Journal of
Marine Science and Engineering

MDPI

Article

A Trajectory-Based Approach to Multi-Session Underwater Visual SLAM Using Global Image Signatures

Antoni Burguera Burguera and Francisco Bonin-Font *

Departament de Matemàtiques i Informàtica, Universitat de les Illes Balears, Carretera de Valldemossa Km 7.5, 07122 Palma, Illes Balears, Spain

* Correspondence: francisco.bonin@uib.es

Received: 9 July 2019; Accepted: 14 August 2019; Published: 17 August 2019

Abstract: This paper presents a multi-session monocular Simultaneous Localization and Mapping (SLAM) approach focused on underwater environments. The system is composed of three main blocks: a visual odometer, a loop detector, and an optimizer. Single session loop closings are found by means of feature matching and Random Sample Consensus (RANSAC) within a search region. Multi-session loop closings are found by comparing hash-based global image signatures. The optimizer refines the trajectories and joins the different maps. Map joining preserves the trajectory structure by adding a single link between the joined sessions, making it possible to aggregate or disaggregate sessions whenever is necessary. All the optimization processes can be delayed until a certain number of loops has been found in order to reduce the computational cost. Experiments conducted in real subsea scenarios show the quality and robustness of this proposal.

Keywords: visual SLAM; multi-session robot; posidonia oceanica

1. Introduction

A crucial task for an Autonomous Underwater Vehicle (AUV) is to build a map of the environment and to estimate its own pose within the map while it is navigating. This task is known as Simultaneous Localization and Mapping (SLAM) [1] and is nowadays a de facto standard for autonomous vehicles, not only in underwater environments.

Since well-established approaches to SLAM, such as Extended Kalman Filter SLAM (EKF-SLAM) or Graph-SLAM [2], are widely used and new approaches mainly concentrate on alleviating their intrinsic limitations [3], SLAM is often considered a solved problem. New research tends to focus on practical- and application-related issues, which depend on or are strongly related to the sensors used to build the map and to estimate the vehicle motion and pose.

Range sensors, such as laser range finders in terrestrial environments or sonar in underwater scenarios, were the modality of choice at first. However, research turned to computer vision as soon as the computational capabilities and price of on-board systems made it possible [4], since cameras provide a much richer representation of the world.

Underwater computer vision poses several problems [5] that only exist up to a much lesser extent in other environments. That is why underwater SLAM solely based on vision [6] is not frequent, as it requires approaches far more robust than its terrestrial counterparts. Visual SLAM and the associated visual loop closing detection processes are usually included in the navigation systems of AUV equipped with other sensors to mitigate the localization drift obtained when navigating only with inertial units (gyroscopes, magnetometers, and accelerometers), visual odometers, or Doppler Velocity Log (DVL) sensors.

However, large-scale or long time operations generate big maps with huge amounts of visual data that can collapse the vehicle computer if they are not treated intelligently [7,8]. A common strategy to overcome this problem is to explore the areas of interest in different, separated, missions called sessions [9–11]. Any low capability of a robot to operate robustly during long periods of time can be alleviated by repeating transits through previously visited areas and by joining all trajectories in a single coordinate frame. When SLAM is performed in these conditions, it is referred to as multi-session SLAM.

In the context of the Augmented Reality Subsea Exploration Assistant (ARSEA) and Twin Robots (TWINBOT) national projects, one or several robots equipped with cameras have to operate cooperatively in tasks such as (a) sea bottom exploration, mapping, and mosaicking for biologic purposes and (b) multi-robot coordinated intervention. All these tasks are done in marine areas which are densely colonized with seagrass and, in particular, with Posidonia oceanica (see further details in Section 10.1). In both projects, multi-session SLAM will be indispensable to aggregate the mapping data computed by all robots that participate in the missions.

Besides, several challenges come up when working in medium and large underwater areas colonized with seagrass, complicating the calculus of visual odometry and loop closings, namely (a) intricate textures; (b) slight dynamics due to the slow oscillation of the seagrass leaves with the water currents; (c) nonexistence of structured scenarios; and (d) lighting and visual hindrances, such as water turbidity, light scattering, flickering, and lack of natural light and visibility in deeper zones.

Furthermore, it is necessary to consider that (1) multi-session SLAM loops between different sessions, known as global loops, are continuously searched during the AUV operation. Therefore, reducing the computational cost of global image matching is crucial. (2) The advantages and drawbacks of the existing approaches for map joining define a compromise between map consistency and computation time; for example, building a full map mixing different sessions leads to more consistency than anchor-node-based solutions [9] at the cost of significantly larger computation times.

This paper presents a novel trajectory-based multi-session visual SLAM and map joining approach which pays special attention on a frontend that searches intersession and intra-session loop closings and a backend to run the single and the global map optimizations. The relevant points and contributions are summarized next:

1. The application of a multi-session loop detector that uses the Hash-Based Loop Closure (HALOC) [12] image global signature (hash) for fast-matching solves the problem of lacking geometry information between different sessions at a considerable speed; as a matter of fact, Negre Carrasco et al. [12] has already shown that HALOC improves the loop closing detection performance with respect to Fast Appearance-Based Mapping (FABMAP) [13] in terms of perceptual aliasing, execution time, and recall and especially in underwater environments.
2. The use of a hash to represent images for multi-session loop closing implies a considerable reduction of data to store and exchange between two different robots, when this application is extended (in forthcoming work) to a multi-robot context. This data reduction will be crucial to make feasible multi-session or multi-robot operations where one robot assumes the task to join all the maps of all robots that participate in the mission.
3. A novel and simple algorithm to join multiple map sessions: This system does not use anchor nodes but it joins two maps through a transformation between the end of one trajectory and the beginning of the other, solving, at the same time, and automatically the-so called initial state problem.
4. A strategy to perform delayed global map optimization (for map joining) to reduce computational load
5. A complete set of software sources available for the scientific community in GitHub public repositories [14–16]
6. A wide set of experiments conducted in Mediterranean marine scenarios, mostly colonized with Posidonia oceanica, which show the validity and robustness of the localization system

To the best of our knowledge, such a trajectory-based approach to multi-session SLAM focused on marine bottoms colonized with seagrass has not been proposed before in the robotic literature.

The paper is structured as follows. Section 2 summarizes the existing research on the subject. Section 3 overviews our proposal, its main components, and how they relate to each other. Section 4 summarizes the notation used throughout the paper. Section 5 describes the block in charge of performing visual odometry. While performing visual odometry, both intra-session and intersession loops are detected as described in Sections 6 and 7, respectively. These loops make it possible to optimize the trajectories as described in Section 8 and to join them as stated in Section 9. Finally, Section 10 presents an extensive set of experimental results using real data gathered in several areas of Mallorca (Spain). The conclusions and some insight for further work are provided in Section 11.

2. Related Work

The so-called initial state problem or kidnapped robot problem [17] refers to the fact that, when a robot starts a new session on the same or in a partially overlapping nearby environment, it does not know its relative pose with respect to another map created previously. One possibility to solve this issue is to localize itself in any map built beforehand. This solution has the advantage of maintaining a single track and a single reference frame. However, this alternative has a strong restriction: the robot must start in a point already mapped. The other possibility is to initiate another map corresponding to this new session, with its own local reference, and then, when the robot detects a point already visited during a previous session, it calculates the transformation between both maps and joins them. This last strategy is known formally as the problem of multi-session simultaneous localization and mapping (SLAM) [18] and consists in combining multiple SLAM trajectories obtained repeatedly over time in the same area by a single or several robots.

In this context, the challenges are basically two: (1) join properly maps gathered in different sessions based on their overlapping parts and (2) improve the current pose and map estimates using previous sessions' data. In general, the aforementioned two main goals are achieved thanks to three building blocks: a visual odometer [19] that matches consecutive images and provides local motion estimates, a loop detector [12,20] that asserts if the autonomous underwater vehicle (AUV) returns to a previously visited place, and an optimizer [21,22] that fuses odometry and loops to consistently improve and join maps as well as to properly estimate the AUV pose. Additional problems appear in multi-session localization because images gathered in several sessions can be extremely different due to changes in the illumination conditions [5] or even the use of different cameras or AUVs.

Existing vision-based multi-session SLAM approaches are mainly focused on terrestrial and aerial environments. Some of them are based on the anchor-nodes method [23], which introduces two main concepts: (a) robot trajectory *anchor*, as the offset of a complete trajectory with respect to a global system of coordinates, and (b) an *encounter*, defined as a measurement or transformation that connects two different poses of two different robots; in multi-session SLAM, the encounters are not direct transforms between robots but indirect via observations of the same area performed at different times. Encounters express additional constraints relating different graphs corresponding to different sessions. For example, McDonald et al. [9] presents a multi-session stereo-vision SLAM approach based on anchor-nodes for indoor and outdoor terrestrial environments.

Contrarily to single session SLAM, multi-session loop detection (i.e., finding encounters) cannot rely on the AUV pose to constrain the search since, at first, the relative pose between sessions is unknown. Instead, global image descriptors [24] such as *Bag of Words* (BoW) are often used. In McDonald et al. [9], the loop closings are detected using a BoW-based solution combined with Incremental Smoothing and Mapping (iSAM) [25] for batch map optimization and Conditional Random Fields (CRF) [26] for feature matching. In [10,27], a new memory management is presented in order to optimize the treatment of the successive graph nodes in multi-session mapping and path planning applications for large-scale indoor office-like environments, using a robot equipped with a RGBD camera and a laser scanner. BoW is used to detect visual global loop closings, the Tree-Based

Network Optimizer (TORO) [28] for graph optimization and the laser together with the RGBD point clouds for map visualization. Latif et al. propose in Reference [29] a new algorithm for loop closure verification in single and multi-session pose graph connectivity: odometry is obtained by means of laser scan matching, with the loop closing candidates using BoW and the *g2o* [21] framework for graph optimization.

All the aforementioned references have been tested only in terrestrial environments, some indoors and outdoors and others only indoors. The literature is extremely scarce in multi-session SLAM addressed, implemented, and tested in underwater scenarios with AUVs. References [18,30,31] are some of the very few pieces of work with these characteristics. The work presented in References [30,31] was designed for the very specific purpose of ship hull inspection and surveillance using AUVs. In this case, the robot moves around the ship hull, at a fixed distance to it, with the camera and the Doppler velocity log (DVL) pointing nadir to the hull. Planes fitting sparse 3-D point clouds obtained by the DVL are used to map the surface, and, in cooperation with a camera, to do single and multi-session SLAM. The Fast Appearance-Based Mapping (FABMAP) [13] framework, which is based on BoW, is used for visual loop closing detection and anchor nodes for the map joining task. However, that field application is far from the scope of the work presented in our study.

The work presented by Williams et al. in Reference [18] is closer to ours in the sense that the surroundings of an ancient shipwreck are surveyed by an AUV that records video sequences for visual mapping and 3-D reconstruction. The vehicle moves over the bottom, with a camera pointing downwards. The different portions of the shipwreck grabbed in different sessions are joined together in a single map using multi-session SLAM techniques.

3. Overview

This section summarizes our proposal. On the one hand, it introduces the main building blocks and the relationships among them. On the other hand, it provides some details to ease further reading and emphasizes the advantages and fields of application of the presented approach.

Our proposal, summarized in Figure 1, focuses on the three building blocks of multi-session visual SLAM, paying special attention to robust and fast place recognition and facilitating the trade-off between consistency and speed. Moreover, our proposal is solely based on vision sensors and an altitude sensor for the pixel/meters scale computation. There is no need for dead reckoning devices.

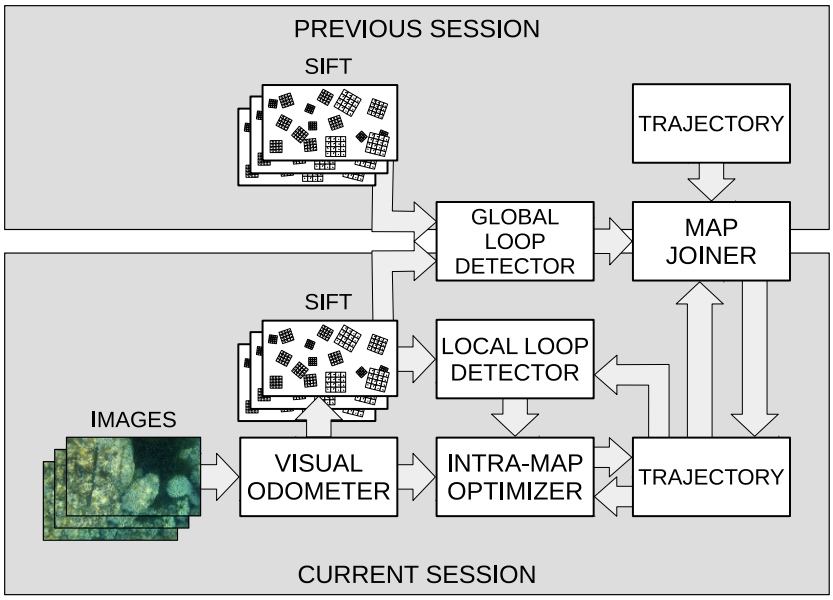


Figure 1. System overview.

A simple yet robust visual odometer based on Scale-Invariant Feature Transform (SIFT) [32] is first described. Its output is used to build the so-called trajectory, which is a set of relative motion estimates between consecutive images.

Two loop detectors are introduced afterwards: the local loop detector, which is in charge of detecting loop candidates inside a region of interest around the AUV in one single session, and the global loop detector, which uses image hashes to find loop candidates between different sessions without any geometrical constraint. In both cases, candidates are confirmed by means of feature matching and Random Sample Consensus (RANSAC) [33].

When used as global image descriptors, image hashes are functions providing fixed length outputs that are similar only in front of similar images [34]. The multi-session loop closing detection task compares image hashes of images of different sessions. Hashes are obtained with HALOC [12] because this approach reduces the perceptual aliasing inherent to clustering algorithms such as BoW [12], it outperforms previous approaches in terms of reliability and computation time, it is particularly well suited in challenging underwater scenarios [35], and it allows a very fast image comparison. Moreover, since the SIFT features used by HALOC have already been computed to perform visual odometry, requiring them does not compromise the execution time.

Two methods based on Extended Kalman Filters (EKF) are used to optimize the trajectories. The first one is the *intra-map optimizer* and is in charge of improving the existing trajectory by means of the local loops. The second one is the *map joiner*, of which the goal is to join different sessions while keeping the intrinsic trajectory structure. The computational complexity [1] of the intra-map optimizer EKF is strongly reduced with respect to other approaches, since the trajectory based approach [6] makes the state vector grow at a significantly lower rate. As for the map joiner EKF, it has a constant execution time because its state has a fixed size. In both cases, linearization errors are alleviated by iteratively re-linearising over subsequent EKF estimates, that is, using an Iterated EKF (IEKF).

The adopted optimization strategies have the following advantages: (a) They allow delayed loop closings, making it possible to store loop data and to optimize the trajectory or join different sessions only when enough computational resources are available; (b) as sessions are joined using a single link, the overhead introduced to join the maps is almost negligible. Moreover, this single linkage approach makes it possible to disaggregate the sessions easily to reduce the computation time when necessary, and (c) contrarily to pose-based SLAM approaches [36], which play with global poses, this approach composes the trajectory with relative motions; then, EKF linearization problems are less relevant since motion covariances are disaggregated and only those involved in each loop take part in the process.

In order to increase the reproducibility of the presented results and to facilitate the use of our algorithms, the whole source code related to the research in this paper has been made publicly available. The links to each specific piece of code are provided throughout the paper.

4. General Notation

Even though specific notation will be introduced when needed, there are some common conventions that are pervasively used throughout the paper. This section focuses on such general notation and can be used as a reference to better understand further sections.

Let I_t denote the image grabbed by the camera at time step t, with its coordinate frame placed at its center with the X and Y axes pointing forward and left, respectively. Also, let a_t denote the altitude at which the image was grabbed. This study assumes that a_t is obtained by external means, such as a stereo-vision altimeter or a DVL. Let us assume also that the underwater vehicle is programmed to navigate at a constant altitude, in an approximated horizontal plane, simplifying the robot motion to a 2-D trajectory.

The normal distribution $X_B^A = N(\hat{X}_B^A, P_B^A)$ models the roto-translation from image I_A to image I_B, with $\hat{X}_B^A$ as its mean and P_B^A as its covariance. Assuming a bottom-looking camera with negligible pitch and roll, $\hat{X}_B^A$ can be expressed as a motion in X (x_B^A) and Y (y_B^A) and a rotation over Z (θ_B^A). That is, $\hat{X}_B^A = (x_B^A, y_B^A, \theta_B^A)^T$.

The trajectory is defined as the set of motions between consecutively grabbed images that fully define the path followed by the AUV. Our proposal is to model the trajectory at time t as $X_t = N(\hat{X}_t, P_t)$, so that $\hat{X}_t$, shown in Equation (1), denotes the mean of X_t and so that P_t is the associated covariance matrix.

$$\hat{X}_t = \begin{pmatrix} (\hat{X}_1^0)^T & (\hat{X}_2^1)^T & \dots & (\hat{X}_t^{t-1})^T \end{pmatrix}^T \tag{1}$$

Equation (2) shows how the relative motion X_j^i between two arbitrary time steps i and j can be recovered as a normal distribution by means of the compounding $\oplus$ and inversion $\ominus$ operators as described in Smith et al. [37].

$$X_j^i = \begin{cases} X_{i+1}^i \oplus X_{i+2}^{i+1} \oplus \cdots \oplus X_j^{j-1} & j > i \\ \ominus X_j^{j-1} \oplus \ominus X_{j-1}^{j-2} \oplus \cdots \oplus \ominus X_{i+1}^i & j < i \\ (0,0,0)^T & j = i \end{cases} \tag{2}$$

The absolute pose X_t^0, which is the pose at time step t relative to the first image, can be easily recovered from the trajectory using the previous equation. Figure 2 summarizes the notation.

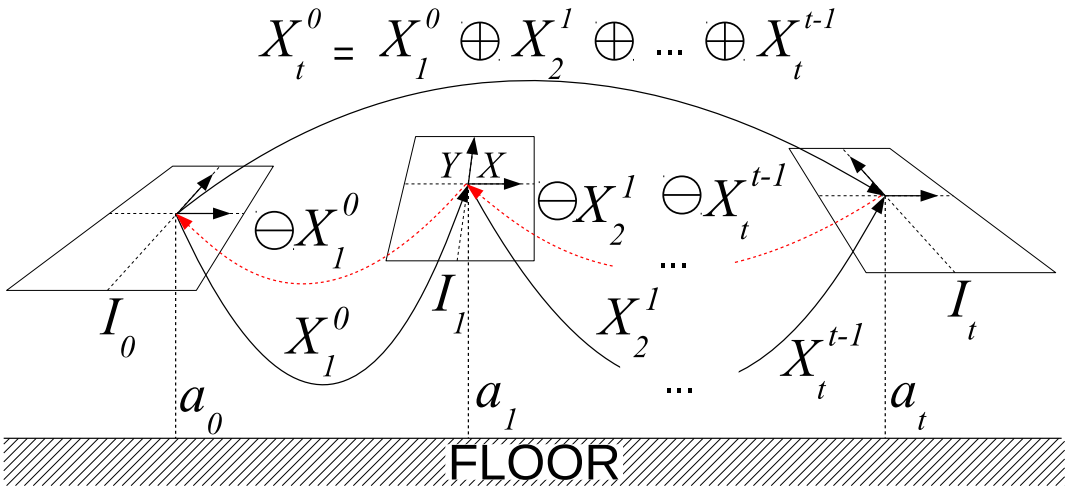


Figure 2. Notation.

5. Visual Odometry

This section centers its attention on the visual odometer, which is in charge of providing local motion estimates between pairs of consecutively gathered images. Our proposal makes use of SIFT feature matching and RANSAC to provide local motions as accurate as possible.

Visual odometry consists of computing the motion X_t^{t-1} between images I_{t-1} and I_t taken by the robot camera at time instants $t-1$ and t. To achieve this goal, the following process, summarized in Figure 3a, is used.

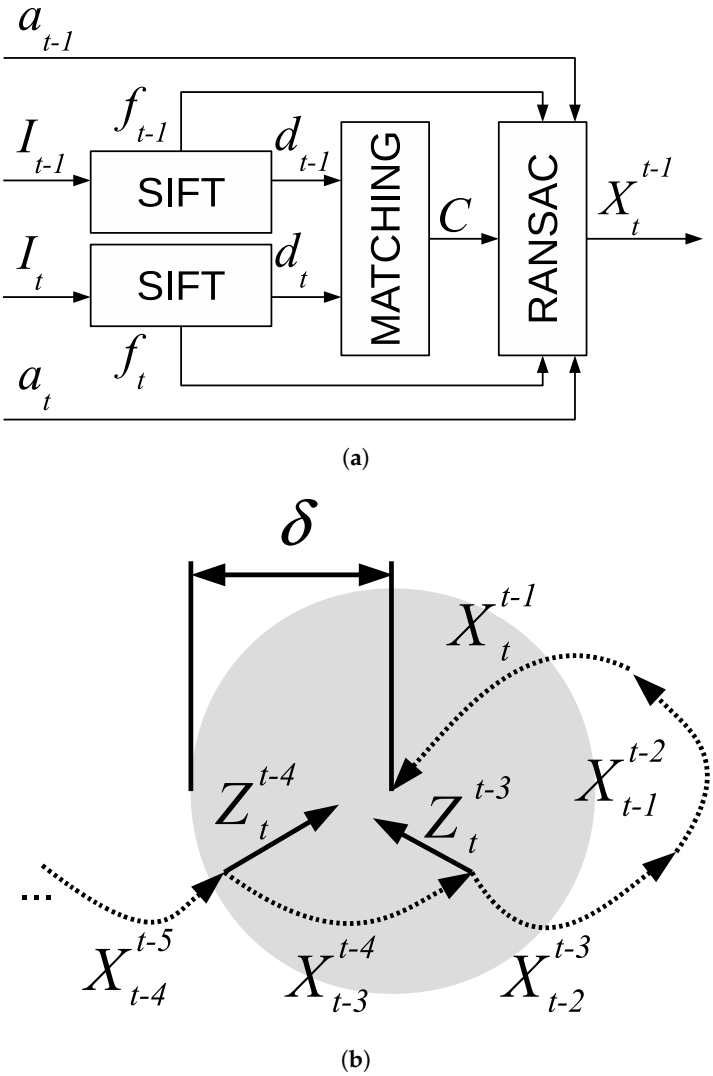


Figure 3. (**a**) Odometry calculation and (**b**) local loop detection.

First, a set of SIFT features and their corresponding descriptors is computed for I_{t-1} and I_t. Let $f_{t-1} = \{f_{t-1,0}, \cdots, f_{t-1,m-1}\}$ and $f_t = \{f_{t,0}, \cdots, f_{t,n-1}\}$ denote the m and n features corresponding to images I_{t-1} and I_t, respectively. A feature simply represents an (x,y) coordinate within the image. Analogously, let $d_{t-1} = \{d_{t-1,0}, \cdots, d_{t-1,m-1}\}$ and $d_t = \{d_{t,0}, \cdots, d_{t,n-1}\}$ denote the SIFT descriptors so that each $d_{i,j}$ describes the feature $f_{i,j}$.

The descriptors d_{t-1} and d_t are matched by means of standard SIFT matching. As a result, a set of feature correspondences C is obtained containing the pairs (i,j) of the matching descriptors $d_{t-1,i}$ and $d_{t,j}$.

At this point, the relative motion between I_{t-1} and I_t could be computed by means of Equation (3) as the one that minimizes the sum of squared distances between the pairs of corresponding features.

$$\hat{X}_t^{t-1} = \arg\min_X \sum_{(i,j)\in C} ||X \oplus f_{t-1,i} - f_{t,j}|| \tag{3}$$

A closed form solution to Equation (3) is available in Lu and Milios [38]. However, since some of the existing correspondences may not be correct, our proposal is to embed Equation (3) into the RANSAC approach shown in Algorithm 1, where $apply\_altitude()$ is a function that converts image feature coordinates from pixel to meters.

Algorithm 1: RANSAC approach to estimate the motion $\hat{X}_B^A$ from image I_A to image I_B.

Input:
f_A, f_B: SIFT features in images I_A and I_B
a_A, a_B: Altitudes corresponding to I_A and I_B
C: Set of correspondences
K: Number of iterations to perform
N_{corr}: Number of correspondences to be randomly selected
N_{min}: Minimum number of correspondences to consider a roto-translation as candidate
ε_{corr}: Maximum allowable error per correspondence

Output:
$fail$: Boolean stating if failed to find $\hat{X}_B^A$
$\hat{X}_B^A$: The estimated roto-translation

begin
  $f'_A \leftarrow apply\_altitude(f_A, a_A)$;
  $f'_B \leftarrow apply\_altitude(f_t, a_B)$;
  $\varepsilon_B^A \leftarrow \infty$; $fail \leftarrow true$;
  for $i \leftarrow 0$ **to** $K-1$ **do**
    $R \leftarrow$ random selection of N_{corr} items from C;
    $X \leftarrow \arg\min_T \sum_{(i,j)\in R} ||T \oplus f'_{A,i} - f'_{B,j}||$;
    $\varepsilon \leftarrow \sum_{(i,j)\in R} ||T \oplus f'_{A,i} - f'_{B,j}||$;
    foreach $(i,j) \in (C-R)$ **do**
      if $\|X \oplus f'_{A,i} - f'_{B,j}\| < \varepsilon_{corr}$ **then**
        $R \leftarrow R \cup \{(i,j)\}$;
      end
    end
    if $|R| > N_{min}$ **then**
      $X \leftarrow \arg\min_T \sum_{(i,j)\in R} ||T \oplus f'_{A,i} - f'_{B,j}||$;
      $\varepsilon \leftarrow \sum_{(i,j)\in R} ||T \oplus f'_{A,i} - f'_{B,j}||$;
      if $\varepsilon < \varepsilon_B^A$ **then**
        $\varepsilon_B^A \leftarrow \varepsilon$; $\hat{X}_B^A \leftarrow X$; $fail \leftarrow false$;
      end
    end
  end
end

Roughly speaking, this algorithm is based on the idea that correct correspondences are consistent among them, thus leading to the same roto-translation, whilst incorrect correspondences are responsible for different roto-translations. Our proposal is to exploit this idea by checking if it is possible to find a consistent subset of C that is large enough to be considered correct.

The algorithm selects a random subset R of C and then computes the roto-translation X as well as the corresponding error ε using only this subset. These values are computed by means of Equation (3) using R instead of C.

Afterwards, each of the nonselected matchings in C is checked. If the error it introduces is below a threshold ε_{corr}, then it is included within R. If at some point the number of items in R surpasses a threshold N_{min}, the roto-translation and the error are computed again using this expanded R, and if the error is below the smallest error until now, the roto-translation is stored as a good model.

This process is iterated a fixed number of times. A description of how to compute the number of iterations depending on the expected number of inliers in the input data is provided in Fischler and Bolles [33]. When the algorithm finishes, it is possible to recover the best roto-translation, which is a robust version of the $\hat{X}_t^{t-1}$ shown in Equation (3). This roto-translation will not be found if partial roto-translations are inconsistent and so R never has enough items. In this case, which is unlikely to happen since consecutive images have sufficient overlap, our proposal is to use $\hat{X}_{t-1}^{t-2}$ instead.

Even though it is out of the scope of this paper, our approach to RANSAC makes it possible to properly guess P_t^{t-1} (covariance of the motion estimated between $t-1$ and t) since it internally computes an error estimate ε_t^{t-1}.

6. Local Loop Detection

A local loop is a loop involving images belonging to one single session. These loops not only provide valuable information to improve each session separately but also are used to improve all the sessions together after joining them. This section is devoted to explaining how the local loops are searched.

The Local Loop Detection (LLD) is in charge of finding loop closings within one single SLAM session. First, the set of loop candidates at time step t is built as shown in Equation (4), where X_t^i is computed by Equation (2), by searching within a predefined radius δ [6].

$$LC_t = \{i : ||\hat{X}_t^i||_2 \leq \delta, i < t-1\} \tag{4}$$

Afterwards, the same process to compute visual odometry described in Section 5 is applied to image pairs I_i and I_t for all $I_i \in LC_t$ in order to build the set of local loops LL_t as shown in Equation (5).

$$LL_t = \{Z_t^i : i \in LC_t \cap \neg fail(i,t)\} \tag{5}$$

In this Equation, $\neg fail(i,t)$ denotes that RANSAC did not fail to find a roto-translation between I_i and I_t. Z_t^i represents the normal $Z_t^i = N(\hat{Z}_t^i, R_t^i)$ so that its mean $\hat{Z}_t^i$ is the roto-translation provided by RANSAC and R_t^i is the associated covariance. This covariance can either be obtained heuristically from the error computed by Algorithm 1 or using the methods described in References [39,40].

In general, the pose according to the trajectory $X_1^0 \oplus ... \oplus X_t^{t-1}$ will not coincide with the pose provided by the loop closings $X_1^0 \oplus ... \oplus X_i^{i-1} \oplus Z_t^i$ because of the measurement errors. The trajectory optimization described in Section 8 is in charge of fusing both sources of information into a consistent trajectory. Figure 3b illustrates these concepts.

7. Global Loop Detection

Global loops are those involving images belonging to different sessions, thus making it possible to establish a relation between them and to join the corresponding trajectories. Since no geometric information between two separated sessions exist, global loops have to rely on robust image matching methods. This section describes our approach to robustly detect global loops.

The goal of Global Loop Detection (GLD) is to find loop closings involving different SLAM sessions. Since in this case it is not possible to geometrically constrain the search region because there is no geometrical relation between both sessions, our proposal is to use a high performance hash-matching approach [12] to select loop candidates and to use RANSAC to validate them.

Let the descriptor matrix D_t of size $n \times m$ contain all the n SIFT descriptors of size m in image I_t. The number of SIFT features is fixed to n for all images. The HALOC [12] hash (sources available in Reference [14]) H_t, which is a vector of which the size is $3m$ independently of the number n_t of features found, is built according to Equations (6) and (7) by projecting each column of D_t onto three different random orthogonal directions each defined by a unit vector u_l of n dimensions.

$$H_t = \begin{pmatrix} (h_{t,0})^T & (h_{t,1})^T & (h_{t,2})^T \end{pmatrix}^T \tag{6}$$

$$h_{t,l}(i) = \sum_{j=0}^{n-1} D_t(j,i) u_l(j), 0 \leq i < m \tag{7}$$

The set of vectors u_l is defined off-line, previous to the SLAM process. The use of random and orthogonal projections avoids providing repetitive information that would reduce the hashing

efficiency [41]. Using three directions has shown to lead to an acceptable trade-off between low hash size and high performance, although other values could be used.

Let V_p denote a previously gathered video sequence and V_c be a video sequence that is currently being gathered and that will eventually capture regions overlapping with V_p. Our proposal is to compute H_t for every image in each sequence. This is extremely fast not only because of the simplicity of HALOC but also because the required SIFT descriptors have previously been computed to achieve odometry. Afterwards, the hash of the current image in V_c is compared to the hash of each image in V_p in order to build the set of candidate global loops GC_t shown in Equation (8). This is also a fast process as a comparison is performed by means of the L1 norm.

$$GC_t = \{i : ||H_i - H_t||_1 \leq \delta', \forall I_i \in V_p\} \tag{8}$$

The value of δ' can be selected depending on the computational resources available. This process is summarized in Figure 4. Finally, the set of global loops GL_t is built as shown in Equation (9).

$$GL_t = \{Z_t^i : i \in GC_t \cap \neg fail(i,t)\} \tag{9}$$

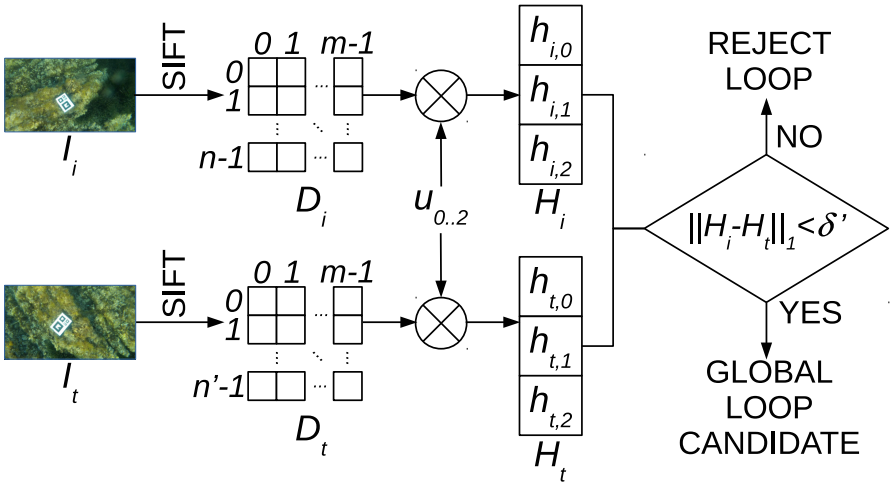


Figure 4. Summary of the Hash-Based Loop Closure (HALOC) operation.

Similarly to LL_t, $\neg fail(i,t)$ denotes that RANSAC did not fail to find a roto-translation between I_i and I_t and Z_t^i is a normal of which the mean $\hat{Z}_t^i$ is the transformation found by RANSAC. The corresponding covariance can be computed using the same methods proposed for the local loops in Section 6.

8. Trajectory Optimization

This section describes how a single trajectory is optimized to meet the constraints imposed by the visual odometry and the detected local loops. As a result of this optimization, the trajectory that best explains both constraints is found. It is important to emphasize that this trajectory can also be the trajectory resulting from joining two sessions.

Let us first describe the process to iteratively build an initial guess of the trajectory by means of the odometric estimates in the current session. This is the so-called trajectory building. Afterwards, a method to optimize the trajectory using an IEKF (sources available in Reference [15]) and the detected local loops is devised. This method, which is referred to as intra-map optimization, constitutes by itself a trajectory-based single-session SLAM approach.

8.1. Trajectory Building

The trajectory itself constitutes the state vector of the abovementioned IEKF. With each new odometric estimate X_t^{t-1}, an initial guess $X_t^- = N(\hat{X}_t^-, P_t^-)$ of the trajectory is constructed according to Equations (10) and (11) by augmenting the state vector in the previous time step.

As for the initial values, following the recommendations in Castellanos et al. [42], $\hat{X}_0$ and P_0 are both set to zero so that the initial AUV pose is assumed to be a global reference frame known without uncertainty.

$$\hat{X}_t^- = \begin{pmatrix} (\hat{X}_{t-1})^T & (\hat{X}_t^{t-1})^T \end{pmatrix}^T \tag{10}$$

$$P_t^- = \text{blkdiag}\{P_{t-1}, P_t^{t-1}\} \tag{11}$$

The covariance P_t^{t-1} of the last odometric estimate can be determined in several ways. For example, it could be heuristically estimated using the ε_B^A computed by Algorithm 1. Also, the two methods presented in References [39,40], which rely on a solid theoretical background, can be used.

If at time t no local loops are detected, this initial guess is consolidated and becomes the trajectory itself ($X_t \leftarrow X_t^-$). If LL_t is not empty, the trajectory is optimized. This optimization is achieved by performing the IEKF update using the detected local loops as measurements. Since only loop closings can lead to changes in the stored trajectory, the state vector will not change during the IEKF prediction and, thus, it is not necessary to perform that step.

It is important to emphasize that Equation (11) alone leads to a block diagonal covariance matrix. However, the intra-map optimization will properly introduce the cross-correlation information by means of the detected local loops.

8.2. Intra-Map Optimization

Since the presence of local loops impose constraints between nonconsecutive items in the trajectory, considering the loop closings as measurements of the state vector makes it possible to optimize the trajectory. Let us model the measurement vector $Z_t = N(\hat{Z}_t, R_t)$ using the detected local loops $Z_t^{ij} = N(\hat{Z}_t^{ij}, R_t^{ij}) \in LL_t$ as a normal of which the mean and covariance are shown in Equations (12) and (13), respectively.

$$\hat{Z}_t = \begin{pmatrix} (\hat{Z}_t^{i0})^T & (\hat{Z}_t^{i1})^T & \dots & (\hat{Z}_t^{in})^T \end{pmatrix}^T \tag{12}$$

$$R_t = \text{blkdiag}\{R_t^{i0}, R_t^{i1}, \dots, R_t^{in}\} \tag{13}$$

The IEKF observation function g_t is in charge of predicting each item in $\hat{Z}_t$ according to the state vector prior estimate X_t^-. To build that function, let us first define in Equation (14) an observation function g_t^i associated to each measurement in Z_t.

$$g_t^i(X_t^-) = \hat{X}_{i+1}^i \oplus \hat{X}_{i+2}^{i+1} \oplus \cdots \oplus \hat{X}_{t-1}^{t-2} \oplus \hat{X}_t^{t-1} \tag{14}$$

The overall observation function g_t can now be constructed by means of Equation (15).

$$g_t = \begin{pmatrix} (g_t^{i0})^T & (g_t^{i1})^T & \dots & (g_t^{in})^T \end{pmatrix}^T \tag{15}$$

Thus, each item in g_t is the guess, according to X_t^-, of the corresponding item in $\hat{Z}_t$. The IEKF observation matrix G_t is the Jacobian matrix of g_t as shown in Equations (16) and (17).

$$G_t = \frac{\partial g_t}{X_t} = \left(\left(\frac{\partial g_t^{i0}}{\partial X_t}\right)^T \left(\frac{\partial g_t^{i1}}{\partial X_t}\right)^T \ \dots \ \left(\frac{\partial g_t^{in}}{\partial X_t}\right)^T \right)^T \tag{16}$$

$$\frac{\partial g_t^i}{\partial X_t} = \left(\frac{\partial g_t^i}{\partial X_1^0}, \frac{\partial g_t^i}{\partial X_2^1}, \dots, \frac{\partial g_t^i}{\partial X_t^{t-1}} \right) \tag{17}$$

According to Equation (14), only the items from X_{i+1}^i onward appear in g_t^i. Thus, the partial derivatives with respect to X_{j+1}^j will be zero for all $j < i$, making it possible to change Equation (17) into Equation (18).

$$\frac{\partial g_t^i}{\partial X_t} = \left(0_{3\times 3i}, \ \frac{\partial g_t^i}{\partial X_{i+1}^i}, \ \dots, \ \frac{\partial g_t^i}{\partial X_t^{t-1}} \right) \tag{18}$$

Each of these partial derivatives could be directly computed. If necessary, some approaches to express them in terms of the Jacobian matrices of the composition transformation have been proposed in Burguera et al. [6].

At this point, an EKF update could be performed by means of Z_t, g_t, and G_t evaluated at X_t^-. However, since G_t is used by the EKF to linearise g_t, the results would be highly influenced by such linearisation, especially when closing large loops. That is why an IEKF is used instead, since it alleviates the linearisation problems [43]. The IEKF update iterates an EKF update relinearising the system at each iteration. The process stops after a fixed number of iterations or when convergence is achieved.

At the jth iteration, the mean $\hat{X}_{t,j}$ and the covariance $P_{t,j}$ can be obtained by simply iterating the EKF. However, the computational cost can be reduced by using Equations (19) and (20). They have been obtained by operating the standard EKF formulation while taking into account the particularities of our proposal.

$$\begin{aligned} \hat{X}_{t,j} &= \hat{X}_{t,j-1} + P_{t,j-1} G_{t,j-1}^T R_t^{-1} (\hat{Z}_t - g_t(\hat{X}_t^-)) - \\ &- P_{t,j-1} P_t^{-1} (\hat{X}_{t,j-1} - \hat{X}_t^-) \end{aligned} \tag{19}$$

$$\begin{aligned} P_{t,j} &= P_t^- - P_t^- G_{t,j-1}^T (G_{t,j-1} P_t^- G_{t,j-1}^T + R_t)^{-1} \cdot \\ &\cdot \ G_{t,j-1} P_t^- \end{aligned} \tag{20}$$

In these Equations, $G_{t,j}$ denotes G_t evaluated at $\hat{X}_{t,j}$, $\hat{X}_{t,0} = X_t^-$ and $P_{t,0} = P_t^-$. The process iterates until $||\hat{X}_{t,j} - \hat{X}_{t,j-1}||$ is below a certain threshold or after a maximum number of iterations is reached. The computed mean and covariance in the last iteration constitute the optimized trajectory.

9. Map Joining

Global loops relate two different sessions and can be used to identify the geometric relationship between two sepparated trajectories. In this section, our proposal to identify such a relationship and to use it to join two trajectories into a single one is described.

In order to preserve the trajectory structure, the goal of map joining (sources available in Reference [16]) is to find the proper transformation between the end of the first trajectory and the start of the second one. Both trajectories, known as sessions, are separated in time, and the map joining task is aligning the current survey to a previously completed map.

Let X_p denote the trajectory of a previous session. Let the first and last images that took part in this trajectory be referred to as I_{ps} and I_{pe}, respectively. Similarly, let X_c denote the current session's trajectory and let its first and last images be denoted by I_{cs} and I_{ce}, respectively. Since the map joining is performed while X_c is being built, I_{ce} will be exactly I_t.

At time step t, one or more global loops may have been found. Each of these loops relate one image in a previous session with I_t in the current session. Our proposal is to store global loops until a

fixed number K of them has been found and to then use them all to join the maps. Let $Z_G = N(\hat{Z}_G, R_G)$, as defined in Equations (21) and (22), denote this set of K accumulated global loops.

$$\hat{Z}_G = \begin{pmatrix} (\hat{Z}_{c0}^{p0})^T & (\hat{Z}_{c1}^{p1})^T & \dots & (\hat{Z}_{cK-1}^{pK-1})^T \end{pmatrix}^T \tag{21}$$

$$R_G = \text{blkdiag}\{R_{c0}^{p0}, R_{c1}^{p1}, ..., R_{cK}^{pK}\} \tag{22}$$

The normals $Z_{ci}^{pi} = N(\hat{Z}_{ci}^{pi}, R_{ci}^{pi})$ represent links between image I_{pi} in the previous session and image I_{ci} in the current one. Each Z_{ci}^{pi} belongs to GL_{ci} (Equation (9)).

Our goal is to find the relative motion between the last image of the previous session I_{pe} and the first image of the current one I_{cs}. Let this relative motion be referred to as $X_{cs}^{pe} = N(\hat{X}_{cs}^{pe}, P_{cs}^{pe})$. Figure 5a summarizes these concepts.

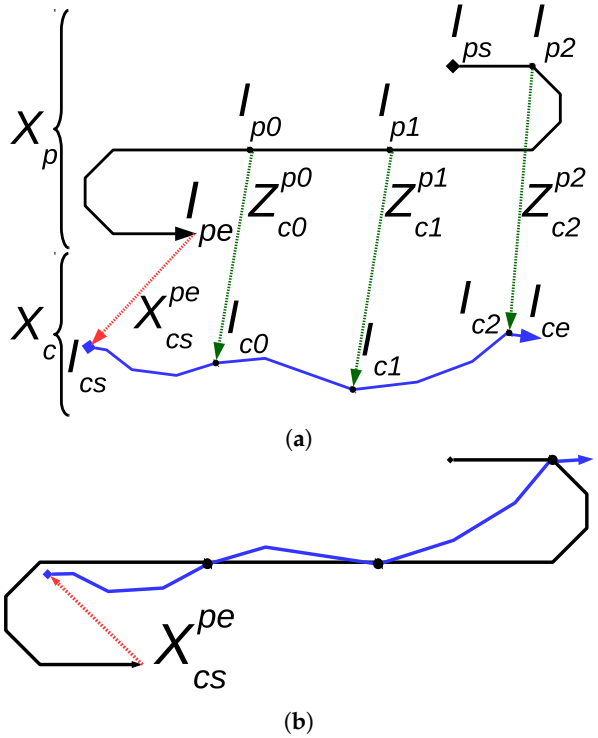


Figure 5. (**a**) Summary of the map joining and (**b**) an example of a joined trajectory.

This paper proposes the use of an IEKF to perform the optimization, with X_{cs}^{pe} as the state vector and Z_G as the measurement. That is, the IEKF will find the X_{cs}^{pe} that better explains all the loop closures. Since X_{cs}^{pe} only depends on the loop closings, which constitute the measurements during the IEKF update, there is no need to perform the IEKF prediction.

Let us begin by building the observation function g_G (Equation (23)) which provides an estimate of each loop in Z_G from the state vector.

$$g_G = \begin{pmatrix} (g_G^0)^T & (g_G^1)^T & \dots & (g_G^{K-1})^T \end{pmatrix}^T \tag{23}$$

Each $g_G^i = \hat{X}_{pe}^{pi} \oplus \hat{X}_{cs}^{pe} \oplus \hat{X}_{ci}^{cs}$ estimates the loop Z_{ci}^{pi}, with $X_{pe}^{pi} = N(\hat{X}_{pe}^{pi}, P_{pe}^{pi})$ as the transformation from the loop closing image I_{pi} to the last image of the previous session I_{pe} and $X_{ci}^{cs} = N(\hat{X}_{ci}^{cs}, P_{ci}^{cs})$ as the transformation from the first image in the second session I_{cs} and the loop closing image I_{ci}. Both can be computed by means of Equation (2).

The Jacobian matrix G_G of the observation function, which is needed by the IEKF optimization, can be computed as shown in Equations (24)–(28). The terms s and c denote sin and cos, respectively.

$$G_G = \begin{pmatrix} (\frac{\partial g_G^0}{X_{cs}^{pe}})^T & (\frac{\partial g_G^1}{X_{cs}^{pe}})^T & \dots & (\frac{\partial g_G^{K-1}}{X_{cs}^{pe}})^T \end{pmatrix}^T \tag{24}$$

$$\frac{\partial g_G^i}{X_{cs}^{pe}} = \begin{pmatrix} c\theta_{pe}^{pi}, & -s\theta_{pe}^{pi}, & -y_{ci}^{cs}c(\theta_{pe}^{pi}+\theta_{cs}^{pe}) - x_{ci}^{cs}s(\theta_{pe}^{pi}+\theta_{cs}^{pe}) \\ s\theta_{pe}^{pi}, & c\theta_{pe}^{pi}, & x_{ci}^{cs}c(\theta_{pe}^{pi}+\theta_{cs}^{pe}) - y_{cs}^{pe}s(\theta_{pe}^{pi}+\theta_{cs}^{pe}) \\ 0, & 0, & 1 \end{pmatrix} \tag{25}$$

$$\hat{X}_{pe}^{pi} = (x_{pe}^{pi}, y_{pe}^{pi}, \theta_{pe}^{pi})^T \tag{26}$$

$$\hat{X}_{cs}^{pe} = (x_{cs}^{pe}, y_{cs}^{pe}, \theta_{cs}^{pe})^T \tag{27}$$

$$\hat{X}_{ci}^{cs} = (x_{ci}^{cs}, y_{ci}^{cs}, \theta_{ci}^{cs})^T \tag{28}$$

Now, the IEKF in Equations (19) and (20) can be applied to find the transformation between both maps. The standard formulation is provided next only for clarity purposes. At the jth iteration of the IEKF, the mean $\hat{X}_{cs,j}^{pe}$ and the covariance $P_{cs,j}^{pe}$ of the state vector can be computed by means of Equations (29)–(31). The term $C_{G,j-1}$ denotes G_G evaluated at $\hat{X}_{cs,j-1}^{pe}$.

$$\hat{X}_{cs,j}^{pe} = \hat{X}_{cs,j-1}^{pe} + K_j(Z_G - g_G(\hat{X}_{cs,j-1}^{pe})) \tag{29}$$

$$P_{cs,j}^{pe} = (I - K_j G_{G,j-1})P_{cs,j-1}^{pe} \tag{30}$$

$$K_j = P_{cs,j-1}^{pe} G_{G,j-1}^T (G_{G,j-1} P_{cs,j-1}^{pe} G_{G,j-1}^T + R_G)^{-1} \tag{31}$$

The process is iterated until $||\hat{X}_{cs,j}^{pe} - \hat{X}_{cs,j-1}^{pe}||$ is below a certain threshold or a maximum number of iterations is reached. When that happens, the final $\hat{X}_{cs,j}^{pe}$ and $P_{cs,j}^{pe}$ constitute the outputs $\hat{X}_{cs}^{pe}$ and P_{cs}^{pe} of the IEKF.

$\hat{X}_{cs,0}^{pe}$ being the initial guess of the state vector, our proposal is to compute it from one of the loops in Z_G, since there is a single solution in the presence of a single loop and, thus, a closed form expression exists. Equation (32) shows such a closed form expression if the first loop Z_{c0}^{p0} is used.

$$\hat{X}_{cs,0}^{pe} = \ominus X_{pe}^{p0} \oplus Z_{c0}^{p0} \oplus (\ominus X_{c0}^{cs}) \tag{32}$$

Since X_{cs}^{pe} expresses a transformation between the end of the previous session and the beginning of the current one, it is extremely simple to use it to join both sessions into a single one. Equations (33) and (34) show the mean and the covariance of the joined trajectory $X_J = N(\hat{X}_J, P_J)$, where P_p and P_c denote the covariances of X_p and X_c, respectively.

$$\hat{X}_J = \begin{pmatrix} (\hat{X}_p)^T & (\hat{X}_{cs}^{pe})^T & (\hat{X}_c)^T \end{pmatrix}^T \tag{33}$$

$$P_J = \text{blkdiag}\{P_p, P_{cs}^{pe}, P_c\} \tag{34}$$

In this way, the resulting trajectory is consistent with both sessions and can be used, from this moment onward, to perform the single-session SLAM described in Section 8.

Additionally, with our proposal being able to simultaneously take into account several global loop closures, different sessions may be kept separated until necessary, reducing the computational cost. Also, the adopted trajectory-based structure with links between different sessions allows to easily separate joined maps whenever is necessary for the sake of computation time.

Figure 5b shows the resulting trajectory after applying the described process to the example in Figure 5a.

The whole process is summarized in Algorithm 2. The function compute_tails is in charge of computing the motion from one of the loop closing images to the end of the previous session and

from the beginning of the current session to the other loop closing image. As stated previously, this is achieved using Equation (2). The function compute_observation computes each of the g_G^i (see Equation (23)) and the corresponding Jacobian matrices (Equations (24)–(28)). Finally, IEKF_update refers to Equations (29)–(31).

Algorithm 2: Map joining.

Input:
Z_G: Set of K global loops
X_p: Previous session trajectory
X_c: Current session trajectory
Output:
X_J: Joined trajectory
begin
  $X_{ce}^{p0}, X_{c0}^{cs} \leftarrow$ compute_tails$(p0, c0, X_p, X_c)$;
  $X_{cs}^{pe} = N\left(\hat{X}_{cs}^{pe}, P_{cs}^{pe}\right) \leftarrow \ominus X_{pe}^{p0} \oplus Z_{c0}^{p0} \oplus (\ominus X_{c0}^{cs})$;
  while *not IEKF convergence* **do**
    $g_G, G_G \leftarrow \emptyset$;
    for $i \leftarrow 0$ **to** $K-1$ **do**
      $X_{ce}^{pi}, X_{ci}^{cs} \leftarrow$ compute_tails(pi, ci, X_p, X_c);
      $g_G^i, G_G^i \leftarrow$compute_observation$(X_{ce}^{pi}, X_{ci}^{cs}, Z_G, X_p, X_c, X_{cs}^{pe})$;
      $g_G \leftarrow \left[g_G^T, \left(g_G^i\right)^T\right]^T$;
      $G_G \leftarrow \left[G_G^T, \left(G_G^i\right)^T\right]^T$;
    end
    $X_{cs}^{pe} \leftarrow$IEKF_update(g_G, G_G, X_{cs}^{pe});
  end
  $\hat{X}_J \leftarrow \left(\left(\hat{X}_p\right)^T, \left(\hat{X}_{cs}^{pe}\right)^T, \left(\hat{X}_c\right)^T\right)^T$;
  $P_J \leftarrow$ blkdiag$\{P_p, P_{cs}^{pe}, P_c\}$;
  $X_J \leftarrow N\left(\hat{X}_J, P_J\right)$;
end

As can be observed, the algorithm begins by computing an initial estimate using the first existing loop in Z_G according to Equation (32). Afterwards, in each IEKF loop, the observation function as well as its Jacobian matrix are iteratively computed by appending a new term for every measurement in Z_G. Finally, when the algorithm converges, the joined trajectory X_J is built by linking X_P and X_C using the obtained X_{cs}^{pe}.

10. Experimental Results

This section presents an extensive set of experiments evaluating the main building blocks of our proposal. These experiments, which have been conducted using real data gathered in coastal areas of Mallorca (Spain) colonized with Posidonia oceanica, assess both qualitatively and quantitatively the ability of our proposal to properly perform multi-session SLAM.

10.1. Experimental Setup

For a first evaluation, two different, partially overlapping video sequences $V1$ and $V2$ were recorded in Port de Valldemossa (Spain) with a bottom-looking camera in a marine environment with sand, rocks, seagrass, and moss. The camera was attached to a diver with the lens axis approximately perpendicular to the bottom. The diver moved on the water surface in an area with an approximate constant depth of 3 m. The lack of any other sensorial data which could be supplied by an AUV makes the localization system a pure vision-based approach.

Some images of these sequences are shown in Figure 6, where it can be observed that the region is populated with Posidonia oceanica, a seagrass that forms dense colonies characterized by its long

and thin leafs. Posidonia is crucial in the maintenance of the Mediterranean marine ecosystems and declared by the European Community a species with special protection. One of the tasks in the ARSEA project includes mapping Posidonia meadows and quantifying their bottom coverage using an AUV and several algorithms to discriminate the Posidonia from the background based on deep learning [44]. Due to the particular texture of the Posidonia and the slight motion of its leafs caused by the water current, tracking stable visual features in consecutive overlapping frames is a challenging task. A high number of outliers and/or a slow feature detection process, matching, or tracking might compromise the accuracy in the calculation of the visual odometry and the registration of images that close loops. However, previous references [45,46] already showed that SIFT is one of the best features to be used in this type of underwater environments in terms of matching and tracking performance. SIFT is also the key feature type in HALOC. Although processing SIFT is slower than other descriptors, the C++ version of HALOC is highly efficient in areas with Posidonia [12,35]. In our opinion and according to our experience and obtained results, given the robustness and traceability of SIFT, spending an additional slight portion of time in the process of RANSAC-based feature detection and matching to obtain more reliable trajectories is more preferable than using other simpler features that present less computational cost than SIFT (thus faster) but can cause larger inaccuracies in the image transformations because they also give a higher number of outliers when tracked/matched.

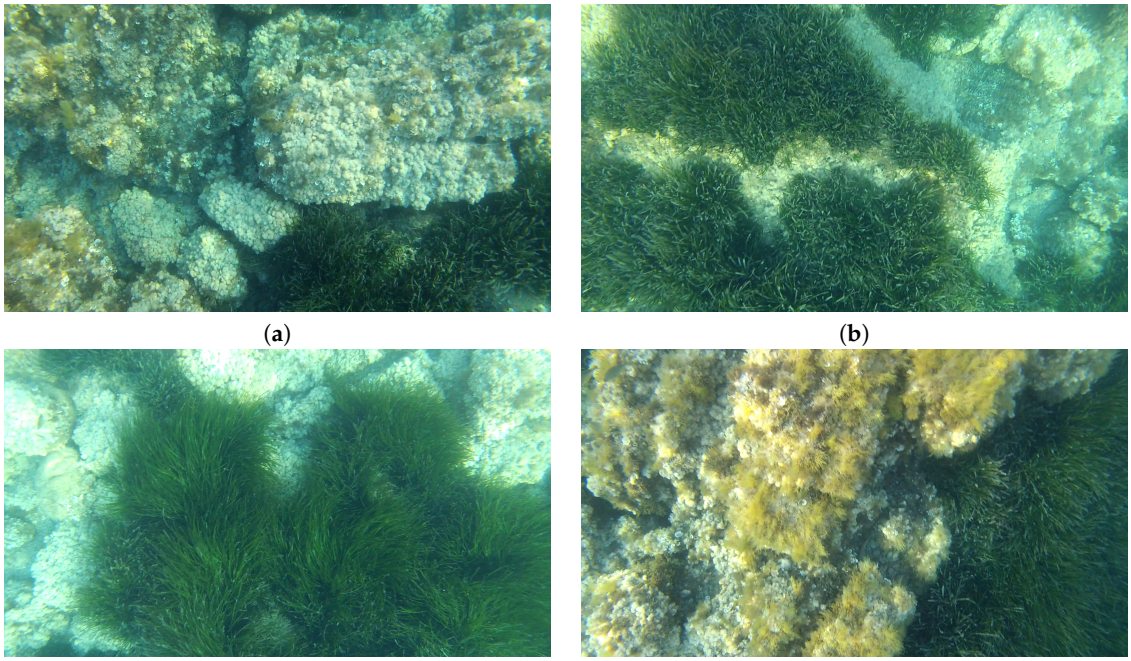


Figure 6. Examples of the images used in the experiments: (**a**,**b**) session 1 and (**c**,**d**) session 2.

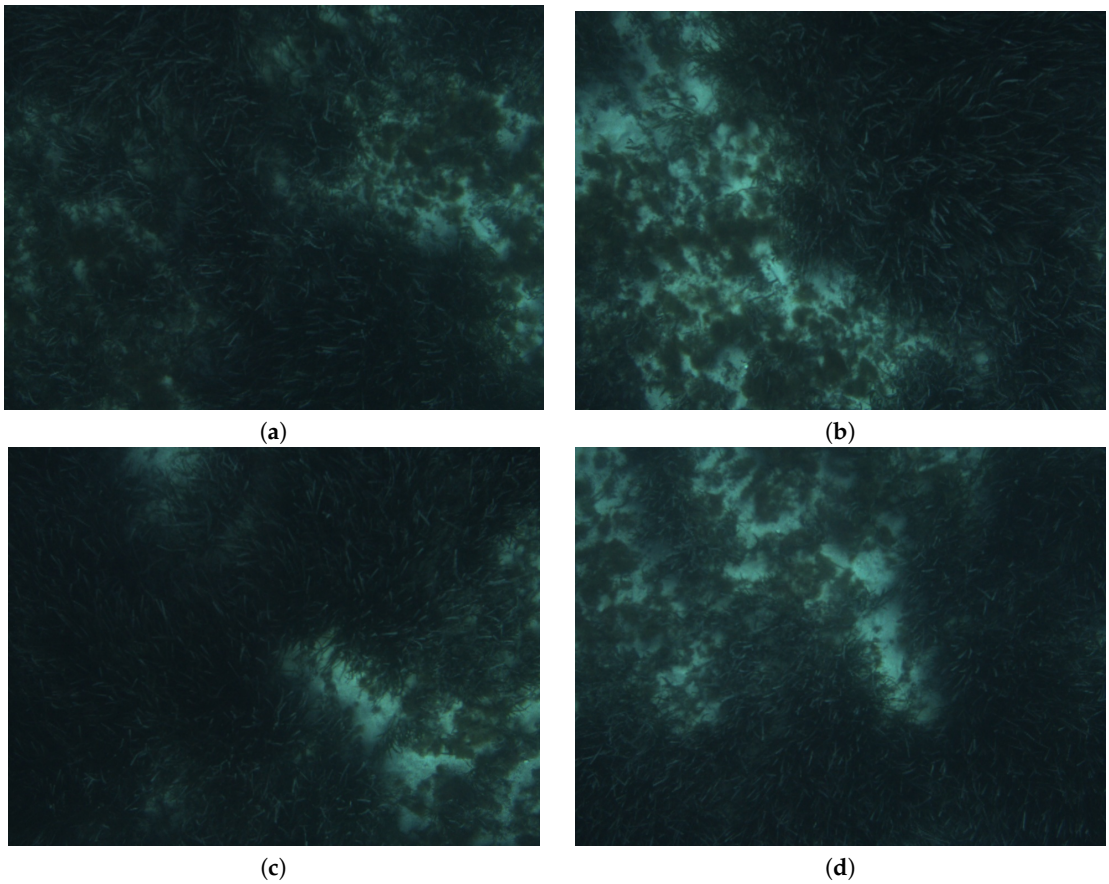
The camera altitude was computed at the beginning of each video sequence by means of a static visual marker of known size, placed at the sea bottom. This marker served as the origin and end point of both trajectories. From the $V1$ sequence, 226 key images were extracted, and 199 were extracted from $V2$.

A second pair of trajectories, $V3$ and $V4$, were recorded also by a diver in Port de Valldemossa far from $V1$ and $V2$ but using the same infrastructure as described above and navigating on the water surface at an approximate constant altitude of 4 m. In this case, the initial altitude was computed thanks to a static structure of known dimensions formed by markers and plastic tubes placed at the sea floor. A total of 209 key frames were selected, the first 152 corresponding to $V3$ and the next 57 corresponding to $V4$.

$V3$ and $V4$ also started and finished over one of the static markers forming the structure. Once this structure was deployed, it was not touched until $V3$ and $V4$ were grabbed. For all sequences, the altitude was assumed to be constant during each session and the video resolution was 1920×1080 pixels, grabbed at 30 frames per second, and prior to their use, all images were scaled down to 320×180 pixels.

Finally, a third pair of video sequences, namely $V5$ and $V6$, were recorded by the SPARUS II AUV [47] property of the University of the Balearic Islands, moving at a programmed constant altitude of 3 m in an area of 16 m depth. The altitude of the robot is well known and obtained from its navigation filter which integrates the DVL, an Inertial Measurement Unit (IMU), a pressure sensor, an Ultra Short Baseline (USBL) localizer, and a stereo 3-D odometer [48]. The aim of these video sequences is to test our proposal in larger environments with complex imagery due to the massive presence of Posidonia on the sea bottom. In particular, $V5$ was obtained along a trajectory of 93 m and $V6$ involved a 114-m mission. While gathering $V5$, the AUV was programmed to perform a rectangular loop of 20×15 m. As for $V6$, the AUV mission was to perform two smaller loops on one side of $V5$.

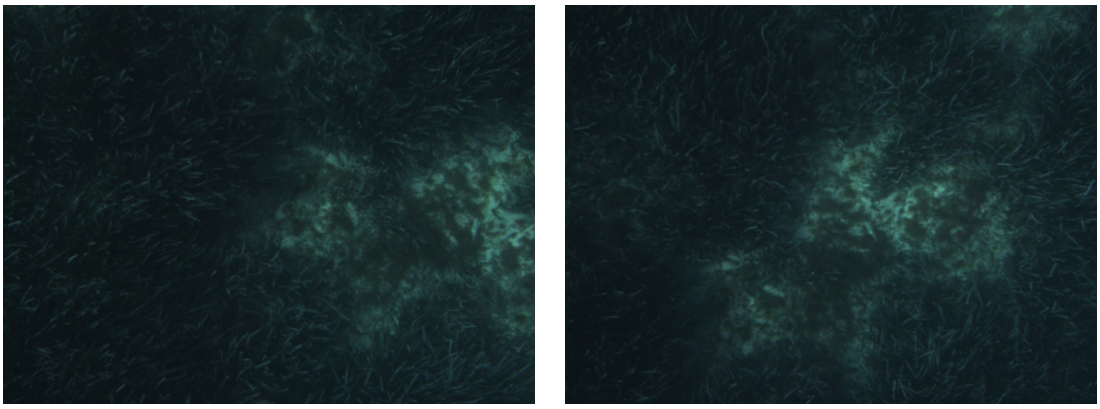
Figure 7 shows some images extracted from $V5$ and $V6$. As it can be observed, illumination was deficient, resulting in dark images with low contrast. In these types of environments, the altitude parameter is extremely important. Cameras at larger altitudes will provide wider fields of view and, thus, the possibility to find more loop closings, but conversely, if the illumination conditions are not optimal, especially at larger depths, the feature matching process can decrease its performance and affect directly the accuracy of the visual odometry calculation and the loop closing detection. In our environments and with our robot and its equipment, altitudes between 3 and 5 m give a good trade-off between image overlap and illumination conditions. A total of 400 key images were extracted, 200 belonging to $V5$ and 200 bwlonging to $V6$. In all these image sets, the Posidonia appears as the darker and/or greener areas, being the clearer areas such as stones, pebbles, or sand in the background.



(**a**) (**b**)

(**c**) (**d**)

Figure 7. *Cont.*



(**e**) (**f**)

Figure 7. Examples of images in $V5$ (**a**,**c**,**e**) and $V6$ (**b**,**d**,**f**): Each row shows images with partial overlap.

Given the extreme difficulty to obtain a ground truth trajectory in underwater environments without a Long Baseline (LBL) positioning infrastructure and specifically in medium or large areas colonized with seagrass, the assessment of the map-joining system has been done not from a vehicle factual ascertainable motion data but by obtaining, (a) on the one side, a qualified mosaic to provide visual evidences of correctness and, (b) on the other side and for the sake of a quantitative evaluation, the difference between a set of 2-D transformations (X_t -(x, y and yaw)) between image pairs that close local and inter-session loops obtained according to the estimated trajectories and the same 2-D transformations X_t obtained manually with Matlab (transformation ground truth) (see Sections 10.3 and 10.4).

10.2. Loop Closure Detection

Similarly to Ozog and Eustice [30], the consistency of a SLAM approach is now partially measured quantitatively by counting the number of visual loop closures and then through the derived parameters of precision, recall, and accuracy.

A total of 13 image pairs (11 in V1 and 2 in V2) were retrieved as single-session loop closings, using feature matching and RANSAC as described in Section 6 around a region surrounding the current image, resulting all true positives (TP). In order to assess the global loop detection using the signature method, all images of both video sequences $V1$ and $V2$ were hashed using HALOC. Equation (8) was applied by means of relating every key image (query image) of $V2$ with the 5 images (so-called candidates) of $V1$ that presented the lowest difference in terms of L1-norm between the hash of the query and the candidate. After this, all these preselected image pairs that presented a number of inliers (SIFT feature matching with RANSAC) higher than 25 were confirmed as loop closings. This threshold was set experimentally, since all the real (single and multi-session) loop closings presented more that 25 inliers and the overwhelmingly majority of image pairs that did not close loops had less than 10. However, this threshold needs to be set as a function of the observed environment. Thirty-five image pair candidates to close global loops (between $V1$ and $V2$) were found, from which 34 were TP and 1 false positive (FP), resulting in a precision, defined as $TP/(TP+FP)$, of 0.97.

Four hundred and twenty-five image pairs were labeled as non-loop closings, including local and global, and verified by visual inspection. Four hundred and two were true negatives (TN) (images that really did not close loops), and 23 turned out to be false negatives (FN) (loop closings classified as non-loop closings). The recall defined as $TP/(TP+FN)$ was 0.7927, and the accuracy defined as $(TP+TN)/(TP+TN+FP+FN)$ was 0.9533.

In summary, although some real loop closings are missed in the detection process (implicit in a recall slightly lower than a 80%), the percentage of true loop closings detected (single and multi-session) with respect to the total of loop closings finally proposed is close to 100%. Figure 8 shows a sample

of 3 pairs of single-session loop closings located in both image sequences $V1$ and $V2$. Dark areas correspond to Posidonia while clear areas correspond to stones and pebbles.

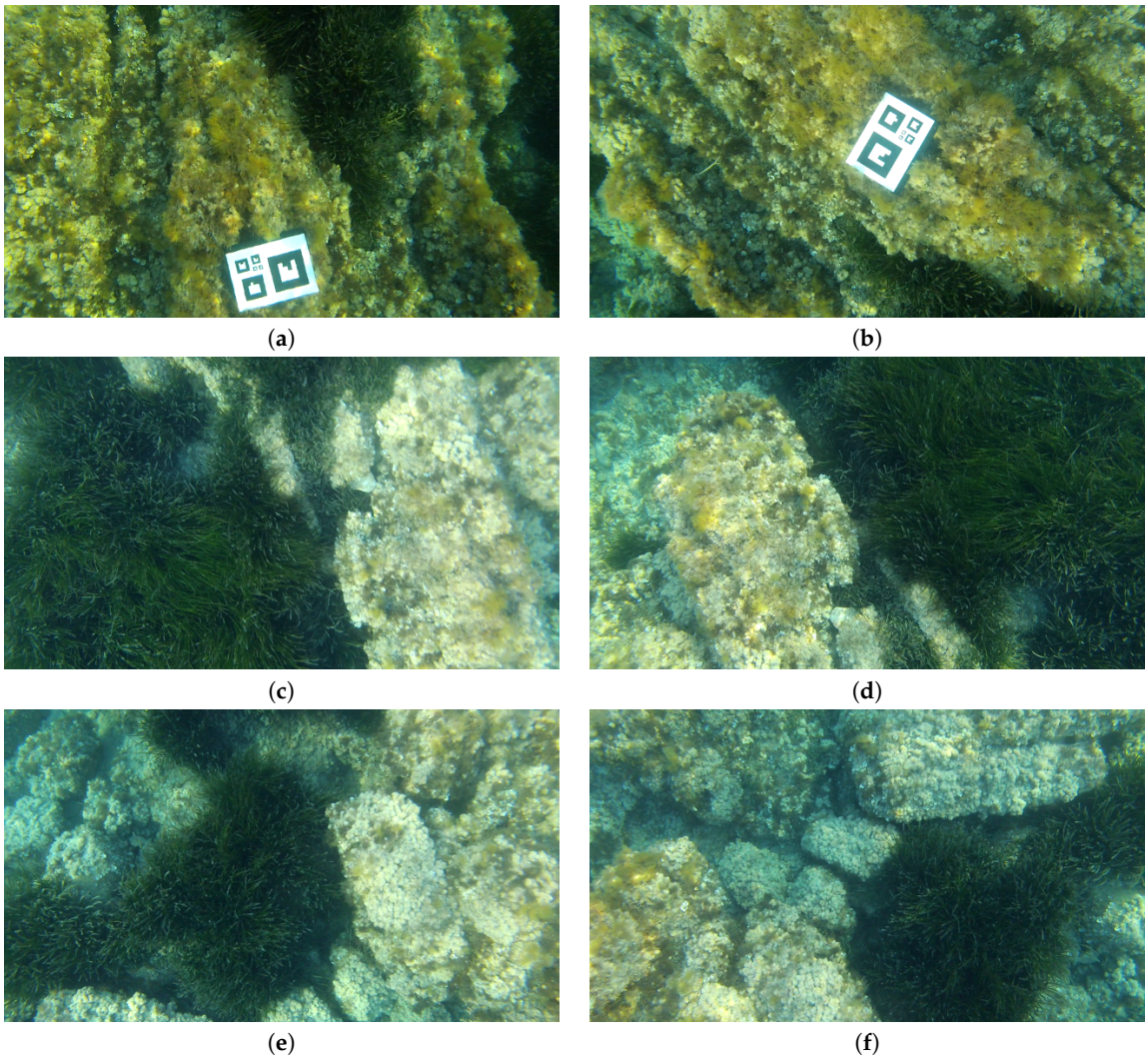


Figure 8. Local loops of sequences $V1$ and $V2$: Each image in the left column closes a loop with the corresponding image in the right column. (**a**,**b**) Sequence $V1$, Image 10 with Image 155. (**c**,**d**) Sequence $V1$, Image 143 with Image 27. (**e**,**f**) Sequence $V2$, Image 119 with Image 63.

Figure 9 shows a sample of 3 pairs of multi-session loop closings between the two image sequences. Again, dark areas show the Posidonia and clearer areas correspond to stones and pebbles.

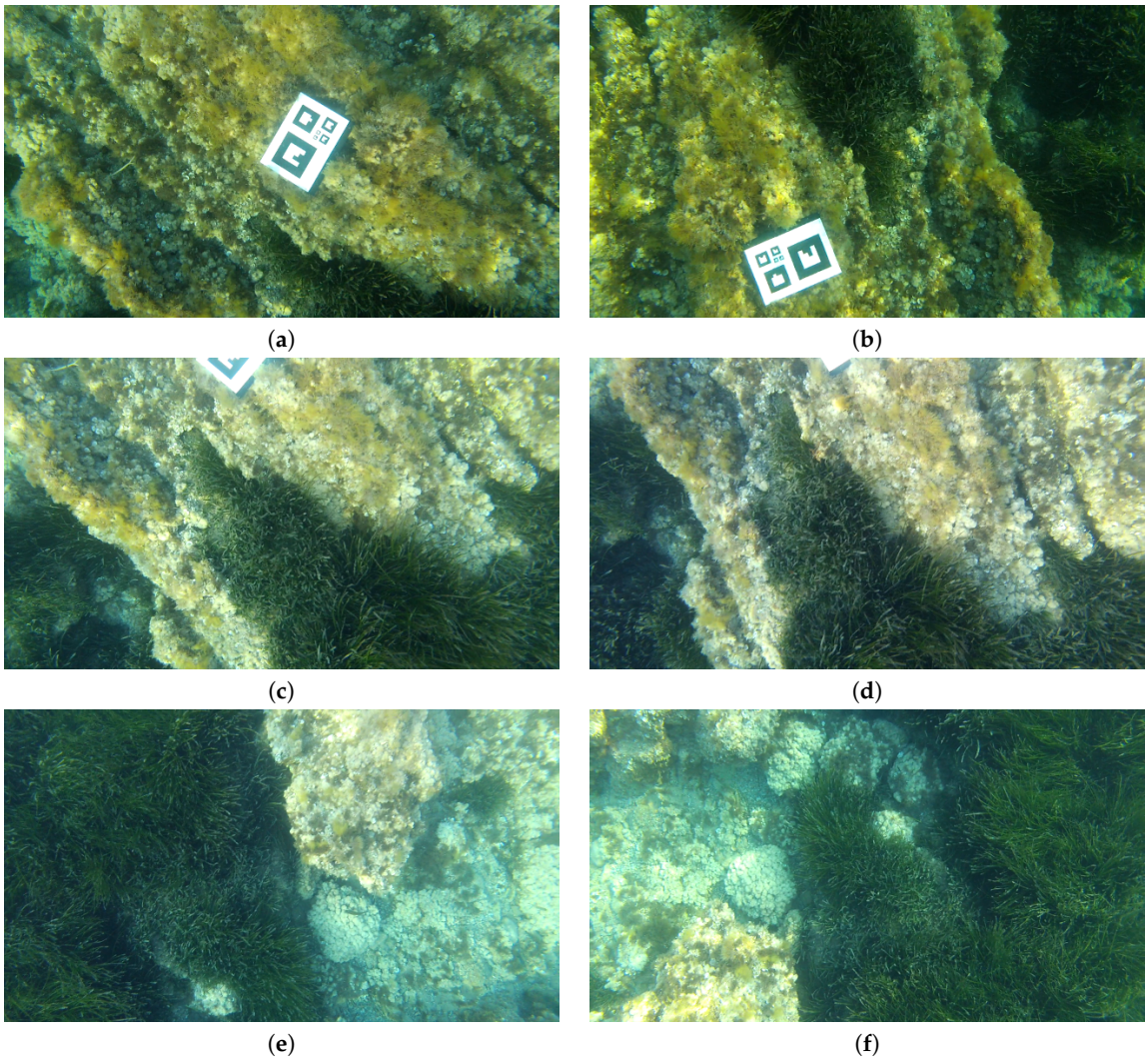


Figure 9. Multi-session loops between sequences $V1$ and $V2$: Each image in the left column closes a loop with the corresponding image in the right column. (**a**,**b**) Image 155 of Sequence $V1$ with Image 2 of Sequence $V2$. (**c**,**d**) Image 152 of Sequence $V1$ with Image 195 of Sequence $V2$. (**e**,**f**) Image 138 of Sequence $V1$ with Image 28 of Sequence $V2$.

An evaluation on $V3$ and $V4$ was done following exactly the same procedure used with $V1$ and $V2$ to retrieve all local and global loop closings. The feature matching threshold was also set to 25. Four hundred image pairs were labeled as negatives (no loop closings), and 130 were labeled as TP loop closings, 21 were labeled as multi-session, and 109 were labeled as single session, giving a precision of 1. The evaluation of these results was also done by visual inspection. From the 400 negatives, 349 were TN and 51 turned out to be FN. The recall result was 0.72, and the accuracy was 0.9. With a threshold of 25, some real loop closings are missed. However, a 100% in precision means that all loop closings that will be used in the SLAM process are true.

Figure 10 shows a sample of 3 pairs of single-session loop closings located in both image sequences $V3$ and $V4$. Dark areas again correspond to Posidonia.

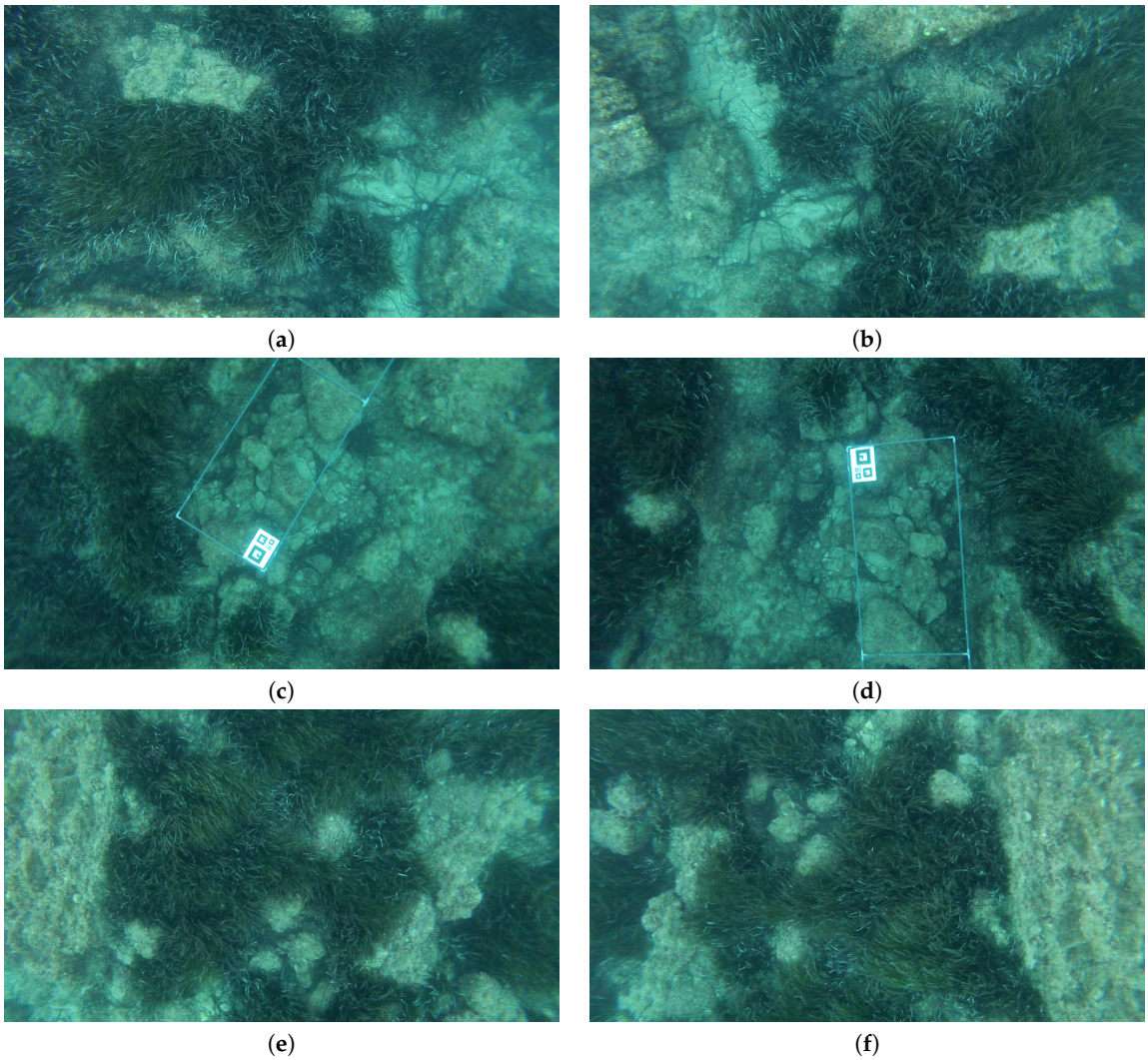


Figure 10. Local loops of sequences $V3$ and $V4$: Each image in the left column closes a loop with the corresponding image in the right column. (**a**,**b**) Sequence $V3$, Image 67 with Image 79. (**c**,**d**) Sequence $V4$, Image 155 with Image 208. (**e**,**f**) Sequence $V4$, Image 162 with Image 171.

Figure 11 shows a sample of 3 pairs of multi-session loop closings between the two image sequences $V3$ and $V4$.

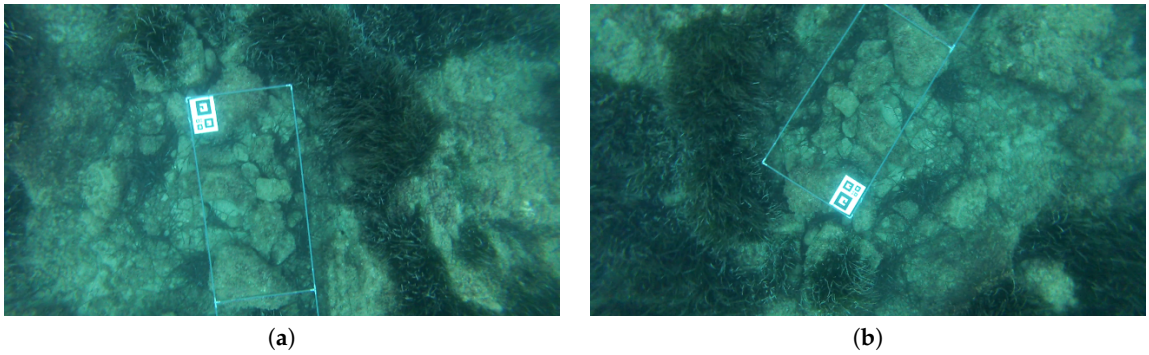


Figure 11. *Cont.*

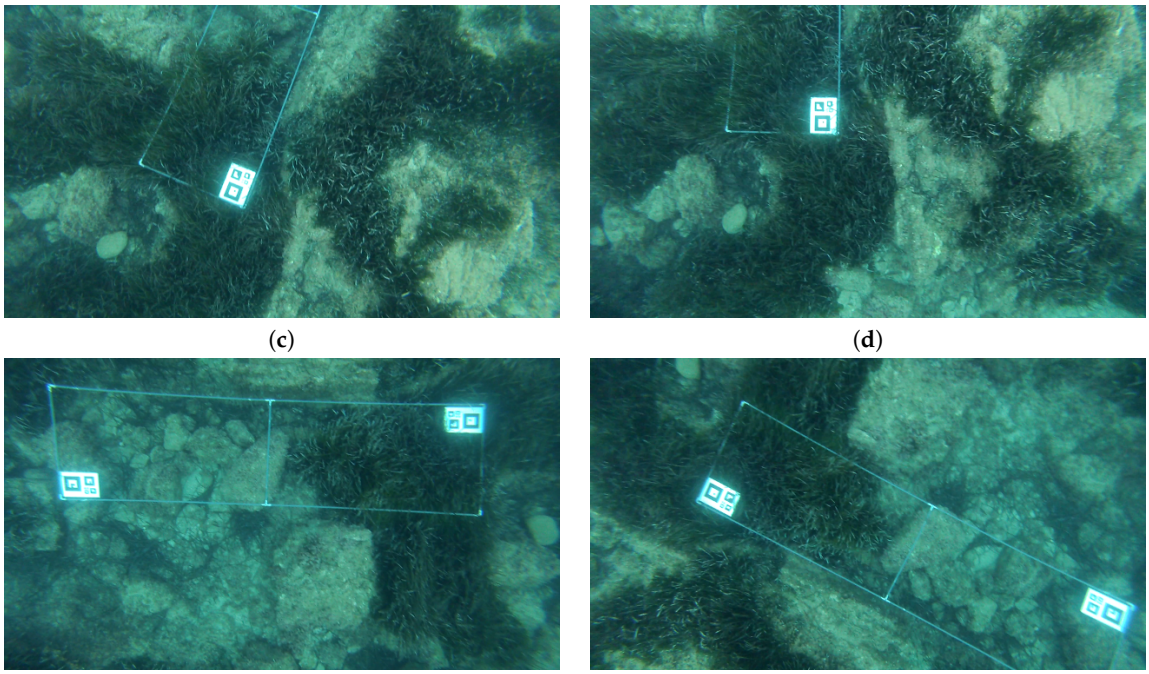


Figure 11. Multi-session loops between sequences $V3$ and $V4$: Each image in the left column closes a loop with the corresponding image in the right column. (**a**,**b**) Image 91 of Sequence $V3$ with Image 155 of Sequence $V4$. (**c**,**d**) Image 145 of Sequence $V3$ with Image 174 of Sequence $V4$. (**e**,**f**) Image 36 of Sequence $V3$ with Image 183 of Sequence $V4$.

A similar evaluation was performed with $V5$ and $V6$. In this case, a total of 59 loops between both sessions was found. Figure 7 shows some examples. The images in the left and right columns belong to $V5$ and $V6$, respectively. As it can be observed due to the kind of imagery that loops are hard to detect by simple visual inspection. Thus, these video sequences clearly show the loop detection capabilities of HALOC in these type of environments.

10.3. Multi-Session SLAM

In order to quantitatively evaluate our proposal, we have selected five image pairs of each video sequence that close a large local loop. By large local loop, we mean that the first and the second image in each pair overlap but that, between the times at which these images were gathered, the camera field of view did not overlap any of them for at least 40 s. Actually, this time surpasses two minutes in five of the ten image pairs and three minutes in two of them.

For each of these image pairs, we have computed manually the exact pose of one image with respect to the other, building two sets of *local ground truth*, $G1$ and $G2$, defined as $G1 = \{G1^i_j = (x1^i_j, y1^i_j, \theta1^i_j), (i,j) \in loops(V1,V1)\}$ and $G2 = \{G2^i_j = (x2^i_j, y2^i_j, \theta2^i_j), (i,j) \in loops(V2,V2)\}$, which include the relative poses between images. Similarly, we have selected five image pairs that close global loops, computing the exact pose of the second image in each pair with respect to the first one, building a *global ground truth* $G3 = \{G3^i_j = (x3^i_j, y3^i_j, \theta3^i_j), (i,j) \in loops(V1,V2)\}$.

Thereafter, let us define the error ek as the average distance between the relative positions according to the ground truth Gk and the corresponding relative positions according to the estimated trajectory as shown in Equation (35). In this Equation, (x^i_j, y^i_j) denotes the relative position of image j with respect to image i according to the trajectory being evaluated.

$$ek = \Big(\sum_{Gk^i_j \in Gk} \sqrt{(xk^i_j - x^i_j)^2 + (yk^i_j - y^i_j)^2} \Big) / |Gk| \tag{35}$$

Overall, the quality of the vehicle pose estimated by the odometry, the single-session SLAM within the first ($e1$) and second video sequences ($e2$) and between both video sequences (multi-session SLAM) can be computed.

Figures 12 and 13 depict the trajectories for $V1$ and $V2$. In these figures, SLAM refers to each single-session separately and MSLAM denotes the multi-session SLAM. The number after MSLAM states the number of global loops used to join the maps. That is, the number refers to K in Equations (21) and (22). Therefore, MSLAM1 corresponds to joining the maps after the first global loop was detected. MSLAM10, MSLAM20, and MSLAM30 delay the map joining until 10, 20, and 30 global loops were found, respectively.

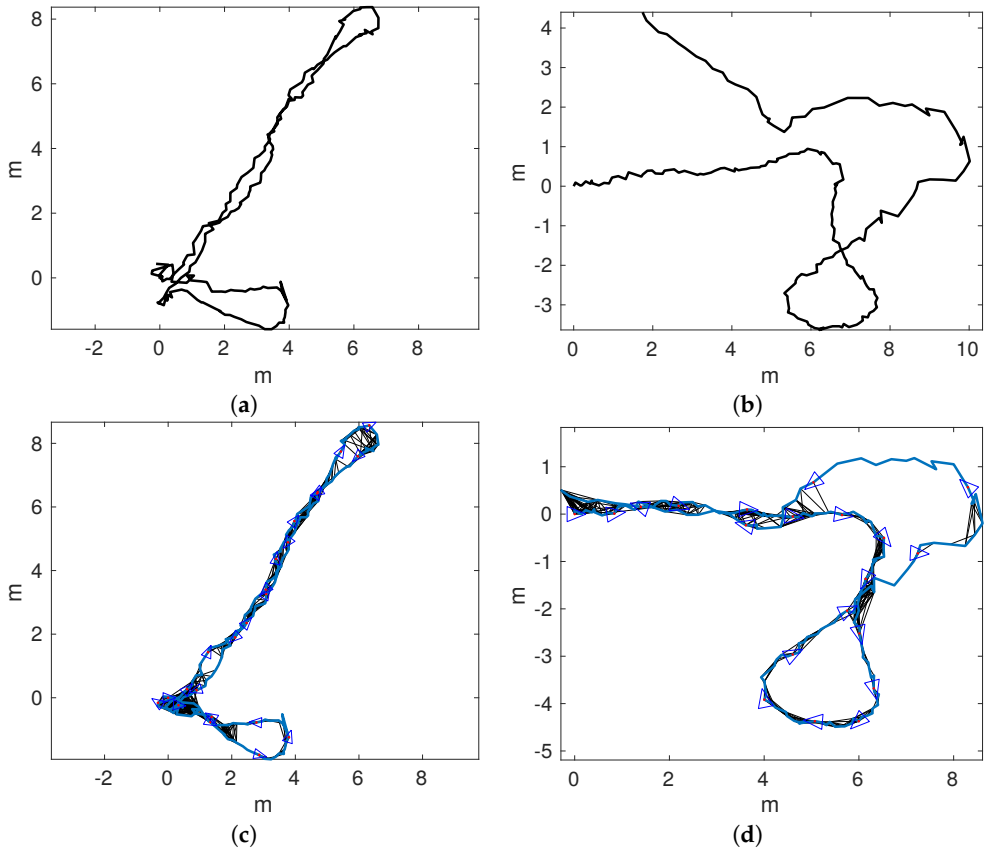


Figure 12. Obtained trajectories for $V1$ and $V2$: (**a**) Session 1 odometry, (**b**) session 2 odometry, (**c**) session 1 SLAM, and (**d**) session 2 SLAM.

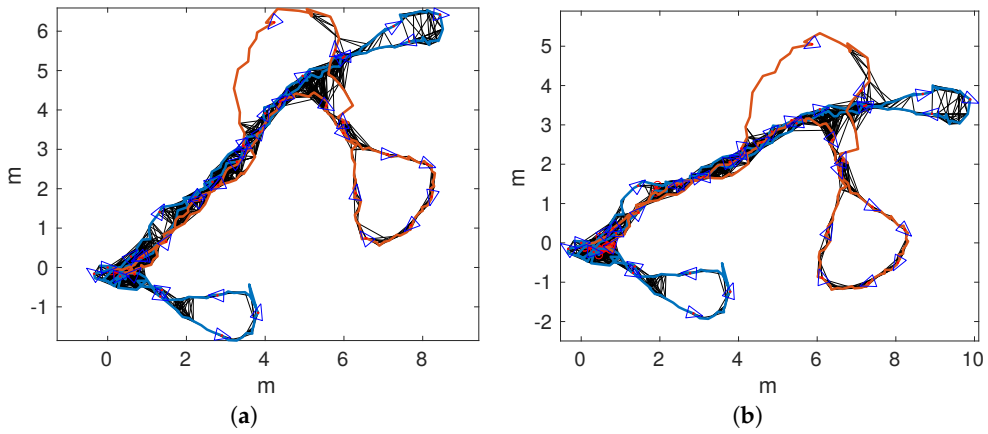


Figure 13. *Cont.*

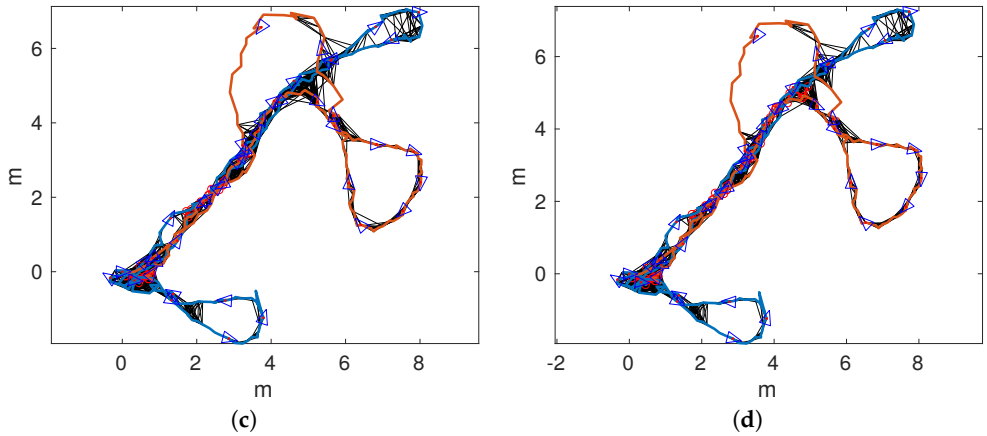


Figure 13. Obtained trajectories for $V1$ and $V2$: (**a**) MSLAM1, (**b**) MSLAM10, (**c**) MSLAM20, and (**d**) MSLAM30. In SLAM, thick lines are the trajectories, thin lines are the detected loops, and the triangles show the estimated AUV orientation.

Table 1 summarizes the obtained quantitative results. The time consumption has been measured in seconds on a Matlab implementation executed in a standard laptop (i7 CPU at 3.1GHz using a single core) running Ubuntu 16.04LTS. The odometry error is expressed in meters, and it is reasonably low during the first session but increases considerably in the second one. Thus, SLAM (either single-session and multi-session) provided a slight improvement to the first session but a very large improvement during the second one. It can also be observed that the results corresponding to each session are similar if we compare the single-session approach to the multi-session one.

Table 1. Summary of the results.

	SESSION 1		SESSION 2		MSLAM	
Method	**e1 (m)**	**Time (seg.)**	**e2 (m)**	**Time (seg.)**	**e3 (m)**	**Time (seg.)**
Odometry	0.473	54.6	3.002	49.0	-	-
SLAM	0.303	148.7	0.352	117.1	-	-
MSLAM1	0.300	-	0.336	-	0.310	488.6
MSLAM10	0.302	-	0.347	-	0.323	465.6
MSLAM20	0.301	-	0.351	-	0.319	432.6
MSLAM30	0.300	-	0.351	-	0.322	385.6

The different delays to join the maps (10, 20, and 30) barely affect the quality of the final estimates since errors are almost constant. To the contrary, the execution time is reduced to 78% if the maps are joined when 30 loops are detected instead of joining them with the first loop. Thus, delaying loop closure is responsible for a large reduction in time consumption.

In the Matlab implementation, the CPU usage was, on average, 24.3% for the odometry, 62.4% for the single-session SLAM, and 90.5% for the MSLAM30 map joining procedure. A CPU usage far below these percentages can be expected with an optimized C++ code instead of Matlab. Although out of the scope of this paper, it is worth mentioning that a C++ implementation is currently being developed by the signing authors under the ROS middleware [49] to be installed in our AUV and to be run online during the missions. Additionally, given the high speed at which the C++ implementation of HALOC is able to propose loop closing candidates [12], the use of the whole system on-line, in our view, is perfectly feasible. Nonetheless, a further complete assessment will be necessary.

Figure 14a,b shows the odometric trajectories of $V3$ and $V4$, respectively. Figure 14c,d depicts the local SLAM trajectories of $V3$ and $V4$, respectively. Finally, Figure 15a–d shows the connected maps, delaying the joining step until 1, 5, 10, and 15 loop closings were detected. Since both sequences

started and finished over the structure, it is easy to see that the odometry drifts at the end of $V3$ and, to a lesser extend, in $V4$.

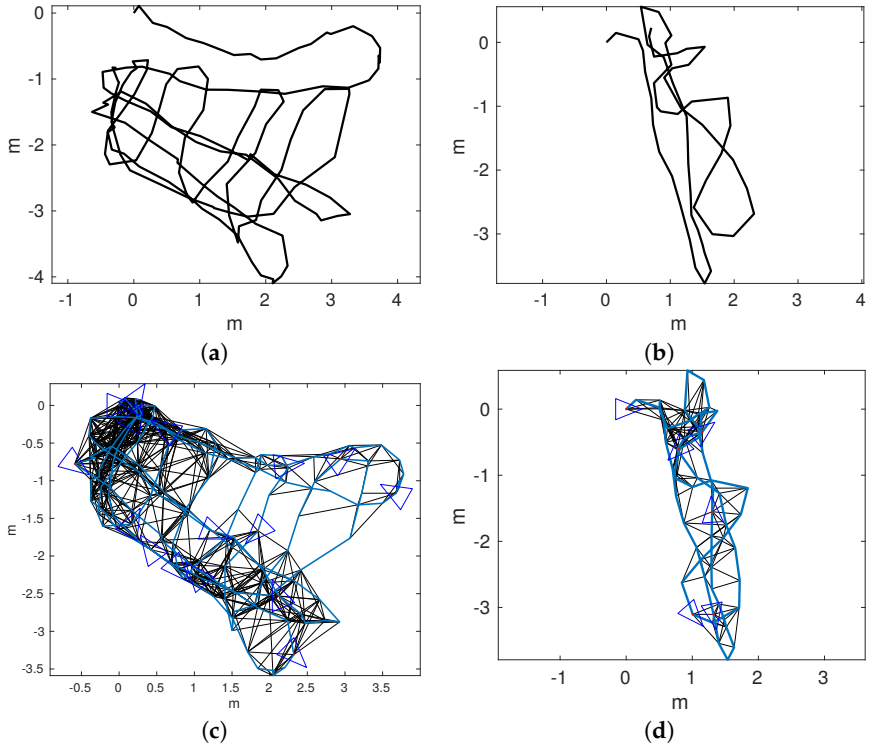


Figure 14. Obtained trajectories for $V3$ and $V4$: (**a**) Session $V3$ odometry, (**b**) session $V4$ odometry, (**c**) session $V3$ SLAM, and (**d**) Session $V4$ SLAM.

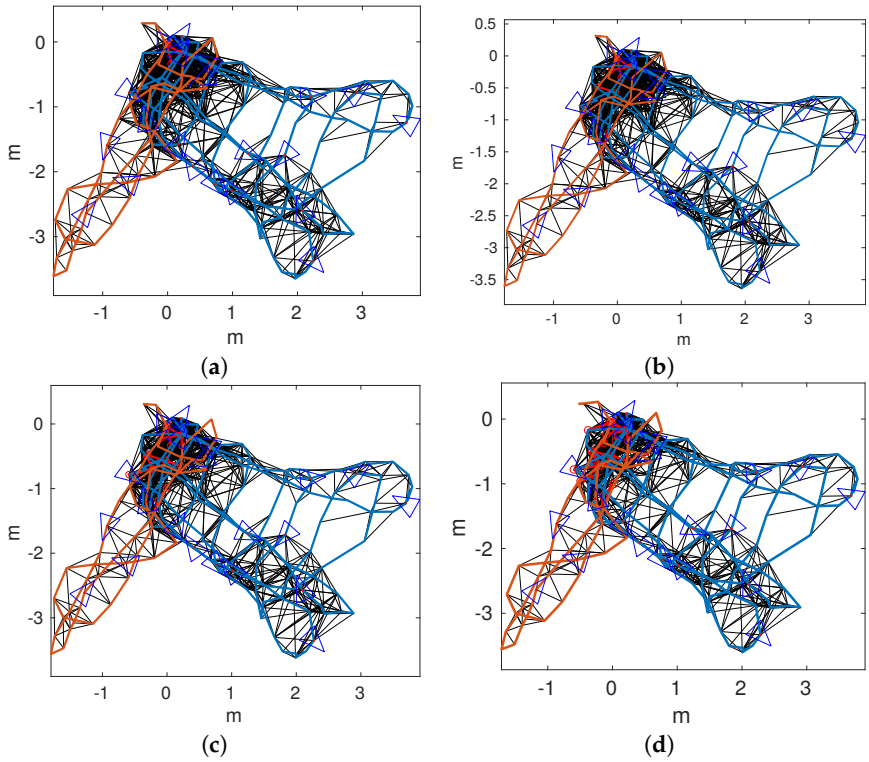
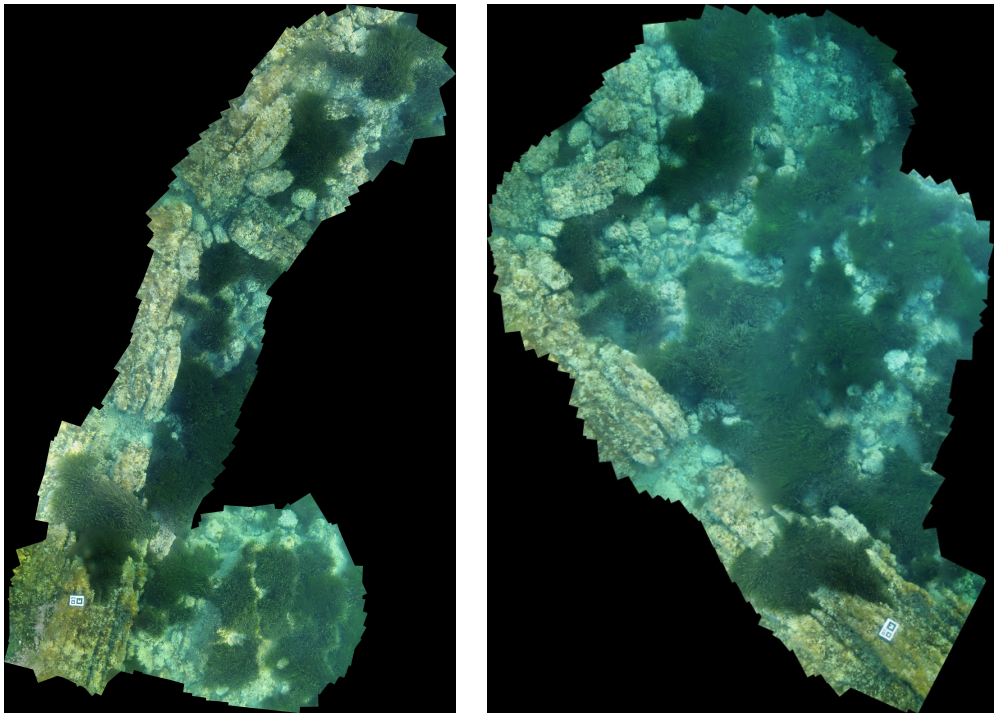


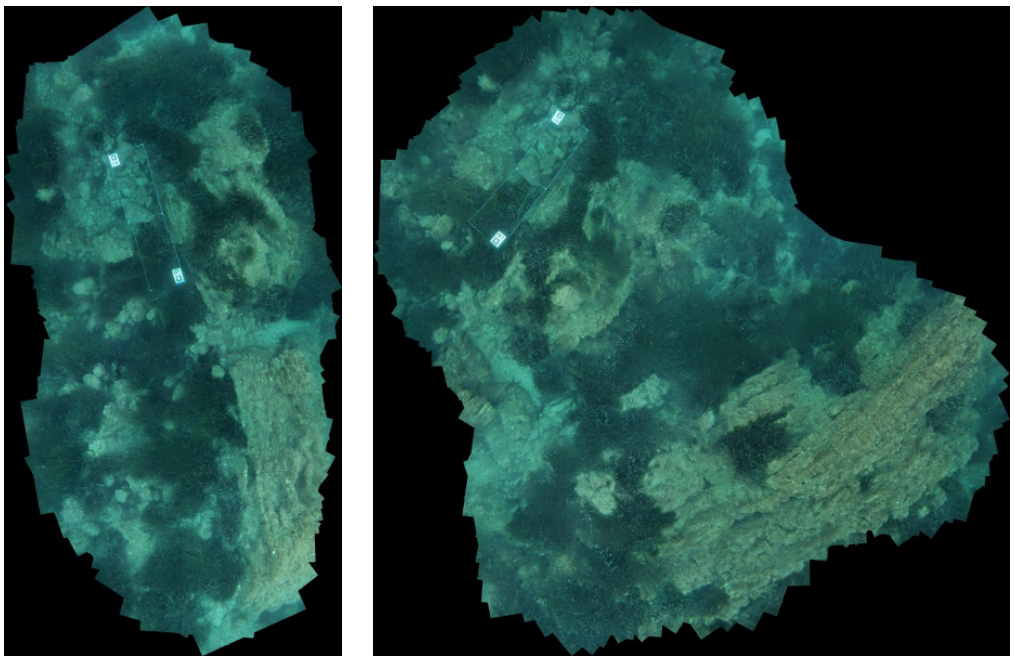
Figure 15. Obtained trajectories for $V3$ and $V4$: (**a**) MSLAM1, (**b**) MSLAM5, (**c**) MSLAM10, and (**d**) MSLAM15. In SLAM, thick lines are the trajectories, thin lines are the detected loops, and the triangles show the estimated AUV orientation.

Figures 16 and 17 show four photo-mosaics corresponding to sequences $V1$, $V2$, $V3$, and $V4$, included for the sake of an easy and fast visual qualitative evaluation of the joined trajectories. Both mosaics were built with the same key frames used in the SLAM processes and *Binary descriptor-based Image Mosaicing* (BIMOS)[50]. BIMOS-related references [50–52] already showed the good performance of this mosaic approach in terms of accuracy and execution time and of application in underwater environments with seagrass (see Bonin-Font et al. [51]). Consequently, its assessment is out of the scope of this paper.



(**a**) (**b**)

Figure 16. (**a**) Photo-mosaic of $V1$ and (**b**) photo-mosaic of $V2$.



(**a**) (**b**)

Figure 17. (**a**) Photo-mosaic of $V3$ and (**b**) photo-mosaic of $V4$.

An illustrative video of the whole process can be seen at https://youtu.be/NW7H5vbYQvU.

The marker visible on mosaics corresponding to $V1$ and $V2$ indicate the start and end of both trajectories. The relative position of the mosaics correspond to $V1$-$V2$ and $V3$-$V4$ with respect the single marker in the formers, and the structure in the later coincide very closely with the form of the resulting joined maps.

A final experiment has been performed, aimed at showing the quality of our proposal in front of larger trajectories and bad illumination conditions. To this end, $V5$ and $V6$ have been used.

Figure 18 shows the obtained trajectories. In particular, Figure 18a,b shows the trajectories according to pure visual odometry. In both cases, significant odometric errors lead to trajectories that barely resemble the actual mission performed by the AUV.

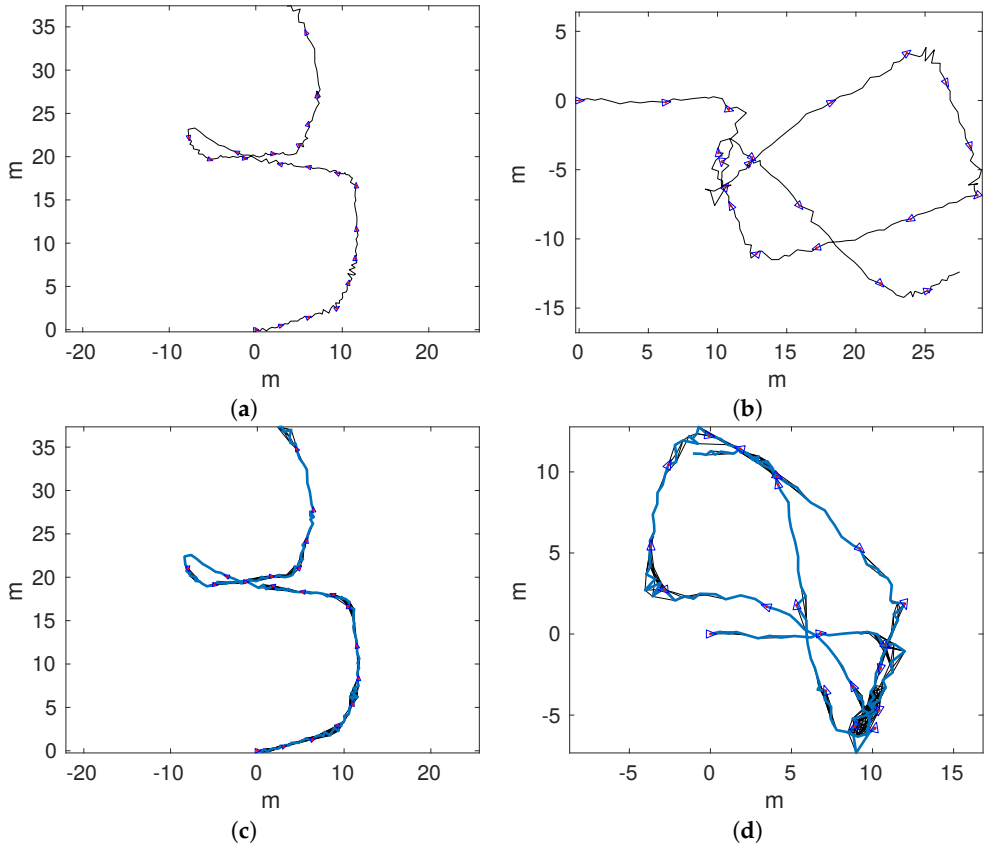


Figure 18. Obtained trajectories for $V5$ and $V6$: (**a**) Session $V5$ odometry, (**b**) session $V6$ odometry, (**c**) session $V5$ SLAM, and (**d**) session $V6$ SLAM.

Figure 18c,d shows the trajectories obtained by single session SLAM. In this case, $V6$ was substantially corrected. However, $V5$ remains almost unchanged with respect to pure odometry. In this case, the odometric error was so large that the geometric constrains prevented the detection of relevant loops.

Figure 19a,d shows the trajectories according to multi-session SLAM if maps are joined after detecting 1, 10, 20, or 30 global loops. As can be observed, in all cases, the overall structure of $V6$ is recovered, thus leading to a large improvement with respect to single session SLAM.

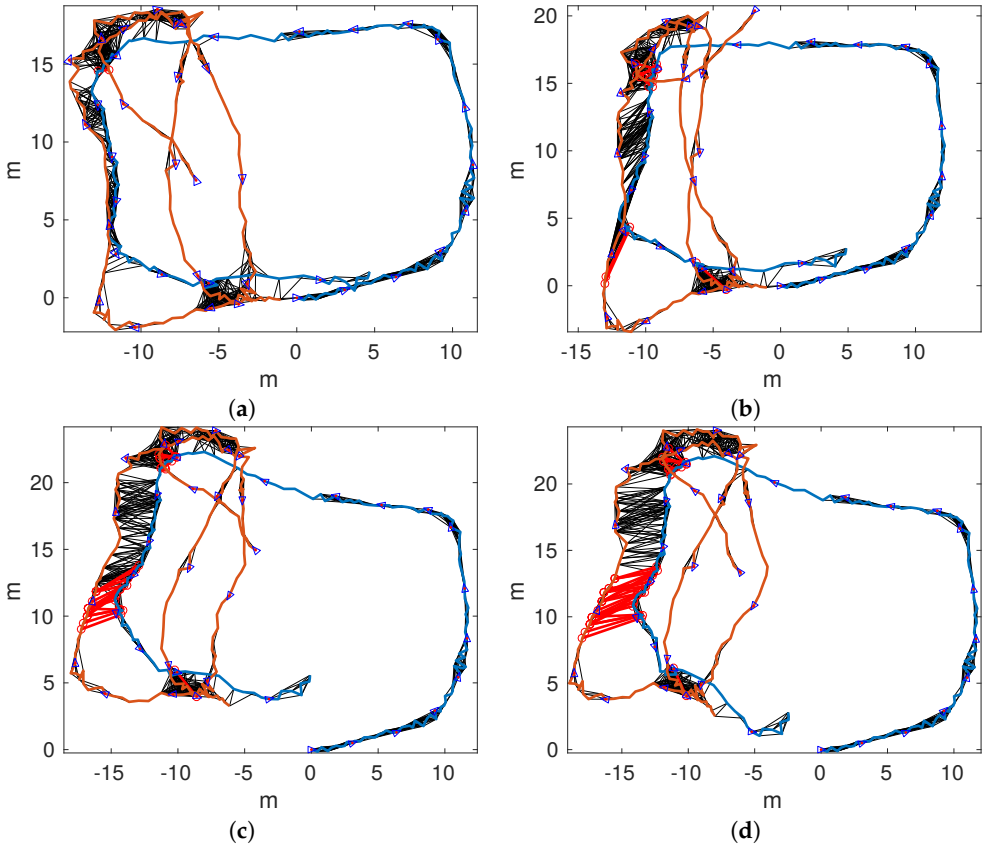


Figure 19. Obtained trajectories for $V5$ and $V6$: (**a**) MSLAM1, (**b**) MSLAM10, (**c**) MSLAM20, and (**d**) MSLAM30. In SLAM, thick lines are the trajectories, thin lines are the detected loops, and the triangles show the estimated AUV orientation.

It can also be observed that the larger the delay to join the trajectories, the worse is the result. For example, for MSLAM20 and MSLAM30, the loops involving the start and the end of $V5$ have not been detected. This decrease of the quality with the delay is reasonable and consistent with previous results: as the delay increases, less local loops are available to improve the joined trajectory.

Table 2 shows the execution times. As can be observed, the multi-session execution time is significantly larger that of single session SLAM. This is mainly due to the fact that few local loops are detected in single session SLAM, especially in $V5$. These results also show that the larger the delay to join the trajectories, the lower the execution time. For example, switching from a delay of one to a delay of 30 leads to a reduction of 38.57% of the execution time.

Table 2. Execution times for $V5$ and $V6$.

	V5	V6	MULTISESSION
Odometry	49.804 s	47.649 s	-
SLAM	66.267 s	82.569 s	-
MSLAM1	-	-	379.228 s
MSLAM10	-	-	262.387 s
MSLAM20	-	-	246.199 s
MSLAM30	-	-	232.960 s

10.4. Robustness

In order to test the robustness of our proposal, we have corrupted the odometric estimates of $V1$ and $V2$ with five different levels of additive zero mean Gaussian noise. The two sigma bounds of these noises ranged from 5 cm in x and y and 5° in orientation for noise level 1 to 25 cm in x and y and 25° in orientation for noise level 5.

Each corrupted odometry has been used to perform single and multi-session SLAM with different loop-closing delays.

MSLAM1, MSLAM10, MSLAM20, and MSLAM30 denote that the maps were joined after detecting 1, 10, 20, and 30 global loops, respectively. The local and the global errors have been recorded in each case. This experiment has been repeated 60 times for each noise level. Table 3 summarizes the obtained results by showing the mean ($\bar{X}$) and the standard deviations (σ) of the obtained errors for each noise level (N.). MSSession denotes the multi-session SLAM. All values are expressed in meters.

Table 3. Error $\bar{X}$ and σ of synthetically corrupted odometry.

N.	Method	Session 1		Session 2		MSSession	
		$\bar{X}$(e1)	σ(e1)	$\bar{X}$(e2)	σ(e2)	$\bar{X}$(e3)	σ(e3)
LEVEL 1	**Odometry**	1.424	0.832	3.441	1.759	-	-
	SLAM	0.304	0.003	0.347	0.002	-	-
	MSLAM1	0.299	0.004	0.352	0.022	0.320	0.004
	MSLAM10	0.300	0.001	0.351	0.002	0.318	0.001
	MSLAM20	0.302	0.001	0.350	0.002	0.318	0.002
	MSLAM30	0.300	0.001	0.350	0.003	0.316	0.001
LEVEL 2	**Odometry**	2.747	1.539	4.565	2.191	-	-
	SLAM	0.304	0.006	0.346	0.005	-	-
	MSLAM1	0.300	0.004	0.346	0.021	0.319	0.004
	MSLAM10	0.300	0.001	0.350	0.005	0.319	0.002
	MSLAM20	0.302	0.001	0.350	0.005	0.318	0.003
	MSLAM30	0.300	0.001	0.352	0.004	0.317	0.003
LEVEL 3	**Odometry**	3.686	1.660	6.081	2.802	-	-
	SLAM	0.302	0.008	0.374	0.111	-	-
	MSLAM1	0.299	0.004	0.361	0.051	0.321	0.015
	MSLAM10	0.300	0.001	0.355	0.019	0.320	0.008
	MSLAM20	0.302	0.001	0.351	0.008	0.318	0.003
	MSLAM30	0.301	0.001	0.352	0.012	0.318	0.007
LEVEL 4	**Odometry**	4.912	2.304	6.520	3.321	-	-
	SLAM	0.302	0.010	0.418	0.298	-	-
	MSLAM1	0.299	0.003	0.368	0.072	0.323	0.025
	MSLAM10	0.300	0.001	0.355	0.017	0.319	0.006
	MSLAM20	0.302	0.002	0.360	0.064	0.321	0.027
	MSLAM30	0.301	0.001	0.351	0.009	0.315	0.007
LEVEL 5	**Odometry**	4.796	2.019	6.363	3.258	-	-
	SLAM	0.301	0.014	0.453	0.406	-	-
	MSLAM1	0.300	0.004	0.368	0.080	0.324	0.026
	MSLAM10	0.300	0.002	0.362	0.043	0.320	0.009
	MSLAM20	0.303	0.001	0.348	0.022	0.316	0.009
	MSLAM30	0.301	0.002	0.355	0.044	0.319	0.021

Figure 20 compares the errors of each tested SLAM approach to those of odometry as a function of the noise level. Firstly, both SLAM errors, single and multi-session, are always below the odometric error. SLAM is able to reduce odometric errors larger than 6 m to approximately 30 cm. Secondly, the resulting SLAM errors are barely influenced by the odometric error. Therefore, within the tested ranges of noise, our proposal is almost independent of the initial conditions.

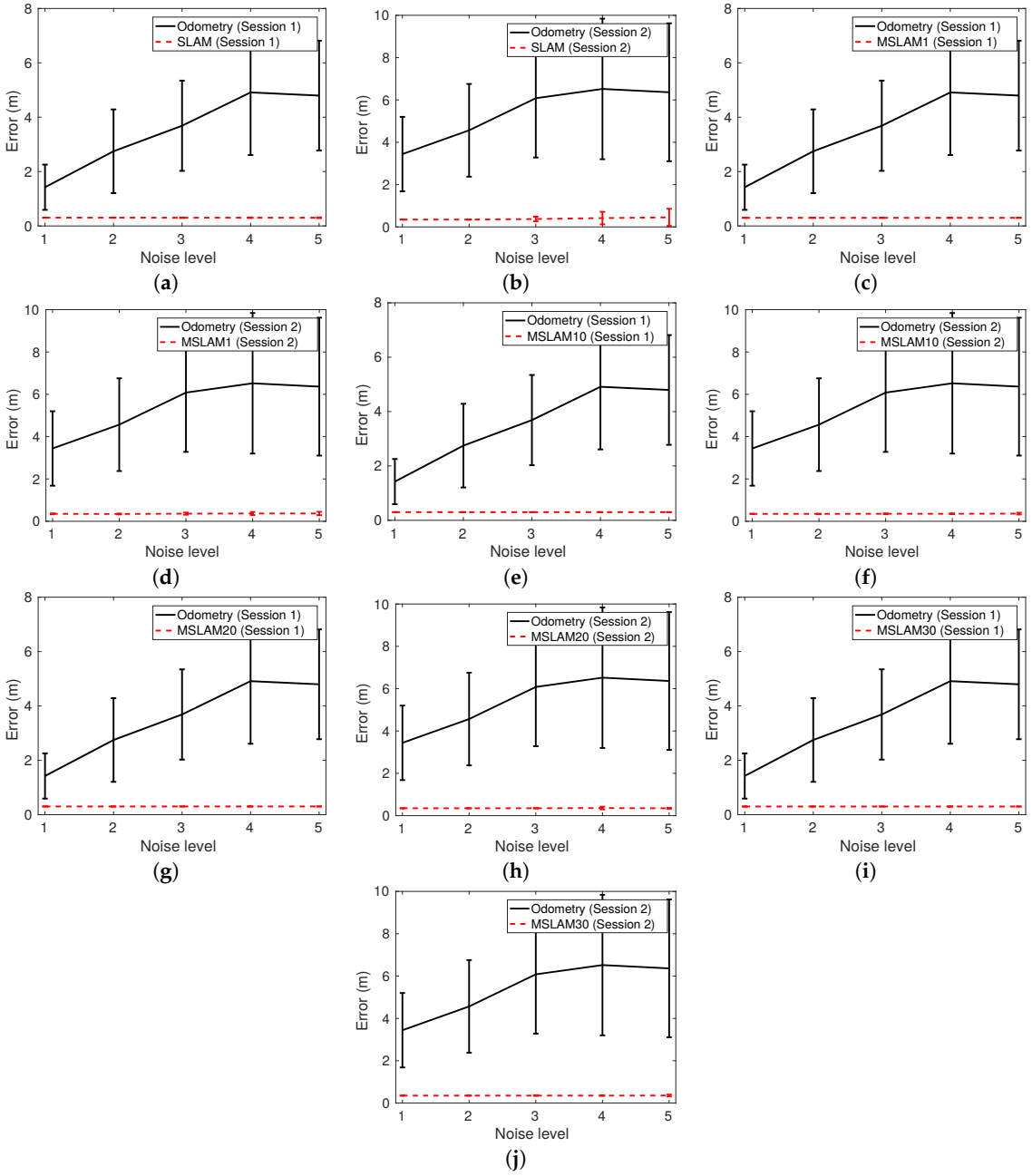


Figure 20. Mean and standard deviations of the errors in distance using synthetically corrupted odometry: (**a**) SLAM session 1 and odometry, (**b**) SLAM session 2 and odometry, (**c**) MSLAM1 session 1 and odometry, (**d**) MSLAM1 session 2 and odometry, (**e**) MSLAM10 session 1 and odometry, (**f**) MSLAM10 session 2 and odometry, (**g**) MSLAM20 session 1 and odometry, (**h**) MSLAM20 session 2 and odometry, (**i**) MSLAM30 session 1 and odometry, and (**j**) MSLAM30 session 2 and odometry.

Additionally, even though the standard deviations of all the SLAM approaches are very small, those corresponding to single session SLAM tend to increase with the noise level. This can be clearly appreciated in Figure 20b. However, the covariances of the error corresponding to multi-session SLAM

are barely influenced by the noise level. This suggests that joining the maps reinforces the stability of the pose estimates.

Finally, these results also show that delaying the map joining has negligible effects on the resulting quality as the errors for MSLAM1, MSLAM10, MSLAM20, and MSLAM30 are almost identical. However, delaying the map joining is responsible for a significant reduction in computation time, so it is advisable to delay them as much as possible.

Figure 21 depicts the means and standard deviations of the MSLAM approach. The initial error barely influences the results except for error level 5, which leads to larger error variability. Nonetheless, even in the worst case, the standard deviation is only 0.021 m. It can also be observed that the delay to join the maps has almost no influence on the final results.

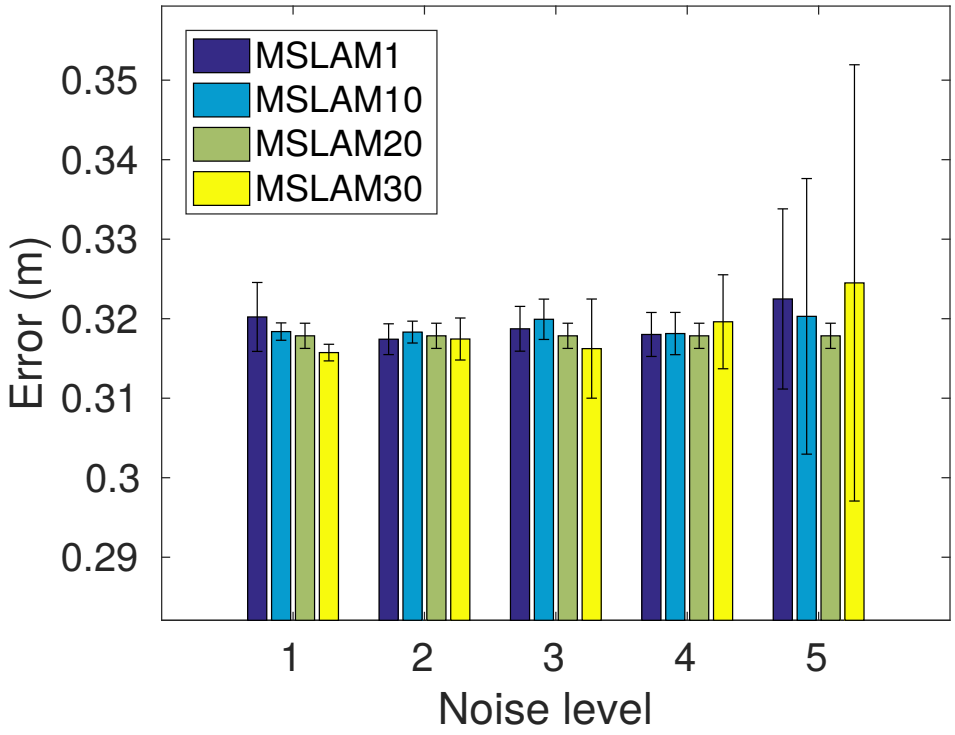


Figure 21. Global error depending on the noise level and the number of loops for map joining.

Figure 22a,b depicts the odometry corresponding to both trajectories corrupted with noise level 5. As can be observed, there is no visual resemblance between them and their non-corrupted counterparts in Figure 12a,b. Figure 22c,d shows the output of our proposed single-session SLAM using the corrupted trajectories, evidencing a huge improvement. Finally, Figure 22e shows the output of MSLAM30. As can be observed, the results (therefore, the measured errors) are extremely similar to the case in which non-corrupted odometry was used (Figure 12h).

Some additional experiments performed with larger noise levels show that the system leads to similar results except for noise levels that are not possible in real operation, such as errors in x or y that surpass the camera field of view. This is due to two main reasons. On the one hand, Algorithm 1 is particularly robust. Because of that, increasing the search radius δ of Equation (4) to account for larger odometric errors leads to larger execution times but not to an increase of false positives. Thus, large values for δ can be used even in low noise situations. On the other hand, global loop detection does not depend on the quality of the odometry. Accordingly, both map joining and the use of global loops to correct the joined trajectory are not affected by the noise level.

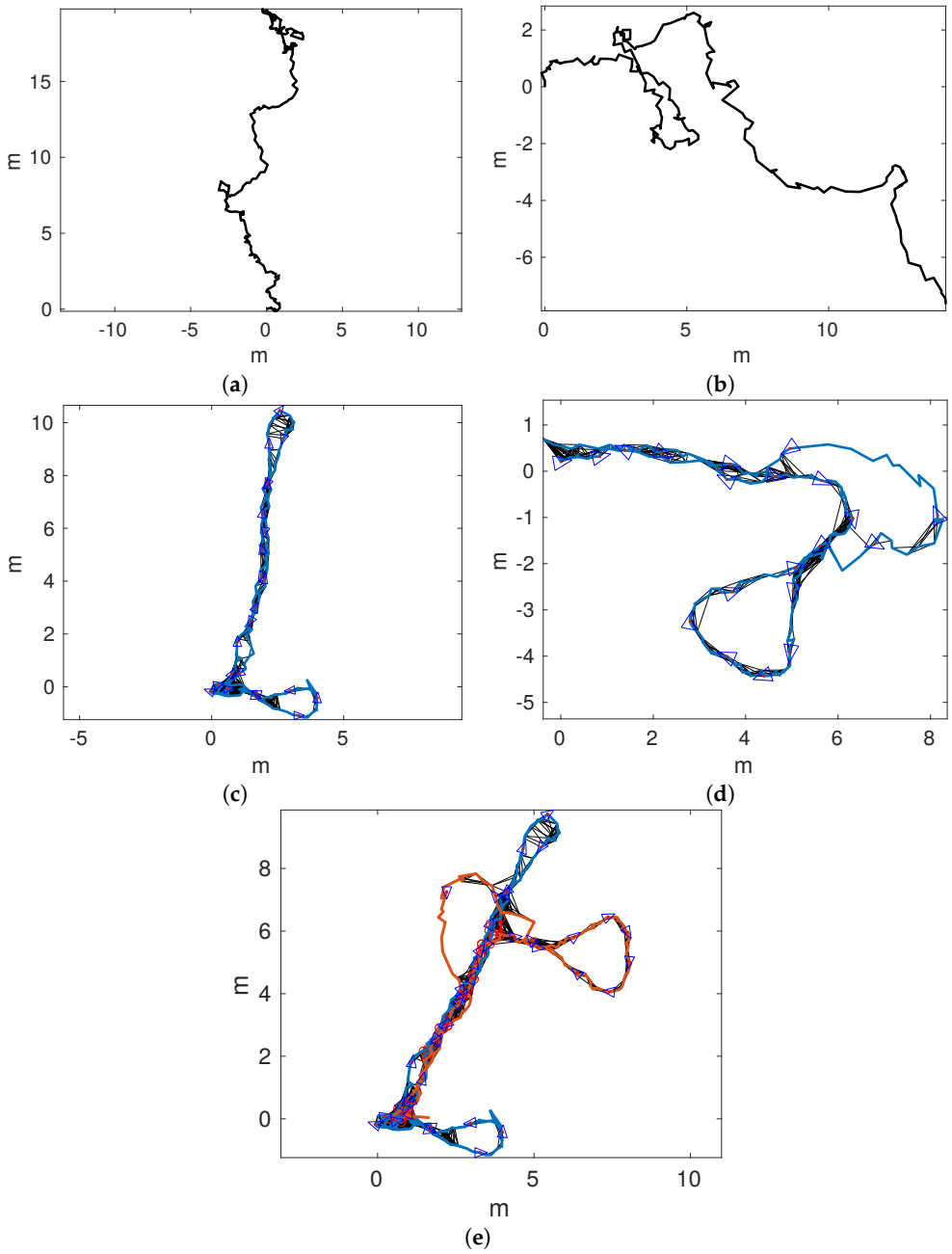


Figure 22. Example of multi-session SLAM using odometry corrupted with noise level 5: (**a**) Corrupted odometry of session 1, (**b**) corrupted odometry of session 2, (**c**) single-session SLAM of session 1, (**d**) single-session SLAM of session 2, and (**e**) multi-session SLAM.

11. Conclusions

In this paper, we have presented a trajectory-based multi-session monocular SLAM approach aimed at underwater environments which has three main blocks: a visual odometer, two different loop detectors, and an optimizer. The visual odometer gives a first estimate of the camera motion in each single session. The single and multi-session loop closing detection add additional pose constraints to each individual sequence and between both sessions. Finally, the optimizer is in charge of improving the trajectories (intra-map optimization) and joining different sessions (map joining).

The main contributions of this paper are as follows: (a) Due to the lack of geometric information relating two different sessions, the multi-session loops are found using an image global signature (hash) fast matching based on HALOC which is already tested on underwater imagery. (b) The capacity of

the map-joining algorithm to preserve the trajectory structure by adding a single link between the joined sessions; (c) to perform a delayed global graph optimization, reducing computational effort without loss of localization accuracy; and (d) to disaggregate sessions, reducing the computational cost whenever is necessary and joining them again later are improved. (e) A complete set of source codes that implement the whole process is available for the community.

Experimental results have been obtained from several video sequences taken with a single camera at subsea environments located in Mallorca. The results of the assessment of each particular component (odometry, loop detection, and optimization) in terms of quality and robustness evidence a high quality in the resulting trajectories. Moreover, the results using extremely corrupted odometry suggest that our proposal is barely influenced by odometric errors. On the other hand, the system has a couple of limitations, which are being addressed in the current ongoing work, and are out of the scope of this paper: (a) The approach is 2-D, and additional data concerning the constant altitude at which the underwater robot is moving is needed but easily obtainable by means of other sensors, such as a stereo altimeter or the DVL; this current approach is being evolved towards a 3-D one, taking into account the 6 Degrees of Freedom (DoF) of the vehicle, starting with the application of a 3-D odometer such as the well-known Viso2 Library [53]. (b) The sooner the map joining task is performed, the best to increase the accuracy of the localization data; once both sessions are joined, the vehicle localization gets into a pure single session SLAM procedure, in which finding local loop closings is easier and they appear with more frequency than intersessions; and delaying the map joining delays inevitably the fine correction of the map.

The forthcoming work includes (a) implementing the whole system in C++, (b) setting out a strategy to get a trajectory ground truth of the underwater missions to be used in comparisons of our method with any other anchor-node-based methods, and (c) adapting the proposal presented in this paper to perform multi-robot visual SLAM.

Author Contributions: Both authors have contributed equally in all aspects of the paper.

Funding: This work is partially supported by the Spanish Ministry of Economy and Competitiveness under contract TIN2014-58662-R (AEI, FEDER, UE) and DPI2017-86372-C3-3-R (AEI, FEDER, UE).

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

References

1. Durrant-Whyte, H.; Bailey, T. Simultaneous localization and mapping (SLAM): part I The Essential Algorithms. *Robot. Autom. Mag.* **2006**, *2*, 99–110. [CrossRef]
2. Thrun, S.; Burgard, W.; Fox, D. *Probabilistic Robotics*; The MIT Press: Cambridge, MA, USA, 2005.
3. Tan, F.; Lohmiller, W.; Slotine, J.J. Analytical SLAM without linearization. *Int. J. Robot. Res.* **2017**, *36*, 1554–1578. [CrossRef]
4. Davison, A.; Calway, A.; Mayol, W. Visual SLAM. *IEEE Trans. Robot.* **2007**, *24*, 1088–1093.
5. Bonin, F.; Burguera, A.; Oliver, G. Imaging systems for advanced underwater vehicles. *J. Marit. Res.* **2011**, *8*, 65–86.
6. Burguera, A.; Bonin-Font, F.; Oliver, G. Trajectory-based visual localization in underwater surveying missions. *Sensors* **2015**, *15*, 1708–1735. [CrossRef]
7. Engel, J.; Schöps, T.; Cremers, D. *LSD-SLAM: Large-Scale Direct Monocular SLAM*; Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Springer: Cham, Switzerland, 2014; Volume 8690 LNCS, pp. 834–849.
8. Mur-Artal, R.; Montiel, J.M.; Tardos, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
9. McDonald, J.; Kaess, M.; Cadena, C.; Neira, J.; Leonard, J.J. Real-time 6-DOF Multi-session Visual SLAM over Large-Scale Environments. *Robot. Auton. Syst.* **2013**, *61*, 1144–1158. [CrossRef]
10. Labbé, M.; Michaud, F. Online global loop closure detection for large-scale multi-session graph-based SLAM. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 2661–2666.

11. Wang, Y.; Huang, S.; Xiong, R.; Wu, J. A framework for multi-session RGBD SLAM in low dynamic workspace environment. *CAAI Trans. Intell. Technol.* **2016**, *1*, 90–103. [CrossRef]
12. Negre Carrasco, P.L.; Bonin-Font, F.; Oliver-Codina, G. Global image signature for visual loop-closure detection. *Auton. Robot.* **2016**, *40*, 1403–1417. [CrossRef]
13. Glover, A.; Maddern, W.; Warren, M.; Stephanie, R.; Milford, M.; Wyeth, G. OpenFABMAP : An Open Source Toolbox for Appearance-based Loop Closure Detection. In Proceedings of the International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012; pp. 4730–4735.
14. Negre, P. Sources of HALOC. Available online: https://github.com/srv/libhaloc (accessed on 16 August 2019).
15. Burguera, A. Sources of Trajectory-Based Visual SLAM. Available online: https://github.com/aburguera/VISUAL_SLAM_2D (accessed on 16 August 2019).
16. Burguera, A. Sources of Multi-session Visual SLAM. Available online: https://github.com/aburguera/MULTISLAM_2D (accessed on 16 August 2019).
17. Seow, Y.; Miyagusuku, R.; Yamashita, A.; Asama, H. Detecting and solving the kidnapped robot problem using laser range finder and wifi signal. In Proceedings of the IEEE International Conference on Real-Time Computing and Robotics, Okinawa, Japan, 14–18 July 2017; pp. 303–308.
18. Williams, S.B.; Pizarro, O.; Foley, B. Return to Antikythera: Multi-session SLAM based AUV Mapping of a First Century B.C. Wreck Site. In *Springer Tracts in Advanced Robotics*; Springer: Cham, Switzerland, 2016; Volume 113, pp. 45–59.
19. Williams, B.; Reid, I. On combining visual SLAM and visual odometry. In Proceedings of the IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–8 May 2010; pp. 3494–3500.
20. Arandjelovic, R.; Gronat, P.; Torii, A.; Pajdla, T.; Sivic, J. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1437–1451. [CrossRef]
21. Kuemmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. g2o: A General Framework for Graph Optimization. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 9–13 May 2011; pp. 3607–3613.
22. Carlone, L.; Aragues, R.; Castellanos, J.A.; Bona, B. A fast and accurate approximation for planar pose graph optimization. *Int. J. Robot. Res.* **2014**, *33*, 965–987. [CrossRef]
23. Kim, B.; Kaess, M.; Fletcher, L.; Leonard, J.; Bachrach, A.; Roy, N.; Teller, S. Multiple Relative Pose Graphs for Robust Coopetative Mapping. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Anchorage, AK, USA, 3–8 May 2010.
24. Angeli, A.; Filliat, D.; Doncieux, S.; Meyer, J.A. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Trans. Robot.* **2008**, *24*, 1027–1037. [CrossRef]
25. Kaess, M.; Ranganathan, A.; Dellaert, F. iSAM: Incremental Smoothing and Mapping. *IEEE Trans. Robot. (TRO)* **2008**, *24*, 1365–1378. [CrossRef]
26. Lafferty, J.D.; McCallum, A.; Pereira, F.C.N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, MA, USA, 28 June–1 July 2001; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2001; pp. 282–289.
27. Labbé, M.; Michaud, F. Long-term Online Multi-session Graph-based SPLAM with Memory Management. *Auton. Robot.* **2018**, *42*, 1133–1150. [CrossRef]
28. Grisetti, G.; Stachniss, C.; Burgard, W. Non-linear Constraint Network Optimization for Efficient Map Learning. *IEEE Trans. Intell. Transp. Syst.* **2009**, *10*, 428–439. [CrossRef]
29. Latif, Y.; Cadena, C.; Neira, J. Robust Loop Closing Over Time for Pose Graph SLAM. *Int. J. Robot. Res.* **2013**, *32*, 1611–1626. [CrossRef]
30. Ozog, P.; Eustice, R.M. Real-time SLAM with Piecewise-planar Surface Model and Sparse 3D Point Clouds. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan, 3–7 November 2013; pp. 1042–1049.
31. Ozog, P.; Carlevaris-Bianco, N.; Kim, A.; Eustice, R.M. Long-term Mapping Techniques for Ship Hull Inspection and Surveillance using an Autonomous Underwater Vehicle. *J. Field Robot.* **2016**, *33*, 265–289. [CrossRef]

32. Lowe, D. Object recognition from local scale-invariant features. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–25 September 1999; pp. 1150–1157.
33. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
34. Jain, U.; Namboodiri, V.; Pandey, G. Compact Environment-Invariant Codes for Robust Visual Place Recognition. In Proceedings of the 14th Conference on Computer and Robot Vision (CRV), Edmonton, AB, Canada, 16–19 May 2017; pp. 40–47.
35. Negre, P.L.; Bonin-Font, F.; Oliver, G. Cluster-based loop closing detection for underwater slam in feature-poor regions. In Proceedings of the IEEE International Conference on Robotics and Automation, Stockholm, Sweeden, 16–21 May 2016; pp. 2589–2595.
36. Mendes, E.; Koch, P.; Lacroix, S. ICP-based pose-graph SLAM. In Proceedings of the SSRR 2016—International Symposium on Safety, Security and Rescue Robotics, Lausanne, Switzerland, 23–27 October 2016; pp. 195–200.
37. Smith, R.; Self, M.; Cheeseman, P. A Stochastic Map for Uncertain Spatial Relationships. In Proceedings of the 4th International Symposium on Robotics Research, Santa Cruz, CA, USA, 9–14 August 1987; MIT Press: Cambridge, MA, USA, 1988; pp. 467–474.
38. Lu, F.L.F.; Milios, E. Robot pose estimation in unknown environments by matching 2D range scans. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 21–23 June 1994; pp. 1–38.
39. Burguera, A.; González, Y.; Oliver, G. Probabilistic sonar scan matching for robust localization. In Proceedings of the IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 3154–3160.
40. Burguera, A.; González, Y.; Oliver, G. On the use of likelihood fields to perform sonar scan matching localization. *Auton. Robot.* **2009**, *26*, 203–222. [CrossRef]
41. Swaminathan, A.; Mao, Y.; Wu, M. Robust and secure image hashing. *IEEE Trans. Inf. Forensics Secur.* **2006**, *1*, 215–230. [CrossRef]
42. Castellanos, J.A.; Neira, J.; Tardós, J.D. Limits to the consistency of EKF-based SLAM. *IFAC Proc. Vol.* **2004**, *37*, 716–721. [CrossRef]
43. Bar-Shalom, Y.; Rong Li, X.; Kirubarajan, T. *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*; John Wiley and Sons, Inc.: Hoboken, NJ, USA, 2001.
44. Martin-Abadal, M.; Guerrero-Font, E.; Bonin-Font, F.; González-Cid, Y. Deep Semantic Segmentation in an AUV for Online Posidonia Oceanica Meadows Identification. *IEEE Access* **2018**, *6*, 60956–60967. [CrossRef]
45. Bonin-Font, F.; Massot, M.; Codina, G.O. Visual Characterization and Automatic Detection of Posidonia Oceanica for Meadows Mapping using an AUV. In Proceedings of the International Workshop on Marine Technology-Colection Instrumentation ViewPoint, Key West, FL, USA, 8–13 November 2015; number 18.
46. Bonin-Font, F.; Gomila, C.C.; Codina, G.O. Hacia la Navegación Visual de un Vehículo Autónomo Submarino en Áreas con Posidonia Oceanica. *Rev. Iberoam. AutomÁtica InformÁtica Ind.* **2017**, *15*, 24–35. [CrossRef]
47. Carreras, M.; Hernandez, J.; Vidal, E.; Palomeras, N.; Ribas, D.; Ridao, P. Sparus II AUV—A Hovering Vehicle for Seabed Inspection. *IEEE J. Ocean. Eng.* **2018**, *43*, 344–355. [CrossRef]
48. Guerrero, E.; Bonin-Font, F.; Negre, P.L.; Massot, M.; Oliver, G. USBL Integration and Assessment in a Multisensor Navigation Approach for AUVs. In Proceedings of the 20th World Congress of the International Federation of Automatic Control (IFAC World Congress), Toulouse, France, 9–14 July 2017.
49. Quigley, M.; Conley, K.; Gerkey, B.P.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An Open-Source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 12–17 May 2009.
50. García, E.; Ortiz, A.; Bonnín, F.; Company, J.P. Fast Image Mosaicing using Incremental Bags of Binary Words. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016.
51. Bonin-Font, F.; Massot, M.; Codina, G.O. Towards Visual Detection, Mapping and Quantification of Posidonia Oceanica using a Lightweight AUV. In Proceedings of the IFAC International Conference on Control Applications in Marine Systems, CAMS 2016, Trondheim, Norway, 13–16 September 2016.

52. García, E.; Ortiz, A. Hierarchical Place Recognition for Topological Mapping. *IEEE Trans. Robot.* **2017**, *33*, 1061–1074. [CrossRef]
53. Geiger, A.; Ziegler, J.; Stiller, C. StereoScan: Dense 3d Reconstruction in Real-time. In Proceedings of the IEEE Intelligent Vehicles Symposium, Baden, Germany, 5–9 June 2011.