

Bathymetric SLAM with No Map Overlap using Gaussian Processes

Stephen Barkby, Stefan B. Williams, Oscar Pizarro, Michael V. Jakuba

Australian Centre for Field Robotics

School of Aerospace Mechanical and Mechatronic Engineering

University of Sydney, Sydney, NSW, 2006, Australia

s.barkby@acfr.usyd.edu.au

Abstract—This paper presents an efficient and featureless approach to Bathymetric Simultaneous Localization And Mapping (SLAM) that utilizes a Rao-Blackwellized Particle Filter (RBPF) and Gaussian Process (GP) Regression to provide loop closures in areas where little to no overlap with previously explored terrain is present. To significantly reduce the memory requirements of this approach (thereby allowing for the processing of large datasets) a novel map representation is also introduced that, instead of directly storing estimates of seabed depth, records the trajectory of each particle and synchronizes them to a common log of bathymetric observations. Upon detecting a loop closure each particle is then weighted by matching new observations to the current predictions generated from a local reconstruction of their map using GP Regression. Here the spatial correlation in the environment is fully exploited, allowing predictions of seabed depth to be generated in areas that may not have been directly observed previously. The particle resampling that is performed therefore not only enforces self-consistency in overlapping sections of the map but additionally enforces self-consistency between neighboring map borders. The results demonstrate how observations of seafloor structure with partial overlap can be used by bathymetric SLAM to improve map self consistency when compared to Dead Reckoning fused with Long-Baseline observations. In addition we show how mapping corrections can still be achieved even when no map overlap is present.

I. INTRODUCTION

The use of Autonomous Underwater Vehicles (AUVs) as mapping platforms is becoming increasingly prevalent in research and industry, due to their advantages over more traditional shipborne and towed sensor systems [15]. Of note is their ability to operate away from the oceans's surface, allowing AUVs to maintain close proximity to the seafloor regardless of depth [9]. With this capability high resolution bathymetry can therefore be obtained using a multibeam depth profiler on an AUV, creating bathymetric maps that are an invaluable resource in marine applications such as geological surveying, pipeline surveys and prospecting/salvage missions [4][3]. However the mapping efforts of AUVs are hindered by the navigation error that accumulates in the vehicle's location estimate, continuing to increase until the AUV obtains a position fix.

GPS observations provide a method of bounding this position error, although they require the AUV to surface. Alternatively an Ultra Short Baseline (USBL) acoustic positioning system can be used to provide range and bearing measurements between the AUV and a support vessel. By using the GPS/Inertial on the ship these observations can

be georeferenced to provide an observation of the vehicle's position while it is underway. However, such a setup provides position estimates less accurate than an equivalent GPS fix and requires the support ship to actively maintain the AUV within the USBL's range for the duration of the mission. This problem is further compounded in deep water deployments where the distances between the ship and the vehicle are large. Long Baseline (LBL) acoustic positioning also provides navigation estimates with bounded error, but this requires additional infrastructure to be in place. Furthermore observations from LBL transponder nets are subject to a trade off between coverage and accuracy and may suffer from multipath and/or occlusions when deployed on seafloor with complex structure [8].

Alternatively Bathymetric distributed Particle SLAM (BP-SLAM) can be used to reduce errors in navigation and mapping [2]. This is accomplished using a Rao-Blackwellized Particle Filter (RBPF) to track the uncertainty in the vehicle states, with each particle maintaining a novel grid-based 2D depth map. The consistency between new observations and predictions from each particle's map are then used to weight the particle set during resampling, forming a method of SLAM that does not require features to be explicitly estimated and tracked.

In this paper we again use the BPSLAM filter but propose a novel map representation that lends itself better to larger scale mapping efforts by reducing the memory requirements needed to store each particle map. The new map structure is based on the observation that each particle map is generated from an identical log of bathymetric observations; it is only the underlying particle trajectories that are different. A considerable amount of memory can therefore be saved if, instead of storing each particle's point cloud or grid map, we store each particle's trajectory and simply link each pose contained within it to a corresponding entry in a log of observations shared by all particles. Technically this in itself is not a true "map" representation but rather an efficient manner in which the raw components to build a particle's map can be stored and accessed without loss of information. Gaussian Process (GP) Regression can then be used to extract depth predictions from each particle's map, even in places that have not previously been explored.

The remainder of this paper is organized as follows. Section II begins by outlining related work. Section III then details the BPSLAM algorithm with the new trajectory

map representation while Section IV presents the results of applying our algorithm to a real, large-scale bathymetric survey. Section V then summarizes our findings.

II. RELATED WORK

One approach to featureless bathymetric SLAM has been reported by Roman [13]. Here a point cloud map model is used to divide a temporal sequence of bathymetry into submaps that are assumed to be error free. Pairwise matching of overlapping submaps constrains the vehicle trajectory and submap origins using a delayed state Kalman filter. This technique has been shown to produce more accurate maps than Dead Reckoning (DR) navigation alone or LBL filtered navigation. However it cannot address errors within the individual submaps that link together to form the overall map. In addition, the tradeoffs in complexity, accuracy and matching performance based on the size and number of submaps are only partially understood.

Alternatively another approach reported by Fairfield [7] uses a Rao-Blackwellized particle filter combined with an occupancy grid-based volumetric map representation, efficiently managed with Deferred Reference Counting Octrees. This method has proven to be successful in generating a consistent 3D bathymetric map in a closed cave environment in real time, where the benefit of continuous localization via measurement of the AUV's proximity with the surrounding cave walls, using both fore and aft mounted sonar arrays, is fully utilized. This approach has also been shown to perform well in an open marine environment when equipped with a single multibeam sonar and a large scale high resolution prior map [8]. However in this case localisation was only performed using the prior map. For bathymetric mapping missions where no prior map information is available opportunities to localize are typically not as abundant as in these two trials.

Our approach also uses a Rao-Blackwellized particle filter to account for the uncertainty in the vehicle's navigation solution and its effect on map-making. However our technique adapts particle filter SLAM to the seafloor in general and utilizes a novel trajectory map representation and GP Regression that can achieve navigation and mapping corrections even with little to no map overlap. These are the principal contributions of this paper.

III. THE BPSLAM FILTER

BPSLAM uses a Rao-Blackwellized particle filter as its foundation, where particle weighting/resampling, based on the self consistency of a map generated by each particle, successively acts to converge the filter down onto the navigation solution that creates the most self-consistent map. The specific structure and filter procedures used by BPSLAM are now described.

A. Particle State Structure

To represent the vehicle state we adopt a similar approach to that advocated by Fairfield [7], in which states that are directly observable using the vehicle's sensors are removed

from the particle filter and are instead tracked with a single EKF that is shared by all particles:

$$\mathbf{x}_v(t_k) = [z_v(t_k), \psi_v(t_k), {}^v\mathbf{v}_v(t_k), \dot{\psi}_v(t_k)] \quad (1)$$

where these states correspond to the vehicle depth, attitude, body velocity and Euler angle rotation rates respectively. This allows the particle filter to focus on tracking the x, y position of the vehicle, leading us to define the particle set $\mathbf{S}(t_k)$ with N particles:

$$\mathbf{S}(t_k) = \begin{pmatrix} x_1(t_k) & \dots & x_N(t_k) \\ y_1(t_k) & \dots & y_N(t_k) \\ p_{ID_1} & \dots & p_{ID_N} \end{pmatrix} \quad (2)$$

where $[x_i(t_k), y_i(t_k)]$ is the hypothesized vehicle position for the i th particle stored in the set and p_{ID_i} is the particle's identification number.

At the beginning of the mission each particle is initialized with an estimate of the vehicle state. This can be relative to a local or georeferenced frame, depending on whether a form of absolute positioning is available e.g. GPS. A simple constant velocity model is then used along with the standard EKF equations to predict and update $\mathbf{x}_v(t_k)$ when new navigation is received, or to predict to the time of a new multibeam observation. The particle set is also predicted forward, except in this case the uncertainty associated with the prediction step is encoded into $\mathbf{S}(t_k)$ by each particle randomly sampling a hypothesis from the distribution of $\mathbf{x}_v(t_{k-1})$ on which to base its prediction.

B. Particle Map Structure

The trajectory map representation stores each particle's map as a trajectory linked to a list of bathymetric observations that is shared by all particles. To reduce the computational expense of copying maps during the particle resampling procedure we additionally utilize Distributed Particle Mapping (DPM) [5][6]. The key idea behind DPM is to retain the original particle's map and have any new particles (*children*) that are resampled from the original particle (*parent*) point to the parent's map rather than copy the map themselves. Extracting a particle's map is then achieved by examining its estimates as well as those inherited by it through the *particle ancestry*. For the trajectory map this corresponds to storing each particle's trajectory as a concatenation of trajectory segments, one that belongs to itself and the rest to its ancestors. This requires that the lineage of each particle be stored in an *ancestry tree*, structured in the form of a list, where each particle (both current and ancestral) has a record in this list that contains the following information:

- **Parent ID** - The ID of the parent that the particle was resampled from.
- **Child List** - A list of all the particle's children.
- **Trajectory** - A list of poses which forms the trajectory branch of this particle.
- **Observations** - A list of indexes (synchronized to the trajectory) that point to sonar swaths in the multibeam log.

The ID of the particle is not stored as a field in this record but is encoded as the position of the record in the list itself, allowing for fast access. A particle's trajectory can therefore be reconstructed at any time by backtracing through its lineage. A particle's map is similarly reconstructed by backtracing through a particle's observations and then using the trajectory (which the observations are synchronized to) to transform them into the global frame. This is demonstrated in Figure 1.

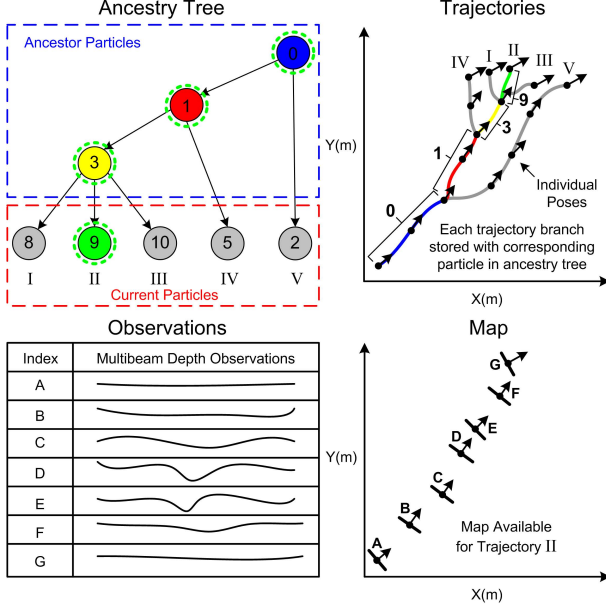


Fig. 1. An example of the trajectory map structure with $N=5$ particles and 7 observations (A-G). Trajectory II is reconstructed by backtracing through the ancestry of the particle stored at this index and connecting together each trajectory segment owned by the particle and its ancestors (shown by the broken circles). This trajectory is then used to transform a list of observations, synchronized to the poses contained within it, into the global frame to create the particle's map.

This structure makes it relatively straightforward to also update each particle's map once particle weighting/resampling has been completed. All that is required is to simply add the particle's current pose and swath index to the back of the trajectory and observation list respectively.

C. GP model

In BPSLAM each particle's map is progressively generated, meaning that the GP model describing the spatial correlation between points in the map (which is completely defined by its mean function $m(X)$ and covariance function $k(X, X')$) must be periodically relearned as new bathymetry is received. Furthermore, each particle will need to perform a separate regression based on its own map during particle weighting. For these reasons we use a simple stationary covariance function developed by Melkumyan [11] to keep this approach computationally tractable. This is given below:

$$k(X, X'; l, \sigma_0) = \begin{cases} \sigma_0 \left[\frac{2 + \cos(2\pi \frac{d}{l})}{3} \left(1 - \frac{d}{l}\right) + \frac{1}{2\pi} \sin(2\pi \frac{d}{l}) \right] & \text{if } d < l \\ 0 & \text{if } d \geq l \end{cases} \quad (3)$$

where d is the xy distance between depth estimates ($d = |X - X'|$). The hyperparameters for this model are periodically relearned by optimizing on the log marginal likelihood [12] using training data extracted from a square patch of the most recent bathymetry (not including the current swath), georeferenced using the Dead Reckoning trajectory¹. The mean function that we use is simply the mean depth of the particle map (i.e. $m(X) = \sum_{i=1}^M \frac{z_i}{M}$).

D. Particle Weighting and Resampling

The particle weighting stage judges each particle by the likelihood that the current range observations $\mathbf{z}(t_k) = \{r_b(t_k)\}_{b=1}^B$ from the multibeam swath would be observed, given the particle's current map (i.e. $w_i = p(\mathbf{z}(t_k) | \mathbf{S}_i(t_0 : t_{k-1}), \mathbf{z}(t_0 : t_{k-1}))$). Once calculated these particle weights are used to randomly resample $\mathbf{S}(t_k)$, where the probability of being resampled is directly proportional to the normalized weights. This allows the particles that are most likely to represent the correct state hypothesis to propagate while removing those which are unlikely, successively converging the particle set towards the true trajectory and map. However particle resampling is only carried out when the *effective particle size*, which is a measure of weight variation, is less than half the number of particles [10]. This is done to prevent removing particles from $\mathbf{S}(t_k)$ prematurely.

The calculation of these particle weights involves four main steps, detailed as follows:

1) *Transforming Range Observations into Global Depth Observations*: The range observations r , modeled with range dependent uncertainty ($\sigma_r^2 = \lambda r^2$, λ constant), are referenced into the global frame F_g using the vehicle and multibeam geometry defined in Figure 2. This converts r into the corresponding depth observation $E_{z,obs}$. Mapping the corresponding uncertainty in range to an equivalent uncertainty in depth ($\sigma_{E_{z,obs}}^2$) is however less straightforward. Fortunately this can be achieved by exploiting the angular correlation between observations in the swath to infer the uncertainty along each observation's depth axis. As such each particle's map, which is a collection of previous depth observations, forms a 2.5D point cloud once reconstructed in the same manner.

2) *Loop Closure Decision*: Loop closures (i.e. particle weighting/resampling) can only be attempted when enough particles have map predictions that exist at the x, y locations of the current depth observations. Naively this would mean that loop closures could be performed at any time during the mission, as GP Regression allows for the calculation of depth predictions anywhere in the map. However, regression far away from a particle map's point cloud will not be useful as all predictions converge toward the mean function at this limit. The rate at which this occurs decreases as the length scale hyperparameter l of the GP model increases, meaning that extrapolated predictions can be performed further away

¹Each particle uses the same GP model, learnt from the DR trajectory that is assumed to be locally consistent.

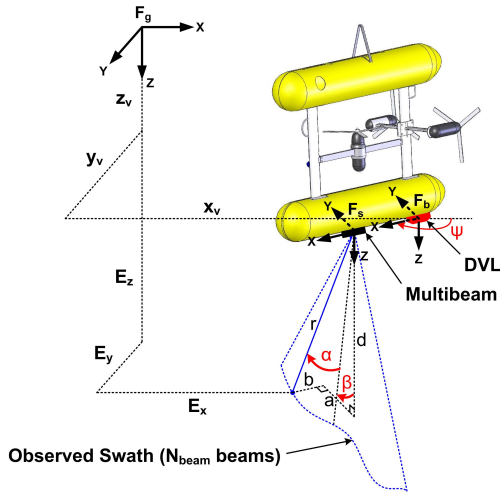


Fig. 2. Relationship between a multibeam observation of range (r), bearing (α), along track angle (β), the state hypothesis ($\bar{\mathbf{x}}_v(t_k)$) and the location of the seabed patch observed ($\mathbf{E}$). b , a and d are the along track, across track and depth of the observation respectively. The location of the Doppler Velocity Log sensor defines the vehicle's coordinate frame, shown here by an arbitrary AUV.

from the training set when the surrounding terrain is more spatially correlated. However performing particle weighting using observations of highly correlated terrain, such as a flat plane, is less likely to discern the particles with the most likely state hypotheses. A loop closure is therefore not attempted if the current length scale is above the user defined threshold l_{max} . For length scales below this threshold particles are only weighted and included in the resampling phase if they possess at least B_{min} observations that fall within $\tau_{lc}l$ distance of each of their respective maps, where τ_{lc} is a scale factor. This effectively allows us to dynamically increase or decrease the acceptable range of extrapolation based on the correlation in the local seabed.

3) *GP Regression*: To attempt a loop closure all particles included in the resampling phase must match their current depth observations (only those within map proximity) to depth predictions extracted from their map. These are calculated using the GP model and a local reconstruction of the particle's map, which is a 2.5D point cloud of the form $\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\}$ where $\mathbf{X} = \{x_i, y_i\}_{i=1}^M$ and $\mathbf{Y} = \{z_i\}_{i=1}^M$ with noise covariance matrix $\mathbf{W} = \text{diag}(\sigma_{E_{z1}}^2 \dots \sigma_{E_{zM}}^2)$, for a set containing M points. A prediction of seabed depth ($E_{z,est}$) at each location ($E_{x,obs}, E_{y,obs}$) can therefore be extracted from the point cloud by performing GP Regression using $\mathcal{D}$ as training data. For the exact regression equations Rasmussen [12] provides an extensive review.

4) *Particle Weighting Formula*: Once regression is complete a weighting can be calculated for these observation/prediction pairs based on their difference. As both are modeled as Gaussian the probability distribution of their difference is also Gaussian. The likelihood that this difference is zero therefore provides a principled method by which the

particle that made the observation can be weighted.

$$w = p((E_{z,est} - E_{z,obs}) = 0) = \frac{e^{-\frac{1}{2} \frac{(\mu_{E_{z,est}} - \mu_{E_{z,obs}})^2}{\sigma_{E_{z,obs}}^2 + \sigma_{E_{z,est}}^2}}}{\sqrt{2\pi(\sigma_{E_{z,obs}}^2 + \sigma_{E_{z,est}}^2)}} \quad (4)$$

Each particle included in the resampling phase will have a different number of observations that have been successfully assigned likelihood measures, the minimum number in this case tracked by the variable L_{min} . A single weighting factor can therefore be generated for each particle by calculating the joint likelihood of L_{min} (i.e. the product of its likelihood measures).

An example of the new map weighting procedure is shown in Figure 3(a). Here previous particle resampling has caused the set to collapse down onto a single past trajectory belonging to the root ancestor Particle 0. Upon receiving the new multibeam swath a loop closure is detected, as the particles have observations that fall within proximity to their maps, shown in Figure 3(b). Each particle then uses a local patch of their own map to generate a prediction (even when no overlap exists) and compares it against the observation. The joint likelihood of these observation/estimate pairs is then calculated. Note that the joint likelihood is a relative measure that only holds meaning if it is generated from the same number of observations for each particle. As such only two observations are used, as increasing this number would require some particles to extrapolate predictions beyond the window where they remain accurate enough for use.

IV. RESULTS

We tested our algorithm on a real mission scenario using bathymetric and navigation logs from a survey undertaken by the AUV *Sentry*, a vehicle owned and operated by the Woods Hole Oceanographic Institution (WHOI) [1].

The survey is of the Haakon Mosby Mud Volcano, located 1250 m deep south of Svalbard. Figure 4 presents a bathymetric map of this region generated from an older survey, along with the new survey pattern that was carried out. To achieve full coverage of this area the tracklines navigated for this mission are a standard “lawnmower” pattern carried out underwater at 20 m altitude to produce swath widths of ~ 60 m. This survey took 10.3 hours to complete and produced on average 11.5 m of overlap between multibeam swaths from neighboring tracklines, which corresponds to $\sim 122\%$ coverage.

The navigation suite installed on *Sentry* comprises a three axis Doppler Velocity Log (DVL) ($\pm 3 \text{ mms}^{-1}$ precision), a surface relative depth sensor and a PHINS Ring Laser Gyro (RLG) that delivers pitch, roll and heading observations with accuracies on the order of 0.01° . Acoustic LBL navigation fixes were also available. These fixes delivered position estimates accurate to ~ 1 m and are only used by the BPSLAM filter to initialize the AUV's position and for measurement updates in an EKF filter separate to BPSLAM, so as to produce a navigation solution by which to compare the performance of the BPSLAM filter. The bathymetric

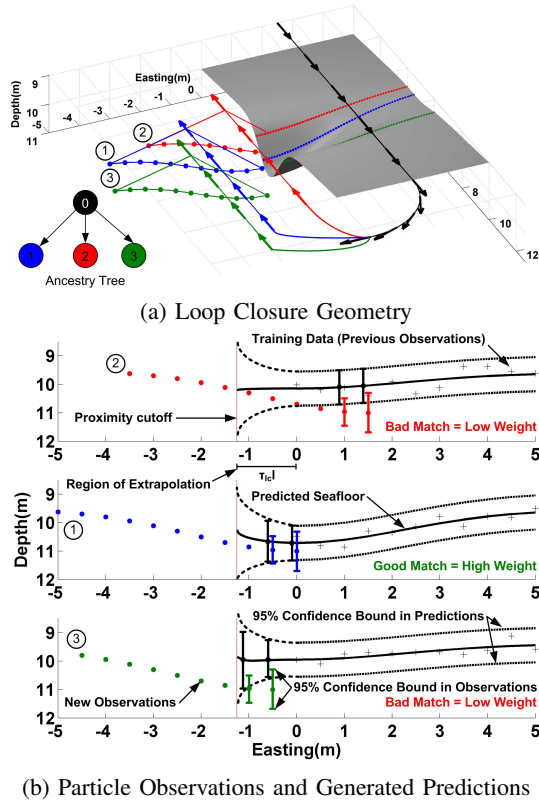


Fig. 3. An example of the GP method used to weight each particle given a set of $N = 3$ with a swath of 11 multibeam observations. In part a) the trajectories for Particle's 1, 2, 3 are shown in blue, red, green respectively. The trajectory of ancestor Particle 0, which they all inherit, is shown in black. The local map section which each particle uses to match its observations against is shown in grey. This map section is different for each particle if it is generated from a trajectory section in the particles ancestry that is not shared. However, in this case the neighboring trajectory within proximity belongs to Particle 0, so the corresponding map is common to all (useful for illustrative purposes). The colored broken lines indicate the slices of the predicted map that are within the plane of each particle's observations, displayed in Part b). Here GP regression is used by each particle to generate predictions to compare against their observations. For illustrative purposes the GP regression is only shown in 2D, where in practice the full 3D map is utilized to generate these predictions. Equation 4 is then used to weight the particles based on these observation/estimate pairs.

sensor used by *Sentry* is a Reson 7125 400 kHz multibeam sonar, providing 480 beams uniformly across 120° .

A. Navigation and Mapping Comparison

For this trial an Intel Xeon 3.00GHz CPU was used to run BPSLAM. As this survey retains consistent overlap (albeit partially) with previously explored terrain the scaling factor τ_{lc} of the trajectory map is set to zero for this run, meaning that particle weighting based on extrapolated predictions will not be attempted. Based on the precision of the navigation estimates available and the duration of the mission we also set the number of particles in the filter to $N = 160$.

Using these settings BPSLAM produced a trajectory that slightly contracted the trackline spacings of the mission when compared to the Dead Reckoning (DR) solution. This large scale correction is also similar to that seen in the LBL fused DR solution, suggesting that BPSLAM's navigation solution is more accurate than that produced by DR. At the end of the

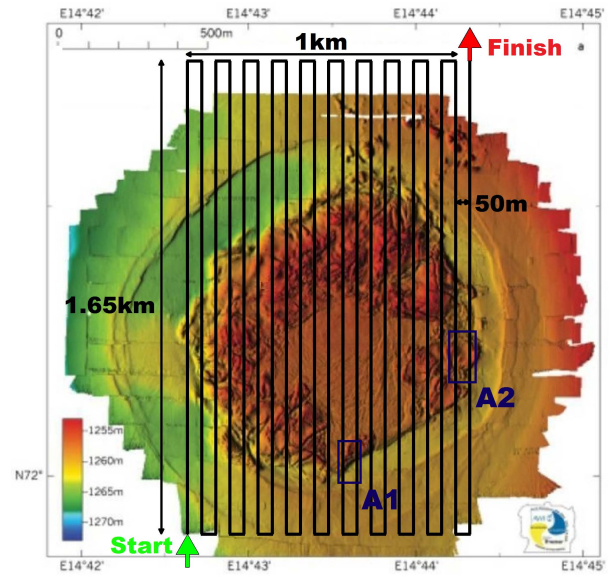


Fig. 4. Previous bathymetric survey of the Haakon Mosby Mud Volcano taken by the ROV Victor 6000. The new survey pattern is indicated by the black line, designed to provide 122% coverage. The small boxes indicate regions where the largest map misregistrations were observed. These are more closely investigated in Figure 6. (Image Source: www.uib.no/geobio)

mission BPSLAM and LBL fused DR provided final position estimates ~ 14.5 m SW of the DR solution. Given that the AUV has traveled ~ 36 km over a period of 10.3 hours this error is relatively small, demonstrating an exceptional performance by the DVL/RLG navigation suite.

Despite this, the accumulation of this small error in DR navigation over time is enough to produce significant error in the corresponding map. Unfortunately, *Sentry*'s bathymetric map and its geological interpretation has yet to be published in a scientific journal. Because of this, we cannot show the full AUV-based map. However the registration errors detected in the full map for each case can be presented, which are determined using a method described by Roman[14] that involves splitting the bathymetry into a sequence of submaps and then calculating the maximum registration error between submap portions [2]. In addition we also provide zoomed in bathymetric maps of two regions where the misregistration error in the maps was largest. These are shown in Figures 5 and 6 respectively.

From Figure 5(a) it can be seen that the use of DR as a navigation solution produces the map with the most artifacts. The inclusion of LBL observations into the navigation solution resolves many of these artefacts, shown in Figure 5(b) by the reduction of the median and 99th percentile registration error. However this has also introduced artefacts in places where there were none before, as evident by the increase in the tail of this histogram. This is also apparent in Figures 6(b) and 6(e) that highlight two main regions in the map where the inclusion of LBL observations has resulted in misregistration of the sonar swaths.

Figure 5(c) demonstrates the performance of the BPSLAM filter when compared to LBL aided navigation. All of the

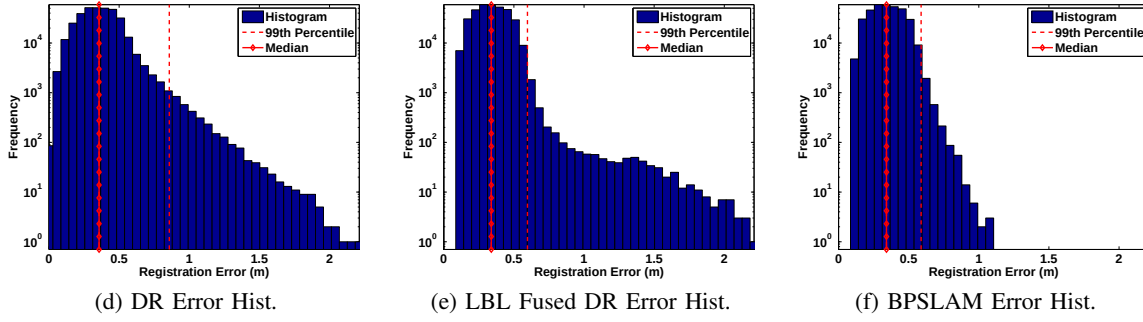


Fig. 5. Registration error histograms generated using three different navigation solutions, detailing the misregistration between overlapping swaths. Comparison shows the LBL fused DR and BPSLAM filters providing a significant reduction in the mapping error when compared to the map produced by DR.

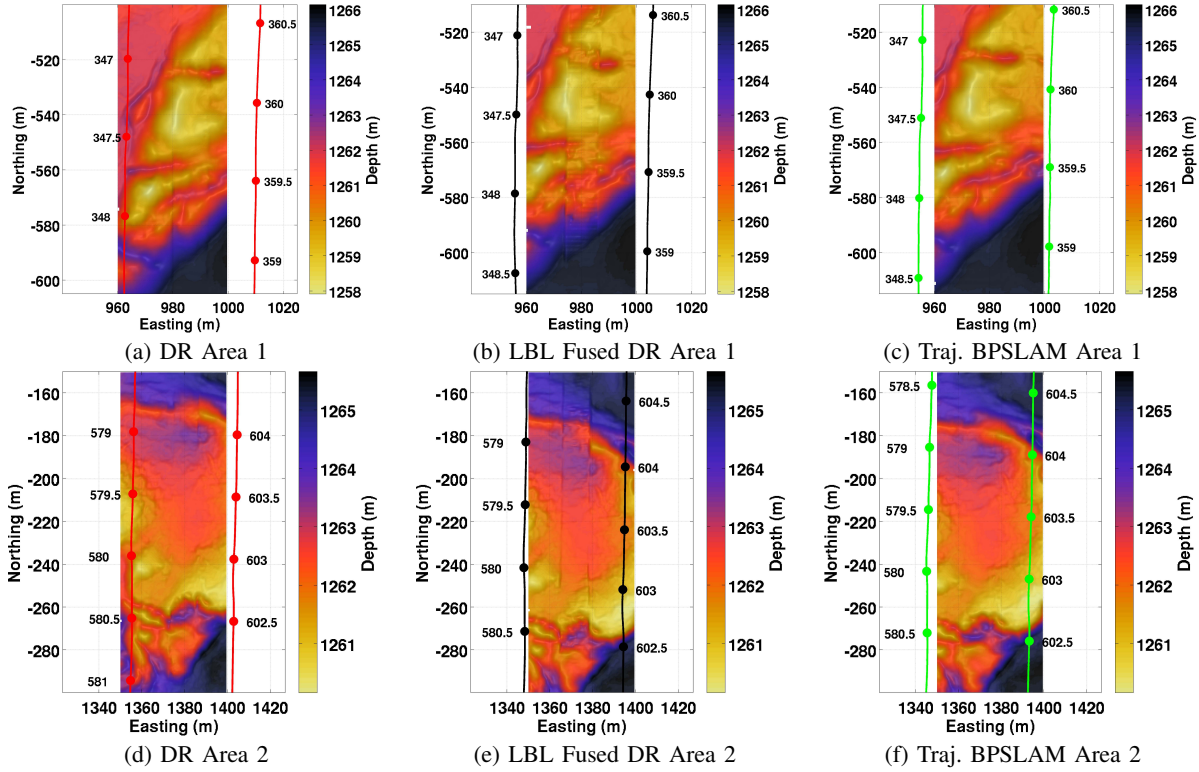


Fig. 6. Zoomed in bathymetric maps generated using three different navigation solutions. The section of the AUV trajectory which creates each map is also shown and timestamped (by minutes from mission start) for reference. Comparison shows a visible misalignment in the maps produced by DR and LBL Fused DR. The map produced by BPSLAM however successfully resolves these errors.

prominent artifacts distorting the non SLAM approaches have been successfully resolved, evident by the reduction in the tail of the histogram when compared to the two non-SLAM approaches.

A close examination of Figure 6 demonstrates how BPSLAM achieved this correction. In Figure 6(c) BPSLAM has identified a navigation solution that, compared to the DR solution in Figure 6(a), shifts its East trackline 2.28 m southwards relative to its West trackline, creating the correct match between neighboring swaths. In comparison the LBL fused DR solution has shifted its East trackline 5.45 m southwards relative to its West trackline. As can be seen in Figure 6(b) this in turn creates a more pronounced registration error than originally seen in the DR solution.

These artefacts in the LBL solution may be owing to the difficult acoustic environment present around the main slope of the volcano. BPSLAM, on the other hand, is able to use the structure of the terrain to correctly align the neighbouring swaths. A similar effect is also seen in Figures 6(d)-(f).

Lastly, for these trials the BPSLAM filter required 2.1 GB of RAM which, given the size of this dataset, demonstrates an exceptional performance in memory usage. It also took 81 minutes to process the mission, which is 13.5 % of the mission time.

B. Map Overlap Comparison

The preceding section has demonstrated how BPSLAM with trajectory maps can perform corrections to navigation and mapping in large scale missions when provided with

only ~ 11.5 m of overlap between multibeam swaths from neighboring tracklines, corresponding to $\sim 122\%$ coverage. What remains to be seen is how the BPSLAM filter performs if the amount of overlap is progressively reduced.

To investigate this behavior the BPSLAM filter was run repeatedly in batches of 25 for a particle set size of $N = 160$. For each batch the maximum width of the multibeam swath was trimmed to a different value, corresponding to 11.5 m, 4.0 m and 0.0 m of swath overlap between neighboring tracklines. Lastly, as an exploratory measure the swaths were trimmed even further so that an average *separation* of 1.0 m between swaths (i.e. -1 m overlap) was achieved. For these trials the scaling factor τ_{lc} was set to 0.3. This corresponded to an average proximity of 2.1 m that had to be achieved between observation and map before a depth prediction was allowed. Upon finishing the average registration error in the map produced by BPSLAM was then calculated for each run. So as not to bias the runs with less overlap, the full swath is used to reconstruct the maps at the end of all runs based on the respective navigation solution generated by BPSLAM, DR and LBL fused DR navigation.

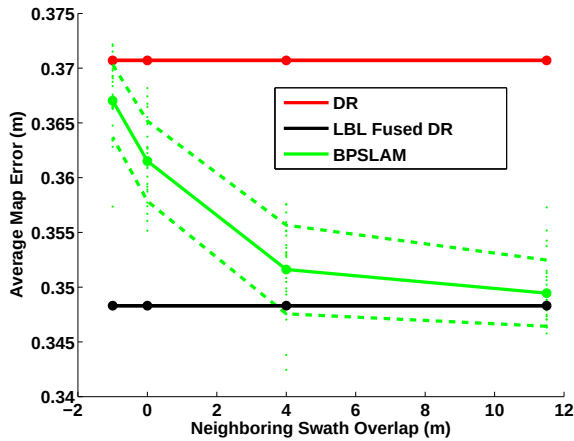


Fig. 7. Variation in the map error as the amount of swath overlap between neighboring tracklines is reduced. The large dots represent the mean error used in each batch of 25 runs (shown by the smaller dots). The outer broken lines represent the $\pm 2\sigma$ bound in each batch.

Figure 7 presents the results of this investigation. For 11.5 m overlap between swaths the map self-consistency produced by BPSLAM rivals and sometimes exceeds that produced by LBL fused DR. As the amount of swath overlap is reduced the BPSLAM mapping solution starts to converge towards the map quality produced by DR. However, as shown the BPSLAM filter is still capable of producing mapping corrections even when no overlap between neighboring swaths exists, consistently out performing Dead Reckoning in this case.

V. CONCLUSIONS

A new trajectory map representation for BPSLAM was proposed. In this framework a significant amount of memory was saved by storing each particle map as a trajectory linked to a bathymetric log of raw observations that is shared by

all particles. Results verified this efficient memory usage, allowing for the processing of this large dataset. In addition the new trajectory map coupled with GP Regression allowed BPSLAM to provide navigation and mapping corrections when minimal overlap was available in the bathymetry, even maintaining these corrections when overlap in the dataset was removed entirely.

ACKNOWLEDGEMENTS

This research was undertaken with funding support from the Australian Research Council and Integrated Marine Observing System (IMOS) facility. The authors would also like to acknowledge Dana Yoerger for generously supplying the Haakon Mosby dataset. Special thanks must also go to Ian Mahon, whose technical support and software modules were instrumental in BPSLAM's implementation. Finally we thank all those who have helped contribute to the development of our GP model including Asher Bender, Fabio Ramos, Arman Melkumyan, Simon O'Callaghan and Alistair Reid.

REFERENCES

- [1] <http://www.whoi.edu/page.do?pid=38095>, March 2011.
- [2] S.A. Barkby, S.B. Williams, O. Pizarro, and M. Jakuba. A Featureless Approach to Efficient Bathymetric SLAM using Distributed Particle Mapping. *Journal of Field Robotics*, 28(1):19–39, Jan. 2011.
- [3] S. Blasco. Application of Multibeam Surveying to Resource Mapping. *FIG XXII International Congress*, TS4.3 Hydrographic Surveying I, April 2002.
- [4] P. Chapman, D. Wills, G. Brookes, and P. Stevens. Visualizing Underwater Environments Using Multifrequency Sonar. *IEEE Computer Graphics and Applications*, 19(5):61–65, Sep. 1999.
- [5] A.I. Eliazar and R. Parr. DP-SLAM 2.0. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '04)*, 2:1314–1320, April 2004.
- [6] A.I. Eliazar and R. Parr. Hierarchical Linear/Constant Time SLAM Using Particle Filters for Dense Maps. *Neural Information Processing Systems Conference*, Dec. 2005.
- [7] N. Fairfield, G.A. Kantor, and D. Wettergreen. Towards particle filter SLAM with three dimensional evidence grids in a flooded subterranean environment. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '06)*, pages 3575–3580, Orlando, FL, May 2006.
- [8] N. Fairfield and D. Wettergreen. Active localization on the ocean floor with multibeam sonar. In *Proceedings of the OCEANS '08 Conference*, Quebec City, Quebec, Sep. 2008.
- [9] W.J. Kirkwood. Development of the dorado mapping vehicle for multibeam, subbottom, and sidescan science missions. *Journal of Field Robotics*, 24(6):487–495, June 2007.
- [10] J.S. Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119, 1996.
- [11] A. Melkumyan and F. Ramos. A Sparse Covariance Function for Exact Gaussian Process Inference in Large Datasets. *International Joint Conferences on Artificial Intelligence (IJCAI '09)*, June 2009.
- [12] C.A. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, MIT, MA, 2006.
- [13] C. Roman and H. Singh. Improved vehicle based multibeam bathymetry using sub-maps and SLAM. In *Proceedings of the Intelligent Robots and Systems (IROS '05)*, pages 3662–3669, Edmonton, Canada, Aug. 2005.
- [14] C. Roman and H. Singh. Consistency based error evaluation for deep sea bathymetric mapping with robotic vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '06)*, pages 3568–3574, Orlando, FL, May 2006.
- [15] H. Singh, L. Whitcomb, D. Yoerger, and O. Pizarro. Microbathymetric Mapping from Underwater Vehicles in the Deep Ocean. *Computer Vision and Image Understanding*, 79:143–161, 2000.