Mapping the Landscape of SLAM Research: A Review

Jeremías Gaia, Eugenio Orosco, Francisco Rossomando, Carlos Soria

Abstract—**Multiple publications have arisen from over three decades of research in the field of simultaneous localization and mapping (SLAM), overwhelming those who wish to delve into this area. The extensive body of work in SLAM has resulted in an abundant and, at times, confusing flow of data, lacking a straightforward explanation of the underlying principles. This article aims to address this issue by providing a clear overview of the general taxonomy of the SLAM universe, assuming the reader is completely unfamiliar with the subject area. As cameras remain the primary sensor choice for SLAM, and considering the vast number of articles available on this topic, special emphasis will be placed on Visual SLAM. Additionally, we will delve into the influence of artificial intelligence on the development of new algorithms. The article incorporates comparative tables to enable readers to assess system performance using benchmark datasets. Moreover, it offers insights into current trends and prospective pathways within the subject area.**

Index Terms—**Simultaneous Localization and Mapping, SLAM, Survey, Visual SLAM, Artificial Intelligence, Intoduction**

I. Introduction

Robots must always be able to determine their location. For mobile robots, whether they move autonomously or with the assistance of an operator, the integration of SLAM systems is imperative [1].

Since the foundations were established in 1986 by authors like Cheeseman and Durrant-Whyte, SLAM has grown exponentially [2], [3]. During the initial two decades (approximately 1986-2006), the primary approaches to solving the problem were rooted in probabilistic filters. Kalman filters, extended Kalman filters, and particle filters emerged as state-of-the-art solutions [4], [5].

With the arrival of passive low-cost video sensors to provide robotic systems with rich visual input, researchers began to integrate cameras into SLAM systems [6]. With the help of non-linear minimization techniques, this data was used to solve the pose of the mobile robot.

Modern SLAM systems have evolved into intricate combinations of subsystems, leveraging diverse sensor data, handling vast amounts of information, and incorporating artificial intelligence (AI) methodologies. In this paper, we present a comprehensive systematic mapping study of the existing literature on SLAM, aiming to provide readers with a comprehensive introduction to the realm of SLAM.

This study makes the following contributions:

All authors are with the Instituto de Automática, at Facultad de Ingeniería, Universidad Nacional de San Juan, Argentina. e-mail:{jgaia,eorosco,frosoma,csoria}@inaut.unsj.edu.ar

Manuscript received April 19, 2025; revised August 26, 2025.

- It provides a global perspective of the SLAM world, showing how it is structured after more than thirty years of research. Several classifications based on the most common elements of SLAM systems have been proposed to this end.
- The key elements in the current SLAM systems architectures are investigated.
- The relevance of artificial intelligence as a tool in current SLAM developments is discussed.
- It provides comparison tables for the reader to evaluate and compare SLAM systems and datasets objectively using

This article is structured as follows: Section II details the guidelines for conducting a systematic search and organizing the information within this document. Section III proposes an overview of the SLAM universe, while Section IV delves further into the specifics of visual SLAM. The impact of artificial intelligence on current systems is discussed in Section V. Comparative analyses of various SLAM methodologies are provided in Section VI. Conclusive insights are presented in Section VIII. Finally, Appendix A contains a summary of the symbols and acronyms employed throughout this article.

II. Search Guidelines

Kitchenham and Charters proposed that Systematic Mapping Studies (also known as Scoping Studies) are designed to provide a wide overview of a research area, to establish if research evidence exists on a topic and provide an indication of the quantity of the evidence [7]. They also designed a number of steps to follow in order to reach a good research result that include: defining the research questions and inclusion/exclusion rules, among others.

Following Kitchenham and Charters idea, this study was conducted based on the following research questions: Is it possible to organize the overwhelming amount of articles regarding the SLAM problem?, Are there any common elements between them?, Is it possible to objectively compare SLAM systems?. Most of the recent systems include cameras: How do Visual SLAM systems work?, Which are their main algorithms?. Finally, since artificial intelligence is a hot topic, it is worth asking: How are AI techniques influencing SLAM systems?.

A preliminary search stage was first conducted to identify existing reviews and assess the volume of relevant studies. As a result, several SLAM surveys and articles were found, and the classification criteria for the contributions in the SLAM field used in this study were established.

Most of the identified survey studies on SLAM have focused solely on specific aspects of the problem. The authors have extensively discussed theoretical issues [8]–[10], multi robot SLAM [11], LiDAR and camera-based SLAM [12], autonomous driving [13], dynamic SLAM [14], and Deep Learning for SLAM [15]. However, these studies often assume that the reader possesses prior knowledge in the field. This highlights the crucial need for an article that comprehensively elucidates the underlying structure of the SLAM universe, assuming the reader is completely unfamiliar with the subject area.

After the preliminary search stage, the following inclusion/exclusion rules were defined:

- Only articles written in Spanish or English were considered.
- Probabilistic SLAM approaches were not included since these kind of systems have already been extensively studied and reviewed [2], [3].
- The search stage was carried out with Google Scholar's search engine. Conference proceedings, journal papers, and unpublished but relevant articles were included.

As the reader may note, there are minimal exclusion constraints, as our intention is to encompass a wide range rather than a specific focus.

Given the significant number of papers presenting results on the EuRoC, KITTI, and 7-Scenes datasets, we focused our comparative analysis on the performance demonstrated by systems on these datasets. The comparison tables were constructed exclusively based on objective (numeric) results, while qualitative and graphical results were deliberately omitted.

III. The SLAM Universe

The purpose of this section is to offer an extensive insight into the domain of SLAM, aiming to establish a foundational understanding of the field. Fig. 1 presents a visual summary of this section. The existing SLAM developments can be organised based on their most common elements or aspects, such as the type of sensors used, the source of robot commands, and how they handle dynamics.

A. According to the Source of the Robot Commands.

A simple and general method to classify SLAM systems is to determine whether a robot or vehicle is commanded by a person. It allows us to distinguish between two types of SLAM systems: active and passive.

*1) **Active SLAM**:* This type of system is capable of traversing and mapping the environment automatically. Human intervention is not required.

They aim to maximize the map information obtained while minimizing a cost function variable. Map uncertainty, power consumption, traveled distance, etc. can be used as minimization variables. An active SLAM (A-SLAM) system has a distinctive workflow that includes: Candidate goal selection; path generation; cost function (also known as utility) computation; and path execution.

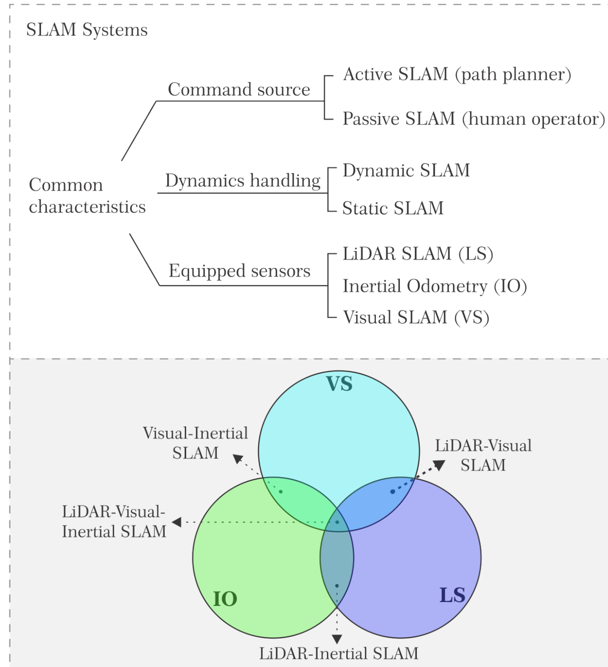

Fig. 1. **Top:** Overview of the SLAM Universe. The most common characteristics identified among state-of-the-art systems are used to classify them. **Bottom:** A detail of the classification of SLAM systems based on the sensors used and their combinations.

Brian Yamauchi established the foundations for selecting candidate goals based on the *nearest frontier* approach [16]. Frontiers refer to the regions along the boundary between explored and unknown territories, essentially representing points on the map situated between known free space and unexplored regions [1]. Frontier-based techniques construct a map while navigating toward a frontier. Whenever it reaches a limit, the system searches for the next frontier. This process is repeated until there are no new frontiers left to detect [17]–[19].

In addition to the nearest frontier approach, other exploration strategies include *cost-utility* [20], *coordinated* [21], *market based* [22] and *integrated* [23]. For a detailed discussion of these exploration approaches, we recommend the survey paper by Juliá et al. [24].

Path generation (also known as "path planning" or "motion planning") and cost function computation are closely linked, since the system's control actions are determined by the variables to be optimized. In this way, A-SLAM can be formulated as an optimal trajectory planning problem [25]–[28]. In this formulation, the control actions calculated by the path generation algorithm are restricted by the optimal trajectory problem.

Other A-SLAM systems concentrate on coverage of a larger area. Bonetto et al. proposed a system that continuously selects its camera heading to maximize environment coverage [29], whereas Carrillo et al. devised a utility function to achieve a balance between exploration and exploitation in [30]. As an additional feature to the traditional A-SLAM workflow, Chen et al. included a collision-free navigation constraint in their system [31].

*2) **Passive SLAM**:* The goal of these systems is to get a precise reconstruction of the robot's path. They are primarily concerned with lowering pose estimation errors while maintaining a consistent map representation.

Passive SLAM systems have an entirely different architecture than A-SLAM systems. According to Cadena et al. [32], a typical SLAM algorithm is comprised of two primary blocks: *front-end* and *back-end*. The *front-end* abstracts sensor data into models that are amenable for estimation, while the *back-end* performs inference on abstracted data produced by the *front-end*.

However, since the publication of ORB-SLAM [33], the architecture presented therein has been replicated in the majority of later advances [34]–[39]. As a result, we propose in this study to refine the *front-end/back-end* model described by [32]. The proposal shown in Fig. 2 aims to fit the current systems architecture.

Three main blocks can be distinguished: *tracking*, *mapping*, and *loop closure detection*.

- **Tracking:** Within this block, all essential tasks and computations are performed to establish the robot's pose, comprising its position and orientation. It receives the system's inputs and perform image rectification and/or feature extraction, synchronisation (multi-sensor systems only), filtering, and other necessary data preparation tasks.
  The tracking block must select representative input measurements to estimate the robot's pose in order to limit the amount of data. For those systems equipped with a camera, these measurements are referred to as *keyframes* [35], [40], [41].
- **Mapping**: The primary goal of this block is to create, develop, and refine a map of the environment. A map is a simplified representation of how the robot perceives the world. The type of sensors used, the image features extracted, and the pose estimation technique have a significant impact on which map representation must be selected [42]. Occupancy grid maps [43], [44], landmark maps [45], and dense representations [46], [47] are examples of mapping techniques.
  The mapping process can be classified into two parts: *local* and *global* mapping [35], [48], [49]. Maintaining a set (also known as *window*) of recent or co-visible views allows the system to perform operations that improve the map's consistency. This local window is updated with every new input to the system. The window (*local* map) is then handled to an optimization technique such as Bundle Adjustment in order to correct the robot's poses [50]. The *global* map is updated every time the loop closure detection algorithm discovers a previously seen location. For a complete description on mapping frameworks, we refer the reader to [51].
- **Loop Closure Detection** (LCD): This module is responsible for establishing long-term data associations. Recognizing previously visited locations aids the system in reducing the total uncertainty caused by odometry drift [52]. This is a decision-making block since it must ensure the system that the robot is in a previously visited location. Bad data associations cause the map to include ambiguous information, which can lead to several localization errors.
  The LCD task is highly dependent on how the system represents its surroundings. One of the most popular methods is to describe the scene as a Bag of Words (BoW) [53], [54]. A vocabulary of visual "words" is built offline in the BoW technique. The system then saves each input image as a collection (bag) of words.
  To detect loop closures, the BoW representation of two images or sequences is compared. The system considers a possible loop closure if the match percentage surpasses a threshold [52], [55]–[61]. A validation step is required to verify if the candidate is in fact a loop closure.

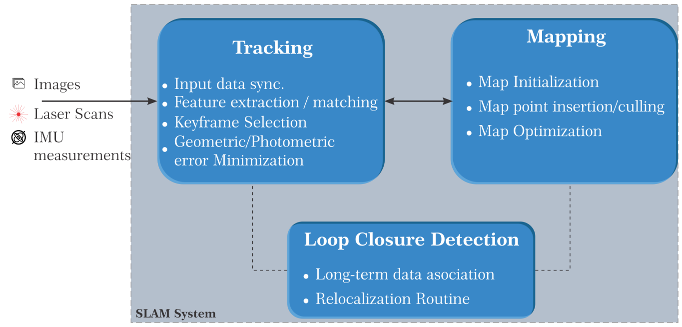

Fig. 2. Architecture of a SLAM system. All the tasks that the system needs to perform SLAM can be grouped into three building blocks: *tracking*, *mapping* and *loop closure detection*. This three blocks can operate sequentially or in parallel.

B. According to the Type of the Sensors Equipped.

LiDAR, visual, and inertial sensors are the three basic instruments used in SLAM systems. The use of these sensors and their combinations allows to classify SLAM systems as shown in Fig. 1.

*1) **LiDAR SLAM**:* Systems included in this category perceive the world through laser scans. Laser rangefinders can generate 2D or 3D point clouds with distance and bearing information for each point. Hector SLAM represents the traditional approach for laser-based SLAM. This system can create 2D occupancy grid maps with low computation resources. However, its main drawback is that it does not detect loop closures [43].

LiTAMIN (LiDAR-based Tracking And MappINg) is a three-dimensional LiDAR SLAM system. The authors proposed a change to the Iterative Closest Point (ICP) algorithm, which is used to optimize pose graphs [62]. A second version of the above mentioned work, combines a modified version of the ICP algorithm with a new point reduction strategy to reduce the computational complexity of the system [63]. Behley *et al.* [64] also used 3D scans in their research. They concentrated their efforts, however, on generating globally consistent maps. They also proposed a novel map-based criterion for LCD in LiDAR SLAM.

Similar to the Visual SLAM workflow, some LiDAR based SLAM systems perform feature extraction on range measurements [65]–[67]. Point features in laser scans describe geometrical characteristics of the scene, such as sharp edges and planar surfaces. A pose estimation is obtained by conducting a feature matching process between the current and previous scans.

Neural networks enable efficient processing of large amounts of information generated by laser scanners. Valente et al. proposed to combine Long Short-Term Memory (LSTM) neural networks and Convolutional Neural Networks (CNNs) to estimate the pose of a vehicle given the laser information [68]. On the other hand, Sun et al. proposed to combine a global localization system powered by a deep neural network, with the iterative Monte Carlo Localization algorithm [69].

*2) **Visual SLAM**:* This category encompasses all advances that use cameras as the primary sensor. This section provides a short introduction to Visual SLAM (V-SLAM), specifically focusing on the concept of visual odometry. Pertinent aspects within this domain are also outlined. An in-depth study about this particular area will be presented in Sec. IV.

V-SLAM techniques are based on visual odometry (VO). It can be defined as the method for estimating an agent's egomotion (human, robot, vehicle, etc.) based only on the input of a single or several cameras [9]. The most popular method for estimating camera poses is a combination of feature matching and RANSAC (Random Sample Consensus) [10], [33], [70]–[72]. The RANSAC algorithm refines the pose estimation after the feature matching procedure delivers an initial estimate of the camera motion.

PTAM (Parallel Tracking And Mapping) can be considered as one of the earliest and most influential Visual SLAM algorithms [73]. The authors proposed splitting tracking and mapping into two threads. Under the assumption that the mapping thread has already created a map of 3D points, visual odometry techniques were used to estimate the relative pose of the camera in the tracking thread. Inspired by PTAM, Mur-Artal et al. presented ORB-SLAM (Oriented FAST and Rotated BRIEF SLAM), where a third thread for *loop closure* was added to the tracking and mapping threads [33].

Leveraging concepts from PTAM, Mur-Artal and colleagues introduced ORB-SLAM (Oriented FAST and Rotated BRIEF SLAM). They expanded the tracking and mapping threads by integrating a third thread dedicated to *loop closure*, as seen in Fig 2. Their contribution became a benchmark, widely embraced by the research community.

Another innovative VO approach involves planar fiducial markers. This minimum modification to the environment can be used to facilitate SLAM in laboratory research. This is the case with recent work that involves placing fiducial planar markers with individual and unique IDs across the path of the robot [74]–[77]. Detection algorithms can quickly process these markers to obtain the robot's pose. Also, whenever one marker is detected for the second time, the system closes the loop.

Open-source state-of-the-art SLAM systems are hard to find. However, Schegel et al. released ProSLAM, an open-source Visual SLAM system aimed towards teaching [37]. This system was built in a modular format to make it simple to comprehend and implement. OpenV-SLAM (Open Visual SLAM) is another open-source visual SLAM framework [78]. A key advantage of OpenV-SLAM is its extensive support for various camera models, including fisheye, equirectangular, and perspective, enhancing its applicability

OpenSLAM is a platform for SLAM researchers which gives them the possibility to share their algorithms with the community. Information about the authors and the original papers is also provided. Since 2018, OpenSLAM has a Github page (https://github.com/OpenSLAM-org) where the source code and installation instructions for several SLAM systems are available.

*3) **Inertial Odometry**:* Incorporating inertial odometry (IO) techniques into SLAM systems has a series of benefits. Reading variations in accelerations, for example, helps LiDAR and Visual frameworks in compensating for camera movement caused by uneven terrain. In the literature, inertial odometry systems are frequently referred to as "Inertial Navigation Systems" (INS).

Strapdown inertial navigation systems (SINS) and pedestrian dead reckoning (PDR) are the two primary types of INS [79], [80]. The former calculates velocity by integration of the total acceleration and computes position by integration of the resultant velocity [81]–[85], whereas the latter estimates trajectories by detecting steps, estimating step length and heading, and updating locations per step [86]–[88].

Inertial navigation can also benefit from deep learning techniques [83], [84]. Chen et al. proposed IoNet, a neural network for inertial odometry drift reduction [89]. AbolDeepIO uses an LSTM network and an inertial measurement unit (IMU) to get a pose estimation from raw inertial data [90]. The aforementioned above, neural networks are used to replace traditional IO approaches. There are, however, systems that employ neural networks to improve standard methods [85], [87], [88].

*4) **LiDAR-Visual SLAM**:* The complementary strengths of cameras and laser scanners are used in LiDAR-visual SLAM approaches. Depth prediction is one of the reasons to use laser rangefinders with cameras [91]–[93]. Depth estimation improves the system in recovering the map's scale in monocular V-SLAM [65].

V-LOAM (Visual-LiDAR Odometry and Mapping) illustrates this relationship, as it uses depth information from laser rangefinders to enhance monocular [94]. Recently, Labbé et al. have proposed to extend their original work RTAB-MAP (Real-Time Appearance-Based MAPping) to support visual and LiDAR SLAM. This new version was published as an open source library by the authors [55], [95].

Systems using an RGB-D camera may also be deemed to fall under this group [96]–[98]. However, unlike the LiDAR-Camera combination, RGB-D cameras are light-sensitive and require supplementary sensors to produce better results [99].

Finally, using LiDAR-Visual systems to detect and classify objects has facilitated the development of autonomous vehicles [100]–[103]. However, when it comes to autonomous driving, calibration is crucial, since a precise 6-DoF transform between the sensors is necessary to accurately estimate the vehicle's pose [104], [105].

*5) **Visual-inertial SLAM (VI-SLAM)**:* Integration of IMU measurements can improve tracking and mapping performance of visual systems [106]–[108] .

Visual-inertial fusion can be accomplished in two ways: loosely coupled and tightly coupled approaches. Loosely coupled techniques use a vision algorithm to estimate the pose, which is then combined with IMU readings in a separate estimation step [109]–[111]. Tightly coupled systems, on the other hand, estimate camera poses using a combined energy cost function that minimizes photometric and IMU measurement errors [112], [113].

The use of inertial sensors in monocular V-SLAM techniques facilitates in the recovery of the map metric scale [34]. VINS-MONO (MONOcular Visual-INertial System) is a tightly coupled VI-SLAM system that leverages IMU measurements to reduce the localization inaccuracy provided by the camera system [114]. As an extension for VINS-MONO, the authors proposed in 2019 VINS-Fusion [115]. The algorithm treats camera and IMU measurements as variables in a factor graph. This optimization method provides a general framework for evaluating multiple camera combinations (IMU+mono, IMU+stereo).

Maplab, an open-source VI-SLAM system, provides users with a console for experimenting with various loop closure detection techniques, multi-session map merging, and map optimization tools [116]. The core component of the system is ROVIOLI (ROVIO with Localization Integration), a tracking/mapping block inspired in the ROVIO system [111].

Analogously to LiDAR-visual systems, initialization plays a pivotal role in visual-inertial systems. In order to achieve reliable measurements, the IMU model parameters must be as well defined as possible. Campos et al. suggested a rigorous process for camera and IMU initialization based on the preintegration concept of Forster et al. [109]. The covariance of the preintegrated components is estimated using IMU measurements and a sensor noise model, reducing the parameter uncertainty [117].

*6) **LiDAR-inertial SLAM**:* LiDAR scans can also benefit from inertial measurements. LIO-SAM (LiDAR Inertial Odometry via Smoothing And Mapping) formulates LiDAR-inertial odometry as a factor graph optimization problem [118]. The system calculates motion from IMU pre-integration and uses it as a starting point for LiDAR odometry optimization. LIPS SLAM (LiDAR-Inertial 3D Plane SLAM) on the other hand, strongly integrates inertial and planar primitive observations for state estimation. When facing dynamic scenarios, the authors used inertial data to better restrict the low frequency LiDAR sensor [119].

Neuhaus and Koß suggested an alternative for LiDAR-Inertial Fusion [120]. They built their system on top of a LiDAR SLAM system, to which they added IMU motion compensation measures. Similarly, Opromolla et al. proposed a hybrid solution that combines attitude data from an IMU with position estimation from a 2D LiDAR sensor for tracking [121].

C. According to How They Handle Dynamic Scene Content.

*1) **Static SLAM Approaches**:* Many SLAM systems are engineered with the assumption of a static environment, where the elements within a scene remain consistent over time. This entails that when a robot revisits a specific location, it should encounter the same set of features.

In real-world circumstances, this assumption is difficult to maintain. Moving objects, people and vehicles present a difficulty for static SLAM techniques. Feature-based approaches, for example, could extract features from pedestrians crossing in front of the camera (see Sec. IV-B). These features will then become a part of the map if the SLAM system does not reject them. This makes it more difficult to use the map for tracking and relocalization.

Some methods can deal with a small number of dynamic features by treating them as outliers [38], [39], [98]. RANSAC is the most commonly used outlier rejection method. This approach fits a model to experimental data. Values that are too far away from the fitting line are discarded.

In feature-based approaches, the RANSAC algorithm is used after the feature matching procedure. The algorithm will correctly fit common features from different images. Those that do not have a strong correspondence will be considered outliers and will be dismissed.

*2) **Dynamic SLAM Approaches**:* Addressing the presence of moving elements within a scene can be achieved by directly eliminating non-static content. In order to obtain a static image, Bescós et al. suggested a method for detecting and removing dynamic objects [72], [122], [123]. The algorithm employs a Mask R-CNN (Mask Region-based CNN) [124] to pixel-wise segment dynamic objects in the input frames. Dynamic pixels are deleted from the image after segmentation. The authors also added an inpainting method to fill in the gaps left by the removed pixels, resulting in a static image. This makes it very difficult for the system to extract features from dynamic objects. Sun et al., on the other hand, proposed to pre-process input data with a motion removal approach, which filters out information on moving objects [97].

Other strategies aim to identify dynamic information in an image without deleting it. For instance, DM-SLAM (SLAM Method for Rigid Dynamic Scenes), employs the segmented output from Mask R-CNN to prevent the tracking system from extracting features from mobile objects [125]. DS-SLAM (Semantic Visual SLAM towards Dynamic Environments) operates similarly to DM-SLAM but relies on a distinct neural network architecture [126]. In contrast, DMS-SLAM (SLAM system for Dynamic Scenes with Multiple sensors) proposes the combination of an ultra-robust feature correspondence algorithm with ORB-SLAM2 [127]. The resulting system provides a general platform that supports monocular, stereo, and RGB-D visual sensors in dynamic scenes. A recent advancement in this domain is STDyn-SLAM, a sophisticated dynamic SLAM system introduced by Esparza et al. [128]. The authors suggest an integrated strategy, merging SegNet segmentation outputs with optical flow techniques applied to stereo cameras, aiming to enhance both visual odometry and 3D map reconstruction.

To improve tracking accuracy, PLD-SLAM (Point Line Dynamic SLAM) proposes extracting point and line characteristics [129]. For motion estimation, features that relate to dynamic areas are removed. Cheng et al. on the other hand, suggested a method based on optical flow to reduce the amount of point features extracted from moving objects [130]. Similarly, DRSO-SLAM [131] also leverages semantic information of the Mask R-CNN. In this case, the authors combined semantics with optical flow to improve the performance of tracking, local mapping, and loop detection in ORB-SLAM2. Dynamic objects optical flow is tracked using pixel-level semantic information and fundamental matrix calculations, in order to isolate them from the static feature points.

Finally, there are more sophisticated approaches to dealing with dynamics. Li and Lee proposed to assign a weight to each image point, that indicates how likely it is that the point relates to a dynamic object [96]. This information is included into an Intensity Assisted Iterative Closest Point (IAICP) algorithm to reduce the effect of moving objects on the motion estimation process.

IV. The Visual SLAM Domain

Cameras are the primary sensor choice for SLAM because low-cost video sensors provide rich information about the environment. Due to the large number of articles in this topic, we propose here to explain the general taxonomy of V-SLAM. Two significant elements that help in understanding how visual SLAM is organised are the type of cameras used in implementations and how the systems process the incoming frames.

A. Camera Types

By camera type, we refer to the physical structure of the device itself. *Monocular* cameras have a single lens, while *stereo* cameras feature two lenses

Monocular SLAM systems perform pose estimation from a single image, configuring the most cheap way to do this task in V-SLAM [49], [132]. Due to its low weight and power consumption with respect to other sensor configurations, monocular cameras are widely used in many applications, but specially on flying robots [133].

This SLAM type presents one significant weakness: Depth estimation. Monocular cameras are incapable of perceiving depth. As a result, the map produced by the SLAM system has scale ambiguity.

To tackle the scale recovery issue, an inertial sensor can be attached to the camera, thus performing Visual-Inertial SLAM as indicated in Sec III-B5. However, Yang et al. proposed an alternative solution to help monocular cameras perceive depth without the need of IMU's. By leveraging Deep Neural Networks (DNNs), their system is able to estimate the stereo disparity from a single image [134]. Another valid (yet expensive) solution would be having multiple robots equipped with monocular cameras, as proposed in [135].

Stereo SLAM techniques do not have scale recovery issues. This is due to the fact that given two images of the same view, the metric scale can be determined with geometric calculations [136], [137]. However, stereo cameras are more expensive than monocular cameras and are equally susceptible to motion blur. As a result, under such conditions, they may require also the assistance of another sensor to obtain the motion estimation.

RGB-D cameras like the well known Kinect and Intel RealSense, have gained significant attention and utility in the SLAM field [138], [139]. These cameras provide both color (RGB) and depth (D) information, enabling more accurate and detailed scene perception. By combining RGB and depth data, SLAM algorithms can extract robust feature points, estimate camera poses, and build dense 3D maps of the environment [140], [141]. The depth information enhances the perception capabilities of SLAM systems, allowing for better understanding of the scene geometry and improved obstacle avoidance.

Event-based cameras, also known as neuromorphic or event-driven cameras, have emerged as a novel sensing technology for SLAM. Unlike traditional cameras, that capture images at a fixed rate, they asynchronously measure per-pixel brightness changes, and output a stream of events that encode the time, location and sign of the brightness changes [142]. This unique characteristic allows event-based cameras to provide high temporal resolution, low latency, and a high dynamic range, making them particularly suitable for fast motion and dynamic scenes [143], [144]. Additionally, event-based cameras consume significantly less power compared to traditional cameras, making them suitable for resource-constrained applications. As event-based cameras continue to evolve, their integration into SLAM systems shows great potential for enhancing real-time perception and navigation capabilities.

It is worth mentioning that most V-SLAM algorithms are designed to work with a single camera type. To address this situation, Sumikura et al. proposed OpenVSLAM (Open Visual SLAM), an open-source framework compatible with a variety of camera types [78]. Users can compare how different types of cameras perform in a pipeline similar to ORB-SLAM using their system.

B. Input Frame Processing

V-SLAM systems perform motion estimation by processing the input images provided by cameras. Two main approaches to image processing can be recognized: *Direct* methods, that use the pixel values of an image to perform motion estimation, and *feature based* or *indirect* methods, that create a representation of the scene before performing the calculations.

Figure 3 depicts a generalized flow diagram of this approaches. Direct methods apply a pixel selection strategy to choose the pixel set to be used for pose estimation. After this step, a direct image alignment process is applied to estimate camera motion. Indirect methods on the other hand, start by extracting a set of features from the input frame. Then, the system tries to find this features in the previous image through a process named feature matching. After that, a rigid-body transformation to represent the camera motion is estiamted with the matched features.

The paper series by Engel et al. [40], [137], [145] is the key to understand how direct approaches work. They set the

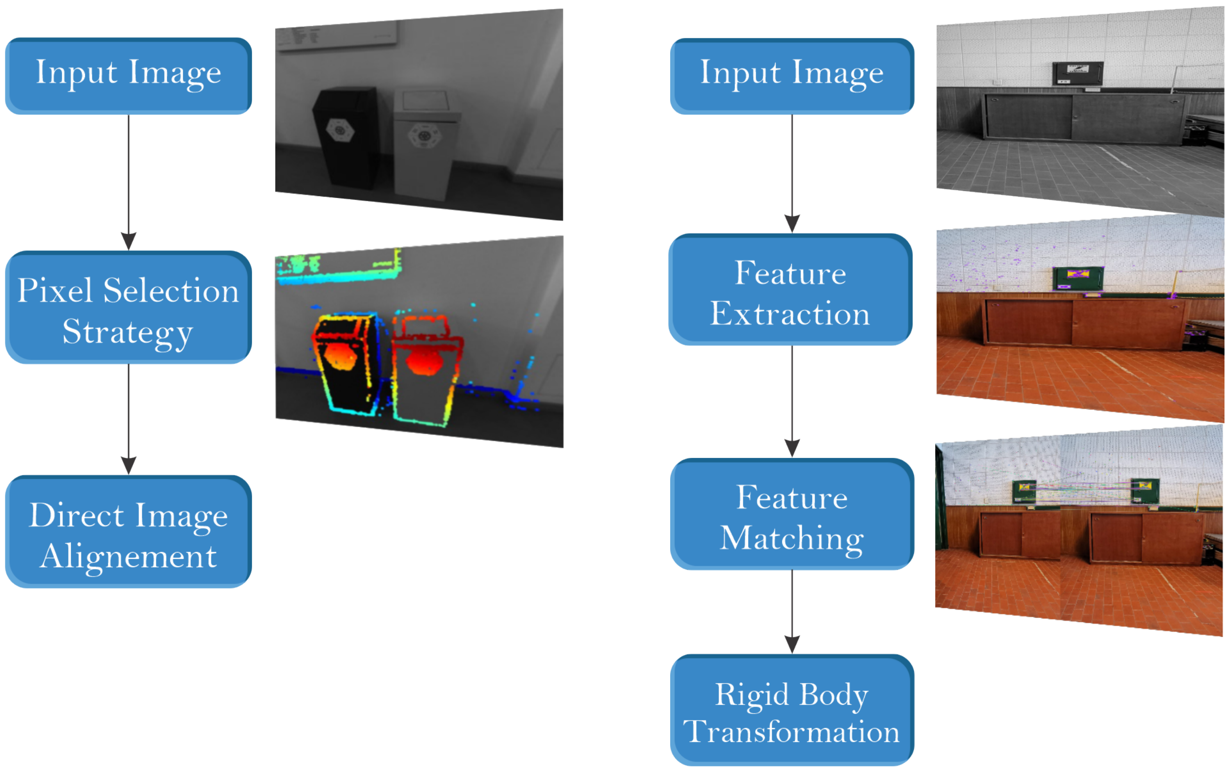

Fig. 3. Generalized flow diagram of the main input frame processing methods. **Left:** direct methods apply pixel selection strategies to select the best pixels to perform motion estimation. **Right:** indirect methods use features to perform calculations. To select the best features, the system combines an image feature extraction algorithm with a feature matching process.

path for many recent articles in this field who followed their ideas [112], [134], [146]–[148]. The formulation of the motion estimation performed by a direct system is explained below, based on Engel's work [40].

Since the available information is the measured intensity level of a set of selected pixels in the image, direct methods estimate the change in camera position performing direct image alignment. This is achieved by optimizing a photometric error such as (1)

$$E_{photo} := \sum_{i \in \mathcal{F}} \sum_{\mathbf{p} \in \mathcal{P}_i} \sum_{j \in obs(\mathbf{p})} E_{\mathbf{p},j}, \quad (1)$$

where i runs over all frames $\mathcal{F}$, $\mathbf{p}$ over all points $\mathcal{P}_i$ in frame i, and j over all observed frames $obs(\mathbf{p})$ in which the point $\mathbf{p}$ is visible. $E_{\mathbf{p},j}$ represents the weighted photometric error of a point $\mathbf{p}$ in keyframe I_i observed from a target frame I_j, with respect to a local neighborhood $\mathcal{N}_p$ of pixels around it. Let

$$E_{\mathbf{p},j} := \sum_{\mathbf{p} \in \mathcal{N}_p} w_{\mathbf{p}} \left\| (I_j[\mathbf{p}'] - b_j) - \frac{t_j e^{a_j}}{t_i e^{a_i}} (I_i[\mathbf{p}] - b_i) \right\|_\gamma, \quad (2)$$

where $\|.\|_\gamma$ is the Huber norm, and $\mathbf{p}'$ represents the projected position of $\mathbf{p}$.

It is worth noting that the exposure times t_i and t_j for images I_i and I_j are required for this formulation. This information isn't always readily available, and so requires the use of a calibrated dataset. As a result, some authors argue that these variables should be removed from the equation (2). Further information on the gradient-dependent weights $w_{\mathbf{p}}$ and brightness transfer function parameters a_i,b_i,a_j,b_J can be found on [40]

Indirect approaches, on the other hand, generate a scene representation before performing calculations. The image's principal points (also known as keypoints) are extracted, and for each of them, a descriptor is calculated. An image feature is comprised of a keypoint and a descriptor. Features contain position information, which allows the camera movement to be tracked by geometric calculations. Therefore, these approaches optimize a geometric error.

There are several local image feature extractors in the literature. Some of the most commonly used algorithms include SIFT [149], SURF [150], ORB [151], BRIEF [152], FAST [153], and BRISK [154]. Neural networks can also be used for feature extraction [155]. GCN (Geometric Correspondence Network) and GCNv2 (version 2) generate binary descriptors designed to replace ORB features in systems like ORB-SLAM 2 [156], [157]. Analogously, the SPHORB algorithm allows to extract ORB features from spherical images [158].

The cornerstone of current *Indirect* methods [36], [38], [39], [57], [158], [159] is ORB-SLAM [33]. A brief description of the tracking thread of this approach is presented below, based on [35].

The tracking algorithm starts by extracting a set of ORB descriptors from the input frame. Then, the system performs descriptor matching with the previous image. For each key point, this process looks for a correspondence in the previous frame. If enough correspondences are found, the system calculates a rigid-body transformation to represent the camera motion. This transformation provides an initial guess for the vehicle's pose. After that, the pose is refined by a process named *motion-only* Bundle Adjustment.

From the input frame, the tracking algorithm extracts a set of ORB descriptors. The machine then compares descriptors with the prior image. This procedure looks for a match in the previous frame for each key point. The system calculates a rigid-body transformation to represent the camera motion if enough correspondences are detected. This transformation yields a first guess about the vehicle's position. After that, a method known as *motion-only* Bundle Adjustment is used to fine-tune the position.

Let $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$ be the camera rotation and translation respectively. The *motion-only* Bundle Adjustment algorithm optimizes these two variables by minimizing the reprojection error between matched 3D points $\mathbf{X}^i \in \mathbb{R}^3$ in world coordinates and keypoints $x^i_{(.)}$, either monocular $x^i_m \in \mathbb{R}^2$ or stereo $x^i_s \in \mathbb{R}^3$, with $i \in \mathcal{X}$ the set of all matches:

$$\{\mathbf{R}, \mathbf{t}\} = \underset{\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum_{i \in \mathcal{X}} \rho \left(\left\| x^i_{(.)} - \pi_{(.)}(\mathbf{R}\mathbf{X}^i + \mathbf{t}) \right\|^2_{\Sigma} \right) \quad (3)$$

where ρ is the Huber cost function and Σ the covariance matrix associated to the scale of the keypoint. The projection function $\pi_{(.)}$ is defined by (4) in the monocular case, and by (5) for stereo cameras.

$$\pi_m \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \end{bmatrix} \quad (4)$$

$$\pi_s \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \\ f_x \frac{X-b}{Z} + c_x \end{bmatrix} \quad (5)$$

where f_x, f_y are the camera's focal lengths, c_x,c_y the principal points and X, Y, Z the matched keypoint world coordinates.

If the system is unable to locate sufficient matches, tracking is considered lost, and relocalization techniques are required to recover it.

Indirect techniques exhibit suboptimal performance in two scenarios: Texture-less environments and repetitive scenarios. Because of the low gradients in the scene, the former prevents feature extraction techniques from working. The latter confuses the loop closure thread because all of the images appear to be from the same location.

Hybrid V-SLAM methods are a third type of V-SLAM system that aims to combine the benefits of direct and indirect approaches [160], [161]. To support direct image alignment methods, this systems rely on the robustness of feature matching, giving them the name of *semi-direct* because of this [41], [162], [163]. In most of the cases, hybrid approaches match the performance of direct and indirect methods. They do not, however, bring about a significant increase in accuracy.

V. The Influence of Artificial Intelligence in Current SLAM systems

Artificial intelligence (AI) and particularly deep learning algorithms, have gained great popularity among academics from different disciplines. This section discusses the influence of AI on the development of algorithms for SLAM.

A. Feature Extraction Process

After being appropriately trained, one of the fundamental strengths of neural networks is their capacity to generate an output with a single pass of information. To accomplish this, researchers have diligently focused on training networks for image feature extraction. This approach seeks to efficiently incorporate iterative algorithms into the network's internal weights, leading to a reduction in computational time [156], [157].

SuperPoint [155] is a fully-convolutional neural network trained in a self-supervised manner able to extract SIFT-like key points and their corresponding descriptors from the image. Another example is the Learned Invariant Feature Transform (LIFT) network [164]. LIFT was end-to-end trained to perform the full feature extraction process: detection, orientation estimation, and feature description. It produces descriptor vectors, which served as the foundational components for LIFT-SLAM [49].

In terms of integration of deep features, DxSLAM [165] showed that feature extraction with deep convolutional neural networks (CNNs) can be seamlessly incorporated into a modern SLAM framework. It utilized a state-of-the-art CNN to detect keypoints in each image frame and to give not only keypoint descriptors, but also a global descriptor of the whole image. These local and global features are then used by different SLAM modules, resulting in much more robustness against environmental changes and viewpoint changes compared with using the hand-crafted features

Weyand et al. harnessed this capability to develop PlaNet, a network capable of geolocating outdoor images anywhere on Earth [166]. In this system, localization is approached as a classification problem, where the world is partitioned into discrete cells. When an image is fed into the network, it computes the likelihood that it corresponds to a particular location.

B. Loop Closure Detection

Conventional methods for loop closure detection face challenges in distinguishing between two locations with subtle differences. Additionally, an error in the loop closure module could potentially lead to the complete loss of the map. These limitations, combined with developments in machine learning, have encouraged researchers to examine data-driven approaches as a complement to existing SLAM systems [15], [167].

It has been demonstrated that the intermediate layers of a deep network can extract information from the inputs that allows for a more accurate characterization of them. In this way, this information can be used to enhance the performance of loop closure detection algorithms. An illustrative instance of this concept can be found in [168] where the authors combined the output of a CNN-based loop closure detection block with the output of traditional sequence-based matching algorithm to handle the viewpoint and condition variance problem.

While the Bag of Words (BoW) algorithm plays a pivotal role in many Loop Closure Detection (LCD) strategies, researchers have been actively exploring ways to enhance or replace it. Memon et al. [169] proposed a solution to speed-up the process of LCD in long trajectories. They combine supervised and unsupervised learning methods to compare scenes faster. On one front, they employ an autoencoder architecture to determine whether the current scene has been previously visited. If the scene has been encountered before, the autoencoder's reconstruction error falls below a predefined threshold, identifying the current frame as a loop closure candidate. Concurrently, they introduce the concept of a "superdictionary" that works in tandem with the BoW dictionary to mitigate the risk of overlooking genuine loop closures. This innovative approach aims to improve the efficiency and accuracy of loop closure detection."

In contrast, Gao et al. [170] proposed a transition from the traditional BoW method to a denoising auto-encoder (SDA), a multi-layer neural network designed to autonomously learn a compressed representation from raw input data. In this approach, newly generated keyframes are introduced as inputs to the SDA, which generates a feature response. Subsequently, a similarity metric is computed with respect to previous keyframes to determine the presence of a loop.

C. Pose Estimation

Pose estimation techniques have undergone a remarkable transformation, thanks to the advancements in neural networks and deep learning. One of the most important contributions to pose estimation with neural networks is the seminal work by Kendall et al. [171]. The PoseNet network outputs the camera pose $\mathbf{p} = [\mathbf{x}, \mathbf{q}]$, given by a 3D translation vector $\mathbf{x}$ and a rotation vector represented by quaternion $\mathbf{q}$. To obtain the camera pose, the authors propose to jointly optimize a translation related term $\mathcal{L}_x$ and a rotation term $\mathcal{L}_q$ defined as:

$$\mathcal{L}_x = \|\hat{\mathbf{x}} - \mathbf{x}\|_2 \qquad \mathcal{L}_q = \|\hat{\mathbf{q}} - \mathbf{q}\|_2 \tag{6}$$

where $\mathbf{x}, \mathbf{q}$ are ground truth and $\hat{\mathbf{x}}, \hat{\mathbf{q}}$ are the values estimated by the net. These two elements are combined into a single Euclidean loss function to be minimized, expressed by

$$L = \mathcal{L}_x + \beta \mathcal{L}_q \quad , \tag{7}$$

where β is a scale factor. Analogously, Walch et al. [172] proposed to regress the camera pose wit a combination of CNN+LSTM networks. They add four LSTM units at after the last fully connected layer of a GoogLeNet, and train the network with (7).

The pose estimation obtained by PoseNet can be improved if the fixed parameter β in (7) is replaced with trainable parameters [173]. The resulting loss function is (8), where $\hat{s}_x$ and $\hat{s}_q$ are learnable parameters for translation and rotation respectively.

$$L = \mathcal{L}_x \exp(-\hat{s}_x) + \hat{s}_x + \mathcal{L}_q \exp(-\hat{s}_q) + \hat{s}_q \tag{8}$$

Recent studies in pose estimation utilize multiple data sources. DeepICN [174], for instance, employs RGB and depth data as inputs for the inverse compositional algorithm (ICA) to estimate relative pose. The authors proposed a modification to the ICA where the image feature extraction and optimization information are learned by the neural networks. An et al. proposed in [175] a visual-LiDAR odometry with a multi-channel recurrent convolutional neural network (RCNN). This method fuses RGB images and LiDAR data to estimate the pose. A hybrid architecture consisting of CNN and multiple BiLSTM layers is proposed in [176]. HVIOnet is an end-to-end trained hybrid network that allows to combine raw camera and raw IMU measurements for position estimation. A key aspect of this work is that by taking raw measurements it eliminates the need for camera calibration adjustments.

D. Depth Perception to Improve Pose Estimation

In the context of Simultaneous Localization and Mapping (SLAM), the utilization of neural networks for depth prediction contributes significantly to pose estimation. Neural networks have proven to be an effective method for accomplishing this goal and overcoming the limitations of monocular cameras in terms of depth perception [177]–[180]. Yang et al. proposed using monocular images to do stereo depth prediction, in order to improve odometry estimations [134]. UnDeepVO (Visual Odometry through Unsupervised Deep learning), on the other hand, is a neural network-based system that can generate a depth map and 6-DoF pose estimation for a monocular image [181].

SfMLearner (Structure from Motion Learner), for instance, implements a direct image alignment method augmented by depth measurements [182]. This approach concurrently deploys a *Depth CNN* and a *PoseNet* network, with the former generating a depth map from monocular imagery and the latter determining camera pose. A similar strategy is seen in the CNN SLAM framework [183], where a predicted dense depth map is fused with depth values estimated by a monocular SLAM system called LSD-SLAM [145]. This integration helps address a common challenge in monocular SLAM: scale estimation.

Inspired by the parallel structure of SfMLearner [182], Yang et al. introduced D3VO (Deep Depth, Deep Pose, and Deep Uncertainty for Monocular Visual Odometry), a SLAM system primarily based on neural networks [148]. D3VO employs a Depth CNN to calculate pixel-wise photometric uncertainty in input images, thereby enhancing the accuracy of depth estimation. The system is further equipped with pose refinement and map optimization modules.

In a different approach detailed in [184], the authors propose a bundle adjustment network (BANet) capable of jointly optimizing the map and camera poses through a differentiable Levenberg–Marquardt algorithm. In order to achieve this goal, the system initially generates several basis depth maps according to the input image, and optimizes the final depth as a linear combination of these basis depth maps using feature-metric bundle adjustment. At the same time, a feature pyramid constructor generates multi-scale feature maps. These depth and feature maps are then fed into the bundle adjustment layer to refine both the camera pose and map estimation.

Traditional direct image alignment algorithms use all pixels to optimize the pose estimation. However, not all pixels contribute to the convergence of the optimizer. FlowNorm [185] introduces an optical flow graph to identify pixels that have a poor impact on optimizer convergence. These pixels are assigned smaller weights, thereby expanding the convergence range of the optimization process.

E. NNs and LiDAR Measurements

Artificial intelligence algorithms are also being used in LiDAR-based techniques [186], [187]. Spampinato et al. proposed a sequential convolutional NN that determines the robot pose given two consecutive laser scans [188]. This network outputs a vector $T = [\Delta x, \Delta y, \Delta \alpha]$, where Δx and Δy are the translation coefficients and $\Delta \alpha$ the rotation coefficient. Chen et al. on the other hand, proposed the OverlapNet for loop closure detection [189]. This two-legged network can estimate the percentage of overlap between two consecutive laser scans. According to the authors, a threshold based on the overlap percentage can be used to determine whether two LiDAR scans are in the same location and/or whether a loop closure can be performed.

At the same time, Lu et al. presented L^3-net (Learning based LiDAR Localization), a global localization method based on 3D convolutions [190]. L^3-net is comprised of three neural networks. A PointNet is used to extract feature descriptors first. Then, the ideal pose is calculated using a combination of a CNN and a recurrent neural network (RNN). Similar to L^3-net, DeepLo (Deep LiDAR Odometry) combines the action of two neural networks. A VertexNet is used to extract features from LiDAR scans, and a PoseNet to estimate the 6-DoF camera pose [191].

F. Aiding VI SLAM

Long Short-Term Memory neural networks are well-known for their ability to classify, process, and predict data from time series. As a result, they're suitable for processing inertial measurements. This feature is used in Visual-Inertial techniques to combine IMU and camera information [192], [193].

Several approaches stand out among Visual-Inertial methodologies. VINet, for instance, combines LSTM and convolutional networks to perform motion estimation [194]. Similarly, DeepVIO [195] consists of a CNN for image flow calculations, an LSTM for IMU measurements processing, and fully connected layers for data fusion. This integrated design enables the system to concurrently compute continuous trajectories by leveraging monocular images and IMU data.

However, LSTMs are note the only option. In [196], Shamwell et al. presented VIOLearner, a system that forgoes the use of LSTMs and instead employs Convolutional Neural Networks for multiple instances of pose refinement. This approach demonstrates the diverse strategies employed within the field of Visual-Inertial techniques for enhancing pose estimation and trajectory tracking.

In recent times, advanced systems have emerged as cutting-edge solutions. In their study [197], Aslan et al. introduced VIIONet (Visual-Inertial Image-Odometry Network), a sophisticated end-to-end trained system designed for estimating the pose of an Unmanned Aerial Vehicle (UAV). The system initially takes image and IMU measurement inputs to generate both camera optical flow (OF) frames and IMU-OF frames. This information is processed separately by two Inception V3 [198] networks to extract abstract features, which are subsequently used in the final step of pose regression. In contrast, SelfVIO (Self-supervised deep learning-based VIO) performs camera pose estimation and depth map recovery in a self supervised way [199]. By using adversarial training and self-adaptive visual–inertial sensor fusion, this system, learns the joint estimation of 6-DoF ego-motion and a depth map of the scene from unlabelled monocular RGB image sequences and inertial measurement unit (IMU) readings.

G. Semantic Image Segmentation

Recent advances in computing power have opened up a new source of data: Semantic image segmentation. Semantic segmentation is a task that involves labeling categories at the pixel level of an image, and it has direct applications in the field of computer vision [200], [201].

SLAM systems can benefit from a more comprehensive understanding of the scene, achieved through the incorporation of data on static/dynamic behavior and object classification [202]–[204]. Take, for instance, VLocNet++ (Visual Localization Network), which introduces an additional term to the loss function (8) that directly accounts for image semantic information [205]. This strategy allows to add geometric and semantic knowledge of the world into the pose regression network.

Jin et al. propposed to add semantic information to the multi-view geometry method in the tracking thread of ORB-SLAM2 to filter out feature points belonging to dynamic objects in the environment [206]. To maintain consistent performance in real-world scenarios, their system underwent training using adversarial transfer learning techniques to mitigate potential performance degradation within the semantic segmentation network.

Furthermore, Zhao et al. introduced a novel system in [207], known as KSF-SLAM (Key Segmentation Frame-based Semantic SLAM), designed for dynamic environments. This system implements a frame selection strategy to determine whether image segmentation is required in the current frame. This optimization approach results in a reduced computational load, as it minimizes the frequency of segmentation procedures.

H. Mapping

The map generated by SLAM systems that integrate artificial intelligence typically benefits indirectly. Consequently, it is challenging to identify articles that specifically contribute to this component of the system. Nevertheless, certain noteworthy contributions can be highlighted. Valada et al. proposed to modify the equation (8) to integrate the relative motion between the current image and the previous predicted pose [208]. This modification helps to maintain the consistency of the map. Fusion++ uses object-level SLAM to produce a more comprehensive map representation [209]. As a result, a 3D map containing reconstructed objects in the scene is generated, allowing the user to better understand the scene observed by the robot. Lastly, it is noteworthy to mention the groundbreaking research by Sucar et al. in [210]. Their work on iMAP (Implicit Mapping and Positioning) introduces a cutting-edge real-time SLAM system. iMAP leverages an implicit neural scene representation and is capable of jointly optimizing a full 3D map and camera poses. A particularly salient feature of this system is its capacity for real-time training, achieved without the reliance on prior data.

I. Practical Considerations About Using AI in SLAM

*1) **Hardware Resources**:* The implementation of neural networks requires careful consideration of hardware components to ensure efficient and effective operation. Firstly, the choice of the central processing unit (CPU) or graphics processing unit (GPU) plays a pivotal role. While CPUs are versatile and suitable for many tasks, GPUs excel in parallel processing, making them particularly well-suited for deep learning tasks.

Memory capacity is another critical aspect, as large neural networks require substantial memory resources. High-speed RAM, such as Graphics Double Data Rate (GDDR) memory for GPUs, is vital for rapid data access during training and inference. Additionally, storage solutions with fast read and write speeds are essential for managing large datasets. Specialized hardware accelerators, like tensor processing units (TPUs) or field-programmable gate arrays (FPGAs), are emerging as dedicated options for neural network tasks.

Finally, it is noteworthy that the majority of AI research applied to SLAM is conducted using desktop computers, while the intended deployment targets mobile robots. This presents a noteworthy challenge given the power limitations of autonomous systems. Systems incorporating diverse data sources or employing complex deep neural networks may demand substantial computational and thermal management capabilities to achieve real-time operation and high precision. Nonetheless, greater computational capacity corresponds to increased power consumption. Conversely, reduced computational capacity reduces power consumption but imposes limitations on the volume of data the system can effectively handle. Consequently, there is a need to calibrate accuracy and timing objectives accordingly. This inherent trade-off is explored further in [133], where multiple systems are evaluated across various hardware platforms.

*2) **Training Sets, Generalization, and Learning Strategies**:* Data preparation is a crucial step in the training of neural networks. The selected training set (images, IMU readings, LiDAR measurements, etc.) must strike a delicate balance. It should span a diverse array of scenarios that the system might encounter during operation, while also minimizing the computational time required for training. The system's ability to generalize effectively in real-world scenarios highly depends on the careful selection of a training dataset.

Generalization poses a challenge when employing neural networks in general, and particularly within the context of SLAM systems. For instance, consider the PLD-SLAM system discussed in Section III-C2. Within its architecture, it incorporates a dynamic object detection module based on MobileNet [129], [203]. Nevertheless, MobileNet is limited to classifying a finite number of objects. Any dynamic object not included during the training phase may encounter misclassification, consequently leading to a decline in system performance.

Similarly, CNN-SVO (CNN Sparse Visual Odometry) [225] serves as another illustration. This system leverages CNNs to reduce pose estimation uncertainty, aiming to enhance the overall algorithmic performance of the original SVO. While it exhibits strong performance when tested with the training dataset, its performance diminishes when subjected to unfamiliar datasets.

However, in recent years, researchers have actively explored alternatives to address this issue. The primary focus was centered on novel network training strategies, giving rise to solutions such as Unsupervised Learning or Self-Supervised Learning [134], [196], [199], Adversarial Training [206], and Reinforcement Learning [226].

VI. Methods Comparison

This section provides a performance comparison of the diverse systems analyzed in this article. We begin by outlining practical considerations for implementing SLAM systems, followed by a concise explanation of the selected metrics for system comparison. We then proceed to present performance comparison tables.

A. Practical Details of SLAM Systems Implementation

*1) **Operating System and System Requirements**:* While this may appear evident, given the rapid pace of technological advancement, verifying the operating system version on which a system is developed is a necessary first step in implementation. The selection of the operating system significantly impacts crucial aspects such as the required library versions and the specific ROS (Robot Operating System) version being employed.

Take ORB-SLAM as an example. Suppose you want to implement and test the system exactly as the authors have published it on their Github page (https://github.com/raulmur/ORB_SLAM). In the system description, the authors delineate the testing environment, which included Ubuntu 12.04 with ROS versions Fuerte, Groovy, and Hydro, as well as Ubuntu 14.04 with ROS Indigo. Furthermore, the system is compatible with OpenCV library version 2.4. It's essential to note that if, during your implementation, any of these specified libraries are no longer accessible, the system may not work as intended.

Another seemingly trivial yet indispensable factor is the evaluation of the hardware prerequisites of the system. While numerous existing developments have undemanding prerequisites, this aspect becomes paramount when implementing SLAM in embedded systems. Systems involving computations like image segmentation or those utilizing artificial intelligence have specific hardware needs that must be taken into account.

TABLE I
COMMONLY USED DATASETS FOR SLAM

Dataset	Year	Sequences	Path Length	Ground Truth	GPS	IMU	Laser Scans	Image			Scenario	Source
KITTI	2012	22	50 km	Yes	Yes	Yes	Yes	Stereo	Grayscale	1392x512 @ 10 fps	Outdoor	[211]
								Stereo	RGB	1392x512 @ 10 fps		
EuRoC	2016	11	0.9 km	Yes	No	Yes	No	Stereo	Grayscale	752x480 @ 20 fps	Indoor	[212]
RobotCar	2015	-	1010.46 km	No	No	Yes	Yes	Stereo	RGB	1280x960	Outdoor	[213]
								Monocular	Grayscale	1024x1024		
TUM RGB-D	2012	39	0.514 km	Yes	No	No	No	Monocular	RGB-D	640x480 @30 fps	Indoor	[214]
TUM Mono	2016	50	-	Yes	No	No	No	Monocular	Grayscale	1280x1024 @ 20-50 fps	Indoor/ Outdoor	[215]
TUM VI	2018	28	20 km	Yes	No	Yes	No	Monocular	Grayscale	1024x1024 @ 20 fps	Indoor/ Outdoor	[216]
Málaga	2009	6	6 km	Yes	Yes	Yes	Yes	Stereo	RGB	1024 x 768 @ 7.5 fps	Outdoor	[217]
Ford Campus	2011	-	6 km	Yes	Yes	Yes	Yes	Omnnidirectional	RGB	1600 x 600 @ 8 fps	Outdoor	[218]
ICL NUIM	2014	8	< 1 km	Yes	No	No	No	Monocular	RGB-D	-	Indoor Artificial	[219]
Michigan North Campus	2013	27	147.4 km	Yes	Yes	Yes	Yes	Omnidirectional	RGB	1600X200 @ 5fps	Indoor/ Outdoor	[220]
MIT Sata	2012		42 km	Yes	No	Yes	Yes	Monocular	RGB-D	-	Indoor	[221]
								Stereo	RGB			
7 scenes	2013	7	-	Yes	No	No	No	Monocular	RGB-D	640x480 @30fps	Indoor	[222]
Málaga Urban	2014	15	36.8 km	No	Yes	Yes	Yes	Stereo	RGB	1024 x 768 @ 20 fps	Outdoor	[223]
MIT DARPA	2010	3	90 km	Yes	Yes	Yes	Yes	Stereo	RGB	376 x 240 @ 10 fps	Outdoor	[224]
								Mono	RGB	752 x 480 @ 22.8 fps		

*2) **Required Sensors**:* While SLAM systems explicitly state this condition (visual, visual-inertial, lidar SLAM, etc.) a priori, it's crucial to delve into the details. Unfortunately, this often entails carefully evaluating the code, as the system may internally require proprietary libraries or specific drivers.

One way to address this is to opt for systems that publish measurements as ROS topics. This simplifies the replacement of sensors within the system. All you need to do is ensure that the new sensor publishes a topic with the same name as the original.

B. Accuracy Metrics

To evaluate the performance of any SLAM system, it is necessary to have evaluation metrics. There are two main ways to estimate the accuracy of a SLAM system: the relative pose error (RPE) and the absolute trajectory error (ATE). The RPE measures the local accuracy over a fixed time interval Δ, while the ATE measures the global consistency of the estimated trajectory.

Given a sequence of poses from the estimated trajectory $\mathbf{P}_1, ..., \mathbf{P}_n \in SE(3)$ and from the ground truth trajectory poses $\mathbf{Q}_1, ..., \mathbf{Q}_n \in SE(3)$, the RPE at time step i is defined as

$$\mathbf{E}_i = \left(\mathbf{Q}_i^{-1}\mathbf{Q}_{i+\Delta}\right)^{-1}\left(\mathbf{P}_i^{-1}\mathbf{P}_{i+\Delta}\right) \tag{9}$$

From a sequence of n poses, $m = n - \Delta$ individual RPEs can be obtained. Taking into account these errors, the root mean squared error (RMSE) over all time indices of the translational and rotational components of (9) is calculated as

$$\begin{aligned} RMSE(\mathbf{E}_{1:n}, \Delta)_{trans} &= \sqrt{\frac{1}{m}\textstyle\sum_{i=1}^{m}\|trans(\mathbf{E}_i)\|^2} \\ RMSE(\mathbf{E}_{1:n}, \Delta)_{rot} &= \sqrt{\frac{1}{m}\textstyle\sum_{i=1}^{m}\|rot(\mathbf{E}_i)\|^2} \end{aligned} \tag{10}$$

These two values help to determine the precision of SLAM systems in terms of both translation and rotation errors. The translational and rotational components of the RPE will be referred to as t_{rel} and r_{rel}, respectively, in the following notation.

Unlike the RPE, the ATE computes the absolute distances between points in the estimated and ground truth trajectories. It is a global measurement that needs both trajectories being aligned in the same coordinate frame beforehand. One way to achieve this goal is to use the Horn method [227], which finds the rigid-body transformation $\mathbf{P}$ that maps the estimated trajectory $\mathbf{S}_{1:n}$ onto the ground truth trajectory $\mathbf{Q}_{1:n}$.

Given a rigid body transformation between the estimated trajectory and the ground truth data, the ATE at time step i can be computed as

$$\mathbf{ATE}_i = \mathbf{Q}_i^{-1}\mathbf{S}\mathbf{P}_i \tag{11}$$

Analogous to the RPE, the RMSE value of (11) is calculated. However, this time only the translational component is considered.

$$ATE_{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\|trans(\mathbf{ATE}_i)\|^2} \tag{12}$$

C. Available Datasets

A SLAM dataset includes a number of trajectories recorded in different scenarios. The measurements acquired by a variety of sensors throughout the execution of the trajectories are different for each dataset. Researchers can use this information to develop and test their algorithms. The obtained results are compared to the reference data provided by the dataset (also referred as ground-truth) through the use of accuracy metrics.

A SLAM dataset contains numerous trajectories recorded across diverse scenarios, storing measurements from different sensors. Researchers leverage this data to develop and test their algorithms. Once their system is tested, they can compare their results to the reference data provided by the dataset (also referred as ground-truth) through the use of accuracy metrics. Therefore, it is important to take into account how many datasets are available and what types of data they contain. Table I lists the properties of some of the most popular SLAM datasets.

The Karlsruhe Institute of Technology and Toyota Technological Institute dataset (KITTI) is one of the most widely used for SLAM applications [211]. It covers around fifty kilometers of a urban environment, divided in twenty-two sequences. This

TABLE II
MEDIAN 6D LOCATION ERROR FOR 7 SCENES DATASET SEQUENCES*

System	Measure	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
DSAC[1] [228]	Trans. [m]	**0,020**	0,040	0,030	0,040	0,050	0,050	1,170
	Rot. [deg]	1,20	1,50	2,70	1,60	2,00	2,00	33,10
DSAC ++ [229]	Trans. [m]	**0,020**	0,020	**0,010**	0,030	0,040	0,040	0,090
	Rot. [deg]	0,50	0,90	0,80	0,70	1,10	1,10	2,60
VLocNet [208]	Trans. [m]	0,036	0,039	0,046	0,039	0,037	0,039	0,097
	Rot. [deg]	1,71	5,34	6,64	1,95	2,28	2,20	6,48
VLocNet ++ [205]	Trans. [m]	0,023	**0,018**	0,016	**0,024**	**0,024**	**0,025**	**0,021**
	Rot. [deg]	1,44	1,39	0,99	1,14	1,45	2,27	1,08
PoseNet [171]	Trans. [m]	0,320	0,470	0,290	0,480	0,470	0,590	0,470
	Rot. [deg]	4,06	7,33	6,00	3,84	4,21	4,32	6,93

* Best results for each sequence in **Bold**.
The symbol '-' stands for *no results reported* and 'X' for tracking failure.
1 Data extracted from [229].

dataset provides information from GPS, IMU, laser scanners and stereo images (both in color and grayscale). Additionally, the sequences are separated for training (00-10) and testing (11-22). It also provides accurate ground truth, making it suitable for all kinds of SLAM systems.

In the context of the European Robotics Challenge (EuRoC), a dataset for visual inertial systems was recorded with a micro-aerial vehicle (MAV) [212]. The EuRoC MAV dataset contains eleven sequences, ranging from slow flights under good visual conditions to dynamic flights with motion blur and poor illumination. The available information includes raw IMU measurements, stereo grayscale images and 6-DOF gruond truth poses, recorded in an indoor industrial area.

Recent neural network-based systems are being tested on the Microsoft 7-scenes dataset [222]. It contains 7 sequences recorded in different indoor environments with a handheld Kinect RGB-D camera. Also, it offers ground truth poses for every frame. This set up makes it a suitable platform for neural network-based algorithms with depth prediction.

D. Comparison Tables

This section presents tables containing performance data for the SLAM systems discussed in this article. Only works who reported objective (i.e. numeric) results were considered to draw comparison tables. Those with qualitative and graphical results were excluded. Simulations were not conducted; instead, the tables were populated with data directly extracted from the original publications of the respective systems. It's important to note that any alteration in this approach is clearly stated inside the tables.

Best results are emphasized with **bold** numbers for translation and underlined numbers for rotation. The '-' symbol denotes sequences where the system failed to report results, whereas the "X" denotes sequences where tracking was lost.

Table II allows to compare the performance of neural network-based methods on the Microsoft 7-scenes dataset. The accuracy of the results is determined by the median pose error acquired after each sequence has been executed.

Although there isn't much difference between the performance of the different systems, the best overall performance can be attributed to VLocNet++. This could be caused by the extra term in the training loss function of this system, that takes into account semantic information when estimating pose.

When evaluating systems using the KITTI dataset researchers prefer to use only the training sequences (00-10). This is related to the KITTI Benchmark Suite's proposal to use testing sequences to develop a performance ranking [230]. LOAM and V-LOAM are the highest performers in the KITTI Benchmark Suite among the systems evaluated in this article at the time of publication. However, because the authors did not publish quantitative results of the system's accuracy in the original article, they are not included in the comparison tables.

Table III shows the systems that provided results for the KITTI dataset's training sequences in terms of the ATE. Looking at the table, it is easy to notice that most systems don't present data for KITTI sequence 01 or lose track of it. This could be related to the fact that Sequence 01 is a highway with a limited number of trackable close objects.

The best results were obtained by Log-SLAM, which has the most constant performance. However, the results for sequences 01, 03, 04, 05, 06, and 10 were not presented by the authors. The LIFT-SLAM results and Ali's proposal stand out among those who finish all of the sequences. It is also worth mentioning that, while Ali's LiDAR SLAM technique completes all of the sequences, its error rate is considerable when compared to the results obtained by other algorithms.

Systems that reported results for the training sequences of the KITTI dataset in terms of the RPE are listed in Table IV (next page). Results for the SfM Learner algorithm were extracted from [146], whereas for ORB-SLAM 2, data was taken from [49].

DeepLo has the best overall performance for the KITTI sequences 00-10. Nevertheless, these results should be read carefully. DeepLo is a neural network-based system that was trained using the 00-08 sequences. This explains his performance in these sequences. Furthermore, as shown by the results for sequences 09 and 10, the system seems unable to generalize to other cases. In light of this, the table additionally includes the second best results for sequences 00-08.

The RMSE of the absolute trajectory error is the standard metric for evaluating the performance of a system in the EuRoC MAV dataset (RMSE ATE). Table V lists the systems that submitted data for this dataset. It is clear that all systems perform consistently and produce similar outputs. However, the proposal of Luo et al. and visual-inertial ORB-SLAM 3 showed the best results.

The evaluation of processing time for SLAM systems is beyond the scope of this paper. However, time response can be critical for rescue and assistance applications, which require real-time data analysis. In this way, a trade-off between three critical aspects must be found: power consumption, time response and accuracy.

Systems that use a multiple data sources (e.g., images, inertial measurements, semantics, etc.) may require a large computational capacity. If high degrees of accuracy and real-time operation are desired, the developer must keep in mind that higher computational capacity means more power consumption. Also, whether or not a SLAM system can be placed on a robot is determined by its power consumption.

Reduced computing capacity, on the other hand, demands lower power consumption while limiting the quantity of data

TABLE III

SYSTEMS WHO REPORTED RESULTS ON THE KITTI SEQUENCES 00-10 IN TERMS OF THE ABSOLUTE TRAJECTORY ERROR (ATE). VALUES EXPRESSED IN METERS*

	KITTI Sequence										
Approach	**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**
LSD-SLAM[1] [145]	0,471	-	0,648	-	-	-	-	4,321	0,798	3,013	-
LoG-SLAM [167]	**0,233**	-	**0,288**	-	-	-	-	**1,316**	**0,233**	**0,716**	5,022
D3VO [148]	-	**1,070**	0,800	-	-	-	**0,670**	-	1,000	0,780	**0,620**
CNN-SVO [225]	17,527	X	50,512	3,459	2,441	8,151	11,509	6,514	10,976	10,687	4,835
DSO[2] [40]	113,184	-	116,811	1,394	**0,422**	47,461	55,617	16,719	111,083	52,225	11,090
LDSO [147]	9,322	11,680	31,980	2,850	1,220	5,100	13,550	2,960	129,020	21,640	17,360
LIFT-SLAM[3] [49]	9.84	X	34.23	**0.97**	**0.42**	11.5	16.58	3.98	82.61	54.91	30.34
Ali *et. al* [67]	6,982	18,779	13,733	1,481	0,810	**3,697**	3,258	3,093	11,473	5,659	4,379
ORB-SLAM [33]	5,33	X	21,28	1,51	1,62	4,85	12,34	2,26	46,68	6,62	8,80

* Best results for each sequence in **Bold**.
The symbol '-' stands for *no results reported* and 'X' for tracking failure.
1 Large-Scale Direct SLAM. Data extracted from [167]
2 Data extracted from [225]
3 Results correspond to the *LIFT-SLAM Fine tuned With EuRoC* version of the algorithm.

TABLE IV

SYSTEMS WHO REPORTED RESULTS ON THE KITTI SEQUENCES 00-10 IN TERMS OF THE RELATIVE POSE ERROR (RPE)*

			KITTI Sequence										
System	**Measure**		**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**
Stereo DSO [146]	t_{rel}	%	0,84	1,43	0,78	0,92	0,65	0,68	0,67	0,83	**0,98**	0,98	**0,49**
	r_{rel}	$[deg/m]$	0,26	0,09	0,21	0,16	0,15	0,19	0,2	0,36	0,25	0,18	0,18
DVSO [134]	t_{rel}	%	0,71	**1,18**	0,84	0,77	**0,35**	0,58	0,71	0,73	1,03	0,83	0,74
	r_{rel}	$[deg/m]$	0,24	0,11	0,22	0,18	0,06	0,22	0,2	0,35	0,25	0,21	0,21
SFMLearner[1] [182]	t_{rel}	%	66,35	35,17	58,75	10,78	4,49	18,67	25,88	21,33	21,9	18,77	14,33
	r_{rel}	$[deg/m]$	6,13	2,74	3,58	3,92	5,24	4,1	4,8	6,65	2,91	3,21	3,3
DeepVIO [195]	t_{rel}	%	11,62	-	4,52	-	-	2,86	-	2,71	2,13	1,38	0,85
	r_{rel}	$[deg/m]$	2,45	-	1,44	-	-	2,32	-	1,66	1,02	1,12	1,03
VIOLearner [196]	t_{rel}	%	1,5	-	1,2	-	-	0,97	-	0,84	1,56	2,27	2,74
	r_{rel}	$[deg/m]$	0,61	-	0,43	-	-	0,51	-	0,66	0,61	1,52	1,35
DeepLO [191]	t_{rel}	%	**0,32**	**0,16**	**0,15**	**0,04**	**0,01**	**0,11**	**0,03**	**0,08**	**0,09**	13,35	5,83
	r_{rel}	$[deg/m]$	0,12	0,05	0,05	0,01	0,01	0,07	0,07	0,05	0,04	4,45	3,53
SuMa [64]	t_{rel}	%	0,7	1,7	1,1	**0,7**	0,4	0,5	**0,4**	**0,4**	1	**0,5**	0,7
	r_{rel}	$[deg/m]$	0,3	0,5	0,4	0,5	0,3	0,2	0,2	0,3	0,4	0,3	0,3
ORB-SLAM [33]	t_{rel}	%	4,46	X	-	9,75	3,71	3,35	8,11	7,43	12,16	26,51	8,65
	r_{rel}	$[deg/m]$	3,28	X	-	2,78	2,15	3,57	2,88	3,58	3,05	11,13	3,62
LIFT-SLAM[2] [49]	t_{rel}	%	3,49	X	9,84	0,86	2,22	5,35	7,05	2,6	28,99	19,16	9,81
	r_{rel}	$[deg/m]$	2,63	X	2,1	0,46	0,5	1,91	2,36	3,64	1,95	2,08	2,2
Stereo LSD-SLAM [137]	t_{rel}	%	**0,63**	2,36	0,79	1,01	0,38	0,64	0,71	0,56	1,11	1,14	0,72
	r_{rel}	$[deg/m]$	0,26	0,36	0,23	0,28	0,31	0,18	0,18	0,29	0,31	0,25	0,33
ORB-SLAM 2[3] [35]	t_{rel}	%	0,7	1,39	**0,76**	0,71	0,48	**0,4**	0,51	0,5	1,05	0,87	0,6
	r_{rel}	$[deg/m]$	0,25	0,21	0,23	0,18	0,13	0,16	0,15	0,28	0,32	0,27	0,27
MVL-SLAM[3] [175]	t_{rel}	%	2,53	3,76	3,95	2,75	1,81	3,49	1,84	3,27	2,75	3,7	4,65
	r_{rel}	$[deg/m]$	0,79	0,80	1,05	1,39	1,48	0,79	0,83	1,51	1,61	1,83	0,51

* Best results for each sequence in **Bold**.
The symbol '-' stands for *no results reported* and 'X' for tracking failure.
1 Data extracted from [146].
2 Results correspond to the *LIFT-SLAM Fine tuned With EuRoC* version of the algorithm.
3 Data extracted from [49].

TABLE V
ABSOLUTE TRAJECTORY ERROR (RMSE) FOR EUROC DATASET SEQUENCES*

System	V1_01	V1_02	V1_03	V2_01	V2_02	V2_03	MH_01	MH_02	MH_03	MH_04	MH_05
LoG-SLAM [167]	0,031	0,018	0,031	-	-	-	0,041	0,026	0,210	0,087	0,057
D3VO [148]	-	-	0,110	-	0,050	0,190	-	-	0,080	-	0,090
LIFT-SLAM[1] [49]	0,100	-	0,370	-	-	-	0,117	0,062	0,053	-	-
ORB-SLAM 2 [35]	0,035	0,020	0,048	0,037	0,035	-	0,035	0,018	0,028	0,119	0,060
LSD-SLAM [145]	0,066	0,074	0,089	-	-	-	-	-	-	-	-
LCSD-SLAM [163]	0,099	0,111	0,825	0,114	0,191	0,238	0,039	0,036	0,045	0,074	0,060
Kimera (RPGO) [110]	0,050	0,110	0,120	0,070	0,100	0,190	0,080	0,090	0,110	0,150	0,240
SVL (Visual) [162]	0,097	0,108	-	-	-	-	0,046	0,045	-	-	-
Luo et al. [41]	0,036	0,035	0,139	**0,026**	0,036	-	**0,018**	**0,015**	0,039	0,093	**0,054**
OKVIS [48]	0,090	0,200	0,240	0,130	0,160	0,290	0,160	0,220	0,240	0,340	0,470
VINS-Mono [114]	0,068	0,084	0,190	0,081	0,160	0,220	0,120	0,120	0,130	0,180	0,210
ORB-SLAM Atlas [38]	0,036	0,022	0,051	0,034	0,028	0,218	0,036	0,021	**0,026**	0,103	**0,054**
VI-SLAM [34]	**0,027**	0,028	X	0,032	0,041	0,074	0,075	0,084	0,087	**0,022**	0,082
ORB-SLAM 3 (Stereo+IMU) [39]	0,038	**0,014**	**0,024**	0,032	**0,014**	**0,024**	0,036	0,033	0,035	0,051	0,082
VINS-Fusion (Mono+IMU) [115]	0,060	0,090	0,180	0,060	0,110	0,260	0,180	0,090	0,170	0,210	0,250
ROVIO [111]	0,100	0,100	0,140	0,120	0,140	0,140	0,210	0,250	0,250	0,190	0,520
SVO(Edgelets+prior) [161]	0,040	0,040	0,070	0,050	0,090	0,790	0,040	0,070	0,270	0,170	0,120
VI-DSO [112]	0,059	0,067	0,096	0,040	0,062	0,174	0,062	0,044	0,117	0,132	0,121
HVIOnet [176]	0,110	0,087	0,190	0,053	0,183	0,183	-	-	-	-	-

* Best results for each sequence in **Bold**.
The symbol '-' stands for *no results reported* and 'X' for tracking failure.
1 Results correspond to the *LIFT-SLAM Fine tuned With EuRoC* version of the algorithm.

that the system can handle. As a result, the accuracy and timing requirements of the system must be modified. In [133], multiple systems were tried on different hardware platforms to further assess this trade-off idea.

Finally, it is worth mentioning the recent work by Bujanca et al. called SLAMBench [231], [232]. It is an SLAM benchmarking platform that allows users to quantify quality-of-result with instrumentation of accuracy, execution time, memory usage and energy consumption for several state-of-the-art systems such as ORB-SLAM2, ORB-SLAM3 DSO,SVO,etc. It also includes a graphical interface to visualize this information, and runs on desktop, laptop, mobile and embedded platforms.

VII. CURRENT TRENDS, POSSIBLE FUTURE DIRECTIONS AND DISCUSSION

This section starts by introducing a discussion: Will SLAM always remain an open problem?. Afterward, we briefly delve into the current trends in SLAM advancements and assess prospective approaches to address the SLAM problem.

A. Discussion: Will SLAM always remain an open problem?

Initially, the fundamental question researchers in SLAM grappled with was: Is there a way to solve the localization and mapping of a mobile robot? A thorough exploration of this topic will reveal to the reader that current research articles increasingly focus on developing highly specific navigation systems that cater to particular needs (e.g., dark environments, underwater, high-speed, in aerial vehicles, etc.), where the functioning of localization and mapping is adapted to these circumstances, rather than presenting a novel solution.

Take mapping, for instance. There are dense map representations, sparse representations, grid maps, and landmarks. Each of these is used interchangeably by researchers based on the resources they have or the objectives they pursue. For some, detail might be crucial (e.g., in disaster zones), while for others, a coarse map suffices (e.g., underwater). Seen in this light, one might conclude that the SLAM problem has already been widely solved, and there is nothing novel to contribute. It suffices to intelligently use the available developments.

On the contrary, we propose that the reader consider that current systems aim to address the following question: Is it possible to create a SLAM system that adheres to specific design specifications, is computationally efficient, and consumes minimal energy? This question has as many answers as there are challenges for researchers. Viewed this way, the SLAM problem will always remain open, as humans continuously face new challenges.

B. The AI Era.

The integration of neural networks has become an enduring trend in recent years. Researchers are consistently exploring innovative ways to leverage neural networks to replace or enhance tasks within the SLAM pipeline [174], [184], [189], [233]. A notable focus has been on evolving neural network training techniques. While it's challenging to encompass all of them in detail, prevalent techniques frequently discussed in the context of SLAM include adversarial training [234], unsupervised learning [181], [199], reinforcement learning [235], [236], and transfer learning [49], [206].

The opening statement in Sec I states "Robots must always be able to determine their location". It would be interesting to complete that sentence by adding "while also adapting and learning in the process". A potential avenue for future exploration could involve systems with continuous learning. Pioneering works such as [210] and [237] have taken initial strides toward achieving this objective, charting a path for the development of more intelligent systems.

The revolutionary impact of artificial intelligence on our day-to-day lives, utilizing tools such as ChatGPT and Bard, brings to light a wealth of possibilities for conceiving navigation systems that autonomously respond to their environments.

C. Collaboration over Combination

Both image processing approaches presented in Sec. IV-B, have their drawbacks. Indirect methods struggle to extract features in textureless environments, being unable to compute the camera pose in such scenarios. This limitation directly affects its trajectory tracking capability, since geometric error calculations are highly dependent on the number of features extracted, and whether they can be tracked in the subsequent frame.

On the other hand, direct methods face challenges in cases of sudden changes in illumination, leading to a loss of tracking. These methods rely on pixel brightness values for pose calculations, and if the pixel values change abruptly, consecutive images cannot be accurately aligned. Therefore, the pose estimation algorithm fails, and tracking is considered lost.

Hybrid image processing approaches that aim to take advantage of both direct and indirect methods can also be found in the literature [160], [161]. The general concept of a hybrid (or semi- direct) system is to take one of the main image processing methodologies and insert it into the other: e.g. to utilize the robustness of feature matching to support direct image alignment algorithms [41], [162], [163]. The goal is to combine the strengths of both approaches to mitigate their respective limitations. However, hybrid approaches do not outperform the main methods separately [162].

The foregoing paragraphs reveal that the fusion of direct and indirect algorithms might not be a definitive solution to the issue of tracking loss. To address the limitations of current systems, a fresh paradigm for tackling the SLAM problem is needed. An alternative approach could involve the collaboration (e.g. parallel execution) of the two main image processing algorithms under the same system, instead of blending them. In this way, the advantages of both systems would be exploited while minimizing their disadvantages. This way, the strengths of both systems can be harnessed while mitigating their respective drawbacks.

VIII. Conclusions

The literature on SLAM can be organized in a straightforward and general manner. Several classifications can be established to explain the overall taxonomy of this field of research, taking into account the most frequent elements that constitute existing systems.

The overwhelming amount of systems in the literature makes it difficult to understand how a SLAM system is actually composed. We proposed a comprehensive description of the contemporary architecture of a SLAM algorithm. Each fundamental component was elucidated, along with insights into their interactions within the system.

Artificial intelligence is a clear trend in this field of research, either by substituting certain building blocks or replicating entire systems using neural networks. We conducted an in-depth analysis of how AI techniques are integrating with SLAM and their implications for current navigation systems.

Furthermore, we devised several comparative tables to facilitate an unbiased assessment and comparison of the accuracy achieved by various existing SLAM advancements. Finally, we explored current trends and future prospects in the field, engaging in a discussion on whether SLAM remains an ongoing challenge.

Appendix

This section contains a summary of symbols used in this paper (Table VI), as well as a list of acronyms and abbreviations (Table VII).

TABLE VI
SYMBOLS SUMMARY

Text reference	Symbol	Description
Photometric error optimization	E_{photo}	Photometric error for direct methods to minimize.
	$\mathcal{F}$	Set of frames considered for direct image alignment.
	$\mathcal{P}_i$	Set of points in frame $\mathcal{F}$ considered for direct image alignment.
	$obs(\mathbf{p})$	Set of observed frames in which a point $\mathbf{p}$ is visible.
	$E_{\mathbf{p},j}$	Weighted photometric error of a point $\mathbf{p}$ in keyframe I_i observed from a target frame I_j.
	$\mathcal{N}_p$	Local neighborhood of pixels around a point $\mathbf{p}$.
	$\Vert\cdot\Vert_\gamma$	Huber norm.
	$\mathbf{p}'$	Projected position of a point $\mathbf{p}$.
	t_i , t_j	Exposure times for images I_i and I_j respectively.
	a_i,b_i,a_j,b_J	Brightness transfer function parameters of a camera.
	$w_\mathbf{p}$	Gradient-dependent weight.
	e	Number e.
Geometric error optimization	$\mathbf{R}$	Camera rotation matrix.
	$\mathbf{t}$	Camera translation vector.
	$\mathbb{R}^3$	Euclidean 3D space.
	$SO(3)$	Group of all rotations about the origin of three-dimensional Euclidean space $\mathbb{R}^3$ under the operation of composition.
	$x^i_{(\cdot)}$	Image keypoints for feature matching.
	$\mathbf{X}^i \in \mathbb{R}^3$	Matched keypoints between images in world coordinates.
	$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$	Matched keypoint world coordinates.
	$\mathcal{X}$	Set of all matched keypoints.
	ρ	Huber cost function.
	Σ	Covariance matrix associated to the scale of a keypoint.
	π_m,π_s	Camera projection function for monocular and stereo cameras respectively.
	f_x,f_y	Camera focal length in x and y directions.
	c_x,c_y	Camera principal point in x and y directions.
	b	Baseline for stereo cameras.
Artifficial intelligence cost functions	$\mathbf{p} = [\mathbf{x}, \mathbf{q}]$	Output camera pose for PoseNet.
	$\mathbf{x}$	3D translation vector.
	$\mathbf{q}$	Rotation vector represented by quaternions.
	$\mathcal{L}_x$	Translation related term for pose estimation.
	$\mathcal{L}_q$	Rotation related term for pose estimation.
	L	Euclidean loss function for pose estimation.
	β	Scale factor used in the Euclidean loss function L.
	$\hat{s}_x$,$\hat{s}_q$	Learnable parameters for translation and rotation to replace β in L.
Relative pose error calculation	$SE(3)$	Special Euclidean Group in 3 dimensions. Is the group of simultaneous rotations and translations for a vector.
	$\mathbf{P}_1, ..., \mathbf{P}_n \in SO(3)$	Sequence of poses from an estimated trajectory.
	$\mathbf{Q}_1, ..., \mathbf{Q}_n \in SO(3)$	Ground truth trajectory poses.
	$\mathbf{E}_i$	Relative pose error at time step i
	$trans(\mathbf{E}_i)$ or t_{rel}	Translational component of $\mathbf{E}_i$.
	$rot(\mathbf{E}_i)$ or r_{rel}	Rotational component of $\mathbf{E}_i$.
	n	Quantity of camera poses considered for the error calculations.
	Δ	Fixed time interval.
	$m = n - \Delta$	Quantity of individual relative pose errors that can be obtained from n camera poses
Absolute trajectory error calculation	$\mathbf{S}_{1:n}$	Estimated trajectory.
	$\mathbf{Q}_{1:n}$	Ground truth trajectory
	$\mathbf{P}$	Rigid body transformation between the estimated trajectory and the ground truth data.
	$trans(\mathbf{ATE})$	Translational component of the absolute trajectory error.

TABLE VII
LIST OF ACRONYMS AND ABBREVIATIONS

Acronym / Abbreviation	Description
A-SLAM	Active SLAM.
AI	Artificial Intelligence.
ATE	Absolute Trajectory Error.
BA	Bundle Adjustment.
BoW	Bag of Words.
BRIEF	Binary Robust Independent Elementary Features.
BRISK	Binary Robust Invariant Scalable Keypoints.
CNN	Convolutional Neural Networks.
CNN-SVO	CNN Sparse Visual Odometry.
D3VO	Deep depth, Deep pose and Deep uncertainty for monocular Visual Odometry.
Deep	When used as part of a name, refers to deep learning based systems.
DeepLo	Deep LiDAR Odometry.
DM-SLAM	SLAM Method for rigid Dynamic scenes.
DoF	Degrees of Freedom.
DS-SLAM	Semantic visual SLAM towards Dynamic environments.
EuRoC MAV	European Robotics Challenge Micro-Aerial Vehicle.
FAST	Features from Accelerated Segment Test.
FCN	Fully Convolutional Network.
GCN	Geometric Correspondence Network.
GCNv2	Geometric Correspondence Network version 2.
HVIOnet	Hybrid Visual Inertial Odometry network
IAICP	Intensity Assisted Iterative Closest Point.
ICP	Iterative Closest Point.
IMU	Inertial Measurement Unit.
INS	Inertial Navigation Systems.
IO	Inertial Odometry.
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute.
L^3-net	Learning based LiDAR Localization.
LCD	Loop Closure Detection.
LiDAR	Light Detection and Ranging.
LiDAR SLAM	LiDAR based SLAM algorithms.
LIO-SAM	LiDAR Inertial Odometry via Smoothing And Mapping.
LIPS SLAM	LiDAR-Inertial 3D Plane SLAM.
LiTAMIN	LiDAR-based Tracking And MappINg.
LOAM	LiDAR Odometry and Mapping.
LoG	Laplacian of a Gaussian operator.
LSD SLAM	Large-Scale Direct SLAM.
LSTM	Long Short-Term Memory neural networks.
Mask R-CNN	Mask Region-based CNN.
MVL-SLAM	Multi-channel Visual-LiDAR SLAM.
NN	Neural Network.
OpenVSLAM	Open Visual SLAM.
ORB	Oriented FAST and Rotated BRIEF.
ORB-SLAM	Oriented FAST and Rotated BRIEF SLAM.
PDR	Pedestrian Dead Reckoning.
PLD-SLAM	Point Line Dynamic SLAM.
PTAM	Parallel Tracking And Mapping.
RANSAC	Random Sample Consensus.
RNN	Recurrent Neural Networks.
ROS	Robot Operating System.
RPE	Relative Pose Error.
RTAB-MAP	Real-Time Appearance-Based MAPping.
SelfVIO	Self-supervised deep learning-based VIO.
SfM	Structure from Motion.
SfMLearner	Structure from Motion Learner.
SIFT	Scale-Invariant Feature Transform.
SINS	Strapdown Inertial Navigation Systems.
SLAM	Simultaneous Localization and Mapping.
SPHORB	SPHerical ORB.
SURF	Speeded Up Robust Features.
Un	When used as part of a name (e.g. UnDeepVO), refers to unsupervised learning.
UnDeepVO	Visual Odometry through Unsupervised Deep learning.
V-LOAM	Visual-LiDAR Odometry and Mapping.
V-SLAM	Visual SLAM.
VI	Visual inertial.
VI-SLAM	Visual inertial SLAM.
VINS-MONO	MONOcular Visual-INertial System.
VIO	Visual inertial odometry.
VIIONet	Visual-Inertial Image-Odometry Network.
VLocNet	Visual Localization Network.
VO	Visual Odometry.

FUNDING

This research did not receive additional support from any organization beyond the authors' academic institutions

CONFLICT OF INTERESTS

The authors have no relevant financial or non-financial interests to disclose.

REFERENCES

[1] I. Lluvia, E. Lazkano, and A. Ansuategi, "Active mapping and robot exploration: A survey," *Sensors*, vol. 21, no. 7, p. 2445, 2021.

[2] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *The international journal of Robotics Research*, vol. 5, no. 4, pp. 56–68, 1986.

[3] H. F. Durrant-Whyte, "Uncertain geometry in robotics," *IEEE Journal on Robotics and Automation*, vol. 4, no. 1, pp. 23–31, 1988.

[4] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[5] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part ii," *IEEE robotics & automation magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[6] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad, "An overview to visual odometry and visual slam: Applications to mobile robotics," *Intelligent Industrial Systems*, vol. 1, no. 4, pp. 289–311, 2015.

[7] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," *Technical report, Ver. 2.3 EBSE Technical Report. EBSE*, 2007.

[8] S. Huang and G. Dissanayake, "A critique of current developments in simultaneous localization and mapping," *International Journal of Advanced Robotic Systems*, vol. 13, no. 5, p. 1729881416669482, 2016.

[9] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[10] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii: Matching, robustness, optimization, and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[11] S. Saeedi, M. Trentini, M. Seto, and H. Li, "Multiple-robot simultaneous localization and mapping: A review," *Journal of Field Robotics*, vol. 33, no. 1, pp. 3–46, 2016.

[12] B. Huang, J. Zhao, and J. Liu, "A survey of simultaneous localization and mapping with an envision in 6g wireless networks," *arXiv preprint arXiv:1909.05214*, 2019.

[13] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: A survey of current trends in autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017.

[14] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual slam and structure from motion in dynamic environments: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 2, pp. 1–36, 2018.

[15] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, "A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence," *arXiv preprint arXiv:2006.12567*, 2020.

[16] B. Yamauchi, "A frontier-based approach for autonomous exploration," *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, pp. 146–151, 1997.

[17] B. Yamauchi, A. Schultz, and W. Adams, "Mobile robot exploration and map-building with continuous localization," *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, vol. 4, pp. 3715–3720, 1998.

[18] C. Zhu, R. Ding, M. Lin, and Y. Wu, "A 3d frontier-based exploration tool for mavs," *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 348–352, 2015.

[19] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast frontier-based information-driven autonomous exploration with an mav," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9570–9576, 2020.

[20] H. H. González-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 829–848, 2002.

[21] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun, "Collaborative multi-robot exploration," *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1, pp. 476–481, 2000.

[22] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," *Proceedings 2002 IEEE international conference on robotics and automation (Cat. No. 02CH37292)*, vol. 3, pp. 3016–3023, 2002.

[23] A. A. Makarenko, S. B. Williams, F. Bourgault, and H. F. Durrant-Whyte, "An experiment in integrated exploration," *IEEE/RSJ international conference on intelligent robots and systems*, vol. 1, pp. 534–539, 2002.

[24] M. Juliá, A. Gil, and O. Reinoso, "A comparison of path planning strategies for autonomous exploration and mapping of unknown environments," *Autonomous Robots*, vol. 33, no. 4, pp. 427–444, 2012.

[25] R. Sim and N. Roy, "Global a-optimal robot exploration in slam," *Proceedings of the 2005 IEEE international conference on robotics and automation*, pp. 661–666, 2005.

[26] J. Vallvé and J. Andrade-Cetto, "Active pose slam with rrt," *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2167–2173, 2015.

[27] C. Leung, S. Huang, and G. Dissanayake, "Active slam using model predictive control and attractor based exploration," *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5031, 2006.

[28] I. Maurović, M. Seder, K. Lenac, and I. Petrović, "Path planning for active slam based on the d* algorithm with negative edge weights," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 8, pp. 1321–1331, 2017.

[29] E. Bonetto, P. Goldschmid, M. Pabst, M. J. Black, and A. Ahmad, "irotate: Active visual slam for omnidirectional robots," *Robotics and Autonomous Systems*, p. 104102, 2022.

[30] H. Carrillo, P. Dames, V. Kumar, and J. A. Castellanos, "Autonomous robotic exploration using a utility function based on rényi's general theory of entropy," *Autonomous Robots*, vol. 42, no. 2, pp. 235–256, 2018.

[31] Y. Chen, S. Huang, and R. Fitch, "Active slam for mobile robots with area coverage and obstacle avoidance," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 3, pp. 1182–1192, 2020.

[32] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[33] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[34] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[35] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[36] F. Nobis, O. Papanikolaou, J. Betz, and M. Lienkamp, "Persistent map saving for visual localization for autonomous vehicles: An orb-slam 2 extension," *2020 Fifteenth International Conference on Ecological Vehicles and Renewable Energies (EVER)*, pp. 1–9, 2020.

[37] D. Schlegel, M. Colosi, and G. Grisetti, "Proslam: graph slam from a programmer's perspective," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3833–3840, 2018.

[38] R. Elvira, J. D. Tardós, and J. M. Montiel, "Orbslam-atlas: a robust and accurate multi-map system," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6253–6259, 2019.

[39] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, 2021.

[40] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[41] H. Luo, C. Pape, and E. Reithmeier, "Hybrid monocular slam using double window optimization," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4899–4906, 2021.

[42] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[43] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," *2011*

IEEE international symposium on safety, security, and rescue robotics, pp. 155–160, 2011.

[44] H. Carrillo, P. Dames, V. Kumar, and J. A. Castellanos, "Autonomous robotic exploration using occupancy grid maps and graph slam based on shannon and rényi entropy," *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 487–494, 2015.

[45] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," *Aaai/iaai*, vol. 593598, 2002.

[46] J. Zhang, W. Sui, X. Wang, W. Meng, H. Zhu, and Q. Zhang, "Deep online correction for monocular visual odometry," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14396–14402, 2021.

[47] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "Codeslam—learning a compact, optimisable representation for dense visual slam," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2560–2568, 2018.

[48] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[49] H. M. S. Bruno and E. L. Colombini, "Lift-slam: a deep-learning feature-based monocular visual slam method," *Neurocomputing*, vol. 455, pp. 97–110, 2021.

[50] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," *International workshop on vision algorithms*, pp. 298–372, 1999.

[51] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2015.

[52] Y. Latif, G. Huang, J. J. Leonard, and J. Neira, "An online sparsity-cognizant loop-closure algorithm for visual navigation.," *Robotics: Science and Systems*, 2014.

[53] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," *Workshop on statistical learning in computer vision, ECCV*, vol. 1, no. 1-22, pp. 1–2, 2004.

[54] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[55] M. Labbe and F. Michaud, "Appearance-based loop closure detection for online large-scale and long-term operation," *IEEE Transactions on Robotics*, vol. 29, no. 3, pp. 734–745, 2013.

[56] D. Filliat, "A visual bag of words method for interactive qualitative localization and mapping," *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3921–3926, 2007.

[57] D. Schlegel and G. Grisetti, "Visual localization and loop closing using decision trees and binary features," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4616–4623, 2016.

[58] J. Chen, J. Li, Y. Xu, G. Shen, and Y. Gao, "A compact loop closure detection based on spatial partitioning," *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, pp. 371–375, 2017.

[59] Z. Yang, Y. Pan, L. Deng, Y. Xie, and R. Huan, "Plsav: Parallel loop searching and verifying for loop closure detection," *IET Intelligent Transport Systems*, vol. 15, no. 5, pp. 683–698, 2021.

[60] N. Kejriwal, S. Kumar, and T. Shibata, "High performance loop closure detection using bag of word pairs," *Robotics and Autonomous Systems*, vol. 77, pp. 55–65, 2016.

[61] H. Zhang, F. Han, and H. Wang, "Robust multimodal sequence-based loop closure detection via structured sparsity.," *Robotics: Science and systems*, 2016.

[62] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "Litamin: Lidar-based tracking and mapping by stabilized icp for geometry approximation with normal distributions," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5143–5150, 2020.

[63] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "Litamin2: Ultra light lidar-based slam using geometric approximation applied with kl-divergence," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11619–11625, 2021.

[64] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments.," *Robotics: Science and Systems*, vol. 2018, p. 59, 2018.

[65] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time.," *Robotics: Science and Systems*, vol. 2, no. 9, pp. 1–9, 2014.

[66] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4758–4765, 2018.

[67] W. Ali, P. Liu, R. Ying, and Z. Gong, "6-dof feature based lidar slam using orb features from rasterized images of 3d lidar point cloud," *arXiv preprint arXiv:2103.10678*, 2021.

[68] M. Valente, C. Joly, and A. de La Fortelle, "An lstm network for real-time odometry estimation," *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1434–1440, 2019.

[69] L. Sun, D. Adolfsson, M. Magnusson, H. Andreasson, I. Posner, and T. Duckett, "Localising faster: Efficient and precise lidar-based robot localisation in large-scale environments," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4386–4392, 2020.

[70] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[71] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2531–2538, 2008.

[72] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[73] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pp. 225–234, 2007.

[74] R. Munoz-Salinas and R. Medina-Carnicer, "Ucoslam: Simultaneous localization and mapping by fusion of keypoints and squared planar markers," *Pattern Recognition*, vol. 101, p. 107193, 2020.

[75] R. Munoz-Salinas, M. J. Marin-Jimenez, and R. Medina-Carnicer, "Spm-slam: Simultaneous localization and mapping with squared planar markers," *Pattern Recognition*, vol. 86, pp. 156–171, 2019.

[76] B. Pfrommer and K. Daniilidis, "Tagslam: Robust slam with fiducial markers," *arXiv preprint arXiv:1910.00679*, 2019.

[77] J. Wang and E. Olson, "Apriltag 2: Efficient and robust fiducial detection," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4193–4198, 2016.

[78] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: A versatile visual slam framework," *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 2292–2295, 2019.

[79] P. G. Savage, "Strapdown inertial navigation integration algorithm design part 1: Attitude algorithms," *Journal of guidance, control, and dynamics*, vol. 21, no. 1, pp. 19–28, 1998.

[80] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication*, pp. 27–35, 2006.

[81] Y. Wu, X. Hu, D. Hu, T. Li, and J. Lian, "Strapdown inertial navigation system algorithms based on dual quaternions," *IEEE transactions on aerospace and electronic systems*, vol. 41, no. 1, pp. 110–132, 2005.

[82] W. Sun, D. Wang, L. Xu, and L. Xu, "Mems-based rotary strapdown inertial navigation system," *Measurement*, vol. 46, no. 8, pp. 2585–2596, 2013.

[83] S. Cortés, A. Solin, and J. Kannala, "Deep learning based speed estimation for constraining strapdown inertial navigation on smartphones," *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2018.

[84] M. Brossard, A. Barrau, and S. Bonnabel, "Rins-w: Robust inertial navigation system on wheels," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2068–2075, 2019.

[85] H. Yan, Q. Shan, and Y. Furukawa, "Ridi: Robust imu double integration," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 621–636, 2018.

[86] M. Ren, K. Pan, Y. Liu, H. Guo, X. Zhang, and P. Wang, "A novel pedestrian navigation algorithm for a foot-mounted inertial-sensor-based system," *Sensors*, vol. 16, no. 1, p. 139, 2016.

[87] B. Wagstaff and J. Kelly, "Lstm-based zero-velocity detection for robust inertial navigation," *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–8, 2018.

[88] C. Chen, Y. Miao, C. X. Lu, L. Xie, P. Blunsom, A. Markham, and N. Trigoni, "Motiontransformer: Transferring neural inertial tracking between domains," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 8009–8016, 2019.

[89] C. Chen, X. Lu, A. Markham, and N. Trigoni, "Ionet: Learning to cure the curse of drift in inertial odometry," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[90] M. A. Esfahani, H. Wang, K. Wu, and S. Yuan, "Aboldeepio: A novel deep inertial odometry network for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 5, pp. 1941–1950, 2019.

[91] J. Ku, A. Harakeh, and S. L. Waslander, "In defense of classical image processing: Fast depth completion on the cpu," *2018 15th Conference on Computer and Robot Vision (CRV)*, pp. 16–22, 2018.

[92] F. Ma and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 4796–4803, 2018.

[93] S. S. Shivakumar, T. Nguyen, I. D. Miller, S. W. Chen, V. Kumar, and C. J. Taylor, "Dfusenet: Deep fusion of rgb and sparse depth information for image guided dense depth completion," *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 13–20, 2019.

[94] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: Low-drift, robust, and fast," *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2174–2181, 2015.

[95] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[96] S. Li and D. Lee, "Rgb-d slam in dynamic environments using static point weighting," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2263–2270, 2017.

[97] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving rgb-d slam in dynamic environments: A motion removal approach," *Robotics and Autonomous Systems*, vol. 89, pp. 110–122, 2017.

[98] Y. Wang and S. Huang, "Motion segmentation based robust rgb-d slam," *Proceeding of the 11th World Congress on Intelligent Control and Automation*, pp. 3122–3127, 2014.

[99] Y. Xu, Y. Ou, and T. Xu, "Slam of robot based on the fusion of vision and lidar," *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, pp. 121–126, 2018.

[100] D. Xu, D. Anguelov, and A. Jain, "Pointfusion: Deep sensor fusion for 3d bounding box estimation," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 244–253, 2018.

[101] K. Shin, Y. P. Kwon, and M. Tomizuka, "Roarnet: A robust 3d object detection based on region approximation refinement," *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2510–2515, 2019.

[102] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, 2018.

[103] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1907–1915, 2017.

[104] J. Levinson and S. Thrun, "Automatic online calibration of cameras and lasers.," *Robotics: Science and Systems*, vol. 2, p. 7, 2013.

[105] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, "Lidar-camera calibration using 3d-3d point correspondences," *arXiv preprint arXiv:1705.09785*, 2017.

[106] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3565–3572, 2007.

[107] D. Zou, Y. Wu, L. Pei, H. Ling, and W. Yu, "Structvio: visual-inertial odometry with structural regularity of man-made environments," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 999–1013, 2019.

[108] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, 2019.

[109] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[110] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1689–1696, 2020.

[111] 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)*, pp. 298–304, 2015.

[112] L. Von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2510–2517, 2018.

[113] V. Usenko, J. Engel, J. Stückler, and D. Cremers, "Direct visual-inertial odometry with stereo cameras," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1885–1892, 2016.

[114] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[115] T. Qin, J. Pan, S. Cao, and S. Shen, "A general optimization-based framework for local odometry estimation with multiple sensors," *arXiv preprint arXiv:1901.03638*, 2019.

[116] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart, "maplab: An open framework for research in visual-inertial mapping and localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1418–1425, 2018.

[117] C. Campos, J. M. Montiel, and J. D. Tardós, "Inertial-only optimization for visual-inertial initialization," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 51–57, 2020.

[118] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5135–5142, 2020.

[119] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, "Lips: Lidar-inertial 3d plane slam," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 123–130, 2018.

[120] F. Neuhaus, T. Koß, R. Kohnen, and D. Paulus, "Mc2slam: Real-time inertial lidar odometry using two-scan motion compensation," *German Conference on Pattern Recognition*, pp. 60–72, 2018.

[121] R. Opromolla, G. Fasano, G. Rufino, M. Grassi, and A. Savvaris, "Lidar-inertial integration for uav localization and mapping in complex environments," *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 649–656, 2016.

[122] B. Bescos, J. Neira, R. Siegwart, and C. Cadena, "Empty cities: Image inpainting for a dynamic-object-invariant space," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5460–5466, 2019.

[123] B. Bescos, C. Cadena, and J. Neira, "Empty cities: A dynamic-object-invariant space for visual slam," *IEEE Transactions on Robotics*, 2020.

[124] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[125] J. Cheng, Z. Wang, H. Zhou, L. Li, and J. Yao, "Dm-slam: A feature-based slam system for rigid dynamic scenes," *ISPRS International Journal of Geo-Information*, vol. 9, no. 4, p. 202, 2020.

[126] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1168–1174, 2018.

[127] G. Liu, W. Zeng, B. Feng, and F. Xu, "Dms-slam: A general visual slam system for dynamic scenes with multiple sensors," *Sensors*, vol. 19, no. 17, p. 3714, 2019.

[128] D. Esparza and G. Flores, "The stdyn-slam: a stereo vision and semantic segmentation approach for vslam in dynamic outdoor environments," *IEEE Access*, vol. 10, pp. 18201–18209, 2022.

[129] C. Zhang, T. Huang, R. Zhang, and X. Yi, "Pld-slam: A new rgb-d slam method with point and line features for indoor dynamic scene," *ISPRS International Journal of Geo-Information*, vol. 10, no. 3, p. 163, 2021.

[130] J. Cheng, Y. Sun, and M. Q.-H. Meng, "Improving monocular visual slam in dynamic environments: an optical-flow-based approach," *Advanced Robotics*, vol. 33, no. 12, pp. 576–589, 2019.

[131] N. Yu, M. Gan, H. Yu, and K. Yang, "Drso-slam: A dynamic rgb-d slam algorithm for indoor dynamic scenes," in *2021 33rd Chinese Control and Decision Conference (CCDC)*, pp. 1052–1058, IEEE, 2021.

[132] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Monocular vision based slam for mobile robots," *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3, pp. 1027–1031, 2006.

[133] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2502–2509, 2018.

[134] N. Yang, R. Wang, J. Stuckler, and D. Cremers, "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 817–833, 2018.

[135] P. Schmuck and M. Chli, "Ccm-slam: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams," *Journal of Field Robotics*, vol. 36, no. 4, pp. 763–781, 2019.

[136] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "A constant-time efficient stereo slam system," *Proceedings of the British machine vision conference*, vol. 1, no. 2009, 2009.

[137] J. Engel, J. Stückler, and D. Cremers, "Large-scale direct slam with stereo cameras," *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1935–1942, 2015.

[138] G.-x. Xin, X.-t. Zhang, X. Wang, and J. Song, "A rgbd slam algorithm combining orb with prosac for indoor mobile robot," in *2015 4th International Conference on Computer Science and Network Technology (ICCSNT)*, vol. 1, pp. 71–74, IEEE, 2015.

[139] Q. Li, X. Wang, T. Wu, and H. Yang, "Point-line feature fusion based field real-time rgb-d slam," *Computers & Graphics*, vol. 107, pp. 10–19, 2022.

[140] Y. Wang, K. Xu, Y. Tian, and X. Ding, "Drg-slam: A semantic rgb-d slam using geometric features for indoor dynamic scene," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1352–1359, IEEE, 2022.

[141] Y. Liu, M. Xu, G. Jiang, X. Tong, J. Yun, Y. Liu, B. Chen, Y. Cao, N. Sun, and Z. Li, "Target localization in local dense mapping using rgbd slam and object detection," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 4, p. e6655, 2022.

[142] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, *et al.*, "Event-based vision: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 1, pp. 154–180, 2020.

[143] W. Chamorro, J. Solà, and J. Andrade-Cetto, "Event-based line slam in real-time," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8146–8153, 2022.

[144] A. G. Gelen and A. Atasoy, "An artificial neural slam framework for event-based vision," *IEEE Access*, 2023.

[145] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," *European conference on computer vision*, pp. 834–849, 2014.

[146] R. Wang, M. Schworer, and D. Cremers, "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3903–3911, 2017.

[147] X. Gao, R. Wang, N. Demmel, and D. Cremers, "Ldso: Direct sparse odometry with loop closure," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2198–2204, 2018.

[148] N. Yang, L. v. Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1281–1292, 2020.

[149] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[150] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *European conference on computer vision*, pp. 404–417, 2006.

[151] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," *2011 International conference on computer vision*, pp. 2564–2571, 2011.

[152] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," *European conference on computer vision*, pp. 778–792, 2010.

[153] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *European conference on computer vision*, pp. 430–443, 2006.

[154] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," *2011 International conference on computer vision*, pp. 2548–2555, 2011.

[155] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 224–236, 2018.

[156] J. Tang, J. Folkesson, and P. Jensfelt, "Geometric correspondence network for camera motion estimation," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1010–1017, 2018.

[157] J. Tang, L. Ericson, J. Folkesson, and P. Jensfelt, "Gcnv2: Efficient correspondence prediction for real-time slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3505–3512, 2019.

[158] J. Li, X. Wang, and S. Li, "Spherical-model-based slam on full-view images for indoor environments," *Applied Sciences*, vol. 8, no. 11, p. 2268, 2018.

[159] T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera, and J. J. Berlles, "S-ptam: Stereo parallel tracking and mapping," *Robotics and Autonomous Systems*, vol. 93, pp. 27–42, 2017.

[160] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," *2014 IEEE international conference on robotics and automation (ICRA)*, pp. 15–22, 2014.

[161] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[162] S.-p. Li, T. Zhang, X. Gao, D. Wang, and Y. Xian, "Semi-direct monocular visual and visual-inertial slam with loop closure detection," *Robotics and Autonomous Systems*, vol. 112, pp. 201–210, 2019.

[163] S. H. Lee and J. Civera, "Loosely-coupled semi-direct monocular slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 399–406, 2018.

[164] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," *European conference on computer vision*, pp. 467–483, 2016.

[165] D. Li, X. Shi, Q. Long, S. Liu, W. Yang, F. Wang, Q. Wei, and F. Qiao, "Dxslam: A robust and efficient visual slam system with deep features," in *2020 IEEE/RSJ International conference on intelligent robots and systems (IROS)*, pp. 4958–4965, IEEE, 2020.

[166] T. Weyand, I. Kostrikov, and J. Philbin, "Planet-photo geolocation with convolutional neural networks," *European Conference on Computer Vision*, pp. 37–55, 2016.

[167] G. Zhang, X. Yan, Y. Xu, and Y. Ye, "Neural guided visual slam system with laplacian of gaussian operator," *IET Computer Vision*, vol. 15, no. 3, pp. 181–196, 2021.

[168] B. Dongdong, W. Chaoqun, B. Zhang, Y. Xiaodong, Y. Xuejun, *et al.*, "Cnn feature boosted seqslam for real-time loop closure detection," *Chinese Journal of Electronics*, vol. 27, no. 3, pp. 488–499, 2018.

[169] A. R. Memon, H. Wang, and A. Hussain, "Loop closure detection using supervised and unsupervised deep neural networks for monocular slam systems," *Robotics and Autonomous Systems*, vol. 126, p. 103470, 2020.

[170] X. Gao and T. Zhang, "Unsupervised learning to detect loops using deep neural networks for visual slam system," *Autonomous robots*, vol. 41, pp. 1–18, 2017.

[171] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," *Proceedings of the IEEE international conference on computer vision*, pp. 2938–2946, 2015.

[172] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-based localization using lstms for structured feature correlation," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 627–637, 2017.

[173] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5974–5983, 2017.

[174] Z. Lv, F. Dellaert, J. M. Rehg, and A. Geiger, "Taking a deeper look at the inverse compositional algorithm," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4581–4590, 2019.

[175] Y. An, J. Shi, D. Gu, and Q. Liu, "Visual-lidar slam based on unsupervised multi-channel deep neural networks," *Cognitive Computation*, vol. 14, no. 4, pp. 1496–1508, 2022.

[176] M. F. Aslan, A. Durdu, A. Yusefi, and A. Yilmaz, "Hvionet: A deep learning based hybrid visual–inertial odometry approach for unmanned aerial system position estimation," *Neural Networks*, vol. 155, pp. 461–474, 2022.

[177] W. Yuan, X. Gu, Z. Dai, S. Zhu, and P. Tan, "Neural window fully-connected crfs for monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3916–3925, 2022.

[178] A. Agarwal and C. Arora, "Attention attention everywhere: Monocular depth prediction with skip attention," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 5861–5870, 2023.

[179] R. Mahjourian, M. Wicke, and A. Angelova, "Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5667–5675, 2018.

[180] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, "Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 340–349, 2018.

[181] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 7286–7291, 2018.

[182] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1851–1858, 2017.

[183] K. Tateno, F. Tombari, I. Laina, and N. Navab, "Cnn-slam: Real-time dense monocular slam with learned depth prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[184] C. Tang and P. Tan, "Ba-net: Dense bundle adjustment network," *arXiv preprint arXiv:1806.04807*, 2018.

[185] K. Wang, K. Wang, and S. Shen, "Flownorm: A learning-based method for increasing convergence range of direct alignment," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2109–2115, IEEE, 2020.

[186] S. Wen, T. Wang, and S. Tao, "Hybrid cnn-lstm architecture for lidar point clouds semantic segmentation," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 5811–5818, 2022.

[187] A. Diab, R. Kashef, and A. Shaker, "Deep learning for lidar point cloud classification in remote sensing," *Sensors*, vol. 22, no. 20, p. 7868, 2022.

[188] G. Spampinato, A. Bruna, I. Guarneri, and D. Giacalone, "Deep learning localization with 2d range scanner," *2021 7th International Conference on Automation, Robotics and Applications (ICARA)*, pp. 206–210, 2021.

[189] X. Chen, T. Läbe, A. Milioto, T. Röhling, O. Vysotska, A. Haag, J. Behley, and C. Stachniss, "Overlapnet: Loop closing for lidar-based slam," *arXiv preprint arXiv:2105.11344*, 2021.

[190] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-net: Towards learning based lidar localization for autonomous driving," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6389–6398, 2019.

[191] Y. Cho, G. Kim, and A. Kim, "Deeplo: Geometry-aware deep lidar odometry," *arXiv preprint arXiv:1902.10562*, 2019.

[192] R. Buchanan, V. Agrawal, M. Camurri, F. Dellaert, and M. Fallon, "Deep imu bias inference for robust visual-inertial odometry with factor graphs," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 41–48, 2022.

[193] Z. Wang, Y. Zhu, K. Lu, D. Freer, H. Wu, and H. Chen, "Attention guided unsupervised learning of monocular visual-inertial odometry," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 651–657, IEEE, 2022.

[194] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[195] L. Han, Y. Lin, G. Du, and S. Lian, "Deepvio: Self-supervised deep learning of monocular visual inertial odometry using 3d geometric constraints," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6906–6913, 2019.

[196] E. J. Shamwell, K. Lindgren, S. Leung, and W. D. Nothwang, "Unsupervised deep visual-inertial odometry with online error correction for rgb-d imagery," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 10, pp. 2478–2493, 2019.

[197] M. F. Aslan, A. Durdu, and K. Sabanci, "Visual-inertial image-odometry network (viionet): A gaussian process regression-based deep architecture proposal for uav pose estimation," *Measurement*, vol. 194, p. 111030, 2022.

[198] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[199] Y. Almalioglu, M. Turan, M. R. U. Saputra, P. P. de Gusmão, A. Markham, and N. Trigoni, "Selfvio: Self-supervised deep monocular visual–inertial odometry and depth estimation," *Neural Networks*, vol. 150, pp. 119–136, 2022.

[200] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, 2017.

[201] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture," *Asian conference on computer vision*, pp. 213–228, 2016.

[202] G. Marchesi, C. Eichhorn, D. A. Plecher, Y. Itoh, and G. Klinker, "Envslam: Combining slam systems and neural networks to improve the environment fusion in ar applications," *ISPRS International Journal of Geo-Information*, vol. 10, no. 11, p. 772, 2021.

[203] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[204] A. Valada, J. Vertens, A. Dhall, and W. Burgard, "Adapnet: Adaptive semantic segmentation in adverse environmental conditions," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4644–4651, 2017.

[205] N. Radwan, A. Valada, and W. Burgard, "Vlocnet++: Deep multitask learning for semantic visual localization and odometry," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4407–4414, 2018.

[206] S. Jin, L. Chen, R. Sun, and S. McLoone, "A novel vslam framework with unsupervised semantic segmentation based on adversarial transfer learning," *Applied Soft Computing*, vol. 90, p. 106153, 2020.

[207] Y. Zhao, Z. Xiong, S. Zhou, Z. Peng, P. Campoy, and L. Zhang, "Ksf-slam: a key segmentation frame based semantic slam in dynamic environments," *Journal of Intelligent & Robotic Systems*, vol. 105, no. 1, p. 3, 2022.

[208] A. Valada, N. Radwan, and W. Burgard, "Deep auxiliary learning for visual localization and odometry," *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 6939–6946, 2018.

[209] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger, "Fusion++: Volumetric object-level slam," *2018 international conference on 3D vision (3DV)*, pp. 32–41, 2018.

[210] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "imap: Implicit mapping and positioning in real-time," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6229–6238, 2021.

[211] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[212] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[213] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[214] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 573–580, 2012.

[215] J. Engel, V. Usenko, and D. Cremers, "A photometrically calibrated benchmark for monocular visual odometry," *arXiv preprint arXiv:1607.02555*, 2016.

[216] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1680–1687, 2018.

[217] J.-L. Blanco, F.-A. Moreno, and J. Gonzalez, "A collection of outdoor robotic datasets with centimeter-accuracy ground truth," *Autonomous Robots*, vol. 27, no. 4, pp. 327–351, 2009.

[218] G. Pandey, J. R. McBride, and R. M. Eustice, "Ford campus vision and lidar data set," *The International Journal of Robotics Research*, vol. 30, no. 13, pp. 1543–1552, 2011.

[219] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, "A benchmark for rgb-d visual odometry, 3d reconstruction and slam," *2014 IEEE international conference on Robotics and automation (ICRA)*, pp. 1524–1531, 2014.

[220] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of michigan north campus long-term vision and lidar dataset," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016.

[221] M. Fallon, H. Johannsson, M. Kaess, and J. J. Leonard, "The mit stata center dataset," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1695–1699, 2013.

[222] B. Glocker, S. Izadi, J. Shotton, and A. Criminisi, "Real-time rgb-d camera relocalization," *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 173–179, 2013.

[223] J.-L. Blanco-Claraco, F.-A. Moreno-Duenas, and J. González-Jiménez, "The málaga urban dataset: High-rate stereo and lidar in a realistic urban scenario," *The International Journal of Robotics Research*, vol. 33, no. 2, pp. 207–214, 2014.

[224] A. S. Huang, M. Antone, E. Olson, L. Fletcher, D. Moore, S. Teller, and J. Leonard, "A high-rate, heterogeneous data set from the darpa urban challenge," *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1595–1601, 2010.

[225] S. Y. Loo, A. J. Amiri, S. Mashohor, S. H. Tang, and H. Zhang, "Cnn-svo: Improving the mapping in semi-direct visual odometry using single-image depth prediction," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5218–5223, 2019.

[226] J. A. Placed and J. A. Castellanos, "A deep reinforcement learning approach for active slam," *Applied Sciences*, vol. 10, no. 23, p. 8386, 2020.

[227] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Josa a*, vol. 4, no. 4, pp. 629–642, 1987.

[228] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother, "Dsac-differentiable ransac for camera localization," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6684–6692, 2017.

[229] E. Brachmann and C. Rother, "Learning less is more-6d camera localization via 3d surface regression," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4654–4662, 2018.

[230] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.

[231] M. Bujanca, X. Shi, M. Spear, P. Zhao, B. Lennox, and M. Luján, "Robust slam systems: Are we there yet?," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[232] M. Bujanca, P. Gafton, S. Saeedi, A. Nisbet, B. Bodin, M. F. O'Boyle, A. Davison, P. Kelly, G. Riley, B. Lennox, M. Luján, and S. Furber, "Slambench 3.0: Systematic automated reproducible evaluation of slam systems for robot vision challenges and scene understanding," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6351–6358, 2019.

[233] R. Duan, Y. Feng, and C.-Y. Wen, "Deep pose graph-matching-based loop closure detection for semantic visual slam," *Sustainability*, vol. 14, no. 19, p. 11864, 2022.

[234] J. Wu, Q. Shi, Q. Lu, X. Liu, X. Zhu, and Z. Lin, "Learning invariant semantic representation for long-term robust visual localization," *Engineering Applications of Artificial Intelligence*, vol. 111, p. 104793, 2022.

[235] S. Wen, Y. Zhao, X. Yuan, Z. Wang, D. Zhang, and L. Manfredi, "Path planning for active slam based on deep reinforcement learning under unknown environments," *Intelligent Service Robotics*, vol. 13, pp. 263–272, 2020.

[236] K. Naveed, M. L. Anjum, W. Hussain, and D. Lee, "Deep introspective slam: Deep reinforcement learning based approach to avoid tracking failure in visual slam," *Autonomous Robots*, vol. 46, no. 6, pp. 705–724, 2022.

[237] A. Safa, T. Verbelen, I. Ocket, A. Bourdoux, H. Sahli, F. Catthoor, and G. Gielen, "Fusing event-based camera and radar for slam using spiking neural networks with continual stdp learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2782–2788, IEEE, 2023.

Jeremías Gaia In 2019, he completed his degree in Electronic Engineering and progressed to pursue a PhD. Concurrently, he serves as a research fellow at CONICET (Consejo Nacional de Investigaciones Científicas y Tecnológicas). His research focuses on high-speed embedded designs, computer vision, robotics, and artificial intelligence. Presently, he holds a position at the Chair of Microprocessors and Digital Electronics.

Eugenio Orosco Eugenio Conrado Orosco was born, raised, and educated in the city of San Juan, Argentina. In 2008, he graduated from the Facultad de Ingeniera de la Universidad Nacional de San Juan (UNSJ) with a bachelor's degree in Electronic Engineering. Then, in 2013, he obtained his PhD from the same university's Doctoral program in Control Systems Engineering. He previously worked as a research fellow and is now an Assistant Researcher at the Consejo Nacional de Investigaciones Cientificas y Tecnológicas (CONICET). At the moment, he is a Tenured Professor in the Laboratorio de Electrónica Digital, Departamento de Electrónica y Automática, UNSJ, teaching Microprocessors and Digital Electronics III. His expertise and research interests include embedded systems, signal processing, and deep learning. He is a human resource trainer at the undergraduate and postgraduate levels, as well as director of different research projects and technology transfers.

Francisco Rossomando Was born in San Juan, Argentina. He received the electronic engineering degree and the master degree in engineering from the Universidad Nacional de San Juan (UNSJ), Argentina, in 1997 and 2002, respectively. From 2002 to 2006, he worked on his doctorate degree at the Universidad Federal of Espirito Santo (ES-Brazil); with a thesis on the modelling and control of hot rolling mills. He also completed an executive MBA in administration and management in science and technology at the Getulio Vargas Foundation (Brazil) He is currently asociate researcher of the National Council for Scientific and Technical Research of Argentina (CONICET), at the Universidad Nacional de San Juan (UNSJ).

Carlos Soria Carlos Miguel Soria was born in Tucumán, Argentina on November 27, 1970. He graduated as an electrical engineer from the Faculty of Exact Sciences of the National University of Tucumán (UNT) in 1996. In 2000 he graduated as Master in Control Systems Engineering from the School of Engineering of the National University of San Juan (UNSJ). He obtained a PhD degree in Control Systems Engineering at the same University. From 1997 to 2000 he received a scholarship from the FOMEC (Fondo para la Mejora en la Educación) program to complete his Master's degree and from 2001 to 2004 he received a scholarship from the Consejo Nacional de Investigaciones Científicas y Tecnológicas (CONICET). He is currently a Professor at the National University of San Juan and an Independent Researcher for the National Council for Scientific and Technological Research (CONICET, Argentina).