Journal of Computer Science and Engineering Research (JCSER)
Volume 2, Issue 1, Date: 30 - June – 2025

A Full Overview of Visual SLAM Algorithms

Rajaa W. Ali [1], Heba Hakim. [2], Dr. Mohammed A. Al-Ibadi [3]
[1, 2, 3] Computer Department, Basrah University, Basrah, Iraq
Email: [1] pgs.rajaa.wejood@uobasrah.edu.iq, [2] hiba.abdulzahrah@uobasrah.edu.iq, [3] mohammed.joudah@uobasrah.edu.iq

***Abstract*—Simultaneous Localization and Mapping, or SLAM, is an essential approach for autonomous robotic systems. Simultaneous mapping and sensor pose estimation are made possible by SLAM in an unknown environment. Visual simultaneous localization and mapping, often known as V-SLAM, is an important field in robotics, particularly for cooperative and interactive mobile robot environments. Faster development of Visual SLAM can be attributed to low-cost sensors, easy integration of additional sensors, and improved ambient information. Numerous strategies and techniques for implementing visual-based SLAM systems are presented in the literature. It might be challenging for a novice in this field to sort through the range of publications, recognize and evaluate the key algorithms, and ultimately select the best one for their intended use. Therefore, we present the three main visual-based SLAM approaches (visual alone, visual inertial, and RGB-D SLAM), emphasizing their salient features and limitations. We also use flowcharts and diagrams to examine the main algorithms of each approach. It tracks the development of SLAM techniques historically and offers contrastive evaluations of concepts and salient ideas. The research examines important Visual SLAM benchmark datasets and offers process-level visualization for every method. This research aims to cover the essential elements and characteristics of SLAM methodologies, providing a foundational resource for understanding and selecting appropriate techniques.**

Keywords—Benchmark; Mobile Robots; RGB-D SLAM; Visual-SLAM; Visual-Inertial SLAM.

I. INTRODUCTION (*HEADING 1*)

In 1986, Smith established the notion of simultaneous localization and mapping (SLAM), which is a basic need for many robotic applications [1,2]. SLAM technology allows mobile robots to generate an environment map and estimate their own location in real time, without requiring any prior environmental information, based on inputs from one or more sensors. In robotics, mapping is essential since it makes landmarks visible and simplifies the use of SLAM. Because of its capacity to carry out navigation and perception concurrently in an unfamiliar area, SLAM has grown in popularity over the last several decades and drawn the interest of numerous scholars [3]. Based on the specific data collecting tools they employ, there are two major categories of SLAM systems in use today.

The foundation of the first kind is provided by light detection and ranging, or LiDAR, sensors [4]. Most autonomous vehicles employ expensive LiDAR based SLAM. The second type, known as visual SLAM [5], offers cheap flow costs and a small volume as benefits. It utilizes an image sensor. It may offer motion estimate if texture data were more plentiful. the portrayal of the environment in visual form. Additionally, when returning to previously recorded places, it might help in estimating the robot's state, moving it, and minimizing estimate errors [8].When the global positioning system (GPS) is unavailable, like in interior scenarios, visual SLAM plays a crucial role because of its rapid ambient awareness and autonomous localization capabilities [6, 7].

The map development process also involves two additional tasks: localization and route planning. According to Stachniss [9], path planning, localization, and mapping are essential functions that allow a robot to comprehend its environment, ascertain its location, and create paths to certain destinations. SLAM is one technique that combines the mapping and localization phases. SLAM algorithms employ data from several sensors. Visual SLAM, or just using visual sensors, may require the use of a monocular RGB camera [18], a stereo camera [19], or an omnidirectional camera (which takes simultaneous photos in all 360-degree directions) [20]. As a result of their restricted visual input, they are more technically demanding [10], or RGB-D cameras (RGB-D SLAM) capture RGB images in addition to depth pixel data [21]. Visual-inertial (VI) SLAM, an inertial measurement unit (IMU) that is small, low-cost, and achieves high accuracy, is essential for many applications that demand lightweight design. It is a crucial component for several applications, including driverless racing vehicles, that require lightweight construction [11].

In order to provide an overview and a basic understanding of the problem of simultaneous localization and mapping (SLAM), Bailey and Durrant-Whyte [12] investigate the recursive Bayesian formulation of the SLAM problem. This approach produces probability distributions, vehicle posture estimates, and absolute or relative landmark placements. A concise synopsis of the graph-based SLAM problem is given by Grisetti et al. (2008). To provide SLAM solution methodologies in mobile robots and its wide application, Taheri et al. [13] provide a useful survey and an effective overview. Basheer et al. [14], Macario Barros et al. [15] separated VSLAM techniques into three classes: visual only (monocular), visual inertial (stereo), and RGB-D SLAM. This division was made in consideration of the studies and surveys of visual aids. Additionally, they put out a number of criteria for decomposing and examining VSLAM algorithms. The first review of VI-SLAM approaches from both an optimization based and filtering-based standpoint is Chen et al. [16]. The RGB-D SLAM system's core concept and structure were first presented by Zhang et al. [17]. Basheer et al. [14] additionally, focuses on the integration of the robotic environment with a robot operating system (ROS) as Middleware. Additionally, Macario Barros et al.

Received: 1-12-2024
Revised: 25-6-2025
Published: 30-6-2025

[15] offer a summary of each approach's primary algorithms using flowcharts and diagrams. Taketomi et al. [23] and Covolan et al. [22] focus on visual only and RGB-D-based techniques and outline the key algorithms, providing an overview of the key ideas utilized in the visual based SLAM systems. al. Servières [24] give a summary of the current V-SLAM and VI-SLAM designs before going on to classify a fresh batch of twelve primary state-of-the-art techniques. Robots that are mobile and adaptive enough to function successfully in new surroundings are essential in today's society. Thus, simultaneous localization and mapping, or SLAM, is an important technique for these robots. Durrant-Whyte (2012) and Mohamed et al. (2008) state that the primary goal of SLAM is to allow for autonomous exploration and navigation of foreign environments by simultaneously creating a map and determining the user's location. It can also make decisions in real-time, so robots don't need to refer to previously made maps. The capacity of the robot to perceive and successfully interact with its environment is enhanced by its usefulness in the extraction, organization, and interpretation of data. Describe the RGB-D SLAM system's fundamental idea and architecture. Previous research has demonstrated the effectiveness of V-SLAM techniques; nevertheless, they are often provided with limited data and unique figures, making it challenging to understand, assess, and select one from the group. Therefore, our effort focuses on simplifying the descriptions of V-SLAM techniques to make them easier for readers to grasp. The main contributions of the study are summarized as follows:

- Examining V-SLAM techniques to identify the most effective robotics tools.
- In order to enhance comprehension of the operational procedures associated with V-SLAM, a graphical and illustrative structural workflow was developed for every approach.
- Determining key factors for the V-SLAM approaches' evaluation and selection criteria.
- Making a table of comparisons with the salient features and parameters of each V-SLAM technique.
- The discussion and display of relevant datasets used in the robotics application domain.

The paper is organized as follows: An introduction of the V-SLAM paradigm that explores its core ideas is provided in Section 2. The key ideas of the three chosen techniques are presented in Section 3. Section 4 delves into the development of V-SLAM and examines the datasets that are most frequently utilized. The guidelines for assessing and choosing visual SLAM techniques are covered in Section 5. The article's conclusion, which summarizes the most important ideas, is found in Section 6.

II. VISUAL BASED SLAM TECHNIQUIES

Three primary processes are involved in visual-based SLAM systems, which employ cameras to create 3D maps from 2D images: initialization, tracking, and mapping (Fig. 1) [10]. Initialization produces an initial map and establishes a global coordinate system. By comparing 2D–3D correspondences, tracking keeps the camera in relation to the map and frequently resolves the Perspective-n-Point (PnP) issue [25, 26]. When additional regions are viewed, mapping enlarges the map. The majority of V-SLAM algorithms rely on intrinsic camera parameters that have been pre-calibrated, whereas extrinsic parameters (rotation and translation) determine camera positions.

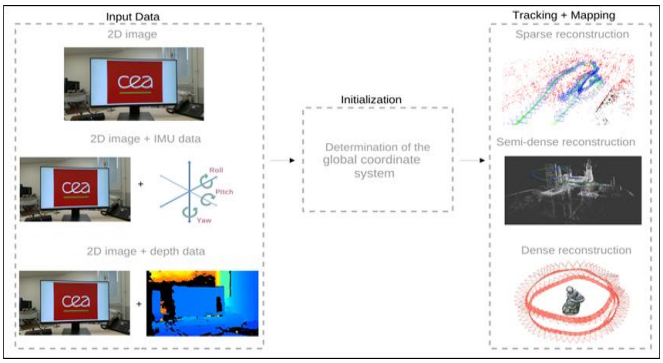

Fig. 1 General elements of a vision-based SLAM. A dense reconstruction (Reprinted from [30]), a semi-dense map, and a sparse map in the MH_01 sequence [28]). Taken from [15].

A 2D image, a 2D image with depth data, or both can be the input for a visual-SLAM system, as shown in Figure 1, depending on the technique employed (visual only, visual inertial, or RGB-D based, respectively). Among the situations that this system may be used to efficiently build and implement are semantic segmentation [32], pixel-wise motion segmentation [31], and filtering techniques [33, 34]. These methods seek to provide a professional approach for an image of the V-SLAM operations. It makes sense to separate the operational framework into four parts, which are listed and covered below.

A. Setting up the System and Collecting Data

In this step of V-SLAM, which involves capturing and processing images. It involves setting up cameras such as RGB-D cameras, depth cameras, or infrared sensors for data gathering and system setup [35]. Camera calibration, which is often the first step in system startup, determines intrinsic and extrinsic properties for accurate mapping and localisation. Effective initialization techniques are essential for precise SLAM tracking and mapping because they minimize error propagation and frequently make use of pre-existing data or manual starting locations [36]. A properly calibrated and initialized system is crucial for efficient VSLAM performance, as demonstrated by the fact that the choice of suitable data acquisition techniques and initialization strategies directly affects the system's capacity to handle a variety of dynamic and varied environments [37].

B. Localization of the System

The second stage of V-SLAM, an important phase in the process overall, has as its main objective determining the position of the system [38]. Pose estimation [42], or localization, is the process by which the system precisely determines where it is in the environment. This step entails estimating the camera's position and orientation in space with respect to the previously constructed map using data from the visual sensor, which is often a camera. Using methods like ORB (Oriented FAST and Rotated BRIEF) or

SIFT (Scale-Invariant Feature Transform), the system recognizes and compares important aspects from the current frame with those seen in previous frames. This phase, which is critical to maintaining the system's accuracy while navigating the environment, frequently incorporates loop closure detection to rectify drift by identifying and realigning with previously visited places [39,40,41]. This process relies on several key components: feature tracking, feature matching, relocalization, and pose estimation. Each of these components plays a crucial role in ensuring that the V-SLAM system can accurately localize itself within the environment, thereby maintaining a consistent and accurate map. To estimate camera motion in visual SLAM, detected characteristics are tracked throughout frames and compared to those in a map or earlier frames. Nearest neighbor search and descriptors aid in finding matches, whereas RANSAC eliminates untrustworthy ones. By comparing the current frame attributes with the map, relocalization recovers the camera's attitude once tracking is lost because to rapid movement or occlusion. Pose estimation, which aligns 2D picture points to their 3D counterparts using the Perspective-n-Point (PnP) technique, uses monitored and matched data to identify the exact camera location and orientation. V-SLAM systems achieve real-time localization and mapping through feature tracking, pose optimization, and relocalization, essential for applications in robotics, augmented reality, and autonomous vehicles.

C. System Map Creation

SLAM systems employ several mapping techniques, including occupancy grids and point clouds, depending on the kind of information, sensor, and application requirements [43, 44]. Localization and mapping work together to keep a trustworthy, current map. Robotics uses grid maps to simulate actual environments. Each cell in the map represents a specific place and stores information about barriers, geography, and occupancy. For robots, feature-based SLAM uses maps that depict environmental elements, such as landmarks, to help in localization and navigation [45, 46]. Specialized sensors produce a 3D point cloud, which visualizes the spatial arrangement to improve comprehension of the surroundings [47]. Keyframe setup during localization results in field modeling, where important spots and feature lines are found for the production of maps [48]. The map is updated in real-time as the robot's location is continually tracked [49]. A key component of feature-based SLAM, bundle adjustments (BAs) improve accuracy by fine-tuning the placement and structure of observed points [50–52].

D. Process Tuning & Loop Closure

Loop closures and system tuning are used to optimize the map in the latter stage of the V-SLAM process. Enhancing the system's precision, dependability, and long-term consistency requires process tuning and loop closures. Process tuning strikes a balance between accuracy, computing efficiency, and resilience by modifying and optimizing a variety of parameters and algorithms, such as pose estimation techniques and feature detection sensitivities. This entails optimizing algorithms and balancing resource limitations with performance, particularly in real-time systems like robots or augmented reality. The system's constant performance is ensured by extensive testing in various scenarios.

By re-aligning the system with previously mapped locations through pose graph optimization, loop closure in V-SLAM fixes map drift and preserves map accuracy, particularly in dynamic situations.

Fig 2 show an explanation of the procedures carried out within V-SLAM. The V-SLAM framework is made up of successive phases arranged to construct the system and process its data.

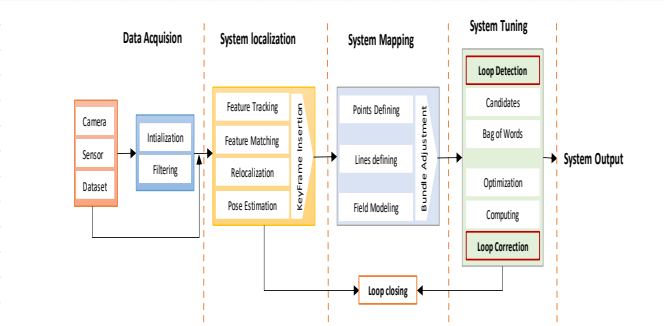

Fig. 2 Adapted from [53]. An overview of the four core components necessary for visual SLAM.

III. VISUAL SLAM MODEL

By utilizing cutting-edge sensors, deep learning, and machine learning, V-SLAM seeks to advance robotics by estimating camera motion and 3D structure in unfamiliar situations [54, 55]. The topology of V-SLAM is divided into three categories, as shown in Fig. 3: visual-inertial SLAM, RGB-D SLAM, and just visual SLAM [57]. Different methods are assessed according to six important criteria: algorithm type, map density, global optimization, loop closure, availability, and embedded implementations [15]. As demonstrated by applications like autonomous driving, the selection of the SLAM approach is contingent upon particular project requirements, including scalability and accuracy [33, 56]. The SLAM algorithms that we have chosen to showcase the best qualities of the three techniques are listed below, arranged by publication year.

A. Visual-Only SLAM

Map points are initialized with uncertainty before being refined by feature-based algorithms in visual-only SLAM systems, which depend on 2D image processing to establish a global coordinate system and rebuild maps. Although monocular cameras are preferred due to their compact size, low cost, and power economy, initialization, scaling, and drift are issues that they must deal with [27]. Although they are bigger and need more processing, stereo cameras can address some of these problems by giving stereo depth in a single image. Through the addition of depth information, increased 3D mapping precision, decreased drift, and support for strong feature matching in demanding situations, RGB-D cameras improve SLAM. Fig. 1 displays the chosen visual only SLAM algorithms, which are described in the next subsections.

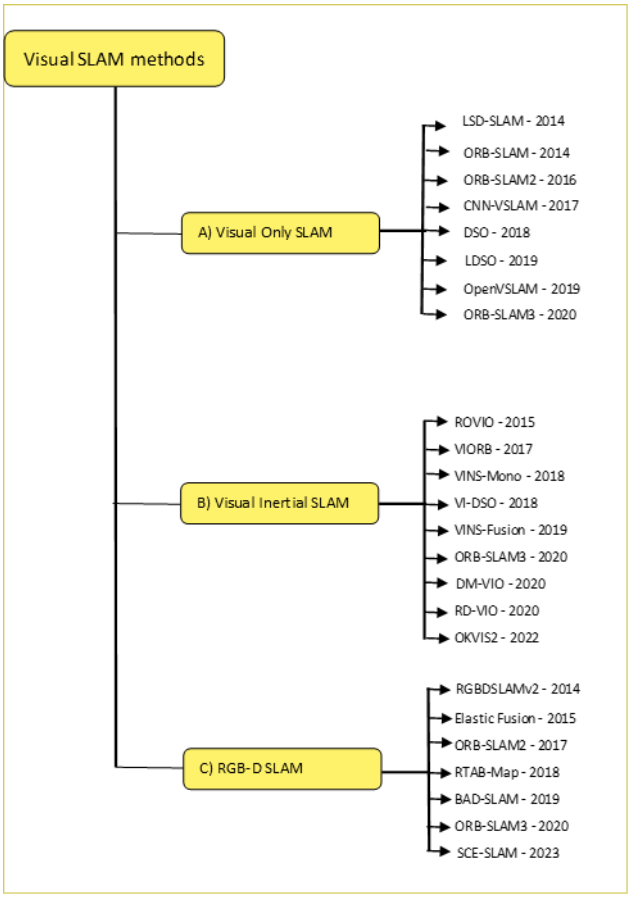

Fig. 3 Three examples of visual SLAM types include RGB-D SLAM, only-visual SLAM, and visual-inertial SLAM.

- ORB-SLAM - 2014

Oriented FAST and Rotated BRIEF SLAM, or ORB-SLAM, is a feature-based SLAM system that may be used in both small and large interior and outdoor settings [127]. Its real-time capabilities and high-quality map reconstruction make it a popular choice for applications such as autonomous navigation, augmented reality, and human-robot interaction [128]. The main features of ORB-SLAM, which can handle both static and dynamic motion clutter, are loop closure, mapping, and tracking [129].

In comparison to existing V-SLAM methods, ORB-SLAM achieves real-time global localization and camera re-localization across different views by enhancing map dynamics, size, and traceability [130,131]. While ORB-SLAM1 is categorized as only-visual, ORB-SLAM2 offers both only-visual and RGB-D SLAM [132,131], while ORB-SLAM3 adds visual-inertial SLAM, demonstrating its adaptability and applicability across a range of applications [133,134].

Four processes comprise the ORB-SLAM methodology: loop closure, local mapping, tracking and sensor input, and output preparation [135, 136]. Version-specific variations in the tracking step—ORB-SLAM1 uses one input, ORB-SLAM2 uses three, and ORB-SLAM3 uses four—have an impact on how well later procedures work. New map points and keyframes are introduced in local mapping, and ORB-SLAM3 enhances feature matching. In versions 2 and 3, the loop closing stage involves bundle adjustment welding and map merging. The output, which includes the required SLAM data and 2D/3D maps, is prepared in the last step [136].

- ORB-SLAM2 – 2016

ORB-SLAM2, a state-of-the-art feature-based algorithm, builds upon ORB-SLAM [59] and operates with three concurrent threads: tracking, local mapping, and loop closure. The tracking thread reduces reprojection error and locates the sensor, while the local mapping thread manages map-related tasks [60].

The loop closure thread in ORB-SLAM2 finds new loops and fixes drift, then adjusts the bundle for motion and structural consistency. For RGB-D, monocular, and stereo techniques, the algorithm employs loop closure and global optimization. But if comparable frames are not recognized, tracking problems may occur, and real-time operation on embedded systems is challenging since pictures must be processed at the same frame rate as they are obtained [61,62]. Figure 4 shows a diagram of the threads in the algorithm. A representation of the algorithm's threads can be found in Fig. 4. Despite the existence of several embedded implementations in the literature, this remains the case. The ORB-SLAM method was executed on a CPU by Yu et al. [63], while Abouzahir et al. [62] built the algorithm on several CPU- and GPU-based platforms and assessed each thread's performance on the platforms.

- CNN VSLAM - 2017

CNN SLAM [64] integrates convolutional neural networks with real-time SLAM by combining maps and depth from monocular SLAM with CNN-predicted semantic segmentation. It uses a key-frame based SLAM approach, where visually distinct frames are refined via pose graph optimization. The method estimates camera positions through frame-to-key-frame transformations, with depth prediction handling scale estimation. It also incorporates loop closure and global optimization. Real-time execution requires a CPU+GPU architecture, and the system's pipeline is illustrated in Figure 5.

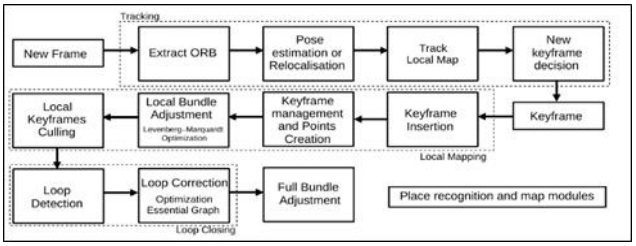

Fig. 4 Diagram representing the ORB-SLAM 2.0 algorithm. Adapted from [59]

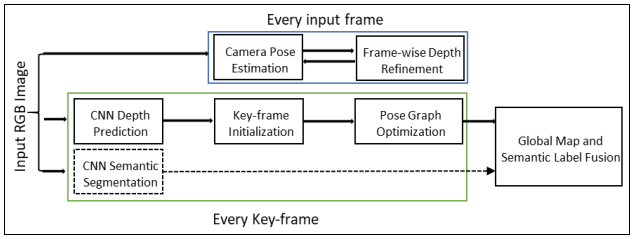

Fig. 5 Diagram representing the CNN-SLAM algorithm. Adapted from [64].

- Direct Sparse Odometry - 2018

A direct probabilistic model and camera motion are combined in Direct Sparse Odometry (DSO), a visual odometry approach that optimizes all model parameters, including geometry expressed as inverse depth. Through the use of an inverse depth map and keyframe window, it performs continuous optimization with local bundle modification while evenly sampling pixels in real-time. By adding posture and loop closure detection, Xiang et al. [67] expanded DSO. Fig. 6 depicts the main phases of DSO.

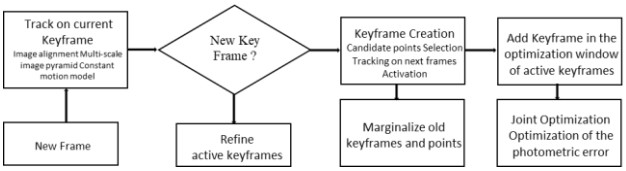

Fig. 6 Diagram representing the DSO algorithm. Adapted from [15]

Pose-graph optimization (DSO) and loop closure detection (LCDSO) are extended to monocular visual SLAM by LDSO, which prioritizes corner features in tracking to preserve robustness in featureless regions. The bag-of-words (BoW) approach is used to identify loop closure possibilities, which are then confirmed by geometric checks and relative pose constraints derived from the combined reduction of 2D and 3D mistakes. The DSO sliding window optimization's co-visibility graph is fused with these limitations.

- Open-VSLAM - 2019

Using ORB as a feature extractor and a graph-based algorithm akin to ORB-SLAM and ProSLAM, OpenVSLAM is a modular, monocular, stereo, and RGBD visual SLAM system [70]. As seen in Fig. 7, the OpenVSLAM program may be loosely categorized into three modules: tracking, mapping, and global optimization. The tracking module determines when to add a new keyframe, which is then sent to the mapping and optimization modules for additional processing, by predicting the camera attitude using posture optimization and keypoint matching.

By triangulating 3D points from keyframes (KFs) and carrying out local bundle adjustment (BA), the mapping module in OpenVSLAM enlarges the map. Pose-graph optimization, global BA, and loop closure are handled by the global optimization module, which uses the g2o optimization framework to solve trajectory and scale drift, especially for monocular camera models [71].

OpenVSLAM provides versatility by supporting map import/export, working with a variety of camera manufacturers and models, and having a cross-platform online browser. Its precision is inferior to that of ORB-SLAM3 and VINS-Fusion, and its absence of integrated loop closure and IMU support causes drift in rapid movements. It has poor real-time performance on low-power devices, is less suitable for large-scale mapping, and performs badly in dynamic or low-texture situations. It also has little community support.

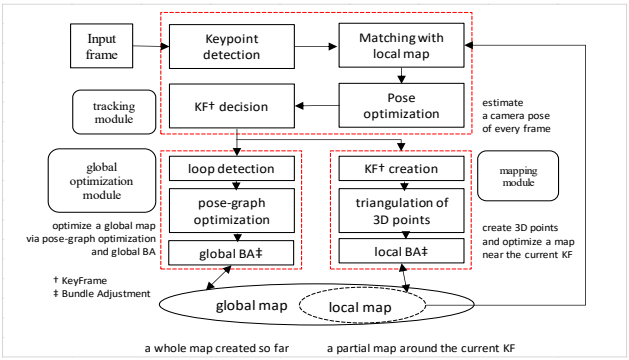

Fig. 7 Main modules of Open-VSLAM: tracking, mapping, and global optimization modules. Adapted from [71].

- ORB-SLAM3 - 2020

A method that combines the ORB-SLAM and VIORB [15] algorithms is the already-discussed ORB-SLAM3. The algorithm is separated into three primary threads, just like its predecessors: loop closure and map merging, rather than loop closing and tracking. Third, there is local mapping. Besides, ORB-SLAM3 [27] maintains an Atlas multi-map representation that includes non-active maps for location recognition and relocalization, as well as an active map utilized by the tracking thread. Map merging is introduced to the final thread, which adheres to the same logic as VIORB in the first two.

Depending on where the overlapping region is, the loop closing and map merging thread uses all of the Atlas maps to find common areas, execute loop correction, merge maps, and switch the active map. An additional significant feature of ORB-SLAM3 is the suggested initialization method, which uses the Maximum-a-Posteriori algorithm separately for the inertial and visual estimates before optimizing them combined. This approach applies loop closures and global optimizations techniques and may be utilized with monocular, stereo, and RGB-D cameras. On the other hand, considerable mistakes in ORB-SLAM3 online performance were shown by the authors in [72]. Although the system performed well in [73], it was unable to analyze all of the sequences and produced erroneous estimates for outdoor sequences.

- LSD-SLAM - 2014

LSD-SLAM is a large-scale, real-time direct monocular SLAM method that is intended for accurate mapping in dynamic settings. Applications like as robots and self-driving automobiles in complex and dynamic environments are perfect for it since it supports a variety of camera combinations and retains accuracy even at lower picture resolutions [29]. The five stages of the workflow used by LSD-SLAM and DVO-SLAM are identical and include data input, picture alignment, loop closure, map optimization, and global optimization. Real-time large-scale mapping is made possible by LSD-SLAM, which combines direct and semi-dense reconstruction approaches. To handle tracking activities efficiently, CPU + FPGA architectures were used in its implementation [137, 68]. But as LSD-SLAM depends on pose-graph optimization, PTAM and ORB-SLAM demonstrated greater accuracy in map estimation [132, 129]. The primary visual-only SLAM algorithms were covered in this section. The key traits and evaluated standards for the

suggested visual-only SLAM algorithms are enumerated in Table 1.

TABLE 1. MAIN ASPECTS RELATED TO THE VISUAL-ONLY SLAM APPROACHES.

Method	Type	Map Density	Global Optimization	Loop Closure	Availability
LSD	Direct	Semi-dense	Yes	Yes	[91]
ORB-SLAM	Feature-based	Sparse	Yes	Yes	[92]
ORB-SLAM2	Feature-based	Sparse	Yes	Yes	[93]
CNN-SLAM	Direct	Semi dense	Yes	Yes	[94]
DSO	Direct	Sparse	No	No	[95]
LDSO	Direct	Sparse	No	Yes	[96]
OpenVSLAM	Hybrid	Sparse	Yes	Yes	[97]
ORB-SLAM3	Feature-based	Sparse	Yes	Yes	[98]

B. *Visual-Inertial SLAM*

By combining inertial measurement units (IMUs) and visual sensors (such as stereo cameras), VI-SLAM improves system performance by producing a more precise and comprehensive description of the surroundings. This hybrid technique, which incorporates IMU data into the environment model, improves accuracy and decreases mistakes in real-world applications such as mobile robots and drones. The next subsections provide explanations of the chosen visual-inertial algorithms, whereas Fig. 3 displays a timeline of those methods.

- *Robust visual inertial odometry - ROVIO – 2015*

By combining optical and inertial data using sophisticated sensor fusion, ROVIO-SLAM [101] enhances navigation accuracy and improves interaction with the surroundings, making it perfect for long-term, low-cost robotic systems operating in difficult environments. To enable reliable mapping and positioning, the procedure consists of three steps [100]: gathering IMU and camera data, processing for feature identification and IMU integration, and producing estimated pose and 3D landmarks.

Despite being effective because of its tightly-coupled visual-inertial fusion, ROVIO lacks loop closure, which makes it less consistent over the long term than ORB-SLAM3 or VINS-Fusion. Lidar-based SLAMs perform better in low-light or texture-poor situations, where it is susceptible to visual deterioration. Furthermore, it performs worse in large-scale mapping, which makes it better suited for scenarios that are more closely regulated or small-scale.

- *Visual Inertial ORB-SLAM – VIORB – 2017*

Based on ORB-SLAM, VIORB [104] is a monocular VI-SLAM system that integrates ORB-based front-end and back-end operations such as graph optimization, loop closure, and relocation. By calculating gyro bias, then fine-tuning scale and gravity, accelerometer bias, and lastly velocity, it accurately initializes scale, velocity, gravity direction, and IMU biases using a special IMU initialization technique. It joins recent keyframes via a co-visibility graph and optimizes them using local bundle modification. Additionally, SLAM solutions that integrate IMU with RGB-D and stereo sensors have been investigated [105].

In the same context, monocular SLAM continuously localizes and recovers the metric scale with great precision, outperforming the state-of-the-art in stereo visual-inertial odometry. For virtual and augmented reality systems, where the expected user viewpoint must not change when the user is in the same workspace, Ra´ el Mur-Artal and Juan D. Tard´os [105] make this zero-drift localization more interesting. Using stereo or RGB-D cameras should help improve accuracy and robustness, and since scale is known, it would also make IMU configuration easier. Relying on the initialization of the monocular SLAM is VIORB IMU initialization's primary flaw.

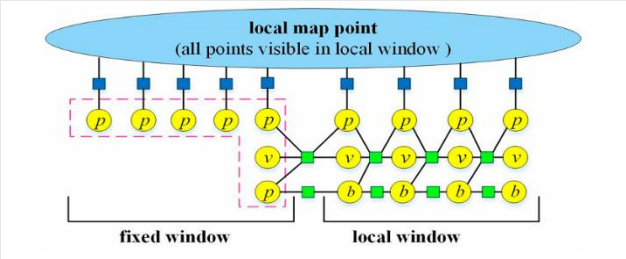

Fig. 8 Reproduced from [105], keyframe in the local map of Visual-Inertial ORB-SLAM.

- VINS-MONO - 2018

With just one camera and one IMU, the monocular visual-inertial system VINS-Mono [69] generates a metric six degrees-of-freedom (DOF) state estimate that may be used for motion tracking and navigation. Because of its small size and effective design, it may be used with drones, ground robots, and mobile devices. It computes roll, pitch, and metric scale and uses IMU data to be resistant to visual tracking loss. In order to minimize drift and improve accuracy, this system combines feature observations with pre-integrated IMU data in a VIO module, together with concurrent global pose optimization.

VINS-Mono's main shortcomings are its reliance on monocular vision without depth sensors, which can result in scale estimation errors, particularly when tracking features over long trajectories or in low-texture environments; the system still experiences residual drift in translation and orientation over time, even with the integration of an IMU; and loop closure techniques, which are crucial for long-term consistency, are not as reliable as approaches like ORB-SLAM3, which provide more developed solutions for relocalization and effectively reusing maps.

- *Direct Sparse Visual-Inertial Odometry - VI-DSO - 2018*

In order to estimate camera locations and sparse scene geometry concurrently, a novel technique for visual-inertial odometry known as VI-DSO [103] minimizes both photometric and IMU measurement errors in a combined energy functional. Unlike key-point based methods, the visual part of the system optimizes a sparse collection of points in a manner similar to bundle correction, while directly reducing a photometric mistake. This enables the system to monitor all pixels, not just corners, with a large enough intensity gradient. IMU data is collected across a

number of frames via measurement pre-integration, and it is utilized as an additional constraint in the optimization between keyframes.

Fig. 10 provides an overview of the VI-DSO method and outlines the main differences between it and the DSO methodology. The VI-DSO is an extension of the DSO algorithm that generates better accuracy and robustness than the original DSO and other algorithms, such ROVIO [70], by accounting for inertial information. However, because bundle modification is dependent on the starting process, it is slow [22]. The method does not conduct global optimization or loop closure detection, and no embedded implementations have been reported in the literature.

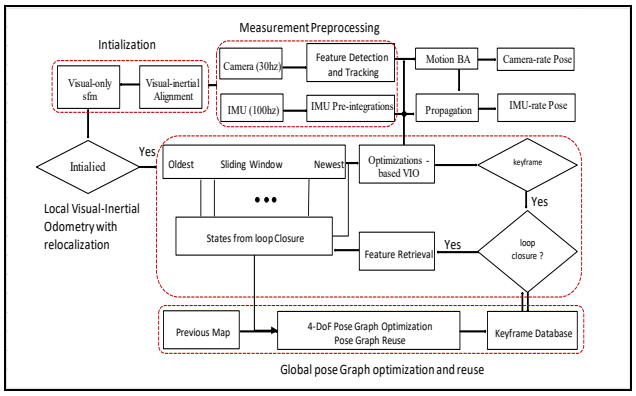

Fig. 9 Block diagram illustrating the full pipeline of a monocular VINS. Taken from [69]

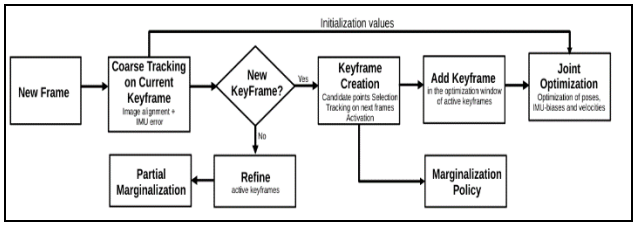

Fig. 10 VI-DSO algorithm representation diagram. Adapted from [15]

- *Delayed Marginalization Visual-Inertial Odometry - DM-VIO – 2020*

DM-VIO is a monocular visual-inertial odometry system that maximizes real-time processing by using delayed marginalization and posture graph bundle modification. DMVIO includes IMU data into marginalization states and allows rapid updates with dependable new linearization points by maintaining a secondary factor graph, which improves accuracy and reduces computing burden [102].
Sparse visual tracking and IMU data are combined with the effective motion estimation method DM-VIO to provide precise real-time applications such as AR and autonomous navigation. DM-VIO avoids feature matching by minimizing photometric error in high-gradient zones, while maintaining precision and lowering computing burden. Keyframes are optimized via a sliding window, which improves camera orientation, velocity, and posture. Nevertheless, delayed marginalization in DM-VIO enhances state estimation, but because of slower updates and difficulties with IMU initialization, it increases complexity and restricts scalability in highly dynamic applications [102].

- RD-VIO: Robust Visual-Inertial Odometry – 2021

RD-VIO, a visual-inertial odometry system developed by Jinyu Li et al. [98], uses the IMU-PARSAC algorithm to handle both pure rotational motions and dynamic surroundings. By breaking up rotating frames into subframes, this two-stage method eliminates problems with pure rotation and enhances keypoint matching with visual and IMU information. With improvements made to a baseline PVIO system to better manage landmark triangulation and modify postures in dynamic settings, Fig. 11 shows the pipeline for RD-VIO. Experiments on the EuRoc and ADVIO datasets show that RD-VIO performs better than baselines and works well on mobile devices, including an AR demo of the iPhone X. Pure inertial odometry or wireless tracking might be useful for maintaining performance, but, as it may lose tracking under extended difficult circumstances.

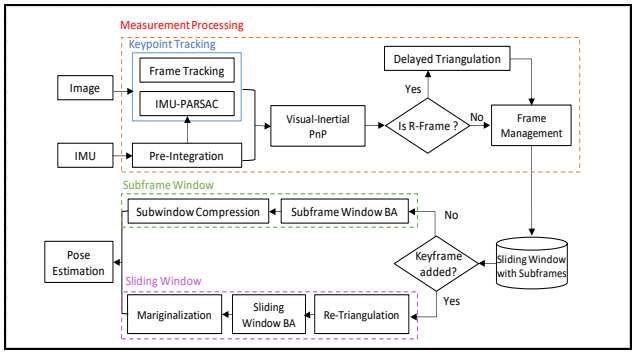

Fig. 11 The pipeline of RD-VIO. Adapted from [98]

- *Open Keyframe-based Visual-Inertial SLAM with Loop Closure OKVIS2 – 2022*

In robotics, augmented reality, and virtual reality (AR/VR), robust and accurate state estimation is still a hurdle, despite the increasing commodity nature of Visual-Inertial Simultaneous Localization and Mapping (VI-SLAM). a comprehensive VI-SLAM solution that addresses problems with long and repeated loop closures in particular. OKVIS2 is a real-time, multi-camera VI-SLAM system with loop closure and location identification capabilities. a multi-camera VI-SLAM system that uses visual reprojection errors, IMU preintegrated error terms, and the marginalization of common observations to generate a factor graph.

A real-time estimator minimizes these in a bounded-size window of recent pose-graph and keyframe frames. Once the loop is closed, it is easy to turn the old pose-graph edges back into landmarks and reprojection errors. Longer loops can also be optimized asynchronously while keeping all states around the loop as components of the optimized variables by reusing the same factor-graph [99]. The VI SLAM system is composed of a frontend and a realtime estimator that process images and IMU messages concurrently whenever a new (multi-)frame is received. To handle loop closures, an asynchronous entire factor graph loop optimization is performed. As shown in Figure 12, the frontend manages a number of tasks, including place recognition, segmentation CNN running, keypoint matching, state initialization, stereo triangulation (from

consecutive frames and from stereo images of the same multi-frame), and, if the latter was successful, re-localization and loop closure initialization. The real-time estimator is then responsible for fixating prior states and constructing pose graph edges by marginalizing old data, and it will optimize the relevant factor graph. Following loop closure, it begins optimizing the complete factor graph and proceeds to turn the edges of the pose graph back into observations. After this asynchronous operation is complete, it synchronizes with the realtime factor graph.

Despite being useful for real-time visual-inertial SLAM, OKVIS2 has issues with accuracy maintenance in dynamic situations and computational overhead brought on by synchronous processing needs. Furthermore, delays may be introduced by its reliance on asynchronous loop optimization, which might compromise localization consistency in practical situations.

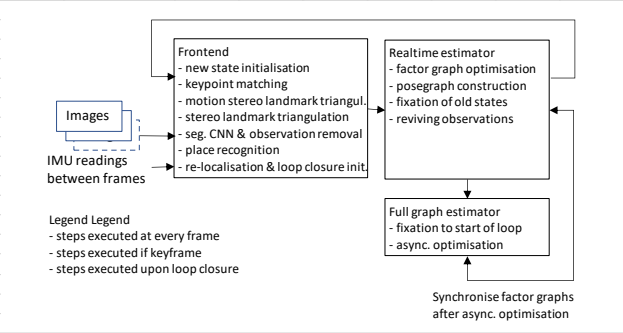

Fig. 12 Overview of OKVIS2 Adapted from [99]

In this part, every one of the seven main visual-inertial SLAM algorithms were independently examined. Table 2 provides an overview of the key characteristics and evaluation standards for the shown visual-inertial SLAM algorithms.

TABLE 2: KEY ELEMENTS OF THE VISUAL-INERTIAL SLAM TEQNIQUES. EVERY STRATGY SHOWS ACLOSLY INTEGRATED SENSORY FUSION.

Method	Type	Map Density	Global Optimization	Loop Closure	Availability
ROVIO	Filtering-based	Sparse	No	No	[106]
VIORB	Optimization-based	Sparse	Yes	Yes	–
VINS-MONO	Optimization-based	Sparse	Yes	Yes	[107]
VI-DSO	Optimization-based	Sparse	No	No	[108]
DM-VIO	Direct	Sparse	No	No	[109]
RD-VIO	Hybrid	Sparse	No	Yes	[110]
OKVIS2	Keyframe-based	Sparse	Yes	Yes	–
ORB-SLAM3	Feature-based	Sparse	Yes	Yes	[98]

R

C. GB-D SLAM

The innovative RGB-D technique integrates depth sensors and RGB-D cameras to estimate and build environmental models. This approach has found applications in several domains, including robotic perception and navigation. It performs well and provides useful information on the spatial surroundings, particularly in inside settings with good lighting. The system can concurrently record color and depth data since RGB-D cameras and depth sensors are coupled. Due to its ability to resolve dense reconstruction on low-textured surface regions, this capability is particularly useful for interior applications. The goal of RGB-D SLAM is to produce a precise three-dimensional reconstruction of the system's surrounding environment, with a focus on gathering geometric data to create a comprehensive three-dimensional model. A summary of the methods applied in this section is provided below:

- RGBDSLAMv2 – 2014

ORB-SLAM2, a complete SLAM solution for monocular, stereo, and RGB-D cameras, has capabilities including map reuse, loop closure, and relocalization. The system runs in a variety of settings in real-time on standard CPUs, from small hand-held interior sequences to drones flying in industrial areas and cars driving around a city. RGBDSLAMv2, one of the most used RGB-D based algorithms, is built on feature extraction [115]. It estimates posture using the ICP approach and estimates the transformation between the matched features using the RANSAC algorithm. To remove the accumulated error, the system then performs a global optimization and loop closure. This method also proposes to use an environment measurement model (EMM) to validate the transformations obtained between the frames. The method's real-time performance is hampered since it depends on SIFT features. RGBDSLAMv2 requires slow sensor movement to work well and has a high processing cost. In Fig. 13, the algorithm is displayed.

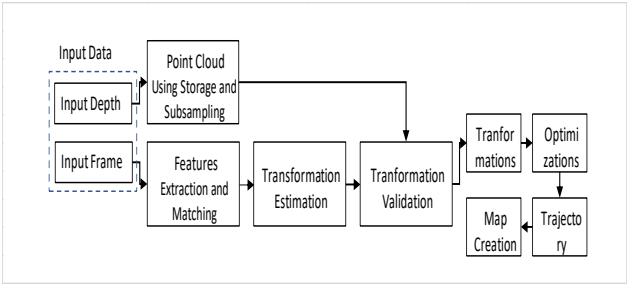

Fig. 13 A schematic illustration of the RGBDSLAMv2 algorithm. Taken from [15].

- Elastic Fusion – 2015

Using an incremental online method and an RGB-D camera, the Elastic Fusion system can capture rich, globally consistent surfel-based maps of room scale settings without the need for pose graph optimization or other postprocessing procedures. This is accomplished by using dense frame-to-model camera tracking, windowed surfel-based fusion, and frequent model refinement using non-rigid surface distortions. Using RGB-D sensors, ElasticFusion is a real-time dense visual SLAM technique that is intended for drift-free 3D reconstruction. By integrating photometric alignment with RGB data and frame-to-model tracking with the ICP method, it approximates the camera posture. To accomplish surface fusion, a truncated signed distance function (TSDF) is used to integrate color and depth data into a global model. One important aspect is the use of a non-rigid deformation graph, which modifies previously mapped regions to enable drift correction and real-time loop closure. The method employs optimization to constantly improve the global model, guaranteeing a reliable and

precise 3D reconstruction. Elastic Fusion suffers in settings with poor texture or fast movement, since tracking becomes erratic. Since memory usage rises sharply in large-scale settings, the method's reliance on a surfel-based approach further restricts its scalability. Furthermore, in crowded or changing landscapes, ElasticFusion's absence of an integrated method for addressing dynamic objects may cause drift or inaccuracy during tracking.

- ORB-SLAM2 - 2017

ORB-SLAM2 is a complete SLAM solution for monocular, stereo, and RGB-D cameras that includes loop closure, relocalization, and map reuse. In a variety of scenarios, such as cars driving through a city, small hand-held interior sequences, and drones flying in industrial environments, the system runs in real-time on standard CPUs. The suggestion by Strasdat et al. [116] states that ORB-SLAM2 uses depth information to generate a stereo coordinate for the recovered components of the picture. In this regard, whether the input is RGB-D or stereo is irrelevant to the system. Unlike all the previous methods, the back-end uses bundle adjustment to generate a globally consistent sparse reconstruction. Because of this, the ORB-SLAM2 method is easy to use and works with standard CPUs. Long-term and globally consistent localization is the goal, not the most complex dense reconstruction. Alternatively, one might fuse depth maps to produce correct reconstruction on-the-fly in a small region, or one could post-process the depth maps from each keyframe after a full BA to create a perfect 3D model of the whole scene utilizing the incredibly precise keyframe poses.

Fig. 14 shows the overall architecture of the system. The system runs three main parallel threads: Localizing the camera entails the following three steps: Three methods are used in tracking: 1) motion-only BA is used to find feature matches on the local map; 2) local mapping is used to manage and optimize the local map; and 3) loop closing is used to find large loops and correct accumulated drift by executing a pose-graph optimization. This thread initiates a fourth thread to finish full BA after the pose-graph optimization.

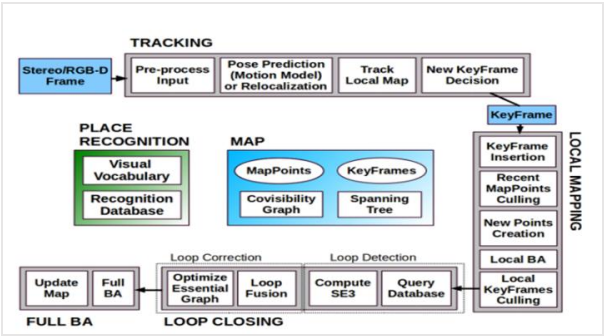

Fig. 14: Three primary parallel threads make up ORBSLAM2: loop closure , local mapping, and tracking. Adapted from [117].

- RTAB-Map – 2018

The RTAB-Map A visual SLAM technique that can be applied to RGB-D and stereo cameras is called real-time appearance-based mapping, or SLAM for short. It's a versatile technique that can handle 2D and 3D mapping tasks based on the sensor and available data. It allows the identification of both stationary and moving 3D objects in the robot's environment through the combination of RGB-D and stereo data for 3D mapping. When LiDAR rays are not able to control the field around the robot, it can be applied in large outdoor environments. Robotic localization and mapping errors can be caused by varying light and environmental interactions. Thus, RTAB's adaptability and resistance to changing light and scenery enable precision operation under challenging conditions. It can easily adjust to function with many cameras or laser rangefinders, and it can manage complex, large-scale scenarios. Moreover, the use of T265 (Intel RealSense Camera) and ultra-wideband (UWB) addresses robot wheel slippage with drifting error control, enhancing system efficiency through precise tracking and the generation of 3D point clouds. The RTAB-MAP SLAM approach involves several procedures in order for it to function. First, tasks including frame creation, sensor integration, and data extraction from RGB-D and stereo cameras are handled by the hardware and front-end stage. At this point, the frames required for the following phase are prepared. The loop closure provides the necessary odometry when the tracking operation and frame processing are finished simultaneously.

Fig. 15 depicts RTAB-Map, the main ROS node. Any kind of odometry may be used for SLAM as it is an external input to RTAB-Map; the choice will rely on what works best for the robot and the application. A graph consisting of nodes and links makes up the structure of the map. Once the sensors are synchronized, the Short-Term Memory (STM) module creates a node and memorizes the raw sensor data, the odometry posture, and additional information that will be needed for later modules (like visual words for Loop Closure and Proximity Detection and local occupancy grid for Global Map Assembling). The "Rtabmap/DetectionRate," which is given in milliseconds, is the determined rate at which nodes are constructed based on how much the data generated by them should overlap each other.

RTAB-Map high computational demand is a major disadvantage that can cause performance issues on devices with limited capabilities, particularly when working with big maps and loop closure detection. Additionally, drift or tracking failures may result from odometry's accuracy declining in settings with little feature variety. Because of its reliance on RGB-D sensors, its application is therefore limited to specific settings, such indoors, where depth data is more trustworthy.

- *Bundle Adjusted Direct RGB-D SLAM BAD-SLAM – 2019*

Simultaneous Localization and Mapping (SLAM) systems rely on the joint optimization of the camera trajectory and the expected 3D map. Bundle adjustment (BA) is the industry standard for this. rapid direct BA formulation, which they employ in a real-time, dense RGB-D SLAM system. As is common with SLAM algorithms, the technique consists of both front-end and back-end components (Figure 16). The frontend tracks the RGB-D camera's motions in real time. It provides first estimates for camera angles and scene dimensions as a result. At a lower frequency, the back-end then fine-tunes the geometry and

the camera trajectory to produce a consistent 3D map. A novel back-end Bundle Adjustment (BA) method for direct RGB-D SLAM is the primary technological contribution.

Sensitivity to sensor calibration and synchronization is a major limitation of BAD-SLAM. Performance can be severely harmed by problems like rolling shutter effects or mismatched depth and RGB data because the system mainly depends on accurate direct measurements from RGB-D cameras. Due to its reliance on extremely precise sensors, BAD-SLAM is less reliable in settings where these requirements are not satisfied or sensor configurations are not ideal.

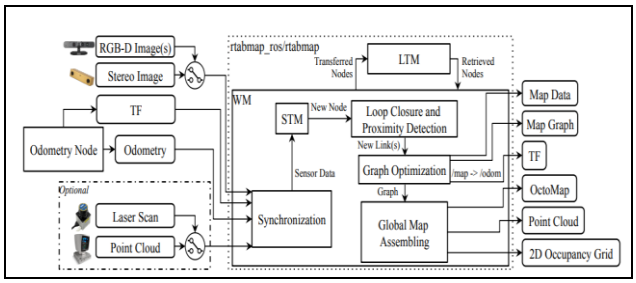

Fig. 15 RTAB-Map, the main ROS node.

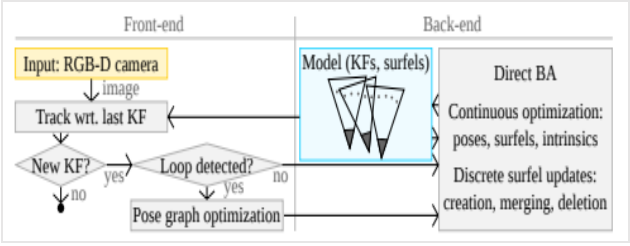

Fig. 16 BAD-SLAM overview. KF stands for keyframe.

- SCE-SLAM - 2023

Spatial coordinate errors SLAM (SCE-SLAM) is a new real-time semantic RGB-D SLAM technique. The purpose of its creation was to overcome the shortcomings of traditional SLAM systems in dynamic operational environments. Combining semantic and geometric data, together with using YOLOv7 for quick object recognition, improved the technique to outperform existing V-SLAM systems, including ORB-SLAM3, and to be more accurate and robust in dynamic circumstances. These improvements make it possible for the SLAM algorithms to be very effective in dynamic applications, which leads to greater adaptability and comprehension of the system environment. Robotic systems can therefore operate in more complex contexts with reduced slippage or mistakes. Robots equipped with SCE-SLAM may also operate more adaptably and with fewer errors, even under challenging lighting situations.

SCE-SLAM has the potential to severely impair the produced maps' accuracy and dependability. Misalignments and distortions in the spatial representation of the environment result from these mistakes, which are caused by inaccurate sensor readings and posture estimation. Reduced navigation performance as a result of SCE-SLAM can make it difficult for autonomous systems to function well in complex and dynamic surroundings.

Three main processes make up the SCE-SLAM method, according to Son et al. (2023). A semantic module is used in the initial stage. As the camera input data is handled, noise is removed using Yolov2 in this module. During the second step, the geometry module analyzes depth pictures and recovers spatial coordinates to prepare the system for integration with ORB SLAM3. The final stage is dedicated to ORB SLAM3 implementation. This link makes the ORB-SLAM3 procedures easier to execute. The procedure working in combination with the loop closure technique produces a system output that is more precise and accurate.

Section C provided a distinct description of the most typical RGB-D-based methods. Table 3 lists the key characteristics and parameters for critical assessment of the given algorithms.

TABLE 3. KEY FEATURES OF RGB-D BASED SLAM TECHNIQUES.

Method	Type	Map Density	Loop Closure	Availability
RGBDSLAMv2	Feature-based	Dense	No	[123]
Elastic Fusion	Direct	Dense	Yes	[124]
ORB-SLAM2	Feature-based	Dense	Yes	[92]
RTAB-Map	Hybrid	Dense/Sparse	Yes	[125]
BAD-SLAM	Direct	Dense	Yes	[126]
SCE-SLAM	Hybrid	Sparse	Yes	?
ORB-SLAM3	Feature-based	Sparse	Yes	[98]

IV. DATASETS AND BENCHMARKING

A fair comparison of all the SLAM algorithms in the literature is necessary to identify which one performs better in particular scenarios. The literature suggests a number of benchmarking datasets with various features to investigate the resilience and capabilities of SLAM. The benchmark dataset that was used to assess the SLAM algorithms that were described in the original publications is made publically available here.

- The **KITTI** dataset, which was created by the Toyota Technological Institute and the Karlsruhe Institute of Technology, includes eight LiDAR data sequences and twenty-two stereo camera sequences that were all taken from actual driving situations. It offers timestamps for synchronization, a range of view of around 60° horizontally for the stereo camera system, and ground truth data for vehicle trajectories and sensor calibration files [75].
- The Technical University of Munich's **TUM RGB-D** dataset offers 39 indoor RGB-D camera video sequences that document a range of scenarios with precise timestamp information and ground truth camera postures. A calibration file that describes the camera system's intrinsic and extrinsic characteristics is included in the dataset. The camera system's field of view is around 60° horizontally and 50° vertically [28]. Furthermore, relative pose error and absolute trajectory error are the two metrics that the authors suggest be used to assess the trajectory's local correctness and global consistency, respectively.

- Eleven sequences of high-resolution pictures and IMU data, together with ground truth trajectories, calibration files, and timestamps, were gathered using micro aerial vehicles and are included in the **Euroc MAV** collection. The camera's diagonal field of view is around 90° [77].
- The **TUM VI** collection includes 25 sequences of synchronized RGB and IMU data that span a 60° horizontal field of view and include comprehensive ground truth poses and calibration files. To guarantee exact temporal alignment, timestamps are supplied for every frame [78].
- The **TUM MONO VO** dataset provides 50 monocular video sequences with calibration information and ground truth postures. Each frame has a timestamp and a horizontal field of view of around 60° [79]. A strong framework for creating and evaluating motion and visual estimating algorithms in a variety of circumstances is offered by these datasets taken together.
- Specifically created for RGB-D SLAM, the **Bonn RGB-D dynamic** dataset comprises dynamic object sequences. Following the same style as TUM RGB-D datasets, it displays RGB-D data together with a 3D point cloud that depicts the changing environment. It goes beyond the confines of regulated spaces and encompasses both indoor and outdoor situations. When creating and assessing algorithms for tasks like object identification, scene comprehension, and robot navigation, it is useful. The fact that this dataset is adaptable enough to handle the complexity of applications utilized in light-challenging fields is noteworthy. Furthermore, it serves as a valuable tool for assessing V-SLAM methods in noisy and dynamic environments where the robot may encounter difficulties interacting with its surroundings and detecting objects.
- There are twelve artificial interior sequences available in the **ICL-NUIM** dataset, each having RGB-D pictures and simulation-generated ground truth trajectories. With a horizontal field of view of around 70°, it gives timestamps and calibration information for every frame [76]. The dataset, which focuses on RGB-D techniques, offers information for assessing the 3D reconstruction using eight artificially created interior settings. The ground truth is a 3D surface model and the calculated trajectory using a SLAM algorithm, while the sequences are generated by a handheld RGB-D camera [80].
- An autonomous driving dataset called **Cityscapes** [87] focuses on instance annotation and pixel-level picture segmentation. Additionally, other datasets—such as NYU RGB-D [37], MS COCO [38], and others—are employed in a variety of settings. With an emphasis on semantic comprehension of urban surroundings, the Cityscapes collection offers high-resolution urban street scenes gathered from 50 locations. With pixel-level labels for semantic segmentation tasks, it provides 20,000 coarsely labeled photos and 5,000 highly annotated images. 30 classes—human, car, flat surface, and other urban elements—are included in the collection. The photos were taken at a resolution of 2048x1024 pixels in a range of weather and lighting circumstances. The pictures have a broad field of vision, which is common for driving situations at street level.
- Another benchmarking dataset for assessing SLAM systems in difficult situations is **Tartan Air** [111]. A variety of weather patterns, moving objects, and changing light are all included in the incredibly lifelike simulated situations used to collect the data. By collecting data in simulations, we are able to give multi-modal sensor data and precise ground truth labels, such as segmentation, optical flow, camera positions, stereo RGB image, and LiDAR point cloud.
- Nguyen et al. [112] published the **NTU VIRAL** dataset, which was collected using an unmanned aerial vehicle (UAV) equipped with a 3DLiDAR, cameras, IMUs, and several Ultra-widebands (UWBs). The information is meant to be used for assessing the performance of aerial operations and autonomous driving. Both indoor and outdoor cases are included.

Table 4 summarizes the main benchmark datasets characteristics presented in this work.

TABLE 4 MAIN ASPECTS RELATED TO THE PRESENTED BENCHMARK DATASETS.

Dataset	Year	Environment.*	Platform	Sensor System	Groundtruth	Availability
TUM RGB-D	2012	Indoor	Robot/Handheld	RGB-D camera	Motion capture	[81]
KITTI	2013	Outdoor	Car	Stereo-cameras 3D laser scanner	INS/GPS	[82]
ICL-NUIM	2014	Indoor	Handheld	RGB-D camera	3D surface model SLAM estimation	[83]
Bonn RGB-D dynamic	2016	Indoor/Outdoor	Handheld	RGB-D camera	Motion capture (partially)	[90]
Cityscapes	2016	Outdoor	Car	Stereo	GPS	[87]
EuRoC	2016	Indoor	MAV	Stereo-cameras IMU	Total Station Motion capture	[84]
TUM Mono VO	2016	Indoor-Outdoor	Handheld	Non-stereo cameras	_	[85]
TUM VI	2018	Indoor-Outdoor	Handheld	Stereo-camera IMU	Motion capture (partially)	[86]
TartanAir	2020	Indoor-Outdoor	photo-realistic simulation environments	RGB cameras, depth sensors, IMU, LIDAR	GPS	[114]
NTU VIRAL	2021	Indoor-Outdoor	UAV	3D lidars, IMUs, time-synchronized cameras, UWBs	GPS	[113]

*Environment: indoor or outdoor.

V. GUIDELINES FOR EVALUATING AND SELECTING VISUAL SLAM METHODS

There are a few things to take into account while selecting a visual SLAM technique. Importantly, the type of sensor used is important: monocular SLAM is less expensive but has scale ambiguity, whereas stereo and RGB-D SLAM provide more accurate depth estimates at a greater computational cost [39]. Applications that need minimal latency must have real-time performance, and GPU acceleration helps techniques like Elastic Fusion [118]. The SLAM system must also be able to adjust to its surroundings; indoor-focused algorithms such as Kinect Fusion perform well in controlled settings but may not be as successful outside [119]. Long-term accuracy is ensured by the system's capacity to manage drift and implement loop

closure methods, as demonstrated by ElasticFusion and ORB-SLAM. Particularly for projects requiring in-depth 3D mapping [36], posture estimation accuracy and map precision should be taken into account. Certain methods are more appropriate for particular situations [120], but SLAM systems should be resilient to external obstacles like occlusions or moving objects. In conclusion, scalability, global optimization, and the accessibility of open-source implementations must be taken into account for long-term development and use [121,122].

VI. CONCLUSIONS

The difficulties and advancements in the field of visual-based SLAM (VSLAM) approaches are highlighted in this study's methodical investigation. Historically, multiple-view geometry and low-level feature matching have been the mainstays of VSLAM systems. There includes discussion of difficulties like recreating low-texture areas and the computational expenses of deep learning techniques, as well as the requirement for extra sensors (such IMUs or stereo cameras) or system previous knowledge. The following six criteria are suggested for choosing SLAM algorithms: availability, global optimization methods, map density, algorithm type, and embedded implementations. The research places a strong emphasis on assessing algorithms according to requirements unique to each application, including scalability, sensor compatibility, and environmental restrictions. The report also proposes future research directions and discusses benchmarking datasets for SLAM algorithm evaluation. An ideal SLAM system that balances real-time performance, precision, and robustness may be selected for a variety of applications by taking these criteria into account.

ACKNOWLEDGMENT

The authors are grateful for the financial support towards this research by the Computer Engineering Department, College of Engineering, Basrah University. Postgraduate Research Grant (PGRG) /2023/HIR/RAJA/ENG/39 (1751-7-3).

REFERENCES

[1] Smith, R.; Cheeseman, P. On the Representation and Estimation of Spatial Uncertainty. Int. J.Robot. Res. 1987, 5, 56–68.

[2] Bailey, T.; Durrant-Whyte, H. Simultaneous localization and mapping (SLAM): Part II. IEEE Robot. Autom. Mag. 2006, 13, 108–117.

[3] Dissanayake, M.W.M.G.; Newman, P.; Clark, S.; Durrant-Whyte, H.F.; Csorba, M.A. Solution to the simultaneous localization and map building (SLAM) problem. IEEE Trans. Robot. Autom. 2001, 17, 229–241.

[4] Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-time loop closure in 2D LIDAR SLAM. In Proceedings of the IEEE International Conference on Robotics and Automation, Stockholm, Sweden, 16–21 May 2016; pp. 1271–1278.

[5] Fuentes-Pacheco, J.; Ruiz-Ascencio, J.; Rendón-Mancha, J.M. Visual simultaneous localization and mapping: Asurvey. Artif. Intell. Rev. (2015). 43, 55–81.

[6] Ido, J.; Shimizu, Y.; Matsumoto, Y.; Ogasawara, T. Indoor Navigation for a Humanoid Robot Using a View Sequence. Int. J. Robot. Res. (2009). 28, 315–325.

[7] Celik, K.; Chung, S.J.; Clausman, M.; Somani, A.K. Monocular vision SLAM for indoor aerial vehicles. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (ICRA), St. Louis, MO, USA, 11–15 October 2009; pp. 1566–1573.

[8] Civera J., Lee S.H. (2019) RGB-D Odometry and SLAM. In: Rosin P., Lai YK., Shao L., Liu Y. (eds) RGB-D Image Analysis and Processing. Advances in Computer Vision and Pattern Recognition. Springer, Cham

[9] G. Grisetti, R. Kümmerle, C. Stachniss and W. Burgard, "A Tutorial on Graph-Based SLAM," in IEEE Intelligent Transportation Systems Magazine, vol. 2, no. 4, pp. 31-43, winter 2010, doi: 10.1109/MITS.2010.939925.

[10] Taketomi, T., Uchiyama, H. & Ikeda, S. Visual SLAM algorithms: a survey from 2010 to 2016. IPSJ T Comput Vis Appl 9, 16 (2017). https://doi.org/10.1186/s41074-017-0027-2

[11] Kabzan, J.; Valls, M.; Reijgwart, V.; Hendrikx, H.; Ehmke, C.; Prajapat, M.; Bühler, A.; Gosala, N.; Gupta, M.; Sivanesan, R.; et al. AMZDriverless: The Full Autonomous Racing System. J. Field Robot. (2020), 37, 1267–1294.

[12] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," in IEEE Robotics & Automation Magazine, vol. 13, no. 3, pp. 108-117, Sept. 2006, doi: 10.1109/MRA.2006.1678144.

[13] Taheri, H., & Xia, Z. C. (2021). SLAM; definition and evolution. Engineering Applications of Artificial Intelligence, 97, 104032. https://doi.org/10.1016/j.engappai.2020.104032

[14] Al-Tawil B, Hempel T, Abdelrahman A and Al-Hamadi A (2024), A review of visual SLAM for robotics: evolution, properties, and future applications. Front. Robot. AI 11:1347985. doi: 10.3389/frobt.2024.1347985

[15] Macario Barros, A.; Michel, M.; Moline, Y.; Corre, G.; Carrel, F. A Comprehensive Survey of Visual SLAM Algorithms. Robotics 2022, 11, 24. https://doi.org/10.3390/robotics11010024

[16] Chen C, Zhu H, Li M, You S. A Review of Visual-Inertial Simultaneous Localization and Mapping from Filtering-Based and Optimization-Based Perspectives. Robotics. 2018; 7(3):45. https://doi.org/10.3390/robotics7030045

[17] Zhang, S., Zheng, L., & Tao, W. (2021). Survey and Evaluation of RGB-D SLAM. IEEE Access, 9, 21367–21387. https://doi.org/10.1109/access.2021.3053188

[18] Munguia-Silva R, Mart'ınez-Carranza J. Autonomous flight using rgb-d slam with a monocular onboard camera only. In: 2018 international conference on electronics, communications and computers (CONIELECOMP). IEEE; 2018. p. 200–6.

[19] Li Y, Lang S. A stereo-based visual-inertial odometry for slam. In: 2019 Chinese automation congress (CAC). IEEE; 2019. p. 594–8.

[20] Wang S, Yue J, Dong Y, Shen R, Zhang X. Real-time omnidirectional visual slam with semi-dense mapping. In: 2018 IEEE intelligent vehicles symposium (IV). IEEE; 2018. p. 695–700.

[21] Jo H, Jo S, Cho HM, Kim E. Efficient 3d mapping with rgb-d camera based on distance dependent update. In: 2016 16th international conference on control, automation and systems (ICCAS). IEEE; 2016. 720 873–875.

[22] Covolan, J.P.; Sementille, A.; Sanches, S. A mapping of visual SLAM algorithms and their applications in augmented reality. In Proceedings of the 2020 22nd Symposium on Virtual and Augmented Reality (SVR), Porto de Galinhas, Brazil, 7–10 November 2020.

[23] Taketomi, T.; Uchiyama, H.; Ikeda, S. Visual SLAM algorithms: A survey from 2010 to 2016. IPSJ Trans. Comput. Vis. Appl. 2017, 9, 1–11.

[24] Servieres, M., Renaudin, V., Dupuis, A., & Antigny, N. (2021). Visual and Visual-Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking. Journal of Sensors, 2021, 1–26. https://doi.org/10.1155/2021/2054828

[25] Klette R, Koschan A, Schluns K (1998) Computer vision: three-dimensional data from images. 1st edn

[26] 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

[27] Campos, C.; Elvira, R.; Rodríguez, J.J.G.; M. Montiel, J.M.; D. Tardós, J. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM. IEEE Trans. Robot. 2021, 37, 1874–1890.

[28] Burri, M.; Nikolic, J.; Gohl, P.; Schneider, T.; Rehder, J.; Omari, S.; Achtelik, M.; Siegwart, R. The EuRoC micro aerial vehicledatasets. Int. J. Robot. Res. 2016, 35, 1157–1163.

[29] Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-Scale Direct Monocular SLAM. In Computer Vision–ECCV 2014; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 834–849.

[30] Bianco, S.; Ciocca, G.; Marelli, D. Evaluating the Performance of Structure from Motion Pipelines. J. Imaging 2018, 4, 98.

[31] Hempel, T., and Al-Hamadi, A. (2020). Pixel-wise motion segmentation for slam in dynamic environments. IEEE Access 8, 164521–164528. doi:10.1109/access.2020.3022506

[32] Liu, Y., and Miura, J. (2021). Rds-slam: real-time dynamic slam using semantic segmentation methods. Ieee Access 9, 23772–23785. doi:10.1109/access.2021.3050617

[33] Wang, Z., Pang, B., Song, Y., Yuan, X., Xu, Q., and Li, Y. (2023). Robust visual-inertial odometry based on a kalman filter and factor graph. IEEE Trans. Intelligent Transp. Syst. 24, 7048–7060. doi:10.1109/tits.2023.3258526

[34] Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-black wellized particle filters. IEEE Trans. Robotics 23, 34–46. doi:10.1109/tro.2006.889486

[35] Beghdadi, A., and Mallem, M. (2022). A comprehensive overview of dynamic visual slam and deep learning: concepts, methods and challenges. Mach. Vis. Appl. 33, 54. doi:10.1007/s00138-022-01306-w

[36] Mur-Artal, R., & Tardós, J. D. (2017). ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. IEEE Transactions on Robotics, 33(5), 1255-1262

[37] Dellaert, F., & Kaess, M. (2006). Fast, Incremental, Consistent Stereo SLAM. Proceedings of Robotics: Science and Systems

[38] Scaradozzi, D., Zingaretti, S., and Ferrari, A. (2018). Simultaneous localization and mapping (slam) robotics techniques: a possible application in surgery. Shanghai Chest 2, 5. doi:10.21037/shc.2018.01.01

[39] Mur-Artal, R., Montiel, J.M.M., & Tardós, J.D. (2015). ORB-SLAM: A Versatile and Accurate Monocular SLAM System. IEEE Transactions on Robotics, 31(5), 1147-1163. DOI: 10.1109/TRO.2015.2463671.

[40] Engel, J., Schöps, T., & Cremers, D. (2014). LSD-SLAM: Large-Scale Direct Monocular SLAM. European Conference on Computer Vision (ECCV), 834-849. DOI: 10.1007/978-3-319-10605-2_54.

[41] Strasdat, H., Montiel, J.M.M., & Davison, A.J. (2012). Visual SLAM: Why Filter? Image and Vision Computing, 30(2), 65-77. DOI: 10.1016/j.imavis.2012.02.002.

[42] Picard, Q., Chevobbe, S., Darouich, M., and Didier, J.-Y. (2023). A survey on real time 3d scene reconstruction with slam methods in embedded systems. arXiv preprint arXiv:2309.05349.

[43] Taheri, H., and Xia, Z. C. (2021). Slam; definition and evolution. Eng. Appl. Artif. Intell. 97, 104032. doi:10.1016/j.engappai.2020.104032.

[44] Fernández-Moral, E., Jiménez, J. G., and Arévalo, V. (2013). Creating metric topological maps for large-scale monocular slam. ICINCO (2), 39–47.

[45] Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Trans. Robotics 23, 34–46. doi:10.1109/tro.2006.889486.

[46] Li, Q., Wang, X., Wu, T., and Yang, H. (2022a). Point-line feature fusion based field real-time rgb-d slam. Comput. Graph. 107, 10–19. doi:10.1016/j.cag.2022.06.013.

[47] Chu, P. M., Sung, Y., and Cho, K. (2018). Generative adversarial network-based method for transforming single rgb image into 3d point cloud. IEEE Access 7, 1021–1029. doi:10.1109/access.2018.2886213

[48] Schneider, T., Dymczyk, M., Fehr, M., Egger, K., Lynen, S., Gilitschenski, I., et al. (2018). maplab: an open framework for research in visual-inertial mapping and localization. IEEE Robotics Automation Lett. 3, 1418–1425. doi:10.1109/lra.2018.2800113

[49] Chen, H., Yang, Z., Zhao, X., Weng, G., Wan, H., Luo, J., et al. (2020). Advanced mapping robot and high-resolution dataset. Robotics Aut. Syst. 131, 103559. doi:10.1016/j.robot.2020.103559.

[50] Acosta-Amaya, G. A., Cadavid-Jimenez, J. M., and Jimenez-Builes, J. A. (2023). Three-dimensional location and mapping analysis in mobile robotics based on visual slam methods. J. Robotics 2023, 1–15. doi:10.1155/2023/6630038

[51] Bustos, A. P., Chin, T.-J., Eriksson, A., and Reid, I. (2019). “Visual slam: why bundle adjust?,” in 2019 international conference on robotics and automation (ICRA) (IEEE), 2385–2391.

[52] Eudes, A., Lhuillier, M., Naudet-Collette, S., and Dhome, M. (2010). “Fast odometry integration in local bundle adjustment-based visual slam,” in 2010 20th International Conference on Pattern Recognition (IEEE), 290–293.

[53] Al-Tawil B, Hempel T, Abdelrahman A and Al-Hamadi A (2024), A review of visual SLAM for robotics: evolution, properties, and future applications. Front. Robot. AI 11:1347985. doi: 10.3389/frobt.2024.1347985

[54] Khoyani, A., and Amini, M. (2023). A survey on visual slam algorithms compatible for 3d space reconstruction and navigation, 01–06.

[55] Acosta-Amaya, G. A., Cadavid-Jimenez, J. M., and Jimenez-Builes, J. A. (2023). Three-dimensional location and mapping analysis in mobile robotics based on visual slam methods. J. Robotics 2023, 1–15. doi:10.1155/2023/6630038

[56] Duan, C., Junginger, S., Huang, J., Jin, K., and Thurow, K. (2019). Deep learning for visual slam in transportation robotics: a review. Transp. Saf. Environ. 1, 177–184. doi:10.1093/tse/tdz019

[57] Theodorou, C., Velisavljevic, V., Dyo, V., and Nonyelu, F. (2022). Visual slam algorithms and their application for ar, mapping, localization and wayfinding. Array 15, 100222. doi:10.1016/j.array.2022.100222

[58] Tourani, A., Bavle, H., Sanchez-Lopez, J. L., and Voos, H. (2022). Visual slam: what are the current trends and what to expect? Sensors 22, 9297. doi:10.3390/s22239297

[59] Mur-Artal, R.; Montiel, J.; Tardos, J. ORB-SLAM: A versatile and accurate monocular SLAM system. IEEE Trans. Robot. 2015, 31, 1147–1163.

[60] Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. IEEE Trans. Robot. 2017, 33, 1255–1262.

[61] Zhan, Z.; Jian, W.; Li, Y.; Yue, Y. A SLAM Map Restoration Algorithm Based on Submaps and an Undirected Connected Graph. IEEE Access 2021, 9, 12657–12674.

[62] Abouzahir, M.; Elouardi, A.; Latif, R.; Bouaziz, S.; Tajer, A. Embedding SLAM algorithms: Has it come of age? Robot. Auton. Syst. 2018, 100, 14–26.

[63] Yu,J.; Gao, F.; Cao, J.; Yu, C.; Zhang, Z.; Huang, Z.; Wang, Y.; Yang, H. CNN-based Monocular Decentralized SLAM on embedded FPGA. In Proceedings of the 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), NewOrleans, LA, USA, 18–22 May 2020; pp. 66–73.

[64] Tateno, K.; Tombari, F.; Laina, I.; Navab, N. CNN-SLAM: Real-Time Dense Monocular SLAM with Learned Depth Prediction. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6565–6574.

[65] J. Engel, V. Koltun and D. Cremers, "Direct Sparse Odometry," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 3, pp. 611-625, 1 March 2018, doi: 10.1109/TPAMI.2017.2658577.

[66] Jin, Q.; Liu, Y.; Man, Y.; Li, F. Visual SLAM with RGB-D Cameras. In Proceedings of the 2019 Chinese Control Conference (CCC), Guangzhou, China, 27–30 July 2019; pp. 4072–4077.

[67] Gao, X.; Wang, R.; Demmel, N.; Cremers, D. LDSO: Direct Sparse Odometry with Loop Closure. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.

[68] Boikos, K.; Bouganis, C.S. A high-performance system-on-chip architecture for direct tracking for SLAM. In Proceedings of the 2017 27th International Conference on Field Programmable Logic and Applications (FPL), Gent, Belgium, 4–6 September 2017; pp. 1–7.

[69] T. Qin, P. Li and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," in IEEE Transactions on Robotics, vol. 34, no. 4, pp. 1004-1020, Aug. 2018, doi: 10.1109/TRO.2018.2853729.

[70] D. Schlegel, M. Colosi and G. Grisetti, "ProSLAM: Graph SLAM from a Programmer's Perspective," 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 2018, pp. 3833-3840, doi: 10.1109/ICRA.2018.8461180.

[71] M. Sumikura, Y. Shibuya, and K. Sakurada, "OpenVSLAM: A Versatile Visual SLAM Framework," Proceedings of the 27th ACM International Conference on Multimedia, pp. 2292-2295, 2019. doi:10.1145/3343031.3350539.

[72] Seiskari, O.; Rantalankila, P.; Kannala, J.; Ylilammi, J.; Rahtu, E.; Solin, A. HybVIO: Pushing the Limits of Real-Time Visual-Inertial Odometry. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 4–8 January 2022; pp. 701–710.

[73] Merzlyakov, A.; Macenski, S. A Comparison of Modern General-Purpose Visual SLAM Approaches. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 9190–9197.

[74] Sturm, J., Engelhard, N., Endres, F., et al. (2012). A benchmark for RGB-D visual odometry, 3D reconstruction, and SLAM. *IEEE International Conference on Robotics and Automation (ICRA)*.

[75] Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI Vision Benchmark Suite. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[76] Huebner, M., & Bischof, H. (2013). ICL-NUIM: A dataset for evaluating visual odometry and SLAM algorithms in indoor environments. *Technical Report*.

[77] Burri, M., Nikolov, S., & Gohl, P. (2016). The Euroc MAV dataset. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[78] Schöps, T., & Cremers, D. (2017). TUM VI benchmark. *Technical Report, Technical University of Munich*.

[79] Sturm, J., & Burgard, W. (2011). TUM Mono VO dataset. *Technical Report, Technical University of Munich*.

[80] Whelan, T.; Kaess, M.; Johannsson, H.; Fallon, M.; Leonard, J.J.; McDonald, J. Real-time large-scale dense RGB-D SLAM with volumetric fusion. Int. J. Robot. Res. 2015, 34, 598–626.

[81] RGB-D SLAM Dataset and Benchmark. Available online: https://vision.in.tum.de/data/datasets/rgbd-dataset (accessed on 18 September 2024).

[82] KITTI-360. Available online: http://www.cvlibs.net/datasets/kitti/ (accessed on 18 September 2024).

[83] ICL-NUIM. Available online: https://www.doc.ic.ac.uk/~ahanda/VaFRIC/iclnuim.html (accessed on 18 September 2024).

[84] The EuRoC MAV Dataset. Available online: https://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets (accessed on 18 September 2024).

[85] Monocular Visual Odometry Dataset. Available online: http://vision.in.tum.de/mono-dataset (accessed on 18 September 2024).

[86] Visual-Inertial Dataset. Available online: https://vision.in.tum.de/data/datasets/visual-inertial-dataset (accessed on 18 September 2024)

[87] Cityscapes. Available online: https://www.cityscapes-dataset.com/ (accessed on 18 September 2024).

[88] NYU RGB-D. Available online: https://cs.nyu.edu/silberman/datasets/ (accessed on 18 September 2024).

[89] MS COCO. Available online: https://paperswithcode.com/dataset/coco (accessed on 18 September 2024).

[90] The Bonn RGB-D Dynamic Dataset is available at: https://www.ipb.uni-bonn.de/data/rgbd-dynamic-dataset (accessed on 18 September 2024).

[91] ORB-SLAM. Available online: https://github.com/raulmur/ORB_SLAM (accessed on 23 September 2024)

[92] ORB-SLAM2. Available online: https://github.com/raulmur/ORB_SLAM2 (accessed on 23 September 2024)

[93] CNNSLAM. Available online: https://github.com/iitmcvg/CNN_SLAM (accessed on 23 September 2024)

[94] DSO: Direct Sparse Odometry. Available online: https://github.com/JakobEngel/dso (accessed on 23 September 2024)

[95] LDSO: Direct Sparse Odometry with Loop Closure. Available online: https://github.com/tum-vision/LDSO (accessed on 23 September 2024)

[96] OpenVSLAM. Available online: https://github.com/xdspacelab/openvslam (accessed on 23 September 2024)

[97] ORB-SLAM3. Available online: https://github.com/UZ-SLAMLab/ORB_SLAM3 (accessed on 23 September 2024)

[98] Schneider, T., Schubert, T., Schmidt, H., & Zell, A. (2021). RD-VIO: Robust visual-inertial odometry for mobile augmented reality in dynamic environments. 2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 218–226. https://doi.org/10.1109/ISMAR52148.2021.00038

[99] Leutenegger, S., Furgale, P., Cadena, C., Dellaert, F., & Siegwart, R. (2022). OKVIS2: Realtime scalable visual-inertial SLAM with loop closure. arXiv preprint arXiv:2206.04135. https://arxiv.org/abs/2206.04135

[100] Picard, Q., Chevobbe, S., Darouich, M., and Didier, J.-Y. (2023). A survey on real time 3d scene reconstruction with slam methods in embedded systems. arXiv preprint arXiv:2309.05349.

[101] M. Bloesch, S. Omari, M. Hutter and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 2015, pp. 298-304, doi: 10.1109/IROS.2015.7353389.

[102] Geneva, P., Yang, Y., Eckenhoff, K., & Huang, G. (2020). DM-VIO: Delayed Marginalization Visual-Inertial Odometry. 2020 IEEE International Conference on Robotics and Automation (ICRA), 5796-5802. https://doi.org/10.1109/ICRA40945.2020.9197424

[103] Von Stumberg, L.; Usenko, V.; Cremers, D. Direct Sparse Visual-Inertial Odometry Using Dynamic Marginalization. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2510–2517.

[104] Mur-Artal, R.; Tardós, J.D. Visual-Inertial Monocular SLAM with Map Reuse. IEEE Robot. Autom. Lett. 2017, 2, 796–803.

[105] Zhao, J., & Shen, S. (2019). A review of visual-inertial simultaneous localization and mapping from filtering-based and optimization-based perspectives. Robotics and Automation Letters, 4(2), 361–368. https://doi.org/10.1109/LRA.2019.2891678

[106] ROVIO. Available online: https://github.com/ethz-asl/rovio (accessed on 2 October 2024).

[107] VINS-Mono. Available online: https://github.com/HKUST-Aerial-Robotics/VINS-Mono (accessed on 2 October 2024).

[108] VI-DSO. Available online: https://github.com/RonaldSun/VI-Stereo-DSO (accessed on 2 October 2024).

[109] DM-VIO. Available online: https://github.com/lukasvst/dm-vio (accessed on 2 October 2024).

[110] RD-VIO. Available online: https://github.com/Jianxff/rd_vio (accessed on 2 October 2024).

[111] Wang,W.;Zhu, D.; Wang, X.; Hu, Y.; Qiu, Y.; Wang, C.; Hu, Y.; Kapoor, A.; Scherer, S. Tartanair: A dataset to push the limits of visual slam. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 4909–4916.

[112] Nguyen,T.M.; Yuan, S.; Cao, M.; Lyu, Y.; Nguyen, T.H.; Xie, L. NTU VIRAL: A Visual-Inertial-Ranging-Lidar dataset, from an aerial vehicle viewpoint. Int. J. Robot. Res. 2021, 41, 270–280.

[113] The NTU VIRAL Dataset is available at: https://ntu-aris.github.io/ntu_viral_dataset (accessed on 4 October 2024).

[114] The TartanAir Dataset is available at: https://theairlab.org/tartanair-dataset (accessed on 4 October 2024).

[115] Endres, F.; Hess, J.; Sturm, J.; Cremers, D.; Burgard, W. 3-D Mapping With an RGB-D Camera. IEEE Trans. Robot. 2014, 30, 177–187.

[116] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual SLAM," in IEEE Int. Conf. Comput. Vision (ICCV), 2011, pp. 2352–2359

[117] Mur-Artal, R., & Tardos, J. D. (2017). ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. IEEE Transactions on Robotics, 33(5), 1255–1262. https://doi.org/10.1109/tro.2017.2705103

[118] Whelan, T., Salas-Moreno, R., Glocker, B., Davison, A. J., & Leutenegger, S. (2015). ElasticFusion: Real-time dense SLAM and light source estimation. The International Journal of Robotics Research, 35(14), 1697-1716. https://doi.org/10.1177/0278364915591230

[119] Izadi, S., Moore, M., Kim, D., et al. (2011). KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera. In Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (pp. 559-568). https://doi.org/10.1145/2047196.2047270

[120] Sturm, J., Engelhard, N., Endres, F., Burgard, W., & Cremers, D. (2012). A benchmark for the evaluation of RGB-D SLAM systems. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 573-580). https://doi.org/10.1109/IROS.2012.6385773

[121] Newcombe, R. A., Davison, A. J., & Reid, I. D. (2011). KinectFusion: Real-time dense surface mapping and tracking. In 2011 10th IEEE International Symposium on Mixed and Augmented Reality (pp. 127-136). https://doi.org/10.1109/ISMAR.2011.6092378

[122] Chen, J., Richard, T., & Jayaraman, P. (2020). A comparative study of dense SLAM systems for RGB-D sensors. Journal of Field Robotics, 37(4), 584-602. https://doi.org/10.1002/rob.21925

[123] rgbdslam. Available online: http://ros.org/wiki/rgbdslam (accessed on 4 October 2024).

[124] ElasticFusion. Available online: https://github.com/mp3guy/ElasticFusion (accessed on 4 October 2024).

[125] RTAPMap. Available online: https://introlab.github.io/rtabmap (accessed on 4 October 2024).

[126] BAD-SLAM. Available online: https://github.com/ETH3D/badslam (accessed on 4 October 2024).

[127] Tourani, A., Bavle, H., Sanchez-Lopez, J. L., and Voos, H. (2022). Visual slam: what are the current trends and what to expect? Sensors 22, 9297. doi:10.3390/s22239297

[128] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. IEEE Trans. robotics 31, 1147–1163. doi:10.1109/tro.2015.2463671

[129] Campos, C., Elvira, R., Rodríguez, J. J. G., Montiel, J. M., and Tardós, J. D. (2021). Orb-slam3: an accurate open-source library for visual, visual–inertial, and multimap slam. IEEE Trans. Robotics 37, 1874–1890. doi:10.1109/tro.2021.3075644

[130] Ragot, N., Khemmar, R., Pokala, A., Rossi, R., and Ertaud, J.-Y. (2019). "Benchmark of visual slam algorithms: orb-slam2 vs rtab-map," in 2019 Eighth International Conference on Emerging Security Technologies (EST) (IEEE), 1–6.

[131] Mur-A, J. D., and Tars, R. (2014). "Orb-slam: tracking and mapping recognizable," in Proceedings of the Workshop on Multi View Geometry in Robotics (MVIGRO)-RSS.

[132] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. IEEE Trans. robotics 31, 1147–1163. doi:10.1109/tro.2015.2463671

[133] Zang, Q., Zhang, K., Wang, L., and Wu, L. (2023). An adaptive orb-slam3 system for outdoor dynamic environments. Sensors 23, 1359. doi:10.3390/s23031359

[134] Ca, C., Elvira, R., Rodríguez, J. J. G., Montiel, J. M., and Tardós, J. D. (2021). Orb slam3: an accurate open-source library for visual, visual–inertial, and multimap slam. IEEE Trans. Robotics 37, 1874–1890. doi:10.1109/tro.2021.3075644

[135] Joo, S.-H., Manzoor, S., Rocha, Y. G., Bae, S.-H., Lee, K.-H., Kuc, T.-Y., et al. (2020). Autonomous navigation framework for intelligent robots based on a semantic environment modeling. Appl. Sci. 10, 3219. doi:10.3390/app10093219

[136] Al-Tawil B, Hempel T, Abdelrahman A and Al-Hamadi A (2024), A review of visual SLAM for robotics: evolution, properties, and future applications. Front. Robot. AI 11:1347985. doi: 10.3389/frobt.2024.1347985.

[137] Boikos, K.; Bouganis, C.S. Semi-dense SLAM on an FPGA SoC. In Proceedings of the 2016 26th International Conference on Field Programmable Logic and Applications (FPL), Lausanne, Switzerland, 29 August–2 September 2016; pp. 1–4.