MSC-VO: Exploiting Manhattan and Structural Constraints for Visual Odometry

Joan P. Company-Corcoles, Emilio Garcia-Fidalgo, and Alberto Ortiz, *Member, IEEE*

***Abstract*—Visual odometry algorithms tend to degrade when facing low-textured scenes —from e.g. human-made environments—, where it is often difficult to find a sufficient number of point features. Alternative geometrical visual cues, such as lines, which can often be found within these scenarios, can become particularly useful. Moreover, these scenarios typically present structural regularities, such as parallelism or orthogonality, and hold the Manhattan World assumption. Under these premises, in this work, we introduce MSC-VO, an RGB-D -based visual odometry approach that combines both point and line features and leverages, if exist, those structural regularities and the Manhattan axes of the scene. Within our approach, these structural constraints are initially used to estimate accurately the 3D position of the extracted lines. These constraints are also combined next with the estimated Manhattan axes and the reprojection errors of points and lines to refine the camera pose by means of local map optimization. Such a combination enables our approach to operate even in the absence of the aforementioned constraints, allowing the method to work for a wider variety of scenarios. Furthermore, we propose a novel multi-view Manhattan axes estimation procedure that mainly relies on line features. MSC-VO is assessed using several public datasets, outperforming other state-of-the-art solutions, and comparing favourably even with some SLAM methods.**

***Index Terms*—Localization, mapping, SLAM.**

I. Introduction

VISUAL Odometry (VO) is the process of estimating the trajectory of a camera within an environment by analysing the sequence of images captured. VO is a key part of a more sophisticated family of methods known as Visual Simultaneous Localization and Mapping (V-SLAM), which typically combine VO with a loop closure detection approach to perform both tasks at the same time. When a previously seen place is revisited, the accumulated drift produced by VO can be alleviated incorporating new constraints into the optimization stage. However, this strategy does not completely remove the camera pose error, so that the overall performance of any SLAM system gets determined by the VO accuracy [1].

Manuscript received September 9, 2021; accepted January 1, 2022. Date of publication January 13, 2022; date of current version February 2, 2022. This work is partially supported by EU-H2020 projects BUGWRIGHT2 (GA 871260) and ROBINS (GA 779776), and by project PGC2018-095709-B-C21 (funded by MCIU/AEI/10.13039/501100011033 and FEDER "Una manera de hacer Europa"). This publication reflects only the authors views and the European Union is not liable for any use that may be made of the information contained therein. *(Corresponding author: Joan P. Company-Corcoles.)*

The authors are with the Department of Mathematics and Computer Science, University of the Balearic Islands, 07122 Palma, Spain, and also with the IDISBA, Institut d'Investigacio Sanitaria de les Illes Balears, 07120 Palma de Mallorca, Spain (e-mail: joanpep.company@uib.es; emilio.garcia@uib.es; alberto.ortiz@uib.es).

Digital Object Identifier 10.1109/LRA.2022.3142900

Many VO and SLAM systems rely on point features because of their wider applicability in general terms [2]. However, in low-textured scenarios, their performance decrease due to the low number of points detected [3]. In this regard, the combination of point and line features has been demonstrated to reduce the number of tracking failures in these environments [3]–[5]. A complementary technique is to take profit of the structural constraints typically present in these scenarios, such as parallelism and/or orthogonality, through a pose-graph optimization strategy [6]. Another well-known strategy, which can be used to reduce the rotation drift in human-made environments, is to adopt the Manhattan World (MW) assumption [7]. This hypothesis assumes a Cartesian coordinate system for the environment and that most part of the geometrical entities present in the scene align to one of its axes, named as Manhattan Axes (MA). This assumption is fundamentally used during the tracking stage [8]–[12]. Nonetheless, these methods do not usually take into account that some indoor environments are not strictly adhering to this assumption, leading to degradation in accuracy or even to tracking failures [13].

Based on the above, this work exploits the benefits of point and line features used in combination with structural constraints and MA alignment to propose a new RGB-D VO framework named as MSC-VO from *Manhattan and Structural Constraints - Visual Odometry*. As already said, the proposed method relies on point and line features, mostly because of their low detection times. Additionally, to address the inaccuracies in depth estimation which result from occlusions, depth discontinuities and RGB-D noise, which is even more notorious for lines than for points, we propose a two-step procedure that can be briefly stated as (1) for each line detected in the image plane, we estimate its 3D line endpoints using a robust fitting procedure, and (2) we next refine the estimated endpoints using the scene structural regularities. Moreover, our approach proposes a novel local map optimization stage which combines point and line reprojection errors along with structural regularities and MA alignment, resulting into more precise local trajectory estimations. Unlike other approaches, where the MW constraints are used during the tracking stage, our solution incorporates the MW assumption during local map optimization, which allows us not to slow down the tracking, which typically requires real-time operation to perform properly. Finally, we propose a novel multi-view MA initialization procedure. A first illustration of the performance of MSC-VO can be found in Fig. 1.

In brief, the most important contributions of this work are:

1) A robust RGB-D VO framework for low-textured environments, which can improve the pose accuracy when

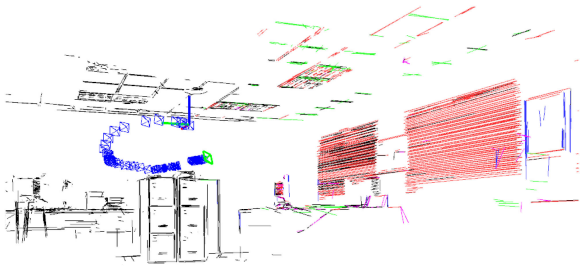

Fig. 1. Example of local map generated by MSC-VO. For a better understanding, only line features are shown. The map corresponds to a human-made environment, which, as expected, is rich in line features. Furthermore, parallel and orthogonal relations between lines are highly present due to the design of these environments. The Manhattan axis line associations are shown using red, green and blue colours, while non-associated lines are labelled in purple- Those lines not included in the covisibility graph are shown in black.

structural regularities and MA alignment are present in the scene. Otherwise, our solution remains operational, as will be shown in the experimental results section.

2) A 3D line endpoint computation method based on the structural information present in the scene.
3) An accurate and efficient 3D local map optimization strategy, which combines reprojection errors with structural constraints and MA alignment.
4) A novel MA initialization procedure that refines the estimation of the traditionally employed Mean Shift algorithm by using multiple frame observations in a multi-graph non-linear least squares formulation.
5) An extensive evaluation of the proposed approach on several public datasets and a comparison with other VO and SLAM state-of-the-art methods.
6) As an additional contribution, the source code MSC-VO is available online for the community.[1]

The rest of the paper is organized as follows: Section II overviews most relevant related works in the field; the proposed framework is introduced in Section III; Section IV reports on the results obtained; and, finally, Section V concludes the paper and suggests some future research lines.

II. Related Work

VO and Visual SLAM algorithms can be roughly classified into two main categories: feature-based and direct methods [1]. Among them, feature-based approaches are typically more robust to illumination changes than direct methods. Despite their impressive results on well-textured scenarios [2], their performance decreases when dealing with low-textured environments [4]. Due to this reason, some authors have opted for the combination of points with other geometric entities, e.g. lines [3]–[5], planes [6], or both [13].

Assuming a MW in human-made environments has demonstrated to be very effective to reduce the rotational drift [8]–[12]. Generally, this premise is taken into account during the tracking stage, being usually decoupled the rotation and the translation parts. Different strategies have been proposed to estimate and track the MA. For example, Zhou *et al.* [8] propose a single Mean Shift iteration that tracks the dominant MA for each frame by using a set of normal vectors. The translational part is computed through three simple 1D density alignments. In [9], the translation estimation is improved through a Kanade-Lucas-Tomasi (KLT) feature tracker. However, these two approaches require the existence of multiple orthogonal planes per frame. To solve this issue, Kim *et al.* [10] combine line and plane features within a Mean Shift-based approach. In addition, they propose to use the reprojection error from the tracked points in the estimation of the translation. In a more recent work, they add an orthogonal plane detection and tracking method [11]. Another solution to improve the tracking accuracy is presented in [12], where the authors introduce the concept of plane orientation relevance to track the MA. More recently, other features are employed in [14], which combines vanishing directions of 3D lines and plane normal vectors to track the MA. In this regard, [13], [14] report that the use of planar features increases the accuracy of the tracking, and, additionally, contributes positively to the estimation of the MA. However, plane detection usually relies on depth estimation, which can fail in some scenarios due to the range limitations and noise of RGB-D cameras [13]; contrarily, line features can be detected directly from the available images. Besides, planes and lines detection require similar computational times if the number of planes is not high; otherwise, the complexity of the underlying processes leads to larger running times for planes. Additionally, to detect and track the MA robustly, these methods typically combine planes with other features, such as lines. Consequently with the aforementioned, our pipeline combines points and lines.

Moreover, the accuracy of the estimated MA determines the correctness of the system during its operation. To reduce these inaccuracies, Li *et al.* [15] describe a method that refines the reference MA by tracking it on each frame, and, thus, obtains multiple reference MA, which are later fused by Kalman Filtering. Following this idea, we propose to refine the position of 3D lines during MA estimation by using a graph-based non-linear error function that includes multiple views of the lines. However, unlike [15], we estimate the MA only once and they remain fixed along the whole sequence.

Local map optimization is usually performed in the back-end to reduce the errors produced during the tracking stage. In this regard, some approaches refine the pose of some previous frames after tracking the MA. For instance, in [16], the authors propose a line-based local optimization method to refine only the translation. However, the rotation is still computed using the decoupled tracking strategy. Moreover, other approaches [6], [14] perform this local optimization by combining point and plane features in conjunction with structural constraints, which have been shown to achieve better results than the decoupled scheme [14].

There exist indoor environments that do not strictly conform to the MW assumption. In these cases, the performance of approaches purely based on it degrades, even leading to tracking failures. To overcome this issue, Zhang *et al.* [6] propose using parallel and perpendicular constraints as an alternative to the MW assumption. Despite its advantages, this method can not reduce the long-term rotation error as the MW assumption does. Another solution is proposed in [13], where the authors use either a decoupled or a non-decoupled tracking strategy depending on whether the scene meets the MW assumption. These strategies permit these works to not only focus on a specific environment.

[1] [Online]. Available: http://github.com/joanpepcompany/MSC-VO

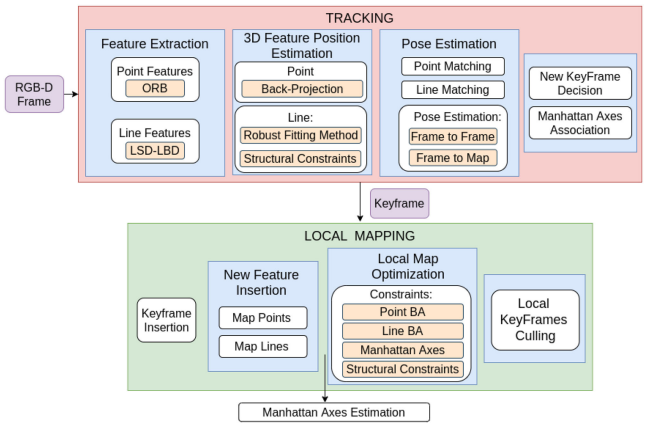

Fig. 2. Overview of MSC-VO.

The related works reviewed above suggest the use of the MW assumption to increase the localization accuracy of VO and SLAM methods. However, using this assumption as a primary source in the tracking procedure can lead to failures in some scenes where the MW assumption is not satisfied, what can restrict those solutions for certain specific environments. As a solution, we propose the incorporation of the MA in local map optimizations. Additionally, we take inspiration from [14], which reports the structural constraints as beneficial for the pose refinement process. To this end, we propose a novel local map optimization approach that combines the point and line reprojection error, the MA alignment and the structural constraints of the scene. Allowing that, the punctual dissatisfaction of some of these constraints does not affect the overall performance. As a result, our method leads to higher localization accuracy and allows working in a wider range of scenarios.

III. MSC-VO Overview

MSC-VO is built on top of the tracking and local mapping components of ORB-SLAM2 [2]. Therefore, it comprises two threads running in parallel, as it is illustrated in Fig. 2. Further details on MSC-VO can be found next.

A. Tracking

The tracking thread is in charge of estimating the position of every frame captured. Additionally, this module decides whether a new keyframe needs to be created. It also associates each new map line with one of the MA, if possible.

1) Feature Extraction: Every frame I_t coming from the RGB-D sensor at time t consists of a colour image I_t^c and a depth image I_t^d. Point and line features are extracted from I_t^c. Points are detected and described using ORB [17], while lines are detected using the Line Segment Detector (LSD) [18] and described using the binary form of the Line Band Descriptor (LBD) [19]. In the following, the location of a point i in image coordinates is denoted as p_i, while each line segment j detected in the image plane is represented by a start point s_j and an end point e_j. Additionally, the normalized line l_j is expressed as:

$$l_j = \frac{e_j - s_j}{\|e_j - s_j\|}. \tag{1}$$

2) 3D Feature Position Estimation: Once points and lines have been detected and described, their 3D positions in camera coordinates are obtained. A point p_i is backprojected using as depth the value corresponding to its 2D position in I_t^d. The resulting 3D position in camera coordinates is denoted as P_i^c. Since lines are more affected than points by depth discontinuities and occlusions, this simple procedure can end up with inaccurate 3D lines. To reduce this effect, we propose a robust two-step method to compute the 3D line endpoints.

First, for every line segment j, we calculate an initial 3D position for its endpoints, denoted by $\{S_j^c, E_j^c\}$, by backprojecting a subset of the points that conforms the line in the image and, next, performing a robust fitting step as in [14]. The 3D normalized line L_j^c is computed similarly to (1). Next, we employ the structural constraints of the scene to refine each detected line. We start by associating parallel and perpendicular lines. To this end, for every possible pair of lines (L_m^c, L_n^c) detected in the current image, we compute the cosine of the angle between the two direction vectors by means of the dot product:

$$\cos\left(L_m^c, L_n^c\right) = \frac{L_m^c \cdot L_n^c}{\|L_m^c\| \|L_n^c\|}. \tag{2}$$

We choose only those pairs (L_m^c, L_n^c) whose cosine value is close to 0 or 1 representing, respectively, perpendicular or parallel lines. The selected pairs are employed to refine their line endpoints by means of non-linear optimization. We use the Levenberg–Marquardt algorithm implemented in g2o [20] to this end. Formally, we define the orientation discrepancy d between lines L_m^c and L_n^c as:

$$d(L_m^c, L_n^c) = \left|\cos\left(L_m^c, L_n^c\right)\right|. \tag{3}$$

Let us denote $\mathbb{L}_\perp$ and $\mathbb{L}_\parallel$ as the sets of, respectively, valid perpendicular and valid parallel line pairs. Given a pair $(L_m^c, L_n^c) \in \mathbb{L}_\perp$, the error term $\mathbf{E}_{m,n}^\perp$ is defined as:

$$\mathbf{E}_{m,n}^\perp = d(L_m^c, L_n^c) \cdot \omega_n^{-1}, \tag{4}$$

where ω_n weights the error term in accordance to the line response returned by the LSD algorithm for segment n. Similarly, for another pair $(L_m^c, L_n^c) \in \mathbb{L}_\parallel$, the error term $\mathbf{E}_{m,n}^\parallel$ is defined as follows:

$$\mathbf{E}_{m,n}^\parallel = \sqrt{1 - d^2(L_m^c, L_n^c)} \cdot \omega_n^{-1}. \tag{5}$$

where $d(\cdot,\cdot) \in [0,1]$.

We define $\mathbf{L}$ as the set of variables to be optimized, which includes those lines that have at least one structural association either on $\mathbb{L}_\perp$ or $\mathbb{L}_\parallel$. We then compute the optimal line end points of $\mathbf{L}$ by minimizing the following cost function:

$$\mathbf{L} = \underset{\mathbf{L}}{\operatorname{argmin}} \left(\sum_{(i,j)\in\mathbb{L}_\perp} \rho\left(\mathbf{E}_{i,j}^\perp\right) + \sum_{(k,o)\in\mathbb{L}_\parallel} \rho\left(\mathbf{E}_{k,o}^\parallel\right) \right), \tag{6}$$

where ρ is the Huber loss function to reduce the influence of outliers. Fig. 3 summarizes the notation of points and lines regarding frame coordinates, and the two error terms defined in this section. As it will be shown in Section IV, using the outlined procedure, the 3D lines estimation accuracy improves, benefiting the whole system.

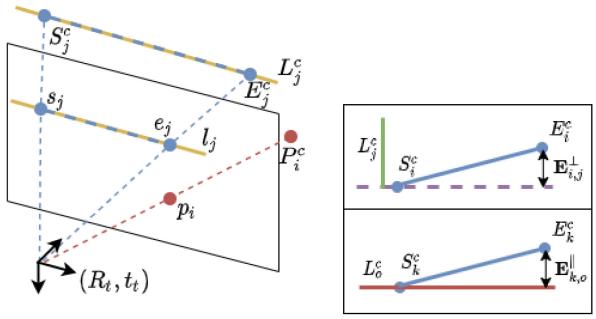

Fig. 3. The left drawing illustrates the notation used for 2D and 3D features, while the right drawings illustrate the line endpoints error terms. S_j^c and E_j^c are the line endpoints to optimize. The right-top drawing shows the error term $\mathbf{E}_{i,j}^{\perp}$ as the cosine of the angle between the normalized line defined by S_i^c and E_i^c and a perpendicular line, shown in green, for a perpendicular association L_j^c. The right-bottom drawing illustrates the parallel error term $\mathbf{E}_{k,o}^{\parallel}$, calculated as the sine of the angle between the normalized line defined by S_k^c and E_k^c and a parallel association L_o^c. (Both cases assume $\omega = 1$.).

3) Pose Estimation: Once features are extracted, an optimization procedure is carried out to estimate the current camera orientation $\mathbf{R}_t \in SO(3)$ and translation $\mathbf{t}_t \in \mathbb{R}^3$. Initially, map points and lines observed in the previous frame are projected to the current frame, assuming a constant velocity motion model. Next, two sets of 2D-3D correspondences, one for points as in [2] and one for lines as in [5], are computed. These associations are then employed to optimize the current camera pose, minimizing the following cost function:

$$\{\mathbf{R_t}, \mathbf{t_t}\} = \underset{\mathbf{R_t}, \mathbf{t_t}}{\operatorname{argmin}} \left(\sum_{i \in \mathbb{P}} \rho\left(\mathbf{E}_i^p\right) + \sum_{j \in \mathbb{V}} \rho\left(\mathbf{E}_j^l\right) \right), \quad (7)$$

where $\mathbb{P}$ and $\mathbb{V}$ are, respectively, the sets of all point and line matches. The error term for the observation of a map point i is defined as:

$$\mathbf{E}_i^p = \| p_i - \pi(R_t P_i^w + t_t) \|^2 \cdot \omega_i^{-1}, \quad (8)$$

where $P_i^w \in \mathbb{R}^3$ is the point in world coordinates corresponding to $p_i \in \mathbb{R}^2$ and ω_i weights the error term in accordance to the response of the ORB detector. The projection function π transforms a 3D point P_i^c in camera coordinates into the image plane using the camera calibration parameters [21]. On the other side, the error term for an observed map line j in the current frame is defined as:

$$\mathbf{E}_j^l = \left\| n_j \cdot \pi(R_t S_j^w + t_t),\ n_j \cdot \pi(R_t E_j^w + t_t) \right\|^2 \cdot \omega_j^{-1}, \quad (9)$$

where $L_j^w = \{S_j^w, E_j^w\}$ is the map line in world coordinates that matches the 2D segment l_j with normal vector n_j. Once the camera pose has been estimated, we project the local map into the current frame to obtain more correspondences, as performed in [2]. The pose is optimized again with the resulting matches.

4) Keyframe Insertion: Once the camera pose has been estimated, the current frame is evaluated to decide whether it should be considered as a new keyframe. We use a similar policy as ORB-SLAM2 [2], but incorporating line correspondences. Unlike ORB-SLAM2, we do not use the condition of a minimum number of features tracked. The rationale behind this idea is that the proposed method is focused on low-textured environments, where typically the number of features tracked per frame can change drastically between scenes. Therefore, it is not possible to fix a reasonable threshold. Instead, we propose to use the ratio between the current frame features that are being tracked in the map, and the sum of these features with the ones that could be potentially created. Once a new keyframe is generated, points and lines are included in the local map and redundant features are culled, as performed in [2]. For each new map line, we search for parallel or perpendicular line correspondences in the local map. Additionally, each line is also associated to an MA, if possible, as explained in the next section.

5) Manhattan Axes Association: Given $\mathcal{M} = \{\mathrm{MA}_0, \mathrm{MA}_1, \mathrm{MA}_2\}$ as the set of Manhattan Axes, when a new keyframe is inserted, each new map line j is associated to axis $M_j \in \mathcal{M}$ whenever possible. To this end, we compare every line L_j^w with each of the three axes: if the value of expression in (3) gets close enough to 1 for axis MA_k, the line is considered as parallel to MA_k, and they are matched, i.e. $M_j = \mathrm{MA}_k$. These associations are used during local map optimization to reduce the camera rotation drift. Notice that, given the combination of structural constraints and this MA alignment, our approach is able to operate even if these axes are not available. The procedure to estimate these MA is explained in Section III-B2.

B. Local Mapping

Whenever a keyframe is inserted, the local mapping thread refines recent keyframe poses and landmarks by a multi-graph optimization process. Furthermore, this thread also estimates the reference MA, if required. Finally, redundant keyframes are culled using the strategy introduced in [2]. Further details can be found next.

1) Local Map Optimization: Once keyframe k is generated, the local optimization procedure refines its pose along with the poses of a set of connected keyframes $\mathcal{K}_c$ obtained from a covisibility graph [2] and all the map points $\mathcal{P}$ and lines $\mathcal{L}$ seen by those keyframes. Other keyframes that observe these points and lines but are not connected to k, denoted by $\mathcal{K}_f$, are included in the optimization, but their poses remain fixed. We denote $\mathbb{P}_k$ and $\mathbb{V}_k$ as the sets of matches between, respectively, points and lines in $\mathcal{P}$ and $\mathcal{L}$ and features in keyframe k. To introduce the structural constraints of the scene into the optimization, we define $\mathbb{L}_{\perp}^k$ and $\mathbb{L}_{\parallel}^k$ as the sets of perpendicular and parallel pairs of lines in $\mathcal{L}$, respectively, co-observed in keyframe k. Finally, we denote as $\mathbb{M}$ the set of map lines that are associated to a MA and that are seen by any keyframe in $\mathcal{K}_c$. Defining $\Gamma = \{P_i^w, L_j^w, R_l, t_l, | i \in \mathcal{P}, j \in \mathcal{L}, l \in \mathcal{K}_c\}$ as the set of variables to be estimated, the optimization problem is defined as:

$$\begin{aligned} \mathbf{\Gamma} = \underset{\mathbf{\Gamma}}{\operatorname{argmin}} \Bigg[& \sum_{k \in \{\mathcal{K}_c \cup \mathcal{K}_f\}} \left(\sum_{i \in \mathbb{P}_k} \rho\left(\mathbf{E}_i^p\right) + \sum_{j \in \mathbb{V}_k} \rho\left(\mathbf{E}_j^l\right) \right) \\ & + \sum_{z \in \mathcal{K}_c} \left(\sum_{(i,j) \in \mathbb{L}_{\perp}^z} \rho\left(\mathbf{E}_{i,j}^{\perp}\right) + \sum_{(i,j) \in \mathbb{L}_{\parallel}^z} \rho\left(\mathbf{E}_{i,j}^{\parallel}\right) \right) \\ & + \sum_{j \in \mathbb{M}} \rho\left(\mathbf{E}_{j,M_j}^{\parallel}\right) \Bigg] \end{aligned} \quad (10)$$

where $\mathbf{E}_{i,j}^{\perp}$, $\mathbf{E}_{i,j}^{\parallel}$, $\mathbf{E}_{i}^{p}$ and $\mathbf{E}_{j}^{l}$ were respectively defined in (4), (5), (8) and (9), and the MA alignment error $\mathbf{E}_{j,M_j}^{\parallel}$ is the error term corresponding to a map line j and its associated Manhattan axis $M_j \in \mathcal{M}$, calculated using (5).

2) Manhattan Axes Estimation: As already said, the Manhattan Axes comprise a set of three orthogonal directions, in world coordinates, which represent the main scene directions. These directions remain fixed over time and, therefore, the MA extraction procedure is performed only once during the whole sequence. Their estimation should be very accurate to prevent misalignments during optimization steps. In this respect, this work proposes a coarse-to-fine MA estimation strategy, where the estimation at the coarsest level is obtained extending the work by Kim *et al.* [10]. The estimated MA are then refined by considering multiple line observations along different keyframes.

For a start, a first estimation of the MA is computed from the first keyframe once it is available using the Mean Shift-based method proposed in [10]. In this first stage, the only features involved are the line direction vectors and the surface normal vectors for a selection of points defined over a grid. The normal vectors are calculated using a modified version of the approach proposed in [22], which is based on integral images to speed up calculations. This procedure is repeated for the next keyframes until valid, though typically noisy, MA are obtained.

Once the local map comprises a sufficient number of keyframes, being denoted by $\mathcal{K}_M$, a non-linear optimization procedure is performed in a second MA refinement stage, using hence the inaccurate MA computed in the first stage as initial guess. Given $\mathcal{M}$ as the set of MA, and defining $\mathbb{V}_k^{\mathrm{MA}_i}$ as the set of map lines associated to the Manhattan axis MA_i observed in keyframe k, the optimization problem can be stated as follows:

$$\mathcal{M} = \underset{\mathcal{M}}{\operatorname{argmin}} \sum_{k \in \mathcal{K}_M} \left(\sum_{j \in \mathbb{V}_k^{\mathrm{MA}_0}} \rho(\mathbf{E}_j^{\mathrm{MA}_0}) + \sum_{j \in \mathbb{V}_k^{\mathrm{MA}_1}} \rho(\mathbf{E}_j^{\mathrm{MA}_1}) + \sum_{j \in \mathbb{V}_k^{\mathrm{MA}_2}} \rho(\mathbf{E}_j^{\mathrm{MA}_2}) \right), \tag{11}$$

where the error term of a line j associated to the axis $M_j \in \mathcal{M}$ is given by:

$$\mathbf{E}_j^{M_j} = \mathbf{E}_{j,M_j}^{\parallel} + \mathbf{E}_{j,M_{j'}}^{\perp} + \mathbf{E}_{j,M_{j''}}^{\perp}, \tag{12}$$

being $M_{j'}$ and $M_{j''}$ the two other MA non-associated to line j. These two last terms enforce the orthogonality among the finally resulting axes. We reduce further the orthogonality error of the MA by means of Singular Value Decomposition (SVD), as also performed in [9], [10], [13].

IV. Experimental Results

To demonstrate the performance of MSC-VO, we conduct various experiments in both synthetic and real image sequences. Additionally, we compare its localization accuracy with some state-of-the-art VO and visual SLAM systems by means of the following datasets:

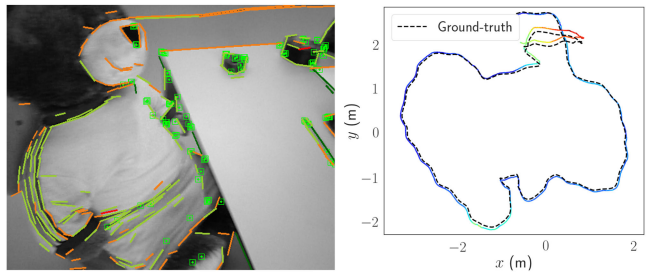

Fig. 4. (left) The MA maybe absent in a scene, e.g. a frame of the *fr3-longoffice* sequence. (right) Trajectory estimated by MSC-VO for this sequence, where no tracking failures are observed.

1) *ICL-NUIM [23]:* This is a synthetic dataset which comprises two main scenes, the living room and the office, coined in our experiments as lr and of, respectively. Furthermore, this is an indoor dataset with large structured areas, where the MW assumption and the structural constraints are highly present. Additionally, this dataset involves some low-textured challenging elements such as floors, ceilings and walls.
2) *TUM RGB-D benchmark [24]:* This is also an indoor dataset that contains several sequences with different structure, illumination and texture conditions. Unlike ICL-NUIM, this is a noisy dataset since a real RGB-D sensor was used.
3) *TAMU RGB-D [25]:* This dataset contains several indoor sequences, among which we employ *Corridor-A* and *Entry-Hall* to validate the final trajectory error (the travel distances are, respectively, 82 m and 54 m).

Regarding the MSC-VO parameters, we have used the default values provided by ORB-SLAM2 authors for the common parts, whereas the remaining parameters have been set experimentally from a single dataset, and they have been kept unaltered for the rest of sequences.

To evaluate the overall performance of MSC-VO, for the ICL-NUIM dataset and the TUM RGB-D benchmark, we use the Root-Mean-Square Error (RMSE) of the Absolute Trajectory Error (ATE) expressed in meters, as computed by the RGB-D TUM benchmark tools [24]. Regarding the TAMU RGB-D dataset, we provide the Trajectory Endpoint Drift (TED) [25], computed as the Euclidean distance between the starting and end points of the path. All the experiments have been performed on an Intel Core i7-9750H @ 2.60GHz / 16 GB RAM, without GPU parallelization.

A. General Performance

For a start, Fig. 4 illustrates the fact that the MA may be absent in a scene, leading to tracking failures for some solutions. In the case of MSC-VO, the fact of involving the MA only in local map optimizations can prevent these failures from occurring. In Fig. 4 (left), we show a frame of the *fr3-longoffice* sequence, for which the MW assumption is not very appropriate. In the image, green, red and blue colours denote the correspondences of a line with a single Manhattan axis, whereas yellow is for 3D lines that do not correspond to any axis and orange is for lines

TABLE I
RMSE OF THE ATE OF MSC-VO (IN METERS)

Sequence	PL-VO	PL-VO-Depth	MSC-VO-OR	MSC-VO
lr-kt0	0.051	0.024	0.012	**0.006**
lr-kt1	0.064	0.048	0.013	**0.010**
lr-kt2	0.054	0.030	0.010	**0.009**
lr-kt3	0.061	0.057	0.040	**0.038**
of-kt0	0.047	0.032	0.030	**0.028**
of-kt1	0.056	0.053	0.025	**0.017**
of-kt2	0.040	0.039	0.019	**0.014**
of-kt3	0.042	0.038	0.031	**0.010**
fr1-xyz	0.015	0.013	0.012	**0.010**
fr1-desk	0.023	0.022	0.024	**0.019**
fr2-xyz	0.011	0.009	0.006	**0.005**
fr2-desk	0.121	0.060	**0.023**	**0.023**
large-cabinet	0.173	0.152	0.131	**0.120**
fr3-longoffice	0.108	0.096	0.034	**0.022**

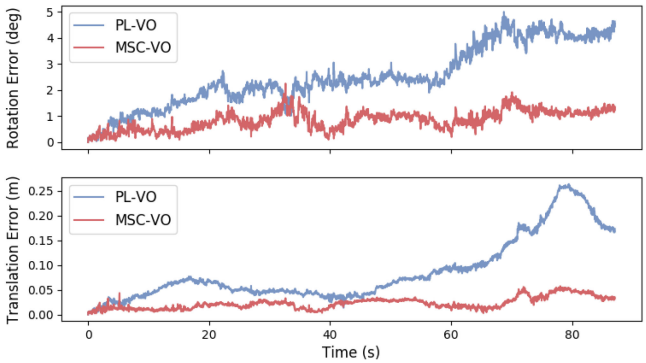

Fig. 5. Rotation and translation error over time for PL-VO and MSC-VO on the *fr3-longoffice* dataset.

TABLE II
TED ON THE TAMU RGB-D DATASET (IN METERS)

Sequence	PL-VO	PL-VO-Depth	MSC-VO-OR	MSC-VO
Corridor-A	2.76	2.29	1.38	**0.91**
Entry-Hall	1.89	1.70	1.26	**1.07**

whose 3D position has not been estimated. Fig. 4 (right) shows that MSC-VO can estimate the whole trajectory.

Next, we compare several versions of MSC-VO to show the effect of the different contributions: *PL-VO* is the part of MSC-VO that just combines point and line features; *PL-VO-Depth* combines PL-VO with the proposed 3D line endpoint estimation method; *MSC-VO-OR* corresponds to a modified version of the proposed solution, where, if a line is associated with an MA and, at the same time, it includes structural constraints, only the MA constraints are considered during the optimization (10); finally, the last case is the full version of MSC-VO. Estimation performance results for multiple sequences can be found in Table I. Moreover, Fig. 5 illustrates the rotation and translation error over time for PL-VO and MSC-VO on the *fr3-longoffice* dataset. Taking PL-VO as the baseline, MSC-VO reduces on average 76.5% and 80% the rotation and translation errors for this dataset.

Table II reports on the TED for each version of MSC-VO for the TAMU RGB-D dataset to assess its performance in long sequences. It is noticeable that each variation of our approach helps to reduce the accumulated drift along the trajectory.

TABLE III
MEAN EXECUTION TIMES (TUM RGB-D BENCHMARK)

Mean Executxion Time (ms)				
Tracking			Local Mapping	
Feat. Extrac. and 3D Pose Estimation	Camera Pose Estimation	Total (Hz)	Local Map Optimization	MA Estimation
23.2	29.1	18	152.6	206.6

TABLE IV
COMPARISON WITH OTHER APPROACHES (TED IN METERS)

Sequence	MSC-VO	ManhSLAM [13]	ORB-SLAM2 [2]
Corridor-A	0.91	**0.51**	3.17
Entry-Hall	**1.07**	1.52	2.18

On the other side, Fig. 6 shows local maps from the same cases as above for the *fr3-longoffice* sequence. The first and second plots result from, respectively, PL-VO and PL-VO-Depth. In the former case, noise from lines depth calculation affects the local map and, consequently, also the pose estimation accuracy. In the second case, this noise is of a lower magnitude, but pose inaccuracies are still observed. The third plot results from MSC-VO with the best local map and the highest localization accuracy. These results show that the local map optimization procedure not only improves the camera pose accuracy, but also refines the map lines. As a result, the misalignment that affects the PL-VO-Depth case is notably reduced. To conclude, the fourth plot shows the trajectories from each approach together with the ground truth, for a further understanding of the pose accuracy achievable on each case.

To finish, average running times for the main stages of MSC-VO can be found in Table III. The averages result from three different sequences of the TUM RBG-D benchmark. As expected, adding line features into point based VO or SLAM methods improves the accuracy and the robustness, though at the expense of increasing the computational complexity [4]. In more detail regarding our solution: (1) the robust fitting method used for 3D line pose estimation increases the low times required to extract line features and adds execution time to the feature extraction stage over other solutions; (2) regarding MA estimation, its execution time is high due to 180.4 ms that are required by the coarsest estimation step, although it needs to be computed only once (in scenarios where the MW assumption holds); and (3) despite local map optimizations require more time than other, more traditional methods based on local bundle adjustment, it can still be fast enough, as they run in a parallel thread. As a general comment, the final frame rate achieved is around 18 Hz.

B. Comparison With Other Solutions

Table V compares MSC-VO regarding localization accuracy with other state-of-the-art approaches, for which the results reported in the original works are reproduced. Best performances are indicated in bold, whereas the second best is shown in bold blue, *n.a.* refers to a not-available value, and × reports a tracking failure. The left side of the table reports on solutions based on the MA assumption that do not perform any global map optimization or loop closure detection (LCD), while the right

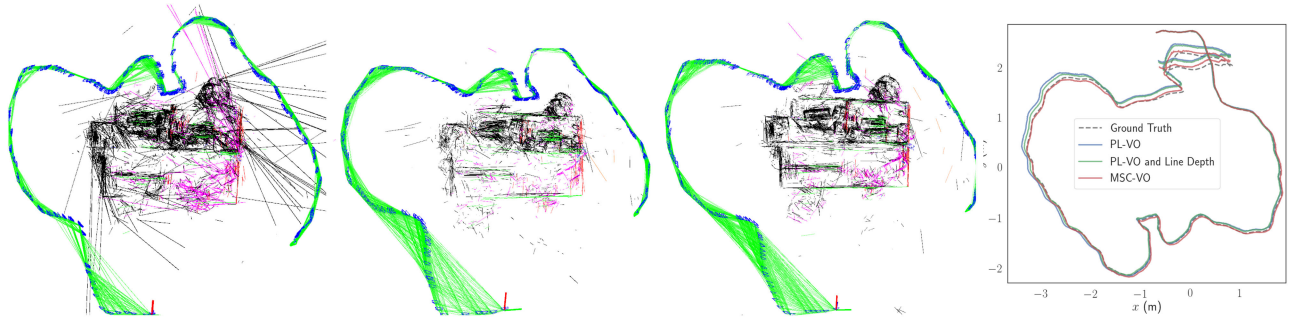

Fig. 6. (left) Local maps for the *fr3-longoffice* sequence and different versions of MSC-VO: 1st – only using points and lines (PL-VO), 2nd – PL-VO using the proposed line depth extraction procedure (PL-VO-Depth), 3rd – full MSC-VO. (right) 2D trajectories for PL-VO, PL-VO-Depth and MSC-VO, respectively shown in blue, green and red, and the ground truth in dashed grey.

TABLE V
RMSE OF THE ATE FOR MSC-VO AND OTHER STATE-OF-THE-ART APPROACHES (IN METERS)

	Without Global Optimization nor LCD						With Global Optimization and/or LCD			
Sequence	MSC-VO	OPVO [9]	LPVO [10]	MWO [8]	SReg [14]	ManhSLAM [13]	ORB-SLAM2 [2]	PS-SLAM [6]	L-SLAM [11]	InfiniTAM [26]
lr-kt0	**0.006**	×	0.015	×	**0.006**	**0.007**	0.025	0.016	0.012	×
lr-kt1	0.010	0.04	0.039	0.32	0.015	0.011	**0.008**	0.018	0.027	**0.006**
lr-kt2	**0.009**	0.06	0.034	0.11	0.020	0.015	0.023	0.017	0.053	**0.013**
lr-kt3	0.038	0.10	0.102	0.40	**0.012**	**0.011**	0.021	0.025	0.143	×
of-kt0	0.028	0.06	0.061	0.31	0.041	**0.025**	0.037	0.032	**0.020**	0.042
of-kt1	0.017	0.05	0.052	1.10	0.020	**0.013**	0.029	0.019	**0.015**	0.025
of-kt2	**0.014**	×	0.039	×	**0.011**	0.015	0.039	0.026	0.026	×
of-kt3	**0.010**	0.04	0.030	1.38	0.014	0.013	0.065	0.012	**0.011**	**0.010**
fr1-xyz	**0.010**	n.a.	n.a.	n.a.	×	**0.010**	**0.010**	**0.010**	n.a.	n.a.
fr1-desk	**0.019**	n.a.	n.a.	n.a.	×	0.027	**0.022**	0.026	n.a.	n.a.
fr2-xyz	**0.005**	n.a.	n.a.	n.a.	×	**0.008**	0.009	0.009	n.a.	n.a.
fr2-desk	**0.023**	n.a.	n.a.	n.a.	×	0.037	0.040	**0.025**	n.a.	n.a.
snot-far	0.077	0.13	0.075	0.47	**0.022**	0.040	×	**0.020**	0.141	0.037
snot-near	×	0.16	0.080	0.95	0.025	0.023	×	**0.013**	0.066	**0.022**
large-cabinet	0.120	0.51	0.279	0.83	**0.071**	0.083	0.124	**0.079**	0.140	0.512
fr3-longoffice	**0.022**	×	0.19	×	n.a.	0.049	**0.028**	n.a.	n.a.	n.a.

× and n.a. respectively stand for *tracking failure* and *not available* value. The best result for each sequence is shown in bold orange and the second best in bold blue.

side of the table is for solutions that benefit from those stages. As can be observed from the ICL-NUIM dataset, the proposed method, which only uses point and lines, achieves competitive results in contrast to other methods that rely on points, lines and planes, such as [13], [14]. Conversely, from the *fr1* and *fr2* sequences, we observe that methods relying on planes are not able to correctly estimate the MA. This is due to the fact that these methods fail to find or track orthogonal planes along the sequence. Contrarily, our approach can estimate the MA on these scenarios, except for the *fr2-desk sequence*, although the structural constraints are fully applicable in this sequence, allowing our approach to remain operational and outperform the rest of solutions. MSC-VO produces a tracking failure in the *snot-near* sequence. We do not observe this behaviour in works relying on planar features, due to the continuous presence of orthogonal planes in the sequence. It is noteworthy that MSC-VO compares favourably with more sophisticated solutions (right side of the table) even without global map optimization or LCD stages.

Finally, Table IV compares the performance of MSC-VO with other solutions in long sequences. On the one hand, we have observed that in the *Corridor-A* sequence most part of the error is due to tracking failures: in these cases, the proposed local map optimization can not fix the problem since no lines are detected in the axis where the errors take place. Despite this is not a common situation, we consider that planes can help to avoid this behaviour due to the continuous detection of the floor. However, it is important to remark that this dataset contains noisy depth data, which highly affects plane detection, and, therefore, the MA assumption does not hold for all frames. As an example, [13] tracks the pose using the MA assumption in, respectively, 15.1% and 12.5% of the frames of *Corridor-A* and *Entry-Hall*. However, MSC-VO uses the MA assumption in all the frames that at least contain one single line associated to an MA, which represents 100% of the frames in both sequences.

V. CONCLUSION AND FUTURE WORK

In this work, we have described MSC-VO, a VO that improves camera pose estimation accuracy in human-made environments. This is achieved by a combined point and line VO approach that leverages the structural regularities of the environment as well as the satisfaction of the MW assumption. On the one side, the structural constraints are used to improve line depth extraction and MA estimation. On the other side, these structural constraints are combined with point and line reprojection errors together with the MW assumption for local map optimization. All these contributions have been shown to increase the accuracy of 3D map lines position estimation and the computed trajectory for MSC-VO. Furthermore, contrary to other state-of-the-art works that use the MW in the tracking stage, our pipeline is designed to deal with the absence of the MA, allowing us to work in a wider range of environments.

Regarding future work, we plan to integrate MSC-VO with an incremental loop closure detection strategy. We are also intent to make use of the structural constraints and the MA alignment for global map optimization.

REFERENCES

[1] N. Yang, R. Wang, and D. Cremers, "Feature-based or direct: An evaluation of monocular visual odometry," 2017, *arXiv:1705.04300*.

[2] R. Mur-Artal and J. D. Tardós, "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.

[3] J. P. Company-Corcoles, E. Garcia-Fidalgo, and A. Ortiz, "LiPo-LCD: Combining lines and points for appearance-based loop closure detection," in *Proc. Brit. Mach. Vis. Conf.*, 2020, pp. 1–13.

[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 *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 4503–4508.

[5] R. Gomez-Ojeda, D. Zuñiga-Noël, F.-A. Moreno, D. Scaramuzza, and J. Gonzalez-Jimenez, "PL-SLAM: A stereo SLAM system through the combination of points and line segments," 2017, *arXiv:1705.09479*.

[6] X. Zhang, W. Wang, X. Qi, Z. Liao, and R. Wei, "Point-plane SLAM using supposed planes for indoor environments," *Sensors*, vol. 19, no. 17, 2019, Art. no. 3795.

[7] J. Coughlan and A. Yuille, "Manhattan world: Compass direction from a single image by bayesian inference," in *Proc. Int. Conf. Comput. Vis.*, 1999, pp. 941–947.

[8] 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. Vis.*, 2016, pp. 3–19.

[9] P. Kim, B. Coltin, and H. J. Kim, "Visual odometry with drift-free rotation estimation using indoor scene regularities," in *Proc. Brit. Mach. Vis. Conf.*, 2017, pp. 62.1–62.12.

[10] 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.

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

[12] L. Wang and Z. Wu, "RGB-D SLAM with manhattan frame estimation using orientation relevance," *Sensors*, vol. 19, no. 5, 2019, Art. no. 1050.

[13] R. Yunus, Y. Li, and F. Tombari, "ManhattanSLAM: Robust planar tracking and mapping leveraging mixture of manhattan frames," 2021, *arXiv:2103.15068*.

[14] Y. Li, R. Yunus, N. Brasch, N. Navab, and F. Tombari, "RGB-D SLAM with structural regularities," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 11581–11587.

[15] H. Li, Y. Xing, J. Zhao, J.-C. Bazin, Z. Liu, and Y.-H. Liu, "Leveraging structural regularity of atlanta world for monocular SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 2412–2418.

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

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

[18] R. Grompone von Gioi, J. Jakubowicz, J. 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.

[19] 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.

[20] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G^2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 3607–3613.

[21] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, MA, USA: Cambridge Univ. Press, 2003, doi: 10.1017/CBO9780511811685.

[22] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, "Real-time plane segmentation using RGB-D cameras," in *RoboCup 2011: Robot Soccer World Cup*, T. Röfer, N. M. Mayer, J. Savage, and U. Saranlı, Eds. Berlin, Germany: Springer, 2012, pp. 306–317.

[23] 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.

[24] 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.

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

[26] V. A. Prisacariu *et al.*, "InfiniTAM v3: A framework for large-scale 3D reconstruction with loop closure," 2017, *arXiv:1708.00783*.