A SLAM-based Approach for Underwater Mapping using AUVs with Poor Inertial Information

Marcus Hammond and Stephen M. Rock
Department of Aeronautics & Astronautics
Stanford University

Abstract—**This paper presents a SLAM-based approach for creating maps of underwater terrain using AUVs with poor inertial information. The initial motivating application for this work was mapping in the non-inertial frame of a free-drifting Antarctic iceberg, but poor inertial information can also occur if low-cost, high drift inertial instrumentation is used in standard mapping tasks, or if DVL bottom lock is lost during the mission. This paper presents a SLAM-based approach in which features are extracted from concatenated multibeam data and descriptors are created, allowing these features to be compared against past terrain as the vehicle traverses the area. There have been a number of previous research efforts that used feature-based SLAM techniques for underwater mapping, but they have generally made assumptions or relied on sensors that are inconsistent with this paper's motivating application, such as a flat bottom, the availability of visual imagery, or manmade fiducial markers. The method presented here uses natural terrain, is robust to water turbidity, and can be used in areas with vertical terrain like the walls of canyons and icebergs. Results are presented on data collected from Monterey Canyon using a vehicle with a high-grade IMU but that lost DVL bottom lock during the mapping run.**

I. Introduction

The work presented in this paper aims to enable high-precision mapping of underwater surfaces using vehicles equipped with low-precision inertial sensors and without external navigation aids. The initial motivating mission for this work was a NASA ASTEP-funded mission to map free-drifting icebergs as part of a larger goal of understanding life in extreme environments. However, it is shown here that this mission can be posed and solved in a form equivalent to the more general problem of mapping with low-precision inertial sensors.

Mapping underwater surfaces using an AUV is typically a multistep process. First, the AUV flies a pattern over the surface collecting multibeam sonar data as well as navigation data from an inertial navigation system (INS); second, an accurate navigation history of the AUV is constructed offline; and finally, the sonar measurements are projected from these location estimates to form a point cloud, which can then processed to produce a final map in a number of formats, like discrete elevation map (DEM) or octree.

Obtaining a precise navigation solution is key to the process. Finding this solution often involves a refinement step in which sections of measured terrain surfaces (e.g. point clouds of sonar data where trajectories cross) are correlated to calculate corrections to the AUVs tracklines. This is known in the robotics community as Simultaneous Localization and Mapping, or SLAM. In underwater mapping, the typical approach has been to use dense, data-driven correlation techniques that do not require explicit representation of the environment, such as ICP, to perform these corrections. While highly accurate, good initial estimates of the AUVs tracklines are generally required for these methods to ensure convergence of the solution. This is often the case when mapping stationary terrain using an AUV equipped with a high-precision INS (e.g. $\leq 0.05\%$ of distance travelled), and when the DVL in that system never loses lock on the terrain.

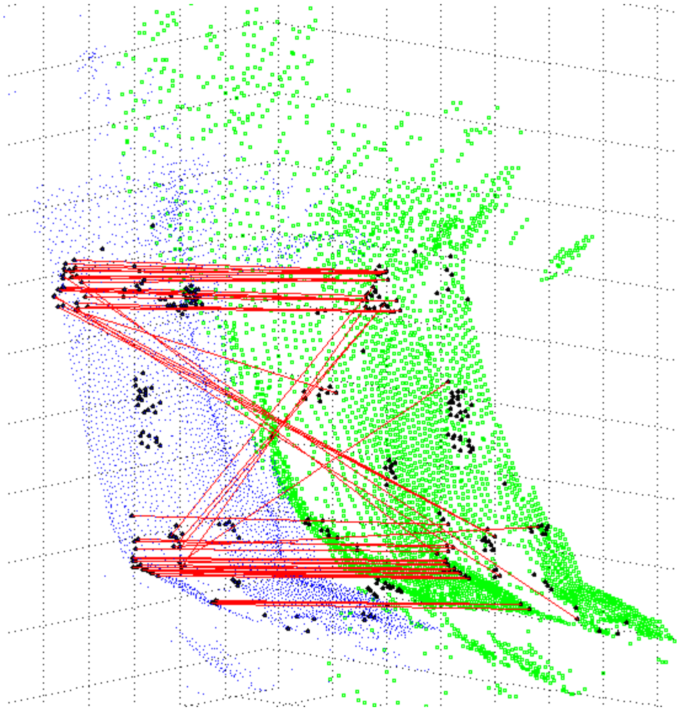


Fig. 1. Features extracted from two passes of SONAR data on a canyon wall. Red lines are hypothesized correspondences, based on comparing descriptors. These correspondences can be used to correct errors in inertial measurement to create accurate maps. Care must be taken to make the algorithm robust to false matches (diagonal lines).

If the terrain is moving, as in the case of an iceberg, or if a low-precision INS is used, then the errors in the tracklines can be large, and convergence may fail when attempting to align the sonar point clouds. For example, typical rotation rates of an iceberg can be 5–10 deg/hr. At normal operating speeds for AUVs, this can translate into an apparent dead-reckoning drift of over 4% of distance traveled [1], even when using a high-precision INS. For low-precision INS systems (e.g. using calibrated MEMs gyros, and fluxgate magnetometer) drift rates can be 1–5% DT even when operating over stationary terrain [2] [3].

The problem of mapping an iceberg was addressed by Kimball [4]. Specifically, he developed a SLAM technique in which the motion of the AUV in the iceberg's reference frame could be estimated offline. However, his solution required explicit modeling of the inertial motion of both the iceberg and the AUV, and was difficult to scale to larger problems with many loop closure events (instance where the same point or feature is observed on repeated passes).

In [5], the iceberg mapping problem was reformulated to eliminate the need for an inertial trajectory estimate, instead developing all the equations in the icebergs frame of reference. Formulated in this way, the motion of the iceberg is indistinguishable from a slowly time-varying gyroscope drift, making the problem equivalent to an inertial navigation and mapping problem with imperfect sensors. The advantage of this formulation is that the resulting problem has been thoroughly documented in the robotics literature, and a number of methods have been developed to solve it.

For the application presented here, feature-based GraphSLAM was selected. GraphSLAM is scalable to large mapping problems, tolerant of large errors in the initial trajectory estimate. This process is enabled by extracting recognizable features from the terrain and using these to establish correspondence between measurements, as illustrated in Figure 1.

This paper describes the use of GraphSLAM to solve the mapping problem when only low-precision inertial information is present (Section III-A). It presents a technique for extracting recognizable features of a terrain from point clouds of sonar data (Section III-B). Finally, it demonstrates the feasibility of the approach using field data collected in Monterey Canyon.

II. Related Work

A. SLAM in other applications

Robotic mapping has been throroughly studied for decades. For a good overview of general methodology, refer to [6]. Since a mobile sensor's measurements are generally recorded in the robot's reference frame, a global map is highly coupled with, and dependent upon, the accuracy of the robot's navigation solution. Even when aids like GPS are used, loop closure is often fine-tuned via scan-matching techniques. This means that nearly all mapping applications are in fact SLAM problems.

A number of methods for solving the SLAM problem have been proposed and studied extensively. These include EKF SLAM, where the pose history and map feature locations are modeled as a single multivariate Gaussian, GraphSLAM where the problem is represented as a Probabilisitic Graphical Model, and FastSLAM, which combines concepts from EKFs and particle filters [7]. These and other methods have been employed in a wide range of applications, from exploring abandoned mines [8] and flooded sinkholes [9] to satellite reconstruction [10].

The application discussed in this paper differs in two key ways from the terrestrial SLAM applications discussed in the literature. First, the applications typically have access to GPS or relatively precise odometry to initialize correlation algorithms for loop closure. The motivating problem for this work has insufficiently precise inertial navigation to reliably initialize scan-matching for loop closure. Data association must be accomplished some other way, which leads to the second challenge. The problem of data correspondence is more difficult underwater. Mapping AUVs do not typically have ranging sensors like Flash LIDAR or structured light cameras that can instantaneously capture dense 3D point clouds for correlation. Additionally, image feature-based SLAM is limited by the attenuation of light in water: while it has been done, applications have been limited to shallow depths and low-turbidity conditions [11]. Again, this is inconsistent with this paper's motivating application, as the AUV operates in conditions where extracting visual features reliably is impossible (great depth, high turbidity, long standoff distances).

The method presented here assumes a single multibeam sonar that returns roughly 500 range-azimuth pairs at each ping, arranged in a fan pattern. With this arrangement, there is not enough information or overlap to do scan-to-scan matching. These sensing limitations are what has driven the strategy to extract descriptive features from the available range data, and use it to detect loop closure events to correct inertial drift.

B. 3D Feature Extraction

The correspondence problem – recognizing when two measurements correspond to the same point in physical space – is a central part SLAM. Solving it is critical, as it is often the only source of new information that can correct accumulated odometry error. A 2006 survey of SLAM methods used for underwater navigation reported that most successfully fielded SLAM techniques up until that point used a data-driven scan matching approach to solving the loop closure problem that does not rely on an explicit representation of features. [12]. These methods typically rely on performing a grid search or ICP correlation between dense point clouds; they work well but can be computationally expensive and require a good initial estimate of alignment to converge to the correct minimum. [13]. The same survey lists a number of early attempts at feature-based SLAM. In [14] and [15], a 2D scanning sonar was used to track manmade sonar target features.

In the past few years there has been great interest and progress made in the development of 3D point cloud feature extraction among the terrestrial robotics community. Many of these algorithms are closely related to 2D image features. This is not surprising, as many of the image feature algorithms treat images as if they were terrain maps, where the image intensity of a given pixel can be thought of as terrain height. Some well-known algorithms that use this idea are Harris Corners [16], SIFT [17], and SURF [18]. The general approach with all of these algorithms is that by considering not only a single point or pixel, but a small neighborhood of points and pixels, stable features can be identified and statistics extracted such that the feature can be identified and matched by later observations. Methods vary in their computational complexity, robustness to noise, invariance under various combinations of transformation, rotation, lighting, etc.

Early work in 3D point cloud feature extraction calculated normal vector direction and local curvature information to characterize interest points. Progress in 3D features has followed along similar lines as their 2D counterparts, with the development of algorithms like Point Feature Histograms and Fast Point Feature Histograms [19], that are to large degree orientation-invariant and useful for classifying different classes of shapes.

Work in both 2D and 3D feature work has focused largely on manmade environments, and being able to perform object recognition from arbitrary vantage points. However, the natural environments in which AUVs typically operate often have much less relief, and far fewer right angles and sharp gradients that are so prevalent in manmade structures. This causes degraded performance in many of the standard methods. Additionally, in typical mapping tasks, certain parts of the robot's ego-motion are well known, and the features of interest are usually fixed in the reference frame of the map, so orientation invariance may in fact be undesirable. With this in mind, a simple, custom approach to extracting features and finding loop closure events will be described below. While the details differ, the intuition that motivates the feature extraction is similar to that of Harris Corners for 2D image processing.

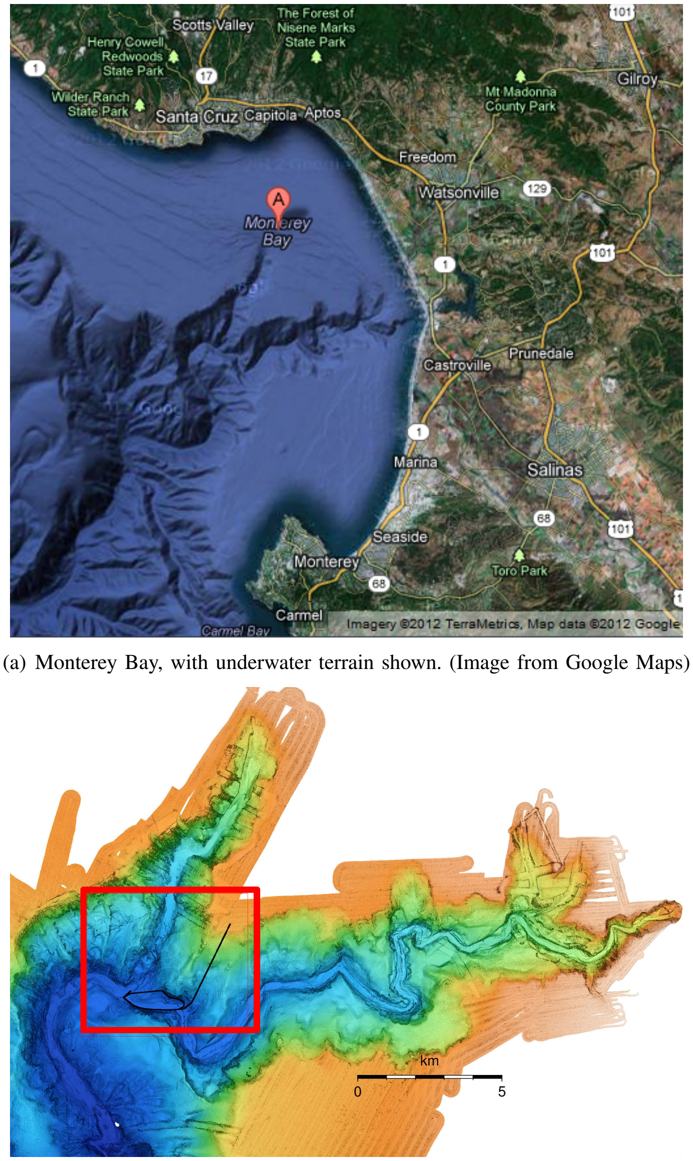
III. Approach

A. *GraphSLAM*

Many techniques exist for solving the SLAM problem described above. The method chosen here is GraphSLAM.

1) Algorithm Overview: The GraphSLAM algorithm models the process of mapping as a probabilistic graph with two types of nodes: pose nodes and feature nodes. The pose nodes represent the vehicle's state at a given time. The feature nodes represent the best estimate of the state of observed landmarks or map features, including their locations and possibly any identifying properties, such as color, albedo, etc. Successive pose nodes are connected with odometry links, with process noise modeled as Gaussian, as in Kalman filters and many others. When a feature is observed by the sensor at time t that pose node is connected to the corresponding feature node via a measurement link. These links are also modeled as Gaussian, with the mean given by the measurement and the covariance a function of the sensor and geometry. To represent all these links, GraphSLAM uses the information form of the Gaussian. A benefit of this is the ability simply to add information as it comes in, postponing solution until all information has been accumulated. A detailed explanation of the method is laid out in [7].

Once constructed in this way, GraphSLAM exploits the problem's structure to solve efficiently even very large mapping problems. Specifically, the problem's sparse, positive-semidefinite structure allows for solution through the method of conjugate gradients. In one notable example, a map of part of Stanford's campus was created using 10^5 poses and 10^8 observations in roughly 30 seconds of cpu time. [7]



(a) Monterey Bay, with underwater terrain shown. (Image from Google Maps)

(b) Detail of Soquel Canyon bathymetry data, showing location of mapped wall. (Image from MBARI)

Fig. 2. Location of test data collection.

The advantages of GraphSLAM are most apparent when incorporating loop closure events, where the same feature is observed at least twice, separated by a long span of time. It is able to use these links to correct errors accumulated during long periods of integrated odometry. In GraphSLAM with known correspondence, these events are known a priori and require no special treatment. In the more challenging GraphSLAM with unknown correspondence, correspondence can be discovered incrementally, with links being made (and even broken) as the solver works toward finding a solution [7].

2) Application to Underwater Mapping: GraphSLAM requires a state representation for poses and for features, a process model, and a measurement model.

The state representation used for the canyon wall mapping mission described in Section IV is

$$X_t = \begin{bmatrix} x & y & z & u & v & w & \phi & \theta & \psi \end{bmatrix}^\intercal \tag{1}$$

The first three terms are the 3-DOF position in the iceberg's frame of reference. u, v, w are the body components of velocity in the iceberg frame (i.e. the values reported by the DVL). ϕ, θ, ψ are the vehicle's roll, pitch, and heading in the iceberg frame, respectively.

In the case of the iceberg, the single greatest source of uncertainty is the iceberg's rotation, so the normal state can be augmented by a bias term as:

$$X_t = \begin{bmatrix} x & y & z & u & v & w & \phi & \theta & \psi & b_z \end{bmatrix}^\intercal \tag{2}$$

where b_z is the apparent gyro drift in the z-axis due to iceberg rotation.

It should be noted that these representations are not unique. If it is suspected that the vehicle's DVL has a bias, this can also be appended to the state. In general, the state representation and the inclusion of error terms will be a function of the error characteristics of the particular vehicle, sensor, and mission profile. It is up to the user to find a representation that adequately describes the observations, but avoids overfitting the data.

The process is modeled as Gaussian, such that

$$X_{t+dT} \sim \mathcal{N}\left(X_t + dX, \Sigma_p\right) \tag{3}$$

where dX is obtained by integrating the current odometry measurements over the time interval dT

$$dX = \begin{bmatrix} \int_0^{dT} {}^I R^V(t) v(t)\,\mathrm{d}t \\ 0 \\ \int_0^{dT} \vec{\omega}\,\mathrm{d}t \\ 0 \end{bmatrix} \tag{4}$$

and Σ_p is the process noise covariance. The zero values appearing in dX indicate that the quantities are expected to change only very slowly over time. If a higher fidelity motion model exists to describe the process, these might be nonzero.

Feature locations and descriptors m_j are also modeled as Gaussian, with measurements tying features to the poses at which they were observed. The measurements $z_{t,j}$ are themselves represented as a 3-dimensional vector in the vehicle's reference frame with some uncertainty ellipse, as well as an abstract descriptor, d, that captures other information useful for determining correspondence.

$$z_{t,j} = \begin{bmatrix} r_x \\ r_y \\ r_z \\ d \end{bmatrix} \tag{5}$$

This information yields a graph constraint of the form

$$\left(z_{t,j} - h(x_t, m_j)\right)^\intercal Q^{-1} \left(z_{t,j} - h(x_t, m_j)\right) \tag{6}$$

with the expected measurement

$$h(x_t, m_j) = {}^I R^V(t)^\intercal \left(m_j - x_t\right) \tag{7}$$

where Q is the measurement covariance matrix and ${}^I R^V(t)$ is the rotation matrix of the vehicle at time t.

B. Extracting Features from Terrain

Using the above measurement model for loop closure is predicated on the ability to detect stable, recognizable features in the terrain. In this context, stability means that the feature is likely to be detected from multiple viewpoints and at different times, and its location is unambiguous. Recognizability means that a feature can be identified as having been observed previously. Extracting stable points from 3D point cloud data involves identifying areas of high curvature. This is generally straightforward. Building recognizable feature descriptors is more challenging.

AUVs used for bathymetric mapping are typically outfitted with multibeam sonars, which use interferometry to calculate range and bearing to many individual points in a single ping. These measurements can be represented as vectors of known length and direction in the vehicle frame emanating from the sensor in a fan pattern as shown in Figure 3. The scans recorded at each time step are concatenated incrementally to form a point cloud of the surface. If the terrain is moving, or if there are large errors in the odometry, the map will appear to warp, causing inconsistencies over time. However, over short time scales, the effects of this warping are small, and can be ignored. Using these submaps built from data collected over short intervals, features can be extracted.

The solution presented here uses a simple custom method that combines elements of several existing algorithms. The approach uses a cascading filter based on curvature to identify points that are likely to be stable over time. First, the data is cleaned of spurious outlying points in a scanwise fashion. For each point in the cleaned data set, the curvature κ is calculated. If the curvature is not sufficiently high, the point is rejected as a possible feature.

$$\kappa = \sqrt{\frac{\left(z''y' - y''z'\right)^2 + \left(x''z' - z''x'\right)^2 + \left(y''x' - x''y'\right)^2}{\left(x'^2 + y'^2 + z'^2\right)^{3/2}}} \tag{8}$$

Each point with sufficient curvature is then analyzed relative to its neighbors in the submap built from multiple scans. Its k-nearest neighbors are extracted and an eigendecomposition is performed on the patch to determine normal vector direction and 3D-curvature magnitude. If this curvature metric is sufficiently high, the point is labeled a feature. For a detailed discussion of this and other methods for extracting 3D curvature of interest points, refer to section 4.3 of [20]

Through empirical analysis, it was found that the inclination of the normal vector from the horizontal was the most reliable

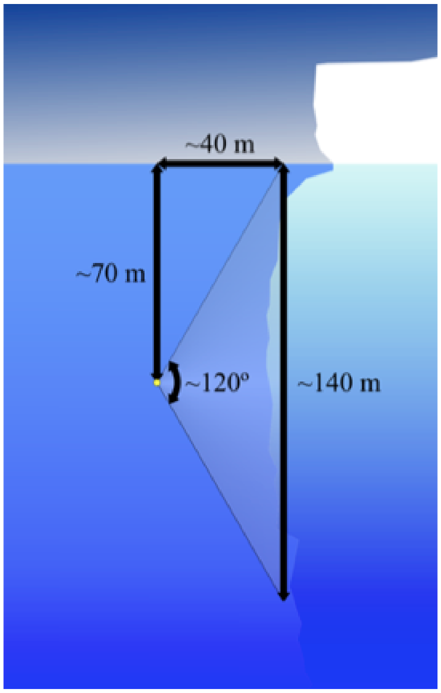


Fig. 3. Orientation of multibeam SONAR with respect to AUV. *Graphic courtesy of Peter Kimball*

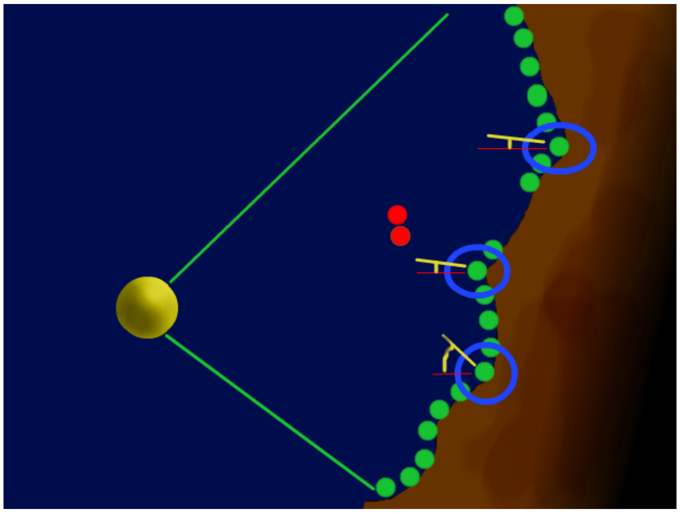


Fig. 4. Sonar returns off a canyon wall. Red points are outliers. Returns circled in blue are likely feature candidates. The yellow tick marks indicate inclination of the normal above the horizontal

discriminator of possible matching features between swaths. Using this, corresponding features were identified between in pre-processing, using random sample consensus (RANSAC) [21] and iterative closest point (ICP) using a 4D Mahalanobis distance metric based on 3D feature position and normal inclination.

$$\text{dist} = [z_{t_1,j}]^{\intercal} \, W \, [z_{t_2,j}] \tag{9}$$

The weights W for this metric were chosen based on the uncertainty covariance for each of the four axes. The inliers that remained after RANSAC were the correspondences used in the GraphSLAM framework.

C. Ceres Solver

To solve the SLAM problem described above, the Ceres Nonlinear Least Squares Graph Solver developed by Google was used. It provides a number of highly optimized solvers and has a flexible user interface allowing great control over the problem definition.

IV. Results

The mapping approach described above was applied to previously collected data from Soquel Canyon in Monterey Bay, shown in Figure 2. The data were recorded using MBARI's mapping AUV, configured with its multibeam sonar mounted as shown in Figure 3. The vehicle flew three passes by a nearly vertical wall of the canyon at three different depths, as shown in Figure 5. For this paper, data from two of the overlapping passes were used. The data set consists of 90 minutes of 3 Hz odometry data (15k poses). Of the roughly 3.5 million raw sonar returns recorded during this time, 4086 made it through the pre-processing feature extraction step, and 157 were found to correspond to an earlier obervation. These 157 correspondences are what drives the GraphSLAM algorithm, allowing it to compensate for integrated dead reckoning errors. The terrain opposite the wall, which was imaged on the return-leg between passes, is relatively flat and no stable features were observed in the region.

A. Feature Extraction on Natural Terrain Data

Figure 5 shows features extracted from three passes by the canyon wall. The green points are raw sonar returns, after cleaning. The red, black, and cyan markers show points that met the curvature criteria to be counted as features. The different colors correspond to on which pass the observation was made.

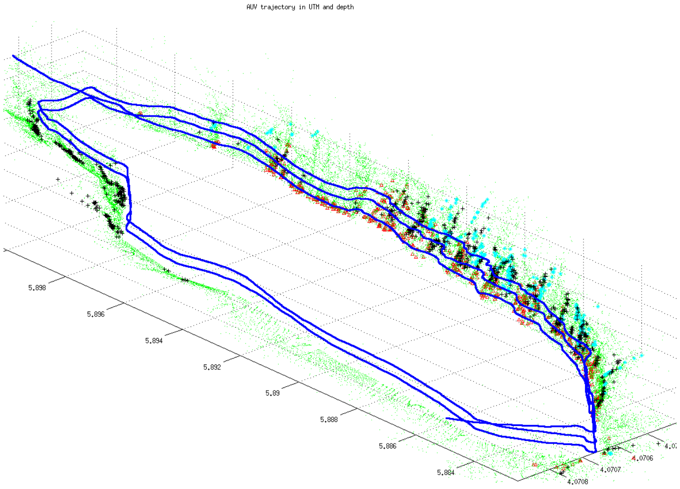
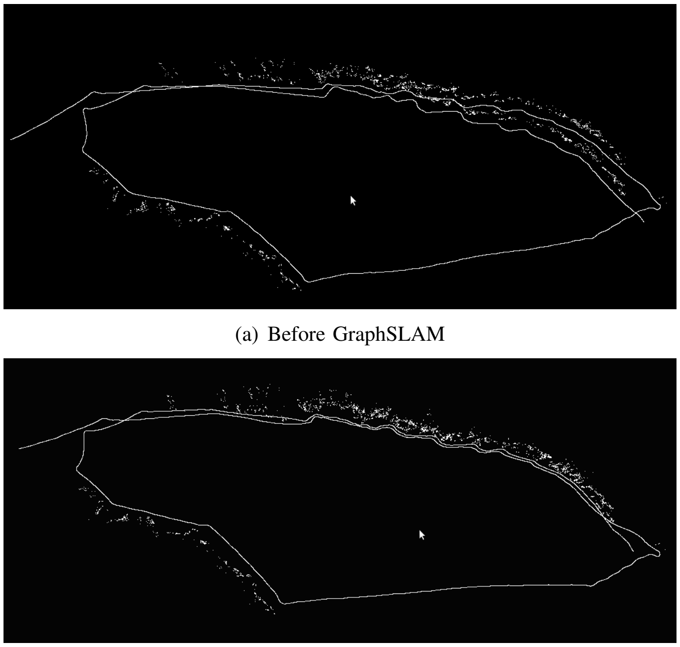


Fig. 5. Features extracted from three passes by the wall of Soquel Canyon, looking Southeast. Color coded by pass.



(a) Before GraphSLAM

(b) After GraphSLAM

Fig. 6. Sonar data from Soquel Canyon in Monterey Bay before and after correction via GraphSLAM, looking South. Note that the inconsistency in the canyon wall due to imperfect inertial guidance is eliminated.

B. Drift Correction on Soquel Canyon Map Data

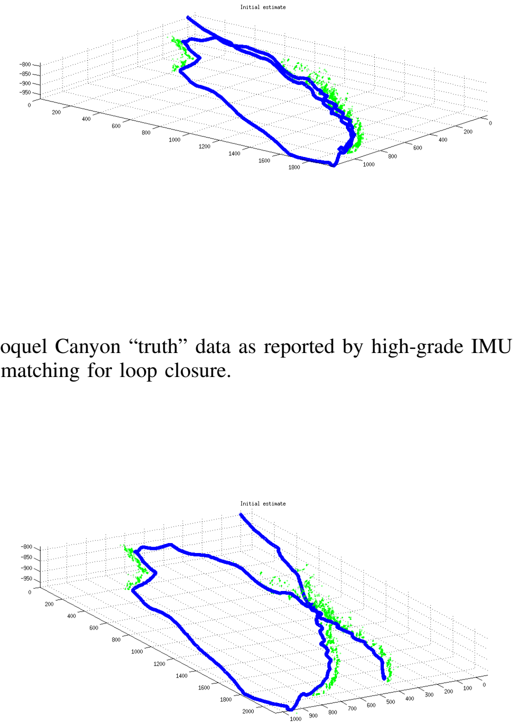
Figure 6 shows the results of GraphSLAM on data collected from two passes by a steep wall of Soquel Canyon in Monterey Bay. The wall is roughly 1.5km in length. DVL lock was lost during the run, resulting in a misalignment of 5 to 10 meters between the two swaths of wall data, even with high-grade inertial measurements. The resulting misalignment is most visible in the top right corner of Figure 6(a).

The algorigthm successfully fuses the two swaths of sonar data into a single, self-consistent map, while leaving the high frequency characteristics of the data mostly intact. It was found that both the convergence rate and the resulting map accuracy as compared with maps generated using high-grade IMU and scan-matching for loop closure were sensitive to the relative weights put on the different terms in the measurement and odometry residual blocks in the SLAM optimization solver. Care must be taken when weighting these terms, so that the errors are spread realistically over the unknown quantities, based on the uncertainty of each.

C. Robustness to Large Heading Uncertainty

To test the algorithm's ability to recover from the types of errors encountered with free-drifting iceberg mapping tasks, a large heading bias was injected into the navigation data. The resulting warping can be seen in Figure 7(b).

The GraphSLAM algorithm was initialized with this large heading bias using the iceberg state representation described above. The results can be seen in Figure 8. While the two passes have been resolved into one self-consistent wall, some warping persists in the resulting map. However, for many applications, such as Terrain Relative Navigation, warping is tolerable, so long as the map is self-consistent.



(a) Soquel Canyon "truth" data as reported by high-grade IMU and scan-matching for loop closure.

(b) Data corrupted with heading bias

Fig. 7. Soquel Canyon data showing the effects of a large time-varying heading bias, simulating the effects of unmodeled iceberg motion or low-grade sensors. Blue points represent the trajectory. Green points are sonar sounding locations.

D. Summary

The problem of mapping with imperfect inertial information, whether from low-precision instruments or environmental motion was posed as a GraphSLAM problem. This problem was then solved on field data collected in Monterey Canyon. To enable this solution method, features were extracted from the natural terrain and used for data correlation on multiple passes. Additionally, robustness of the algorithm to large initial errors was demonstrated by adding a heading rate bias, consistent with the magnitude of motion that could be expected when mapping an iceberg. In both cases, GraphSLAM was able to use terrain information to estimate its navigation errors and create a self-consistent map.

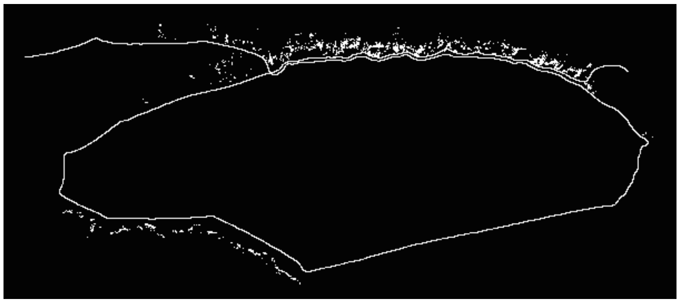


Fig. 8. Results of GraphSLAM on Soquel Canyon data after being initialized with large heading bias to simulate unmodeled iceberg motion. Data from two passes by the wall have been resolved, but some warping remains.

Acknowledgment

This work was funded by NASA ASTEP Grant NNX11AR62G with additional support from the Monterey Bay Aquarium Research Institute (MBARI).

References

[1] P. Wadhams, M. Kristensen, and O. Orheim, "The response of antarctic icebergs to ocean waves," *Journal of Geophysical Research: Oceans (1978–2012)*, vol. 88, no. C10, pp. 6053–6065, 1983.

[2] N. Barbour and G. Schmidt, "Inertial sensor technology trends," *Sensors Journal, IEEE*, vol. 1, no. 4, pp. 332–339, 2001.

[3] D. Meduna, *Terrain relative navigation for sensor-limited systems with application to underwater vehicles*. Phd, Stanford University, August 2011.

[4] P. Kimball and S. M. Rock, "Estimation of iceberg motion for mapping by auvs," in *IEEE-OES Autonomous Underwater Vehicles Conference (AUV)*, (Monterey, CA), 08/2010 2010.

[5] M. Hammond, "Enabling auv mapping of free-drifting iceberg without external navigation aids," in *IEEE OCEANS Conference*, (St. John's, Newfoundland), 10 2014.

[6] S. Thrun *et al.*, "Robotic mapping: A survey," *Exploring artificial intelligence in the new millennium*, pp. 1–35, 2002.

[7] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents, MIT Press, 2005.

[8] S. Thrun, D. Hahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker, "A system for volumetric robotic mapping of abandoned mines," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 3, pp. 4270 – 4275 vol.3, 2003.

[9] N. Fairfield, G. Kantor, and D. Wettergreen, "Towards particle filter SLAM with three dimensional evidence grids in a flooded subterranean environment," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 3575 –3580, May 2006.

[10] S. Augenstein, *Monocular Pose and Shape Estimation of Moving Targets, for Autonomous Rendezvous and Docking*. Phd, Stanford University, 06/2011 2011.

[11] S. Williams, O. Pizarro, M. Jakuba, and N. Barrett, "AUV benthic habitat mapping in south eastern tasmania," in *Field and Service Robotics* (A. Howard, K. Iagnemma, and A. Kelly, eds.), vol. 62 of *Springer Tracts in Advanced Robotics*, pp. 275–284, Springer Berlin / Heidelberg, 2010.

[12] J. C. Kinsey, R. M. Eustice, and L. L. Whitcomb, "A survey of underwater vehicle navigation: Recent advances and new challenges," in *IFAC Conference of Manoeuvering and Control of Marine Craft*, 2006.

[13] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Robotics-DL tentative*, pp. 586–606, International Society for Optics and Photonics, 1992.

[14] I. T. Ruiz, Y. Petillot, D. M. Lane, and C. Salson, "Feature extraction and data association for auv concurrent mapping and localisation," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 3, pp. 2785–2790, IEEE, 2001.

[15] S. Williams, G. Dissanayake, and H. Durrant-Whyte, "Towards terrain-aided navigation for underwater robotics," *Advanced Robotics*, vol. 15, no. 5, pp. 533–549, 2001.

[16] M. Harris, C. & Stephens, "A combined corner and edge detector," in *4th Alvey Vision Conference*, (Manchester,), p. 147151, 1988.

[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[18] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*, pp. 404–417, Springer, 2006.

[19] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 3212–3217, IEEE, 2009.

[20] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," *KI-Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, 2010.

[21] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.