Taketomi *et al. IPSJ Transactions on Computer Vision and Applications* (2017) 9:16
DOI 10.1186/s41074-017-0027-2

IPSJ Transactions on Computer Vision and Applications

REVIEW PAPER

Open Access

Visual SLAM algorithms: a survey from 2010 to 2016

Takafumi Taketomi[1*], Hideaki Uchiyama[2] and Sei Ikeda[3]

Abstract
SLAM is an abbreviation for simultaneous localization and mapping, which is a technique for estimating sensor motion and reconstructing structure in an unknown environment. Especially, Simultaneous Localization and Mapping (SLAM) using cameras is referred to as visual SLAM (vSLAM) because it is based on visual information only. vSLAM can be used as a fundamental technology for various types of applications and has been discussed in the field of computer vision, augmented reality, and robotics in the literature. This paper aims to categorize and summarize recent vSLAM algorithms proposed in different research communities from both technical and historical points of views. Especially, we focus on vSLAM algorithms proposed mainly from 2010 to 2016 because major advance occurred in that period. The technical categories are summarized as follows: feature-based, direct, and RGB-D camera-based approaches.

Keywords: Survey, Visual SLAM, Computer vision, Augmented reality, Robotics

1 Introduction

Simultaneous Localization and Mapping (SLAM) is a technique for obtaining the 3D structure of an unknown environment and sensor motion in the environment. This technique was originally proposed to achieve autonomous control of robots in robotics [1]. Then, SLAM-based applications have widely become broadened such as computer vision-based online 3D modeling, augmented reality (AR)-based visualization, and self-driving cars. In early SLAM algorithms, many different types of sensors were integrated such as laser range sensors, rotary encoders, inertial sensors, GPS, and cameras. Such algorithms are well summarized in the following papers [2–5].

In recent years, SLAM using cameras only has been actively discussed because the sensor configuration is simple and the technical difficulties are higher than others. Since the input of such SLAM is visual information only, the technique is specifically referred to as visual SLAM (vSLAM). vSLAM algorithms have widely proposed in the field of computer vision, robotics, and AR [6]. Especially, they are suitable for camera pose estimation in AR systems because the configuration of the systems can be simple such as camera-mounted tablets or smartphones. One of the important requirements in AR systems is real-time response to seamlessly and interactively merge real and virtual objects. To achieve the response with a limited computational resource on a light-weighted hand-held device, various low computational-cost vSLAM algorithms have been proposed in the literature. The application of such vSLAM algorithms is not limited to AR systems. For example, it is also useful for unmanned autonomous vehicles (UAV) in robotics [7]. Even though vSLAM algorithms have been proposed for different purposes in different research communities, they basically share overall parts of technical core ideas and can be used to achieve different purposes each other. Therefore, we categorize and summarize such algorithms as a survey paper.

In this paper, we review real-time vSLAM algorithms, which remarkably evolve forward in the 2010s. In general, the technical difficulty of vSLAM is higher than that of other sensor-based SLAMs because cameras can acquire less visual input from a limited field of views compared to 360° laser sensing which is typically used in robotics. From such input, camera poses need to be continuously estimated and the 3D structure of an unknown environment is simultaneously reconstructed. The early work of vSLAM using a monocular camera was based on tracking and mapping feature points in 2000s. This is called

*Correspondence: takafumi-t@is.naist.jp
[1]Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, 630-0192 Nara, Japan
Full list of author information is available at the end of the article

"feature-based approach." To cope with texture-less or feature-less environments, vSLAM without detecting feature points and directly with a whole image for tracking and mapping has been proposed. This is called "direct approach." With the advent of low-cost RGB-D sensors such as Microsoft Kinect, vSLAM algorithms with both a monocular image and its depth have been proposed. Therefore, the existing vSLAM algorithms introduced in this paper are categorized according to feature-based, direct, and RGB-D camera-based approaches. This paper will be helpful for readers who want to start to learn the basic framework of vSLAM, the difference among the algorithms, and the progress from 2010 to 2016. Also, remaining technical problems are discussed for further research topics and several benchmarking methodologies for comparing different algorithms are provided so that readers can have some perspectives for next research direction.

The remainder of the paper is organized as follows. In Sections 2 and 3, the elements of vSLAM and related techniques of vSLAM including visual odometry are introduced. In Sections 4, 5, and 6 where existing vSLAM algorithms are summarized, feature-based, direct, and RGB-D-based vSLAM algorithms are, respectively, introduced. In Section 7, remaining technical problems in vSLAM algorithms are discussed. In Section 8, datasets for evaluating a performance of vSLAM algorithms are introduced. Finally, we present the conclusion in Section 9. Note that there exist survey papers on vSLAM algorithms proposed till 2011 [8, 9]. These papers are also useful for understanding our survey on newer algorithms.

2 Elements of vSLAM

In this section, we first introduce the basic framework followed by most of vSLAM algorithms since late 2000s.

2.1 Basic modules

The framework is mainly composed of three modules as follows.

1. Initialization
2. Tracking
3. Mapping

To start vSLAM, it is necessary to define a certain coordinate system for camera pose estimation and 3D reconstruction in an unknown environment. Therefore, in the initialization, the global coordinate system should first be defined, and a part of the environment is reconstructed as an initial map in the global coordinate system. After the initialization, tracking and mapping are performed to continuously estimate camera poses. In the tracking, the reconstructed map is tracked in the image to estimate the camera pose of the image with respect to the map. In order to do this, 2D–3D correspondences between the image and the map are first obtained from feature matching or feature tracking in the image. Then, the camera pose is computed from the correspondences by solving the Perspective-n-Point (PnP) problem [10, 11]. It should be noted that most of vSLAM algorithms assumes that intrinsic camera parameters are calibrated beforehand so that they are known. Therefore, a camera pose is normally equivalent to extrinsic camera parameters with translation and rotation of the camera in the global coordinate system. In the mapping, the map is expanded by computing the 3D structure of an environment when the camera observes unknown regions where the mapping is not performed before.

2.2 Additional modules for stable and accurate vSLAM

The following two additional modules are also included in vSLAM algorithms according to the purposes of applications.

- Relocalization
- Global map optimization

The relocalization is required when the tracking is failed due to fast camera motion or some disturbances. In this case, it is necessary to compute the camera pose with respect to the map again. Therefore, this process is called "relocalization." If the relocalization is not incorporated into vSLAM systems, the systems do not work anymore after the tracking is lost and such systems are not practically useful. Therefore, a fast and efficient method for the relocalization has been discussed in the literature. Note that this is also referred to as kidnapped robot problems in robotics.

The other module is global map optimization. The map generally includes accumulative estimation error according to the distance of camera movement. In order to suppress the error, the global map optimization is normally performed. In this process, the map is refined by considering the consistency of whole map information. When a map is revisited such that a starting region is captured again after some camera movement, reference information that represents the accumulative error from the beginning to the present can be computed. Then, a loop constraint from the reference information is used as a constraint to suppress the error in the global optimization.

Loop closing is a technique to acquire the reference information. In the loop closing, a closed loop is first searched by matching a current image with previously acquired images. If the loop is detected, it means that the camera captures one of previously observed views. In this case, the accumulative error occurred during camera movement can be estimated. Note that the closed-loop detection procedure can be done by using the same techniques as relocalization. Basically, relocalization is done for recovering a camera pose and loop detection is done for obtaining geometrically consistent map.

Pose-graph optimization has widely been used to suppress the accumulated error by optimizing camera poses [12, 13]. In this method, the relationship between camera poses is represented as a graph and the consistent graph is built to suppress the error in the optimization. Bundle adjustment (BA) is also used to minimize the reprojection error of the map by optimizing both the map and the camera poses [14]. In large environments, this optimization procedure is employed to minimize estimation errors efficiently. In small environments, BA may be performed without loop closing because the accumulated error is small.

2.3 Summary

As listed above, the framework of vSLAM algorithms is composed of five modules: initialization, tracking, mapping, relocalization, and global map optimization. Since each vSLAM algorithm employs different methodologies for each module, features of a vSLAM algorithm highly depend on the methodologies employed. Therefore, it is important to understand each module of a vSLAM algorithm to know its performance, advantages, and limitations.

It should be noted that tracking and mapping (TAM) is used instead of using localization and mapping. TAM was first used in Parallel Tracking and Mapping (PTAM) [15] because localization and mapping are not simultaneously performed in a traditional way. Tracking is performed in every frame with one thread whereas mapping is performed at a certain timing with another thread. After PTAM was proposed, most of vSLAM algorithms follows the framework of TAM. Therefore, TAM is used in this paper.

3 Related technologies

vSLAM, visual odometry, and online structure from motion are designed for estimating camera motion and 3D structure in an unknown environment. In this section, we explain the relationship among them.

3.1 Visual odometry

Odometry is to estimate the sequential changes of sensor positions over time using sensors such as wheel encoder to acquire relative sensor movement. Camera-based odometry called visual odometry (VO) is also one of the active research fields in the literature [16, 17]. From the technical point of views, vSLAM and VO are highly relevant techniques because both techniques basically estimate sensor positions. According to the survey papers in robotics [18, 19], the relationship between vSLAM and VO can be represented as follows.

$$\text{vSLAM} = \text{VO} + \text{global map optimization}$$

The main difference between these two techniques is global map optimization in the mapping. In other words, VO is equivalent to the modules in Section 2.1. In the VO, the geometric consistency of a map is considered only in a small portion of a map or only relative camera motion is computed without mapping. On the other hand, in the vSLAM, the global geometric consistency of a map is normally considered. Therefore, to build a geometrically consistent map, the global optimization is performed in the recent vSLAM algorithms.

The relationship between vSLAM and VO can also be found from the papers [20, 21] and the papers [22, 23]. In the paper [20, 22], a technique on VO was first proposed. Then, a technique on vSLAM was proposed by adding the global optimization in VO [21, 23].

3.2 Structure from motion

Structure from motion (SfM) is a technique to estimate camera motion and 3D structure of the environment in a batch manner [24]. In the paper [25], a SfM method that runs online was proposed. The authors named it as real-time SfM. From the technical point of views, there is no definitive difference between vSLAM and real-time SfM. This may be why the word "real-time SfM" is not found in recent papers.

As explained in this section, vSLAM, VO, and real-time SfM share many common components. Therefore, we introduce all of them and do not distinguish these technologies in this paper.

4 Feature-based methods

There exist two types of feature-based methods in the literature: filter-based and BA-based methods. In this section, we explain both methods and provide the comparison. Even though some of the methods were proposed before 2010, we explained them here because they can be considered as fundamental frameworks for other methods.

4.1 MonoSLAM

First monocular vSLAM was developed in 2003 by Davison et al. [26, 27]. They named it MonoSLAM. MonoSLAM is considered as a representative method in filter-based vSLAM algorithms. In MonoSLAM, camera motion and 3D structure of an unknown environment are simultaneously estimated using an extended Kalman filter (EKF). 6 Degree of freedom (DoF) camera motion and 3D positions of feature points are represented as a state vector in EKF. Uniform motion is assumed in a prediction model, and a result of feature point tracking is used as observation. Depending on camera movement, new feature points are added to the state vector. Note that the initial map is created by observing a known object where a global coordinate system is defined. In summary, MonoSLAM is composed of the following components.

- Map initialization is done by using a known object.
- Camera motion and 3D positions of feature points are estimated using EKF.

The problem of this method is a computational cost that increases in proportion to the size of an environment. In large environments, the size of a state vector becomes large because the number of feature points is large. In this case, it is difficult to achieve real-time computation.

4.2 PTAM

To solve the problem of a computational cost in MonoSLAM, PTAM [15] split the tracking and the mapping into different threads on CPU. These two threads are executed in parallel so that the computational cost of the mapping does not affect the tracking. As a result, BA that needs a computational cost in the optimization can be used in the mapping. This means that the tracking estimates camera motion in real-time, and the mapping estimates accurate 3D positions of feature points with a computational cost. PTAM is the first method which incorporates BA into the real-time vSLAM algorithms. After publishing PTAM, most vSLAM algorithms follow this type of multi-threading approaches.

In PTAM, the initial map is reconstructed using the five-point algorithm [28]. In the tracking, mapped points are projected onto an image to make 2D–3D correspondences using texture matching. From the correspondences, camera poses can be computed. In the mapping, 3D positions of new feature points are computed using triangulation at certain frames called keyframes. One of the significant contributions of PTAM is to introduce this keyframe-based mapping in vSLAM. An input frame is selected as a keyframe when a large disparity between an input frame and one of the keyframes is measured. A large disparity is basically required for accurate triangulation. In contrast to MonoSLAM, 3D points of feature points are optimized using lobal BA with some keyframes and global BA with all keyframes with the map. Also, in the tracking process, the newer vision of PTAM employ a relocalization algorithm [29]. It uses a randomized tree-based feature classifier for searching the nearest keyframe of an input frame. In summary, PTAM is composed of the following four components.

- Map initialization is done by the five-point algorithm [28].
- Camera poses are estimated from matched feature points between map points and the input image.
- 3D positions of feature points are estimated by triangulation, and estimated 3D positions are optimized by BA.
- The tracking process is recovered by a randomized tree-based searching [29].

Compared to MonoSLAM, in PTAM, the system can handle thousands of feature points by splitting the tracking and the mapping into different threads on CPU.

There have been proposed many extended PTAM algorithms. Castle et al. developed a multiple map version of PTAM [30]. Klein et al. developed a mobile phone version of PTAM [31]. In order to run PTAM on mobile phones, input image resolution, map points, and number of keyframes are reduced. In addition, they consider rolling shutter distortion in BA to get an accurate estimation result because a rolling shutter is normally installed in most mobile phone cameras due to its cheap cost. Since PTAM can reconstruct a sparse 3D structure of the environment only, the third thread can be used to reconstruct a dense 3D structure of the environment [32, 33].

4.3 Comparison between MonoSLAM and PTAM

The difference between the EKF-based mapping in MonoSLAM and the BA-based mapping with the keyframes in PTAM was discussed in the literature [34]. According to the literature, to improve an accuracy of vSLAM, it is important to increase the number of feature points in a map. From this point of view, the BA-based approach is better than the EKF-based approach because it can handle large number of points.

4.4 Techniques on global map optimization

Geometric consistency of the whole map is maintained by using BA for the keyframes as explained above. However, in general, BA suffers from a local minimum problem due to the numerous number of parameters including camera poses of the keyframes and points in the map. Pose-graph optimization is a solution to avoid this problem in the loop closing as described in Section 2. In the loop closing, camera poses are first optimized using the loop constraint. After optimizing the camera poses, BA is performed to optimize both 3D positions of feature points and the camera poses. For the loop closing, a visual information-based approach is employed [35]. They used a bag-of-words-based image retrieval technique to detect one of the keyframes which view is similar with the current view [36].

In a vSLAM system [35], a stereo camera is selected as a vision sensor. In this case, the scale of the coordinate system is fixed and known. However, in monocular vSLAM cases, there is a scale ambiguity and a scale may change during camera movement if global BA is not performed. In this case, a scale drift problem occurs and the scale of the coordinate system at each frame may not be consistent. In order to correct the scale drift, camera poses should be optimized in 7 DoF. Strasdat et al. [37] proposed a method for optimizing 7 DoF camera poses based on similarity transformation.

As an extension of PTAM, ORB-SLAM [38] includes BA, vision-based closed-loop detection, and 7 DoF pose-graph optimization. As far as we know, ORB-SLAM is the most complete feature-based monocular vSLAM system. ORB-SLAM is extended to the stereo vSLAM and the RGB-D vSLAM [39].

4.5 Summary

Figure 1 shows the summary of feature-based methods. MonoSLAM was developed in 2003 [26]. Both the tracking and the mapping are sequentially and simultaneously using EKF. PTAM was developed in 2007 [15]. They proposed to separate the tracking and the mapping into different threads on CPU. This multi-threading approach enables to handle thousands of feature points in the map. In large environments, it is difficult to get global optimal of the map and the camera poses due to the local minimum problem in BA. To avoid this problem, closed-loop detection and pose-graph optimization can be used before BA. ORB-SLAM [38] includes multi-threaded tracking, mapping, and closed-loop detection, and the map is optimized using pose-graph optimization and BA, and this can be considered as all-in-one package of monocular vSLAM. Since ORB-SLAM is an open source project[1], we can easily use this whole vSLAM system in our local environment.

In this section, we introduced feature point-based vSLAM algorithms. Feature point-based vSLAM algorithms normally employ handcrafted feature detectors and descriptors and can provide stable estimation results in rich textured environments. However, it is difficult to handle curved edges and other complex cues by using such handcrafted features. In some special cases such as poor textured environments, line features have been used as image features [40, 41]. Moreover, feature points and edgelets are combined to achieve robust estimation against to motion-blurred input images [42].

5 Direct methods

In contrast to feature-based methods in the previous section, direct methods directly use an input image without any abstraction using handcrafted feature detectors and descriptors. They are also called feature-less approaches. In general, photometric consistency is used as an error measurement in direct methods whereas geometric consistency such as positions of feature points in an image is used in feature-based methods. In this section, we introduce some leading direct methods.

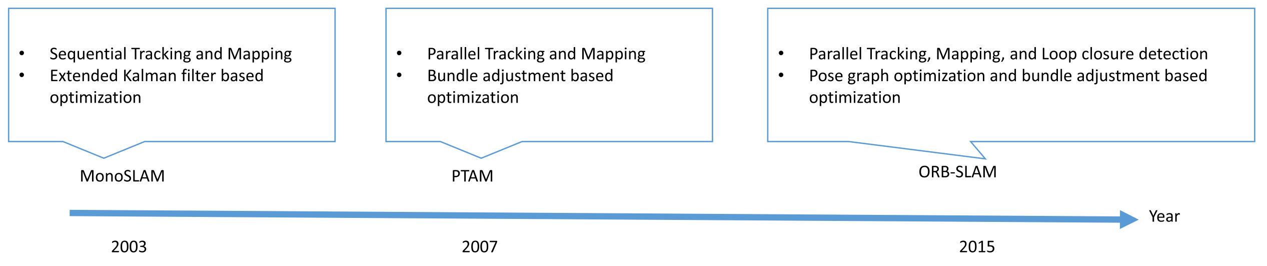
5.1 DTAM

Newcombe et al. proposed a fully direct method [43] called DTAM. In DTAM, the tracking is done by comparing the input image with synthetic view images generated from the reconstructed map. This is simply equivalent to registration between an image and the 3D model of a map and is efficiently implemented on GPU in DTAM. The mapping is done by using multi-baseline stereo [44], and then, the map is optimized by considering space continuity [45] so that 3D coordinates of all pixels can be computed. The initial depth map is created using a stereo measurement like PTAM. In summary, DTAM is composed of the following three components.

- Map initialization is done by the stereo measurement.
- Camera motion is estimated by synthetic view generation from the reconstructed map.
- Depth information is estimated for every pixels by using multi-baseline stereo, and then, it is optimized by considering space continuity.

The DTAM algorithm is optimized for achieving real-time processing on mobile phones [46]. Basically, these methods [43, 46, 47] are designed for fast and online 3D modeling.

It should be noted that Stühmer et al. previously proposed a variational approach for estimating depth information for every pixels [47]. They use similar cost function for the mapping as DTAM. However, in this method, PTAM [15] was used for the tracking. Therefore, the tracking is the feature-based method and is not a fully direct method.


• Sequential Tracking and Mapping
• Extended Kalman filter based optimization
• Parallel Tracking and Mapping
• Bundle adjustment based optimization
• Parallel Tracking, Mapping, and Loop closure detection
• Pose graph optimization and bundle adjustment based optimization
MonoSLAM
PTAM
ORB-SLAM
Year
2003
2007
2015
Fig. 1 Summary of feature-based methods

5.2 LSD-SLAM

LSD-SLAM is another leading method in direct methods. The core idea of LSD-SLAM follows the idea from semi-dense VO [20]. In this method, reconstruction targets are limited to areas which have intensity gradient compared to DTAM which reconstructs full areas. This means that it ignores textureless areas because it is difficult to estimate accurate depth information from images. In the mapping, random values are first set as initial depth values for each pixels, and then, these values are optimized based on photometric consistency. Since this method does not consider the geometric consistency of the whole map, this method is called visual odometry.

In 2014, semi-dense VO was extended to LSD-SLAM [21]. In LSD-SLAM, loop-closure detection and 7 DoF pose-graph optimization as described in the previous sections are added to the semi-dense visual odometry algorithm [20]. In summary, LSD-SLAM is composed of the following four components.

- Random values are set as an initial depth value for each pixel.
- Camera motion is estimated by synthetic view generation from the reconstructed map.
- Reconstructed areas are limited to high-intensity gradient areas.
- 7 DoF pose-graph optimization is employed to obtain geometrically consistent map.

Basically, these semi-dense approaches [20, 21] can achieve real-time processing with CPU. In addition, they optimized the LSD-SLAM algorithm for mobile phones by considering the CPU architecture for them [48]. In the literature [48], they also evaluated the accuracy of the LSD-SLAM algorithm for low-resolution input images. LSD-SLAM is extended to stereo cameras and omni-directional cameras [49, 50].

5.3 SVO and DSO

Forster et al. proposed semi-direct VO (SVO) [51]. Although the tracking is done by feature point matching, the mapping is done by the direct method. In feature-based methods, feature descriptors and the Lucas-Kanade tracker [52] are used to find correspondences. In contrast to feature-based methods, camera motion is estimated by minimizing photometric errors surrounding feature points. This method can be regarded as sparse version of DTAM and LSD-SLAM.

Recently, Engel et al. propose the direct sparse odometry (DSO) [53]. In contrast to SVO, DSO is a fully direct method. In order to suppress accumulative error, DSO removes error factors as much as possible from geometric and photometric perspectives. In DSO, the input image is divided into several blocks, and then, high intensity points are selected as reconstruction candidates. By using this strategy, points are spread within the whole image. In addition, to achieve highly accurate estimation, DSO uses both geometric and photometric camera calibration results. It should be noted that DSO considers local geometric consistency only. Therefore, DSO is classified into VO, not vSLAM.

5.4 Summary

Figure 2 shows the summary of direct methods. Direct methods can be categorized according to map density. Dense methods [43, 47] generate a dense map computed such that depth values are estimated for every pixels in each keyframe. These methods can be useful for realtime 3D modeling with GPU. In contrast to dense methods, semi-dense [21] and sparse [51, 53] methods focus on the applications based on the tracking of sensor poses. These methods can run in real-time on CPUs.

6 RGB-D vSLAM

Recently, structured light-based RGB-D cameras [54] such as Microsoft Kinect [55] become cheap and small. Since such cameras provide 3D information in real-time, these cameras are also used in vSLAM algorithms.

6.1 Difference with monocular vSLAM

By using RGB-D cameras, 3D structure of the environment with its texture information can be obtained directly. In addition, in contrast to monocular vSLAM algorithms, the scale of the coordinate system is known because 3D structure can be acquired in the metric space.

The basic framework of depth (D)-based vSLAM is as follows. An iterative closest point (ICP) algorithm [56] have widely been used to estimate camera motion. Then, the 3D structure of the environment is reconstructed by combining multiple depth maps. In order to incorporate RGB into depth-based vSLAM, many approaches had been proposed as explained below.

It should be noted that most of consumer depth cameras are developed for indoor usages. They project IR patterns into an environment to measure the depth information. It is difficult to detect emitted IR patterns in outdoor environments. In addition, there is a limitation of a range of depth measurement such that the RGB-D sensors can capture the environment from 1 to 4 m.

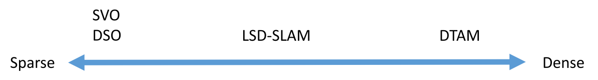


Fig. 2 Comparison of direct methods based on map density

6.2 KinectFusion

Newcombe et al. proposed KinectFution in 2011 [57]. In KinectFusion, a voxel space is used for representing the 3D structure of the environment. The 3D structure of the environment is reconstructed by combining obtained depth maps in the voxel space, and camera motion is estimated by the ICP algorithm using an estimated 3D structure and the input depth map, which is depth-based vSLAM. KinectFusion is implemented on GPU to achieve real-time processing.

Kahler et al. achieve realtime processing of KinectFusion on mobile devices [58]. To reduce a computational cost, they use a voxel block hashing in the mapping process. RGB-D vSLAM suffer from amount of data. In the literature [59], they reduce amount of data by unifying co-planar points.

6.3 SLAM++

Salas-Moreno et al. proposed an object level RGB-D vSLAM algorithm [60]. In this method, several 3D objects are registered into the database in advance, and these objects are recognized in an online process. By recognizing 3D objects, the estimated map is refined, and 3D points are replaced by 3D objects to reduce the amount of data.

As a similar algorithm, Tateno et al. proposed a real-time segmentation method for RGB-D SLAM [61]. Segmented objects are labeled, and then, these objects can be used as recognition targets.

6.4 Techniques on RGB-D VO and global map optimization

For the tracking, RGB images are also used in RGB-D vSLAM algorithms. In the literature [62, 63], relative camera motion is estimated by tracking feature points between successive frames. A translation matrix is then estimated using tracked feature points, and this translation matrix is refined by the ICP algorithm using depth maps. On the other hand, photometric consistency-based camera motion tracking methods have been proposed [22, 23, 64]. This photometric consistency-based camera motion tracking is also employed monocular camera-based dense vSLAM methods [20, 21, 43].

In order to obtain a geometrically consistent map, pose-graph optimization and deformation graph optimization are used in RGB-D vSLAM algorithms. Kerl et al. used pose-graph optimization to reduce the accumulative error [23]. This pose-graph optimization is almost the same as loop closure in monocular vSLAM algorithms. Whelan et al. used pose-graph optimization for camera motion refinement and deformation graph optimization for map refinement, respectively, [65]. In contrast to other works [23], the estimated map is also refined. In [66], deformation graph optimization is frequently used for certain frames, and camera motion is estimated by matching between a RGB-D image and a reconstructed model. They showed geometrically consistent model that can be acquired using deformation graph optimization as often as possible.

Note that RGB-D SLAM APIs are provided in consumer devices such as Google Tango[2] and Structure Sensor[3]. Especially, Google Tango provides a stable estimation result by combining internal sensor information.

7 Open problems

In practical situations, vSLAM faces some problems. In this section, we list the following problems: purely rotational motion, map initialization, estimating intrinsic camera parameters, rolling shutter distortion, and scale ambiguity.

7.1 Pure rotation

When users move a device in handheld augmented reality applications, purely rotational motion sometimes occurs. This is a problem because disparities cannot be observed during purely rotational motion with monocular vSLAM. To solve this problem, in the literature [67, 68], different projection models are used to handle general camera motion and purely rotational motion. For example, homography-based tracking is used for purely rotational motion and 6 DoF camera tracking is used for other camera motion. As another approach, two types of 3D point representation is used dependent on camera motion [69]. Points which can be observed with large disparities are represented as 3D points, and points which cannot be observed with large disparities are represented as 3D rays. In the tracking process, 3D ray information is also used to estimate camera motion. They use distances between 3D rays and feature points in the input image as reprojection errors.

Note that purely rotational motion is not a problem in RGB-D vSLAM. This is because tracking and mapping processes can be done by using obtained depth maps. On the other hand, monocular camera-based vSLAM cannot continue mapping during pure rotation movement.

7.2 Map initialization

Map initialization is important to achieve accurate estimation in vSLAM. Basically, in order to obtain an accurate initial map, baseline should be wide. However, in practical scenarios, it may be difficult to do ideal camera motion by novice people. To solve this problem, Mulloni et al. proposed an user-friendly initialization [70]. They used 2D/3D guides for instructing ideal camera motion for map initialization. Arth et al. proposed 2.5D map-based initialization for outdoor environments [71]. By using this method, vSLAM can be initialized in a global coordinate system on the earth.

Reference objects such as fiducial markers and known 3D objects have also been used to get a global coordinate system, and initial camera poses are estimated by tracking reference objects. In order to extend a trackable area, vSLAM is incorporated with it. Vuforia[4] provides marker-based SLAM initialization. In literature [72, 73], they use a known 3D object as a reference, and the known object shape is used to refine the map.

7.3 Estimating intrinsic camera parameters

Most vSLAM algorithms assume known intrinsic camera parameters. This means that camera calibration should be done before using vSLAM applications, and intrinsic camera parameters should be fixed during vSLAM estimation process. However, it is annoying for novice people. In the literature, they achieve intrinsic camera parameter estimation during vSLAM [74]. Intrinsic camera parameters are gradually converged during vSLAM estimation process. On the other hand, intrinsic camera parameter change can be handled [75]. They remove camera zooming effect by estimating focal length change based on an offline self-calibration technique [76].

7.4 Rolling shutter distortion

To achieve accurate camera pose estimation, it is important to consider a shutter type. Most vSLAM algorithms assume a global shutter, and these algorithms estimate one camera pose for each frame. However, most consumer cameras including RGB-D cameras employ rolling shutter due to its cost. In rolling shutter cameras, each row of a captured image is taken by different camera poses. It is obviously difficult to estimate camera poses of each row directly. In general, an interpolation-based approach is used to estimate rolling shutter camera pose estimation. In the literature [77–79], they use a spline function to interpolate a camera trajectory.

7.5 Scale ambiguity

Absolute scale information is needed in some vSLAM applications with monocular vSLAM. In order to obtain absolute scale information, user's body is used in the literature [80, 81]. Lee et al. used user's hand to determine an absolute scale and a global coordinate system [80]. Knorr et al. used user's face information to determine the absolute scale [81]. There is an assumption such that the size difference of these body parts is small within people. Therefore, these vSLAM systems can estimate scale information accurately.

As another approach, several sensors such as accelerometer, gyro, and magnetic sensor on mobile phones can also be used. In the literature [82], scale information is estimated by using accelerometer. They use frequency-domain filtering technique to remove sensor noise.

8 Benchmarking

To achieve fair comparison between vSLAM algorithms, benchmarking is obviously important and its methodologies have been discussed in recent years. Here, we introduce some benchmarking dataset as follows.

TrakMark provides image sequences with 6 DoF camera motion and intrinsic camera parameters [83]. In TrakMark, image sequences are divided into three scenarios: virtualized environments, indoor environments, and outdoor environments[5]. TrakMark assumes to be used for evaluating a performance of vSLAM algorithms in AR/MR research community. They also proposed an evaluation criteria from AR/MR research perspective. In AR/MR applications, image space errors are the most important because it is OK if the overlay of virtual objects onto an image is natural. In TrakMark, they employed the projection error of virtual object (PEVO) as a criteria for evaluating vSLAM algorithms [84]. In this criteria, virtual points are projected onto the input images using estimated and ideal camera poses, and then, distances are measured in the image space.

Martull et al. newly provided a stereo dataset which follows Tsukuba dataset [85]. Tukuba stereo dataset has been used for evaluating stereo algorithms. They created new Tukuba stereo dataset using computer graphics. Image sequences, camera poses, and depth maps for each frame are provided in the dataset. Image sequences are created using different camera trajectories and lighting conditions.

TUM RGB-D benchmarking dataset provides RGB-D image sequences with 6 DoF camera poses [86]. Camera poses are obtained using a motion capture system, which can be considered more accurate than vSLAM. They propose relative pose error (PRE) and absolute trajectory error (ATE) for evaluating local and global errors, respectively.

KITTI dataset is designed for evaluating vision systems in a driving scenario and includes many types of data [87]. In the dataset, visual odometry dataset is provided. Ground truth camera poses are obtained using RTK-GPS. In KITTI dataset webpage[6], evaluation results are listed. The results of LSD-SLAM and ORB-SLAM algorithms can be found in the Web page.

In contrast to other dataset, SLAMBench provides a framework for evaluating vSLAM algorithms from accuracy and energy consumption [88]. In addition, KinectFusion implementation is included in SLAMBench in different options (C++, OpenMP, OpenCL, and CUDA).

On-site benchmarking have been organized in International Symposium on Mixed and Augmented Reality (ISMAR) since 2008, which is called "tracking competition." In the tracking competition, participants need to do specific tasks given by organizers using own vSLAM systems. Unlike dataset-based evaluation, participants can control

Table 1 Comparison of representative algorithms

	Method	Map density	Global optimization	Loop closure
Mono-SLAM [26]	Feature	Sparse	No	No
PTAM [15]	Feature	Sparse	Yes	No
ORB-SLAM [38]	Feature	Sparse	Yes	Yes
DTAM [43]	Direct	Dense	No	No
LSD-SLAM [21]	Direct	Semi-dense	Yes	Yes
SVO [51]	Semi-direct	Sparse	No	No
DSO [53]	Direct	Sparse	No	No
KinectFusion [57]	RGB-D	Dense	No	No
Dense visual SLAM [23]	RGB-D	Dense	Yes	Yes
ElasticFusion [66]	RGB-D	Dense	Yes	Yes
SLAM++ [60]	RGB-D	Dense	Yes	Yes

camera movement based on current tracking results. Therefore, the tracking competition can evaluate vSLAM algorithms as an interactive system.

9 Conclusions

In this paper, we introduced recent vSLAM algorithms mainly from 2010 to 2016. Basically, vSLAM algorithms are composed of initialization, camera motion estimation, 3D structure estimation, global optimization, and relocalization. Recently, direct methods are active research field in monocular vSLAM. RGB-D vSLAM has also been developed in recent years because many consumer RGB-D cameras can be obtained with a cheap price. In AR/MR research community, practical problems have been solved. Even though vSLAM algorithms have been developed since 2003, vSLAM is still an active research field.

To understand the difference between different methods, those modules should be compared. Table 1 shows the summary of representative methods. Each algorithm has different characteristics. We need to choose an appropriate algorithm by considering a purpose of an application.

This paper focused on recent vSLAM algorithms using cameras only. As another approach, SLAM algorithms which are using visual and inertial data are called visual-inertial SLAM. By combining visual and inertial data, we can get more stable estimation results. Also, in the literature [77, 82], they are using sensor information to solve scale estimation and rolling shutter distortion compensation. Currently, smartphone and tablet devices have cameras, GPS, gyroscope, and accelerometer. In the future, we believe sensor fusion is one direction to realize robust and practical vSLAM systems.

To learn the elements of vSLAM algorithms, we provide ATAM[7] which is a vSLAM toolkit for beginners [89]. It includes monocular vSLAM algorithm including real scale estimation from a chessboard. Users can easily install and modify ATAM because the source code was designed to be well structured and only dependent on OpenCV [90] and cvsba [91].

Endnotes

[1] https://github.com/raulmur/ORB_SLAM2.
[2] https://get.google.com/tango/.
[3] https://structure.io/.
[4] https://developer.vuforia.com/.
[5] http://trakmark.net.
[6] http://www.cvlibs.net/datasets/kitti/.
[7] https://github.com/CVfAR/ATAM.

Authors' contributions
TT and HU collected and summarized visual SLAM papers. SI is an adviser and helped to draft the manuscript. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, 630-0192 Nara, Japan. [2]Kyushu University, 744 Motooka, Nishi-ku, 819-0395 Fukuoka, Japan. [3]Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, 525-8577 Shiga, Japan.

Received: 16 January 2017 Accepted: 17 May 2017
Published online: 02 June 2017

References
1. Chatila R, Laumond JP (1985) Position referencing and consistent world modeling for mobile robots. In: Proceedings of International Conference on Robotics and Automation Vol. 2. pp 138–145
2. Durrant-Whyte H, Bailey T (2006) Simultaneous localization and mapping: part i. Robot Autom Mag IEEE 13(2):99–110
3. Bailey T, Durrant-Whyte H (2006) Simultaneous localization and mapping (slam): Part ii. IEEE Robot Autom Mag 13(3):108–117
4. Thrun S, Leonard JJ (2008) Handbook of robotics. Chap. Simultaneous localization and mapping

5. Aulinas J, Petillot YR, Salvi J, Lladó X (2008) The slam problem: a survey. In: Proceedings of Conference on Artificial Intelligence Research and Development: Proceedings of International Conference of the Catalan Association for Artificial Intelligence. pp 363–371
6. Billinghurst M, Clark A, Lee G (2015) A survey of augmented reality. Found Trends Human-Computer Interact 8(2-3):73–272
7. Engel J, Sturm J, Cremers D (2012) Camera-based navigation of a low-cost quadrocopter. In: Proceedings of International Conference on Intelligent Robots and Systems. pp 2815–2821
8. Ros G, Sappa A, Ponsa D, Lopez AM (2012) Visual slam for driverless cars: a brief survey. In: Intelligent Vehicles Symposium (IV) Workshops
9. Fuentes-Pacheco J, Ruiz-Ascencio J, Rendón-Mancha JM (2015) Visual simultaneous localization and mapping: a survey. Artif Intell Rev 43(1):55–81
10. Klette R, Koschan A, Schluns K (1998) Computer vision: three-dimensional data from images. 1st edn
11. Nister D (2004) A minimal solution to the generalised 3-point pose problem. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Vol. 1. pp 560–5671
12. Grisetti G, Kümmerle R, Stachniss C, Burgard W (2010) A tutorial on graph-based slam. Intell Transp Syst Mag IEEE 2(4):31–43
13. Kümmerle R, Grisetti G, Strasdat H, Konolige K, Burgard W (2011) g2o: A general framework for graph optimization. In: Proceedings of International Conference on Robotics and Automation. pp 3607–3613
14. Bundle adjustment a modern synthesis. In: Triggs B, McLauchlan PF, Hartley RI, Fitzgibbon AW (eds) (2000) Vision algorithms: theory and practice. pp 298–372
15. Klein G, Murray DW (2007) Parallel tracking and mapping for small AR workspaces. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 225–234
16. Nistér D, Naroditsky O, Bergen J (2004) Visual odometry. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Vol. 1. p 652
17. Yousif K, Bab-Hadiashar A, Hoseinnezhad R (2015) An overview to visual odometry and visual slam: applications to mobile robotics. Intell Ind Syst 1(4):289–311
18. Scaramuzza D, Fraundorfer F (2011) Visual odometry [tutorial]. Robot Autom Mag IEEE 18(4):80–92
19. Fraundorfer F, Scaramuzza D (2012) Visual odometry: Part ii: matching, robustness, optimization, and applications. Robot Autom Mag IEEE 19(2):78–90
20. Engel J, Sturm J, Cremers D (2013) Semi-dense visual odometry for a monocular camera. In: Proceedings of International Conference on Computer Vision. pp 1449–1456
21. Engel J, Schöps T, Cremers D (2014) LSD-SLAM: large-scale direct monocular SLAM. In: Proceedings of European Conference on Computer Vision. pp 834–849
22. Kerl C, Sturm J, Cremers D (2013) Robust odometry estimation for RGB-D cameras. In: Proceedings of International Conference on Robotics and Automation. pp 3748–3754
23. Kerl C, Sturm J, Cremers D (2013) Dense visual SLAM for RGB-D cameras. In: Proceedings of International Conference on Intelligent Robots and Systems. pp 2100–2106
24. Agarwal S, Furukawa Y, Snavely N, Simon I, Curless B, Seitz SM, Szeliski R (2011) Building rome in a day. Commun ACM 54(10):105–112
25. Civera J, Grasa OG, Davison AJ, Montiel J (2010) 1-point ransac for extended kalman filtering: application to real-time structure from motion and visual odometry. J Field Robot 27(5):609–631
26. Davison AJ (2003) Real-time simultaneous localisation and mapping with a single camera. In: Proceedings of International Conference on Computer Vision. pp 1403–1410
27. Davison AJ, Reid ID, Molton ND, Stasse O (2007) Monoslam: real-time single camera SLAM. Pattern Anal Mach Intell IEEE Trans 29(6):1052–1067
28. Nistér D (2004) An efficient solution to the five-point relative pose problem. Pattern Anal Mach Intell IEEE Trans 26(6):756–770
29. Williams B, Klein G, Reid I (2007) Real-time SLAM relocalisation. In: Proceedings of International Conference on Computer Vision. pp 1–8
30. Castle R, Klein G, Murray DW (2008) Video-rate localization in multiple maps for wearable augmented reality. In: 2008 12th IEEE International Symposium on Wearable Computers. pp 15–22. doi:10.1109/ISWC.2008.4911577
31. Klein G, Murray DW (2009) Parallel tracking and mapping on a camera phone. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 83–86
32. Newcombe RA, Davison AJ (2010) Live dense reconstruction with a single moving camera. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. pp 1498–1505
33. Pradeep V, Rhemann C, Izadi S, Zach C, Bleyer M, Bathiche S (2013) MonoFusion: real-time 3D reconstruction of small scenes with a single web camera. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 83–88
34. Strasdat H, Montiel JM, Davison AJ (2012) Visual SLAM: why filter? Image Vision Comput 30(2):65–77
35. Mei C, Sibley G, Cummins M, Newman P, Reid I (2009) A constant-time efficient stereo slam system. In: Proceedings of British Machine Vision Conference. pp 54–15411
36. Cummins M, Newman P (2008) FAB-MAP: probabilistic localization and mapping in the space of appearance. Int J Robot Res 27(6):647–665
37. Strasdat H, Montiel J, Davison AJ (2010) Scale drift-aware large scale monocular slam. In: Proceedings of Robotics: Science and Systems. p 5
38. Mur-Artal R, Montiel JMM, Tardós JD (2015) ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Trans Robot 31(5):1147–1163. doi:10.1109/TRO.2015.2463671
39. Mur-Artal R, Tardós JD (2016) ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. CoRR. abs/1610.06475
40. Eade E, Drummond T (2009) Edge landmarks in monocular slam. Image Vis Comput 27(5):588–596
41. Hirose K, Saito H (2012) Fast line description for line-based SLAM. In: Proceedings of the British Machine Vision Conference
42. Klein G, Murray D (2008) Improving the agility of keyframe-based SLAM. In: Proceedings of European Conference on Computer Vision. pp 802–815
43. Newcombe RA, Lovegrove SJ, Davison AJ (2011) DTAM: dense tracking and mapping in real-time. In: Proceedings of International Conference on Computer Vision. pp 2320–2327
44. Okutomi M, Kanade T (1993) A multiple-baseline stereo. Pattern Anal Mach Intell IEEE Trans 15(4):353–363
45. Rudin LI, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. Phys D Nonlinear Phenom 60(1):259–268
46. Ondruska P, Kohli P, Izadi S (2015) MobileFusion: real-time volumetric surface reconstruction and dense tracking on mobile phones. IEEE Trans Vis Comput Graph 21(11):1251–1258
47. Stühmer J, Gumhold S, Cremers D (2010) Real-time dense geometry from a handheld camera. In: Goesele M, Roth S, Kuijper A, Schiele B, Schindler K (eds). Springer, Berlin, Heidelberg, pp 11–20
48. Schöps T, Engel J, Cremers D (2014) Semi-dense visual odometry for AR on a smartphone. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 145–150
49. Caruso D, Engel J, Cremers D (2015) Large-scale direct SLAM for omnidirectional cameras. In: Proceedings of International Conference on Intelligent Robots and Systems
50. Engel J, Stueckler J, Cremers D (2015) Large-scale direct SLAM with stereo cameras. In: Proceedings of International Conference on Intelligent Robots and Systems
51. Forster C, Pizzoli M, Scaramuzza D (2014) SVO: fast semi-direct monocular visual odometry. In: Proceedings of International Conference on Robotics and Automation. pp 15–22
52. Baker S, Matthews I (2004) Lucas-kanade 20 years on: a unifying framework. Int J Comput Vis 56(3):221–255
53. Engel J, Koltun V, Cremers D (2016) Direct sparse odometry. CoRR. abs/1607.02565
54. Geng J (2011) Structured-light 3d surface imaging: a tutorial. Adv Opt Photon 3(2):128–160
55. Zhang Z (2012) Microsoft kinect sensor and its effect. MultiMedia IEEE 19(2):4–10
56. Besl PJ, McKay ND (1992) A method for registration of 3-d shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256
57. Newcombe RA, Izadi S, Hilliges O, Molyneaux D, Kim D, Davison AJ, Kohi P, Shotton J, Hodges S, Fitzgibbon A (2011) KinectFusion: real-time dense surface mapping and tracking. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 127–136
58. Kahler O, Prisacariu V, Ren C, Sun X, Torr P, Murray D (2015) Very high frame rate volumetric integration of depth images on mobile devices. IEEE Trans Vis Comput Graph 21(11):1241–1250

59. Salas-Moreno RF, Glocker B, Kelly PHJ, Davison AJ (2014) Dense planar SLAM. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 157–164
60. Salas-Moreno RF, Newcombe RA, Strasdat H, Kelly PHJ, Davison AJ (2013) SLAM++: simultaneous localisation and mapping at the level of objects. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. pp 1352–1359
61. Tateno K, Tombari F, Navab N (2016) When 2.5D is not enough: Simultaneous reconstruction, segmentation and recognition on dense SLAM, IEEE International Conference on Robotics and Automation (ICRA). pp 2295–2302
62. Henry P, Krainin M, Herbst E, Ren X, Fox D (2012) RGB-D mapping: using Kinect-style depth cameras for dense 3D modeling of indoor environments. Int J Robot Res 31(5):647–663
63. Endres F, Hess J, Engelhard N, Sturm J, Cremers D, Burgard W (2012) An evaluation of the RGB-D SLAM system. In: Proceedings of International Conference on Robotics and Automation. pp 1691–1696
64. Steinbrücker F, Sturm J, Cremers D (2011) Real-time visual odometry from dense RGB-D images. In: Proceedings of IEEE International Conference on Computer Vision Workshops. pp 719–722
65. Whelan T, Kaess M, Leonard JJ, McDonald J (2013) Deformation-based loop closure for large scale dense RGB-D SLAM. In: Proceedings of International Conference on Intelligent Robots and Systems. pp 548–555
66. Whelan T, Leutenegger S, Moreno RS, Glocker B, Davison A (2015) ElasticFusion: dense slam without a pose graph. In: Proceedings of Robotics: Science and Systems. doi:10.15607/RSS.2015.XI.001
67. Gauglitz S, Sweeney C, Ventura J, Turk M, Höllerer T (2012) Live tracking and mapping from both general and rotation-only camera motion. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 13–22
68. Pirchheim C, Schmalstieg D, Reitmayr G (2013) Handling pure camera rotation in keyframe-based SLAM. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 229–238
69. Herrera C, Kim K, Kannala J, Pulli K, Heikkilä J, et al (2014) Dt-slam: deferred triangulation for robust SLAM. In: Proceedings of 3D Vision Vol. 1. pp 609–616
70. Mulloni A, Ramachandran M, Reitmayr G, Wagner D, Grasset R, Diaz S (2013) User friendly SLAM initialization. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 153–162
71. Arth C, Pirchheim C, Ventura J, Schmalstieg D, Lepetit V (2015) Instant outdoor localization and SLAM initialization from 2.5 d maps. IEEE Trans Vis Comput Graph 21(11):1309–1318
72. Bleser G, Wuest H, Stricker D (2006) Online camera pose estimation in partially known and dynamic scenes. In: Proceedings of International Symposium on Mixed and Augmented Reality. pp 56–65
73. Tamaazousti M, Gay-Bellile V, Collette SN, Bourgeois S, Dhome M (2011) Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. pp 3073–3080
74. Civera J, Bueno DR, Davison A, Montiel JMM (2009) Camera self-calibration for sequential bayesian structure from motion. In: Proceedings of International Conference on Robotics and Automation. pp 403–408
75. Taketomi T, Heikkilä J (2015) Focal length change compensation for monocular slam. In: Proceedings of International Conference on Image Processing. pp 1–5
76. Pollefeys M, Koch R, Gool LV (1999) Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. Int J Comput Vis 32(1):7–25
77. Lovegrove S, Patron-Perez A, Sibley G (2013) Spline Fusion: a continuous-time representation for visual-inertial fusion with application to rolling shutter cameras. In: Proceedings British Machine Vision Conference. pp 93.1–93.12
78. Kerl C, Stueckler J, Cremers D (2015) Dense continuous-time tracking and mapping with rolling shutter RGB-D cameras. In: Proceedings of International Conference on Computer Vision. pp 2264–2272
79. Kim JH, Cadena C, Reid I (2016) Direct semi-dense SLAM for rolling shutter cameras. In: Proceedings of International Conference on Robotics and Automation. pp 1308–1315
80. Lee T, Höllerer T (2008) Multithreaded hybrid feature tracking for markerless augmented reality. IEEE Trans Vis Comput Graph 15(undefined):355–368
81. Knorr SB, Kurz D (2016) Leveraging the user's face for absolute scale estimation in handheld monocular SLAM. In: Proceedngs of International Symposium on Mixed and Augmented Reality. pp 11–17
82. Mustaniemi J, Kannala J, Särkkä S, Matas J, Heikkilä J (2016) Inertial-based scale estimation for structure from motion on mobile devices. CoRR. abs/1611.09498
83. Tamura H, Kato H (2009) Proposal of international voluntary activities on establishing benchmark test schemes for ar/mr geometric registration and tracking methods. In: International Symposium on Mixed and Augmented Reality. pp 233–236
84. Makita K, Okuma T, Ishikawa T, Nigay L, Kurata T (2012) Virtualized reality model-based benchmarking of AR/MR camera tracking methods in TrakMark. In: IEEE ISMAR 2012 Workshop on Tracking Methods and Applications. pp 1–4
85. Martull S, Martorell MP, Fukui K (2012) Realistic CG stereo image dataset with ground truth disparity maps. In: ICPR2012 Workshop TrakMark2012. pp 40–42
86. Sturm J, Engelhard N, Endres F, Burgard W, Cremers D (2012) A benchmark for the evaluation of RGB-D SLAM systems. In: Proceedings of International Conference on Intelligent Robots and Systems
87. Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? The kitti vision benchmark suite. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition
88. Nardi L, Bodin B, Zia MZ, Mawer J, Nisbet A, Kelly PHJ, Davison AJ, Luján M, O'Boyle MFP, Riley GD, Topham N, Furber SB (2015) Introducing SLAMBench, a performance and accuracy benchmarking methodology for SLAM. In: IEEE International Conference on Robotics and Automation. pp 5783–5790
89. Uchiyama H, Taketomi T, Ikeda S, do Monte Lima JPS (2015) [POSTER] Abecedary tracking and mapping: a toolkit for tracking competitions. In: IEEE International Symposium on Mixed and Augmented Reality. pp 198–199
90. Open Source Computer Vision. http://opencv.org/. Accessed 24 May 2017
91. cvsba: an OpenCV wrapper for sba library. http://www.uco.es/investiga/grupos/ava/node/39. Accessed 24 May 2017