Submapping SLAM based on acoustic data from a 6-DOF AUV ⋆

Josep Aulinas * **Chee Sing Lee** * **Joaquim Salvi** *
Yvan R. Petillot **

* *Computer Vision and Robotics group (ViCoRob), University of Girona, 17071 Girona, Spain (e-mail: {cslee,jaulinas,qsalvi}@eia.udg.edu)*
** *Oceans Systems Lab, School of EPS, Heriot-Watt University, EH144AS Edinburgh, United Kindgom (e-mail: Y.R.Petillot@hw.ac.uk)*

Abstract: Autonomous Underwater Vehicles (AUVs) need positioning systems besides the Global Positioning System (GPS), since GPS does not work in underwater scenarios. Possible solutions are the Simultaneous Localization and Mapping (SLAM) algorithms. SLAM algorithms aim to build a map while simultaneously localizing the vehicle within this map. However, they offer limited performance when faced with large scale scenarios. For instance, they do not create consistent maps for large areas, mainly because uncertainties increase with the scale of the scenario. In addition, the computational cost increases with the map size. The use of local maps reduces computational cost and improves map consistency. Following this idea, in this paper we propose a new SLAM approach that uses independent local maps together with a global level stochastic map. The global level contains the relative transformations between local maps. These local maps are updated once a new loop is detected. Local maps that are sharing a high number of features are updated through fusion, maintaining the correlation between landmarks and vehicle. Experimental results on real data obtained from the REMUS-100 AUV show that our approach is able to obtain large map areas consistently.

Keywords: Autnonomous vehicles, robotics, navigation

1. INTRODUCTION

Mapping and localization techniques are necessary for many underwater applications. Some examples of these applications are underwater cartography, geological mapping, off-shore structures inspection, studies of biodiversity or deep-water archaeology. Different underwater vehicles have been developed in order to explore completely unknown underwater regions, for instance the so called Autonomous Underwater Vehicles (AUVs). An AUV is equipped with onboard sensors, which provide information about the vehicle, such as speeds, orientations or accelerations, and about the environment, such as 3D clouds of points from the sea floor or the relative location of salient features with respect to the vehicle. This information is very valuable to calculate the approximate position of the vehicle.

Terrestrial and aerial vehicles can localize themselves with Global Positioning System (GPS). However, underwater, GPS can not be used because electromagnetic waves are strongly attenuated through the medium of water. A standard for bounded `xyz` navigational position measurements for underwater vehicles is the long-baseline (LBL) acoustic transponder system (Hunt et al. (1974)). LBL operates on the principle of time-of-flight and it is been proven to operate up to a range of 10 km (Whitcomb et al. (1999)). The main drawback of LBL is that it requires two or more acoustic transponder beacons to be tethered to the sea floor. Short-baseline (SBL) systems provide more accurate positioning information, but suffer from the same drawbacks than the LBL. Internal sensors, such as the Inertial Measurement Unit (IMU) and the Doppler Velocity Log (DVL) do not give absolute localization, therefore the localization problem suffers from drift due to odometric noise. Furthermore, the detection of salient features in the environment is a complex task due to measurement noise. These noises makes the mapping and localization a difficult challenge. Simultaneous Localization and Mapping (SLAM), also known as Concurrent Mapping and Localization (CML), is one of the fundamental challenges of robotics (Durrant-Whyte and Bailey (2006)). The SLAM problem involves a joint task of simultaneously estimating the map and localizing the vehicle inside this map.

A well known and widely used SLAM approach is the Extended Kalman Filter SLAM (EKF-SLAM) (Smith et al. (1988)). EKF-SLAM represents the vehicle's pose and the location of a set of environment features in a joint state vector. This vector is estimated and updated by the EKF. The EKF provides a suboptimal solution due to

⋆ The authors also acknowledge the support of the research project DPI-2007-66796-C03-02 funded by the Spanish Ministry of Science and Innovation. J. Aulinas holds a University of Girona scholarship (BR-07/03). C.S. Lee holds a grant from the Catalan government (IUE/2365/2009)

several approximations and assumptions, which result in divergences (Castellanos et al. (2007)). In large areas, EKF complexity grows with the number of landmarks, because each landmark is correlated to all other landmarks. This means that EKF memory complexity is $O(n^2)$ and a time complexity of $O(n^2)$ per step, where n is the total number of features stored in the map.

The use of submaps has been shown to address the problems of consistency and computational complexity. An early example of this strategy is the Decoupled Stochastic Map (DSM) (Leonard and Feder (2000)). The DSM uses non-statistically independent submaps. Therefore the correlations are broken introducing inconsistency in the map. Similar inconsistencies were seen in (Aulinas et al. (2010)), where submaps were assumed to be independent but still shared information. Different techniques, such as the Constrained Local Submap Filter (CLSF) (Williams et al. (2002)) or Local Map Joining (MJS) (Tardós et al. (2002)) produce efficient global maps by consistently combining completely independent local maps. The Divide and Conquer SLAM (DCS) (Paz et al. (2008)) is capable to recover the global map in approximately $O(n)$ time. The Constant Time SLAM (CTS) (Leonard and Newman (2003)), the Atlas approach (Bosse et al. (2004)), and the Hierarchical SLAM (HS) (Estrada et al. (2005)) store the link between local maps by means of an adjacency graph. The HS imposes loop constraints on the adjacency graph, producing a better estimation of the global level map. The Conditionally Independent Local Maps (CILM) (Piniés and Tardós (2008)), is based on sharing information between consecutive submaps. This way, a new local map is initialized considering the *a-priori* knowledge.

These submapping techniques demonstrate that using submaps, both linearization errors and computational cost can be addressed at the same time, improving the consistency of EKF-SLAM (Castellanos et al. (2007)). Only few of them have been tested on underwater scenarios (Williams (2001); Roman and Singh (2007)), where some extra constraints have to be taken into account. Firstly, the terrain sensing is limited to either acoustics (Ribas (2008)) or near-field vision (Eustice (2005)), because electromagnetic waves are strongly attenuated in the water. Secondly, underwater scenarios are in general unstructured and require 3D navigation (6-DOF motion), while most current SLAM solutions are used on man-made (geometrically simple) indoor spaces, where a 2D map representation is sufficient. Therefore, the use of SLAM on AUV navigations requires further testing and improvements.

The main contribution of our approach is the strategy used to decide whether to fuse the submaps. This decision is made on the basis that fusing two maps that share many landmarks will produce a better update than fusing two maps that only share a few landmarks. The experiments done with real data show a a bounded effect of the linearization error and also a precise reconstruction of the map since the drift suffered in shorter distances is smaller, and the data association can be more robustly solved as compared to other state of the art techniques.

The rest of the paper is structured as follows: Section 2 describes the novelty of our SLAM approach. The standard EKF and the map fusion approach are also presented. Section 3 describes the experimental setup and the results obtained using a 6-DOF vehicle. Finally, conclusions and future work are presented in Section 4.

Algorithm I: Selective Submap Joining SLAM

```
begin mission
while navigating do
    x̂_i, P̂_i = EKF SLAM() ← (Build submap M_i)
    x̂_G, P̂_G = build_global_map(x̂_i, P̂_i)
    H_Loop = check_possible_loops(x̂_G, P̂_G)
    for j = H_Loop do
        refer M_i and M_j to a common base reference
        H_ij = data_association(x̂_i, x̂_j, P̂_i, P̂_j)
        if H_ij > threshold then
            x̂_ij, P̂_ij = map_fusion(x̂_i, P̂_i, x̂_j, P̂_j, H_ij)
            x̂_G, P̂_G = update_global_map(x̂_ij, P̂_ij)
        endif
    endfor
endwhile
```

2. SELECTIVE SUBMAP JOINING BASED SLAM

The basis of the Selective Submap Joining SLAM (SSJS) (see Algorithm I) lies in the EKF-based SLAM. A sequence of EKF-based submaps is built, as explained in Subsection 2.1. The size of these submaps is predefined by the total number of features per map and by the uncertainty boundaries. The links between local maps are stored in a global level map, as described in Subsection 2.2. This graph information allows checking whether a loop closing event is occurring, following the strategy presented in Subsection 2.3. The main novelty of our approach lies in the fact that upon loop closure, we decide to fuse two maps or to keep them independent depending on the number of common landmarks, in contrast to other approaches that fuse maps regardless of the information they share (Williams (2001); Tardós et al. (2002); Estrada et al. (2005); Paz et al. (2008)).

2.1 Map Building

A map is built using a standard EKF algorithm (Algorithm II). The EKF estimates the state, at a certain time step k, of a dynamic non-linear system from a series of incomplete and noisy measurements, as its mean x_k and the covariance P_k. The algorithm iterates continously through three steps: prediction, observation and update. The prediction stage uses the motion model f to estimate the current state $\widehat{\mathbf{x}}_k$ from the previous time step $\mathbf{x}_{k-1}$, and control inputs (i.e. odometry) $\mathbf{u}_k$, if available. (see (1)). The hat notation denotes an estimate based only on this prediction, before corrections from sensor input.

$$\widehat{\mathbf{x}}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) \qquad \widehat{\mathbf{P}}_k = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T \quad (1)$$

In general, the motion model is a non-linear function, which requires the following linearizations for predicting the state covariance at time k:

$$\mathbf{F}_k = \frac{\partial f}{\partial \mathbf{x}} \mid_{\mathbf{x}_{k-1}} \qquad \mathbf{G}_k = \frac{\partial f}{\partial \mathbf{x}} \mid_{\mathbf{u}_k} \quad (2)$$

Algorithm II: Map Building, EKF-SLAM

map_initialization()
$\mathbf{z}_0, \mathbf{R}_0$ = get_measurements()
$\mathbf{x}_0, \mathbf{P}_0$ = add_features()
for $k = 1$ until end of map **do**
  $\mathbf{x}_{od}, \mathbf{Q}_{od}$ = get_odometry()
  $\widehat{\mathbf{x}}_{k|k-1}, \widehat{\mathbf{P}}_{k|k-1}$ = EKF_prediction($\widehat{\mathbf{x}}_{k-1}$,$\widehat{\mathbf{P}}_{k-1}$, $\mathbf{x}_{od}$,$\mathbf{Q}_{od}$)
  $\mathbf{z}_k, \mathbf{R}_k$ = get_measurements()
  $\mathcal{H}_k$ = data_association($\widehat{\mathbf{x}}_{k|k-1}$,$\mathbf{z}_k$,$\widehat{\mathbf{P}}_{k|k-1}$,$\mathbf{R}_k$)
  $\widehat{\mathbf{x}}_k, \widehat{\mathbf{P}}_k$ = EKF_update($\widehat{\mathbf{x}}_{k|k-1}$,$\widehat{\mathbf{P}}_{k|k-1}$, $\mathbf{z}_k$,$\mathbf{R}_k$,$\mathcal{H}_k$)
  $\widehat{\mathbf{x}}_k, \widehat{\mathbf{P}}_k$ = add_features($\widehat{\mathbf{x}}_k$,$\widehat{\mathbf{P}}_k$,$\mathbf{z}_k$,$\mathbf{R}_k$,$\mathcal{H}_k$)
endfor
return: $\mathcal{M}_i = \{\widehat{\mathbf{x}}_k, \widehat{\mathbf{P}}_k\}$

*NOTE: end of map could either be the end of the mission or a predefined interruption, for instance a uncertainty boundary has been reached.

During the observation phase the vehicle's onboard sensors provide measurements about the true EKF state. These are stored in an observation vector $\mathbf{z}_k$. In addition, an observation model gives the predicted sensor readings from the state estimate. This is represented by the non-linear function $\widehat{\mathbf{z}}_k = h(\widehat{\mathbf{x}}_k)$. The *a-priori* predictions from the observation model $\widehat{\mathbf{z}}_k$ are associated with the current measurements using the Joint Compatibility Branch and Bound (JCBB) by (Neira and Tardós (2001)). The JCBB is very robust because it considers the relative location between features. Knowing the data association $\mathcal{H}_k$, the innovation vector ν_k and the associated innovation covariance matrix $\mathbf{S}_k$ can then be calculated (See (3)). These are then used for the EKF update stage.

$$\begin{aligned} \widehat{\mathbf{z}}_k &= f(\widehat{\mathbf{x}}_k) \\ \nu_k &= \mathbf{z}_k - \widehat{\mathbf{z}}_k \\ \mathbf{S}_k &= \mathbf{H}_k \widehat{\mathbf{P}}_{k-1} \mathbf{H}_k^T + \mathbf{R}_k \end{aligned} \tag{3}$$

The term $\mathbf{R}_k$ represents the covariance of the zero mean white Gaussian observation noise ($\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$), and $\mathbf{H}_k$ is the linearization of the observation model given by (4)

$$\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}} \mid_{\widehat{\mathbf{x}}_k} \tag{4}$$

The update stage provides an improved *a-posteriori* state estimate using ν_k and the Kalman gain $\mathbf{W}_k$.

$$\begin{aligned} \mathbf{W}_k &= \widehat{\mathbf{P}}_k \mathbf{H}_k^T \mathbf{S}_k^{-1} \\ \mathbf{x}_k &= \widehat{\mathbf{x}}_k + \mathbf{W}_k \nu_k \\ \mathbf{P}_k &= (\mathbf{I} - \mathbf{W}_k \mathbf{H}_k) \widehat{\mathbf{P}}_k (\mathbf{I} - \mathbf{W}_k \mathbf{H}_k)^T + \mathbf{W}_k \mathbf{R}_k \mathbf{W}_k^T \end{aligned} \tag{5}$$

The observations that were not associated to any existing landmark in the map are considered as new map features, and x_k and P_k are augmented to include the new features.

2.2 Submapping and Global Map Building

Local maps $\mathcal{M}_i \quad \mathcal{M}_{i+1} \quad ... \quad \mathcal{M}_j$ are built sequentially. The reference frame of a submap is at the vehicle's starting point. This starting point of a local map $\mathcal{M}_{i+1}$ coincides with the last position of the previous map $\mathcal{M}_i$. Therefore, the relative transformation between two consecutive maps ${}^{\mathcal{M}_i}_{\mathcal{M}_{i+1}}\mathcal{T}$ is the vehicle's pose at the last position of $\mathcal{M}_i$. This link is stored in a global level map $\mathbf{x}_G$ together with the vehicle uncertainty at that position $\mathbf{P}_G$, see (6). A new link is then stored every time a local map reaches its end. The information contained in this global level is very important to detect loop events, since a local map can be referred *w.r.t.* the frame reference of any other local map.

$$\mathbf{x}_G = \begin{bmatrix} . \\ {}^{\mathcal{M}_{i-1}}_{\mathcal{M}_i}\mathcal{T} \\ {}^{\mathcal{M}_i}_{\mathcal{M}_{i+1}}\mathcal{T} \\ . \end{bmatrix} \quad \mathbf{P}_G = \begin{bmatrix} . & . & . & . \\ . & \sigma^2_{{}^{\mathcal{M}_{i-1}}_{\mathcal{M}_i}\mathcal{T}} & 0 & . \\ . & 0 & \sigma^2_{{}^{\mathcal{M}_i}_{\mathcal{M}_{i+1}}\mathcal{T}} & . \\ . & . & . & . \end{bmatrix} \tag{6}$$

2.3 Loop Closing Strategy

A sequence of submaps and a global level stochastic map have been built. Navigating in loops improves the map consistency by revisiting previously seen portions of the map which have higher levels of certainty, and propagating the improved estimations through subsequently visited regions. Our loop closing procedure begins with a search within the global map level, every time a submap is finished. According to the proximity of different submaps in the global level, loop closing hypotheses are made. Afterwards, the data association between those maps defined as loop closing candidates is computed. If the correspondences between maps are higher than a *threshold*, they are joined and fused to a single map. Together with the map fusion, the corresponding link in the global level is corrected. This correction is directly obtained from the map fusion since the links within the fused maps are correlated and updated with all the information.

The map joining approach used is the one presented by Tardós et al. (2002). Two independent local maps $\mathcal{M}_i^{\mathcal{B}} = (\mathbf{x}_i^{\mathcal{B}}, \mathbf{P}_i^{\mathcal{B}})$, that contain a set of n features $F_1, ..., F_n$, and $\mathcal{M}_j^{\mathcal{B}'} = (\mathbf{x}_j^{\mathcal{B}'}, \mathbf{P}_j^{\mathcal{B}'})$, that contains a set of m features $E_1, ..., E_m$, are joined by referring them both to the same reference. Therefore, the features from both maps are expressed relative to the same base B forming a joint state vector $\mathbf{x}_{i+j}^{\mathcal{B}}$ and covariance matrix $\mathbf{P}_{i+j}^{\mathcal{B}}$ (see (7)).

$$\mathbf{x}_{i+j}^{\mathcal{B}} = \begin{bmatrix} \mathbf{x}_i^{\mathcal{B}} \\ \mathbf{x}_j^{\mathcal{B}} \end{bmatrix} \qquad \mathbf{P}_{i+j}^{\mathcal{B}} = \mathbf{J}_i \mathbf{P}_i^{\mathcal{B}} \mathbf{J}_i^T + \mathbf{J}_j \mathbf{P}_j^{\mathcal{B}} \mathbf{J}_j^T \tag{7}$$

Where the matrices for the Jacobian $\mathbf{J}_i$ and $\mathbf{J}_j$ are in (8).

$$\mathbf{J}_i = \frac{\partial \mathbf{x}_{i+j}^{\mathcal{B}}}{\partial \mathbf{x}_i^B} \qquad \mathbf{J}_j = \frac{\partial \mathbf{x}_{i+j}^{\mathcal{B}}}{\partial \mathbf{x}_j^B} \tag{8}$$

Assuming that the number of features shared between $\mathcal{M}_i^{\mathcal{B}}$ and $\mathcal{M}_j^{\mathcal{B}}$ is above the threshold, a data association algorithm such as the JCBB is carried out. Features from $\mathcal{M}_j^{\mathcal{B}}$ are understood as new measurements for the features from $\mathcal{M}_i^{\mathcal{B}}$. Therefore, a non-linear measurement matrix $\widehat{\mathbf{z}}_{ij_i} = \mathbf{h}_{ij_i}$ mapping a feature $E_{j_i}^{\mathcal{B}}$ corresponding to a feature $F_i^{\mathcal{B}}$ needs to be linearized by means of Jacobian computation (see (9)).

$$\mathbf{H}_{ij_i} = \left[\frac{\partial \mathbf{h}_{ij_i}}{\partial \mathbf{x}_{v_i}^B} \; \mathbf{0} \; ... \; \mathbf{0} \; \frac{\partial \mathbf{h}_{ij_i}}{\partial \mathbf{x}_{F_i}^B} \; \mathbf{0} \; ... \; \mathbf{0} \; \frac{\partial \mathbf{h}_{ij_i}}{\partial \mathbf{x}_{E_{j_i}}^{B'}} \; \mathbf{0} \; ... \; \mathbf{0} \right] \tag{9}$$

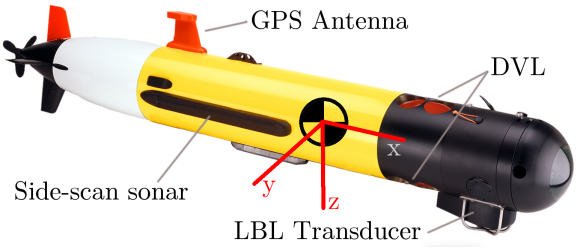

Fig. 1. The REMUS-100 AUV.

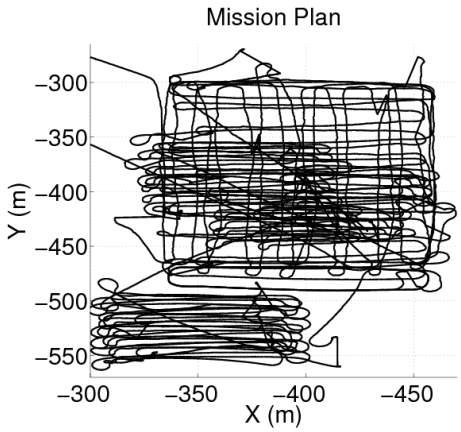

Fig. 2. Vehicle trajectory for the entire mission.

Afterwards, the local map information and links between maps are improved using the EKF update equations (see (5)). Once the joint maps have been updated, the rows and columns corresponding to common landmarks from $\mathcal{M}_j$ are removed from the joint state to avoid repetitions.

3. EXPERIMENTAL RESULTS

The AUV REMUS-100 in Fig. 1 was used to test our algorithms with real-world data. The vehicle was sent underwater to perform a reconaissance mission. During the mission the vehicle navigated a large area, about 300m × 400m. The trajectory consisted of a large number of loops, i.e. revisiting the same area several times (Fig. 2). The vehicle maintained a depth of 12-14 meters. The scenario's sea floor was rather flat (oscillating around 16 meters), but with several salient objects. The total navigation time was almost 4 hours. Our algorithms were then tested offline using the navigation and measurement data gathered during the mission.

The vehicle was equipped with a DVL and IMU, giving navigation data relative to the vehicle reference frame such as velocities, orientations and depth. In addition, the vehicle was carrying a side-scan sonar pointing both ways, starboard and port. From the navigation information provided by the sensors, the vehicle state can be defined by a 9D-vector, composed of the 6-DOF vehicle pose $(x\ y\ z\ \phi\ \theta\ \psi)^T$ and the vehicle frame linear velocities $(v_x\ v_y\ v_z)^T$.

The map was composed of objects, rocks and other detectable features. These features' states are defined as 3D points $(x_{l_i}\ y_{l_i}\ z_{l_i})^T$. Note that the 3D point of an object represents the gravity centre of the object. Feature extraction from sonar measurements was accomplished by the Seebyte AutoTracker software (Evans et al. (2003)). In addition to feature information, the side-scan sonar provided a measure of the altitude, i.e. the distance from the sensor to the seabed. Therefore, the joint state vector estimate $\widehat{\mathbf{x}}$ for our problem contains both vehicle state and map information, as shown in (10).

$$\widehat{\mathbf{x}} = (x_V\ \ y_V\ \ z_V\ \ \phi_V\ \ \theta_V\ \ \psi_V\ \ v_x\ \ v_y\ \ v_z\ \ldots \\ \ldots\ x_{l_1}\ y_{l_1}\ z_{l_1}\ \ldots\ x_{l_i}\ y_{l_i}\ z_{l_i}\ \ldots\ x_{l_n}\ y_{l_n}\ z_{l_n})^T \tag{10}$$

The motion model for the REMUS-100 is a 6-DOF constant velocity kinematics model as shown in (11). Where ${}^{\mathbf{x}_{k-1}}_{\mathbf{x}_k}\mathcal{R}$ is the rotation matrix necessary to go from instant $k-1$ to instant k (as explained in Appendix A).

$$\begin{bmatrix} x_k \\ y_k \\ z_k \\ \phi_k \\ \theta_k \\ \psi_k \\ v_{x_k} \\ v_{y_k} \\ v_{z_k} \end{bmatrix} = \begin{bmatrix} {}^{k-1}_{k}\mathcal{R} \begin{bmatrix} v_{x_{k-1}}dt \\ v_{y_{k-1}}dt \\ v_{z_{k-1}}dt \end{bmatrix} + \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \end{bmatrix} \\ \begin{matrix} \phi_{k-1} \\ \theta_{k-1} \\ \psi_{k-1} \\ v_{x_{k-1}} \\ v_{y_{k-1}} \\ v_{z_{k-1}} \end{matrix} \end{bmatrix} \tag{11}$$

The observation model gives the predicted sensor pose from the last known position, and is represented by the non-linear function $\widehat{z}_k = h(\mathbf{x}_k)$ and its linearized version $\mathbf{H}_k$ in (12).

$$\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}} \Big|_{\mathbf{x}_k} = \begin{bmatrix} \mathbf{H}_{k,o} \\ \mathbf{H}_{k,v} \\ \mathbf{H}_{k,d} \end{bmatrix} \tag{12}$$

Through the sensors of our system, we obtain measurements for the vehicle's orientation, linear speeds and the depth, the altitude (seabed depth), and salient feature positions (see Fig. 3). The linearized observation models are the ones shown in (13) for the orientations, (14) for the velocities, and (15) for the depth.

$$\mathbf{H}_{k,o} = \begin{bmatrix} 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ \ldots\ 0 \\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ \ldots\ 0 \\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ \ldots\ 0 \end{bmatrix} \tag{13}$$

$$\mathbf{H}_{k,v} = \begin{bmatrix} 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ \ldots\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ \ldots\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ \ldots\ 0 \end{bmatrix} \tag{14}$$

$$\mathbf{H}_{k,d} = [\,0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \ldots\ 0\,] \tag{15}$$

3.1 Computational Time

Two of the algorithm's parameters, the number of features per submap and the threshold for map joining, were tuned to evaluate computational performance. Figs. 4 and 5 show that for our dataset, the shortest execution times occur for submaps containing between 10 and 15 features, while joining submaps that have above 40% of their features in common.

3.2 Consistency Analysis

Map consistency is shown qualitatively in Fig. 6. This figure compares the trajectory given by dead reckoning

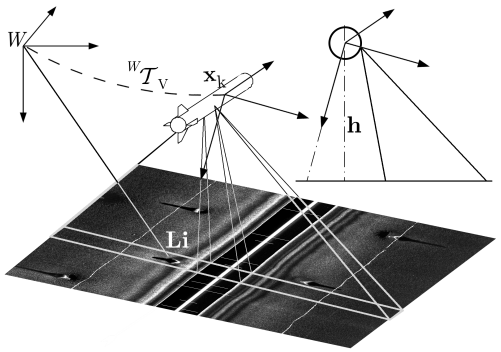

Fig. 3. Schematic representation of the side-scan sonar measurement procedure.

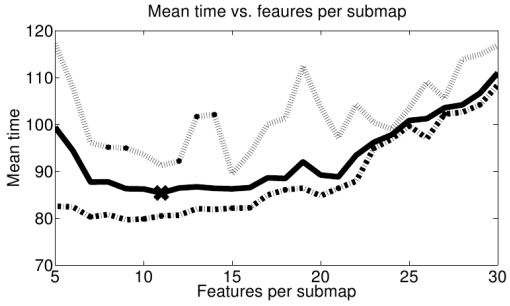

Fig. 4. Minimum, maximum, and mean mission execution times for a given submap size while varying the map joining threshold.

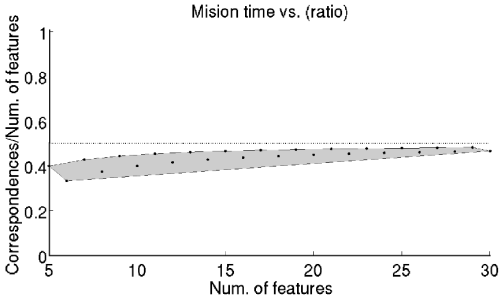

Fig. 5. Threshold values which gave the minimum mission execution time for each submap size.

to the one obtained by our SSJS and LBL. The LBL provides an accurate positioning of the vehicle, therefore it can be considered as the ground truth. Notice that our SSJS SLAM approach clearly improved the dead reckoning (DR). Another example that shows the consistency of our approach is shown in Fig. 7. This figure shows that the discrepancy between LBL and SSJS is always kept inside the uncertainty boundaries, which means that the filter will not cause divergences due to overconfidence. However, when the size of the submaps is increased, the estimation becomes overconfident, which will lead to inconsistencies. This was an expected result, as with large submaps the approach tends to be a standard EKF, leading to the conclusion that submap size is an important parameter for map consistency as well as computational time.

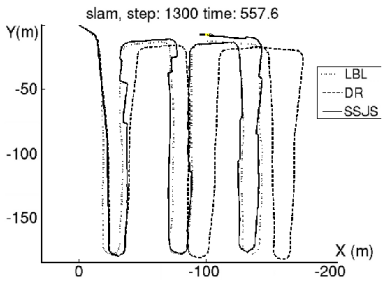

Fig. 6. Detailed excerpt from trajectory estimates. The results shows that our SSJS SLAM provides a path estimate that is closer to the LBL ground truth, compared to dead reckoning (DR).

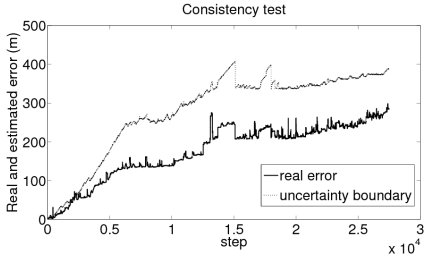

Fig. 7. Example of a consistent configuration for our approach, where the discrepancy between LBL and SSJS (real error) is always lower than the uncertainty boundary (estimated error).

4. CONCLUSIONS

The Selective Submap Joining SLAM approach presented in this paper has been demonstrated to be suitable to consistently map large scale scenarios. The main contribution of our approach is the local map fusion strategy. The local map fusion strategy determines whether or not to fuse two local maps, depending on the amount of information they share. We have shown that our map creates consistent maps under the REMUS-100 Dataset.

As future work, new datasets will be used, as well as the feature extraction algorithm will be added to the SLAM approach in order to check the whole method on-line.

REFERENCES

Aulinas, J., Salvi, J., Lladó, X., and Petillot, Y. (2010). Local map update for large scale slam. *Electronics Letters*, 46(8), 564–566. doi:10.1049/el.2010.2271. URL `http://link.aip.org/link/?ELL/46/564/1`.

Bosse, M., Newman, P., Leonard, J., and Teller, S. (2004). SLAM in large scale cyclic environments using the atlas framework. *International Journal Robotics Research*, 23(12), 1113–1139.

Castellanos, J.A., Martínez-Cantin, R., Tardós, J.D., and Neira, J. (2007). Robocentric map joining: Improving the consistency of EKF-SLAM. *Robotics and Autonomous Systems*, 55(1), 21–29.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping (SLAM): Part I. *IEEE Robotics and Automation Magazine*, 13(2), 99–108.

Estrada, C., Neira, J., and Tardós, J. (2005). Hierarchical SLAM: real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4), 588–596.

Eustice, R. (2005). *Large-area visually augmented navigation for autonomous underwater vehicles*. Master's thesis, Massachusetts Institute of Technology / Woods Hole Oceanographic Institution Joint Program.

Evans, J., Petillot, Y., Redmond, P., Wilson, M., and Lane, D. (2003). AUTOTRACKER: AUV embedded control architecture for autonomous pipeline and cable tracking. In *IEEE OCEANS*, volume 5, 2651 – 2658.

Hunt, M., Marquet, W., Moller, D., Peal, K., Smith, W., and Spindel, R. (1974). An acoustic navigation system, technical report WHOI-74-6. Technical report, Woods Hole Oceanographic Institution.

Leonard, J. and Newman, P. (2003). Consistent, convergent, and constant-time SLAM. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, 1143–1150. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Leonard, J.J. and Feder, H.J.S. (2000). A computationally efficient method for large-scale concurrent mapping and localization. In *Proceedings of the Ninth International Symposium on Robotics Research*, 169–176. Springer-Verlag.

Neira, J. and Tardós, J. (2001). Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6), 890 – 897.

Paz, L.M., Tardós, J., and Neira, J. (2008). Divide and conquer: EKF SLAM in O(n). *IEEE Transactions on Robotics*, 24(5), 1107–1120.

Piniés, P. and Tardós, J. (2008). Large scale SLAM building conditionally independent local maps: Application to monocular vision. *IEEE Transactions on Robotics*, 24(5), 1094–1106.

Ribas, D. (2008). *Underwater SLAM for Structured Environment Using an Imaging Sonar*. Ph.D. thesis, University of Girona.

Roman, C.N. and Singh, H. (2007). A self-consistent bathymetric mapping algorithm. *Journal of Field Robotics*, 24(1–2), 23–50.

Smith, R., Self, M., and Cheeseman, P. (1988). A stochastic map for uncertain spatial relationships. In *Proceedings of the 4th International Symposium on Robotics Research*, 467–474. MIT Press, Cambridge, MA, USA.

Tardós, J.D., Neira, J., Newman, P.M., and Leonard, J.J. (2002). Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotics Research*, 21(4), 311–330.

Whitcomb, L., Yoerger, D., and Singh, H. (1999). Combined doppler/LBL based navigation of underwater vehicles. In Durham (ed.), *International Symposium on Unmanned Untethered Submersible Technology*.

Williams, S.B. (2001). *Efficient Solutions to Auntonomous Mapping and Navigation Problems*. Ph.D. thesis, Australian Center for Field Robotics, University of Sydney.

Williams, S.B., Dissanayake, G., and Durrant-whyte, H. (2002). An efficient approach to the simultaneous localisation and mapping problem. In *Proceedings IEEE International Conference Robotics and Automation*, volume 1, 406–411.

Appendix A. 6-DOF TRANSFORMATIONS

Following the coordinate frame notation introduced in (Smith et al. (1988)) we define a 6-DOF pose of frame j with respect to frame i as in (A.1).

$$\mathbf{x}_{ij} = [x_{ij},\ y_{ij},\ z_{ij},\ \phi_{ij},\ \theta_{ij},\ \psi_{ij}]^T \tag{A.1}$$

In this notation, $x_{ij},\ y_{ij},\ z_{ij}$ are the 3-vector translation form i to j expressed in frame i, and $\phi_{ij},\ \theta_{ij},\ \psi_{ij}$ is a 3-vector of xyz-convention roll, pitch and yaw Euler angles. From these Euler angles the 3×3 orthonormal rotation matrix that rotates frame j into frame i is defined as in (A.2).

$${}^i_j\mathbf{R} = rotz(\psi)^T roty(\theta)^T rotz(\phi)^T \tag{A.2}$$

Given the orthonormal rotation (A.2) together with the translation, we can obtain the homogeneous 4×4 corresponding transformations matrices ${}^i_j\mathcal{T}$ and ${}^j_i\mathcal{T}$ as in (A.3).

$${}^i_j\mathcal{T} = \begin{bmatrix} {}^i_j\mathbf{R} & {}^i_j\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \qquad {}^j_i\mathcal{T} = \begin{bmatrix} {}^i_j\mathbf{R}^T & -{}^i_j\mathbf{R}^T {}^i_j\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{A.3}$$

Given the pose vector $\mathbf{x}_{ij}$ and $\mathbf{x}_{jk}$, the frame k can be expressed *w.r.t.* frame i according to (A.4). The associated Jacobian to the composition of transformations is given by (A.5). The Jacobians are necessary to compute covariance estimates.

$$\begin{aligned} {}^i_k\mathcal{T} &= {}^i_j\mathcal{T}\ {}^j_k\mathcal{T} \\ \mathbf{x}_{ik} &= \mathbf{x}_{ij} \oplus \mathbf{x}_{jk} = [x_{ik},\ y_{ik},\ x_{ik},\ \phi_{ik},\ \theta_{ik},\ \psi_{ik}]^T \end{aligned} \tag{A.4}$$

$$J_\oplus = \frac{\partial \mathbf{x}_{ik}}{\partial(\mathbf{x}_{ij}, \mathbf{x}_{jk})} = [J_{\oplus 1}\ J_{\oplus 2}] = [\frac{\partial \mathbf{x}_{ik}}{\partial \mathbf{x}_{ij}}\ \frac{\partial \mathbf{x}_{ik}}{\partial \mathbf{x}_{jk}}] \tag{A.5}$$

The notation $\oplus$ and $\ominus$ is used to denote the composition and inversion of transformations described by Smith et al. (1988). In order to reverse a coordinate frame, the inverse transformation (A.6) together with its associated Jacobian (A.7) is needed. This inverse transformation provides the coordinate frame i with respect to frame j (i.e. $\mathbf{x}_{ji}$), given the pose $\mathbf{x}_{ij}$.

$$\begin{aligned} {}^j_i\mathcal{T} &= {}^i_j\mathcal{T}^{-1} \\ \mathbf{x}_{ji} &= \ominus\mathbf{x}_{ij} = [x_{ji},\ y_{ji},\ x_{ji},\ \phi_{ji},\ \theta_{ji},\ \psi_{ji}]^T \end{aligned} \tag{A.6}$$

$$J_\ominus = \frac{\partial \mathbf{x}_{ji}}{\partial \mathbf{x}_{ij}} \tag{A.7}$$

Finally, given the poses $\mathbf{x}_{ij}$ and $\mathbf{x}_{ik}$ it is possible to obtain the relative pose $\mathbf{x}_{jk}$ by combining the two previous operations, as expressed in (A.8) together with its associated Jacobian in (A.9).

$$\begin{aligned} {}^j_k\mathcal{T} &= {}^j_i\mathcal{T}\ {}^i_k\mathcal{T} = {}^i_j\mathcal{T}^{-1}\ {}^i_k\mathcal{T} \\ \mathbf{x}_{jk} &= \mathbf{x}_{ji} \oplus \mathbf{x}_{ik} = \ominus\mathbf{x}_{ij} \oplus \mathbf{x}_{ik} = \\ &= [x_{jk},\ y_{jk},\ x_{jk},\ \phi_{jk},\ \theta_{jk},\ \psi_{jk}]^T \end{aligned} \tag{A.8}$$

$$J_{\ominus\oplus} = \frac{\partial \mathbf{x}_{jk}}{\partial(\mathbf{x}_{ij}, \mathbf{x}_{ik})} = [J_{\oplus 1} J_\ominus \quad J_{\oplus 2}] \tag{A.9}$$