PL-SLAM: A Stereo SLAM System Through the Combination of Points and Line Segments

Ruben Gomez-Ojeda, Francisco-Angel Moreno, David Zuñiga-Noël, Davide Scaramuzza, and Javier Gonzalez-Jimenez

***Abstract*—Traditional approaches to stereo visual simultaneous localization and mapping (SLAM) rely on point features to estimate the camera trajectory and build a map of the environment. In low-textured environments, though, it is often difficult to find a sufficient number of reliable point features and, as a consequence, the performance of such algorithms degrades. This paper proposes PL-SLAM, a stereo visual SLAM system that combines both points and line segments to work robustly in a wider variety of scenarios, particularly in those where point features are scarce or not well-distributed in the image. PL-SLAM leverages both points and line segments at all the instances of the process: visual odometry, keyframe selection, bundle adjustment, etc. We contribute also with a loop-closure procedure through a novel bag-of-words approach that exploits the combined descriptive power of the two kinds of features. Additionally, the resulting map is richer and more diverse in three-dimensional elements, which can be exploited to infer valuable, high-level scene structures, such as planes, empty spaces, ground plane, etc. (not addressed in this paper). Our proposal has been tested with several popular datasets (such as EuRoC or KITTI), and is compared with state-of-the-art methods such as ORB-SLAM2, revealing a more robust performance in most of the experiments while still running in real time. An open-source version of the PL-SLAM C++ code has been released for the benefit of the community.**

***Index Terms*—Bundle adjustment (BA), line segment features, loop closure, stereo visual simultaneous localization and mapping (SLAM).**

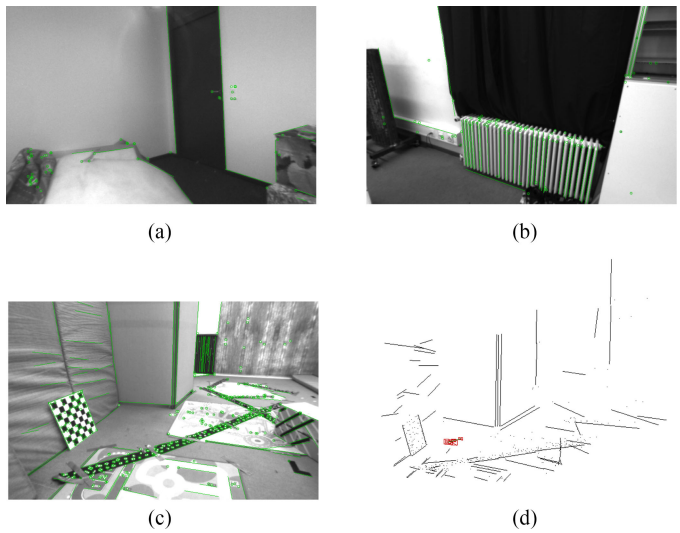


Fig. 1. Low-textured environments are challenging for *feature-based* SLAM systems based on traditional keypoints. In contrast, line segments are usually common in human-made environments, and apart from an improved camera localization, the built maps are richer as they are populated with more meaningful elements (3-D line segments). (a) *lt-easy*. (b) *euroc/V2-01-easy*. (c) *euroc/V1-01-easy*. (d) Map from (c).

I. Introduction

In RECENT years, visual simultaneous localization and mapping (SLAM) has been firmly progressing toward the degree of reliability required for fully autonomous vehicles: mobile robots, self-driving cars, or unmanned aerial vehicles. In a nutshell, the SLAM problem consists of the estimation of the vehicle trajectory given as a set of poses (position and orientation) while simultaneously building a map of the environment. Apart from self-localization, a map becomes useful for obstacle avoidance, object recognition, task planning, etc. [1].

As a first-level classification, SLAM systems can be divided into *topological* (e.g., [2]–[5]) and *metric* approaches. In this paper, we focus on the latter, which take into account the *geometric* information of the environment and build a physically meaningful map of it [6], [7]. These approaches can be further categorized into *direct* and *feature-based* systems.

Direct methods estimate the camera motion by minimizing the photometric errors between consecutive frames under the assumption of constant brightness along the local parts of the sequences (examples of this approach can be found elsewhere [8]–[10]). While this group of techniques has the advantage of working directly with the input images regardless of any intermediate representation, they are very sensitive to brightness changes (this phenomena was addressed in [11]) and constrained to narrow baseline motions. In contrast, *feature-based* methods employ an indirect representation of the images, typically in the form of point features that are tracked along the successive frames and then employed to recover the pose by minimizing the projection errors [12], [13].

Manuscript received March 20, 2018; accepted January 16, 2019. Date of publication April 2, 2019; date of current version May 31, 2019. This paper was recommended for publication by Associate Editor J. Piater and Editor T. Murphey upon evaluation of the reviewers' comments. This work was supported in part by the Spanish Government under Project DPI2017-84827-R and under Grant BES-2015-071606 and in part by the Andalusian Government under Project TEP2012-530. *(Corresponding author: Ruben Gomez-Ojeda.)*

R. Gomez-Ojeda, F.-A. Moreno, D. Zuñiga-Noël, and J. Gonzalez-Jimenez are with the Machine Perception and Intelligent Robotics Group, University of Malaga, 29016 Málaga, Spain (e-mail: rubengooj@gmail.com; famoreno@uma.es; dzuniga@uma.es; javiergonzalez@uma.es).

D. Scaramuzza is with the Robotics and Perception Group, Department of Informatics, University of Zurich, 8006 Zürich, Switzerland, and also with the Department of Neuroinformatics, University of Zurich and ETH Zurich, 8092 Zürich, Switzerland (e-mail: davide.scaramuzza@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2019.2899783

It is noticeable that the performance of any of the above-mentioned approaches usually decreases in low-textured environments in which it is typically difficult to find a large set of keypoint features. The effect in such cases is an accuracy impoverishment and, occasionally, the complete failure of the system. Many of such low-textured environments, however, contain planar elements that are rich in linear shapes, so it would be possible to extract line segments from them. We claim that these two types of features (keypoints and segments) complement each other and its combination leads to a more versatile, robust, and stable SLAM system. Furthermore, the resulting maps comprising both three-dimensional (3-D) points and line segments provide more structural information from the environment than point-only maps, as can be seen in the example shown in Fig. 1(d). Thus, applications that perform high-level tasks such as place recognition, semantic mapping, or task planning, among others, can significantly benefit from the richer information that can be inferred from them.

These benefits, though, come at the expense of a higher computational burden in both detecting and matching line segments in images [14], and also in dealing effectively with segment-specific problems such as partial occlusions, line disconnection, etc., which complicate feature tracking and matching as well as the residual computation for the map and pose optimization. Such hurdles are the reason why the number of solutions that have been proposed in the literature to SLAM or structure from motion (SfM) with line features (e.g., [15]–[19]) is so limited. Besides, the few solutions we have found only perform robustly in highly structured environments while showing unreliable results when applied to more realistic ones such as those recorded in the EuRoC or KITTI datasets. In this paper, we address the segment-specific tracking and matching issues by discarding outliers through the comparison of the length and the orientation of the line features, whereas for the residual computation, we represent segments in the map by their endpoints coordinates. Thus, the residuals between the observed segments and their corresponding lines in the map are computed by the distance between the projections of those endpoints on the image plane and the infinite lines associated to the observed ones. This way, we are able to build a consistent cost function that seamlessly encompasses both point and line features.

These two kinds of features are also employed to robustly detect loop closures during camera navigation, following a new bag-of-words (BoW) approach that combines the advantages of using each of them to perform place recognition. In summary, we propose a novel and versatile stereo visual SLAM system, coined PL-SLAM, which builds upon our previous visual odometry (VO) approach presented in [20], and combines both point and line segment features to perform real-time camera localization and mapping. The main contributions of this paper are as follows.

1) The first open-source stereo SLAM system that employs point and line segment features in real time, hence being capable of operating robustly in low-textured environments where traditional point-only approaches tend to fail. Because of the consideration of both kinds of features, our proposal also produces rich geometrical maps.
2) A new implementation of the bundle adjustment (BA) process that seamlessly accounts for both kinds of features while refining the poses of the keyframes (KFs).
3) An extension of the BoW approach presented in [21] that takes into account the description of both points and line segments to improve the loop-closure process.

A set of illustrative videos showing the performance of the proposed system and an open-source version of the developed C++ PL-SLAM library is publicly available at http://mapir.uma.es and https://github.com/rubengooj/pl-slam.

II. Related Work

Feature-based SLAM is traditionally addressed by tracking keypoints along successive frames and then minimizing some error function (typically based on reprojection errors) to simultaneously estimate the poses and the map [22]. Among the most successful proposals, we can highlight FastSLAM [23], PTAM [24] [25], SVO [26] [10], and, more recently, ORB-SLAM [13], which relies on a fast and continuous tracking of ORB features [27], and a local bundle adjustment (LBA) step with the continuous observations of the point features. All these approaches, though, tend to fail or reduce their accuracy in low-textured scenarios where the lack of repeatable and reliable features usually hinders the feature tracking process. In the following, we review the state of the art of visual SLAM systems based on alternative image features to keypoints: i.e., edgelets, lines, or line segments.

One of the remarkable approaches that employs *line* features is the one in [28], where Smith *et al.* proposed an algorithm to integrate them into a monocular extended Kalman filter SLAM system (EKF-SLAM). In the cited paper, the line detection relies on an hypothesize-and-test method that connects several nearby keypoints to achieve real-time performance. Other works employ *edge* landmarks as features in monocular SLAM, as the one reported in [29], which does not only include the information of the local planar patch as in the case of keypoints, but also considers local edge segments, hence introducing new valuable information as the orientation of the so-called *edgelets*. In that work, they derive suitable models for those kinds of features and use them within a particle-filter SLAM system, achieving nearly real-time performance. More recently, Forster *et al.* [10] also introduced edgelets in combination with intensity corners in order to improve robustness in environments with little or high-frequency texture.

A different approach, known as *model based*, incorporates prior information about the orientation of the landmarks derived from line segments. Particularly, the method in [30] presents a monocular two-dimensional (2-D) SLAM system that employs vertical and horizontal lines on the floor as features for both motion and map estimation. For that, they propose two different parameterizations for the vertical and the horizontal lines: vertical lines are represented as 2-D points on the floor plane (placed at the intersection point between the line and such plane), whereas horizontal lines are represented by their two endpoints placed on the floor. Finally, the proposed model is incorporated into an EKF-SLAM system. Another model-based approach is

reported in [31], where Zhou *et al.* introduce structural lines in an extension of a standard EKF-SLAM system. The dominant directions of the lines are estimated by computing their vanishing points under the assumption of a Manhattan world [32]. All these model-based approaches, though, are limited to very structured scenarios and/or planar motions, as they rely solely on line features.

The works in [16] and [33] address a generic approach that compares the impact of eight different landmark parameterization for monocular EKF-SLAM, including the use of point and line features. Nevertheless, such systems are only validated through analytic and statistical tools that assumed already known data association and that, unlike our proposal, do not implement a complete front-end that detect and track the line segments. Another technique for building a 3-D line-based SLAM system has been proposed in the recent work [34]. For that, the authors employ two different representations for the line segments: the Plücker line coordinates for the initialization and 3-D projections, and an orthonormal representation for the back-end optimization. Unfortunately, neither the source code is available nor the employed dataset contains any ground truth, therefore, it has not been possible to compare with our proposal.

Recently, line segment features have also been employed for monocular pose estimation in combination with points, due to the bad-conditioned nature of this problem. For that, Gomez-Ojeda *et al.* [35] extended the semidirect approach in [26] with line segments. Thanks to this pipeline, line segments can be propagated efficiently throughout the image sequence while refining the position of the endpoints under the assumptions of high frame rate and very narrow baseline.

Finally, by the time of the first submission of this paper, a work with the same name (PL-SLAM, [36]) was published extending the monocular algorithm ORB-SLAM to the case of including line segment features computed through the line segment detector (LSD) detector [37]. Apart from being a monocular system (unlike our stereo approach), their proposal deals with line tracking and matching in an essentially different way: they propagate the line segments by their endpoints and then perform descriptor-based tracking, which decreases the time performance of ORB-SLAM. Besides this computational drawback, when working with features detected with the LSD detector, the variance of the endpoints becomes quite pronounced, especially in challenging illumination conditions or very low-textured scenes, making more difficult wide-baseline tracking and matching between line features in nonconsecutive frames. Our PL-SLAM approach, in contrast, does not make any assumption regarding the position of the lines endpoints so that our tracking front-end allows us to handle partially occluded line segments, endpoints variance, etc., for both the stereo and frame-to-frame tracking, hence becoming a more robust approach to point-and-line SLAM.

III. PL-SLAM Overview

The general structure of the PL-SLAM system proposed here is depicted in Fig. 2, whereas its main modules will be presented in the following sections. This structure is strongly based on the scheme first proposed by ORB-SLAM [13] and also implements three different threads: *visual odometry*, *local mapping*, and *loop closure*. This efficient distribution allows for a continuous tracking of the VO module while the local mapping and the loop closure ones are processed in the background only when a new keyframe is inserted (other approaches that exploits parallel threads can be found elsewhere [8], [24]). As will be further described, our proposal also takes some of the ORB-SLAM ideas as basis for developing our point-and-line SLAM system.

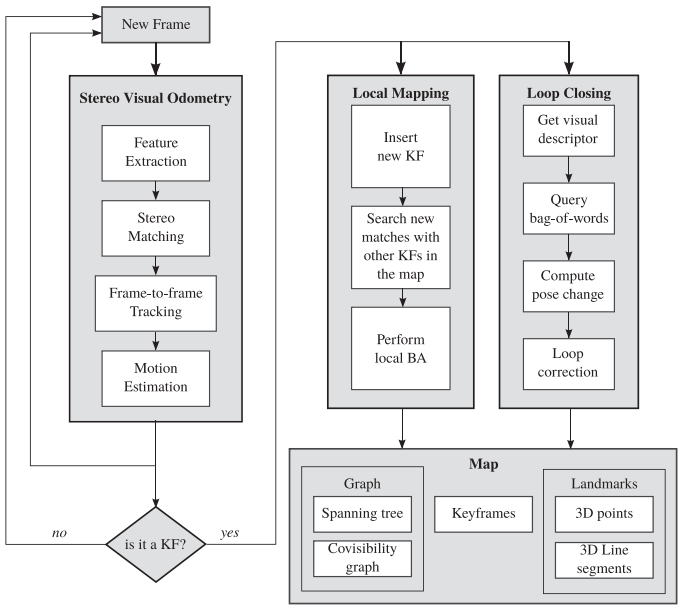


Fig. 2. Scheme of the stereo PL-SLAM system.

A. Map

The map consists of the following:

1) a set of KFs;
2) the detected 3-D landmarks (both keypoints and line segments);
3) a covisibility graph;
4) a spanning tree.

The KFs contain the observed stereo features and their descriptors, a visual descriptor of the corresponding left image computed through a visual vocabulary, as explained later in Section VI-A, and the information of the 3-D camera pose.

Regarding the landmarks, we store the list of observations and the most representative descriptor for each detected landmark. Besides, specifically for points, we also keep its estimated 3-D position, whereas for the line segments, we keep both their direction and the estimated 3-D coordinates of their endpoints.

Finally, the covisibility information is modeled by a graph (as in [38]), where each node represents a KF, and the edges between KFs are created only if they share a minimum number of landmarks (which in this paper is set to 20), allowing for real-time BA of the local map. Please refer to Fig. 3 for an example of a covisibility graph.

In order to perform a faster loop-closure optimization, we also form the so-called *essential graph*, which is less dense than

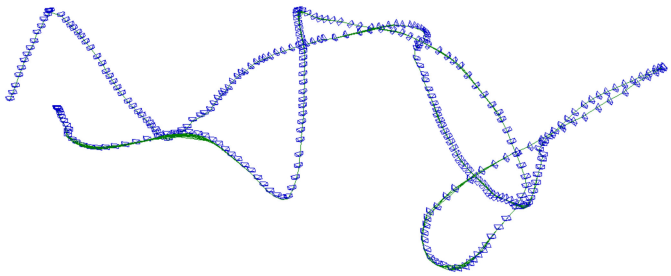


Fig. 3. Covisibility graph in the sequence *lt-first* for which we have represented the edges connecting the keyframes with green lines.

the covisibility graph as an edge between two KFs is created only when they share more than 100 landmark observations. Finally, the map also contains a *spanning tree*, which stands for the minimum connected representation of a graph that includes all the KFs. Both the essential graph and the use of a spanning tree for loop closure are ideas originally proposed in [13].

B. Feature Tracking

We perform feature tracking through the stereo visual odometry algorithm from our previous work [20]. In a nutshell, we track image features (points and segments) from a sequence of stereo frames and compute their 3-D position and their associated uncertainty represented by covariance matrices. The 3-D landmarks are then projected to the new camera pose, and the projection errors are minimized in order to obtain both the camera pose increment and the covariance associated to such estimation. This process is repeated every new frame, performing simply frame to frame VO, until a new KF is inserted into the map. Further discussion about the feature tracking procedure will be formally addressed in Section IV.

Once a KF is inserted into the map, two procedures are run in parallel: local mapping and loop-closure detection.

C. Local Mapping

The local mapping procedure looks for new feature correspondences between the new KF, the last one, and those connected to the last one in the covisibility graph. This way, we build the so-called *local map* of the current KF, which includes all the KFs that share at least 20 landmark observations with the current one as well as all the landmarks observed by them. Finally, an optimization of all the elements within the local map (i.e., KF poses and landmarks positions) is performed. A detailed description of this procedure will be presented in Section V.

D. Loop Closure

In parallel to local mapping, loop-closure detection is carried out by extracting a visual descriptor for each image, based on a BoW approach, as will be described in Section VI. All the visual descriptors of the captured KFs during camera motion are stored in a database, which is later employed to find similar frames to the current one. The best match will be considered a loop-closure candidate only if the local sequence surrounding this KF is also similar. Finally, the relative $SE(3)$ transformation between the current KF and the loop-closure candidate is estimated so that if a proper estimation is found, all the KFs poses involved in the loop are corrected through a pose-graph optimization (PGO) process.

It is important to remark that the stereo visual odometry system runs continuously at every frame while both the local mapping and loop-closure detection procedures are launched in the background (in separated threads) only when a new KF is inserted, thus allowing our system to reach real-time performance. In the event of a new keyframe being inserted in the system while the local mapping thread is still being processed, the keyframe is temporary stored until the map is updated and then a new local mapping process is launched.

It is worth mentioning that as declared before, these mapping and loop-closure pipelines are identical to the ones presented in ORB-SLAM, being aimed to reduce (along with the incorporation of recent sparse algebra techniques) the high computational burden that general BA involves. Within the BA framework, our proposal belongs to the so-called *relative* techniques (e.g., [39]–[41]), which have gained great popularity in the last years as an alternative to the more costly *global* approaches (e.g., [24] and [42]).

IV. Feature Tracking

This section reviews the most important aspects of our previous work [20], which deals with the visual odometry estimation between consecutive frames, and also with the KF decision policy. Briefly, both points and line segments are tracked along a sequence of stereo frames (see Fig. 1), and, then, both the 3-D motion of the camera and its uncertainty are computed through the minimization of the projection errors. Note that this process only performs the optimization of the camera poses and not the 3-D position of the tracked features, whose coordinates in the map are refined during the LBA procedure explained in the next section.

Since the stereo cameras employed in the experiments are precalibrated, the initialization of these features is performed in a single stereo shot by straightforwardly employing their extrinsic parameters to determine the 3-D position of both the observed keypoints and lines.

A. Point Features

In this paper, we use the well-known ORB method [27] due to its great performance for keypoint detection, and the binary nature of the descriptor it provides, which allows for a fast, efficient keypoint matching. In order to reduce the number of outliers, we only consider measurements that fulfill that the best match in the left image corresponds to the best match in the right one, i.e., they are mutual best matches. Finally, we also filter out those matches whose distance in the descriptor space with the second best match is less than twice the distance with the best match, to ensure that the correspondences are meaningful enough.

B. Line Segment Features

The LSD method [37] has been employed to extract line segments, providing high precision and repeatability. For stereo matching and frame-to-frame tracking, we augment line segments with a binary descriptor provided by the line band descriptor (LBD) method [43], which allows us to find correspondences between lines based on their local appearance. Similarly to the case of points, we check that both candidate features are mutual best matches, and also that the feature is meaningful enough. Finally, we take advantage of the useful geometrical information that line segments provide in order to filter out those line matches with different orientations and lengths, and those with a high difference on the disparities of the endpoints. Notice that this filter helps the system to retain a larger amount of structural lines, which allows the formation of more consistent maps based on points and lines [see Fig. 1(d)].

C. Motion Estimation

Once we have established the correspondences between two stereo frames, we back-project both the keypoints and the line segments from the first frame to the next one. Then, we iteratively estimate the camera ego-motion through a robust Gauss–Newton minimization of the line and keypoint projection errors. In order to deal with outliers, we employ a Pseudo-Huber loss function and perform a two-step minimization, as proposed in [44]. Finally, we obtain the incremental motion estimation between the two consecutive frames, which can be modeled by the following normal distribution:

$$\boldsymbol{\xi}_{t,t+1} \sim \mathcal{N}(\boldsymbol{\xi}^*_{t,t+1}, \boldsymbol{\Sigma}_{\boldsymbol{\xi}^*_{t,t+1}}) \tag{1}$$

where $\boldsymbol{\xi}^*_{t,t+1} \in \mathfrak{se}(3)$ is the 6-D vector of the camera motion between the frames t and $t+1$, and $\boldsymbol{\Sigma}_{\boldsymbol{\xi}^*_{t,t+1}}$ stands for the covariance of the estimated motion, approximated by the inverse of the Hessian of the cost function in the last iteration.

D. Keyframe Selection

For deciding when a new KF is inserted in the map, we have followed the approach in [45], which employs the uncertainty of the relative motion estimation. Thus, following (1), we transform the uncertainty from the covariance matrix into a scalar, named *entropy*, through the following expression:

$$h(\boldsymbol{\xi}) = 3(1 + \log(2\pi)) + 0.5 \log(|\boldsymbol{\Sigma}_{\boldsymbol{\xi}}|). \tag{2}$$

Then, for a given KF, i we check the ratio between the entropy from the motion estimation between the previous KF i and the current one $i+u$ and that between the previous KF i and its first consecutive frame $i+1$, i.e.,

$$\alpha = \frac{h(\boldsymbol{\xi}_{i,i+u})}{h(\boldsymbol{\xi}_{i,i+1})}. \tag{3}$$

If the value of α lies below some pre-established threshold, which in our experiments has been set to 0.9, then the frame $i+u$ is inserted to the system as a new KF. Notice that to compute the expression in (2), we need the uncertainty of the pose increment between nonconsecutive frames. Since (1) only estimates the incremental motion between consecutive frames, a series of such estimations are composed through first-order Gaussian propagation techniques to obtain the covariance between two nonconsecutive KFs.

V. Local Mapping

This section describes the behavior of the system when a new KF is inserted, which essentially consists in performing the BA of the so-called *local map*, i.e., those KFs connected with the current one by the covisibility graph and the landmarks observed by those local KFs.

A. Keyframe Insertion

Every time the visual odometry thread selects a KF, we insert it into the SLAM system and optimize the local map. First, we refine the estimation of the relative pose change between the current and the previous KFs, since the one provided by the VO is estimated by composing the relative motions between the intermediate frames. For that, we perform data association between the KFs, taking into account the geometrical restrictions described in Section IV and obtaining a consistent set of common features observed in them. Then, we perform a similar optimization than the one presented in Section IV-C, for which we employ the pose provided by the VO thread as the initial estimation for a Gauss–Newton minimization. Once we have computed the relative pose change between the KFs, we insert the current one into the system, including the following.

1) An index for the keyframe.
2) The information of its 3-D pose, which comprises an absolute pose and the relative pose from the previous KF, along with their associated uncertainties.
3) The new 3-D landmarks, which are initialized by storing both their 2-D image coordinates and their descriptors. The new observations of the already existing landmarks are also added to the map.

Finally, we also look for new correspondences between the unmatched feature observations from the current frame, and the landmarks in the local map.

B. Local Bundle Adjustment

After inserting the KF, the next step is to perform a BA of the local map. As stated before, this map is formed by all the KFs connected with the current one in the covisibility graph (i.e., those that share at least 20 landmarks) and also all the landmarks observed by the local KFs. For that, let us define the vector $\boldsymbol{\psi}$ that contains the variables to be optimized, which are the $\mathfrak{se}(3)$ pose of each KF $\boldsymbol{\xi}_{iw}$, the 3-D position of each point $\mathbf{X}_{wj}$, and also the 3-D positions of the endpoints for each line segment: $\{\mathbf{P}_{wk}, \mathbf{Q}_{wk}\}$. Then, we minimize the projection errors between the observations and the landmarks projected to the frames where they were observed

$$\boldsymbol{\psi}^* = \underset{\boldsymbol{\psi}}{\operatorname{argmin}} \sum_{i \in \mathcal{K}_l} \left[\sum_{j \in \mathcal{P}_l} \mathbf{e}_{ij}^\top \boldsymbol{\Sigma}_{\mathbf{e}_{ij}}^{-1} \mathbf{e}_{ij} + \sum_{k \in \mathcal{L}_l} \mathbf{e}_{ik}^\top \boldsymbol{\Sigma}_{\mathbf{e}_{ik}}^{-1} \mathbf{e}_{ik} \right] \tag{4}$$

where $\mathcal{K}_l$, $\mathcal{P}_l$, and $\mathcal{L}_l$ refer to the groups of local KFs, points, and line segments, respectively.

In this expression, the projection error $\mathbf{e}_{ij}$ stands for the 2-D distance between the observation of the jth map point in the ith KF, and can be expressed as

$$\mathbf{e}_{ij} = \mathbf{x}_{ij} - \boldsymbol{\pi}(\boldsymbol{\xi}_{iw}, \mathbf{X}_{wj}) \tag{5}$$

where the function $\boldsymbol{\pi} : \mathfrak{se}(3) \times \mathbb{R}^3 \mapsto \mathbb{R}^2$ first places the jth 3-D point $\mathbf{X}_{wj}$ (in world coordinates) into the local reference system of the ith KF, i.e., $\mathbf{X}_{ij}$, and then projects this point to the image. The use of line segments is slightly different, since we cannot simply compare the position of the endpoints as they might be displaced along the line or occluded from one frame to the next one. For that, we take as error function the distances between the projected endpoints of the 3-D line segment and its corresponding infinite line in the image plane. In this case, the error $\mathbf{e}_{ik}$ between the kth line observed in the ith frame, is given by

$$\mathbf{e}_{ik} = \begin{bmatrix} \mathbf{l}_{ik} \cdot \boldsymbol{\pi}(\boldsymbol{\xi}_{iw}, \mathbf{P}_{wk}) \\ \mathbf{l}_{ik} \cdot \boldsymbol{\pi}(\boldsymbol{\xi}_{iw}, \mathbf{Q}_{wk}) \end{bmatrix} \tag{6}$$

where $\mathbf{P}_{wk}$ and $\mathbf{Q}_{wk}$ refer to the 3-D endpoints of the line segments in the world coordinate system and $\mathbf{l}_{ik}$ is the equation of the infinite line that corresponds to the kth line segment in the ith KF, which can be obtained with the cross product between the 2-D endpoints of the line segments in homogeneous coordinates, i.e.: $\mathbf{l}_{ik} = \mathbf{p}_{ik} \times \mathbf{q}_{ik}$.

The problem in (4) can be iteratively solved by following the Levenberg–Marquardt optimization approach, for which we need to estimate both the Jacobian and the Hessian matrices

$$\Delta\boldsymbol{\psi} = \left[\mathbf{H} + \lambda \, \mathrm{diag}(\mathbf{H})\right]^{-1} \mathbf{J}^\top \mathbf{W}\mathbf{e} \tag{7}$$

where the error vector $\mathbf{e}$ contains all the projection errors $\mathbf{e}_{ij}$ and $\mathbf{e}_{ik}$. This equation, along with the following update step:

$$\boldsymbol{\psi}' = \boldsymbol{\psi} \boxplus \Delta\boldsymbol{\psi} \tag{8}$$

can be applied recursively until convergence, resulting in the optimal $\boldsymbol{\psi}$, from which we can update the position of the local KFs and landmarks. Notice that the update equation cannot be directly applied to the whole vector, given the different nature of the variables in $\boldsymbol{\psi}$.

It is important to remark that each observation error $\mathbf{e}_{ij}$ or $\mathbf{e}_{ik}$, only depends on a single KF $\boldsymbol{\xi}_{iw}$, and a single landmark $\mathbf{X}_{wj}$ or $\{\mathbf{P}_{wk}, \mathbf{Q}_{wk}\}$. Hence, the Hessian matrix can be formed by adding the influence of each observation to its corresponding block, as shown in (9) shown at the bottom of this page, where the contribution of two single features to the Hessian is presented (the full matrix is formed by $\mathbf{H} = \sum_{i \in \mathcal{K}_l} \sum_{j \in \mathcal{P}_l} \sum_{k \in \mathcal{L}_l} \mathbf{H}_{i,jk}$).

Notice that for the rest of observations that belong to the KFs that are not part of the local map, their Jacobian matrices $\frac{\partial \mathbf{e}_{ij}}{\partial \boldsymbol{\xi}_{iw}}$ and $\frac{\partial \mathbf{e}_{ik}}{\partial \boldsymbol{\xi}_{iw}}$ are equal to zero, since here we only optimize the local map, whereas the rest of the KFs remain fixed.

It should also be underlined that in (4) the influence of the errors in both points and lines is weighted with $\boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ij}}$ and $\boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ik}}$, respectively, which stand for the inverses of the covariance matrixes associated to the uncertainty of each projection error. In practice, though, it is more effective to set such covariances to the identity matrix and follow a similar approach to the one described in Section IV-C as it introduces robust weights and also deals with the presence of outlier observations.

Finally, we remove from the map those landmarks with less than three observations, as they are less meaningful.

VI. Loop Closure

In this paper, we adopt a BoW approach based on the binary descriptors extracted for both the keypoints and the line segments in order to robustly cope with data association and loop-closure detection.

In short, the BoW technique consists in summarizing all the information extracted from an image (in our proposal, the descriptors of keypoints and line segments) into a *word* vector, employing for that a vocabulary that has been built offline from different image datasets. Then, as the camera moves, the words computed from the grabbed images are stored in a database that is later employed to search for the most similar image to the current keyframe.

In the following, we first address the process of detecting loop closures from the created BoWs, and then describe the correction of the pose estimations of the KFs involved in the loop.

A. Loop-Closure Detection

The detection of loop closures involves both to find an image similar to the one being currently processed and to estimate the relative pose change between them, as described next.

$$\mathbf{H}_{i,jk} = \left[\begin{array}{c|c|c} \vdots \; \frac{\partial \mathbf{e}_{ij}}{\partial \boldsymbol{\xi}_{iw}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ij}} \frac{\partial \mathbf{e}_{ij}}{\partial \boldsymbol{\xi}_{iw}} + \frac{\partial \mathbf{e}_{ik}}{\partial \boldsymbol{\xi}_{iw}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ik}} \frac{\partial \mathbf{e}_{ik}}{\partial \boldsymbol{\xi}_{iw}} \; \vdots & \vdots \; \frac{\partial \mathbf{e}_{ij}}{\partial \boldsymbol{\xi}_{iw}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ij}} \frac{\partial \mathbf{e}_{ij}}{\partial \mathbf{X}_{wj}} \; \vdots & \vdots \; \frac{\partial \mathbf{e}_{ik}}{\partial \boldsymbol{\xi}_{iw}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ik}} \frac{\partial \mathbf{e}_{ik}}{\partial \{\mathbf{P}_{wk}, \mathbf{Q}_{wk}\}} \; \vdots \\ \hline \vdots \; \frac{\partial \mathbf{e}_{ij}}{\partial \mathbf{X}_{wj}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ij}} \frac{\partial \mathbf{e}_{ij}}{\partial \boldsymbol{\xi}_{iw}} \; \vdots & \vdots \; \frac{\partial \mathbf{e}_{ij}}{\partial \mathbf{X}_{wj}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ij}} \frac{\partial \mathbf{e}_{ij}}{\partial \mathbf{X}_{wj}} \; \vdots & \vdots \; \mathbf{0} \; \vdots \\ \hline \vdots \; \frac{\partial \mathbf{e}_{ik}}{\partial \{\mathbf{P}_{wk}, \mathbf{Q}_{wk}\}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ik}} \frac{\partial \mathbf{e}_{ik}}{\partial \boldsymbol{\xi}_{iw}} \; \vdots & \vdots \; \mathbf{0} \; \vdots & \vdots \; \frac{\partial \mathbf{e}_{ik}}{\partial \{\mathbf{P}_{wk}, \mathbf{Q}_{wk}\}}^\top \boldsymbol{\Sigma}^{-1}_{\mathbf{e}_{ik}} \frac{\partial \mathbf{e}_{ik}}{\partial \{\mathbf{P}_{wk}, \mathbf{Q}_{wk}\}} \; \vdots \end{array}\right] \tag{9}$$

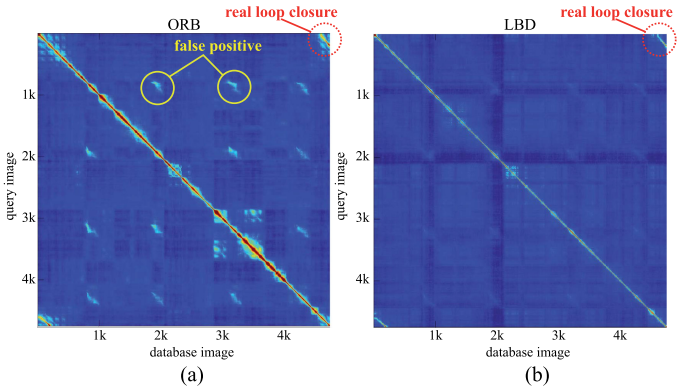


Fig. 4. Similarity matrices for a certain dataset where the (a) ORB keypoint-only bag-of-words approach yields false positives that are not present in the (b) LBD line-only approach.

1) Visual Place Recognition: Specifically, we have employed the method presented in [21], which was initially developed for BRIEF binary descriptors, and subsequently adapted to ORB keypoints. Since, in this paper, segments are also augmented with binary descriptors, we propose to build two specific visual vocabularies and databases for them. This way, at each time step, the most similar images in the databases of keypoints and line segments are retrieved in parallel in order to look for loop closures. This dual search is motivated by the fact that some scenes may be described more distinctively by lines than by keypoints or vice versa. Thus, employing both methods and merging their results allow us to refine the output of database queries, incurring in a small computational footprint.

To illustrate this, we first define a *similarity matrix* as the matrix that contains in each row the similarity values, in the range [0, 1] of a certain image with all the images stored in the database. Then, we compute such matrices from a sequence recorded in a corridor that goes around a square area.

Concretely, the matrix in Fig. 4(a) has been computed employing only ORB keypoints to build both the vocabulary and the database, whereas the one in Fig. 4(b) relies only on lines. The color palette goes from blue (score $= 0$) to red (score $= 1$). As can be noted, some yellowish areas appear in the first matrix in places where the images look similar according to the keypoints (specifically, after turning at the corners of the corridor). This indicates potential loop closures although, in fact, they are just false positives. The second (line-only) matrix, though, does not present this behavior so that it may be employed to discard them. On the other hand, the first matrix presents more distinctiveness, since the difference in score is generally larger for nonsimilar images than in the line-only matrix. Therefore, the image similarities yielded by querying both feature databases may be combined to improve robustness when detecting potential loop closures.

In this paper, we propose to weight the results from both features (s_k for keypoints and s_l for lines) according to two criteria, namely *strength* and *dispersion*. The former weights the similarity score proportionally to the number of features of a certain type (keypoint or line) in the set of features detected in the image, whereas the latter takes into account the dispersion of the features in the image (the more disperse the higher the weight will be). This yields a more robust total similarity score for the image (s_t)

$$s_t = 0.5\left(n_k/\left(n_k+n_l\right)+d_k/\left(d_k+d_l\right)\right)s_k + 0.5\left(n_l/\left(n_k+n_l\right)+d_l/\left(d_k+d_l\right)\right)s_l. \quad (10)$$

In this equation, n_k and n_l are the number of keypoints and lines extracted in the image, respectively, whereas the dispersion values for the keypoints and the lines (d_k and d_l, respectively) are computed as the square root of the sum of the variances in the x and y coordinates of the found features. For the case of the lines, such x and y coordinates are taken from their midpoint.

Note that this formulation gives the same importance to both kinds of features (hence the 0.5 factor), although this could be tuned according to the environment (e.g., if the images are expected to be low textured, it might be more convenient to down-weight the keypoint result with respect to the lines one). Nevertheless, the results will not change significantly with the weighting factor, and a finer optimization would not be worthy as long as both kinds of features have influence in the total score.

We have empirically evaluated this strategy in comparison to four other alternatives following the classification framework employed in [46] for four different datasets: Oxford dataset [47], sequence 4 in Malaga dataset [48], sequence 7 in KITTI dataset [49], and i3tf dataset [34]. Concretely, the compared alternatives consisted in taking into account the following:

1) only the score yielded from querying the *keypoint* BoW (s_k);
2) only the score yielded from querying the *line* BoW (s_l);
3) both s_k and s_l but following only the *strength* criterion (strategy #1);
4) both s_k and s_l but following only the *dispersion* criterion (strategy #2).

We have tested them on synchronized sequences without any loop closures, so that the elements in the diagonal matrix are 1, as they correspond to the same image in both the database and the query. Subsequently, we have selected, for each query image, the k most similar images from the database (i.e., those with largest total score), and have considered a match as an inlier (true positive match) if it was close enough to the diagonal with a tolerance of d frames. Finally, we have varied the tolerance and measured the ratio of inliers for all the strategies to generate the precision-recall curves in the abovementioned datasets.

As shown in Fig. 5, all three combined strategies outperform the point and line-only approaches, whereas strategy #3 [that corresponding to (10)] performs slightly better than the other two in all the evaluated datasets.

2) Estimating the Relative Motion: Once we have a loop-closure candidate, we still need to discard false positives that could have not been detected with the abovementioned approach. This is achieved by recovering the relative pose between the two KFs involved in the loop closure (namely *current* and *old* KFs from now on). For that, we first look for matches between the features from both KFs in a similar way to the one described in Section III while also searching for new correspondences between the current KF and the local map associated to the

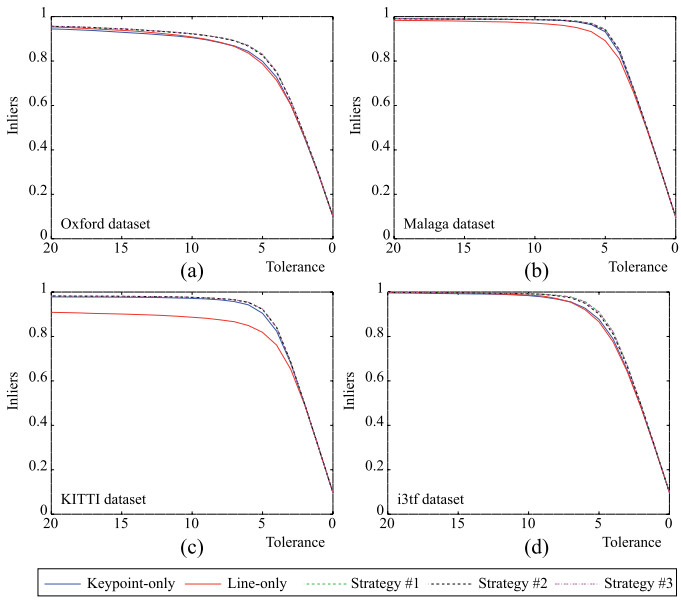


Fig. 5. Precision-recall curves for four different datasets. (a) Oxford dataset. (b) Sequence 4 in Malaga dataset. (c) Sequence 7 in KITTI dataset. (d) i3tf dataset, for the 10 most similar images in the dataset.

old one. Then, we estimate a valid transformation $\hat{\boldsymbol{\xi}}_{ij} \in \mathfrak{se}(3)$ that relates both KFs following the approach described in Section IV-C. Finally, since an erroneous detection of a loop closure (false positive) would produce a very negative impact on the SLAM system, we check the consistency of the loop-closure candidate with the following tests.

1) The maximum eigenvalue of the covariance matrix $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\xi}}_{ij}}$ is inferior to 0.01.
2) The obtained translation and rotation cannot rise over 0.50 m and 3.00°, respectively.
3) The inliers ratio in the estimation is higher than 50%.

Regarding the first criterion, a large value of the eigenvalues of the uncertainty matrix [see (1)] is often an indicator of an ill-conditioned Hessian matrix, most probably due to the presence of a large number of outliers in the feature matching set. Ensuring that the maximum eigenvalue of the covariance matrix is below a certain threshold allows us to detect potentially incorrect loop closures candidates and discard them.

In the case of the second criterion, we also set a maximum translation and rotation limit for the estimated pose, as BoW-based approaches typically provide positive matches that are very similar in appearance and pose, so that a large change in pose between the involved frames usually indicates a wrong loop-closure detection. Finally, the third criterion sets a minimum ratio of detected inliers after the optimization process, since motion estimation is strongly affected by the presence of outliers and incorrect associations from visual place recognition.

B. Loop Correction

After estimating all consecutive loop closures in our trajectory, we then fuse both sides of the loop closure correcting the error distributed along the loop. This is typically solved by formulating the problem as a PGO, where the nodes are the KFs inside the loop, and the edges are given by both the essential graph and the spanning tree. For that, let us define the following error function as the $\mathfrak{se}(3)$ difference between the transformation that relates the KFs $\hat{\boldsymbol{\xi}}_{ij}$ to the current observation of the same transformation:

$$\mathbf{r}_{ij}(\boldsymbol{\xi}_{iw}, \boldsymbol{\xi}_{jw}) = \log\left(exp(\hat{\boldsymbol{\xi}}_{ij}) \cdot exp(\boldsymbol{\xi}_{jw}) \cdot exp(\boldsymbol{\xi}_{iw})^{-1}\right) \tag{11}$$

where the operators $\log : SE(3) \mapsto \mathfrak{se}(3)$ and $\exp : \mathfrak{se}(3) \mapsto SE(3)$ refer to the well-known logarithm and exponential maps. Notice that in the case of a regular edge, the value of $\hat{\boldsymbol{\xi}}_{ij}$ coincides with the estimation of $\boldsymbol{\xi}_{ij}$ in the first step of the optimization, and hence the error in these edges is initially zero.

This PGO problem is solved using the `g2o` library [50] yielding the optimal pose of the KFs included in the optimization, i.e., the essential graph and the spanning tree, when considering the loop-closure edges. Finally, we update the pose of the KFs along with the pose of the landmarks observed by them, and we also merge the local maps of both sides of the loop by first fusing the landmarks matched while estimating their relative motion (see Section VI-A), and then looking for new correspondences between the unmatched landmarks.

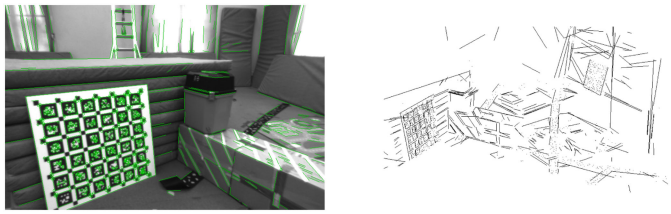
VII. Experimental Validation

In this section, we evaluate the performance of PL-SLAM in several video sequences from different datasets, in order to demonstrate the robustness of our proposal, which does not fail in any of the considered sequences, even in the low-textured ones. Besides, we also provide an estimation of both the camera trajectory and the map while computing a measurement of the committed error with respect to the ground truth. Concretely, we have tested our proposal against the EuRoC MAV dataset [51], a low-textured dataset specifically recorded for assessing the effect of adding lines to the visual SLAM system, and the well-known KITTI video sequences [49].

It is important to remark that the error metric employed in this paper for the EuRoC MAV and the low-textured datasets is the one proposed in [52] as relative pose error (RPE), which computes the relative error in translation between the estimated camera poses and the ground truth. Using RPE as a metric allows us to obtain comparable error values for all the experiments, regardless the presence and number of loop closures in the camera trajectory, as relative measurements are not significantly affected by them, unlike absolute pose error measurements. For the KITTI dataset, though, we employ the standardized metrics provided by the KITTI benchmark to obtain the error measurements presented in Table III.

In the following, we present examples of the trajectories and maps estimated by PL-SLAM, together with the average errors committed by our proposal, a *point-only* version of our system (P-SLAM), a *line-only* version of our system (L-SLAM), and ORB-SLAM2, which is considered one of the state-of-the-art methods for stereo visual SLAM. For the latter, we have employed the open-source implementation of its last version [53].

We have also tried to compare our method against the one proposed in [34], but unfortunately, as their approach to perform line segment tracking is based on an optical flow algorithm, their proposal fails when applied to datasets with large motions



(a) (b)

Fig. 6. Mapping results in the *V1-01-easy* sequence from the EuRoC MAV dataset. (a) Features tracked between two consecutive keyframes. (b) Resulting 3-D map for the sequence. The checkerboard and the boxes in the scene are clearly reflected in the left part of the map, whereas more noisy features can be found in the rest, as a consequence of factors such as nontextured surfaces, high illumination, etc.

TABLE I
RELATIVE TRANSLATIONAL RMSE ERRORS IN THE EUROC MAV DATASET [51]

Sequence	P-SLAM	L-SLAM	PL-SLAM	ORB-SLAM2
MH-01-easy	0.0811	0.0588	0.0416	**0.0251**
MH-02-easy	0.1041	0.0566	**0.0522**	0.0638
MH-03-med	0.0588	**0.0371**	0.0399	0.0712
MH-04-dif	-	0.1090	0.0641	**0.0533**
MH-05-dif	0.1208	0.0811	0.0697	**0.0414**
V1-01-easy	0.0583	0.0464	0.0423	**0.0405**
V1-02-med	0.0608	-	**0.0459**	0.0617
V1-03-dif	0.1008	-	**0.0689**	-
V2-01-easy	0.0784	0.0974	**0.0609**	-
V2-02-med	0.0767	-	**0.0565**	0.0666
V2-03-dif	0.1511	-	**0.1261**	-

A dash indicates that the experiment failed.
The boldface type are to highlight the better results.

between frames. Therefore, we could not include their results in this paper.

All the experiments have been run on an Intel Core i5-6600 CPU @ 3.30GHz and 16-GB RAM without GPU parallelization.

A. EuRoC MAV dataset

The EuRoC MAV dataset [51] consists of 11 stereo sequences recorded with an MAV flying across three different environments: two indoor rooms and one industrial scenario, containing sequences that present different challenges depending on the speed of the drone, illumination, texture, etc.

As an example, we show the central part of the map built from the *V1-02-easy* sequence in Fig. 6(b) where two different parts are clearly visible. The first one shows the features extracted from the nonstructured part of the environment (refer to the right side of the map), presenting a relatively large amount of small and noisy line segments, which make difficult the interpretation of that part of the scene. In contrast, at the bottom left part of the figure, we can observe the structured part of the environment, which is clearly represented in the map through a set of line segments that depicts a checkerboard and a bunch of boxes. This example reflects that the maps built from line segments are geometrically richer than those created from only points, so that they can be employed to extract high-level meaningful information from them.

Finally, Table I shows the mean relative translational RMSE of the motion estimation in the different sequences included in the dataset. It can be observed that for indoor and structured scenarios, the inclusion of line segment features in the system is very beneficial to improve the robustness of the system and the estimation of the camera trajectory. In this case, both the *point-only* and the *line-only* approaches yield worse results than PL-SLAM, whereas ORB-SLAM2 fails in some sequences, as keypoint tracking is prone to be lost. PL-SLAM, on the contrary, successfully estimates the camera trajectory in all the sequences.

TABLE II
RELATIVE TRANSLATIONAL RMSE ERRORS IN LOW-TEXTURED SEQUENCES RECORDED WITH GT DATA FROM AN OPTITRACK SYSTEM

Sequence	P-SLAM	L-SLAM	PL-SLAM	ORB-SLAM2
lt-easy	-	0.1412	**0.1243**	0.1391
lt-medium	-	0.1998	**0.1641**	-
lt-difficult	-	0.1801	**0.1798**	-
lt-rot-difficult	0.2411	0.2247	**0.2034**	0.2910

A dash indicates that the experiment failed.
The boldface type are to highlight the better results.

TABLE III
MEAN RELATIVE RMSE FOR THE KITTI DATASET [49]

	P-SLAM		L-SLAM		PL-SLAM		ORB-SLAM2	
Seq.	t_{rel}	R_{rel}	t_{rel}	R_{rel}	t_{rel}	R_{rel}	t_{rel}	R_{rel}
00	2.47	0.95	3.32	6.67	2.36	0.89	1.82	0.58
01	8.25	5.85	-	-	5.80	2.32	1.24	0.32
02	2.54	1.04	6.62	9.42	2.35	0.91	1.81	0.46
03	3.88	1.71	6.83	7.14	3.74	1.54	2.79	0.44
04	2.14	0.42	-	-	2.21	0.30	1.07	0.18
05	2.03	0.96	3.06	2.25	1.74	0.88	1.03	0.37
06	4.37	3.01	5.99	8.29	3.51	2.72	1.25	0.56
07	2.65	1.35	3.58	6.52	1.83	1.03	0.94	0.46
08	2.73	1.34	6.15	9.23	2.18	1.15	1.78	0.57
09	1.90	1.02	6.92	4.66	1.68	0.92	1.11	0.42
10	1.92	1.32	6.86	5.55	1.21	0.99	1.09	0.46

The translation errors are expressed in %, whereas the rotation errors are also expressed relatively to the translation in deg/100 m. A dash indicates that the experiment failed.

B. Low-Textured Scenarios

We have also assessed the performance of the compared methods in challenging low-textured scenarios. For that, we have recorded a set of four stereo sequences (namely *lt-easy*, *lt-medium*, *lt-difficult*, and *lt-rot-difficult*) in a room equipped with an OptiTrack system,[1] which provides the ground truth of the camera trajectory. The resulting covisibility graph yielded by our PL-SLAM system for the sequence *lt-medium* is shown in Fig. 3 where a loop closure between the initial and final parts of the trajectory can be observed. The experiments in these sequences (refer to Table II) reveal that while point-based approaches either fail to recover the trajectory or yield worse results than in previous scenarios, the two methods based on line segments are capable of robustly estimating the camera path in all sequences, even achieving a good performance in terms of accuracy.

C. KITTI Dataset

Finally, we have tested PL-SLAM on the well-known KITTI dataset [49], using the 11 sequences that provide ground truth and yielding the results presented in Table III. Note that this is an urban dataset with highly textured image sequences and,

[1][Online]. Available: http://optitrack.com/

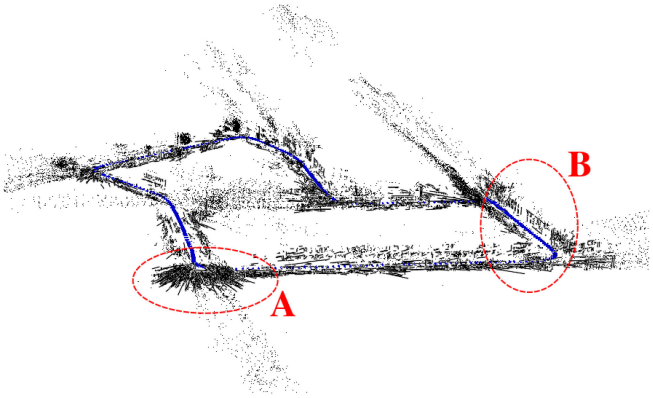


Fig. 7. Map (in black) comprising points and line segments, and the trajectory (in blue) obtained with PL-SLAM from an outdoor environment in the sequence *KITTI-07*. The map presents noisy measurements in some parts (e.g., zone A), and structural lines from the environment, such as parts of the buildings (e.g., zone B).

as expected, the exploitation of line segments barely increases the accuracy, since point features are more than sufficient for a proper operation of a keypoint-based SLAM system. Still, the reasons why we have tested our proposal against the KITTI sequences are twofold: first, the lack of publicly accessible datasets containing both low-textured scenes and ground truth, and second, the KITTI dataset has become a standard when assessing visual SLAM.

For this dataset, we have employed the standard error measurements proposed by the KITTI benchmark site where the translational errors are expressed in % of the trajectory length while the rotational part is expressed in deg/100 m of the trajectory. It is important to highlight that these results are obtained from applying the KITTI benchmarking scripts to the trajectories estimated by the evaluated methods. This also includes ORB-SLAM2, for which we have generated the estimated trajectory for each sequence with both the code and the parameters they publicly provide in their repository. A different (and probably more tuned) set of parameters might cause the difference between their results obtained in this evaluation and those presented in the ORB-SLAM2 original paper, which could not be reproduced.

In any case, and regarding the robustness of the system, both PL-SLAM and ORB-SLAM2 successfully complete the trajectory estimation for all the sequences, as expected for such a textured dataset. Unsurprisingly, the results confirm worse performance of the *line-only* approach in these outdoor scenarios, even failing at properly estimating the trajectory of the stereo camera in some of the sequences (those recorded in rural environments, which lacks structural lines).

As an illustrative example, Fig. 7 depicts the trajectory and the map estimated by PL-SLAM in the sequence *KITTI-07*. As can be seen in the zone marked as A in the figure, the presence of line segments can introduce some "noise" in the maps, as not all the detected lines have a physical meaning, i.e., some lines do not belong to structural parts of the environment. Nevertheless, in other parts of the sequence, relevant information of the scene structure has been correctly captured in the map. This can be observed in the zone marked as B in the figure, where the buildings can be clearly observed, leading to a descriptive representation of the scene. On the contrary, the presence of noisy points in the map is less noticeable to the human eye, as they do not provide as much spatial information as line segments.

TABLE IV
AVERAGE RUNTIME OF EACH PART OF THE ALGORITHM

Visual Odometry	KITTI 1241 × 376 10 fps	EuRoC MAV 752 × 480 20 fps	Low-Textured 752 × 480 20 fps
P-SLAM	12.2 ms	8.7 ms	8.1 ms
L-SLAM	54.6 ms	47.6 ms	46.1 ms
PL-SLAM	66.0 ms	49.7 ms	40.0 ms
ORB-SLAM2	98.1 ms	69.0 ms	61.4 ms
Local Mapping			
P-SLAM	38.9 ms	37.3 ms	35.8 ms
L-SLAM	37.4 ms	36.0 ms	34.5 ms
PL-SLAM	43.8 ms	40.6 ms	42.1 ms
ORB-SLAM2	230.0 ms	162.0 ms	102.0 ms
Loop Closing			
P-SLAM	11.3 ms	3.5 ms	3.7 ms
L-SLAM	9.5 ms	3.9 ms	3.4 ms
PL-SLAM	28.0 ms	4.7 ms	4.5 ms
ORB-SLAM2	9.1 ms	3.6 ms	4.4 ms
Keyframe Rate			
P-SLAM	4.78 f/kf	6.36 f/kf	3.28 f/kf
L-SLAM	3.68 f/kf	4.45 f/kf	3.25 f/kf
PL-SLAM	5.06 f/kf	5.36 f/kf	4.78 f/kf
ORB-SLAM2	2.77 f/kf	5.26 f/kf	3.10 f/kf

Finally, Fig. 8 depicts the estimated trajectory obtained with PL-SLAM in three sequences from the KITTI dataset that presents different number of loop closures. It can be noted that the importance of correcting the drift in long sequences to obtain accurate absolute solutions [see Fig. 8(a) and (c)], in contrast to the results obtained in sequences without loop closures, as the one presented in Fig. 8(b).

D. Performance

Regarding the time performance, we present Table IV that shows the average processing time of each part of the PL-SLAM algorithm, for each of the tested datasets. Thanks to the efficient implementation of [20], our VO thread achieves real-time performance in average for all combinations of features (i.e., points, lines, and points and lines) and for all the datasets, since the acquisition time for the KITTI sequences is 10 fps and for both the EuRoC MAV and the *low-textured* datasets is 20 fps, whereas our proposal performs in 15, 20, and 25 fps, respectively.

On the other hand, the LBA can be processed at around 20 Hz, which is fast enough for our purposes, as it runs in a parallel thread while the VO thread is continuously processing new frames. Finally, the *loop-closure* management, although being the most time consuming step of the algorithm, presents acceptable values. Please notice that these are average values, and for each particular sequence, loop closure varies substantially: those presenting small loops can perform loop closure in few milliseconds, whereas sequences with loops involving a significant amount of KFs and landmarks spend higher processing

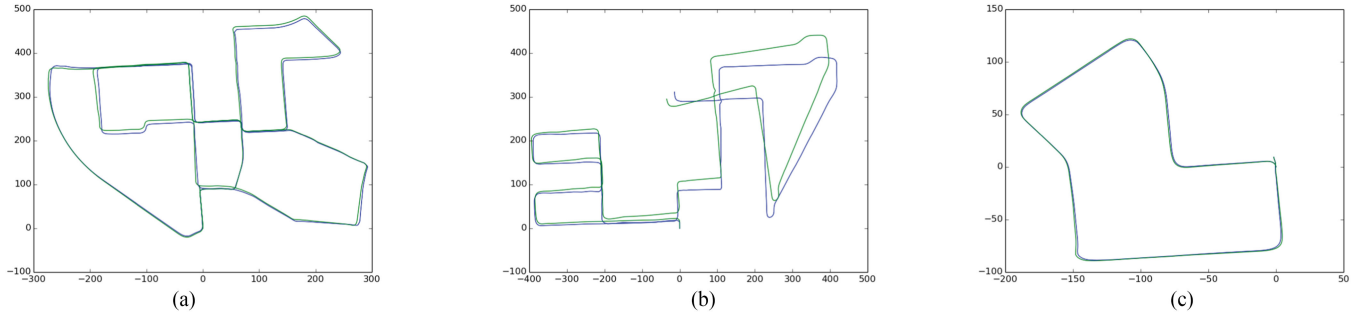


Fig. 8. Some trajectories estimated with PL-SLAM (in green) from the KITTI dataset (ground-truth in blue). (a) Trajectory estimated in the sequence *KITTI-00*, where a large amount of loop closures can be found. (b) Sequence *KITTI-08* does not present any loop closure, and hence the drift along the trajectory is not corrected. (c) Finally, the sequence *KITTI-07* presents a loop closure between the initial and final parts of the trajectory.

time than either the local mapping or the visual odometry procedures. In any case, since loop closure is computed in a parallel thread (and not at every frame), the rest of the system can still run in real time.

Finally, in order to illustrate the real-time capability of our system, we have also provided in Table IV the rate of captured frames before inserting a new keyframe for each dataset, thus giving a picture of how often the system demands a local mapping and loop-closure management.

E. Discussion

The experimental validation presented in this paper proves that PL-SLAM operates more robustly than point-only approaches, especially in low-textured scenarios where keypoints are difficult to extract and track. Moreover, our implementation achieves real-time performance for all the considered datasets, which include an heterogeneous set of both indoor and outdoor scenarios.

Regarding the accuracy, we have measured the relative RMSE between KFs for both the EuRoC MAV and the low-textured datasets, as relative measurements are not influenced by the presence of loop closures, hence leading to results that can be more comparable between different sequences. On the other hand, we have employed the metrics proposed in the KITTI benchmark (i.e., absolute translational errors expressed in % of the trajectory length and absolute rotational errors expressed in deg/100 m of the trajectory) in our experiments with the KITTI sequences, as they have become standardized for such dataset. PL-SLAM's results reveal better performance, in terms of accuracy, than its *point-only* and *line-only* approaches, and somewhat inferior performance to the ORB-SLAM2 method while at the same time, providing more robustness in challenging, low-textured scenes in where point-only approaches are prone to fail. This inferior performance is mainly explained by the fact that our approach does not perform LBA in every frame, unlike ORB-SLAM2, hence slightly increasing the final drift of the trajectory, especially in those sequences without loop closures.

Finally, and since our system architecture is strongly based on that of ORB-SLAM, we would like to highlight the following essential differences between these two approaches.

1) The inclusion of line segments as image features, which allows us to achieve robust camera localization in scenarios where keypoint-only methods usually perform poorly or even fail.
2) The inclusion of binary line descriptors in the loop-closure procedure, in order to make it more robust.
3) The implementation of the visual odometry thread as a frame-to-frame incremental motion estimation to meet the computational constraints that line segments introduce, unlike ORB-SLAM2, which continuously performs LBA between recent frames.

VIII. Conclusion

In this paper, we proposed a novel stereo visual SLAM system that extends our previous VO approach in [20], and that was based on the combination of both keypoints and line segment features. Our proposal, coined PL-SLAM, contributed with a robust and versatile system capable of working in all types of environments, including low-textured ones while producing geometrically meaningful maps. For that, we developed the first open-source SLAM system that runs in real time and simultaneously employs keypoints and line segment features. Our implementation was developed from scratch and was based on a BA solution that seamlessly deals with the combination of different kinds of features. Moreover, we extended the place recognition BoW approach in [21] for the case of simultaneously employing points and line segments, in order to enhance the loop-closure process. Our approach was tested on well-known datasets such as EuRoC MAV or KITTI, as well as in a sequence of stereo images recorded in a challenging low-textured scenario. In these experiments, PL-SLAM was compared with ORB-SLAM2 [53], a *point-only* system and a *line-only* system, obtaining superior performance in terms of robustness in most of the dataset sequences while still operating in real time.

With respect to the accuracy, our proposal gets similar results to ORB-SLAM2 for the EuRoC MAV and low-textured datasets when using the metric defined in [52], which computes the relative RMSE between KFs. On the other hand, the experiments with the KITTI dataset shows somewhat superior performance of the ORB-SLAM2 method following its standardized metrics for both absolute translational and rotational errors.

For future work, our implementation can benefit from faster keypoint front-ends, such as the ones in SVO [10], [26] and PL-SVO [35], where authors reduced the computational time of the feature tracking with a semidirect approach that estimates the position of the features as a consequence of the motion estimation, or from alternative tracking techniques [54] to improve robustness in difficult illumination conditions. Finally, our algorithm can be employed to obtain more accurate and refined maps by applying some SfM or multistereo techniques [18], [55] in order to filter the structural lines, hence obtaining more meaningful information of the structured parts of the environment.

REFERENCES

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM)," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–116, Sep. 2006.

[2] M. J. Milford, G. F. Wyeth, and D. Prasser, "RatSLAM: A hippocampal model for simultaneous localization and mapping," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2004, pp. 403–408.

[3] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *Int. J. Robot. Res.*, vol. 27, no. 6, pp. 647–665, 2008.

[4] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 1643–1649.

[5] M. Milford, "Vision-based place recognition: how low can you go?" *Int. J. Robot. Res.*, vol. 32, no. 7, pp. 766–789, 2013.

[6] G. Klein and D. Murray, "Parallel tracking and mapping on a camera phone," in *Proc. 8th ACM Int. Symp/Mixed Augmented Reality*, Orlando, FL, USA, Oct. 2009, pp. 83–86.

[7] F.-A. Moreno, J.-L. Blanco, and J. Gonzalez-Jimenez, "A constant-time SLAM back-end in the continuum between global mapping and submapping: Application to visual stereo SLAM," *Int. J. Robot. Res.*, vol. 35, no. 9, pp. 1036–1056, 2016.

[8] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proc. IEEE Int. Conf. Comput. Vision*, 2011, pp. 2320–2327.

[9] J. Engel, T. Schöps, and D. Cremers, "LSD -SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 834–849.

[10] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robot.*, vol. 33, no. 2, pp. 249–265, Apr. 2017.

[11] 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.

[12] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robot. Autom. Mag.*, vol. 18, no. 4, pp. 80–92, Dec. 2011.

[13] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, 2015.

[14] A. Bartoli and P. Sturm, "Structure-from-motion using lines: Representation, triangulation, and bundle adjustment," *Comput. Vision Image Understanding*, vol. 100, no. 3, pp. 416–441, 2005.

[15] C. Mei and E. Malis, "Fast central catadioptric line extraction, estimation, tracking and structure from motion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 4774–4779.

[16] J. Solà, T. Vidal-Calleja, and M. Devy, "Undelayed initialization of line segments in monocular SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2009, pp. 1553–1558.

[17] L. Zhang and R. Koch, "Hand-held monocular SLAM based on line segments," in *Proc. Irish Mach. Vision Image Process. Conf.*, 2011, pp. 7–14.

[18] M. Hofer, M. Maurer, and H. Bischof, "Line 3-D: Efficient 3-D scene abstraction for the built environment," in *Proc. German Conf. Pattern Recognit.*, 2015, pp. 237–248.

[19] J. Briales and J. Gonzalez-Jimenez , "A minimal closed-form solution for the perspective three orthogonal angles (P3oA) problem: Application to visual odometry," *J. Math. Imag. Vision*, vol. 55, no. 3, pp. 266–283, 2016.

[20] R. Gomez-Ojeda and J. Gonzalez-Jimenez, "Robust stereo visual odometry through a probabilistic combination of points and line segments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 2521–2526.

[21] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012.

[22] C. Cadena *et al.*, "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.

[23] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. AAAI Nat. Conf. Artif. Intell.*, 2002, pp. 593–598.

[24] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE ACM Int. Symp. Mixed Augmented Reality*, 2007, pp. 225–234.

[25] G. Klein and D. Murray, "Improving the agility of keyframe-based SLAM," in *Proc. Comput. Vision*, 2008, pp. 802–815.

[26] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semidirect monocular visual odometry," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 15–22.

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

[28] P. Smith, I. Reid, and A. J. Davison, "Real-time monocular SLAM with straight lines," *Proc. Brit. Mach. Vision Conf.*, 2006, pp. 17–26.

[29] E. Eade and T. Drummond, "Edge landmarks in monocular SLAM," *Image Vision Comput.*, vol. 27, pp. 588–596, Apr. 2009.

[30] G. Zhang and I. H. Suh, "Building a partial 3-D line-based map using a monocular SLAM," in *Proc. IEEE Conf. Robot. Autom.*, 2011, pp. 1497–1502.

[31] 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. 9545, no. 4, pp. 1364–1375, Apr. 2015.

[32] J. M. Coughlan and A. L. Yuille, "Manhattan world: Compass direction from a single image by Bayesian inference," in *Proc. 7th IEEE Int. Conf.Comput. Vision*, 1999, pp. 941–947.

[33] J. Solà, T. Vidal-Calleja, J. Civera, and J. M. M. Montiel, "Impact of landmark parametrization on monocular EKF-SLAM with points and lines," *Int. J. Comput. Vision*, vol. 97, pp. 339–368, Sep. 2011.

[34] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh, "Building a 3-D line-based map using stereo SLAM," *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1364–1377, Dec. 2015.

[35] R. Gomez-Ojeda, J. Briales, and J. González-Jiménez, "PL-SVO: Semidirect monocular visual odometry by combining points and line segments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 4211–4216.

[36] 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.

[37] R. G. VonGioi, 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.

[38] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual SLAM," in *Proc. Int. Conf. Comput. Vision*, Nov. 2011, pp. 2352–2359.

[39] D. Sibley, C. Mei, I. D. Reid, and P. Newman, "Adaptive relative bundle adjustment.," in *Proc. Robot., Sci. Syst.*, vol. 32, 2009, paper 33.

[40] G. Sibley, C. Mei, I. Reid, and P. Newman, "Vast-scale outdoor navigation using adaptive relative bundle adjustment," *Int. J. Robot. Res.*, vol. 29, no. 8, pp. 958–980, 2010.

[41] F.-A. Moreno, J.-L. Blanco, and J. Gonzalez-Jimenez, "A constant-time SLAM back-end in the continuum between global mapping and submapping: Application to visual stereo SLAM," *Int. J. Robot. Res.*, vol. 35, no. 9, pp. 1036–1056, 2016.

[42] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. Robot.*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.

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

[44] F.-A. Moreno, J.-L. Blanco, and J. González-Jiménez, "ERODE: An efficient and robust outlier detector and its application to stereovisual odometry," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 4691–4697.

[45] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 2100–2106.

[46] M. Lopez-Antequera, R. Gomez-Ojeda, N. Petkov, and J. Gonzalez-Jimenez, "Appearance-invariant place recognition by discriminatively training a convolutional neural network," *Pattern Recognit. Lett.*, vol. 92, pp. 89–95, 2017.
[47] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *Int. J. Robot. Res.*, vol. 28, pp. 595–599, May 2009.
[48] J.-L. Blanco-Claraco, F.-Á. Moreno-Dueñas, and J. González-Jiménez, "The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario," *Int. J. Robot. Res.*, vol. 33, no. 2, pp. 207–214, 2014.
[49] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 3354–3361.
[50] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G 2 O: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 3607–3613.
[51] M. Burri *et al.* "The EuRoC microaerial vehicle datasets," *Int. J. Robot. Res.*, vol. 35, pp. 1157–1163, 2016.
[52] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 573–580.
[53] 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.
[54] R. Gomez-Ojeda and J. Gonzalez-Jimenez, "Geometric-based line segment tracking for HDR stereo sequences," in *Proc. Int. Conf. Intell. Robots Syst.*, 2018, pp. 69–74.
[55] M. Pizzoli, C. Forster, and D. Scaramuzza, "REMODE: Probabilistic, monocular dense reconstruction in real time," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 2609–2616.

Ruben Gomez-Ojeda was born in Spain in 1988. He received the B.S.–M.S. degree in industrial engineering and the M.S. degree in mechatronics in 2012 and 2014, respectively, from the University of Malaga, Malaga, Spain, where he is currently working toward the Ph.D. degree in computer vision and robotics with the Machine Perception and Intelligent Robotics Group, under the supervision of J. Gonzalez-Jimenez.

In 2016, he was a Visiting Researcher with the Robotics and Perception Group, University of Zurich, Zürich, Switzerland, with Davide Scaramuzza. In 2018, he was an Intern with Oculus VR, Zurich, Switzerland, under the supervision of Matia Pizzoli, where he rejoined in 2019. His research interests include vision-based navigation, place recognition, autonomous robotics, and AR/VR.

Francisco-Angel Moreno was born in Spain in 1981. He received the B.S. degree in technical telecommunications engineering from the University of Jaen, Jaén, Spain, in 2002, the M.S. degree in telecommunications engineering and Ph.D. degree in computer vision from the Machine Perception and Intelligent Robotics group under the supervision of J. Gonzalez-Jimenez and J.-L. Blanco from the University of Malaga, Málaga, Spain, in 2007 and 2015, respectively.

During his Ph.D., he did two research stays, the first one in 2010 at the University of Bristol, Bristol, U.K., and the second one in the University of Lincoln, Lincoln, U.K., in 2013. He is currently a provisional Assistant Professor of Industrial Engineering with the University of Malaga. His research interests include vision-based navigation, telepresence robotics, and human–machine interaction.

David Zuñiga-Noël was born in Spain in 1993. He received the B.S. degree in computer science and the M.S. degree in mechatronics in 2016 and 2017, respectively, from the University of Malaga, Spain, where he is currently working toward the Ph.D. degree in computer vision and robotics with the Machine Perception and Intelligent Robotics Group.

His research interests include vision-based navigation, autonomous robotics, and sensor fusion.

Davide Scaramuzza was born in Italy in 1980. He received the Ph.D. degree in robotics and computer vision from ETH Zurich, Zurich, Switzerland, in 2008.

He was a Postdoc with the University of Pennsylvania, Philadelphia, PA, USA. He is currently an Associate Professor of Robotics and Perception with the Department of Informatics (University of Zurich) and the Department of Neuroinformatics (University of Zurich and ETH Zurich). From 2009 to 2012, he led the European project "sFly," which introduced the PX4 autopilot and pioneered visual-SLAM-based autonomous navigation of microdrones. He coauthored the book *Introduction to Autonomous Mobile Robots* (MIT Press, 2004) and more than 100 papers on robotics and perception. His research interests include the intersection of robotics, computer vision, and neuroscience.

Prof. Scaramuzza was the recipient of the Misha Mahowald Neuromorphic Engineering Award, the IEEE Robotics and Automation Society Early Career Award, the SNSF-ERC Starting Grant, a Google Research Award, the European Young Research Award, and several conference and journal paper awards for his research contributions.

Javier Gonzalez-Jimenez was born in Spain in 1962. He received the B.S. degree in electrical engineering from the University of Seville, Seville, Spain, in 1987, and the Ph.D. degree in robotics from the Department of Ingenieria de Sistemas y Automatica, University of Malaga, Málaga, Spain, in 1993.

He is currently the Head of the Machine Perception and Intelligent Robotics (MAPIR) Group and Full Professor in robotics and computer vision with the University of Malaga. In 1990–1991, he was with Field Robotics Center, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, working on mobile robots as part of his Ph.D. Since 1996, he has been heading Spanish and European projects on mobile robotics and perception, in which he has authored/coauthored more than 50 JCR-ISI papers, 100 international conferences, and three books.