2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

# From Points to Planes - Adding Planar Constraints to Monocular SLAM Factor Graphs

Charlotte Arndt[1,2], Reza Sabzevari[1] and Javier Civera[2]

***Abstract*— Planar structures are common in man-made environments. Their addition to monocular SLAM algorithms is of relevance in order to achieve more complete and higher-level scene representations. Also, the additional constraints they introduce might reduce the estimation errors in certain situations. In this paper we present a novel formulation to incorporate plane landmarks and planar constraints to feature-based monocular SLAM. Specifically, we enforce in-plane points to lie exactly in the plane they belong to, propagating such information to the rest of the states. Our formulation, differently from the state of the art, allows us to incorporate general planes, independently of depth information or CNN segmentation being available (although we could also use them). We evaluate our method in several sequences of public databases, showing accurate plane estimations and pose accuracy on par with state-of-the-art point-only monocular SLAM.**

## I. Introduction

A typical, though important, challenge for robots and autonomous vehicles is accurate localisation in an unknown environment. While this can be done with external systems like GNSS, those are not always available and often not accurate enough. In the case of GPS, for example, the localisation quality indoors or in urban canyons is often insufficient. An alternative is to use on-board sensors to build a map of the environment and to localise the robot with respect to it. This approach is called Simultaneous Localisation and Mapping (SLAM), and has been a highly researched topic for many years [1].

Among the many sensors used for localisation, monocular cameras stand out because of their high availability and low costs. The most popular monocular SLAM systems, for example ORB-SLAM2 [2] or DSO [3], use points as the primary, and most often the only, landmark type in the map. However, points do not preserve any high-level information about the semantics and geometric structure of the scene. Over the years there have been significant efforts to use semantic information in Structure from Motion [4], [5] and mapping [6] or to incorporate elements like lines [7], [8] or objects [9]–[11] into monocular SLAM systems.

In this paper, we focus on planes. They are geometric structures which are commonly found in man-made environments and a good representation for semantically relevant elements like the floor or walls as well as surfaces of furniture. This makes them promising entities to enhance the map representation of SLAM systems. Also, they have the potential to reduce the estimation errors, as they add constraints with a large spatial extent.

In the following, we present, a monocular system based on the state-of-the-art ORB-SLAM2 [2] that incorporates planes into the map and planar constraints into the optimization. Our main contribution is modelling the joint optimization of planes, points belonging to the planes and regular 3D points using factor graphs. In contrast to other sensing modalities, planes are not directly observed from the monocular images and hence have to be initialized from the point cloud of the underlining sparse map and updated via the points belonging to them. We evaluate our algorithm on several sequences from public datasets, demonstrating an accurate estimation of scene planes and camera pose errors on par with the point-only monocular SLAM baseline.

The rest of the paper is structured as follows: In Section II the related work on SLAM systems using planes is reviewed in detail and differences to our work are highlighted. Section III introduces our formulation for the joint optimization of keyframes, planes and points. Section IV details our experimental setup as well as the results. Finally, Section V contains the conclusions and lines for future work.

## II. Related work

### A. RGB-D Plane SLAM

RGB-D cameras have become popular sensors for indoor vision applications. The depth channel allows us to produce 3D reconstructions at scale even from a single view.

With RGB-D cameras, it is possible to measure planes directly. Based on this, Kaess [12] introduces a minimal representation of planes for SLAM suitable for optimisation with incremental solvers. Similar to quaternions, which are commonly used to represent rotations, a plane in homogeneous representation can be normalised to lie on the unit sphere $\mathbb{S}^3$ and be updated in the tangent space using the exponential mapping. This resulted in a planar mapping system using only planes as landmarks. Hosseinzadeh et al. [13] included this minimal formulation in their RGB-D SLAM system which uses points, planes and quadrics representing objects in the map. Additionally, they include constraints between planes based on the Manhattan world assumption and a tangential constraint between objects and planes. They are able to show substantial improvement compared to the state-of-the-art point-based RGB-D ORB-SLAM2 [2]. In our approach, we use such parametrization in

[1]CV-Lab, Coorporate Research, Robert Bosch GmbH, Hildesheim, Germany, charlotte.arndt2@de.bosch.com, reza.sabzevari@de.bosch.com

[2]Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, Zaragoza 50018, Spain, jcivera@unizar.es

JC was in part supported by research grants from the Spanish Government (PGC2018-096367-B-I00) and the Aragón Government (DGA-T45 17R/FSE).

the monocular case. This is challenging as the plane cannot be measured directly but only indirectly via the points that belong to it.

### B. Monocular Plane SLAM

In contrast to RGB-D images, monocular ones do not generally contain the geometric information to estimate the parameters of a plane from a single image. Nevertheless, different monocular SLAM systems incorporating planes have been proposed.

Concha and Civera [14], [15] use triangulated planes to complete sparse and semi-dense monocular maps, but do not optimize the planes jointly with the map states. Gee et al. [16] is an early work on integrating planes into the joint SLAM optimization. The authors proposed a monocular EFK SLAM where planes are discovered from the 3D map points. Once a plane is detected, its points are then converted into a 2D representation within the plane, enforcing their planarity. In our work, we incorporate their planarity constraint into a factor graph formulation and integrate it into the state-of-the-art system ORB SLAM [2].

Yang et al. [17] estimate wall planes from wall-floor boundaries, which they detect using a CNN. The plane landmarks improve the robustness of their SLAM system in scenes with little texture. A follow up work [18] adds objects and modifies the plane detection. Pixelwise semantic segmentation is used to find initial wall-ground boundaries and the detections are refined using a CRF. Nevertheless, both systems are limited to wall planes, while our formulation can work with general planes, independently of how they are extracted.

In [11] a monocular SLAM system using points, planes and objects is presented. The plane are extracted by a neural network which makes it possible to generate plane measurements from monocular images directly. Objects are represented as quadrics.

Summing up, all methods presented except [16] either 1) use an RGB-D camera or a neural network to detect planes, or 2) do not include planes in the joint multi-view map optimization. In our work we address the monocular case, using the minimal parametrization proposed in [12] to optimise planes in the manifold. We combine this with the monocular planar constraint from [16]. Finally, we develop a factor graph-based formulation to optimise planes, points and keyframes and integrate this with ORB SLAM [2].

## III. SLAM with Points and Planes

In this section we present our formulation for the optimisation of points and planes in a SLAM system.

### A. Mapping and Tracking States

We define our map state $\boldsymbol{x}_M$ as follows

$$\boldsymbol{x}_M = \left(\boldsymbol{c}_1^\top \boldsymbol{c}_i^\top \dots \boldsymbol{P}_1^\top \boldsymbol{P}_j^\top \dots \boldsymbol{\pi}_1^\top \boldsymbol{\pi}_k^\top \dots \boldsymbol{y}_1^{1\top} \dots \boldsymbol{y}_l^{k\top} \dots\right)^\top \tag{1}$$

where $\mathcal{C} = \{\boldsymbol{c}_i \mid i = 1, \dots, m \; ; \boldsymbol{c}_i \in SE(3)\}$ represents the poses of a set of keyframes, $\mathcal{P} = \{\boldsymbol{P}_j \mid j = 1, \dots, n \; ; \boldsymbol{P}_i \in \mathbb{R}^3\}$ are regular 3D points that do not belong to planes, $\mathcal{K} = \{\boldsymbol{\pi}_k \mid k = 1, \dots, p \; ; \boldsymbol{\pi}_k \in \mathbb{P}^3\}$ are the 3D planes, and $\mathcal{Y} = \{\boldsymbol{y}_l^k \mid k = 1, \dots, p \; ; l = 1, \dots, n_k \; ; \boldsymbol{y}_l^k \in \mathbb{R}^2\}$ are the in-plane points, expressed in the reference system of the local plane.

The tracking state $\boldsymbol{x}_t \in SE(3)$ contains the camera pose in the current time step $t$.

### B. Plane Representation

There are many different alternatives to plane representation. A plane in homogeneous representation is given as $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3, \pi_4)^\top \in \mathbb{P}^3$. Another common representation is using the normal vector $\mathbf{n} = (n_x, n_y, n_z)^\top$ and the distance to the origin $d$, where:

$$\boldsymbol{n} = \frac{(\pi_1, \pi_2, \pi_3)^\top}{\sqrt{\pi_1^2 + \pi_2^2 + \pi_3^2}} \tag{2}$$

$$d = \frac{-\pi_4}{\sqrt{\pi_1^2 + \pi_2^2 + \pi_3^2}} \tag{3}$$

To optimise a plane, it is best to use a minimal representation which has only as many parameters as the degrees of freedom of a plane, namely 3. Additionally, it should not contain singularities. This can be achieved by optimising in the tangent space of $S^3$ as proposed in [12]. Looking back at Equation (2) and Equation (3), it can be seen that the first three elements of a plane in its homogeneous representation form a vector part closely related to the normal vector of the plane, while the last element is a scalar related to the distance of the plane from the origin. Furthermore, by normalising the homogeneous coordinates of a plane, we have

$$\boldsymbol{\pi}' = \boldsymbol{\pi} / \|\boldsymbol{\pi}\| \in S^3 \tag{4}$$

Another representation of this unit sphere $S^3$ is in the context of quaternions. A quaternion $\boldsymbol{q}$ can be normalised to a unit quaternion $\boldsymbol{q}'$ which is frequently used to represent and optimise rotations.

$$\boldsymbol{q}' \in S^3 = \{\boldsymbol{q} \in \mathbb{R}^4 : \|\boldsymbol{q}\| = 1\} \tag{5}$$

Both representations cover the same space and therefore, they can be optimised in the same tangent space of $S^3$ using the exponential and log mapping derived by Grassia [19]

$$\boldsymbol{q}^{(s+1)} = \exp(\boldsymbol{\omega})\boldsymbol{q}^{(s)} \tag{6}$$

$$\exp(\boldsymbol{\omega}) = \begin{pmatrix} \frac{1}{2} sinc(\frac{1}{2}\|\boldsymbol{\omega}\|)\boldsymbol{\omega} \\ \cos(\frac{1}{2}\|\boldsymbol{\omega}\|) \end{pmatrix} , \tag{7}$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ represents an incremental update of the plane. $\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}/\|\boldsymbol{\omega}\|$ is the axis of the normal vector and $\|\boldsymbol{\omega}\|$ is related to the distance of the plane from the origin.

### C. In-plane points

Once a plane is introduced in the monocular map, the 3D points belonging to such a plane can be represented as 2D points in the local coordinate frame of the plane (we will denote such points as in-plane points). In this manner, we enforce the points to be not only close to but exactly in the plane. As we use homogeneous coordinates during the optimisation, that means a conversion between different representations of the plane is needed to parameterize the points.

The position of an in-plane point relative to the plane origin is represented as a linear combination of two arbitrary orthogonal vectors contained in the plane, $\boldsymbol{v}_0$ and $\boldsymbol{v}_1$. For convenience, the origin of plane $\boldsymbol{\pi}_k$ in the world coordinate system $\boldsymbol{O}_k^w$ is defined as the point in the plane which is the closest to the origin of the world frame.

$$\boldsymbol{O}_k^w = \begin{pmatrix} -dn_x & -dn_y & -dn_z \end{pmatrix}^\top \tag{8}$$

As $\boldsymbol{v}_0$ is perpendicular to the plane normal $\mathbf{n}$ and of unit length, we have the two equations $\boldsymbol{v}_0 \cdot \boldsymbol{n} = 0$ and $\|\boldsymbol{v}_0\| = 1$. Arbitrarily setting the $y$-component to zero allows us to solve for the remaining two components.

$$\boldsymbol{v}_0 = \begin{pmatrix} \frac{-n_z}{n_x\sqrt{1+(\frac{n_z}{n_x})^2}} & 0 & \frac{1}{\sqrt{1+(\frac{n_z}{n_x})^2}} \end{pmatrix}^\top \tag{9}$$

As $\boldsymbol{v}_1$ is perpendicular to both $\boldsymbol{v}_0$ and $\boldsymbol{n}$, it can be determined as $\boldsymbol{v}_1 = \boldsymbol{v}_0 \times \boldsymbol{n}$.

Finally, we convert a 3D point in the world coordinate system $\boldsymbol{P}_j \in \mathbb{R}^3$ into a 2D in-plane point $\boldsymbol{y}_l^k \in \mathbb{R}^2$ in the coordinate system of its plane by implicitly projecting it onto the plane along the normal direction.

$$\boldsymbol{y}_l^k = \begin{bmatrix} (\boldsymbol{P}_j - \boldsymbol{O}_k^w) \cdot \boldsymbol{v}_0 \\ (\boldsymbol{P}_j - \boldsymbol{O}_k^w) \cdot \boldsymbol{v}_1 \end{bmatrix} \tag{10}$$

In the course of this operation, $\boldsymbol{P}_j$ is removed from the set of 3D points and $\boldsymbol{y}_l^k$ is added to the set of in-plane points instead.

### D. Local Bundle Adjustment using Point and Plane Constraints

Our measurements $\mathcal{Z} = \{\boldsymbol{z}_{1,1}, \ldots, \boldsymbol{z}_{i,j}, \ldots, \boldsymbol{z}_{i_k,j_k,k}, \ldots\}$ are the observations of regular points ($\boldsymbol{z}_{i,j}$) and in-plane points ($\boldsymbol{z}_{i_k,j_k,k}$) in the images taken by the camera.

The cost function to be optimised during the least-squares estimation is the sum of the reprojection errors of all points in all the images.

$$C = \sum_{i,j} \rho_h(\boldsymbol{e}_{i,j}^T \boldsymbol{\Omega}_{i,j}^{-1} \boldsymbol{e}_{i,j}) + \sum_{i,k,l} \rho_h(\boldsymbol{e}_{i,k,l}^T \boldsymbol{\Omega}_{i,k,l}^{-1} \boldsymbol{e}_{i,k,l}) \tag{11}$$

We use the robust Huber cost function $\rho_h$ and $\boldsymbol{\Omega}_{i,j}$ is the covariance matrix for the keypoint at the scale it was detected.

The projection equation is similar for 3D points and in-plane points. For 3D points, the keypoint $\boldsymbol{z}_{i,j}$ visible in the image of camera $\boldsymbol{c}_i$ is associated with point $\boldsymbol{P}_j$ and compared to its reprojected position.

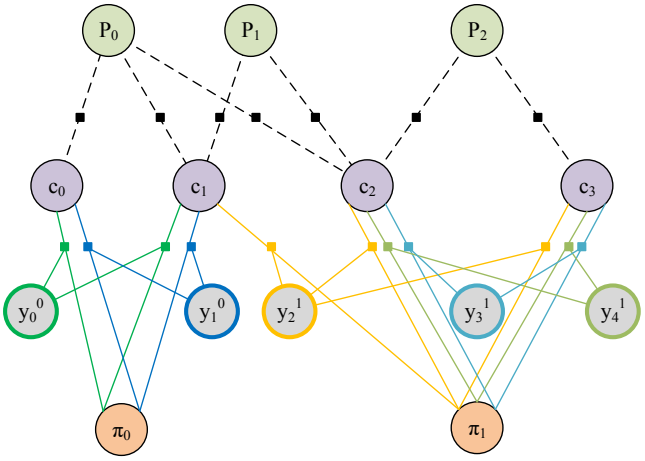



Fig. 1: Factor graph of the proposed optimisation. Vertices for 3D points are in green, camera poses in purple, planes in orange and in-plane points in grey with colored outline. Ternary factors connect camera poses, in-plane points and planes. The factors and edges leading to them are colored according to the in-plane point they are connected with. Figure best viewed in color.

$$\boldsymbol{e}_{i,j} = \boldsymbol{z}_{i,j} - proj(\boldsymbol{c}_i, \boldsymbol{P}_j) \tag{12}$$

An in-plane point $\boldsymbol{y}_l^k$ in a plane $\boldsymbol{\pi}_k$ can be converted to the world coordinate system $\boldsymbol{Y}_{k,l}^w$ at any time using

$$\boldsymbol{Y}_{k,l}^w = \begin{bmatrix} \boldsymbol{v}_0 & \boldsymbol{v}_1 \end{bmatrix} \cdot \boldsymbol{y}_l^k + \boldsymbol{O}_k^w \tag{13}$$

Afterwards, the point can be projected into the camera frame with the usual projection equation to calculate the error to the associated keypoint.

$$\boldsymbol{e}_{i,k,l} = \boldsymbol{z}_{i,k,l} - proj(\boldsymbol{c}_i, \boldsymbol{Y}_{k,l}^w(\boldsymbol{\pi}_k, \boldsymbol{y}_l^k)) \tag{14}$$

Fig. 1 shows the resulting factor graph visualising the optimisation described by the equations above. All points in the same plane are indirectly connected with each other via the shared plane parameters. This in turn results in a constraint that might have a much larger extent than the field of views of the camera frames, which has a positive effect in the accuracy of the estimation.

### E. Implementation details

For this paper, we have integrated our formulation for planes and in-plane points into the monocular pipeline of the state-of-the-art system ORB-SLAM2 [2]. This influences several parts of the system, besides the bundle adjustment mentioned in section III-D, which are described in the following.

**Camera Tracking.** During tracking, only the camera pose is optimised while all points, regular and in-plane points, are kept fixed. Therefore, in-plane points are used in nearly the same way as 3D points, requiring only its transformation to the reference frame $w$ using Equation 13 before using the standard projection model.

**Pose Graph Optimization and Loop Closure.** To close a loop, both ends of the map need to be aligned followed by a global bundle adjustment. We first detach the in-plane points from the plane temporarily, in order to compute an initial alignment. After loop closure, the plane is re-estimated from the set of points that previously belonged to it. Some of the points might now be outliers to the re-estimated plane and converted into regular 3D points. Finally, a global bundle adjustment is performed on all points, planes and keyframe poses.

**Local map for local bundle adjustment.** In ORB-SLAM2, the local bundle adjustment is performed on a window around the current keyframe, which is selected based on covisibility information. To make sure the plane is optimised correctly, we add all points belonging to the plane to the optimisation. Additionally, all keyframes in which these points are visible are added but remain fixed.

**Plane insertion and management.** The initialisation and extension of planes is implemented as part of the mapping thread. In contrast to related papers where the authors extract planes from RGB-D data [12], [13] or additional information from CNNs [11], [17], we use a method based on the information already available in the SLAM map.

In our plane discovery algorithm, we use RANSAC to find planes from the 3D point cloud. Singular value decomposition (SVD) is used to calculate the plane parameters. New points are added to existing planes if their distance to the plane is within a certain threshold. As the system is monocular and therefore reconstructs only up to scale, this threshold is without scale. Another method for determining if points form a plane is semantic information. If semantic labels are available, a plane can be fitted to the points which belong to a class that can be represented with a plane (e.g., walls, floor, ceiling, doors or paintings).

Points can be removed from a plane if they are determined to be outliers during the optimisation. For this, we adapted the outlier conditions used in ORB SLAM which are based on the $\chi^2$ test [20]. We assume a two-dimensional Chi-squared distribution and use $0.95$ as the confidence threshold. If they fail to pass the test for $80\%$ of their edges in the factor graph, in-plane points are removed from the plane. Otherwise the outlier observations are eliminated. When a point is removed from its plane, it is converted into a regular 3D point with the same position in the world coordinate frame $w$. Freed from the plane constraint affecting it, following optimizations can move it to a position better supported by the image observations.

## IV. EXPERIMENTS

In the following section, we present experimental results on three sequences from the TUM RGB-D dataset [21], the flyroom sequence from [22] and a sequence from the InteriorNet dataset [23]. We chose these sequences because they show scenes containing one or more planes, and because being publicly available facilitates future comparisons. The supplementary video shows exemplary runs of the conducted experiments for a further impression of the system performance.

### *A. Experiments with one dominant plane.*

In the first experiments we examine the effect of adding a single dominant plane. We selected three sequences from the TUM benchmark, which consist of mainly planar structures and a sequence from the flyroom dataset, where the drone with a downward-facing camera flies multiple circles over a textured floor. Sample frames from all these sequences and views of the final map are shown in Fig. 2.

For the TUM sequences, the plane discovery is based on RANSAC. A plane hypothesis is accepted if it contains at least $12.5\%$ of all map points as inliers. Once a plane with enough supporting points is found, it is inserted. Plane discovery continues on the full point cloud and if a plane hypothesis with more inliers is found, it replaces the previous plane. For the drone sequence a plane is fitted to all points and never replaced. For repeatability, we chose to insert the plane after 50 keyframes.

As the mapping thread is slowed down by plane discovery and the optimization, we play the sequences at reduced speed when inserting planes. This leads to a comparable number of map points and keyframes and hence to a fair comparison. The drone sequence is recorded with a faster camera motion and more affected by the slow-down of the mapping thread. Therefore, this sequence is played 20 times slower while the sequences from the TUM dataset are played 6 times slower.

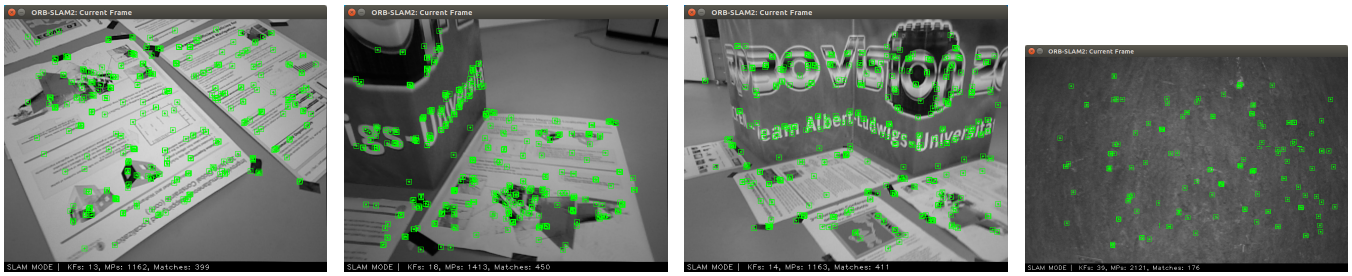
Due to the unobservability of the real scale in monocular SLAM, we use different point-to-plane thresholds $d$ in the different sequences as given in the captions of Fig. 2a to 2d.

The results are displayed in Table I. We compute the absolute pose error against the ground truth trajectory and report the RMSE of the best out of 10 runs as well as the average. The variation over different runs is also visualised in the boxplots in Fig. 2 showing the distribution of the RMSE for these 10 runs and the average, so 11 entries in total.

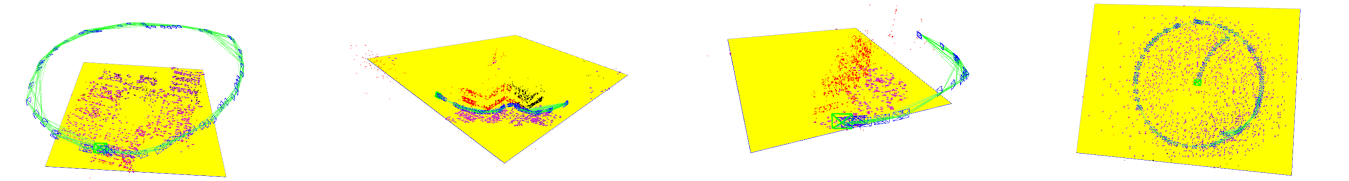
The addition of the dominant plane improves the accuracy slightly in all four sequences for both the best run and the mean over all runs. Additionally, the maps of each sequence, shown in the right colum of Fig. 2, show a successful reconstruction of the dominant plane of the scene, which is the floor plane in all these sequences.

TABLE I: RMSE of the APE for the best run and the average over 10 runs, values in cm

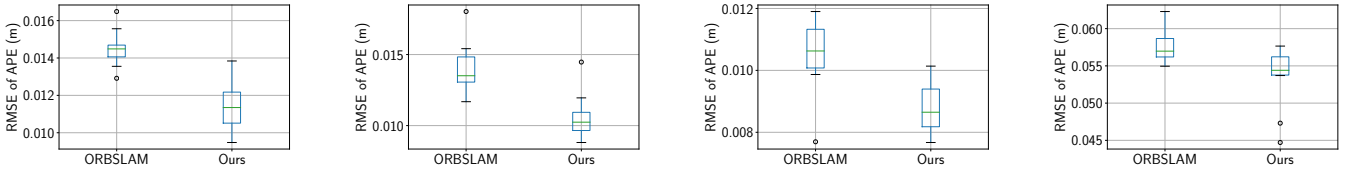
| sequence | ORB SLAM2 [2] | | Ours | |
|---|---|---|---|---|
| | best | mean | best | mean |
| nostr_txt_near [21] | 1.2914 | 1.4486 | **0.9478** | **1.1392** |
| str_txt_near [21] | 1.1682 | 1.3992 | **0.881** | **1.0569** |
| str_txt_far [21] | 0.7694 | 1.0592 | **0.7671** | **0.8812** |
| flyroom_low_nobox [22] | 5.4957 | 5.7687 | **4.4711** | **5.3702** |



(a) nostr_txt_near [21] $d = 0.001$ (b) str_txt_near [21] $d = 0.005$ (c) str_txt_far [21] $d = 0.01$ (d) flyroom [22] $d = 0.01$



(e) Map at the end of each sequence




(f) Boxplots showing the distribution of the RMSE values of 10 individual runs and the mean run for each sequence

Fig. 2: Frames, maps and RMSE boxplots of experiments with one dominant plane. Each column stands for one sequence.

### *B. Experiments with multiple planes*

Additionally, we performed experiments with multiple planes which were extracted using semantic information. We use one sequence from the InteriorNet dataset [23] which provides many synthetic indoor scenes with semantic segmentation annotations.

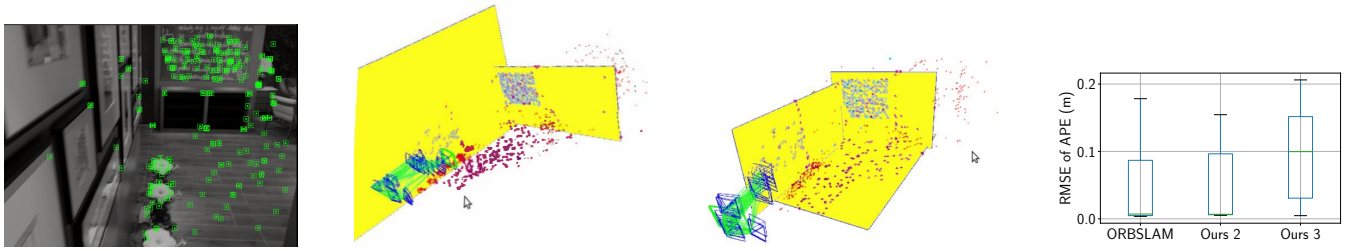
To extract the planes, we used the ground truth semantic segmentation for classes which can be represented with planes. We choose the classes *picture*, *floor* and *other structure*. The latter is the class belonging to a wall decoration made from multiple pictures. For each new keyframe in the mapping thread, the ground truth masks are loaded and each keypoint that lies inside one of the masks is labeled with the corresponding class. A plane is fitted through the point cloud of a class after at least 50 points with that label are present.

We performed experiments with two planes (*picture* and *other structure*) and three planes and compare the results against plain monocular ORB SLAM. We use sequence HD5/3FO4JYDQGRUB with trajectory 1 and the images rendered with motion blur. The trajectories in the dataset are randomly generated and can contain motions unsuitable for camera tracking. In this sequence there is a sudden turn of the camera while only the floor is visible. We only use the sequence up to frame 725 which is 50 frames before tracking failure to remove any effects this could have on the experiments. As before, for the experiment with planes, the sequence is played 6 times slower, for both SLAM systems to have a comparable number of keyframes. We report the RMSE of the APE averaged over ten runs and the best run in Table II. A boxplot with the distribution over ten runs and the mean is shown in Fig. 3.

We show a comparable performance between ORB SLAM and our method with two or three planes. According to Table II, ORB SLAM had the best run overall while our method with two planes achieves better results on average. By introducing a third plane, the performance is slightly reduced. A possible reason for this could be a higher number of incorrect point-to-plane associations for the floor plane, which degenerate the trajectory accuracy. We are currently using the same point-to-plane threshold for all planes. It seems appropriate for the *picture* and *other structure* plane. In the case of the floor plane, there are multiple 3D points caused by the flowers shown in Fig. 3a in the middle, which are close enough to be incorrectly associated with the plane. A more sophisticated thresholding mechanism might therefore improve the results.

TABLE II: RMSE of the APE for the best run and the average over 10 runs on one sequence from InteriorNet [23], values in centimeters

| ORB SLAM2 [2] | | Ours (2 planes) | | Ours (3 planes) | |
|---|---|---|---|---|---|
| best | mean | best | mean | best | mean |
| **0.3642** | 5.4459 | 0.5173 | **5.1753** | 0.4724 | 9.9551 |




(a) Example frame (b) Map with two planes (c) Map with three planes (d) Boxplot

Fig. 3: (a) Sample frame of the InteriorNet [23] sequence ($d = 0.025$) with tracked points; (b) side view of the estimated keyframes, points and planes for the 2-planes experiment; (c) side view of the estimated keyframes, points and planes for the 3-planes experiment; (d) boxplot showing the distribution of the RMSE values of 10 individual runs and the mean for ORB SLAM and our version with 2 and 3 planes.

## V. Conclusions

In this paper we introduce a novel formulation to include planes in monocular feature-based SLAM. By enforcing points to be exactly on the plane, it is possible to reconstruct accurate planes from monocular SLAM and jointly optimise them with 3D points and camera poses. Our evaluation on different sequences from public datasets showed a slight improvement in pose estimation accuracy over the point-only baseline and a reasonable reconstruction of the plane in experiments with one or multiple planes. While our optimisation does not depend on a specific method for an initial plane hypothesis, the method at hand needs to be robust enough. In our experiments, we used a simple RANSAC based method and ground truth semantic information. In general, more sophisticated methods for extraction of planes from point clouds would be necessary for general scenes. Instead of ground truth semantic annotations, a CNN for semantic segmentation can be used. For future work, it would be interesting to investigate the effects of different sizes of planes in relation to the map and the number of points in the planes. Additionally, individual, adaptive point-to-plane thresholds could improve the multi-plane performance.

## References


[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-d cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, 2017.

[3] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[4] S. Y. Bao and S. Savarese, "Semantic structure from motion," in *Proc. IEEE CVPR '11*. IEEE, 2011, pp. 2025–2032.

[5] S. Y. Bao, M. Bagra, Yu-Wei Chao, and S. Savarese, "Semantic structure from motion with points, regions, and objects," in *Proc. IEEE CVPR '12*. IEEE, 2012, pp. 2703–2710.

[6] A. Concha, W. Hussain, L. Montano, and J. Civera, "Incorporating scene priors to dense monocular mapping," *Autonomous Robots*, vol. 39, no. 3, pp. 279–292, 2015.

[7] J. Sola, T. Vidal-Calleja, J. Civera, and J. M. M. Montiel, "Impact of landmark parametrization on monocular ekf-slam with points and lines," *International journal of computer vision*, vol. 97, no. 3, pp. 339–368, 2012.

[8] 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)*. IEEE, 2017, pp. 4503–4508.

[9] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. Montiel, "Towards semantic slam using a monocular camera," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1277–1284.

[10] S. Yang and S. Scherer, "CubeSLAM: Monocular 3-D Object SLAM," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 925–938, 2019.

[11] M. Hosseinzadeh, K. Li, Y. Latif, and I. Reid, "Real-time monocular object-model aware sparse SLAM," in *Proc. IEEE ICRA '19*. IEEE, 2019, pp. 7123–7129.

[12] M. Kaess, "Simultaneous localization and mapping with infinite planes," in *Proc. IEEE ICRA '15*. IEEE, 2015, pp. 4605–4611.

[13] M. Hosseinzadeh, Y. Latif, T. Pham, N. Suenderhauf, and I. Reid, "Structure aware SLAM using quadrics and planes," in *Computer Vision ACCV 2018*, C. V. Jawahar, H. Li, G. Mori, and K. Schindler, Eds. Springer International Publishing, 2019, pp. 410–426.

[14] A. Concha and J. Civera, "Using superpixels in monocular SLAM," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 365–372.

[15] ——, "DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 5686–5693.

[16] A. P. Gee, D. Chekhlov, W. W. Mayol-Cuevas, and A. Calway, "Discovering planes and collapsing the state space in visual SLAM." in *BMVC*, 2007, pp. 1–10.

[17] S. Yang, Y. Song, M. Kaess, and S. Scherer, "Pop-up SLAM: Semantic monocular plane SLAM for low-texture environments," in *Proc. IEEE IROS '16*. IEEE, 2016, pp. 1222–1229.

[18] S. Yang and S. Scherer, "Monocular object and plane SLAM in structured environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3145–3152, 2019.

[19] F. S. Grassia, "Practical parameterization of rotations using the exponential map," *Journal of Graphics Tools*, vol. 3, no. 3, pp. 29–48, 1998.

[20] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., 2001, DOI: 10.1002/0471221279.

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

[22] Z. Zhang, H. Rebecq, C. Forster, and D. Scaramuzza, "Benefit of large field-of-view cameras for visual odometry," in *Proc. IEEE ICRA '16*. IEEE, 2016, pp. 801–808.

[23] W. Li, S. Saeedi, J. McCormac, R. Clark, D. Tzoumanikas, Q. Ye, Y. Huang, R. Tang, and S. Leutenegger, "InteriorNet: Mega-scale multi-sensor photo-realistic indoor scenes dataset," in *British Machine Vision Conference (BMVC)*, 2018. [Online]. Available: https://interiornet.org/