Robotics and Autonomous Systems 93 (2017) 27–42

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

S-PTAM: Stereo Parallel Tracking and Mapping

Taihú Pire [a,*], Thomas Fischer [a], Gastón Castro [a], Pablo De Cristóforis [a], Javier Civera [b], Julio Jacobo Berlles [a]

[a] University of Buenos Aires, Argentina
[b] University of Zaragoza, Spain

HIGHLIGHTS

- Parallel Nature of the SLAM problem is exploited achieving real-time performance.
- Stereo constraints are used for point initialization, mapping and tracking phases.
- Real-time loop detection and correction are included in the system.
- Local Bundle Adjustment runs in parallel to refine local co-visible area.
- Wheel odometry can be used to feed the Stereo SLAM system.

ARTICLE INFO

Article history:
Available online 20 April 2017

Keywords:
SLAM
Visual SLAM
Stereo SLAM
Stereo vision
Loop closure

ABSTRACT

This paper describes a real-time feature-based stereo SLAM system that is robust and accurate in a wide variety of conditions – indoors, outdoors, with dynamic objects, changing light conditions, fast robot motions and large-scale loops. Our system follows a parallel-tracking-and-mapping strategy: a tracking thread estimates the camera pose at frame rate; and a mapping thread updates a keyframe-based map at a lower frequency. The stereo constraints of our system allow a robust initialization – avoiding the well-known bootstrapping problem in monocular systems–and the recovery of the real scale. Both aspects are essential for its practical use in real robotic systems that interact with the physical world.

In this paper we provide the implementation details, an exhaustive evaluation of the system in public datasets and a comparison of most state-of-the-art feature detectors and descriptors on the presented system. For the benefit of the community, its code for ROS (Robot Operating System) has been released.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

A robust and accurate self-localization and mapping of the surrounding areas is an essential competence to perform robotic tasks autonomously in a wide variety of applications and scenarios. Due to the sensor noise, constructing and updating the map of an unknown environment has to be done simultaneously with the estimation of the robot pose within it. Such problem is usually referred with the acronym SLAM, standing for Simultaneous Localization and Mapping, and has been the object of active research during the last two decades.

Most of the early works on SLAM made use of a laser rangefinder as the main sensor [1], in combination with wheel odometry. More recently, visual sensors – either passive [2] or active [3]–have become the dominant choice. The odometric information has become less relevant, making visual SLAM suitable for other applications like Augmented and Virtual Reality. The affordable, small and light now-a-day cameras can provide high resolution data in real-time. Their range is unlimited – at the assumable price of a large depth uncertainty for small parallax pixels–, in contrast to the range limits of laser sensors. Moreover, cameras are passive sensors and therefore do not interfere with each other, and unlike *Structured light range sensors* (SLRS), they can be used in both indoor and outdoor environments. These characteristics make cameras the best choice for a general multi-purpose mobile robotic platform.

For the above reasons, visual SLAM has become one of the most studied topics in the latest decade. And nowadays it is possible to achieve robust and accurate visual SLAM results in real time. However, some significant challenges remain, particularly for monocular configurations – namely highly dynamic environments or fast camera motions. In these scenarios a stereo cameras offers a higher degree of robustness. Triangulating the depth from a single view – and hence initializing points with small uncertainty– allows to initialize the system robustly and augment the map with

* Corresponding author.
E-mail addresses: tpire@dc.uba.ar (T. Pire), tfischer@dc.uba.ar (T. Fischer), gcastro@dc.uba.ar (G. Castro), pdecris@dc.uba.ar (P. De Cristóforis), jcivera@unizar.es (J. Civera), jacobo@dc.uba.ar (J. Jacobo Berlles).

http://dx.doi.org/10.1016/j.robot.2017.03.019
0921-8890/© 2017 Elsevier B.V. All rights reserved.

undelayed low-uncertainty depth information. In addition a stereo setting allows to recover the real scale and avoid the scale drift. While a monocular-inertial combination (e.g., [4]) can also be used to extract the real scale of the scene, the reader should notice that the two sensor settings are complementary. Inertial sensors are not reliable in periods of constant velocity motion. Stereo cameras, on the other hand, are equivalent to monocular ones in low-parallax configurations – large scene depths compared to its baseline. A stereo-inertial combination (as in [5]) can be used to avoid their individual limitations. RGB-D sensors also provide the real scale of the scene for SLAM and have the added value of dense depth measurements (for example, [6]). However, the depth measurements are range-limited and they cannot work under direct sunlight, so they are limited to indoor scenes and lack the generality of stereo cameras.

In this work we present a real-time SLAM system using a stereo camera, henceforth referred to as S-PTAM. Stereo cameras allow to match the same visual point-landmarks on a pair of synchronized views, recovering their real depth accurately if the parallax is high. As the robot moves through the environment it is possible to track the visual landmarks frame after frame, improve their depth estimation and track the robot pose. In the experiments of this paper, the stereo setting plays a key role in some challenging cases of dynamic objects and changing lighting conditions.

Feature-based visual SLAM approaches rely on the quality and quantity of local image features. On the one hand, the accuracy of the localization heavily depends on the homogeneous deployment of features in images and the ability to track them for long periods, even from different points of view and lighting conditions. On the other hand, if the number of points in the map grows too quickly, it may slow down the whole system. To be able to keep the response of the system under real-time constraints, images have to be dropped or other parts of the system, like optimization routines, must use less computational resources. Currently, there exist several local image feature extractors. A feature extractor is a combination of a salient point (called *keypoint*) detection procedure and a computation of a unique signature (called *descriptor*) for each such a detected point. The most commonly used detectors are SIFT [7], SURF [8], STAR [9], GFTT [10], FAST [11], AGAST [12], and the relatively recently proposed ORB [13], while among the most used descriptors we can mention SIFT, SURF, ORB, BRIEF [14], BRISK [15], and LATCH [16]. In this work, we also evaluate the impact of different state-of-the-art feature extractors on the performance of the visual SLAM localization method to find the best option.

Following the approach of Parallel Tracking and Mapping (PTAM) [17], S-PTAM divides the problem into two main parallel tasks: camera tracking and map optimization. These tasks run 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 to achieve a flexible, robust and accurate stereo SLAM system. Its main characteristics can be summarized as follows:

- The SLAM problem is heavily parallelized achieving real-time performance, whilst minimizing inter-thread dependency.
- The stereo constraints are used for point initialization, mapping and tracking, improving the accuracy and robustness of the system.
- Real-time loop detection and correction are included in the system. The loop detection is performed using appearance-based image matching and the loop correction by optimizing a pose graph representation of the map.
- A maintenance process that runs in an independent thread iteratively refines the map (Bundle Adjustment) in a local co-visible area, improving global consistency.
- Although the method works with the only input of a stereo sequence, wheel odometry can also be used for further accuracy and robustness.
- Binary features are used to describe visual point-landmarks, thus reducing the storage requirements and the matching cost.

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. This paper builds on our previous work [18], being the additional contributions: (1) a more extended and detailed description of the whole system, (2) the design and implementation of a real-time loop closure algorithm, (3) an assessment of the impact of most state-of-the-art image feature extractors on the performance of the system and (4) a more extended and exhaustive evaluation of the system in several public datasets.

2. Related work

Although SLAM in general and stereo SLAM in particular are two broad topics with a vast extent of associated bibliography, we will describe here the main research lines and the works that are more related to ours.

[19,20] and [21] can be considered some of the earliest works on stereo SLAM. The first one estimates an edge map from a trinocular sensor. The second one estimates a piece-wise planar reconstruction of a room-sized scenario from some dozens stereo pairs. And the third one, a seminal work, estimates a sparse map of SIFT features. [22] describes an active stereo SLAM that uses an Extended Kalman Filter (EKF). Filtering was the main approach for SLAM on its early days, and in addition to the EKF the particle filters [23,24] were another popular choice. [25] is an example of a stereo SLAM based on particle filters.

The progress in stereo SLAM algorithms has gone hand in hand with other visual settings, in particular with the monocular ones. Based on the EKF feature-based monocular SLAM of [2] – the first one demonstrating real time in room-sized scenes–, [26] proposed a formulation using a stereo camera. They incorporate an inverse depth point parametrization [27], a joint compatibility test [28] for the rejection of spurious matches and a submapping strategy [29] to estimate robust, accurate and larger maps (e.g., of building halls or squares).

EKF-based approaches were demonstrated to be inconsistent in the long term due to the integration of the linearization errors [30]. A decade later [31] showed that they are also less efficient – in terms of information processing per time unit–than a parallel tracking and mapping approach, the latest becoming the dominant algorithm to the current days. PTAM [17] is one of the first and most representative systems on this line, originally designed for small Augmented Reality applications. This system divides the tracking and mapping estimation into two separate threads, exploiting the availability of multi-core processors. The first thread tracks the camera motion at every frame assuming a known map. The second thread estimates a 3D map for a subset of keyframes at a lower frame rate, which allows the use of non-linear batch optimization techniques such as Bundle Adjustment [32] – a gold standard in Structure from Motion.

Stereo Visual Odometry (VO) presents a tight relation with stereo SLAM. The former aims only to local consistency and the latter to global consistency, but both use similar methods in many of

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

its parts. The parallel tracking and mapping approach and the non-linear local optimization are present in most of the best performing stereo VO systems (e.g., [33]).

The stereo SLAM research has focused on the last decade on a higher robustness, a higher accuracy and larger maps, with small variations in the fundamentals – with the exceptions detailed on the two last paragraphs of this section. FrameSLAM [34], for example, addresses global localization for large trajectories (up to 10 km in real-time) using stereo in combination with GPS and IMU sensors in some experiments. It uses stereo VO to estimate the incremental motion while a pose graph models the global pose. The pose graph is built by marginalizing the point features and even some of the poses, resulting in what they call a skeleton, that allows a fast global optimization while the camera is localized with the local map.

Pose graphs are present in the main stereo SLAM works in order to reduce the complexity while keeping the global structure. RSLAM [35] models the map as a sequence of relative poses and the landmarks in their local camera frames. To provide an accurate local map RSLAM uses an active region of frames (the most closest frames in terms of distance) to perform Bundle Adjustment. The active region defines the landmarks visible from the current frame. Representing the local environment around the robot consists in projecting the active region into the current frame. Landmarks with base frames (where the landmark's 3-D coordinates are kept) belonging to poses from the active region are projected into the current frame by composing the transforms along the edges. In this framework, loop closure consists in creating a new edge that can then be used to transfer 3-D landmark estimates into the current frame and therefore evaluate their projection in the image. Accordingly, the system does not provide a global map consistency.

[36] proposes a double window optimization approach instead of the common active window approach allowing to deal with loopy camera motions. In a loopy camera motion the number of keyframes at the boundary is relatively large with respect to the total number of keyframes within the active window, and fixing them hampers convergence. The double window optimization approach deal with this kind of movements defining an inner window and an outer window. The inner window uses point-pose constraints and it is supported by the outer window which uses pose–pose constraints. In this way, while the inner window serves to model the local area as accurately as possible, the pose-graph in the outer window acts to stabilize the periphery.

ORB-SLAM2 [37] is the more representative feature-based SLAM nowadays. It updates the PTAM framework by making several state-of-the-art additions to obtain a more robust and accurate performance in larger scenarios. Among others the system uses [38] for loop closure detection, [39] to update the pose graph accounting for the monocular scale drift, and the covisibility map technique proposed in [36] for large trajectories.

Our system S-PTAM is a feature-based stereo and inherits the best practices of the above referenced works. We use local Bundle Adjustment in the neighborhood of the current frame to have a locally consistent feature map for accurate tracking and a pose graph modeling the global structure to correct the drift if we revisit places. The main difference of S-PTAM with the above system is that we initialize the map features in the tracking thread at frame rate, resulting in a higher resilience in fast camera motions and the capability of the creation of map points during loop closing optimizations. Mainly because of these features, but also partly to other implementation details, we outperform several state-of-the-art baselines in public databases.

Recently, visual SLAM and visual odometry started using direct methods [40] that minimize the photometric error of high-gradient pixels (in contrast to the geometric error of salient pixels in the image) in order to estimate the map and camera poses. As its key benefit, these algorithms are able to estimate more dense maps than the traditional feature-based ones described above. Their accuracy should be better, as they integrate the information of more pixels and avoid the artifacts that the feature extraction process might produce. However, [41] reported a higher accuracy for state-of-the-art feature-based methods, possibly due to the lower maturity of the direct approach. Notice that the results in the KITTI dataset [42] agree with this latest paper and the accuracy of the best direct SLAM method [43] is still lower than the one of ORB-SLAM2 [37] and our work S-PTAM.

Early direct SLAM/VO works used a stereo setting [44,45]. Other works using a monocular camera have produced fully dense and accurate maps with a TV-regularization of the photometric solution [46] or the addition of scene priors and learned patterns [47]. Some other works estimate a semidense map of the highest-gradient pixels to avoid the large errors produced by the TV-regularization in low-gradient areas (e.g., [43]). Currently, the best representative of direct SLAM using stereo cameras is [48].

3. Notation

SE(3) transformation $\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}$. $\boldsymbol{R}$ stands for a rotation matrix and $\boldsymbol{t}$ for a translation vector. $\boldsymbol{T}$ is a transformation belonging to the Lie Group, $SE(3)$, the group of rigid-body motions in 3D. In particular, we use $\boldsymbol{E}^{\mathrm{CW}}$ as the transformation which represents a camera pose that transforms a point in world coordinates frame $\boldsymbol{x}^{\mathrm{W}} = \left[x^{\mathrm{W}} y^{\mathrm{W}} z^{\mathrm{W}} 1\right]^{\top}$ to a point in camera coordinates frame $\boldsymbol{x}^{\mathrm{C}} = \left[x^{\mathrm{C}} y^{\mathrm{C}} z^{\mathrm{C}} 1\right]^{\top}$, that is:

$$\boldsymbol{x}^{\mathrm{C}} = \boldsymbol{E}^{\mathrm{CW}} \boldsymbol{x}^{\mathrm{W}}. \tag{1}$$

Motion matrix noted with $\boldsymbol{M}$, is a 4×4 matrix (belongs to $SE(3)$) which represents the changes in camera pose by left-multiplication, $\boldsymbol{E}^{\mathrm{CW}} = \boldsymbol{M}^{\mathrm{C}} \boldsymbol{E}^{\mathrm{CW}}_{\mathrm{prev}}$. In Lie Groups, the motion matrix $\boldsymbol{M}$ could be represented by a six-vector $\boldsymbol{\mu} = (t_x, t_y, t_z, \theta^{\mathrm{roll}}, \theta^{\mathrm{pitch}}, \theta^{\mathrm{yaw}})$, where the first three elements correspond to translation and the last three elements correspond to rotation angles. The motion vector $\boldsymbol{\mu}$ and motion matrix $\boldsymbol{M}$ are related by:

$$\boldsymbol{M} = \exp(\boldsymbol{\mu}) = e^{\sum_{j=1}^{6} \mu_j \boldsymbol{G}_j}, \tag{2}$$

where $\boldsymbol{G}_j$ with $j = 1 \cdots 6$ are the group generator matrices. They result from the partial derivatives of motion matrices with respect to the motion parameters evaluated in $\boldsymbol{\mu} = 0$, that is $\frac{\partial \boldsymbol{M}}{\partial \mu_j} = \boldsymbol{G}_j$. For further information on Lie Groups the reader is referred to [49].

Measurement noted with letter $\boldsymbol{z} = \begin{bmatrix} u \\ v \end{bmatrix}$, is the true 2D position that matches with the projected 3D point on the camera's image plane.

Map point noted with $\boldsymbol{p}$, is an ordered pair $\left(\boldsymbol{x}^{\mathrm{W}}, \boldsymbol{d}\right)$ which contains the 3D point $\boldsymbol{x}^{\mathrm{W}}$ and its associated descriptor $\boldsymbol{d}$.

Stereo keyframe noted with letter $\boldsymbol{K}$, is a stereo pair of images with the associated stereo camera pose.

Map is defined as the set of map points and the set of stereo keyframes.

A point in camera reference frame $\boldsymbol{x}^{\mathrm{C}}$, projects into the image as

$$\begin{bmatrix} \hat{\mathbf{u}} \\ \hat{\mathbf{v}} \end{bmatrix} = P\left(\boldsymbol{x}^{\mathrm{C}}\right). \tag{3}$$

We project the 3D points in the image plane using the well-known pinhole camera model

$$P\left(\boldsymbol{x}^{\mathrm{C}}\right) = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \end{bmatrix} \begin{bmatrix} \frac{x^{\mathrm{C}}}{z^{\mathrm{C}}} \\ \frac{y^{\mathrm{C}}}{z^{\mathrm{C}}} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{f_u x^{\mathrm{C}}}{z^{\mathrm{C}}} + u_0 \\ \frac{f_v y^{\mathrm{C}}}{z^{\mathrm{C}}} + v_0 \end{bmatrix}, \tag{4}$$

where we assume that the images are rectified, f_u and f_v are the focal length in the horizontal and vertical coordinates, and $[u_0, v_0]^\top$ is the image position of the principal point.

4. Method

Fig. 1 shows a scheme of the main components and the computation flow of S-PTAM.

Our system defines the global reference frame at the camera pose in the first frame of the sequence. An initial map is estimated by matching and triangulating salient point features in the first stereo pair. For every frame after the tracking thread estimates the 6DOF pose for each stereo frame by minimizing the re-projection error between the projected map points and their correspondences. The system selects a subset of keyframes that will be used in a second thread to estimate the map at a lower rate. The map points are triangulated from the stereo matches of each keyframe, and added to the map. The mapping thread is constantly minimizing the local re-projection error by refining all the map points and the stereo poses using Bundle Adjustment. We use a pose graph to maintain the global consistency of the map. Point correspondences are actively searched between keyframes in order to strengthen the constraints of the pose graph. The map is a shared resource between tracking, mapping and loop closing threads.

To deal with the accumulated errors in large trajectories, S-PTAM runs a loop closure detection in a third thread. This thread searches for loop closure candidates using the visual appearance of features. We confirm the potential candidates by a robust motion estimation from features correspondences. This relative motion estimation is then added to the pose graph, that is optimized to accommodate such constraint.

The next sections of this paper provide a more detailed explanation of each component of the system.

4.1. Feature extraction and description

S-PTAM relies on matching local image features for localization and mapping. The pose of each stereo frame is estimated from the correspondences between the 3D map features and the 2D image features. Every local feature that does not have a map correspondence is triangulated from the stereo matches and added to the map. The mapping system also searches for correspondences between keyframes and map points. The viewpoint invariance and the cost are the two key aspects for local features in SLAM, as wide-baseline matching improves the accuracy and a high extraction/description cost reduces the budget for other tasks.

For the present system, the GFTT [10] algorithm was selected to detect the image key-points, and the BRISK [15] extractor to describe their features. This combination of feature detector and descriptor algorithms was chosen based on a thorough evaluation of state-of-the-art feature detectors and descriptors – see the details in Section 6.2.

4.2. Pose tracking

Our pose tracking thread consists of 4 sequential steps.

4.2.1. Matching

We project each map point inside the viewing frustum of the predicted stereo pose and search for the match in a neighborhood of the point. A reasonable prediction of the current camera pose is necessary in order to perform such 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* can be used instead. Matching between map points and features is carried out comparing the descriptors. As binary descriptors are used, the Hamming-distance is calculated. If the distance is below a given threshold the match is valid, otherwise it is discarded.

4.2.2. Pose refinement

In order to estimate the current camera pose $\boldsymbol{E}^{\mathrm{CW}}$ in the global reference frame W, we compose the previous camera pose $\boldsymbol{E}^{\mathrm{CW}}_{\mathrm{prev}}$ with the relative motion $\boldsymbol{M}^{\mathrm{C}}$ in the local camera frame

$$\boldsymbol{E}^{\mathrm{CW}} = \boldsymbol{M}^{\mathrm{C}} \boldsymbol{E}^{\mathrm{CW}}_{\mathrm{prev}}, \tag{5}$$

To find the relative motion $\boldsymbol{M}^{\mathrm{C}}$ we use the following equation

$$\boldsymbol{J}\boldsymbol{\mu} = \boldsymbol{\Delta z}\left(\boldsymbol{\mu}_{\mathrm{prev}}\right) \tag{6}$$

where $\boldsymbol{\mu}$ is composed of the relative motion parameters in vector form $\boldsymbol{\mu} = (t_x, t_y, t_z, \theta_{roll}, \theta_{pitch}, \theta_{yaw})^\top$, $\boldsymbol{\Delta z}$ is the re-projection error (only depending on the camera motion $\boldsymbol{\mu}$, as we consider the map fixed), and $\boldsymbol{J}$ is the Jacobian of the re-projection error with respect to the camera motion parameters. Each element J_{ij} of the Jacobian is computed as

$$J_{ij} = \frac{\partial \boldsymbol{\Delta} z_i(\boldsymbol{\mu})}{\partial \mu_j} = \frac{\partial \left(\begin{bmatrix} u \\ v \end{bmatrix}_i - P\left(\exp(\boldsymbol{\mu}) \boldsymbol{E}^{\mathrm{CW}}_{\mathrm{prev}} \boldsymbol{x}^{\mathrm{W}}_i\right)\right)}{\partial \mu_j} \tag{7}$$

$$= -\frac{\partial P\left(\boldsymbol{x}^{\mathrm{C}}_i\right)}{\partial \boldsymbol{x}^{\mathrm{C}}_i} \frac{\partial \boldsymbol{x}^{\mathrm{C}}_i}{\partial \mu_j}, \tag{8}$$

where

$$\frac{\partial P\left(\boldsymbol{x}^{\mathrm{C}}_i\right)}{\partial \boldsymbol{x}^{\mathrm{C}}_i} = \begin{bmatrix} \frac{f_u}{z^{\mathrm{C}}} & 0 & -\frac{f_u x^{\mathrm{C}}}{z^{\mathrm{C}2}} \\ 0 & \frac{f_v}{z^{\mathrm{C}}} & -\frac{f_v y^{\mathrm{C}}}{z^{\mathrm{C}2}} \end{bmatrix} \tag{9}$$

and

$$\frac{\partial \boldsymbol{x}^{\mathrm{C}}_i}{\partial \mu_j} = \boldsymbol{G}_j \boldsymbol{E}^{\mathrm{CW}}_{\mathrm{prev}} \boldsymbol{x}^{\mathrm{W}}_i. \tag{10}$$

The motion vector $\boldsymbol{\mu}$ is found by solving the Eq. (6). In order to do this, given a set $S = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N\}$ of matched measurements, the new value for $\boldsymbol{\mu}$ is obtained by minimizing an objective function as follows

$$\boldsymbol{\mu}' = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i \in S} \rho\left(\boldsymbol{J}_i \boldsymbol{\mu} - \boldsymbol{\Delta z}_i(\boldsymbol{\mu}_{\mathrm{prev}})\right), \tag{11}$$

where $\rho(.)$ is the Huber function used to reduce the effect of outliers. The minimization in (11) is performed using the well-known Levenberg–Marquardt algorithm.

4.2.3. Keyframes selection and map points creation

Once the current pose is estimated, a frame is selected to be a *keyframe* if 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. Other visual SLAM systems (like PTAM [17]) create new map points once the keyframe is processed by the mapping thread that may not be immediate, depending on the level of congestion, potentially causing a tracking failure. In contrast, S-PTAM immediately creates and incorporates the new points into the map after the tracking step to avoid the loss of potential map matches on the upcoming frames. Finally, the keyframe is queued into the map refinement thread, to be processed as soon as possible.

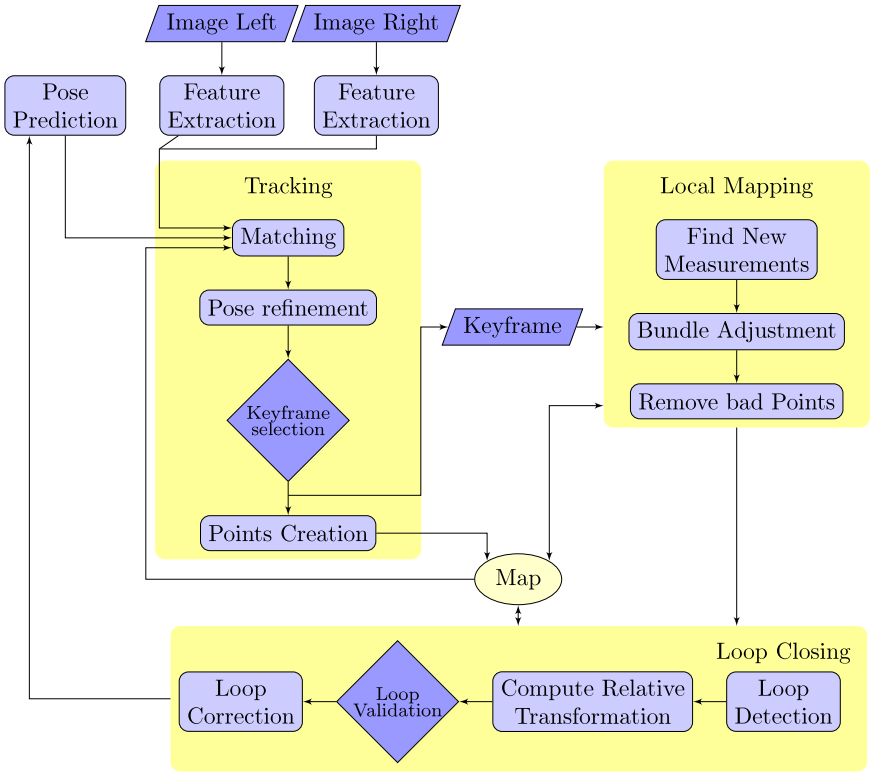


Fig. 1. S-PTAM overview.

4.3. Local mapping

This section details the stereo mapping algorithm that uses multiview and stereo constraints to refine the estimated map (keyframe poses and salient points' positions). Our system follows mainly the local mapping approach presented in [17], extending it with the stereo constraints.

The refinement of the camera poses (keyframe map) and the 3D points (point cloud map) is done with a particular case of least squares estimation called Bundle Adjustment, that minimizes the re-projection error of every point in every image. The problem can be stated as follows: given an initial set of N keyframe poses $\{\boldsymbol{E}_1, \ldots, \boldsymbol{E}_N\}$, an initial set of M 3D points $\boldsymbol{x}^{\mathrm{W}} = \{\boldsymbol{x}_1^{\mathrm{W}}, \ldots, \boldsymbol{x}_M^{\mathrm{W}}\}$ and a family of measurement sets $\{S_1, \ldots, S_N\}$, where each set S_j contains the measurement $\boldsymbol{z}_{ij}$ of the ith point in the jth keyframe, the simultaneous estimation of the multiple cameras and the point cloud is achieved by solving

$$\boldsymbol{J}\begin{bmatrix}\boldsymbol{\mu}\\ \boldsymbol{x}\end{bmatrix} = \Delta z\left(\boldsymbol{\mu}_{\mathrm{prev}}, \boldsymbol{x}_{\mathrm{prev}}^{\mathrm{W}}\right), \tag{12}$$

where

$$\Delta z\left(\boldsymbol{\mu}, \boldsymbol{x}^{\mathrm{W}}\right) = \boldsymbol{z} - P\left(\exp(\boldsymbol{\mu})\boldsymbol{E}_{\mathrm{prev}}^{\mathrm{CW}}\boldsymbol{x}^{\mathrm{W}}\right) \tag{13}$$

is the reformulated re-projection error where the dependence of the 3D point is included. In a way analogous to the minimization used in 4.2.2, we must minimize the double summation in (14)

$$\left\{\boldsymbol{\mu}'_{j=2\cdots N}, \boldsymbol{x}'^{\mathrm{W}}_{i=1\cdots M}\right\} = \underset{\{\{\boldsymbol{\mu}\},\{\boldsymbol{x}^{\mathrm{W}}\}\}}{\operatorname{argmin}} \sum_{j=1}^{N}\sum_{i\in S_j} \rho_{\sigma_T}\left(\psi_{ji}\right), \tag{14}$$

where

$$\psi_{ji} = \boldsymbol{J}_{ji}\begin{bmatrix}\boldsymbol{\mu}_j\\ \boldsymbol{x}_i^{\mathrm{W}}\end{bmatrix} - \Delta z_i\left(\boldsymbol{\mu}_{\mathrm{prev},j}, \boldsymbol{x}_{\mathrm{prev},i}^{\mathrm{W}}\right).$$

Observe that $\boldsymbol{\mu}_1$ is fixed during the Bundle Adjustment refinement. This is because the first keyframe is given zero uncertainty, as it defines the world reference frame. Given that the vector of parameters is divided into two groups (cameras and points) the Jacobian can be decomposed as

$$J = \left[\frac{\partial \Delta z\left(\boldsymbol{\mu}, \boldsymbol{x}^{\mathrm{W}}\right)}{\partial \boldsymbol{\mu}} \middle| \frac{\partial \Delta z\left(\boldsymbol{\mu}, \boldsymbol{x}^{\mathrm{W}}\right)}{\partial \boldsymbol{x}^{\mathrm{W}}}\right].$$

The computation of the Jacobian is performed as follows. The part corresponding to the camera pose has the form given by (7), whereas the part corresponding to the point cloud parameters have the form

$$\boldsymbol{J}_{ji} = \frac{\partial \Delta z(\boldsymbol{\mu}_j, \boldsymbol{x}_i^{\mathrm{W}})}{\partial \boldsymbol{x}_i^{\mathrm{W}}} \tag{15}$$

$$= \frac{\partial\left(\begin{bmatrix}u\\ v\end{bmatrix}_i - P\left(\exp(\boldsymbol{\mu}_j)\boldsymbol{E}_{\mathrm{prev}}^{\mathrm{C}_j\mathrm{W}}\boldsymbol{x}_i^{\mathrm{W}}\right)\right)}{\partial \boldsymbol{x}_i^{\mathrm{W}}} \tag{16}$$

$$= -\frac{\partial P\left(\boldsymbol{x}_i^{\mathrm{C}_j}\right)}{\partial \boldsymbol{x}_i^{\mathrm{C}_j}}\frac{\partial \boldsymbol{x}_i^{\mathrm{C}_j}}{\partial \boldsymbol{x}_i^{\mathrm{W}}}. \tag{17}$$

The first partial derivative is given by (9) and the second partial derivative results from

$$\frac{\partial \boldsymbol{x}_i^{\mathrm{C}_j}}{\partial \boldsymbol{x}_i^{\mathrm{W}}} = \frac{\partial\left(\boldsymbol{M}^{\mathrm{C}_j}\boldsymbol{E}_{\mathrm{prev}}^{\mathrm{C}_j\mathrm{W}}\boldsymbol{x}_i^{\mathrm{W}}\right)}{\partial \boldsymbol{x}_i^{\mathrm{W}}} = \boldsymbol{R}. \tag{18}$$

To this point, we have addressed the Bundle Adjustment multi-view constraints in one of the stereo images (without loss of generalization, the left one). Adding the stereo constraint slightly differs from the above. The relative motion between the left and the right cameras is fixed, so we can obtain the pose of the right camera from the left camera using

$$\boldsymbol{E}^{\mathrm{RW}} = \boldsymbol{E}^{\mathrm{RL}}\boldsymbol{M}^{L}\boldsymbol{E}_{\mathrm{prev}}^{\mathrm{LW}}. \tag{19}$$

Now, we can use the right camera measurements to add stereo constraints to Bundle Adjustment. These constraints are given by

$$\boldsymbol{z}^{\mathrm{R}} = P^{\mathrm{R}}\left(\boldsymbol{E}^{\mathrm{RL}}\boldsymbol{M}^{L}\boldsymbol{E}^{\mathrm{LW}}_{\mathrm{prev}}\boldsymbol{x}^{\mathrm{W}}\right). \tag{20}$$

Summing up, a 3D–2D point constraint is modeled with (4) for the left camera and with Eq. (20) for the right camera. The Jacobian rows related to the right camera measurements have the form

$$\boldsymbol{J}^{\mathrm{R}}_{ji} = \frac{\partial \boldsymbol{\Delta} z^{\mathrm{R}}(\boldsymbol{\mu}_j, \boldsymbol{x}^{\mathrm{W}}_i)}{\partial \boldsymbol{x}^{\mathrm{W}}_i} \tag{21}$$

$$= \frac{\partial\left(\begin{bmatrix} u^{\mathrm{R}} \\ v^{\mathrm{R}} \end{bmatrix}_i - P^{\mathrm{R}}\left(\boldsymbol{E}^{\mathrm{RL}}\exp(\boldsymbol{\mu}_j)\boldsymbol{E}^{\mathrm{L}_j\mathrm{W}}_{\mathrm{prev}}\boldsymbol{x}^{\mathrm{W}}_i\right)\right)}{\partial \boldsymbol{x}^{\mathrm{W}}_i} \tag{22}$$

$$= \begin{bmatrix} \frac{f_u}{z^{\mathrm{R}}} & 0 & -\frac{f_u x^{\mathrm{R}}}{z^{\mathrm{R}^2}} \\ 0 & \frac{f_v}{z^{\mathrm{R}}} & -\frac{f_v y^{\mathrm{R}}}{z^{\mathrm{R}^2}} \end{bmatrix} \boldsymbol{R}^{\mathrm{RL}}\boldsymbol{R}. \tag{23}$$

Notice that, if the stereo camera is rectified, then the transformation between cameras is a pure translation in the x-axis (baseline) and the intrinsic parameters are the same, therefore $y^{\mathrm{L}} = y^{\mathrm{R}}$ and $z^{\mathrm{L}} = z^{\mathrm{R}}$ and (21) can be rewritten as

$$\boldsymbol{J}^{\mathrm{R}}_{ji} = \begin{bmatrix} \frac{f_u}{z^{\mathrm{L}}} & 0 & -\frac{f_u x^{\mathrm{R}}}{z^{\mathrm{L}^2}} \\ 0 & \frac{f_v}{z^{\mathrm{L}}} & -\frac{f_v y^{\mathrm{L}}}{z^{\mathrm{L}^2}} \end{bmatrix} \boldsymbol{R}. \tag{24}$$

Finally the Jacobian can be expressed as

$$\boldsymbol{J}_{ji} = \begin{bmatrix} \frac{f_u}{z^{\mathrm{L}}} & 0 & -\frac{f_u x^{\mathrm{L}}}{z^{\mathrm{L}^2}} \\ 0 & \frac{f_v}{z^{\mathrm{L}}} & -\frac{f_v y^{\mathrm{L}}}{z^{\mathrm{L}^2}} \\ \frac{f_u}{z^{\mathrm{L}}} & 0 & -\frac{f_u x^{\mathrm{R}}}{z^{\mathrm{L}^2}} \end{bmatrix} \boldsymbol{R}. \tag{25}$$

4.4. Loop closure

Handling large environments requires a system capable of recognizing already-visited places and optimizing the map and the trajectory, in order to reduce the accumulated drift and maintain a globally consistent map model. To accomplish this task, the process is divided into three phases: the detection of revisited places, the estimation of the relative transformation and the loop correction.

In the detection phase, the keyframes provided by the local mapping are described by a bag-of-binary-words using a previously trained visual vocabulary. The computed bag-of-binary-words are used to create a keyframe database as proposed in [50]. For each new keyframe the database is queried to obtain those previously added keyframes that are similar in terms of appearance.

If a loop closure candidate is found, the relative transformation between the queried keyframe and the loop candidate keyframe is estimated. This transformation will serve to measure the accumulated error and to validate the proposed loop determined by the matched keyframes.

Once a loop has been considered valid, a correction is propagated among all the keyframes of the loop providing an initial seed for a later pose graph optimization obtaining a more accurate solution that reduces the accumulated drift error.

4.4.1. Loop detection

Loop detection is achieved making use of the efficient appearance-based method proposed in [38]. Each new keyframe $\boldsymbol{K}_i$ is described as a bag-of-words vector $\mathbf{v}_i$ and the keyframe database is queried scoring any previously added $\mathbf{v}_j$ that shares words with $\mathbf{v}_i$ following the normalized similarity score:

$$\eta\left(\mathbf{v}_i, \mathbf{v}_j\right) = \frac{s\left(\mathbf{v}_i, \mathbf{v}_j\right)}{s\left(\mathbf{v}_i, \mathbf{v}_{i-1}\right)},$$

where $\mathbf{v}_{i-1>0}$ is the bag of words representation of the previous inserted keyframe, and the similarity score between two bags of words $s(\mathbf{v}_i, \mathbf{v}_j)$ is an L_1-score which lies in $[0, 1]$:

$$s\left(\mathbf{v}_i, \mathbf{v}_j\right) = 1 - \frac{1}{2}\left|\frac{\mathbf{v}_i}{|\mathbf{v}_i|} - \frac{\mathbf{v}_j}{|\mathbf{v}_j|}\right|.$$

In the case that the highest normalized similarity score exceeds a predefined threshold, its respective keyframe is considered a match and a potential loop candidate.

4.4.2. Map points matching and geometric verification

Once the current keyframe $\boldsymbol{K}_{\mathcal{C}}$ has been successfully matched with a loop candidate keyframe $\boldsymbol{K}_{\ell}$, the relative transformation $\boldsymbol{T}^{\mathrm{C}_{\mathcal{C}}\mathrm{C}_{\ell}}$ existing between $\boldsymbol{K}_{\mathcal{C}}$ and $\boldsymbol{K}_{\ell}$ must be computed. This transformation will be used to perform the correction on the detected loop. In order to be avoid false positive loops, an initial estimation of the relative transformation, along with the set of map points inliers associated, are computed in a first step. An initial transformation between the matched keyframes is computed performing RANSAC with a P3P (Perspective-3-Point) solver [51] over 3D–2D correspondences established between map points observed by $\boldsymbol{K}_{\mathcal{C}}$ and features extracted in $\boldsymbol{K}_{\ell}$. If the percentage of inliers exceeds a given threshold then the detected loop is considered valid and a general PnP (Perspective-n-Point) solver [52] is used, over the inlier matches, to estimate a more accurate relative transformation. The resulting $\boldsymbol{T}^{\mathrm{C}_{\mathcal{C}}\mathrm{C}_{\ell}}$ is finally refined with a non-linear optimization.

4.4.3. Loop correction and keyframes optimization

At first, the loop correction process estimates an initial update using the computed relative transformation. This correction is performed by propagating the $\boldsymbol{T}^{\mathrm{C}_{\mathcal{C}}\mathrm{C}_{\ell}}$ transformation through the keyframes between $\boldsymbol{K}_{\mathcal{C}}$ and $\boldsymbol{K}_{\ell}$.

Let $\{\boldsymbol{E}^{\mathrm{C}_{\mathcal{C}}\mathrm{W}}, \ldots, \boldsymbol{E}^{\mathrm{C}_{j+1}\mathrm{W}}, \boldsymbol{E}^{\mathrm{C}_j\mathrm{W}}, \ldots, \boldsymbol{E}^{\mathrm{C}_{\ell}\mathrm{W}}\}$ the keyframes poses belonging to the detected loop, the propagation is defined by:

$$\boldsymbol{E}^{\mathrm{C}_{\mathcal{C}}\mathrm{W}}_{prop} = \boldsymbol{T}^{\mathrm{C}_{\mathcal{C}}\mathrm{C}_{\ell}}\boldsymbol{E}^{\mathrm{C}_{\ell}\mathrm{W}}$$

$$\boldsymbol{E}^{\mathrm{C}_j\mathrm{W}}_{prop} = Interpolate_j\left(\boldsymbol{E}^{\mathrm{C}_j\mathrm{W}}, \boldsymbol{E}^{\mathrm{C}_j\mathrm{C}_{j+1}}\boldsymbol{E}^{\mathrm{C}_{j+1}\mathrm{W}}_{prop}\right)$$

$$\boldsymbol{E}^{\mathrm{C}_{\ell}\mathrm{W}}_{prop} = \boldsymbol{E}^{\mathrm{C}_{\ell}\mathrm{W}},$$

where the low index *prop* refers to the camera poses after the propagation. $Interpolate_j\,(*, *)$ performs a linear pose interpolation between the non-corrected and corrected camera pose of $\boldsymbol{K}_j$ according to its distance to where the loop was detected. In this way, keyframes closer to the current keyframe (and therefore closer to the loop point) will be corrected more strongly, whereas the correction for the keyframes located farther from the loop point will be smoothed. This was achieved in practice using quaternion representation and spherical linear interpolation (Slerp).

After the initial loop correction, a pose graph optimization is carried out to get a more accurate solution. At last, each point on the map is corrected by applying the same transformation that was applied to its original keyframe from where it was triangulated.

4.4.4. Map update and components synchronization

To allow the system to operate in real time along with the loop correction extension, two properties must take place:

- The tracking thread must remain operational at real time being able to work with any map point needed and create new keyframes if required.

- The local mapping thread must be able to perform the bundle adjustment over keyframes and map points inside a defined sliding window.

To ensure these two properties, after a loop has been validated, the most recent subset of keyframes selected by the mapping thread is defined as the safe mapping window. Then, the initial loop correction propagation and pose graph optimization are performed over an internal copy of all the keyframes present. Thereafter, the map update process is divided into three stages:

- Update corrected keyframes and map points outside of the defined mapping window.
- Update corrected keyframes and map points inside the defined mapping window applying any optimization that may have been introduced by the local bundle adjustment since the start of the loop closing process.
- Correct any keyframe created and added to the map after the internal copy has been made. This correction is achieved applying the same rigid transformation that has been applied for the correction of the current keyframe.

The tracking and mapping threads need to be paused only during the last two stages, where only a small fraction of the map is updated. After the map update, the only remaining part is to notify, to the pose predictor, the transformation that must be applied to the ongoing trajectory.

Throughout the process, the tracking may encounter map points that are being actively corrected by the loop closing process, but it is expected that the tracking dismisses those map points while projecting and matching.

5. Implementation details

In this section we explain in detail some relevant implementation decisions that allow the system to run in real time on a mobile platform, minimizing the impact on the pose estimation accuracy.

As keypoint detection and extraction is time consuming, the feature processing for each image of the stereo pair is 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, only the map points that are in a covisibility area are considered. The covisibility area, for an incoming frame, is determined by all the points that are shared among near keyframes. These keyframes are those that observed the points tracked by the previous frame. Then, this map points are filtered by camera frustum culling. Points initialized from a very different point of view (more than 45 degrees) are also discarded.

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. For valid matches, the descriptors stored in map points are updated with image features descriptors. The update of map points descriptors allows to track them for a longer period of time.

Global map optimization through Bundle Adjustment becomes prohibitive for large scale environments. Consequently we perform only local optimizations. 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. Our experiments show that the queue size never exceeds four keyframes.

We use the library DBoW2 library [38] for loop closure, configured with the threshold $\alpha = 0.3$, temporal consistency $k = 0$ and no geometric check. A visual vocabulary of 6 levels with 10 clusters per level was trained with a combination of indoor and outdoor image sequences from the MIT Stata Center Dataset [53] and the Málaga Urban Dataset [54] with a total of 10 thousand images.

The openGV library [55] was used for solving the pose estimation needed for the loop correction with a central absolute variant of the methods aforementioned. A detected loop is validated if 80% of the 3D–2D correspondences were found as inliers.

The g^2o (General Graph Optimization) library [56] was used to perform Levenberg–Marquardt minimization during tracking, Bundle Adjustment and pose graph optimization after loop closure. Other graph optimization libraries, Vertigo [57,58] and GTSAM [59], were considered as alternative to g^2o. In [56] is shown a comparison between g^2o and GTSAM library. The comparison showed that g^2o outperform GTSAM. On the other hand, Vertigo is an extension of g^2o and GTSAM which can solve pose graph optimization problems even with the presence of false positive loop closures. During the experimentation with S-PTAM no false positive loop closure occurs (given the strong validation process carried out during the loop detection), and thus the use of Vertigo was dismissed.

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

6. Experiments

6.1. Error metric

To assess the final impact of the different experiment configurations on the accuracy of S-PTAM, we extend a commonly used metric [42,60] specifically designed for evaluating the performance of SLAM systems.

Let $\boldsymbol{x}_k$ be the estimated pose at frame k and $\boldsymbol{x}_k^*$ the corresponding ground truth pose. Let us note the set of differences (or motions) between two frames of a sequence as $\boldsymbol{\delta}_{i,j} = \boldsymbol{x}_j \ominus \boldsymbol{x}_i$, where $\oplus$ is the standard motion composition operator and $\ominus$ it is inverse [61]. Analogously $\boldsymbol{\delta}^*_{i,j} = \boldsymbol{x}^*_j \ominus \boldsymbol{x}^*_i$.

The *relative error* committed between frames i and j becomes $\boldsymbol{\delta}_{i,j} \ominus \boldsymbol{\delta}^*_{i,j}$, and the aforementioned metric is defined as the root–mean–square error (RMSE) over δ. It differs from the original metric [60] by taking the square root, which helps in interpreting numerical results, since the measurement units are the same as for the data. Moreover, to obtain meaningful numerical results, we need to separate the translational $\boldsymbol{\epsilon}_t$ and rotational $\boldsymbol{\epsilon}_\theta$ part of this errors, since they are different in nature, separation which was also suggested by the original authors [60]:

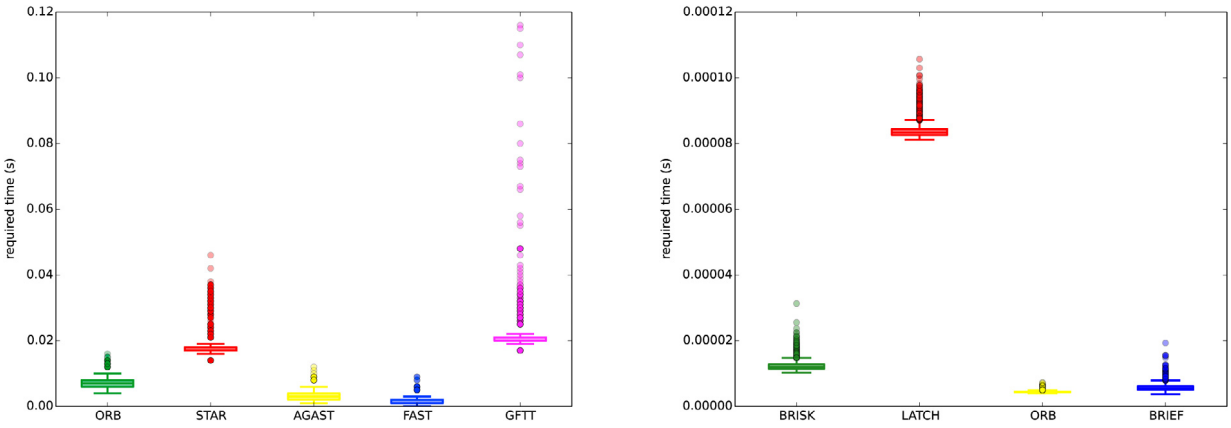
$$\boldsymbol{\epsilon}_t = \sqrt{\frac{1}{N}\sum_{i,\,j} trans\left(\boldsymbol{\delta}_{i,j} \ominus \boldsymbol{\delta}^*_{i,j}\right)^2}$$

$$\boldsymbol{\epsilon}_\theta = \sqrt{\frac{1}{N}\sum_{i,\,j} rot\left(\boldsymbol{\delta}_{i,j} \ominus \boldsymbol{\delta}^*_{i,j}\right)^2}$$

where N is the number of relative displacements $\boldsymbol{\delta}_{i,j}$. In practice, the inverse motion composition operation between two poses $\boldsymbol{x}_j \ominus \boldsymbol{x}_i$ can be computed from the corresponding transformation matrices representing each pose, namely $\boldsymbol{T}_{\boldsymbol{x}_i}$ and $\boldsymbol{T}_{\boldsymbol{x}_j}$ as

$$\boldsymbol{x}_i \ominus \boldsymbol{x}_j = \boldsymbol{T}^{-1}_{\boldsymbol{x}_j}\boldsymbol{T}_{\boldsymbol{x}_i}.$$

This equations intentionally leave open the choice of which relative displacements $\boldsymbol{\delta}_{i,j}$ are included in the metric. As discussed by the original authors [60], the choices will highlight different properties of the data. In our case, we strive for local consistency, which is better highlighted by taking displacements as small as



(a) Feature detection time for each image.

(b) Descriptor extraction time for each feature.

Fig. 2. Feature extraction times. Data was measured over all KITTI training sequences (except 01). Boxes represent interquartile range (*IQR*), whiskers reach to $-1.5 \times IQR$ and $1.5 \times IQR$, and the points represent data beyond those ranges, considered outliers. The line inside the box represents the median.

possible. Therefore relative displacements are taken between consecutive frames, yielding:

$$\epsilon_t = \sqrt{\frac{1}{N}\sum_k trans\left(\delta_{k,k+1} \ominus \delta^*_{k,k+1}\right)^2}$$

$$\epsilon_\theta = \sqrt{\frac{1}{N}\sum_k rot\left(\delta_{k,k+1} \ominus \delta^*_{k,k+1}\right)^2}.$$

6.2. Evaluation of feature extractors

In this section, we assess the impact of image feature extractors on the performance of the S-PTAM system in terms of pose accuracy and computational requirements. Although the evaluation is performed using the S-PTAM system, the obtained results can be generalized for other stereo feature-based SLAM systems.

An image feature extractor can usually be split into two phases, detection and description. The feature detector is used to find salient areas in the image, while the feature descriptor captures and synthesizes the information in a local neighborhood of the selected area. A brief overview of the most commonly used feature extractor and descriptor algorithms, which were considered for comparison, is presented below.

STAR – A modified version of the CenSurE (Center Surrounded Extrema) [9] detector, which is computationally less demanding at the expense of lower precision.

FAST – Features from Accelerated Segment Test [11] is a feature detector focused on lowering the computational cost.

AGAST – Adaptive and Generic Accelerated Segment Test, a corner detector based on FAST. Unlike FAST, AGAST does not have to be trained for a specific scene, but it dynamically adapts to the environment while processing an image [12].

GFTT – A detector focused on selecting features relevant to motion tracking by analyzing the amount of information they provide for that particular task [10].

BRIEF – Binary Robust Independent Elementary Features [14] is a descriptor that describes an image area using a number of intensity comparisons of random pixel pairs. It is saved as a binary string, which reduces the computational complexity of the subsequent matching.

ORB – Oriented FAST and Rotated BRIEF [13] is another attempt to achieve a scale and rotation invariant BRIEF, as a computationally efficient alternative to SIFT and SURF. It uses a modified version of the FAST detector to achieve low computational cost, computing orientation information in the process.

BRISK – Binary Robust Invariant Scalable Keypoints [15] is a scale and rotation invariant version of BRIEF, but unlike BRIEF, it uses a deterministic comparison pattern.

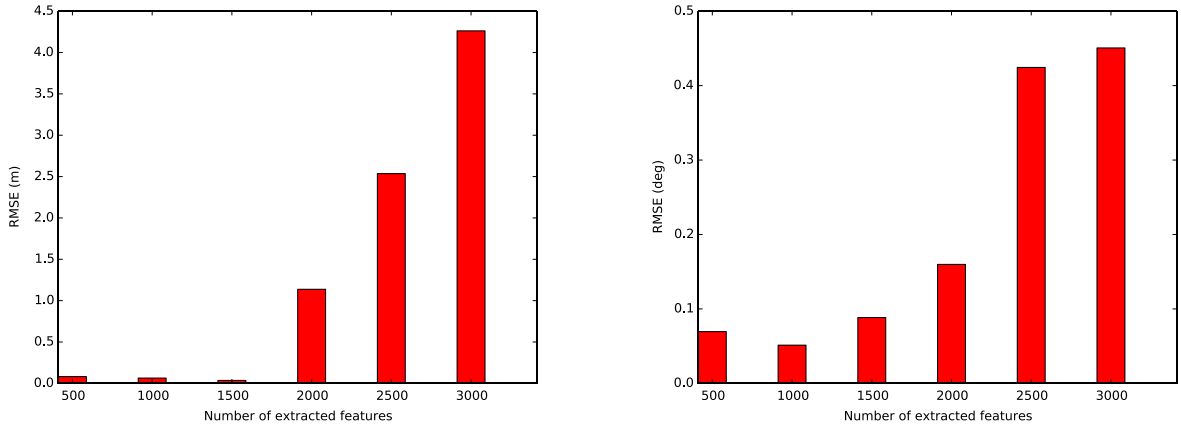
LATCH – Learned Arrangements of Three Patch Codes is a binary descriptor extractor which compute each value descriptor vector through the comparison of patches instead of solely pixels as BRIEF or BRISK extractors. By comparing patches, visual information with more spatial information support is considered for each of the descriptor's bits, and their values are therefore less sensitive to noise [16].

Given the high computational cost of SIFT and SURF feature extractors, they are not considered here, since the system is expected to run in real time. The ORB descriptor relies on its own detector (ORB). For each of the BRIEF, BRISK and LATCH descriptors, the combination with GFTT, FAST, AGAST and STAR detectors are considered. This amounts to thirteen detector–descriptor pairs that are evaluated in the present work.

The experiments are performed over the KITTI Vision Benchmark Suite [42], which provides a reliable ground truth for several training sequences. The stereo camera mounted on the front has a 60 cm baseline and a resolution of 1344×391 pixels and runs at a frame rate of 10 Hz. The training sequences sums up to 23.000 stereo frames. Results are computed over all the training sequences, except for sequence 01 which records a car driving in a highway at high speed, together with a low feature scene, rendering the system ill-conditioned for visual odometry.

The quality of each feature extractor is measured directly as the error 6.1 committed when running the S-PTAM system over the sequences with that particular configuration, with respect to the provided ground truth. In Table A.6, the parameters used for each feature extractor are detailed. To make the drift produced by each feature extractor observable, the loop closure module was disabled for this set of experiments.

An important requirement for every feature extractor is to track the camera pose in real time. First of all the extraction cost should be low. In Fig. 2 it can be seen how the different detectors and descriptor extractors perform in the context of the KITTI dataset. Note that the extraction time required by each method depends heavily on the processing power and resolution of the images. On the other hand, the number of extracted features also has a direct impact on the performance, since the map, and thus the amount of tracked points, scale with it. In Fig. 2a, GFTT is the most unstable detector, presenting time demanding outliers. In real-time operation, each outlier may cause the lost of a stereo frame. However, losing a few scattered frames does not compromise



(a) Translation RMSE errors. (b) Rotation RMSE errors.

Fig. 3. RMSE translation and rotation errors obtained by S-PTAM running in on-line fashion on sequence 04, with ORB/ORB as feature extractor. Six experiments were carried out changing the number of features to be extracted.

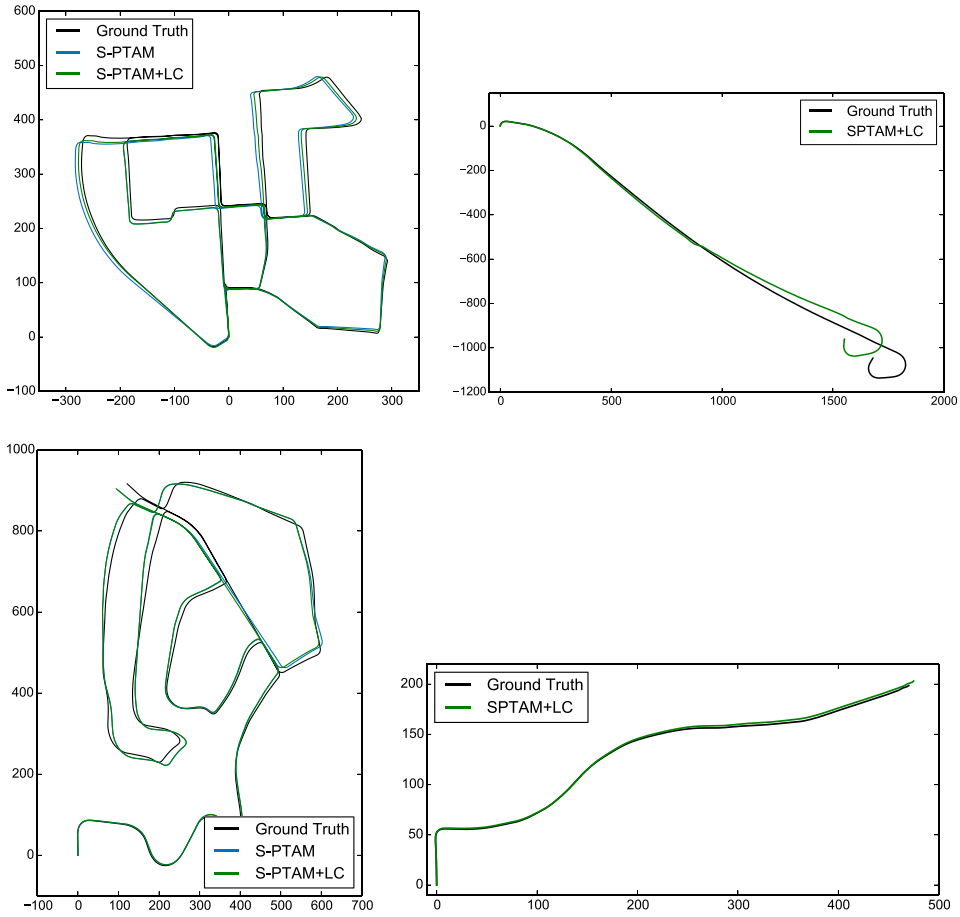


Fig. 4. Trajectories of the 00, 01, 02, 03 sequences.

S-PTAM's localization as we shall see in the on-line experiments presented in Section 6.3.

Analyzing the accuracy obtained by S-PTAM running in off-line mode (having enough time to process each stereo frame), it is possible to see which feature extractor achieves the best accuracy. Tables 1 and 2 show the RMSE translation and rotation errors achieved upper bounding the number of features to be extracted per frame. Upper bounds start at 500 features, given that almost all extractors fails to localize with lesser number of features. STAR/BRISK combination was the only one that was able to operate over all sequences with a 250 features upper bound. The ORB/ORB and GFTT/BRISK combinations outperform the others under the evaluated error metric. It is important to clarify that the number of features detected by GFTT remains around ∼500 features despite the selected upper bounds. This is determined by the detector's characteristics and its implementation. The tables show that in general at greater number of features extracted, a greater number of features are tracked, and therefore a better accuracy is obtained. In particular, ORB/ORB clearly presents the aforementioned tendency. Nevertheless, the performance of the

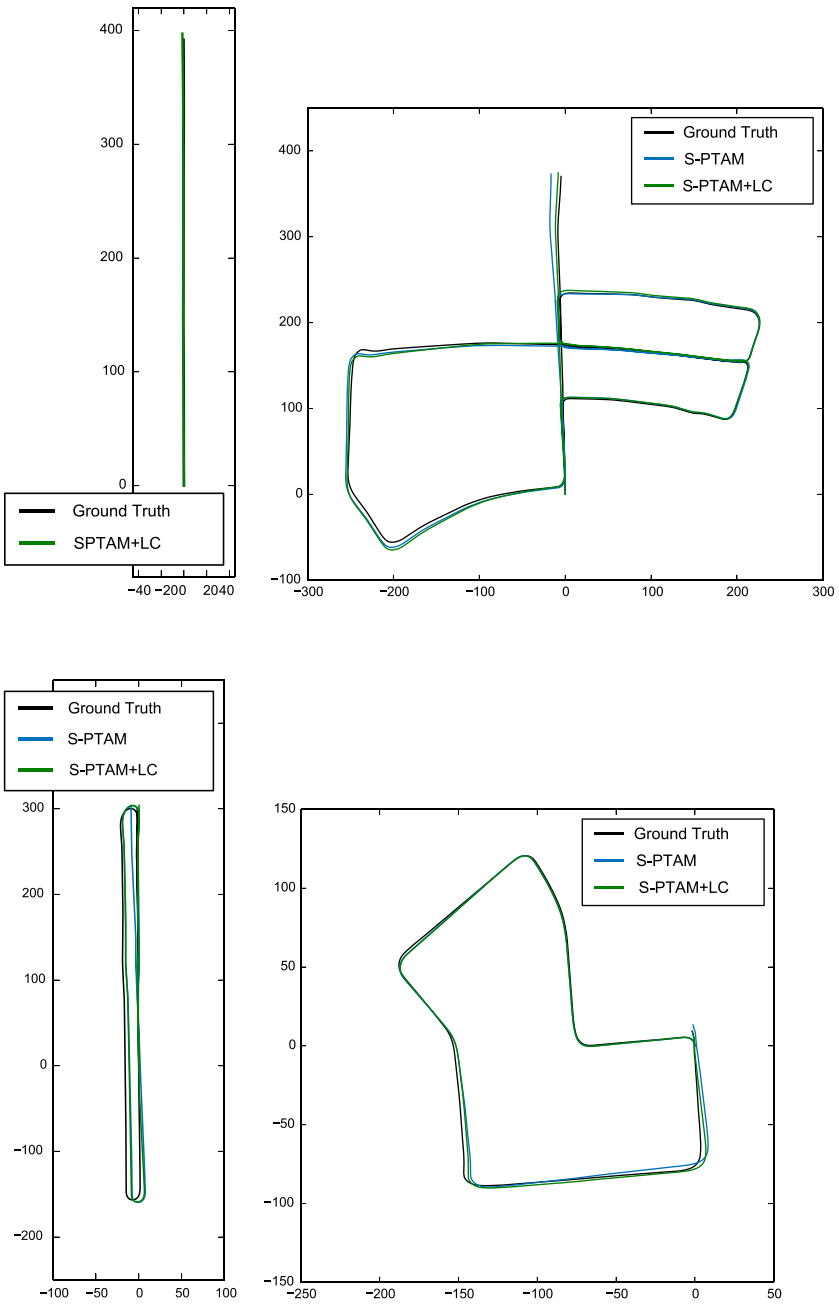


Fig. 5. Trajectories of the 04, 05, 06, 07 sequences.

system gets compromised if too many features are tracked. For real-time operation the number of features to be extracted should be carefully selected. Fig. 3 shows the translation and rotation RMSE errors obtained by S-PTAM running in on-line mode with ORB/ORB features. Errors decrease until ∼1500 features, and increases rapidly thereafter.

The experiments presented in the next section were carried out using the GFTT algorithm for the detection of features and BRISK was selected as the feature descriptor. At first this decision appears to be in conflict with previous works [62,63] on binary feature evaluation. In [63] the FAST/BRIEF extractor is recommended in the same context as the experiments conducted in the present work. However, it does not consider the complexity of further processing the huge amount of points extracted, and bases the detector choice solely on its speed. In [62], the BRIEF descriptor is preferred over BRISK, but BRISK is only paired with the AGAST detector. In [63] BRISK is not even considered.

6.3. Loop closure experiments

To assess the accuracy, robustness and computational cost of the S-PTAM system with the loop closure extension, the KITTI dataset and the Indoor Level 7 S-Block dataset [64] were used. They cover both outdoor large driving scenarios as well as indoor robotics respectively. The KITTI dataset, 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. This dataset presents dynamic objects, changing light conditions and fast camera motions. The Level 7 S-Block dataset corresponds to a wheeled robot moving around an office environment under artificial illumination conditions. The stereo camera mounted on the robot has a ∼30 cm baseline and a resolution of 1280 × 1960 pixels at a frame rate of 12 Hz. During the trajectory several loop closures are made

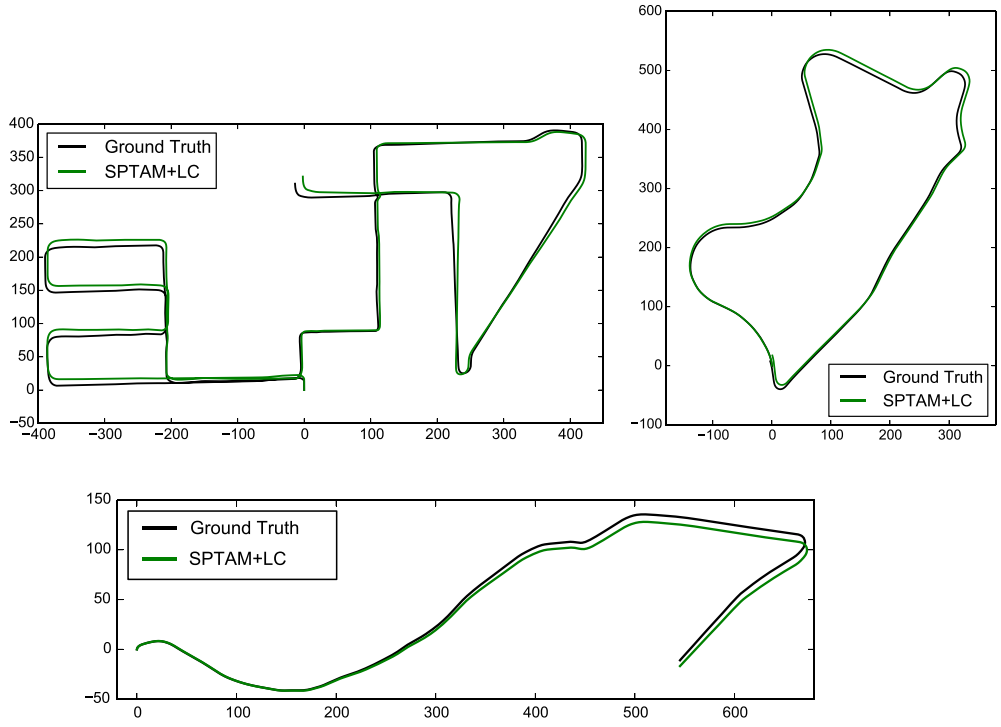


Fig. 6. Trajectories of the 08, 09, 10 sequences.

Table 1
Translation RMSE errors obtained for each feature extractor over all KITTI training sequences (except 01) limiting the number of features to be extracted. Good (small) relative error implies local consistency, sufficient for navigation.

	Features extracted per frame				
Extractor	500	1000	1500	2000	2500
AGAST / BRIEF	0.0601	0.0448	0.0441	0.0491	0.0452
AGAST / BRISK	0.0425	0.0361	0.0357	0.0357	0.0356
AGAST / LATCH	0.0887	0.0677	0.0796	0.1089	0.0793
FAST / BRIEF	0.0563	0.0511	0.0538	0.0473	0.0601
FAST / BRISK	0.0423	0.0342	0.034	0.0337	0.0345
FAST / LATCH	0.0906	0.0661	0.0891	0.1213	0.0727
GFTT / BRIEF	0.0333	0.0322	0.0322	0.0319	0.0317
GFTT / BRISK	**0.0301**	**0.0299**	**0.0299**	**0.0293**	0.0294
GFTT / LATCH	0.0443	0.0419	0.0419	0.0423	0.042
ORB / ORB	0.0462	0.0363	0.0321	0.0305	**0.0293**
STAR / BRIEF	0.0393	0.0368	0.037	0.0375	0.0371
STAR / BRISK	0.0449	0.0436	0.0415	0.0521	0.0529
STAR / LATCH	0.0589	0.0525	0.0496	0.0496	0.0495

Table 2
Rotation RMSE errors obtained for each feature extractor over all KITTI training sequences (except 01) limiting the number of features to be extracted. Good (small) relative error implies local consistency, sufficient for navigation.

	Features extracted per frame				
Extractor	500	1000	1500	2000	2500
AGAST / BRIEF	0.095	0.0833	0.0795	0.0835	0.0825
AGAST / BRISK	0.0881	0.0742	0.0768	0.0784	0.0767
AGAST / LATCH	0.119	0.0984	0.0976	0.0973	0.0955
FAST / BRIEF	0.0939	0.0851	0.0814	0.0813	0.0853
FAST / BRISK	0.0838	0.0764	0.0755	0.0737	0.0746
FAST / LATCH	0.1142	0.1001	0.0965	0.1002	0.0966
GFTT / BRIEF	0.0771	0.0745	0.0746	0.0744	0.0741
GFTT / BRISK	**0.0756**	**0.0741**	0.0741	0.0746	0.0738
GFTT / LATCH	0.0897	0.0863	0.0898	0.0866	0.0892
ORB / ORB	0.087	0.0762	**0.0724**	**0.0699**	**0.068**
STAR / BRIEF	0.0798	0.076	0.0747	0.0745	0.0759
STAR / BRISK	0.0848	0.08	0.0765	0.0768	0.0766
STAR / LATCH	0.102	0.0914	0.0895	0.0913	0.0915

over an extended period of time (more than 30 min). For all these experiments, a standard laptop with an Intel Core i7 @ 2.8 GHz processor and 16 GB RAM was used.

6.3.1. The KITTI benchmark suite

Figs. 4–6 show the performed trajectories estimated by S-PTAM with the loop closure extension compared with the ground truth. For those sequences where loops were detected (00, 02, 05, 06 and 07), a comparison with S-PTAM without loop closure is presented. Implemented methods for the loop detection and validation have shown to be robust as no false positives have occurred in any of the evaluated sequences. Fig. 7 shows the loops that were detected over the sequence 00, the Z-axis represents time and red lines link pairs of keyframes that were matched as positive loops. The loop correction proved to be able to operate without disrupting the tracking continuity. Figs. 8 and 9 show the absolute translation and rotation error respectively at each moment of the sequence 00. In such figures, absolute errors of the system with and without loop closure extension are presented. It can be seen that the first loop correction occurred after a significant period of time without any loops (where accumulated drift error increases substantially), significantly improves the global localization of the system. When a loop correction occurs, the translation error get adjusted to the values registered at the time that the place was first visited. In Fig. 8, between seconds 350 to 400, the car revisits a section previously mapped. It is interesting to note that the absolute error is not further reduced with higher numbers of detected loops. This is due to the accumulated error being already eliminated by the first loop closed in that segment. Error peaks in the figure correspond to areas with low texture or high-speed turns. During the ∼4 km trajectory followed by the car, the maximum absolute localization error was less than 15 m.

Table 3 shows the performance of the loop detection and geometric validation methods on sequences that presents loops in trajectory (00, 02, 05, 06, 07 and 09). Loop associations proposed in [65] were used as ground truth. The appearance-based loop

Table 3
Precision and recall results on KITTI sequences.

Sequence	#Loops	Appearance			Appearance + Geometric Validation		
		#Detections	%Precision	%Recall	#Validations	%Precision	%Recall
KITTI00	732	2747	13,14%	49,32%	45	100%	6,14%
KITTI02	234	3010	3,12%	40,17%	5	100%	2,14%
KITTI05	320	1596	12,4%	61,88%	28	100%	8,75%
KITTI06	269	412	24,03%	36,8%	16	100%	5,95%
KITTI07	13	434	2,07%	69,23%	1	100%	7,69%
KITTI09	17	836	0,60%	29,41%	0	/	0%

Table 4
Tracking phase average processing time.

Tracking phase	Time (ms)
Feature Extraction and description	31.53
Get Points (inside Frustum)	4.37
Matching	4.71
Pose Update	0.99
AddKeyFrame	13.62
Total	55.22

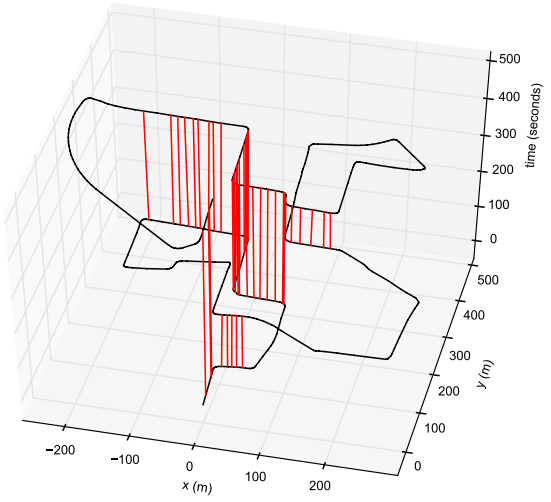


Fig. 7. Loop detections on sequence 00. Red lines link pairs of matched keyframes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

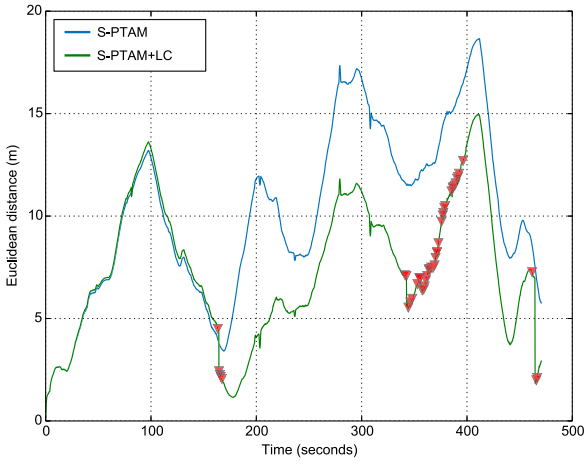


Fig. 8. Absolute translation error on sequence 00. Red markers show when a loop was validated. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

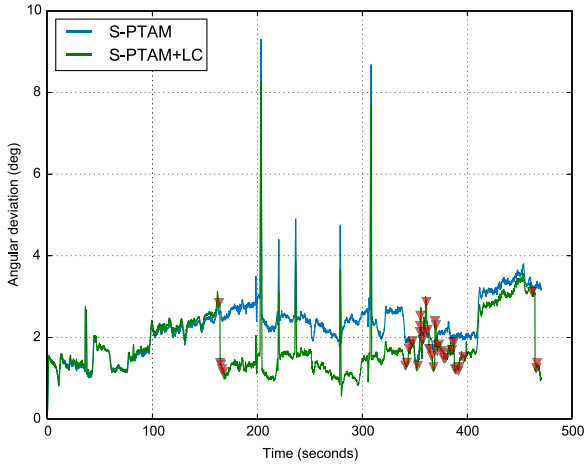


Fig. 9. Absolute rotation error on sequence 00. Red markers show when a loop was validated. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

detection is permissive, generating a high number of detections with a large percentage of false positives. In contrast, the geometric validation implemented rejects false positives with 100% precision at expense of a lower recall. The number of loops finally validated is proportionally low compared to the amount of loops defined in the ground truth. As we mention before, several loop corrections in close succession do not significantly improve the global localization, the method focuses on fewer loops with higher number of inlier correspondences. S-PTAM fails to detect a loop that occurs in the last 17 frames of sequence 09.

Table 4 shows the average temporal performance, measured for the costliest subroutines of the tracking process. Despite of the time consumed by the loop closure procedures, the tracking thread runs at ∼18 Hz.

6.3.2. Level 7 s-block dataset

Unlike the KITTI dataset, which presents a low number of widely separated loops, the Level 7 dataset features a high number of them. Besides testing S-PTAM in a different environment, it also allows to make a proper evaluation of the loop closure extension in terms of time requirements. In Fig. 10 the trajectory estimated by S-PTAM with and without the loop closure extension is presented along with the loops that have been validated. The loop validation process implemented shows to be robust and accurate, given that, even when the scene is highly repetitive no false positive loops are detected.

Figs. 11 and 12 show that the map update (outside the usage mapping window) and loop correction processes scale linearly with the number of keyframes. Note that the gaps between measurements indicate that there was no loop detected in that timespan. During the loop correction process, tracking and mapping threads are paused up to 4 ms. This allows the continuous growth

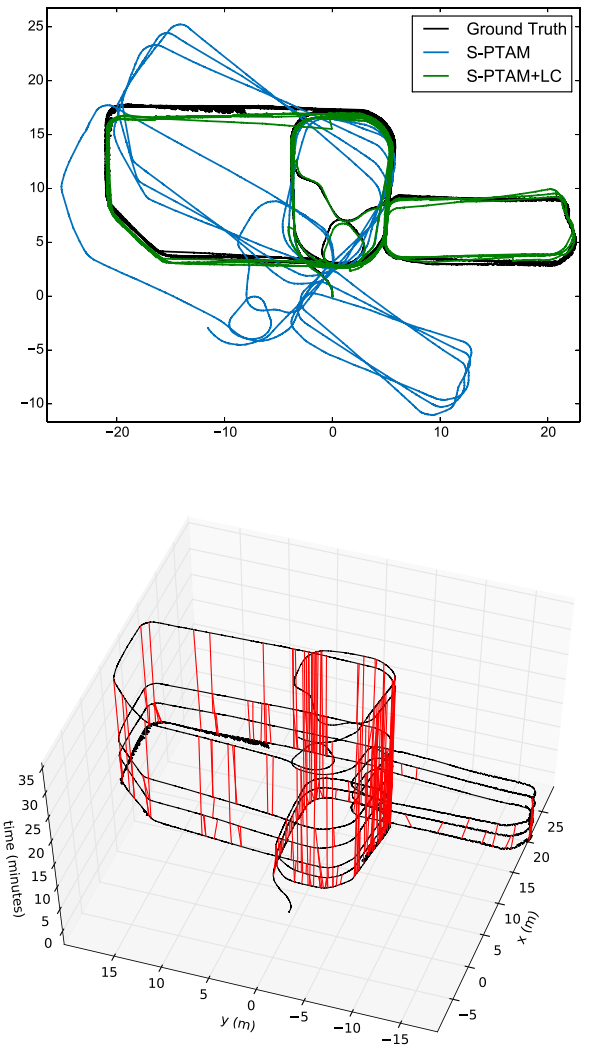


Fig. 10. Estimated trajectory and loops detected over time. Note that ground-truth information presents segments with little noise.

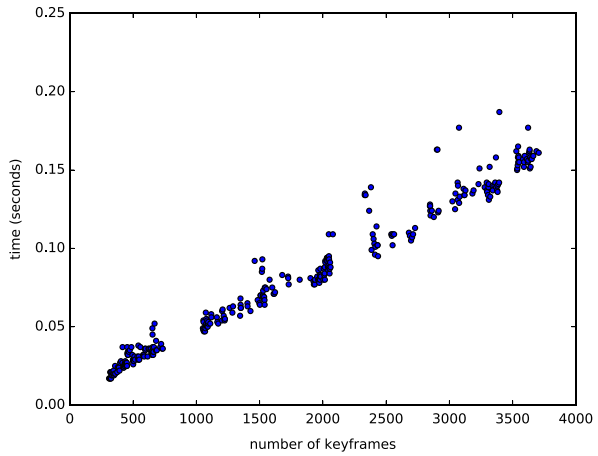


Fig. 11. Map update times (for keyframes outside the usage mapping window).

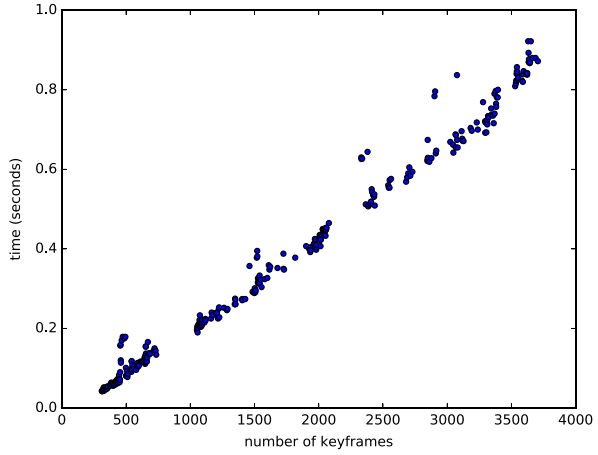


Fig. 12. Loop correction (initial loop correction and pose graph optimization) time over the number of keyframes in the map.

Table 5
Comparison of S-PTAM, ORB-SLAM2 and S-LSD-SLAM in KITTI Benchmark.

Method	Translation Error	Rotation Error
ORB-SLAM2	**1.15** %	0.0027 [deg/m]
S-PTAM	1.19%	**0.0025** [deg/m]
S-LSD-SLAM	1.20%	0.0033 [deg/m]

of the map along with an uninterrupted tracking, even during the loop closing process.

6.4. Comparison with other slam systems

This section aims to compare S-PTAM with other state-of-the-art SLAM systems.

6.4.1. The KITTI benchmark comparison

The KITTI benchmark presents an exhaustive comparison of several state-of-the-art SLAM systems in the context of outdoor driving scenarios. In the benchmark, the errors measured are a form of relative mean square errors (MSE), normalized over distances and velocities. See [42] for further details on how this errors are computed.

In Table 5, which is an excerpt of the ranking on the benchmark website [66], S-PTAM is compared to the stereo version of ORB-SLAM2 [37] and the S-LSD-SLAM [48] system. Both are state-of-the-art reference systems in the visual SLAM community. ORB-SLAM2 presents the best translation error whilst S-PTAM presents the best rotation error. The direct stereo SLAM system S-LSD-SLAM, performs worse than the feature-based ones in this dataset.

6.4.2. Level 7 s-block dataset

In this section, we compare ORB-SLAM2 and S-PTAM systems over the Level7 dataset. Fig. 13 shows the trajectory estimated by both systems; and Figures Figs. 14 and 15 present the absolute translation and rotation errors obtained. The figures show that S-PTAM and ORB-SLAM2 have comparable accuracy and both present similar error peaks around the same areas.

7. Conclusions

In this paper, we present a mature stereo SLAM system for robot localization called S-PTAM. S-PTAM 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 SLAM problem is heavily parallelized, separating tracking and map refinement routines, while minimizing inter-thread dependency. Moreover, to make the system scale better in large scale maps,

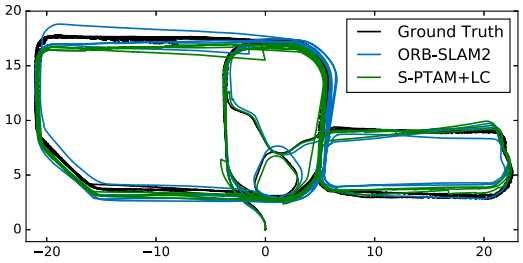


Fig. 13. Comparison between trajectories estimated by ORB-SLAM2 and S-PTAM. Note that ground-truth information presents segments with little noise.

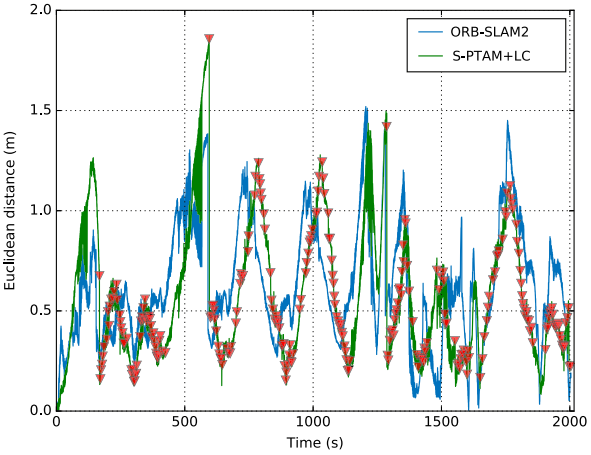


Fig. 14. Absolute translation error estimated by ORB-SLAM2 and S-PTAM. Note that ground-truth information presents segments with little noise.

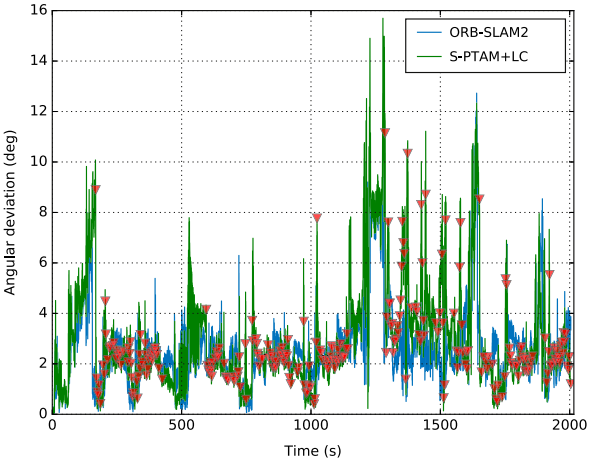


Fig. 15. Absolute rotation error estimated by ORB-SLAM2 and S-PTAM. Note that ground-truth information presents segments with little noise.

a loop closure module was developed. This extension was designed in order to does not disrupt tracking and local mapping threads, allowing to the system operates in real-time.

This work also assesses the impact of image feature extractors on the performance of S-PTAM in terms of pose accuracy and computational requirements. From this evaluation, we conclude that the GFTT key-point detector and BRISK descriptor combination gives a good trade-off between computation demanding and accuracy for real-time applications. Although the evaluation was performed using the S-PTAM system, the obtained results can be extrapolated to other stereo feature-based SLAM systems.

Table A.6
Parameters used for the feature detectors and descriptors in the evaluation of Section 6.2. The parameters not appearing in the list use the default value in the OpenCV 3 implementation. We used the Hamming distance as the metric for the descriptor similarity, as all of them are binary.

Detector/Descriptor	Parameter	Value
STAR	`responseThreshold`	20
FAST	`threshold`	60
AGAST	`threshold`	60
GFTT	`minDistance`	15.0
BRIEF	`hammingThreshold`	25
ORB	`nLevels`	1
	`hammingThreshold`	50
BRISK	`hammingThreshold`	100
LATCH	`hammingThreshold`	45
	`rotationInvariance`	false

The accuracy of the method was tested in public outdoor and indoor datasets, comparing results against the provided ground truth. The presented datasets presents different conditions, such as dynamic objects, changing light conditions and fast camera motions combined with low camera frame-rate. Furthermore, experiments were performed with simulated time to test the real-time performance of the system. Results indicate that the accuracy of S-PTAM is comparable to state-of-the-art approaches for mobile robot localization.

Although S-PTAM can deal with arbitrary camera motions, abrupt motion changes may produce localization failures. An approach to deal with this limitation is to feed the motion model with angular velocity and linear acceleration measurements provided by an inertial measurement unit (IMU). IMU integration into S-PTAM was explored in [67].

Acknowledgments

The research was supported by UBACYT project No. 20020130300035BA and the Program Missions Abroad VII of the Argentinian Ministry of Education under the project No. 41- ##-0091. It was also partially funded by the Spanish project DPI2015-67275, the Aragón regional project "Grupo DGA T04-FSE" and the University of Zaragoza via the project JIUZ-2015-TEC-03.

Appendix A. Parameters of the feature extractors

See Table A.6.

References

[1] J.A. Castellanos, J.M.M. Montiel, J. Neira, J.D. Tardos, The SPmap: a probabilistic framework for simultaneous localizationand map building, IEEE Trans. Robot. Autom. (ISSN: 1042-296X) 15 (5) (1999) 948–952. http://dx.doi.org/10.1109/70.795798.

[2] A.J. Davison, I.D. Reid, N.D. Molton, O. Stasse, MonoSLAM: Real-time single camera slam, IEEE Trans. Pattern Anal. Mach. Intell. (ISSN: 0162-8828) 29 (6) (2007) 1052–1067. http://dx.doi.org/10.1109/TPAMI.2007.1049.

[3] C. Kerl, J. Sturm, D. Cremers, Dense visual slam for RGB-D cameras, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2013, pp. 2100–2106. http://dx.doi.org/10.1109/IROS.2013.6696650.

[4] A. Concha, G. Loianno, V. Kumar, J. Civera, Visual-inertial direct SLAM, in: 2016 IEEE International Conference on Robotics and Automation, ICRA, 2016, pp. 1331–1338. http://dx.doi.org/10.1109/ICRA.2016.7487266.

[5] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, P. Furgale, Keyframe-based visual–inertial odometry using nonlinear optimization, Int. J. Robot. Res. (ISSN: 0278-3649) 34 (3) (2015) 314–334. http://dx.doi.org/10.1177/0278364914554813.

[6] T. Whelan, S. Leutenegger, R.F. Salas-Moreno, B. Glocker, A.J. Davison, Elastic Fusion: Dense slam without a pose graph,in: Proceedings of Robotics: Science and Systems, RSS, 2015. http://dx.doi.org/10.15607/RSS.2015.XI.001.

[7] D. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. (ISSN: 0920-5691) 60 (2) (2004) 91–110. http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.

[8] H. Bay, T. Tuytelaars, L. Van Gool, SURF: Speeded up robust features, in: Proceedings of the European Conference on Computer Vision, ECCV, in: Lecture Notes in Computer Science, vol. 3951, Springer, Berlin, Heidelberg, ISBN: 978-3-540-33832-1, 2006, pp. 404–417. http://dx.doi.org/10.1007/11744023_32.

[9] M. Agrawal, K. Konolige, M. Blas, Censure: center surround extremas for realtime feature detection and matching, in: Proceedings of the European Conference on Computer Vision, ECCV, in: Lecture Notes in Computer Science, vol. 5305, Springer, Berlin Heidelberg, ISBN: 978-3-540-88692-1, 2008, pp. 102–115. http://dx.doi.org/10.1007/978-3-540-88693-8_8.

[10] J. Shi, C. Tomasi, Good features to track, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 1994, pp. 593–600. http://dx.doi.org/10.1109/CVPR.1994.323794.

[11] E. Rosten, T. Drummond, Machine learning for high-speed corner detection, in: Proceedings of the European Conference on Computer Vision, ECCV, in: Lecture Notes in Computer Science, vol. 3951, Springer, Berlin, Heidelberg, ISBN: 978-3-540-33832-1, 2006, pp. 430–443. http://dx.doi.org/10.1007/11744023_34.

[12] E. Mair, G.D. Hager, D. Burschka, M. Suppa, G. Hirzinger, Adaptive and generic corner detection based on the accelerated segment test, in: K. Daniilidis, P. Maragos, N. Paragios (Eds.), Proceedings of the ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II, Springer, Berlin, Heidelberg, ISBN: 978-3-642-15552-9, 2010, pp. 183–196. http://dx.doi.org/10.1007/978-3-642-15552-9_14.

[13] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, ORB: an efficient alternative to sift or surf, in: Proceedings of the IEEE International Conference on Computer Vision, ICCV, 2011, pp. 2564–2571. http://dx.doi.org/10.1109/ICCV.2011.6126544.

[14] M. Calonder, V. Lepetit, C. Strecha, P. Fua, BRIEF: binary robust independent elementary features, in: Proceedings of the European Conference on Computer Vision, ECCV, in: Lecture Notes in Computer Science, vol. 6314, Springer, Berlin, Heidelberg, ISBN: 978-3-642-15560-4, 2010, pp. 778–792. http://dx.doi.org/10.1007/978-3-642-15561-1_56.

[15] S. Leutenegger, M. Chli, R.Y. Siegwart, BRISK: binary robust invariant scalable keypoints, in: Proceedings of the IEEE International Conference on Computer Vision, ICCV, in: ICCV'11, IEEE Computer Society, Washington, DC, USA, ISBN: 978-1-4577-1101-5, 2011, pp. 2548–2555. http://dx.doi.org/10.1109/ICCV.2011.6126542.

[16] G. Levi, T. Hassner, LATCH: Learned arrangements of three patch codes, Computing Research Repository, CoRR abs/1501.03719 (2015). http://arxiv.org/abs/1501.03719.

[17] G. Klein, D. Murray, Parallel tracking and mapping for small AR workspaces, in: Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, ISMAR, IEEE Computer Society, Washington, DC, USA, ISBN: 978-1-4244-1749-0, 2007, pp. 1–10. http://dx.doi.org/10.1109/ISMAR.2007.4538852.

[18] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, J.J. Berlles, Stereo parallel tracking and mapping for robot localization, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2015, pp. 1373–1378. http://dx.doi.org/10.1109/IROS.2015.7353546.

[19] J.A. Castellanos, J.M.M. Montiel, J. Neira, J.D. Tardós, Sensor influence in the performance of simultaneous mobile robot localization and map building, in: Proceedings of the 6th International Symposium on Experimental Robotics, Springer, London, ISBN: 978-1-84628-541-7, 2000, pp. 287–296. http://dx.doi.org/10.1007/BFb0119407.

[20] L. Iocchi, K. Konolige, M. Bajracharya, Visually realistic mapping of a planar environment with stereo, in: Experimental Robotics VII, Springer, 2001, pp. 521–532.

[21] S. Se, D. Lowe, J. Little, Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks, Int. J. Robot. Res. 21 (8) (2002) 735–758.

[22] A.J. Davison, D.W. Murray, Simultaneous localization and map-building using active vision, IEEE Trans. Pattern Anal. Mach. Intell. 24 (7) (2002) 865–880.

[23] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, FastSLAM: A factored solution to the simultaneous localization and mapping problem, in: Proceedings of the Eighteenth National Conference on Artificial Intelligence, American Association for Artificial Intelligence, Menlo Park, CA, USA, ISBN: 0-262-51129-0, 2002, pp. 593–598. http://dl.acm.org/citation.cfm?id=777092.777184.

[24] M. Montemerlo, S. Thrun, D. Roller, B. Wegbreit, FastSLAM 2.0: An Improved particle filtering algorithm for simultaneous localization and mapping that provably converges, in: Proceedings of the 18th International Joint Conference on Artificial Intelligence, in: IJCAI'03, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003, pp. 1151–1156. http://dl.acm.org/citation.cfm?id=1630659.1630824.

[25] R. Sim, P. Elinas, M. Griffin, Vision-based slam using the rao-blackwellised particle filter, in: In IJCAI Workshop on Reasoning with Uncertainty in Robotics, 2005, pp. 9–16.

[26] L.M. Paz, P. Piniés, J.D. Tardós, J. Neira, Large-scale 6-DOF SLAM with stereo-in-hand, IEEE Trans. Robot. (ISSN: 1552-3098) 24 (5) (2008) 946–957. http://dx.doi.org/10.1109/TRO.2008.2004637.

[27] J. Civera, A.J. Davison, J.M.M. Montiel, Inverse depth parametrization for monocular SLAM, IEEE Trans. Robot. (ISSN: 1552-3098) 24 (5) (2008) 932–945. http://dx.doi.org/10.1109/TRO.2008.2003276.

[28] J. Neira, J.D. Tardós, Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robot. Autom. (ISSN: 1042-296X) 17 (6) (2001) 890–897. http://dx.doi.org/10.1109/70.976019.

[29] L.M. Paz, J.D. Tardós, J. Neira, Divide and conquer: ekf slam in o(n), IEEE Trans. Robot. 24 (5) (2008) 1107–1120. http://dx.doi.org/10.1109/TRO.2008.2004639. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4631503&isnumber=4663225.

[30] S.J. Julier, J.K. Uhlmann, A counter example to the theory of simultaneous localization and map building, in: Proceedings of the IEEE International Conference on Robotics and Automation (IROS), vol. 4, 2001, pp. 4238–4243. http://dx.doi.org/10.1109/ROBOT.2001.933280.

[31] H. Strasdat, J.M.M. Montiel, A.J. Davison, Editors choice article: Visual SLAM: Why filter? Image Vis. Comput. (ISSN: 0262-8856) 30 (2) (2012) 65–77. http://dx.doi.org/10.1016/j.imavis.2012.02.009.

[32] B. Triggs, P.F. McLauchlan, R.I. Hartley, A.W. Fitzgibbon, Bundle adjustment — A modern synthesis, in: Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings, Springer, Berlin, Heidelberg, ISBN: 978-3-540-44480-0, 2000, pp. 298–372. http://dx.doi.org/10.1007/3-540-44480-7_21.

[33] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, P. Sayd, Generic and real-time structure from motion using local bundle adjustment, Image Vis. Comput. (ISSN: 0262-8856) 27 (8) (2009) 1178–1193. http://dx.doi.org/10.1016/j.imavis.2008.11.006. http://www.sciencedirect.com/science/article/pii/S0262885608002436.

[34] K. Konolige, M. Agrawal, FrameSLAM: From bundle adjustment to real-time visual mapping, IEEE Trans. Robot. (ISSN: 1552-3098) 24 (5) (2008) 1066–1077. http://dx.doi.org/10.1109/TRO.2008.2004832.

[35] C. Mei, G. Sibley, M. Cummins, P. Newman, I. Reid, RSLAM: a system for large-scale mapping in constant-time using stereo, Int. J. Comput. Vis. (ISSN: 0920-5691) 94 (2) (2011) 198–214. http://dx.doi.org/10.1007/s11263-010-0361-7.

[36] H. Strasdat, A.J. Davison, J.M.M. Montiel, K. Konolige, Double window optimisation for constant time visual slam, in: Proceedings of the IEEE International Conference on Computer Vision, ICCV, 2011, pp. 2352–2359. http://dx.doi.org/10.1109/ICCV.2011.6126517.

[37] R. Mur-Artal, J.D. Tardós, ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras, CoRR abs/1610.06475 (2016). http://arxiv.org/abs/1610.06475.

[38] D. Galvez-Lopez, J.D. Tardós, Bags of binary words for fast place recognition in image sequences, IEEE Trans. Robot. (ISSN: 1552-3098) 28 (5) (2012) 1188–1197. http://dx.doi.org/10.1109/TRO.2012.2197158.

[39] H. Strasdat, J.M.M. Montiel, A.J. Davison, Scale drift-aware large scale monocular slam, in: Y. Matsuoka, H.F. Durrant-Whyte, J. Neira (Eds.), Proceedings of Robotics: Science and Systems, 2010. http://dx.doi.org/10.15607/RSS.2010.VI.010.

[40] M. Irani, P. Anandan, About direct methods, in: Proceedings of the International Workshop on Vision Algorithms: Theory and Practice, in: Proceedings of the IEEE International Conference on Computer Vision, ICCV, Springer-Verlag, London, UK, ISBN: 3-540-67973-1, 2000, pp. 267–277. http://dl.acm.org/citation.cfm?id=646271.685624.

[41] R. Mur-Artal, J.D. Tardós, Probabilistic semi-dense mapping from highly accurate feature-based monocular slam, in: Proceedings of Robotics: Science and Systems, RSS, 2015. http://dx.doi.org/10.15607/RSS.2015.XI.041.

[42] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: The KITTI dataset, Int. J. Robot. Res. (ISSN: 0278-3649) 32 (11) (2013) 1231–1237. http://dx.doi.org/10.1177/0278364913491297.

[43] J. Engel, T. Schöps, D. Cremers, LSD-SLAM: Large-scale direct monocular SLAM, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Proceedings of the European Conference on Computer Vision, ECCV, Springer International Publishing, Cham, ISBN: 978-3-319-10605-2, 2014, pp. 834–849. http://dx.doi.org/10.1007/978-3-319-10605-2_54.

[44] A.I. Comport, E. Malis, P. Rives, Real-time quadrifocal visual odometry, Int. J. Robot. Res. 29 (2–3) (2010) 245–266.

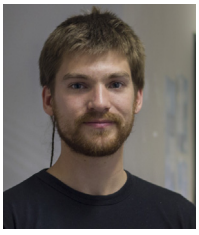
[45] T. Tykkälä, A. Comport, A dense structure model for image based stereo slam, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2011, pp. 1758–1763. http://dx.doi.org/10.1109/ICRA.2011.5979805.

[46] R.A. Newcombe, S.J. Lovegrove, A.J. Davison, DTAM: Dense tracking and mapping in real-time, in: Proceedings of the IEEE International Conference on Computer Vision, ICCV, in: ICCV '11, IEEE Computer Society, Washington, DC, USA, ISBN: 978-1-4577-1101-5, 2011, pp. 2320–2327. http://dx.doi.org/10.1109/ICCV.2011.6126513.

[47] A. Concha, W. Hussain, L. Montano, J. Civera, Incorporating scene priors to dense monocular mapping, Auton. Robots (ISSN: 1573-7527) 39 (3) (2015) 279–292. http://dx.doi.org/10.1007/s10514-015-9465-9.

[48] J. Engel, J. Stückler, D. Cremers, Large-scale direct slam with stereo cameras, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, IROS, IEEE, 2015, pp. 1935–1942. http://dx.doi.org/10.1109/IROS.2015.7353631.
[49] V.S. Varadarajan, Lie groups, lie algebras, and their representations, in: Graduate Texts in Mathematics, Springer-Verlag, New York, ISBN: 9780387909691, 1984. http://dx.doi.org/10.1007/978-1-4612-1126-6.
[50] R. Mur-Artal, J.D. Tardós, Fast relocalisation and loop closing in keyframe-based slam, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA, 2014, pp. 846–853. http://dx.doi.org/10.1109/ICRA.2014.6906953.
[51] L. Kneip, D. Scaramuzza, R. Siegwart, A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2011, pp. 2969–2976. http://dx.doi.org/10.1109/CVPR.2011.5995464.
[52] L. Kneip, H. Li, Y. Seo, Upnp: an optimal o(n) solution to the absolute pose problem with universal applicability, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Proceedings of the European Conference on Computer Vision, ECCV, in: Lecture Notes in Computer Science, vol. 8689, Springer International Publishing, ISBN: 978-3-319-10589-5, 2014, pp. 127–142. http://dx.doi.org/10.1007/978-3-319-10590-1_9.
[53] M.F. Fallon, H. Johannsson, M. Kaess, J.J. Leonard, The mit Stata Center Dataset, The International Journal of Robotics Research 32 (14) (2013) 1695–1699. http://dx.doi.org/10.1177/0278364913509035.
[54] J.-L. Blanco, F.-A. Moreno, J. González, A collection of outdoor robotic datasets with centimeter-accuracy ground truth, Auton. Robots (ISSN: 0929-5593) 27 (4) (2009) 327–351. http://dx.doi.org/10.1007/s10514-009-9138-7. http://www.mrpt.org/Paper:Malaga_Dataset_2009.
[55] L. Kneip, P. Furgale, OpenGV: A unified and generalized approach to real-time calibrated geometric vision, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA, 2014, pp. 1–8. http://dx.doi.org/10.1109/ICRA.2014.6906582.
[56] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard, G2o: a general framework for graph optimization, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA, 2011, pp. 3607–3613. http://dx.doi.org/10.1109/ICRA.2011.5979949.
[57] N. Sünderhauf, P. Protzel, Switchable constraints for robust pose graph SLAM, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2012, pp. 1879–1884. http://dx.doi.org/10.1109/IROS.2012.6385590.
[58] N. Sünderhauf, http://www.openslam.org/vertigo.
[59] F. Dellaert, M. Kaess, Square Root SAM: Simultaneous localization and mapping via square root information smoothing, Int. J. Robot. Res. 25 (12) (2006) 1181–1203. http://dx.doi.org/10.1177/0278364906072768. http://ijr.sagepub.com/content/25/12/1181.abstract.
[60] R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, A. Kleiner, On measuring the accuracy of SLAM algorithms, Auton. Robots (ISSN: 1573-7527) 27 (4) (2009) 387–407. http://dx.doi.org/10.1007/s10514-009-9155-6.
[61] R. Smith, M. Self, P. Cheeseman, Estimating uncertain spatial relationships in robotics, in: Proceedings of the 1987 IEEE International Conference on Robotics and Automation, vol. 4, 1987, p. 850. http://dx.doi.org/10.1109/ROBOT.1987.1087846.
[62] J. Heinly, E. Dunn, J.-M. Frahm, Comparative Evaluation of Binary Features, in: A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, C. Schmid (Eds.), Proceedings of the ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II, Springer, Berlin, Heidelberg, ISBN: 978-3-642-33709-3, 2012, pp. 759–773. http://dx.doi.org/10.1007/978-3-642-33709-3_54.
[63] A. Schmidt, M. Kraft, M. Fularz, Z. Domagała, Comparative assessment of point feature detectors in the context of robot navigation, J. Automation Mob. Robot. Intell. Syst. 7 (1) (2013) 11–20.
[64] L. Murphy, T. Morris, U. Fabrizi, M. Warren, M. Milford, B. Upcroft, M. Bosse, P. Corke, Experimental comparison of odometry approaches, in: J.P. Desai, G. Dudek, O. Khatib, V. Kumar (Eds.), Experimental Robotics, in: Springer Tracts in Advanced Robotics, vol. 88, Springer International Publishing, ISBN: 978-3-319-00064-0, 2013, pp. 877–890. http://dx.doi.org/10.1007/978-3-319-00065-7_58.
[65] R. Arroyo, P.F. Alcantarilla, L.M. Bergasa, J.J. Yebes, S. Bronte, Fast and effective visual place recognition using binary codes and disparity information, in: Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2014, pp. 3089–3094. http://dx.doi.org/10.1109/IROS.2014.6942989.
[66] The KITTI Vision Benchmark Suite, http://www.cvlibs.net/datasets/kitti/eval_odometry.php.
[67] T. Fischer, T. Pire, P. Čížek, P. De Cristóforis, J. Faigl, Stereo vision-based localization for hexapod walking robots operating in rough terrains, in: Intelligent Robots and Systems, IROS, 2016 IEEE/RSJ International Conference on, IEEE, 2016, pp. 2492–2497.

Taihú Pire was born in Rosario, Argentina, in 1986. He received the computer science degree in 2010 from the National University of Rosario in Argentina. He is currently a Ph.D. student and assistant teacher at the University of Buenos Aires in Argentina. Currently, his research interests are in develop new Visual SLAM algorithms.



Thomas Fischer was born in Lomé, Togo, in 1988. He receive his master degree in Computer Science in 2013 from the University of Buenos Aires. Since then he works as a full time Ph.D. student for CONICET at the Laboratory of Robotics and Embedded Systems from the University of Buenos Aires, and also as a Teacher Assistant at the same university. Currently, his research interests include camera vision, visual SLAM, autonomous real time navigation and exploration, and hexapod platforms.

Gastón Castro was born in Buenos Aires, Argentina, in 1989. Computer science student at the University of Buenos Aires, with a master thesis subject on loop closure. Applied for Ph.D. student position at the Laboratory of Robotics and Embedded Systems from the University of Buenos Aires. Teacher Assistant at the same university. His research interests include computer vision, visual SLAM methods, loop closure and robotics.

Pablo De Cristóforis received the Ph.D. degree in Computer Science from the University of Buenos Aires, Argentina in 2013. He is currently a Research Assistant at the National Council of Scientific and Technological Research (CONICET), Argentina. His research interests include autonomous vision-based navigation and exploration, visual SLAM and 3D vision algorithms for mobile robotics.

Javier Civera was born in Barcelona, Spain, in 1980. He received the industrial engineering degree in 2004 and the Ph.D. degree in 2009, both from the University of Zaragoza in Spain. He is currently an Associate Professor at the University of Zaragoza, where he teaches courses in computer vision, machine learning and artificial intelligence. He has participated in several EUfunded, national and technology transfer projects related with vision and robotics and has been funded for research visits to Imperial College (London) and ETH (Zrich). He has coauthored around 30 publications in top conferences and journals, receiving around 1750 references (GoogleScholar). Currently, his research interests are in the use of 3D vision, distributed architectures and learning algorithms to produce robust and real-time vision technologies for robotics, wearables and AR applications.

Julio C. Jacobo-Berlles received the Eng. degree in electronics and the Ph.D. degree in computer science from the University of Buenos Aires, Buenos Aires, Argentina, in 1983 and 2005, respectively. Currently, he is an Adjoint Professor at the Computer Science Department, Faculty of Exact and Natural Sciences, University of Buenos Aires. His research interests include microwave remote sensing, image processing, computer vision, and robotics.