ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras

Raúl Mur-Artal and Juan D. Tardós

***Abstract*—We present ORB-SLAM2, a complete simultaneous localization and mapping (SLAM) system for monocular, stereo and RGB-D cameras, including map reuse, loop closing, and relocalization capabilities. The system works in real time on standard central processing units in a wide variety of environments from small hand-held indoors sequences, to drones flying in industrial environments and cars driving around a city. Our back-end, based on bundle adjustment with monocular and stereo observations, allows for accurate trajectory estimation with metric scale. Our system includes a lightweight localization mode that leverages visual odometry tracks for unmapped regions and matches with map points that allow for zero-drift localization. The evaluation on 29 popular public sequences shows that our method achieves state-of-the-art accuracy, being in most cases the most accurate SLAM solution. We publish the source code, not only for the benefit of the SLAM community, but with the aim of being an out-of-the-box SLAM solution for researchers in other fields.**

***Index Terms*—Localization, mapping, RGB-D, simultaneous localization and mapping (SLAM), stereo.**

I. Introduction

Simultaneous localization and mapping (SLAM) has been a hot research topic in the last two decades in the computer vision and robotics communities, and has recently attracted the attention of high-technological companies. SLAM techniques build a map of an unknown environment and localize the sensor in the map with a strong focus on real-time operation. Among the different sensor modalities, cameras are cheap and provide rich information of the environment that allows for robust and accurate place recognition. Therefore, visual SLAM solutions, in which the main sensor is a camera, are of major interest. Place recognition is a key module of a SLAM system to close loops (i.e., detect when the sensor returns to a mapped area and correct the accumulated error in exploration) and to relocalize the camera after a tracking failure, due to occlusion or aggressive motion, or at system reinitialization.

Manuscript received October 20, 2016; revised March 3, 2017; accepted April 19, 2017. Date of publication June 12, 2017; date of current version October 2, 2017. This paper was recommended for publication by Associate Editor J. Piater and Editor T. Murphey upon evaluation of the reviewers' comments. This work was supported in part by the Spanish government under Project DPI2015-67275, in part by the Aragón regional government under Project DGA T04-FSE, and in part by the Ministerio de Educación Scholarship FPU13/04175. *(Corresponding author: Raúl Mur-Artal.)*

R. Mur-Artal was with the Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, 50018 Zaragoza, Spain, until January 2017. He is currently with Oculus Research, Redmond, WA 98052 USA (e-mail: raul.murartal@oculus.com).

J. D. Tardós is with the Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, 50018 Zaragoza, Spain (e-mail: tardos@unizar.es).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org.

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

Digital Object Identifier 10.1109/TRO.2017.2705103

Visual SLAM can be performed by using just a monocular camera, which is the cheapest and smallest sensor setup. However, as depth is not observable from just one camera, the scale of the map and estimated trajectory is unknown. In addition, the system bootstrapping require multiview or filtering techniques to produce an initial map as it cannot be triangulated from the very first frame. Last but not least, monocular SLAM suffers from scale drift and may fail if performing pure rotations in exploration. By using a stereo or an RGB-D camera, all these issues are solved and allow for the most reliable visual SLAM solutions.

In this paper, we build on our monocular ORB-SLAM [1] and propose ORB-SLAM2 with the following contributions:

1) the first open-source[1] SLAM system for monocular, stereo, and RGB-D cameras, including loop closing, relocalization, and map reuse;
2) our RGB-D results show that by using bundle adjustment (BA), we achieve more accuracy than state-of-the-art methods based on iterative closest point (ICP) or photometric and depth error minimization;
3) by using close and far stereo points and monocular observations, our stereo results are more accurate than the state-of-the-art direct stereo SLAM;
4) a lightweight localization mode that can effectively reuse the map with mapping disabled.

Fig. 1 shows examples of ORB-SLAM2 output from stereo and RGB-D inputs. The stereo case shows the final trajectory and sparse reconstruction of the sequence 00 from the KITTI dataset [2]. This is an urban sequence with multiple loop closures that ORB-SLAM2 was able to successfully detect. The RGB-D case shows the keyframe poses estimated in sequence fr1_room from the TUM RGB-D Dataset [3], and a dense pointcloud, rendered by backprojecting sensor depth maps from the estimated keyframe poses. Note that our SLAM does not perform any fusion like KinectFusion [4] or similar, but the good definition indicates the accuracy of the keyframe poses. More examples are shown on the attached video.

In the rest of this paper, we discuss related work in Section II, we describe our system in Section III, then present the evaluation results in Section IV, and end with the conclusion in Section V.

II. Related Work

In this section, we discuss related work on stereo and RGB-D SLAM. Our discussion, as well as the evaluation in Section IV is focused only on SLAM approaches.

A. Stereo SLAM

A remarkable early stereo SLAM system was the work of Paz *et al.* [5]. Based on conditionally independent divide and conquer extended-Kalman-filter SLAM, it was able to operate in larger environments

[1] https://github.com/raulmur/ORB_SLAM2

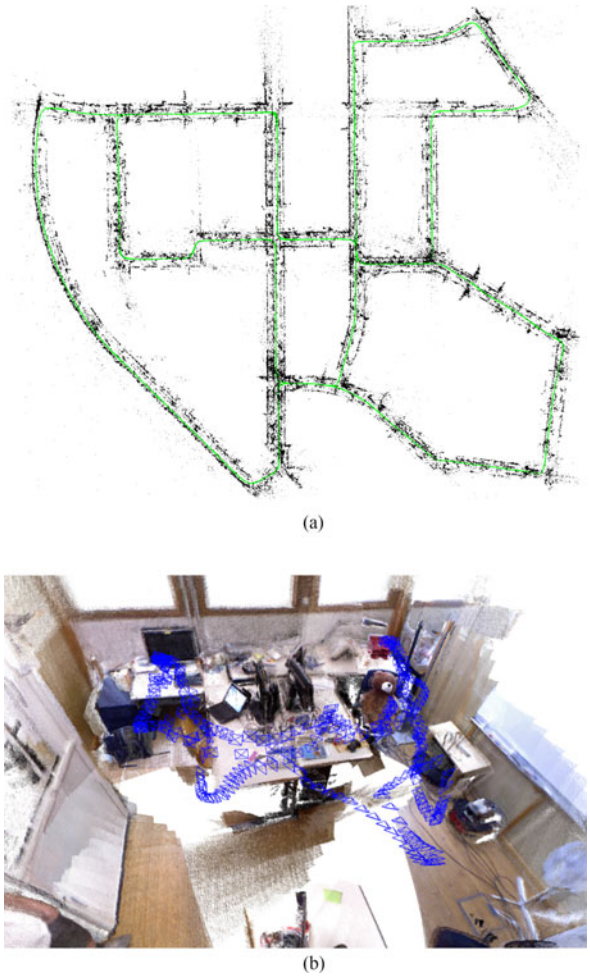

Fig. 1. ORB-SLAM2 processes stereo and RGB-D inputs to estimate camera trajectory and build a map of the environment. The system is able to close loops, relocalize, and reuse its map in real time on standard CPUs with high accuracy and robustness. (a) Stereo input: Trajectory and sparse reconstruction of an urban environment with multiple loop closures. (b) RGB-D input: Keyframes and dense pointcloud of a room scene with one loop closure. The pointcloud is rendered by backprojecting the sensor depth maps from estimated keyframe poses. No fusion is performed.

than other approaches at that time. Most importantly, it was the first stereo SLAM exploiting both close and far points (i.e., points whose depth cannot be reliably estimated due to little disparity in the stereo camera), using an inverse depth parameterization [6] for the latter. They empirically showed that points can be reliably triangulated if their depth is less than $\sim$40 times the stereo baseline. In this paper, we follow this strategy of treating in a different way *close* and *far* points, as explained in Section III-A.

Most modern stereo SLAM systems are keyframe based [7] and perform BA optimization in a local area to achieve scalability. The work of Strasdat *et al.* [8] performs a joint optimization of BA (point-pose constraints) in an inner window of keyframes and pose graph (pose-pose constraints) in an outer window. By limiting the size of these windows, the method achieves constant time complexity, at the expense of not guaranteeing global consistency. The relative simultaneous localisation and mapping system (RSLAM) of Mei *et al.* [9] uses a relative representation of landmarks and poses and performs relative BA in an active area, which can be constrained for constant time. RSLAM is able to close loops, which allow to expand active areas at both sides of a loop, but global consistency is not enforced. The recent S-PTAM by Pire *et al.* [10] performs local BA, however it lacks large loop closing. Similar to these approaches, we perform BA in a local set of keyframes so that the complexity is independent of the map size and we can operate in large environments. However, our goal is to build a globally consistent map. When closing a loop, our system aligns first both sides, similar to RSLAM so that the tracking is able to continue localizing using the old map, and then, performs a pose-graph optimization that minimizes the drift accumulated in the loop, followed by full BA.

The recent stereo large-scale direct SLAM (LSD-SLAM) of Engel *et al.* [11] is a semidense direct approach that minimizes photometric error in image regions with high gradient. Not relying on features, the method is expected to be more robust to motion blur or poorly textured environments. However, as a direct method, its performance can be severely degraded by unmodeled effects like rolling shutter or non-Lambertian reflectance.

B. RGB-D SLAM

One of the earliest and most famed RGB-D SLAM systems was the KinectFusion of Newcombe *et al.* [4]. This method fused all depth data from the sensor into a volumetric dense model that is used to track the camera pose using ICP. This system was limited to small workspaces due to its volumetric representation and the lack of loop closing. Kintinuous by Whelan *et al.* [12] was able to operate in large environments by using a rolling cyclical buffer and included loop closing using place recognition and pose graph optimization.

Probably the first popular open-source system was the RGB-D SLAM of Endres *et al.* [13]. This is a feature-based system, whose front end computes frame-to-frame motion by feature matching and ICP. The back end performs pose-graph optimization with loop closure constraints from a heuristic search. Similarly, the back end DVO-SLAM by Kerl *et al.* [14] optimizes a pose graph where keyframe-to-keyframe constraints are computed from a visual odometry that minimizes both photometric and depth error. DVO-SLAM also searches for loop candidates in a heuristic fashion over all previous frames, instead of relying on place recognition.

The recent ElasticFusion of Whelan *et al.* [15] builds a surfel-based map of the environment. This is a map-centric approach that forget poses and performs loop closing applying a nonrigid deformation to the map, instead of a standard pose-graph optimization. The detailed reconstruction and localization accuracy of this system is impressive, but the current implementation is limited to room-size maps as the complexity scales with the number of surfels in the map.

As proposed by Strasdat *et al.* [8], our ORB-SLAM2 uses depth information to synthesize a stereo coordinate for extracted features on the image. This way our system is agnostic of the input being stereo or RGB-D. Differently to all above methods our back end is based on bundle adjustment and builds a globally consistent sparse reconstruction. Therefore our method is lightweight and works with standard central processing units (CPUs). Our goal is long term and globally consistent localization instead of building the most detailed dense reconstruction. However from the highly accurate keyframe poses one could fuse depth maps and get accurate reconstruction on-the-fly in a local area or post-process the depth maps from all keyframes after a full BA and get an accurate 3-D model of the whole scene.

III. ORB-SLAM2

ORB-SLAM2 for stereo and RGB-D cameras is built on our monocular feature-based ORB-SLAM [1], whose main components are summarized here for reader convenience. A general overview of the system

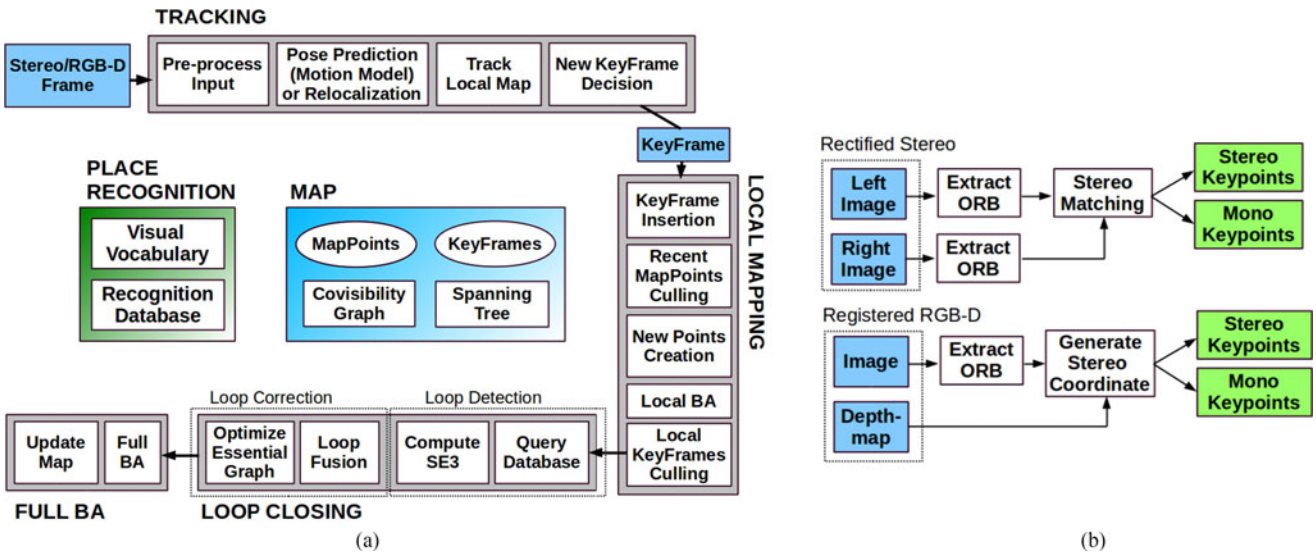

Fig. 2. ORB-SLAM2 is composed of three main parallel threads: tracking, local mapping, and loop closing, which can create a fourth thread to perform full BA after a loop closure. The tracking thread preprocesses the stereo or RGB-D input so that the rest of the system operates independently of the input sensor. Although it is not shown in this figure, ORB-SLAM2 also works with a monocular input as in [1]. (a) System threads and modules. (b) Input preprocessing.

is shown in Fig. 2. The system has three main parallel threads: 1) the tracking to localize the camera with every frame by finding feature matches to the local map and minimizing the reprojection error applying motion-only BA; 2) the local mapping to manage the local map and optimize it, performing local BA; and 3) the loop closing to detect large loops and correct the accumulated drift by performing a pose-graph optimization. This thread launches a fourth thread to perform full BA after the pose-graph optimization, to compute the optimal structure and motion solution.

The system has embedded a place recognition module based on DBoW2 [16] for relocalization, in case of tracking failure (e.g., an occlusion) or for reinitialization in an already mapped scene, and for loop detection. The system maintains a covisibiliy graph [8] that links any two keyframes observing common points and a minimum spanning tree connecting all keyframes. These graph structures allow to retrieve local windows of keyframes so that tracking and local mapping operate locally, allowing to work on large environments, and serve as structure for the pose-graph optimization performed when closing a loop.

The system uses the same ORB features [17] for tracking, mapping, and place recognition tasks. These features are robust to rotation and scale and present a good invariance to camera autogain and autoexposure, and illumination changes. Moreover, they are fast to extract and match allowing for real-time operation and show good precision/recall performance in bag-of-word place recognition [18].

In the rest of this section, we present how stereo/depth information is exploited and which elements of the system are affected. For a detailed description of each system block, we refer the reader to our monocular publication [1].

A. Monocular, Close Stereo, and Far Stereo Keypoints

ORB-SLAM2 as a feature-based method preprocesses the input to extract features at salient keypoint locations, as shown in Fig. 2(b). The input images are then discarded and all system operations are based on these features so that the system is independent of the sensor being stereo or RGB-D. Our system handles monocular and stereo keypoints, which are further classified as close or far.

Stereo keypoints are defined by three coordinates $\mathbf{x}_s = (u_L, v_L, u_R)$, being (u_L, v_L) the coordinates on the left image and u_R the horizontal coordinate in the right image. For stereo cameras, we extract ORB in both images and for every left ORB we search for a match in the right image. This can be done very efficiently assuming stereo rectified images so that epipolar lines are horizontal. We then generate the stereo keypoint with the coordinates of the left ORB and the horizontal coordinate of the right match, which is subpixel refined by patch correlation. For RGB-D cameras, we extract ORB features on the RGB image and, as proposed by Strasdat *et al.* [8], for each feature with coordinates (u_L, v_L), we transform its depth value d into a virtual right coordinate

$$u_R = u_L - \frac{f_x b}{d} \tag{1}$$

where f_x is the horizontal focal length and b is the baseline between the structured light projector and the infrared camera, which we approximate to 8 cm for Kinect and Asus Xtion. The uncertainty of the depth sensor is represented by the uncertainty of the virtual right coordinate. In this way, features from stereo and RGB-D input are handled equally by the rest of the system.

A stereo keypoint is classified as *close* if its associated depth is less than 40 times the stereo/RGB-D baseline, as suggested in [5], otherwise it is classified as *far*. Close keypoints can be safely triangulated from one frame as depth is accurately estimated and provide scale, translation, and rotation information. On the other hand far points provide accurate rotation information but weaker scale and translation information. We triangulate far points when they are supported by multiple views.

Monocular keypoints are defined by two coordinates $\mathbf{x}_m = (u_L, v_L)$ on the left image and correspond to all those ORB for which a stereo match could not be found or that have an invalid depth value in the RGB-D case. These points are only triangulated from multiple views and do not provide scale information, but contribute to the rotation and translation estimation.

B. System Bootstrapping

One of the main benefits of using stereo or RGB-D cameras is that, by having depth information from just one frame, we do not need a specific structure from motion initialization as in the monocular case.

At system startup we create a keyframe with the first frame, set its pose to the origin, and create an initial map from all stereo keypoints.

C. Bundle Adjustment with Monocular and Stereo Constraints

Our system performs BA to optimize the camera pose in the tracking thread (motion-only BA), to optimize a local window of keyframes and points in the local mapping thread (local BA), and after a loop closure to optimize all keyframes and points (full BA). We use the Levenberg–Marquardt method implemented in g2o [19].

Motion-only BA optimizes the camera orientation $\mathbf{R} \in SO(3)$ and position $\mathbf{t} \in \mathbb{R}^3$, minimizing the reprojection error between matched 3-D points $\mathbf{X}^i \in \mathbb{R}^3$ in world coordinates and keypoints $\mathbf{x}^i_{(\cdot)}$, either monocular $\mathbf{x}^i_{\mathrm{m}} \in \mathbb{R}^2$ or stereo $\mathbf{x}^i_{\mathrm{s}} \in \mathbb{R}^3$, with $i \in \mathcal{X}$ the set of all matches

$$\{\mathbf{R}, \mathbf{t}\} = \underset{\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum_{i \in \mathcal{X}} \rho \left(\left\| \mathbf{x}^i_{(\cdot)} - \pi_{(\cdot)} \left(\mathbf{R}\mathbf{X}^i + \mathbf{t} \right) \right\|^2_{\Sigma} \right) \tag{2}$$

where ρ is the robust Huber cost function and Σ the covariance matrix associated to the scale of the keypoint. The projection functions $\pi_{(\cdot)}$, monocular π_{m} and rectified stereo π_{s}, are defined as follows:

$$\pi_{\mathrm{m}} \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \end{bmatrix}, \quad \pi_{\mathrm{s}} \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \\ f_x \frac{X-b}{Z} + c_x \end{bmatrix} \tag{3}$$

where (f_x, f_y) is the focal length, (c_x, c_y) is the principal point, and b the baseline, all known from calibration.

Local BA optimizes a set of covisible keyframes $\mathcal{K}_L$ and all points seen in those keyframes $\mathcal{P}_L$. All other keyframes $\mathcal{K}_F$, not in $\mathcal{K}_L$, observing points in $\mathcal{P}_L$ contribute to the cost function but remain fixed in the optimization. Defining $\mathcal{X}_k$ as the set of matches between points in $\mathcal{P}_L$ and keypoints in a keyframe k, the optimization problem is the following:

$$\{\mathbf{X}^i, \mathbf{R}_l, \mathbf{t}_l | i \in \mathcal{P}_L, l \in \mathcal{K}_L\} = \underset{\mathbf{X}^i,\mathbf{R}_l,\mathbf{t}_l}{\operatorname{argmin}} \sum_{k \in \mathcal{K}_L \cup \mathcal{K}_F} \sum_{j \in \mathcal{X}_k} \rho \left(E(k,j) \right)$$

$$E(k,j) = \left\| \mathbf{x}^j_{(\cdot)} - \pi_{(\cdot)} \left(\mathbf{R}_k \mathbf{X}^j + \mathbf{t}_k \right) \right\|^2_{\Sigma}. \tag{4}$$

Full BA is the specific case of local BA, where all keyframes and points in the map are optimized, except the origin keyframe that is fixed to eliminate the gauge freedom.

D. Loop Closing and Full BA

Loop closing is performed in two steps, first, a loop has to be detected and validated, and second, the loop is corrected optimizing a pose graph. In contrast to monocular ORB-SLAM, where scale drift may occur [20], the stereo/depth information makes scale observable and the geometric validation and pose-graph optimization no longer require dealing with scale drift and are based on rigid body transformations instead of similarities.

In ORB-SLAM2, we have incorporated a full BA optimization after the pose graph to achieve the optimal solution. This optimization might be very costly, and therefore, we perform it in a separate thread, allowing the system to continue creating map and detecting loops. However, this brings the challenge of merging the bundle adjustment output with the current state of the map. If a new loop is detected while the optimization is running, we abort the optimization and proceed to close the loop, which will launch the full BA optimization again. When the full BA finishes, we need to merge the updated subset of keyframes and points optimized by the full BA, with the nonupdated keyframes and points that where inserted while the optimization was running. This is done by propagating the correction of updated keyframes (i.e., the transformation from the nonoptimized to the optimized pose) to nonupdated keyframes through the spanning tree. Nonupdated points are transformed according to the correction applied to their reference keyframe.

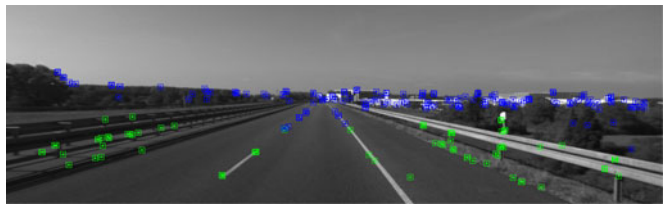

Fig. 3. Tracked points in KITTI 01 [2]. Green points have a depth less than 40 times the stereo baseline, while blue points are further away. In this kind of sequences, it is important to insert keyframes often enough so that the amount of close points allows for accurate translation estimation. Far points contribute to estimate orientation but provide weak information for translation and scale.

E. Keyframe Insertion

ORB-SLAM2 follows the policy introduced in monocular ORB-SLAM of inserting keyframes very often and culling redundant ones afterwards. The distinction between close and far stereo points allows us to introduce a new condition for keyframe insertion, which can be critical in challenging environments where a big part of the scene is far from the stereo sensor, as shown in Fig. 3. In such environment, we need to have a sufficient amount of close points to accurately estimate translation, therefore, if the number of tracked close points drops below τ_t and the frame could create at least τ_c new close stereo points, the system will insert a new keyframe. We empirically found that $\tau_t = 100$ and $\tau_c = 70$ works well in all our experiments.

F. Localization Mode

We incorporate a localization mode, which can be useful for lightweight long-term localization in well-mapped areas, as long as there are not significant changes in the environment. In this mode, the local mapping and loop closing threads are deactivated and the camera is continuously localized by the tracking using relocalization if needed. In this mode, the tracking leverages visual odometry matches and matches to map points. Visual odometry matches are matches between ORB in the current frame and 3-D points created in the previous frame from the stereo/depth information. These matches make the localization robust to unmapped regions, but drift can be accumulated. Map point matches ensure drift-free localization to the existing map. This mode is demonstrated in the accompanying video.

IV. Evaluation

We have evaluated ORB-SLAM2 in three popular datasets and compared to other state-of-the-art SLAM systems, using always the results published by the original authors and standard evaluation metrics in the literature. We have run ORB-SLAM2 in an Intel Core i7-4790 desktop computer with 16-GB RAM. In order to account for the nondeterministic nature of the multithreading system, we run each sequence five times and show median results for the accuracy of the estimated trajectory. Our open-source implementation includes the calibration and instructions to run the system in all these datasets.

TABLE I
COMPARISON OF ACCURACY IN THE KITTI DATASET

	ORB-SLAM2 (stereo)			Stereo LSD-SLAM		
Sequence	t_{rel} (%)	r_{rel} (deg/100 m)	t_{abs} (m)	t_{rel} (%)	r_{rel} (deg/100 m)	t_{abs} (m)
00	0.70	**0.25**	1.3	**0.63**	0.26	**1.0**
01	**1.39**	**0.21**	10.4	2.36	0.36	**9.0**
02	**0.76**	0.23	5.7	0.79	0.23	**2.6**
03	**0.71**	**0.18**	**0.6**	1.01	0.28	1.2
04	0.48	**0.13**	0.2	**0.38**	0.31	0.2
05	**0.40**	**0.16**	**0.8**	0.64	0.18	1.5
06	**0.51**	**0.15**	**0.8**	0.71	0.18	1.3
07	**0.50**	**0.28**	0.5	0.56	0.29	0.5
08	**1.05**	0.32	**3.6**	1.11	**0.31**	3.9
09	**0.87**	0.27	**3.2**	1.14	**0.25**	5.6
10	**0.60**	**0.27**	**1.0**	0.72	0.33	1.5

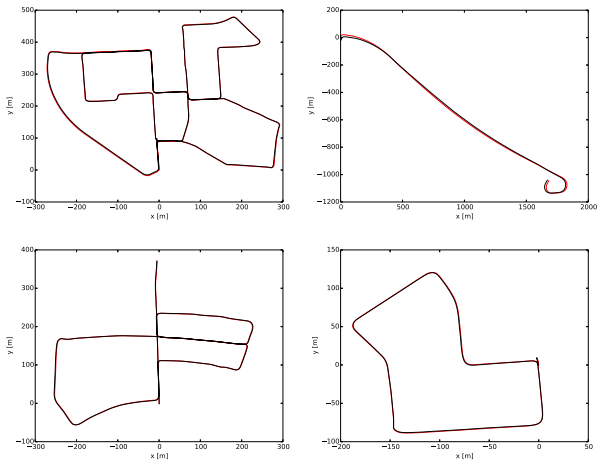

Fig. 4. Estimated trajectory (black) and ground truth (red) in KITTI 00, 01, 05, and 07.

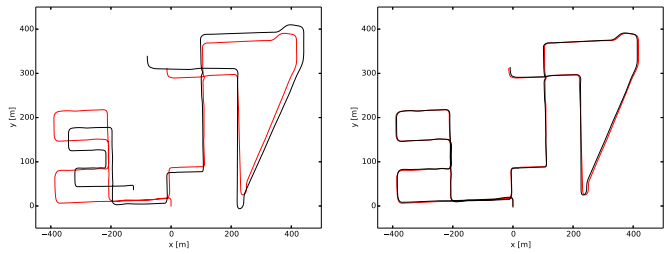

Fig. 5. Estimated trajectory (black) and ground truth (red) in KITTI 08. Left: Monocular ORB-SLAM [1], right: ORB-SLAM2 (stereo). Monocular ORB-SLAM suffers from severe scale drift in this sequence, especially at the turns. In contrast, the proposed stereo version is able to estimate the true scale of the trajectory and map without scale drift.

A. KITTI Dataset

The KITTI dataset [2] contains stereo sequences recorded from a car in urban and highway environments. The stereo sensor has a ~54-cm baseline and works at 10 Hz with a resolution after rectification of 1240×376 pixels. Sequences 00, 02, 05, 06, 07, and 09 contain loops. Our ORB-SLAM2 detects all loops and is able to reuse its map afterwards, except for sequence 09 where the loop happens in very few frames at the end of the sequence. Table I shows results in the 11 training sequences, which have public ground truth, compared to the state-of-the-art Stereo LSD-SLAM [11], to our knowledge the only stereo SLAM showing detailed results for all sequences. We use two different metrics, the absolute translation root-mean-square error (RMSE) t_{abs} proposed in [3], and the average relative translation t_{rel} and rotation r_{rel} errors proposed in [2]. Our system outperforms Stereo LSD-SLAM in most sequences, and achieves in general a relative error lower than 1%. The sequence 01, see Fig. 3, is the only highway sequence in the training set and the translation error is slightly worse. Translation is harder to estimate in this sequence because very few close points can be tracked, due to high speed and low frame rate. However, orientation

TABLE II
EUROC DATASET

Sequence	ORB-SLAM2 (stereo)	Stereo LSD-SLAM
V1_01_easy	**0.035**	0.066
V1_02_medium	**0.020**	0.074
V1_03_difficult	**0.048**	0.089
V2_01_easy	0.037	-
V2_02_medium	0.035	-
V2_03_difficult	X	-
MH_01_easy	0.035	-
MH_02_easy	0.018	-
MH_03_medium	0.028	-
MH_04_difficult	0.119	-
MH_05_difficult	0.060	-

Comparison of Translation RMSE (m).

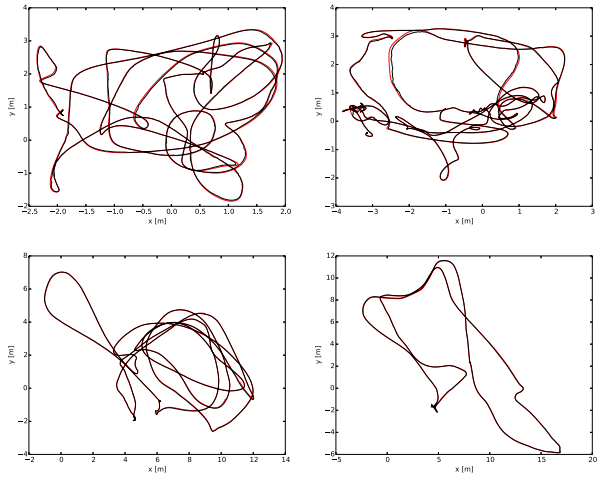

Fig. 6. Estimated trajectory (black) and ground truth (red) in EuRoC V1_02_medium, V2_02_medium, MH_03_medium, and MH_05_difficult.

TABLE III
TUM RGB-D DATASET

Sequence	ORB-SLAM2	Elastic-Fusion (RGB-D)	Kintinuous	DVO SLAM	RGBD SLAM
fr1/desk	**0.016**	0.020	0.037	0.021	0.026
fr1/desk2	**0.022**	0.048	0.071	0.046	-
fr1/room	0.047	0.068	0.075	**0.043**	0.087
fr2/desk	**0.009**	0.071	0.034	0.017	0.057
fr2/xyz	**0.004**	0.011	0.029	0.018	-
fr3/office	**0.010**	0.017	0.030	0.035	-
fr3/nst	0.019	**0.016**	0.031	0.018	-

Comparison of translation RMSE (m).

Fig. 7. Dense pointcloud reconstructions from estimated keyframe poses and sensor depth maps in TUM RGB-D *fr3_office*, *fr1_room*, *fr2_desk*, and *fr3_nst*.

TABLE IV
TIMING RESULTS OF EACH THREAD IN MILISECONDS (MEAN ± 2 STD. DEVIATIONS)

Settings	Sequence	V2_02	07	fr3_office
	Dataset	EuRoC	KITTI	TUM
	Sensor	Stereo	Stereo	RGB-D
	Resolution	752×480	1226×370	640×480
	Camera FPS	20 Hz	10 Hz	30 Hz
	ORB Features	1000	2000	1000
Tracking	Stereo Rectification	3.43 ± 1.10	-	-
	ORB Extraction	13.54 ± 4.60	24.83 ± 8.28	11.48 ± 1.84
	Stereo Matching	11.26 ± 6.64	15.51 ± 4.12	0.02 ± 0.00
	Pose Prediction	2.07 ± 1.58	2.36 ± 1.84	2.65 ± 1.28
	Local Map Tracking	10.13 ± 11.40	5.38 ± 3.52	9.78 ± 6.42
	New Keyframe Decision	1.40 ± 1.14	1.91 ± 1.06	1.58 ± 0.92
	Total	41.66 ± 18.90	49.47 ± 12.10	25.58 ± 9.76
Mapping	Keyframe Insertion	10.30 ± 7.50	11.61 ± 3.28	11.36 ± 5.04
	Map Point Culling	0.28 ± 0.20	0.45 ± 0.38	0.25 ± 0.10
	Map Point Creation	40.43 ± 36.10	47.69 ± 29.52	53.99 ± 23.62
	Local BA	137.99 ± 248.18	69.29 ± 61.88	196.67 ± 213.42
	Keyframe Culling	3.80 ± 8.20	0.99 ± 0.92	6.69 ± 8.24
	Total	174.10 ± 278.80	129.52 ± 88.52	267.33 ± 245.10
Loop	Database Query	3.57 ± 5.86	4.13 ± 3.54	2.63 ± 2.26
	SE3 Estimation	0.69 ± 1.82	1.02 ± 3.68	0.66 ± 1.68
	Loop Fusion	21.84	82.70	298.45
	Essential Graph Opt.	73.15	178.31	281.99
	Total	108.59	284.88	598.70
BA	Full BA	349.25	1144.06	1640.96
	Map Update	3.13	11.82	5.62
	Total	396.02	1205.78	1793.02
Loop size (#keyframes)		82	248	225

can be accurately estimated, achieving an error of 0.21 degrees per 100 meters, as there are many far point that can be long tracked. Fig. 4 shows some examples of estimated trajectories.

Compared to the monocular results presented in [1], the proposed stereo version is able to process the sequence 01 where the monocular system failed. In this highway sequence, see Fig. 3, close points are in view only for a few frames. The ability of the stereo version to create points from just one stereo keyframe instead of the delayed initialization of the monocular, consisting on finding matches between two keyframes, is critical in this sequence not to lose tracking. Moreover the stereo system estimates the map and trajectory with metric scale and does not suffer from scale drift, as seen in Fig. 5.

B. EuRoC Dataset

The recent EuRoC dataset [21] contains 11 stereo sequences recorded from a microaerial vehicle (MAV) flying around two different rooms and a large industrial environment. The stereo sensor has a ∼11-cm baseline and provides WVGA images at 20 Hz. The sequences are classified as *easy*, *medium,* and *difficult* depending on MAV's speed, illumination, and scene texture. In all sequences, the MAV revisits the environment and ORB-SLAM2 is able to reuse its map, closing loops when necessary. Table II shows absolute translation RMSE of ORB-SLAM2 for all sequences, comparing to Stereo LSD-SLAM, for the results provided in [11]. ORB-SLAM2 achieves a localization precision of a few centimeters and is more accurate than Stereo LSD-SLAM. Our tracking get lost in some parts of *V2_03_difficult* due to severe motion blur. As shown in [22], this sequence can be processed using IMU information. Fig. 6 shows examples of computed trajectories compared to the ground truth.

C. TUM RGB-D Dataset

The TUM RGB-D dataset [3] contains indoors sequences from RGB-D sensors grouped in several categories to evaluate object reconstruction and SLAM/odometry methods under different texture, illumination, and structure conditions. We show results in a subset of sequences where most RGB-D methods are usually evaluated. In Table III, we compare our accuracy to the following state-of-the-art methods: ElasticFusion [15], Kintinuous [12], DVO-SLAM [14], and RGB-D SLAM [13]. Our method is the only one based on bundle adjustment and outperforms the other approaches in most sequences. As we already noticed for RGB-D SLAM results in [1], depthmaps for *freiburg2* sequences have a 4% scale bias, probably coming from miscalibration, that we have compensated in our runs and could partly explain our significantly better results. Fig. 7 shows the point clouds that result from backprojecting the sensor depth maps from the computed keyframe poses in four sequences. The good definition and the straight contours of desks and posters prove the high accuracy localization of our approach.

D. Timing Results

In order to complete the evaluation of the proposed system, we present in Table IV, timing results in three sequences with different image resolutions and sensors. The mean and two standard deviation ranges are shown for each thread task. As these sequences contain one single loop, the full BA and some tasks of the loop closing thread are executed just once and only a single time measurement is reported. The average tracking time per frame is below the inverse of the camera frame rate for each sequence, meaning that our system is able to work in real time. As ORB extraction in stereo images is parallelized, it can be seen that extracting 1000 ORB features in the stereo WVGA images of V2_02 is similar to extracting the same amount of features in the single VGA image channel of fr3_office.

The number of keyframes in the loop is shown as reference for the times related to loop closing. While the loop in KITTI 07 contains more keyframes, the covisibility graph built for the indoor fr3_office is denser, and therefore, the loop fusion, pose-graph optimization, and full BA tasks are more expensive. The higher density of the covisibility graph makes the local map contain more keyframes and points, and therefore, local map tracking and local BA are also more expensive.

V. Conclusion

We have presented a full SLAM system for monocular, stereo, and RGB-D sensors, able to perform relocalization, loop closing, and reuse its map in real time on standard CPUs. We focus on building globally consistent maps for reliable and long-term localization in a wide range of environments as demonstrated in the experiments. The proposed localization mode with the relocalization capability of the system yields a very robust, zero drift, and lightweight localization method for known environments. This mode can be useful for certain applications, such as tracking the user viewpoint in virtual reality in a well-mapped space.

The comparison to the state-of-the-art shows that ORB-SLAM2 achieves in most cases the highest accuracy. In the KITTI visual odometry benchmark ORB-SLAM2 is currently the best stereo SLAM solution. Crucially, compared with the stereo visual odometry methods that have flourished in recent years, ORB-SLAM2 achieves zero-drift localization in already mapped areas.

Surprisingly our RGB-D results demonstrate that if the most accurate camera localization is desired, bundle adjustment performs better than direct methods or ICP, with the additional advantage of being less computationally expensive, not requiring GPU processing to operate in real time.

We have released the source code of our system, with examples and instructions so that it can be easily used by other researchers. ORB-SLAM2 is to the best of our knowledge the first open-source visual SLAM system that can work either with monocular, stereo, and RGB-D inputs. Moreover our source code contains an example of an augmented reality application[2] using a monocular camera to show the potential of our solution.

Future extensions might include, to name some examples, nonoverlapping multicamera, fisheye, or omnidirectional cameras support, large-scale dense fusion, cooperative mapping, or increased motion blur robustness.

References

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

[2] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[3] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 573–580.

[4] R. A. Newcombe *et al.*, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. IEEE Int. Symp. Mixed Augmented Reality*, 2011, pp. 127–136.

[5] L. M. Paz, P. Piniés, J. D. Tardós, and J. Neira, "Large-scale 6-DOF SLAM with stereo-in-hand," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 946–957, Oct. 2008.

[6] J. Civera, A. J. Davison, and J. M. M. Montiel, "Inverse depth parametrization for monocular SLAM," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 932–945, Oct. 2008.

[2]https://youtu.be/kPwy8yA4CKM

[7] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Visual SLAM: Why filter?" *Image Vis. Comput.*, vol. 30, no. 2, pp. 65–77, 2012.

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

[9] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "RSLAM: A system for large-scale mapping in constant-time using stereo," *Int. J. Comput. Vision*, vol. 94, no. 2, pp. 198–214, 2011.

[10] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, and J. J. Berlles, "Stereo parallel tracking and mapping for robot localization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 1373–1378.

[11] J. Engel, J. Stueckler, and D. Cremers, "Large-scale direct SLAM with stereo cameras," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 1935–1942.

[12] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald, "Real-time large-scale dense RGB-D SLAM with volumetric fusion," *Int. J. Robot. Res.*, vol. 34, no. 4-5, pp. 598–626, 2015.

[13] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D mapping with an RGB-D camera," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 177–187, Feb. 2014.

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

[15] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "ElasticFusion: Real-time dense SLAM and light source estimation," *Int. J. Robot. Res.*, vol. 35, no. 14, pp. 1697–1716, 2016.

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

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

[18] R. Mur-Artal and J. D. Tardós, "Fast relocalisation and loop closing in keyframe-based SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 846–853.

[19] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 3607–3613.

[20] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular SLAM," in *Proc. Robot., Sci. Syst.*, 2010.

[21] M. Burri *et al.*, "The EuRoC micro aerial vehicle datasets," *Int. J. Robot. Res.*, vol. 35, no. 10, pp. 1157–1163, 2016.

[22] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular SLAM with map reuse," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 796–803, Apr. 2017.