Authors' accepted manuscript: IEEE Transactions on Robotics, 2019. The final publication is available at: https://doi.org/10.1109/TRO.2019.2899783

PL-SLAM: a Stereo SLAM System through the Combination of Points and Line Segments

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

***Abstract*—Traditional approaches to stereo visual 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 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 3D elements, which can be exploited to infer valuable, high-level scene structures like planes, empty spaces, ground plane, etc. (not addressed in this work). Our proposal has been tested with several popular datasets (such as KITTI and EuRoC), and is compared to state of the art methods like ORB-SLAM, 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 will be released for the benefit of the community.**

***Index Terms*—Stereo Visual SLAM, line segment features, bundle adjustment, loop closure**

I. Introduction

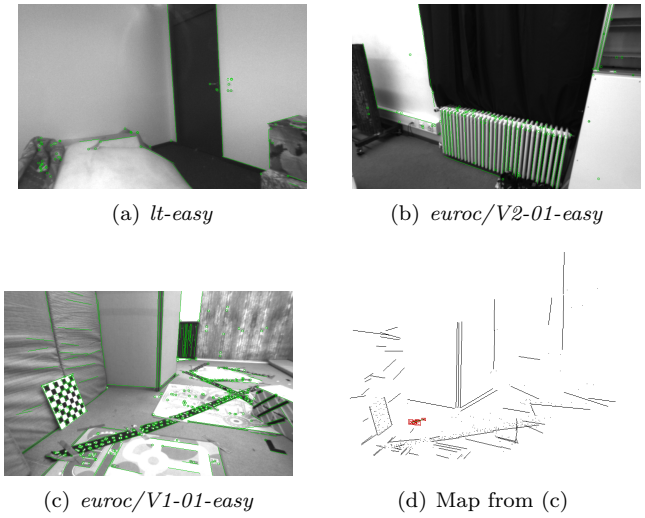
In recent years, visual Simultaneous Localization And Mapping (SLAM) is firmly progressing towards the degree of reliability required for fully autonomous vehicles: mobile robots, self-driving cars or Unmanned Aerial Vehicles (UAVs). In a nutshell, the SLAM problem consists of the estimation of the vehicle 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.

This work has been supported by the Spanish Government (project DPI2017-84827-R and grant BES-2015-071606) and the Andalusian Government (project TEP2012-530).

R. Gomez-Ojeda, F.A. Moreno, D. Zuñiga-Noël, and J. Gonzalez-Jimenez are with the Machine Perception and Intelligent Robotics (MAPIR) Group, University of Malaga. (email: rubengooj@gmail.com).

D. Scaramuzza is with the Robotics and Perception Group, Dep. of Informatics, University of Zurich, and Dep. of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland.



(a) *lt-easy* (b) *euroc/V2-01-easy* (c) *euroc/V1-01-easy* (d) Map from (c)

Figure 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 (3D line-segments).

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 classified into *direct* and *feature-based* systems. The first group, i.e. *direct* methods, estimates 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 for recovering the pose by minimizing the projection errors [12], [13].

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 3D points and segments provide more structural information from the environment than point-only maps, as can be seen in the example shown in Figure 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 like 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 KITTI or EuRoC datasets. In this work, 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, while, for the residual computation, we represent segments in the map with 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 robot navigation, following a new bag-of-words 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 approach presented in [20], and combines both point and line segment features to perform real-time robot localization and mapping. The main contributions of this work are:

- 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, while obtaining similar accuracy in the rest of the scenarios. Because of the consideration of both kinds of features, our proposal also produces rich geometrical maps.
- A new implementation of the bundle adjustment process that seamlessly accounts for both kinds of features while refining the poses of the keyframes.
- An extension of the bag-of-words 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 proposed system and an open source version of the developed C++ PL-SLAM library are 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 re-projection errors) to estimate the robot poses [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 step with the continuous observations of the point features. However, all of the previous approaches 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 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 the authors propose an algorithm to integrate them into a monocular Extended Kalman Filter SLAM system (EKF-SLAM). In the refered paper, the line detection relies on an hypothesize-and-test method that connects several near 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, authors in [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 2D 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 2D points on the floor plane (placed the intersection point between the line and such plane), while horizontal

lines are represented by their two end-points placed on the floor. Finally, the proposed models is incorporated into an EKF-SLAM system. Another model-based approach is reported in [31], where the authors 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], [33] address a generic approach that compares the impact of eight different landmark parametrization 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 3D 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 3D projections, and an orthonormal representation for the back-end optimization. Unfortunately, neither the source code is available nor the employed dataset contain any ground-truth, therefore it has not been possible to carry out a comparison against 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, in [35] the authors extended the semi-direct 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 end-points 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 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 increases the computational burden of ORB-SLAM. Besides this computational drawback, when working with features detected with the LSD detector, the variance of the endpoints becomes quite pronounced, specially in challenging illumination conditions or very low-textured scenes, making more difficult wide-baseline tracking and matching between line features in non-consecutive 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 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.

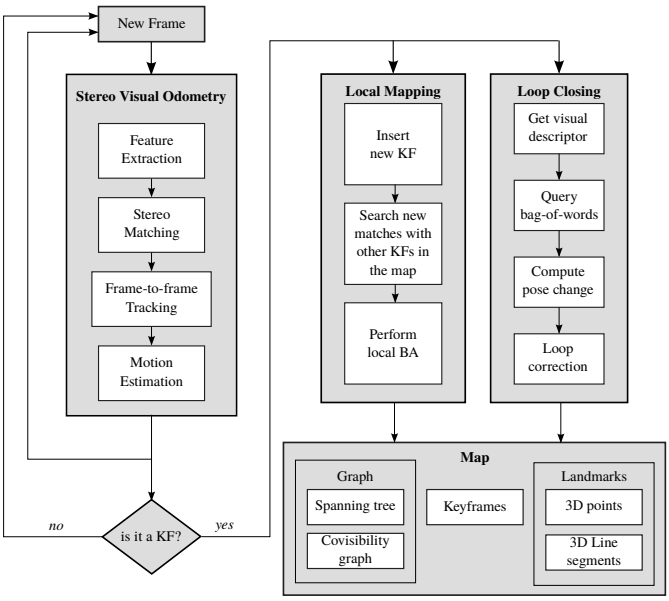


Figure 2. Scheme of the stereo PL-SLAM system.

III. PL-SLAM Overview

The general structure of the PL-SLAM system proposed here is depicted in Figure 2, and its main modules are described in the following sections. As it is common to other SLAM systems (being ORB-SLAM [13] the most popular method nowadays), our proposal is also based on 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.

Map. The map consists of i) a set of keyframes (KFs), ii) the detected 3D landmarks (both keypoints and line segments), iii) a covisibility graph and iv) a spanning tree.

The keyframes 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 3D 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 3D position while, for the line segments, we keep both their direction and the estimated 3D coordinates of their endpoints.

Finally, the covisibility information, as in [38], is modeled by a graph: each node represents a KF, and edges between KFs are created only if they share a minimum number of landmarks, which in this work is set to 20 landmarks (see Figure 3 for an example), allowing for real-time bundle adjustment along the local map.

Similarly, in order to perform a faster loop closure optimization, we also form the so-called *essential graph*, which is less dense than the covisibility graph because an edge between two KFs is created when they share more

than 100 landmark observations. Finally, the map also contains a spanning tree, which is the minimum connected representation of a graph that includes all the KFs.

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 3D position and their associated uncertainty represented by covariance matrices. The 3D 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 to the map. Further discussion about this 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.

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 (KF poses and landmarks positions) is performed. A detailed description of this procedure will be presented in Section V.

Loop Closure. In parallel to local mapping, a loop closure detection is carried out by extracting a visual descriptor for each image, based on a bag-of-words approach, as will be described in Section VI. All the visual descriptors of the captured frames 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 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.

These mapping and loop closure approaches are identical to the ones followed in ORB-SLAM, being aimed to reduce the high computational burden that general BA involves (along with the incorporation of recent sparse algebra techniques). 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], [42]).

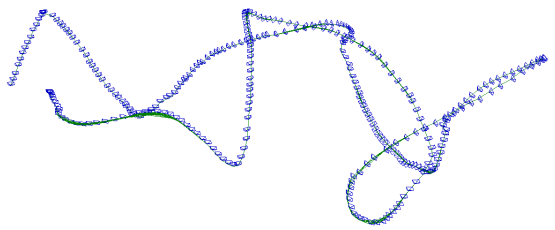


Figure 3. Covisibility graph in the sequence *lt-first* for which we have represented the edges connecting the keyframes with green lines.

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. Basically, both points and line segments are tracked along a sequence of stereo frames (see Figure 1), and then the 3D motion of the camera (and also its uncertainty) is computed by minimizing the projection errors.

A. Point Features

In this work 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 which 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 Line Segment Detector (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 endpoint. Notice that this filter helps the system to retain a larger amount of structural

lines, which allows for the formation of more consistent maps based on points and lines (see Figure 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 modelled 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 6D 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 equation (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 Equation (2), we need the uncertainty of the pose increment between non-consecutive frames. Since Equation (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 non-consecutive KFs.

V. Local Mapping

This section describes the behavior of the system when a new KF is inserted, which essentially consists in performing the bundle adjustment 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:

1) An index for the keyframe.
2) The information of its 3D pose, which comprises an absolute pose and the relative pose from the previous KF, along with their associated uncertainties.
3) The new 3D landmarks, which are initialized by storing both their 2D 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 bundle adjustment 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 3D position of each point $\mathbf{X}_{wj}$, and also the 3D 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 2D distance between the observation of the j-th map point in the i-th 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 j-th 3D point $\mathbf{X}_{wj}$ (in world coordinates) into the local reference system of the i-th 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 3D line segment and its corresponding infinite line in the image plane. In this case, the error $\mathbf{e}_{ik}$ between the k-th line observed in the i-th 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 3D 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 k-th line segment in the i-th KF, which can be obtained with the cross product between the 2D 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:

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

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 appending the influence of each observation to its corresponding block, as showed in Equation (9). Notice that, for the rest of observations that belong to the KFs that are not part of the local map, their Jacobian matrixes $\frac{\partial \mathbf{e}_{ij}}{\partial \boldsymbol{\xi}_{iw}}$ and $\frac{\partial \mathbf{e}_{ik}}{\partial \boldsymbol{\xi}_{iw}}$ are equal to zero, since here we only optimize the local map while 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 3 observations, as they are less meaningful.

VI. Loop Closure

In this work, we adopt a bag of words (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 off-line 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 seek 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 keyframes 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.

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 our work, segments are also augmented with binary descriptors, we propose to build both specific visual vocabularies and databases for them. This way, at each time step, the most similar images in the databases of keypoints and 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 segments 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 Figure 4(a) has been computed employing only ORB keypoints to build both the vocabulary and the database, while the other (Figure 4(b)) relies only on segments. 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 non-similar 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.

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

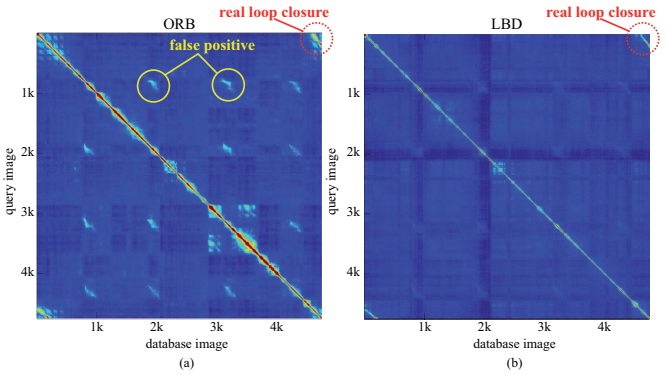


Figure 4. Similarity matrices for a certain dataset where the (a) ORB keypoint-only bag-of-words approach yields false positives that are not present in the (b) LBD line-only approach.

In this work, 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, while 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):

$$\begin{aligned} s_t = 0.5\left(n_k/\left(n_k+n_l\right) + d_k/\left(d_k+d_l\right)\right) s_k + \\ 0.5\left(n_l/\left(n_k+n_l\right) + d_l/\left(d_k+d_l\right)\right) s_l, \end{aligned} \tag{10}$$

where n_k and n_l are the number of keypoints and segments extracted in the image, respectively, and d_k and d_l are the dispersion values, which 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 segments, the midpoint coordinates are employed. Note that this formulation gives the same importance to both kinds of features (hence the 0.5 factor), although it 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). 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]. The compared approaches were: i) using just s_k, ii) using just s_l, iii) using both s_k and s_l but taking into account only the *strength* criteria, and iv) using both s_k and s_l but taking into account only the *dispersion* criteria. Our proposed strategy yielded better results overall in terms of precision-recall for all the 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 above mentioned 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 *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:

i) The maximum eigenvalue of the covariance matrix $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\xi}}_{ij}}$ is inferior to 0.01.
ii) The obtained translation and rotation cannot rise over 0.50 meters and 3.00 degrees, respectively.
iii) 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 provides 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 pose-graph optimization (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:

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

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 (please, refer to 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 scenarios from different datasets, for which we estimate both the trajectory and the map in several video sequences. We also compare our approach with the stereo version of ORB-SLAM [51] by employing its open source implementation, which is considered one of the state-of-the-art methods for stereo visual SLAM.

All the experiments have been run on an Intel Core i5-6600 CPU @ 3.30GHz and 16GB RAM without GPU parallelization. In order to fairly compare all the sequences, we have only considered the relative errors between the KFs positions, disregarding the accuracy of the absolute poses since it dramatically varies depending on whether or not the sequence presents loop closures. 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 between frames. Therefore, we could not include their results in this paper.

In the following, we present examples of the trajectories and maps estimated by PL-SLAM, together with the average errors committed by our proposal, ORB-SLAM, a *point-only* system (P-SLAM), and a *line-only* system (L-SLAM).

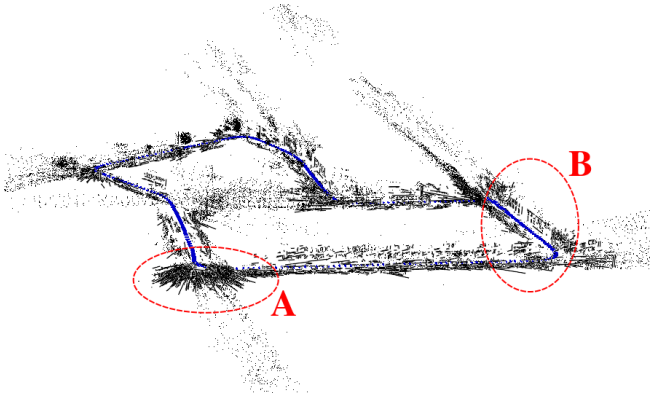


Figure 5. Map (in black) comprising points and line segments, and the trajectory (in blue) obtained with PL-SLAM from an outdoor environment in the sequence *KITTI-07*. The map presents noisy measurements in some parts (e.g. zone A), and lines from the environment, such as parts of the buildings (e.g. zone B).

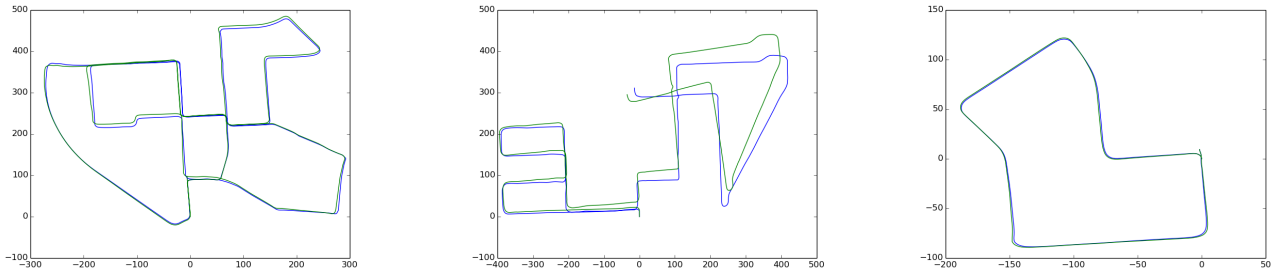
A. KITTI dataset

First, we have tested PL-SLAM on the well-known KITTI dataset [49], using the 11 sequences that provide ground truth, yielding the results presented in Table I. Note that this is an urban dataset with highly textured image sequences and, as expected, the exploitation of line segments barely increases the accuracy, since the number of detected points is generally sufficient for a proper operation of the SLAM system.

Still, PL-SLAM shows a slightly superior performance for most of the datasets in comparison to the *point-only* approach and also to the ORB-SLAM system, specially in the rotation estimation. Unsurprisingly, the results confirm worse performance of the *line-only* system 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).

As an illustrative example, Figure 5 depicts the trajectory and the map estimated by PL-SLAM (LSD) 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 significant 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 the zone marked as B in the figure, where the buildings can be clearly noticed, 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.

Finally, Figure 6 depicts the estimated trajectory obtained with PL-SLAM (LSD) in three sequences from the KITTI dataset that present different number of loop closures. It can be noted the importance of correcting the drift in long sequences to obtain accurate absolute



(a) (b) (c)

Figure 6. 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) The 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.

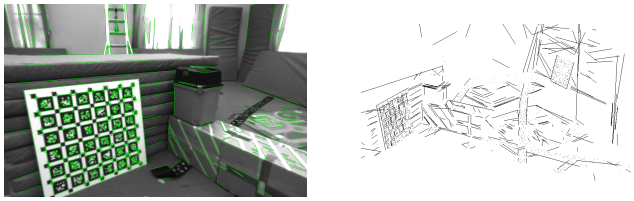
Table I
MEAN RESULTS IN THE KITTI DATASET [49]. THE TRANSLATION ERRORS ARE EXPRESSED IN %, WHILE THE ROTATION ERRORS ARE ALSO EXPRESSED RELATIVELY TO THE TRANSLATION IN $deg/100m$. A DASH INDICATES THAT THE EXPERIMENT FAILED.

	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	2.57	3.00	3.29	7.99	**2.38**	**2.32**	2.51	5.30
01	5.59	2.32	-	-	3.23	2.17	**1.51**	**1.65**
02	2.34	1.97	6.23	12.35	**2.20**	**1.86**	2.31	3.88
03	3.68	2.96	6.33	19.17	3.40	**3.17**	**3.31**	5.52
04	2.30	1.16	-	-	1.57	**1.10**	**1.45**	2.21
05	1.94	2.25	2.58	7.07	**1.67**	**1.85**	1.75	4.52
06	2.48	1.76	3.81	9.71	2.02	**1.28**	**1.45**	2.86
07	2.46	3.82	2.71	6.71	**1.57**	**2.60**	2.20	5.43
08	**2.31**	**2.61**	6.97	13.67	2.42	2.65	2.50	5.03
09	1.57	**1.99**	6.56	11.55	**1.49**	2.12	1.63	3.95
10	1.64	2.80	5.82	11.77	**1.61**	**2.79**	1.81	6.43

solutions (refer to Figure 6(a,c)), in contrast to the results obtained in sequences without loop closures, as the one presented in Figure 6(b). Nevertheless, relative translation and rotation errors are similar for the three sequences, as shown in Table I.

B. EuRoC MAV dataset

The EuRoC MAV dataset [52] consists of 11 stereo sequences recorded with a 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 Figure 7(b), where two different parts are clearly visible. The first one shows the features extracted from the non-structured 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



(a) (b)

Figure 7. Mapping results in the *V1-01-easy* sequence from the EuRoC MAV dataset. (a) Features tracked between two consecutive keyframes. (b) Resulting 3D map for the sequence. The checkerboard and the boxes in the scene are clearly reflected in the left part of the map, while more noisy features can be found in the rest, as a consequence of factors like non-textured surfaces, high illumination, etc.

Table II
RELATIVE RMSE ERRORS IN THE EUROC MAV DATASET [52]. A DASH INDICATES THAT THE EXPERIMENT FAILED.

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**	-

be employed to extract high-level meaningful information from them.

Finally, Table II shows the mean relative RMSE errors 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 estimate camera motion and to improve the system robustness. In this case, both the *point-only* and the *line-only* approaches yield worse results than PL-SLAM, while ORB-SLAM fails in several sequences since feature tracking is prone to be lost.

C. Low-textured Scenarios

Finally, we have assessed the performance of the compared methods in challenging low-textured scenarios. For

Table III
Relative RMSE errors in low-textured sequences recorded with GT data from an OptiTrack system. A dash indicates that the experiment failed.

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

that, we have recorded a set of stereo sequences 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 (LSD) system for the sequence *lt-medium* is shown in Figure 3, where a loop closure between the initial and the final part of the trajectory can be observed. These experiments (see Table III) 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, achieving a good performance in terms of accuracy.

D. Performance

Finally, regarding the computational burden, 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 for all combinations of features (i.e. points, lines, and points and lines) for the datasets with lower resolution (752×480), and nearly real time in the KITTI dataset, and in all cases our approach performs faster than ORB-SLAM2, even when considering the two different features. On the other hand, the local bundle adjustment (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, it can be seen that the loop closure management is the most time consuming step of the algorithm although it is computed in a parallel thread (and not at every frame), so that the rest of the system can still run in real time.

E. Discussion

As our system architecture is similar to in ORB-SLAM [13] we would like to clarify the essential differences between the two approaches: i) 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 fails, ii) the inclusion of binary line descriptors in the loop closure procedure, in order to make it more robust, and iii) the implementation of the visual odometry thread as a frame-to-frame incremental motion estimation to meet the computational constraints that line segments introduces, unlike ORB-SLAM2, which performs motion-only Bundle

[1]http://optitrack.com/

Table IV
Average runtime of each part of the algorithm.

VO	KITTI 1241 × 376	EuRoC MAV 752 × 480	Low-Textured 752 × 480
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

Adjustment between recent frames. In any case, we do not claim to obtain more accuracy than ORB-SLAM2 in common environments. In fact, both ORB-SLAM2 and our approach perform similarly in such environments, with slightly superior results for ORB-SLAM2 in accuracy and for our approach in computational burden. In essence, we claim more robustness in low-textured ones, where the number of point features dramatically decreases. Is in this kind of scenarios where our proposal achieves better performance.

Finally, although it was mentioned in the previous section, it is important to highlight that the results achieved by ORB-SLAM2 in these experiments have been computed by evaluating keyframe-to-keyframe pose estimation errors, which is a different metric than the one employed in the ORB-SLAM2 original paper, hence the differences in the results.

VIII. Conclusions

In this paper we have proposed a novel stereo visual SLAM system that extends our previous VO approach in [20], and that is based on the combination of both keypoints and line segment features. Our proposal, coined PL-SLAM, contributes 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 have developed the first open source SLAM system that runs in real time and that simultaneously employs keypoints and line segment features. Our ad-hoc implementation has been developed from scratch and its based on a bundle adjustment solution that seamlessly deals with the combination of different kinds of features. Moreover, we have extended the place recognition bag-of-words approach in [21] for the case of simultaneously employing points and line segments in order to enhance the loop-closure process. Our approach has been tested on popular benchmarking datasets such as KITTI, or EuRoC MAV, as well as in a sequence of stereo images recorded in a challenging low-textured scenario. PL-SLAM has been compared with ORB-SLAM [13], a

point-only system and a line-only system, obtaining superior performance in terms of both accuracy and robustness in most of the dataset sequences.

For future work, our implementation can benefit from better 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 semi-direct approach that estimates the position of the features as a consequence of the motion estimation. Finally, our algorithm can be employed to obtain more accurate and refined maps by applying some SfM or Multi-Stereo techniques [18], [53] 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 Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99–116, 2006.

[2] M. J. Milford, G. F. Wyeth, and D. Prasser, " RatSLAM: A Hippocampal Model for Simultaneous Localization and Mapping ," *Proceeding of the 2004 IEEE international Conference on Robotics & Automation*, pp. 403–408, 2004.

[3] M. Cummins and P. Newman, " FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research*, 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," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1643–1649, 2012.

[5] M. Milford, "Vision-based place recognition: how low can you go?," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 766–789, 2013.

[6] G. Klein and D. Murray, "Parallel Tracking and Mapping on a Camera Phone," in *Proc. Eigth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'09)*, (Orlando), pp. 83–86, October 2009.

[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," *The International Journal of Robotics Research*, 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 *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2320–2327, IEEE, 2011.

[9] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European Conference on Computer Vision*, pp. 834–849, Springer, 2014.

[10] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[11] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[12] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *Robotics & Automation Magazine, IEEE*, vol. 18, no. 4, pp. 80–92, 2011.

[13] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[14] A. Bartoli and P. Sturm, "Structure-from-motion using lines: Representation, triangulation, and bundle adjustment," *Computer Vision and 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 *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 4774–4779, IEEE, 2006.

[16] J. Solà, T. Vidal-Calleja, and M. Devy, "Undelayed initialization of line segments in monocular SLAM," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pp. 1553–1558, 2009.

[17] L. Zhang and R. Koch, "Hand-held monocular SLAM based on line segments," in *Machine Vision and Image Processing Conference (IMVIP), 2011 Irish*, pp. 7–14, IEEE, 2011.

[18] M. Hofer, M. Maurer, and H. Bischof, "Line3D: Efficient 3D Scene Abstraction for the Built Environment," in *German Conference on Pattern Recognition*, pp. 237–248, Springer, 2015.

[19] J. Briales and J. Gonzalez-Jimenez, "A Minimal Closed-form Solution for the Perspective Three Orthogonal Angles (P3oA) Problem: Application To Visual Odometry," *Journal of Mathematical Imaging and 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 *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2521–2526, May 2016.

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

[22] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard, "Simultaneous Localization And Mapping: Present, Future, and the Robust-Perception Age," *CoRR*, vol. abs/1606.05830, 2016.

[23] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Aaai/iaai*, pp. 593–598, 2002.

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

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

[26] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 15–22, IEEE, 2014.

[27] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2564–2571, IEEE, 2011.

[28] P. Smith, I. Reid, and a. J. Davison, "Real-Time Monocular SLAM with Straight Lines," *Procedings of the British Machine Vision Conference 2006*, pp. 3.1–3.10, 2006.

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

[30] G. Zhang and I. H. Suh, "Building a partial 3D line-based map using a monocular SLAM," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1497–1502, IEEE, 2011.

[31] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, "StructSLAM : Visual SLAM with Building Structure Lines," *IEEE Transactions on Vehicular Technology*, vol. 9545, no. c, pp. 1–1, 2015.

[32] J. M. Coughlan and A. L. Yuille, "Manhattan world: Compass direction from a single image by bayesian inference," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2, pp. 941–947, IEEE, 1999.

[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," *International Journal of Computer Vision*, vol. 97, pp. 339–368, sep 2011.

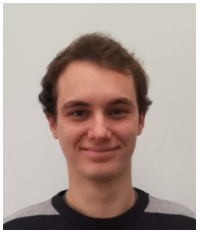
[34] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh, "Building a 3-D Line-Based Map Using Stereo SLAM," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1364–1377, 2015.

[35] R. Gomez-Ojeda, J. Briales, and J. González-Jiménez, "PL-SVO: Semi-Direct Monocular Visual Odometry by Combining Points and Line Segments," in *Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4211–4216, IEEE/RSJ, 2016.

[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 *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 4503–4508, IEEE, 2017.

[37] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2010.
[38] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual SLAM," in *2011 International Conference on Computer Vision*, pp. 2352–2359, Nov 2011.
[39]
[40] G. Sibley, C. Mei, I. Reid, and P. Newman, "Vast-scale outdoor navigation using adaptive relative bundle adjustment," *The International Journal of Robotics Research*, 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," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1036–1056, 2016.
[42] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.
[43] L. Zhang and R. Koch, "An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency," *Journal of Visual Communication and Image Representation*, 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 *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 4691–4697, IEEE, 2013.
[45] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2100–2106, IEEE, 2013.
[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 Recognition Letters*, 2017.
[47] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *The International Journal of Robotics Research*, 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," *The International Journal of Robotics Research*, 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 *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 3354–3361, IEEE, 2012.
[50] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 3607–3613, IEEE, 2011.
[51] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
[52] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016.
[53] M. Pizzoli, C. Forster, and D. Scaramuzza, "REMODE: Probabilistic, monocular dense reconstruction in real time," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 2609–2616, IEEE, 2014.

Ruben Gomez-Ojeda (1988, Spanish) is a PhD student associated with the Machine Perception and Intelligent Robotics (MAPIR) group at the University of Malaga, under the supervision of J. Gonzalez-Jimenez. He received his B.S.-M.S. in Industrial Engineering in 2012, and his M.S. in Mechatronics in 2014 from the University of Malaga, Spain. In 2016, he was a Visiting Researcher at the Robotics and Perception Group from the University of Zurich with D.Scaramuzza. His research interests include vision based navigation, place recognition, and autonomous robotics.



David Zuñiga-Noël (1993, Spanish) received his B.S. degree in Computer Science in 2016 and his M.S. in Mechatronics in 2017, both of them obtained at the University of Malaga, Spain. In 2016 he joined the Machine Perception and Intelligent Robotics (MAPIR) group, where he started to work toward the PhD degree in Computer Vision and Robotics. His research interests include vision based navigation, autonomous robotics, and sensor fusion.

Francisco-Angel Moreno (1981, Spanish) received his B.S. degree in Technical Telecommunications Engineering from the University of Jaen in 2002. He received his M.S. degree in Telecommunications Engineering from the University of Malaga in 2007. In 2009 he joined the MAPIR group where he received his PhD degree in 2015 under the supervision of Javier Gonzalez-Jimenez and Jose-Luis Blanco. During his PhD he did two research stays, the first one in 2010 at the University of Bristol, and the second one in the University of Lincoln in 2013. His research interests include vision based navigation, telepresence robotics and human-machine interaction.

Davide Scaramuzza (1980, Italy) received the Ph.D. degree in robotics and computer vision from ETH Zurich, Zurich, Switzerland, in 2008, and a Postdoc at University of Pennsylvania, Philadelphia, PA, USA. He is a Professor of Robotics with University of Zurich, where he does research at the intersection of robotics, computer vision, and neuroscience. From 2009 to 2012, he led the European project sFly, which introduced the world's first autonomous navigation of microdrones in GPS-denied environments using visual-inertial sensors as the only sensor modality. He coauthored the book Introduction to Autonomous Mobile Robots (MIT Press). Dr. Scaramuzza received an SNSF-ERC Starting Grant, the IEEE Robotics and Automation Early Career Award, and a Google Research Award for his research contributions.

Javier Gonzalez-Jimenez (1962, Spanish) is the head of the MAPIR group and full professor at the University of Malaga. Prof. Gonzalez-Jimenez received his B.S. degree in Electrical Engineering from the University of Seville in 1987. He joined the Department of "Ingenieria de Sistemas y Automatica" at the University of Malaga in 1988 and received his Ph.D. from this University in 1993. In 1990-1991 he was at the Field Robotics Center, Robotics Institute, Carnegie Mellon University (USA) working on mobile robots as part of his PhD. Since 1996 he has been heading Spanish and European projects on mobile robotics and perception. In these areas he is (co)author of more than 50 JCR-ISI papers, 100 international conferences and 3 books.