

Online Visual Gyroscope for Autonomous Cars

Abstract—Knowing the exact position and rotation is a crucial necessity for the navigation of autonomous robots. Even in outdoor environments GPS signals are not always accessible for estimating online rotation and position of robots. Also inertial aided navigation methods have their own defects such as the drift of gyroscope or inaccuracy of accelerometer in agile motions and environmental sensitivity of compass.

In this article, we have introduced a novel online visual gyroscope that can estimate the rotation of a moving car with analyzing the images of a monocular camera installed on it. Our real time visual gyroscope utilizes an efficient method of rotation estimation between each pair of camera frames neither considering 3D points nor vanishing points. Instead, our approach assumes a fixed depth for the majority of matched key points between two frames which is more prevalent in outdoor environments like the case of autonomous car. We also analyzed different methods of extracting 2D correspondences between two frames and concluded the optimum factors for a real time implementation. Based on these determinations, we evaluated our visual gyroscope in several datasets from KITTI benchmark and showed that it can estimate the rotation with the rate of 10 frames per second and has the average drift of 0.08 deg per frame.

I. INTRODUCTION

Accurate estimation of the position and orientation of a robot or any other kind of autonomous vehicle is a necessity for its navigation process. Traditionally, several navigation sensors such as gyroscope, GPS, accelerometer and compass were used in order to address this problem. Nowadays due to the accessibility of robots to the GPS data in outdoor environments and utilization of fusion algorithms, the (x,y) coordinates of the robot and its rotation can be estimated with a relatively good precision [1]. However, each one of these sensors has its own drawbacks and especially in some cases all or a number of them are not available. The GPS signals even in outdoor environments are not always accessible and in some locations the robot may miss its connection to the satellite transferring signals. The gyroscope and accelerometer, on the other hand, are always accessible but have other defects. The gyroscope suffers from drift and unknown initial state and the accelerometer is sensitive to agile movements [2]. Thus we can say still there is not a final solution for this issue.

Contemporary advances in image processing techniques and computer vision applications, have persuaded researchers to contemplate camera as a new navigation sensor for estimating rotation and translation of a moving system like a robot. In this article that we have focused merely on the rotation estimation, the camera can be used as a visual gyroscope for computing robot's orientation. The term of visual gyroscope has been recently introduced by Hartmann et al. in [3] for merely estimating rotation of camera. They have used a series of camera frames and estimated the relative rotation for each pairs of frames like a gyroscope sensor. Then they optimize all of relative rotations for a group of frames in a global coordinate frame. The main drawback of this work is its slow rate of

processing especially for an acceptable amount of accuracy. It also can not indicate the online rotation for each frame due to the necessity of optimization for a sequence of camera frames.

The contribution of this paper is to use a prevalent feature of camera images in autonomous cars in order to propose an efficient method of estimating camera rotation. This new approach does not have the constraints used in previous works such as dependency on the existence of vanishing points. It also does not rely on building 3D point correspondences between two camera frames; therefore, it can handle the issue of rotation estimation between each pair of camera frames expeditiously. All of these features of the proposed method makes it suitable to be utilized as a visual gyroscope for online estimation of camera rotation. This paper also examines different ways of implementing feature detection and matching between two frames regarding their accuracy and speed. Using all of these evaluations, we provided the methodology of implementing a practical visual gyroscope that can be used in most of outdoor environments by autonomous cars or other vehicles.

II. RELATED WORKS

Up to now several methods have been proposed for the visually rotation estimation of moving cameras. SLAM and SFM based methods (e.g. [4] and [5]) can give the relative rotation between frames as a part of their results; however, these methods are not efficient for merely rotation estimation due to their time consuming 3D map building. Visual odometry based methods (e.g. [6], [7]), on the other hand, try to figure out the rotation and translation of the camera by analyzing each pair of frames key points without regarding of building 3D points.

Among visual odometry methods, there have been some works that merely estimate the rotation of the camera [3], [8], [9], [10], [11], [12], [13], [14]. Using vanishing points in the camera frame for rotation estimation, most of these methods are not applicable for outdoor environments that plague with lack of vanishing points. [3], [12] are another methods that estimate the rotation for a group of images in a global coordinate frame. These methods, for having good accuracy, should analyze a large number of frames and need a longer time to compute relative rotation of each frame.

Kneip et al. recently proposed two geometric methods [13], [14] for the rotation estimation between two camera frames that each one has its own challenges for practical applications like our autonomous car. The method proposed in [13] provides up to 20 solutions for the rotation estimation between two frames among which finding the best answer is a crucial task. Also the optimization method proposed in [14] needs an initial rotation that makes it inapplicable for estimating large rotations. These methods and also some other popular methods have been implemented in a library called "OpenGV" [15]

that was significantly helpful for our experiments in order to evaluate previously related approaches.

III. VISUAL GYROSCOPE

In this section we will introduce steps that we followed in order to implement a visual gyroscope that estimates the rotation between each pairs of camera frames. This procedure can generally be decomposed into two separate parts. In the first step, two sequent camera frames are analyzed in order to find correspondent points between them. For this purpose, key points of the images are detected and the nearest points from each frame are matched together as correspondences. In the second step, the rotation between these two groups of correspondences is estimated. Hence at first we introduce a previous related method that finds the rotation between 3D correspondences and then explain how we applied this method on the 2D correspondences. In the following we will cover these two steps in more detail.

A. Feature Detection and Matching

We used feature detection and matching methods in order to find correspondent key points in each pairs of camera frames. Although in some previous works such as [16] researchers devised a new method of key point detection, we decided to use a prevalent key point detector instead of focusing on proposing a new one. For this purpose, we evaluated several previously introduced feature detectors and descriptors to find out an efficient combination of them with good accuracy and speed in rotation estimation. Finally we selected the FAST key point detector [17] and SIFT [18], BRIEF [19], BRISK [20], ORB [21] and FREAK [22] feature descriptors to be considered in our future experiments and evaluated their efficiencies. We also investigated the accuracy and speed of each descriptor with different number of key points retained from feature detection process. A comparison on the average running time and also average rotation error for each descriptor with different retained key points is visible in the Fig. 1 and Fig. 2. The characteristics of the system that we used for this comparison is explained in the fourth section of this paper. One can choose an appropriate feature descriptor and suitable number of retained key points based on the constraints that is needed such as speed, accuracy or both of them. For our visual gyroscope, we mainly prefer to have a real time processing rate (ex. 10 frames per second), so we decided to choose fast descriptors with minimum number of retained key points.

B. Rotation only estimation

In this section, we will focus on the procedure of estimating rotation between two sequent camera frames that is needed in order to compute roll, pitch and yaw angles at each moment. For this purpose we will first review a previously proposed method of mere rotation estimation by Arun et al. [23] that uses 3D point correspondences between two frames. Then we will introduce our method that adopt Arun's method for mere rotation estimation without any 3D point calculation.

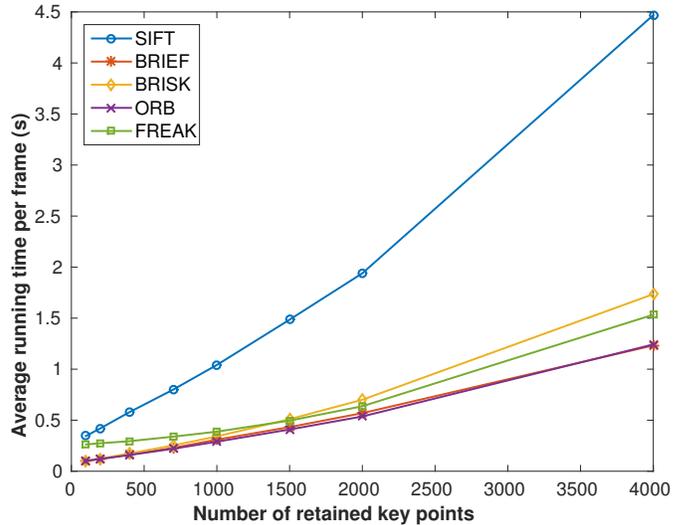


Fig. 1. Average running time of rotation estimation for each frame using different descriptors with different retained points

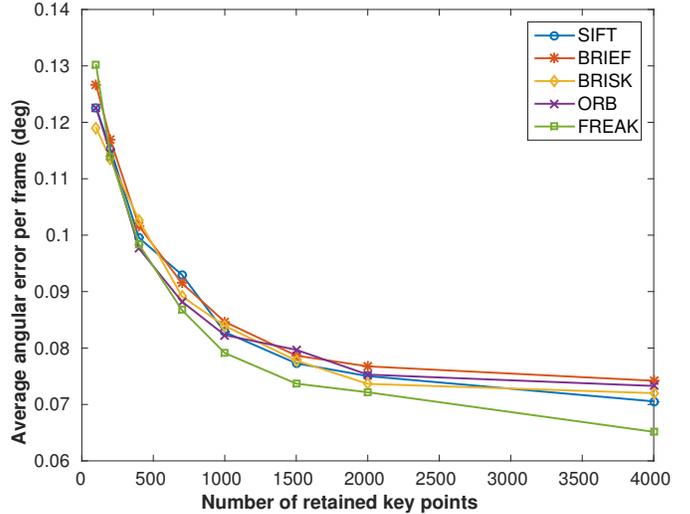


Fig. 2. Average angular error for each frame using different descriptors with different retained points

1) *Rotation only estimation from 3D points:* Up to now several methods have been proposed for estimating rotation and translation matrices between 3D correspondences; however, for our visual gyroscope we are only interested to compute the rotation matrix. Hence we decided to first decouple the rotation and translation effects in the transformation between two frames and then merely focus on the rotation matrix estimation. For this purpose an efficient method has been proposed by Arun et al. [23] that tries to omit the effect of translation on 3D points and then estimate pure rotation between them. In this method, two correspondent 3D points (p_i and p'_i ; $i = 1, 2, \dots, N$) are supposed to have this relationship with each other:

$$p'_i = Rp_i + T \quad (1)$$

For removing the effect of translation, Arun et al. proposed to calculate relative location of each 3D point from the mean point of its relevant set. We call these currently produced points

q_i and q'_i for the first and second 3D sets respectively that are computed using these equations:

$$q_i = p_i - \frac{1}{N} \sum_{k=1}^N p_k \quad (2)$$

$$q'_i = p'_i - \frac{1}{N} \sum_{k=1}^N p'_k \quad (3)$$

Now according to Arun et al., we can say that q_i and q'_i have merely the rotation transformation between each other and this equation confirms for them:

$$q'_i = R q_i \quad (4)$$

Arun's method defines H as the covariance matrix between q_i and q'_i points like below. Then it derives the rotation matrix from computing the Singular Value Decomposition (SVD) of H:

$$H = \sum_{i=1}^N q_i q_i'^t \quad (5)$$

$$H = U \Lambda V^t \quad (6)$$

Finally the rotation matrix (R) between 3D correspondences based on the Arun's method is achieved through this equation:

$$R = V U^t \quad (7)$$

Note that valid rotation matrix should have determinant equal to +1, in other case (pretty rare case) the algorithm would fail [23].

2) *Our method for rotation only estimation without 3D points:* In most of previously proposed methods, in order to find rotation and translation between two sequent camera frames, at first the 3D location of matched correspondent key points should be built through a process called feature triangulation [24]. Being helpful only for robust translation magnitude estimation, feature triangulation is not really necessary for merely rotation estimation between two frames. Thus we will provide a rotation estimation solution for our visual gyroscope based on bearing vectors. In this section at first we will show how bearing vectors are computed for each set of correspondences. Then we will express how we applied Arun's rotation estimation method on our computed bearing vectors without triangulation and building 3D points.

Bearing vectors are 3D unit vectors in a unit sphere around the camera center pointing to specific 3D locations in the world space. Using the pinhole camera model, the bearing vector b_i for each image point measurement (u_i, v_i) can be computed by an inverse calibration matrix multiplication [24]:

$$b_i = \frac{1}{\|K^{-1}(u_i, v_i, 1)^t\|} K^{-1}(u_i, v_i, 1)^t \quad (8)$$

Now in order to apply Arun's method on the bearing vectors instead of 3D points, we assume that most of detected bearing vectors are referring to 3D points with equal distance from the camera center. Based on this assumption, the location of each 3D point p_i has this relationship with its bearing vector b_i :

$$p_i = \lambda * b_i + n_i \quad (9)$$

Which λ refers to the distance coefficient that is fixed for all 3D points and n_i is the variation. It is obvious that the amount of n_i varies for each 3D point; however, we neglect its value in comparison with the distance coefficient for high depth environments that the location of 3D points have enough great distance from the camera. This situation usually occurs when we work on outdoor environments especially the automobile context which we can write this equation for the bearing vectors:

$$p_i = \lambda * b_i \quad (10)$$

Note that we consider the λ as a fixed but unknown value because we didn't computed the exact location of 3D points. Now we can apply Arun's method for estimating decoupled rotation between two sets of bearing vectors (b_i and b'_i) computed by the above methodology. Hence we can write:

$$H = \sum_{i=1}^N (\lambda b_i) (\lambda b'_i)^t \quad (11)$$

$$H = \lambda^2 \sum_{i=1}^N b_i b_i'^t \quad (12)$$

The rotation matrix based on the Arun's method is derived from the SVD of H matrix:

$$H = U \lambda^2 \Lambda V^t \quad (13)$$

$$R = V U^t \quad (14)$$

As it is visible in the above equation, we can conclude that the λ coefficient does not have any effect in the calculation of rotation matrix and knowing its value is not necessary for this regard. Thus we can compute the rotation matrix between two sequent camera frames without utilizing 3D correspondences by applying the Arun's method on our 2D bearing vectors.

C. Outlier rejection

Due to the fact that in most of 2D correspondences between two sequent frames there are some outliers, we applied the famous RANSAC scheme [25] on the two sets of bearing vectors in order to choose the best rotation with maximum number of inliers. For all of datasets that we used in our experiments, the RANSAC number of iterations and distance threshold values were assigned to 50 and 0.000002 respectively.

IV. EXPERIMENTS AND RESULTS

In this section we will study detailed results from our experiments that aim to evaluate the efficiency of proposed visual gyroscope. Two important determinants have been considered to record in all experiments: 1) Rotation estimation accuracy and 2) The running time. We will see that these two factors have an inverse relationship with each other; therefore, an online visual gyroscope has less accuracy comparing with an offline one.

A. Datasets and Hardware

We have used 14 datasets from Karlsruhe [26], [27] and KITTI [28] benchmarks that were gathered from a camera on top of a moving car. All datasets have been selected from different environments with different motions and conditions. The total number of camera frames that we considered in our experiments was more than 3300 frames using all 14 datasets. In these dataset we assumed the IMU rotation data as our ground truth due to its high accuracy and short duration of each dataset (so the IMU drift is negligible). We also decided to only investigate the yaw angle due to the fact that rotations around other axes doesn't have enough comparable changes for a moving car. The processor used for executing computer vision algorithms and other geometric parts was a Dual-Core Pentium CPU with frequency of 2.8 GHz and 4 gigabytes of RAM.

B. Number of Key Points

In this section we want to address the effect of number of key points computed in each pair of camera frames on the accuracy and the speed of visual gyroscope. Hence we devised several experiments in which each time some specific number of detected key points have been used. According to the Fig. 2, that shows the average yaw error for different descriptors with different retained key points, considering more key points in the computations can decrease the average yaw error and lead to more accurate results. On the other hand, it is visible in the Fig. 1 that increasing the number of retained key points will increase the running time of rotation estimation for each pair of camera frames. Thus we should consider a trade off between the accuracy and the speed of computations. Since the proposed visual gyroscope should operate in a real time rate computation, about 10 frames per second, we decided to use BRIEF feature descriptor and assigned the value of 100 as the number of retained key points in our experiments. It is obvious that these parameters can be altered for achieving best results in different processors with different speeds.

C. Accuracy of the real time visual gyroscope

In order to evaluate the accuracy of the proposed gyroscope, it was utilized in order to estimate relative rotation between each pairs of camera frames in all datasets. Due to the constraint of real time operation, BRIEF descriptor with number of 100 key points were selected as expressed in previous section. Using these parameters, the visual gyroscope is capable of estimating the rotation of camera frames in a rate of 10 frames per second which is considered an acceptable rate for our datasets with the same frame rate.

Frame to frame yaw degree was derived from the estimated rotation matrix R using the equation 15. Then the difference of estimated yaw with the ground truth was computed as the frame to frame yaw error. Fig. 3 shows the histogram of absolute error of estimated yaw degrees for each pairs of camera frames using all 14 datasets. As it is visible, in most cases the proposed gyroscope has a an accuracy below 0.2 deg in each frame. It also in some cases has errors greater that 0.2 deg and in some exceptions it has errors above 1 degree per frame.

$$Yaw_{vision} = \arctan(-R(3, 1), R(3, 2)^2 + R(2, 2)^2) \quad (15)$$

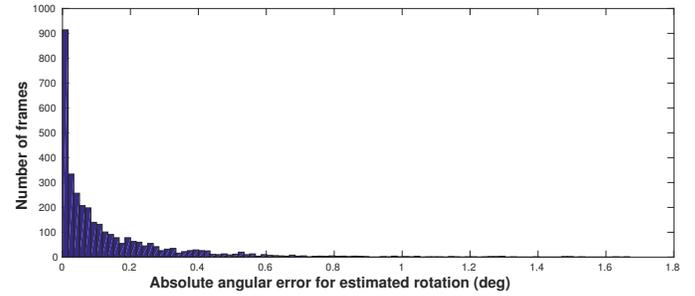


Fig. 3. Histogram of frame to frame absolute error

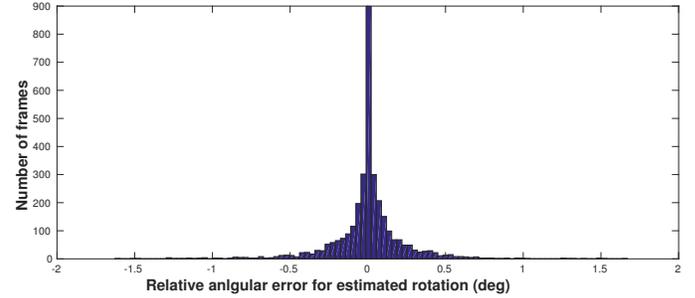


Fig. 4. Histogram of frame to frame relative error

We also computed the overall yaw degree in each frame as the accumulation of frame to frame yaws in all previous frames. Fig. 5 shows the result of yaw estimation for 9 different datasets from KITTI benchmark. It is visible that based on the conditions existed in each dataset, such as automobile's speed and the environment, the proposed gyroscope has different precisions.

D. Drift estimation

Fig. 3 shows that in several cases the visual gyroscope has a frame to frame error greater than zero and this may evoke a huge amount of drift in the estimated yaw after a long integration over previous frames. However, it is obvious in the Fig. 5 that the proposed visual gyroscope does not have such great drift in comparison with the ground truth even for long datasets. In order to investigate this issue in more detail, we provided the histogram of frame to frame relative error instead of absolute error in Fig. 4. This Fig. shows that the noise of proposed visual gyroscope has a zero mean Gaussian distribution. Thus we can conclude that by integration over huge number of frames the amount of error would be decreased due to its Gaussian distribution.

In order to address the accuracy of the proposed gyroscope by considering the Gaussian distribution of its error, we followed a similar methodology called Allan Variance method used for the determination of gyroscopes drift (ex. [29]). Thus we defined several windows with lengths varied from 1 to 40 and for each window we computed the average error of estimated rotation considering all datasets. If we call the average drift for window of length l , $Drift_l$, and assume that we have N frames to analyze and want to compute the

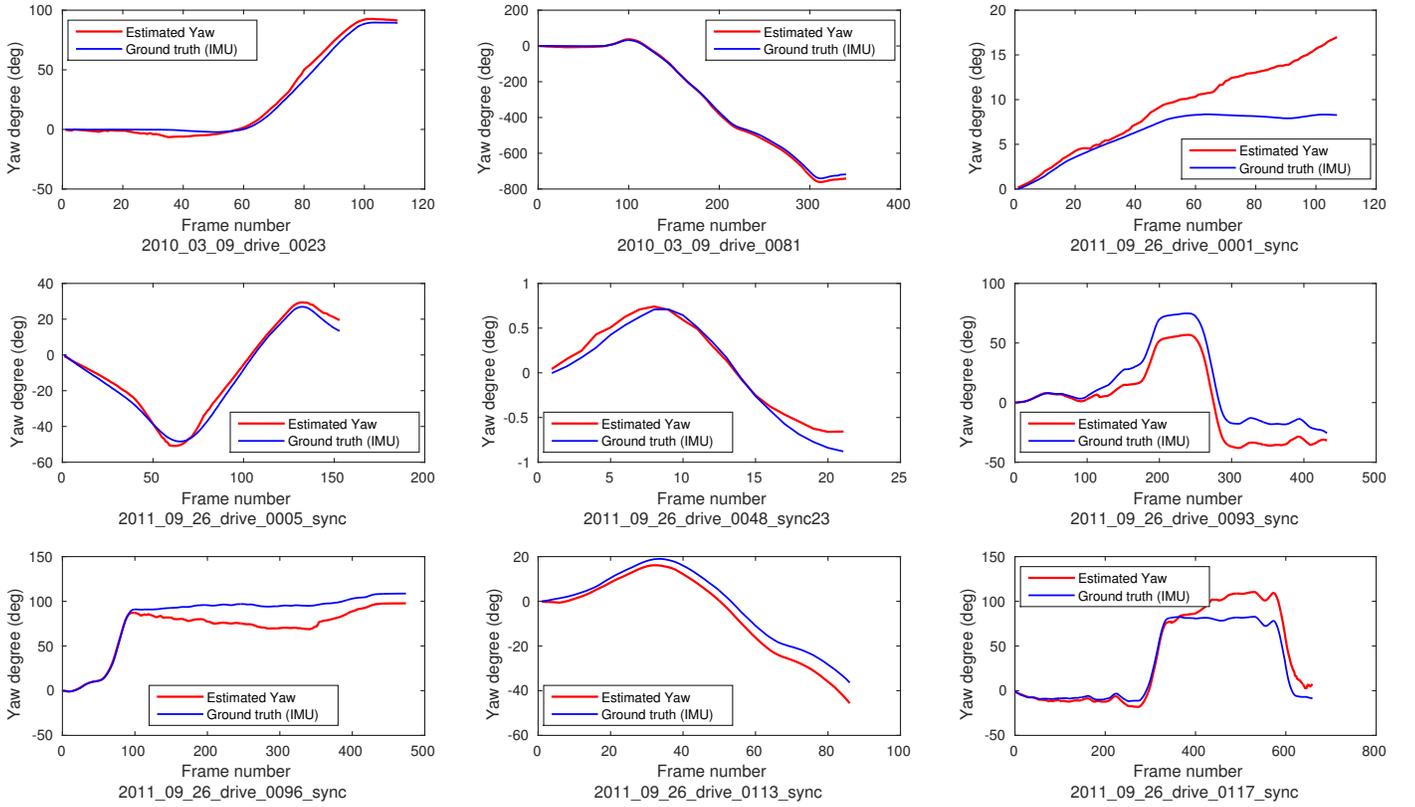


Fig. 5. Estimated and ground truth yaw degree for 9 datasets.

average drift for this window length we can write:

$$Drift_l = \frac{1}{N/l} \sum_{i=1}^{N/l} \sum_{k=i*l}^{(i+1)*l-1} error_frame_k \quad (16)$$

We computed the average error for all window sizes from 1 to 40 and the result of these experiments is shown in fig. 6. It is apparently visible in this diagram that the amount of rotation estimation error is decreased by increasing the length of window in experiments (from about 0.13 to 0.8 deg/frame) and after a while the average error is fixed around 0.08 deg/frame. Hence we can conclude that the amount of drift for the proposed visual gyroscope is about 0.08 deg/frame.

V. CONCLUSION AND FUTURE WORKS

We have introduced a fast method of estimating rotation between two camera frames without considering 3D points or vanishing points. Instead, the proposed method assumes a fixed depth for the majority of matched key points in outdoor environments especially for autonomous cars. Using this presumption, we modified a previously introduced efficient method of rotation estimation for 3D correspondences on our 2D matched key points. We evaluated the accuracy of the proposed method as a fast and accurate visual gyroscope by estimating the yaw degree of a moving car in several datasets from KITTI benchmark. We also investigated other aspects for implementing a real time visual gyroscope such as optimum feature descriptor and number of key points in our computations. Finally we showed that this real time gyroscope can estimate the yaw degree with the rate of 10 frames per

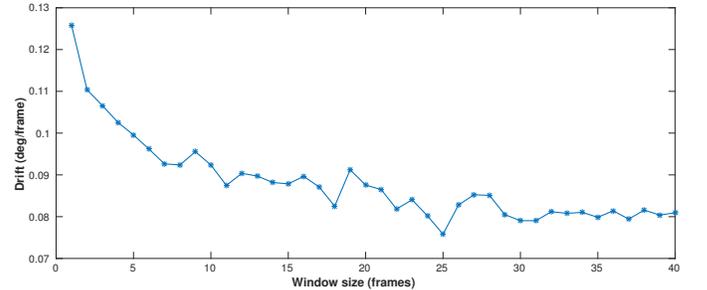


Fig. 6. Estimated drift using different window sizes for the proposed visual gyroscope

second and with the average drift of 0.08 degree per frame. Results of utilizing this real time visual gyroscope on several datasets prove its efficiency and ability to be considered as an independent gyroscope in the future works. The main idea that should be considered in the future works is that efficient approaches of accelerating the speed of the algorithm can lead to utilizing more feature points in rotation computations and finally improve accuracy of the visual gyroscope. Moreover, some vision based techniques such as loop closure detection can also be used in order to inhibit the effect of drift for long time operations.

ACKNOWLEDGMENT

The authors would like to thank Laurent Kneip for his invaluable comments about the paper and also his contributions for evaluating previous rotation estimation methods.

REFERENCES

- [1] C.-X. Li, W.-M. Liu, and J.-W. Wu, "High-precision positioning algorithm based on gps," in *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, vol. 5, 2010, pp. 364–368.
- [2] R. Begg and M. Palaniswami, *Computational Intelligence for Movement Sciences: Neural Networks and Other Emerging Techniques*, 1st ed. Idea Group Publishing, 2006.
- [3] W. Hartmann, M. Havlena, and K. Schindler, "Visual gyroscope for accurate orientation estimation," in *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, 2015, pp. 286–293.
- [4] A. Davison, I. Reid, N. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 6, pp. 1052–1067, 2007.
- [5] C. Wu, "Visualsfm: A visual structure from motion system, <http://ccwu.me/vsfm>, 2011," 2011.
- [6] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, 2004, pp. I-652–I-659 Vol.1.
- [7] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 3946–3952.
- [8] M. Antone and S. Teller, "Automatic recovery of relative camera rotations for urban scenes," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, 2000, pp. 282–289 vol.2.
- [9] V. Huttunen and R. Pich, "A monocular camera gyroscope," *Gyroscope and Navigation*, vol. 3, no. 2, pp. 124–131, 2012.
- [10] C. Kessler, C. Ascher, N. Frietsch, M. Weinmann, and G. Trommer, "Vision-based attitude estimation for indoor navigation using vanishing points and lines," in *Position Location and Navigation Symposium (PLANS), 2010 IEEE/ION*, 2010, pp. 310–318.
- [11] 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 *Ubiquitous Positioning, Indoor Navigation, and Location Based Service (UPINLBS), 2012*, 2012, pp. 1–8.
- [12] D. Martinec and T. Pajdla, "Robust rotation and translation estimation in multiview reconstruction," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, 2007, pp. 1–8.
- [13] L. Kneip, R. Siegwart, and M. Pollefeys, "Finding the exact rotation between two images independently of the translation," in *Computer Vision ECCV 2012*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7577, pp. 696–709.
- [14] L. Kneip and S. Lynen, "Direct optimization of frame-to-frame rotation," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, 2013, pp. 2352–2359.
- [15] L. Kneip and P. Furgale, "Opengv: A unified and generalized approach to real-time calibrated geometric vision," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, pp. 1–8.
- [16] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, 2011, pp. 963–968.
- [17] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision ECCV 2006*, ser. Lecture Notes in Computer Science, A. Leonardis, H. Bischof, and A. Pinz, Eds. Springer Berlin Heidelberg, 2006, vol. 3951, pp. 430–443.
- [18] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [19] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision ECCV 2010*, ser. Lecture Notes in Computer Science, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Springer Berlin Heidelberg, 2010, vol. 6314, pp. 778–792.
- [20] S. Leutenegger, M. Chli, and R. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, Nov 2011, pp. 2548–2555.
- [21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, Nov 2011, pp. 2564–2571.
- [22] A. Alahi, R. Ortiz, and P. Vanderghenst, "Freak: Fast retina keypoint," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, June 2012, pp. 510–517.
- [23] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3-d point sets," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-9, no. 5, pp. 698–700, 1987.
- [24] L. Kneip, "Real-time scalable structure from motion: From fundamental geometric vision to collaborative mapping," Ph.D. dissertation, ETH, 2012.
- [25] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.
- [26] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *Intelligent Vehicles Symposium (IV)*, 2011.
- [27] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian Conference on Computer Vision (ACCV)*, 2010.
- [28] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.
- [29] N. El-Sheimy, H. Hou, and X. Niu, "Analysis and modeling of inertial sensors using allan variance," *Instrumentation and Measurement, IEEE Transactions on*, vol. 57, no. 1, pp. 140–149, Jan 2008.