Stereo Parallel Tracking and Mapping for robot localization

Taihú Pire[1] Thomas Fischer[1] Javier Civera[2] Pablo De Cristóforis[1] and Julio Jacobo Berlles[1]

***Abstract*— This paper describes a visual SLAM system based on stereo cameras and focused on real-time localization for mobile robots. To achieve this, it heavily exploits the parallel nature of the SLAM problem, separating the time-constrained pose estimation from less pressing matters such as map building and refinement tasks. On the other hand, the stereo setting allows to reconstruct a metric 3D map for each frame of stereo images, improving the accuracy of the mapping process with respect to monocular SLAM and avoiding the well-known bootstrapping problem. Also, the real scale of the environment is an essential feature for robots which have to interact with their surrounding workspace. A series of experiments, on-line on a robot as well as off-line with public datasets, are performed to validate the accuracy and real-time performance of the developed method.**

I. Introduction

Simultaneous Localization and Mapping (SLAM), is a key skill for any truly autonomous mobile system. The goal of SLAM is the joint estimation of both the robot's pose and a model of its surrounding environment (i.e. the map). This information is essential for the robot to safely interact within its workspace. A vast number of SLAM implementations have appeared over the last decade. Most of the early works use a laser rangefinder as the main sensor [1]. More recently, visual sensors have become the dominant choice, either passive [2] or active [3]. Presently, affordable, small and light, now-a-day cameras can provide high resolution data in real-time and virtually unlimited measurement ranges in contrast to laser rangefinders. Moreover, cameras are passive sensors and therefore do not interfere with each other when deployed in the same environment, and unlike *Structured light range sensors* (SLRS), they can be used in both indoor and outdoor environments. These characteristics make cameras an ideal choice for multi-purpose mobile robotic platforms.

In this work we present a SLAM system using a stereo camera as the main sensor for robot localization. Stereo cameras allow to detect the same visual point-landmarks on a pair of synchronized images, which can be used to recover their depth information and incrementally build a sparse point-cloud representation of the environment. They also allow to compute the real scale of the map, which is essential for many mobile robot applications. As the robot moves trough the environment it is possible to track the visual landmarks frame after frame, and estimate the current pose within it.

[1] Taihú Pire, Thomas Fischer, Pablo De Cristóforis and Julio Jacobo Berlles are with the Computer Science Department, University of Buenos Aires, Argentina. {tpire, tfischer, pdecris, jacobo}@dc.uba.ar

[2] Javier Civera is with the Robotics, Perception and Real-Time Group, University of Zaragoza, Spain. {jcivera}@unizar.es

Following the approach of Parallel Tracking and Mapping (PTAM) [4], our stereo SLAM system, henceforth referred to as S-PTAM, also divides the problem into two main parallel tasks: camera tracking and map optimization. These tasks are executed in two different threads, only sharing the map between them. The tracking thread matches features, creates new points and estimates the camera pose for every new frame, and the mapping thread iteratively refines the nearby point-landmarks that compose the map.

S-PTAM was developed from scratch to achieve a Stereo SLAM system that overcomes the limitations of PTAM when it comes to robot navigation. The main characteristics of the system can be described as follows:

- The parallel nature of the SLAM problem is exploited achieving real-time performance, whilst minimizing inter-thread dependency.
- A stereo camera is used, avoiding the monocular bootstrapping problem [5] and allowing to compute the metric scale of the mapped environment without any prior information.
- A maintenance process executed in a independent thread iteratively runs map refinement (Bundle Adjustment) operations in a local covisible area, which improves global consistency.
- Although the method may work exclusively on camera sensors, if odometry is available, it can be used for early pose prediction and more effective feature matching.
- Stereo constraints are enforced on the pose- and map-refinement algorithms, improving robustness.
- Binary features are used to describe visual point-landmarks, thus reducing storage requirements and increasing matching speed.

The implementation of S-PTAM is open source and publically available[1]. It is built upon the ROS (Robot Operating System) framework to ease distribution and integration.

II. Related Work

Early stereo works in SLAM were based on probabilistic approaches like Extended Kalman Filters [6], and Particle Filters [7]. However, in recent work [8], filter-based approaches were compared to keyframe-based methods which used global optimization techniques like Bundle Adjustment, and the latter ones were found to achieve the best balance between precision and computational cost.

One of the most actively developed keyframe-based approaches is PTAM [4]. This method has been widely used to solve the pose estimation problem in unknown environments

[1] http://github.com/lrse/sptam

in several scenarios [9], [10]. While PTAM was originally presented for monocular systems and Virtual Reality applications, stereo variants have since been developed. An early example is FrameSLAM [11], which introduces the idea of selecting only a subset of frames (*skeleton*) for mapping, thus reducing the complexity of map related operations such as *loop closure*. However, features are used only for local visual odometry and discarded afterwards, losing any chance of performing navigation.

Large-scale environments are tackled in more recent approaches [12], [13] by proposing strategies based on variations of the relative Bundle Adjustment [14] method, which allows for constant-time estimation in terms of map size. Other recent stereo SLAM works focus on improving the tracking aspect instead. One example [15] consists in an improved strategy for selection of points, based on the optical flow, to be used during tracking which reduces the uncertainty of the camera motion. Finally, [16] presents a modified version of PTAM, which includes depth information, and is similar to the approach proposed in this work. However, just as in the original version of PTAM, the mapper thread creates new map points after performing a global Bundle Adjustment on each incoming keyframe. If the map grows large enough, this causes the tracking to loose any reference to the map, thus making it unsuitable for large scale environments.

III. Method Overview

S-PTAM starts with the canonical pose that defines the world reference frame for the system and an empty map that is initialized by triangulating matching features in the first pair of stereo images. From then on, the tracking thread estimates the current robot pose for each new incoming stereo frame, minimizing the re-projection error between features on the images and their corresponding map points. Occasionally, some frames are strategically selected (*keyframes*), which incorporate new points triangulated from stereo features, thus augmenting the map. They also add new spatial constraints to the map points, which potentially improves the accuracy of the model. Simultaneously, the map refinement thread is constantly trying to minimize the re-projection error by adjusting all points and keyframes in a bundle. The map is the only shared resource between both threads. By increasing the accuracy of the map, future localization will also be improved. The main tasks of the method are detailed below.

A. Tracking

Here we describe the sequential four steps taken by the tracking process on arrival of each new pair of stereo images.

1) Feature extraction: Our visual SLAM method relies on local image features. These features are matched to existing map-points and tracked for each frame, or failing that, used to create new ones by matching stereo correspondences. As features should be detected and matched from different viewpoints, descriptors robust to pose changes are required. Computational cost is also an important factor in the selection of a particular extractor/descriptor combination.

For the present work, the BRIEF [17] descriptor was selected for its performance. The lack of rotational invariance was not considered an issue based on the assumption that the mobile (ground) robot moves on a locally planar terrain.

The Shi-Tomasi [18] detection algorithm was selected for extraction. Although the method performs slower than other state-of-the-art detection algorithms such as FAST [19], it enforces a "good" spatial distribution of the selected points over the image by means of non-maximum suppression, which significantly increases the robustness of the pose refinement during tracking.

2) Matching: Next, we want to define a set of spatial constraints that relate the existing map points $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$ to the newly extracted image features. For each point inside the viewing frustum of the predicted stereo cameras, matching is performed on the descriptors found in a close neighborhood of the points projection on the image. A prediction of the current camera pose is necessary in order to perform said projection. In our case, dead-reckoning based on wheel odometry is used, since it is available in most ground based robotic vehicles. If it were not, a *decaying velocity model* or even the camera pose computed at the previous frame can be used instead.

3) Pose refinement: The preliminary pose prediction $\boldsymbol{\mu} = (t_x, t_y, t_z, \theta_r, \theta_p, \theta_y)$ of the current stereo camera system needs to be adjusted. This is done by using the well known Levenberg Marquardt algorithm, which iteratively minimizes the re-projection errors through the non-linear least squares equation

$$\boldsymbol{\mu}' = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \sum_k \left(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\right)^2 \tag{1}$$

where $\boldsymbol{\mu}'$ is the adjusted camera pose, and $\hat{\boldsymbol{x}}_k$ and $\boldsymbol{x}_k$ are the projection and *measurement*, respectively, of the map point $\boldsymbol{X}_k$ in one or both cameras. If $\boldsymbol{X}_k$ is matched to features (u_L, v_L) in the left camera and (u_R, v_R) in the right camera, then we call it a stereo match, and $\boldsymbol{x}_k = (u_L, v_L, u_R)$. Since the images are rectified, this enforces the constraint $v_L = v_R$. On the other hand, if it matches a feature in a single camera, $\boldsymbol{x}_k = (u, v)$. Analogous coordinates are also used for the projections $\hat{\boldsymbol{x}}_k$. Let $(\hat{u}_L, \hat{v}_L) = \boldsymbol{P}_L(\boldsymbol{\mu})\boldsymbol{X}_k$ and $(\hat{u}_R, \hat{v}_R) = \boldsymbol{P}_R(\boldsymbol{\mu})\boldsymbol{X}_k$, where $\boldsymbol{P}_L(\boldsymbol{\mu})$ and $\boldsymbol{P}_R(\boldsymbol{\mu})$ are the left and right projection matrices for the left and right camera respectively. If $\boldsymbol{x}_k$ is a stereo measurement then $\hat{\boldsymbol{x}}_k = (\hat{u}_L, \hat{v}_L, \hat{u}_R)$. If it is a left or right monocular measurement then $\hat{\boldsymbol{x}}_k = (\hat{u}_L, \hat{v}_L)$ or $\hat{\boldsymbol{x}}_k = (\hat{u}_R, \hat{v}_R)$ respectively.

The effect of outliers on the refined pose can be reduced by modifying the error function in (1) using a robust M-estimator (in our case the Huber weight function) and a weighted least squares approach, as proposed in [4]. The Levenberg Marquardt algorithm requires the symbolic Jacobian of this error function with respect to $\boldsymbol{\mu}$, but as in [4], the much simpler Jacobian of the residual function $\boldsymbol{r}_k = \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k$ may be used instead. The derivation of $\boldsymbol{J}_k = \frac{\partial \boldsymbol{r}_k}{\partial \boldsymbol{\mu}}$ is presented in appendix I.

4) Keyframes selection: Finally, the current frame is selected to be a *keyframe* when the number of tracked points is less than 90% of the points tracked in the last keyframe. If so, the remaining unmatched features from the stereo pair are triangulated to create new map points. Since the camera images are stereo-rectified, potential matches for a feature are found using a row-based search on the other image, greatly lowering the computational cost with respect to a brute force approach.

B. Mapping

The multiview and stereo constraints are used to adjust the map of sparse salient point features and keyframes. Our approach follows the monocular system presented in [4] and extends it by enforcing the stereo constraints.

Bundle Adjustment refers to the simultaneous refinement of a set of camera poses (keyframes) and 3D points (the map) reducing the re-projection error of every point in every image. It can be seen as a particular case of least squares estimation, which again can be solved with the approach proposed in section III-A.3.

Given a set of m camera poses $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m\}$, a set of n 3D points $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$ and set of measurements $\boldsymbol{S} = \{\boldsymbol{x}_{ij}\}$, where each $\boldsymbol{x}_{ij}$ is the position of the detected feature for point $\boldsymbol{X}_i$ on the image plane of the j-th keyframe, the method can be seen as minimizing the equation

$$\left\{\boldsymbol{\mu}'_j, \boldsymbol{X}'_i\right\} = \underset{\left\{\boldsymbol{\mu}_j, \boldsymbol{X}_i\right\}}{\operatorname{argmin}} \sum_{\boldsymbol{x}_{ij} \in \boldsymbol{S}} \left(\boldsymbol{x}_{ij} - \hat{\boldsymbol{x}}_{ij}\right)^2 .$$

It is important to note that the camera pose $\boldsymbol{\mu}_1$ is considered fixed during Bundle Adjustment refinement. This is because the first keyframe is considered to have zero uncertainty, as it defines the world reference frame for the method.

Given the sparse block nature of the Jacobian, a *sparse* Levenberg-Marquardt implementation is used, which heavily exploits this kind of structure to improve computational cost. Each block of the Jacobian can decomposed as

$$\boldsymbol{J}_{ij} = \left[\begin{array}{c|c} \frac{\partial \boldsymbol{r}_{ij}}{\partial \boldsymbol{\mu}_j} & \frac{\partial \boldsymbol{r}_{ij}}{\partial \boldsymbol{X}_i} \end{array} \right] .$$

for $\boldsymbol{r}_{ij} = \boldsymbol{x}_{ij} - \hat{\boldsymbol{x}}_{ij}$. The symbolic derivation of each term of $\boldsymbol{J}_{ij}$ for left and right cameras is presented in appendices I and II respectively.

IV. Implementation Details

In this section we detail some relevant implementation decisions that were taken to allow the system to execute in real-time on a mobile robotic platform, minimizing the impact on robot pose estimation accuracy.

Since keypoint detection and descriptor extraction is heavily time consuming, this process is separated from the tracking on another thread. A pipeline is used under the assumption that the tracking process is considerably faster than the image analysis. This allows to process an incoming stereo frame while still tracking the previous one. Feature computation for each image of the stereo pair is also split into two parallel threads.

Another bottleneck of the tracking phase is matching map points to recently extracted features. Since the map size scales linearly with the traveled distance, checking all points becomes infeasible on the long run. Because of this, map points are initially filtered by camera frustum culling. Points whose descriptors were detected from a radically different angle of view (e.g. 45 degrees) are also discarded.

Then, the remaining map points are projected onto the image plane to check for matches against the detected features. To speed up this process, detected features are grouped by spatial hashing into grid cells. The matching of a map point is then restricted to the features inside a neighborhood around its projection.

If a frame is selected to act as a keyframe, it is queued into the map refinement thread to be processed as soon as possible. Unlike PTAM, where new map points are created once the keyframe is processed by the mapping thread, which may not be immediate depending on the level of congestion, S-PTAM immediately creates and incorporates the new points after the tracking step and before queuing, to avoid the loss of potential map matches on upcoming frames.

Global map optimization through Bundle Adjustment, as proposed in [4], becomes prohibitive for large scale environments. Consequently we perform only local optimization. The Local Bundle Adjustment (LBA) only refines a fixed number of queued keyframes, along a set of already refined nearby keyframes, and the corresponding subset of visible map points. Unlike PTAM, which runs LBA once for each single keyframe, S-PTAM grabs up to ten queued keyframes to avoid starvation. Experiments show that the queue size never exceeds four keyframes.

The $\mathbf{g^2o}$ (General Graph Optimization) library [20] was used to perform Levenberg Marquardt minimization during both tracking and Bundle Adjustment.

The source code was built upon the ROS framework, in order to promote its usage by the robotics research community.

V. Experiments

To assess the accuracy, robustness and computational cost of the S-PTAM method, various sequences from public datasets such as MIT Stata Center Data Set [21] and the Karlsruhe KITTI dataset [22] were used, since both of them provide Ground-Truth measurements. They cover indoor robotic as well as outdoor large driving scenarios respectively. The latter, although not strictly robot localization, provides a standard benchmarking framework which helps to compare the performance of our method to other state-of-the-art stereo vision based SLAM systems. Furthermore, the S-PTAM method was also evaluated running on-line on a wheeled robot in indoor environments. For all these experiments, a standard laptop with an Intel Core i7 @ 2.20 GHz processor and 8 GB RAM was used. Below we present results for the mentioned sequences.

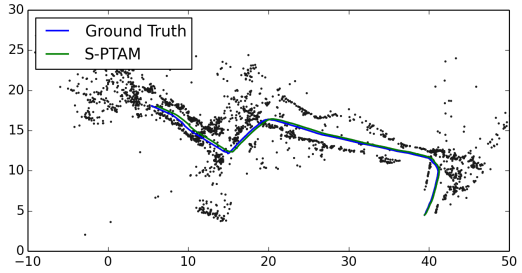

Fig. 1: Estimated trajectory performed by S-PTAM on MIT sequence 2012-01-27-07-37-01 compared to the ground truth as well as the generated map. Distances are given in meters.

A. MIT Stata Center dataset

This dataset is a collection of several indoor sequences taken with a PR2 robot inside the MIT Stata Center building. In particular we show the results for sequence 2012-01-27-07-37-01, which includes some unconnected portions of ground-truth data. Although S-PTAM was tested on the whole sequence, we only show the results over one of these sub-sequences where ground truth is available. During the nearly 50 m trajectory followed by the robot the maximum computed error was 0.6 m.

Fig. 2 and Fig. 3 illustrate the relative error (green), which characterizes local consistency. It is computed by comparing differences between subsequent poses in the estimated sequence as well as the ground truth. This allows to assess the tracking performance independently of the incremental errors associated with incremental methods. As expected, the relative tracking error does not increase with traveled distance and never exceeds 20 cm. The peaks in localization errors indicates zones where it is difficult to perform tracking. This is caused, for example, by lack of features or confusing environments, such as those with heavy reflections.

In Fig. 3 at 14 seconds a peak shows an error of 80 degrees, after which the method immediately recovers by tracking the previously constructed map. The peak is related to an area which presents a high-speed turn with lack of decent features causing difficulties for tracking. Nonetheless, S-PTAM shows to be robust against this kind of outliers.

B. KITTI benchmark suite

The KITTI benchmark suite is a benchmarking framework which provides multiple training and test sequences of a car travelling through different urban environments. The stereo camera mounted on the front has a ∼60 cm baseline and a resolution of 1344×391 pixels at a frame rate of 10 Hz.

The results obtained by S-PTAM were commited to the KITTI Benchmark website where its accuracy can be compared to other state-of-the-art methods. At the moment of this publication, S-PTAM was tenth on the ranking, and on the seventh place among stereo-vision approaches.

Here we show results obtained using the training sequence labeled 00. In Fig. 4 we see the performed trajectories as estimated by S-PTAM and the ground truth. Fig. 5 and Fig. 6 show the achieved accuracy, where again the peaks denote

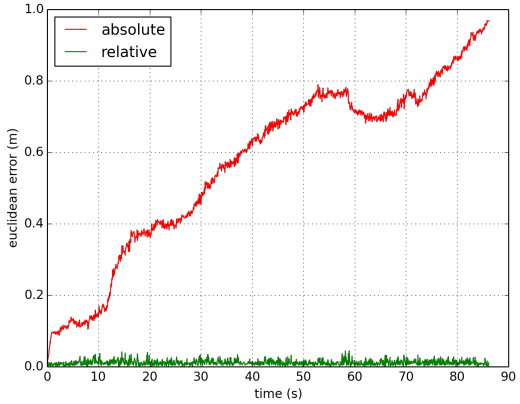

Fig. 2: Euclidean distance of the trajectory computed for the MIT sequence 1 with respect to the ground truth poses.

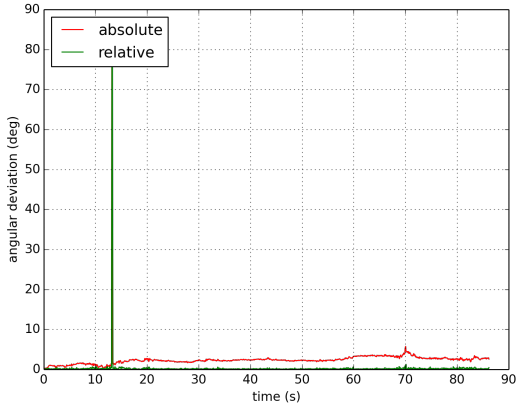

Fig. 3: Angular deviation of the trajectory computed for the MIT sequence 1 with respect to the ground truth poses.

areas with low texture or high-speed turns. Note that the relative localization error does not increase as a function of traveled distance. During the ∼4 km trajectory followed by the car, the maximum absolute localization error was of 16 m.

C. Robot experiment

To test the computational performance of the S-PTAM method running in real-time on a mobile robot, an experiment was performed using a *Pioneer-3AT* mobile robot with a stereo rig consisting of two *Pointgrey Firefly MV* cameras. In this experiment, the cameras had a resolution of 640×480 pixels and they triggered at a rate of 12 Hz, the configuration baseline was ∼14.1 cm. The robot was driven ∼320 meters around university corridors running S-PTAM method in real-time on a laptop computer. Wheel odometry was used for early pose prediction. Fig. 7 shows the robot's path along as the map performed by S-PTAM. The reduced number of map points is due to low texture walls of the corridor. Given that no ground-truth was available for this experiment, a round trip path was performed. For the return, the robot was driven backward so as to localize itself within the previously constructed map. The start and end poses were ensured to be

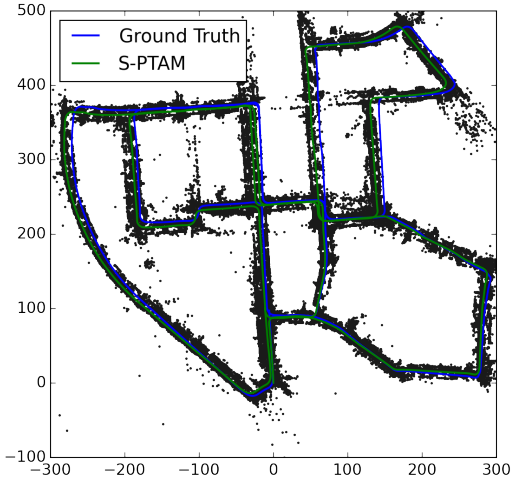

Fig. 4: Estimated trajectory performed by S-PTAM on KITTI dataset 00 compared to the ground truth as well as the generated map. Distances are given in meters.

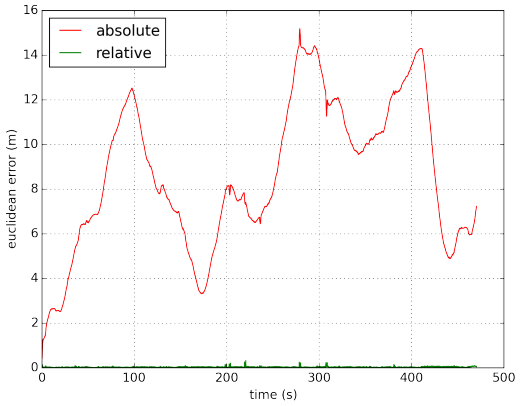

Fig. 5: Euclidean distance of trajectory in Fig. 4 with respect to the ground truth poses.

the same by manually matching markings on the floor. The difference between start and end pose given by S-PTAM was 0.0086 m in translation and 0.18 deg in rotation.

Table I shows the average temporal performance measured for the costliest subroutines of the tracking process. The Tracking thread runs at ∼17.98 Hz, which should be enough to control most autonomous mobile ground robots in real-time.

TABLE I: Tracking phase average processing time.

Tracking phase	time in ms
Feature Extraction and description	20
Get Points (inside Frustum)	1
Matching	2
AddKeyFrame	0.1
Pose Update	32.5
Total	55.6

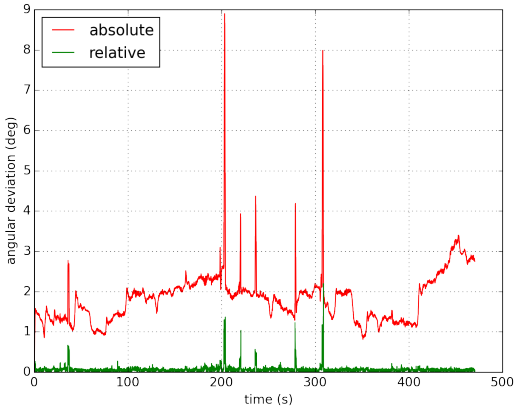

Fig. 6: Angular deviation of the trajectory in Fig. 4 with respect to the ground truth poses.

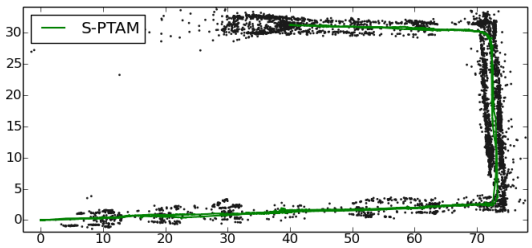

Fig. 7: Estimated trajectory performed by S-PTAM as well as the generated map. Distances are given in meters.

VI. Conclusions

A stereo SLAM system for robot localization called S-PTAM was presented. This method incrementally builds a point-based sparse map representation of the workspace, using a stereo camera, and tracks the camera pose within it. To allow S-PTAM to run in large scale environments and respond in real-time, the parallel nature of the SLAM problem is heavily exploited, separating tracking and map refinement routines, while minimizing inter-thread dependency.

The accuracy of the method was tested in public outdoor and indoor datasets, comparing results against the provided ground truth where it was available. Furthermore, experiments were performed on a real robot to test the on-line real-time performance. Results indicate that the precision of the system is comparable to state-of the art approaches for mobile robot localization.

Although stereo SLAM is a well-studied topic in the robotics community, there seems to be a lack of open-source implementations, preventing experimentation and further analysis of existing methods. In this spirit and as a part of this work we release a software package for the presented method built upon the ROS framework.

In future work, we hope to achieve autonomous robot navigation and exploration using the constructed map.

APPENDIX I
CAMERA JACOBIAN DERIVATION

We want to take the derivative of the residual function $\boldsymbol{r}_{ij} = \boldsymbol{x}_{ij} - \hat{\boldsymbol{x}}_{ij}$ with respect to the camera pose parameters $\boldsymbol{\mu}_j = (t_x, t_y, t_z, \theta_{roll}, \theta_{pitch}, \theta_{yaw})$. Since $\boldsymbol{x}_{ij}$ does not depend on $\boldsymbol{\mu}_j$ we are left with computing $\boldsymbol{J}_{ij} = -\frac{\partial \hat{\boldsymbol{x}}_{ij}}{\partial \boldsymbol{\mu}_j}$. Using the chain rule we rewrite

$$\frac{\partial \hat{\boldsymbol{x}}_{ij}}{\partial \boldsymbol{\mu}_j} = \frac{\partial \hat{\boldsymbol{x}}_{ij}}{\partial \boldsymbol{X}_{ij}} \frac{\partial \boldsymbol{X}_{ij}}{\partial \boldsymbol{\mu}_j},$$

where $\boldsymbol{X}_{ij}$ is the point $\boldsymbol{X}_i$ in the coordinate frame of the j-th camera. Using the classic pinhole camera model, the projection is computed as

$$\hat{\boldsymbol{x}}_{ij} = \begin{bmatrix} \frac{f_u x}{z} + u_0 \\ \frac{f_v y}{z} + v_0 \end{bmatrix},$$

where f_u and f_v are the focal lengths of the camera and x, y and z are the coordinates of $\boldsymbol{X}_{ij}$. It follows easily that

$$\frac{\partial \hat{\boldsymbol{x}}_{ij}}{\partial \boldsymbol{X}_{ij}} = \begin{bmatrix} \frac{f_u}{z} & 0 & \frac{-f_u x}{z^2} \\ 0 & \frac{f_v}{z} & \frac{-f_v y}{z^2} \end{bmatrix}.$$

We can compute $\boldsymbol{X}_{ij}$ as $exp(\boldsymbol{\mu}_j)\boldsymbol{X}_i$, where exp denotes the exponential map, and using the corresponding generator matrices $\boldsymbol{G}_{k=1..6}$ the remaining derivative becomes

$$\begin{aligned} \frac{\partial \boldsymbol{X}_{ij}}{\partial \boldsymbol{\mu}_j} &= \begin{bmatrix} \boldsymbol{G}_1 \boldsymbol{X}_i | & \cdots & | \boldsymbol{G}_6 \boldsymbol{X}_i \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & z & -y \\ 0 & 1 & 0 & -z & 0 & x \\ 0 & 0 & 1 & y & -x & 0 \end{bmatrix}, \end{aligned}$$

where x, y and z are the coordinates of the point $\boldsymbol{X}_i$ in the reference frame.

APPENDIX II
POINT JACOBIAN DERIVATION

Following a reasoning analogous to that in appendix I, we need to compute the derivative

$$\boldsymbol{J}_{ij} = -\frac{\partial \hat{\boldsymbol{x}}_{ij}}{\partial \boldsymbol{X}_{ij}} \frac{\partial \boldsymbol{X}_{ij}}{\partial \boldsymbol{X}_i}.$$

Since the first term was already derived in I, we focus on the second one. $\boldsymbol{X}_{ij}$ can be rewritten as $\boldsymbol{T}_j \boldsymbol{X}_i$, where $\boldsymbol{T}_j = [\boldsymbol{R}_j | \boldsymbol{t}_j]$ is the transformation that takes a point from the reference frame to the j-th camera frame. Then, it follows that

$$\frac{\partial \boldsymbol{X}_{ij}}{\partial \boldsymbol{X}_i} = \boldsymbol{R}_j.$$

REFERENCES

[1] J. Castellanos, J. Montiel, J. Neira, and J. Tardos, "The SPmap: a probabilistic framework for simultaneous localizationand map building," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 5, pp. 948–952, 1999.

[2] A. J. Davison, N. D. Molton, I. D. Reid, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. June, pp. 1052–1067, 2007.

[3] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2013, pp. 2100–2106.

[4] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[5] S. Gauglitz, C. Sweeney, J. Ventura, M. Turk, and T. Hollerer, "Live tracking and mapping from both general and rotation-only camera motion," in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2012, pp. 13–22.

[6] L. M. Paz, P. Pinies, J. Tardos, and J. Neira, "Large-Scale 6-DOF SLAM With Stereo-in-Hand," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 946–957, Oct 2008.

[7] E.-y. Wu, G.-y. Li, Z.-y. Xiang, and J.-l. Liu, "Stereo vision based SLAM using Rao-Blackwellised particle filter," *Journal of Zhejiang University SCIENCE A*, vol. 9, no. 4, pp. 500–509, 2008. [Online]. Available: http://dx.doi.org/10.1631/jzus.A071361

[8] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Editors Choice Article: Visual SLAM: Why Filter?" *Image Vision Comput.*, vol. 30, no. 2, pp. 65–77, Feb. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.imavis.2012.02.009

[9] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart, "Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 3056–3063.

[10] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2012, pp. 2815–2821.

[11] K. Konolige and M. Agrawal, "FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, Oct 2008.

[12] H. Strasdat, A. Davison, J. M. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual SLAM," in *IEEE International Conference on Computer Vision (ICCV)*, Nov 2011, pp. 2352–2359.

[13] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "RSLAM: A System for Large-Scale Mapping in Constant-Time Using Stereo," *Int. J. Comput. Vision*, vol. 94, no. 2, pp. 198–214, Sept. 2011. [Online]. Available: http://dx.doi.org/10.1007/s11263-010-0361-7

[14] G. Sibley, "Relative bundle adjustment," *Department of Engineering Science, Oxford University, Tech. Rep*, vol. 2307, no. 09, 2009.

[15] F. Bellavia, M. Fanfani, F. Pazzaglia, and C. Colombo, "Robust selective stereo slam without loop closure and bundle adjustment," in *Image Analysis and Processing – ICIAP 2013*, ser. Lecture Notes in Computer Science, A. Petrosino, Ed. Springer Berlin Heidelberg, 2013, vol. 8156, pp. 462–471.

[16] S. Scherer, D. Dube, and A. Zell, "Using depth in visual simultaneous localisation and mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2012, pp. 5216–5221.

[17] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision ECCV 2010*, ser. Lecture Notes in Computer Science, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Springer Berlin Heidelberg, 2010, vol. 6314, pp. 778–792.

[18] J. Shi and C. Tomasi, "Good features to track," in *1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94.*, Jun 1994, pp. 593–600.

[19] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proceedings of the 9th European Conference on Computer Vision - Volume Part I*, ser. ECCV'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 430–443.

[20] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 3607–3613.

[21] M. Fallon, H. Johannsson, M. Kaess, D. Rosen, E. Muggler, and J. Leonard, "Mapping the MIT Stata Center: Large-Scale Integrated Visual and RGB-D SLAM," in *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, Jul 2012.

[22] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *International Journal of Robotics Research (IJRR)*, 2013.