Combination of Planes and Image Edges for a Visual Odometer for Weakly Textured Structured Environments

Joan P. Company-Corcoles[1] and Alberto Ortiz[1]

Abstract— **Camera trajectory estimation has generated a lot of interest during the last decades, especially for robotic positioning. It is well-known that outdoors positioning mainly relies on GPS, whereas one of the main used methods in indoor positioning is visual odometry. As it is well known, visual odometers get typically in trouble when the environment is weakly textured. Facing this situation, this paper develops and tests a novel visual odometer that combines image edges and planar information to estimate the trajectory of an RGB-D camera in environments that lack texture. We also present a plane matching method based on a graph matching technique. To conclude, a comparison of the proposed odometer for two well-known datasets and other visual odometers and SLAM systems is reported. The comparison shows our method as more accurate as for the estimation of the position in indoor places where visual features are poor, while similar values are obtained for other indoor environments.**

I. INTRODUCTION

In recent years, there has been a great advance in the development of Simultaneous Localization and Mapping techniques (SLAM). Regarding localization, it comprises two main tasks, odometry and loop closing. When a new camera frame is received, the odometry is responsible for estimating the new camera pose by calculating the transformation between frames. At the same time, the loop closing takes care of finding if the current frame has been previously seen. Accumulating the transformations along time make the camera position error grow continuously. When a loop closure is detected, the accumulated odometry error is reduced because of the introduction of a constraint between the current frame and its correspondence. It is very important that during all the odometry process the error between frames keeps as low as possible and the tracking does not get lost. These odometry problems usually appear in state-of-the-art methods in structured indoor environments lacking texture.

Among the variety of current sensors for visual odometry, we employ an RGB-D sensor since it is a light and cheap device able to extract a color and a depth image in featureless areas, where the monocular and stereo sensors are not able to extract depth.

Visual odometers can be classified on the basis of how the tracking stage is implemented. Currently, most odometers are based on keypoints. To obtain the transformation between frames, this method detects and describes keypoints in two images and match them [1]. Since in weakly textured environments the number of keypoints is low, the estimation tends to be imprecise. Another important tracking technique is direct image alignment [2]. This method determines the image alignment that makes most of the pixel intensity agree. This approach does not work properly either in untextured places or where the illumination changes between frames. Another important tracking method is the iterative closest point (ICP) [3]. Unlike the previous methods, which are based on decreasing the photometrical error, ICP reduces the geometrical error. This method iteratively approaches each 3D point from one point cloud to the closest 3D point of the other point cloud. This method requires either a good initial guess transformation between frames or that the two 3D point clouds are sufficiently well-aligned.

During last years, different techniques to register frames using edges, lines and planes have been developed. These structures are abundant in a structured environment, what leads to better tracking results in this kind of environments despite the absence of texture. Using these features individually have some drawbacks, while combining them with other features tend to improve the registration performance. Some combined methods use lines and keypoints [4] [5], lines, keypoints and planes [6] [7], keypoints and planes [8] [9], semi-direct methods and planes [10], direct methods and edges [11], points with planar patches [12] and edges with depth maps [13].

The first contribution of this work is a visual odometer that combines information of planar primitives and the edges extracted from the image, where both of them are still present in structured environments lacking texture. Furthermore, another advantage of edges is that they are less affected by illumination changes between frames because edge extractors use differential information between pixel intensities. On the other hand, the plane abstraction counteracts the depth noise introduced by RGB-D cameras. Moreover, the number of planes obtained from a depth image is lower than the number of keypoints, what means less storage as well as less computational resources.

Our second contribution is a plane matching method based on graphs which is used by the proposed odometer in order to find plane correspondences between frames. Most current plane matching methods first extract the transformation between frames by other techniques and then assume that a plane correspondence exist [10]. However, if the transformation estimation fails, i.e. the match is non-existent or is wrong, the registration process also fails. Another kind

This work is partially supported by projects ROBINS (EU-H2020, GA 779776), PGC2018-095709-B-C21 (MCIU/AEI/FEDER, UE), PROCOE/4/2017 (Govern Balear, 50 % P.O. FEDER 2014-2020 Illes Balears) and the scholarship BES-2015-071804

[1] Corresponding Author: Joan P. Company-Corcoles, UIB - University of the Balearic Islands, 07122 - Palma de Mallorca, Spain; E-mail: joanpep.company@uib.es.

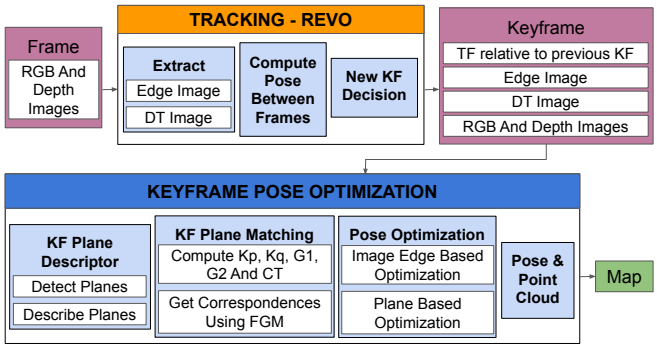

Fig. 1: General view of the proposed system.

of technique for plane matching obtains the transformation and the correspondences by Random Sample Consensus (RANSAC) [8]. For this method, it is necessary to have at least three planes whose normal spans $\mathbb{R}^3$, what sometimes is a constraint difficult to satisfy in these environments.

The plane matching algorithm most similar to ours is described in [14], where the authors use an interpretation tree where they search for plane correspondences among the planar patches of two images. In this graph, the authors represent both planar features and the planar relationships with neighbouring planar patches. Regarding this method, we introduce a different graph matching technique, as well as a different method to represent the graph associated to each image.

We report on extensive experiments for two well-known SLAM and odometers evaluation datasets to assess our approach. These experimental results show that our approach improves the accuracy of the camera trajectory in weakly textured environments, while it produces similar results on other environments.

II. General view of the system

The information about the environment is introduced in our system as camera frames, one colour image and one depth image. Next, each frame is processed by the Robust Edge-Based Visual Odometry (REVO) algorithm [15], which extracts edges, detects keyframes and estimates the transformation between keyframes. When a new keyframe is generated, we extract a coloured point cloud by combining the information from the colour and the depth images, detect [16] and describe the existing planes, and match them using our plane matcher. A refined camera transformation between keyframes is obtained after an optimization process that combines edges information and planar features. Finally, the refined pose and the generated point cloud are introduced in a map which represents the environment.

II-A. Plane description and matching

Given two graphs, where one encodes the planar information of the reference keyframe Kf_r and the other encodes the planar information of the current keyframe Kf_c, the graph matching module determines the correspondences between them. In order to find these correspondences, we use the Factorized Graph Matching (FGM) algorithm described in [17]. The authors present the FGM in a keypoint matching application, hence we have adapted it to work with planes. To compare two graphs and to obtain their correspondences, we compute two affinity matrices that represent the similarities between the graph nodes and the graph edges, which are called Kp and Kq respectively. The topology of the graph, which is represented by the starting and the ending node of each graph edge, is contained in an incidence matrix, where G_1 corresponds to the Kf_r graph and G_2 to the Kf_c. The possible matching candidates between graphs are represented in the Ct matrix.

In our system, the Kp matrix encode the colour similarities of each plane of the Kf_r with each plane of the Kf_c. The colour distribution of each plane is represented by a 3x16 histogram, one block for each RGB channel. We use the Bhattacharya distance to measure histogram dissimilarity.

Regarding the Kq matrix (which represents the relationships between planar patches of the same keyframe), in our case, its components store a weighted distance comprising the distance between the respective normal vectors and the distance from one to another plane in case of parallel patches.

Finally, matching candidates between graphs are defined by the C_t matrix. A plane from the Kf_r graph is a candidate with a plane of graph Kf_c if the orientation between them is the same, where the plane orientation has been classified as horizontal, vertical or oblique.

III. Position refinement using edges and planes

As usual, camera motion can be expressed through the rigid body transformation $T_{k,k-1} \in \mathbb{R}^{4\times4}$:

$$T_{k,k-1} = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}, \tag{1}$$

where $R_{k,k-1} \in SO(3)$ is a rotation matrix, and $t_{k,k-1} \in \mathbb{R}^{3\times1}$ is a translation vector. The set $T_{1:n} = T_{(1,0)}, ..., T_{(n,n-1)}$ expresses the camera motion between frames from 0 to n.

The estimation error in the transformation between consecutive keyframes is what we want to reduce through the optimization stage. We use least squares to minimize this error, where we combine geometric and photometric information, respectively represented by planar information and image edges. We also use the transformation between keyframes provided by REVO as an initial guess, in order to achieve a faster convergence during optimization. The transformation error E is expressed as:

$$E = E_{edge} + w_{pl} E_{pl}, \tag{2}$$

where w_{pl} is the relative importance between the two terms of the cost function. E prioritizes planar error over edge error. E_{edge} and E_{pl} are detailed in the following sections.

To solve the least squares problem we use the Ceres software [18]. We describe the rotation term using the Rodrigues formulation. This formulation provides an efficient way of representing a rotation, which is described by the angle of

rotation θ around a unitary vector $v = [x, y, z]$. This formulation has no singularities, unlike Euler angles, apart from a discontinuity in π radians, which we properly avoid within the system. On the other side, instead of using vector (x, y, z) and angle θ, we use a combination of both elements into a single vector $c_{rod} = [r_x, r_y, r_z] = [x/\theta, y/\theta, z/\theta]$, whereas to invert this representation we use $\theta_i = 1/\sqrt{r_x^2 + r_y^2 + r_z^2}$ for the angle and $[x, y, z] = [r_x \cdot \theta_i, r_y \cdot \theta_i, r_z \cdot \theta_i]$ for the vector. Using this single vector in the optimization process reduces the amount of parameters to optimize.

III-A. Optimization based on image edges

We have chosen a similar optimization process than in REVO [15] with two differences. The first one is that we do not use the pyramidal system that they propose. This method is often used to register two sequences with big displacement between them. In our case, we have the initial transformation between keyframes and the plane optimization term to solve this problem. By means of the second modification we adjust the importance of each evaluated point depending on its depth. This is related to the fact that the sensor is less accurate for further points. The weight applied to each point is $w_d = Z(p)^{-2}$, where $Z(p)$ is the depth of the point.

The error term for edges E_{edge} is described by (3). It accounts for all the errors for each edge point $p_{i,c}$ from the current image. We also use a Huber loss function $\delta_H()$ in order to tolerate large residuals.

$$E_{edge} = \sum_{p_{i,c}} \delta_H(r_e(p_{i,c}) \cdot w_d(Z(p_{i,c}))) \tag{3}$$

The error associated to each edge r_e is obtained by the Euclidean distance between the projection τ onto Kf_r, using the edge depth information $Z(p_{i,c})$ as well as the transformation between frames T_{rc}, of the evaluated edge from the current image and the position of the closest edge in the reference image, obtained by evaluating the Distance Transform (DT):

$$r_e = DT(\tau(T_{rc}, p_{i,c}, Z(p_{i,c}))) \tag{4}$$

III-B. Optimization based on plane primitives

This side of the optimization is intended to minimize the distance of the boundary points of a planar patch from one keyframe to the corresponding planar patch of the other keyframe, similarly to [6]. We use boundary points because they result to be an overall and simplified representation of a plane shape, and they also reduce the number of points the system has to process.

The error term for planes E_{pl} accounts for all the errors of the boundary points of planar patches in Kf_r projected onto the corresponding patches of Kf_c:

$$E_{pl} = \sum_{v_{i,r} \in Pl_{j,r}} \sum_{\pi_k \in Pl_{k,c}} S_{v_i,\pi_k} \cdot \delta_C(r_{pl} \cdot w_d(Z(v_i))), \tag{5}$$

where plane matching is described by S, with $S_{v_i,\pi_k} = 1$ if $Pl_{j,r}$ and $Pl_{k,c}$ match, where $v_{i,r} \in Pl_{j,r}$ is a boundary point, and π_k is the plane equation of patch k, $\delta_C()$ is the Cauchy loss and w_d has already been explained.

In the previous formulation, the error from a boundary point $v_{i,r}$ denoted by r_{pl}, is calculated as the perpendicular distance of $v_{i,r}$ to the corresponding planar patch once transformed to the current keyframe:

$$r_{pl} = n_{k,c}^T \cdot R_{rc} \cdot v_{i,r} + n_{k,c}^T \cdot Tr_{rc} - d_{k,c} \tag{6}$$

where the plane equation π_k involves the patch normal vector n and the distance from the plane to the camera optical center d, and (R_{rc}, Tr_{rc}) are, respectively, the rotation and translation from Kf_r to Kf_c.

(a) Pl-EVO (b) REVO

Fig. 2: Generated 3D map from Pl-EVO and from REVO.

IV. Results

In this section, we report on comparison results between the proposed odometer and some state-of-the-art visual odometers and full SLAM systems. In the comparison, we use two well-known public datasets, the TUM RGB-D benchmark [19] and the synthetic dataset ICL-NUIM [20]. The metric used to compare with other algorithms is the Root-Mean-Square-Error (RMSE).

Table I collects the RMSE values for the camera motion estimators involved in the comparison. Each row corresponds to a sequence of a dataset, whereas the columns corresponds to different SLAM systems such as [1] [2] [4] [21] and visual odometers [13] [21] and our visual odometer Pl-EVO. In the table, column REVO corresponds to the algorithm available online, which only register two consecutive frames, whereas column REVO E+D+Opt corresponds to the algorithm described in [13], where they optimize for the N previous frames to improve camera motion estimation.

Table I suggests that the Pl-EVO obtains better results than the state-of-the-art odometers and SLAM systems for indoor environments lacking texture, without the need for relocalization or loop closing strategies and only performing a register between two keyframes. Moreover, we obtain similar results to the other systems for unstructured indoor environments.

In Pl-EVO, when a keyframe is processed, a map of the environment is generated using the optimized pose and the point cloud associated. By way of illustration, Figure 2 shows the results of REVO and Pl-EVO. It can be observed that one single wrong register misaligns all the consecutive frames for the case of REVO.

TABLE I: Absolute trajectory error calculated using RMSE (cm)

Sequence	Pl-EVO	REVO	Edge SLAM †	ORB-SLAM†	LSD-SLAM†	Edge VO†	PL-SLAM†	REVO E+D+Opt‡
fr1/xyz*	3.56	4.86	1.31	**0.9**	9.0	16.51	1.21	1.55
fr2/xyz*	1.50	1.90	0.49	**0.3**	2.15	21.41	0.43	-
fr3/str_notex_far	**1.60**	2.38	6.71	×	×	41.76	×	2.17
ICL/office0*	**2.78**	6.70	3.21	5.67	×	×	-	-
ICL/office1	1.23	2.30	19.5	×	×	×	×	**0.98**
ICL/office3*	**0.89**	2.96	4.58	16.18	×	×	×	-

– This information is not available.
× The system has not been able to initialize, or the motion estimator gets lost during the sequence processing.
† The results come from [21].
‡ The results come from [13]
∗ It shows a possible loop closing dataset.

V. Conclusions and future work

In this work, we have presented a novel visual odometer capable of operating within weakly textured environments. We have shown that Pl-EVO obtains better results in structured environments lacking texture, and similar results in the rest of indoor environments. These improvements have been achieved through the combination of planar primitives and edge data during the registration process.

A plane matching technique used in the proposed odometer has also been described. Planes correspondences are found by the use of a graph matching technique. Unlike other methods, this process does not require previous knowledge of the transformation between frames to find correspondences.

As for future work, we plan to integrate inertial data from an Inertial Measurement Unit (IMU) inside the motion estimation process, as well as into the plane matching task, to enhance global system performance.

References

[1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "Orb-SLAM: A versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, Oct 2015.

[2] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 834–849.

[3] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, Oct 2011, pp. 127–136.

[4] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "PL-SLAM: Real-time monocular visual SLAM with points and lines," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 4503–4508.

[5] X. Zuo, X. Xie, Y. Liu, and G. Huang, "Robust visual SLAM with point and line features," *arXiv preprint arXiv:1711.08654*, 2017.

[6] M. Dou, L. Guan, J.-M. Frahm, and H. Fuchs, "Exploring high-level plane primitives for indoor 3D reconstruction with a hand-held RGB-D camera," in *Computer Vision - ACCV 2012 Workshops*, J.-I. Park and J. Kim, Eds. Springer Berlin Heidelberg, 2013, pp. 94–108.

[7] P. F. Proença and Y. Gao, "Probabilistic combination of noisy points and planes for RGB-D odometry," *CoRR*, vol. abs/1705.06516, 2017. [Online]. Available: http://arxiv.org/abs/1705.06516

[8] Y. Taguchi, Y. Jian, S. Ramalingam, and C. Feng, "Point-plane SLAM for hand-held 3D sensors," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*, May 2013, pp. 5182–5189.

[9] E. Ataer-Cansizoglu, Y. Taguchi, S. Ramalingam, and T. Garaas, "Tracking an RGB-D camera using points and planes," in *2013 IEEE International Conference on Computer Vision Workshops*, Dec 2013, pp. 51–58.

[10] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, "Keyframe-based dense planar SLAM," in *IEEE International Conference on Robotics and Automation (ICRA), Singapore*, 2017.

[11] X. Wang, W. Dong, M. Zhou, R. Li, and H. Zha, "Edge enhanced direct visual odometry." in *BMVC*, 2016.

[12] R. Li, Q. Liu, J. Gui, D. Gu, and H. Hu, "A novel RGB-D SLAM algorithm based on points and plane-patches," in *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, Aug 2016, pp. 1348–1353.

[13] F. Schenk and F. Fraundorfer, "Combining edge images and depth maps for robust visual odometry," in *Proc. 28th British Machine Vision Conference*, 2017, pp. 1–12.

[14] E. Fernández-Moral, P. Rives, V. Arévalo, and J. González-Jiménez, "Scene structure registration for localization and mapping," *Robotics and Autonomous Systems*, vol. 75, pp. 649 – 660, 2016.

[15] F. Schenk and F. Fraundorfer, "Robust edge-based visual odometry using machine-learned edges," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 1297–1304.

[16] C. Feng, Y. Taguchi, and V. R. Kamat, "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6218–6225.

[17] F. Zhou and F. De la Torre, "Factorized graph matching," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 9, pp. 1774–1789, 2016.

[18] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[19] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 573–580.

[20] A. Handa, T. Whelan, J. McDonald, and A. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China, May 2014.

[21] S. Maity, A. Saha, and B. Bhowmick, "Edge SLAM: Edge points based monocular visual SLAM," in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, Oct 2017, pp. 2408–2417.