Visual Gyroscope for Accurate Orientation Estimation

Wilfried Hartmann Michal Havlena Konrad Schindler
Institute of Geodesy and Photogrammetry, ETH Zürich, Switzerland
firstname.lastname@geod.baug.ethz.ch

Abstract

A visual gyroscope is a device which estimates camera 3D rotation using image input, in our case a monocular video. Contrary to traditional Structure-From-Motion (SFM) or visual SLAM, we address the case where only *the rotation must be found, whereas no translation estimate is desired. That case can be solved without computing an explicit 3D map of the environment, thus avoiding computationally expensive bundle adjustment. Instead, a simple linear method is used to obtain globally consistent rotations from relative rotation estimates between image pairs. We show that the obtained camera orientations are accurate w.r.t. ground truth collected with a navigation-grade (dGPS-supported) IMU, and reach 3D bearing errors below 1° over a 1-minute time interval for $> 90\%$ of all cases. Efficient computation is achieved by employing GPU-enabled feature extraction and matching. To warrant on-line performance for sequences of arbitrary lengths we run the global rotation estimation in a sliding-window fashion and show that the accuracy of the camera orientations obtained by chaining the partial solutions stays high. Finally, we compare the proposed visual gyroscope to a publicly available SFM software and experimentally demonstrate the importance of a very large field-of-view for accurate rotation estimation.*

1. Introduction

The ability to self-localize in an unknown environment is a pre-requisite for many applications of image metrology and image understanding. Knowing the location and viewing direction of a moving camera can be a goal in itself, as for example in robot navigation *e.g.* [5, 16, 6, 1], or it can serve as auxiliary function to constrain object detection [17, 26], tracking [7, 4] and mapping [21, 28].

In situations where the position of the observer is already known with sufficient accuracy, *e.g.* thanks to other sensors like GPS, it can be wasteful to solve the full SLAM problem, including the more brittle translation and scale estimation based only on visual input. Here, we address a particular, more restricted part of the localization problem: we design a visual gyroscope, *i.e.* a system that estimates the 3D rotation w.r.t. a reference direction, from video recorded by a monocular camera while moving in an unknown environment – see the illustration in Fig. 1. Our aim is to match high-quality IMUs, which achieve bearing errors on the order of $\leq 1^\circ$ per minute.

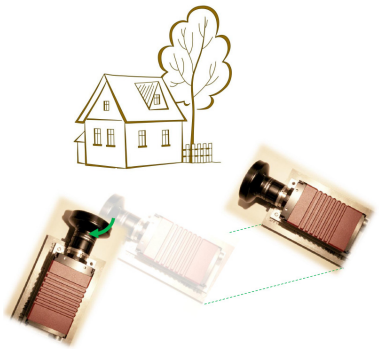


Figure 1: A visual gyroscope measures the changes in camera orientation between different frames. The camera itself undergoes a general motion but if the position does not need to be estimated, one can gain efficiency by omitting its computation.

A classical approach to solve this task, by including an additional hardware sensor, would be to measure absolute bearing with a compass. Unfortunately, the needle of a magnetic compass is deflected by metal objects. A gyrocompass, which is not affected by metal objects, is too large, heavy, and expensive for typical mobile systems. Therefore, it makes sense to measure absolute orientations only at suitable locations and to determine the orientation relative to those reference orientations in other places. Relative rotation, *i.e.* orientation change, can be determined with IMUs by measuring accelerations and integrating them to yield rotation changes. The main limitation of IMUs is that they drift due to error accumulation. Except for expensive high-end devices the drift is rather strong (typically $> 1^\circ$ / min.). On the contrary, a visual gyroscope will have only little drift as long as the same landmarks remain in the field-of-view

over many frames, and can exploit loop closures to correct drift. A further disadvantage of IMUs is that orientation is lost as soon as they are switched off, whereas a visual gyroscope can re-initialize itself when switched on near the last known position. On the other hand, visual navigation requires an environment with a sufficient number of natural features.

The contribution of this paper is two-fold. First, we propose a visual gyroscope based on a fish-eye lens camera; the algorithmic backbone of the system are a robust and efficient method for estimating pairwise relative orientation between frames [22] and a method for computing globally consistent camera rotations without having to recover camera locations or 3D world points [19]. Second, we propose a sliding window approach with two windows running in parallel, so as to turn orientation estimation into an on-line process. The accuracy stays high compared to the off-line process that uses all frames.

The proposed visual gyroscope is evaluated with respect to per-frame ground truth from an independent sensor, namely a navigation-grade GPS/IMU unit. In this way, the accuracy can be evaluated not only at selected check-points (*e.g*. points visited repeatedly) but at every frame. At the same time the reference is completely independent of the camera, and thus unaffected by systematic effects such as calibration errors. We also compare to a recent SFM system [30], a representative of standard SLAM / SFM methods that recover full camera poses and a 3D point cloud of the environment. The evaluation on a 4-minute long video sequence shows that the accuracy of the proposed approach is similar (in fact slightly better) compared to the full SFM solution. Furthermore, we show that, as expected, a very large field-of-view plays an important role for accurate orientation estimation.

2. Related Work

Compared to the enormous literature about Structure-From-Motion (SFM) or visual SLAM, there are rather few papers dealing only with the rotational part of camera motion. The visual gyroscopes described in the literature can mostly be classified into two categories. One group of methods relies on vanishing point detection, and the other one uses additional sensors, such as classical gyroscopes, to support the rotation estimation.

In [12] a monocular camera gyroscope is presented which estimates the orientation based on detecting orthogonal vanishing points in individual frames. To that end line segments are extracted from the images. This approach therefore works well in urban environments, where architectural features provide dominant vanishing directions, and fails in natural environments. Similar approaches include [23, 14].

A sensor fusion method using a gyroscope as well as a camera is presented in [9]. That method is also designed for man-made environments which have predominantly vertical and horizontal line features. Another sensor fusion approach is presented in [11]. Both methods build on the Kalman filter to perform the fusion.

In [15] a rather untypical approach for calculating camera orientation from a single frame is presented. Motion blur in the images is used to estimate the rotation. First, the axis of rotation is found and then the blur length is estimated. The method is fast and elegant, but inherently rather inaccurate, and it only works for purely rotational motion with no significant translation.

There are a few SFM methods which estimate consistent camera rotations before solving for the translations, mostly also relying on vanishing points, *e.g*. [2, 24]. An exception is [19], where consistent rotations are estimated by solving a linear constraint system constructed from pairwise epipolar geometries between overlapping images. This method allows one to determine the camera rotations for all images in a global coordinate frame, in such a way that they are as consistent as possible with the two-view orientations, without having to triangulate 3D scene points (and thus also without finding camera translations). Since most SLAM systems aim to reconstruct the full camera path, and often also the 3D environment, the method does not seem to have found widespread use. However, it is well suited for the visual gyroscope task and plays a central role in our system.

3. Method

Technically, the visual gyroscope problem for a given input frame boils down to estimating the relative rotation w.r.t. a frame with known orientation, *e.g*. the starting frame of the sequence. In practice, the two frames will often be too far apart (or even not overlapping at all), so that several consecutive relative rotations must be chained to connect them. To minimize error build-up and circumvent the decision which of the many possible chains to use, we propose to jointly optimize over all observed pairwise orientations [19].

The processing pipeline assumes an internally calibrated camera, so that the positions of the extracted and matched SIFT features [18] can be converted to ray directions and epipolar geometries for pairs of images can be estimated in the form of essential matrices. The five-point algorithm [25] inside a RANSAC sampling loop [8] is often used to this end. However, rather than committing to the initial result, we prefer to perform a local optimization inside the sampling loop [22]. This further improves the robustness compared to standard RANSAC schemes and contributes to improving the accuracy of rotation estimation [19], which deteriorates if erroneous relative rotations are included.

Once more we point out that for a visual gyroscope it is *not* necessary to estimate a 3D point cloud of the environ-

ment (the "structure" part of SFM, respectively the "mapping" part of SLAM). In fact, it has been observed that camera rotations extracted directly from pairwise epipolar geometries are often more accurate than those obtained from a full 3D reconstruction, because distant points cannot be triangulated reliably [27]. By skipping the structure computation, and especially its optimization with bundle adjustment, one can significantly reduce the computational cost compared to standard SFM or visual SLAM approaches. In the following we describe the pipeline in more detail.

3.1. Image Matching

Standard SIFT feature points and descriptors [18] are used to establish image correspondences. Our implementation uses the open source SiftGPU library [29]. It is sometimes suggested to locally undistort fish-eye images to perspective projection before descriptor computation, to compensate the strong image distortions. We found that matching fish-eye images directly using SIFT descriptors works sufficiently well, thus we prefer to extract the descriptors from the raw images, without local undistortion. The SiftGPU library [29] is also used for image matching. In order to find the best match not only the distance threshold is used but also the minimal ratio between the best and second-best match, as recommended by Lowe [18].

3.2. Relative Orientation

The matches are used to estimate the relative orientation between pairs of images. Since we use a fish-eye camera it is convenient to work with ray directions rather than in pixel coordinates, hence we convert image points to unit vectors with the known calibration.

If at least 50 potential matches for an image pair are found, we proceed to epipolar geometry estimation. In this step we use USAC, a state-of-the-art random sampling framework [22] that integrates a number of well-proven extensions of the classical RANSAC scheme. Essential matrices are computed from minimal sets of five matches sampled at random, using the five-point method [25, 20]. Each essential matrix is scored by its support from the set of SIFT matches, using the Sampson error (the 1st-order approximation to the geometric epipolar error) to discriminate inliers from outliers. Afterwards a local optimization is performed in order to find the essential matrix which best fits to the inlier matches.

Using standard unoriented projective geometry, we fix the coordinate system at the first camera, find the projection matrix of the second one using the cheirality constraint, and extract the rotation from it [10].

3.3. Global Rotation Estimation

The next step aims to optimize all observed pairwise rotations jointly, placing them in a common coordinate frame in such a way that the overall discrepancies between them are minimized. Although the pairwise rotations are already fairly reliable thanks to the local optimization, we only include rotations supported by at least 50 verified matches to avoid grossly wrong constraints.

Formally, we are searching for the rotations $\{\mathtt{R}^i, i = 1 \dots m\}$ of all m cameras in a common coordinate system. The joint optimization over a set of observed pairwise rotations $\mathtt{R}^{ij}$ can then be written as a linear system [19], simply by observing that for any two frames which have already been oriented relative to each other

$$\mathtt{R}^j - \mathtt{R}^{ij}\mathtt{R}^i \stackrel{!}{=} \mathtt{0}_{3\times 3} \ , \tag{1}$$

subject to orthonormality constraints. Although it is in principle possible to enforce the non-linear orthonormality constraints when solving (1), it is more efficient not to do so and to search for approximate, *i.e.* non-orthonormal, rotation matrices instead: System (1) can be decomposed in three smaller subsystems

$$\mathbf{r}_k^j - \mathtt{R}^{ij}\mathbf{r}_k^i = \mathbf{0}_{3\times 1} \tag{2}$$

for $k = 1, 2, 3$, where $\mathbf{r}_k^i$ are columns or $\mathtt{R}^i$. Note that $\mathbf{r}_1^i$ $\mathbf{r}_2^i$, and $\mathbf{r}_3^i$ are actually three linearly independent solutions of the same system. Once can therefore stack all the unknowns $\mathbf{r}^i$ in a single column vector $\mathbf{z}$ of length $3m$; and also stack all the $l \leq m(m-1)/2$ known pairwise rotations into a $3l \times 3m$ matrix $\mathtt{A}$, placing $\mathtt{I}_{3\times 3}$ and $-\mathtt{R}^{ij}$ in column blocks corresponding to j and i, respectively. In this way one ends up with a system

$$\mathtt{A}\mathbf{z} = \mathbf{0}_{3l\times 1} \ . \tag{3}$$

In the ideal noiseless case, $\mathtt{A}^\mathtt{T}\mathtt{A}$ has a three-dimensional null space and the stacked columns of the sought approximate rotations comprise its basis. Practically, three linearly independent least-square solutions to system (3) can be found: $\mathbf{z}_1$, $\mathbf{z}_2$, and $\mathbf{z}_3$ are obtained as the eigenvectors corresponding to the three smallest eigenvalues of $\mathtt{A}^\mathtt{T}\mathtt{A}$. Approximate rotations are then constructed by stacking corresponding subvectors of $\mathbf{z}_k$ into 3×3 matrices. Finally, the constructed matrices are projected onto the manifold of orthonormal matrices with singular value decomposition (SVD).

Comparison to naive chaining. Obviously, one could also place all rotations in a common reference frame by iteratively chaining relatively oriented pairs until all images have been covered. The solution will then depend on which pairwise rotations are chained, and no loop closing will occur due to the tree structure of chaining (while existing loops are implicitly closed by the proposed global estimation). Still, one may ask whether chaining is not sufficient. In that case one faces the following trade-off: on one hand,

nearby images typically have more (and also more accurately localized) matches, leading to more accurate pairwise rotation estimates; on the other hand, longer chains will decrease the accuracy due to error accumulation (the error of an open chain grows with the square root of its length). In our experiments, we tested chaining with different spacings between the frames, and found that in practice all variants fail to reach the 1° target, and are significantly less accurate than the global approach, see Sec. 5.

3.4. Sliding Frame Window

The major drawback of the employed rotation estimation technique is that its time complexity grows quadratically with the length of the input image sequence. This is due to the growing number of image pairs which need to be matched. As a first step one could restrict image matching to nearby images in the sequence. With this, the ability to exploit loop closures would be lost already but still no consistent rotation estimates could be returned before the very last frame of the sequence, when the linear system is solved.

We thus use a sliding window, so as to allow for a continuous output of the current best orientation estimate. There is a natural trade-off: The results are more accurate with a larger window, while the runtime is lower with a smaller one. We show experimentally that it is possible to get close to real-time performance and maintain acceptable accuracy.

For each sliding window of length n all possible frame pairs are matched. These relative orientations are then used as the input for joint rotation estimation [19], to get n orientations which are consistent inside the window.

The goal is now to output the rotations $\{\mathtt{R}^i, i = 1 \dots m\}$ of all m cameras in a common coordinate system, given the rotations $\{\mathtt{R}^r_w, r = 1 \dots n, w = 1 \dots \frac{m}{n}\}$ computed for the sliding frame windows (enumerated by w). The first $\frac{n}{2}$ rotations of the first sliding frame window are in fact the sought rotations in the common coordinate system:

$$\mathtt{R}^r = \mathtt{R}^r_1, r = 1 \dots \tfrac{n}{2}. \quad (4)$$

As each sliding frame window has a half-overlap with the next one, see Fig. 2, the remaining output rotations (except for the last $\frac{n}{2}$ rotations) can be computed as weighted averages of pairs of corresponding rotations. First, the transformations that bring the coordinate systems of the individual sliding frame windows to the common coordinate system must be obtained:

$$\bar{\mathtt{R}}_{w+1} = \bar{\mathtt{R}}_w \mathtt{R}^s_w \mathtt{R}^{t\mathtt{T}}_{w+1}, \quad (5)$$

with $s = \frac{3\frac{n}{2}+1}{2}$ and $t = \frac{\frac{n}{2}+1}{2}$, where s and t are the indices for the same rotation in sliding frame windows w and $w+1$, respectively. $\bar{\mathtt{R}}_1$ is the identity matrix.

The actual formula used to compute the remaining rotations in the common coordinate system by averaging the corresponding consistent rotations from sliding frame windows w and $w+1$ then is:

$$\mathtt{R}^{r+w\frac{n}{2}} = (1-\alpha)\bar{\mathtt{R}}_w \mathtt{R}^{r+\frac{n}{2}}_w + \alpha \bar{\mathtt{R}}_{w+1} \mathtt{R}^r_{w+1}, r = 1 \dots \tfrac{n}{2}, \quad (6)$$

where $\alpha = \frac{r-1}{\frac{n}{2}-1}$ are the weights for individual rotations. Note that the actual averaging is performed in quaternion notation to ensure that the result is a rotation matrix.

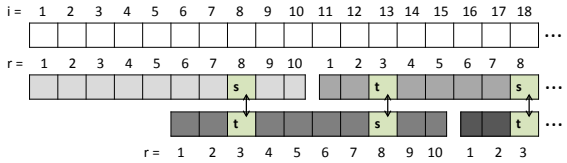


Figure 2: Sliding frame window approach showing four windows of length 10 with their index r and the resulting orientations $\mathtt{R}^i$ on top with their index i. The frames labeled s and t are used to compute the transformation into the common coordinate system.

4. Hardware and Dataset

Our experimental hardware consists of a camera and a rigidly mounted navigation-grade IMU/GNSS unit recording ground truth trajectory. We point out that the IMU recordings are used *only* for evaluation. The proposed visual gyroscope uses solely video input, without any sensor fusion.

4.1. Visual Gyroscope Camera

In order to reliably separate camera rotation from translation, it is advantageous to have a very large field-of-view. However, truly omni-directional systems are not always practically feasible because they would need to be mounted at a prominent place to avoid obstructing parts of the view field. As a compromise, a fish-eye camera with a 185° circular field-of-view was selected, with a diameter of 1,320 pixels and recording at 4 fps.[1]

The camera intrinsics (defining the mapping from pixels to ray directions in a camera-centric frame) have been calibrated off-line using the generic camera model of [13], which has been designed to cover all relevant central projections by approximating the mapping from image pixels to unit rays in camera coordinates with a unifying polynomial/trigonometric function. The model has 23 parameters: the usual 4 parameters for the position of the principal point and the scaling of the two axes, to map from image pixels to the sensor plane; 5 parameters $k_{1\dots5}$ for the radially symmetric part; and 14 parameters for the asymmetric part.

[1] AVT Prosilica GX1920 with Fujinon FE185C057HA-1 lens.

4.2. Ground Truth Recordings – IMU

In order to assess the accuracy of the proposed visual gyroscope, independent measurements are needed which are more accurate than the accuracy goal of 1°. For this purpose we employ a navigation-grade IMU, the Applanix POS LV 210 system [3]. The system consists of a high-quality IMU, a surveying-type dual frequency GNSS antenna rigidly connected to the IMU, and a processing unit that converts the measurements into position and orientation data. The system is used together with a static reference GNSS station located nearby, in order to use differential GPS and cancel out atmospheric influences on the GPS signal.

To measure ground truth for the visual gyroscope, one must make sure that *(i)* the camera is rigidly attached to the IMU and *(ii)* the data recorded by the IMU are synchronized with the acquired images. Note, since we are only interested in relative rotations it is not necessary to know the exact transformation between the IMU and camera coordinate frames (boresight alignment and offset). The shutter signal from the camera is sent via a dedicated cable directly to the POS LV computer system, which time-stamps all recorded data with accurate GPS clock timings. The GPS time signal also serves to synchronize (in post-processing) with the GNSS reference station, which records at 1 Hz during the entire image acquisition. IMU measurements are recorded at 100 Hz, to ensure that there always is an IMU reading within 5 ms of the moment the camera is triggered.

4.3. Evaluation Dataset

The dataset used for evaluation was recorded in a handheld manner in June 2013, in a mixed urban/natural environment. The velocity varies from almost no motion to fast walking speed, and motion patterns include sequences with predominantly translation, others with nearly pure rotation (a failure mode of many visual SLAM methods with which the visual gyroscope shall be able to cope), as well as mixed translation and rotation. The accuracy of post-processed orientation data from the IMU was estimated as 0.25° by the Applanix POSPac software.

For the purpose of this paper, a sequence of 900 frames (nearly 4 minutes) is used. It includes both urban and natural environments, and also different motion patterns and speeds. Furthermore, within the 4 minutes the same path was traversed twice, so that loop-closing is possible. A trajectory estimated with full 6DOF SFM (using VisualSFM [30] on perspective cutouts) closely matches the IMU ground truth, showing that the visual data is not contaminated by any major systematic errors (Fig. 3).

5. Experiments and Results

The proposed visual gyroscope is demonstrated on the outdoor dataset (Sec. 4.3), by comparing to the GNSS/IMU ground truth. Both, the global and the sliding frame window approach are evaluated. Furthermore, we experimentally compare to full SFM with bundle adjustment and examine the benefit of wide and very wide field-of-view for rotation estimation, by running the visual gyroscope for narrower fields-of-view, respectively generating perspective cutouts with different opening angles and reconstructing from those cutouts with VisualSFM [30].

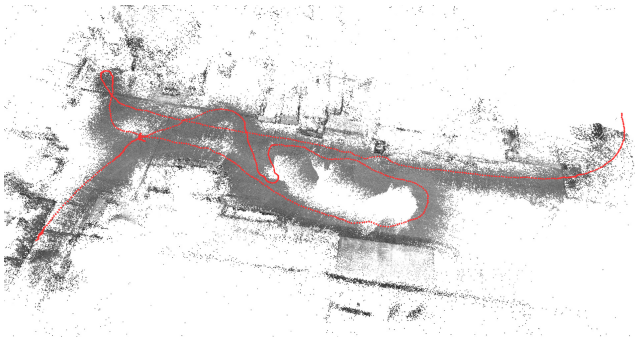


Figure 3: Subsequence of 900 frames used for evaluation. 3D point cloud (gray dots) and camera poses (red pyramids) of the model computed by VisualSFM (Sec. 5.4). Note that the camera returns to a previously visited location after approximately 400 frames.

5.1. Accuracy of Rotation Chaining

The sequence was processed as described above (Sec. 3.2). The hardware is a standard Intel i7-based 3.2GHz desktop computer and the code is written in C++. The SIFT feature extraction takes ≈50 ms per image, matching and epipolar geometry estimation with local optimization takes between 7 and 14 ms per image pair, depending on the number of matches. The time for the actual chaining—a simple multiplication—is negligible.

First, we empirically investigate the error accumulation when chaining relative rotations obtained from image pairs with different temporal spacings (measured in frames). The following *error metric* is used: the angles provided by the IMU ground truth (yaw, pitch, and roll) are converted to a rotation matrix $\mathtt{R}_{GT}$. That matrix is then multiplied with the inverse of the rotation $\mathtt{R}_{VG}$ estimated by the visual gyroscope to obtain a residual rotation $\Delta\mathtt{R} = \mathtt{R}_{VG}^{\mathsf{T}}\mathtt{R}_{GT}$ (which in the error-free case would be the identity). $\Delta\mathtt{R}$ is converted to axis-angle representation, and the angle part—*i.e.* the minimal 3D rotation angle to compensate the difference—is what we call the error of the estimated rotation.

The fixed chaining intervals of lengths 5, 10, and 15 frames are compared in Fig. 4. The longer the interval, the smaller the number of relative rotations that need to be chained in order to reach a certain frame but, on the other hand, the less accurate the individual relative rotations. In

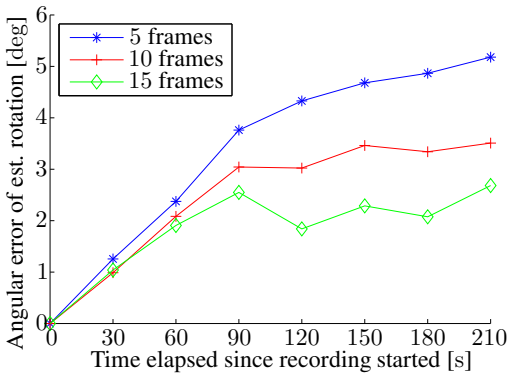


Figure 4: Angular errors of the relative rotations estimated by chaining using different fixed intervals. The angular error is around 2° for a 1 minute long trajectory.

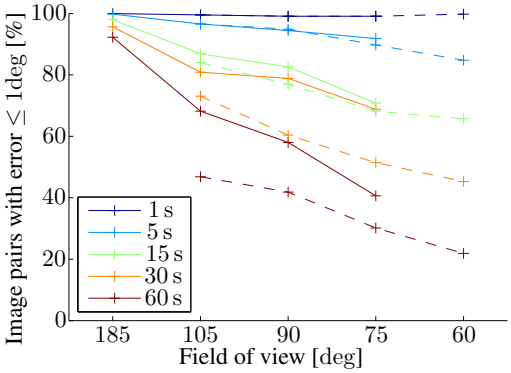


Figure 5: Dependence of rotation accuracy on the field-of-view. The graph shows the fraction of observed cases within the sequence where the angular error remains $< 1°$, over different time intervals. Solid lines denote the proposed visual gyroscope, dashed lines denote VisualSFM.

order to include all frames of the sequence in the evaluation, the accumulated error between first and last frame is computed for five different trajectories starting at frames 1, 2, 3, 4, and 5 respectively, and averaged. In Fig. 4, one can clearly see that the error accumulates slower for longer intervals, *i.e.* the error accumulation over the chain length has a stronger influence than the lower number of correspondences between more distant frame pairs. However, further increasing the spacing is not possible, since for larger spacings some of the estimated pairwise orientations are grossly wrong, causing the visual gyroscope to fail.

Overall, we conclude that as expected shorter chains are to be preferred, but even the longest spacing possible under realistic conditions does not reach the target of $\leq 1°$ error.

5.2. Accuracy of Globally Consistent Rotations

For the second experiment, globally consistent rotations (Sec. 3.3) were estimated for the entire sequence, from

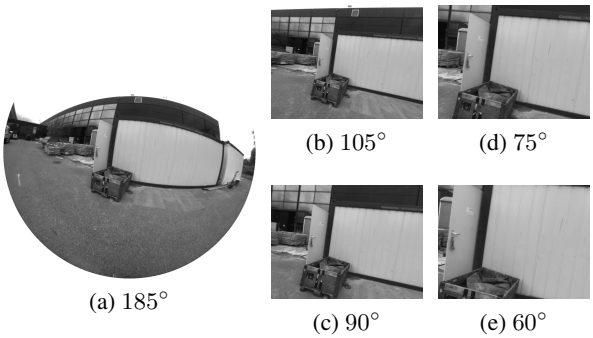


Figure 6: Original image and perspective cutouts with different fields-of-view. Note that the narrowest horizontal field-of-view of 60° offers little that is useful for camera orientation, as most of the image is occupied by the white container.

47,853 pairwise geometries. The optimization for all 900 frames took under 10 s in C++. Note that this time is negligible compared to feature point extraction and matching, and also many times faster than even a highly optimized bundle adjustment needed by SFM or full SLAM.

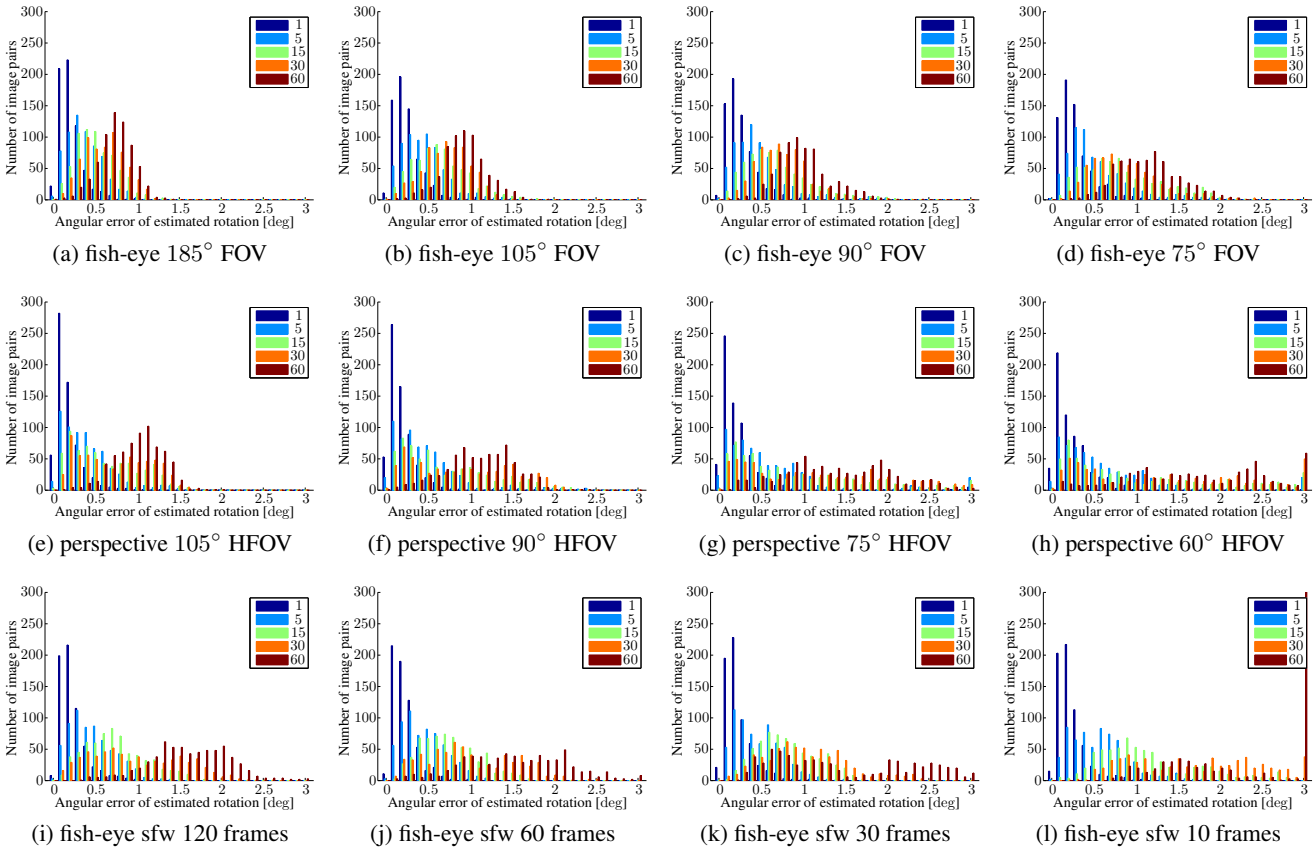
Once globally consistent rotations have been found, we compute the angular error for all image pairs of different fixed temporal intervals. Five different intervals are compared: 1, 5, 15, 30, and 60 seconds (corresponding to 4, 20, 60, 120, and 240 frames, respectively), see Fig. 7a for the histograms of angular errors. As expected, the distribution peaks at larger errors as the temporal spacing increases (*i.e.* more steps are on average required to bridge the interval). The fraction of image pairs whose relative rotations are accurate to $\leq 1°$ is visualized in Fig. 5 (leftmost point on the x-axis). For the 1 minute interval, the accuracy goal is reached in $> 90\%$ of all cases.

5.3. Accuracy for Sliding Frame Window

We now go on to compare the accuracy of the batch result over 900 frames with sliding windows of 10, 30, 60 and 120 frames (Sec. 3.4). The corresponding computation times are shown in (Tab. 1). The histograms are shown in Figs. 7i–7l for decreasing size of the sliding frame window. Note that in this case, the method cannot benefit from the loop closure present in the sequence. For the short intervals of 1 and 5 seconds (blue) the accuracy stays high. For the longer intervals of 15, 30 and 60 seconds the accuracy drops with decreasing frame window size. When using only a 10 frame window the accuracy drops drastically, therefore we consider 30–60 frames as a good choice for both fast computation and reasonably accurate results.

5.4. Accuracy for Different Fields of View

The fish-eye lens used in this work has a 185° circular field-of-view. We simulate a narrower view field by using



(a) fish-eye 185° FOV (b) fish-eye 105° FOV (c) fish-eye 90° FOV (d) fish-eye 75° FOV

(e) perspective 105° HFOV (f) perspective 90° HFOV (g) perspective 75° HFOV (h) perspective 60° HFOV

(i) fish-eye sfw 120 frames (j) fish-eye sfw 60 frames (k) fish-eye sfw 30 frames (l) fish-eye sfw 10 frames

Figure 7: Histograms of rotation errors over different time intervals, when reducing the field-of-view *(a)-(h)* or when using the sliding frame window approach with the visual gyroscope *(i)-(l)*. Note, *(a)-(d)* visual gyroscope; *(e)-(h)* full SFM with perspective cutouts.

Window size [frames]	10	30	60	120
Average time [ms/frame]	369	611	999	1,284

Table 1: Average computation time per frame as a function of (temporal) sliding window size. Timings are indicative, the code is not fully optimized.

only interest points within the 105°, 90° and 75° circle. To ensure possible degradations are not caused by the simplified visual gyroscope approach, but happen also when applying the gold standard, we synthetically create perspective cutouts from the original images for 105, 90, 75, and 60 degree HFOV, using the geometric camera calibration (Fig. 6). On the cutouts, we run full SFM using VisualSFM [30], a state-of-the-art SFM toolkit. Note that a 60° HFOV already corresponds a "wide-angle" lens, *e.g.* for a camera with a 35 mm sensor it means a 30 mm lens.

VisualSFM exhaustively attempts to match all pairs of images and connect the image set as densely as possible, thus the estimated camera poses are an upper limit for what can be reached with more selective SFM/SLAM approaches that only use correspondences between some of the image pairs. The reconstructed trajectories appear to be free of blunders and all images were connected in a single model in most cases (27 images failed to connect for 60° HFOV).

After extracting relative rotation information from the estimated camera poses one can draw histograms of the angular error distribution. The histograms are shown in Figs. 7e–7h for decreasing HFOV. One can clearly see that the accuracy of rotation estimation significantly decreases with the decreasing field-of-view. Even for the widest HFOV of 105°, the accuracy goal of 1° per minute is reached in less than 50% of the time (Fig. 5).

Comparing the visual gyroscope with full SFM at a view field of 105°, 90°, and 75° shows (Figs. 7b–7d) that the proposed approach achieves a comparable error distribution (in fact even a slightly higher proportion of errors are below our goal of 1°), *i.e.* dispensing with 3D structure computation and bundle adjustment does not impair the rotation estimates, but rather improves them, see Fig. 5.

6. Conclusions and Outlook

We have presented a visual gyroscope software capable of recovering camera rotations without estimating full camera pose and 3D scene structure. The proposed batch system, which inherently uses loop closure, proved to perform well in the experimental evaluation with ground truth from a navigation-grade GPS/IMU system. Highly accurate orientation data with error within 1° for $> 90\%$ of all 1-minute sub-sequences were delivered. The proposed on-line (sliding frame window) approach achieves a constant computational time per window, but the accuracy drops by a factor of two, mainly due to the lack of a mechanism for detecting loop closure. We would like to address this problem in our future work.

Moreover, we have shown that a very large field-of-view (fish-eye lens or panoramic setup) as well as a relatively large, redundant set of pairwise frame-to-frame orientations are required to reach that accuracy, even when using full SFM and bundle adjustment.

Furthermore, the visual gyroscope has only a small drift and thus better long-term accuracy than consumer-grade IMUs, as long as the same landmarks remain visible for some time. Conversely, it will inherently have lower short-term accuracy than even a cheap IMU. Hence, we want to further increase the accuracy with sensor fusion between the two complementary devices.

References

[1] M. Agrawal and K. Konolige. FrameSLAM: From bundle adjustment to real-time visual mapping. *IEEE T-RO*, 24(5):1066–1077, 2008.

[2] M. E. Antone and S. Teller. Automatic recovery of relative camera rotations for urban scenes. In *CVPR*, 2000.

[3] Applanix Corp. Applanix POSPac mobile mapping suite. http://www.applanix.com/products/land/pospac-mms.html, 2013.

[4] W. Choi and S. Savarese. Multiple target tracking in world coordinate with single, minimally calibrated camera. In *ECCV*, 2010.

[5] A. Davison. Real-time simultaneaous localisation and mapping with a single camera. In *ICCV*, 2003.

[6] E. Eade and T. Drummond. Monocular slam as a graph of coalesced observations. In *ICCV*, 2007.

[7] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *CVPR*, 2008.

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, June 1981.

[9] J. R. Goulding. Biologically-inspired image-based sensor fusion approach to compensate gyro sensor drift in mobile robot systems that balance. In *MFI*, 2010.

[10] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[11] D. Hong, H. Lee, H. Cho, Y. Park, and J. H. Kim. Visual gyroscope: Integration of visual information with gyroscope for attitude measurement of mobile platform. In *ICCAS*, 2008.

[12] V. Huttunen and R. Piché. A monocular camera gyroscope. *Gyroscopy and Navigation*, 3(2):124–131, 2012.

[13] J. Kannala and S. S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE T-PAMI*, 28:1335–1340, 2006.

[14] C. Kessler, C. Ascher, N. Frietsch, M. Weinmann, and G. Trommer. Vision-based attitude estimation for indoor navigation using vanishing points and lines. In *PLANS*, 2010.

[15] G. Klein and T. Drummond. A single-frame visual gyroscope. In *BMVC*, 2005.

[16] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, 2007.

[17] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool. Integrating recognition and reconstruction for cognitive traffic scene analysis from a moving vehicle. In *DAGM*, 2006.

[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.

[19] D. Martinec and T. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *CVPR*, 2007.

[20] P. Moulon. openMVG: "open Multiple View Geometry". http://imagine.enpc.fr/~moulonp/openMVG/, 2012.

[21] M. Pollefeys et al. Detailed real-time urban 3d reconstruction from video. *IJCV*, 78(2-3):143–167, 2008.

[22] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J. Frahm. USAC: A Universal Framework for Random Sample Consensus. *IEEE T-PAMI*, 35(8):2022–2038, Aug 2013.

[23] L. Ruotsalainen, J. Bancroft, G. Lachapelle, H. Kuusniemi, and R. Chen. Effect of camera characteristics on the accuracy of a visual gyroscope for indoor pedestrian navigation. In *UPINLBS*, 2012.

[24] S. Sinha, D. Steedly, and R. Szeliski. A multi-stage linear approach to structure from motion. In *Trends and Topics in Computer Vision*, Springer LNCS, 2012.

[25] H. Stewénius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *ISPRS J. Photogrammetry and Remote Sensing*, 60:284–294, 2006.

[26] P. Sudowe and B. Leibe. Efficient use of geometric constraints for sliding-window object detection in video. In *ICVS*, 2011.

[27] J. Tardif, Y. Pavlidis, and K. Daniilidis. Monocular visual odometry in urban environments using an omdirectional camera. In *IROS*, 2008.

[28] R. Timofte and L. Van Gool. Multi-view manhole detection, recognition, and 3d localisation. In *ICCV Workshops*, 2011.

[29] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). http://cs.unc.edu/~ccwu/siftgpu/, 2007.

[30] C. Wu. VisualSFM: A Visual Structure from Motion System. http://ccwu.me/vsfm, 2011.