


Article

Comparison of Three Off-the-Shelf Visual Odometry Systems

Alexandre Alapetite ^{1,*} , Zhongyu Wang ¹, John Paulin Hansen ², Marcin Zajaczkowski ² and Mikołaj Patalan ²

¹ Alexandra Institute, Njalsgade 76, 2300 Copenhagen S, Denmark; zhongyu.wang@alexandra.dk

² Department of Technology, Management and Economics, Technical University of Denmark, Diplomvej 371, 2800 Kongens Lyngby, Denmark; jpha@dtu.dk (J.P.H.); marcin.zajaczkowski95@gmail.com (M.Z.); patalan.m@gmail.com (M.P.)

* Correspondence: alexandre.alapetite@alexandra.dk

Received: 11 May 2020; Accepted: 13 July 2020; Published: 21 July 2020



Abstract: Positioning is an essential aspect of robot navigation, and visual odometry an important technique for continuous updating the internal information about robot position, especially indoors without GPS (Global Positioning System). Visual odometry is using one or more cameras to find visual clues and estimate robot movements in 3D relatively. Recent progress has been made, especially with fully integrated systems such as the RealSense T265 from Intel, which is the focus of this article. We compare between each other three visual odometry systems (and one wheel odometry, as a known baseline), on a ground robot. We do so in eight scenarios, varying the speed, the number of visual features, and with or without humans walking in the field of view. We continuously measure the position error in translation and rotation thanks to a ground truth positioning system. Our result shows that all odometry systems are challenged, but in different ways. The RealSense T265 and the ZED Mini have comparable performance, better than our baseline ORB-SLAM2 (mono-lens without inertial measurement unit (IMU)) but not excellent. In conclusion, a single odometry system might still not be sufficient, so using multiple instances and sensor fusion approaches are necessary while waiting for additional research and further improved products.

Keywords: odometry; camera; positioning; navigation; indoor; robot

1. Introduction

Robot localization within its environment is one of the fundamental problems in the field of mobile robotics [1]. One way of tracking this problem is to use vision-based odometry (VO), that is capable of accurately localizing robots' position with low drift over long trajectories even in challenging conditions. Many VO algorithms were developed that are categorized into direct, semi-direct and feature-based on what image information is used in order to estimate egomotion [2]. The hardware setup varies, as camera images can be captured in monocular or stereo vision. Many augmentations of VO are available, which perform sensor fusion of computed egomotion with other sensors that can refine the trajectory such as inertial measurement unit (IMU) [3–5], depth sensors [6,7], and LIDAR (light detection and ranging) [8,9].

An important feature to improve the quality of VO is to use SLAM (simultaneous localization and mapping) [10] supplemented by “loop closure”: this means building a database of images while moving, so that when the robot comes back to an already seen location—i.e., with a view similar enough than one of those in the database—it will relocalize itself, thereby cancelling any drift since the last time the robot was at that same location, which results in a much more robust long-time positioning [11]. We only assess the “localization” part of SLAM in this article, not its “mapping” component.

There are many benchmark comparisons of VO algorithms, usually focusing on one of VO applications. Benchmarks compare various VO algorithms in terms of translation and rotation error, memory usage, computation time and CPU (central processing unit) consumption. Previous research [12] compared monocular visual-inertial odometry for six degrees of freedom (6DoF) trajectories of flying robots. [13] assessed performance of VO algorithms using image and depth sensors (RGB-D) for an application of mobile devices. Prior research [14] evaluated VO techniques in challenging underwater conditions. The KITTI [15] dataset features benchmark for visual odometry where researchers can evaluate their algorithms on 11 unknown beforehand trajectories and the best performing VO in terms of rotation and translation error are listed online. [16] compared filtering-based methods and optimization-based methods of VI-SLAM through experiments, it also foresees the trend of running SLAM system on dedicated hardware, for example Intel RealSense.

The purpose of this research is mainly to assess the accuracy of a new commercial hardware–software technology—the RealSense T265 from Intel—and compare it with a few other alternatives. Our approach is similar to [17] (unpublished at the time of our experiments)—which we will confirm—but in another setup, with more comparisons, and additional constraints on processing power. The conducted evaluation of VO solutions is especially meant for practitioners, serving as a guideline in choosing the right VO solution. ZED Mini and RealSense provide out-of-a-box hardware that contain dedicated software solutions, bringing the best of its hardware–software synergy. The RealSense and ZED Mini performance will be compared to well-established ORB-SLAM2 algorithm. ORB-SLAM2 was chosen for this purpose, since it is one of the most widely used SLAM algorithms for VO purpose, therefore, it is easier for readers to establish common frame of reference regarding accuracy performance. The RealSense, ZED Mini and ORB-SLAM2 are all running SLAM loop closure. Moreover, the VO solutions will be compared to wheel odometry [1] (optical encoders)—again, mainly to provide another well-known reference—and evaluated against ground truth obtained by motion capture system OptiTrack.

2. Materials and Methods

2.1. Test Environment

The experiments were conducted indoors, in a controlled area, on a flat, non-slippery surface. As visible in Figure 1, we used some pieces of dark textile to make the scene more, or less, feature-rich, i.e., to adjust the quantity of visual clues in the robot’s field of view. Indeed, visual odometry systems are especially challenged when facing uniform surfaces such as a long white wall. Another important parameter affecting the quality of visual odometry is whether those visual clues are static (not moving) or whether some of them might be moving (dynamic). In order to compare the robustness of the different odometry systems over moving visual elements, we asked three persons to walk repeatedly along the walls of the experiment area.

It is important to note that in order to ensure that we are testing the different systems in a fair way, all visual odometry systems were running in parallel, meaning that there were exposed to exactly the same environment. We believe this is an interesting approach to compare VO systems, at least when it comes to the data acquisition, and it is similar to the approach used with mobile phones in [18]. In our case, we also ran the computation live (which is needed for RealSense and ZED Mini), which might favor the RealSense solution (which runs on its own hardware) because ZED Mini and ORB-SLAM2 run as software on our computer board.



Figure 1. A photo of the test environment, with the robot, and illustrating how we reduce the number of visible features.

2.2. Ground Truth

An OptiTrack [19] system was used as a ground truth system. It is a motion capture system that is capable of tracking objects with positional error less than 0.3 mm and rotational error less than 0.05° , using seven Prime 13 cameras [20] (cf. Figure 2), which can detect passive markers placed on the tracked object. Five markers placed on the top of the robot were used to track robots 6DoF position. The pivot point was marker location where the final position was calculated for. In the experiment, pivot point was located in the center of the camera, which is was ~ 25 cm in front of the center of the robot.



Figure 2. A photo of one of the seven OptiTrack cameras.

2.3. Robot Setup

The platform on which the tests were performed is a Parallax Arlo [21] Robot Platform System (cf. Figure 1), commercialized by Parallax Inc. [22]. This platform was utilized as the physical framework for visual odometry research. Two standard wheels with motors on the sides of the robot and two castor wheels in front and back, cause the platform to be nonholonomic. The platform has two battery packs (12 V, 7 Ah) connected to DHB-10 Motor Controller and Propeller Activity board. It also supplied the Nvidia Jetson TX2 and Raspberry Pi 3. The Activity board was connected to the Raspberry Pi which delivered the control signals for the motors.

To improve the efficiency of the visual odometry computations, one has divided the vision systems into two independent parts, running in parallel. First one running under the Raspberry Pi 3, having Arlobot's Activity board and Intel RealSense T265 connected to. The other system was running on Nvidia Jetson TX2, which has significantly higher computation possibilities (8 GB of memory, 6 CPU cores, GPU with 256 CUDA cores). The board was powered by the 12 V output from the battery pack mounted on the Arlobot's platform. The Nvidia Jetson had connected only one external camera—ZED Mini delivered by Stereolabs. Apart from holding the ZED Mini computations, this station held the ORB-SLAM2 algorithm on itself as well. ZED Mini took advantage of the board's GPU, while ORB-SLAM2 used mostly a single CPU core.

Both cameras (Intel RealSense T265 and ZED Mini) were mounted on top of each other in the front of the Arlobot platform. The mounting position was shifted from the robot's rotation axis to the front of it by 25 cm.

2.4. Software Setup

The robot software packages operate on ROS [23] (Robot Operating System, from the Open Source Robotics Foundation), more precisely ROS Kinetic under the GNU/Linux distribution Ubuntu 16.04 LTS. One has used modified "ROS packages for ArloBot" on Raspberry Pi to obtain communication with the "Parallax Activity board" [24] (microcontroller) on the robot.

2.5. Intel RealSense Tracking Camera T265

The RealSense T265 camera is a tracking camera that was released by Intel in March 2019 at a price of 199 USD. It includes two fisheye lens sensors as well as an inertial measurement unit (IMU). The visual SLAM algorithm runs directly on built-in Intel Movidius Myriad 2 VPU. This gives very low latency between movement and its reflection in the pose, as well as low power consumption that stays around 1.5 W. Since all the computations are performed in real-time onboard it does not require any computations to be held on the master computer.

2.6. ZED Mini

The ZED Mini [25] is a visual-inertial depth camera, which features dual high-speed 2K image sensors and a 110° field of view. With an eye separation of 63 mm, the camera senses depth from 0.1 m to 12 m with improved accuracy and fewer occlusions in the near range. Using visual-inertial odometry technology, inertial measurements (IMU) are fused at 800 Hz with visual data from the stereo camera. Sensor fusion allows for accurate tracking even when visual odometry gets lost due to insufficient amount of feature matches. The image acquisition was done at a resolution of 720p and a frequency of 20 Hz (the best trade-off we found between blur and resolution). The ZED Mini odometry software was able to process frames in stereo at ~19.5 Hz, taking advantage of the GPU (graphics processing unit) compute capability of the Nvidia Jetson computer board.

2.7. ORB-SLAM2

ORB-SLAM2 is a complete SLAM system [26] for monocular, stereo and RGB-D cameras that achieve state-of-the-art accuracy in many environments (cf. Figure 3). In this study, the monocular setup was used. We chose mono ORB-SLAM2 because stereo ORB-SLAM2 was too computationally heavy for the computer board used in the experiments, resulting in a too low framerate, especially when run in parallel with the other odometries. ORB-SLAM2 was only able to take advantage of a single CPU core, not of the other cores nor of the GPU.

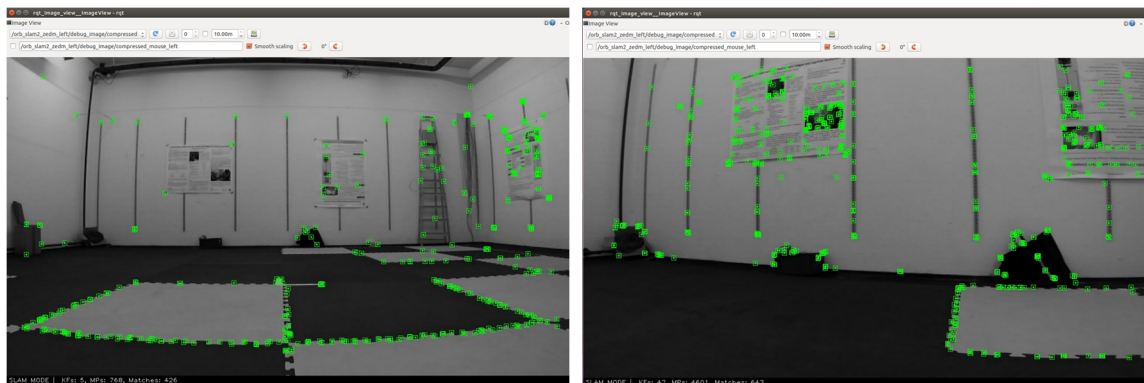


Figure 3. Two screenshots of ORB-SLAM2 representation of the detected features, far from walls (**left**) and close to a wall (**right**), in a scenario with many features (e.g., additional items, posters), visualized via the robot operating system (ROS) tool `rqt_image_view`. The full camera data can be downloaded (cf. Supplementary Materials).

The main three components that are executed in parallel are: (1) tracking thread which estimates pose of current frame and optimizes its position minimizing the reprojection error applying motion-only bundle adjustment (BA); (2) local mapping thread that saves new keyframes, performs local BA and saves visual words for later (Bag of Words) (BoW) place recognition [27]; and (3) the loop closure to detect large loops using BoW approach and refine trajectory first performing pose-graph optimization and lastly performing full BA in order to obtain optimal structure and motion solution.

It is important to note that ORB-SLAM2 does not integrate any IMU by default, and has thus less sensor data to work with than the RealSense T265 and ZED Mini solutions. Furthermore, ORB-SLAM2 appeared very much computationally heavy in our setup, being able to process frames in monocular mode at only ~5.5 Hz (i.e., with many dropped frames). We believe it is still fair to test ORB-SLAM2 this way, as the computer board we used is not low-end for a small robot or drone.

Finally, as ORB-SLAM2 was operating in monocular mode, we did an offline calculation and optimization of the scale factor (cf. Section 3: Data Analysis). We also analyzed the gains when more computing power was available.

2.8. Scenarios

For each scenario, the robot starts by driving forward for three meters. Following this, it makes three full turns plus 180° (1260° in total) around its own spot. The process is repeated four times during each scenario.

We repeated the experiments for three different parameters, giving a total of eight combinations (cf. Table 1):

- Quantity of visual features: We changed the number of visual features in the field of view: either “many” with several paper posters on the walls to increase the number of visual clues, or “few” with mostly grey walls. The floor is unchanged between conditions.
- Robot speed: We made the robot drive at two different reference speeds: either “fast” with 1.07 m/s linear speed for ~ 2.52 rad/s angular speed (when the robot turns), or “slow” with 0.36 m/s linear (i.e., ~ 3 times slower) speed for ~ 0.35 rad/s angular speed (i.e., ~ 7 times slower).
- Moving visual elements: We made the visual environment more, or less stable: either “static” with nothing moving, or “dynamic” with some persons constantly walking along the walls around the room.

Table 1. List of scenarios to test the eight combinations of three conditions.

Quantity of Visual Features	Robot Speed	With Moving Visual Elements (Dynamic) or without (Static)
Many	Slow	Static
Many	Slow	Dynamic
Few	Slow	Static
Few	Slow	Dynamic
Many	Fast	Static
Many	Fast	Dynamic
Few	Fast	Static
Few	Fast	Dynamic

We picked that specific path to fit into the area of our lab covered by the ground truth system, while assessing both translation and rotation. Furthermore, the robot would drive through an already seen path, giving a good chance for the SLAM algorithms to perform relocalization when seeing some known scenes.

3. Data Analysis

The datasets come from three different sources, namely OptiTrack system, Raspberry Pi and Jetson TX2. The first thing to do is to transform the data into same format. Because the robot runs only in a 2D plane, the position of different methods can be transformed into robot position (x , y) and robot orientation θ . Afterwards, the three datasets are synchronized and merged into one. In order to analyze the performance of different visual odometry systems relative to the OptiTrack, some columns of the dataset such as velocity and OptiTrack are interpolated (filled with previous values if the cells are empty) since the OptiTrack data does not come at the same timestamp as the others. Before calculating the errors, the ORB-SLAM2 data is scaled and the scale coefficient is found by gradient decent (i.e., we found the optimal scale factor), using the first part of each scenario (one seventh of the data points). Besides, the robot wheel odometry data also needs to be transformed to the camera center so that all measurements are in the same coordinate system. An example of how the data looks like at this stage can be seen in Figure 4.

The error of the visual odometry system is evaluated as translation error and rotation error, where the translation error is calculated by the distance offset relative to the ground truth and the rotation error is calculated by the angle offset. In addition, the incremental of the errors over time is also computed.

See details in the “Supplementary Materials” section for the source code and the data.

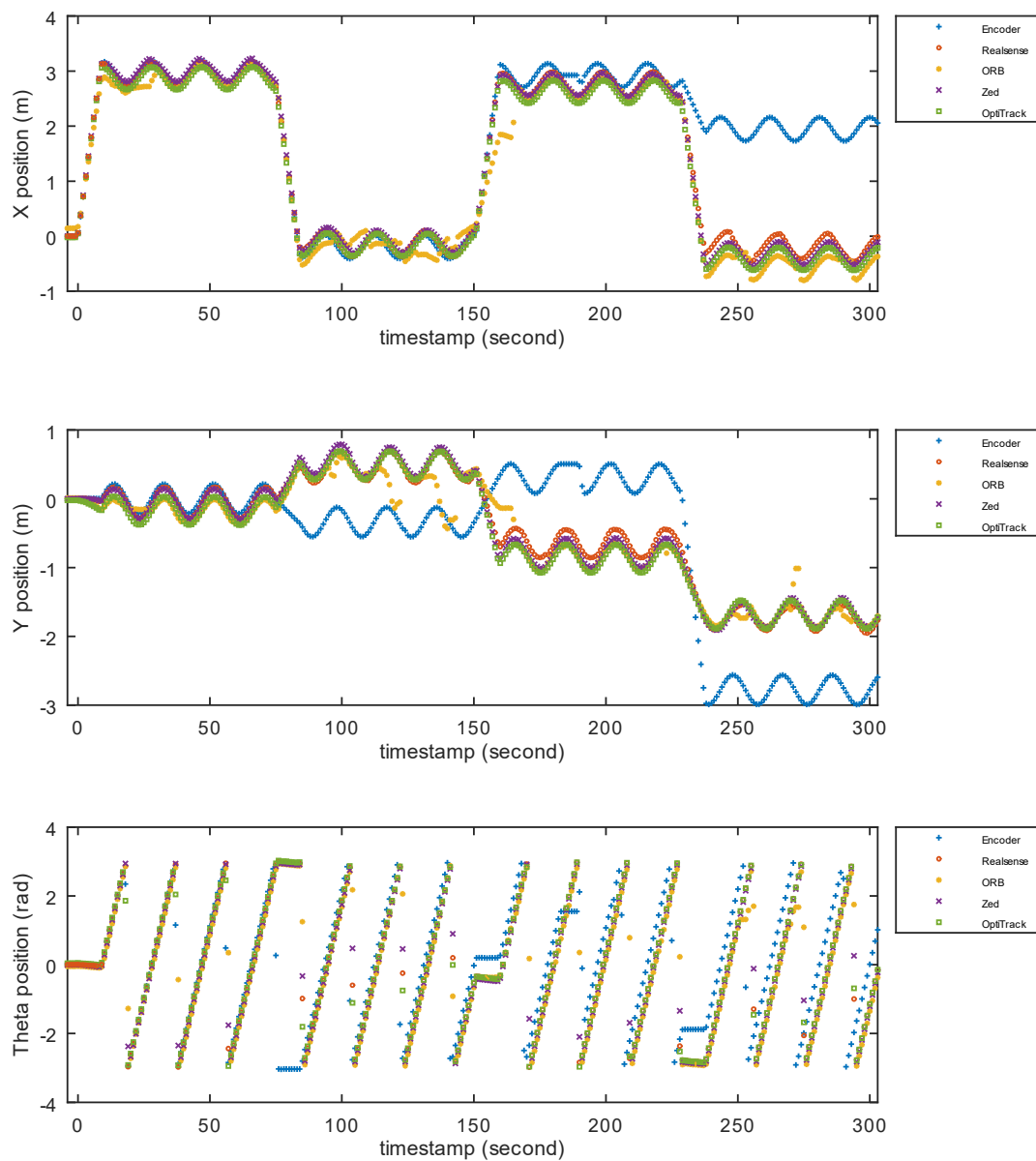


Figure 4. Example of raw data during a single experiment run. The scenario was “many features, slow, static environment). The upper graph reports the x-position of the robot over time (five minutes in this example) according to each odometry, which can be compared with the ground truth (OptiTrack series, green squares). The middle graph if for the y-position of the robot, while the bottom graph is for the rotation (orientation) of the robot.

4. Results

4.1. Descriptive Statistics

In order to get a better understanding of the data, a first round of descriptive statistics is performed. The two most informative visualizations are reported in Figures 5 and 6, respectively for translation error (i.e., robot $\{x, y\}$ position estimation error) and for rotation error (i.e., robot orientation error).

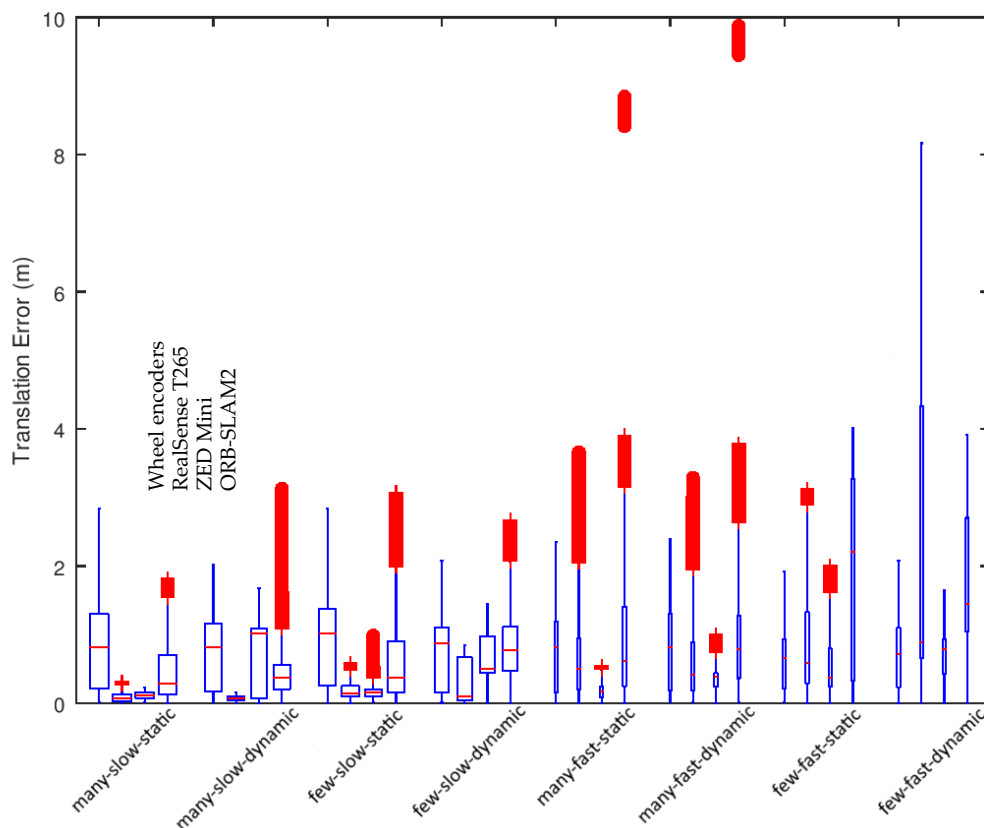


Figure 5. For each of the scenarios (“many slow static”, “many slow dynamic”, “few slow static”, “few slow dynamic”, “many fast static”, “many fast dynamic”, “few fast static”, “few fast dynamic”), we report the median translation error (red horizontal line), the 75% observed translation errors (blue rectangle box), the 95% observed translation errors (blue error lines), as well as outliers (red crosses above the rest). For each scenario, from left to right, are reported the odometries Wheel encoders, RealSense T265, ZED Mini, ORB-SLAM2.

We observe that wheel odometry (based on optical encoders) always provide a poor translation (Figure 5) and rotation (Figure 6) estimation but does so in a quite consistent manner: wheel odometry is indeed not much affected by the scenarios—not even speed—which is not surprising in non-sliding condition. The measurements are more consistent during translations than during rotations.

The scenario with many features, slow speed, and static scene was without surprise the one with the best results for all odometries. Expectedly, we observe that the visual odometry systems are much affected by the challenging scenarios, with sometimes big errors for all of them, especially ORB-SLAM2, and especially during rotations.

For all visual odometries, a typical example of event leading to outliers is when there is a loss of tracking, followed by an accumulation of errors, until a sharp relocalization by loop closure.

As clearly visible in particular on Figure 5, aside for rotating, speed had the greatest detrimental effect on the quality of the visual odometries. The number of visual features had a clear, but lesser impact. Finally, the fact that some visual clues where moving or not in the field of view did not impact the accuracy as much as the other factors (and less than what we were expecting).

Other versions of the above figures are provided in Appendix A (with another visualization, some data smoothing, and more CPU power for ORB-SLAM2).

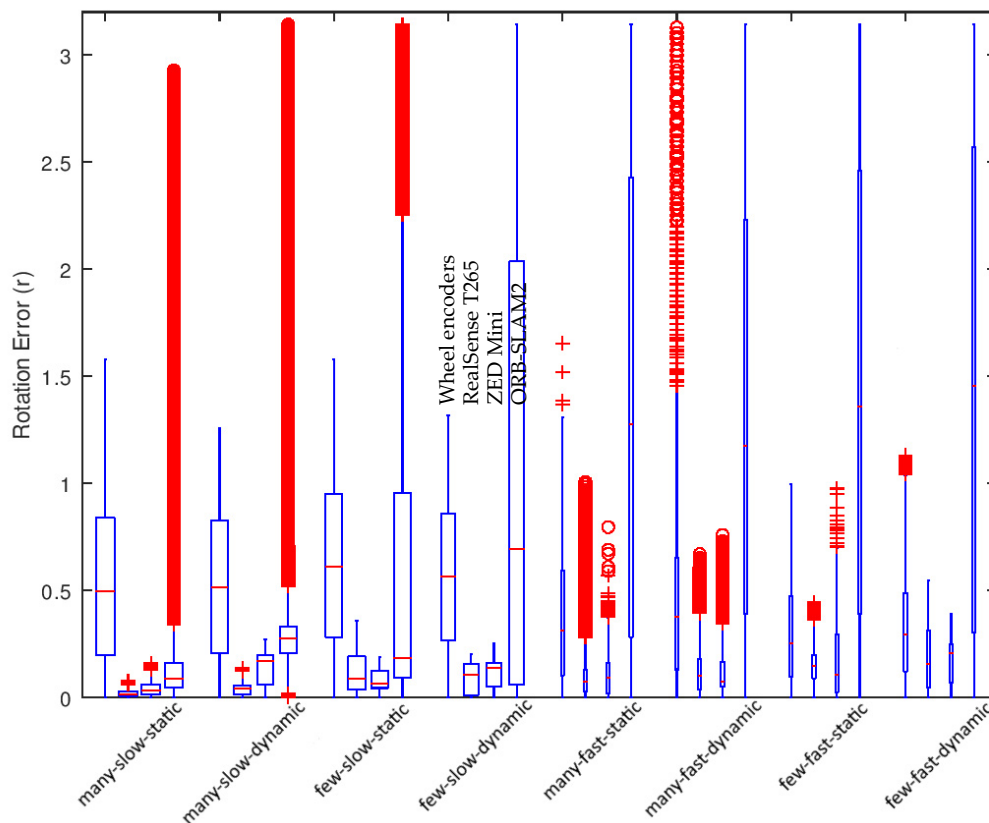


Figure 6. For each of the scenarios (“many slow static”, “many slow dynamic”, “few slow static”, “few slow dynamic”, “many fast static”, “many fast dynamic”, “few fast static”, “few fast dynamic”), we report the median rotation error (red horizontal line), the 75% observed rotation errors (blue rectangle box), the 95% observed rotation errors (blue error lines), as well as outliers (red crosses above the rest). For each scenario, from left to right, are reported the odometries Wheel encoders, RealSense T265, ZED Mini, ORB-SLAM2.

4.2. Statistical Analysis

In order to help identifying relevant differences, we did a light statistical analysis, with a series of *t*-Tests of the type “two-sample assuming unequal variances” from the “Analysis ToolPak” of Excel (Microsoft Office 365 version 1910). Table 2 contains results for the average translation error, while Table 3 contains the average rotation error, across all scenarios.

Table 2. Summary of the *t*-Tests of the average translation error (in meters), between odometries.

		Wheel Encoders	RealSense T265	ZED Mini	ORB-SLAM2
Mean		0.8432 m	0.6706 m	0.4249 m	1.1710 m
Variance		0.0144	1.3356	0.1120	0.3683
Observations		24	24	24	24
<i>p</i> -value	Wheel encoders		2.37×10^{-1}	1.54×10^{-6} **	7.79×10^{-3} **
	RealSense T265			1.63×10^{-1}	3.44×10^{-2} *
	ZED Mini				3.25×10^{-6} **

p-values marked with one asterisk “*” are better than 0.05; two asterisks “**” when better than 0.01.

From Table 2, one can see that there is a significant difference in quality of the positioning between the odometry systems when it comes to translation error, except between wheel odometry and RealSense, and between RealSense and ZED Mini. Likewise from Table 3, one can see that there is no significant difference between the RealSense and the ZED Mini, while other odometries exhibit a significant difference between each-other in terms of rotation error.

Table 3. Summary of the t-Tests of the average rotation error (in radians), between odometries.

	Wheel Encoders	RealSense T265	ZED Mini	ORB-SLAM2
Mean	0.4682 rad	0.1057 rad	0.1061 rad	1.0409 rad
Variance	0.0167	0.0092	0.0060	0.3314
Observations	24	24	24	24
<i>p</i> -value	Wheel encoders	2.80×10^{-14} **	1.63×10^{-14} **	3.52×10^{-5} **
	RealSense T265		4.93×10^{-1}	2.21×10^{-8} **
	ZED Mini			2.05×10^{-8} **

p-values marked with two asterisks “**” when better than 0.01.

As such, the statistical analysis confirmed the main trends observed in the descriptive statistics.

4.3. Main Findings

Wheel odometry is not much affected by the different scenarios, not even by the change of speed, leading to more consistent values, especially during translation. This is not surprising because the floor surface remained identical. However, the standard deviation of wheel odometry is typically higher than for the visual odometries, making it generally less precise, especially during the easy scenarios (i.e., one or more of: low speed, many features, static environment).

But the scenarios do have a significant effect on visual odometries. In our tests, speed had the greatest effect (in the “fast” scenarios, linear speed was ~3 times higher and angular speed ~7 times higher), followed by the number of features, while the static vs. dynamic environment had the smallest effect.

Among the visual odometries, ORB-SLAM2 has the poorer results in our experiments, both in translation ($p < 4 \times 10^{-2}$) and rotation ($p < 4 \times 10^{-5}$), and for all scenarios. This materializes in a higher imprecision, a higher standard deviation, and more outliers than other methods. This is not surprising due to running in monocular mode and without IMU. Without IMU sensor fusion, if the visual tracking gets lost, all the subsequent position estimations will deviate highly until a familiar scene is found with loop closure.

Except for a few outliers, the RealSense T265 and the ZED Mini have comparable results in average ($p > 0.1$), both in terms of translation error and rotation error. The RealSense T265 is a bit more negatively affected by speed than ZED Mini, especially during translation.

Finally, we also tried to post-process the odometry data offline to smooth it and reduce outliers (cf. Figures A2 and A5 in Appendix A), but this did not change the main findings.

5. Discussion

5.1. Camera Lens Types

The RealSense T265 has a wide field of view, making it able to potentially spot many more visual features than the ZED Mini, but the drawback is in principle a poorer image quality. In our experiments, in the end, it did not seem to make a significant difference, although we cannot tell which part of the results is due to the lens and which part is due to a difference of processing. The RealSense T265 would arguably have an advantage in scenarios where the interesting visual features are in the periphery of the camera field of view, i.e., not visible by cameras with narrower field of views.

5.2. Processing Power

On a robot or drone, aspects such as total weight, price, and power consumption are essential factors. On those factors, the RealSense T265 globally wins over the ZED Mini and ORB-SLAM2, as it comes with built-in data processing, while the other visual odometries require an additional powerful computer board such as an NVIDIA Jetson or similar.

Noticeably, the stereo version of ORB-SLAM2 was too computationally heavy for the computer board used in the experiments and could therefore not run in scenarios requiring real-time odometry.

We believe it is fair to include aspects such as processing power requirement when picking an odometry method. It is important to note that the quality of the ORB-SLAM2 odometry would have been a bit higher, had we used a more powerful computer board: ORB-SLAM2 uses a single CPU core fully (the other processes being run on the other CPU cores and not using them fully) and achieved to process frames at only about ~5.5 Hz in average. This slow performance compared to ZED Mini is partially due to the fact that ZED Mini takes advantage of GPU compute capability, which ORB-SLAM2 could not do.

In order to compare with the maximum quality that the baseline ORB-SLAM2 would have achieved with unlimited processing power, we re-ran ORB-SLAM2 offline on the collected data, and found light obvious improvements, but without changing our findings, i.e., a resulting quality still lower than RealSense T265 and ZED Mini (cf. Figures A3 and A6 in Appendix A).

5.3. Multiple Sensors & Sensor Fusion

In our experiments, we compared the different method with one single sensor for each of them, but it would be possible to combine several cameras for potentially better quality. This is especially doable for larger robots.

A similar approach is to combine different types of sensors with a sensor fusion approach. Outside, such a sensor fusion could be done, with e.g., GPS (Global Positioning System), and indoor for instance with fiducial markers such as ArUco markers or other 2D-barcodes.

Noticeably, the RealSense T265 offers a built-in sensor fusion mechanism that can be fed with wheel odometry, but this was outside the scope of those experiments.

5.4. Limitations of Black-Box Systems

While it would be interesting for the academic discussion to be able to tune or disable various internal mechanisms of the RealSense T265 or of the ZED Mini (for instance the SLAM loop closure or IMU), those systems are relatively closed and cannot be inspected in details by third-parties like is possible to do, with e.g., an open source software algorithm. We, therefore, content ourselves with an overall assessment of their relative merits, independently of their internal choices. In particular, this prevents us from comparing ORB-SLAM2 (which does not have any IMU data) with the sole visual odometry methods included in RealSense T265 or of the ZED Mini, which would be fairer for ORB-SLAM2.

5.5. Limitations of the Experimental Design

The project supporting those experiments only had the resources to perform a controlled assessment in a relatively narrow use-case, i.e., short trajectory ground vehicle motion. We have, therefore, neither assessed the relative performance of the various odometry systems in 3D (e.g., drones) nor for longer trajectories. However, our experiments are a subset of the expected motions that can be seen in the more challenging 3D movements and/or longer navigations, and we can therefore expect that the limitations we observed also occur in those phases of the more challenging conditions. Anecdotaly, we did come to similar conclusions when running our ground robots on longer indoor scenarios, but this was outside the lab and thus without a ground truth setup, so we cannot provide good quality data for the longer trajectories.

6. Conclusions

In the specific tested use-case, i.e., short trajectory ground vehicle motion with limited processing power, the experiments show that the Intel RealSense T265 compares well with off-the-shelf state of the art, especially when accounting for price, built-in processing power, and sensor fusion abilities. In our experiments, the ZED Mini and the RealSense T265 provide comparable results. This confirms another recent evaluation of the T265 [17], which also found that its localization is more reliable than the baseline ORB-SLAM2. However, a single RealSense T265 does not solve the visual odometry

challenge fully. Therefore, even for basic indoor navigation needs, several sensors or techniques must be combined. For the time being, visual odometry remains a domain with room for additional research and improvements. In particular, we start observing great advances on visual-inertial odometry from the world of augmented reality [28] with Google ARCore [29], Apple ARKit [18], Microsoft Hololens [30], which comparison with the RealSense T265 is left for a future publication.

Supplementary Materials: The source code of the data analysis, as well as the raw data (in ROS Bag format) for the different odometry systems is available online from: https://github.com/DTUR3/visual_odometry_comparison.

Author Contributions: Conceptualization, A.A. and J.P.H.; data curation, Z.W.; formal analysis, A.A. and Z.W.; funding acquisition, A.A. and J.P.H.; investigation, M.Z. and M.P.; methodology, A.A.; project administration, A.A. and J.P.H.; resources, J.P.H.; software, Z.W, M.Z. and M.P.; supervision, A.A.; validation, A.A.; visualization, A.A. and Z.W.; writing—original draft, A.A., M.Z. and M.P.; writing—review and editing, A.A. and J.P.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by InfinIT (the innovation network for IT in Denmark) and by the Bevica Foundation, Denmark.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

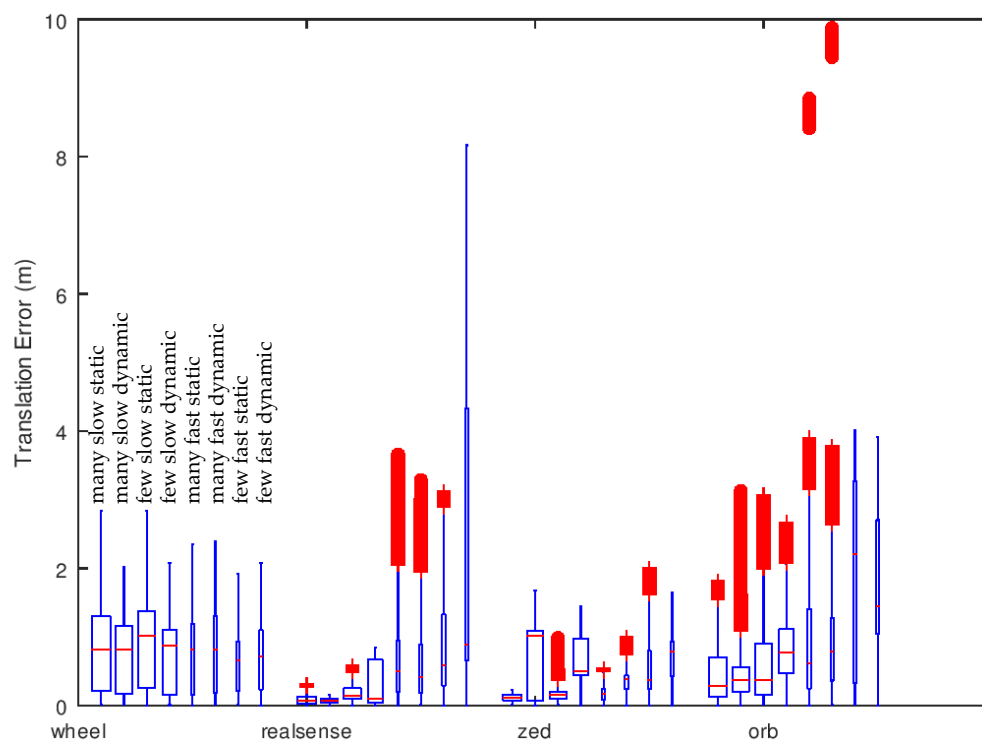


Figure A1. For each of the odometries (Wheel encoders, RealSense T265, ZED Mini, ORB-SLAM2), we report the median translation error (red horizontal line), the 75% observed translation errors (blue rectangle box), the 95% observed translation errors (blue error lines), as well as outliers (red crosses above the rest). For each odometry, from left to right, are reported the scenarios: “many slow static”, “many slow dynamic”, “few slow static”, “few slow dynamic”, “many fast static”, “many fast dynamic”, “few fast static”, “few fast dynamic”.

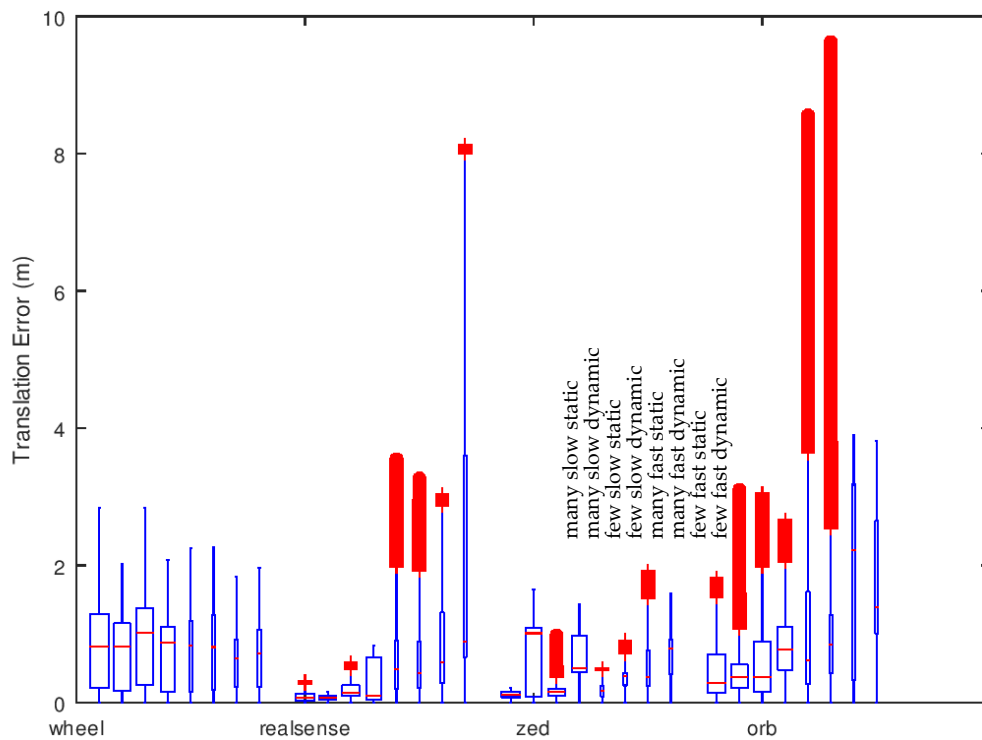


Figure A2. Same as Figure A1, but after soothing the data on 500 points across all sensors, which is about ~1.76 s.

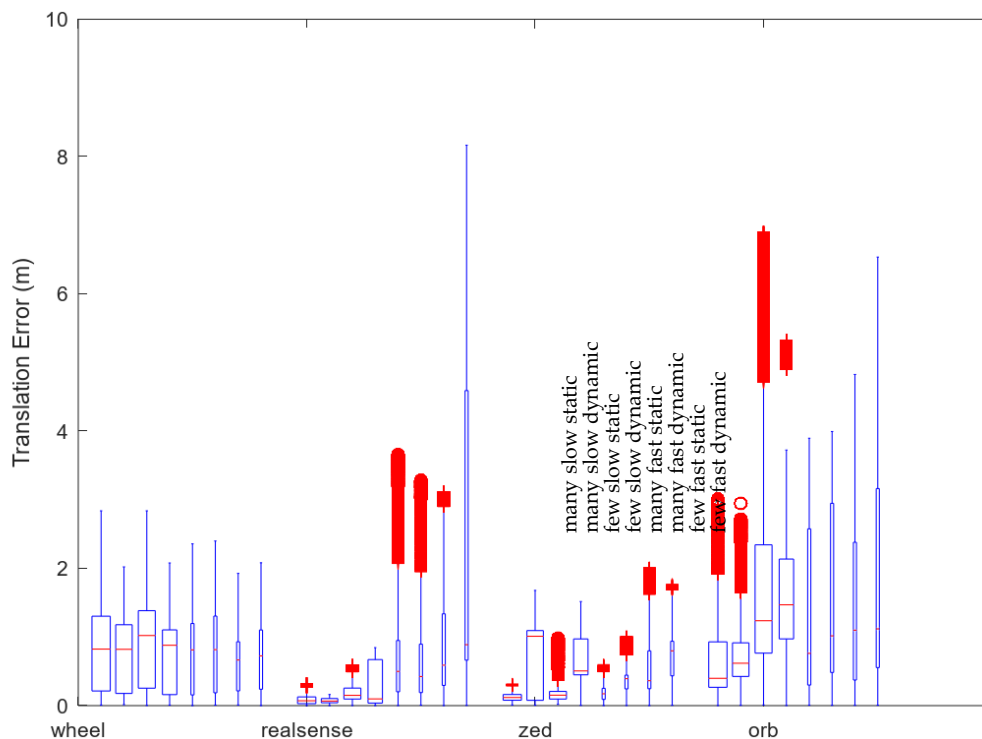


Figure A3. Same as Figure A1, but with ORB-SLAM2 computed offline (i.e., without central processing unit (CPU) processing power limitations) and excluding the first seconds of the experiments used for initialization (so the results for other odometries are not exactly those of Figure A1 either).

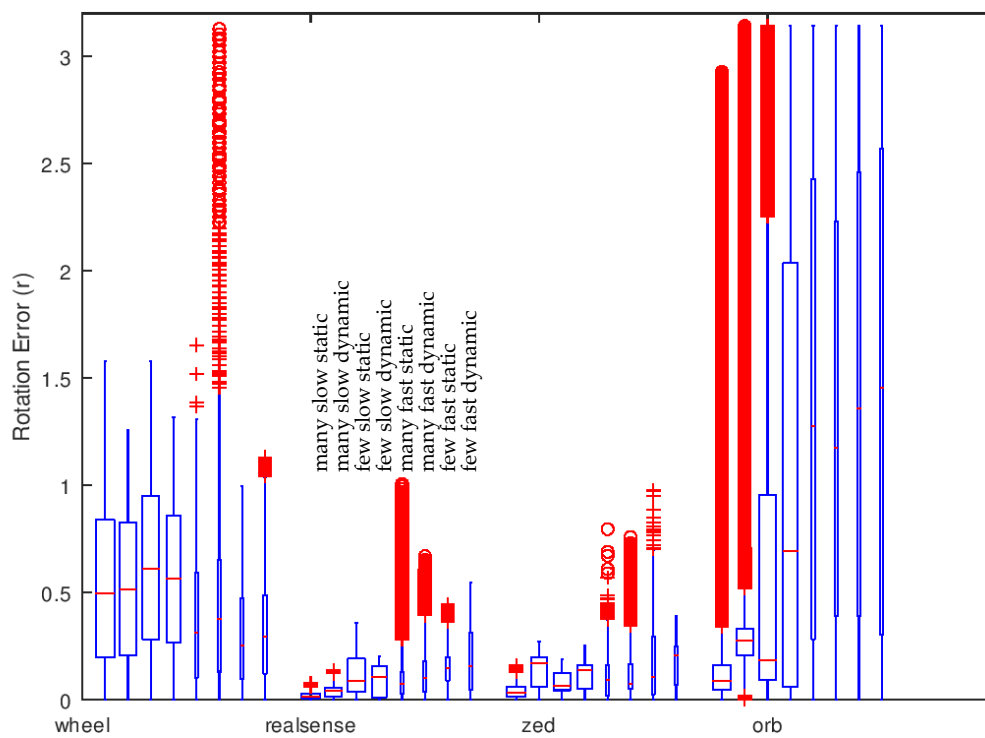


Figure A4. For each of the odometries (wheel encoders, RealSense T265, ZED Mini, ORB-SLAM2), we report the median rotation error (red horizontal line), the 75% observed rotation errors (blue rectangle box), the 95% observed rotation errors (blue error lines), as well as outliers (red crosses above the rest). For each odometry, from left to right, are reported the scenarios: “many slow static”, “many slow dynamic”, “few slow static”, “few slow dynamic”, “many fast static”, “many fast dynamic”, “few fast static”, and “few fast dynamic”.

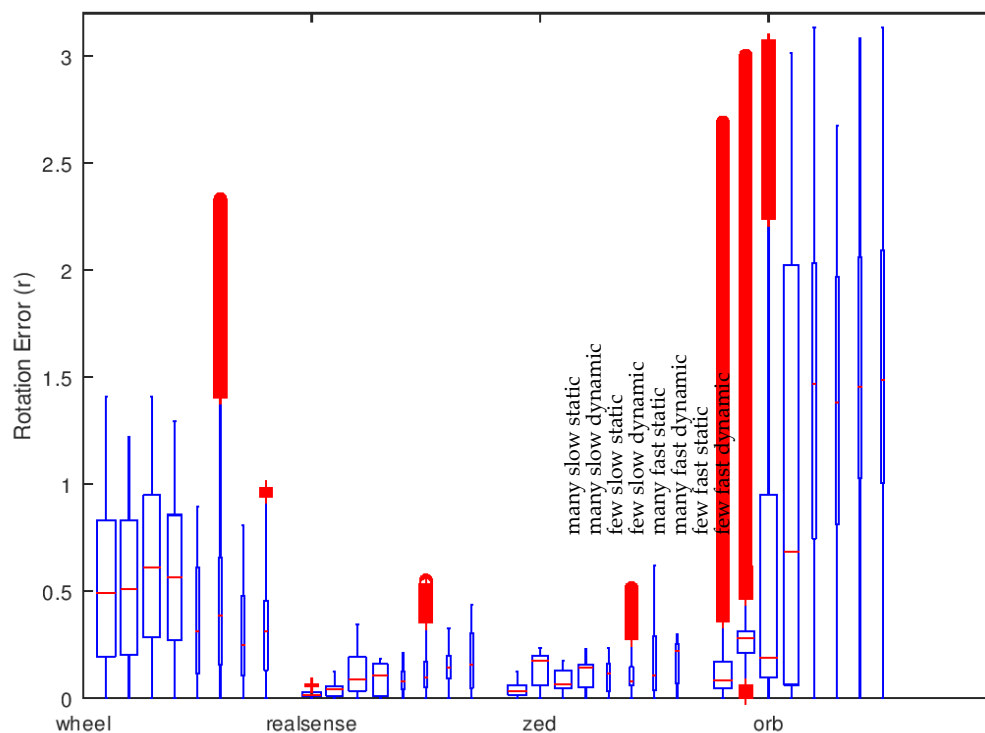


Figure A5. Same as Figure A4, but after soothing the data on 500 points across all sensors, which is about ~1.76 s.

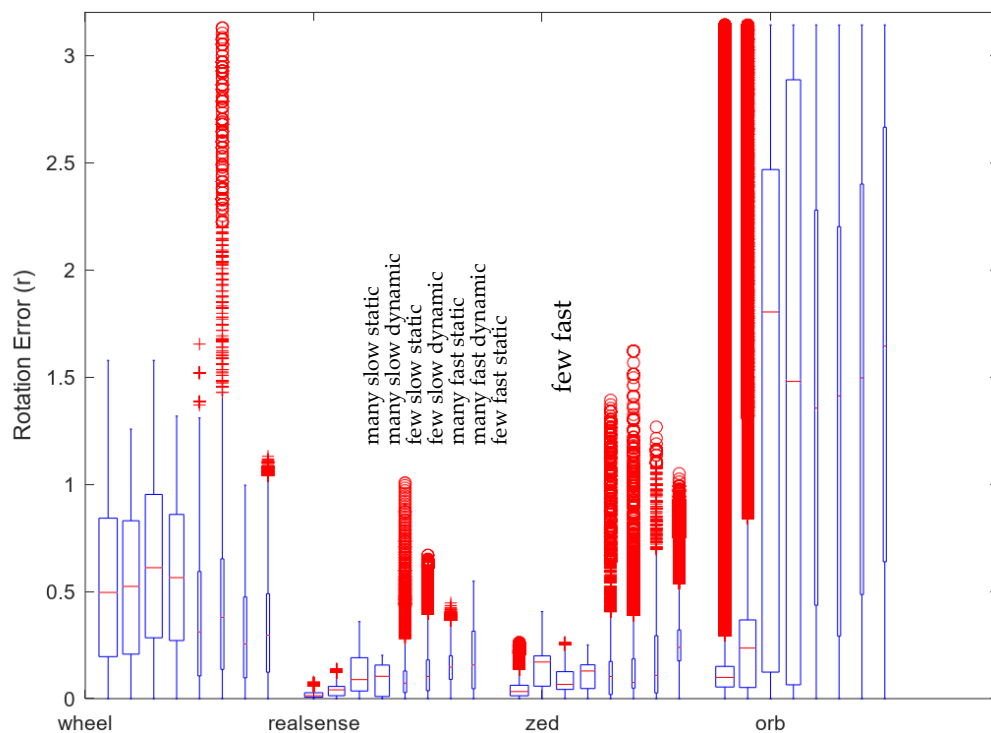


Figure A6. Same as Figure A4, but with ORB-SLAM2 computed offline (i.e., without CPU processing power limitation) and excluding the first seconds of the experiments used for initialization (so the results for other odometries are not exactly those of Figure A4 either).

References

1. Mohamed, S.A.S.; Hagbayan, M.; Westerlund, T.; Heikkonen, J.; Tenhunen, H.; Plosila, J. A Survey on Odometry for Autonomous Navigation Systems. *IEEE Access* **2019**, *7*, 97466–97486. [\[CrossRef\]](#)
2. Aqel, M.O.A.; Marhaban, M.H.; Saripan, M.I.; Ismail, N.B. Review of visual odometry: Types, approaches, challenges, and applications. *SpringerPlus* **2016**, *5*, 1897. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Qin, T.; Li, P.L.; Shen, S.J. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [\[CrossRef\]](#)
4. Mur-Artal, R.; Tardós, J.D. Visual-Inertial Monocular SLAM With Map Reuse. *IEEE Robot. Autom. Lett.* **2017**, *2*, 796–803. [\[CrossRef\]](#)
5. Usenko, V.; Engel, J.; Stücker, J.; Cremers, D. Direct visual-inertial odometry with stereo cameras. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1885–1892. [\[CrossRef\]](#)
6. Whelan, T.; Johannsson, H.; Kaess, M.; Leonard, J.; McDonald, J. Robust real-time visual odometry for dense RGB-D mapping. In Proceedings of the IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013. [\[CrossRef\]](#)
7. Steinbrücker, F.; Sturm, J.; Cremers, D. Real-time visual odometry from dense RGB-D images. In Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops), Barcelona, Spain, 6–13 November 2011; pp. 719–722. [\[CrossRef\]](#)
8. Graeter, J.; Wilczynski, A.; Lauer, M. Limo: Lidar-monocular visual odometry. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018. [\[CrossRef\]](#)
9. Shin, Y.; Park, Y.S.; Kim, A. Direct Visual SLAM Using Sparse Depth for Camera-LiDAR System. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 5144–5151. [\[CrossRef\]](#)
10. Durrant-Whyte, H.; Bailey, T. Simultaneous localization and mapping: Part I. *IEEE Robot. Autom. Mag.* **2006**, *13*, 99–110. [\[CrossRef\]](#)

11. Gutmann, J.S.; Konolige, K. Incremental mapping of large cyclic environments. In Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, 8–9 November 1999; pp. 318–325. [\[CrossRef\]](#)
12. Delmerico, J.; Scaramuzza, D. A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2502–2509. [\[CrossRef\]](#)
13. Angladon, V.; Gasparini, S.; Charvillat, V.; Pribanić, T.; Petković, T.; Đonlić, M.; Ahsan, B.; Bruel, F. An evaluation of real-time RGB-D visual odometry algorithms on mobile devices. *J. Real-Time Image Proc.* **2019**, *16*, 1643–1660. [\[CrossRef\]](#)
14. Joshi, B.; Rahman, S.; Kalaitzakis, M.; Cain, B.; Johnson, J.; Xanthidis, M.; Karapetyan, N.; Hernandez, A.; Li, A.Q.; Vitzilaios, N.; et al. Experimental Comparison of Open Source Visual-Inertial-Based State Estimation Algorithms in the Underwater Domain. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019. [\[CrossRef\]](#)
15. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [\[CrossRef\]](#)
16. Chen, C.; Zhu, H.; Li, M.; You, S. A Review of Visual-Inertial Simultaneous Localization and Mapping from Filtering-Based and Optimization-Based Perspectives. *Robotics* **2018**, *7*, 45. [\[CrossRef\]](#)
17. Ouerghi, S.; Ragot, N.; Boutteau, R.; Savatier, X. Comparative Study of a commercial tracking camera and ORB-SLAM2 for person localization. In Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Valletta, Malta, 27–29 February 2020; pp. 357–364. [\[CrossRef\]](#)
18. Cortés, S.; Solin, A.; Rahtu, E.; Kannala, J. ADVIO: An Authentic Dataset for Visual-Inertial Odometry. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 419–434. [\[CrossRef\]](#)
19. OptiTrack for robotics. Available online: <https://optitrack.com/motion-capture-robotics/> (accessed on 15 July 2020).
20. OptiTrack Prime 13. Available online: <https://optitrack.com/products/prime-13/> (accessed on 15 July 2020).
21. Arlo Complete Robot System. Parallax Inc. Available online: <https://www.parallax.com/product/28966> (accessed on 15 July 2020).
22. Arlo Robotic Platform System. Parallax Inc. Available online: <https://www.parallax.com/product/arlo-robotic-platform-system> (accessed on 15 July 2020).
23. ROS. Available online: <https://www.ros.org> (accessed on 15 July 2020).
24. Propeller Activity Board, WX. Parallax Inc. Available online: <https://www.parallax.com/product/32912> (accessed on 15 July 2020).
25. Introducing ZED Mini. Stereolabs. Available online: <https://www.stereolabs.com/blog/introducing-zed-mini/> (accessed on 15 July 2020).
26. Mur-Artal, R.; Tardós, J.D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [\[CrossRef\]](#)
27. Gálvez-López, D.; Tardos, J.D. Bags of binary words for fast place recognition in image sequences. *IEEE Trans. Robot.* **2012**, *28*, 1188–1197. [\[CrossRef\]](#)
28. Feigl, T.; Porada, A.; Steiner, S.; Löf-fler, C.; Mutschler, C.; Philippsen, M. Localization Limitations of ARCore, ARKit, and Hololens in Dynamic Large-Scale Industry Environments. In Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, Valletta, Malta, 27–29 February 2020; pp. 307–318.
29. Zhang, X.; Yao, X.; Zhu, Y.; Hu, F. An ARCore Based User Centric Assistive Navigation System for Visually Impaired People. *Appl. Sci.* **2019**, *9*, 989. [\[CrossRef\]](#)
30. Kästner, L.; Lambrecht, J. Augmented-Reality-Based Visualization of Navigation Data of Mobile Robots on the Microsoft Hololens-Possibilities and Limitations. In Proceedings of the IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), Bangkok, Thailand, 18–20 November 2019; pp. 344–349. [\[CrossRef\]](#)

