Structure-SLAM: Low-Drift Monocular SLAM in Indoor Environments

Yanyan Li, Nikolas Brasch, Yida Wang, Nassir Navab, and Federico Tombari

***Abstract*—In this letter a low-drift monocular SLAM method is proposed targeting indoor scenarios, where monocular SLAM often fails due to the lack of textured surfaces. Our approach decouples rotation and translation estimation of the tracking process to reduce the long-term drift in indoor environments. In order to take full advantage of the available geometric information in the scene, surface normals are predicted by a convolutional neural network from each input RGB image in real-time. First, a drift-free rotation is estimated based on lines and surface normals using spherical mean-shift clustering, leveraging the weak Manhattan World assumption. Then translation is computed from point and line features. Finally, the estimated poses are refined with a map-to-frame optimization strategy. The proposed method outperforms the state of the art on common SLAM benchmarks such as ICL-NUIM and TUM RGB-D.**

***Index Terms*—SLAM, visual learning.**

I. Introduction

VISUAL Simultaneous Localization and Mapping (V-SLAM) systems are important for autonomous robots and augmented reality, as they are used to estimate poses and reconstruct unknown environments. In numerous SLAM use cases and applications, monocular cameras are the most common sensors in indoor scenarios. Indoor environments are often characterized by a lack of textured surfaces, and by irregularly distributed feature points. In particular, low-textured walls, floor and ceiling are difficult to deal with by both state-of-the-art feature-based methods [1] as well as direct methods [2], [3]. For low-textured scenes, SLAM systems combining point and line features have been proposed to target low-textured scenes, e.g. Stereo-PLSLAM [4], PLVO [5], Mono-PLSLAM [6] and [7], extending the working scenarios to low-textured environments with visible structural edges. Since the map is built from a sequence of input frames, small errors accumulate over time, resulting in drift which affects dense reconstruction by leading to misaligned surfaces and artifacts.

There are two main strategies to overcome these errors. Loop closure detection [1], [8] combined with pose graph optimization detects previously seen landmarks and optimizes the pose graph based on the new constraints, thus correcting the accumulated drift. Loop closure, however, brings in an extra computational burden and removes the drift only when revisiting the same place. Another strategy consists of assuming an underlying (global) structure in the world frame, then each tracked frame can be directly aligned to this world structure instead of the last frame or keyframes. The most common formulation of a structured scene is the Manhattan World (MW) [9], [10] where the environment shown in Fig. 1(a) consists of geometric structures (planes and lines) oriented in one of three orthogonal orientations. It is particularly useful in indoor environments where structures such as walls, floor and ceilings often show consistent alignment over multiple rooms, enabling a global alignment.

The MW approach is an efficient method to keep the accumulated drift low by providing a drift-free strategy for rotation estimation, as the rotational component is the main source of overall drift [11], [12].

The state of the art of monocular approaches relying on a MW [9], [10] are based on parallel and orthogonal lines alone, as it is difficult to extract 3D information, except for vanishing points, from a monocular RGB image, which is a quite strong limitation for most scenarios. Furthermore, indoor environments often consist of large planar regions with few features for pose estimation. RGB-D methods [12], [13], directly measure the structure of the scene in the form of depth maps, this allows them to compute dense surface normals for each pixel.

Inspired by recent works based on convolutional neural networks (CNN) and scene geometry prediction approaches from a single view [14], [15], we propose a monocular SLAM framework which leverages the underlying scene structure to carry out low-drift SLAM even in presence of low-textured environments, in the form of densely predicted normal maps from a CNN, analogously to existing works based on dense RGB-D sensors.

Specifically, we propose the following contributions:

- A low drift real-time monocular SLAM framework for structured environments, with decoupled rotation and translation
- Dense monocular normal estimation for rotation estimation leveraging the MW assumption

Manuscript received February 24, 2020; accepted July 23, 2020. Date of publication August 11, 2020; date of current version August 21, 2020. This work was supported by China Scholarship Council. This letter was recommended for publication by Associate Editor J. Andrade-Cetto and Editor S. Behnke upon evaluation of the reviewers' comments. *(Yanyan Li and Nikolas Brasch contributed equally to this work.) (Corresponding author: Yanyan Li.)*

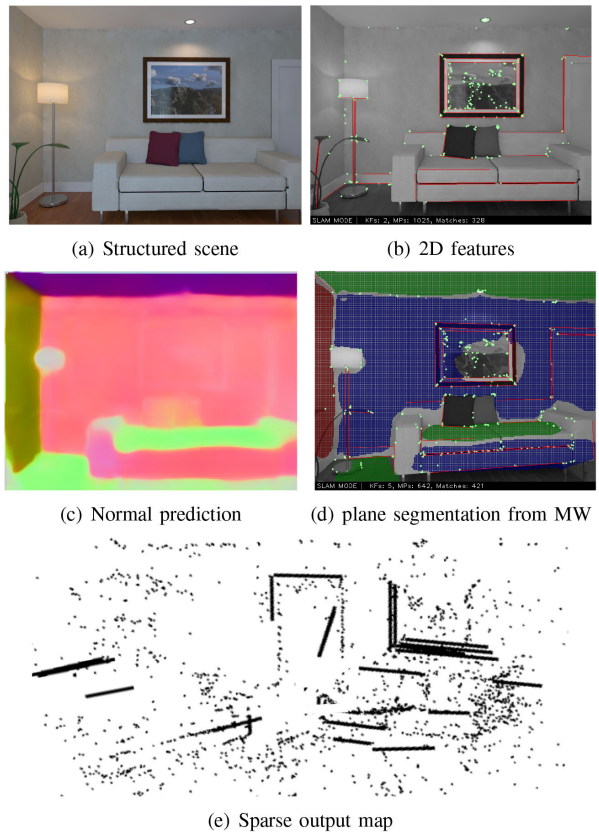
Yanyan Li, Nikolas Brasch, and Yida Wang are with the Technical University of Munich, 80333 München, Germany (e-mail: yanyan.li@tum.de; nikolas.brasch@tum.de; yida.wang@tum.de).

Nassir Navab is with the Technical University of Munich, 80333 München, Germany, and also with the Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: nassir.navab@tum.de).

Federico Tombari is with the Technical University of Munich, 80333 München, Germany, and also with the Google, 8002 Zürich, Switzerland (e-mail: tombari@in.tum.de).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/LRA.2020.3015456



(a) Structured scene (b) 2D features

(c) Normal prediction (d) plane segmentation from MW

(e) Sparse output map

Fig. 1. The proposed approach targets low-textured indoor scenes to carry out low-drift monocular SLAM based on dense normal prediction and leveraging the Manhattan World assumption.

- A method for translation estimation relying on point and line features

We evaluate numerically on common SLAM benchmarks such as ICL-NUIM [16] and TUM RGB-D [17] showing that the proposed approach outperforms the state of the art in monocular SLAM.

II. Related Work

A. Monocular SLAM

PTAM [18] is a monocular, keyframe-based SLAM system which was the first work to introduce the idea of splitting camera tracking and mapping into parallel threads, and demonstrate to be successful for real time augmented reality applications in small-scale environments. Strasdat *et al.* [8] present a large scale monocular SLAM system in which the front-end bases on optical flow implemented on a GPU, followed by FAST feature matching and motion-only BA, and a back-end based on sliding-window BA. As a complete SLAM pipeline, ORB-SLAM [1] combines feature based tracking, sparse point mapping, descriptor based re-localization and loop closure altogether. In addition to point features several works propose the use of lines [4], [5] for low-textured environments, we propose to use additional dense structural information in the form of predicted normal maps.

Inspired by the recent success of deep learning based depth prediction, CNN-SLAM [19] incorporates a neural network which estimates depth information within the popular LSD-SLAM [2] framework to create dense scene reconstructions in metric scale, where depth predictions are used to initialize the SLAM system and merged continuously with the semi-dense depth maps optimized by the SLAM system. Instead of estimating depth maps only for key-frames in CNN-SLAM, our approach predicts surface normals from every RGB frame in real-time. In CodeSLAM [20], a neural network learns a compact latent representation for the structure of a scene conditioned on the RGB image, showing that the joint optimization of both structure and pose can improve monocular pose estimation. By predicting normal maps instead of depth maps we avoid the necessary differentiation operation which could introduce noise. Predicting normal maps also seems to generalize better between datasets as depth does.

B. RGB-D SLAM

Probabilistic-VO [7] combines points together with lines and planes for pose estimation while modeling their uncertainties. Due to the combination of 2D-3D point and line correspondences and 3D-3D plane matches, a weighting between reprojection and euclidean errors must be chosen empirically. CPA-SLAM [21] extended DVO-SLAM with global plane landmarks. Pose estimation and soft assignment of depth measurements to planes are computed in an Expectation-Maximization framework. KDP-SLAM [22] combines photometric and geometric loss based on plane segments instead of points for frame-to-frame pose estimation and additionally aligns plane segments with global planes in a Smoothing and Mapping (SAM) framework.

C. Manhattan World

Straub *et al.* [11] and Zhou *et al.* [23] show that the main source of drift in traditional feature-based systems is caused by the rotation estimation.

Even if the MW assumption is a good constraint for indoor SLAM, it is difficult to enforce it in monocular methods because only limited 3D information can be obtained. Zhou *et al.* [10] applies J-linkage [24] to classify parallel line segments into different groups and estimate the dominant direction from the vanishing points. If depth maps are available, surface normals can be computed directly. Joo *et al.* [25] provide a branch-and-bound framework for Manhattan Frame estimation. MVO [23] propose a unit sphere mean shift method to find the rotation matrix between the Manhattan World and the camera system. For the translational part, they compute and align density distributions of points in each orthogonal direction, avoiding the costly matching of points. OPVO [26] use planes to estimate the Manhattan Frame rotation, limiting its application to environments with at least 2 orthogonal planes. LPVO [12] adds vanishing points of lines for the rotation estimation. Both use point based methods for translation estimation. L-SLAM [13] replaces the graph based translation estimation from LPVO with a Kalman filter based SLAM update, using the LPVO translation estimation in the prediction step. Compared with [12], [13], we build an initialization module based on points, lines and

predicted normals. Further more, a refinement module is added to optimize the pose after the decoupled initialization.

III. Scene Structure Analysis

The structural information used in the system is analyzed in this section. First, we describe the methods for extraction and triangulation of points and lines; Then, an architecture for surface normal prediction is introduced.

A. Points and Lines Analysis

Point features, due to their descriptiveness, compactness and robustness to illumination changes, are the most common features used in visual SLAM systems. In our method, ORB features [27] are adopted which are fast enough to extract and robust enough to get matched. Since it's hard to extract sufficient feature points for robust pose estimation in low-textured environments, we further supplement them with line segments extracted and encoded using the LSD [28] and LBD [29] accordingly.

Similar to ORB-SLAM [1], once the 2D point features $p_n = (u_n, v_n)$ and line segments $l_m = (p_{m,s}, p_{m,e})$ are extracted in the new keyframe F_i, new features are triangulated to 3D points P_n and lines L_m with correspondences located on other connected keyframes.

Due to the factorization of rotation and translation estimation, it is possible to estimate the pose even in cases with pure rotation and no translation or with small parallax, which would not be possible with pure monocular feature based approaches. The rotation can be estimated from the Manhattan World Frame, this means fewer landmarks are needed to obtain the remaining 3 degrees of freedom for the translation.

B. Surface Normal Prediction

We use learned knowledge to reason about the 3D environment, instead of measuring dense depth values directly. Therefore, a 2D convolutional architecture(CNN) is trained to segment planar regions and predict pixel-wise surface normals. The proposed CNN is composed of a ResNet101-FPN [14] encoder for feature extraction and a two-branch decoder for planar area segmentation and normal estimation. As the planar and non-planar regions are unbalanced in indoor scenarios, we use the balanced cross entropy loss for training

$$\mathscr{L}_p = -1(1-w)\sum_{i\in P}\log p_i - w\sum_{i\in P_{neg}}\log(1-p_i), \quad (1)$$

where P and P_{neg} represent planar and non-planar regions, respectively. p_i represents the probability of the ith pixel being located in a planar region. We use w to balance the contributions of planar and non-planar pixels. Then the loss function for the normal estimation is filtered by the planar mask.

$$\mathscr{L}_n = -\frac{1}{n}\sum_{i\in P} n_i \cdot n_i^*, \quad (2)$$

where n_i and n_i^* are the predicted normal and ground truth normal for the ith pixel.

IV. Initialization

In this section, we describe the strategy of computing the relative poses between two frames and reconstructing an initial map. In order to be robust to different motions, we decouple pose estimation into rotation and translation which is explained further in the following paragraphs.

Rotation. First, we assume that there is a Manhattan coordinate system M shown in Fig. 3, we compute the relative rotation R_{C_1M} from Manhattan coordinate frame M to the first frame C_1 by clustering the normal map v_i^s of C_1 on the unit Gaussian sphere [12], [23] centered on the M. Following [12], [23], we project the normals onto the tangent plane of each Manhattan world axis r_n, where $n \in [1, 2, 3]$, for the current estimation. Instead of testing several random matrices, we found that setting R_{C_0M} to identity and running multiple mean-shift iterations is enough to obtain a good estimate. In order to remove noise from normal maps, we only consider the vectors $v_{in}^{s'}$ which are close to the axis r_n.

Then, the refined surface normal vectors $v_{in}^{s'}$ are projected to two-dimensional vectors $m_{in}^{'}$ in the nth tangential plane. We compute the cluster mean $s_n^{'}$ for the nth tangential plane under a Gaussian kernel by

$$s_n^{'} = \frac{\sum_{in} e^{-c\|m_{in}^{'}\|^2} m_{in}^{'}}{\sum_{in} e^{-c\|m_{in}^{'}\|^2}} \quad (3)$$

where c is a hyper parameter that defines the width of the kernel, which is set to 2 in our experiments. Then, we transform the cluster centers back onto the Gaussian sphere as s_n, which are used to update the angle between the camera and the MW axis $\hat{r}_n$ combining with the current rotation Q_n,

$$\hat{r}_n = Q_n s_n, \quad (4)$$

here $Q_n = [r_{mod(n,3)}, r_{mod(n+1,3)}, r_{mod(n+2,3)}]$ and $mod()$ is a modulus operation. The tangent plane and the cluster centers are iteratively computed until the rotation estimate is converged. Then we obtain $R_{C_1M} = [\hat{r}_1, \hat{r}_2, \hat{r}_3]^T$.

Translation. As for the translation estimation, 2D correspondences of points $[p_i^1, p_i^2]$ between two frames and their relative rotation $R_{C_1C_2}$ are used

$$X_i^2 = \begin{bmatrix} x_i^2 \\ y_i^2 \\ z_i^2 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} X_i^1 + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad (5)$$

where X_i^j represents a 3D point in the jth camera. By eliminating the scale z_i^2, we obtain

$$\begin{bmatrix} \tilde{x}_i^2 \\ \tilde{y}_i^2 \\ 1 \end{bmatrix} = \begin{bmatrix} (r_1 \cdot X_i^1 + t_1)/(r_3 \cdot X_i^1 + t_3) \\ (r_2 \cdot X_i^1 + t_2)/(r_3 \cdot X_i^1 + t_3) \\ 1 \end{bmatrix} \quad (6)$$

where $[\tilde{x}_i^j \ \tilde{x}_i^j \ 1]^T$ represents the ith normalized 3D point in the jth camera frame. Since X_i^1 is also a 3D point, we need to eliminate z_i^1 and build

$$\begin{bmatrix} -\tilde{y}_i^1 t_3 + t_2 \\ \tilde{x}_i^1 t_3 - t_1 \\ -\tilde{x}_i^1 t_2 + \tilde{y}_i^1 t_1 \end{bmatrix}^T \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \begin{bmatrix} \tilde{x}_i^1 \\ \tilde{y}_i^1 \\ 1 \end{bmatrix} = 0 \quad (7)$$

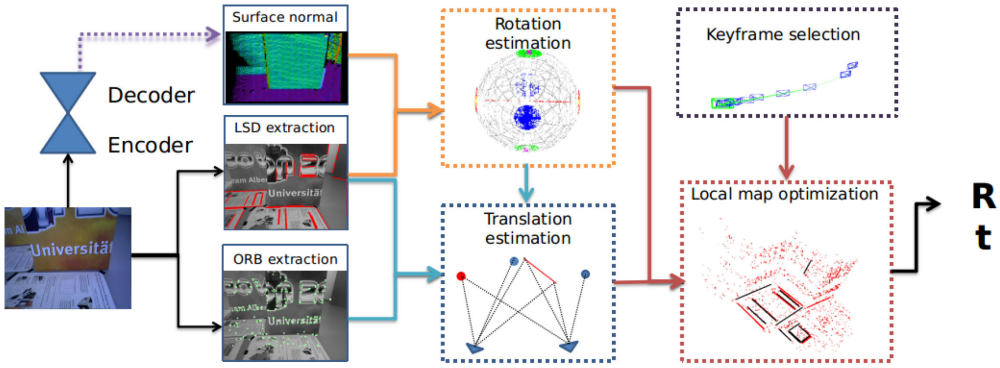


Fig. 2. Proposed SLAM framework (StructureSLAM). In the front-end, the encoder-decoder network predicts dense surface normals. In parallel, point and line features are extracted from the RGB image. In the back-end, first the scene structure in the form of normals and lines is used to estimate the global rotation of the camera. Then, the remaining 3-DoF for the translation are obtained using point and line features. The initial pose estimate is validated and refined using the local map. Keyframes are selected based on the availability of point and line features.

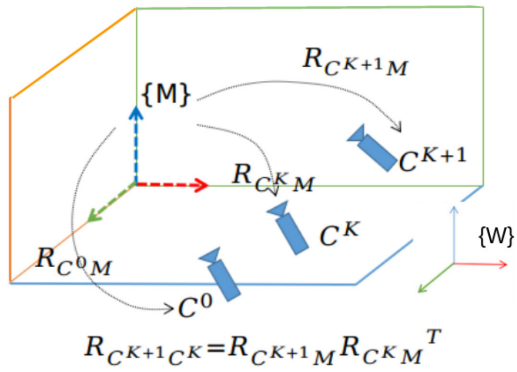


Fig. 3. Rotation estimation between multiple frames via the Manhattan world.

where $[\tilde{x}_i^j\ \tilde{x}_i^j\ 1]^T = K^T(u_i^j\ v_i^j\ 1)$ and K is the intrinsic matrix of the camera [12]. $(u_i^j\ v_i^j)$ is the ith pixel in the jth frame. Based on eq. (6) and eq.(7), we construct a translation relationship between those 2D correspondences. Then, we solve the system in eq. (7) using SVD to obtain the translation.

V. Tracking

Instead of estimating rotation and translation between two frames, we estimate the rotation between each frame and the underlying Manhattan World. The residual rotation errors are independent of the sequence length and cannot be propagated between frames. Point and line correspondences are used to estimate translation (3 DoFs) by a combination of frame-to-frame and frame-to-map methods.

A. Manhattan Rotation Estimation

This section describes the rotation estimation between camera and Manhattan system.

Given the surface normals and mask of planar regions from the network, we follow the mean-shift clustering approach, as desribed in IV, to find the dominant axes on the euclidean sphere and estimate the rotation R_{C_KM}. Since normal maps might contain errors due to the networks inference process, the clustering approach is used to remove outliers first. Furthermore, the initial rotation will be refined in following sections.

B. Translation Estimation

After obtaining the rotation matrix, we use the points and line segments to estimate the 3-DoF translational motion, which requires less features than the full 6-DoF estimation. We re-project the 3D points from the last frame to the current one and define the error function, based on the re-projection error, as follows,

$$e_{k,j}^p = p_k - \pi(R_{k,j}P_j + t_{k,j}) \tag{8}$$

here $\pi()$ is the projection function. Since the rotation matrix $R_{k,j}$ has already been estimated in the last step, we fix the rotation and only optimize the translation using the right half of the Jacobian matrix for eq. (8),

$$\frac{\partial e_{k,j}^p}{\partial \xi} = \begin{bmatrix} \frac{xyf_x}{z^2} & -\frac{z^2+x^2}{z^2}f_x & \frac{yf_x}{z} & -\frac{f_x}{z} & 0 & \frac{xf_x}{z^2} \\ \frac{z^2+y^2}{z^2}f_y & -\frac{xyf_y}{z^2} & -\frac{xf_y}{z} & 0 & -\frac{f_y}{z} & \frac{yf_y}{z^2} \end{bmatrix} \tag{9}$$

For the lines we obtain the normalized line function from the 2D endpoints p_{start} and p_{end} as follows,

$$l = \frac{p_{start} \times p_{end}}{\|p_{start}\|\|p_{end}\|} = (a, b, c) \tag{10}$$

We formulate the error function based on the point-to-line distance between l and the projected 3D endpoints P_{start} and P_{end} from the matched 3D line in the keyframe. For each endpoint P_x, the error function can be noted as,

$$e_{k,j}^l = l\pi(R_{k,j}P_x + t_{k,j}) \tag{11}$$

The Jacobian matrix for the line error eq. (11) is given by

$$\frac{\partial e_{k,j}^l}{\partial \xi} = \begin{bmatrix} -\frac{f_y l_y z^2 + f_x l_x xy + f_y l_y y^2}{z^2}, & \frac{f_x l_x z^2 + f_x l_x x^2 + f_y l_y xy}{z^2}, \\ -\frac{f_x l_x y - f_y l_y x}{z}, & \frac{f_x l_x}{z}, \\ \frac{f_y l_y}{z}, & -\frac{f_x l_x x + f_y l_y y}{z^2} \end{bmatrix} \tag{12}$$

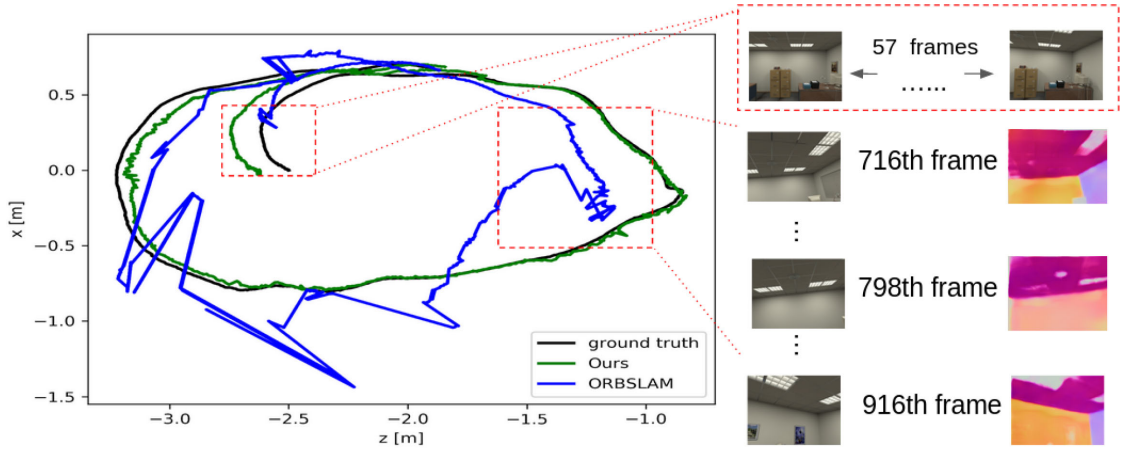


Fig. 4. Trajectory analysis, comparing the proposed method, ORB-SLAM and the ground-truth on the "of-k3" sequence in the ICL NUIM dataset.

The combined least squares cost for points and lines can be written as

$$t* = argmin \sum_{j\in(k-2,k-1)}^{M} ({e^p_{k,j}}^T e^p_{k,j} + {e^l_{k,P_x}}^T e^l_{k,P_x}) \quad (13)$$

The system is solved using the Levenberg-Marquardt algorithm.

C. Fallback and Pose Refinement

The pose estimate is based on the MW assumption. In cases of Non-Manhattan Worlds or where the Manhattan Frame is not visible in the current frame the estimated pose will be incorrect. To check whether the pose estimate obtained from the previous steps is correct we project all features from the last n keyframes onto the current frame and compute the re-projection error. By applying a threshold to filter the features we require a minimum number of inliers to accept the pose.

When not enough inliers are found we fall back to a frame-to-frame tracking method until we estimate a pose that agrees again with the Manhattan World. As a fallback we first track the new frame based on the last frame using an efficient re-projection search scheme [30] for points and lines, using the same least squares method as for the translation, this time using the full Jacobian matrix. In the case we do not get a good solution, measured based on the number of inliers, we try to estimate the pose based on the last keyframe using descriptor matching for the points [30] and re-projection based search for the lines. To reduce the drift, in the final step we optimize the pose of the new frame based on a local map constructed from the last n keyframes [30]. Here we do not use the MW assumption anymore, as we found that the initial rotation estimation is enough to reduce the drift and errors in the predicted normal maps can lead to inconsistent pose estimates.

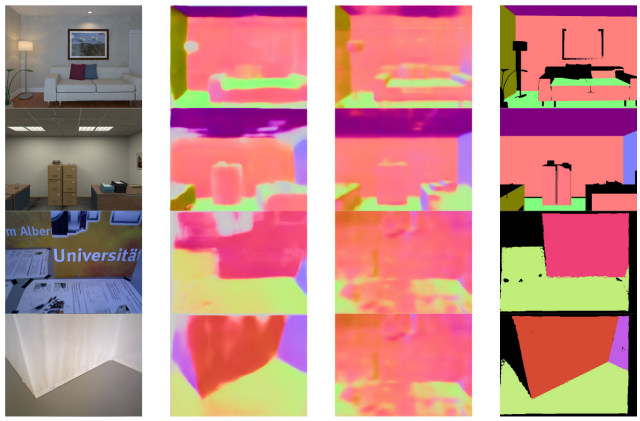
In contrast to other work, based on Manhattan frames for rotation estimation this heuristic allows us to fall back to a purely feature based pose estimation in case the estimate from the MW pose estimation is wrong or not available.

VI. Experiments

Implementation details: We train the network implemented for normal estimation based on the ScanNet [31] dataset with a batch size of 32 for 8 epochs. The backbone is pretraind on ImageNet [32] for feature extraction and PlaneReconstruction [14] for understanding plane regions. We use the Adam optimizer with a learning rate of 10^{-4} and a weight decay of 10^{-5}. Our model is trained in an end-to-end manner and can predict normal maps in real-time. As a baseline we use the original GeoNet [15] model trained by the authors for 400 k iterations on NYU-DepthV2 [33]. Models used in the experiments are not fine-tuned on other datasets. All experiments were carried out with an Intel Core i7-8700 CPU (with @3.20 GHz) and a NVIDIA 2080 Ti GPU. We run each sequence 5 times and show median results for the accuracy of the estimated trajectory. We evaluate our proposed SLAM system on public datasets and compare its performances with other state-of-the-art methods. The evaluation metrics used in the experiments are the absolute trajectory error (ATE) and the relative pose error (RPE) [17], which measure the absolute and relative pose differences between the estimated and the ground truth motion.

Evaluation and datasets: In order to evaluate our method, on the one hand, we compare against several monocular SLAM frameworks, as CNN-SLAM [19] that connects SLAM with predicted depth maps based on keyframes, LSD-SLAM [2] that is popular direct method and ORB-SLAM [1]. We align the trajectories for ORB-SLAM, LSD and the proposed method to the ground truth trajectories using a similarity transformation [1] due to the unknown real scale. On the other hand, we run our SLAM architecture with different normal maps to evaluate the importance of accurate normals, by switching our normals with the ones from the state-of-the-art, but not real-time capable network, GeoNet [15] and normal maps computed from the depth maps provided by the dataset using [34].

- ICL-NUIM dataset [16] is a synthetic indoor datasets that provide RGB images, depth maps and ground-truth camera poses. There are two scenes, named "living room" and "office" which are noted as "lr" and "of" in our experiments.



(a) Input (b) Ours (c) GeoNet [15] (d) AHC [34]

Fig. 5. Results of normal prediction model on ICL-NUIM (top) and TUM-RGBD (bottom) scenes for different approaches.

TABLE I
PERFORMANCE OF THE SURFACE NORMAL PREDICTION ON THE SCANNET [31] TEST SET

Error			Accuracy		
mean	median	rmse	$<11.25^\circ$	$<22.5^\circ$	$<30^\circ$
15.2°	7.8°	24.4°	0.672	0.797	0.841

- TUM RGB-D dataset [17] was collected using a real RGB-D sensor in real scenes as well as specially designed scenes to challenge current SLAM algorithms, featuring challenging scenes with good structure, but without texture.
- HRBB4 dataset [35] which has 12,000 frames of 640×320 pixels recorded by a monocular camera in a corridor.

A. Normal Prediction

Fig. 5 presents qualitative results on unseen images of different normal estimation methods. In our method, we mask out the lampshade (first row) and small boxes (second row), as these regions are classified as non-planar. The first two rows, show common examples for indoor environments. Both of them show good results, GeoNet shows smaller inaccuracies. For the last two rows, which are very uncommon scenes, the planar region detection and normal estimation of our model are still generating reasonable results, while the quality of the normal predictions from GeoNet decreased severely.

The agglomerative hierarchical clustering (AHC) algorithm [34] is an efficient method to detect planes in a depth map. However it difficult to detect planes (like in the third an forth row) where the quality of the depth maps decrease due to a highly slanted surface. In the Table I, the performance of the network is evaluated on the ScanNet [31] dataset generated by [36] against the ground truth.

B. Pose Estimation

In order to evaluate our method in different environments, we select structured image sequences from the ICL-NUIM dataset [16] and the TUM RGB-D dataset [17]. Table II shows the RMSE for all methods on several sequences, 'lr' and 'of' stand for the living room and office room sequences in the ICL-NUIM dataset. 's-t-near' and 's-not-near' are the structure-texture-near and structure-notexture-near sequences in the TUM RGB-D dataset, respectively. 's-t-near' and 's-t-far' are showing the same environment consisting of multiple textured planes, 's-not-near' and 's-not-far' consist of a similar structure, but without texture.

From the six row to the eight row, different normal maps are given to the same backbone. It is obvious that using AHC-based normal maps (obtained from ground truth depth map) obtain the best results compared to other methods. It also shows the potential of our SLAM architecture, given precise normal maps. Performances from $-w$ Ours (combination of our normal prediction network and the backbone) is more robust than $-w$ GeoNet (combination of GeoNet and the backbone), especially in the 's-not-far' sequence. For those non-textured images, it is difficult for GeoNet to predict accurate normals. In the backbone, conic areas around each axis are used during the sphere mean-shift method to filter the normal maps, this allows the handling of normal outliers up to a certain point. In cases were the number of outliers is too high, it is difficult to obtain a good rotation from the back-end of the architecture. Different to the monocular methods, LPVO [12] works directly with RGB-D images, which prevents scale drift and allows tracking directly on the depthmap. In comparison our method achieves comparable performance without the use of a depth sensor.

Our method obtains good results and shows robust performance in all five sequences. In the first two sequences, the difference between the point based ORB-SLAM and our method, that connects structure and geometric information, is not significant. However ORB-SLAM is not able to find enough point matches over a sequence of frames and looses tracking in some of the sequences, these are marked with a cross ($\times$). Our method, which additionally uses lines for the translation estimation achieves even better results.

When we compare -w Ours, -w GeoNet with ORB-SLAM in textured sequences, they obtain similar results because those sequences have a sufficient number of features distributed evenly on each frame. However for indoor environments, like Fig. 4, it is difficult to obtain enough point features because of large non-textured planar regions. In the 'of-kt3' sequence, there is little change in the first 57 frames, so ORB-SLAM cannot initialize successfully, because it needs enough points for homography/fundamental model selection. After initialization, it is also challenging for ORB-SLAM to track via the point-based motion model. For our case, the initial rotation matrix is estimated by the mean-shift method instead of estimating the essential or homography matrices. This means we can deal with pure rotational motion. Furthermore, points and line segments are used for 3 DoFs translation only, which is more robust even in large non-textured scene.

In order to present the robustness of our method, we compute the RPE for those sequences, which can be processed robustly by ORB-SLAM and our method. For 's-t-far' and 's-t-near' that are textured sequences, ORB-SLAM and the proposed method have similar performences. The relative translation errors for

TABLE II
COMPARISON OF TRANSLATION RMSE (M) FOR ICL-NUIM [16] AND TUM RGB-D [17] SEQUENCES USING MONOCULAR CAMERA. WE USE **BOLD** NUMBERS TO MARK THE BEST RESULT PER SEQUENCE. $-w$ MEANS THAT THE PROPOSED FRAMEWORK USES THE CORRESPONDING SURFACE NORMALS. $\times$ INDICATES THAT THE ALGORITHM FAILS DUE TO LOST TRACKING

Methods	lr-kt1	lr-kt2	lr-kt3	of-kt1	of-kt2	of-kt3	s-t-near	s-t-far	s-not-near	s-not-far
LSD-SLAM [2]	0.059	0.323	-	**0.157**	0.213	-	-	0.214	-	-
CNN-SLAM [19]	0.540	0.211	-	0.790	0.172	-	-	0.037	-	-
ORB-SLAM [1]	0.024	0.061	0.035	$\times$	**0.031**	0.326	0.016	0.015	$\times$	$\times$
LPVO [12]	*0.04*	*0.03*	*0.10*	*0.05*	*0.04*	*0.03*	*0.11*	*0.17*	*0.08*	*0.07*
-w Ours	**0.016**	**0.045**	0.046	$\times$	**0.031**	0.065	**0.014**	**0.014**	**0.065**	**0.281**
-w GeoNet [15]	$\times$	0.047	**0.026**	$\times$	0.048	**0.043**	0.107	**0.014**	0.068	$\times$
-w AHC [34]	*0.016*	*0.028*	*0.020*	*0.763*	*0.021*	*0.020*	*0.015*	*0.013*	*0.015*	*0.220*

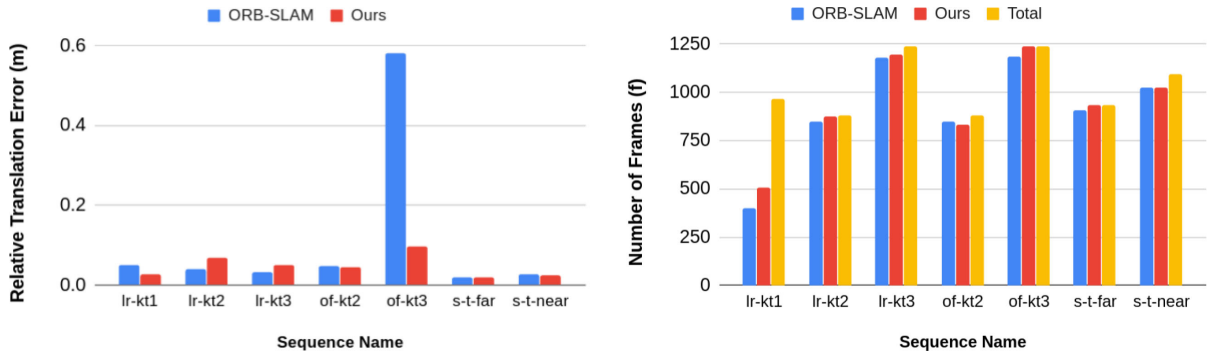


Fig. 6. Relative translational error comparison between ORB-SLAM and our method on different sequences (left) and a comparison of the average runtime length on each sequence before tracking is lost (right).

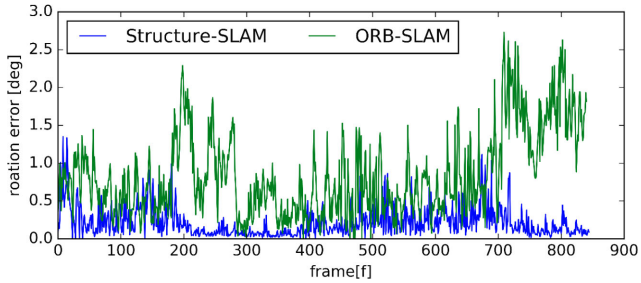


Fig. 7. rotation error comparison between ORB-SLAM and our method on sequence lr-k2.

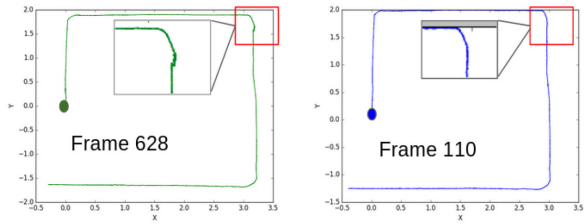


Fig. 8. The estimated trajectories of the camera on the HRBB4 [35] dataset. Left: ORB-SLAM, Right: Structure-SLAM.

the sequence 'of-kt3' in Fig. 6 (left) is significantly larger for ORB-SLAM, which corresponds to the result presented in Fig. 4. As shown in Fig. 7, the proposed method, Structure-SLAM, is more stable in rotation estimation compared with ORB-SLAM.

We also compare the number of frames tracked by different methods. Compared with ORB-SLAM, our method retrieves the camera pose more reliable. Especially in 'lr-kt2,' 'of-kt3' and 's-t-far,' our method initializes fast and tracks all frames in the sequences, as can be seen in sequence 'of-kt3' in Fig. 6 on the right. Similar results can be found for HRBB4 in Fig. 8. Compared with ORB-SLAM which only initializes after the 628th frame, our method is able to initialization much earlier around frame 110. Furthermore, the proposed method shows a more stable behaviour in the upper right corner of the corridor where the environment changes drastically.

VII. CONCLUSION

We have proposed a SLAM system for monocular cameras based on points, lines and surface normals. Using the Manhattan World assumption for rotation estimation and point and line features for windowed translation estimation we achieve state-of-the-art performance. We have shown that normals, learned from a single RGB image, can be used to estimate the rotation between frames leveraging the MW assumption. Compared to other state-of-the-art methods based on global rotation estimation, in our method there exists a fallback level using points and lines to estimate the full pose, in case no Manhattan frame can be found. This enables the tracking over short sequences to later re-localize within the Manhattan world. In the future, global bundle adjustment could be used to correct the frames during these sequences without global frames. Furthermore, we would like to leverage the learned structure information for the translation estimation as well.

REFERENCES

[1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, 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 *Proc. Springer Eur. Conf. Comput. Vision*, 2014, pp. 834–849.

[3] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, Mar. 2018.

[4] R. Gomez-Ojeda, F.-A. Moreno, D. Zuñiga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "PL-SLAM: A stereo SLAM system through the combination of points and line segments," *IEEE Trans. Robot.*, vol. 35, no. 3, pp. 734–746, Jun. 2019.

[5] Y. Lu and D. Song, "Robust RGB-D odometry using point and line features," in *Proc. IEEE Int. Conf. Comput. Vision*, 2015, pp. 3934–3942.

[6] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Morenonoguer, "PL-SLAM: Real-time monocular visual SLAM with points and lines," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 4503–4508.

[7] P. F. Proença and Y. Gao, "Probabilistic RGB-D odometry based on points, lines and planes under depth uncertainty," *Robot. Auton. Syst.*, vol. 104, pp. 25–39, 2018.

[8] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular SLAM," *Robot.: Sci. Syst. VI*, vol. 2, no. 3, pp. 73–80, 2010.

[9] H. Li, J. Yao, J.-C. Bazin, X. Lu, Y. Xing, and K. Liu, "A monocular SLAM system leveraging structural regularity in manhattan world," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2518–2525.

[10] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, "StructSLAM: Visual SLAM with building structure lines," *IEEE Trans. Veh. Technol.*, vol. 64, no. 4, pp. 1364–1375, Apr. 2015.

[11] J. Straub, N. Bhandari, J. J. Leonard, and J. W. Fisher, "Real-time manhattan world rotation estimation in 3D," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2015, pp. 1913–1920.

[12] P. Kim, B. Coltin, and H. J. Kim, "Low-drift visual odometry in structured environments by decoupling rotational and translational motion," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 7247–7253.

[13] P. Kim, B. Coltin, and H. J. Kim, "Linear RGB-D SLAM for planar environments," in *Proc. Springer Eur. Conf. Comput. Vision*, 2018, pp. 333–348.

[14] Z. Yu, J. Zheng, D. Lian, Z. Zhou, and S. Gao, "Single-image piece-wise planar 3D reconstruction via associative embedding," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 1029–1037.

[15] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia, "GeoNet: Geometric neural network for joint depth and surface normal estimation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 283–291.

[16] A. Handa, T. Whelan, J. McDonald, and A. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 1524–1531.

[17] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2012, pp. 573–580.

[18] G. Klein and D. Murray, "Parallel tracking and mapping on a camera phone," in *Proc. IEEE Int. Symp. Mixed Augmented Reality*, 2009, pp. 83–86.

[19] K. Tateno, F. Tombari, I. Laina, and N. Navab, "CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 6565–6574.

[20] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "CodeSLAM–learning a compact, optimisable representation for dense visual SLAM," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 2560–2568.

[21] L. Ma, C. Kerl, J. Stückler, and D. Cremers, "CPA-SLAM: Consistent plane-model alignment for direct RGB-D SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1285–1291.

[22] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, "Keyframe-based dense planar SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 5110–5117.

[23] Y. Zhou, L. Kneip, C. Rodriguez, and H. Li, "Divide and conquer: Efficient density-based tracking of 3D sensors in Manhattan worlds," in *Proc. Asian Conf. Comput. Vision*, 2016, pp. 3–9.

[24] R. Toldo and A. Fusiello, "Robust multiple structures estimation with J-linkage," in *Proc. Springer Eur. Conf. Comput. Vision*, 2008, pp. 537–547.

[25] K. Joo, T.-H. Oh, J. Kim, and I. S. Kweon, "Globally optimal manhattan frame estimation in real-time," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 1763–1771.

[26] P. Kim, B. Coltin, and H. J. Kim, "Visual odometry with drift-free rotation estimation using indoor scene regularities," in *Proc. Brit. Mach. Vision Conf.*, 2017, pp. 7–19.

[27] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE Int. Conf. Comput. Vision*, 2011, pp. 2564–2571.

[28] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 4, pp. 722–732, Apr. 2010.

[29] L. Zhang and R. Koch, "An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency," *J. Vis. Commun. Image Representation*, vol. 24, no. 7, pp. 794–805, 2013.

[30] R. Murartal 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, Oct. 2017.

[31] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2432–2443.

[32] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2009, pp. 248–255.

[33] P. K. N. Silberman, D. Hoiem, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Springer Eur. Conf. Comput. Vision*, 2012, pp. 746–760.

[34] C. Feng, Y. Taguchi, and V. R. Kamat, "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 6218–6225.

[35] Y. Lu, D. Song, and J. Yi, "High level landmark-based visual navigation using unsupervised geometric constraints in local bundle adjustment," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 1540–1545.

[36] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa, "PlaneNet: Piecewise planar reconstruction from a single RGB image," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 2579–2588.