MDPI

Article

Semantic Mapping for Autonomous Subsea Intervention

Guillem Vallicrosa * , Khadidja Himri * , Pere Ridao * and Nuno Gracias *

Underwater Robotics Research Center (CIRS), Computer Vision and Robotics Institute (VICOROB), Universitat de Girona, Parc Científic i Tecnològic de la UdG. C/Pic de Peguera 13, 17003 Girona, Spain
* Correspondence: gvallicrosa@eia.udg.edu (G.V.); khadidja.himri@udg.edu (K.H.); pere@eia.udg.edu (P.R.); ngracias@silver.udg.edu (N.G.)

Abstract: This paper presents a method to build a semantic map to assist an underwater vehicle-manipulator system in performing intervention tasks autonomously in a submerged man-made pipe structure. The method is based on the integration of feature-based simultaneous localization and mapping (SLAM) and 3D object recognition using a database of a priori known objects. The robot uses Doppler velocity log (DVL), pressure, and attitude and heading reference system (AHRS) sensors for navigation and is equipped with a laser scanner providing non-coloured 3D point clouds of the inspected structure in real time. The object recognition module recognises the pipes and objects within the scan and passes them to the SLAM, which adds them to the map if not yet observed. Otherwise, it uses them to correct the map and the robot navigation if they were already mapped. The SLAM provides a consistent map and a drift-less navigation. Moreover, it provides a global identifier for every observed object instance and its pipe connectivity. This information is fed back to the object recognition module, where it is used to estimate the object classes using Bayesian techniques over the set of those object classes which are compatible in terms of pipe connectivity. This allows fusing of all the already available object observations to improve recognition. The outcome of the process is a semantic map made of pipes connected through valves, elbows and tees conforming to the real structure. Knowing the class and the position of objects will enable high-level manipulation commands in the near future.

Keywords: 3D object recognition; point clouds; global descriptors; semantic segmentation; semantic information; Bayesian probabilities; laser scanner; underwater environment; pipeline detection; inspection, maintenance and repair; AUV

1. Introduction

State-of-the-art autonomous underwater vehicles (AUVs) are commonly used for seafloor mapping in predominantly flat environments using multiple sensors, including side-scan sonar (SSS), multibeam echosounder (MBES), forward-looking sonar (FLS) and cameras, among others. The use of unmanned underwater vehicles (UUVs) for inspection, maintenance and repair (IMR) applications is nowadays limited to the use of remotely operated vehicles (ROVs) in inspection and/or intervention tasks. Nevertheless, during the last decade, the research community has made a significant effort defining a new class of UUV, the intervention autonomous underwater vehicle (I-AUV). This class of vehicles is expected to replace intervention ROVs in IMR tasks in the future [1]. Though several autonomous manipulation tasks have already been demonstrated, often only in water tank conditions, most are just proof of concept demonstrations oriented to very particular targets. Tasks such as valve turning [2,3], connector plug/unplug [4] and object search and recovery [5] are clear examples. Nevertheless, in all these tasks, custom algorithms have usually been used to detect and track a particular manipulation goal. Often the targets have been labeled with markers to simplify the problem, or the robot was limited to performing a particular manipulation action over a particular target object. In contrast, a truly autonomous I-AUV should be able to obtain and use semantic knowledge of its

surroundings. As such, the vehicle should be capable of identifying which objects are around it, which class they belong to, and which tasks can be performed on them. For instance, if a safety valve has to be manipulated in case of an alarm, the I-AUV needs to know which valve, where it is and how it can be opened or closed. This leads to the semantic map concept—a map containing the objects position and their specific class. Semantic mapping is a key technique to endow the I-AUV with autonomous reasoning capabilities.

1.1. Objectives

This paper tackles the semantic map building problem for an I-AUV equipped with a real-time high-resolution laser scanner and working on IMR operations. It extends our prior work [6], where a point feature-based 3D object recognition method was proposed. The method used Bayesian estimation as a probabilistic framework to integrate multiple detections into a single, and more robust, object class identification. To do so, it was necessary to track objects along the sequentially grabbed scans. For this purpose, a Interdistance Joint Compatibility Branch and Bound (IJCBB) object tracking method was proposed, which was able to track the objects in the presence of navigation glitches due to sporadic failures of the Doppler velocity log (DVL) measurements. Moreover, the method exploited semantic information related to object pipe connectivity (number of pipes connected to the object) to constrain the potential set of compatible object classes used during the Bayesian estimation. Nevertheless, the IJCBB must establish at least three pairings between two scans to be able to register them. Otherwise, the tracking fails and the object detections in this scan cannot contribute to the Bayesian estimation. On the other hand, the iterative nature of the tracking algorithm reduces the drift, but is not able to cancel it. Therefore, the natural next step is to employ simultaneous localization and mapping (SLAM) techniques using the pipes and objects as features to build a drift-less consistent map of the structure. Using conventional data association algorithms, between the objects in a scan and the objects in the SLAM, it is possible to track the objects and apply the Bayesian estimation. The outcome of the process is a semantic map of pipes and objects, which provides the I-AUV with an accurate navigation as well as with the semantic knowledge of the manipulable objects around it.

1.2. Contributions

The main contributions of the present paper are the following:

- A feature-based extended Kalman filter (EKF) SLAM method is proposed which uses line and point features to represent the pipes and the objects, respectively. The method solves two problems: (1) it provides a drift-less navigation; and (2) it assigns a globally consistent identifier to every object in every scan, enabling Bayesian estimation. When conveniently combined with the object recognition results, it becomes a semantic map endowing the I-AUV with the semantic knowledge required to perform high-level commands, such as *Open Valve X*, for example.
- It provides a method for plane segmentation which partitions the point cloud according to the average maximum curvature and classifies the partitions either as planes or as a curved region. The method allows separation of the flat surfaces corresponding to the walls of the water tank, where the experiment was performed, from the pipe structure itself.
- It provides an extension to the semantic object segmentation method already proposed in [6], ensuring the correct segmentation of the valve handle, which proved problematic in the previous paper.

1.3. Structure of the Paper

The remainder of the paper is organized as follows. Section 2 describes the state of the art on underwater SLAM, object recognition and semantic mapping. Section 3 describes the object recognition pipeline from the segmentation of the scans to the Bayesian recognition. Section 4 describes the feature-based SLAM for object and pipe feature tracking. Section 5

describes the experimental setup and the results obtained. Sections 6 and 7 provide conclusions and future work on the results obtained.

2. State of the Art

2.1. Underwater SLAM

Many outdoor field robots rely on absolute measurements to bound the dead reckoning (DR) navigation drift, such as the Global Positioning System (GPS). However, in underwater robotics, those sensors are unavailable due to electromagnetic attenuation; underwater robots instead have to rely on acoustic localization methods such as long baseline (LBL) [7], short baseline (SBL) [8], ultra-short baseline (USBL) [9] or GPS intelligent buoys (GIB) [10]. Those methods require deployment of the beacons and/or a support vessel to provide the GPS positioning to be composed with the measured acoustic position. Unfortunately, those methods restrict the vehicle to a predefined zone (LBL) or decrease their precision with increasing depth of the vehicle (SBL, USBL and GIB).

A solution to overcome these issues and have a completely independent AUV is to correlate the vehicle sensor measurements with a map of the environment to reliably locate its position with Terrain-Based Navigation (TBN) techniques [11]. However, precise maps are not widely available, and so many researchers rely on SLAM methods, where the robot incrementally builds a model of the environment and simultaneously uses it to estimate its position within it.

Underwater SLAM can be categorised according to the type of sensors used to perceive the environment. On the one hand, vision-based sensors perceive the environment at high rates and high precision, but they are very sensitive to water visibility, which greatly limits their range. On the other hand, acoustic-based sensors provide low-rate and low-precision measurements regardless of visibility. Regarding acoustic SLAM, we can further classify SLAM into feature-based and featureless methods. Feature-based methods are generally used in man-made environments, where features are easier to extract [12–14], while featureless methods are primarily used in natural environments [15–21].

In contrast, underwater vision-based SLAM relies heavily on visual features extracted from the texture of the environment [22–27]. If the environment is texture-less, an alternative is to use laser-camera systems, where the laser produces the necessary texture to extract point clouds from the environment. Initial developments of this approach relied on a fixed laser scanner that, combined with the vehicle motion, produces the point clouds [28,29], but suffers from navigation drift.

A new laser scanner based on a moving mirror provides scans at a maximum rate of 6 Hz, fast enough to allow the vehicle drift during a single scan to be neglected [30]. This laser scanner has already been tested on motion planning applications in an unknown environment [31] and in a pose-based SLAM for mapping [32]. In the present work, we focus on the application of this laser scanner to semantically extract features that serve as input for the SLAM algorithm and ease the recognition of the object features on pre-trained models of the different objects.

2.2. Object Recognition

Object recognition is a domain of 3D scene exploration and understanding associated with applications such as autonomous driving and housekeeping robots. 3D object recognition has emerged thanks to pre-existing 2D methods translated into 3D and the advanced availability of different types of 3D sensors.

In the field of object recognition based on point clouds, several surveys have been carried out in which methods and ideas based on global and local descriptors have been presented [33–35]. Global recognition methods interpret the entire object as a unique vector of values, while local recognition methods focus more on a local region and are computed from salient points. Recently, deep learning has gained increasing attention. The following two publications are representative examples. In [36], Guo et al. summarized deep learning methods applied to 3D point clouds. The authors aimed to select the most

relevant applications for point cloud understanding, considering 3D shape classification, 3D object detection and tracking, and 3D point cloud segmentation. They evaluated the quality of the performance of state-of-the-art methods based on deep learning and compared the methods with different publicly available datasets. In Tian et al. [37], the authors proposed a dynamic graph convolutional broad network (DGCB-Net) for feature extraction and object recognition from point clouds, and their method was tested on several public datasets and one dataset which they collected.

However, fewer papers have focused on underwater application scenarios, with the exception of the paper by Martin et al. [38], in which a processing pipeline is presented, based on the use of a deep PointNet neural network. The proposed method was able to detect pipes and valves from 3D RGB point clouds in underwater environments using a generated dataset to train and test the network. Recent work by Pereira et al. [39] is also based on a deep learning approach, where a convolutional neural network was used for recognizing a docking structure from point clouds. Their methods were evaluated with simulated and real datasets.

Although deep learning approaches have been reported to have attained accurate results, such methods are very demanding in terms of the amount of training data to ensure proper learning generalization. In the case of man-made structures observed by sensors that provide only colourless point clouds, the collection of the required training data is a difficult and time-consuming task.

The work described in Martin et al. [38] used a similar man-made structure as the one in our work, comprising valves interconnected by pipes. Furthermore, the experiments in both papers were conducted in an underwater environment. Their work is directly related to the problem we are trying to solve, i.e., the recognition of man-made objects underwater, because it formed part of the same research project TWINBOT [40] in which both groups participated. In the following paragraphs, we provide a comparison of the two works, which highlights the trade-offs between the two approaches.

- In the present work, we have used a feature-based SLAM approach to object recognition using a 3D point cloud with no RGB information, obtained with a laser scanner. The process can be summarised as follows:
  - The segmentation of the ground was performed using the methods explained in Section 3.2. The segmentation of pipes was performed separately from the recognition of the objects;
  - Five object classes were defined in the experiments, which were segmented based on the pipe connections;
  - The knowledge database was generated from the object's CAD model using a process described in our previous article [33]. The test data was collected in the test pool of our laboratory, and included 1268 point clouds for individual objects, extracted from 245 laser scans;
  - The main recognition performance results are found in Section 5.4.
- In the work of Martin et al. [38], a deep learning approach was applied for the detection of pipes and valves. The network used, as input, 3D point clouds with RGB information obtained from stereo cameras, and the following steps were performed:
  - Ground truth data were manually created from the point cloud, and divided into three classes: pipes, valves and background;
  - Two datasets were used. The first dataset was acquired in a test tank and contained 262 point clouds. This dataset was divided into two subsets, the first containing 236 point clouds which were used to train the network and the remainder used as test samples. The second dataset was collected in the sea and included 22 point clouds that were used only as a test set;
  - 13 experiments were conducted varying the hyper-parameters in the training phase: batch size, learning rate, block-stride and number of points;

- To assess the performance of the neural network and estimate how the model is expected to perform, a 10-fold cross-validation was performed. Overall, 9 subsets of 213 point clouds were used for training, and 1 subset of 23 point clouds was used for testing. The final classification result was obtained by averaging the performance of these ten different results;
- From the results presented, it can be seen that the background class was predominant, followed by the pipe and valve classes in both pool and sea experiments.

2.3. Semantic Mapping

Semantic mapping started indoors with scene recognition [41–44] and then moved outdoors. It has been applied on various input data, such as cameras [45,46], depth cameras [47,48] or laser scanners (usually LIDARs) [49–51]. Implementations vary from surpervised to unsupervised methods, where semantic classes are a priori unknown.

Adding semantic information to underwater maps contributes to a better spatial awareness of nearby terrains and objects, enabling higher-level tasks to be performed. This is especially important for IMR tasks where robots have to be aware of the different components and how to interact with them.

In the underwater environment, it has been mainly used for semantic image segmentation [52,53], which can also be applied to exploration [54]. To the best of the authors' knowledge, semantic mapping has not yet been applied to point clouds obtained underwater with a laser scanner for IMR tasks, and thus, this paper goes beyond the state of the art.

3. Object Recognition Pipeline

As can be seen in Figure 1, the object recognition pipeline is divided into several modules. First, the floor and lateral walls/slopes of the water tank where the experiment takes place are segmented and subtracted from the scanned point-cloud. Then, pipes are detected and the resulting point cloud is used as input for the semantic object segmentation. Having extracted the planes and the pipes from the scan, objects are segmented.
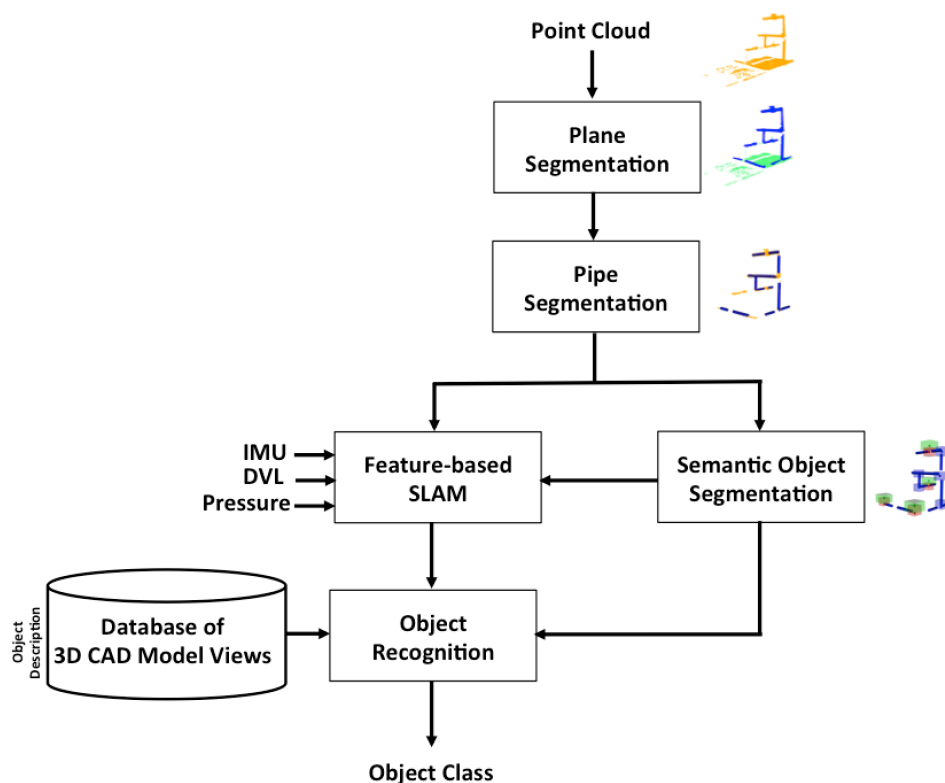

Figure 1. 3D object recognition pipeline.

A feature-based SLAM is continuously running, integrating DVL, pressure and attitude and heading reference system (AHRS) measurements. The input pipes and objects are used as features of the SLAM, which simultaneously estimates the robot pose, and the position of the already-observed pipes and objects. Therefore, solving the association of the objects segmented from the scan with those already mapped, it is possible to associate a global identifier with them. Finally, the object recognition module uses the point feature descriptors of the partial views of the segmented objects, matching them against those stored in the object database, identifying the object class. Since the global identifier of the observed object instance is known thanks to the SLAM output, it is possible to use several past object class estimations to compute its global object class, achieving more robust results. Hereafter, the different modules are described in more detail.

3.1. Object Data Base

A database of point clouds was created (Table 1), containing overlapping partial views of isolated objects. These views were created from 3D CAD models and captured using a virtual camera. This database was useful for the design of simulated experiments and for their statistical analysis, as presented in our previous work [6]. Details on the creation of the database can be found in the same publication.

Table 1. Polyvinylchloride (PVC) pressure pipe objects used in the experiments (reprinted with permission from ref. [6]. 2021 Sensors)

PVC Objects	Id Name	Size (mm^3)	PVC Objects Views (12)
	1-Ball-Valve	$198 \times 160 \times 120$	
	2- Elbow	$122.5 \times 122.5 \times 77$	
	3- R-Tee	$122.5 \times 168 \times 77$	
	4- R-Socket	$88 \times 75 \times 75$	
	5- Butterfly-Valve	$287.5 \times 243 \times 121$	
	6- 3-Way-Ball-Valve	$240 \times 160 \times 172$	

3.2. Plane Segmentation

In our previous work [6], planes were detected using random sample consensus (RANSAC). Unfortunately, in several scans, the principal plane detected did not correspond to the floor or the walls of the water tank. Sometimes, points belonging to different pipes and even objects, and others belonging to the slopes, became co-planar, forming the most significant plane in the scene. However, removing it would wrongly eliminate a significant number of points in the pipes and objects, making the recognition more challenging. To avoid this problem, an alternative method is proposed in this paper.

The problem of plane segmentation can be seen as an unsupervised classification problem, where the goal is to group the points into regions defined according to their curvature, which is an attribute describing the local geometry around a point. In Point Cloud Library (PCL), the curvature of a point is computed performing an eigen-decomposition of the points in the neighbourhood. The eigenvector corresponding to the smallest eigenvalue provides the direction of the normal, and the other two provide the tangent plane. The curvature κ is defined as the ratio between the smallest eigenvalue and the addition of the three eigenvalues:

$$\kappa = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \text{ where } \lambda_0 < \lambda_1 < \lambda_2. \tag{1}$$

To remove the planar surfaces, first we segmented the point cloud into several regions using the region-growing method [55]. The algorithm begins by selecting as a seed point the one with least curvature. Then, the region is computed by growing the seed to those adjacent points in the neighbourhood whose angles between normals (the normal of the seed and the local normal at the point) are within a pre-defined threshold. Next, the points within the region with a curvature below a threshold are considered as new seeds, and the algorithm is iterated until no more seeds are available. At this point, the first region has been segmented and the algorithm is applied again to the rest of the point cloud. The result is a set of regions having a smooth evolution of the angle among their normals. The regions are separated either for having a sudden change in their normals (smoothness), or because they are spatially separated, as shown in Figure 2. The threshold angle between the normal vectors was set to 30 degrees. If the points are on the same plane, then the normals of the fitting planes of these two points are approximately parallel.

Second, the resulting regions are classified into two categories based on an empirical threshold on their mean curvature (Fig. 2). We evaluated the curvature of each region in a neighbourhood of 50 points and chose an empirical threshold of 0.025 (Fig. 3). Each region from the growing regions result is classified as: (a) points on flat areas such as the bottom and the slopes on both sides of the water tank, or (b) points on the rest of the cloud, such as objects and pipes of the structure.

Subsequently, the flat regions are deleted, and the remainder are merged into a single region containing the non-flat areas to be further processed. The proposed plane segmentation method is shown in Algorithm 1.

Algorithm 1: Plane Segmentation

---

1 **function** RegionsGrowingSegmentation(***in:*** *scan*, ***out:*** R_I):

　　// Returns the set of region R_i detected in the scan using Growing
　　　Regions Algorithm

2 **return** $\{R_I\}$

3 **function** MergeRegions(***in:*** R_I, ***out:*** S_{R_I}, P_{R_I}):

4 　**if** $(R_{curvature} > \tau_d)$ **then** 　　　　　　　　　　　　　　　　　// Non planar?

　　　// Returns a pipes and objects (structure) regions (S_{R_I}) result
　　　　of merging the input set of non-plane regions

5 　**else**

　　　// Returns a plane regions (P_{R_I}) result of merging the input
　　　　set of plane regions

6 **return** $\{< S_{R_I}, P_{R_I} >\}$

7 **procedure** PlaneSegmentation(***in:*** *scan*; ***out:*** S_{R_I}, P_{R_I}):

8 　R_I=RegionsGrowingSegmentation(*scan*) 　　　　　　// Set of regions R_i

9 　**forall** $R_i \in R_I$ **do**

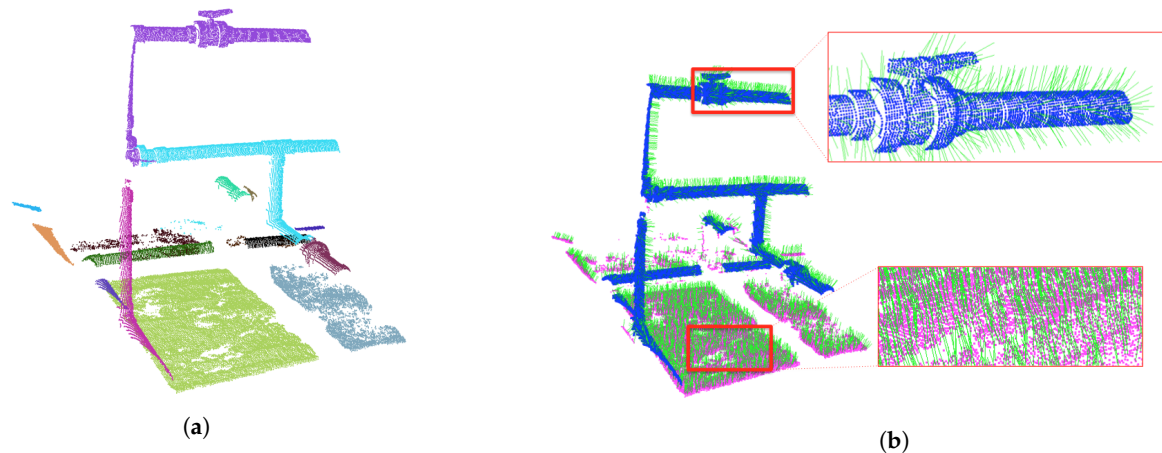10 　　$\{< S_{R_I}, P_{R_I} >\}$=MergeRegions(R_i)

Figure 2. Plane segmentation. (**a**) Outcome regions of the region-growing method. (**b**) Segmentation of the point cloud into two regions with normals in green: I) non-flat areas in blue, and II) flat areas in pink.
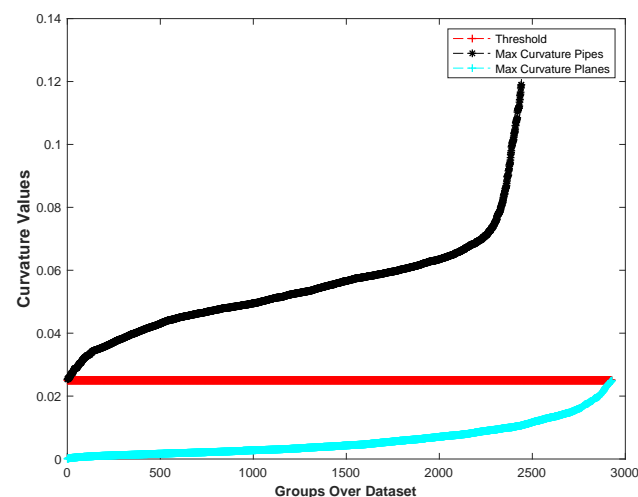

Figure 3. Mean curvature threshold separating the pipes from the flat areas. The horizontal axis represents, for all the dataset scans, the regions obtained using the region-growing method. The vertical axis provides, for each region, its mean curvature.

3.3. Pipe Detection

For detecting pipes in the current scan, a method based on the RANSAC implementation in PCL was used. This method models the pipes as cylinders with seven parameters, consisting of the 3D position of a point on the axis, axis direction, and cylinder radius. The scan is divided into two categories, namely the pipe cloud category and non-pipe cloud category. Since the radius of the pipes is known and objects have a maximum size, only the segmented cylinders with length more than 0.30 m and maximum radius of 0.064 m are considered as pipes. To calculate the endpoints of the pipes, the selected set of points is projected onto the pipe axis, and the points at the extreme ends are considered as the limits of the pipe.

Scan deformations caused by motion-induced distortions during the acquisition of the laser scan [32] can occasionally lead to two different detections being generated for the same pipe. The solution for such cases as well as details on the implementation of the pipe detection are provided in [6]. An example of pipe detection with their respective endpoints is given in Figure 4.
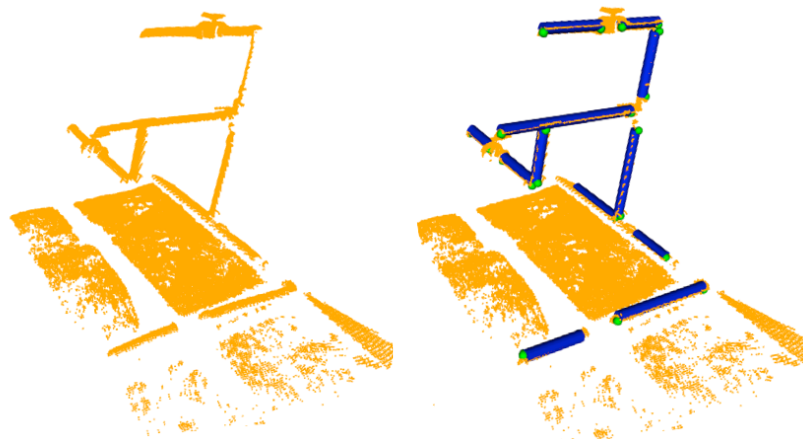
Figure 4. Pipe detection: (**left**) 3D laser scan point cloud; (**right**) Pipes in blue with their respective endpoints in green.

3.4. Semantic Object Segmentation

The proposed semantic 3D object segmentation is inspired and motivated by the fact that objects are found at the extremities of pipes. Knowledge about these objects includes detailed information about the connectivity of the objects and structural knowledge, such as the fact that valves with two parallel connections are characterised by handles, which is an important feature for objects like butterfly valves to distinguish them from their homologous valves. In addition, functional knowledge is needed for these features, allowing the robot to infer whether the valves can be turned on or off based on the position of a handle. To this end, the semantic segmentation problem can be formulated as follows: Given an object with one or two parallel connections (Figure 5), it is possible to find a potential handle, as shown in the right part of the figure, where objects with one or two parallel connections are segmented using a 'mushroom' shape (green cube on the top of the red one). The base is defined for the body of the object and the parallel pipe shape for the potential handle, while if the object has more connections or two perpendicular connections, only the base is segmented, as shown with blue cubes.
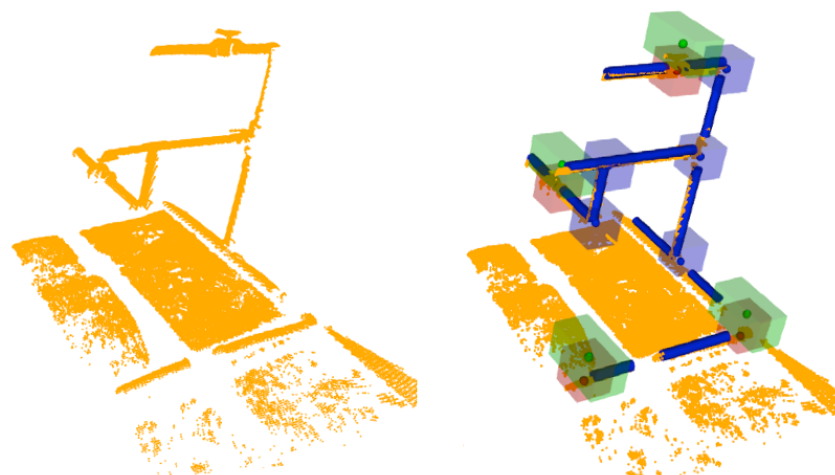

Figure 5. Semantic object segmentation: (**left**) 3D laser scan point cloud; (**right**) Example of segmentation and how objects with different connectivity are treated differently.

3.5. 3D Object Recognition Based on Global Descriptors

Object recognition is an essential part of building a semantic map of the environment. In [33] we studied and compared several descriptors using synthetic and real data. The best results involving experimental data were achieved using the Clustered Viewpoint Feature Histogram (CVFH) [56] descriptor, which is therefore used in this paper.

3.6. Bayesian Recognition

A disadvantage of object recognition with a single-view approach is that multiple objects may have similar views. A study based on the confusion matrices for the various objects was carried out in our previous work [33]. Given a set of observations of a particular object, we can use confusion matrices to determine how many observations were recognized as *object-class-n*, where *n* indicates the class name of the object. Given this information, we can estimate a probability for each class as well as the confusion between classes, which is used to implement a Bayesian estimation method to improve object recognition results.

For this purpose, several observations were combined to calculate the probability that an object belongs to each object class. The selected object was assigned to the class with the highest probability. This method required continuous observation of the same objects across the scans, so a tracking method was required to iteratively compute the Bayesian probabilities. In our previous work [6], this tracking was performed using a navigation-less variant of the Joint Compatibility Branch and Bound (JCBB) algorithm, based on the distances between objects within a scan, and referred to as IJCBB. In the present work, we use the SLAM solution described in Section 4, which achieves significantly higher performance.

3.7. Bayesian Estimation

In order to solve the common problem of ambiguous observations caused by having only partial views of the objects in the scans, a Bayesian estimator is applied. In [33] we have already computed the object confusion matrix; this matrix is used as an estimate of the required conditional probabilities. The object class recognised with the global descriptor is denoted as Z_C. X is the actual class of this object, and *Ball-Valve*, *Elbow*, *R-Tee*, *R-Socket*, *Butterfly-Valve*, *3-Way-Valve* are potential class candidates, sub-indexed with numbers 1 to 6 respectively. $P(Z_C|X_i)$ indicates the probability that the object is recognised as class Z_C when its actual class is X_i. If $C = i$ then it is a true positive (TP), otherwise ($C \neq i$) it is a false positive (FP).

3.8. Semantic-Based Recognition

By knowing the number of pipes connected to the object and their geometry, the recognition rate can be further improved. This method was presented in [6] and is briefly summarized here for completion.

The information about the number of pipe connections and their geometry is used to reduce the set of possible classes for a given object by considering only those classes that are compatible with that configuration. For example, if we know that an object is connected to 3 pipes, then only 2 candidate classes are possible: the *R-Tee* and the *3-Way-Valve*. Thus, the Bayesian probabilities are computed only for the compatible candidate classes and considered zero for the rest.

Four different geometric configurations may arise:

Configuration 1 Three pipes: two collinear and one orthogonal. This group contains the *R-Tee* and the *3-Way-Valve*;

Configuration 2 Two orthogonal pipes: This group contains the *Elbow* but also the members of the previous group, since it is possible that the third pipe has not yet been observed;

Configuration 3 Two collinear pipes: All objects are included in this group, except the *Elbow* and the R-Sockets. The remaining objects admit a collinear connection to two pipes;

Configuration 4 Single or no connection: All objects are considered as potential candidates.

It can be seen from Table 2 that these configurations have a hierarchy in the sense that the first is the most restrictive, the second is less restrictive and encompasses the objects of the first group, and so on. One exception is group 3, for 2 collinear pipes where the *Elbow* of group 2 is not present. It is worth noting that the laser scanning process often provides only partial views of the objects due to occlusions and the limited field of view. As such, a certain object may appear as connected to a single pipe in the first observation, then connected to three pipes on the second observation and then just to a single pipe in the third observation. Since objects are mapped in the SLAM, we can use the knowledge of the previously observed configurations to better compute the probabilities. As an example, if an object is observed in configuration 1 and then configuration 2, then the probabilities for the second observation will be computed as for configuration 1 (which is the most restrictive).

Table 2. Semantic connection of objects. The number of pipes connected to an object is indicated by n_p (reprinted with permission from ref. [6]. 2021 Sensors).

Type of Connection	Pipe Disposition			Potential Object Candidates
	n_p	$=$	$\perp$	
	3	2	1	
	2	0	2	
	2	2	0	
	1|0	1|0	1|0	

4. Simultaneous Localization and Mapping for Object and Pipe Tracking

Once the pipes and objects are segmented from the scans, they are sent as input to a SLAM algorithm that integrates AUV navigation with those features in order to improve navigation and track the features, keeping a single global ID for each of them. The output of the SLAM to the semantic Bayesian recognition are the global IDs for each object detected in the scan. This ensures that different observations of the same object are used together to better estimate the object class.

4.1. Line Feature Representation

The pipes are represented using an ortho-normal line representation [57] consisting of three angles of rotation ($\alpha\,\beta\,\gamma$) and the shortest distance from the frame origin to the line ρ (2) (Figure 6).

$$L = \begin{bmatrix} \alpha & \beta & \gamma & \rho \end{bmatrix} \tag{2}$$
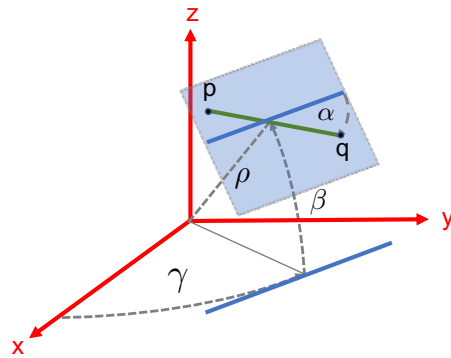
Figure 6. Line feature parametrization.

Given the segment endpoints (p and q) provided by the pipe detection algorithm (see Section 3.3), the ortho-normal representation is computed using Plücker coordinates [58].

$$n = p \times q \tag{3}$$
$$v = q - p \tag{4}$$
$$n_u = n/\|n\| \tag{5}$$
$$v_u = v/\|v\| \tag{6}$$
$$r_u = v_u \times n_u \tag{7}$$
$$R = \begin{bmatrix} r_u & v_u & n_u \end{bmatrix} = Rot(\gamma, z)Rot(\beta, y)Rot(\alpha, x) \tag{8}$$
$$\rho = \|n\|/\|v\| \tag{9}$$

where v_u represents the line direction and n_u is perpendicular to the plane formed by the two endpoints and the frame origin (Figure 7). The three angles of rotation can be extracted from the rotation matrix R as:

$$\alpha = \text{atan2}(v_{u_z}, n_{u_z}) \tag{10}$$
$$\beta = \text{asin}(r_{u_z}) \tag{11}$$
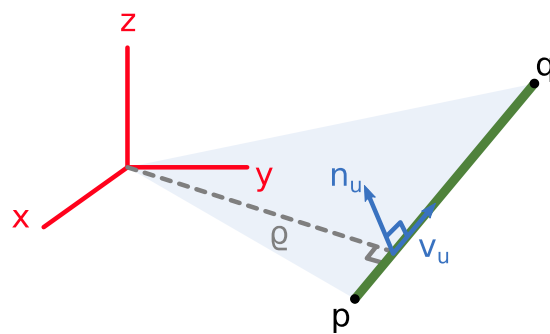$$\gamma = \text{atan2}(r_{u_y}, r_{u_x}) \tag{12}$$

Figure 7. Ortho-normal representation of a pipe segment.

The line is computed from the pipe endpoints which are known in the vehicle sensor frame $\{S\}$, which is the frame of reference of the point cloud. Therefore it is initially referenced to $\{S\}$ and has to be transformed to the world frame $\{W\}$:

$$^W\rho \cdot {}^W\boldsymbol{n}_u = {}^W\boldsymbol{R}_S \cdot {}^S\rho \cdot {}^S\boldsymbol{n}_u + {}^W\boldsymbol{t}_S \times \left({}^W\boldsymbol{R}_S \cdot {}^S\boldsymbol{v}_u\right) \tag{13}$$

$$^W\boldsymbol{v}_u = {}^W\boldsymbol{R}_S \cdot {}^S\boldsymbol{v}_u \tag{14}$$

where $^W\boldsymbol{R}_S$ and $^W\boldsymbol{t}_S$ are, respectively, the rotation and the translation that transform from the sensor frame $\{S\}$ to the world frame $\{W\}$.

Similarly, the opposite transformation is computed as:

$$^S\rho \cdot {}^S\boldsymbol{n}_u = {}^W\boldsymbol{R}_S^T \cdot {}^W\rho \cdot {}^W\boldsymbol{n}_u - {}^W\boldsymbol{R}_S^T \cdot {}^W\boldsymbol{t}_S \times {}^W\boldsymbol{v}_u \tag{15}$$

$$^S\boldsymbol{v}_u = {}^W\boldsymbol{R}_S^T \cdot {}^W\boldsymbol{v}_u \tag{16}$$

From this, we can calculate the frame change Jacobians with respect to the line representation in the frame and to the sensor position in the world.

4.2. State Vector

The state is represented with the Gaussian random vector $\mathbf{x}(k)$:

$$\mathbf{x}(k) = \begin{bmatrix} \mathbf{x}_v(k) & \mathbf{f}_1(k) & \mathbf{f}_2(k) & \dots & \mathbf{f}_n(k) \end{bmatrix}^T \tag{17}$$

defined by 2 parameters, the mean:

$$\widehat{\mathbf{x}}(k) = \begin{bmatrix} \widehat{\mathbf{x}}_v(k) & \widehat{\mathbf{f}}_1(k) & \widehat{\mathbf{f}}_2(k) & \dots & \widehat{\mathbf{f}}_n(k) \end{bmatrix}^T \tag{18}$$

and the covariance matrix $\boldsymbol{P}(k)$, which provides the covariance of the vehicle and the feature lines, as well as their cross-correlations:

$$\boldsymbol{P}(k) = E\left([\mathbf{x}(k) - \widehat{\mathbf{x}}(k)][\mathbf{x}(k) - \widehat{\mathbf{x}}(k)]^T\right) = \begin{bmatrix} \boldsymbol{P}_v(k) & \boldsymbol{P}_{vf_1}(k) & \dots & \boldsymbol{P}_{vf_n}(k) \\ \boldsymbol{P}_{f_1v}(k) & \boldsymbol{P}_{f_1}(k) & \dots & \boldsymbol{P}_{f_1f_n}(k) \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{P}_{f_nv}(k) & \boldsymbol{P}_{f_nf_1}(k) & \dots & \boldsymbol{P}_{f_n}(k) \end{bmatrix} \tag{19}$$

The vehicle state $\mathbf{x}_v = [x\ y\ z\ \phi\ \theta\ \psi\ u\ v\ w]^T$ has nine dimensions, including vehicle position $[x\ y\ z]^T$ and the vehicle orientation $[\phi\ \theta\ \psi]$, both represented in the world reference frame $\{W\}$. This frame is located at the water surface, being aligned with the north (i.e., north-east-down (NED) reference frame). The linear velocities $[u\ v\ w]^T$, instead, are referenced to the vehicle's frame $\{B\}$. This is also the minimum dimension of the state vector at the beginning of the execution. The state vector is initialized with the vehicle at rest on the surface when the first depth, AHRS and DVL measurements are received.

The line and object features $[\mathbf{f}_1(k)\ \mathbf{f}_2(k)\ \dots\ \mathbf{f}_n(k)]$ are static and defined in the world reference frame $\{W\}$. Line features are represented with ortho-normal coordinates (see Section 4.1) and objects are represented by their coordinates xyz. The number of line features in the state vector is represented by n_l and the number of object features is represented by n_o, with the total number of features being $n = n_l + n_o$.

4.3. Prediction

A six degrees of freedom (DoF) constant-velocity kinematics model is used to predict the vehicle state evolution from time $k-1$ to time k. The attitude rate of change (Euler angle derivatives), available from the AHRS, is used as the system input ($\mathbf{u}(k) = [\dot{\phi}\ \dot{\theta}\ \dot{\psi}]^T$). The uncertainty is modeled as a white Gaussian noise in linear acceleration (w_l) and attitude velocity (w_a). This model can be formulated as:

$$\mathbf{x}_v(k|k-1) = f(\mathbf{x}_v(k-1), \mathbf{u}(k), \mathbf{w}(k)) \tag{20}$$

$$\mathbf{x}_v(k|k-1) = \begin{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \boldsymbol{Rot}(\phi,\theta,\psi) \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} \Delta t + \mathbf{w}_l \frac{\Delta t^2}{2} \right) \\ \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} + (\mathbf{u} + \mathbf{w}_a)\Delta t \\ \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \mathbf{w}_l \Delta t \end{bmatrix} \tag{21}$$

where Δt is the time between $k-1$ and k, and $\mathbf{w} = [\mathbf{w}_l\ \mathbf{w}_a] \sim \mathcal{N}(\mathbf{0}, Q)$ is a white Gaussian noise representing the uncertainty of the linear acceleration $\mathbf{w}_l = [\mathbf{w}_{\dot{u}}\ \mathbf{w}_{\dot{v}}\ \mathbf{w}_{\dot{w}}]$ and the attitude velocity $\mathbf{w}_a = \left[\mathbf{w}_{\dot{\phi}}\ \mathbf{w}_{\dot{\theta}}\ \mathbf{w}_{\dot{\psi}}\right]$. In contrast, the features are static and are kept constant throughout the prediction. Hence, the whole state can be predicted using:

$$\mathbf{x}(k|k-1) = \begin{bmatrix} f(\mathbf{x}_v(k-1), \mathbf{u}(k), \mathbf{w}(k)) & \mathbf{f}_1(k-1) & \mathbf{f}_2(k-1) & \ldots & \mathbf{f}_n(k-1) \end{bmatrix}^T \tag{22}$$

4.4. Navigation Sensor Updates

The different navigation sensors present on the vehicle (pressure sensor, DVL and AHRS) provide direct observations of the state vector. Therefore, a linear observation model can be used. The general model in this case is:

$$\mathbf{z}(k) = \boldsymbol{H}(k) \cdot \mathbf{x}(k|k-1) + \boldsymbol{m}(k) \tag{23}$$

where z is the measurement vector, and $\boldsymbol{m} \equiv \mathcal{N}(\mathbf{0}, \boldsymbol{R})$ is a white Gaussian noise vector with $\mathbf{0}$ mean and covariance $\boldsymbol{R}$. The size of the observation matrix $\boldsymbol{H}$, as well as the size of $\boldsymbol{R}$, changes between the different types of observations.

A pressure sensor produces a 1 DoF position measurement which is a direct observation of the vehicle's depth (i.e., z position). Therefore, the resulting observation matrix is:

$$\boldsymbol{H}_{\textbf{DEPTH}}(k) = \begin{bmatrix} 0 & 0 & 1 & \mathbf{0}_{1\times 6} & \mathbf{0}_{1\times(4n_l+3n_o)} \end{bmatrix} \tag{24}$$

and R_{DEPTH} is the covariance of the pressure sensor:

$$R_{\text{DEPTH}} = \sigma_{\text{DEPTH}}^2 \tag{25}$$

An AHRS produces 3 DoF angular measurements, which are direct observations of the vehicle attitude (Euler angles). The resulting observation matrix is:

$$\boldsymbol{H}_{\textbf{AHRS}}(k) = \begin{bmatrix} \mathbf{0}_{3\times 3} & I_{3\times 3} & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times(4n_l+3n_o)} \end{bmatrix} \tag{26}$$

and the covariance matrix $\boldsymbol{R}_{\textbf{AHRS}}$ is a 3×3 square matrix with the uncertainties of each angle observation:

$$\boldsymbol{R}_{\textbf{AHRS}}(k) = \begin{bmatrix} \sigma_\phi^2 & 0 & 0 \\ 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & \sigma_\psi^2 \end{bmatrix} \tag{27}$$

A DVL produces 3 DoF velocity measurements, which are direct observations of the vehicle velocity in its own frame:

$$\boldsymbol{H}_{\textbf{DVL}}(k) = \begin{bmatrix} \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & I_{3\times 3} & \mathbf{0}_{3\times(4n_l+3n_o)} \end{bmatrix} \tag{28}$$

and the covariance matrix R_{DVL} is a 3×3 square matrix with the uncertainties of each velocity estimation.

$$R_{\mathbf{DVL}}(k) = \begin{bmatrix} \sigma_u^2 & 0 & 0 \\ 0 & \sigma_v^2 & 0 \\ 0 & 0 & \sigma_w^2 \end{bmatrix} \tag{29}$$

4.5. Line Feature Observation

From the pipe detector (see Section 3.3), line features are received as pairs of endpoints in the sensor frame $\{S\}$. A first merging filter is used to join collinear segments onto bigger segments. This is done by checking the point-to-line distance of the endpoints against the line defined by the other segment and vice-versa. If all the distances are below a threshold, the segments are joined and the longest possible segment from the two pairs of endpoints is retained (Figure 8).
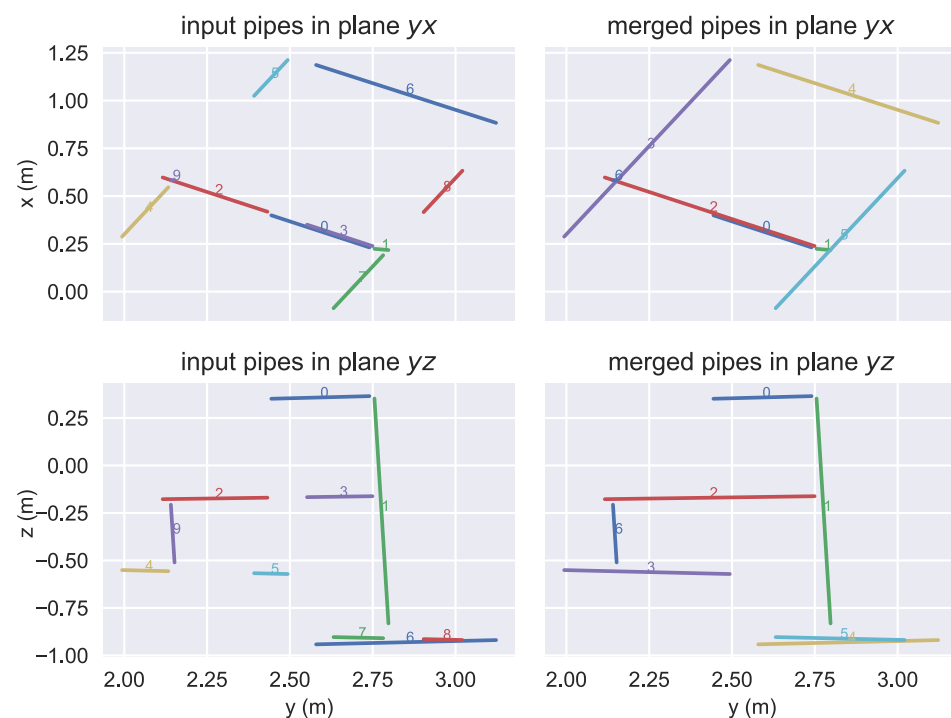

Figure 8. (**left**) Original pipes received from the pipe detector. (**right**) Merged pipes before SLAM update.

As observations of a highly angular structure, the angular threshold between lines is not very sensitive, and in this case, a maximum value of 0.175 rad is used. However, the distance threshold is more sensitive due to the existence of parallel lines. A maximum value around the half distance between the closest lines in the real structure, 0.3 m, is used.

The merged segments are converted to the line feature representation in the sensor frame using Equations (3)–(12). The first step in the feature update process is feature association. Already mapped features in the state vector are transformed to the sensor frame together with their uncertainty. A JCBB algorithm is used to ensure consistency in the associations, as opposed to standard individual compatibility [59]. Once this association is solved, we have two kinds of observations: re-observed features that were already in the state vector, or new features that are candidates to be added to the state vector.

For better representation of the line features when observing the results, the endpoints provided by the pipe detector are saved and re-projected to their associated line at the end of every feature observation.

4.5.1. Line Feature Re-Observation

Given a feature observation $z(k)$, associated with an already mapped feature f_j, the non-linear observation equation is defined as:

$$z(k) = h_{f_j}(x(k), v_j(k)) = h_j(x_v(k), f_j(k), v_j(k)) \tag{30}$$

$$v_j(k) \equiv \mathcal{N}(\mathbf{0}, R_{f_j}(k)) \tag{31}$$

where the h_j function uses the the robot pose x_v and the feature parameters $f_j = [^W\alpha\ ^W\beta\ ^W\gamma\ ^W\rho]$ are represented in the world frame to transform the line parameters to be referenced to the sensor frame. To do so, first, (3)–(7) are used to compute the vectors ^Wr_u, ^Wv_u, ^Wn_u and $^W\rho$. Next, Equations (15)–(16) are used to compute their counterparts in the sensor frame and, finally, (10)–(12) compute the angles of the new line parametrization in the sensor frame.

The linearised observation matrix is given by:

$$H_{f_j}(k) = \left. \frac{\partial h_{f_j}(x(k), v(k))}{\partial x(k)} \right|_{x(k) = \hat{x}(k)} \tag{32}$$

$$\mathbf{H_{f_j}}(k) = \begin{bmatrix} J_{1_j}(k) & \mathbf{0} & \cdots & \mathbf{0} & J_{2_j}(k) & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}_{4 \times (9 + 4n_l + 3n_o)} \tag{33}$$

where J_{1_j} is a 4×9 Jacobian matrix that represents the partial derivative of transforming f_j from the world frame $\{W\}$ to the sensor frame $\{S\}$ with respect to the vehicle state, and J_{2_j} is a 4×4 Jacobian matrix that represents the partial derivative of transforming f_j from the world frame $\{W\}$ to the sensor frame $\{S\}$ with respect to the features in the world frame ^Wf_j:

$$J_{1_j}(k) = \left. \frac{\partial h_j(x_v(k), f_j(k))}{\partial x_v(k)} \right|_{x_v = \hat{x}_v(k), f_j(k) = \hat{f}_j(k)} \tag{34}$$

$$J_{2_j}(k) = \left. \frac{\partial h_j(x_v(k), f_j(k))}{\partial f_j(k)} \right|_{x_v(k) = \hat{x}_v(k), f_j(k) = \hat{f}_j(k)}. \tag{35}$$

Next, observation matrices are stacked to form a single observation matrix:

$$H(k) = \begin{bmatrix} H_{f_1}(k) \\ H_{f_2}(k) \\ \dots \\ H_{f_s}(k) \end{bmatrix}_{4s \times (9 + 4n_l + 3n_o)} \tag{36}$$

with s being the number of observed features. Similarly, the covariance matrices R_i are used to form a block diagonal matrix of uncertainty:

$$R(k) = \begin{bmatrix} R_{f_1}(k) & \mathbf{0}_{4 \times 4} & \cdots & \cdots \\ \mathbf{0}_{4 \times 4} & R_{f_2}(k) & \cdots & \cdots \\ \vdots & \vdots & \ddots & \mathbf{0}_{4 \times 4} \\ \vdots & \vdots & \mathbf{0}_{4 \times 4} & R_{f_s}(k) \end{bmatrix}_{4s \times 4s}. \tag{37}$$

Then, a standard EKF update is applied using these matrices.

4.5.2. New Line Feature Observation

After updating the filter with all the feature observations which have been associated to map features, the remaining non-associated features are considered as candidates to be incorporated to the state vector. Since the structure is known to have only vertical or horizontal pipes, the candidate features are tested against this condition in order to discard outliers.

To add a feature $\mathbf{f}_i$ observed in the sensor frame $\{S\}$ to the state vector, it is compounded with the current vehicle position to obtain the feature in the world frame $\{W\}$. We denote this operation with the $\odot$ operator to distinguish it from the vehicle-point compounding using the $\oplus$ operator, traditionally defined in the SLAM literature as:

$$\mathbf{f}_j(k) = \mathbf{x}_v(k) \odot \mathbf{f}_i(k) \tag{38}$$

Let the stochastic map at time step k be defined by the stochastic vector $x(k) \sim \mathcal{N}(\hat{x}(k), P(k))$. Then, the augmented state vector, including the new feature, is given by:

$$x_+(k) \equiv \mathcal{N}(\hat{x}_+(k), P_+(k)) \tag{39}$$

where:

$$\hat{\mathbf{x}}_+(k) = \begin{bmatrix} \hat{\mathbf{x}}(k) & \hat{\mathbf{x}}_v(k) \odot \hat{\mathbf{f}}_i(k) \end{bmatrix}^T \tag{40}$$

and:

$$P_+(k) = \begin{bmatrix} P(k) & [P_v^T(k)\, P_{f_1 v}^T(k) \ldots P_{f_m v}^T(k)]^T J_{1\odot}(k)^T \\ [P_v(k)\, P_{vf_1}(k) \ldots P_{vf_m}(k)] J_{1\odot}(k) & J_{1\odot}(k) P_v(k) J_{1\odot}^T(k) + J_{2\odot}(k) R_{f_j}(k) J_{2\odot}^T(k) \end{bmatrix} \tag{41}$$

where $J_{1\odot}$ is a 4×9 Jacobian matrix that represents the partial derivative of transforming $\mathbf{f}_i$ from the sensor frame $\{S\}$ to the world frame $\{W\}$ with respect to the vehicle state, and $J_{2\odot}$ is a 4×4 Jacobian matrix that represents the partial derivative of transforming $\mathbf{f}_i$ from the sensor frame $\{S\}$ to the world frame $\{W\}$ with respect to the feature in the sensor frame:

$$J_{1\odot}(k) = \left. \frac{\partial \mathbf{x}_v(k) \odot \mathbf{f}_i(k)}{\partial \mathbf{x}_v(k)} \right|_{x_v = \hat{x}_v(k), f_i(k) = \hat{f}_i(k)} \tag{42}$$

$$J_{2\odot}(k) = \left. \frac{\partial \mathbf{x}_v(k) \odot \mathbf{f}_i(k)}{\partial \mathbf{f}_i(k)} \right|_{x_v = \hat{x}_v(k), f_i(k) = \hat{f}_i(k)} \tag{43}$$

Once a feature is added to the state vector, its endpoints are also saved for future re-observations.

4.6. Object Feature Observation

From the object semantic segmentation, object features are received as *xyz* positions in the sensor frame $\{S\}$. The first step before the update is the feature association. Already-mapped features in the state vector are transformed to the sensor frame together with their uncertainty. As for the line features, a JCBB algorithm is used to ensure consistency in the associations. Once this association is solved, we have two kinds of observations: re-observed features that were already in the state vector or new features that are candidates to be added to the state vector.

4.6.1. Object Feature Re-observation

As in the previous case, each feature observation $\mathbf{z}(k)$ associated with an already mapped feature $\mathbf{f}_j$ has an observation Equation (30). In this case, since we use point features instead of lines, a different h_j function is used:

$$h_j(x_v(k), f_j(k)) = \ominus x_v(k) \oplus f_j(k). \tag{44}$$

where $\oplus$ and $\ominus$ are the conventional compounding and inverse compounding operations commonly used in the SLAM literature.

Given the point feature observation (32), computing the observation matrix H_{f_j} (33) involves computing the Jacobians J_{1_j} (34) and J_{2_j} (35) of the point feature observation function h_j given in (44). In this case, the matrix size is $3 \times (9 + 4n_l + 3n_o)$ since the points are tri-dimensional. In a similar way as was used in Section 4.5.1, the stacked observation matrix $H(k)$ can be computed as shown in (36), though, in this case, its dimension is

$3s \times (9 + 4n_l + 3n_o)$. Finally, the covariance matrix of the observation can be built as a bloc diagonal matrix as shown in (37) being, in this case, a $3s \times 3s$ matrix.

Then, a standard EKF update is applied using these matrices.

4.6.2. New Object Feature Observation

As for the line features, after updating all the object position observations which have been associated to point map features, the remaining non-associated features are considered as candidates to be incorporated to the state vector. The process followed to map the newly discovered objects is equivalent to the one conducted with the pipe lines. The main difference is how the world reference feature position is computed:

$$\mathbf{f}_j(k) = \mathbf{x}_v(k) \oplus \mathbf{f}_i(k) \tag{45}$$

which, in this case, uses the conventional vector compounding operation. Therefore, the vector augmentation equations are equivalent to (40) and (41), substituting $\odot$ by $\oplus$, $J_{1\odot}$ by $J_{1\oplus}$ and $J_{2\odot}$ by $J_{2\oplus}$. Please note that in this case, the h_j function used to compute the Jacobians is now the one reported in (44).

5. Experimental Results

5.1. Experimental Setup

The underwater test scene consisted of an industrial structure comprising pipes and valves, with an approximate size of 1.4 m width, 1.4 m depth and 1.2 m height (Figure 9). For the testing, this structure was positioned at the bottom of a 5 m deep water tank, while the Girona500 AUV [60] moved in a trajectory around it while always facing the underwater structure. The laser scanner measurements were obtained at a distance ranging from 2 to 3.5 m from the underwater structure at a rate of 0.5 Hz. Maintaining a constant distance to the observed structure ensures better results as observed in [61]. The dataset was acquired and stored in a Robot Operating System (ROS) bagfile to be processed offline, consisting of the AUV navigation data (DVL at 5 Hz, pressure at 8 Hz and AHRS at 20 Hz), and the point clouds of 245 laser scans gathered with our laser scanner.
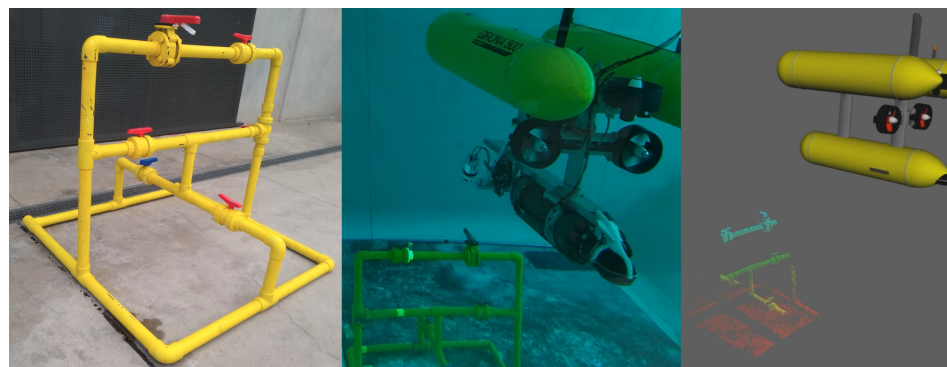
Figure 9. Experimental setup in the water tank with the Girona500 AUV. (**left**) Industrial structure before deployment. (**center**) Underwater view of the water tank during the experiments. (**right**) The 3D visualizer with a scan of the structure.

The 245 scans were processed, containing a total of 1268 object observations of 20 unique objects from 6 different classes, and 1778 pipe observations of 12 unique pipes. More details on the experimental setup can be found in [6].

A video showcasing the segmentation and SLAM results can be found in https://www.youtube.com/watch?v=flFoUrDN-rc

5.2. SLAM Results

The proposed SLAM algorithm with line and object features was compared first with the same algorithm without the feature updates, consisting of a DR navigation. Since no features are used in DR, the resulting map contains all the observations received from the semantic segmentation module (Figure 10a). Nevertheless, the SLAM solution provides a consistent map with all the pipes and objects from the structure (Figure 10b). Note that the lower corner is never observed in this dataset, and thus, the corner object, as well as the full length of the bottom pipes, are not included in the final map.

To assert the convergence on the state estimation for pipes and objects, one can look at the volume of the uncertainty bounding ellipsoid, which can be computed as $\prod_i \lambda_i$, where λ_i are the eigen-values of the uncertainty matrix corresponding to the feature. For better numerical stability, by avoiding multiplications of small numbers that can lead to numerical errors, volumes can be calculated in the logarithmic space as $\sum_i \log(\lambda_i)$. Figures 11 and 12 show how the uncertainty-bounding ellipsoids for each feature decrease through time with each re-observation of the feature and maintain constant values when the features are not re-observed.
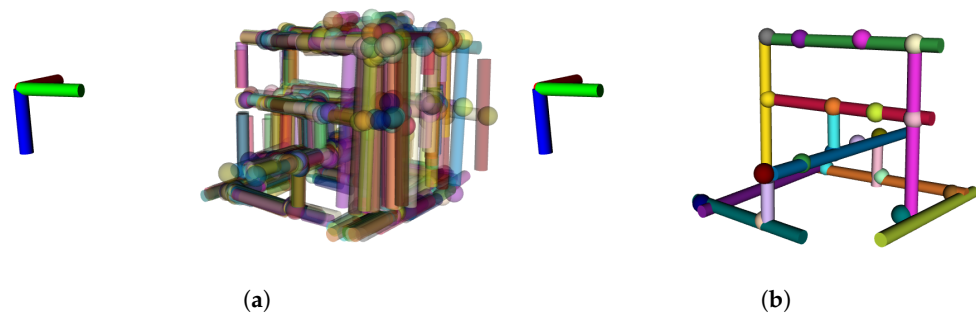

(**a**) (**b**)

Figure 10. Comparison of maps obtained from DR and SLAM with the NED reference frame. (**a**) DR with 1778 pipes and 1268 objects. (**b**) SLAM with 12 pipes and 20 objects.
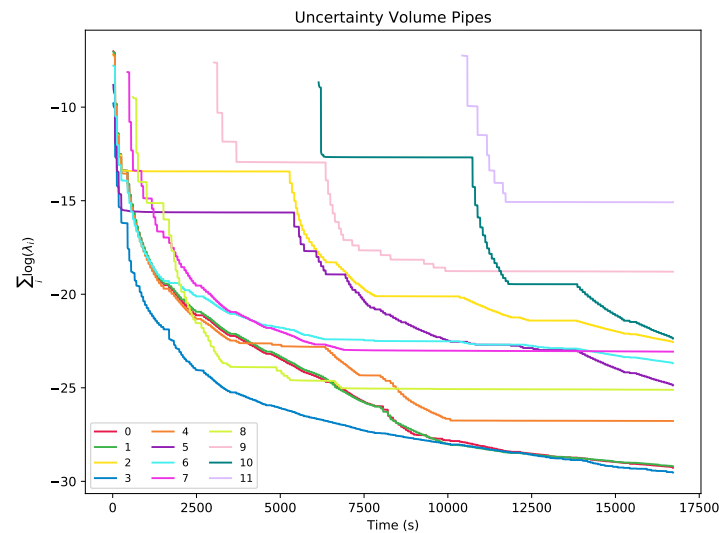

Figure 11. Uncertainty volumes with regards to experiment time for the 12 pipe features.
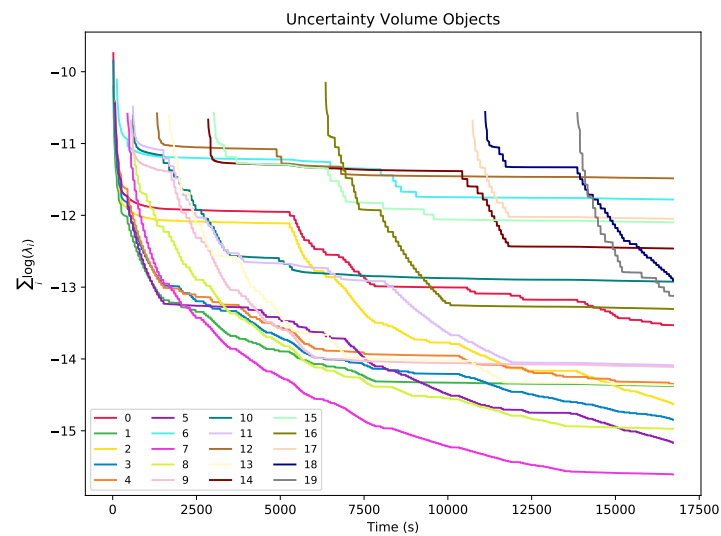
Figure 12. Uncertainty volumes with regards to experiment time for the 20 object features.

Looking at the vehicle state vector, we can observe that vehicle positions in the *xy* plane reach a significantly smaller uncertainty than the DR solution (Figure 13).
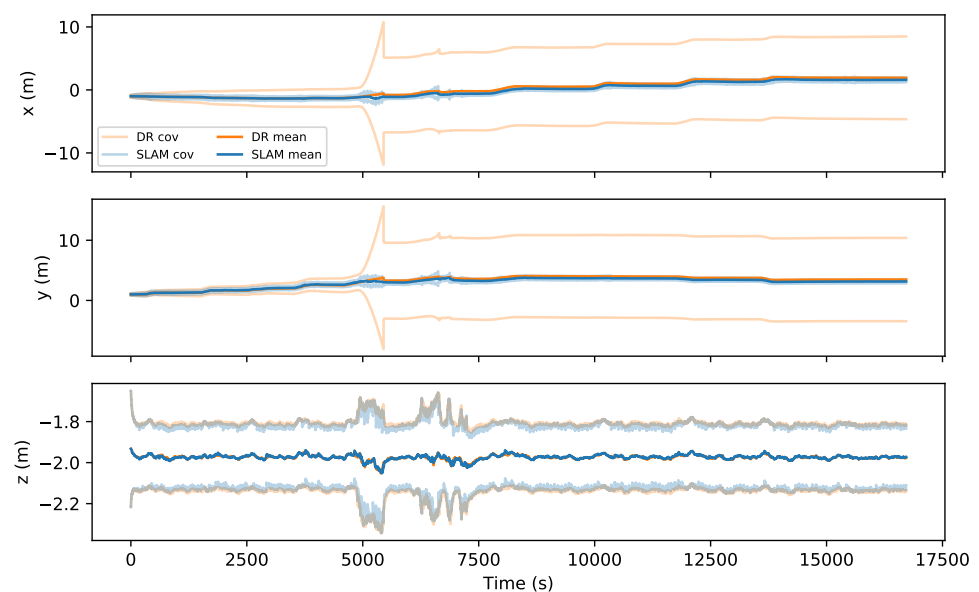
Figure 13. Comparison between DR and SLAM mean values and $\pm 2\sigma$ covariances for robot position.

It is worth noting that the DR covariance shows several peaks of uncertainty due to the DVL not being able to provide velocity measurements to bound the error. This can be clearly seen in the vehicle state velocity (Figure 14), where the DVL failures are more clearly seen by the growing uncertainty. DVL failures are common in water tank experiments due to the beams impacting the vertical walls. However, the SLAM solution greatly reduces the uncertainty during those events, providing a more accurate estimation.

Figure 14. Comparison between DR and SLAM mean values and $\pm 2\sigma$ covariances for robot velocity.

5.3. Object Segmentation Results

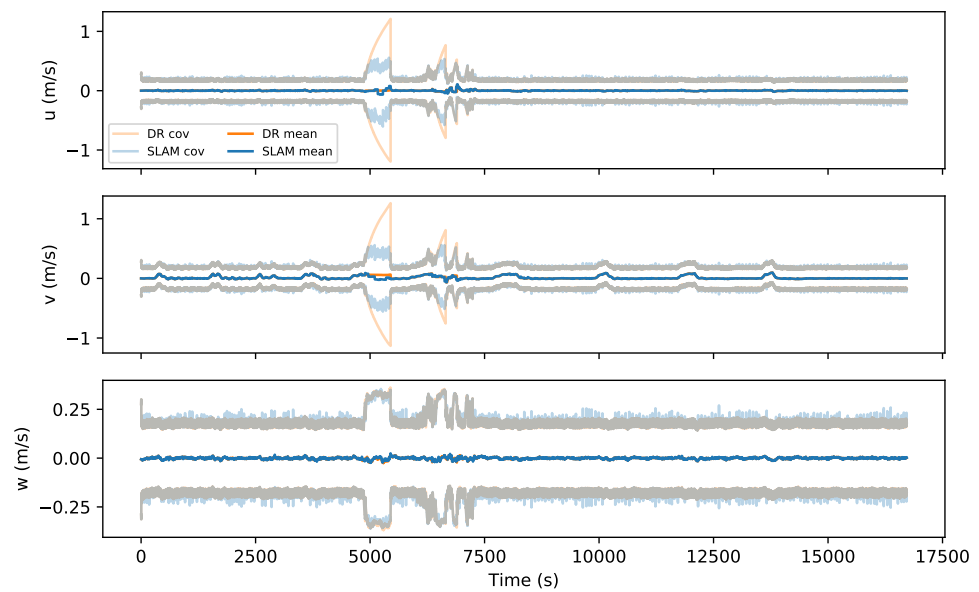Significantly better results have been achieved using the segmentation method described in Section 3.4 compared to our previous solution. The new method correctly segments the handle of the valve, which is a salient feature of this object. This can be appreciated in Figure 15, which is a good example of a *Butterfly* object segmentation. This improvement leads to a better recognition rate with the CVFH descriptor.
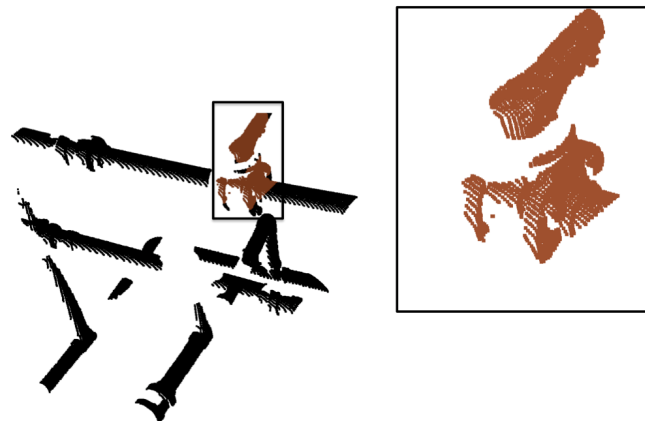
Figure 15. Segmented view of the butterfly valve along with the handle.

5.4. Object Recognition Results

Figure 16 and Table 3 show the confusion matrices of the object recognition method. The row labelled *SYN* shows the confusion matrix, which was computed using synthetic data and the CVFH descriptor only. This confusion matrix is the same as the one presented in [33] and is included here for comparison. The rows labeled *DESC*, *BAYS* and *SEM* show the experimental results of applying the method described in this paper to the dataset reported above. These rows show the confusion matrices when using the CVFH alone, together with the Bayesian estimate, and the result of incorporating the semantic information about the pipe connectivity. The figure shows that, in general, for each row

(*SYN*, *DESC*, *BAYS* and *SEM*), the column related to the ground truth class is always the one with the highest recognition rate. It also shows that, in general, the recognition rate grows when incorporating Bayesian estimation and semantic information. Please note that we separate the results (*DESC*, *BAYS* and *SEM*) to provide an insight into how the method works. Nevertheless, the row *SEM*, which corresponds to the output of the complete recognition pipeline, incorporates both Bayesian estimation and semantic information. Therefore, focusing on this row, it can be clearly seen that a good recognition rate is obtained for all objects, with *R-tee* being the most challenging one, since it is often confused with the *3-way-valve*.

On the other hand, Table 4 shows the assessment of the results based on the accuracy, precision, recall and F1 score [62]. Three object classes, namely *Ball-Valve*, *Elbow* and *Butterfly-Valve*, have a balanced trend between recall and precision, resulting in a high F1 score that improves progressively from the descriptor-based to the Bayesian, and then to the semantics-based method.

The *3-Way-Valve* has a high recall, meaning that the system works well recognising it when actually scanning (TP) and that there is a low number of False Negatives (FNs). However, it has a low precision, meaning that the number of FPs is high. Unfortunately, this leads to a poorer F1 score. The high number of FPs (*R-Tees* wrongly detected as *3-way-valves*) may be explained by the fact that most of the *R-Tees* are located at the bottom, on the floor. This means that these objects are far from the laser scanner, and therefore, their point cloud is noisier and of lower resolution (i.e., the point density is considerably lower). The *R-Tees* are particularly sensitive to noise. As can be seen in the database, the object views have smooth continuous curvatures compared to the scanned ones, which produce noisy surfaces. These noisy surfaces distort the results of the descriptor, given that the descriptor is based on the computation of surface normals from the point cloud.

In contrast, the *R-tee* class achieves high precision (low number of FPs) but low recall (high number of FNs) due, again, to the high number of *R-Tees* detected as *3-way-valves*.

Table 3. Confusion matrices expressed as a numerical %. The objects 1, 2, 3, 4, 5 represent, respectively: a *Ball-Valve*, an *Elbow*, a *R-Tee*, a *3-Way-Valve*, and a *Butterfly-Valve*. We highlighted the best recognition for each object, which coincides with the correct class and the usage of semantic information.

Descriptors	Experiment	Ball Valve					Elbow					R-Tee					3-Way-Ball-Valve					Butterfly-Valve				
		1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
CVFH	SYN	63	10	7	19	2	2	75	7	1	1	4	27	65	1	1	9	3	1	84	1	17	5	1	21	54
	DESC	60.46	18.11	1.28	9.44	10.71	2.74	80.82	4.11	9.59	2.74	1.68	38.13	19.18	28,78	12.23	39.64	5.41	0.9	49.55	4.5	21.76	12.35	7.06	25.29	33.53
	BAYS	82.65	0	0	0	17.35	1.37	90.41	1.37	6.85	0	6.47	10.55	51.32	19.42	12.23	1.8	0	0	95.5	2.7	0	0	0	0	100
	SEM	82.65	0	0	0	17.35	0	73.97	17.81	8.22	0	0	3,12	64.75	32,13	0	0	0	0	100	0	0	0	0	0	100

Table 4. Assessment of the recognition performance through accuracy, recall, precision and F1 score. We highlighted the best results in blue color, which are all consistent with the use of semantic information, except for *3-Way-Ball-Valve*, where the best F1 score was achieved using the Bayesian estimation method.

| Descriptors	Experiment	Average				Ball Valve				Elbow				R-Tee				3-Way-Ball-Valve				Butterfly-Valve			
		Accuracy	Recall	Precision	F1-Score	Accuracy	Recall	Precision	F1-Score	Accuracy	Recall	Precision	F1-Score	Accuracy	Recall	Precision	F1-Score	Accuracy	Recall	Precision	F1-Score	Accuracy	Recall	Precision	F1-Score
CVFH	DESC	0.42	0.49	0.46	0.38	0.4	0.60	0.72	0.66	0.42	0.81	0.19	0.30	0.42	0.19	0.79	0.31	0.42	0.50	0.21	0.29	0.42	0.34	0.36	0.35
	BAYS	0.76	0.84	0.73	0.74	0.76	0.83	0.92	0.87	0.76	0.90	0.60	0.72	0.76	0.51	1	0.68	0.76	0.95	0.55	0.70	0.76	1	0.58	0.74
	SEM	0.80	0.84	0.78	0.78	0.8	0.83	1	0.91	0.80	0.74	0.81	0.77	0.80	0.65	0.95	0.77	0.80	1	0.44	0.61	0.80	1	0.71	0.83
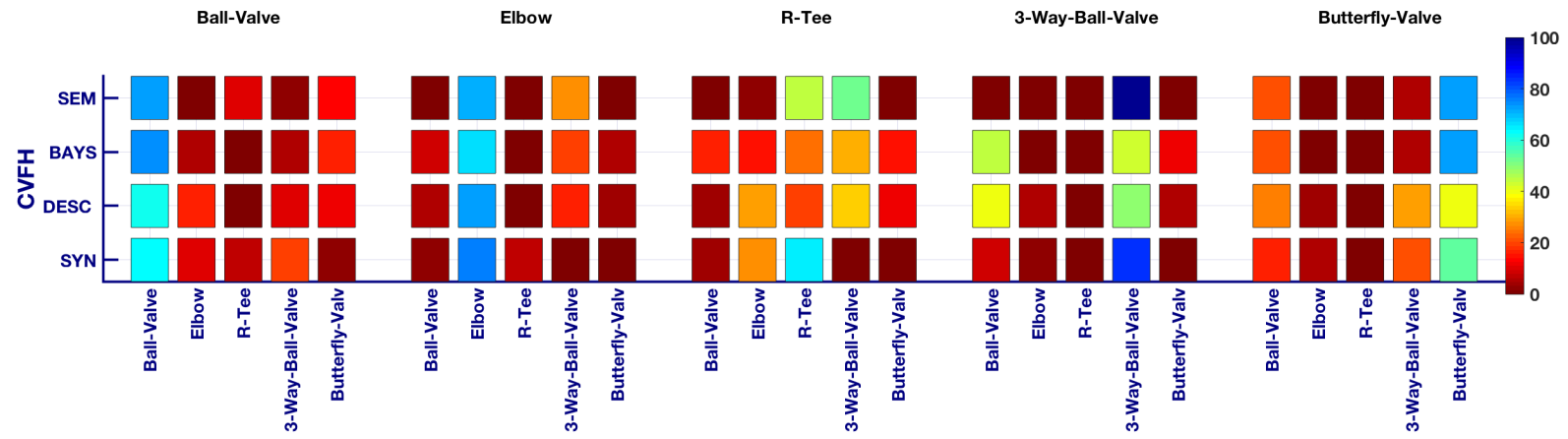
Figure 16. Confusion matrices.

6. Conclusions

This paper has presented a semantic mapping method using non-coloured point clouds and navigation sensor data. The method includes semantic segmentation (of planes, pipes and objects) paired with a feature-based SLAM filter and a semantic-based recognition based on multiple views of each tracked object. The methods were tested against real data gathered with an AUV in a water tank with a man-made pipe structure.

Semantic segmentation attained better performance in selecting the sets of points belonging to each object than in our previous work. This reduced the negative impact of the presence of points belonging to pipes that made recognition more difficult. The "mushroom" shape bounding box used over the pipe intersections allowed the computation of object candidates with the potential presence of handles, thus enabling a better crop of the input scan that tightly encapsulates the object with handle to be recognized.

Feature-based SLAM provided an accurate object tracking that allowed the integration of multiple views of the same object acquired at different times in order to better estimate their class. Moreover, it produced a consistent map of the structure while also providing navigation corrections that compensated for the effects of inconsistencies in navigation due to errors in DVL measurements. The integration of the recognition and the SLAM module, where information is passed back and forth, was instrumental to the higher performance of the approach and to the ability to create a semantic map of all recognized objects.

7. Future Work

Future research plans will continue in the direction of combining the representations of SLAM and object recognition to provide a more accurate and detailed 3D semantic map while providing recognition with more complete views. The object recognition approach we used, based on SLAM, mainly consists of two modules: a Bayesian semantic information-based method for recognition and the SLAM system. Since SLAM provides long-term consistent navigation, one future improvement will be to use this navigation to fuse several scans, which will provide more comprehensive views of the objects. Having more complete views has the potential to improve both the accuracy of object recognition and the reliability of pose estimates, especially in challenging scenarios with significant changes in viewpoints.

A longer term strategy to improve the observation quality is to perform view planning in order to reduce the ambiguity caused by poorly observed objects. Such view planning should be multi-objective in the sense of taking into account multiple objects simultaneously and should be guided towards the next best views that solve the ambiguity between the most probable classes for each object. Continuing this work, future efforts will be directed toward the goal of grasping and manipulating such objects.

Author Contributions: Conceptualization, G.V., K.H., P.R. and N.G.; investigation, G.V., K.H., P.R. and N.G.; methodology, G.V., K.H., P.R. and N.G.; software, G.V., K.H.; supervision, P.R. and N.G.; writing, G.V., K.H., P.R. and N.G. All authors have read and agreed to the published version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

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

References

1. Ridao, P.; Carreras, M.; Ribas, D.; Sanz, P.J.; Oliver, G. Intervention AUVs: The next challenge. *Annu. Rev. Control.* **2015**, *40*, 227–241. doi:https://doi.org/10.1016/j.arcontrol.2015.09.015.

2. Cieslak, P.; Ridao, P.; Giergiel, M. Autonomous underwater panel operation by GIRONA500 UVMS: A practical approach to autonomous underwater manipulation. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation, Seattle, WA, USA, 26–30 May 2015; pp. 529–536. doi:10.1109/ICRA.2015.7139230.

3. Carreras, M.; Carrera, A.; Palomeras, N.; Ribas, D.; Hurtós, N.; Salvi, Q.; Ridao, P. Intervention Payload for Valve Turning with an AUV. In *Computer Aided Systems Theory—EUROCAST 2015*; Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 877–884.

4. Youakim, D.; Ridao, P.; Palomeras, N.; Spadafora, F.; Ribas, D.; Muzzupappa, M. MoveIt!: Autonomous Underwater Free-Floating Manipulation. *IEEE Robot. Autom. Mag.* **2017**, *24*, 41–51. doi:10.1109/MRA.2016.2636369.

5. Sanz, P.J.; Ridao, P.; Oliver, G.; Casalino, G.; Petillot, Y.; Silvestre, C.; Melchiorri, C.; Turetta, A. TRIDENT An European project targeted to increase the autonomy levels for underwater intervention missions. In Proceedings of the 2013 OCEANS-San Diego, San Diego, CA, USA, 23–27 September 2013; pp. 1–10.

6. Himri, K.; Ridao, P.; Gracias, N. Underwater Object Recognition Using Point-Features, Bayesian Estimation and Semantic Information. *Sensors* **2021**, *21*, 1807. doi:10.3390/s21051807.

7. Kinsey, J.C.; Whitcomb, L.L. Preliminary field experience with the DVLNAV integrated navigation system for oceanographic submersibles. *Control. Eng. Pract.* **2004**, *12*, 1541–1549. doi:10.1016/j.conengprac.2003.12.010.

8. Thomas, H.G. GIB Buoys: An Interface Between Space and Depths of the Oceans. In Proceedings of the 1998 Workshop on Autonomous Underwater Vehicles (Cat. No.98CH36290), Cambridge, MA, USA, 21 August 1998 ; pp. 181–184. doi:10.1109/AUV.1998.744453.

9. Mandt, M.; Gade, K.; Jalving, B. Integrateing DGPS-USBL position measurements with inertial navigation in the HUGIN 3000 AUV. In Proceedings of the 8th Saint Petersburg International Conference on Integrated Navigation Systems, St. Petersburg, Russia, 28–30 May 2001.

10. Alcocer, A.; Oliveira, P.; Pascoal, A. Study and implementation of an EKF GIB-based underwater positioning system. *Control. Eng. Pract.* **2007**, *15*, 689–701. doi:https://doi.org/10.1016/j.conengprac.2006.04.001.

11. Melo, J.; Matos, A. Survey on advances on terrain based navigation for autonomous underwater vehicles. *Ocean. Eng.* **2017**, *139*, 250–264. doi:https://doi.org/10.1016/j.oceaneng.2017.04.047.

12. Ribas, D.; Ridao, P.; Domingo, J.D.; Neira, J. Underwater SLAM in Man-Made Structured Environments. *J. Field Robot.* **2008**, *25*, 898–921. doi:10.1002/rob.

13. He, B.; Liang, Y.; Feng, X.; Nian, R.; Yan, T.; Li, M.; Zhang, S. AUV SLAM and experiments using a mechanical scanning forward-looking sonar. *Sensors* **2012**, *12*, 9386–9410.

14. Fallon, M.F.; Folkesson, J.; McClelland, H.; Leonard, J.J. Relocating underwater features autonomously using sonar-based SLAM. *IEEE J. Ocean. Eng.* **2013**, *38*, 500–513.

15. Burguera, A.; González, Y.; Oliver, G. The UspIC: Performing Scan Matching Localization Using an Imaging Sonar. *Sensors* **2012**, *12*, 7855–7885. doi:10.3390/s120607855.

16. Mallios, A.; Ridao, P.; Ribas, D.; Carreras, M.; Camilli, R. Toward autonomous exploration in confined underwater environments. *J. Field Robot.* **2016**, *33*, 994–1012.

17. Vallicrosa, G.; Ridao, P. H-SLAM: Rao-Blackwellized Particle Filter SLAM Using Hilbert Maps. *Sensors* **2018**, *18*, 1386. doi:10.3390/s18051386.

18. Fairfield, N.; Kantor, G.; Wettergreen, D. Towards particle filter SLAM with three dimensional evidence grids in a flooded subterranean environment. In Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA 2006), Orlando, FL, USA, 15–19 May 2006; pp. 3575–3580. doi:10.1109/ROBOT.2006.1642248.

19. Roman, C.; Singh, H. A Self-Consistent Bathymetric Mapping Algorithm. *J. Field Robot.* **2007**, *24*, 23–50. doi:10.1002/rob.

20. Barkby, S.; Williams, S.B.; Pizarro, O.; Jakuba, M. A Featureless Approach to Efficient Bathymetric SLAM Using Distributed Particle Mapping. *J. Field Robot.* **2011**, *28*, 19–39. doi:10.1002/rob.

21. Palomer, A.; Ridao, P.; Ribas, D. Multibeam 3D Underwater SLAM with Probabilistic Registration. *Sensors* **2016**, *16*, 560. doi:10.3390/s16040560.

22. Eustice, R.; Pizarro, O.; Singh, H. Visually Augmented Navigation in an Unstructured Environment Using a Delayed State History. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '04), New Orleans, LA, USA, 26 April–1 May 2004; pp. 25–32. doi:10.1109/ROBOT.2004.1307124.

23. Williams, S.; Mahon, I. Simultaneous Localisation and Mapping on the Great Barrier Reef. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '04), New Orleans, LA, USA, 26 April–1 May 2004; pp. 1771–1776.

24. Eustice, R.; Singh, H.; Leonard, J.; Walter, M.; Ballard, R. Visually Navigating the RMS Titanic with SLAM Information Filters. In *Proceedings of the Robotics Science and Systems*; MIT Press: Cambridge, MA, USA, 2005.

25. Johnson-Roberson, M.; Pizarro, O.; Williams, S.B.; Mahon, I. Generation and Visualization of Large-Scale Three-Dimensional Reconstructions from Underwater Robotic Surveys. *J. Field Robot.* **2010**, *27*, 21–51. doi:10.1002/rob.

26. Gracias, N.; Ridao, P.; Garcia, R.; Escartin, J.; Cibecchini, F.; Campos, R.; Carreras, M.; Ribas, D.; Magi, L.; Palomer, A.; et al. Mapping the Moon: Using a lightweight AUV to survey the site of the 17th Century ship 'La Lune'. In Proceedings of the MTS/IEEE OCEANS Conference, Bergen, Norway, 10–14 June 2013.

27. Campos, R.; Gracias, N.; Palomer, A.; Ridao, P. Global Alignment of a Multiple-Robot Photomosaic using Opto-Acoustic Constraints. *IFAC-PapersOnLine* **2015**, *48*, 20–25. doi:10.1016/j.ifacol.2015.06.004.

28. Inglis, G.; Smart, C.; Vaughn, I.; Roman, C. A pipeline for structured light bathymetric mapping. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 4425–4432.

29. Massot-Campos, M.; Oliver, G.; Bodenmann, A.; Thornton, B. Submap bathymetric SLAM using structured light in underwater environments. In Proceedings of the 2016 IEEE/OES Autonomous Underwater Vehicles (AUV), Tokyo, Japan, 6–9 November 2016; pp. 181–188. doi:10.1109/AUV.2016.7778669.

30. Palomer, A.; Ridao, P.; Forest, J.; Ribas, D. Underwater Laser Scanner: Ray-Based Model and Calibration. *IEEE/ASME Trans. Mechatronics* **2019**, *24*, 1986–1997. doi:10.1109/TMECH.2019.2929652.

31. Palomer, A.; Ridao, P.; Youakim, D.; Ribas, D.; Forest, J.; Petillot, Y. 3D Laser Scanner for Underwater Manipulation. *Sensors* **2018**, *18*, 1086. doi:10.3390/s18041086.

32. Palomer, A.; Ridao, P.; Ribas, D. Inspection of an underwater structure using point-cloud SLAM with an AUV and a laser scanner. *Journal of Field Robotics* **2019**, *36*, 1333–1344.

33. Himri, K.; Ridao, P.; Gracias, N. 3D Object Recognition Based on Point Clouds in Underwater Environment with Global Descriptors: A Survey. *Sensors* **2019**, *19*, 4451.

34. Guo, Y.; Bennamoun, M.; Sohel, F.; Lu, M.; Wan, J. 3D object recognition in cluttered scenes with local surface features: a survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2270–2287.

35. Alexandre, L.A. 3D descriptors for object and category recognition: a comparative evaluation. In Proceedings of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal, 7–12 October 2012; Volume 1, p. 7.

36. Guo, Y.; Wang, H.; Hu, Q.; Liu, H.; Liu, L.; Bennamoun, M. Deep learning for 3D point clouds: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, doi:10.1109/TPAMI.2020.3005434.

37. Tian, Y.; Chen, L.; Song, W.; Sung, Y.; Woo, S. DGCB-Net: Dynamic Graph Convolutional Broad Network for 3D Object Recognition in Point Cloud. *Remote. Sens.* **2021**, *13*, 66.

38. Martin-Abadal, M.; Piñar-Molina, M.; Martorell-Torres, A.; Oliver-Codina, G.; Gonzalez-Cid, Y. Underwater Pipe and Valve 3D Recognition Using Deep Learning Segmentation. *J. Mar. Sci. Eng.* **2020**, *9*, 5.

39. Pereira, M.I.; Claro, R.M.; Leite, P.N.; Pinto, A.M. Advancing Autonomous Surface Vehicles: A 3D Perception System for the Recognition and Assessment of Docking-Based Structures. *IEEE Access* **2021**, *9*, 53030–53045.

40. Pi, R.; Cieślak, P.; Ridao, P.; Sanz, P.J. TWINBOT: Autonomous Underwater Cooperative Transportation. *IEEE Access* **2021**, *9*, 37668–37684. doi:10.1109/ACCESS.2021.3063669.

41. Nüchter, A.; Hertzberg, J. Towards semantic maps for mobile robots. *Robot. Auton. Syst.* **2008**, *56*, 915–926.

42. Balaska, V.; Bampis, L.; Boudourides, M.; Gasteratos, A. Unsupervised semantic clustering and localization for mobile robotics tasks. *Robot. Auton. Syst.* **2020**, *131*, 103567.

43. Kostavelis, I.; Gasteratos, A. Learning spatially semantic representations for cognitive robot navigation. *Robot. Auton. Syst.* **2013**, *61*, 1460–1475.

44. Kim, D.I.; Sukhatme, G.S. Semantic labeling of 3d point clouds with object affordance for robot manipulation. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 5578–5584.

45. Civera, J.; Gálvez-López, D.; Riazuelo, L.; Tardós, J.D.; Montiel, J.M.M. Towards semantic SLAM using a monocular camera. In Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; pp. 1277–1284.

46. Tang, Z.; Wang, G.; Xiao, H.; Zheng, A.; Hwang, J.N. Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Salt Lake City, UT, USA, 18–22 June 2018; pp. 108–115.

47. Shao, T.; Xu, W.; Zhou, K.; Wang, J.; Li, D.; Guo, B. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph. (TOG)* **2012**, *31*, 1–11.

48. Song, S.; Yu, F.; Zeng, A.; Chang, A.X.; Savva, M.; Funkhouser, T. Semantic scene completion from a single depth image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1746–1754.

49. Dewan, A.; Oliveira, G.L.; Burgard, W. Deep semantic classification for 3d lidar data. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 3544–3549.

50. Chen, S.W.; Nardari, G.V.; Lee, E.S.; Qu, C.; Liu, X.; Romero, R.A.F.; Kumar, V. Sloam: Semantic lidar odometry and mapping for forest inventory. *IEEE Robot. Autom. Lett.* **2020**, *5*, 612–619.

51. Milioto, A.; Vizzo, I.; Behley, J.; Stachniss, C. Rangenet++: Fast and accurate lidar semantic segmentation. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 4213–4220.

52. Liu, F.; Fang, M. Semantic segmentation of underwater images based on improved Deeplab. *J. Mar. Sci. Eng.* **2020**, *8*, 188.

53. Miguelanez, E.; Patron, P.; Brown, K.E.; Petillot, Y.R.; Lane, D.M. Semantic knowledge-based framework to improve the situation awareness of autonomous underwater vehicles. *IEEE Trans. Knowl. Data Eng.* **2010**, *23*, 759–773.

54. Girdhar, Y.; Dudek, G. Exploring underwater environments with curiosity. In Proceedings of the 2014 Canadian Conference on Computer and Robot Vision, Montreal, QC, Canada, 6–9 May 2014, pp. 104–110.

55. Rabbani, T.; Heuvel, F.; Vosselman, G. Segmentation of point clouds using smoothness constraint. *Int. Arch. Photogramm. Remote. Sens. Spat. Inf. Sci.* **2006**, *36*, 248–253.

56. Aldoma, A.; Vincze, M.; Blodow, N.; Gossow, D.; Gedikli, S.; Rusu, R.B.; Bradski, G. CAD-model recognition and 6DOF pose estimation using 3D cues. In Proceedings of the 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), Barcelona, Spain, 6–13 November 2011; pp. 585–592.

57. Yang, Y.; Huang, G. Aided inertial navigation: Unified feature representations and observability analysis. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 3528–3534.

58. Ruifang, D.; Frémont, V.; Lacroix, S.; Fantoni, I.; Changan, L. Line-based monocular graph SLAM. In Proceedings of the 2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), Daegu, Korea, 16–18 November 2017; pp. 494–500.

59. Neira, J.; Tardós, J.D. Data association in stochastic mapping using the joint compatibility test. *IEEE Trans. Robot. Autom.* **2001**, *17*, 890–897.

60. Ribas, D.; Palomeras, N.; Ridao, P.; Carreras, M.; Mallios, A. Girona 500 AUV: From Survey to Intervention. *IEEE/ASME Trans. Mechatronics* **2012**, *17*, 46–53. doi:10.1109/TMECH.2011.2174065.

61. Himri, K.; Ridao, P.; Gracias, N.; Palomer, A.; Palomeras, N.; Pi, R. Semantic SLAM for an AUV using object recognition from point clouds. *IFAC-PapersOnLine* **2018**, *51*, 360–365.

62. Johnson, J.M.; Khoshgoftaar, T.M. Survey on deep learning with class imbalance. *J. Big Data* **2019**, *6*, 1–54.