

Visual SLAM: What are the Current Trends and What to Expect?

Ali Tourani*, Hriday Bavle†, Jose-Luis Sanchez-Lopez‡, and Holger Voos§

*†‡§ University of Luxembourg, Interdisciplinary Centre for Security, Reliability, and Trust (SnT),
L-1855 Luxembourg, Luxembourg

§University of Luxembourg, Department of Engineering, L-1359 Luxembourg, Luxembourg

Email: *ali.tourani@uni.lu, †hriday.bavle@uni.lu, ‡joseluis.sanchezlopez@uni.lu, §holger.voos@uni.lu

Abstract—Vision-based sensors have shown significant performance, accuracy, and efficiency gain in Simultaneous Localization and Mapping (SLAM) systems in recent years. In this regard, Visual Simultaneous Localization and Mapping (VSLAM) methods refer to the SLAM approaches that employ cameras for pose estimation and map generation. We can see many research works that demonstrated VSLAMs can outperform traditional methods, which rely only on a particular sensor, such as a Lidar, even with lower costs. VSLAM approaches utilize different camera types (*e.g.*, monocular, stereo, and RGB-D), have been tested on various datasets (*e.g.*, KITTI, TUM RGB-D, and EuRoC) and in dissimilar environments (*e.g.*, indoors and outdoors), and employ multiple algorithms and methodologies to have a better understanding of the environment. The mentioned variations have made this topic popular for researchers and resulted in a wide range of VSLAMs methodologies. In this regard, the primary intent of this survey is to present the recent advances in VSLAM systems, along with discussing the existing challenges and trends. We have given an in-depth literature survey of forty-five impactful papers published in the domain of VSLAMs. We have classified these manuscripts by different characteristics, including the novelty domain, objectives, employed algorithms, and semantic level. We also discuss the current trends and future directions that may help researchers investigate them.

Index Terms—Visual SLAM, Computer Vision, Robotics.

I. INTRODUCTION

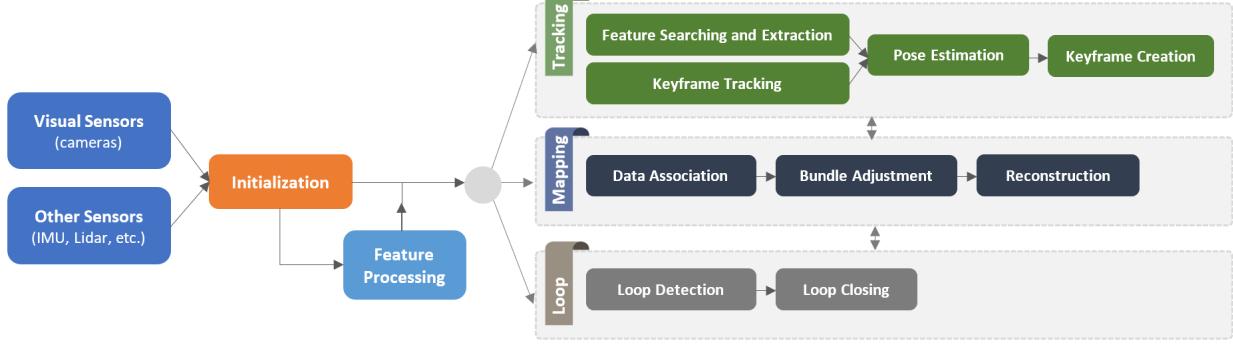
Simultaneous Localization and Mapping (SLAM) refers to the process of estimating an unknown environment’s map while monitoring the location of an *agent* at the same time [1]. Here, the *agent* can be a domestic robot [2], an autonomous vehicle [3], a planetary rover [4], even an Unmanned Aerial Vehicle (UAV) [5], [6] or an Unmanned Ground Vehicle (UGV) [7]. In situations where a prior map of the environment is unavailable or the robot’s location is unknown, SLAM can be utilized to cover a wide range of applications. In this regard, and considering the ever-growing applications of robotics, SLAM has gained huge attention among industry and research community members in recent years [8], [9].

SLAM systems may use various sensors to collect data from the environment, including laser-based sensors, acoustic, and vision sensors [10]. The vision sensors category covers any variety of visual data detectors, including monocular, stereo, event-based, omnidirectional, and Red Green Blue-Depth (RGB-D) cameras. A robot equipped with a vision sensor uses the visual data provided by cameras to estimate

the position and orientation of the robot with respect to its surroundings [11]. The process of using vision sensors to perform SLAM is particularly called Visual Simultaneous Localization and Mapping (VSLAM). Utilizing visual data in SLAM applications has the advantages of cheaper hardware requirement, more straightforward object detection and tracking, and the ability to provide rich visual and semantic information [12]. The captured images (or video frames) can also be used for vision-based applications, including semantic segmentation and object detection, as they store a wealth of data for processing. The mentioned characteristics have recently made VSLAM a trending topic in robotics and prompted robotics and Computer Vision (CV) experts to perform considerable studies and investigations in the last decades. Consequently, VSLAM can be found in various types of applications where it is essential to reconstruct the 3D model of the environment, such as autonomous, Augmented Reality (AR), and service robots [13].

As a general benchmark introduced by [14] to tackle high computational cost, SLAM approaches mainly contain two introductory threads to be executed in parallel, known as *tracking* and *mapping*. Hereby, a fundamental classification of the algorithms used in VSLAM is how researchers employ distinct methods and strategies in each thread. The mentioned solutions look differently at SLAM systems based on the type of data they use, making them dividable into two categories: *Direct* and *Indirect (feature-based)* methods [15]. *Indirect* methods extract feature points (*i.e.*, keypoints) obtained from textures by processing the scene and keep track of them by matching their descriptors in sequential frames. Despite the computationally expensive performance of feature extraction and matching stages, these methods are precise and robust against photo-metric changes in frame intensities. *Direct* algorithms, on the other hand, estimate camera motions directly from pixel-level data and build an optimization problem to minimize the photo-metric error. By relying on photogrammetry, these methods utilize all camera output pixels and track their replacement in sequential frames regarding their constrained aspects, such as brightness and color. These characteristic enable *direct* approaches to model more information from images than *indirect* techniques and enable a higher-accuracy 3D reconstruction. However, while

Fig. 1. The flowchart of a standard visual SLAM approach. Regarding the direct/indirect methodology utilized, the functionality of some of these modules may change or ignored.



direct methods work better in texture-less environments and do not require more computation for feature extraction, they often face large-scale optimization problem [16]. The pros and cons of each approach encouraged researchers to think about developing *Hybrid* solutions, where a combination of both approaches are considered. *Hybrid* methods commonly integrate the detection stage of *indirect* and *direct*, in which one initializes and corrects the other.

Additionally, as VSLAMs mainly include a Visual Odometry (VO) front-end to locally estimate the path of the camera and a SLAM back-end to optimize the created map, the variety of modules used in each category results in implementation variations. VO provides preliminary estimation of the location and pose of the robot based on local consistencies and sent to the back-end for optimization. Thus, the primary distinction between VSLAM and VO is whether or not to take into account the global consistency of the map and the predicted trajectory. Several state-of-the-art VSLAM applications also include two additional modules: loop closure detection and mapping [15]. They are responsible for recognizing previously visited locations for more precise tracking and map reconstruction based on the camera pose.

To summarize, Fig.1 shows the overall architecture of a standard VSLAM approach. Accordingly, the system's inputs may also integrate with other sensor data, such as Inertial Measurement Unit (IMU) and Lidar, to provide more information rather than visual data. Moreover, regarding the *direct* or *indirect* methodology used in a VSLAM pipeline, the functionality of the visual features processing module might be changed or ignored. For instance, the "Feature Processing" stage is only employed in *indirect* approaches. Another factor is utilizing some particular modules such as loop closing detection and bundle adjustment for improved execution.

This paper surveys forty-five VSLAM papers and classifies them into various categories according to diverse aspects. We hope our work will present a reference for the robotics community researchers working to improve VSLAM techniques. The rest of the paper is organized as follows: Section II reviews the evolutionary stages in VSLAM methods that lead to the currently existing systems. We introduce and discuss other published surveys in VSLAM domain in Section III.

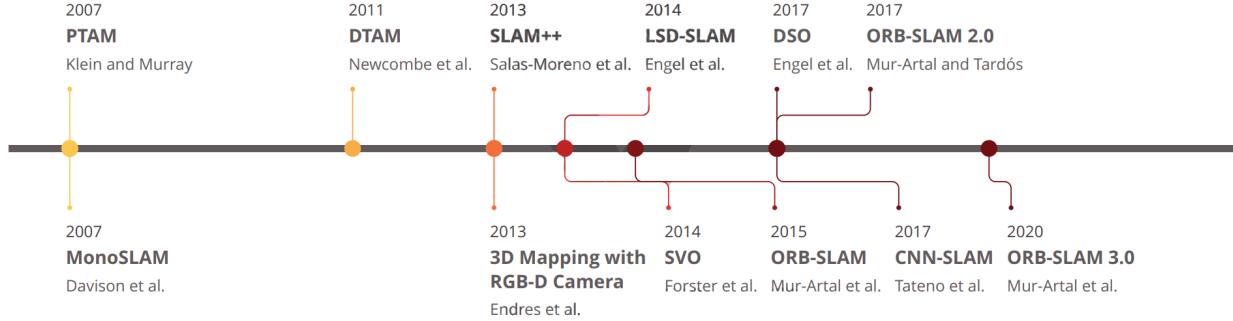
An abstract-level of various VSLAM modules are presented in Section IV and a classification of state-of-the-arts based on the main contributions are available in Section V. We will then talk about unresolved challenges and potential trends in this field in Section VI. The paper finally concludes in Section VII.

II. EVOLUTION OF VISUAL SLAM ALGORITHMS

VSLAM systems have matured over the past years, and several frameworks have played a vital role in this development process. To provide a general picture, Fig.2 illustrates the milestones of the widely-referred VSLAM approaches that impacted the community and have been used as baselines for other frameworks.

Accordingly, the first endeavor in the literature to implement a real-time monocular VSLAM system was developed by Davison *et al.* in 2007 where they introduced a framework titled *Mono-SLAM* [17]. Their indirect framework could estimate the camera motion and 3D elements found in the world using the Extended Kalman Filter (EKF) algorithm [18]. *Mono-SLAM* began the primary action in the VSLAM domain despite the lack of global optimization and loop closure detection modules. However, the maps reconstructed by this method only include landmarks and do not offer further detailed information about the area. Klein *et al.* in [14] proposed Parallel Tracking and Mapping (PTAM) in the same year, in which they divided the entire VSLAM system into two primary threads: *tracking* and *mapping*. This multi-threading baseline was approved by many subsequent works in later works, which will be discussed in this paper. The main idea in their approach was to reduce the computational cost and apply parallel processing to achieve real-time performance. While the *tracking* thread estimates camera motion in real-time, *mapping* predicts the 3D positions of feature points. PTAM was also the first approach to utilize Bundle Adjustment (BA) for jointly optimizing the camera poses and the created 3D map. It uses Features from Accelerated Segment Test (FAST) [19] corner detector algorithm for key-points matching and tracking. Despite better performance than *Mono-SLAM*, the algorithm has complex design and requires user input in the first stage. A direct approach for measuring depth values and motion parameters for map construction was Dense Tracking

Fig. 2. Milestones of highly impactful Visual SLAM approaches.



and Mapping (DTAM) introduced by Newcombe *et al.* [20] in 2011. DTAM was a real-time framework equipped with *dense mapping* and *dense tracking* modules and could determine camera poses by aligning the entire frames with a given depth map. To construct the environment map, the mentioned stages estimate the depth of the scene and the motion parameters, respectively. Although DTAM can provide a detailed presentation of the map, it demands high computational cost to perform in real-time. As another indirect approach in the domain of 3D mapping and pixel-based optimization, Endres *et al.* [21] in 2013 proposed a method that could work with RGB-D cameras. Their method performs in real-time and is focused on low-cost embedded systems and small robots, but it cannot produce accurate results in featureless or challenging scenarios. At the same year, Salas-Moreno *et al.* [22] proposed one of the first endeavors in utilizing semantic information in a real-time SLAM framework, titled SLAM++. Their system employs RGB-D sensor outputs and performs 3D camera pose estimation and tracking to shape a pose graph. A pose graph is a graph in which the nodes represent pose estimates and are connected by edges representing the relative poses between nodes with measurement uncertainty [23]. The predicted poses will then be optimized by merging the relative 3D poses obtained from semantic objects in the scene.

With ripening the baseline of VSLAM, researchers focused on improving the performance and precision of these systems. In this regard, Forster *et al.* in 2014 proposed a hybrid VO approach known as Semi-direct Visual Odometry (SVO) [24] as a part of VSLAM architectures. Their method could merge feature-based and direct approaches to perform sensors' motion estimation and mapping tasks. SVO could work with both monocular and stereo cameras and was equipped with a pose refinement module to minimize re-projection errors. However, the main drawbacks of SVO are employing a short-term data association and the inability to perform loop closure detection and global optimization. LSD-SLAM [25] is another influential VSLAM method introduced by Engel *et al.* in 2014 and contains *tracking*, *depth map estimation*, and *map optimization* threads. The method could reconstruct large-scale maps using its pose-graph estimation module and was equipped with global optimization and loop closure detection feat. The

weakness of LSD-SLAM is its challenging initialization stage that requires all points in a plane, making it a computationally intensive approach. Mur-Artal *et al.* introduced two accurate indirect VSLAM approaches that have attracted the attention of many researchers so far: ORB-SLAM [26] and ORB-SLAM 2.0 [27]. These methods can accomplish localization and mapping in well-textured sequences and perform high-performance position recognition using Oriented FAST and Rotated BRIEF (ORB) features. The first version of ORB-SLAM is able to compute both the camera position and the environment's structure using the keyframes collected from camera locations. The second version is the extension to ORB-SLAM with three parallel threads, including *tracking* for finding feature correspondences, *local mapping* for map management operations, and *loop closing* for detecting new loops and correcting the drift error. Although ORB-SLAM 2.0 can work with both monocular and stereo camera setups, it cannot be used for autonomous navigation due to reconstructing maps with unknown scales. Another drawback of this approach is its inability to work in texture-less areas or environments with repetitive patterns. The most recent version of this framework, named ORB-SLAM 3.0, was proposed in 2021 [28]. It works with various camera types, such as monocular, RGB-D and stereo-vision, and provides improved pose estimation outputs.

In recent years and with the significant influences of deep learning in various domains, deep neural network-based approaches could resolve many issues by providing higher recognition and matching rates. Similarly, replacing hand-crafted with learned features in VSLAM is one of the solutions suggested by many recent deep learning-based methods. In this regard, Tateno *et al.* presented an approach based on Convolutional Neural Networks (CNNs) that processes the input frames for camera pose estimation and uses keyframes for depth prediction, anointed CNN-SLAM [29]. Segmenting camera frames into smaller sections to provide better understanding of the environment is one of the ideas in CNN-SLAM to provide parallel processing and real-time performance. As a different methodology, Engel *et al.* also introduced a new trend in direct VSLAM algorithms titled Direct Sparse Odometry (DSO) [30] that merges a direct approach and a sparse reconstruction to extract the highest intensity points in image

blocks. By tracking sparse sets of pixels, it considers the image formation parameters and uses an indirect tracking method. It should be noted that DSO can only provide perfect accuracy if the photo-metrically calibrated cameras are used and fails to achieve high-accuracy results using regular cameras.

To recap, milestones in the VSLAM systems evolution process reveal that recent approaches focus on the parallel execution of multiple dedicated modules. These modules shaped general-purpose techniques and frameworks compatible with a broad range of sensors and environments. The mentioned characteristic enables them to be executable in real-time and be more flexible in terms of performance improvement.

III. RELATED SURVEYS

There are various survey papers available in the domain of VSLAM that present a general review of the different existing approaches. Each of these papers review the major advantages and disadvantages of employing VSLAM approaches. Macario Barros *et al.* [31] divided approaches in three different classes: visual-only (monocular), visual-inertial (stereo), and RGB-D. They also proposed various criteria for simplifying analyzing VSLAM algorithms. However, they did not include other vision sensors, such as event camera-based ones, which we will discuss later in IV-A. Chen *et al.* [32] reviewed a wide range of traditional and semantic VSLAM publications. They divided the SLAM development era into *classical*, *algorithmic-analysis*, and *robust-perception* stages and introduced hot issues there. They also summarized classical frameworks that employ direct/indirect methodologies and investigated the impact of deep learning algorithms in semantic segmentation. Although their work provides a comprehensive study of the advanced solutions in this domain, the classification of approaches is only restricted to the *feature types* employed in feature-based VSLAMs. Jia *et al.* [33] surveyed numerous manuscripts and presented a brief comparison between graph optimization-based methods and deep learning-equipped approaches. Despite presenting a proper comparison, their discussion cannot be generalized due to reviewing a limited number of papers. In another work, Abaspur Kazerouni *et al.* [34] covered various VSLAM methods, utilized sensory equipment, datasets, and modules and simulated several indirect approaches for comparison and analysis. They contribute only to the feature-based algorithms -*e.g.*, *HOG*, *Scale-Invariant Feature Transform (SIFT)*, and *Speeded Up Robust Features (SURF)* and deep learning-based solutions. Bavle *et al.* [35] analyzed the situational awareness aspects in various SLAM and VSLAM applications and discussed their missing points. They could conclude that operating the lacking situational awareness features could enhance the performance of the current research works.

Other surveys studied the latest VSLAM approaches focused on a particular topic or trend. For instance, Duan *et al.* [15] investigated the progress of deep learning in visual SLAM systems for transportation robotics. The authors summarized the advantages and drawbacks of utilizing various

deep learning-based methods in VO and loop closure detection tasks in their paper. The significant advantage of using deep learning approaches in VSLAMs is the accurate feature extraction in pose estimation and the overall performance calculation. In another work in the same field, Arshad and Kim [36] focused on the impact of deep learning algorithms in loop closure detection using visual data. They reviewed various VSLAM papers and analyzed the long-term autonomy of robots in different conditions. Singandhupe and La [37] reviewed the impact of VO and VSLAM in driverless vehicles. They collected approaches that have been evaluated on the KITTI dataset, enabling them to have a brief description of the advantages and demerits of each system. Cheng *et al.* [32] in a similar manuscript reviewed the VSLAM-based autonomous driving systems and raised the future development trends of such systems. Some other researchers surveyed VSLAM works with the ability to work in real-world conditions. For instance, Saputra *et al.* [38] targeted the variations of VSLAM techniques operating in dynamic and rough environments and discussed the reconstruction, segmentation, tracking, and parallel execution of threads problems.

Regarding the mentioned surveys, the current survey has particularities that set it apart from other surveys presented so far and provides a comprehensive review of the VSLAM systems presented in different venues. In this regard, the major contributions of this survey compared to other available VSLAM surveys are:

- Categorizing various recent VSLAM publications regarding the main contributions, criteria, and objectives of researchers in proposing new solutions,
- Analyzing the current trends of VSLAM systems by profoundly investigating different approaches regarding dissimilar aspects,
- Introducing the potential contributions of VSLAM for researchers

IV. VSLAM SETUP CRITERIA

Considering various VSLAM approaches, we can classify different setups and configurations available into categories mentioned below:

A. Sensors and Data Acquisition

The early-stage implementation of a VSLAM algorithm introduced by Davison *et al.* [17] was equipped with a monocular camera for trajectory recovery. Monocular cameras are the most common vision sensors for a wide range of tasks, such as object detection and tracking [39]. Stereo cameras, on the other hand, contain two or more image sensors, enabling them to perceive depth in the captured images, which leads to more accurate performance in VSLAM applications. These camera setups are cost-efficient and provide informative perception for higher accuracy demands. RGB-D cameras are other variations of visual sensors used in VSLAMs and supply both the depth and colors in the scene. The mentioned vision sensors can provide rich information about the environment in straightforward circumstances -*e.g., proper lighting and motion speed*- but they

often struggle to cope with conditions where the illumination is low or the dynamic range in the scene is high.

In recent years, event cameras have also been used in various VSLAM applications. These low latency bio-inspired vision sensors generate pixel-level brightness changes instead of standard intensity frames when a motion is detected, leading to a high dynamic range output with no motion blur impact [40]. In contrast with standard cameras, event-based sensors supply trustworthy visual information during high-speed motions and wide-range dynamic scenarios but fail to provide sufficient information when the motion rate is low. Although event cameras can outperform standard visual sensors in severe illumination and dynamic range conditions, they mainly generate unsynchronized information about the environment. This makes traditional vision algorithms unable to process the outputs of these sensors [41]. Additionally, using the spatio-temporal windows of events along with the data obtained from other sensors can provide rich pose estimation and tracking information.

Moreover, some approaches use a multi-camera setup to counter the common issues of working in a real-world environment and improve localization precision. Utilizing multiple visual sensors aid in situations where complicated problems such as occlusion, camouflage, sensor failure, or sparsity of trackable texture occurs by providing cameras with overlapping field of views. Although multi-camera setups can resolve some data acquisition issues, camera-only VSLAMs may face various challenges such as motion blur when encountering fast-moving objects, features mismatching in low or severe illumination, dynamic object ignorance in scenarios with high pace changes, *etc.* Hence, some VSLAM applications may equip with multiple sensors alongside cameras. Fusing the events and standard frames [42] or integrating other sensors such as Lidars [43] and IMUs to VSLAM are some of the existing solutions.

B. Target Environments

As a strong presumption in many traditional VSLAM practices, the robot works in a static world with no sudden or unanticipated changes. Consequently, although many systems could demonstrate successful application in specific settings, some unexpected changes in the environment (*e.g.*, the existence of moving objects) are likely to cause complications for the system and degrade the state estimation quality to a large extent. Systems that work in dynamic environments usually employ algorithms such as Optical Flow or Random Sample Consensus (RANSAC) [44] to detect movements in the scene, classify the moving objects as outliers, and skip them while reconstructing the map. Such systems utilize either *geometry/semantic* information or try to improve the localization scheme by combining the results of these two [45].

Additionally, we can classify different environments into *indoor* and *outdoor* categories as a general taxonomy. An *outdoor* environment can be an *urban* area with structural landmarks and massive motion changes such as buildings and road textures, or an *off-road* zone with a weak motion

state such as moving clouds and vegetation, the texture of the sand, *etc.* As a result of this, the amount of trackable points in *off-road* environments is less than the *urban* areas, which increases the risk of localization and loop closure detection failure. *Indoor* environments, on the other hand, contain scenes with entirely different global spatial properties, such as corridors, walls, and rooms. We can anticipate that while a VSLAM system might work well in one of the mentioned zones, it might not show the same performance in other environments.

C. Visual Features Processing

As discussed in Section I, detecting visual features and utilizing feature descriptors information for pose estimation is an inevitable stage of indirect VSLAM methodologies. These approaches employ various feature extraction algorithms to understand the environment better and track the feature points in consecutive frames. Feature extraction stage contains a wide range of algorithms, including SIFT [46], SURF [47], FAST [19], Binary Robust Independent Elementary Features (BRIEF) [48], ORB [49], *etc.* Among them, ORB features have the advantage of fast extraction and matching without losing huge accuracy compared to SIFT and SURF [50].

The problem with some of the mentioned methods is that they cannot effectively adapt to various complex and unforeseen situations. Thus, many researchers employed CNNs to extract deep-seated features of images for various stages, including VO, pose estimation, and loop closure detection. These techniques may represent supervised or unsupervised frameworks according to the functionality of the methods.

D. System Evaluation

While some of the VSLAM approaches, especially those with the capability of working in dynamic and challenging environments are tested on robots in real-world conditions, many research works have used publicly available datasets to demonstrate their applicability. In this regard, the *RAWSEEDS Dataset* by Bonarini *et al.* [51] is a well-known multi-sensor benchmarking tool, containing indoor, outdoor, and mixed robot trajectories with ground-truth data. It is one of the oldest publicly available benchmarking tools for robotic and SLAM purposes. *Scenenet RGB-D* by McCormac *et al.* [52] is another favored dataset for scene understanding problems, such as semantic segmentation and object detection, containing five million large-scale rendered RGB-D images. The dataset also contains pixel-perfect ground-truth labels and exact camera poses and depth data, making it a potent tool for VSLAM applications. Many recent works in the domain of VSLAM and VO have tested their approaches on the *TUM RGB-D* dataset [53]. The mentioned dataset and benchmarking tool contains color and depth images captured by a Microsoft Kinect sensor and their corresponding ground-truth sensors trajectories. Also, *NTU VIRAL* by Nguyen *et al.* [54] is a dataset collected by a UAV equipped with 3D lidars, cameras, IMUs, and multiple Ultra-widebands (UWBs). The dataset

contains indoor and outdoor instances and targeted for evaluating autonomous driving and aerial operation performances.

Moreover, *EuRoC MAV* by Burri *et al.* [55] is another popular dataset containing images captured by a stereo camera, along with synchronized IMU measurements and motion ground-truth. Collected data in *EuRoC MAV* are classified into easy, medium, and difficult categories according to the surrounding conditions. *OpenLORIS-Scene* by Shi *et al.* [56] is another publicly available dataset for VSLAM works, containing a wide range of data collected by a wheeled robot equipped with various sensors. It provides proper data for monocular and RGB-D algorithms, along with odometry data from wheel encoders. As a more general purpose dataset used in VSLAM applications, *KITTI* [57] is popular collection of data, captured by two high-resolution RGB and grayscale video cameras on a moving vehicle. *KITTI* provides accurate ground-truth using GPS and laser sensors, making it a highly popular dataset to be used in mobile robotics and autonomous driving. The *TartanAir* [58] is another benchmarking dataset for evaluation of SLAM algorithms under challenging scenarios. Additionally, the Imperial College London and National University of Ireland Maynooth (ICL-NUIM) [59] dataset is another VO dataset containing handheld RGB-D camera sequences, which has been used as a benchmark for many SLAM works.

In contrast with the previous datasets, some other datasets contain data acquired using particular cameras instead of regular ones. For instance, the *Event Camera Dataset* introduced by Mueggler *et al.* [60] is a dataset with collected samples using an event-based camera for high-speed robotic evaluations. Dataset instances contain inertial measurements and intensity images captured by a motion-capture system, making it a suitable benchmark for VSLAMs equipped with event cameras.

The mentioned datasets are used in multiple VSLAM methodologies according to their sensor setups, applications, and target environments. These datasets mainly contain cameras' extrinsic and intrinsic calibration parameters and ground-truth data. The summarized characteristics of the datasets and some instances of each are shown in Table I and Fig.3, respectively.

E. Semantic Level

Semantic information is required for the robot to understand the scene around it and make more profitable decisions. In many recent VSLAM works, adding semantic-level information to the geometry-based data is preferred to the pure geometry-based approaches, enabling them to deliver conceptual knowledge of the surroundings [61]. In this regard, a pre-trained Object Recognition module can add semantic information to the VSLAM models [62]. One of the most recent approaches is employing CNNs in VSLAM applications. In general, semantic VSLAM approaches contain four primary components described below [43]:

- *Tracking module*: it uses the two-dimensional feature points extracted from consecutive video frames to estimate the camera pose and construct three-dimensional

map points. Calculation of the camera pose and construction of the 3D map points build the baselines of the localization and mapping processes, respectively.

- *Local mapping module*: by processing two sequential video frames, a new 3D map point is created, which is used along with a BA module for an improved camera pose.
- *Loop closing module*: by comparing the keyframes to the extracted visual features and assessing the similarities between them, it tunes the camera pose and optimizes the constructed map.
- *Non-Rigid Context Culling (NRCC)*: the main goal of employing NRCC is to filter temporal objects from video frames in order to reduce their detrimental impact on the localization and mapping stages. It mainly contains a masking/segmentation process for separating various unstable instances in frames, such as people. Since it leads to lower the number of feature points to be processed, NRCC simplifies the computational part and results in a more robust performance.

Accordingly, utilizing semantic level in VSLAM approaches can improve the uncertainty in pose estimation and map reconstruction. However, the current challenge here is to correctly use the extracted semantic information without hugely impacting the computational cost.

V. VSLAM APPROACHES BASED ON THE MAIN OBJECTIVES

In order to pinpoint VSLAM approaches that achieve rich outcomes and present robust architectures, we collected and filtered out highly cited publications published in top-notch venues in the recent years from Google Scholar¹ and well-known Computer Science bibliography databases: Scopus² and DBLP³. We also studied the manuscripts referred to in the mentioned publications and purified the ones most relevant to the VSLAM domain. After exploring the papers, we could categorize the collected publications based on their main objectives to solve a particular problems into sub-sections presented below:

A. Objective I: Multi-sensor Processing

This category covers the range of VSLAM approaches that employ various sensors to understand the environment better. While some techniques rely on *only cameras* as the employed visual sensors, others combine *various sensors* to enhance the accuracy of their algorithm.

1) Employing Multiple Cameras:

As it might be difficult to recreate the 3D trajectories of moving objects with a single camera, some researchers suggest using multiple cameras instead. For instance, *CoSLAM*⁴ is a VSLAM system introduced by Zou and Tan [63], which uses separate cameras deployed on various platforms to reconstruct

¹<https://scholar.google.com/>

²<https://www dblp.org/>

³<https://www scopus.com>

⁴<https://github.com/danping/CoSLAM>

TABLE I
COMMONLY USED DATASETS FOR VSLAM APPLICATIONS. *GT* IN THE TABLE REFERS TO THE AVAILABILITY OF GROUND-TRUTH VALUES.

Dataset Name	Year	Environment		GPS	lidar	sonar	IMU	Utilized Sensors				GT
		indoor	outdoor					mono	stereo	RGB-D	event	
RAWSEEDS [51]	2006	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓
KITTI [57]	2012		✓	✓	✓		✓	✓	✓			✓
ICL-NUIM [59]	2014	✓								✓		✓
TUM RGB-D [53]	2016	✓					✓			✓		✓
EuRoC MAV [55]	2016	✓					✓	✓	✓			✓
Event Camera Dataset [60]	2017		✓				✓				✓	✓
SceneNet RGB-D [52]	2017	✓					✓			✓		✓
OpenLORIS-Scene [56]	2020	✓			✓		✓	✓	✓	✓		✓
TartanAir [58]	2020	✓	✓		✓		✓	✓	✓	✓		✓
NTU VIRAL [54]	2021	✓	✓		✓		✓	✓				✓

robust maps. Their system combines multiple cameras moving around independently in a dynamic environment and reconstructs the map regarding their overlapping fields of view. The process makes it easier to rebuild dynamic points in 3D by mixing intra- and inter-camera pose estimation and mapping. *CoSLAM* tracks visual features using the Kanade-Lucas-Tomasi (KLT) algorithm and operates in static and dynamic contexts, including indoors and outdoors, where the relative positions and orientations may shift over time. The primary drawback of this method is requiring sophisticated hardware to interpret numerous camera outputs and increased computational cost by adding more cameras.

For challenging off-road settings, Yang *et al.* [64] developed a multi-camera cooperative panoramic vision VSLAM approach. Their approach gives each camera independence to increase the performance of the VSLAM system under challenging conditions, such as occlusion and texture sparsity. In order to determine the matching range, they extract ORB features from cameras' overlapping fields of view. Additionally, they employed a deep learning technique based on a CNN to recognize similar features for loop closure detection. For the experiment, the authors used a dataset produced by a panoramic camera and an integrated navigation system.

MultiCol-SLAM is another an open-source VSLAM framework with multi-camera configurations by Urban and Hinz [65]. They use their previously created model, *MultiCol*, to enhance ORB-SLAM utilizing a keyframe-based process that supports multiple fisheye cameras. They added a Multi-Keyframes (MKFs) processing module to ORB-SLAM, which collects turns images into keyframes. Authors also proposed the idea of multi-camera loop closing, in which loop closures are detected from MKFs. Although their method operates in real-time, it requires a significant computer power because several threads must run simultaneously.

2) Employing Multiple Sensors:

Some other approaches proposed fusing various sensors and using vision- and inertial-based sensors outputs for better performance. In this regard, a low-cost indirect lidar-assisted VSLAM called *CamVox*⁵ was proposed by Zhu *et al.* [66] and demonstrated reliable performance and accuracy. Their method uses ORB-SLAM 2.0 and combines the unique capabilities

offered by Livox lidars as the premium depth sensors with the outputs from RGB-D cameras. The authors used an IMU to synchronize and correct the non-repeating scanned locations. Their contribution is presenting an autonomous lidar-camera calibration method that operates in uncontrolled environments. Real-world tests on a robot platform indicate that *CamVox* performs in real-time while processing the environment.

Authors in [67] proposed a multi-modal system titled *VIRAL* (*Visual-Inertial-Ranging-Lidar*) *SLAM* that couples camera, lidar, IMU, and UWB. They also presented a map-matching marginalization scheme for visual features based on the local map constructed from lidar point clouds. The visual components are extracted and tracked using BRIEF algorithm. The framework also contains a synchronization scheme and trigger for the utilized sensors. They tested their approach on simulation environments and their generated dataset titled NTU VIRAL [54] that contains data captured by camera, lidar, IMU, and UWB sensors. However, their approach is computationally intensive due to handling synchronization, multi-threading, and sensor conflict resolution.

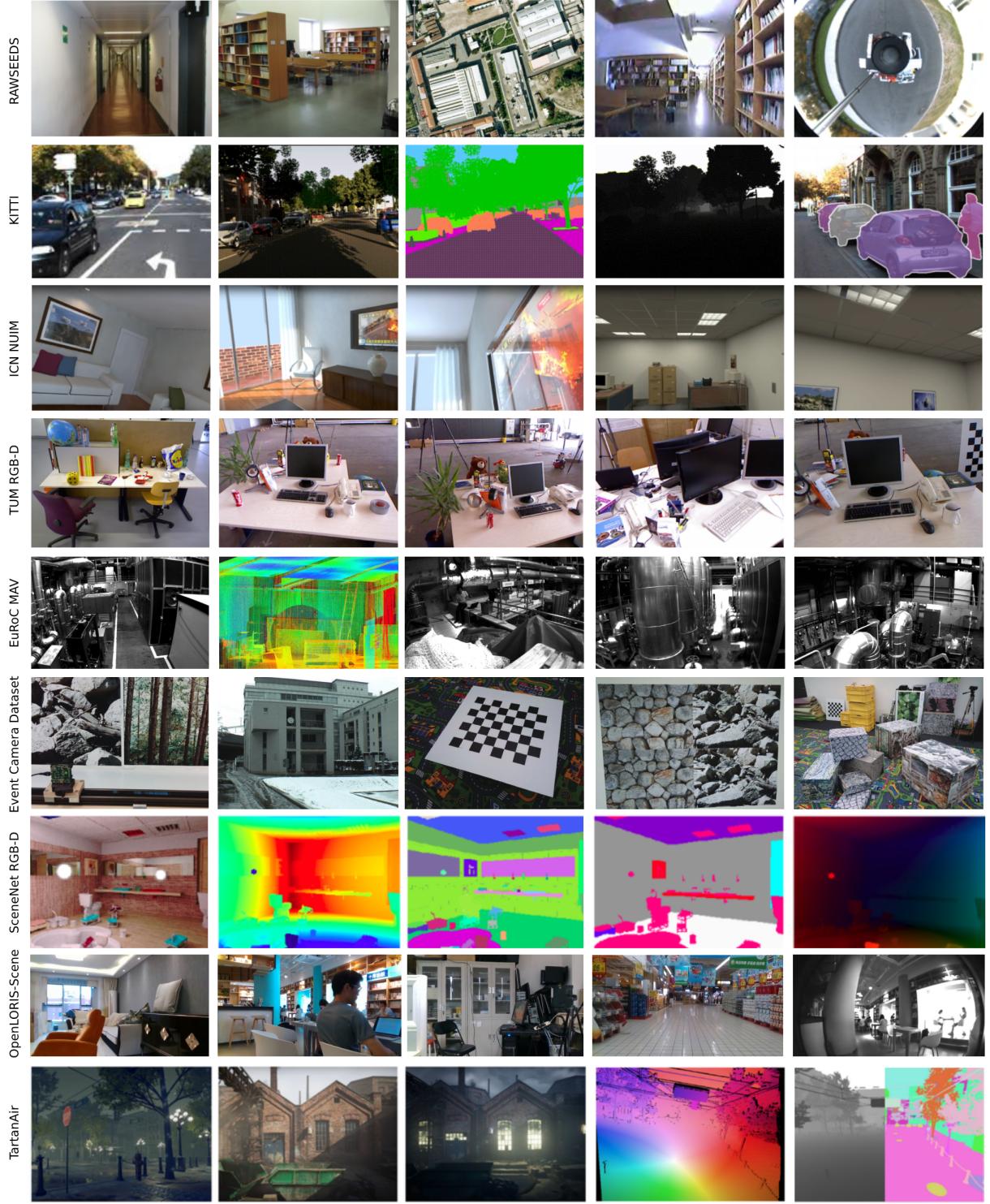
Vidal *et al.* [42] proposed integrating events, camera frames, and IMU in parallel configurations for reliable position estimation in high-speed settings. Their *Ultimate SLAM*⁶ system is based on an event camera and a keyframe-based nonlinear optimization pipeline introduced in [68]. They use the FAST corner detector and the Lucas-Kanade tracking algorithm for feature detection and tracking, respectively. *Ultimate SLAM* avoids motion blur problems brought on by high-speed activity and operates in dynamic situations with varied lighting conditions. The efficiency of this technique on the "Event Camera Dataset" was obvious in comparison to alternative event-only and conventional camera configurations. The authors also tested Ultimate SLAM on an autonomous quadrotor equipped with an event camera to show how their system can manage flight conditions that are impossible for conventional VO platforms to handle. The major challenge in *Ultimate SLAM* is the synchronization of events with standard frame outputs.

A tightly-coupled monocular camera and UWB range sensors were suggested by Nguyen *et al.* [69] for VSLAM. They use a combination of feature-based (visible) and feature-less (UWB) landmarks to create a map. It operates effectively

⁵<https://github.com/ISEE-Technology/CamVox>

⁶https://github.com/uzh-rpg/rpg_ultimate_slam_open

Fig. 3. Instances of some of the most popular visual SLAM datasets used for evaluation in various papers. The characteristics of these datasets can be found in Table I.



when UWB is exposed to multi-path effects in congested surroundings. They built their indirect method on ORB-SLAM and employ ORB characteristics for pose estimation. They tested their system on a generated dataset with hand-carried movements simulating an employed aerial robot. The syn-

chronization of the camera and UWB sensor is one of the difficulties in this case, but it has been overcome by employing a new camera pose with its related timestamp for each new image.

B. Objective II: Pose Estimation

Methods classified in this category focus on how to improve the pose estimation of a VSLAM approach using various algorithms.

1) Employing Lines/Points Data:

In this regard, Zhou *et al.* [70] suggested employing building structural lines as useful features to determine the camera pose. Structural lines are associated with dominant directions and encode global orientation information, resulting in improved predicted trajectories. *StructSLAM*, the mentioned method, is a 6-Degree of Freedom (DoF) VSLAM technique that operates in both low-feature and featureless conditions. It employs EKF to estimate variables based on the current directions in the scene. For evaluation, the indoor scenes dataset from RAWSEEDS 2009 and a set of generated sequential images dataset were used.

*Point and Line SLAM (PL-SLAM)*⁷, a VSLAM system based on ORB-SLAM optimized for non-dynamic low texture settings, was introduced by Pumarola *et al.* [71]. The system simultaneously fuses line and point features for improved posture estimation and helps running in situations with few feature points. The authors tested *PL-SLAM* on their generated dataset and TUM RGB-D. The drawback of their method is the computational cost and the essence of using other geometric primitives, *e.g.*, planes, for a more robust accuracy.

Gomez-Ojeda *et al.* [72] introduced *PL-SLAM*⁸ (different from the framework with the same name by Pumarola *et al.* in [71]), an indirect VSLAM technique that uses points and lines in stereo vision cameras to reconstruct an unseen map. They merged segments obtained from points and lines in all VSLAM modules with visual information taken from successive frames in their approach. Using the ORB and Line Segment Detector (LSD) algorithms, points and line segments are retrieved and tracked in subsequent stereo frames in *PL-SLAM*. The authors tested *PL-SLAM* on EuRoC and KITTI datasets and could outperform the stereo version of ORB-SLAM 2.0 in terms of performance. One of the main drawbacks of *PL-SLAM* is the computational time required for the feature tracking module and considering all structural lines to extract information about the environment.

A degeneracy avoidance technique for monocular point- and line-based VSLAM systems was introduced by Lim *et al.* [73]. A strong Optical Flow-based line tracking module that extracts line characteristics, filters out short lines in each frame, and matches the previously identified lines is another contribution of their method. To demonstrate the efficacy of their technique and show that it was superior to the established point-based approaches, they tested their system on EuRoC MAV dataset. The system lacks an adaptive approach to identify the correct optimization parameters, notwithstanding the strong findings.

2) Using Extra Features:

Dual Quaternion Visual SLAM (DQV-SLAM), a framework

for stereo-vision cameras that uses a broad Bayesian framework for 6-DoF posture estimation was proposed in [74]. In order to prevent the linearization of the nonlinear spatial transformation group, their approach uses progressive Bayes updates. For point clouds of maps and Optical Flow, *DQV-SLAM* uses ORB features to enable reliable data association in dynamic circumstances. On KITTI and EuRoC datasets, the method could estimate experiment results reliably. However, it lacks a probabilistic interpretation for stochastic modeling of poses and is computationally demanding for sampling approximation-based filtering.

Muñoz-Salinas *et al.* [75] developed a technique using artificial squared planar markers to recreate a large-scale interior environment map. Their real-time *SPM-SLAM* system can solve the ambiguity issue of pose estimation using markers if at least two of them are visible in each video frame. They created a dataset with video sequences of markers placed in two rooms joined by a door for examination. Although *SPM-SLAM* is cost-effective, it only works when numerous planar markers are scattered around the area while at least two are visible for marker connection recognition. Moreover, the ability of their framework to handle dynamic changes in the scene is not measured.

3) Deep Learning:

In another approach, Bruno and Colombini [76] proposed *LIFT-SLAM*, which combines deep learning-based feature descriptors with the conventional geometry-based systems. They expanded the ORB-SLAM system's pipeline and employed a CNN to extract features from images, using the learned features to provide more dense and precise matches. For purposes of detection, description, and orientation estimation, *LIFT-SLAM* fine-tunes a Learned Invariant Feature Transform (LIFT) deep neural network. Studies using the KITTI and EuRoC MAV datasets' indoor and outdoor instances revealed that *LIFT-SLAM* outperforms conventional feature-based and deep learning-based VSLAM systems in terms of accuracy. However, the weaknesses of the method are its computationally intensive pipeline and un-optimized CNN design, which leads to near real-time performance.

Naveed *et al.* [77] proposed a deep learning-based VSLAM solution with a reliable and consistent module, even on routes with extreme turns. Their approach outperformed several VSLAMs and used a deep reinforcement learning network trained on realistic simulators. Furthermore, they provided a baseline for active VSLAM evaluation and could properly generalize across actual indoor and outdoor environments. The network's path planner developed the ideal path data, which is received by its base system, ORB-SLAM. They produced a dataset with actual navigation episodes in challenging and texture-less environments for evaluation.

As another approach, *RWT-SLAM* is a deep feature matching-based VSLAM framework the authors in [78] proposed for weakly textured situations. Their method, which is based on ORB-SLAM, is fed with feature masks from an enhanced LoFTR [79] algorithm for local image feature matching. A CNN architecture and the LoFTR algorithm were

⁷<https://github.com/HarborC/PL-SLAM>

⁸<https://github.com/rubengooj/pl-slam>

used to extract coarse-level and fine-level descriptors in the scene, respectively. *RWT-SLAM* is examined on the TUM RGB-D and OpenLORIS-Scene datasets, as well as a real-world dataset gathered by the authors. However, their system is computationally demanding despite the robust feature matching results and performance.

C. Objective III: Real-world Viability

Approaches in this category have the primary objective of being used in various environments and working under several scenarios. We notice that the references in this section are highly integrated with *semantic* information extracted from the environment and present an end-to-end VSLAM application.

1) Dynamic Environments:

In this regard, a VSLAM system titled *DS-SLAM*⁹ has been introduced by Yu *et al.* [61], which can be used in dynamic contexts and offers semantic-level information for map construction. The system is built upon ORB-SLAM 2.0 and contains five threads: *tracking*, *semantic segmentation*, *local mapping*, *loop closing*, and *dense semantic map construction*. To exclude dynamic items before the pose estimation process and increase localization accuracy, *DS-SLAM* employs the Optical Flow algorithm with a real-time semantic segmentation network called *SegNet* [80]. *DS-SLAM* has been tested in real-world settings and with RGB-D cameras, as well as on the TUM RGB-D dataset. However, despite its high accuracy in localization, it faces semantically segmentation limitations and computationally intensive features.

Semantic Optical Flow SLAM (SOF-SLAM), an indirect VSLAM system built upon the RGB-D mode of ORB-SLAM 2.0, is another method in highly dynamic environments proposed by Cui and Ma [45]. Their approach uses the Semantic Optical Flow dynamic feature detection module, which extracts and skips the changing features concealed in the semantic and geometric information provided by ORB feature extraction. In order to deliver accurate camera pose and environment reports, *SOF-SLAM* makes use of *SegNet*'s pixel-wise semantic segmentation module. In extremely dynamic situations, experimental findings on the TUM RGB-D dataset and in real-world settings demonstrated that *SOF-SLAM* performs better than ORB-SLAM 2.0. However, the ineffective method of non-static feature recognition and reliance on just two consecutive frames for this purpose are *SOF-SLAM*'s weakness points.

Using the Optical Flow method to separate and eliminate dynamic feature points, Cheng *et al.* [81] suggested a VSLAM system for dynamic environments. They have utilized the ORB-SLAM pipeline's structure and supplied it with fixed feature points generated from typical monocular camera outputs for precise posture estimation. The system indicated operates in featureless circumstances by sorting Optical Flow values and using them for feature recognition. According to experimental results on the TUM RGB-D dataset, the suggested system functions well in dynamic indoor circumstances.

However, the system's configuration uses an offline threshold for motion analysis, making it difficult to use in a variety of dynamic environment situations.

Another VSLAM strategy was released by Yang *et al.* [82] that reconstructs the environment map using semantic segmentation network data, a motion consistency detection technique, and geometric restrictions. Their approach, which is based on ORB-SLAM 2.0's RGB-D variant, performs well in dynamic and indoor environments. Only the stable features from the scene are retained using an improved ORB feature extraction technique, while the dynamic characteristics are disregarded. The features and the semantic data will then be combined to create a static semantic map. Evaluation findings on the Oxford and TUM RGB-D datasets demonstrated the effectiveness of their approach in enhancing location accuracy and creating semantic maps with a wealth of data. However, their system can run into problems in corridors or places with less information.

2) Deep Learning-based Solutions:

In another work by Li *et al.* [83] called *DXSLAM*¹⁰, deep learning is used to find keypoints that resemble SuperPoints and to produce both the general descriptors and the images' keypoints. They trained the cutting-edge deep CNN HF-NET to produce frame- and keypoint-based descriptions by extracting local and global information from each frame. They also used the offline Bag of Words (BoW) method to train a visual vocabulary of local characteristics for precise loop closure recognition. *DXSLAM* operates in real-time without using a Graphics Processing Unit (GPU) and is compatible with contemporary CPUs. Even if such qualities are not specifically addressed, it has a great ability to resist dynamic changes in dynamic contexts. *DXSLAM* has been tested on TUM RGB-D and OpenLORIS-Scene datasets and both indoor and outdoor images and could achieve more accurate results than ORB-SLAM 2.0 and DS-SLAM. However, the major disadvantages of this method are complex architecture for feature extraction and incorporating deep features into an old SLAM framework.

In another approach, Li *et al.* [84] developed a real-time VSLAM technique for extracting feature points based on deep learning in complicated situations. The method can run on a GPU and supports the creation of 3D dense maps and is a multi-task CNN for feature extraction with self-supervision capabilities. The CNN output is binary code strings with a fix-length of 256, making it possible to be replaced by more conventional feature point detectors like ORB. It comprises three threads for reliable and timely performance in dynamic scenarios: *tracking*, *local mapping*, and *loop closing*. The system that supports monocular and RGB-D cameras using ORB-SLAM 2.0 as a baseline. The authors tested their methodology on the TUM dataset and two datasets collected in a corridor and an office using a Kinect camera for the experiments.

Steenbeek and Nex in [85] a real-time VSLAM technique that uses a CNN for accurate scene interpretation and map reconstruction. Their solution utilizes monocular camera

⁹<https://github.com/ivipsourcecode/DS-SLAM>

¹⁰<https://github.com/ivipsourcecode/dxslam>

streams from a UAV during flight and employs a depth-estimating neural network for reliable performance. The mentioned method is based on ORB-SLAM 2.0 and makes use of visual cues collected from indoor environments. Additionally, the CNN is trained on more than 48,000 indoor examples and operates the pose, space depth, and RGB inputs to estimate scale and depth. The TUM RGB-D dataset and a real-world test using a drone were used to evaluate the system, which demonstrated enhanced pose estimation accuracy. However, the system struggles in situations without texture and needs both CPU and GPU resources for real-time performance.

3) Using Artificial Landmarks:

A technique called *UcoSLAM*¹¹ [86] by Muñoz-Salinas and Medina-Carnicer outperforms conventional VSLAM systems by combining natural and man-made landmarks and automatically calculating the scale of the surroundings using fiducial markers. *UcoSLAM*'s primary driving force is to combat natural landmarks' instability, repetition, and poor tracking qualities. It can operate in surroundings without tags or features since it can operate in keypoints-only, markers-only, and mixed modes. To locate map correspondences, optimize re-projection errors, and re-localize in the event of tracking failure, *UcoSLAM* has a tracking mode. Additionally, it has a marker-based loop closure detection system and can describe features using any descriptor, including ORB and FAST. Despite all the plus points of *UcoSLAM*, the system executes in multiple threads, making it a time-consuming approach.

4) Wide-range of Setups:

Another VSLAM strategy for dynamic indoor and outdoor situations is *DMS-SLAM* [87], which supports monocular, stereo, and RGB-D visual sensors. The system employs sliding window and Grid-based Motion Statistics (GMS) [88] feature matching methods to find static feature locations. Using the ORB-SLAM 2.0 system as its foundation, *DMS-SLAM* tracks the static features recognized by the ORB algorithm. The authors tested their suggested methodology on the TUM RGB-D and KITTI datasets and outperformed cutting-edge VSLAM algorithms. Additionally, because the feature points on the dynamic objects were removed during the tracking step, *DMS-SLAM* performs more quickly than the original ORB-SLAM 2.0. Despite the benefits described, the suggested solution encounters difficulties in situations with little texture, fast motion, and highly dynamic environments.

D. Objective IV: Resource Constraint

In another category, some of the VSLAM methodologies are built for devices with limited computational resources compared to other standard devices. For instance, VSLAM systems designed for mobile devices and robots with embedded systems are included in this category.

1) Devices with Limited Processing Capabilities:

In this regard, *edgeSLAM* is a real-time, edge-assisted semantic VSLAM system for mobile and resource-constrained devices proposed by Xu *et al.* [89]. It employs a series of fine-grained

modules to be used by an edge server and the associated mobile devices rather than requiring heavy threads. A semantic segmentation module based on the Mask-RCNN technique is also included in *edgeSLAM* to improve segmentation and object tracking. The authors put their strategy into practice on an edge server with several commercial mobile devices, such as cellphones and development boards. By reusing the findings of the object segmentation, they avoided duplicate processing by adapting system parameters to different network bandwidth and latency situations. *EdgeSLAM* has been evaluated on monocular vision instances of TUM RGB-D, KITTI, and the created dataset for experimental settings.

For stereo camera setups, Schlegel, Colosi, and Grisetti [90] suggested a lightweight feature-based VSLAM framework titled *ProSLAM*¹² that achieves results on par with cutting-edge techniques. Four modules make up their approach: the *triangulation* module, which creates 3D points and associated feature descriptors; the *incremental motion estimation* module, which processes two frames to determine the current position; the *map management* module, which creates local maps; and the *re-localization* module, which updates the world map based on the similarities of local maps. *ProSLAM* retrieves the 3D position of the points using a single thread and leverages a small number of well-known libraries for a system that is simple to create. According to the experiments on KITTI and EuRoC datasets, their approach could achieve robust results. However, it shows weakness in rotation estimation and does not contain any bundle adjustment module.

Bavle *et al.* [91] proposed *VPS-SLAM*¹³, a lightweight graph-based VSLAM framework for aerial robotics. Their real-time system integrates geometrical data, several object detection techniques, and visual/visual-inertial odometry for pose estimation and building the semantic map of the environment. Low-level characteristics, IMU measurements, and high-level planar information are all used by *VPS-SLAM* to reconstruct sparse semantic maps and predict robot states. The system leverages the lightweight version of You Only Look Once v2.0 (YOLO2) [92] trained on the COCO dataset [93] for object detection due to its real-time and computationally effective performance. They used a hand-held camera setup and an aerial robotic platform equipped with an RGB-D camera for testing. The TUM RGB-D dataset's indoor instances were used to test their methodology, and they were able to provide results that were on par with those of well-known VSLAM methods. However, only a small number of objects (*e.g.*, chairs, books, and laptops) can be used by their VSLAM system to build a semantic map of the surrounding area.

Another real-time indoor VSLAM method was proposed by Tseng *et al.* [94] that requires a low-cost setup. The authors also presented a technique for estimating the number of frames and visual elements required for a reasonable degree of localization accuracy. Their solution is based on the OpenVSLAM [95] framework and makes use of it for emergencies that

¹²https://gitlab.com/srrg-software/srrg_proslam

¹³https://github.com/hridaybavle/semantic_slam

¹¹<https://sourceforge.net/projects/ucoslam/>

arise in the real world, such as gaining access to specific targets. The system acquires the scene's feature map for precise pose estimation by applying the Efficient Perspective-n-Point (EPnP) and RANSAC algorithms. According to tests conducted in a building, their device can deliver accurate findings under difficult lighting conditions.

2) Computation Offloading:

Ben Ali *et al.* [96] suggested using edge computing to enable the offloading of resource-intensive operations to the cloud and reduce the computational burden on the robot. They modified the architecture of ORB-SLAM 2.0 in their indirect framework, Edge-SLAM¹⁴, by maintaining the tracking module on the robot and delegating the remainder to the edge. By splitting the VSLAM pipeline between the robot and the edge device, the system can maintain both a local and a global map. With fewer resources available, they could still execute properly without sacrificing accuracy. They used the TUM RGB-D dataset and two different mobile devices to generate a custom indoor environment dataset using RGB-D cameras for evaluation. However, one of their approach's drawbacks is the architecture's complexity due to the decoupling of various SLAM modules. Another setback is that their system works only in short-term settings, and utilizing Edge-SLAM in long-term scenarios (*e.g.*, multiple days) would face performance degradation.

E. Objective V: Versatility

VSLAM works categorized in this class are focused on straightforward development, utilization, adaptation, and extension.

In this regard, Sumikura *et al.* [95] introduced *OpenVSLAM*¹⁵, a highly adaptable open-source VSLAM framework seeks to be quickly developed upon and called by other third-party programs. Their feature-based approach is compatible with multiple camera types, including monocular, stereo, and RGB-D, and can store or reuse the reconstructed maps for later usage. *OpenVSLAM* performs better in terms of tracking accuracy and efficiency than ORB-SLAM and ORB-SLAM 2.0 due to its powerful ORB feature extractor module. However, the open-source code of the system has been discontinued owing to worries over code similarities that infringed on the rights to ORB-SLAM 2.0.

To bridge the gap between real-time capabilities, accuracy, and resilience, Ferrera *et al.* [97] developed *OV²SLAM*¹⁶ that works with monocular and stereo-vision cameras. By limiting the extraction of features to keyframes and monitoring them in subsequent frames through eliminating photo-metric errors, their method lessens the computational load. In this sense, *OV²SLAM* is a hybrid strategy that combines the virtues of the direct and indirect categories of VSLAM algorithms. Using well-known benchmarking datasets including EuRoC, KITTI, and TartanAir in both indoor and outdoor experiments, it

was demonstrated that *OV²SLAM* surpasses several popular techniques in terms of performance and accuracy.

Another approach in this category, titled *DROID-SLAM*¹⁷, a deep learning-based visual SLAM for monocular, stereo, and RGB-D cameras, is proposed by Teed and Deng [98]. They could attain greater accuracy and robustness than well-known monocular and stereo track methods. Their solution operates in real-time and consists of *back-end* (for bundle adjustment) and *front-end* (for keyframe collection and graph optimization) threads. *DROID-SLAM* has already been taught using monocular camera examples, therefore it does not need to be trained again to use stereo and RGB-D inputs. The approach minimizes the projection error, like indirect methods, while not requiring any pre-processing for feature identification and matching. A feature extraction network comprising downsampling layers and residual blocks processes each input image to create dense features. *DROID-SLAM* has been tested on well-known datasets, including TartanAir, EuRoC, and TUM RGB-D, and could achieve acceptable results.

Bonetto *et al.* in [99] propose *iRotate*¹⁸, an active technique for omnidirectional robots with RGB-D cameras. Additionally, a module for spotting obstructions in the camera's area of vision is employed in their approach. By offering observation coverage of previously unexplored places and previously visited locations, *iRotate*'s primary objective is to lessen the distance the robot must go to map the environment. The mentioned method uses a VSLAM framework with graph features as its back-end. The authors could attain outcomes that were on par with those of cutting-edge VSLAM methods by providing comparisons in simulation and on a real three-wheel omnidirectional robot. However, the major weakness of their method is that the robot might face start-stop cases in which the local paths are re-planned.

F. Objective VI: Visual Odometry

Approaches in this category aim to determine the position and orientation of the robot with the highest possible accuracy.

1) Deep Neural Networks:

In this regard, the *Dynamic-SLAM* framework was proposed in [100] that leverages deep learning for accurate pose prediction and suitable environment comprehension. As part of a semantic level module for optimized VO, the authors employed a CNN to identify moving objects in the environment, which helped them lower the pose estimate error brought on by improper feature matching. Additionally, *Dynamic-SLAM* uses a selective tracking module to ignore dynamic locations in the scene and a missed feature corrective algorithm for speed invariance in adjacent frames. Despite the excellent results, the system requires huge computational costs and faces the risk of misclassifying dynamic/static objects due to a limited number of defined semantic classes.

Bloesch *et al.* [101] proposed the *Code-SLAM*¹⁹ direct technique, which offers a condensed and dense representation

¹⁴<https://github.com/droneslab/edgeslam>

¹⁵<https://github.com/xdspacelab/openvslam>

¹⁶<https://github.com/ov2slam/ov2slam>

¹⁷<https://github.com/princeton-vl/DROID-SLAM>

¹⁸https://github.com/eliabntt/irotate_active_slam

¹⁹<https://github.com/silviutroscof/CodeSLAM>

of the scene geometry. Their VSLAM system, which only functions with monocular cameras, is an enhanced version of *PTAM* [14]. They divided intensity images into convolutional features and fed them to a depth auto-encoder using a CNN trained on intensity images from the SceneNet RGB-D dataset. Indoor examples of the EuRoC dataset's have been used to test *Code-SLAM*, and the findings were promising in terms of accuracy and performance.

*DeepVO*²⁰, an end-to-end VO framework using a Deep Recurrent Convolutional Neural Network (RCNN) architecture for monocular settings, was proposed by Wang *et al.* [102]. Their approach uses deep learning to automatically learn the appropriate features, model sequential dynamics and relations, and infer poses directly from color frames. The *DeepVO* architecture includes a CNN called FlowNet for computing optical flow from sequential frames and two Long Short-Term Memory (LSTM) layers for estimating the temporal changes based on the feed provided by the CNN. The framework can simultaneously extract visual characteristics and perform sequential modeling by combining CNN and Recurrent Neural Network (RNN). *DeepVO* can incorporate geometry with the knowledge models learned for an enhanced VO. However, it cannot be utilized to replace conventional geometry-based VO approaches.

Parisotto *et al.* [103] proposed a DeepVO-like end-to-end system using a Neural Graph Optimization (NGO) step instead of LSTMs. Their approach operates a loop closure detection and correction mechanism based on temporally distinct poses. NGO uses two attention-optimization methods to jointly optimize the aggregated predictions made by convolutional layers of local pose estimation modules and delivers global pose estimations. They experimented with their technique on 2D and 3D mazes and outperformed DeepVO's performance and accuracy levels. The mentioned approach needs to be connected to a SLAM framework to supply the re-localization signals.

In another work, one of the most extensive VSLAM frameworks titled *DeepFactors*²¹ introduced by Czarnowski *et al.* [104] for densely rebuilding the environment map with monocular cameras. For a more reliable map reconstruction, their real-time solution performs joint optimization of the pose and depth, makes use of probabilistic data, and combines learned and model-based approaches. The authors modified the *CodeSLAM* framework and added missing components such as local/global loop detection. The system is evaluated on the ICL-NUIM and TUM RGB-D datasets after being trained on roughly 1.4 million ScanNet [105] images. *DeepFactors* improves the idea of the *CodeSLAM* framework and focuses on code optimization in traditional SLAM pipelines. However, due to the computational costs of the modules, this approach requires employing GPUs to guarantee real-time performance.

2) In-depth Adjacent Frame Processing:

In another work, and by reducing the photometric and geomet-

ric errors between two pictures for camera motion detection, the authors of [106]²² developed a real-time dense SLAM approach for RGB-D cameras, improving their prior method, [107]. Their keyframe-based solution expands *Pose SLAM* [108] that only keeps non-redundant poses for producing a compact map, adds dense visual odometry characteristics, and effectively utilizes the information from camera frames for a reliable camera motion estimation. Authors also employed an entropy-based technique to gauge keyframe similarity for loop closure detection and drift avoidance. However, their approach still needs work in the areas of loop closure detection and keyframe selection quality.

In another work introduced by Li *et al.* [109], real-time dynamic object removal is accomplished using a feature-based VSLAM approach known as *DP-SLAM*. This method uses a Bayesian probability propagation model based on the likelihood of the keypoints derived from moving objects. The variation of geometry restrictions and semantic data can be overcome by *DP-SLAM* using the moving probability propagation algorithm and iterative probability updates. It is integrated with ORB-SLAM 2.0 and has been tested on the TUM RGB-D dataset. Despite the accurate results, the system only works in sparse VSLAMs and faces high computational costs because of the iterative probability updater module.

Pair-Navi, a suggested indoor navigation system by Dong *et al.* [110], reuses a previously traced path by an agent for usage in the future by other agents. Hence, a previous traveler known as the *leader* captures the trace information, such as turnings and particular ambient qualities, and gives it to a later *follower* that needs to travel to the same destination. While the *follower* uses a re-localization module to determine its location concerning the reference trace, the *leader* incorporates visual odometry and trajectory creation modules. To recognize and remove dynamic items from the video feature set, the system employs a Mask Region-based CNN (Mask R-CNN). They tested *Pair-Navi* on a set of generated datasets and several smartphones for the experiments.

3) Various Feature Processing:

Another approach in this category is a text-based VSLAM system called *TextSLAM*, proposed by Li *et al.* [111]. It incorporates text items retrieved from the scene using the FAST corner detection technique into the SLAM pipeline. Texts include a variety of textures, patterns, and semantic meanings, making the approach more efficient to use them to create 3D text maps of high quality. *TextSLAM* uses texts as reliable visual fiducial markers, parametrizes them after the first frame in which they are found, and then projects the 3D text object onto the target image to locate it again. They also presented a new three-variable parameterization technique for initializing instantaneous text features. Using a monocular camera and a dataset created by the authors, experiments were conducted in indoor and outdoor settings, and the results were highly accurate. Operating in text-free surroundings, interpret-

²⁰<http://senwang.gitlab.io/DeepVO/>

²¹<https://github.com/jczarnowski/DeepFactors>

²²<https://vision.in.tum.de/data/software/dvo>

ing short letters, and requiring the storage of enormous text dictionaries are the three fundamental challenges of *TextSLAM*.

Xu *et al.* [43] proposed an indirect VSLAM system built upon a modified ORB-SLAM that enables high-accuracy localization and user interaction using the Occupancy Grid Mapping (OGM) method and a new 2D mapping module. Their system can reconstruct the environment map that shows the presence of a barrier as an equally spaced field of variables using the OGM, makes it possible to navigate continuously and in real-time while planning a route. The experimental examination of a generated dataset shows their approach functions in GPS-denied conditions. However, their technique struggles to function well in dynamic and complicated environments and has trouble appropriately matching the features in corridors and featureless conditions.

CPA-SLAM, a direct VSLAM method for RGB-D cameras that makes use of planes for tracking and graph optimization, was proposed by Ma *et al.* [112]. *Frame-to-keyframe* and *frame-to-plane* alignments are regularly integrated in their technique. They also introduced an image alignment algorithm for tracking the camera with respect to a reference keyframe and aligning the image with the planes. The keyframe data is used by *CPA-SLAM*, which looks for the closest short temporal and geographical distances to track. The system's real-time performance of their tracking system was tested in with- and without-plane settings, with analyses performed on TUM RGB-D and ICL-NUIM datasets with indoor and outdoor scenes. However, it only supports a small number of geometric shapes, *i.e.*, planes.

VI. PINPOINTING THE CURRENT TRENDS

A. Statistics

Regarding the classification of the surveyed papers in various aspects presented above, we have visualized the processed data in Fig.4 to find the current trends in VSLAM. In sub-figure "a", We can see that the majority of the proposed VSLAM systems are standalone applications that implement the whole procedure of localization and mapping using visual sensors from scratch. While ORB-SLAM 2.0 and ORB-SLAM are other *base* platforms employed to make a new framework, minimal approaches are based on other VSLAM systems, such as PTAM and PoseSLAM. Moreover, and in terms of the objectives of VSLAM applications, what tops the chart in sub-figure "b" is improving the *Visual Odometry* module. Thus, most of the recent VSLAMs are trying to resolve the problems of current algorithms in determining the position and orientation of the robots. *Pose estimation* and *Real-world viability* are further fundamental objectives of proposing new VSLAM papers. Concerning the dataset used for evaluation in the surveyed papers, sub-figure "c" illustrates that most works have been tested on the *TUM RGB-D* dataset. This dataset has been employed as the primary or one of the multiple baselines for evaluation in the reviewed manuscripts. Additionally, many researchers tend to perform experiments on the datasets *generated* by them. We can assume that the primary motivation for generating a dataset was to show how

a VSLAM method works in real-world scenarios and if it can be used as an end-to-end application. *EuRoC MAV* and *KITTI* are the next popular datasets for evaluation in VSLAM works, respectively. Another interesting information extracted from sub-figure "d" is concerning the impact of employing semantic data when using the VSLAM system. We can see that the majority of the reviewed papers do not include semantic data while processing the environment. We presume the reason behind not utilizing the semantic data are:

- The computational cost for training a model that recognizes objects and utilizing it for semantic segmentation is considerable in many cases, which might raise the processing time.
- The majority of the geometry-based VSLAM works are designed to work as plug-and-play devices so that they can employ camera data for localization and mapping with the least possible effort.
- Incorrect information extracted from the scene can also lead to more added noise to the process.

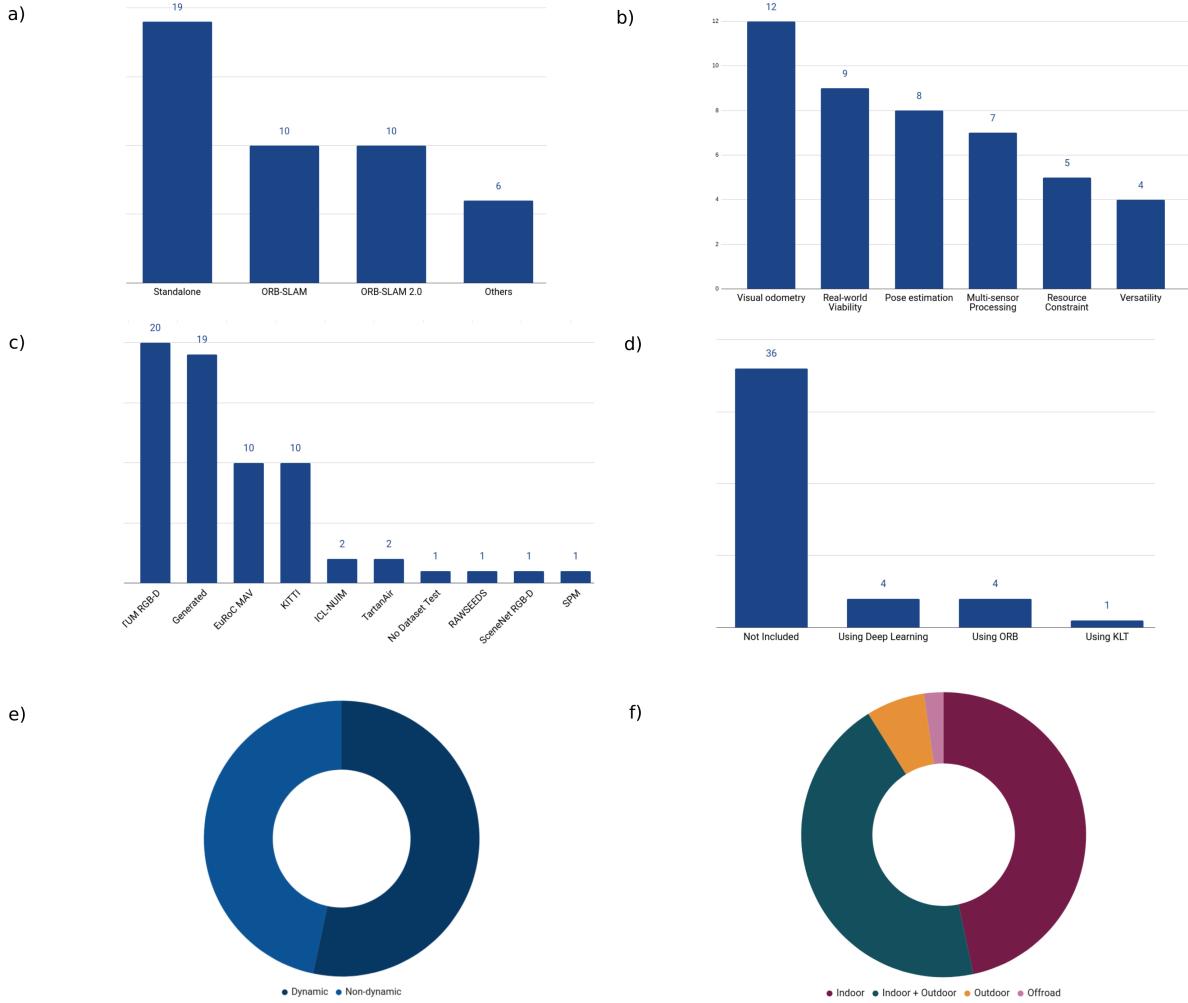
When the environment is considered, we can see in sub-figures "e" that more than half of the approaches can also work in dynamic environments with challenging conditions, while the remaining systems are only focused on environment with no dynamic changes. Moreover, and in sub-figure "f", most of the approaches work in "indoor settings" or "both indoor and outdoor environments," while the rest of the papers have only been tested outdoor conditions. It should be mentioned that approaches that can only work in a particular circumstance with restricted assumptions might not produce the same accuracy if employed in other scenarios. That is one of the main reasons why some approaches only concentrate on a particular situation.

B. Analyzing the Current Trends

The current survey has reviewed the state-of-the-art visual SLAM approaches that have absorbed massive attention and demonstrated their principal contributions in this field. Despite the wide range of reliable solutions and improvements in various modules of VSLAM systems over the past years, there are still many high potential fields and unsolved issues that investigating in them lead to more robust approaches in the future evolution of SLAMs. In light of the wide range of visual SLAM approaches, we hereby propose currently trending areas for investment and introduce the following open research directions:

Deep Learning: deep neural networks have shown encouraging results in various applications, including VSLAM [15], making them a significant trend in multiple fields of study. Due to their learning capabilities, these architectures have shown a considerable potential to be utilized as reliable feature extractors to tackle different issues in VO and loop closure detection. CNNs can aid VSLAMs in precise object detection and semantic segmentation, and can outperform traditional feature extraction and matching algorithms for correctly recognizing hand-crafted features. It has to be mentioned that since deep learning-based methods have been trained on datasets

Fig. 4. Analyzing the current trends of VSLAM approaches: a) base SLAM system employed to implement a new approach, b) the primary objective of the approach, c) various datasets that the proposed methods were testing on, d) the impact of utilizing semantic data in the proposed methods, e) the amount of dynamic objects existing in the environment, f) various types of environments the system tested on.



with large amounts of diversified data and limited object classes, there is always a risk of misclassification of dynamic points and causing false segmentation. Thus, it might lead to lower segmentation accuracy and pose estimation error.

Information Retrieval and Computational Cost Trade-off: generally, the processing cost and the quantity of information in the scene should always be balanced. In this perspective, dense maps allow VSLAM applications to record high-dimensional complete scene information, but doing so in real-time would be computationally demanding. Sparse representations, on the other hand, would fail to capture all the needed information due to their lower computational cost. It should also be noted that real-time performance is directly related to the camera's frame rate, and frame losses in peak processing times can negatively affect the VSLAM system's performance, regardless of algorithms performance. Moreover, VSLAMs typically take advantage of tightly-coupled modules and modifying one module may adversely affect others, which makes the balancing task more challenging.

Semantic Segmentation: Providing semantic information while creating the map of the environment can bring about very useful information for the robot. Identifying objects in the camera's field of view -*e.g., doors, windows, people, etc.*- are a trendy topic in current and future VSLAM works, as the semantic information can be used in pose estimation, trajectory planning, and loop closure detection modules. With the widespread usage of object detection and tracking algorithms, semantic VSLAMs will undoubtedly be among the future solutions in this domain.

Loop Closing Algorithms: One of the critical issues in any SLAM system is the drift problem and losing the feature tracks caused by accumulated localization errors. Detection of drifts and loop closures to identify previously visited places contributes to high computation latency and cost in VSLAM systems [89]. The main reason is that the complexity of loop closure detection increases with the size of the reconstructed map. Moreover, combining the map data collected from various locations and refining the estimated poses are

very complex tasks. With this, optimization and balancing of the loop closure detection module have a massive potential for improvement. One of the common approaches for detecting loop closures is improving image retrieval by training a visual vocabulary based on local features and then aggregating them.

Working in Challenging Scenarios: Working in a texture-less environment with few salient feature points often leads to drift errors in position and orientation in robots. As one of the primary challenges in VSLAM, this error may lead to system failure. Thus, considering complementary scene-understanding methods in feature-based approaches, such as object detection or line features, would be a trendy topic.

VII. CONCLUSIONS

This paper presented a broad range of SLAM works in which visual data collected from cameras play a significant role. We categorized the recent works of VSLAM systems based on various characteristics of their approaches, such as the experimental environment, novelty domain, object detection and tracking algorithms, semantic level viability, performance, etc. We also reviewed the critical contributions of the works and the existing drawbacks and challenges according to the authors' claims, future version improvements, and the issues addressed in other related methods. Another contribution of the paper is discussing the current trends of VSLAM systems and the existing open issues that will be investigated more by researchers.

REFERENCES

- [1] A. R. Khairuddin, M. S. Talib, and H. Haron, "Review on simultaneous localization and mapping (slam)," in *2015 IEEE international conference on control system, computing and engineering (ICCSCE)*. IEEE, 2015, pp. 85–90.
- [2] I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Röning, "Magnetic field-based slam method for solving the localization problem in mobile robot floor-cleaning task," in *2011 15th international conference on advanced robotics (ICAR)*. IEEE, 2011, pp. 198–203.
- [3] Q. Zou, Q. Sun, L. Chen, B. Nie, and Q. Li, "A comparative analysis of lidar slam-based indoor navigation for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- [4] D. Geromichalos, M. Azkarate, E. Tsardoulas, L. Gerdes, L. Petrou, and C. Perez Del Pulgar, "Slam for autonomous planetary rovers with global localization," *Journal of Field Robotics*, vol. 37, no. 5, pp. 830–847, 2020.
- [5] T. Yang, P. Li, H. Zhang, J. Li, and Z. Li, "Monocular vision slam-based uav autonomous landing in emergencies and unknown environments," *Electronics*, vol. 7, no. 5, p. 73, 2018.
- [6] J. Li, Y. Bi, M. Lan, H. Qin, M. Shan, F. Lin, and B. M. Chen, "Real-time simultaneous localization and mapping for uav: A survey," in *Proc. of International micro air vehicle competition and conference*, vol. 2016, 2016, p. 237.
- [7] Z. Liu, H. Chen, H. Di, Y. Tao, J. Gong, G. Xiong, and J. Qi, "Real-time 6d lidar slam in large scale natural terrains for ugv," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 662–667.
- [8] A. Gupta and X. Fernando, "Simultaneous localization and mapping (slam) and data fusion in unmanned aerial vehicles: Recent advances and challenges," *Drones*, vol. 6, no. 4, p. 85, 2022.
- [9] 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.
- [10] M. Zaffar, S. Ehsan, R. Stolkin, and K. M. Maier, "Sensors, slam and long-term autonomy: a review," in *2018 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. IEEE, 2018, pp. 285–290.
- [11] X. Gao and T. Zhang, *Introduction to Visual SLAM: From Theory to Practice*. Springer Nature, 2021.
- [12] M. Filipenko and I. Afanasyev, "Comparison of various slam systems for mobile robot in an indoor environment," in *2018 International Conference on Intelligent Systems (IS)*. IEEE, 2018, pp. 400–407.
- [13] Y.-J. Yeh and H.-Y. Lin, "3d reconstruction and visual slam of indoor scenes for augmented reality application," in *2018 IEEE 14th International Conference on Control and Automation (ICCA)*. IEEE, 2018, pp. 94–99.
- [14] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.
- [15] C. Duan, S. Junginger, J. Huang, K. Jin, and K. Thurow, "Deep Learning for Visual SLAM in Transportation Robotics: A Review," *Transportation Safety and Environment*, vol. 1, no. 3, pp. 177–184, 01 2020. [Online]. Available: <https://doi.org/10.1093/tse/tdz019>
- [16] M. Outahar, G. Moreau, and J.-M. Normand, "Direct and indirect vslam fusion for augmented reality," *Journal of Imaging*, vol. 7, no. 8, p. 141, 2021.
- [17] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.
- [18] M. I. Ribeiro, "Kalman and extended kalman filters: Concept, derivation and properties," *Institute for Systems and Robotics*, vol. 43, p. 46, 2004.
- [19] D. G. Viswanathan, "Features from accelerated segment test (fast)," in *Proceedings of the 10th workshop on image analysis for multimedia interactive services, London, UK*, 2009, pp. 6–8.
- [20] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*. IEEE, 2011, pp. 2320–2327.
- [21] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-d mapping with an rgb-d camera," *IEEE transactions on robotics*, vol. 30, no. 1, pp. 177–187, 2013.
- [22] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.
- [23] E. Mendes, P. Koch, and S. Lacroix, "Icp-based pose-graph slam," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2016, pp. 195–200.
- [24] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 15–22.
- [25] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.
- [26] 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.
- [27] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgbd cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
- [28] 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*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [29] 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*, 2017, pp. 6243–6252.
- [30] 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.
- [31] A. Macario Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A comprehensive survey of visual slam algorithms," *Robotics*, vol. 11, no. 1, p. 24, 2022.
- [32] W. Chen, G. Shang, A. Ji, C. Zhou, X. Wang, C. Xu, Z. Li, and K. Hu, "An overview on visual slam: From tradition to semantic," *Remote Sensing*, vol. 14, no. 13, p. 3010, 2022.
- [33] Y. Jia, X. Yan, and Y. Xu, "A survey of simultaneous localization and mapping for robot," in *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, vol. 1. IEEE, 2019, pp. 857–861.

- [34] I. A. Kazerouni, L. Fitzgerald, G. Dooly, and D. Toal, "A survey of state-of-the-art on visual slam," *Expert Systems with Applications*, p. 117734, 2022.
- [35] H. Bavle, J. L. Sanchez-Lopez, E. F. Schmidt, and H. Voos, "From slam to situational awareness: Challenges and survey," *arXiv preprint arXiv:2110.00273*, 2021.
- [36] S. Arshad and G.-W. Kim, "Role of deep learning in loop closure detection for visual and lidar slam: A survey," *Sensors*, vol. 21, no. 4, p. 1243, 2021.
- [37] A. Singandhupe and H. M. La, "A review of slam techniques and security in autonomous driving," in *2019 third IEEE international conference on robotic computing (IRC)*. IEEE, 2019, pp. 602–607.
- [38] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual slam and structure from motion in dynamic environments: A survey," *ACM Comput. Surv.*, vol. 51, no. 2, feb 2018. [Online]. Available: <https://doi.org/10.1145/3177853>
- [39] M. He, C. Zhu, Q. Huang, B. Ren, and J. Liu, "A review of monocular visual odometry," *The Visual Computer*, vol. 36, no. 5, pp. 1053–1065, 2020.
- [40] 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.
- [41] J. Jiao, H. Huang, L. Li, Z. He, Y. Zhu, and M. Liu, "Comparing representations in tracking for event camera-based slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1369–1376.
- [42] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 994–1001, 2018.
- [43] L. Xu, C. Feng, V. R. Kamat, and C. C. Menassa, "An occupancy grid mapping enhanced visual slam for real-time locating applications in indoor gps-denied environments," *Automation in Construction*, vol. 104, pp. 230–245, 2019.
- [44] 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.
- [45] L. Cui and C. Ma, "Sof-slam: A semantic visual slam for dynamic environments," *IEEE access*, vol. 7, pp. 166 528–166 539, 2019.
- [46] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [47] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.
- [48] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*. Springer, 2010, pp. 778–792.
- [49] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.
- [50] E. Karami, S. Prasad, and M. Shehata, "Image matching using sift, surf, brief and orb: performance comparison for distorted images," *arXiv preprint arXiv:1710.02726*, 2017.
- [51] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos, "Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets," in *In proceedings of IROS*, vol. 6, 2006, p. 93.
- [52] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation?" in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2678–2687.
- [53] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580.
- [54] T.-M. Nguyen, S. Yuan, M. Cao, Y. Lyu, T. H. Nguyen, and L. Xie, "Ntu viral: A visual-inertial-ranging-lidar dataset, from an aerial vehicle viewpoint," *International Journal of Robotics Research*, 2021.
- [55] 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.
- [56] X. Shi, D. Li, P. Zhao, Q. Tian, Y. Tian, Q. Long, C. Zhu, J. Song, F. Qiao, L. Song *et al.*, "Are we ready for service robots? the openloris-scene datasets for lifelong slam," in *2020 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2020, pp. 3139–3145.
- [57] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.
- [58] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4909–4916.
- [59] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, "A benchmark for rgb-d visual odometry, 3d reconstruction and slam," in *2014 IEEE international conference on Robotics and automation (ICRA)*. IEEE, 2014, pp. 1524–1531.
- [60] E. Mueggler, H. Rebecq, G. Gallego, T. Delbrück, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.
- [61] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Dsslam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1168–1174.
- [62] S. Wen, P. Li, Y. Zhao, H. Zhang, F. Sun, and Z. Wang, "Semantic visual slam in dynamic environment," *Autonomous Robots*, vol. 45, no. 4, pp. 493–504, 2021.
- [63] D. Zou and P. Tan, "Coslam: Collaborative visual slam in dynamic environments," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 2, pp. 354–366, 2012.
- [64] Y. Yang, D. Tang, D. Wang, W. Song, J. Wang, and M. Fu, "Multi-camera visual slam for off-road navigation," *Robotics and Autonomous Systems*, vol. 128, p. 103505, 2020.
- [65] S. Urban and S. Hinz, "Multicol-slam-a modular real-time multi-camera slam system," *arXiv preprint arXiv:1610.07336*, 2016.
- [66] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "Camvox: A low-cost and accurate lidar-assisted visual slam system," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5049–5055.
- [67] T.-M. Nguyen, S. Yuan, M. Cao, T. H. Nguyen, and L. Xie, "Viral slam: Tightly coupled camera-imu-uwb-lidar slam," *arXiv preprint arXiv:2105.03296*, 2021.
- [68] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *British Machine Vision Conference*. University of Zurich, 2017, pp. 1–8.
- [69] T. H. Nguyen, T.-M. Nguyen, and L. Xie, "Tightly-coupled ultra-wideband-aided monocular visual slam with degenerate anchor configurations," *Autonomous Robots*, vol. 44, no. 8, pp. 1519–1534, 2020.
- [70] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, "Structslam: Visual slam with building structure lines," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1364–1375, 2015.
- [71] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Pl-slam: Real-time monocular visual slam with points and lines," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 4503–4508.
- [72] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "Pl-slam: A stereo slam system through the combination of points and line segments," *IEEE Transactions on Robotics*, vol. 35, no. 3, pp. 734–746, 2019.
- [73] H. Lim, Y. Kim, K. Jung, S. Hu, and H. Myung, "Avoiding degeneracy for monocular visual slam with point and line features," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 675–11 681.
- [74] S. Bultmann, K. Li, and U. D. Hanebeck, "Stereo visual slam based on unscented dual quaternion filtering," in *2019 22th International Conference on Information Fusion (FUSION)*. IEEE, 2019, pp. 1–8.
- [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] 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.
- [77] 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.
- [78] Q. Peng, Z. Xiang, Y. Fan, T. Zhao, and X. Zhao, "Rwt-slam: Robust visual slam for highly weak-textured environments," *arXiv preprint arXiv:2207.03539*, 2022.
- [79] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou, "Loftr: Detector-free local feature matching with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 8922–8931.
- [80] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.
- [81] 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.
- [82] S. Yang, C. Zhao, Z. Wu, Y. Wang, G. Wang, and D. Li, "Visual slam based on semantic segmentation and geometric constraints for dynamic indoor environments," *IEEE Access*, vol. 10, pp. 69 636–69 649, 2022.
- [83] 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)*. IEEE, 2020, pp. 4958–4965.
- [84] G. Li, L. Yu, and S. Fei, "A deep-learning real-time visual slam system based on multi-task feature extraction network and self-supervised feature points," *Measurement*, vol. 168, p. 108403, 2021.
- [85] A. Steenbeek and F. Nex, "Cnn-based dense monocular visual slam for real-time uav exploration in emergency conditions," *Drones*, vol. 6, no. 3, p. 79, 2022.
- [86] R. Muñoz-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.
- [87] 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.
- [88] J. Bian, W.-Y. Lin, Y. Matsushita, S.-K. Yeung, T.-D. Nguyen, and M.-M. Cheng, "Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4181–4190.
- [89] J. Xu, H. Cao, D. Li, K. Huang, C. Qian, L. Shangguan, and Z. Yang, "Edge assisted mobile semantic visual slam," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1828–1837.
- [90] D. Schlegel, M. Colosi, and G. Grisetti, "Proslam: Graph slam from a programmer's perspective," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3833–3840.
- [91] H. Bavle, P. De La Puente, J. P. How, and P. Campoy, "Vps-slam: Visual planar semantic slam for aerial robotic systems," *IEEE Access*, vol. 8, pp. 60 704–60 718, 2020.
- [92] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.
- [93] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [94] P.-Y. Tseng, J. J. Lin, Y.-C. Chan, and A. Y. Chen, "Real-time indoor localization with visual slam for in-building emergency response," *Automation in Construction*, vol. 140, p. 104319, 2022.
- [95] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: a versatile visual slam framework," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2292–2295.
- [96] A. J. Ben Ali, Z. S. Hashemifar, and K. Dantu, "Edge-slam: edge-assisted visual simultaneous localization and mapping," in *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, 2020, pp. 325–337.
- [97] M. Ferrera, A. Eudes, J. Moras, M. Sanfourche, and G. Le Besnerais, "Ov²slam: A fully online and versatile visual slam for real-time applications," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1399–1406, 2021.
- [98] J. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in Neural Information Processing Systems*, vol. 34, 2021.
- [99] E. Bonetto, P. Goldschmid, M. Pabst, M. J. Black, and A. Ahmad, "Irotoate: Active visual slam for omnidirectional robots," *Robotics and Autonomous Systems*, vol. 154, p. 104102, 2022.
- [100] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, "Dynamic-slam: Semantic monocular visual localization and mapping based on deep learning in dynamic environment," *Robotics and Autonomous Systems*, vol. 117, pp. 1–16, 2019.
- [101] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "Codeslam—learning a compact, optimisable representation for dense visual slam," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2560–2568.
- [102] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2043–2050.
- [103] E. Parisotto, D. Singh Chaplot, J. Zhang, and R. Salakhutdinov, "Global pose estimation with an attention-based recurrent network," in *Proceedings of the ieee conference on computer vision and pattern recognition workshops*, 2018, pp. 237–246.
- [104] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison, "Deepfactors: Real-time probabilistic dense monocular slam," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 721–728, 2020.
- [105] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.
- [106] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 2100–2106.
- [107] ———, "Robust odometry estimation for rgb-d cameras," in *2013 IEEE international conference on robotics and automation*, 2013, pp. 3748–3754.
- [108] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-based compact pose slam," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 78–93, 2009.
- [109] A. Li, J. Wang, M. Xu, and Z. Chen, "Dp-slam: A visual slam with moving probability towards dynamic environments," *Information Sciences*, vol. 556, pp. 128–142, 2021.
- [110] E. Dong, J. Xu, C. Wu, Y. Liu, and Z. Yang, "Pair-navi: Peer-to-peer indoor navigation with mobile visual slam," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1189–1197.
- [111] B. Li, D. Zou, D. Sartori, L. Pei, and W. Yu, "Textslam: Visual slam with planar text features," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2102–2108.
- [112] L. Ma, C. Kerl, J. Stückler, and D. Cremers, "Cpa-slam: Consistent plane-model alignment for direct rgb-d slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1285–1291.