automation

MDPI

Communication

Modelling Software Architecture for Visual Simultaneous Localization and Mapping

Bhavyansh Mishra [1], Robert Griffin [1,2] and Hakki Erhan Sevil [1,*]

[1] Intelligent Systems & Robotics, University of West Florida, Pensacola, FL 32514, USA; bm37@students.uwf.edu (B.M.); rgriffin@ihmc.org (R.G.)
[2] Institute for Human and Machine Cognition (IHMC), Pensacola, FL 32502, USA
* Correspondence: hsevil@uwf.edu

Abstract: Visual simultaneous localization and mapping (VSLAM) is an essential technique used in areas such as robotics and augmented reality for pose estimation and 3D mapping. Research on VSLAM using both monocular and stereo cameras has grown significantly over the last two decades. There is, therefore, a need for emphasis on a comprehensive review of the evolving architecture of such algorithms in the literature. Although VSLAM algorithm pipelines share similar mathematical backbones, their implementations are individualized and the ad hoc nature of the interfacing between different modules of VSLAM pipelines complicates code reuseability and maintenance. This paper presents a software model for core components of VSLAM implementations and interfaces that govern data flow between them while also attempting to preserve the elements that offer performance improvements over the evolution of VSLAM architectures. The framework presented in this paper employs principles from model-driven engineering (MDE), which are used extensively in the development of large and complicated software systems. The presented VSLAM framework will assist researchers in improving the performance of individual modules of VSLAM while not having to spend time on system integration of those modules into VSLAM pipelines.

Keywords: visual simultaneous localization and mapping (VSLAM); software architecture for VSLAM; components of VSLAM systems

Citation: Mishra, B.; Griffin, R.; Sevil, H.E. Modelling Software Architecture for Visual Simultaneous Localization and Mapping. *Automation* **2021**, 2, 48–61. https://doi.org/10.3390/automation2020003

Academic Editor: Felipe N. Martins

Received: 22 February 2021
Accepted: 31 March 2021
Published: 2 April 2021

1. Introduction

Visual simultaneous localization and mapping (VSLAM) is a technique used to estimate camera motion and to generate 3D maps of environments using vision-based sensors. VSLAM is an essential building block in several robotic, automotive, and augmented/mixed reality (AR/MR) applications [1]. In the presence of sufficient illumination and visually distinctive environments, VSLAM can be used to track the 6 degrees-of-freedom (DOF) pose of the camera as well as to generate globally consistent 3D maps and camera trajectories with minimal drift over large-scale motion [2]. Therefore, VSLAM can also be viewed as a combination of visual odometry and loop closure [3]. Most VSLAM implementations share common components such as filtering, feature detection, and image alignment in the front-end for generating sensor-agnostic representations of the input data. The back-end usually consists of keyframe management, optimization, and map improvement for estimating motion and structure with global consistency and minimal error [4]. There still remain open challenges in VSLAM implementations such as resource allocation, distributed mapping, and optimizing map storage [4]. The research on improving the state-of-the-art in VSLAM is still ongoing, and several different approaches and implementations of VSLAM are being continuously proposed [5–8]. The newly proposed VSLAM approaches, conforming to their experimental software nature, are also vulnerable to the same issues that arise commonly in complex software engineering systems. This necessitates developing a better understanding of the core components and interfaces within VSLAM systems captured by an overarching software architecture for VSLAM.

The ability to create a generalized software model from commonly occurring components in any robotic application can speed up the overall application development process by improving code reusability and by reducing the time required for system design, development, integration, and maintenance [9]. Therefore, a software architecture for VSLAM that not only satisfies the original performance requirements but also offers long-term maintainability, reliability, stability, and robustness could significantly assist researchers while also promoting a commonality of language and a flexibility in composition. A framework for VSLAM could be highly beneficial in the long-term if it were designed with the requirements for robust perception under consideration [4], as well.

This paper aims to present an overview of VSLAM algorithms and, from this overview, to propose a model of VSLAM using components and defining interfaces with specific message types for assisting in the design, development, and testing of individual modules of VSLAM. Such an approach offers three key advantages: (i) better transparency in troubleshooting the software system, (ii) the ability to independently modify individual components, (iii) improved efficiency in system integration, and (iv) a better conceptual model of the system in general. The survey on VSLAM algorithms from 2010 to 2016 by Taketomi et al. [3] and the review of the SLAM literature up to 2016 by Cadena et al. [4] explain the high-level framework employed by state-of-the-art SLAM algorithms up until 2016. However, their works more generally review SLAM and do not provide detailed discussion on vision-based SLAM modules. Fraundorfer and Scaramuzza [10] systematically explained visual odometry algorithms developed between 1980 and 2011 but did not cover techniques that employ loop closures for global optimization. To the best of our knowledge, the current literature only reviews VSLAM algorithms up until 2016. In this work, we include more recent advances while also introducing a standardized comparison. Most notably, we introduce a novel framework to address algorithm reusability and architecture modularity to assist in the development of new VSLAM approaches. Therefore, the main contributions of this paper include (i) an analysis of past trends in design and performance of VSLAM algorithms; (ii) a comprehensive review of open source libraries and packages used in their implementations; and (iii) a general overview of VSLAM algorithms developed between 2000 and 2019; and finally, leveraging these other contributions, (iv) a component-based model of VSLAM modules.

The rest of this paper is organized into the following sections: Section 2 summarizes the approaches that have been used to implement various VSLAM algorithms. Section 3 develops an accumulated list of modules commonly found in VSLAM implementations for both feature-based and direct approaches. Section 4 proposes a model for VSLAM based on its evolving architecture from the literature. Section 5 lists the tools, libraries, and packages used in VSLAM module implementations. Section 6 provides an overview of the performance of various implementations of VSLAM against standard benchmark datasets for VSLAM. Finally, Section 7 presents the conclusion.

2. Overview of Approaches in VSLAM

The majority of currently existing visual SLAM algorithms primarily consist of two stages: the *front-end*, which handles transforming sensor data into a representation in feature space and establishing constraints in robot motion and sensor measurements, and the *back-end*, which consumes the constraints generated by the front-end and performs an optimization to maximize the *maximum a posteriori* estimate of the unknown poses and landmarks. Representation of the back-end as a factor graph has been de facto standard for formulating the core graph-based structure of SLAM problems due to its intuitive and generalized nature.

2.1. Front-End Modules for VSLAM

Often, the data received from perception sensors is large, dense, and too complex to be represented directly in the core graph of constraints. Due to this reason, it is common to derive constraints from salient portions of the sensor data such as features and patches with

high image gradients. Depending on which approach is chosen, feature-based and direct approaches stem out. Specific implementation of the front-end also varies based on the type of sensor configuration used such as monocular, stereo, RGB-D, etc. or some combination of them. Most of the modules of VSLAM can be used with these sensor configurations interchangeably. The majority of VSLAM algorithms have been developed using feature-based [10] and direct [11,12] approaches. Hybrid approaches that combine the selected benefits from both direct and feature-based approaches have also been proposed and are described below.

2.1.1. Feature-Based

Feature-based VSLAM methods track keypoint correspondences over successive image frames and perform a joint optimization of camera pose and 3D world point locations using a framework known as bundle adjustment. These methods require rotation and scale-invariant feature detection, effective matching, and algorithms such as Random Sample Consensus (RANSAC) for robust outlier removal [10]. Similar to most VSLAM approaches, feature-based methods also require modules such as pose-graph optimization, local mapping, and global map optimization to achieve high performance overall [13,14].

Initialization: The initialization module refers to the initial local map and camera pose generation, and the first setup of the global world frame for the algorithm. The front-end usually performs the initialization by triangulating an initial set of keypoints using two-view reconstruction algorithms such as the five-point algorithm [15], eight-point algorithm [16], or objects with known geometry [14,17]. In feature-based monocular VSLAM, two-view initialization becomes delayed since at least two keyframes are needed for local mapping and tracking [5]. However, in stereo setups, a single stereo-pair can be used to initialize SLAM by directly estimating parameters of the essential matrix, which is a matrix that relates 2D image points corresponding to the same 3D landmark points in two different images [14,18]. Approaches such as model selection between homography (for planar scenes) and fundamental matrix (for general scenes) have also been proposed [5]. Although most approaches estimate local map and pose by estimating parameters of the homography or the essential matrix between two keyframes by matching point correspondences, recent line correspondence-based methods that assume a constant rotation model over three successive keyframes have been explored [19].

Feature detection: Feature detection is used to generate keypoints from corners (Harris [20], Shi-Tomasi [21], Moravec [22], Forstner [23], FAST [24], and KLT [21]), blobs (SIFT [25], SURF [26], CENSUR [27], and ORB [28]) or line segments [29–33] for correspondence matching. Robust visual odometry requires feature descriptors to exhibit localization accuracy, repeatability, computational efficiency, robustness, distinctiveness, and invariance [10]. Most feature detectors consider the local peaks and crests on the feature-response function, usually after a non-maxima suppression, to be the detected features in the image [20,25]. Recently, both point and line features were combined for better performance overall [8,19].

Feature matching/tracking: Feature matching and tracking are necessary to find corresponding feature points in successive images, known as correspondences [14]. The brute force approach to feature matching employs a comparison of the pairwise sum of squared distances or normalized cross correlation between the feature points detected in first and second images. However, due to the quadratic computational complexity of such an approach, other techniques have been explored utilizing data structures such as search trees, bag-of-words, and hash tables for fast lookup of neighboring feature points in both images [5,34]. In the presence of wide-baseline motion between the two images, corresponding feature points are searched at their expected positions in the second image using image motion and distortion models [21]. Such an approach predicts the expected location of the feature point by assuming a motion model between the two images. Epipolar match-

ing is also used to reduce the search space in the second image to only the points along the epipolar lines corresponding to the feature points in the first image [35].

Keypoint outlier removal: Often, corresponding feature points are mismatched, and due to a high number of such outliers in the set of correspondences, RANSAC is performed over a camera motion model [10]. Usually a perspective-n-point (PnP) or efficient PnP (EPnP) algorithm is used to generate a camera motion model upon which RANSAC [36] is performed. The 5-point, 6-point, 7-point, and 8-point algorithms are widely used PnP variations in the literature. However, varying degrees of accuracy have also been achieved with lower numbers of point correspondences. For line correspondences, Pumarola et al. [19] explored EPnPL [37] for model fitting.

2.1.2. Direct

Direct methods avoid feature extraction and are designed to process image data directly using numerical optimization techniques for minimizing photometric cost using most of the pixels in the image to obtain the initial estimates of the structure and motion [11]. Usually, variants of image alignment techniques based on the famous Lucas–Kanade algorithm [38] are used to iteratively estimate the parameters of camera motion and 3D structure of the scene. Modules such as pose-graph optimization, loop closure, and keyframe management are also used in direct methods to improve overall performance and robustness [6,11].

Image alignment: Dense tracking using image alignment has been performed primarily using variants of the optical flow algorithm proposed by Lucas and Kanade [38] to estimate warping parameters between consecutive images [39]. The camera motion and depth map are jointly optimized for a large number of pixels in the image with the cost function

$$\arg\min_{p} \sum_{x} \|I_1(x) - W(I_2(x), p)\|^2, \tag{1}$$

where $W(I, p)$ is the warp function applied to image I_1 with parameters p for transforming it close to the second image I_2. Baker et al. [39] provide a detailed overview of four different configurations of the Lucas–Kanade algorithm (forward additive, inverse additive, forward compositional, and inverse compositional) based on the direction of warping between the template and image as well as on whether the update rule is additive or compositional.

Optimizer: Dense tracking methods require minimization of the energy function consisting of photometric residuals from all pixels based on brightness constancy [12]. A warping model is selected with appropriate parametrization for optimization [11]. The parameters of this warping transform are optimized with update steps based on steepest-descent, Newton, Gauss–Newton, and Levenberg–Marquardt formulations [39]. The overall framework of direct methods can be modelled as a maximum a posteriori (MAP) estimation [4]. The image alignment-based dense and semi-dense camera pose tracking optimization is essentially a maximization over the negative log likelihood of the photometric error over all pixels [40]. The negative log of the prior term therefore becomes the regularization term in the energy that is optimized. In DTAM [7], a non-convex energy functional was proposed with a photometric error data term and a regularization term that uses a spatial smoothness prior in an inverse depth parameterization.

2.1.3. Hybrid Approaches

In semi-direct methods, camera motion is estimated using feature-based keypoints tracked using optical flow. The mapping is performed using direct approach on keyframes that are generated after large baseline motion of the camera from the last keyframe [40]. Semi-dense methods combine the computational efficiency of feature-based methods and

the ability to work in featureless conditions of direct methods to estimate motion and structure in real time [41].

2.1.4. Other Common Modules

The front-end in VSLAM is also responsible for the task of data association. In the short term, the front-end associates measurements to variables in consecutive camera frames. In the long term, the front-end is responsible for finding loop closures to generate looping constraints on the graph of keyframes for back-end optimization [4].

Keyframe management: Keyframes are specific frames captured by the camera when large disparities are observed. Keyframe-based techniques offer robustness in map generation due to better 3D landmark triangulation [42]. The frequency at which keyframes are chosen and discarded can be used to establish an adaptive usage of processing and memory resources used by the algorithm. The technique of bundle adjustment (BA) is usually performed on a subset (window) of keyframes for local mapping to reduce the computational complexity and to improve convergence of the optimization. Various strategies for insertion and culling of keyframes based on factors such as the number of co-visible points and disparity with the previous keyframe, and computational resource limits have been used in the literature [5,43].

The keyframe pose-point constraints are usually stored in the form of a covisibility graph with a threshold number of shared landmarks for efficient BA and pose-graph optimization [44,45].

Relocalization and loop detection: VSLAM systems fail on occlusions and fast camera movements when tracking cannot be performed. Techniques using place recognition and data structures such as co-visibility graphs and spanning tree of keyframes are used to detect previously observed landmarks and to either relocalize the camera or identify loop constraints on the pose graph of the system [46]. The performance of relocalization improves when place recognition techniques such as bag-of-words (DBoW2 [47]) are used [5].

Loop closure: Visual odometry systems lack global map consistency checks and are therefore prone to drift from the ground truth over time as motion estimation error accumulates [10]. This can be corrected by detecting constraints from instances of the system coming back to a previously observed location, also known as place recognition [48]. The visual vocabulary for both local and global image descriptors can be used to generate loop candidates [10]. Techniques using visual bag-of-words have been frequently used in the past to implement loop closure detection [49]. The detected loop candidates are then verified for geometric consistency using epipolar constraints against previously observed keypoints [50].

3. Back-End Modules of VSLAM

Feature-based, direct, and hybrid VSLAM approaches all share the overall framework since they take camera image frames as input and generate camera motion models and 3D models of the scene as the output. They, however, differ in some of the core modules associated with local mapping and the optimization framework. This section provides a description of the most commonly found internal modules in both feature-based and direct approaches. Figure 1 also provides a general illustration of the feature-based VSLAM pipelines specifically and loosely generalizes the state-of-the-art pipelines from the literature such as ORB-SLAM [5], PL-SLAM [8], and PTAM [14].

Visual odometry systems accumulate both rotational and translational drift errors over time due to uncertainties in camera poses and 3D landmark positions. This issue is resolved by optimizing the old map points and camera poses with newly found geometrical constraints such as loop detection and multiple overlapping keyframes [42]. Several techniques such as performing BA on keyframes after loop constraint detection, and alternating

between pose-graph optimization and BA [5] have been proposed that perform global bundle adjustment while also keeping the required computation within resource limits for real-time operation [3].

Bundle adjustment: Bundle adjustment is the estimation of camera positions T, camera orientations R, and 3D landmark positions X through a joint optimization on the reprojection error between the landmark projections onto the keyframes $\pi(X)$ and matched keypoints. The optimization on reprojection error is defined as follows:

$$\arg\min_{R_i,T_i,X_j} \sum_{i}^{n} \sum_{j}^{m} \|z_{ij} - \pi(X_j, R_i, T_i)\|^2, \tag{2}$$

where n is the number of images, m is the number of point projections in the ith image and j iterates over the observation of m landmark points visible in the ith image as z_{ij}.

The nonlinear nature of such an optimization problem requires the use of nonlinear optimization algorithms such as Levenberg–Marquardt and Gauss–Newton. Local BA is used to obtain local 3D landmark positions around the current keyframe or a window of keyframes [13]. Full or global BAs can be performed on all keyframes, however, with increased computational cost. For multiple types of features (points, edgelets, and lines), cost functions are custom designed based on the reprojection constraints [8].

Factor graph optimization: The camera poses and their relative transforms are usually represented as a probabilistic graph, where the camera poses and landmark positions are the nodes and the probabilistic constraints between them are the edges [51]. Visual odometry algorithms usually accumulate drift over time, which can be modelled as the propagation of error in both poses and transforms through the pose graph [10]. Given initial noisy estimates for the variables such as camera poses and landmark positions, a factor graph can be used to calculate the maximum a posteriori (MAP) inference to optimize the graph further for consistency and to remove drift. An error term generated from the expected camera pose is summed up to a cost function over all edge constraints,

$$\arg\min_{x_i} -log(P(x_0) \prod_{i} P(x_i|Z)), \tag{3}$$

which on expanding the logarithm expression and dropping the constant prior factor becomes a sum of exponents as follows:

$$\arg\min_{x_i} \sum_{i} P(x_i|z_i), \tag{4}$$

where an assumption of Gaussian distributions as $P(x_i|z_i) = exp(h(x_i) - z_i)^2_\Sigma$ with a nonlinear measurement function $h(.)$ yields

$$\arg\min_{x_i} \sum_{i} (h(x_i) - z_i)^2_\Sigma. \tag{5}$$

This shows that, under Gaussian assumption, MAP inference is equivalent to solving a nonlinear least squares. At this point, a nonlinear optimizer such as the Gauss–Newton, Levenberg–Marquardt, or Dogleg algorithms could be used to generate the MAP estimate for the posterior $P(X|Z)$ [51]. Such techniques have also been referred to as *graph-based SLAM*, *pose-graph SLAM*, and *smoothing* in the literature.

Loop constraints in a factor graph provide important information for extended consistency of the global map and for reduction of error in camera poses and their relative transforms. Several approaches for loop constraint detection have been proposed in the past using both local [52] and global [53,54] image feature descriptors.

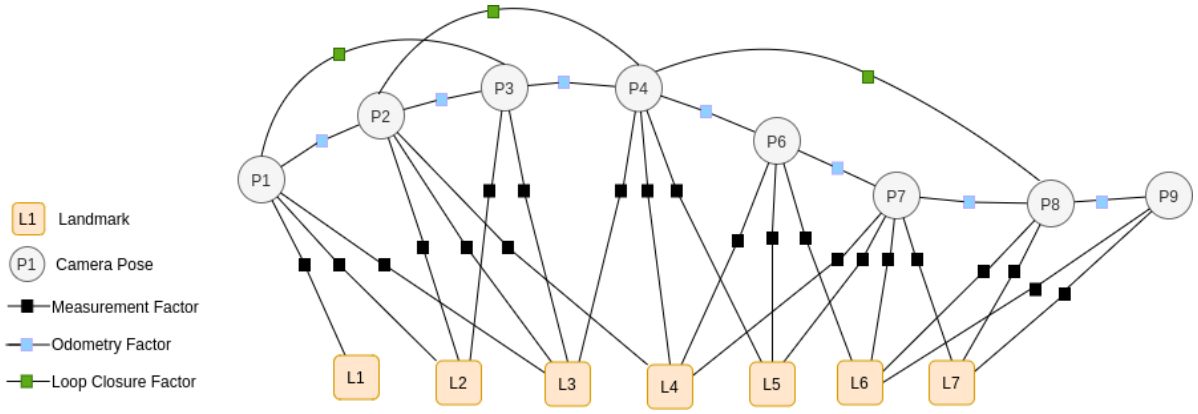

Figure 1. The factor graph is the central back-end structure for the state-of-the-art visual simultaneous localization and mapping (VSLAM) pipelines. This probabilistic graphical model stores probabilistic constraints of various kinds such as relative landmark positions and odometry measurements between camera poses. A factor graph is a bi-partite graph where the two types of nodes are variables (unknowns) and factors (constraints). Such a graphical model is used to perform maximum a posteriori inferencing to extract $P(X|Z)$.

4. Software Model for VSLAM Architecture

The principles of component-based software engineering help model a software system in terms of components, connectors, and rules to define the model topology using those components and connectors [9]. This section presents a novel architectural framework for VSLAM that not only has the potential to promote code reusability and maintainability but also can function to model existing state-of-the-art VSLAM systems. The following subsection defines VSLAM as consisting of both *active components* that perform computations on the processor and *passive components* such as data structures and databases that primarily store different kinds of information throughout the execution of the active components. We believe such a delineation of VSLAM modules into active and passive components as well as the interfaces between them can help make research on individual components of the VSLAM pipeline independent of the other components and, therefore, allow researchers to focus on improving particular components of VSLAM rather than on expending their efforts on system integration.

4.1. Architecture

The objects and their connections shown in Figure 2 illustrate the proposed architecture visually. This architecture consists of *active component* types and the interfaces between them as specified below:

1. *Feature detector*: **requires** an image frame and descriptor type and **provides** image coordinates of the detected feature points and feature descriptors;
2. *Feature matcher*: **requires** feature coordinates and descriptors from two images and **provides** feature point correspondences between two images;
3. *Local mapper*: **requires** keypoint correspondences and camera intrinsics, and **provides** local 3D map points;
4. *Local pose estimator*: **requires** feature point correspondences from two images and **provides** a $T \in SE(3)$ transform between camera image and last keyframe;
5. *Keyframe manager*: **requires** a new image frame and its feature points, and **provides** a keyframe decision and keyframe update;
6. *Loop detector*: **requires** a last keyframe and a new image frame and **provides** a loop closure constraint for a factor graph;
7. *Nonlinear optimizer*: **requires** a factor graph of measurement, motion, and loop constraints and **provides** an optimized graph based on maximum a posteriori inferencing; and
8. *Analyzer (for benchmarking purposes)*: **requires** the camera pose and 3D map points and **provides** accuracy measurements tested against benchmarking datasets.

The following *passive components* assist the *active components* in storing and retrieving different types of information as specified:

1. *Keyframe list*: **stores** the keyframes collected by the keyframe manager based on the keyframe generation and culling conditions and continuously optimized by the factor-graph nonlinear optimizer component
2. *Factor graph*: **stores** a graph of all the constraints from odometry; landmark measurements; loop closures; and other sensors such as Inertial Measurement Unit (IMU) factors, kinematics factors, etc.

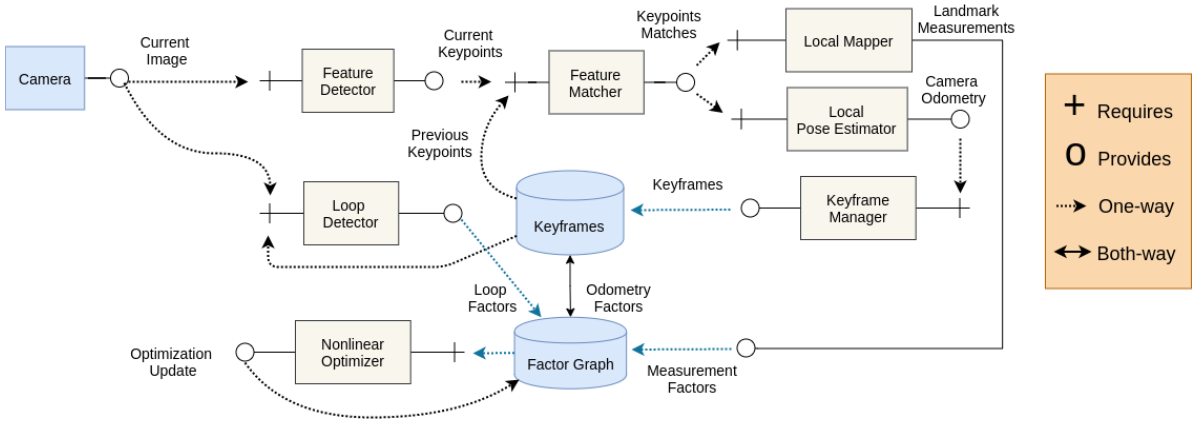

Figure 2. Architecture for VSLAM showing components (active and passive), interfaces, and data flow with appropriate message types.

4.2. Data Flow

This paper defines the data flow through the architecture as consisting of the messages that are exchanged at the interfaces between both the active and passive components. Selection of the appropriate set of message types can have a significant impact on the overall performance of the system. Based on the previous literature on VSLAM, we define a specific set of message types to be an appropriate selection for transfer of data across VSLAM components.

Keyframe messages: A keyframe message consists of the current image and the camera pose associated with the image. Keyframes are generated, stored, retrieved, and deleted by the keyframe manager component based on factors such as availability of wide-baseline images, limitations on computational resources, and detection of loop closures. The keyframe messages are stored in the keyframe database in the form of a covisibility graph and spanning tree for fast access using both the global map optimizer and the keyframe manager components. This definition gains inspiration from the survey by Younes et al. [42].

Keypoint messages: The keypoint message consists of detected feature point coordinates of a correspondence pairs between two images. The filtered keypoint messages are used by the keyframe manager to generate keyframes when a threshold number of keypoints are detected in consecutive image frames. The keypoint messages are associated with the 3D landmark measurements in the sensor frame.

Image messages: An image message consists of the normal RGB images generated by a monocular camera with specified height and width parameters. Raw images are only processed by the feature detector and loop detector components to create sensor agnostic data for subsequent components to use. Image messages are not passed or stored anywhere in raw format to conserve space and to maintain computational efficiency.

Map point messages: The map point messages consist of the 3D coordinates of the map points as generated by the local mapper. The map points are stored in the map database and occasionally optimized by the factor graph optimizer.

Camera pose messages: The camera pose messages consist of the 6 DoF position and orientation information needed to uniquely define the extrinsic parameters of the camera for a given time instant. The initial camera pose is generated by the initializer component and then later updated by the local mapper component.

4.3. Additional Modules

Resource monitor: The standard for robust perception as described by Cadena et al. in their review paper on visual odometry considers *resource awareness* as one of the important characteristics of any robust perception system. We therefore recommend an additional component in the VSLAM pipeline termed "resource monitor", which runs in the background and continuously monitors the CPU, GPU, memory, storage, and network usage metrics and dynamically changes the quantitative parameters of the VSLAM modules to vary the amount of processing time and space needed on-the-go accordingly.

Visualizer: To further enhance the productivity of researchers in VSLAM, we also recommend that particular emphasis be applied in the development of a graphical user interface, called the "visualizer" component, which extracts data from every interface between the VSLAM components and offers clear and reliable visualization of the different messages that flow through the VSLAM pipeline.

5. Algorithms, Tools, and Libraries

Since most of the VSLAM back-end problems can be formulated as optimization problems, software libraries that offer fast and accurate numerical optimization are indispensable to the current state-of-the-art in VSLAM implementations. In VSLAM pipelines, modules involving bundle adjustment, pose-graph optimization, and direct image alignment require efficient and real-time optimization over cost functions with thousands of parameters. This has only been achieved due to open-source implementations of software packages for large optimization problems including linear, nonlinear, constrained, and unconstrained ones.

Georgia Tech Smoothing and Mapping (GTSAM) is a smoothing and mapping (SAM) library for robotics and vision in C++ built using Bayesian networks and factor graphs. The General Graph Optimization (g2o) package offers a powerful back-end for solving optimization problems where constraints can be formulated as nodes and edges, such as in graph-based SLAM and bundle adjustment [51]. Ceres is a high-performance C++ library for solving large optimization problems such as nonlinear least squares with bounds and general unconstrained optimization [55]. The Incremental Smoothing and Mapping (iSAM and iSAM2) libraries are sparse nonlinear optimization libraries implemented in C++ that calculate incremental updates using efficient matrix factorization [56].

6. Benchmarking and Datasets

In this section, we discuss several datasets that have been used for evaluating visual SLAM systems [57]. The following subsections of this paper describe these datasets and compare the performance of VSLAM implementations that have been evaluated on each dataset by the respective authors of the VSLAM implementations.

KITTI Odometry: The KITTI Odometry dataset [57] consists of 22 stereo sequences taken from four Flea4 cameras, 3D laser scans taken from a Velodyne HDL-64E, and GPS-RTK readings taken on their self-driving platform over a length of 39.2 km in outdoor conditions. The sensor fusion and localization system based on OXTS RT3003 is used to define the ground truth for benchmarking. The KITTI dataset has been used for evaluation by ORB-SLAM [13], DSO [58], LDSO [59], GDVO [60], and Stereo LSD-VO [61], as shown in

Table 1. Since ORB-SLAM2 is the most complete state-of-the-art VSLAM pipeline in feature-based methods (includes almost all the modules of VSLAM as proposed in the model), it achieves the best performances on this dataset. Stereo DSO [62] and Stereo LSD [61] can be observed in Table 1 to be a close seconds in terms of relative and absolute translational errors, respectively.

Table 1. Absolute trajectory error (RMSE) on the KITTI Odometry Benchmark Dataset for state-of-the-art VSLAM implementations.

	ATE/RMSE			T_abs				T_rel	
(ATE/RMSE)	**ORB-SLAM**	**Mono DSO**	**LDSO**	**ORB-SLAM2**	**St. LSD**	**GDVO**	**Stereo DSO**	**ORB-SLAM2**	**St LSD-VO**
Seq 00	5.33	126.7	9.322	1.3	1	4.9	0.84	0.83	1.09
Seq 01	-	165.03	11.68	10.4	9	5.2	1.43	1.38	2.13
Seq 02	21.28	138.7	31.98	5.7	2.6	6.1	0.78	0.81	1.09
Seq 03	1.51	4.77	2.85	0.6	1.2	0.3	0.92	0.71	1.16
Seq 04	1.62	1.08	1.22	0.2	0.2	0.2	0.65	0.45	0.42
Seq 05	4.85	49.85	5.1	0.8	1.5	1.8	0.68	0.64	0.9
Seq 06	12.34	113.57	13.55	0.8	1.3	1.5	0.67	0.82	1.28
Seq 07	2.26	27.99	2.96	0.5	0.5	0.8	0.83	0.78	1.25
Seq 08	46.48	120.17	129.02	3.6	3.9	2.4	0.98	1.07	1.24
Seq 09	6.62	74.29	21.64	3.2	5.6	2.2	0.98	0.82	1.22
Seq 10	8.68	16.32	17.36	1	1.5	1.1	0.49	0.58	0.75

European Robotics Challenge (EuRoC): The EuRoC dataset [63] consists of 11 sequences captured from the on-board stereo camera and IMU of a quadrotor in varying motion and illumination conditions. The external motion capture system and 3D scans of the environment are provided as the ground truth for evaluation of the SLAM systems. The EuRoC dataset has been used in evaluating ORB-SLAM2 [13], LSD-SLAM [61], SVO [17], DSO [58], and PL-SLAM [8] as shown in Table 2. This is one of the most challenging datasets for any VSLAM pipeline as the motion consists of rapid translations and rotations, as observed from a micro-aerial vehicle. ORB-SLAM2 can again be observed to offer one of the best performances compared to other state-of-the-art VSLAM pipelines in Table 2.

Table 2. Translational error (RMSE) on the European Robotics Challenge (EuRoC) Micro-Aerial Vehicle (MAV) Dataset for state-of-the-art VSLAM implementations.

	Trans (RMSE)			T_abs			T_rel		
Seq.	**St ORB-SLAM2**	**St LSD-SLAM**	**St SVO**	**MonoVO ORB-SLAM**	**Mono DSO**	**MonoVO LSD**	**P-SLAM**	**L-SLAM**	**PL-SLAM**
V1_01	0.035	0.066	0.04	0.04	0.12	1.24	0.0583	0.0464	0.0423
V1_02	0.02	0.074	0.04	-	0.11	1.11	0.0608	-	0.0459
V1_03	0.048	0.089	0.07	-	0.93	-	0.1008	-	0.689
V2_01	0.037	-	0.05	0.02	0.04	-	0.0784	0.0974	0.0609
V2_02	0.035	-	0.09	0.07	0.13	-	0.0767	-	0.0565
V2_03	-	-	0.79	-	1.16	-	0.1511	-	0.1261
MH_01	0.035	-	0.04	0.03	0.05	0.18	0.0811	0.0588	0.0416
MH_02	0.018	-	0.05	0.02	0.05	0.56	0.1041	0.0566	0.0522
MH_03	0.028	-	0.06	0.02	0.18	2.69	0.0588	0.0371	0.0399
MH_04	0.119	-	0.17	0.2	0.24	2.13	-	0.109	0.0641
MH_05	0.06	-	0.12	0.19	0.11	0.85	0.1208	0.0811	0.0697

TUM RGB-D: The TUM RGB-D dataset [64] consists of color and depth sequences captured from a Microsoft Kinect in two different environments at 30 Hz. The ground truth is defined by the time-synchronized 6 DoF poses provided by an external motion capture system at 100 Hz. The TUM RGB-D dataset has been used for evaluation by PL-SLAM [8], ORB-SLAM [5], LSD-SLAM [6], Semidense-VO [41], and PL-SVO [65], as shown in Table 3. The PTAM [14] and ORB-SLAM [5] pipelines deliver the best performances on this dataset out of the solely vision-based SLAM pipelines given in Table 3.

Table 3. Absolute trajectory error on the TUM RGB-D dataset.

AKfT (RMSE)	LSD-SLAM	PL-SLAM	ORB-SLAM	PTAM
f1_xyz	9	1.46	1.38	1.15
f2_xyz	2.15	1.49	0.54	0.2
floor	38.07	9.42	8.71	-
kidnap	-	60.11	4.99	2.63
office	38.53	5.33	4.05	-
NstrTexFar	18.31	37.6	-	34.74
NstrTexNear	7.54	1.58	2.88	2.74
StrTexFar	7.95	1.25	0.98	0.93
StrTexNear	-	7.47	1.5451	1.04
deskPerson	31.73	6.34	5.95	-
sitHalfsph	7.73	9.03	0.08	0.83
WalkXyz	5.87	9.05	1.48	-
WalkXyz	12.44	-	1.64	-
WalkHalfsph	-	-	2.09	-

Other datasets: The ICL-NUIM dataset [66] consists of RGB-D sequences, ground truth camera poses, and 3D surface models for four different trajectories generated in a synthetic environment. This is the first dataset to provide a 3D surface ground truth for assessing reconstruction accuracy in SLAM systems. The ICL-NUIM dataset has been used for the evaluation of SVO, ORB-SLAM, DSO, and LSD-SLAM by Forster et al. [17] and of PL-SVO by Gomez-Ojeda et al. [65]. The New College dataset [67] consists of 30 GB of data collected as 5 DoF odometry, omnidirectional, and stereo camera sequences captured by a robotic vehicle driving through college campus. The New College dataset has been used for evaluation of RSLAM [68] and ORB-SLAM [5]. Other notable datasets used for evaluation of the VSLAM systems include RobotCar dataset [69], TrakMark [70], SLAMBench2 [71], and the dataset proposed by Martull et al. [72].

7. Conclusions

Numerous approaches to VSLAM including filter-based, feature-based, direct, semi-direct, and semi-dense methods have been developed and tested in the past. Exploration of VSLAM has fueled research in several different areas of computer vision such as feature detection, place recognition, numerical optimization methods, and visual geometry. The landscape of VSLAM pipelines continues to grow, with new approaches being continuously proposed. A standardized model of VSLAM pipelines is required now more than ever before. Appropriate delineation of module boundaries and definition of the interfaces between them can help improve conceptual understanding, can optimize performance, and can offer high code reuse by bringing consistency in implementation techniques. Several aspects for robust 3D perception using monocular and stereo visual cameras have not been addressed sufficiently enough for high reliability, efficiency, and performance. This paper reviews VSLAM implementations through the lens of a consistent model, compares their performances as presented in the literature, and provides a concise summary of open source libraries that have been used to develop VSLAM systems.

Author Contributions: Conceptualization, methodology, investigation, and writing—original draft preparation, B.M.; supervision, project administration, and writing—review and editing, R.G. and H.E.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pascoe, G.; Maddern, W.; Tanner, M.; Piniés, P.; Newman, P. NID-SLAM: Robust Monocular SLAM Using Normalised Information Distance. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1446–1455. [CrossRef]
2. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef] [PubMed]
3. Taketomi, T.; Uchiyama, H.; Ikeda, S. Visual SLAM algorithms: A survey from 2010 to 2016. *IPSJ Trans. Comput. Vis. Appl.* **2017**, *9*, 16. [CrossRef]
4. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [CrossRef]
5. Mur-Artal, R.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
6. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2014; pp. 834–849.
7. Newcombe, R.A.; Lovegrove, S.J.; Davison, A.J. DTAM: Dense tracking and mapping in real-time. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2320–2327. [CrossRef]
8. Gomez-Ojeda, R.; Moreno, F.; Zuñiga-Noël, D.; Scaramuzza, D.; Gonzalez-Jimenez, J. PL-SLAM: A Stereo SLAM System Through the Combination of Points and Line Segments. *IEEE Trans. Robot.* **2019**, *35*, 734–746. [CrossRef]
9. Brugali, D.; Scandurra, P. Component-based robotic engineering (Part I) [Tutorial]. *IEEE Robot. Autom. Mag.* **2009**, *16*, 84–96. [CrossRef]
10. Fraundorfer, F.; Scaramuzza, D. Visual odometry: Part ii: Matching, robustness, optimization, and applications. *IEEE Robot. Autom. Mag.* **2012**, *19*, 78–90. [CrossRef]
11. Cremers, D. Direct methods for 3D reconstruction and visual SLAM. In Proceedings of the 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA), Nagoya, Japan, 8–12 May 2017; pp. 34–38. [CrossRef]
12. Irani, M.; Anandan, P. About Direct Methods. In Proceedings of the International Workshop on Vision Algorithms: Theory and Practice, Corfu, Greece, 21–22 September 2000.
13. 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. [CrossRef]
14. Klein, G.; Murray, D. Parallel Tracking and Mapping for Small AR Workspaces. In Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan, 13–16 November 2007; pp. 225–234. [CrossRef]
15. Nister, D. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 756–770. [CrossRef] [PubMed]
16. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: New York, NY, USA, 2003.
17. Forster, C.; Zhang, Z.; Gassner, M.; Werlberger, M.; Scaramuzza, D. SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems. *IEEE Trans. Robot.* **2017**, *33*, 249–265. [CrossRef]
18. Herrera, C.D.; Kim, K.; Kannala, J.; Pulli, K.; Heikkilä, J. DT-SLAM: Deferred Triangulation for Robust SLAM. In Proceedings of the 2014 2nd International Conference on 3D Vision, Tokyo, Japan, 8–11 December 2014; Volume 1, pp. 609–616. [CrossRef]
19. Pumarola, A.; Vakhitov, A.; Agudo, A.; Sanfeliu, A.; Moreno-Noguer, F. PL-SLAM: Real-time monocular visual SLAM with points and lines. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 4503–4508. [CrossRef]
20. Harris, C.G.; Pike, J. 3D positional integration from image sequences. *Image Vis. Comput.* **1988**, *6*, 87–90. [CrossRef]
21. Jianbo, S. Good features to track. In Proceedings of the 1994 IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 21–23 June 1994; pp. 593–600. [CrossRef]
22. Moravec, H.P. *Obstacle Avoidance And Navigation in the Real World by a Seeing Robot Rover*; Technical Report; Stanford University: Stanford, CA, USA, 1980.
23. Fan, H.; Ling, H. Parallel Tracking and Verifying: A Framework for Real-Time and High Accuracy Visual Tracking. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 5487–5495. [CrossRef]
24. Rosten, E.; Drummond, T. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 430–443.
25. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
26. Bay, H.; Tuytelaars, T.; Van Gool, L. Surf: Speeded up robust features. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 404–417.
27. Agrawal, M.; Konolige, K.; Blas, M.R. Censure: Center surround extremas for realtime feature detection and matching. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 102–115.

28. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571. [CrossRef]
29. Mei, C.; Malis, E. Fast central catadioptric line extraction, estimation, tracking and structure from motion. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 4774–4779. [CrossRef]
30. Solà, J.; Vidal-Calleja, T.; Devy, M. Undelayed initialization of line segments in monocular SLAM. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; pp. 1553–1558. [CrossRef]
31. Hofer, M.; Maurer, M.; Bischof, H. Efficient 3D Scene Abstraction Using Line Segments. *Comput. Vis. Image Underst.* **2017**, *157*, 167–178. [CrossRef]
32. Zhang, L.; Koch, R. Hand-Held Monocular SLAM Based on Line Segments. In Proceedings of the 2011 Irish Machine Vision and Image Processing Conference, Dublin, Ireland, 7–9 September 2011; pp. 7–14. [CrossRef]
33. Briales, J.; Gonzalez-Jimenez, J. A Minimal Closed-form Solution for the Perspective Three Orthogonal Angles (P3oA) Problem: Application To Visual Odometry. *J. Math. Imaging Vis.* **2016**, *55*, 266–283. [CrossRef]
34. Özuysal, M.; Lepetit, V.; Fleuret, F.; Fua, P. Feature harvesting for tracking-by-detection. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 592–605.
35. Pritchett, P.; Zisserman, A. Wide baseline stereo matching. In Proceedings of the Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271), Bombay, India, 4–7 January 1998; pp. 754–760. [CrossRef]
36. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
37. Vakhitov, A.; Funke, J.; Moreno-Noguer, F. Accurate and linear time pose estimation from points and lines. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 583–599.
38. Lucas, B.D.; Kanade, T. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence-Volume 2*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1981; pp. 674–679.
39. Baker, S.; Matthews, I. Lucas-kanade 20 years on: A unifying framework. *Int. J. Comput. Vis.* **2004**, *56*, 221–255. [CrossRef]
40. Forster, C.; Pizzoli, M.; Scaramuzza, D. SVO: Fast semi-direct monocular visual odometry. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 15–22. [CrossRef]
41. Engel, J.; Sturm, J.; Cremers, D. Semi-dense Visual Odometry for a Monocular Camera. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 1449–1456. [CrossRef]
42. Younes, G.; Asmar, D.; Shammas, E.; Zelek, J. Keyframe-based Monocular SLAM. *Robot. Auton. Syst.* **2017**, *98*, 67–88. [CrossRef]
43. Strasdat, H.; Montiel, J.; Davison, A.J. Scale drift-aware large scale monocular SLAM. *Robot. Sci. Syst. VI* **2010**, *2*, 7.
44. Mei, C.; Sibley, G.; Newman, P. Closing loops without places. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 18–22 October 2010; pp. 3738–3744. [CrossRef]
45. Strasdat, H.; Davison, A.J.; Montiel, J.M.M.; Konolige, K. Double window optimisation for constant time visual SLAM. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2352–2359. [CrossRef]
46. Williams, B.; Klein, G.; Reid, I. Real-Time SLAM Relocalisation. In Proceedings of the 2007 IEEE 11th International Conference on Computer Vision, Rio de Janeiro, Brazil, 14–21 October 2007; pp. 1–8. [CrossRef]
47. Gálvez-López, D.; Tardos, J.D. Bags of binary words for fast place recognition in image sequences. *IEEE Trans. Robot.* **2012**, *28*, 1188–1197. [CrossRef]
48. Lowry, S.; Sünderhauf, N.; Newman, P.; Leonard, J.J.; Cox, D.; Corke, P.; Milford, M.J. Visual Place Recognition: A Survey. *IEEE Trans. Robot.* **2016**, *32*, 1–19. [CrossRef]
49. Nilsback, M.; Zisserman, A. A Visual Vocabulary for Flower Classification. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 17–22 June 2006; Volume 2, pp. 1447–1454. [CrossRef]
50. Angeli, A.; Filliat, D.; Doncieux, S.; Meyer, J. Fast and Incremental Method for Loop-Closure Detection Using Bags of Visual Words. *IEEE Trans. Robot.* **2008**, *24*, 1027–1037. [CrossRef]
51. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. g2o: A general framework for graph optimization. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 3607–3613.
52. Mikolajczyk, K.; Tuytelaars, T.; Schmid, C.; Zisserman, A.; Matas, J.; Schaffalitzky, F.; Kadir, T.; Van Gool, L. A comparison of affine region detectors. *Int. J. Comput. Vis.* **2005**, *65*, 43–72. [CrossRef]
53. Jogan, M.; Leonardis, A. Robust localization using panoramic view-based recognition. In Proceedings of the 15th International Conference on Pattern Recognition, ICPR-2000, Barcelona, Spain, 3–7 September 2000; Volume 4, pp. 136–139.
54. Ulrich, I.; Nourbakhsh, I. Appearance-based place recognition for topological localization. In Proceedings of the 2000 ICRA, Millennium Conference, IEEE International Conference on Robotics and Automation, Symposia Proceedings (Cat. No.00CH37065), San Francisco, CA, USA, 24–28 April 2000; Volume 2, pp. 1023–1029. [CrossRef]
55. Agarwal, S.; Mierle, K. Ceres Solver. Available online: http://ceres-solver.org (accessed on 21 March 2021).

56. Kaess, M.; Ranganathan, A.; Dellaert, F. iSAM: Incremental Smoothing and Mapping. *IEEE Trans. Robot.* **2008**, *24*, 1365–1378. [CrossRef]
57. Geiger, A.; Lenz, P.; Urtasun, R. Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; IEEE Computer Society: Washington, DC, USA, 2012; pp. 3354–3361.
58. Engel, J.; Koltun, V.; Cremers, D. Direct Sparse Odometry. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 611–625. [CrossRef] [PubMed]
59. 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; pp. 2198–2204. [CrossRef]
60. Zhu, J. Image Gradient-based Joint Direct Visual Odometry for Stereo Camera. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), Melbourne, Australia, 19–25 August 2017; pp. 4558–4564.
61. Engel, J.; Stückler, J.; Cremers, D. Large-scale direct SLAM with stereo cameras. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 1935–1942. [CrossRef]
62. Wang, R.; Schwörer, M.; Cremers, D. Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 3923–3931. [CrossRef]
63. Burri, M.; Nikolic, J.; Gohl, P.; Schneider, T.; Rehder, J.; Omari, S.; Achtelik, M.W.; Siegwart, R. The EuRoC micro aerial vehicle datasets. *Int. J. Robot. Res.* **2016**, *35*, 1157–1163. [CrossRef]
64. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A benchmark for the evaluation of RGB-D SLAM systems. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 573–580. [CrossRef]
65. Gomez-Ojeda, R.; Briales, J.; Gonzalez-Jimenez, J. PL-SVO: Semi-direct Monocular Visual Odometry by combining points and line segments. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 4211–4216. [CrossRef]
66. Handa, A.; Whelan, T.; McDonald, J.; Davison, A.J. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 1524–1531. [CrossRef]
67. Smith, M.; Baldwin, I.; Churchill, W.; Paul, R.; Newman, P. The new college vision and laser data set. *Int. J. Robot. Res.* **2009**, *28*, 595–599. [CrossRef]
68. Mei, C.; Sibley, G.; Cummins, M.; Newman, P.; Reid, I. RSLAM: A system for large-scale mapping in constant-time using stereo. *Int. J. Comput. Vis.* **2011**, *94*, 198–214. [CrossRef]
69. Maddern, W.; Pascoe, G.; Linegar, C.; Newman, P. 1 Year, 1000km: The Oxford RobotCar Dataset. *Int. J. Robot. Res.* **2017**, *36*, 3–15. [CrossRef]
70. Tamura, H.; Kato, H. Proposal of international voluntary activities on establishing benchmark test schemes for AR/MR geometric registration and tracking methods. In Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality, Orlando, FL, USA, 19–22 October 2009; pp. 233–236.
71. Bodin, B.; Wagstaff, H.; Saeedi, S.; Nardi, L.; Vespa, E.; Mayer, J.; Nisbet, A.; Luján, M.; Furber, S.; Davison, A.; et al. SLAMBench2: Multi-Objective Head-to-Head Benchmarking for Visual SLAM. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018.
72. Martull, S.; Peris, M.; Fukui, K. Realistic CG stereo image dataset with ground truth disparity maps. In Proceedings of the ICPR Workshop TrakMark2012, Tsukuba, Japan, 11–15 November 2012; Volume 111, pp. 117–118.