The Simultaneous Localization and Mapping (SLAM)-An Overview

Abstract

Positioning is needed for many applications related to mapping and navigation, either in civilian or military domains. The significant developments in satellite-based techniques, sensors, telecommunications, computer hardware and software, image processing, etc. positively influenced solving the positioning problem efficiently and instantaneously. Accordingly, the mentioned development empowered the applications and advancement of autonomous navigation. One of the most interestingly developed positioning techniques is what is called in robotics Simultaneous Localization and Mapping (SLAM). The SLAM problem solution has witnessed a quick improvement in the last decades, either using active sensors like the RAdio Detection and Ranging (Radar) and Light Detection and Ranging (LiDAR) or passive sensors like cameras. Definitely, positioning and mapping is one of the main tasks for geomatics engineers, and therefore it's of high importance for them to understand the SLAM topic, which is not easy because of the huge documentation and algorithms available and the various SLAM solutions in terms of the mathematical models, complexity, the sensors used, and the type of applications. In this paper, a clear and simplified explanation of SLAM from a geometrical viewpoint is introduced, avoiding going into the complicated algorithmic details behind the presented techniques. In this way, a general overview of SLAM is presented, showing the relationship between its different components and stages, like the core part of the front-end and back-end, and their relation to the SLAM paradigm. Furthermore, we explain the major mathematical techniques of filtering and pose graph optimization, either using visual or LiDAR SLAM, and introduce a summary of the efficient contribution of deep learning to the SLAM problem. Finally, we address examples of some existing practical applications of SLAM in our reality.

Keywords: *SLAM, Visual Odometry, Graph Pose Optimization, Extended Kalman Filter, Deep Learning.*

I. Introduction

SLAM is an acronym for Simultaneous Localization And Mapping which is a technology that enables a robot to map an unknown environment and position itself based on the built map at the same time [1] and frequently with the absence of exterior positioning systems such as Global Navigation Satellite System GNSS [2]. The starting point for SLAM was at the third IEEE International Conference on Robotics and Automation which was held in San Francisco, in 1986. While the first use of the term “SLAM” was at the Seventh International Symposium of Robotics Research which was held in Munich, Germany, in 1995 by Durrant-Whyte, et al. [3]. Nowadays, SLAM is a major factor behind autonomous unmanned aerial vehicles (UAVs), unmanned ground vehicles (UGVs), self-driving cars, augmented/virtual reality, and various autonomous indoor and outdoor mobile mapping applications [4, 5]. To accomplish this SLAM step, the robot should have on-board sensors that apply several measurements along the robot trajectory. Major robot sensors used can be the camera, LiDAR, GNSS receiver/antenna, and the Inertial Measurement Unit IMU. When the SLAM algorithm is based on camera sensors it is called visual SLAM, while when based on laser scanners is called LiDAR SLAM [6, 7].

One of the important sensors mounted on the robot is the IMU, which is used to measure the linear and rotational acceleration of the robots. In more detail, IMU is a

combination of triaxial accelerometers that measure dynamic acceleration and gravity and triaxial gyroscopes measure angular velocity.

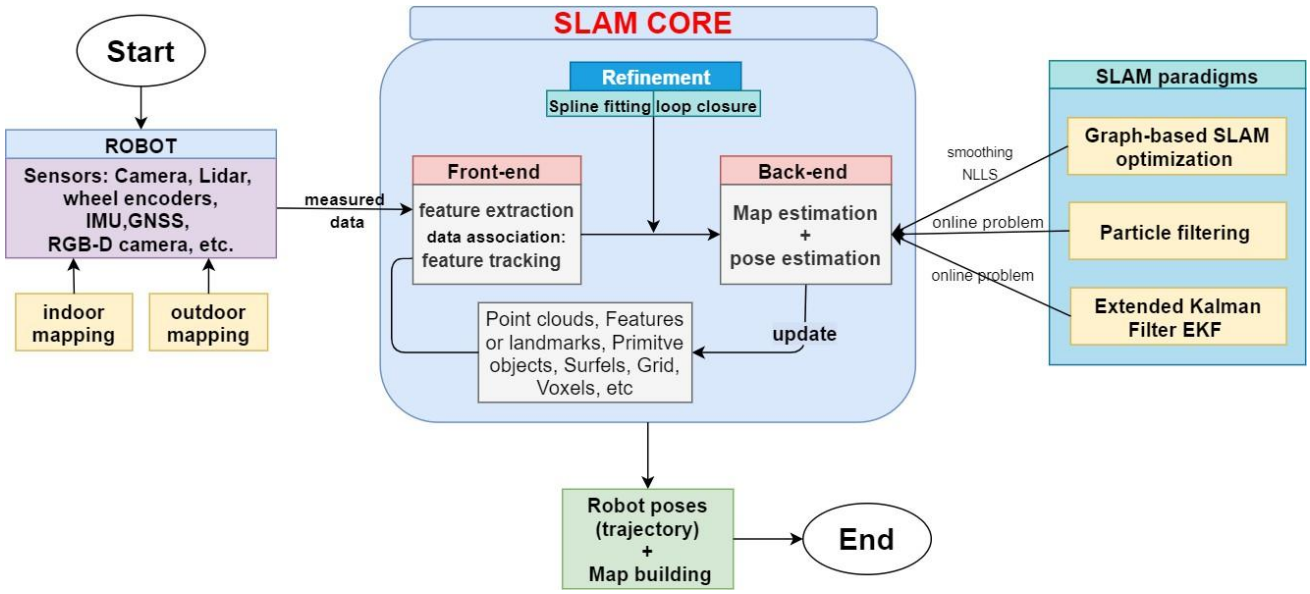

Fig.1. SLAM techniques overview.

Sometimes triaxial magnetometers are integrated with them. Accordingly, the speed of the robot and the traveled distance can be estimated from the IMU measurements which can be used for odometry. However, the IMU-based navigation suffers from accumulated error over time since the positions are computed using the dead reckoning method [8]. Therefore, the IMU in robots manufactured for use outdoors is usually integrated with GNSS, which permits the acquisition of the absolute position to a high level of accuracy in open-sky environments.

This GNSS measurement provides direct information about the location of the robot independently from the previous location estimates and therefore no positional accumulated errors occurred further [4]. Nevertheless, the integration of the GNSS with the IMU leads to a more robust navigation system, especially in urban environments where the quality of GNSS signals sometimes degrades or even outages in blocked areas [9]. Consequently, SLAM is a challenging problem either for indoor mapping applications or GNSS denied outdoor environments like in urban canyon and forests where sophisticated techniques are required for having a reliable localization and mapping solution.

The traditional solution for SLAM at GNSS denied areas is to fix landmarks (reference points or control points) that can be easily identified from the robot sensors. However, this approach is costly and difficult to be used in large-scale environments.

The core implementation of a SLAM system incorporates two main components: the front-end and the back-end as shown in Figure 1. In the front-end component, the detection and tracking of features from imaging sensors (Visual SLAM) either monocular or stereoscopic can be applied [10, 11]. This is very well known in computer vision and photogrammetry and called the structure from motion SfM technique [12]. The corresponding features in consecutive images/scans are associated; this is the so-called data association.

In LiDAR SLAM, the front end step is applied by scanning the environment attained from a moving robot [13, 14] and the successive point clouds are coregistered using the well-known scan matching techniques like the iterative closets point ICP [15, 16].

Hence, features can be sparse as landmarks or dense as point clouds depending on the sensor type and technique used for the feature detection and tracking. Whenever the robot moves, new landmarks or point clouds are detected and tracked. The features can also be lines [17], planes [35], or surfels [18] extracted from the LiDAR data.

However, the performance of these feature-based SLAM algorithms is mainly based on the success of the detection and tracking methods within the front-end step. For instance, image-based SLAM will fail in textureless areas and provide a low ability to deal with poorly textured ones, and planar feature-based SLAM will fail in environments that lack planar structures.

Therefore the outliers that can be found in this step may highly mislead the following step of the back end where the pose of the robot and the location of the landmarks is estimated.

The back-end can be applied using complex mathematical techniques either using filtering or smoothing techniques as will be explained in the next section and involved mainly with

the estimation or update of the landmarks and the robot positions in a reference coordinate system [13].

In this paper, we provide an overview of the SLAM solution showing the relationship between its different components and stages. We address the general problem from different types of data and illustrate the different techniques with the advantages and disadvantages of each. Finally, we list some existing practical SLAM-based applications in our reality. The paper can serve as a tutorial for SLAM users or even non-specialized readers.

The remainder of this paper is structured as follows: In the following section, we present the SLAM paradigm. In Section 3, we commence with a brief overview of SLAM in the 2D and 3D space domain, then we proceed with SLAM enhancement techniques in Section 4. Section 5 presents SLAM sensor-based techniques. The different map representations in SLAM are discussed in section 6. Section 7 introduces a summary of the deep learning efficient contribution to the SLAM. Next, some different SLAM-based applications in our reality are listed in Section 8. Finally, the paper ends with conclusions in Section 9.

II. SLAM Paradigm

SLAM should work perfectly in a well-identified environment with available high certainty in Robot positions. However, in the real-life scenario, SLAM must deal with high uncertainties in the surroundings and with imperfect knowledge of the Robot positions. Accordingly, the SLAM problem is generally defined employing probabilistic tools because of the inherent sensor measurement noise [11].

Currently, many solutions are found to the SLAM problem which can be classified either as filtering or smoothing approaches (Fig.1). Filtering approaches are more suitable for on-line robot state and map estimation. The estimate is supplemented and refined by immediate integration of the new sensor measurements as the robot moves. Techniques like Kalman filters [19] and particle filters [20] are a major example of this filtering SLAM type and are typically designed as on-line SLAM techniques.

Filtering approaches are applied in two main steps: a prediction step and an update step. Generally, they are considered as a maximum a posterior (MAP) method in which measurements from sensors like the IMUs are used to estimate the prior distribution of the robot pose. The IMU measurements are combined with the measurements taken mostly by a camera or a LiDAR to build the likelihood distribution [21]. In a typical SLAM filtering approach, the IMU sensor measurements are used in the prediction step to predict the motion of the vehicle (odometry) [22]. While the measured features on images and the estimated camera pose are used as a likelihood distribution to update the predictions in the update step [21]. Filtering approaches will be more explained in the next section.

On the other hand, smoothing approaches like the graph SLAM, estimate the full robot trajectory by processing the full set of the sensor measurements. These smoothing approaches are categorized as the full SLAM problem, and they normally rely on least-squares adjustment techniques and optimization [23, 24] which is considered as an advantage over the filtering process in terms of accuracy. On the other hand, the main disadvantage of the graph SLAM is the high memory consumption as it combines all the pose estimates in the computation procedure. While procedures like the Kalman filtering consider the last pose only and the motion model [25] to enable an online implementation and use the loop closure to increase the accuracy.

In both filtering and smoothing approaches, it is now clear to the reader the significant amount of mathematical formulation required to have a final reliable robot poses and constructed map [11]. This expensive computing cost is a challenge when executing SLAM on robot hardware. Computation is usually performed on limited processing power microprocessors while to achieve accurate SLAM localization, it is important to execute either image processing or point cloud alignment at a high rate [26]. Furthermore, optimization calculations like loop closure are costly computational processes. Altogether, it is a challenge to execute such computationally expensive processing on robot microcomputers. Different solutions are proposed to overcome this problem like parallel processing, using multi-core CPUs, or embedded GPUs to improve the processing speeds [26].

A) Extended Kalman Filter EKF

The Extended Kalman Filter EKF technique is based on tracking a Gaussian belief of the robot and assumes all the measurements have a Gaussian noise behavior. EKF can be applied in the following main steps: predict state, predict measurement, apply the real measurements, associate the data, and finally update. In SLAM, EKF determines the position and orientation of a robot by verifying its state $\hat{x}_k$ and its uncertainty P_k from the noisy IMU measurements. Then the real measurement information captured by the camera or the LiDAR is integrated to improve the state prediction (pose) in an updated step $\hat{x}_k^+$. The updated pose state and its uncertainty will be fed back as ($\hat{x}_{k-1}$, P_{k-1}) for modeling a new prediction and update (Fig.2). Further reading about the EKF computational approach can be found in [27].

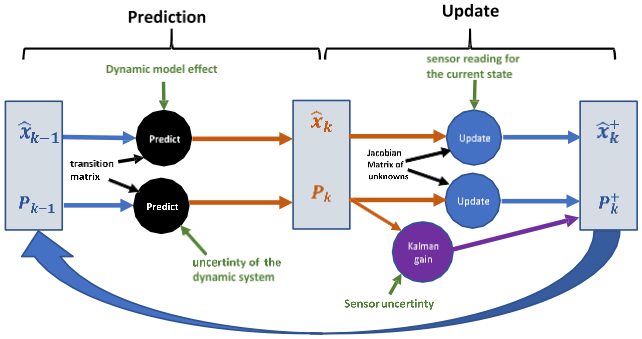

Fig. 2. General workflow of Kalman filter [27].

The formulating of the Kalman filter can be summarized as follows where we have two Gaussian distributions [27]:

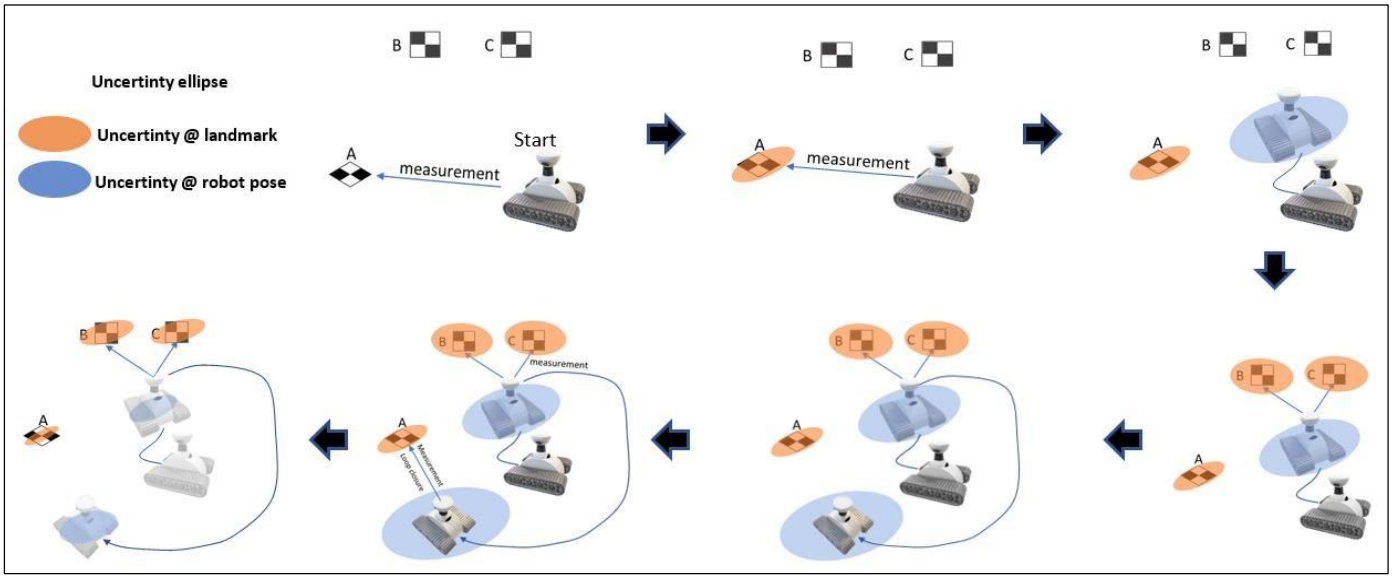

Fig.3. Illustration of the prediction, uncertainty propagation, and update using filtering SLAM

1. The predicted measurement $B_k\hat{x}_k$ and its uncertainty with $B_k\hat{x}_k B_k^t$ where B is the jacobian matrix.
2. The observed measurement $(\vec{z}_k, R_k)$ where the current sensor state is $\vec{z}_k$ and its uncertainty R_k.

The Kalman gain matrix is computed as:

$$FK = B_k P_k B_k^t (B_k P_k B_k^t + R_k)^{-1} \quad (1)$$

Then, the update state calculations can be applied as:

$$K' = P_k B_k^t (B_k P_k B_k^t + R_k)^{-1} \quad (2)$$

$$\hat{x}_k^+ = x_k + K'(\vec{z}_k - B_k\hat{x}_k) \quad (3)$$

$$P_k^+ = P_k - K' B_k P_k \quad (4)$$

The mentioned equations are used to update the system in a repetitive way where $\hat{x}_k^+$ is the new estimate and together with P_k^+ are replaced back into a new iteration of prediction and continue until a stopping criterion is satisfied.

In Fig. 3, the concept of online SLAM using filtering is shown where SLAM estimates the most recent robot pose state based upon the previous states. A sequence of illustrations is given where the robot starts measuring landmark A using a mounted sensor [20]. At the start location, a zero uncertainty at the robot pose is assumed while predicting an uncertainty value at the landmark as propagated by the measurement uncertainty. Then when the robot moves, its pose is calculated with a Gaussian-based uncertainty. Two other landmarks B and C are measured and their positions with the associated uncertainty are estimated. The robot continues its movement with an updated predicted pose and uncertainty and when detects a revisited feature (landmark A) a loop closure calculations are applied. Thus, the uncertainty in the robot poses and landmarks improves significantly.

Ullah, et al. [28] have developed two SLAM algorithms for robot localization, the first one is based on the linear KF and the second one is based on the EKF. Although EKF is one of the most common filtering techniques, it has some disadvantages like being difficult to implement in practice. Moreover, it is not a very accurate method for complicated nonlinear systems or with high uncertainty problems [21].

B) Graph SLAM

A widely used SLAM technique is the graph formulation which involves constructing a graph with connected nodes (Fig.4). Every node represents a robot pose or a measured landmark and in which the edges between the nodes represent the sensor measurement that constrains the connected poses [29]. After completing the graph construction, the essential problem is to find the optimal alignment of the nodes that is maximally consistent with the measurements (smoothing). As a result, this requires solving a large-scale constrained minimization problem. Accordingly, the graph SLAM problem is divided into two tasks: 1) graph construction and 2) graph optimization [29].

The constraints (edges) are constructed based on the sensor measurements either as odometry measurements between subsequent robot locations or computed from the alignment of images or laser scans captured at two different robot poses. Afterward, optimization should be applied to find the best graph configuration that satisfies the constraints [29]. However, this optimization task is challenging to implement on a long navigation trajectory dealing with a large system of nonlinear equations [14]. However, the sparse structure of the matrices can allow cost-effective algorithms to perform this SLAM global optimization. Currently, few open source libraries are available to deal with the solution of large-scale sparse systems like the g2o library [30], Ceres Solver [31], or the GTSAM [32].

In visual SLAM, the graph SLAM optimization is similar to the well know photogrammetric Bundle adjustment where the positions of the observed feature points (landmarks) are estimated simultaneously with robot poses as shown in Fig.4 [33]. It should be noted that in some applications the positions of the landmarks are fixed a priori (control points), and then SLAM may not be needed if the localization can be done reliably concerning the known landmarks [2].

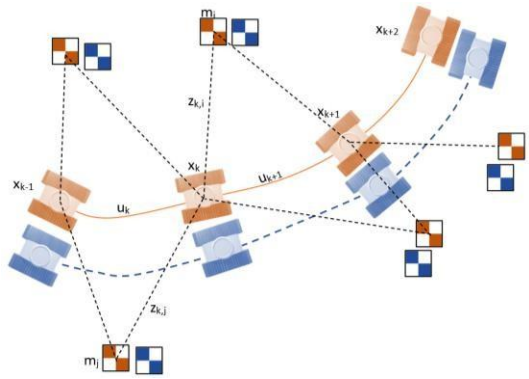

Fig.4. Graph SLAM concept. Orange represents true locations and blue represents the estimated locations.

Fig. 4 illustrates the variables that define the graph which consist of:

x_k : the state vector describing the pose of the vehicle at time k.
u_k : the motion control vector applied the time k-1.
m_i : a vector describing the location of the ith fixed landmark.
$z_{k,i}$: a sensor measurement is taken from the robot to the jth landmark at time k.

Consequently, the objective of a maximum likelihood approach is to find the optimal configuration of the graph nodes (x variables) that minimizes $F(x)$ all the observations z [29]:

$$F(x) = \arg\min \sum_{\langle i,j\rangle \epsilon C} v_{ij}^{t} \Omega_{ij} v_{ij} \qquad (5)$$

where v_{ij} indicates the residual errors or the difference between the projected observation $\hat{z}_{ij}$ and real observation z_{ij} measured by the robot sensor. These observations z_{ij} includes the position and orientation information. Ω_{ij} represent the information matrix which is also called the weight matrix.
The minimization problem is normally solved using the nonlinear least-squares adjustment using either the Gauss-Newton or the Levenberg-Marquardt methods. However, for large-scale problems, this optimization may imply a memory consumption that grows quadratically in the number of variables [2].

III. SLAM IN 2D AND 3D SPACE DOMAIN

SLAM can also be classified into 2D and 3D SLAM as described in the following subsections.

A) 2D SLAM

2D SLAM is three degrees of freedom (3DOF) process, namely position (x, y) and orientation (yaw). It is based on the assumption that the robot moves in a plane, thus 2D SLAM establishes a 2D map of the surrounding area and provides a 2D position and orientation of the robot in this map. The existence of many 2D SLAM algorithms was a motivation to study and analyze them to guide the interested researchers in either improving or innovating their algorithms. For example, Kümmerle, et al. [34] compared several SLAM algorithms based on their output trajectories. The availability of ground truth in the comparison method is not necessary as well as the analyzed algorithms may use different techniques and sensors.

Different datasets were used in the study in order to determine the level of generality of each algorithm. Moreover, they have provided an objective benchmark dataset to help other researchers in the mapping field in testing and evaluating their algorithms. A subsequent study on the pros and cons of the available 2D SLAM algorithms until 2013 has been conducted by Santos, et al. [35]. The evaluation process is mainly based on the quality of the output map instead of the trajectory. They have chosen five 2D LiDAR SLAM techniques to be tested under the same conditions. Those techniques were HectorSLAM, Gmapping, KartoSLAM, CoreSLAM, and LagoSLAM. All the mentioned techniques were implemented using the Robot Operating System (ROS). which is a prominent framework that enables researchers in the robotic field to execute their algorithms [36]. Several 2D simulations and real-world tests are conducted for the evaluation. For the simulation experiments in MRL Arena (4.57×4.04 m), Gmapping and HectorSLAM perform quite better than others in generating the map (~0.4 cm error), while the generated map by CoreSLAM has the highest error (~ 11.8 cm). In the real world MRL Arena, KartoSLAM generates the map with the lowest error (~1.03 cm) and in contrast to others, it does not vary too much than its error in the simulation test (~0.55 cm). Overall, in the real world environment all techniques provide less accurate results than the virtual environment.

Google Cartographer [37] is one of the most recent 2D SLAM-based systems. The Viametris i-MMS system employs an online 2D SLAM for positioning in indoor environments [38]. Li, et al. [39] designed 2D SLAM-based navigation system that utilizes GNSS/IMU integration to navigate outdoors and IMU/LiDAR integration for indoor areas. Recently, the 2D LiDAR SLAM-based mobile robot has achieved success in indoor rescue missions [40].

A) 3D SLAM

After the successful implementations of 2D SLAM in mapping an environment and localizing the mapping system in 2D space, many researchers have turned towards study the applicability of 3D SLAM [41]. One of the main reasons behind this trend is the significant changes in roll and pitch values of the mapping system while on the move. 3D SLAM is six degrees of freedom (6DOF) process, namely position (x, y, z) and orientation (roll, pitch, yaw). In comparison with 2D SLAM, it is a more complex and highly computationally intensive process, but it is more efficient to model the general

motion of a platform. In case of a limited motion to a 2D space, 2D SLAM is sufficient for localization, and a perpendicular sensor is used for mapping in the third dimension [42]. They have used the horizontal laser range finder (Sick) to estimate the location within the plane while the vertical one supplies the system with the needed information for 3D mapping in the indoor environment. Weingarten and Siegwart [43] rotated the 2D laser range finder (Sick) to generate a 3D point cloud of an indoor environment that feeds 3D SLAM. Some systems that employ 3D SLAM have been designed to explore the 3D space in some cases like human navigation and rescue operations [44]. Recently, the 3D SLAM becomes indispensable to operate mobile service robots in unknown environments [45].

IV. SLAM Enhancement techniques

Two enhancement techniques are normally applied to refine the SLAM performance namely loop closure and trajectory interpolation.

A) Loop Closure

Similar to the well-known technique used in traversing and geodetic network adjustment in Geomatics, the last step in SLAM is to apply a loop closure. This is the final refinement step to have a globally consistent SLAM solution, particularly over long trajectories. This is necessary even when using highly accurate sensors because they are still prone to some amount of random uncertainty and will lead to a trajectory drift [46], which in turn, can result in a significant misclosure at the end of a loop.

Accordingly, loop closure is the process of revisiting the same stored mapping area by either new image frames or LiDAR scans and connecting between them by a constraint. This SLAM front-end step of loop closure will significantly reduce the accumulated drift in the final estimated map and robot poses (Fig.5) [4, 29].

A powerful computational approach is required to match features in the new images or scans concerning all the previously detected features in real-time which is impractical and consumes memory especially over long trajectories [4]. Therefore, some techniques like the Bags of Words technique are initialized to tackle this issue [2].

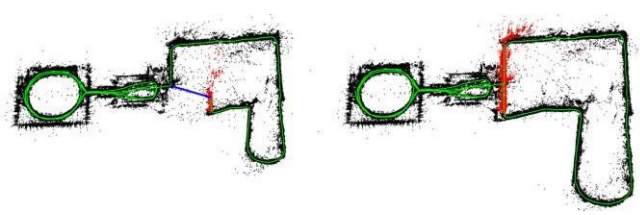

Fig. 5. Loop closure illustration. Left: before loop closure. Right after loop closure. [13]

To validate the loop closure, additional geometric verification steps are needed to determine their quality. In visual SLAM applications, geometric verification and outlier rejection are applied using RANSAC [47]. While in laser-based SLAM approaches, loop closure can be tested by the goodness of the alignment between the current laser scan point cloud and the previously scanned point cloud [2]. This can be measured by checking the histogram of point cloud normal as a descriptor for achieving loop closure [33].

B) Continuous robot trajectory interpolation

As described earlier, SLAM is applied using several sensors onboard the robot like the IMUs, LiDARs, cameras, odometers, and GNSS receivers/antenna. Accordingly, every sensor may operate at a different frequency than the other sensors. One solution is to consider the trajectory of the robot as a continuous function of time either as a nonparametric Gaussian method or as a spline function [46].

A common preference is a cubic B-spline [48] where a sequence of spline time spaced knots can be stored. At every spline knot, the pose is defined by the three rotations $(\omega, \varphi, k)$and three translations (T_X, T_Y, T_Z). The modelling of e.g. $\omega(t)$ by a B-spline function is given in Equation (2) [49].

$$\omega(t) = \sum_i \alpha_{\omega,i} B_i(t) \qquad (6)$$

where $\alpha_{\omega,i}$is the spline coefficient for angle ω to be estimated on interval i.

Then the trajectory is defined at any time by computing the weighted sum of the four closest knots [46]. Thus, a total of 10 variables per second is required for the optimization which will reduce the size of the variables to be estimated and the trajectory will be smooth without motion distortions (Fig.6).

V. SLAM Sensor-based techniques

SLAM can be classified into two main techniques concerning the map measuring sensor which is either by using a camera or a LiDAR. Accordingly, two terminologies are found: Visual SLAM and LiDAR SLAM.

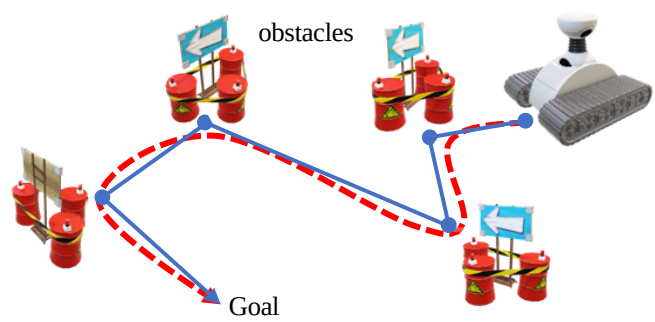

Fig. 6.Spline smoothing (red) of the robot trajectory.

A) Visual SLAM

The Visual SLAM technique is based on using images taken from a camera mounted on a robot. The cameras used can be optical with wide-angle lenses, fisheye lenses, or panoramic like the ladybug camera [50]. Other camera types like the RGB-D cameras [51] are also widely used for indoor mapping applications.

When the Robot is equipped with a single camera, the SLAM technique is called **monocular** SLAM where the depth estimation is challenging. However, using fixed-position landmarks that can be automatically detected in the images

like coded targets and the inertial measurement units IMUs will highly support the visual SLAM technique. When the IMU is used, the SLAM technique is called **visual odometry** which utilizes the motion sensor data derived to estimate a robot's change in position over time.

In literature, visual SLAM well-known techniques are structure from motion (SfM), visual odometry, and bundle adjustment [26]. It should be noted that visual SLAM can rely on sparse point cloud features like ORB-SLAM [10, 11] or rely on dense point cloud features such as DTAM or LSD-SLAM [26]. The well-known visual SLAM techniques like ORB-SLAM [11] can be applied using the following steps (Fig.7) where the first two steps represent the front-end component.

Map Initialization: based on 2D ORB feature correspondences in two overlapped image frames, a relative orientation is applied to estimate the robot (camera) initial pose. The relative orientation can be applied using the fundamental matrix or homography. A triangulation is applied to estimate the initial map of 3D points or landmarks.

Tracking: for each new image frame, apply feature matching in the new frame to features in the previous keyframe. Since the matched features have their 3D positions defined in the previous step, the robot pose is estimated using resection techniques like the perspective-n-points (PnP) method [52-54]. The estimated camera pose is refined by tracking the local map again.

Local Mapping: in this back-end step, the current image frame is used to build new map points. This is applied by adjusting the Robot pose and the map 3D points using bundle adjustment which minimizes the errors of the projected 3D map points into the current image.

Loop Closure: When a revisited place is detected using the Bags of Words technique, the loop closure refinement is applied and all the poses are refined using graph optimization or bundle adjustment.

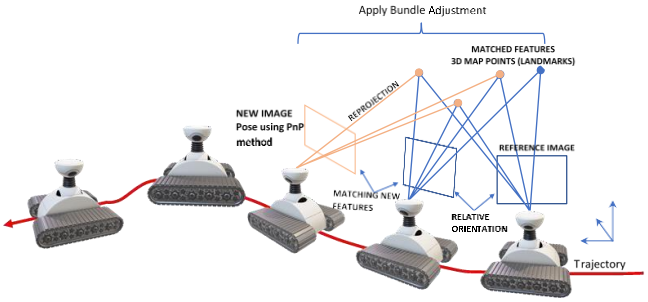

Fig. 7. Monocular visual SLAM technique concept.

Compared to LiDAR SLAM, visual SLAM is a more preferred approach in terms of cost which uses significantly less expensive cameras compared to LiDARs. However, visual SLAM may not be precise as the LiDAR SLAM and could be slower. Another disadvantage of visual SLAM is being very sensitive to the changes in the scene illumination and appearance, and textureless environment. Finally, visual SLAM has the advantage of better scene coverage than LiDAR [6] unless multiple LiDARs are used.

B) LiDAR – Based SLAM

LiDARs (laser scanners) output data is generally 2D or 3D point cloud data and offers high-precision range measurements and performs efficiently for SLAM map construction [26]. The most commonly used scanners are the 2D Hokuyo laser range-finder [55] for indoor mapping [49, 56-58] and the multibeam 3D-LiDAR Velodyne [59] for indoor and outdoor mapping [16, 60].

Similar to SLAM, another term called LOAM is also used in literature to indicate for LiDAR odometry and mapping as a 3D technique [33, 61]. LiDAR-based SLAM has gained researchers' attention because of its high accuracy and the increasing number of open-source implementations, especially for localization and mapping in dynamic indoor environments [62].

Generally, LiDAR-based SLAM enables the robot movement estimation incrementally by registering the successively scanned point clouds. The estimated traveled distance along the trajectory is used for localizing the robot while building the map through the point cloud co-registration using in most cases the iterative closest point (ICP) algorithm [63]. Current LiDAR-based SLAM techniques are relying on derived features out of the point clouds to accomplish the estimation. Reduced map representations like voxel/grid-based methods or point sub-sampling, will effectively decrease the data amount used for the co-registration [18]. Compared to the visual-based SLAM, in this approach, we can work reliably over the variations in lighting conditions or seasons by exploiting the geometric structure like planes (Fig.6) out of the scanned point clouds [7].

On the other hand, the challenge to register the successive LiDAR point clouds is caused by the difficulty to find sufficient correspondences for the co-registration and this may result in losing the robot path. In Fig.8, the initial robot trajectory based on INS observations (yellow) supposed to be refined after the registration, loop closure, and optimization. However, the trajectory got deviated (blue) because of the misalignment between the successive point clouds caused by the insufficient feature correspondences.

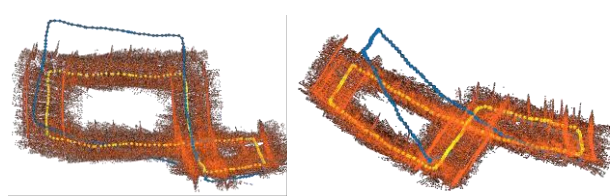

Fig. 8. Misaligned LiDAR SLAM.

Furthermore, point cloud co-registration normally entails pretty demanding computations and then necessitates optimizing the processes for a fast implementation [21]. In addition, the geometry of the LiDAR observations should be strong enough to reliably estimate the robot pose otherwise, the robot slides in some direction.

Therefore, localization of the robots will highly improve when fusing other sensor measurements such as wheel odometry, GNSS, and IMU data [26] and further apply the loop closure technique. Recently, Karam, et al. [64] have

developed several strategies for IMU-LiDAR SLAM integration in which they utilized the IMU measurements to support LiDAR SLAM in overcoming some problematic areas. The IMU contribution in their strategies is not limited to the pose prediction but also the IMU observations participate in the pose estimation. Their results showed the ability of the IMU to support the LiDAR SLAM and prevent the drift in case of insufficient LiDAR observations.

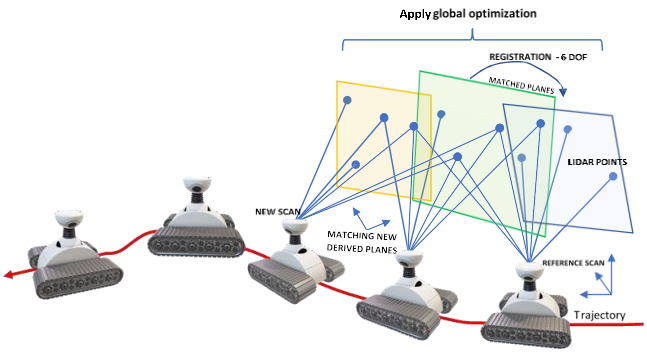

Fig. 9. LiDAR SLAM concept using derived planes co-registration.

Even though, it is still challenging in some cases to get an accurate SLAM result when scanning textures of shiny objects like the glass leading to an inadequate performance [21]. Finally, LiDARs are currently expensive instruments and that makes them unworkable for extensive operations [62].

In Fig.9 a simplified concept for LiDAR SLAM is shown where the planar features out of the scanned point are derived and co-registered. Whenever features are revisited, a loop closure is applied using optimization techniques and then the map is updated.

Finally, it's worth mentioning the existence of hybrid SLAM approaches that combine the Visual and LiDAR SLAM techniques together. Visual LiDAR Odometry and Mapping (V-LOAM) is such an example of an integrated technique where the IMU measurements deliver prior data about the sensor motion to a visual odometry unit, which in turn delivers prior data to the LiDAR matching unit [7].

VI. MAP REPRESENTATION

As mentioned, SLAM is concerned with the sensor pose and the map of the surrounding environment and they are depending on each other during the robot navigation. The map in SLAM can be represented mostly by a sparse set of landmarks, dense point clouds, or by volumetric representation.

A) Sparse map representation: this is the most common map representation in SLAM by a set of sparse 2D or 3D landmarks along the trajectory related to distinctive features in the environment either points, lines, or planes. Then the SLAM technique is referred to as feature-based representations which are mostly known in visual SLAM as the structure from motion technique [2].
The main disadvantage of this sparse representation is the need to have available distinctive features in the mapping environment which might be a problem in poor textured places.

B) Dense map representation: this is mostly a dense unstructured point cloud representation which is also used for obstacle avoidance or rendering. In SLAM, sensors like stereo cameras, RGB-D cameras, or LiDARs are widely used with a dense point cloud representation. The main disadvantage of this dense representation is the need to [2]:
1) Store a large amount of data while they give a low level of information about the geometry.
2) High-performance computing power in real-time.
Accordingly, one solution is to derive geometric primitives like planes [49], cylinders, surfels [4], etc. from the point clouds and then efficiently use them for the registration between successive scans within the SLAM pipeline.

C) Grid-based map representation: this map spatial-partitioning representation is applied by defining adjacent regular geometric primitives either as voxels or 2D grids [2]. The value of each grid cell indicates its state that can be free, occupied, or unknown (Fig.10) which is based on a predicted probability and defined as the *Occupancy Grid* [65]. Hence, the occupancy value of a grid cell is defined using a probabilistic method that has as an input estimated measurement from the robot sensor to the map point (like distances or angles). Then it is possible to update the grid cell values whenever a new measurement is achieved using a Bayesian technique [66]. This updating step of the grid cells' status will continue while the robot is moving and sensing the environment. The resulted grid map can be used for obstacle avoidance, path planning, and pose estimation. Consequently, this representation has the main advantage of being accurate and easy to create [66]. Fig.10 illustrates an example of the 3D grid representation (voxels) of the environment while a robot is moving inside [67].

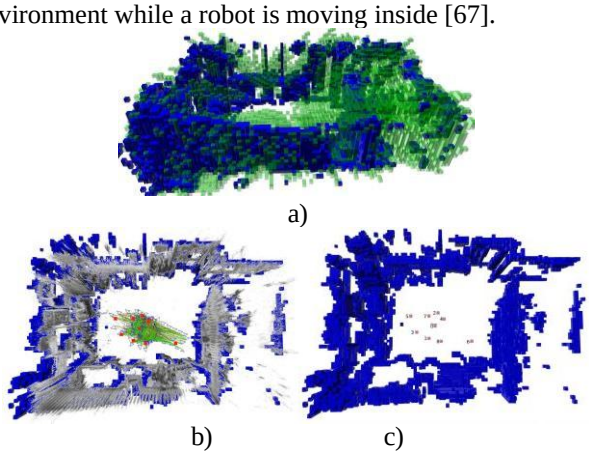

Fig. 10. 3D occupancy grid representation. a) 3D occupancy map defined by blue occupied voxels and green free voxels. b) Top view of a point cloud superimposed on the derived voxels. c) Occupied voxels are shown in blue [67].

VII. Deep Learning for SLAM

In recent years, research regarding SLAM is applied using deep learning techniques to replace the traditional visual odometry approach which is based on geometric processing.

The motivation to use such machine learning techniques is to keep a good positioning performance in difficult environments and to overcome the possible inaccurate scale estimation of visual SLAM [68]. One example is to directly derive the inter-frame pose between two images captured from a moving robot.

Moreover, deep learning is used to estimate the six degrees of freedom (DoF) of a camera (rotation and translation) as well as estimating the depth distance of the objects in the captured single images [4, 33, 69]. It is worth mentioning that most of the early proposed deep learning research on SLAM was only focused on visual odometry for localization without mapping which is recently developed to solve the full SLAM problem [68].

Recently, Sarlin, et al. [70] introduced the new terms of SuperPoint, SuperGlue, and SuperMap. SuperPoint [71] is aimed to replace the geometric-based interest points (like SIFT) with convolutional neural networks CNN feature points. Thereby, feature points and their descriptors are computed together without patches and then enable real-time processing on a GPU. The SuperGlue which is a mix of the Graph Neural Networks and the optimal transport is aimed to improve the feature matching by learning. This approach is promising to successfully achieve matching at extreme wide-baseline stereo images in real-time Fig.11. Sarlin, et al. [70] are continuing the work for a further step of the SuperMap to reach an end-to-end Deep Visual SLAM.

Fig.12 is designed to summarize all the SLAM-related deep learning techniques applied in recent years to enable the reader to have an overview of the recent contribution of deep learning to SLAM.

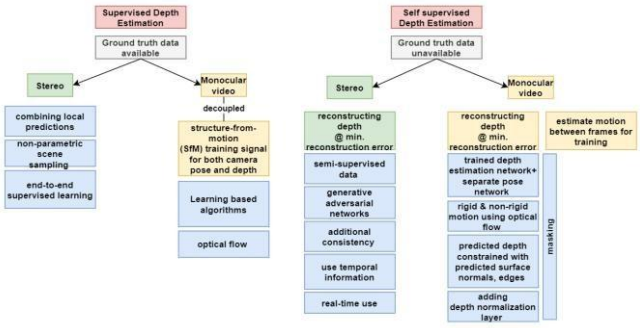

Fig.12. Different research output to apply deep learning for depth estimation in stereo and monocular modes.

VIII. Applications

SLAM is widely used in different applications mainly for mobile mapping tasks in GNSS denied or degraded environments like indoor environments, urban canyons, dense forests, and underwater unmanned missions [4].

Robotic unmanned ground vehicles (UGVs) and unmanned aerial vehicles (UAVs) represent potential future machines that utilized SLAM techniques in different areas (Fig.13a,13b). As an example in the oil and gas industry, the robots would be equipped with sensors that can detect natural gas leaks as well as hazardous substances. This would help to avoid accidents and to keep employees safe.

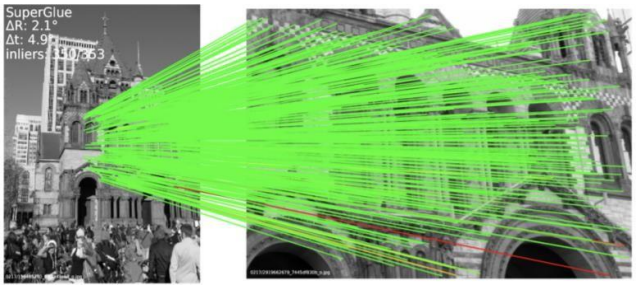

Fig.11. SuperPoint+SuperGlue matching result for a challenging case with two different scales and perspective images [70].

A new generation of car parking autonomous robots using SLAM was found by Stanley Robotics (Fig.13c) introduced a parking service in airports that will save passengers time while also reducing vehicle emissions [72]. The construction and mining industry also invested in UGVs and UAVs which include: moving materials, bulldozing, digging trenches, situational awareness, asset inspection, and excavations (Fig.13d) [73].

Another SLAM-based UAV is introduced by Emesent company [74] with its AL2 Hovermap (Fig.13e). AL2 Hovermap efficiently collects the data automatically based on SLAM techniques in challenging GNSS denied environments like in underground mining mapping missions. Fig.13b shows the HUSKY robot vehicle which performs SLAM to help to predict rock bursts and rock falls [75]. SKEYETECH a fully autonomous drone for security and safety applications is shown in Fig.13f. Another autonomous UAV product is found by Skydio [76] which is designed for real-time 3D mapping, motion planning, scene understanding, and obstacle avoidance. Recently, Boston Dynamics released a versatile Spot robot dog that relies on SLAM to navigate autonomously. Spot has a 6 DOF arm which gives the ability to grasp objects and open doors [77, 78]. More recently, the FARO Focus laser scanner has been attached to Spot for automated 3D scanning (Fig. 14).

Among many SLAM applications, we summarize the following fields:

- Autonomous driving.
- Rescue tasks for high-risk or difficult navigation environments.
- Deep-sea exploration and mining.
- Augmented reality where virtually rendered objects need to fit in the real-life 3D environment.
- Virtual reality where users would like to interact with objects in the virtual environment/gaming.
- Visual surveillance systems.
- Infrastructure inspection and 3D reconstruction.

IX. Conclusions

In this paper, a simplified and clear explanation is introduced about the SLAM technique for the scientific community as well as nonspecialized readers. Terminologies like visual odometry, loop closure, 2D SLAM, and pose-graph optimization are briefly explained with illustrative figures like in Fig.3, Fig.4, and Fig.9.

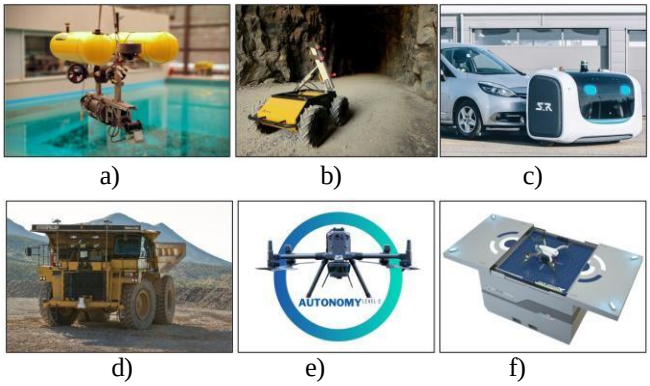

Fig.13. a) SLAM for underwater applications using Girona 500 AUV [79]. b) HUSKY robot for measuring deep mines [75]. c) Stan parking robotic [72]. d) UGV for construction industry [73]. e) Emesent Hovermap [74]. f) SKEYETECH fully autonomous drone [80].

Fig. 14. The integration of FARO Focus scanner and Spot robot dog [81].

Two main mathematical approaches for solving SLAM have been presented namely the Extended Kalman Filter and the pose-graph optimization. The advantages and disadvantages of both mentioned techniques were indicated. The contribution of artificial intelligence and deep learning in predicting scene depth for solving SLAM in challenging environments is summarized in Fig.12.

Readers of the paper who are interested to know about the SLAM problem and its major elements regardless of their scientific specialty are expected to benefit from the overview given in this paper. For a more advanced and deep understanding of SLAM, readers are advised to investigate the several references cited in the paper.

Noticeably, one of the SLAM problem key solutions is the loop closure technique which was introduced in surveying and geodesy science a very long time before adopted in robotics. , it is highly recommended for the specialist in the Geomatics field to focus on the uprising techniques offered by the other scientific fields in computer science, robotics, etc., and to contribute to the advancement of SLAM problems and other uprising problems in autonomous navigation. Furthermore, an invitation to the academic institutions in surveying and geodesy fields to upgrade their curriculums and adapt techniques related to autonomous navigation and artificial intelligence, etc. as we believe these topics will highly impact the development of those sciences.

References

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

[2] C. Cadena *et al.*, "Simultaneous Localization And Mapping: Present, Future, and the Robust-Perception Age," *IEEE Transactions on Robotics*, vol. 32, 06/18 2016, doi: 10.1109/TRO.2016 .2624754.

[3] H. Durrant-Whyte, D. Rye, and E. Nebot, "Localization of autonomous guided vehicles," in *Robotics Research*, London, G. Giralt and G. Hirzinger, Eds., 1996// 1996: Springer London, pp. 613-625.

[4] A. Yao. "Teaching robots presence: what you need to know about SLAM." Comet Labs Research Team.https://blog.cometlabs.io/teaching-robotspresence- what-you-need-to-know-about-slam-9bf0ca037553 (accessed 30 August 2020).

[5] S. Prabhu. "Introduction to SLAM (Simultaneous Localisation and Mapping)." ARreverie Techno-logy. http://www.arreverie.com/blogs/introduction-simultaneous-localisation-and-mapping/ (accessed 30 August 2020).

[6] C. Pao. "How are Visual SLAM and LiDAR used in robotic navigation?" CEVA. https://www.ceva-dsp.com/ourblog/how-are-visual-slam-and-lidar-used-in-robotic-navigation/ (accessed 30 August 2020).

[7] Y. Nava, "Visual-LiDAR SLAM with loop closure," PhD thesis,Master's thesis, KTH Royal Institute of Technology, 2018.

[8] Y. Baudoin and M. K. Habib, *Using Robots in Hazardous Environments: Landmine Detection, De-Mining and Other Applications*. Elsevier, 2010.

[9] A. Angrisano, M. Petovello, and G. Pugliano, "Benefits of combined GPS/GLONASS with low-cost MEMS IMUs for vehicular urban navigation," (in eng), *Sensors (Basel)*, vol. 12, no. 4, pp. 5134-5158, 2012, doi: 10.3390/s120405134.

[10] R. Mur-Artal, J. Montiel, and J. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, pp. 1147-1163, 10/01 2015, doi: 10.1109/TRO.2015.2463671.

[11] M. Inc. "Monocular visual simultaneous localization and mapping." https://nl.mathworks.com/help/ vision/examples/monocular-visual-simultaneous-localization-and-mapping.html (accessed 30 August 2020).

[12] R. Szeliski, "Structure from motion," in *Computer Vision: Algorithms and Applications*, R. Szeliski Ed. London: Springer London, 2011, pp. 303-334.

[13] J. O. Esparza-Jiménez, M. Devy, and J. L. Gordillo, "Visual EKF-SLAM from heterogeneous landmarks," (in eng), *Sensors (Basel)*, vol. 16, no. 4, p. 489, 2016, doi: 10.3390/s16040489.

[14] S. Agarwal, K. S. Parunandi, and S. Chakravorty, "Robust pose-graph slam using absolute orientation sensing," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 981-988, 2019.

[15] J. Procházková and D. Martišek, *Notes on Iterative Closest Point Algorithm*. Proc. in 17th Conference on Applied Mathematics, 876–884, 2018.

[16] A. Filgueira, P. Arias, and M. Bueno, "Novel inspection system, backpack-based, for 3D modelling of indoor scenes," International Conference on Indoor Positioning and Indoor Navigation (IPIN), 4-7 October 2016, Alcalá de Henares, Spain, (October), 4–7..

[17] Y.-H. Choi, T.-K. Lee, and S.-Y. Oh, "A line feature based SLAM with low grade range sensors using geometric constraints and active exploration for mobile robot," *Autonomous Robots*, vol. 24, no. 1, pp. 13-27, 2008, doi: 10.1007/s10514-007-9050-y.

[18] J. Behley and C. Stachniss, *Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments*. Proc. of Robotics: Science and Systems; Pittsburgh, PA; 2018.

[19] H. W. Sorenson, "Kalman filtering techniques," in *Advances in Control Systems*, vol. 3, C. T. Leondes Ed.: Elsevier, 1966, pp. 219-292.

[20] M. Xanthidis, C. Stachniss, P Allen, P. Furgale, M. Chli, M. Huttter, M. Rufli, D. Scaramuzza and R. Siegwart."SLAM Tutorial " ETH Zurich. http://www.cs.columbia.edu/~allen/F17/NOTES/slam_pka.pdf (accessed 30 August 2020).

[21] S. A. S. Mohamed, M. Haghbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, and J. Plosila, "A Survey on Odometry for Autonomous Navigation Systems," *IEEE Access*, vol. 7, pp. 97466-97486, 2019.

[22] S. Karam, V. Lehtola, and G. Vosselman, "Integrating a low-cost mems imu into a laser-based slam for indoor mobile mapping," *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. XLII-2/W17, pp. 149-156, 2019, doi: 10.5194/isprs-archives-XLII-2-W17-149-2019.

[23] E. M. Mikhail and F. E. Ackermann. Observations and least squares. Washington, D.C: University Press of America, 1982. http://catalog.hathitrust.org/api/volumes/oclc/8551922.html.

[24] S. S. Rao, *Engineering Optimization - Theory and Practice, 4th ed*. New Jersey, USA: JOHN WILEY & SONS, INC., 2009.

[25] J. Folkesson and H. I. Christensen, "Closing the loop with graphical SLAM," *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 731-741, 2007.

[26] MathWorks. "SLAM (Simultaneous Localization and Mapping)." https://nl.mathworks.com/discov-ery/slam.html (accessed 30 August 2020).

[27] B. Alsadik, *Adjustment models in 3D geomatics and computational geophysics: with MATLAB examples*. Elsevier Science, 2019.

[28] I. Ullah, X. Su, X. Zhang, and D. Choi, "Simultaneous localization and mapping based on kalman filter and extended kalman filter," *Wireless Communications and Mobile Computing*, vol. 2020, p. 2138643, 2020/06/08 2020, doi: 10.1155/2020/2138643.

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

[30] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. "g2o: a general framework for graph optimization." https://openslam-org.github.io/g2o.html (accessed 30 August 2020).

[31] "Ceres Solver." Google Inc. http://ceres-solver.org/ (accessed 30 August 2020).

[32] "Georgia Tech Smoothing and Mapping Library GTSAM." https://github.com/borglab/gtsam acce-ssed 30 August 2020).

[33] D. Lu, "Vision-enhanced lidar odometry and mapping," MSc. Thesis, Carnegie Mellon University, Pittsburgh, PA, 2016.

[34] R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss and A. Kleiner. "On measuring the accuracy of SLAM algorithms," *Autonomous Robots*, vol. 27, no. 4, p. 387, 2009, doi: 10.1007/s10514-009-9155-6.

[35] J. M. Santos, D. Portugal, and R. P. Rocha, "An evaluation of 2D SLAM techniques available in Robot Operating System," *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 21-26, 2013, pp. 1-6, doi: 10.1109/SSRR.2013.6719348.

[36] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E Berger, R. Wheeler and A. Ng. "ROS: an open-source Robot Operating System". *ICRA workshop on open source software*. Vol. 3. No. 3.2. 2009..

[37] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," *IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1271-1278, doi: 10.1109/ICRA.2016.7487258.

[38] C. Thomson, G. Apostolopoulos, D. Backes, and J. Boehm, "Mobile laser scanning for indoor modelling," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-5/W2, 10/16 2013, doi: 10.5194/isprsannals-II-5-W2-289-2013.

[39] N. Li, *L. Guan, Y. Gao, S. Du, M. Wu, X. Guang and X. Cong*, "Indoor and outdoor low-cost seamless integrated navigation system based on the integration of INS/GNSS/LIDAR system," *Remote Sensing*, vol. 12, no. 19, p. 3271, 2020. [Online]. Available:https://www.mdpi.com/20724292/12/19/3271.

[40] X. Zhang, J. Lai, D. Xu, H. Li, and M. Fu, "2D LiDAR-based SLAM and path planning for indoor rescue using mobile robots," *Journal of Advanced Transportation*, vol. 2020, p. 8867937, 2020/11/17 2020, doi: 10.1155/2020/8867937.

[41] A. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proceedings Ninth IEEE International Conference on Computer Vision*, 13-16 Oct. 2003 2003, pp. 1403-1410 vol.2, doi: 10.1109/ICCV.2003.1238654.

[42] I. Mahon and S. Williams, "Three-Dimensional Robotic Mapping," PRoc. Australasian Conference on Robotics and Automation. 2003..

[43] J. Weingarten and R. Siegwart, "EKF-based 3D SLAM for structured environment reconstruction," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 3834-3839, doi: 10.1109/IROS.2005.1545285.

[44] M. Baglietto, A. Sgorbissa, D. Verda, and R. Zaccaria, "Human navigation and mapping with a 6DOF IMU and a laser scanner," *Robotics and Autonomous Systems*, vol. 59, no. 12, pp. 1060-1069, 2011, doi: https://doi.org/10.1016/j.robot .2011.08.005.

[45] S. Zhang and S. Qin, "An Approach to 3D SLAM for a mobile robot in unknown indoor environment towards service operation,"*Chinese Automation Congress (CAC)*, 2018 2018, pp. 2101-2105, doi: 10.1109/CAC.2018.8623105.

[46] D. Lu. "LiDAR mapping with ouster 3D sensors." Ouster. https://ouster.com/blog/lidar-mapping-with-ouster-3d-sensors/ (accessed 30 August 2020).

[47] S. Y. Chen and Y. F. Li, "Vision sensor planning for 3–D model acquisition," presented at the IEEE transactions on systems, man, and cybernetics. 2005.

[48] G. Vosselman, "Design of an indoor mapping system using three 2D laser scanners and 6 DOF SLAM," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-3, pp. 173-179, 08/07 2014, doi: 10.5194/isprsannals-II-3-173-2014.

[49] S. Karam, G. Vosselman, M. Peter, S. Hosseinyalamdary, and V. Lehtola, "Design, calibration, and evaluation of a backpack indoor mobile mapping system," *Remote Sensing*, vol. 11, p. 905, 2019, doi: 10.3390/rs11080905.

[50] FLIR."Ladybug5." https://www.flir.com/products/ladybug5plus/ (accessed 30 August 2020).

[51] REALSENSE. "Depth Camera D415." https://www.-intelrealsense.com/depth-camera-d415/ (accessed 30 August 2020).

[52] B. Alsadik, "A modified method for image triangulation using inclined angles," *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. XLI-B3, pp. 453-458, 2016, doi: 10.5194/isprs-archives-XLI-B3-453-2016.

[53] L. Kneip, D. Scaramuzza, and R. Siegwart, "A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation," in *CVPR 2011*, 2011, pp. 2969-2976, doi: 10.1109/CVPR.2011.5995464.

[54] E. W. Grafarend and J. Shan, "Closed-form solution of P4P or the three-dimensional resection problem in terms of Möbius barycentric

coordinates," *Journal of Geodesy*, vol. 71, no. 4, pp. 217-231, 1997, doi: 10.1007/s001900050089.

[55] Hokuyo Automatic Co., Ltd. https://www.hokuyo-aut.jp/ (accessed 19 February, 2021).

[56] G. Chen, J. Kua, S. Shum, N. Naikal, M. Carlberg, and A. Zakhor, "Indoor localization algorithms for a human-operated backpack system," 2010.

[57] T. Liu, M. Carlberg, G. Chen, J. Chen, J. Kua, and A. Zakhor, "Indoor localization and visualization using a human-operated backpack system," in *2010 International Conference on Indoor Positioning and Indoor Navigation*, Zurich, Switzerland, 15-17,2010, pp. 1-10, doi: 10.1109/IPIN.2010 .5646820.

[58] Viametris. "vMS3D." https://www.viametris. com/vms3d (accessed July 21, 2020).

[59] Velodyne. https://velodynelidar.com/ (accessed 1 October, 2018).

[60] NavVis. "Versatile Reality Capture with NavVis VLX _ NavVis." https://www.navvis.com/ (accessed 19 February, 2021).

[61] B. Huang, J. Zhao, and J. Liu, "A Survey of Simultaneous Localization and Mapping with an Envision in 6G Wireless Networks*"*. 2019.

[62] 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, doi: https://doi.org/10.1016/j.autcon.2019.04.011.

[63] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239-256, 1992, doi: 10.1109/34.121791.

[64] S. Karam, V. Lehtola, and G. Vosselman, "Strategies to integrate IMU and LiDAR SLAM for indoor mapping," *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. V-1-2020, pp. 223-230, 2020, doi: 10.5194/isprs-annals-V-1-2020-223-2020.

[65] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46-57, 1989.

[66] A. A. S. Souza, R. Maia, and L. M. G. Gonçalves, "3D probabilistic occupancy grid to robotic mapping with stereo vision," in *Current Advancements in Stereo Vision*: IntechOpen, 2012.

[67] L. v. Stumberg, V. C. Usenko, J. Engel, J. Stückler, and D. Cremers, "From monocular SLAM to autonomous drone exploration," *European Conference on Mobile Robots (ECMR)*, pp. 1-8, 2017.

[68] C. Zhao, K. Sun, P. Purkait, T. Duckett, and R. Stolkin, "Learning Monocular Visual Odometry with Dense 3D Mapping from Dense 3D Flow," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 6864-6871, doi: 10.1109/IROS.2018.8594151.

[69] C. Godard, O. M. Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 3827-3837, doi: 10.1109/ICCV.2019.00393.

[70] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning Feature Matching With Graph Neural Networks." Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 4938-4947

[71] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: self-supervised interest point detection and description," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2018, pp. 224-236

[72] R. KNIGHT. "UGV's Gaining Ground." inside unmanned systems. https://insideunmannedsystems .com/ugvs-gaining-ground/ (accessed 30 August 2020).

[73] Stanley Robotics. https://stanley-robotics.com/ (acces-sed 30 August 2020).

[74] Emesent. "Hovermap AL2." https://www.em-esent.io/ (accessed 30 August 2020.

[75] "Husky performs slam on stereonets to help predict rock falls and rock bursts." Queen's University. https://clearpathrobotics.com/husky-queens-mining-slam-stereonets/ (accessed 30 August 2020).

[76] Skydio. https://www.skydio.com/pages/skydioautonomy (accessed 30 August 2020).

[77] B. Dynamics. "Spot Arm." https://www.bostondy-namics.com/spot-arm (accessed 5 January 2021).

[78] E. Ackerman. "Boston Dynamics' Spot Robot Dog." IEEE Spectrum. https://spectrum.ieee.org/autom-aton/robotics/industrial-robots/boston-dynamics-spot-robot-dog-now-available (accessed 5 January, 2021).

[79] N. Palomeras, M. Carreras, and J. Andrade-Cetto, "Active SLAM for autonomous underwater exploration," *Remote Sensing*, vol. 11, p. 2827, 11/28 2019, doi: 10.3390/rs11232827.

[80] Azur Drones. "Skeyetech, fully autonomous drone for safety and security." https://www.azurdrones. com/product/skeyetech/ (accessed 30 August 2020).

[81] FARO. "FARO Trek 3D Laser Scanning Integration." https://www.faro.com/en/Products/Hardware/Trek-3D-Laser-Scanning-Integration (accessed 21 March, 2021).