some major VO algorithms, ours evaluated more algorithms. For the sake of simplicity, we term the datasets just EuRoC and TUM VI, respectively.
4) Various sensor configurations: While [23] evaluated only MVIO algorithms, our benchmark evaluates all the scale-aware sensor configurations, including MVIO, SVIO, and Stereo, except for RGB-D. In this benchmark, localization performances can be compared, depending on sensor configurations, as well as algorithms.
5) Comprehensive results: The standard evaluation metrics of VO are absolute trajectory error (ATE) and relative pose error (RPE). We evaluate both the metrics for all combinations of sequences algorithms five times each. Inspired by DSO [24], the error results are sorted in an ascending way, and drawn as a graph. This clearly shows which algorithm has lower or larger error, and how many sequences each algorithm succeeds in tracking.

The rest of this paper is organized as follows: Section 2 introduces the philosophy and implementation of the benchmark system. Section 3 presents the evaluation results for each dataset. Then the last section concludes the paper.

2. SAVO Benchmark

SAVO benchmark is an open-source software that has three components. First, the Docker system is adopted to enable anyone to reproduce the same benchmark system. The second is the algorithm sources, which are modified to output pose results in the same format. The last component is the evaluation system that measures our evaluation metrics from the results. The following subsections present their details.

2.1 Docker-Based System

There have been several VO or vSLAM benchmark papers [19, 20, 22, 23] other than the dataset papers, but none of them are open-source, and hence reproducible. Although the evaluated algorithms are also open-source, building the sources requires many dependencies, and some dependencies of the algorithms conflict with each other, or with the current system. For example, most of the VO algorithms depend on the specific version of ROS, but it requires the specific OS version. In addition, installing ROS affects the whole system, so it can affect the dependency chains of other projects. Therefore, we needed an isolated OS dedicated for this benchmark. By using Docker (https://www.docker.com/), this requirement is easily met with the least burden to the desktop system. We define a docker image that includes all the dependencies for the evaluated algorithms, and that even enables graphical viewers. The benchmark system can be reproduced in any system where Docker is installed. Since our system provides scripts to automate the building process, just executing a few scripts finishes the setup procedure. Once the setup is done, the algorithms can be run and evaluated.

2.2 Evaluated Algorithms

SAVO benchmark evaluated the state-of-the-art algorithms. While some algorithms are committed to a single sensor configuration, others support multiple configurations. Although we selected four algorithms, there are eight configurations in total, as follows: ROVIOLI [17] (MVIO), ORB-SLAM2 [6] (Stereo), VINS-fusion [13, 25, 26] (Stereo, MVIO, SVIO), and SVO2 [16] (Stereo, MVIO, SVIO). The core principal of the algorithms are briefly introduced here:

- ORB-SLAM2 extended the famous ORB-SLAM [4] to stereo and RGB-D cameras. Stereo measurements are represented by stereo pixels, and 3D landmark points in the local map points are optimized to reduce the reprojection error to stereo measurements by bundle adjustment. ORB-SLAM2 here is a VO version, which disabled the loop detection function. Only the stereo configuration is evaluated in this benchmark.
- ROVIOLI is the online front-end part of the MATLAB [17] system. ROVIOLI relies on ROVIO [12] as a basic MVIO algorithm, and is integrated with the global map builder and the frame localization module. ROVIO adopted an EKF-based direct method, where the innovation vector is computed from image intensities, not from feature positions. When it starts in an unknown environment, like our test, it is localized by ROVIO, while independently building a global map for later use.

- SVO2 is a successor of SVO, which is a semi-direct visual odometry algorithm. In SVO, features are extracted only at keyframes but the correspondences are found by direct motion estimation, rather than by descriptor matching. Then feature locations are refined, and camera poses and 3-D map points are optimized by bundle adjustment using the refined feature locations. SVO2 extended SVO to utilize multiple cameras and motion priors from IMU.
- VINS-fusion optimizes point reprojection errors and inertial constrains over the time window. Its IMU state vector includes velocity, acceleration, and biases, as well as pose. The full state vector consists of IMU state vectors over the window, extrinsic parameters between a camera and an IMU, and inverse distances of features. The VIO module is seamlessly integrated with the relocalization module to remove the drift error.

2.3 Algorithm Modifications and Settings

To evaluate the pose estimation performance of the algorithms, slight modifications of the algorithms are needed to log output poses into a file. There were four common modifications. First, the resulting pose format is unified to TUM's format [20] with additional frame process time (FPT): ($timestamp$, x, y, z, qx, qy, qz, qw, fpt). The poses are stacked during running a sequence, and then after the sequence is finished, they are dumped into a file.

The second change is to add configuration files for the new TUM VI. While all the algorithms have configuration files for EuRoC in common, they have nothing for TUM VI, since the dataset was released after the algorithms. We copied the EuRoC configuration files, and edited them, referring the calibration results of TUM VI. However, the IMU parameters about noise and bias were not touched, because the IMU calibration results of both datasets are not very different, and the IMU parameters for EuRoC are customized by the authors not using the raw calibration results.

The third one is about the image format of the TUM VI dataset. Since the dataset provides 16-bit gray images, it has to be converted to 8-bit images to extract features, and match them with the built-in parameters in the algorithms. For example, ORB-SLAM2 could not initialize a map with raw 16-bit images, because ORB descriptor distances are far larger than those from 8-bit images. However, a naïve conversion by dividing by 256 is not a good idea. It loses too much photometric resolution, and the resulting images are too dark. The pixel intensities are divided by 180, to keep the photometric resolution in 8-bits as much as is possible.

The last common change is to finish the process automatically. For algorithms to move onto the next sequence automatically, they must be finished at the end of sequences. The whole evaluation process is governed by python scripts that launch algorithms for each sequence multiple times.

There were also algorithm specific modifications and settings. In ORB-SLAM2, the process to undistort omnidirectional camera model was added to handle the TUM VI dataset. The algorithm assumes the horizontally aligned stereo camera, but the TUM VI dataset is a bit misaligned. The row offset parameter was added to correctly find stereo matches. The GUIs to show images and features were enabled, but we did not launch external visualization tools like RViz of ROS. Since the FPTs are measured only in the main processing algorithm, GUI does not affect the timing results. The FPTs of ROVIOLI were not measured, since it has complex internal structure with parallel thread, and we could not measure the processing time from frame input to pose output.

2.4 Evaluation Metrics

VO algorithms are evaluated by pose estimation accuracy. Given the ground-truth trajectory as a sequence of rigid transformations $\{\mathbf{P}_t\}_{t=1,\ldots,N}$, the estimated trajectory $\{\mathbf{Q}_t\}_{t=1,\ldots,N}$ is evaluated by computing difference from the ground truth. For evaluation, poses in the two trajectories are associated by timestamps, and only associated pose pairs are taken into account in evaluation. Our benchmark evaluates error metrics by averaging errors except for the highest 1%, because sometimes outlier poses occur, due to the instability of the state estimators. Some algorithms can detect the wrong state by themselves and relocalize, but a couple of outliers affect the overall results too much. By removing the highest 1%, more genuine performances can be evaluated.

The popular evaluation metrics are ATE and RPE. ATE measures trajectory level error, which is sensitive to rotational errors. It is computed after aligning the estimated trajectory with the ground-truth trajectory, to prevent initial errors from dominating the total error. As only scale-aware algorithms are evaluated, scale adjustment is not required. The trajectory aligning transformation $\mathbf{T}$ and ATE are computed by the following equations:

$$\mathbf{T} = \underset{\mathbf{T}}{\operatorname{argmin}} \sum_{t} \left\| trans(\mathbf{Q}_t^{-1}\mathbf{T}\mathbf{P}_i) \right\|, \tag{1}$$

$$\mathrm{ATE} = \frac{1}{\mathrm{N}} \sum_{\mathrm{t}} \left\| trans(\mathbf{Q}_{\mathrm{t}}^{-1} \mathbf{T} \mathbf{P}_{\mathrm{i}}) \right\| . \quad (2)$$

On the other hand, RPE measures relatively short-term drifts. The relative poses between fixed time interval poses are evaluated. The interval time Δ was selected as 10 seconds, long enough to see tracking drifts. If the interval is too short, like a second, most of the errors are close to zero, except when it suffers from large drifts. In RPE, both rotational error and translational error are evaluated as follows:

$$\mathrm{RPTE} = \frac{1}{\mathrm{N}} \sum_{\mathrm{t}} \left\| trans \left((\mathbf{Q}_{\mathrm{t}}^{-1} \mathbf{Q}_{\mathrm{t}+\Delta})^{-1} (\mathbf{P}_{\mathrm{t}}^{-1} \mathbf{P}_{\mathrm{t}+\Delta}) \right) \right\| , \quad (3)$$

$$\mathrm{RPRE} = \frac{1}{\mathrm{N}} \sum_{\mathrm{t}} angle \left((\mathbf{Q}_{\mathrm{t}}^{-1} \mathbf{Q}_{\mathrm{t}+\Delta})^{-1} (\mathbf{P}_{\mathrm{t}}^{-1} \mathbf{P}_{\mathrm{t}+\Delta}) \right) , \quad (4)$$

where RPTE and RPRE mean the relative pose translational error and the relative pose rotational error, respectively, and the $angle()$ function computes a rotation angle of the given rigid transformation.

3. Evaluation Results

The algorithms introduced in Section 2.2 were evaluated using EuRoC and TUM VI. These two datasets were selected because they have inertial measurements necessary for VIO algorithms, and provide accurate ground-truth poses. While EuRoC provides ground-truth information over whole trajectories, TUM VI supplies the ground truth only in the room where every sequence starts and ends. Comparing the datasets, EuRoC has been extensively used to evaluate VO and vSLAM algorithms for the last few years. TUM VI is relatively new, hence less used, but presents more and longer sequences, and the sequence are more challenging, due to its highly dynamic motion. The main results of this benchmark are evaluation results of the VO algorithms from the two datasets. EuRoC has images with the resolution of 752×480. TUM VI provides two-level resolutions of 1024×1024 and 512×512. The lower resolution was used in the evaluation for fast processing.

The evaluation results are presented as comprehensive figures for intuitive comparison. For simplicity, the algorithm names are abbreviated as follows: ORB SLAM2 with stereo input and no loop detection is denoted by *orb2_vo_stereo*, and ROVIOLI with MVIO configuration is termed *rovioli_mvio*. VINS-fusion algorithms with MVIO, SVIO, and Stereo configurations are represented as *vinsfs_mvio, vinsfs_svio*, and *vinsfs_stereo* respectively. SVO2 algorithms with the same configurations are termed *svo2_mvio, svo2_svio,* and *svo2_stereo*, respectively.

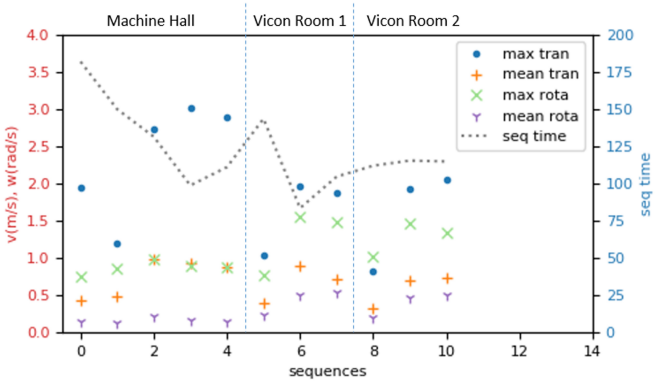

Figure 1. The translational and rotational speeds of the sequences of EuRoC. The mean and maximum values of the speeds are depicted by markers. The translational and rotational speeds were computed in m/s and rad/s, respectively. The dotted line represents the sequence time lengths.

We ran algorithms on the docker image of Ubuntu 16.04 with ROS while the system OS was Ubuntu 18.04. The hardware system included i7-8700 and 16 GB memory.

3.1 EuRoC MAV Dataset

EuRoC has 11 sequences from the three environments: Machine hall, and Vicon rooms 1 and 2. Before presenting the evaluation results, we give the sequence information related to the tracking difficulty. The difficulty of VO or VIO depends on many variables: visual textures of the environment, trajectory length, frame rate, translational and rotational speeds, and so on. Figure 1 shows translational and rotational speeds and sequence time lengths of the 11 sequences, where the speeds are statistically represented by the mean and maximum values. Although the speeds are computed over 0.5 seconds, not per frame, to avoid temporary outliers, the maximum values are far larger than the mean values. The EuRoC paper presented the mean speeds of the sequences, but we consider that the maximum speed is more important, because the tracking algorithms usually get lost at extreme speeds. The sequence time lengths varied moderately, ranging from (83 to 181 seconds).

To evaluate the pose tracking accuracy of the algorithms, each algorithm was executed 5 times for each sequence. The estimated poses were saved in files, and used to compute the evaluation metrics. There are 11 sequences in EuRoC, and hence 55 runs for each algorithm. Figure 2 shows ATEs of the eight algorithms. The 55 ATEs from each algorithm were sorted and marked in the figure. This error figure easily visu-

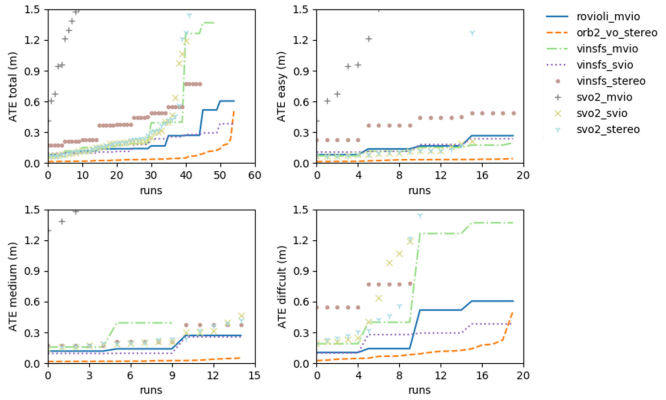

Figure 2. ATE results from EuRoC are drawn in the four SEPs that show the results from total sequences (upper left), *easy* sequences (upper right), *medium* sequences (lower left), and *difficult* sequences (lower right).

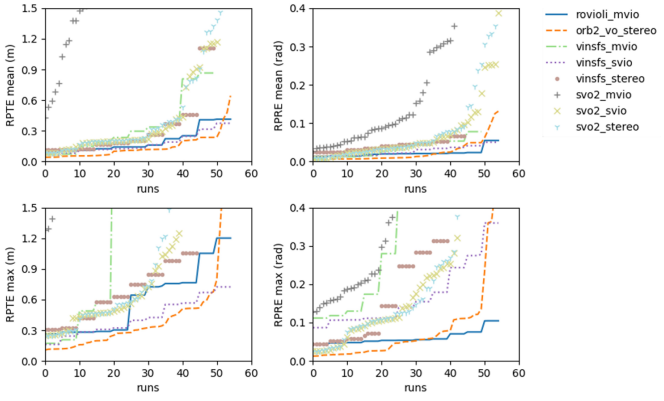

Figure 3. RPE results from EuRoC are drawn in the four SEPs. The left column shows translational error, while the right shows rotational error. The first and second rows present the mean and maximum errors, respectively.

alizes which algorithm yielded lower errors, and is named as a sorted error plot (SEP). Figure 2 consists of the four SEPs. The upper left one is the total results from all the sequences. The remaining three figures represent the results from subsets of sequences categorized by the difficulty labels: *easy*, *medium*, and *difficult*. There are 4, 3, and 4 sequences labeled as *easy*, *medium*, and *difficult*, respectively. The overall results showed that the oldest orb2_vo_stereo still produced the best accuracy, while svo2_mvio results in the worst performance. Comparing the subfigures, the overall ATEs increased with the difficulty levels. At the *difficult* level, VINS-fusion and SVO2 algorithms failed to track in many runs. Another interesting point is the difference between sensor configurations. VINS-fusion and SVO2 have a family of algorithms with different sensor configurations, respectively. In the VINS-fusion family, the performances of the member algorithms are ordered from better to worse as SVIO $>$ MVIO $>$ Stereo. Since SVIO includes inertial information as well as stereo image stream, it resulted in the most accurate trajectory, based on the rich input information. VINS-fusion seems to be more dependent on inertial information. On the other hand, in the SVO2 family, SVIO and Stereo showed similar performances, but MVIO yielded far worse results. SVO2 seems to rely more on visual information than does VINS-fusion.

Figure 3 shows the RPE results, where subfigures show different metrics of all the runs. RPTE and RPRE are shown in the left and right columns respectively. The upper figures present the mean values, while the lowers present the maximum values. The overall performance rankings are similar to the ATE results. Orb2_vo_stereo, vinsfs_svio, and rovioli_mvio showed the best

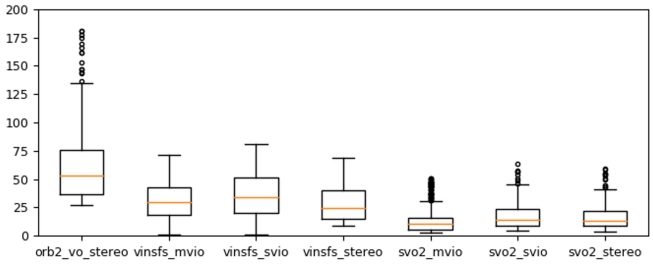

Figure 4. Boxplots of FPTs of the seven algorithms in milliseconds. The FPTs are measured with the *Machine Hall 1* sequence.

local tracking performances. Svo2_mvio resulted in large drifts. The ratio between the mean and maximum is larger in RPRE than in RPTE, which implies tracking rotation is more difficult than tracking translation in highly dynamic motion.

Since in robotic systems, localization solutions usually have to provide the current pose in real time, processing time is an important issue. We measured FPTs only from the *Machine Hall 01* sequence. Each algorithm was executed five times for the sequence, and the timing results were concatenated. Since the larger number of samples is likely to result in larger outliers, only $1,000$ FPTs are evenly sampled, and used to draw a boxplot. The evaluation environment is comprised of i7-8700 CPU and 32 GB RAM. Figure 4 exhibits the FPTs of the eight algorithms in boxplots. ORB SLAM2 took the longest time, 58.08 ms on average, while Svo2_mvio took just 10.41 ms on average. VINS-fusion algorithms resulted in mid-level FPTs without outliers.

Overall, orb2_vo_stereo, vinsfs_svio, and rovioli_mvio showed the best localization results in EuRoC, but their FPT results were larger than those of the others. The SVO2 algorithms were

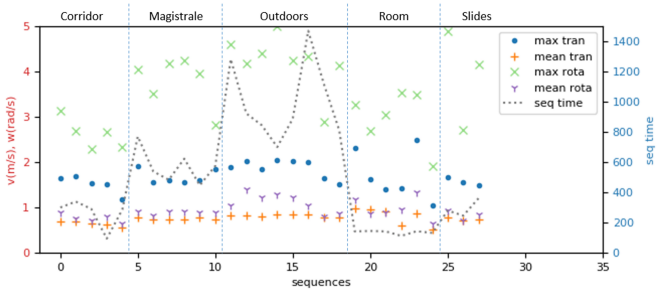

Figure 5. Time lengths and translational and rotational speeds of the TUM VI sequences. Both the mean and maximum speeds are indicated by markers, and time lengths are indicated by a dotted line.

faster than the others, but the tracking performance was slightly worse than the others with Stereo and SVIO configurations.

3.2 TUM VI Dataset

TUM VI contains 28 sequences in 5 types of environments: room, corridor, magistrale (large hall), outdoors, and slides. While the images of EuRoC were captured by a camera on an MAV, TUM VI used a handheld camera. TUM VI sequences have extremely dynamic motions by a swinging arm, which makes VO difficult. Figure 5 shows translational and rotation speeds, and sequence time length. Compared to the EuRoC results in Figure 1, the translational speeds of TUM VI are similar to or smaller than that of EuRoC, but the rotational speeds of TUM VI are overwhelmingly faster than EuRoC in both mean and maximum values. Sequence time lengths vary a lot depending on environment types. The shortest sequence is 'corridor4' at 93 seconds, and the longest is 'outdoors6' at 1,469 s. Due to the higher difficulty of the sequences, the algorithms frequently failed to track sequences.

Figure 6 presents the ATE results from TUM VI. The overall ATEs are large, and many algorithms disappeared or diverged soon in the figure. The algorithms initialized and started tracking at different frames, and might fail at any frame. Therefore, every resulting pose sequence has a different time range. Algorithms that tracked a sequence for only a short time and failed are not considered reliable, but those pose sequences may result in smaller ATEs. To remove this illusion, we only evaluated the resulting pose sequences that tracked more than half of the ground truth pose sequences. The missing data in Figure 6 is due to diverged error or a short period of tracking time. There are supposed to be the results of a total 140 runs for each algorithm, but most algorithms could not track more than half of the runs, as shown in the upper left graph in Figure 6. The

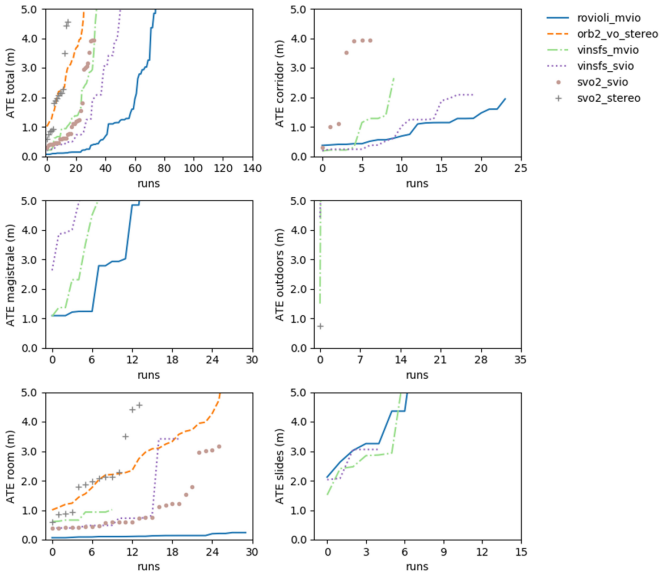

Figure 6. ATE results from TUM VI are depicted in the six SEPs. The upper left SEP shows the total ATEs from all sequences. The remaining five SEPs show ATEs of specific environments, where the environment name is annotated in the vertical axis legends.

most crucial reasons are the highly dynamic motion of the TUM VI sequences, as well as the longer sequence lengths. All the algorithms succeeded in tracking relatively many sequences from the room (lower left) where visual textures were rich, motion was relatively slow, and sequence lengths were short. On the other hand, most of the algorithms failed in the outdoor sequences, where features were far away, motion was fast, and sequence lengths were more than 10 minutes. Comparing the algorithms, only rovioli_mvio showed reliable results in the corridor, room, and slide sequences. Vinsfs_svio showed relatively good results, while the other algorithms failed a lot. Svo2_mvio and vinsfs_stereo even disappeared, because they failed in all the runs.

The RPE results are similar to the ATEs, as shown in Figure 7. Rovioli_mvio presented the best results in both translational and rotation drifts. Vinsfs_svio and svo2_svio look similar in the mean translational error, but vinsfs_svio is better in rotational error, which results in the lower ATEs of vinsfs_svio. Orb2_vo_stereo showed the best performance in EuRoC, but it is the worst among the available algorithms in TUM VI. Another stereo based algorithm, svo2_stereo, also resulted in large errors and many failures. The main reason is that VO algorithms without inertial information are vulnerable to highly dynamic motion, especially fast rotations, and the TUM VI sequences frequently include fast rotations. Inertial information stabilizes

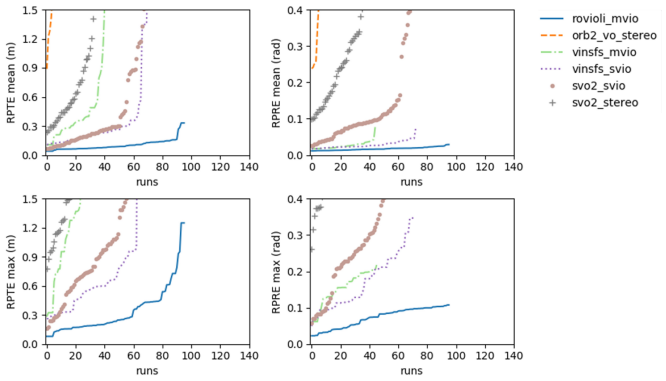

Figure 7. RPE results from TUM VI are presented as the four SEPs. The left column shows RPTEs, while the right shows RPREs. The first and second rows present the mean and maximum errors, respectively.

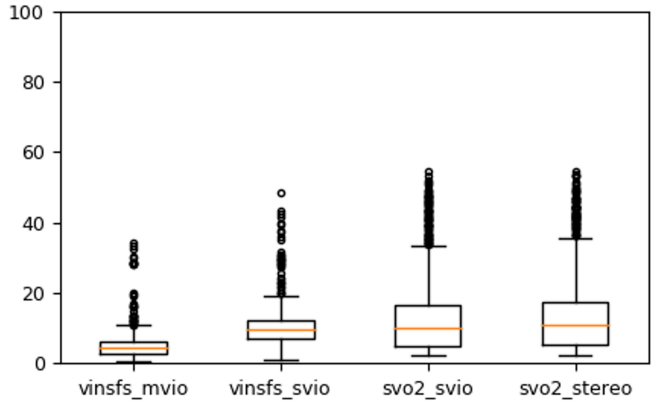

Figure 8. Boxplots of FPTs of the four algorithms in milliseconds. The FPTs are measured with the *Corridor 1* sequence.

the ego-motion tracking performance in difficult conditions, such as fast motion, and textureless or dark environments.

The FPTs of the algorithms are measured only with the *Corridor 1* sequence. Each algorithm was executed five times, and only $1,000$ FPTs were evenly sampled from the entire FPTs. The boxplots of the samples are drawn for the valid algorithms, as shown in Figure 8. ORB SLAM2 could not made valid trajectories, and the FPTs of ROVIOLI were not measurable. Compared with the FPTs from EuRoC, the FPTs of the VINS-fusion algorithms are reduced, while the SVO2 algorithms maintain a similar level to EuRoC. Since the VINS-fusion algorithms are feature-based methods, FPTs depend on environment textures. While the machine hall images of EuRoC are full of textures, the corridor environment is filled with weaker textures, which results in lower processing time. There were several outliers out of Figure 8 that were close to 500 ms in vinsfs_svio, which could have been taken for global graph optimization.

The experimental results indicate that rovioli_mvio is a generally reliable VO algorithm, while when motion is smooth, orb2_vo_stereo is a good option. Dynamicity of motion affected the tracking performances a lot. Stereo only algorithms could not track fast rotations, and even VIO algorithms frequently failed by fast rotation and lack of features. In processing time, orb2_vo_stereo ran at about 20 fps, but with lower computation power, it would be slower. Other algorithms, VINS-fusion and SVO2, run sufficiently at real-time.

4. Conclusions

We introduced the SAVO benchmark system that compares the latest open-source VO algorithms with the latest datasets as an open-source software. The Docker system helped us execute the system in any PC platform, and manage dependencies in an independent environment. This benchmark is publicly available, and readily usable by using a number of scripts to automate the installation and execution process.

The algorithms were evaluated with EuRoC and TUM VI datasets in terms of ATE, RPE, and FPT. ORB SLAM2 with stereo inputs showed the lowest error results in EuRoC, and ROVIOLI was the best in TUM VI. Most of the algorithms have been evaluated with EuRoC, but not with TUM VI. Evaluation with TUM VI revealed that inertial information boosts the robustness of VO algorithms, but VIO algorithms still suffer from highly dynamic motions, despite the wide FoV of TUM VI images. VO and VIO algorithms are considered matured, but there are still problems to solve.

Since this work is open-source, it can be further expanded by anyone to evaluate new algorithms that are not presented in the current benchmark. A new algorithm can be adopted by the adjustment introduced in Section 2.3. Our plan is to update the latest algorithms, for example, Stereo DSO, which has no publicly available source codes by the authors, although there are implementations made by third parties. In addition we are planning to develop a new VO algorithm, and evaluate it in this benchmark.

Acknowledgments

This research was funded by the Soonchunhyang University Research Fund (No. 20180404) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2018R1C1B5086360).

References

[1] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: real-time single camera SLAM," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 29, pp. 1052-1067, 2007. http://doi.org/10.1109/TPAMI.2007.1049

[2] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nara, Japan, 2007, pp. 1-10. http://doi.org/10.1109/ISMAR.2007.4538852

[3] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual SLAM algorithms: a survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, article no. 16, 2017. https://doi.org/10.1186/s41074-017-0027-2

[4] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147-1163, 2015. https://doi.org/10.1109/TRO.2015.2463671

[5] J. Engel, T., Schops, and D. Cremers, "LSD-SLAM: large-scale direct monocular SLAM," in *Computer Vision-ECCV 2014*. Cham: Springer, 2014, pp. 834-849. https://doi.org/10.1007/978-3-319-10605-2_54

[6] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: an open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255-1262, 2017. https://doi.org/10.1109/TRO.2017.2705103

[7] M. Persson, T. Piccini, M. Felsberg, and R. Mester, "Robust stereo visual odometry from monocular techniques," in *Proceedings of 2015 IEEE Intelligent Vehicles Symposium (IV)*, Seoul, Korea, 2015, pp. 686-691. https://doi.org/10.1109/IVS.2015.7225764

[8] R. Wang, M. Schworer, and D. Cremers, "Stereo DSO: large-scale direct sparse visual odometry with stereo cameras," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 2017, pp. 3903-3911. https://doi.org/10.1109/ICCV.2017.421

[9] J. Engel, J. Stuckler, and D. Cremers, "Large-scale direct SLAM with stereo cameras," in *Proceedings of 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015, pp. 1935-1942. https://doi.org/10.1109/IROS.2015.7353631

[10] G. Hartmann, F. Huang, and R. Klette, "Landmark initialization for unscented Kalman filter sensor fusion in monocular camera localization," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 13, no. 1, pp. 1-11, 2013. https://doi.org/10.5391/IJFIS.2013.13.1.1

[11] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314-334, 2015. https://doi.org/10.1177/0278364914554813

[12] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Proceedings of 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015, pp. 298-304. https://doi.org/10.1109/IROS.2015.7353389

[13] T. Qin, P. Li, and S. Shen, "VINS-mono: a robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004-1020, 2018. https://doi.org/10.1109/TRO.2018.2853729

[14] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodge, and A. Fitzgibbon, "Kinectfusion: real-time dense surface mapping and tracking," in *Proceedings of 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, Basel, Switzerland, 2011, pp. 127-136. https://doi.org/10.1109/ISMAR.2011.6092378

[15] M. Zollhofer, P. Stotko, A. Gorlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, "State of the art on 3D reconstruction with RGB-D cameras," *Computer Graphics Forum*, vol. 37, no. 2, pp. 625-652, 2018. https://doi.org/10.1111/cgf.13386

[16] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249-265, 2016. https://doi.org/10.1109/TRO.2016.2623335

[17] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart, "maplab: an open

framework for research in visual-inertial mapping and localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1418-1425, 2018. https://doi.org/10.1109/LRA.2018.2800113

[18] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proceedings of 2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, 2012, pp. 3354-3361. https://doi.org/10.1109/CVPR.2012.6248074

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

[20] J. Engel, V. Usenko, and D. Cremers, "A photometrically calibrated benchmark for monocular visual odometry," 2016, Available: https://arxiv.org/abs/1607.02555

[21] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157-1163, 2016. https://doi.org/10.1177%2F0278364915620033

[22] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stuckler, and D. Cremers, "The TUM VI benchmark for evaluating visual-inertial odometry," in *Proceedings of 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 2018, pp. 1680-1687. https://doi.org/10.1109/IROS.2018.8593419

[23] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *Proceedings of 2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, 2018, pp. 2502-2509. https://doi.org/10.1109/ICRA.2018.8460664

[24] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611-625, 2018. https://doi.org/10.1109/TPAMI.2017.2658577

[25] T. Qin, J. Pan, S. Cao, and S. Shen, "A general optimization-based framework for local odometry estimation with multiple sensors," 2019, Available: https://arxiv.org/abs/1901.03638

[26] T. Qin, S. Cao, J. Pan, and S. Shen, "A general optimization-based framework for global pose estimation with multiple sensors," 2019, Available: https://arxiv.org/abs/1901.03642

Hyukdoo Choi received his B.S. and Ph.D. degrees in electrical and electronic engineering from Yonsei University, Seoul, Korea, in 2009 and 2014, respectively. He is currently an assistant professor of Department of Electronic and Information Engineering in Soonchunhyang University. His main research interests include machine learning, computer vision, simultaneous localization and mapping (SLAM) and deep learning.
E-mail: hyukdoo.choi@sch.ac.kr