Relative Localization for Aerial Manipulation with PL-SLAM

A. Pumarola, Alexander Vakhitov, Antonio Agudo, F. Moreno-Noguer and A. Sanfeliu

Abstract This chapter explains a precise SLAM technique, PL-SLAM, that allows to simultaneously process points and lines and tackle situations where point-only based methods are prone to fail, like poorly textured scenes or motion blurred images where feature points are vanished out. The method is remarkably robust against image noise, and that it outperforms state-of-the-art methods for point based contour alignment. The method can run in real-time and in a low cost hardware.

1 Introduction

The precise localization of an aerial robot is crucial for manipulation. In this section, we tackle the task of precise localization relative to a close up workspace for robot inspection and manipulation. The method requires robustness to poorly textured surfaces and, when the tracker is lost, relocalize the robot when passing over an already seen area. SLAM methods have proven effective to accurately estimate trajectories while keeping record of previously seen areas.

A. Pumarola (✉) · A. Agudo · F. Moreno-Noguer · A. Sanfeliu
CSIC-UPC, Institut de Robótica i Informática Industrial,
Llorens i Artigas 4-6, 08028 Barcelona, Spain
e-mail: apumarola@iri.upc.edu

A. Agudo
e-mail: aagudo@iri.upc.edu

F. Moreno-Noguer
e-mail: fmoreno@iri.upc.edu

A. Sanfeliu
e-mail: sanfeliu@iri.upc.edu

A. Vakhitov
Skolkovo Institute of Science and Technology, Ulitsa Nobelya, 3, 121205 Moskva, Moscow Oblast, Russia
e-mail: a.vakhitov@skoltech.ru

A. Ollero and B. Siciliano (eds.), *Aerial Robotic Manipulation*,
Springer Tracts in Advanced Robotics 129,
https://doi.org/10.1007/978-3-030-12945-3_17

Since the groundbreaking Parallel Tracking And Mapping (PTAM) [1] algorithm was introduced by Klein and Murray in 2007, many other real-time visual SLAM approaches have been proposed, including the feature point-based ORB-SLAM [2], and the direct-based methods LSD-SLAM [3] and RGBD-SLAM [4] that optimize directly over image pixels. Among them, the ORB-SLAM seems to be the current state-of-the-art, yielding better accuracy than the direct methods counterparts. However, it is prone to fail when dealing with poorly textured frames or when feature points are temporary vanished out due to, e.g., motion blur. This kind of situations are often encountered in man-made workspaces. However, despite the lack of reliable feature points, these environments may still contain a number of lines that can be used in a similar way.

Building upon the ORB-SLAM framework, we present PL-SLAM (Point and Line SLAM) [5], a solution that can simultaneously leverage points and lines information. Lines are parameterized by their endpoints, whose exact location in the image plane is estimated following a two-step optimization process. This representation is robust to occlusions and mis-detections and enables integrating the line representation within the SLAM machinery. The resulting approach is very accurate in poorly textured environments, and also, improves the performance of ORB-SLAM in highly textured sequences.

2 PL-SLAM Method

PL-SLAM pipeline highly resembles that of ORB-SLAM, in which we have integrated the information provided by line features (see Fig. 1). We next briefly review the main building blocks in which line operations are performed. For a description of the operations involving point features, the reader is referred to [2]. First, point and line features are detected using [6] an LSD [7], respectively. Then, after having obtained an initial set of map-to-image point and line feature pairs, all features of the local map are projected onto the image to find further correspondences. If the image contains sufficient new information about the environment, it is flagged as a keyframe and its corresponding points and lines are triangulated and added to the map. To discard possible outliers, features seen from less than three viewpoints or in less than 25% of the frames from which they were expected to be seen are discarded too (culling). Point and line features position in the map are optimized with a local BA. Note in Fig. 1 that we do not use lines for loop closing. Matching lines across the whole map is too computationally expensive. Hence, only point features are used for loop detection.

We next describe the line parameterization and error function as well as their integration within the main building blocks of the SLAM pipeline, namely bundle adjustment and global re-localization (Table 1).

In order to extend the ORB-SLAM [2] to lines, we need a proper definition of the reprojection error and line parameterization. Following [8], let $\mathbf{P}, \mathbf{Q} \in \mathbb{R}^3$ be the 3D endpoints of a line, $\mathbf{p}_\mathrm{d}, \mathbf{q}_\mathrm{d} \in \mathbb{R}^2$ their 2D detections in the image plane, and

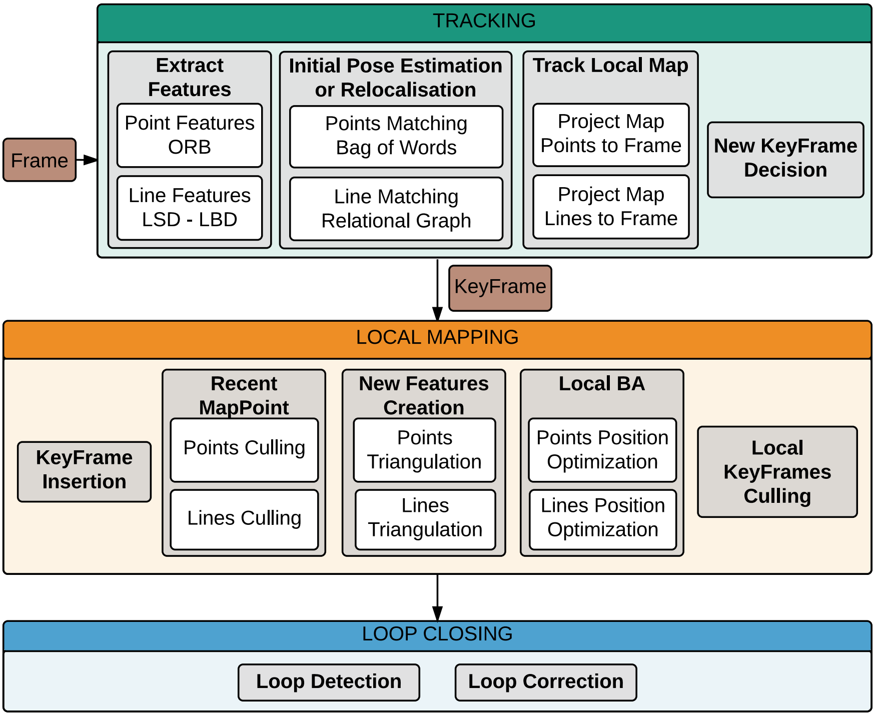

Fig. 1 PL-SLAM pipeline, an extension of the ORB-SLAM [2] pipeline. The system is composed by three main threads: *Tracking*, *Local Mapping* and *Loop Closing*. The *Tracking* thread estimates the camera position and decides when to add new keyframes. Then, *Local Mapping* adds the new keyframe information into the map and optimizes it with BA. The *Loop Closing* thread is constantly checking for loops and correcting them

$\mathbf{p}_{\mathrm{d}}^{\mathrm{h}}, \mathbf{q}_{\mathrm{d}}^{\mathrm{h}} \in \mathbb{R}^3$ theirs corresponding homogeneous coordinates. From the latter we can obtain the normalized line coefficients as:

$$\mathbf{l} = \frac{\mathbf{p}_{\mathrm{d}}^{\mathrm{h}} \times \mathbf{q}_{\mathrm{d}}^{\mathrm{h}}}{\left|\mathbf{p}_{\mathrm{d}}^{\mathrm{h}} \times \mathbf{q}_{\mathrm{d}}^{\mathrm{h}}\right|}. \tag{1}$$

The *line reprojection error* $\mathrm{E}_{\mathrm{line}}$ is then defined as the sum of point-to-line distances E_{pl} between the projected line segment endpoints, and the detected line in the image plane (see Fig. 2-right). That is:

$$\mathrm{E}_{\mathrm{line}}(\mathbf{P}, \mathbf{Q}, \mathbf{l}, \boldsymbol{\theta}, \mathbf{K}) = \mathrm{E}_{\mathrm{pl}}^2(\mathbf{P}, \mathbf{l}, \boldsymbol{\theta}, \mathbf{K}) + \mathrm{E}_{\mathrm{pl}}^2(\mathbf{Q}, \mathbf{l}, \boldsymbol{\theta}, \mathbf{K}), \tag{2}$$

with:

$$\mathrm{E}_{\mathrm{pl}}(\mathbf{P}, \mathbf{l}, \boldsymbol{\theta}, \mathbf{K}) = \mathbf{l}^{\top}\pi(\mathbf{P}, \boldsymbol{\theta}, \mathbf{K}), \tag{3}$$

Table 1 Symbols used in the development of the PL-SLAM method

Definition	Symbol
2D detections of lines endpoints $\left[\mathbf{P}\ \mathbf{Q}\right]$	$\mathbf{p}_d, \mathbf{q}_d \in \mathbb{R}^2$
Homogeneous 2D coordinates of endpoints detections $\mathbf{p}_d, \mathbf{q}_d \in \mathbb{R}^2$	$\mathbf{p}_\mathrm{d}^\mathrm{h}, \mathbf{q}_\mathrm{d}^\mathrm{h} \in \mathbb{R}^3$
Detected line coefficients	$\mathbf{l}$
Line reprojection error	$\mathbf{E}_\mathrm{line}$
Camera calibration matrix (internal parameters)	$\mathbf{K}$
ith camera parameters, $\boldsymbol{\theta}_i = \mathbf{R}_i, \mathbf{t}_i$	$\boldsymbol{\theta}_i$
Projection of $\mathbf{P}$ into the image plane of camera $(\boldsymbol{\theta}, \mathbf{K})$	$\pi(\mathbf{P}, \boldsymbol{\theta}, \mathbf{K})$
Detected line reprojection error	$\mathrm{E}_\mathrm{line,d}$
Detected point to line error	$\mathrm{E}_\mathrm{pl,d}$
Projection of the jth point $\mathbf{X}_j \in \mathbb{R}^3$ into the ith keyframe	$\widetilde{\mathbf{x}}_{i,j}$
Estimation error for $\mathbf{X}_j$ in ith keyframe	$\mathrm{e}_\mathrm{i,j}$
Cost function to minimize during bundle adjustment	C
Hubert robust cost function	ρ
Covariance matrices for the detection scales	$\boldsymbol{\Omega}_{i,j}, \boldsymbol{\Omega}'_{i,j}, \boldsymbol{\Omega}''_{i,j}$

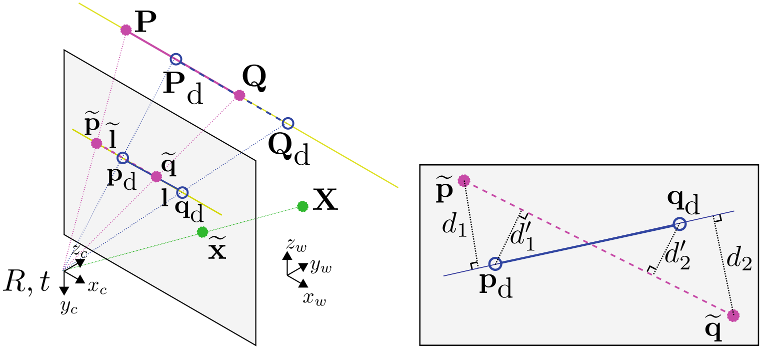

Fig. 2 Left: Notation. Let $\mathbf{P}, \mathbf{Q} \in \mathbb{R}^3$ be the 3D endpoints of a 3D line, $\widetilde{\mathbf{p}}, \widetilde{\mathbf{q}} \in \mathbb{R}^2$ their projected 2D endpoints to the image plane and $\widetilde{\mathbf{l}}$ the projected line coefficients. $\mathbf{p}_\mathrm{d}, \mathbf{q}_\mathrm{d} \in \mathbb{R}^2$ the 2D endpoints of a detected line, $\mathbf{P}_\mathrm{d}, \mathbf{Q}_\mathrm{d} \in \mathbb{R}^3$ their real 3D endpoints, and $\mathbf{l}$ the detected line coefficients. $\mathbf{X} \in \mathbb{R}^3$ is a 3D point and $\widetilde{\mathbf{x}} \in \mathbb{R}^2$ its corresponding 2D projection. Right: Line-based reprojection error. d_1 and d_2 represent the *line reprojection error*, and d'_1 and d'_2 the *detected line reprojection error* between a detected 2D line (blue solid) and the corresponding projected 3D line (pink dashed)

where $\mathbf{l}$ are the detected line coefficients, $\pi(\mathbf{P}, \boldsymbol{\theta}, \mathbf{K})$ represents the projection of the endpoint $\mathbf{P}$ onto the image plane, given the internal camera calibration matrix $\mathbf{K}$, and the camera parameters $\boldsymbol{\theta} = \{\mathbf{R}, \mathbf{t}\}$ that includes the rotation and translation parameters, respectively.

Note that in practice, due to real conditions such as line occlusions or misdetections, the image detected endpoints $\mathbf{p}_\mathrm{d}$ and $\mathbf{q}_\mathrm{d}$ will not match the projections of the endpoints $\mathbf{P}$ and $\mathbf{Q}$ (see Fig. 2-left). Therefore, we define the *detected line* *reprojection error* as:

$$\mathrm{E}_{\mathrm{line,d}}(\mathbf{p}_{\mathrm{d}}, \mathbf{q}_{\mathrm{d}}, \mathbf{l}) = \mathrm{E}^2_{\mathrm{pl,d}}(\mathbf{p}_{\mathrm{d}}, \mathbf{l}) + \mathrm{E}^2_{\mathrm{pl,d}}(\mathbf{q}_{\mathrm{d}}, \mathbf{l}), \tag{4}$$

where $\mathbf{l}$ is the projected 3D line coefficients and the detected point-to-line error is $\mathrm{E}_{\mathrm{pl,d}}(\mathbf{p}_{\mathrm{d}}, \mathbf{l}) = \mathbf{l}^\top \mathbf{p}_{\mathrm{d}}$.

Based on the methodology proposed in [8], a recursion over the detected reprojection line error will be applied in order to optimize the pose parameters $\boldsymbol{\theta}$ while approximating $\mathrm{E}_{\mathrm{line,d}}$ to the line error $\mathrm{E}_{\mathrm{line}}$ defined on Eq. (2).

The camera pose parameters $\boldsymbol{\theta} = \{\mathbf{R}, \mathbf{t}\}$ are optimized at each frame with a *Bundle Adjustment with Points and Lines* strategy that constrains $\boldsymbol{\theta}$ to lie in the SE(3) group. For doing this, we build upon the framework of the ORB-SLAM [2] but besides feature point observations, we include the lines as defined in the previous subsection. We next define the specific cost function we propose to be optimized by the BA that combines the two types of geometric entities.

Let $\mathbf{X}_j \in \mathbb{R}^3$ be the generic jth point of the map. For the ith keyframe, this point can be projected onto the image plane as:

$$\widetilde{\mathbf{x}}_{i,j} = \pi(\mathbf{X}_j, \boldsymbol{\theta}_i, \mathbf{K}), \tag{5}$$

where $\boldsymbol{\theta}_i = \{\mathbf{R}_i, \mathbf{t}_i\}$ denotes the specific pose of the ith keyframe. Given an observation $\mathbf{x}_{i,j}$ of this point, we define following 3D error:

$$\mathbf{e}_{i,j} = \mathbf{x}_{i,j} - \widetilde{\mathbf{x}}_{i,j} \ . \tag{6}$$

Similarly, let us denote by $\mathbf{P}_j$ and $\mathbf{Q}_j$ the endpoints of the jth map line segment. The corresponding image projections (expressed in homogeneous coordinates) onto the same keyframe can be written as:

$$\widetilde{\mathbf{p}}^{\mathrm{h}}_{i,j} = \pi(\mathbf{P}_j, \boldsymbol{\theta}_i, \mathbf{K}), \tag{7}$$

$$\widetilde{\mathbf{q}}^{\mathrm{h}}_{i,j} = \pi(\mathbf{Q}_j, \boldsymbol{\theta}_i, \mathbf{K}) \ . \tag{8}$$

Then, given the image observations $\mathbf{p}_{i,j}$ and $\mathbf{q}_{i,j}$ of the jth line endpoints, we use Eq. (1) to estimate the coefficients of the observed line $\widetilde{\mathbf{l}}_{i,j}$. We define the following error vectors for the line:

$$\mathbf{e}'_{i,j} = (\widetilde{\mathbf{l}}_{i,j})^\top (\mathbf{K}^{-1} \mathbf{p}^{\mathrm{h}}_{i,j}), \tag{9}$$

$$\mathbf{e}''_{i,j} = (\widetilde{\mathbf{l}}_{i,j})^\top (\mathbf{K}^{-1} \mathbf{q}^{\mathrm{h}}_{i,j}). \tag{10}$$

The errors (9, 10) are in fact instances of the point-to-line error (3). As explained in [8] they are not constant w.r.t. shift of the endpoints $\mathbf{P}_j$, $\mathbf{Q}_j$ along the corresponding 3D line, which serves as implicit regularization allowing us to use such a non-minimal line parametrization in the BA.

Observe that representing lines using their endpoints we obtain comparable error representations for points and lines. We can therefore build a unified cost function that integrates each of the error terms as:

$$C = \sum_{i,j} \rho \left(\mathbf{e}_{i,j}^{\top} \boldsymbol{\Omega}_{i,j}^{-1} \mathbf{e}_{i,j} + \mathbf{e}_{i,j}^{'}{}^{\top} \boldsymbol{\Omega}_{i,j}'{}^{-1} \mathbf{e}_{i,j}' + \mathbf{e}_{i,j}^{''}{}^{\top} \boldsymbol{\Omega}_{i,j}''{}^{-1} \mathbf{e}_{i,j}'' \right) \tag{11}$$

where ρ is the Huber robust cost function and $\boldsymbol{\Omega}_{i,j}$, $\boldsymbol{\Omega}_{i,j}'$, $\boldsymbol{\Omega}_{i,j}''$ are the covariance matrices associated to the scale at which the keypoints and line endpoints were detected, respectively.

An important component of any SLAM method is the *Global Relocalization*, an approach to relocalize the camera when the tracker is lost. This is typically achieved by means of a PnP algorithm, that estimates the pose of the current (lost) frame given correspondences with 3D map points appearing in previous keyframes. On top of the PnP method, a RANSAC strategy is used to reject outliers correspondences.

In the ORB-SLAM, the specific PnP method that is used is the EPnP [9], which however, only accepts point correspondences as inputs. In order to make the approach appropriate to handle lines for relocalization, we replace the EPnP by the recently published EPnPL [8], which minimizes the *detected line reprojection error* of Eq. (4).

Furthermore, EPnPL is robust to partial line occlusion and mis-detections. This is achieved by means of a two-step procedure in which first minimizes the reprojection error of the detected lines and estimates the line endpoints $\mathbf{p}_{\text{d}}$, $\mathbf{q}_{\text{d}}$. These points, are then shifted along the line in order to match the projections $\mathbf{p}_{\text{d}}$, $\mathbf{q}_{\text{d}}$ of the 3D model endpoints $\mathbf{P}$, $\mathbf{Q}$ (see Fig. 2). Once these matches are established, the camera pose can be reliably estimated.

3 Experiments for Validation of the Method

To validate the method, Table 2 presents the the localization accuracy of PL-SLAM [5] against other state-of-the-art Visual SLAM methods, including ORB-SLAM [2], PTAM [1], LSD-SLAM [3] and RGBD-SLAM [4] using the TUM RGB-D benchmark [10]. The metric used for the comparison is the Absolute Trajectory Error (ATE), provided by the evaluation script of the benchmark. Before computing the error, all trajectories are aligned using a similarity warp except for the RGBD-SLAM [4] which is aligned by a rigid body transformation. All experiments were carried out with an Intel Core i7-4790 (4 cores @3.6 GHz), 8Gb RAM and ROS Hydro [11]. Due to the randomness of the some stages of the pipeline, e.g., initialization, position optimization or global relocalization, all experiments were run five times and we report the median of all executions.

Note that PL-SLAM consistently improves the trajectory accuracy of ORB-SLAM in all sequences (see Fig. 3 for a comparison example). Indeed, it yields the best result in all but two sequences, for which PTAM performs slightly better. Neverthe-

Table 2 Localization accuracy in the TUM RGB-D Benchmark [10]

Absolute keyframe trajectory RMSE [cm]						
TUM RGB-D Sequence	PL-SLAM Classic Init	PL-SLAM Line Init	ORB-SLAM	PTAM†	LSD-SLAM†	RGBD-SLAM†
f1_xyz	1.21	1.46	1.38	**1.15**	9.00	1.34
f2_xyz	0.43	1.49	0.54	**0.2**	2.15	2.61
f1_floor	7.59	9.42	8.71	–	38.07	**3.51**
f2_360_kidnap	3.92	60.11	4.99	**2.63**	–	393.3
f3_long_office	**1.97**	5.33	4.05	–	38.53	–
f3_nstr_tex_far	ambiguity detected	37.60	ambiguity detected	34.74	**18.31**	–
f3_nstr_tex_near	2.06	**1.58**	2.88	2.74	7.54	–
f3_str_tex_far	**0.89**	1.25	0.98	0.93	7.95	–
f3_str_tex_near	1.25	7.47	1.5451	**1.04**	–	–
f2_desk_person	**1.99**	6.34	5.95	–	31.73	6.97
f3_sit_xyz	**0.066**	9.03	0.08	0.83	7.73	–
f3_sit_halfsph	**1.31**	9.05	1.48	–	5.87	–
f3_walk_xyz	**1.54**	ambiguity detected	1.64	–	12.44	–
f3_walk_halfsph	**1.60**	ambiguity detected	2.09	–	–	–

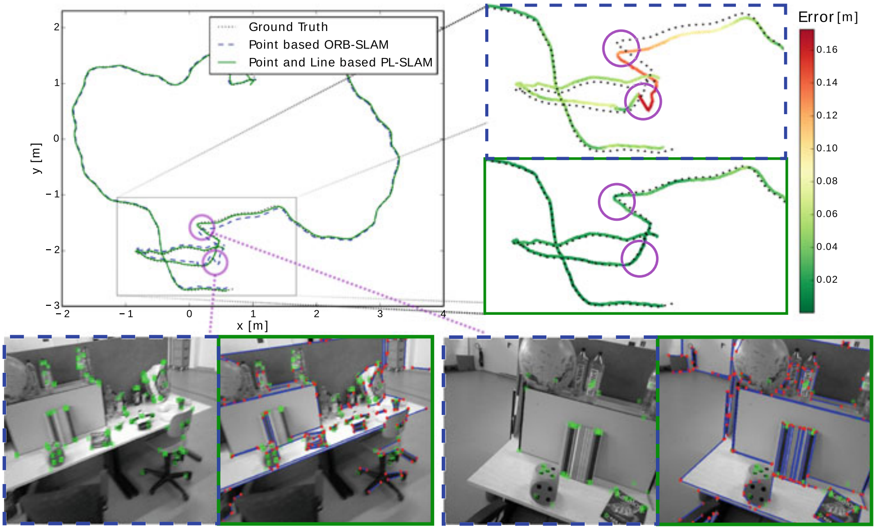

Fig. 3 ORB-SLAM [2] vs PL-SLAM [5]. Comparison of the trajectories obtained using the state-of-the-art point-based method ORB-SLAM and the proposed PL-SLAM, in a TUM RGB-D sequence. The black dotted line shows the ground truth, the blue dashed line is the trajectory obtained with ORB-SLAM, and the green solid line is the trajectory obtained with PL-SLAM. Note how the use of lines consistently improves the accuracy of the estimated trajectory

less, PTAM turned not to be so reliable, as in 5 out of all 12 sequences it lost track. LSD-SLAM and RGBD-SLAM also lost track in 3 and 7 sequences, respectively. PL-SLAM builds upon the architecture of the state-of-the-art ORB-SLAM and modifies its original pipeline to operate with line features without significantly compromising its efficiency.

Real-life experiments where done in the Karting experimental site where most of the methods where tested in real-life conditions in the AEROARMS project. The method was implemented using a monochromatic camera located at the bottom of the aerial robot. Figure 4 shows the points and lines detected using the PL-SLAM, and the trajectory followed by the aerial robot before (BSC) and after (ASC) scale convergence. The scale is computed in the beginning of the fly to obtain the real scale and once the estimated map scale has converged, we compare the estimated distance of the robot from the ground against the real one. To obtain the real height, a laser pointer was installed in the bottom of the robot pointing to the ground and corrected with the relative angles of the robot.

The Karting presents a challenging scenario as it contains: (1) visual features at a long distance and (2) apparent lines. Long distant features are prone to error as small movements of the robot correspond to large displacements of the observed point. Small errors in the estimation of this points would cause large penalties in optimization of the 3D map. To overcome these penalties, long distant points were removed by the points culling filter. Similarly, the apparent lines (not real lines) of

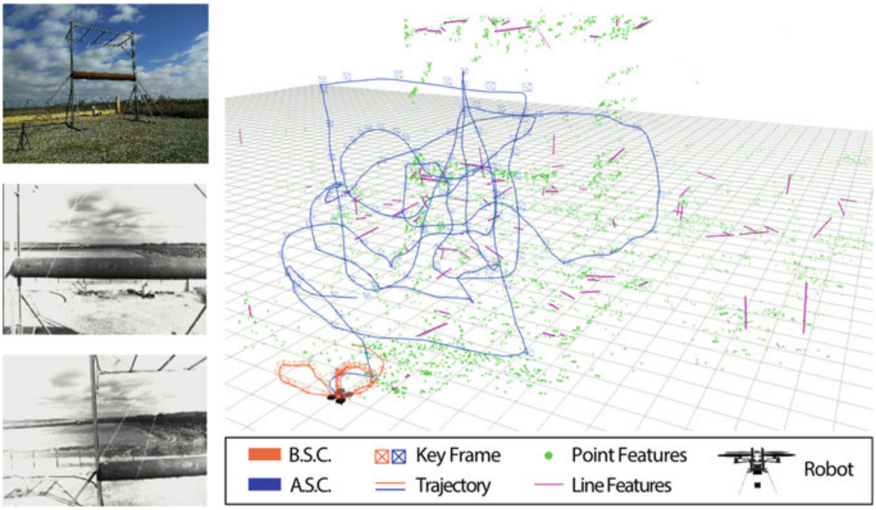

Fig. 4 Real-life experiments in the Karting experimental site. The figure shows the pipe in the Karting experimental site and the trajectory obtained after (blue) and before (red) scale convergence. The figure also shows the points and line detected with the method

the scene, e.g. the pipe contours, were also filtered with the same mechanism to prevent the Bundle Adjustment from trying to fit non-real scene landmarks in the 3D map. The experiment concluded that the method can robustly operate on challenging scenarios with noisy landmarks.

4 Conclusions

In this chapter we have presented PL-SLAM [5], an approach to visual SLAM that allows to simultaneously process points and lines and tackle situations where point-only based methods are prone to fail, like poorly textured scenes or motion blurred images where feature points are vanished out. We have also developed a novel line-based map initialization approach, which estimates camera pose and 3D map from 5 line correspondences in three consecutive images. This approach holds on the assumption of constant and small inter-frame rotation in these three images. We have shown that this indeed is a good approximation for many situations and showed consistent improvement w.r.t. current competing methods results when evaluating the full pipeline on the TUM RGB-D benchmark. To the best of our knowledge, the continuous contours based relative localization approach has not been studied before, even though it provides a very natural measure of alignment error without the need of correspondences. The experiments concluded that the method is remarkably robust against image noise, and that it outperforms state-of-the-art methods for point-based contour alignment. The method was also tested in the Karting experimental site were most of the AEROARMS methods were tested. The method can run in real-time and in a low cost hardware.

References

1. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: ISMAR, pp. 225–234. IEEE, New York (2007)
2. Mur-Artal, R., Montiel, J.M.M., Tardós, J.D.: ORB-SLAM: a versatile and accurate monocular slam system. TRO **31**(5), 1147–1163 (2015)
3. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM. In: ECCV, pp. 834–849. Springer, Berlin (2014)
4. Endres, F., Hess, J., Sturm, J., Cremers, D., Burgard, W.: 3-D mapping with an RGB-D camera. TRO **30**(1), 177–187 (2014)
5. Pumarola, A., Vakhitov, A., Agudo, A., Sanfeliu, A., Moreno-Noguer, F.: PL-SLAM: Real-time monocular visual SLAM with points and lines. In: International Conference in Robotics and Automation (2017)
6. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to sift or surf. In: 2011 IEEE international conference on Computer Vision (ICCV), pp. 2564–2571. IEEE, New York (2011)
7. von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: LSD: a line segment detector. IPOL **2**, 35–55 (2012)
8. Vakhitov, A., Funke, J., Moreno-Noguer, F.: Accurate and linear time pose estimation from points and lines. In: ECCV (2016)
9. Moreno-Noguer, F., Lepetit, V., Fua, P.: Accurate non-iterative O(n) solution to the pnp problem. In: ICCV, pp. 1–8. IEEE, New York (2007)
10. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGB-D SLAM systems. In: IROS (2012)
11. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y:. ROS: an open-source robot operating system. In: ICRAW, vol. 3, pp. 5. Kobe, Japan (2009)