Sensors & Transducers Journal, Vol.0, Issue 0, Month 2012, pp.

www.sensorsportal.com

Sensors & Transducers

ISSN 1726-5479

http://www.sensorsportal.com

Visual Odometry in Dynamical Scenes

Dong ZHANG and Ping LI

Department of Control Science and Engineering, Zhejiang University, Hangzhou, Zhejiang, 310027 China

Phone: +86-139-8946-3600, Email: zhangs@zju.edu.cn

Received: */Accepted:* */Published:*

Abstract: This paper presents a novel method for visual odometry in dynamical scenes. As an imaging sensor, video camera has a lot of applications by using computer vision methods. Among them, visual odometry is one of the most important applications. However, the traditional method could only work well in static scenes and this is a major drawback which severely limits its application in real-world scenes. The proposed method employs 3-D motion segmentation method to segment the feature point trajectories into different motions, and accurately tracks the background points for camera position and orientation estimation, thus allowing the application of visual odometry method in real world. *Copyright © 2012 IFSA.*

Keywords: Visual odometry, Computer vision, Robot navigation, Structure from motion)

1. Introduction

As an imaging sensor, video camera has a lot of applications by using computer vision methods. Visual odometry (VO) ([1]) is an important computer vision method for robot navigation ([2-4]) and autonomous driving system ([5]). It extracts feature points from the image sequence and employs multiple view geometry approaches to estimate the camera positions and orientations.

However, there are some drawbacks of traditional visual odometry method ([1]) that limits its application in real world, and hinders its integration into electronics products (e.g. cellular phones, autonomous driving system etc.). A major drawback is that traditional visual odometry method could only work well in static scenes (where the only motion in the camera view is the background), and would generally fail in dynamical scenes (where there are some other moving objects in the camera view other than the background). Fig.1 shows a group of sample image frames captured in dynamical scenes. It is very difficult to estimate camera positions and orientations in such scenes since there are some big moving objects in the scenes (e.g. the moving cars). However, the proposed method could

work well while traditional visual odometry method fails.

Fig. 1 Sample image frames captured in dynamical scenes.

The reason why traditional VO method fails in dynamical scenes is that it does not distinguish between the background and the independently moving objects. In this case, there would be feature points extracted both from the background and independently moving objects and traditional VO method use all the feature points to estimate the camera positions and orientations. It is obvious that the feature points on the independently moving objects would affect the estimation results. Traditional VO method use RANSAC ([6]) to eliminate the outliers, so if there are only a small number of feature points extracted from the independently moving objects, RANSAC could help traditional VO method to get the correct estimation. However, if there are a lot of feature points belonging to the moving objects, RANSAC scheme would fail and could not give accurate results. In the extreme cases in which there are more feature points from the moving objects than from the background, RANSAC scheme would lead to totally wrong results (i.e. it would wrongly select the moving objects to be the background). Recently, a good framework for multi-body visual SLAM (Simultaneous Localization and Mapping) ([7, 8]) was proposed to solve such problems for robot navigation applications. However, it employs two-view motion segmentation methods which would induce large errors in real-world applications, and the smoothly moving camera assumption in the framework limits its applications.

The proposed method solves this problem by using background tracking. Generally, in an image sequence, there would be image frames captured both from static scenes and dynamical scenes. First we extract feature points (in this paper, we use SIFT [9] for all the "feature point") from the frames. Then, we use any of the frames from static scenes to initialize a background model (i.e. a feature point set belonging to the background). In applications (e.g. robot navigation), we usually use the first few frames for background initialization. For the frames from the dynamical scenes, we match the feature points of consecutive frames, connect them into trajectories and employ 3-D motion segmentation method ([10]) to segment them into different motions. Since all points from the background share the same motion, we examine feature points from each of the motions to find the motion that have some matches with the previous background model and update the background models with it. This is an online process and it could be continued until the end of the image sequence. In the same time, we use feature points of the background model to estimate the camera positions and orientations, thus we form a robust visual odometry method that could work well in dynamical scenes.

This paper is organized as following: Section 2 introduces the proposed background tracking method; Section 3 gives the algorithm of visual odometry in dynamical scenes; Section 4 shows some results to verify the effectiveness of our method and Section 5 concludes.

2. Method

Background tracking is the main contribution of this paper. It is the background tracking scheme that enables our method to work in dynamical scenes. For background tracking, first we need to initialize a background model and then we need to track and update it. Here the "background model" means the feature point set that belong to the background.

For background model initialization, we need to extract feature points from an image frame that captured in a static scene. Generally speaking, an image sequence would include frames both from static scenes and dynamical scenes, so for an arbitrary image sequence, we can use the frames from static scenes for background initialization. In robot navigation, it is a common practice to include an initialization step for the sensors (e.g. GPS [11]) and it is very easy to do the background initialization for the proposed method in applications. For instance, in robot navigation, the only need is to ensure the first few frames of image are captured in static scenes, and after this, all the navigation is fully automatic.

For background tracking, we need to employ the newly developed 3-D motion segmentation method ([10, 12]). For F consecutive image frames (in the experiments, we use 3 consecutive frames as a group), we extract feature points from them, match and connect them to be feature point trajectories. Assume these trajectories could be segmented into n different motions using 3-D motion segmentation. Let W be the trajectory matrix that $W = [W_1 \quad W_2 \quad \ldots \quad W_n]\Gamma$, in this $\Gamma \in \mathbf{R}^{P \times P}$ is an unknown permutation matrix and

$$W_i|_{i=1}^{n} = \begin{bmatrix} x_{i,(11)} & \cdots & x_{i,(1P_i)} \\ \vdots & \ddots & \vdots \\ x_{i,(F1)} & \cdots & x_{i,(FP_i)} \end{bmatrix} = [X_{i,(*1)} \quad \cdots \quad X_{i,(*P_i)}] = [X_{i,(1*)} \quad \cdots \quad X_{i,(F*)}]^T \tag{1}$$

, where $x_{i,(**)}$ are the feature point locations in the images, P_i is the number of the trajectories of the ith motion, $X_{i,(*j)}$ is the jth trajectory and $X_{i,(k*)}$ is the feature points of the kth frame of the ith motion. The goal of 3-D motion segmentation methods ([13-16]) is to find the permutation matrix Γ thus explicitly get the point trajectories for each motion from W_1 to W_n.

Assume the background models of F^{s-1} frames begin at the $(s-1)$th frame are known. It's obvious that the background models of these F^{s-1} frame could be expressed as

$$W_{back}^{s-1} = [X_{back,(1*)}^{s-1} \,\big|\, \cdots \quad X_{back,(r*)}^{s-1} \,\big|\, \cdots \quad X_{back,(F^{s-1}*)}^{s-1}]^T \tag{2}$$

, in which $X_{back,(r*)}^{s-1}$ is the background model of the rth frame in these frames. For background updating, the goal is to get the background models of F^s frames begin at the sth frame. There are $F_{overlap}^s$ overlapping frames between these F^s frames and previous F^{s-1} frames. Also, there are F_{fresh}^s "fresh" frames, so $F^s = F_{overlap}^s + F_{fresh}^s$. The background models of the F^s frames could be updated using the overlapping frames as following. By using 3-D motion segmentation method (in this paper, we use GPCA [13]), the feature point trajectories in these F^s frames could be segmented into n^s motions $W^s = [W_1^s \quad W_2^s \quad \cdots \quad W_{n^s}^s]\Gamma$. For the overlapping frames, the feature point set $X_{back,(r*)}^s = \{x_{back,(rj)}^s\}|_{j=1}^{P_{back}^s}$ in rth frame that belongs to the new background models would have a nonempty intersection with $X_{back,((r+1)*)}^{s-1}$ (the corresponding frame in the previous F^{s-1} frames), because all the background points have the same motion. In short, it is $X_{back,(r*)}^s \bigcap X_{back,((r+1)*)}^{s-1} \neq \varnothing$.

The proposed method examines from W_1^s to $W_{n^s}^s$ to see which one conforms to the requirement above and obtain the updated background models W_{back}^s. This process could be continued until the end of the image sequence.

In real-world applications, there are noises for the feature point locations, so the result of 3-D motion segmentation is not perfect, thus there would be some wrongly segmented points. Therefore, instead of

examine which motion of W_1^s to $W_{n^s}^s$ have matches with previous background model. We select the motion that with the highest matching rate with the previous model to update the new background model.

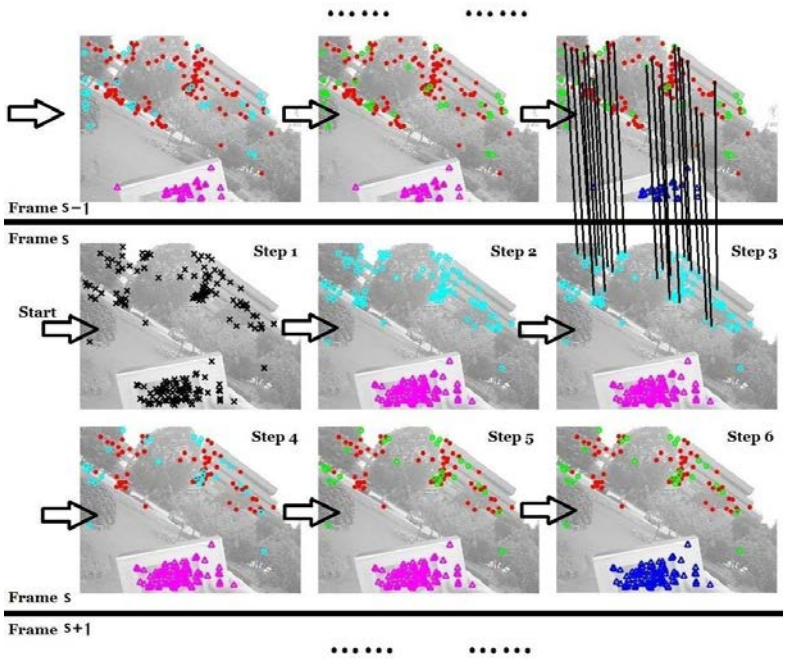

Fig. 2 Background Model Updating Process Details. The first row is the last 3 step of background model updating process of frame (s-1), the second and the third row are the whole process of background model updating of frame s.

Fig.2 shows the details of this background model updating process of one frame in an image sequence. In this figure, the first row is frame $(s-1)$ (in this particular example, $s=6$) and the second and the third row are the background model updating process of frame s. We use these 2 frames to illustrate the detailed process of the background model updating. Suppose we have the background model of frame $(s-1)$ (markers of hollow green circles and solid red circles belong to this category) in row 1 column 3, we want to get the background models of frame s. There are 6 steps: **Step 1** (row 2, column 1), we get the feature point trajectories from a image sequence begins at frame s (in this example frame from 6 to 8), we mark all the trajectories at frame s with black 'x' markers. **Step 2** (row 2, column 2), we use 3-D motion segmentation techniques to segment the feature point trajectories into different motions. In this frame, there are two motions, the cyan circles are one motion and the magenta triangles are another motion. **Step 3** (row 2, column 3), we examine the points in both the motions to see if they have some matches with previous background model begins at frame $(s-1)$. In this case, we do find some matching points (some of them are shown as the linked lines in the figure). **Step 4** (row 3, column 1), these matching points are considered to be a subset of the background points in frame s and marked as solid red circles. **Step 5** (row 3, column 2), as discussed above, we know that point trajectories that with the same motion as the matching points are also a subset of the background points, so we consider them as new background points and marked them as hollow green circles. Thus we got the new background model of frame s. **Step 6** (row 3, column 3), other motions are all considered to be foreground points. This process can continue as long as the image sequence.

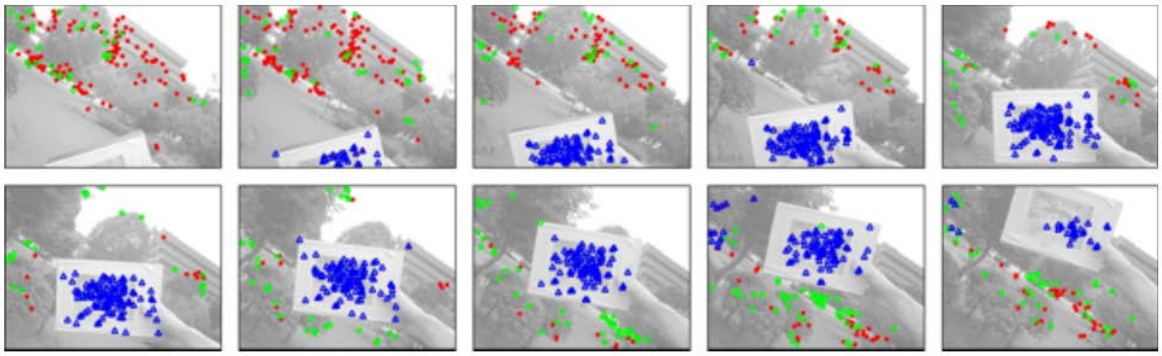

Fig. 3 Background Model Updating Process. Markers of hollow green circles are "fresh" background feature points, markers of solid red circles are matching background points and blue triangles are foreground points. For each updating step, 1 fresh frame is added and 3 frames for a group to extract the point trajectories for 3-D motion segmentation.

Fig.3 shows the updating process of the background models of sample frames of an image sequence. Note that, the background totally changed through the image sequence and there are far more foreground points in some frames but the extracted backgrounds are very accurate.

3. Algorithm

Algorithm 1 shows the complete algorithm.

Algorithm 1: Visual Odometry in Dynamical Scenes

```
// Background Initialization
```
1 Extract feature points from image frames captured in static scenes to initialize a background model.
2 **while** *(not end of the image sequence)* **do**
```
    // Background Tracking
```
3 Extract feature points from a group of consecutive frames.
4 Match these feature points into trajectories.
5 Employ 3-D motion segmentation techniques to segment the trajectories into different motions.
6 Select the motion with the highest matching rate with the previous background model to update the background models.
```
    // Camera Positions and Orientation Estimation
```
7 Online estimate the camera positions and orientations with the background models using multiple view geometry methods
8 **end**

Algorithm 1

4. Experimental Results

The proposed method is tested both for camera position and orientation estimation. The reference positions and orientations are obtained by a DGPS (Differential Global Positioning System) and an INNALABS AHRS (Attitude Heading Reference System) respectively and synchronized with the image sequences.

4.1 Camera Position Estimation

In experiment 1, the image sequence is captured by a moving camera in real-world scenes, in which there is a big independently moving object in the scenes (frame 23 to 31 and frame 108 to 115). Fig.4 shows the results. There are 3 sample frames in the bottom of the figure: the left one is the first frame, the middle one is the 25th and the right one is the 111th frame. Arrows shows the locations of the frames in the estimated trajectories. It is obvious that the estimation of traditional VO method induces huge errors after the frames of dynamical scenes and never come back while the proposed method could work well with only a small accumulative error which is acceptable [1]).

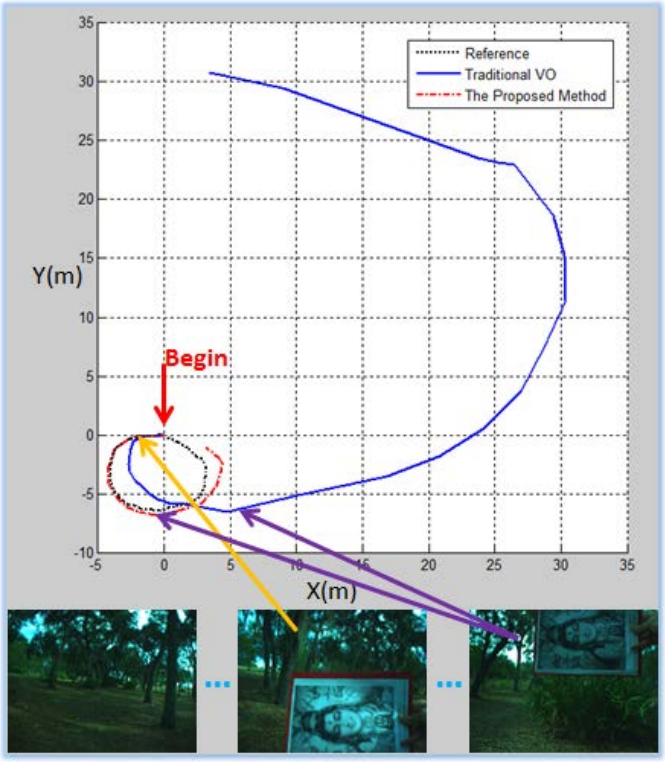

Fig. 4 Experiment 1, camera position estimation for an image sequence captured in real-world scenes. There is an independently moving object in the scenes for frame 23 to 31 and frame 108 to 115.

In experiment 2, the image sequence is also captured by a moving camera in real-world scenes which contains a big moving object from frame 18 to frame 27. It shows in Fig.5 that the proposed method definitely outperform traditional VO method.

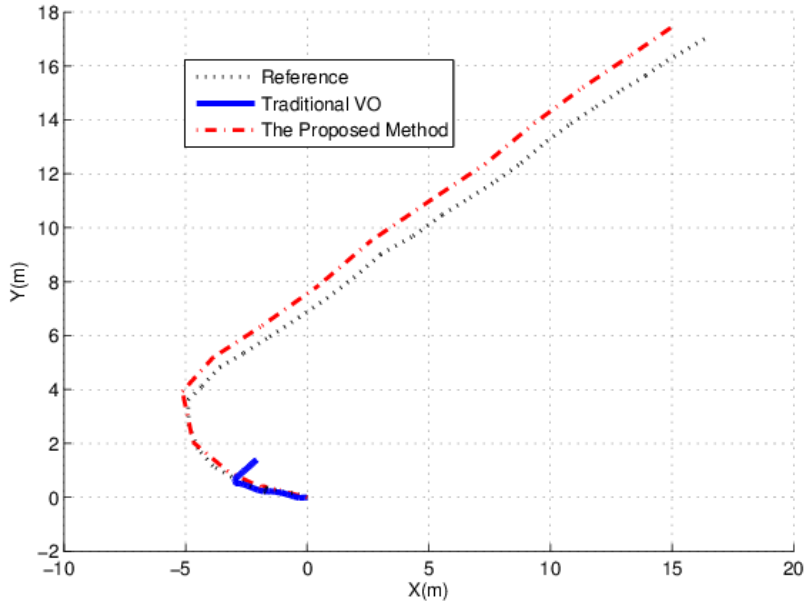

Fig. 5 Experiment 2, camera position estimation for an image sequence captured in real-world scenes. There is an independently moving object in the scenes for frame 18 to 27.

4.2 Camera Orientation Estimation

In the above two experiments, the big independently moving object do not have rotations within the image sequence, so the camera orientation estimations of traditional VO and the proposed method are similar. Due to limited space, here we only show another experiment that traditional VO method also fails in estimation of the camera orientation.

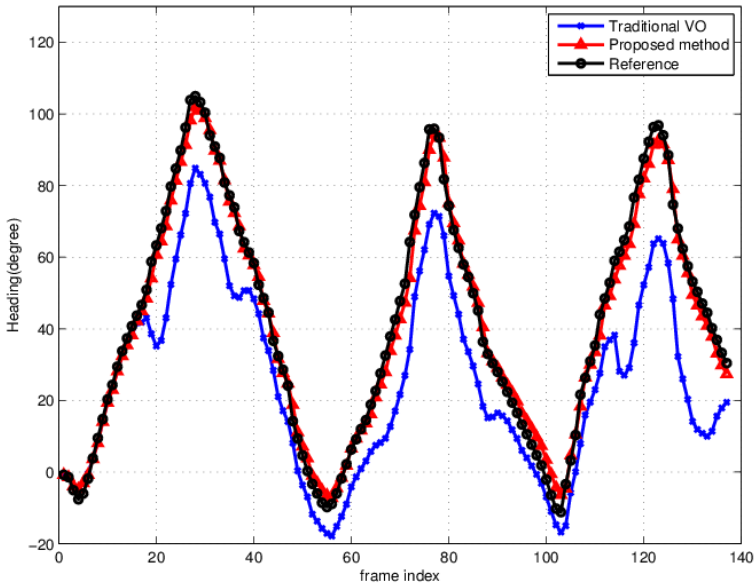

Fig. 6 Experiment 3, camera orientation estimation (heading) for an image sequence captured in real-world scenes.

In experiment 3, there is an object moving (which also includes rotation) in the scenes. It is shown in Fig.6, Fig.7 and Fig.8 that the proposed method could always estimate the camera orientation within a small error range while traditional VO method have huge estimation errors for some of the image frames.

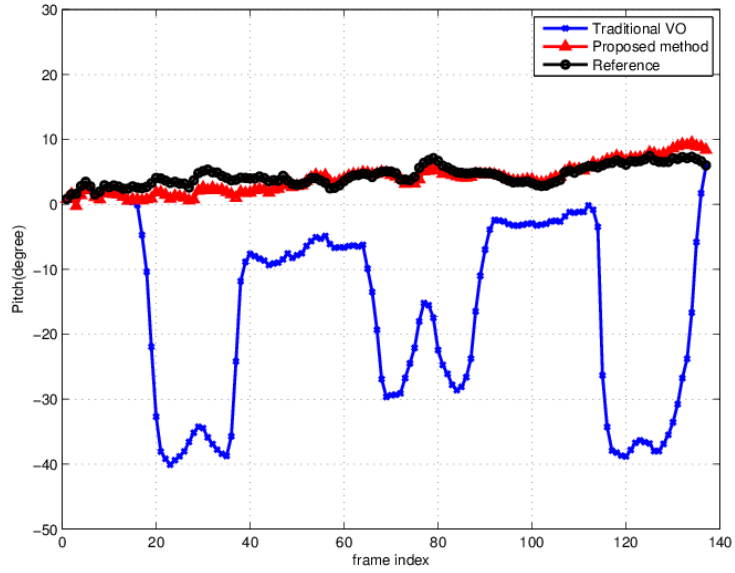

Fig. 7 Experiment 3, camera orientation (pitch) estimation for an image sequence captured in real-world scenes.

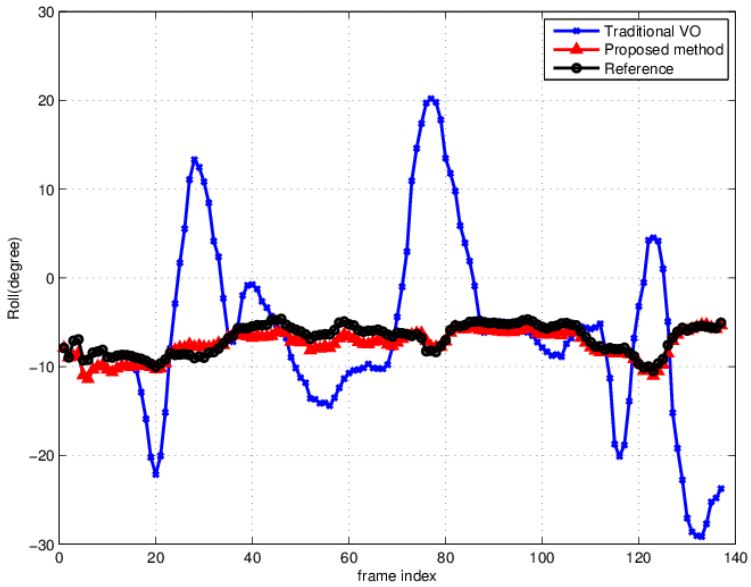

Fig. 8 Experiment 3, camera orientation (roll) estimation for an image sequence captured in real-world scenes.

4.2 Quantitative Comparisons

We also compared the average estimation errors of the proposed method and the traditional visual odometry method in Table.1. These results show that the proposed method has much smaller errors than traditional visual odometry method for estimating the camera positions and orientations.

Table 1 Average Estimation Error Comparison

		Our Method	Traditional VO
Average Position Error (m)	Sequence 1	**0.54**	10.54
	Sequence 2	**0.54**	8.78
Average Orientation Error (º)	Heading	**2.70**	16.47
	Pitch	**1.06**	20.32
	Roll	**0.84**	7.39

5. Conclusions

This paper has proposed an effective method for visual odometry in dynamical scenes. It achieves the robustness by explicitly tracking the background. The experimental results verify that the method is very accurate in real-world scenes while traditional VO method fails.

References

[1] D. Nister, O. Naroditsky and J. Bergen. *Visual odometry*. in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004,

[2] K. Konolige, M. Agrawal and J. Sola, *Large-scale visual odometry for rough terrain.* Robotics Research, vol.66, n.STAR, pp.201--212, 2011

[3] A. Comport, E. Malis and P. Rives, *Real-time quadrifocal visual odometry.* The International Journal of Robotics Research, vol.29, n.2-3, pp.245--266, 2010

[4] C. H. Tseng and D. J. Jwo, *Designing fuzzy adaptive nonlinear filter for land vehicle ultra-tightly coupled integrated navigation sensor fusion.* Sensors and Transducers, vol.128, n.5, pp.1-16, 2011

[5] A. Diosi, S. Segvic, A. Remazeilles and F. Chaumette, *Experimental Evaluation of Autonomous Driving Based on Visual Memory and Image-Based Visual Servoing.* IEEE Transactions on Intelligent Transportation Systems, vol.12, n.99, pp.1-14, 2011

[6] M. Fischler and R. Bolles, *Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography.* Communications of the ACM, vol.24, n.6, pp.381-395, 1981

[7] A. Kundu, K. Krishna and J. Sivaswamy. *Moving Object Detection by Multi-View Geometric Techniques from a Single Camera Mounted Robot*. in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp.4306

[8] S. Wangsiripitak and D. W. Murray. *Avoiding moving outliers in visual SLAM by tracking moving objects*. in *IEEE International Conference on Robotics and Automation*, 2009, pp.375-380

[9] D. G. Lowe, *Distinctive image features from scale-invariant keypoints.* International Journal of Computer Vision, vol.60, n.2, pp.91-110, 2004

[10] R. Tron and R. Vidal. *A benchmark for the comparison of 3-D motion segmentation algorithms*. in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp.1-8

[11] D. J. Jwo and S. H. Wang, *Adaptive fuzzy strong tracking extended Kalman filtering for GPS navigation.* IEEE Sensors Journal, vol.7, n.5, pp.778--789, 2007

[12] S. Rao, R. Tron, R. Vidal and Y. Ma, *Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.32, n.10, pp.1832-1845, 2010

[13] R. Vidal, Y. Ma and S. Sastry, *Generalized principal component analysis (GPCA).* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.27, n.12, pp.1945-1959, 2005

[14] J. Yan and M. Pollefeys, *A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate.* Lecture Notes in Computer Science, vol.3954, pp.94-106, 2006

[15] R. Hartley and R. Vidal. *The multibody trifocal tensor: Motion segmentation from 3 perspective views*. in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004, pp.769

[16] K. Schindler, U. James and H. Wang, *Perspective n-view multibody structure-and-motion through model selection.* Lecture Notes in Computer Science, vol.3951, n.1, pp.606-619, 2006