VISUAL ODOMETRY FOR RGB-D CAMERAS FOR DYNAMIC SCENES

Haleh Azartash, Kyoung-Rok Lee and Truong Q. Nguyen

ECE Department, UC San Diego, La Jolla CA

ABSTRACT

In this paper, we propose an accurate estimation of the camera motion in a dynamic environment from RGB-D videos. To better exclude the moving object portion of the scene from the stationary background, we use image segmentation. Next, dense pixel matching between the current and reference color images is performed to construct the 3D point cloud for dense motion estimation. At the end, we perform motion optimization, i.e., to find the combination of motion parameters that minimizes the remainder difference between the reference and the current image. We validate our proposed method across two benchmark sequences and show that our approach is more accurate than the existing solutions. We show that our method reduces the RMSE by 6.55% and 7.16% for stationary and dynamic scenes, respectively.

Index Terms— Visual odometry, Dynamic scene, ICP, Segmentation, motion optimization

1. INTRODUCTION

The problem of estimating a moving rig's motion has been studied for the past three decades. Several methods have been developed for this purpose, such as wheel odometry, global positioning system (GPS), inertial measurement units (IMUs) and more recently, visual odometry (VO). Visual odometry is a term defined by Nister [1] which explains the process of estimating the relative pose of a moving vehicle using only the images from one (or more) camera(s) mounted on the vehicle between different time intervals. There have been many studies on visual odometry for stationary environments using monocular, stereo [2, 3, 4, 30, 9], and more recently, RGB-D [5, 8] cameras including optical flow-based and sparse feature-based methods. Another popular method is different varieties of iterative closest points (ICP) where the corresponding points in the two consecutive images are iteratively found. In [11], an energy-based method based on minimizing the underlying backprojection error is presented. Despite the vast pool of existing methods for stationary environments, the dynamic environment where in addition to the camera motion, the objects in the scene are also moving is far less explored. In this paper, we present a novel method to estimate the motion of an RGB-D camera in a non-stationary environment using image segmentation and dense ICP motion estimation for each segment. The main purpose of using image segmentation is to cluster the correlated image pixels into distinct segments in order to separate the moving object segment from the stationary background. Even though there is no guarantee that the moving object is categorized into a single segment, based on the assumption that the moving object only occupy a small portion of the image with respect to the background, it is expected that the majority of the segments belong to the stationary part of the scene. The main contributions of the proposed method are as follows.

- We use image segmentation to separate the moving part of the scene from the stationary part.
- We perform the motion estimation per segment, hence the pixels from the moving object will be occluded from the final motion estimation
- We optimizing the motion parameters by minimizing the remainder error using the motion parameters of each segment.

The remainder of the paper is organized as follows. In Section 2 the related work is reviewed. Section 3 shows the problem formulation and the details of the proposed algorithm is presented in Section 4. The results are shown in Section 5.

2. RELATED WORK

The majority of the existing works on visual odometry focused on ground and micro-air vehicles for stationary scenes [14, 27, 24, 10]. The task is performed by estimating the rigid body transformation between two images at consecutive time intervals. One of the most popular methods for VO is feature tracking accompanied by an outlier removal method, e.g., RANSAC [16, 17] and graph-based consistency algorithm [15]. The advantage of the feature tracking method is that it only uses a fraction of total number of pixels per frame for motion estimation which significantly reduces the computation time. However, interest points in the image are mostly at the edges or corners of the objects in the scene. In RGB-D cameras, if the objects are close to the camera, the depth information at the borders of the objects are lost due to parallex between the color and infrared camera, therefore, several of the selected feature points do not have a corresponding depth information. Thus, the number of suitable feature points reduces dramatically which in turn result in inaccurate estimation. Dense estimators, however, use the information of the whole image for motion estimation [27, 11, 12]. Steinbrucker et al. [11], uses a direct method which aligns images together to estimate VO using the color and depth images obtained from Microsoft Kinect sensor for stationary scenes. An alternative dense method is to use ICP algorithm to align 3D point clouds [28, 29] which is used in our proposed algorithm. In the ICP method, first the 3D point clouds of the corresponding points in the current and reference images are constructed.

This work is supported in part by NSF grant CCF-1065305 and by the Technology Development Program for Commercializing System Semiconductor funded by the Ministry of Knowledge Economy (MKE, Korea). (No. 10041126, Title: International Collaborative R&BD Project for System Semiconductor).

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

Then, these two point clouds are iteratively aligned to find the nearest neighbor between them.

In contrast to static VO, dynamic visual odometry has not been investigated extensively. Due to the fact that many of the aforementioned methods assume the motion to be rigid body motion, moving objects within the scene cause erroneous motion parameters. This becomes more important since in many applications of VO, estimation error will propagate to the trajectory of the overall motion for simultaneous localization and mapping (SLAM). In [31], Kitt et al. proposed a method for VO for dynamic environment using feature classification on stereo images. However, in this method, the classifier needs to be learned in advance which includes hand labeling training examples. Kerl et. al [5] proposed a direct motion estimation method with a probabilistic formulation where a motion prior based on a constant velocity model is used.

3. PROBLEM FORMULATION

The inputs of our proposed method are two pairs of image, (I_k, D_k) and (I_{k+1}, D_{k+1}) obtained by a RGB-D camera (i.e., Microsoft Kinect). I_k and D_k represent the color (RGB) and the corresponding depth map at time k, respectively. The camera motion is defined by a 3×3 rotation matrix, $\mathbf{R} \in SO(3)$ and a 3×1 translation vector, $\mathbf{t} \in \mathbb{R}^3$. For each 3D point in the scene at time k, $P_k(X, Y, Z)$, the corresponding point in the image plane is $p_k(x, y)$ as shown in Eq. (1).

$$\begin{aligned} Z &= D(x, y) \\ (X, Y, Z)^\top &= \left(\frac{(x + c_x) \cdot Z}{f_x}, \frac{(y + c_y) \cdot Z}{f_y}, Z\right)^\top \end{aligned} \tag{1}$$

Here, f_x and f_y are camera's focal length and (c_x, c_y) is the principal point of the camera. We define $\mathbf{T}$ as the transformation between the images at two consecutive time intervals which is shown in Eq. (2).

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \overline{\mathbf{0}} & 1 \end{bmatrix} \tag{2}$$

where $\overline{\mathbf{0}}$ is a 1×3 zero vector. The camera motion for each 3D point can be described by Eq. (3) where $[P \quad 1]^\top$ is the homogenous coordinate of point P.

$$\begin{bmatrix} P_{k+1} \\ 1 \end{bmatrix} = \mathbf{T} \begin{bmatrix} P_k \\ 1 \end{bmatrix} \tag{3}$$

The goal is to find the transformation matrix that accurately describes the motion of the camera.

4. VISUAL ODOMETRY FOR RGB-D

In this section, the details of the proposed method is presented. Our approach consists of the following steps.

- *Image segmentation*: The input image is divided into n segments in order to separate the moving parts of the scene from the stationary part. The number of segments is decided based on the similarities between element within a segment and dissimilarities between the neighboring segments.
- *Dense 3D point cloud construction*: To construct the 3D points, the pixels of the reference and current frames are matched. The coordinates of the corresponding points and the depth information from the depth map images (D_k) construct the 3D point cloud. The object boundary points with no depth information are ignored in point cloud construction.
- *Motion estimation*: The motion of each segment is estimated using the ICP method. The obtained motion parameters ($\mathbf{R}_i$ and $\mathbf{t}_i \;\; i = 1, \cdots, n$) are sorted based on the number of the 3D points used for motion estimation.
- *Optimization of the motion parameters*: To find the true motion parameter from the R_i and T_i, the current image is warped by each transformation from the segments. The final motion parameter set is the combination of the motion parameters that minimizes the remainder difference between the reference image and warped current image.

4.1. Image Segmentation

In order to separate the motion of the moving object from the rest of scene, the input RGB images, I_t, are first segmented. There are several different image and video segmentation methods available [18, 19, 20]. In this paper we apply the segmentation method presented in [18] by Felzenszwalb et al. which only uses the spatial information of the RGB images. This method uses graph-based representation of the image as a predicate for measuring the evidence for a boundary between two regions. The predicate is based on the similarity between the elements within a segment and dissimilarity between the elements of the neighboring segments. For the segmentation method one can set different parameters that define the minimum size of the segments or the dissimilarity measurement between the two neighboring segments. After Segmentation, if the moving object is segmented into multiple segments rather than a single one, based on the assumption that the moving object is small with respect to the stationary part of the scene, it is highly expected that we have segments that only contain the background (stationary) pixels. Note that this depends on the minimum size of the segments. In Section 5, we show the values that we have used in our implementation.

4.2. Dense 3D Point Cloud Construction

After segmentation, corresponding points between all pixels in each segment in I_t and pixels in I_{t+1} should be computed. The similarity measure for the matching process is normalized cross correlation (NCC) as shown in Eq. (4).

$$\frac{\sum_{(i,j) \in W} I_1(i, j) \cdot I_2(x + i, y + j)}{\sqrt[2]{\sum_{(i,j) \in W} I_1^2(i, j) \cdot \sum_{(i,j) \in W} I_2^2(x + i, y + j)}} \tag{4}$$

Due to the arbitrary nature of the camera motion, the search area to find the corresponding point in the current image (with maximum NCC value) is a square window instead of the horizontal scanline used in the rectified images. Once the corresponding points are established, the depth images, D_k and D_{k+1} are used to construct the 3D point cloud. Note that, the

points where depth information is not available are discarded from the point cloud.

4.3. ICP motion estimation

The iterative closest point (ICP) algorithm is widely used for 3D shape alignment problem. The main goal of the method is to estimate the relative pose between two 3D sets of points. The ICP was introduced by Chen and Medioni [21] and Besl and Mckay [22] and many ICP variants have since been proposed [28]. In the algorithm, the relative transformation between two point clouds is iteratively estimated. More specifically, corresponding pairs in overlapped area are recovered using initial transformation based on geometric position. Then, the relative rotation and translation are estimated by minimizing error metric based on Euclidean distance between each corresponding pair. The refined transformation parameters are then used in the next iteration. These processes are repeated for a specific number of iterations. Due to the fact that the camera motion between each time interval is small in our experimental results, we used identity matrix and zero vector for initial rotation matrix and translation vector respectively. However, if the camera's motion is fast, the previous frame motion is used as the initial motion parameters.

4.4. Motion Optimization

Once the motion parameters for each segment is computed, the final motion of the frame must be determined. In order to find the final motion parameters, in the proposed method we use a combination of all motion parameter candidates, $\mathbf{R}_i$ and $\mathbf{t}_i$ $i = 1, \cdots, n$, that minimizes the difference between the reference frame and the current warped frames (maximizing the photoconsistency) as shown in Eq. (5). The main reason for using the linear combination of the motion parameters is that in the segmented image, there are multiple segments from the stationary sections of the scene that have similar size (number of valid pixels in the point cloud) and similar (not identical) motion parameters. By applying the linear combination we find the best motion parameters that maximizes the photoconsistency. In order to increase the accuracy of the results, the motion parameters are sorted based on the (a) the number of 3D points in the point cloud of each segment and (b) the remainder error from the ICP. The estimations from the smallest point cloud and largest remainder error are removed from the optimization process.

$$e_i \;=\; [I_k - I_w^i]^2 \tag{5}$$

where I_w^i is the current warped image using transformation $\mathbf{T}_i$ from Eq. (2) constructed from $\mathbf{R}_i$ and $\mathbf{t}_i$ $i = 1, \cdots, n$. In order to obtain the current warped frame, I_{k+1}, in 2D coordinate using the 3D motion parameters, the 3D point cloud of the current image is constructed using the depth information, D_{k+1}. Then, the obtained point cloud is transformed using $\mathbf{T}_i$. The transformed 3D point cloud is shown as $\mathbf{T}_i(I_{k+1})$. Then, $\mathbf{T}_i(I_{k+1})$ is reprojected into image plane using Eq. (6) and the camera parameters.

$$\begin{aligned} p_i(x,y) \;&=\; G(P_i(X,Y,Z)) \\ p_i(x,y) \;&=\; (\frac{P_{ix} f_x}{P_{iz}} \;-\; c_x \;,\; \frac{P_{iy} f_y}{P_{iz}} \;-\; c_y)^\top \end{aligned} \tag{6}$$

where P_x, P_y and P_z are the x, y and z coordinates of the points in the warped 3D point cloud and $p(x, y)$ is the 2D points in the reprojected image plane. The set of all $p(x, y)$ creates I_w^i. The final transformation is a linear combination of the $\mathbf{T}_i$ obtained for each segment. The best combination is obtained by minimizing the final residual error as shown in Eq. (7).

$$\begin{aligned} e \;&=\; [I_k \;-\; \Sigma\alpha_i \; I_w^i]^2 \\ &\Sigma\alpha_i = 1 \end{aligned} \tag{7}$$

where α_i are the appropriate weights assigned to the transformation for each segment that minimizes residual error. The minimization process is shown in Eq. (8). This results in the correct motion parameters having a higher weight than those from the moving object due to the assumption that the moving object occupies a small portion of the scene.

$$\begin{aligned} \min_{\alpha} e \;&= \min_{\alpha_i} \; [I_k \;-\; \Sigma\alpha_i \; I_w^i]^2 \\ \min_{\alpha} e \;&= \min_{\alpha_i} \; [I_k \;-\; \Sigma\alpha_i \; \mathbf{T}_i(I_{k+1})]^2 \end{aligned} \tag{8}$$

The final transformation between I_k and I_{k+1} is obtained as shown Eq. (9).

$$\mathbf{T} \;=\; \Sigma\alpha_i \; \mathbf{T_i} \tag{9}$$

5. EXPERIMENTAL RESULTS AND DISCUSSIONS

5.1. Experimental setup

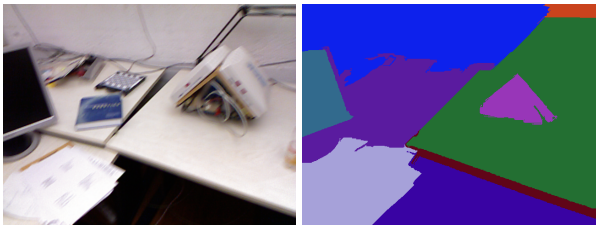
In this section, the comparison results of the proposed method and the method developed by Engelhard et al. [25] is presented. In our experiments, we use five different sequences from the benchmark database [26]. The evaluation is done using the online tool from the benchmark database. In this paper we show the results for only two sequences, $fr1\backslash desk2$ (stationary sequence) and $fr1\backslash desk2\backslash withPerson$ (dynamic sequence). [1] Note that in this database the camera is calibrated and the color and depth image are both rectified and synchronized (the images are from the same time stamp). Therefore, the corresponding depth information of each pixel ($p(x, y)$), is stored in the same pixel location ((x, y)) in the depth image, $D(x, y)$. There are no prior assumptions regarding the camera motion (such as the velocity or direction of the camera), the only assumption is that the moving parts of the scene in the dynamic sequence are small with respect to the image size. The resolution of both sequences are 640×480. For comparison, the root mean square error (RMSE), the average error (mean) and the standard deviation (std.) of the obtained results from the proposed method and other method are compared with the ground truth results from the database. In the comparison process, the overall camera trajectories are compared to each other (not frame-by-frame). Note that in our proposed method, no drift reduction method is used which would improve the results significantly.

5.2. Experiments

We evaluate the accuracy of the proposed method using three benchmark sequences and compare the results with the

[1] The additional results are shown at our website (`http://videoprocessing.ucsd.edu/~haleh/Haleh_Azartash/Research.html`).

ground truth and the existing method by Engelhard et al. [25]. In these experiments, the error between the estimated motion and the reference ones is measured in m/s. For each segment, we present the results for different segmentation option for the minimum size of the segment, S_{min} $(1/100)\times$, $(1/200)\times$ and $(1/300)\times$ frame size of 640×480 . Note that having a smaller S_{min}, the number of segments in each image would increase. The effect of different S_{min}'s is shown in Figure 1 and Figure 2 for stationary and dynamic environments, respectively. As shown in Figure 1d, the smaller value of S_{min}

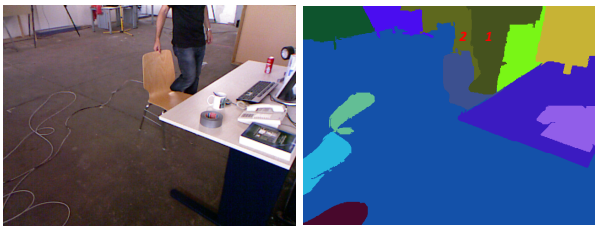


(a) Original frame (b) S_{min} = 3072

(c) S_{min} = 1536 (d) S_{min} = 1024

Fig. 1: The effect of minimum segmentation size

yields small segments in the image (outline by the red line). In the motion optimization process where S_{min} is 1024, α_i for the small blobs are much lower than the larger segments. This is due to the fact that the larger segments have more points in the 3D point cloud which results in a more accurate motion estimation. In Figure 2, the person is moving in the



(a) Original frame (b) S_{min} = 3072

(c) S_{min} = 1536 (d) S_{min} = 1024

Fig. 2: The effect of minimum segmentation size

frame and he occupies a small portion of the scene. As shown in Figure 2b-Figure 2d, the persons' silhouette is confined in two segments, Segment 1 and Segment 2 in Figure 2b. As expected, the α_i values for these two segments are lower than that of other segments. Also, the smaller blobs in Figure 2c and Figure 2d has lower α_i than those of larger segments.

According to the above discussion, it is shown that even though the minimum value for the segment size results in different segmentation, its effect on the final motion parameters is reduced by the optimization step for both static and dynamic scenes. Also, note that even though the area of the moving person was not restricted into one segment, since the transformation $\mathbf{T}_i$ for these segments was not aligned with the rest of the image, the weight of these two segments in the final $\mathbf{T}$ were insignificant.

The comparison results are shown in Table 1 and Table 2. In the table the results of the proposed method for three different values of S_{min} are compared to the method proposed in [25]. The table elements indicate the RMSE, mean and std. values compared to the ground truth. For the stationery scene, as shown in Table 1, comparing the results for different S_{min} shows that for this sequence, since there is no motion in the scene, the larger segmentation size has more accurate value.

Table 1: Motion estimation results for stationary scene

Segment Size	RMSE	Mean	Std.
S_{min} 1024	0.012899	0.012941	0.006854
S_{min} 1536	0.01457	0.013086	**0.006545**
S_{min} 3072	**0.014181**	**0.012803**	0.0067002
Engelhard et al. [25]	0.015175	0.013485	0.006961
Error Reduction%	6.55	5.06	5.97

Table 2: Motion estimation results for dynamic scene

Segment Size	RMSE	Mean	Std.
S_{min} 1234	0.009762	0.008521	0.006960
S_{min} 1536	0.096350	0.00898	**0.005784**
S_{min} 3072	**0.010649**	**0.0089379**	0.006084
Engelhard et al. [25]	0.011470	0.009643	0.006210
Error Reduction%	7.16	7.31	2.03

6. CONCLUSION

In this paper, we presented a new approach for visual odometry for dynamic environments. In the proposed approach, we use image segmentation to separate the moving parts of the scene from the static background. Then, dense motion estimation using ICP is performed for each segment. In order to find the final motion, the current image is warped with each candidate motion parameters from all segments and then the linear combination of transformation that minimizes the difference between the reference and current images is chosen as the final motion transformation. The motion optimization process assigns proper weights to each motion parameter candidate. The experimental results show that the moving part of the scene are isolated to a few segments and the assigned weights for those segments are very small. We show that our proposed method reduces the RMSE by 6.55% and 7.16% compared with those of [25] for stationary and dynamic scenes, respectively. Currently, the only drawback of our proposed method is the execution time. For our future work, we plan to implement the proposed method in real-time in order to make it suitable for robot navigation.

7. REFERENCES

[1] D. Nister, O. Naroditsky and J. Bergen, "Visual Odometry", *Proc. CVPR*, 2004, pp. 652–659.

[2] D. Scaramuzza and F. Fraundorfer, "Visual Odometry: Part I - Matching, Robustness, and Applications", *Robotics Automation Magazine*, 18(4), 2011.

[3] D. Scaramuzza and F. Fraundorfer, "Visual Odometry: Part II - The First 30 Years and Fundamentals", *Robotics Automation Magazine*, 19(1), 2012.

[4] K. Konolige, M. Agrawal, R. Bolles, C. Cowan, M. Fischler, and B. Gerkey, "Outdoor mapping and navigation using stereo vision", *Proc. ISER*, 2007.

[5] C. Kerl and J. Sturm and D. Cremers,"Robust Odometry Estimation for RGB-D Cameras", *Proc. ICRA*, 2013.

[6] S. Weiss, M. Achtelik, M. Chli, and R. Siegwart, "Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV", *Proc. ICRA*, 2012.

[7] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter", *Proc. IROS*, 2012.

[8] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera", *Proc. ISRR*, 2011.

[9] A. Geiger, J. Ziegler and C. Stiller, "StereoScan: Dense 3d reconstruction in real-time", *Proc. Intelligent Vehicles Symposium*, 2011.

[10] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the RGB-D SLAM system", *Proc. ICRA*, 2012.

[11] F. Steinbrucker, J. Sturm, and D. Cremers, "Real-time visual odometry from dense RGB-D images", in Workshop on Live Dense Reconstruction at *ICCV*, 2011.

[12] C. Audras, A. Comport, M. Meilland, and P. Rives, "Real-time dense RGB-D localisation and mapping", *Proc. ICRA*, 2011.

[13] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "3-D motion and structure from 2-D motion causally integrated over time: Implementation", *Proc. ECCV*, 2000.

[14] F. Bonin-Font, A. Ortiz and G. Oliver,"Visual Navigation for Mobile Robots: A Survey", Journal of Intelligent and Robotic Systems 53(3), 2008, pp. 263–296.

[15] A. Howard," Real-time stereo visual odometry for autonomous ground vehicles", emphProc. IROS, 2008.

[16] A. E. Johnson, S. B. Goldberg, Y. Cheng and L. H. Matthies,"Robust and efcient stereo feature tracking for visual odometry", *Proc. ICRA*, 2008.

[17] K. Ni, H. Jin and F. Dellaert ,"GroupSAC: Efficient consensus in the presence of groupings", *Proc. ICCV*, 2011.

[18] P. F. Felzenszwalb and D. P. Huttenlocher."Efficient Graph-Based Image Segmentation", *International Journal of Computer Vision*, 59(2), 2004.

[19] M. Grundmann, V. Kwatra, M. Han and I. Essa,"Efficient Hierarchical Graph Based Video Segmentation", *Proc. CVPR*, 2010.

[20] J. Lezama, K. Alahari, J. Sivic and I. Laptev,"Track to the Future: Spatio-temporal Video Segmentation with Long-range Motion Cues", *Proc. CVPR*, 2011.

[21] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images", *Proc. ICRA*, 1991.

[22] P.J. Besl and N. McKay, "A method for registration of 3-D shapes", *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(2), 1992.

[23] S. Rusinkiewicz and M. Levoy,""Efficient Variants of the ICP Algorithm", *Proc. 3DIM*, 2001.

[24] J. Stuhmer, S. Gumhold, and D. Cremers, "Real-time dense geometry from a handheld camera", *Proc. DAGM*, 2010.

[25] N. Engelhard, F. Endres, J. Hess, J. Sturm and W. Burgard," Real-time 3D visual slam with a hand-held camera", *Proc. of the RGB-D Workshop on 3D Perception in Robotics*, 2011.

[26] http://vision.in.tum.de/data/datasets/rgbd-dataset.

[27] A. Comport, E. Malis, and P. Rives,"Accurate Quadri-focal Tracking for Robust 3D Visual Odometry", *Proc. ICRA*, 2007.

[28] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm", emphProc. 3DIM, 2001.

[29] J. Stückler and S. Behnke, "Model learning and real-time tracking using multi-resolution surfel maps", *Proc. AAAI*, 2012.

[30] H. Azartash, N. Banai and T. Q. Nguyen, "Integrated Stereo Ego Motion Estimation Framework With Sparse Depth Map", *Proc. IEEE ATC*, 2012, pp. 124–129.

[31] B. Kitt, F. Moosmann and C. Stiller, "Moving on to dynamic environments: Visual odometry using feature classification", *Proc. IROS*, 2010.