Evaluation of Recent Approaches to Visual Odometry from RGB-D Images

Sergey Alexandrov[1] and Rainer Herpers[1,2,3]

[1] Department of Computer Science, Hochschule Bonn-Rhein-Sieg, Germany
[2] Department of Computer Science and Engineering, York University, Canada
[3] Faculty of Computer Science, University of New Brunswick, Canada
sergey.alexandrov@smail.inf.h-brs.de, rainer.herpers@h-brs.de

Abstract. Estimation of camera motion from RGB-D images has been an active research topic in recent years. Several RGB-D visual odometry systems were reported in literature and released under open-source licenses. The objective of this contribution is to evaluate the recently published approaches to motion estimation. A publicly available dataset of RGB-D sequences with precise ground truth data is applied and results are compared and discussed. Experiments on a mobile robot used in the RoboCup@Work league are discussed as well. The system showing the best performance is capable of estimating the motion with drift as small as 1 $^{\mathrm{cm}}/_{\mathrm{s}}$ under special conditions, though it has been proven to be robust against shakey motion and moderately non-static scenes.

1 Introduction

The objective of a visual odometry system is to compute a continuous camera trajectory through examination of the changes that the motion induces on the images. Research in this area has a long history in robotics [1]. Initially motivated by the NASA Mars exploration program in the 80s, over the years it has yielded systems that were applied on in- and out-door wheeled robots, cars, aerial and underwater vehicles, and even quadruped dog-like robots [2].

The wide choice of vision systems that were used to implement visual odometry includes monocular, stereo, omni-directional and multi-camera setups. Recently low-cost RGB-D cameras were introduced to the market. These visual sensors combine a conventional RGB video camera with a depth sensor, and deliver color images with pre-registered per-pixel depth information at the standard video frame rate. RGB-D cameras have relatively low power consumption and weight which made them suitable and effective sources of information about the environment (and thus the robot's motion). This motivated many researches to explore the possibility of implementing an RGB-D visual odometry system.

Over the past few years a number of such systems were reported in the literature [3,4,5,6,7]. A closely related topic: mapping with RGB-D cameras, also enjoyed considerable attention [8,9,10,11]. The latter emphasizes globally consistent alignment of all captured data, but often involves an odometry subsystem that performs preliminary local alignment. A number of distinct approaches with various benefits and deficits, constraints, and computational complexities exist.

S. Behnke et al. (Eds.): RoboCup 2013, LNAI 8371, pp. 444–455, 2014.

A comparative evaluation of the recent state of the art RGB-D visual odometry systems has been conducted in this contribution. The scope of the evaluation is limited to the software released open-source and capable of real-time performance on the CPU core of a common laptop. Our interest in this area is motivated by participation in the Robocup@Work league. Our target platform is an omni-directional mobile base equipped with an RGB-D camera on which a visual odometry system should be applied. A publicly available RGB-D dataset with various environments and 6 degrees of freedom motions is used for evaluation as well as a dataset captured by ourselves on the target platform.

The remainder of the contribution is organized as follows. Section 2 summarizes the state of the art in RGB-D visual odometry and closely related fields. The evaluated open-source systems are introduced in more detail in Section 3. Section 4 describes the datasets, types of experiments, and metrics that were used to evaluate the systems. The evaluation results and discussion are provided in Section 5. Section 6 summarizes the results and draws the conclusions.

2 Related Work

One core problem of visual odometry is the frame-to-frame transform computation. That is, given two data frames captured by a moving camera at two different time points, compute the rigid transform that relates the positions of the camera. The existing approaches to solve this problem for an RGB-D camera are broadly classified into three groups based on the way they treat the data.

The methods of the first group, often called sparse or feature-based methods, are straightforward adaptations of the classical Structure from Motion stereo odometry pipeline [12]. They compute the motion between frames by solving the absolute orientation problem for a sparse set of 3D points, each of which is a salient and repeatable feature of the environment. This is usually embedded inside a robust estimation framework to tolerate outlier points, which result from incorrect correspondences and independently moving objects. Huang et al. [3] demonstrated such a system for a quadrotor micro aerial vehicle. It is capable of running in real time and performs sufficiently well to allow for autonomous flight in static indoor environments with rich visual features. Du et al. [9] designed an odometry system for interactive environment modeling. It employs a novel hypothesis grading approach which computes visibility conflict criterion for the dense depth data. Recently Domínguez et al. [5] presented a system with an explicit feature filtering stage where a set of rules is applied to keep only the most reliable and consistent in terms of relative 3D positions features.

The second group often referred to as dense or direct methods, solve the motion estimation problem by iteratively aligning the frames to minimize a certain error function. For example: usage of geometrical error between 3D surfaces as defined by the depth images yields a well-understood Iterative Closest Point (ICP) algorithm. Pomerleau et al. [4] presented a modular and efficient ICP library and odometry system. It does not consider color information and is capable of running in real time. Alternatively, Steinbrücker et al. [6] minimize photometric error between intensity images. Their algorithm excels at small displacements but assumes small camera displacements and a static environment. Recently Kerl et al. [7] presented a generalized and extended version which is more accurate and can tolerate moderate dynamics in the environment.

The third group contains hybrid methods. In the pioneering work on RGB-D visual odometry Henry et al. [8] presented a two-stage algorithm. Feature-based method produces an initial motion estimate which is then used as a starting point for the ICP algorithm. The authors reported experiments that showed that the proposed algorithm outperforms its components, however the computational complexity prevents it from running in real time. A similar combined approach was exercised later by by Endres et al. [10] and Hu et al. [11].

3 Evaluated RGB-D Visual Odometry Systems

3.1 Dense Visual Odometry

Dense Visual Odometry (`DVO`) was developed by Kerl et al. [7] and released open-source[1] as a ROS package. The approach is based on the photo-consistency assumption. It states that the same scene point observed by a camera at two consecutive time instants should have the same intensity in both images. The difference in the intensity between the first and the warped second image is defined as the residual and is regarded as a function of the camera motion. The algorithm proceeds by finding the transform that maximizes the posterior probability of the camera motion given the residual image.

The authors focused exclusively on frame-to-frame motion computation and the system maintains the pose by merely multiplying in estimated transforms. Errors in each estimation are accumulated as well. A technique that is often applied to reduce the drift is keyframe insertion. For an incoming frame instead of estimating transform between it and the previous one, the system computes the transform between it and the keyframe. The latter is periodically updated according to a pre-defined rule. This slows down drift accumulation as only the estimation errors at the moments of keyframe insertion are summed. To allow a fair comparison with other systems, `DVO` was augmented with a simple keyframing technique. A keyframe is inserted when the transform estimation failed or when the output transform exceeds 4 cm or 0.05 radians. Sometimes `DVO` produces erroneous estimations without reporting a failure. Such cases are recognized when the transform is greater than 8 cm or 0.08 radians. These numbers are based on the empirical evaluation detailed in Section 5.

3.2 Fast Odometry from Vision

The Fast Odometry from Vision (`FOVIS`) system was developed by Huang et al. [3] and released open-source[2]. It is based on the standard stereo odometry pipeline and adopts a number of optimization techniques reported in the

[1] `https://github.com/tum-vision/dvo`

[2] `https://code.google.com/p/fovis`

literature. The FAST detector with adaptive threshold is used for feature detection. Uniform distribution of features in the frame is achieved using bucketing technique. An initial estimation of rotation is produced through direct minimization of the sum of squared pixel errors between frames. This accelerates feature matching and reduces the fraction of invalid matches by restricting the search window. Nevertheless, a certain amount of incorrectly associated features remain, and need to be pruned by the inlier detection algorithm. It proceeds by computing the graph of consistent (in terms of pair-wise spatial relations) feature matches and finds the maximum clique in it. This is considered to be the set of inliers. Finally the motion is estimated by solving the absolute orientation problem and then iteratively refining the solution by minimizing the re-projection error by discarding the outliers which have survived all the previous processing.

3.3 PointMatcher

PointMatcher (`PM`) is a modular ICP library developed by Pomerleau and Magnenat and released open-source[3]. The transform computation between two frames is regarded as registration of two point clouds. This process is implemented as a reconfigurable chain of modules: data filters, matchers, error minimizers, etc. The authors supply default configuration obtained as a result of extensive tests [4].

Usage of full-resolution depth images is computationally prohibitive and does not allow real-time performance. There are two options how to reduce the amount of data and hence the running time of the algorithm. Either the depth images should be down-scaled, or the point clouds obtained from the full-resolution depth images should be aggressively downsampled. Our initial experiments showed that the former yields better results, so for the evaluation the depth images were down-scaled to 160 × 120 pixels in a pre-processing step.

4 Evaluation Strategy

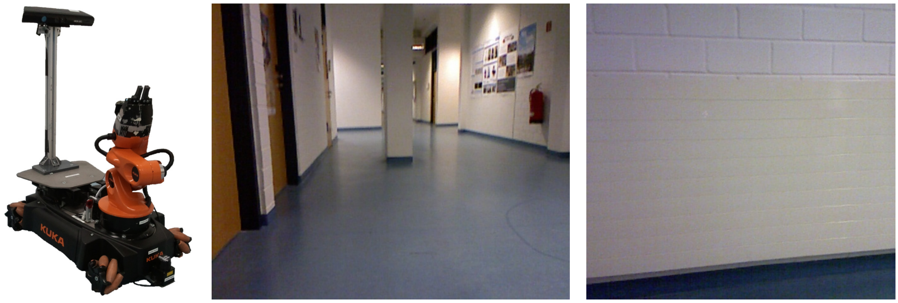
4.1 RGB-D Datasets

A quantitative evaluation of visual odometry systems requires a dataset of image sequences for which the intrinsic camera parameters and the ground truth trajectories are known. One such dataset was recently published by Sturm et al. [13]. It contains a large number of RGB-D image sequences captured with a Microsoft Kinect camera in a typical office and in a large industrial hall. The data are recorded at a resolution of 640 × 480 pixels and at a frame-rate of 30 Hz. Sequences are accompanied by time-synchronized trajectories of the camera estimated with a high-precision motion capture system. The authors state that the relative error on the frame-to-frame basis is below 1 mm and 0.5°. Thus, these trajectories could be used as ground truth data for visual odometry evaluation.

The KUKA youBot [14] is currently the most widely used robot platform in the RoboCup@Work league. It has an omni-directional base with four swedish

[3] `https://github.com/ethz-asl/libpointmatcher`

wheels. Our team has customized it by mounting a sensor tower on the back platform. It hosts a Microsoft Kinect camera which is pointed forwards and is slightly tilted (see Fig. 1a). The robot localizes itself using a laser scanner and a pre-built map of the environment. We drove the robot around a lab and a campus corridor to record several data sequences with trajectories up to 27 m long. The corridor is challenging for visual odometry as the texture is scarce and the structures are ambiguous there, as demonstrated in Fig. 1b and Fig. 1c.



(a) KUKA youBot (b) Campus corridor (c) Textureless wall

Fig. 1. The robot and the environment used for evaluation

4.2 Frame-to-Frame Transform Computation

The principal component of a visual odometry system is the module that estimates the relative transform between two given frames. The quality of this estimation is the limiting factor for the accuracy of the system as a whole [1]. In order to analyze the estimation errors the transform computation module of each system is isolated and a large number of RGB-D frame pairs with known ground truth transforms between them are applied.

We isolate the transform computation module of each system, feed it a large number of RGB-D frame pairs with known ground truth transforms between them and analyze the estimation errors.

Formally, given a pair of consecutive RGB-D frames, ground truth transform between them $\mathbf{T}_g$, and the transform estimated by an odometry module $\mathbf{T}_e$, the estimation error is defined as $\mathbf{E} = \mathbf{T}_g^{-1}\mathbf{T}_e$. $Trans\,(\mathbf{T})$ and $rot\,(\mathbf{T})$ are used to denote the translational and rotational components of a transform $\mathbf{T}$.

Analysis of the collected data yields helpful insights. First, we studied how the magnitude of errors $trans\,(\mathbf{E})$ and $rot\,(\mathbf{E})$ depend on the actual transforms $\mathbf{T}_g$. This information allows to understand the limitations the odometry systems impose on the maximum motion velocities. Secondly, we studied how the relative estimation error which is defined as $\eta_{trans} = \frac{trans(\mathbf{E})}{trans(\mathbf{T}_g)}$ and $\eta_{rot} = \frac{rot(\mathbf{E})}{rot(\mathbf{T}_g)}$ (for translational and rotational components) is related with the estimated transforms $\mathbf{T}_e$. In other words, for each possible output transform the average error that this transform bears is empirically calculated. Knowing this allows to make informed choices of the strategy for keyframe insertion.

In order to have representative statistics about the errors, a large number of sample frame pairs that cover the spectrum of possible transforms is required. We boost the number of sample frame pairs by considering frames in the image sequences that are not strictly adjacent. In specific, for each frame a subset of its successors are selected so that the ground truth transform between the frames is limited by 30 cm and 0.15 radians. This way the number of examples is increased up to almost a million and cover the whole spectrum of translation and rotation combinations (of course limited by the mentioned numbers).

4.3 Odometry for Unconstrained Motion

In this group of experiments the visual odometry systems are evaluated with the sequences from the RGB-D dataset and the performance is assessed with the Relative Pose Error (RPE) metric proposed by Sturm et al. [13].

Formally, given an estimated trajectory consisting of a set of poses $\mathbf{P}_1, \ldots, \mathbf{P}_n$ and a ground truth trajectory consisting of a set of poses $\mathbf{Q}_1, \ldots, \mathbf{Q}_n$ the RPE error at a time step i is defined as follows:

$$\mathbf{E}_i = \left(\mathbf{Q}_i^{-1}\mathbf{Q}_{i+\Delta_t}\right)^{-1} \left(\mathbf{P}_i^{-1}\mathbf{P}_{i+\Delta_t}\right), \qquad (1)$$

where Δ_t is a fixed time interval which is set to 1 s here, so that the value of RPE could be interpreted as drift per second, a natural and comprehensible metric. Following Sturm et al. we summarize RPE distributions by computing root-mean-square error (RMSE). This is in contrast with commonly used median or mean values which give less influence to gross errors yielding optimistic results.

4.4 Odometry for Planar Motion

In this group of experiments, the in-house dataset is applied to evaluate the performance of the visual odometry systems on a ground mobile robot. Both intrinsic and extrinsic camera parameters are known. This allows us to project the pose estimates computed by the systems on the ground plane and zero out pitch and roll rotations. The resulting trajectories were compared with the output of the laser-based localization system. The latter is not precise enough to make quantitative assessments. However, it is globally accurate so the amount of drift could be visually assessed.

5 Evaluation Results

5.1 Frame-to-Frame Transform Computation

First the evaluation of frame-to-frame transform computation was conducted. The sequences from the "Testing and Debugging" and "Handheld SLAM" groups were used with exclusion of "large_with_loop", "large_no_loop" sequences because they do not have entire ground truth trajectory. This provided us with nearly

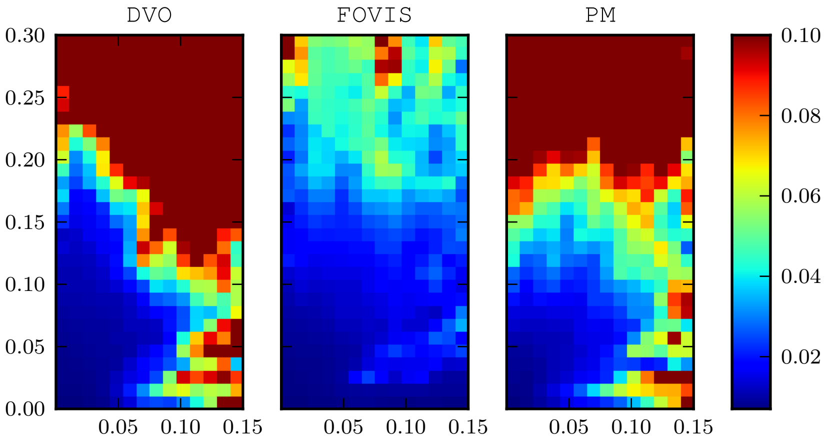


Fig. 2. Magnitude of the translational component of the estimation error (in meters, color-coded) versus ground truth transform rotation (in radians along x-axis) and translation (in meters along y-axis) for the three transform computation modules `DVO` (left), `FOVIS` (middle), and `PM` (right).

one million frame pairs. For each sample the magnitudes of the translational and rotational components of the estimation error were computed.

Figure 2 displays the relation between translational components of estimation errors $trans\,(\mathbf{E})$ and ground truth transforms. All the samples are distributed in a 2D histogram and the mean of each cell is encoded with color. To ensure the same color mapping in all histograms errors above 10 cm were truncated. Thus dark brown corresponds to an error of 10 cm or more. Note that the samples for which the estimation module declared a failure are not included. We observed that `FOVIS` can handle the whole spectrum of examined transforms with a reasonable accuracy. However, its performance significantly degrades when the translation is more than 15 cm. Both `DVO` and `PM` are more restricted. The range of transforms that result in good estimates is limited to about 5° and 15 cm, and in this region `DVO` is consistently move accurate than `PM`. The distribution of rotational components of estimation errors $rot\,(\mathbf{E})$ is very similar and is not reproduced here due to space constraints.

Next the relative error in estimated transform versus its magnitude was considered. Figure 3 presents the translational η_{trans} and rotational η_{rot} components. The data are again distributed in a 2D histogram and the mean of each cell is encoded with color. Relative errors above 30% were truncated, thus dark brown corresponds to a relative error of 30% or more. This value could be interpreted as the measure of reliability of an estimated transform. For example: when the translational component of a transform output by `DVO` is greater than 4 cm and is less than 8 cm and the rotational component is less than 0.05 radians the translational error will be about 15% on average. Therefore, it makes sense to insert a keyframe when the estimated transform falls in the region with the smallest average relative error which will lead to the slowest drift accumulation.

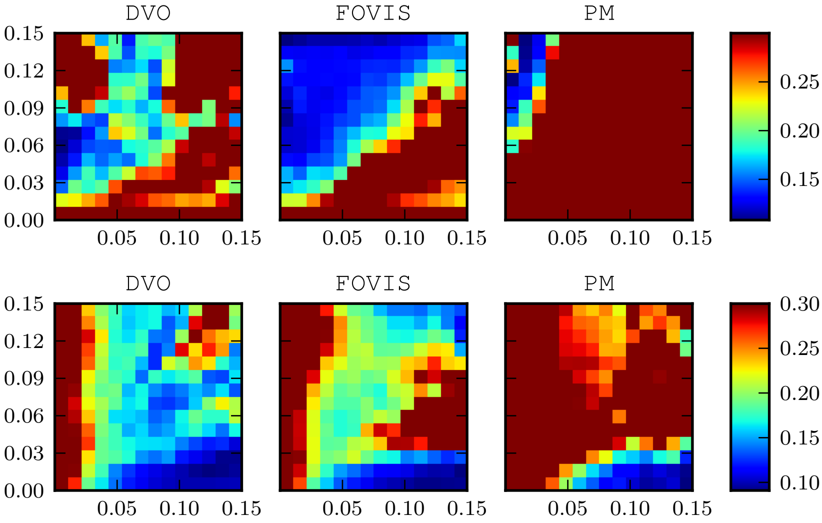


Fig. 3. Relative error η_{trans} in the translational component (top row) and η_{rot} in the rotational component (bottom row) versus estimated transform rotation (in radians along x-axis) and translation (in meters along y-axis) for the three transform computation modules DVO (left), FOVIS (middle), and PM (right).

5.2 Odometry for Unconstrained Motion

The visual odometry systems were evaluated on all sequences from the RGB-D dataset and the summarized results are presented in Table 1.

The sequences in “Handheld SLAM” group contain free flying camera motions with various average velocities and environments. It has been shown that DVO is typically marginally better than FOVIS in terms of translational drift, however in several cases (“fr1/360” and “fr1/desk2”) it has a larger drift. The rotational drift, on the other hand, is slightly less for FOVIS. PM consistently gives considerably larger drift. One notable case (“fr2/360_hemisphere”) is where the performance of all systems significantly degrades. We attribute this to the fact that in a high percentage of frames in the sequence only the ceiling of the industrial hall is visible. It is distant and hence the depth data is inaccurate. The distribution of RPE of all sequences of the group excluding the aforementioned sequence is presented in Fig. 4. DVO and FOVIS performance is almost identical. However, the latter has a smaller spread but marginally larger median value.

In the “Robot SLAM” sequences the camera is mounted on a Pioneer robot which is driven around in an industrial hall. Wires scattered over the concrete ground produce a severe jittering in the video. Furthermore, the structures are often found at a large distances from the camera. It has been shown that DVO is consistently better than FOVIS. PM completely fails in these sequences as the

Table 1. Evaluation results for the three visual odometry systems on the sequences from RGB-D dataset. The second group column shows the average translational ($^{m}/_{s}$) and rotational ($^{deg}/_{s}$) velocities in a sequence according to the ground truth data. The third and fourth group columns show RMSE of translational drift ($^{m}/_{s}$) and rotational drift ($^{deg}/_{s}$) respectively for each odometry system.

	Avg. velocity		Translational drift			Rotational drift		
Sequence name	trans	rot	DVO	FOVIS	PM	DVO	FOVIS	PM
Testing and Debugging								
fr1/xyz	0.244	8.92	**0.025**	0.027	0.082	1.53	**1.44**	3.84
fr1/rpy	0.062	50.15	**0.036**	0.053	0.080	10.47	16.42	**6.07**
fr2/xyz	0.058	1.72	0.005	**0.004**	0.075	**0.30**	0.31	2.74
fr2/rpy	0.014	5.77	0.011	**0.005**	0.058	0.54	**0.36**	2.45
Handheld SLAM								
fr1/360	0.210	41.60	0.129	**0.083**	0.113	4.18	**2.53**	4.93
fr1/floor	0.258	15.07	**0.054**	**0.054**	0.163	2.40	**2.12**	6.90
fr1/desk	0.413	23.33	**0.049**	0.055	0.210	7.17	**6.86**	12.07
fr1/desk2	0.426	29.31	0.066	**0.057**	0.109	4.77	**4.65**	6.38
fr1/room	0.334	29.88	**0.054**	0.064	0.115	2.53	**2.23**	6.29
fr2/360_hemisphere	0.163	20.57	**0.111**	0.132	0.685	**2.62**	6.81	11.39
fr2/desk	0.193	6.34	**0.011**	0.013	0.065	0.56	**0.54**	2.49
fr3/long_office_household	0.249	10.19	**0.013**	0.014	0.052	**0.54**	0.64	2.22
Robot SLAM								
fr2/pioneer_360	0.225	12.05	**0.063**	0.163	0.649	**4.01**	5.80	8.11
fr2/pioneer_slam	0.261	13.38	**0.071**	0.142	0.412	**2.86**	3.17	6.74
fr2/pioneer_slam2	0.190	12.21	**0.061**	0.132	0.462	**3.31**	3.61	5.49
fr2/pioneer_slam3	0.164	12.34	**0.091**	0.102	0.324	2.75	**1.92**	6.41
Structure vs. Texture								
fr3/nostruct_notext_far	0.196	2.71	0.200	**0.183**	0.220	6.30	3.67	**3.13**
fr3/nostruct_notext_near	0.319	11.24	0.300	**0.295**	0.321	**9.89**	10.70	12.95
fr3/nostruct_text_far	0.299	2.89	0.186	**0.092**	0.301	2.37	**1.70**	2.55
fr3/nostruct_text_near	0.242	7.43	0.048	**0.017**	0.254	1.46	**0.86**	8.41
fr3/struct_notext_far	0.166	4.00	0.110	0.109	**0.066**	2.69	3.09	**0.73**
fr3/struct_notext_near	0.109	6.25	0.142	0.105	**0.021**	7.75	5.21	**0.97**
fr3/struct_text_far	0.193	4.32	**0.012**	0.014	0.028	**0.45**	0.50	0.99
fr3/struct_text_near	0.141	7.68	0.040	**0.014**	0.051	1.60	**0.71**	1.91
Dynamic objects								
fr2/desk_with_person	0.121	5.34	**0.013**	0.033	0.082	**0.45**	0.87	2.59
fr3/sitting_static	0.011	1.70	**0.008**	0.014	0.040	**0.25**	0.33	1.21
fr3/sitting_xyz	0.132	3.56	**0.014**	0.030	0.055	**0.52**	0.97	1.88
fr3/sitting_halfsphere	0.180	19.09	**0.036**	0.056	0.108	**1.31**	6.98	4.92
fr3/sitting_rpy	0.042	23.84	**0.045**	0.075	0.145	**0.88**	5.13	4.84
fr3/walking_static	0.012	1.39	0.194	**0.153**	0.268	3.26	**2.18**	5.29
fr3/walking_xyz	0.208	5.49	0.405	**0.256**	0.419	7.38	**4.76**	6.71
fr3/walking_halfsphere	0.221	18.27	0.325	**0.240**	0.380	**5.30**	5.31	8.02
fr3/walking_rpy	0.091	20.90	0.406	**0.306**	0.429	**7.01**	8.66	8.51

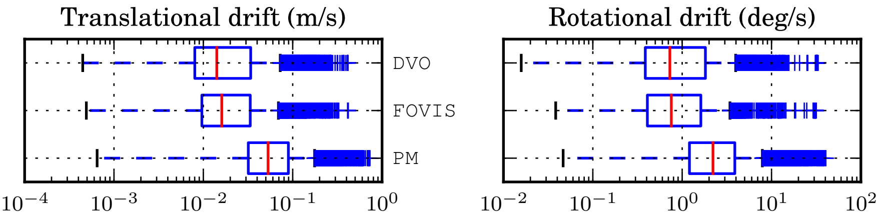


Fig. 4. Aggregated translational drift (left) and rotational drift (right) in the "Handheld SLAM" sequences. `DVO` and `FOVIS` demonstrated similar performance, `PM` has significantly larger drift.

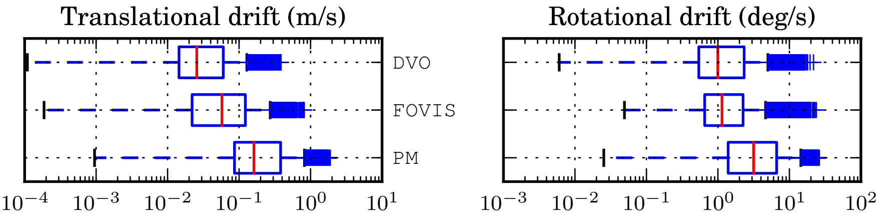


Fig. 5. Aggregated translational drift (left) and rotational drift (right) in the "Robot SLAM" sequences. `DVO` outperforms the other systems in terms of translational drift.

translational drift that it exposes is twice as large as the average velocity. The aggregate RPE of all the sequences of the group is presented in Fig. 5.

"Structure vs. Texture" sequences expose environments with different amount of structure and visual texture. In the most challenging environment with no texture and no structure all systems completely fail producing drift that is of the same or larger magnitude than the average velocity. In the setup with no structure and visual texture `FOVIS` is superior to `DVO` and `PM` fails as is has virtually no data to compute on. In the sequences with structure and no visual features `PM` as expected outperforms other systems. It is interesting to note that in the sequences with structure `PM` performs significantly better than in the other groups. The difference is that here the structures are within approximately one meter and always contain several intersecting planes. Therefore, we conclude that `PM` is particularly suitable when there is always a structure at the close distance.

The sequences from the "Dynamic Objects" group have people moving around in the scene. In the first five sequences the amount of spurious motion is small and `DVO` performs significantly better. In the last four sequences people move faster and occupy most of the image area. `FOVIS` gives much less drift, nevertheless we hold that all the systems failed to produce useful results.

5.3 Odometry for Planar Motion

Furthermore, a number of experiments with the youBot platform as described in Section 4.1 were performed. Figure 6 presents the estimated planar trajectories of the three visual odometry systems for one of the sequences (in red). The ground truth trajectory as output by the laser scanner based localization system is plotted in green. The trajectory estimated by DVO is similar to the actual path. One noticeable deviation happens in the bottom-right corner where the robot rotated in front of a flat textureless wall (see Fig. 1c). FOVIS also showed a good estimation with a similar flaw in the same spot which, however, manifested itself in a wrong turn. The estimate of PM also follows the real path on the whole, however is more erroneous. In general, we observed that the output trajectories mostly preserve the distances and are precise in rotations as well.

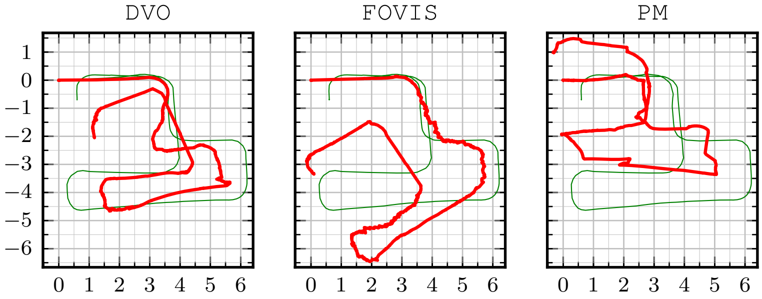


Fig. 6. Planar trajectories of the robot estimated by the odometry systems. Ground truth and estimated trajectories are plotted in thin green and thick red respectively.

6 Conclusions

An evaluation of the three state of the art visual odometry systems (DVO, FOVIS, and PM) was presented. Their frame-to-frame transform estimation components were considered in isolation. The feature-based approach implemented in FOVIS is able to handle the widest spectrum of motions. We presented the relative error in transform estimation as a function of estimated transform which provides an insight for devising a strategy of keyframe insertion for odometry systems.

The algorithms were evaluated on the publicly available RGB-D dataset with precise ground truth. When the motion is smooth and the environment has rich texture and structure both FOVIS and DVO performed equally well giving translational and rotational drift at the level of 1 cm/s and 0.5 deg/s. When the motions are faster and abrupt the performance degrades to nearly 5 cm/s and 5 deg/s. In a particularly challenging group of sequences with excessive jitter DVO clearly outperformed the other methods. It also demonstrated robustness against moderately dynamic scenes where its drift is significantly lower than the others.

The representative of direct methods that minimize geometric error (PM) has performed significantly worse. In the favourable conditions when rich structure is present at the close distance from the camera, it gives drift at the level of 2 cm/s to 6 cm/s and 1 deg/s to 2.5 deg/s. In more challenging conditions it mostly failed producing drift of the same magnitude as the average camera motion. As expected, it outperformed the other systems in a textureless environment.

The systems were evaluated on in-house data collected on an omni-directional robot. The estimated trajectories follow the path of the robot accurately enough.

We conclude that both DVO and FOVIS could be equally well used as odometry systems. The former performs slightly better in normal conditions and is more robust against shakey camera motion and moderately non-static scenes. They both have the same failure mode when the environment has poor visual texture. If this is the target environment for a robot then PM should be considered.

Acknowledgments. Financial support of the BMBF in the FHprofUnt program line, project 6-MIG, grant No 1759X08 and the DAAD PPP project, grant No 50750255, “FPGA based Computer and Machine Vision” is gratefully acknowledged.

References

1. Scaramuzza, D., Fraundorfer, F.: Visual Odometry (Tutorial). IEEE Robotics & Automation Magazine 18(4), 80–92 (2011)
2. Howard, A.: Real-time Stereo Visual Odometry for Autonomous Ground Vehicles. In: Proc. of IROS (2008)
3. Huang, A.S., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D., Roy, N.: Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera. In: Int. Symposium on Robotics Research, ISRR (2011)
4. Pomerleau, F., Magnenat, S., Colas, F., Liu, M., Siegwart, R.: Tracking a Depth Camera: Parameter Exploration for Fast ICP. In: Proc. of IROS (2011)
5. Domínguez, S., Zalama, E., García-Bermejo, J.G., Worst, R., Behnke, S.: Fast 6D Odometry Based on Visual Features and Depth. In: Lee, S., Yoon, K.-J., Lee, J. (eds.) Frontiers of Intelligent Auton. Syst. SCI, vol. 466, pp. 5–16. Springer, Heidelberg (2013)
6. Steinbrücker, F., Sturm, J., Cremers, D.: Real-Time Visual Odometry from Dense RGB-D Images. In: Workshops at ICCV (2011)
7. Kerl, C., Sturm, J., Cremers, D.: Robust Odometry Estimation for RGB-D Cameras. In: Proc. of ICRA (2013)
8. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. In: Proc. of the Int. Symposium on Experimental Robotics (2010)
9. Du, H., Henry, P., Ren, X., Cheng, M., Goldman, D., Seitz, S., Fox, D.: Interactive 3D Modeling of Indoor Environments with a Consumer Depth Camera. In: Proc. of the Int. Conf. on Ubiquitous Computing (2011)
10. Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., Burgard, W.: An Evaluation of the RGB-D SLAM System. In: Proc. of ICRA (2012)
11. Hu, G., Huang, S., Zhao, L., Alempijevic, A., Dissanayake, G.: A Robust RGB-D SLAM Algorithm. In: Proc. of IROS (2012)
12. Sünderhauf, N., Protzel, P.: Stereo Odometry - A Review of Approaches. Electrical Engineering (2007)
13. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A Benchmark for the Evaluation of RGB-D SLAM Systems. In: Proc. of IROS (2012)
14. Bischoff, R., Huggenberger, U., Prassler, E.: KUKA youBot - A Mobile Manipulator for Research and Education. In: Proc. of ICRA (2011)