IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 13, 2018, accepted April 16, 2018, date of publication April 27, 2018, date of current version June 5, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2830819

A Multibeam-Based SLAM Algorithm for Iceberg Mapping Using AUVs

PETTER NORGREN AND ROGER SKJETNE, (Member, IEEE)
Department of Marine Technology, Norwegian University of Science and Technology, N-7491 Trondheim, Norway

Corresponding author: Petter Norgren (petter.norgren@ntnu.no)

This work was supported in part by the Research Council of Norway (RCN) through the Center of Excellence Autonomous Marine Operations and Systems, under Project RCN-223254, and in part by the Center for Research-Based Innovation Sustainable Arctic Marine and Coastal Technology, under Project RCN-203471.

ABSTRACT Using autonomous underwater vehicles (AUVs) for mapping underwater topography of sea-ice and icebergs, or detecting keels of ice ridges, is foreseen as enabling technology in future arctic marine operations. Wind, current, and Coriolis forces affect an iceberg's trajectory, making automated mapping difficult. This paper presents a method aiming at enabling autonomous iceberg mapping using AUVs equipped with a multibeam echosounder by estimating the position and orientation of the iceberg. The method is based on a bathymetric simultaneous localization and mapping (SLAM) algorithm, namely, the bathymetric distributed particle filter SLAM (BPSLAM) algorithm. The proposed method estimates the AUV's pose in an iceberg-fixed coordinate system. The relative states can be used for both guiding the vehicle to achieve complete coverage, as well as estimation of a consistent iceberg topography. The algorithm also provides an estimate of the iceberg's drift velocity – an important parameter for the AUV trajectory planning as well as any related ice management (IM) operations. Two new weighting algorithms for the BPSLAM method are proposed, enabling batch processing of multibeam echosounder (MBE) measurements to ensure real-time operation without discarding information. The proposed method is demonstrated using a real iceberg topography taken from the PERD iceberg sightings database, with simulated AUV and MBE range measurements. The algorithm is also evaluated on a real world bathymetric dataset, collected using the HUGIN HUS AUV.

INDEX TERMS Ocean engineering, AUV, SLAM, multibeam, arctic technology, iceberg management, ice surveillance.

I. INTRODUCTION

In Arctic marine operations, where there may exist a threat of sea-ice, and/or icebergs, ice management (IM) is often used to mitigate the risk posed by any ice features. Eik defines in [1] IM as the sum of all activities where the objective is to reduce, or avoid, actions from any kind of ice feature. An IM system includes methods for detecting, tracking, and forecasting sea-ice, ridges, and icebergs. The consequences of failure, especially in the fragile Arctic ecosystem, are severe and detailed information about the current ice conditions will be important to reduce the risk.

To develop accurate iceberg drift models for iceberg trajectory forecasting, detailed knowledge of iceberg keel geometry is necessary. Kubat *et al.* present in [2] an operational approach to estimate iceberg drift using an empirical model of the keel cross-sectional area depending on waterline height. Detailed iceberg surveys would improve the statistical foundation for the development of such empirical models [3]. Another modelling application that would greatly benefit from iceberg keel surveys is iceberg deterioration models, as stated by Murphy and Carriers in [4]. Such models would require repeated and systematic surveys of the same iceberg.

Autonomous underwater vehicles (AUVs) have been used in increasingly complex Arctic operations since the first reported AUV deployment in the Arctic in 1972, presented by Francois and Nodland in [5]. AUVs are suitable for a wide array of tasks due to the vehicle's high spatial and temporal coverage. AUVs are also unaffected by the potentially harsh surface conditions in the Arctic during their mission. Forrest *et al.* present in [6] an experiment where an AUV equipped with an interferometric sidescan sonar was used to capture draft measurements of the underside of a fragment of the Peterman Ice Island. Forrest *et al.* state that planning the AUV mission for mapping the drifting and rotating iceberg was

the most challenging part of the operation. This motivates an autonomous mapping scheme, but before the guidance problem can be solved, the robot must know its location relative to the environment it is mapping.

Localization is the problem of determining a robot's position and orientation in a reference frame. When performing mapping of an unknown environment, the robot must know its location in order to build a map. Conventional underwater robotic navigation is usually performed using a combination of acoustic positioning systems and dead-reckoning by inertial navigation. The common methods for acoustic positioning are long-baseline (LBL) and ultrashort-baseline (USBL), and where LBL requires two or more transponders to be deployed in the operational area, USBL usually requires a support ship to stay close to the AUV during the survey. The time between updates is also relatively long for acoustic systems, depending on range and other operational factors. Inertial navigation does not require external instrumentation, but the navigation uncertainty grows unbounded unless position fixes are acquired, either by using global navigation satellite systems (GNSS) or acoustic positioning.

Simultaneous localization and mapping (SLAM) is a method that attempts to build a map of the unknown environment, while using the same map to determine the robot's location inside the map. Since the robot's location is also needed for building the map, SLAM can be considered a chicken-or-egg problem, making it a hard problem to solve [7]. SLAM can be used to bound the error when performing inertial navigation without external instrumentation, and therefore SLAM is considered to be a requirement for truly autonomous operations [8]. Thus, much work has been conducted in the field of SLAM the last three decades (see e.g. [8]–[10] and references therein).

Either solutions to the SLAM problem can consider the full trajectory of the robot, dubbed the full SLAM problem, or it can consider only the current pose of the robot. The latter is called online SLAM. This paper will only consider online SLAM, since the full SLAM problem is typically too computationally demanding to be solved online, and a mapping scheme requires an online solution. Furthermore, SLAM can be divided into two main groups based on its map representation. First, feature-based SLAM only store landmark locations (which may be updated upon reobservation). Detecting a landmark from sensor readings (e.g. laser, camera, sonar), and perhaps even more important, determining if it is the correct landmark, require a feature detector. Featureless approaches, on the other hand, rely on a sensor model to evaluate observations and update the map (often a grid map). Featureless approaches are often preferred in environments where clearly identifiable features are sparse.

The method presented in this paper is based on the bathymetric distributed particle SLAM (BPSLAM) algorithm presented by Barkby *et al.* in [11]–[13]. The BPSLAM algorithm, which is a featureless, grid-map based approach to the online SLAM problem, was chosen as iceberg topography mapping has many similarities with bathymetric mapping, such as lacking clearly identifiable features and utilizing MBE as the main mapping sensor. The BPSLAM algorithm also has real-time properties. This is an important aspect since the objective is to autonomously explore and map an iceberg, requiring the SLAM outputs to be available in real-time. Preliminary work on the algorithm presented in this paper can be found in [14].

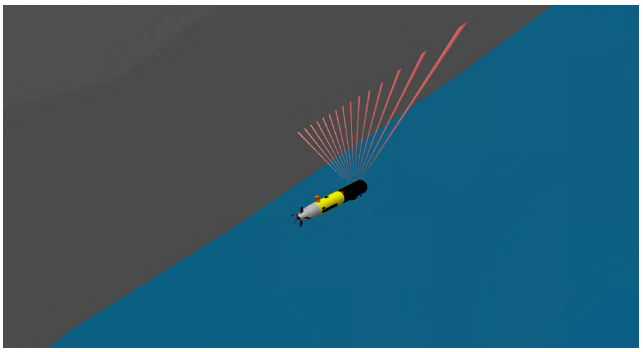

FIGURE 1. Illustration of an AUV performing under-ice mapping using a multibeam echosounder looking up at the ice.

The contributions in the work outlined in this paper are: 1) development of an AUV/iceberg relative motion model; 2) an online iceberg mapping filter estimating the relative position and velocity between the iceberg and the AUV using an extended Kalman filter (EKF) and particle filter SLAM; 3) development and evaluation of two new weighting schemes for BPSLAM to enable batch processing of the MBE measurements and to adapt the update rate of the SLAM algorithm to ensure real-time operation; 4) development of an Arctic AUV simulator; and 5) verification of the method on a real-world dataset collected with the HUGIN HUS AUV [15] in the Trondheimsfjord.

A. NOTATION

Matrices are written in capital letters and vectors are written in small letters. The dimension of each variable will be defined. A variable in the Euclidean space with dimension n is denoted $\mathbb{R}^n$, while matrices of dimension $n \times m$ are denoted $\mathbb{R}^{n\times m}$. The total time derivative of a variable $x(t)$ is denoted $\dot{x}$. Superscript denotes the reference frame to which a given vector is expressed. For example, p^n is a position in the north-east-down (NED) frame. The reference frames used are: NED (n), BODY (b), ICE (i), and BEAM (m). A rotation matrix $R_a^b \in \mathrm{SO}(n)$ between reference frames uses a subscript for the frame transformed from, and superscript for the frame transformed to. For example, rotating from BODY to NED is denoted R_b^n. For horizontal 3 degree-of-freedom (DOF) motion, with states (x, y, ψ), we use the simpler notation $R(\psi) = R_z, \psi \in SO(3)$, representing a rotation of a reference frame about the z-axis by an angle ψ. A 3DOF state with position and heading is also referred to as a pose.

II. BACKGROUND

A. AUVs IN ARCTIC OPERATIONS

AUVs have the unique capability of being able to operate autonomously under the ice for an extended period of time. One of the first AUVs performing complex tasks in the Arctic is the cable-laying AUV Theseus, developed by International Submarine Engineering (ISE), which successfully laid 175 km of fiber-optical cable between Jolliffe Bay and Ice Camp Knossos in Canada, under a 2.7 meter thick ice cover [16], [17]. Another application of AUVs in the Arctic is under-ice surveys. Wadhams *et al.* present in [18] the use of a Maridan Martin 150 AUV that gathered sidescan imagery of the underside of the ice in 2002 – the first of its kind acquired by an AUV. From the sidescan sonar data, the authors were able to identify first-year, multi-year, brash, and frazil ice. Similarly, the Autosub-II AUV was used to obtain the first under-ice multibeam measurements in 2004 [19].

Large scale oceanographic surveys are necessary to assess how climate change in the Arctic is affected by inflow of warm Atlantic water to the Fram Strait into the Arctic Ocean, and the corresponding effects on the global climate. The Atlantic Layer Tracking Experiment was designed to elucidate how Arctic and global climate change are interrelated by surveying the water column with a custom designed AUV, capable of ranges between 1500 and 3000 km and with a depth rating of up to 4500 m. Missions conducted on Arctic latitudes as high as 82° north are presented by Bellingham *et al.* in [20].

More recently in 2010, the Explorer AUV, developed by ISE, was launched from Boden Island and collected under-ice bathymetry for 12 days without surfacing; see [21] and [22]. The AUV transited from the main camp to a remote camp on a drifting ice floe 320 km away, where underwater charging and data upload were demonstrated. From the remote camp, several bathymetric surveys were conducted, and a total of 1000 km of bathymetric data were acquired under ice during the operation. For a survey on AUVs in Arctic operations the reader is referred to [23], and references therein.

B. BATHYMETRY SLAM

Several SLAM algorithms for bathymetric mapping have utilized MBE as the main sensor for observing the environment. Many of the bathymetric SLAM algorithms are featureless approaches that stem from earlier work on terrain-aided navigation (TAN), which only considers the localization problem given a predefined map of the environment (see [24] for a survey on TAN). The reason featureless approaches are often used for bathymetric surveys is the lack of clearly identifiable features on the seabed. An interesting TAN algorithm, which can be seen as a hybrid between TAN and SLAM, is presented by Hagen *et al.* in [25], detailing a method for line-to-line terrain navigation. In this method, the terrain of the previous survey line in a lawn-movern pattern is used to bound the navigational error using synthetic aperture sonar on the HUGIN AUV.

A featureless bathymetric SLAM approach using EKF SLAM is presented by Roman and Singh in [26], where the point cloud collected from the MBE is stored in submaps, and the submaps are pair-wise matched using correlation and an iterative closest point (ICP) algorithm. Other methods dealing with featureless bathymetric SLAM and submaps matched with ICP, extended from 2.5D (2D map with elevation, which is often used as map representation for bathymetry) to 3D, are presented in [27]–[30]. An alternative method using factor graph SLAM and submaps matched with ICP is presented in [31], which is a full SLAM algorithm providing smoothing of the full trajectory.

Fairfield *et al.* propose in [32], [33] a different approach to handle complex 3D underwater environments. By using an occupancy grid of cubic volume elements (voxels) as its map, a complex 3D geometry can be represented. The method uses a Rao-Blackwellized particle filter (RBPF) at the core of the SLAM algorithm (see [34], [35] for details on RBPF), and the evidence of occupancy in a certain voxel is updated by a log-likelihood update function. Fairfield *et al.* state in [33] that the algorithm is robust to noise and that real-time constraints are achieved by adaptively changing the particle number.

In [36]–[38], Eliazar and Parr introduce the distributed particle filter SLAM (DPSLAM) algorithm – a featureless real-time SLAM algorithm using an ancestry tree to store relations between particles. The ancestry tree algorithm makes it possible to use only one map for all particles (RBPF implementations usually require one map per particle), thus eliminating costly map-copy operations. While DPSLAM is intended for laser range sensor, Barkby *et al.* suggest an alternative, namely the BPSLAM, which also uses RBPF and an ancestry tree, but is tailored for use with MBE and bathymetric surveys [11]–[13]. Whereas the original BPSLAM is an online SLAM method, Barkby *et al.* adopt the method to solve the full SLAM problem in [39], [40], and eliminate the requirement for overlap in sensor data by using Gaussian processes.

C. ICEBERG SLAM

Kimball and Rock report in [41] an extension to TAN intended for AUVs, that uses data captured from a side-mounted MBE on a ship to estimate the iceberg-relative ship track as well as iceberg motion. Kimball and Rock further extend the proposed method in [42]–[44]. The method presented in [44] utilizes a sideways-mounted Doppler velocity log (DVL) and MBE, and it optimizes the estimated iceberg trajectory and the measurement positions with respect to map consistency. The estimates of the iceberg trajectory, topography, and rotation are computed after a full circumnavigation of the iceberg, making this an offline SLAM approach with one loop-closure. An important aspect of the work presented by Kimball and Rock is that the iceberg is not instrumented, meaning estimates of iceberg motion must be calculated by the underwater vehicle itself. Hammond and Rock build on the work by Kimball and Rock in [45], and remove the need for external navigation aids in the iceberg

trajectory estimation problem. The result is a method that generates a consistent iceberg topography map, not needing external positioning systems (like a ship with USBL, or LBL transponders), but without an estimate of the iceberg trajectory. In [46], the same authors present a GraphSLAM approach to underwater mapping with poor inertial information, with the intended application being iceberg mapping.

Kunz presents in [47] a SLAM algorithm for mapping of ice floes using an AUV with upwards-mounted MBE and DVL. The presented method is a solution to the full SLAM problem, utilizing pose graph optimization. In [47, Ch. 5], Kunz presents an AUV mission conducted under Antarctic sea-ice, and the results show how the proposed method performed in estimating the ice drift. During the mission, the AUV guidance system was able to track the translation of the sea-ice from measurements by the DVL, but the guidance system could not account for the rotation of the sea-ice, as this was not estimated before after the mission was completed.

III. PROBLEM FORMULATION

The problem considered is using an AUV equipped with an MBE to map the underwater topography of an iceberg using featureless bathymetric SLAM based on a particle filter estimator. The main challenge to be solved is accurate mapping of the underwater iceberg geometry in an iceberg-fixed coordinate frame that is translating and rotating at an assumed constant velocity.

SLAM algorithms work by utilizing a map of the environment to estimate the state of the vehicle. The objective is to estimate the relative state between the AUV and the iceberg, and therefore, information about the iceberg is required. If no a priori information about the iceberg is available, the AUV must build a map of the iceberg while simultaneously performing localization of the vehicle inside the map. An update of the estimated iceberg state will therefore only be performed upon loop-closures, that is, upon re-observation of a previously mapped area.

Three coordinate reference frames are used to describe the motions of the iceberg and AUV. These are the Earth-fixed North-East-Down (NED) frame, assumed inertial and denoted the ''global'' frame $\{n\}$, the AUV body-fixed frame denoted $\{b\}$, and the iceberg-fixed frame denoted $\{i\}$; see Figure 2.

We make the following assumptions for the development of our algorithm:

1) Iceberg drift velocity (incl. rotational rate) is constant.
2) An upper bound on the iceberg's size is known.
3) Drift velocity of the iceberg is much smaller than the AUV velocity.
4) Iceberg topography does not change during the survey.

The first assumption stems from the inherent inability of BPSLAM to estimate the drift velocity and rotation when the AUV is between loop closures. This motivates the use of an active SLAM approach to optimize performance by minimizing time between loop closures. In this paper,

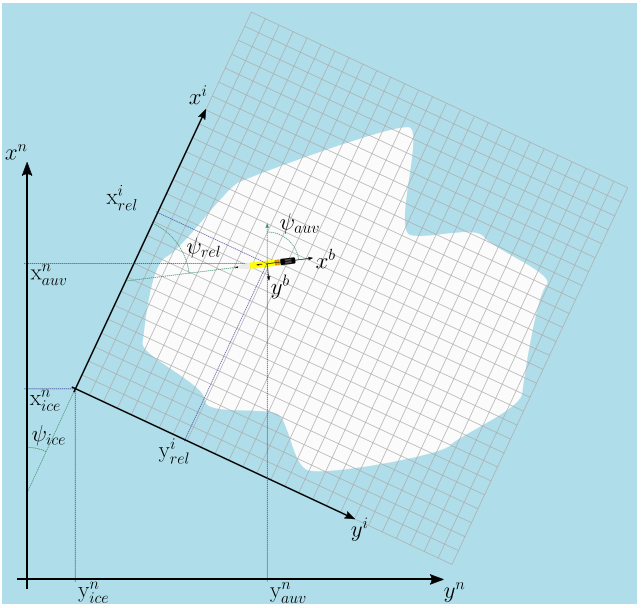

FIGURE 2. Iceberg-AUV relative coordinate system.

however, we will not study the development of an active SLAM guidance scheme. Thus, we assume there exists a guidance system generating a lawn-mover pattern (in the iceberg-fixed frame), for which the MBE coverage area overlaps from leg to leg – resulting in frequent loop-closures after the first leg is completed. An a priori estimate of the drift velocity is also assumed known. This will greatly improve the performance of the estimator, since no update can be made before the first loop closure, and without any prior information, this error will propagate throughout the survey.

The BPSLAM algorithm uses a fixed size grid map, with fixed resolution, thus requiring knowledge of an upper bound on the iceberg length and width in the survey setup. Knowledge of this size bound is reasonable, since it typically is provided by other systems, e.g., ship radar, UAVs, or satellites. However, if the AUV would be operating as a standalone system, without remote intervention, a possible solution would be to first run the AUV in an ''iceberg size detection'' mode, where the boundaries and keel depth of the iceberg are determined; see e.g. [48], [49]. This could also be the phase where a priori estimates of the linear drift is made, using measurements from an upwards-looking DVL. According to Yulmetov *et al.* in [50], the rotation of the icebergs is mainly affected by tidal currents and the Coriolis effect, and in a given area multiple icebergs have been observed to exhibit the same rotational trend. Thus, an a priori rotational rate may be estimated based on the operational area.

The third assumption is an operational constraint – the AUV must be able to move faster than the environment it is mapping. To fulfil the constraint on relatively small loop closure periods, the AUV must move fast relative to the iceberg. This is similar to mapping bathymetry in the presence of ocean currents – the AUV must have a much

larger velocity than the current to maneuver properly and obtain good data [51].

The fourth assumption is necessary, since we are utilizing the shape of the iceberg and loop closures to determine where the AUV is relative to the iceberg. If we disregard iceberg calving, this is a reasonable assumption, since other effects that change the iceberg shape occurs at a much larger timescale than a mapping survey.

A. AUV/ICEBERG RELATIVE MOTION MODEL

The ice-relative AUV pose in the i-frame, $\eta^i_{rel} = \begin{bmatrix} x^i_{rel} & y^i_{rel} & \psi_{rel} \end{bmatrix}^\top$, can be expressed in terms of the AUV and iceberg poses in the n-frame by

$$\eta^i_{rel} = R^\top(\psi_{ice})(\eta^n_{auv} - \eta^n_{ice}), \tag{1}$$

where ψ_{ice} denotes the heading of the iceberg relative to the NED-frame and $R(\psi_{ice}) \in SO(3)$. By defining the relative velocity of the AUV in the b-frame as $\nu^b_{rel} = \nu^b_{auv} - \nu^b_{ice} \in \mathbb{R}^3$, the relative motion model can be expressed as

$$\dot{\eta}^i_{rel} = R(\psi_{rel})\nu^b_{rel} - S(r_{ice})\eta^i_{rel} + \omega_1, \tag{2}$$

$$\dot{\nu}^b_{rel} = \dot{\nu}^b_{auv} + S(r_{rel})R^\top(\psi_{rel})\nu^i_{ice} - R^\top(\psi_{rel})\dot{\nu}^i_{ice} + \omega_2, \tag{3}$$

where $S = -S^\top$ is a 3 DOF skew-symmetric matrix [52, Ch.2] with the rotational rate, r, about the z-axis as input. $\psi_{rel} = \psi_{auv} - \psi_{ice}$ is the relative heading between the AUV and the iceberg, and r_{rel} and r_{ice} are the relative rotational rate and the rotational rate of the iceberg, respectively; see Figure 2. ω_1 and ω_2 represent process noise, which is assumed to be zero-mean Gaussian (i.e. no bias). The AUV acceleration $\dot{\nu}^b_{auv}$ is typically captured by the kinetic model [52]

$$\dot{\nu}^b_{auv} = M^{-1}\left(\tau - C(\nu^b_{auv})\nu^b_{auv} - D(\nu^b_{auv})\nu^b_{auv}\right). \tag{4}$$

This can be included as part of the filter dynamics, or it can be taken as a signal from a separate onboard navigation system. Since a 3 DOF model is used in the estimator, the depth of the AUV is not estimated. Measurements of the depth are still used for processing topography measurements, however.

A pure kinematic model is sufficient for the iceberg, since we have assumed its velocity ν^i_{ice} to be constant. However, to account for slow variations in its acceleration, we model the acceleration as driven by a stochastic process according to

$$\begin{aligned} \dot{\eta}^n_{ice} &= R(\psi_{ice})\nu^i_{ice}, \\ \dot{\nu}^i_{ice} &= \omega_3, \end{aligned} \tag{5}$$

where $\omega_3 \in \mathbb{R}^3$ is zero-mean Gaussian white noise. By defining the state vector

$$x = \begin{bmatrix} \eta^i_{rel} & \nu^b_{rel} & \eta^n_{ice} & \nu^i_{ice} \end{bmatrix}^\top, \tag{6}$$

the state space representation of the motion model is described by

$$\dot{x} = f(x, u) + \omega$$

$$= \begin{bmatrix} R(\psi_{rel})\nu^b_{rel} - S(r_{ice})\eta^i_{rel} + \omega_1 \\ u + S(r_{rel})R^\top(\psi_{rel})\nu^i_{ice} - R^\top(\psi_{rel})\dot{\nu}^i_{ice} + \omega_2 \\ R(\psi_{ice})\nu^i_{ice} \\ \omega_3 \end{bmatrix}, \tag{7}$$

where $u = \dot{\nu}^b_{auv}$ and $\omega = \begin{bmatrix} \omega_1 & \omega_2 & 0 & \omega_3 \end{bmatrix}^\top$.

B. THE TOPOGRAPHY MAP

The topography map is represented by a grid map M with fixed size and resolution, where the size must be set according to the size bounds of the iceberg. The resolution depends on the type of MBE used, the number of beams, and the distance between the AUV and the mapped environment.

In the framework of a particle filter, let $\mathcal{M} \subset \mathbb{Z}^2$ be the set of indices belonging to the grid map, and let a topography estimate be defined by

$$\Lambda_{p_{id}} = \begin{Bmatrix} p_{id} & \xi & \Omega & t \end{Bmatrix}, \tag{8}$$

where p_{id} is the particle id that made the estimate, ξ is the information vector of the topography estimate, with corresponding information matrix Ω, and t is the timestamp when the estimate was made. From this, the topography map can be expressed as

$$M(i,j) = \begin{Bmatrix} \Lambda_{p_1} & \Lambda_{p_2} & \dots & \Lambda_{p_k} \end{Bmatrix}, \tag{9}$$

where $(p_1, p_2, \dots, p_k)$ is the set of particles that have made an update to grid element $(i,j) \in \mathcal{M}$. If no particle has made an update to element (i,j), then $M(i,j) = \varnothing$.

C. MOTION MODEL MEASUREMENTS

Measurements of a subset of the states of the relative motion model are required to reduce the dead-reckoning errors. We will assume that the AUV's orientation is available at each update, while the position is available sporadically. For simplicity, only the equations for the full position and orientation update are presented here. This gives the measurement model

$$\eta^n_{auv} = R(\psi_{ice})\eta^i_{rel} + \eta^n_{ice}. \tag{10}$$

To avoid large errors in the global position, the AUV can get acoustic positioning fixes (e.g. using LBL or USBL) with update rate depending on measurement system, availability, and navigation system accuracy. Further, the relative linear velocity can be assumed measured using an upwards-looking DVL when the AUV is below the iceberg. For simplicity, the velocity measurement is assumed available at every update. The relative rotation rate cannot be measured directly, but the rotational (yaw) rate of the AUV is assumed measured using an onboard inertial measurement unit (IMU). This gives

$$\begin{aligned} \begin{bmatrix} u_{rel} \\ v_{rel} \\ r_{auv} \end{bmatrix}^b &= \nu^b_{rel} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} R^\top(\psi_{rel})\nu^i_{ice}, \\ &= \nu^b_{rel} + diag\begin{pmatrix} 0 & 0 & 1 \end{pmatrix}\nu^i_{ice}. \end{aligned} \tag{11}$$

Note that we do not assume that the velocity of the AUV is measured, since this can be hard to accomplish if acoustic position updates are not periodically available (and of

sufficient quality). If the AUV velocity is available as well, it is straightforward to include these measurements, and that would improve the estimation of the iceberg velocity significantly. Finally, the position and orientation of the iceberg that are estimated in the SLAM algorithm are used to update the relative model. Since a particle filter has been chosen as SLAM implementation, the current pose of the iceberg will be determined from a weighted average of the particle poses

$$\eta^n_{ice} = \frac{\sum_{i=1}^{N_p} w_i \eta^n_{ice,i}}{\sum_{i=1}^{N_p} w_i}, \tag{12}$$

where N_p is the number of particles, and w_i is the particle weight of particle i. Thus, we now have the measurement vector

$$y = \begin{bmatrix} x^n_{auv} & y^n_{auv} & \psi_{auv} & u^b_{rel} & v^b_{rel} & r_{auv} & x^n_{ice} & y^n_{ice} & \psi_{ice} \end{bmatrix}^\top, \tag{13}$$

and by combining (10), (11), and the iceberg position measurement we get the combined measurement model

$$y = h_m(x) = \begin{bmatrix} R(\psi_{ice})\eta^i_{rel} + \eta^n_{ice} \\ v^b_{rel} + diag\begin{pmatrix}0 & 0 & 1\end{pmatrix} v^i_{ice} \\ \eta^n_{ice} \end{bmatrix}. \tag{14}$$

D. TOPOGRAPHY MEASUREMENTS

To perform loop-closures using the SLAM algorithm observations of the environment must be compared to a measurement model. This paper employs a method similar to the observation model used in BPSLAM presented in Barkby et al. [13]. MBE observations are modeled as sets of range (r), cross-track angle (α), and along-track angle (β), i.e., $\mathrm{z} = \begin{bmatrix}\mathrm{r} & \alpha & \beta\end{bmatrix}^\top$. The along-track angle is typically zero, unless the MBE is mounted with a tilt angle. The across-track angles can be constant or varying, depending on the MBE used. Adding the angles to the observation model also provide the possibility to model uncertainty caused by the size of the beam width (unlike laser range measurements, the MBE beams have a non-negligible footprint). The observation model for one beam observation is given by

$$\mathrm{z} = h_s(\mathrm{E}^i, p^i_{rel}, \Theta_{rel}) + \omega_{MBE}, \tag{15}$$

where h_s is the SLAM measurement function that uses the ensonified grid in the map, E^i, and the relative position and attitude, to calculate an expected observation. We define the relative position of the AUV in the NED-frame as $\mathrm{p}^n_{rel} = \begin{bmatrix}x^n_{rel} & y^n_{rel} & z^n_{rel}\end{bmatrix}^\top = \begin{bmatrix}x^n_{auv} & y^n_{auv} & z^n_{auv}\end{bmatrix}^\top - \begin{bmatrix}x^n_{ice} & y^n_{ice} & 0\end{bmatrix}^\top$, and the relative attitude as $\Theta_{rel} = \begin{bmatrix}\phi_{auv} & \theta_{auv} & \psi_{rel}\end{bmatrix}$. Note that we cannot use the relative AUV/iceberg state that is input to the SLAM algorithm directly, since the relative state will differ from particle to particle. Using (10), the relative state of the j^{th} particle can be expressed as

$$\begin{aligned} \eta^n_{rel,j} &= \hat{\eta}^n_{auv} - \eta^n_{ice,j}, \\ &= R(\hat{\psi}_{ice})\hat{\eta}^i_{rel} + \hat{\eta}^n_{ice} - \eta^n_{ice,j}, \end{aligned} \tag{16}$$

where $\hat{\cdot}$ denotes the estimated states input to the SLAM algorithm, and $\eta^n_{ice,j}$ is the iceberg position estimated by particle j. The ensonified grid can now be found with the following relation

$$\mathrm{E}^i = R^\top(\psi_{ice})\mathrm{p}^n_{rel} + R^i_b(\Theta_{rel})R^b_m(\Theta_{beam})\vec{\mathrm{k}}\mathrm{r}_m. \tag{17}$$

It is assumed that the roll and pitch is measured using the IMU, and the depth of the AUV is assumed available through pressure sensor measurements. $\Theta_{beam} = \begin{bmatrix}\alpha_m & \beta_m & 0\end{bmatrix}^\top$ is the measured beam angles (often fixed), and r_m is the beam range measured by the MBE. $\vec{\mathrm{k}}$ is the unit vector in the direction of the z-axis, and the sign of $\vec{\mathrm{k}}$ is positive if the MBE is mounted downwards, and negative if mounted upwards. It should be noted that the observations are used to determine what grid is ensonified in the measurement model. This simplification is disregarding the data association problem that arises by the uncertainties in the measurements. However, solutions that take this into account have been found to be too computationally complex [13]. The measurement function can now be expressed as

$$h_s(\mathrm{E}^i, p^i_{rel}, \Theta_{rel}) = \begin{bmatrix} \sqrt{b^2 + a^2 + d^2} \\ \arctan\left(\frac{a}{d}\right) \\ \arctan\left(\frac{b}{d}\right) \end{bmatrix}, \tag{18}$$

$$\begin{bmatrix} b \\ a \\ d \end{bmatrix} = R^b_i(\Theta_{rel})diag\begin{pmatrix}1 & 1 & sgn(\mathrm{k})\end{pmatrix}\left(\mathrm{E}^i - \mathrm{p}^i_{rel}\right). \tag{19}$$

For upwards-mounted MBE, the sign of the vertical component must be reversed, shown by the signum-function in (19). Note that some MBE models can output processed xyz-points, that is, they output $p_{mbe} = \begin{bmatrix}dx & dy & dz\end{bmatrix}^\top$, corrected for roll, pitch, sound speed, and ray bending. This is the case for the multibeam used by the HUGIN AUV, the Kongsberg EM2040. To utilize this processing, (17) can be modified to

$$\mathrm{E}^i = R^\top(\psi_{ice})\left(\mathrm{p}^n_{rel} + R(\psi_{auv})p_{mbe}\right). \tag{20}$$

E. PROBLEM STATEMENT

Suppose that an AUV equipped with an upwards-mounted MBE is mapping a drifting and rotating iceberg, using BPSLAM. Let $\hat{\eta}^i_{rel}$ and $\hat{\eta}^n_{ice}$ be the estimate of the AUV's relative pose in the i-frame and the estimate of the pose of the iceberg in the n-frame, respectively, while $\hat{v}^i_{ice}$ is the estimate of the iceberg drift velocity. The primary objective is to design an estimator, such that

$$\lim_{t\to\infty} |\hat{\eta}^i_{rel}(t) - \eta^i_{rel}(t)| = 0, \tag{21}$$

$$\lim_{t\to\infty} |\hat{\eta}^n_{ice}(t) - \eta^n_{ice}(t)| = 0. \tag{22}$$

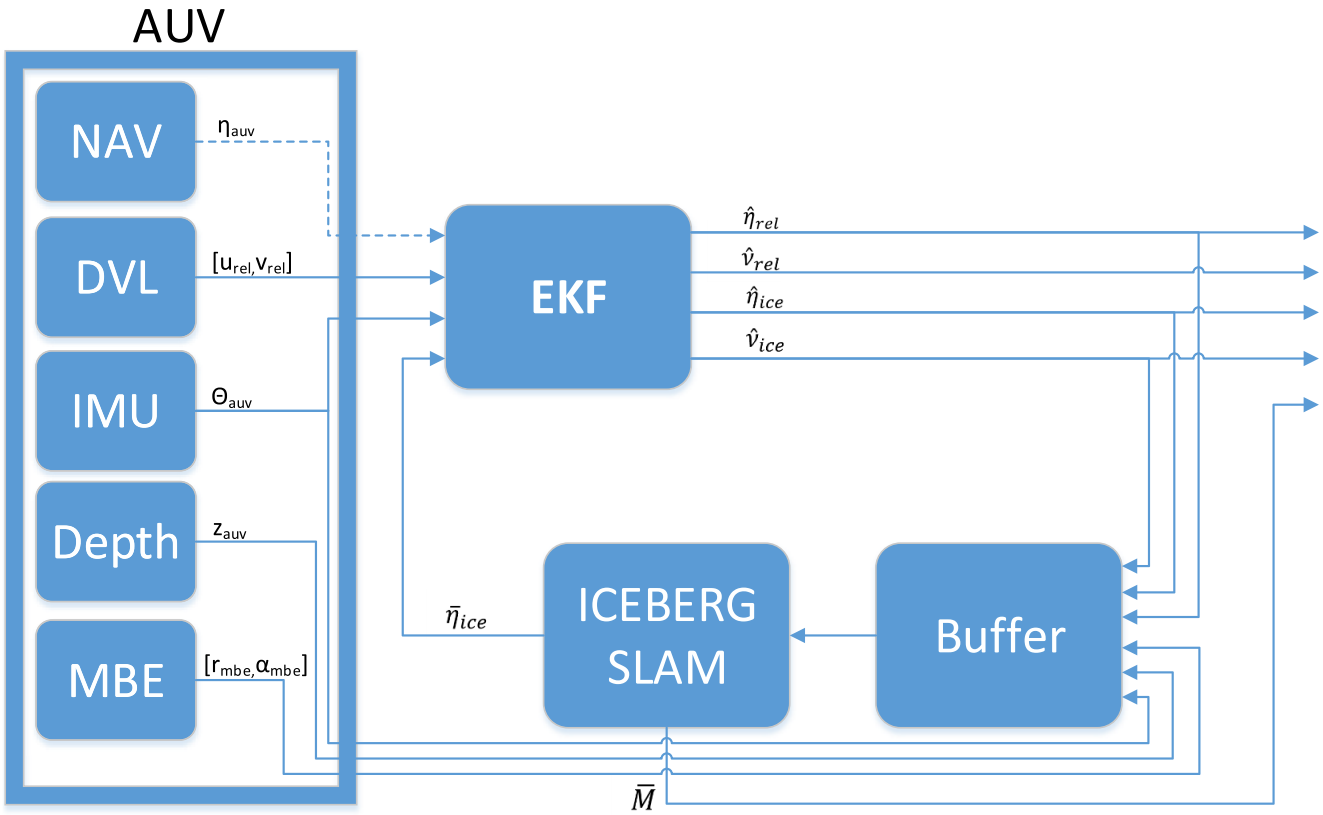

FIGURE 3. Blockdiagram of the iceberg mapping estimator.

The secondary objective is to estimate the drift velocity v^i_{ice} of the iceberg in the i-frame, such that

$$\lim_{t\to\infty} |\hat{v}^i_{ice}(t) - v^i_{ice}(t)| = 0. \tag{23}$$

The third objective is to estimate the topography of the iceberg in real-time in a fixed size grid map, such that

$$\lim_{t\to\infty} |\hat{M}(t) - M| = 0, \tag{24}$$

where $\hat{M}(t)$ is the map topography estimate.

IV. MOTION ESTIMATOR FOR ICEBERG MAPPING

The iceberg mapping and motion estimator presented in this paper has two layers. The bottom layer is the SLAM layer, which estimates the topography of the iceberg, as well as the iceberg's position in the NED-frame. The top layer consists of an EKF estimating the relative position and velocity between the AUV and the iceberg. The EKF uses the weighted average state from the SLAM algorithm from (12) as the measured iceberg pose, while the other measurements are from the AUV navigation system and the DVL. The velocity of the iceberg is also estimated in the top layer EKF. A block diagram of the estimator is shown in Figure 3.

A. RELATIVE MOTION ESTIMATOR

The relative motion estimator is implemented as a standard discrete EKF. The state of the EKF is defined in (6), and the state transition is given by (7), while the EKF predict equations are given by

$$\hat{x}_{k|k-1} = \hat{x}_{k-1|k-1} + \Delta_T f(\hat{x}_{k-1|k-1}, u_{k-1}), \tag{25}$$
$$P_{k|k-1} = F_{k-1} P_{k-1|k-1} F^\top_{k-1} + Q_{k-1}, \tag{26}$$

where Δ_T is the timestep, Q is the model covariance matrix, and P is the error covariance matrix. $F \approx I + \Delta_T \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1|k-1}}$ is the discretized Jacobian of (7). The EKF update equations are given by

$$K_k = P_{k|k-1} H^\top_{m,k} \left(H_{m,k} P_{k|k-1} H^\top_{m,k} + R_k\right)^{-1}, \tag{27}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \left(y_k - h_m(\hat{x}_{k|k-1})\right), \tag{28}$$
$$P_{k|k} = (I - K_k H_{m,k}) P_{k|k-1}, \tag{29}$$

where h_m is the measurement model defined in (14), and $H_{m,k} = \left.\frac{\partial h_m}{\partial x}\right|_{\hat{x}_{k|k-1}}$ is the Jacobian of h_m. R_k is the measurement covariance, K_k is the Kalman gain, and y_k is the measurement at timestep k, defined in (13).

B. ICEBERG BPSLAM

The SLAM algorithm chosen for the iceberg mapping problem is based on the BPSLAM algorithm presented by Barkby *et al.* in [11]–[13]. BPSLAM uses a scheme called distributed particle mapping (DPM) [37] to avoid repeated

map copy and delete operations, since these have a significant impact on computational complexity. By storing all particle estimates in one grid map, and tracking the particle ancestry in an ancestry tree, one map structure can be used for all particles. The BPSLAM method is an efficient, featureless RBPF SLAM algorithm.

Particle filters are especially suitable for featureless approaches due to its ability to provide multiple hypotheses. This means that the filter will at all times retain multiple possible iceberg poses, called particles, and all hypoteses will be evaluated against the observed environment to find the best fit. The overall steps in the BPSLAM algorithm is shown in Algorithm 1. Details of the BPSLAM algorithm can be found in [11]–[13], and previous work by the authors on the Iceberg SLAM algorithm can be found in [14]. The following section provides an overview of the Iceberg BPSLAM algorithm and details the modifications provided in this paper.

Algorithm 1 Bathymetric Distributed Particle Filter SLAM

1: **Initialize** each particle from initial distribution and add any prior information to the map.
2: **while** running **do**
3: **for** N_p **do**
4: **Propagate** by sampling from motion model.
5: **Weight** according to map agreement.
6: **Update** maps of particles.
7: **end for**
8: **if** $N_{eff} < \frac{N_p}{2}$ **then**
9: **Resample** based on importance weight.
10: **Prune** ancestry tree.
11: **Update** particle set.
12: **end if**
13: **end while**

1) PARTICLE PROPAGATION

The particle filter will at all times keep a set of N_p active particles

$$S_t = \begin{Bmatrix} \eta^n_{ice,1} & \cdots & \eta^n_{ice,N_p} \\ \mathrm{p}_{id,1} & \cdots & \mathrm{p}_{id,N_p} \\ \mathrm{w}_1 & \cdots & \mathrm{w}_{N_p} \end{Bmatrix}, \tag{30}$$

where p_{id} and w_k are the particle id and weight of the k^{th} particle. The set also contains the particle state vector, $\eta^n_{ice,k}$. At each timestep, each particle is propagated according to a proposal distribution

$$\dot{\eta}^n_{ice,k} = R(\hat{\psi}_{ice})\hat{v}^i_{ice} + \omega^n_{ice}, \tag{31}$$

where $\hat{v}^i_{ice}$ is the estimated iceberg velocity from the top layer estimator (see Figure 3). Equation (31) is similar to the iceberg motion model in (5), except that the SLAM filter does not estimate the iceberg velocity. Since the SLAM algorithm utilizes measurements of the iceberg topography to estimate the iceberg pose, it was deemed more suitable to have the velocity estimate in the top level estimator where measurements of relative velocity could be included. $\omega^n_{ice} \sim \mathcal{N}(0, \sigma^2_{v_{ice}}) \in \mathbb{R}^3$ is driving the particle filter with variance $\sigma^2_{v_{ice}} \in \mathbb{R}^3$. The variance should be on the order of the expected iceberg velocity variations, plus measurement noise and dead-reckoning errors.

2) PARTICLE WEIGHTING

To assess whether or not a particle is a good estimate of the iceberg pose, each particle is evaluated against the map contained in the particle filter. Obviously it will not be possible to assess if a particle is good or bad if no prior map exist. If the observations contain areas overlapping the previously mapped terrain (either fully or partially), it is possible to determine how well the observations fit with the expected results from the map, by using the sensor model described in (18). Three different methods for weighting the measurements are discussed - the original BPSLAM method from [13] (converted to log-space for numerical stability); a modified version of the BPSLAM method; and an ICP based method.

a: BPSLAM WEIGHTING

Using the log-likelihood function for normal distributions for beam j gives

$$\ln\left(P\left[\left(\hat{E}_{z,j} - \bar{E}_{z,j}\right) = 0\right]\right) = -\frac{1}{2}\frac{\left(\hat{E}_{z,j} - \bar{E}_{z,j}\right)^2}{\sigma^2_{\hat{E}_{z,j}} + \sigma^2_{\bar{E}_{z,j}}} - \ln\left(\sqrt{2\pi\left(\sigma^2_{\hat{E}_{z,j}} + \sigma^2_{\bar{E}_{z,j}}\right)}\right), \tag{32}$$

where $j \in \mathcal{W}_k \subset \mathcal{Z}^+$ is an observation index in the set of all valid observations for a given particle. $\hat{E}_{z,j}$ and $\sigma^2_{\hat{E}_{z,j}}$ are the estimated topography and its corresponding variance estimate, which are stored per grid for all particles that have made an update to that particular grid. $\bar{E}_{z,j}$ can be found using (17) and $\sigma^2_{\bar{E}_{z,j}}$ can be estimated through backward transport using [13],

$$\sigma^2_{\bar{E}_z} = \left(H^\top_{s,k} R^{-1}_{obs} H_{s,k}\right)^{-1}, \tag{33}$$

where $H_{s,k}$ is the Jacobian of the measurement function in (18), and $R_{obs} = diag\left(\sigma^2_r\ \sigma^2_\alpha\ \sigma^2_\beta\right)$ is the covariance of the observations. A weight will only be calculated for a given observation if it ensonifies a grid already containing a topography estimate, i.e. it belongs to the set of valid observations $\mathcal{W}_k$.

The particle weight can be calculated from the joint likelihood of the beam weights

$$\begin{aligned} \mathrm{w}_k = f_{z_k|x_k,M} &= \prod_{\forall j \in \mathcal{W}_{s,k}} P\left[\left(\hat{E}_{z,j} - \bar{E}_{z,j}\right) = 0\right] \\ &= \sum_{\forall j \in \mathcal{W}_{s,k}} \ln\left(P\left[\left(\hat{E}_{z,j} - \bar{E}_{z,j}\right) = 0\right]\right). \end{aligned} \quad (34)$$

where $\mathcal{W}_{s,k} \subseteq \mathcal{W}_k$ is a subset of the indices of the beam weights determined by sampling $N_{min} = \min(|\mathcal{W}_k|)$ for $k = 1 \ldots N_p$ random indices from $\mathcal{W}_k$ into $\mathcal{W}_{s,k}$ with equal probability. The subset $\mathcal{W}_{s,k}$ will be different for each particle, since the set $\mathcal{W}_k$ will differ. This means that only N_{min} of the beam weights will be used to determine the particle weight. The particles will have differing number of valid weights (due to different number of overlapping observations). We can define a criterion for including a particle in the resampling step by setting a minimum overlap, γ. By for instance setting $\gamma = 50\%$, a minimum of 50% of the ensonified grids must contain a previous estimate for the particle to be included in the resampling step. Note that this is not the same as saying 50% of the MBE beams must produce a valid weight, since different beams may ensonify the same grid. If a particle is not included in the resampling process it will not be assigned a weight, and it cannot be removed or spawn new particles during resampling.

b: MODIFIED BPSLAM WEIGHTING

The original BPSLAM method process measurements sequentially, i.e. each new MBE swath initiates a full particle filter step (see Algorithm 1). Since resampling and prune are relatively computationally expensive, a method that allows measurements to be buffered and processed batch wise (multiple swaths at the time) was desired. Furthermore, batch processing of swaths will allow adaptive sampling rate in the SLAM algorithm, which can be important to ensure real-time operation during run-time transients. Therefore, a new weighting method for the BPSLAM algorithm is proposed, based on the work by Hagen *et al.* in [25]. First, the beam likelihood is calculated as in (34), but with $\mathcal{W}_{s,k} = \mathcal{W}_k$, and then modified according to

$$\mathrm{w}_k = f_{z_k|x_k,M}^{\frac{\alpha(x_k)}{m(x_k)}}, \quad (35)$$

where $0 < \alpha(x_k) < 1$ is a measure of the actual terrain information, which depends on terrain variation, map noise, and sensor noise (see [25] for details). This modifier makes the algorithm more robust, expecially in segments with little terrain variations [25]. The modifier $m(x_k)$ represents the number of grid points supported at x_k, allowing use of all available beam information, rather than only the minimum number of beams as in the original BPSLAM method. Note that this will also improve the run-time for the algorithm for parallel execution, since calculating N_{min} requires thread synchronization in a practical implementation, which produces significant overhead.

c: PARTICLE FILTER ICP WEIGHTING

A method that is frequently used for EKF SLAM is the iterative closest point method for matching point clouds. This weighting method is proposed as an alternative to the two BPSLAM weighting methods, and is also developed with the intention of batch processing MBE measurements to allow real-time execution. The details of the ICP algorithm is outside the scope of this paper, but the ICP implementation has been taken from the source code provided by Bouaziz *et al.* [53], and an overview of the ICP algorithm can be found in [53] and references therein.

Let $E_{j,k}^i \in \mathcal{E}_k$ be observation j of particle k calculated from (17), where the set $\mathcal{E}_k$ represents the set of all observations at the current timestep, referred to as the patch. The output of the ICP algorithm is a rigid body transformation $T(d, \Theta)$ transforming the input data points d (our observations) to a set of model points M (our map). The rigid body transformation is calculated by minimizing a cost function. The particle weight can now be calculated from [54]

$$f_{z_k|x_k,M} = e^{\sum_{\forall j \in \mathcal{E}_k} \|E_{j,k} - T_k(E_{j,k}, \Theta_j)\|^2}, \quad (36)$$

$$\mathrm{w}_k = f_{z_k|x_k,M}^{\frac{1}{m(x_k)}}, \quad (37)$$

where the modifier $m(x_k)$ is the same as in the modified BPSLAM method.

The ICP weighting algorithm does not take the beam and grid uncertainty into account, like the two former methods, but provides a best fit between measurements and map points. Altering the ICP method to account for this uncertainty could be an interesting extension of the algorithm in further work.

3) MAP UPDATE

After weighting each particle, the map must be updated. Since we assume a static topography, the predict step can be omitted in the filter estimating the topography. By following the BPSLAM methodology [13] and by using the dual form of EKF, the Extended information filter (EIF), the update equations can be formulated as

$$\tilde{\mathrm{E}}_z = \tilde{\Omega}^{-1}\tilde{\xi}, \quad (38)$$

$$\Omega = \tilde{\Omega} + H_s^\top R_{obs}^{-1} H_s, \quad (39)$$

$$\xi = \tilde{\xi} + H_s^\top R_{obs}^{-1}\left[\mathrm{z} - h_s(\mathrm{E}^i, p_{rel}^i, \Theta_{rel}) + H_s\tilde{\mathrm{E}}_z\right], \quad (40)$$

where $\xi \in \mathbb{R}$ is the topography estimate vector in information form, $\Omega \in \mathbb{R}$ is the information matrix, and E_z is the topography estimate of the selected grid. The notation $\tilde{\cdot}$ indicates a priori estimate since time subscript has been omitted.

4) PARTICLE RESAMPLING

In a particle filter, resampling is necessary to achieve convergence, but it can also be dangerous since it limits the memory of the filter and can potentially lead to situations where important particles are removed (particle depletion). Since the goal of resampling is to remove unlikely particles, while keeping particles that have good correspondence with the map, it only

makes sense to perform resampling if the particle weights differ significantly. A method to accomplish this is to define an *effective particle size* as presented in [13] and [55],

$$N_{eff} = \frac{1}{\sum_{i=1}^{N_r} \bar{w}_i^2}, \tag{41}$$

where $\bar{w}_i$ is the normalized weight for particle i. Resampling is only performed when $N_{eff} < N_r/2$, where N_r represents the number of particles eligible for resampling.

If the effective particle size is small enough and resampling is allowed to continue, the resampling step is performed according to the principle of Sequential-Importance-Resampling (SIR), where the normalized particle weight defines the probability of a particle being drawn into the new particle set. If a certain particle with id equal to j is drawn, it is inserted into the ancestry tree as a child of particle j and assigned a new particle id. The systematic resampling algorithm has been selected since this method has been shown to provide smaller variance and lower computational cost than the other commonly used resampling strategies [56].

C. SUMMARY OF ESTIMATION ALGORITHM

To wrap up the section describing the developed estimator, a short summary is provided. The reader is referred to Section III-E providing the problem statement, and Figure 3 providing a graphical overview of the estimator.

The top layer estimator is a standard EKF, running at high rate to continuously provide updated estimates to e.g. a guidance system. The top layer provides estimates of the relative position and velocity between the AUV and the iceberg, as well as estimates of iceberg position and velocity (see state vector in (6)), propagated according to (7). The inputs to the EKF are the measurements in (13), where the AUV pose and angular rate, as well as the relative velocity, are external signals, while the iceberg pose is an output of the bottom layer SLAM algorithm.

The input to the bottom layer is the relative pose and velocity from the top layer, which are used to propagate the particles according to (31). At each SLAM step, Algorithm 1 is executed, before a new SLAM estimate is generated according to (12), which is used to update the top layer estimator. In between SLAM steps, all MBE measurements are timestamped (along with the relative position, attitude, and velocity) and buffered. This architecture allows adaptive SLAM timesteps, which will handle run-time transients, as long as the average run-time is well below the real-time constraints.

V. ARCTIC AUV SIMULATOR

Performing AUV operations in the Arctic are expensive and risky, and all new methods must be thoroughly tested before being implemented on real systems in the field. Therefore, a numerical model capable of simulating the desired environment is required. This section presents part of the simulator used in conjunction with the work presented in this paper.

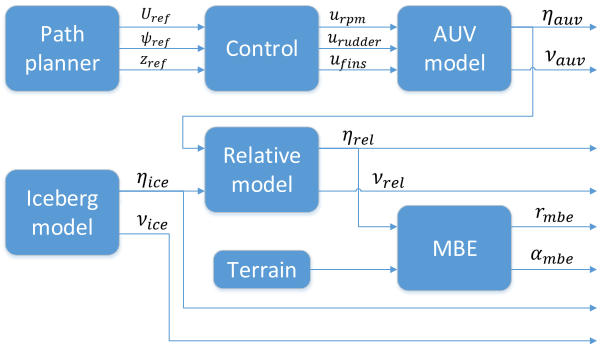

FIGURE 4. Simplified blockdiagram of the Arctic AUV simulator.

A block diagram of the simulator can be found in Figure 4. An earlier version of the simulator has been presented in [49].

A. AUV MODEL

To analyze the behavior of the developed method under realistic AUV motions, a 6 DOF model is used. The measurements from the MBE sensor are greatly affected by AUV motions like rolling and pitching, since the sonar head is fixed to the the AUV body. The targeted AUV during the simulator development is the REMUS 100 AUV [57]. The model parameters have been taken from a REMUS 100 model presented in [58].

The general 6 DOF equations of motion of a marine craft can be written on vectorial form [52]

$$\dot{\eta} = J_\Theta(\eta)\nu, \tag{42}$$

$$M\dot{\nu}_r + C(\nu_r)\nu_r + D(\nu_r)\nu_r + g(\eta) = \tau + \tau_{env}, \tag{43}$$

where $\eta = \left[p^n_{auv}\ \Theta_{nb}\right]^\top$ is the position and orientation vector of the AUV. $p^n_{auv} \in \mathbb{R}^3$ denotes the position of the AUV in the n-frame, while $\Theta_{nb} \in \mathcal{S}^3$ is a vector of Euler angles. $\nu \in \mathbb{R}^6$ contains the linear and angular velocities of the AUV, expressed in the b-frame, and τ, $\tau_{env} \in \mathbb{R}^6$ are the forces and moments acting on the AUV in the body-fixed frame from the control surfaces and the environmental loads, respectively. $\nu_r = \nu - \nu_c \in \mathbb{R}^6$ is the relative velocity vector, where ν_c is the velocity vector of the ocean currents. In the model we have assumed irrotational ocean currents.

Equation (43) defines the kinetics of the AUV, while the velocity transformation from the b-frame to the n-frame is expressed in (42). $M = M_{RB} + M_A \in \mathbb{R}^{6\times6}$ is the rigid-body inertia and added mass of the AUV, while the centripetal and Coriolis rigid-body and added mass are denoted $C(\nu) = C_{RB}(\nu) + C_A(\nu_r) \in \mathbb{R}^{6\times6}$. $D(\nu_r) \in \mathbb{R}^{6\times6}$ represents hydrodynamic damping, and restoring forces are given by $g(\eta) \in \mathbb{R}^6$. The coefficients for these matrices can be found in [58]. For more details about the simulator, the reader is refered to [49].

B. BEAM RANGE SIMULATOR

The beam range simulator used in conjunction with the AUV dynamics has been developed by Holsen in [59] and extended by the authors, and a description of the beam simulator will

be given in the following. The vector describing the direction of each beam is given by

$$\frac{r^n_{MBE,i}}{|r^n_{MBE,i}|} = R^n_b(\Theta_{nb})R^b_{MBE,i}(\Theta_{bMBE,i})\vec{k}, \tag{44}$$

where $r^n_{MBE,i} \in \mathbb{R}^3$ is the position vector of the i^{th} beam. $\Theta_{bMBE,i} \in \mathcal{S}^3$ is the angles of the beams expressed relative to the b-frame, $R^b_{MBE,i}(\Theta_{bMBE,i})$ is the rotation matrix between the i^{th} beam and the b-frame, and $\vec{k} = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^\top$.

The range of each beam is estimated through an iterative procedure. Let e_i be the current estimate of the i^{th} beam range. We calculate the beam-end position, $p^n_{MBE,i} = \begin{bmatrix}x_{MBE,i} & y_{MBE,i} & z_{MBE,i}\end{bmatrix}^\top$ from

$$p^n_{MBE,i} = p^n_{AUV} - \frac{r^n_{MBE,i}}{|r^n_{MBE,i}|}e_i. \tag{45}$$

Since the mapped surface is above the AUV, (45) has a negative sign. If the AUV is performing bathymetric mapping, the sign in (45) must be reversed. The surface, which in this case represents the water surface or the iceberg, is represented as a 3D digital terrain map, where each entry in the matrix has a north-, east-, and depth-coordinate. To get the depth at a given position, bilinear interpolation between the four nearest neighbours is used, resulting in a depth at a given position, $Z_{surface}(x, y)$. The error between the beam-end position and the surface is

$$e_{Z,MBE} = z_{MBE,i} - Z_{surface}(x_{MBE,i}, y_{MBE,i}). \tag{46}$$

If the error is positive, the beam range is too small, while with a negative error the beam range is too large. The beam range is increased by a constant increment until the error becomes negative, then binary search is used to reduce the error to a sufficient accuracy. Binary search could have been used for the whole search; however, this has shown to cause the beams to hit the wrong surface (beams hitting the water surface when they should have been hitting the iceberg) under some conditions.

VI. RESULTS

The following section presents an assessment of the performance of the iceberg mapping estimator detailed throughout this paper. A real iceberg topography, from the PERD iceberg database [60] is used in a simulation study. The selected iceberg, no. R11i01, is a wedged iceberg with dimension 160 by 135 meters, a sail height of 31 meters, and a draft of 110 meters. The results highlights the estimators performance with regard to the problem statement in Section III-E. A run-time analysis is also presented to evaluate the algorithms real-time potential, to assess the feasibility of using it in closed-loop with an active guidance algorithm.

In order to verify the performance of the algorithm, we first test it on a static seabed dataset. This is acquired by the HUGIN HUS AUV equipped with an EM2040 MBE [15] in November 2017 in the Trondheimsfjord in Norway. The EM2040 has 400 beams spread out over a varying swath sector, and can output processed xyz-points, corrected for AUV roll and pitch, sound speed, and ray bending. The objective of this preliminary test is to verify that the SLAM algorithm works well on a standard seabed bathymetry test case. The processed dataset is depicted in Figure 5(c). The bathymetry is collected from an area with large variations in topography to be comparable with mapping of an iceberg.

A. CASE STUDY: STATIC REAL-WORLD BATHYMETRY

The first results presented are the bathymetric field tests. Figure 5(d) shows the AUV navigation system trajectory (real-time solution) in solid black and the processed offline NavLab-solution in solid red. NavLab is a post-processing tool developed by the Norwegian Defence Research Establishment and provides a good estimate of the ground truth [61]. The NavLab estimate of the AUV trajectory is deduced by merging all available information from sensor measurements and mathematical error models through a complex post-processing estimator [61]. Optimal smoothing is also applied to the estimates, since all data, both before and after the current timestep, are available. If we compare the AUV navigation with the NavLab-trajectory in Figure 5(d) and look at the AUV navigation error in Figure 5(b) we see that the AUV navigation has been quite poor during the survey, with a maximum error norm $e = \sqrt{p_x^2 + p_y^2}$ of 36.40 meters, even with the aid of ship-mounted USBL. The large changes in the AUV navigation (especially on the left side of the figure) are caused by new USBL position fixes (noisy) entering the navigation solution. In the results presented here, we only study the SLAM algorithm, and thus, the AUV navigation has not been included in the SLAM results (i.e., pure dead-reckoning).

The SLAM trajectories are shown in Figure 5(d), and its corresponding error when compared to the NavLab-trajectory is shown in Figure 5(b). The errors for the SLAM methods are all greater than 10 meters, but the trajectories for the BPSLAM and the modified BPSLAM are similar, with maximum errors of 13.32 and 12.93 meters, respectively. The BPSLAM method suffers from particle depletion, as can be seen from Figure 5(a). This happens when the particle standard deviation in both directions are reduced to zero. The standard deviation tells us about the spread of the particles in a given direction, but a low standard deviation is not always desirable since this also reflects the available hypotheses of the particle filter. Particle depletion is when all particles stem from one or very few particles after a resampling, i.e. a low number of hypotheses. If neither of these particles reflect the correct state, it will be hard for the particle filter to recover. The 3σ-bound shown in Figure 5(a) should therefore ideally reduce to a limit defined by the measurement and map uncertainty, and map resolution. This seems to be the case for the ICP and the modified version. A sudden decrease in the 3σ-bound happens upon resampling, where unlikely particles are resampled from more likely ones. The modified BPSLAM

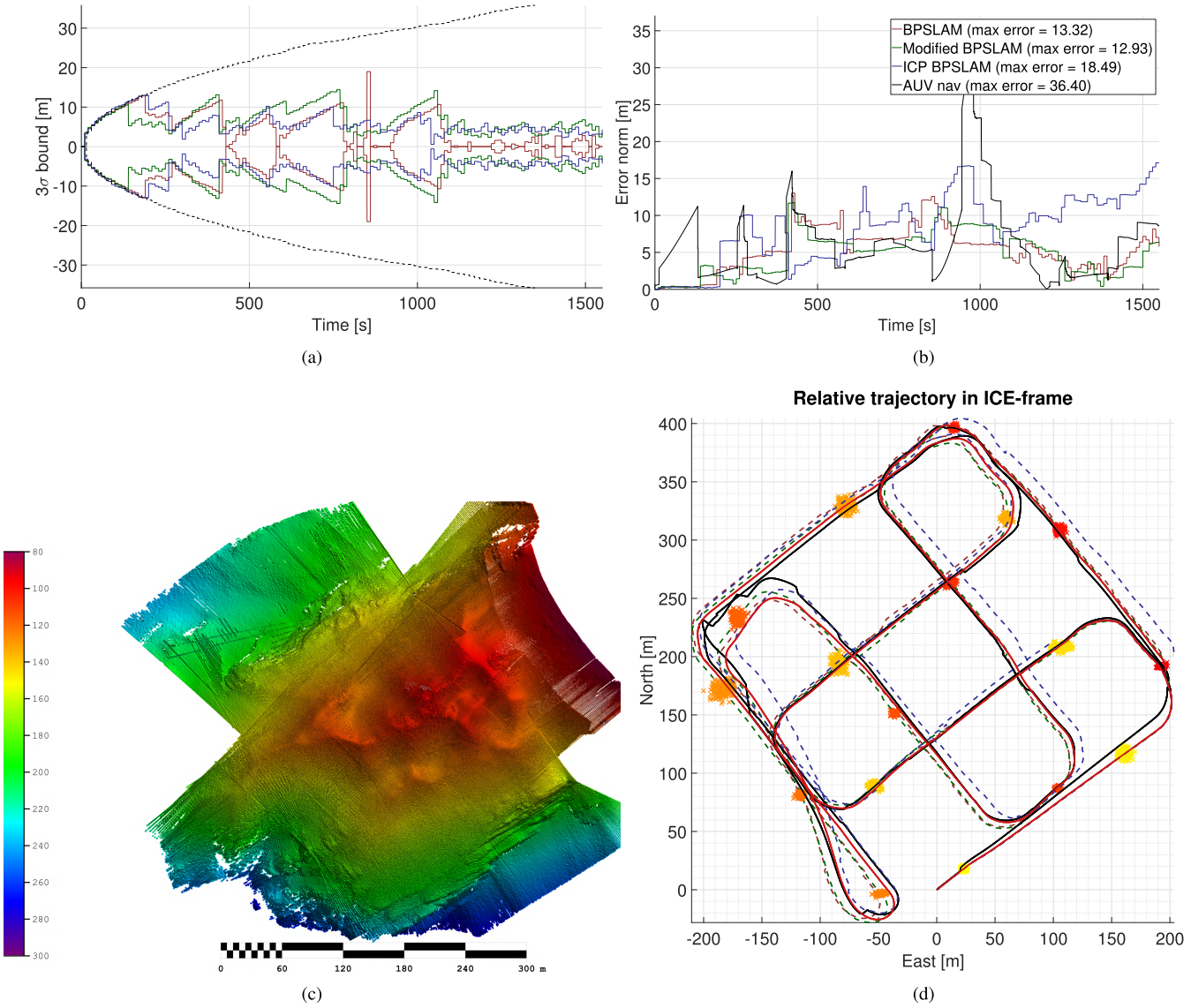

FIGURE 5. HUGIN dataset: SLAM pose estimation results. a) Particle cloud 3σ-bound for SLAM estimate of HUGIN north position. Black line shows particle cloud 3σ-bound with no resampling. b) Shows the error norm of the north and east position for the average HUGIN pose from the particle cloud compared to the NavLab solution. c) Processed bathymetry from the HUGIN AUV. d) HUGIN trajectory plot with particle cloud snapshots from the modified BPSLAM algorithm and trajectories from all SLAM methods and the AUV navigation system (including the NavLab-solution in solid red color).

method is more robust to the particle depletion problem. The error for the ICP method is larger, which is believed to stem from convergence problems in the ICP algorithm (this method aborts after a given number of iterations). Due to the inaccurate AUV navigation, we cannot say anything about the expected error using the SLAM method, but all methods show significant improvement when compared to the real-time solution aided by USBL.

B. CASE STUDY: ICEBERG MAPPING

In the simulations presented in this section, the iceberg is drifting with a constant speed and a constant rotational velocity. Yulmetov *et al.* [50] study the drift of multiple icebergs using trackers, and report a mean iceberg speed in the range $0.08 - 0.28$ m/s, and a maximum iceberg speed of $0.41 - 1.66$ m/s. Further, the authors report a mean iceberg rotational rate of $1 - 2$ revolutions per 24 hours, which is about $15 - 30$ degrees/hour. They also report extreme rotational rates (of short durations) in the order of more than 200 degrees/hour. Based on this information the drift speed of the iceberg was set to 0.3 m/s with a course angle of 45 degrees (relative to north). The standard deviations for the noise driving the particle filter in (31) was set to 1.0 m/s for the linear velocity and to 200 degrees/hour for the rotational velocity.

The multibeam is configured with the Imagenex DeltaT multibeam in mind, with 120 beams spread equally over a sector of 120 degrees, and 3 degrees beamwidth. The range resolution for the DeltaT multibeam is 0.02% [62], but to account for uncertainties in beam angle, and iceberg

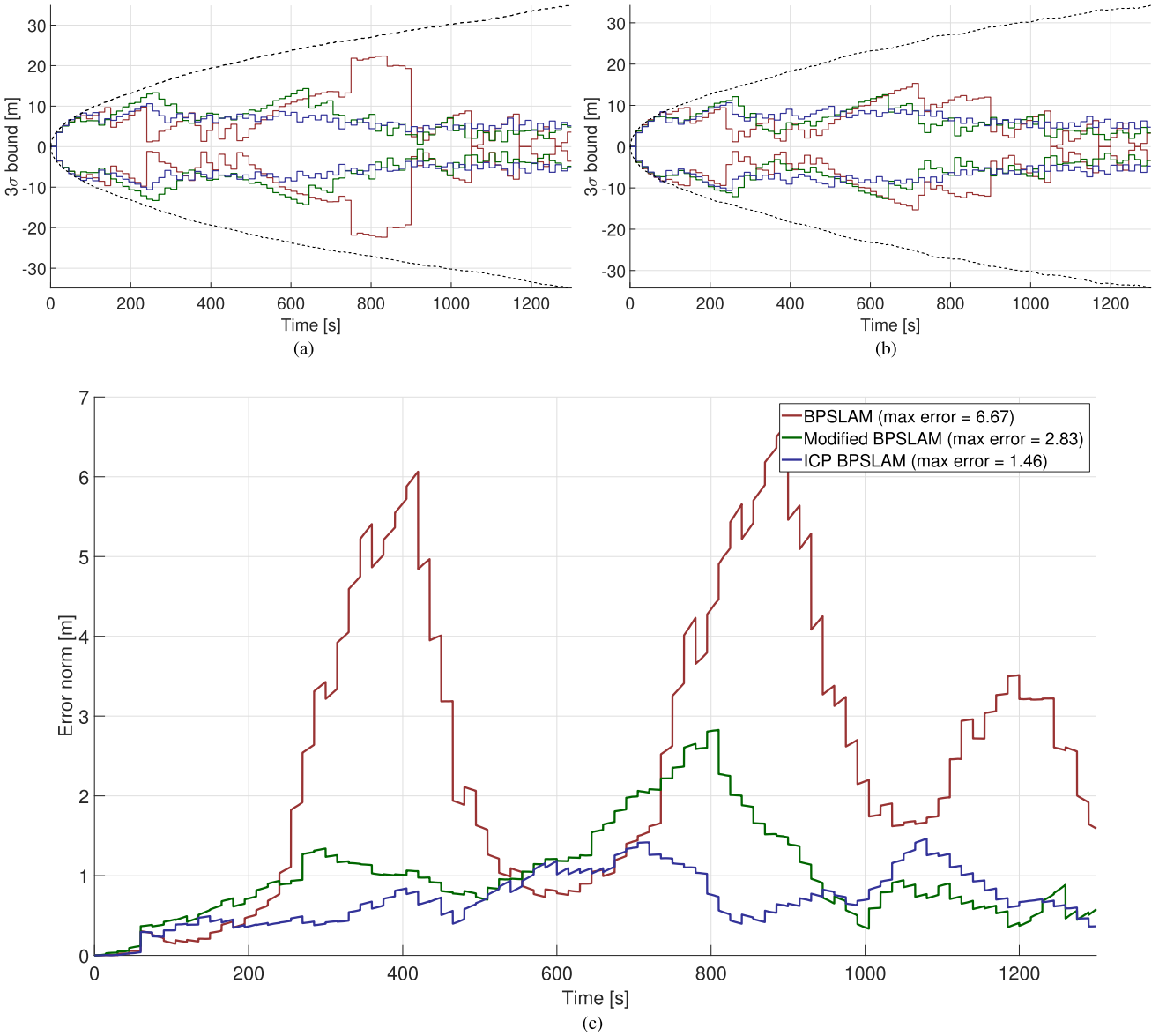

FIGURE 6. Case 1: SLAM and EKF iceberg pose estimation results. a) Particle cloud 3σ-bound for SLAM estimate of iceberg north position. Black line shows particle cloud 3σ-bound with no resampling. b) Particle cloud 3σ-bound for SLAM estimate of iceberg east position. c) Shows the error norm of the north and east position for the iceberg pose estimate in the top level EKF estimator.

roll and pitch, a higher noise level of 0.5% was chosen for the range measurements. Therefore, the standard deviation for the multibeam is set to $\sigma_r = 0.5$ meters (a range of 100 meters is assumed), and the standard deviation for the multibeam cross-track and along-track error is set to half of the beamwidth. In the SLAM algorithm, the map size was set to 225 meters, and the resolution was set to 1.0 meter. Further, the patch size is set to 15 seconds, which means that at each SLAM timestep, a total of 150 MBE swaths will be processed (the MBE is running at 10 Hz). The number of particles used is 200, unless otherwise specified.

1) CASE 1 - ICEBERG WITH LINEAR DRIFT

This case serves as a baseline for the estimator, and no rotation has been applied. The intial linear velocity is also assumed to be known. The results from this case is shown in Figure 6. From Figure 6(c), it is clear that the ICP version of BPSLAM outperforms the other methods in term of relative position error, which in the ICP SLAM case has a maximum error norm of 1.46 meters, while the original and the modified version has 6.67 and 2.83 meters, respectively. The improvement does come at a cost, which is discussed in Section VI-C. The error of the modified BPSLAM converges to about the same error as the ICP method at the end of the simulation, but is shown to consistently result in a larger maximum error compared to the ICP method through multiple simulations. From the particle cloud 3σ-bound, shown in Figure 6(a) and 6(b), it can be seen that the original BPSLAM method has the largest variations in particle cloud standard deviation. We can also see that the particle cloud

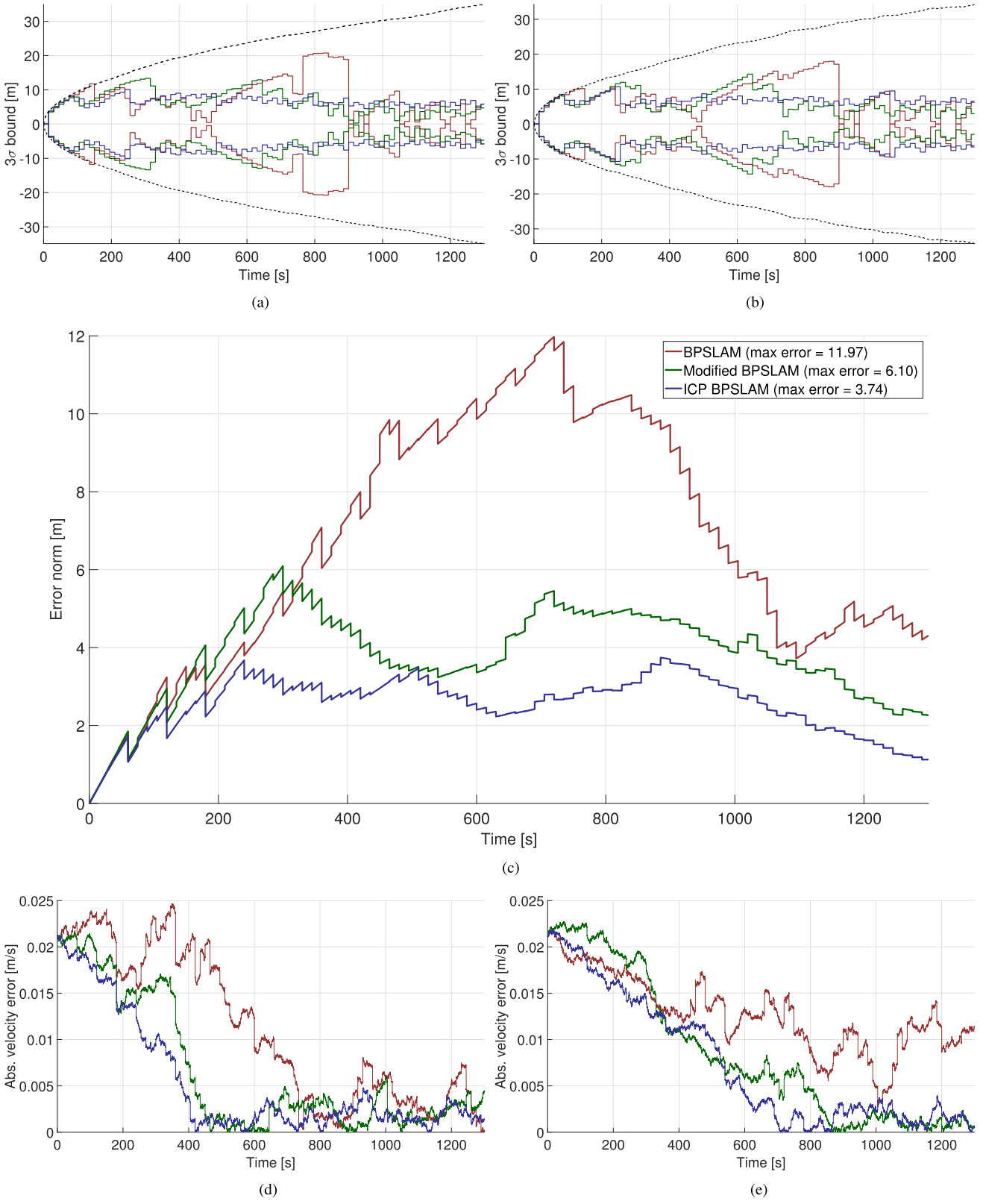

FIGURE 7. Case 2: SLAM and EKF iceberg pose estimation results. a) Particle cloud 3σ-bound for SLAM estimate of iceberg north position. Black line shows particle cloud 3σ-bound with no resampling. b) Particle cloud 3σ-bound for SLAM estimate of iceberg east position. c) Shows the error norm of the north and east position for the iceberg pose estimate in the top level EKF estimator. d) and e) Iceberg velocity error in the north and east direction, respectively.

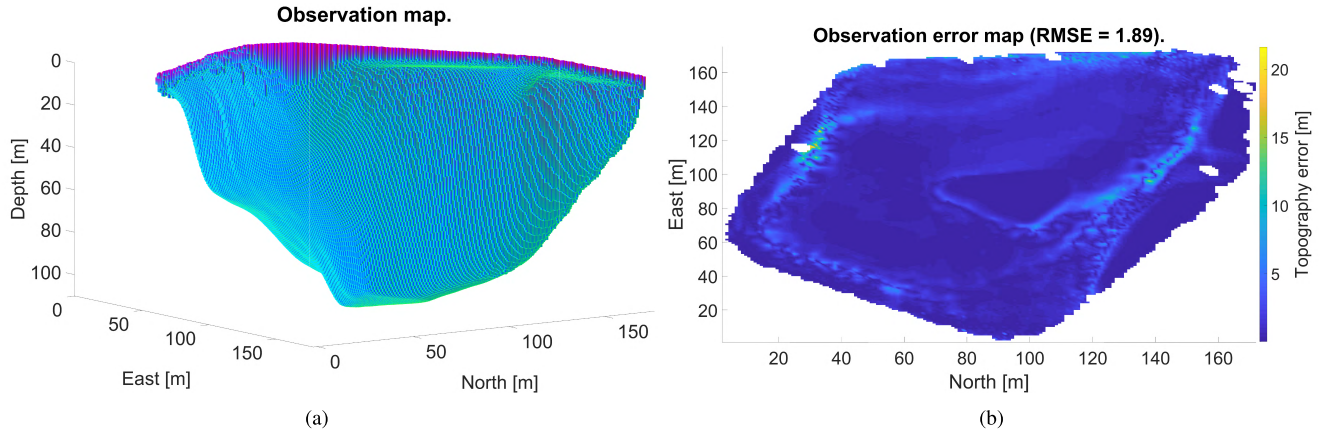

FIGURE 8. Iceberg topography map extracted from the SLAM algorithm. a) Shows the topography extracted from the particle with the highest weight at the end of the simulation. b) Topography error map showing the error when compared to the real topography.

standard deviation goes to zero, or close to zero, multiple times. This is particle depletion, and if we look at the error at the time of depletion, we see that the estimate is not correct – resulting in a situation that is hard to recover from. The reason for the particle depletion is that some observations are unjustly weighted much lower than others. Since beam weights are selected by random sampling when only partial overlap is achieved, this can lead to a larger variation in particle weights than it should be. In the modified BPSLAM, this effect is reduced by including all beams and adjusting the weight with the modifier $1/m(x)$. It is believed that the effect seen with the original BPSLAM method will be reduced for applications where full overlap is required. It should also be noted that the original BPSLAM is only utilizing 1 out of 150 swaths for determining resampling since the SLAM algorithm is running once every 15 seconds. Running the algorithm on every MBE swath amplified the problem, leading to severe particle depletion and estimator divergence.

Figure 8(a) illustrates the estimated iceberg topography collected from the particle with the highest weight at the end of the simulation. Figure 8(b) shows the observation map error when compared to the actual map. The error is larger in areas with high topography gradient, due to the fixed resolution of the map and since the topography estimation is sensitive to small position errors in these areas. The overall root-mean-squared error (RMSE) over all the grids that contain a measurement is 1.89 meters for the simulation in Case 1 with the ICP method.

2) CASE 2 - ICEBERG WITH LINEAR DRIFT AND UNCERTAIN INITIAL VELOCITY

In Case 2 the simulation parameters were the same as in Case 1, but initial drift speed and direction were assumed to be uncertain and were set to be 10% off from the actual values. Figure 7(d) and 7(e) illustrate the velocity estimate errors. The estimates converge, but rather slowly, since the states have been assumed to be constant. We can also see that the velocity estimate converges faster for the north-direction than for the east-direction. This is believed to stem from the AUVs survey direction, which is mainly in the east-west during the first 650 seconds, and north-south during the rest of the simulation (see trajectory plot in Figure 10).

In Figure 7(c) we can see that the maximum error is larger for all methods, but the position estimates in the ICP and the modified version converge to within a few meters once the velocity estimates have converged.

3) CASE 3 - ICEBERG WITH LINEAR AND ROTATIONAL DRIFT AND UNCERTAIN INITIAL VELOCITY

Case 3 studies how iceberg rotation affects the estimates, and the results are depicted in Figure 9. A rotational rate of 30 degrees/hour was applied to the iceberg, with initial uncertainty of 10%. Figures 9(a) - 9(c) show similar results as seen in Case 2, but again with slightly larger maximum errors. Figures 9(d) - 9(e) display particle cloud heading 3σ-bound and the rotational rate estimate error, respectively. From the plot of the heading 3σ-bound it is clear that it is difficult to estimate the rotation of the iceberg. The estimate of the rotational rate further supports this claim, since the estimates have troubles with converging. It looks like the original BPSLAM estimate converges, but multiple simulations have shown this to be a coincidence. The rotation estimate for the ICP SLAM is better, but from Figure 9(d) we see that the 3σ-bound is not significantly reduced.

Figure 10 shows the relative trajectory and the NED-trajectory during the simulation of Case 3. Figure 10(a) illustrates snapshots of the particle cloud (projected from the iceberg origin to the relative AUV position for the illustration). We see that the state estimate get a steady state error, which is due to the inaccurate velocity estimates at the start of the simulation.

C. RUN-TIME ANALYSIS

The run-time of a SLAM algorithm is always an important criteria, and especially important when the algorithm is intended for use with a guidance algorithm. Figure 11

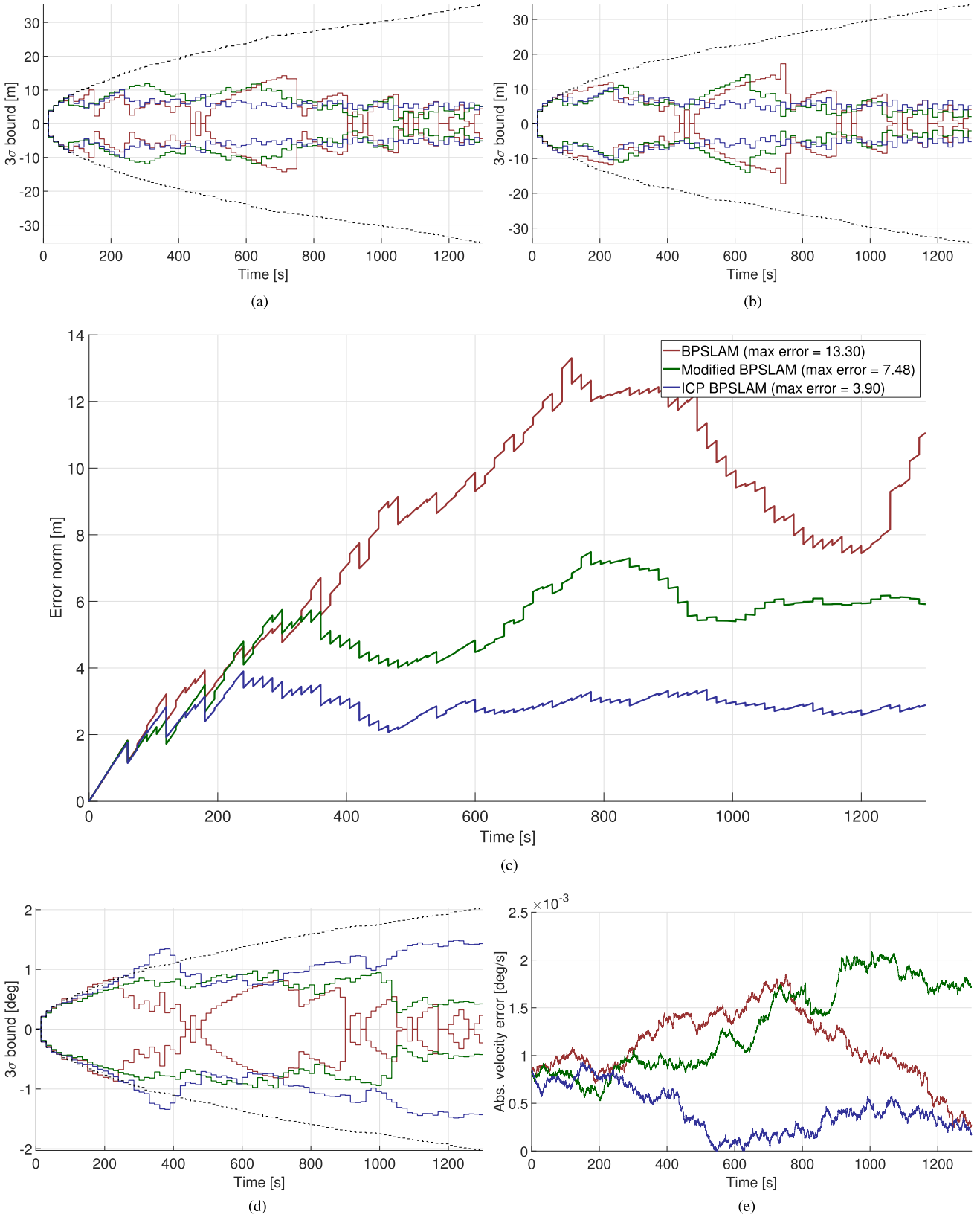

FIGURE 9. Case 3: SLAM and EKF iceberg pose estimation results. a) Particle cloud 3σ-bound for SLAM estimate of iceberg north position. Black line shows particle cloud 3σ-bound with no resampling. b) Particle cloud 3σ-bound for SLAM estimate of iceberg east position. c) Shows the error norm of the north and east position for the iceberg position estimate in the top level EKF estimator. d) Particle cloud 3σ-bound for SLAM estimates of iceberg heading. e) Absolute heading error for iceberg orientation estimate in the top level EKF estimator.

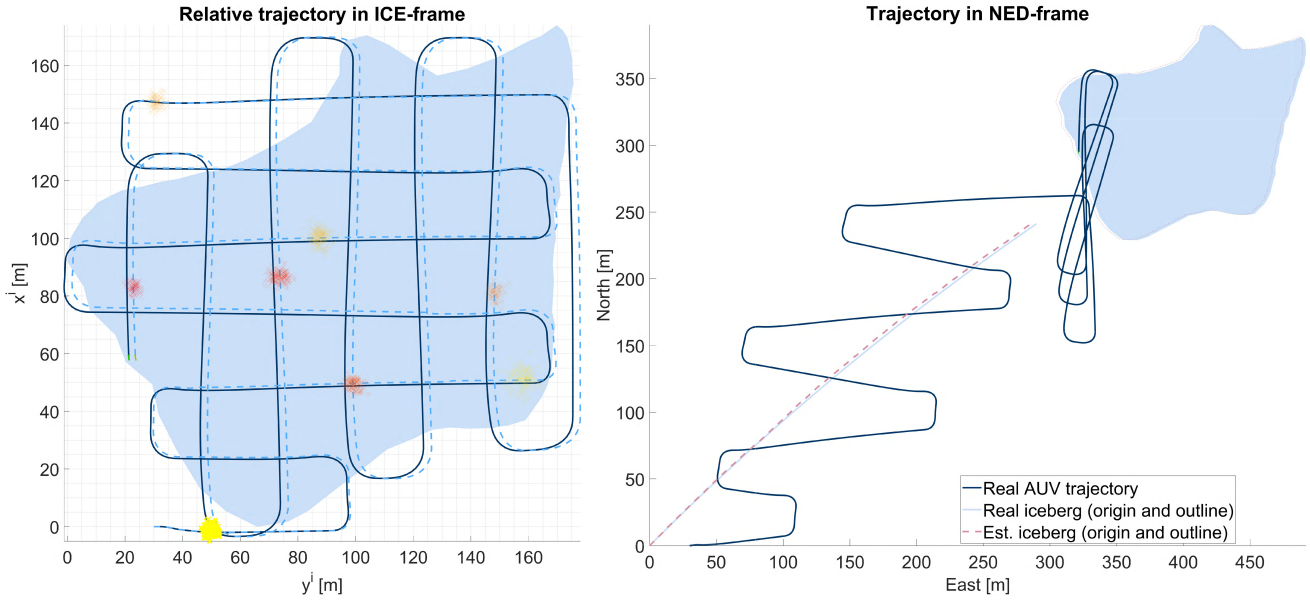

FIGURE 10. Relative and global AUV and iceberg trajectory for Case 3. a) Shows the true and estimated relative AUV trajectory with snapshots of projected iceberg origin point cloud (older clouds are more yellow, newer clouds are more red). b) Global AUV and iceberg trajectory showing the real and estimated iceberg outline at end of simulation.

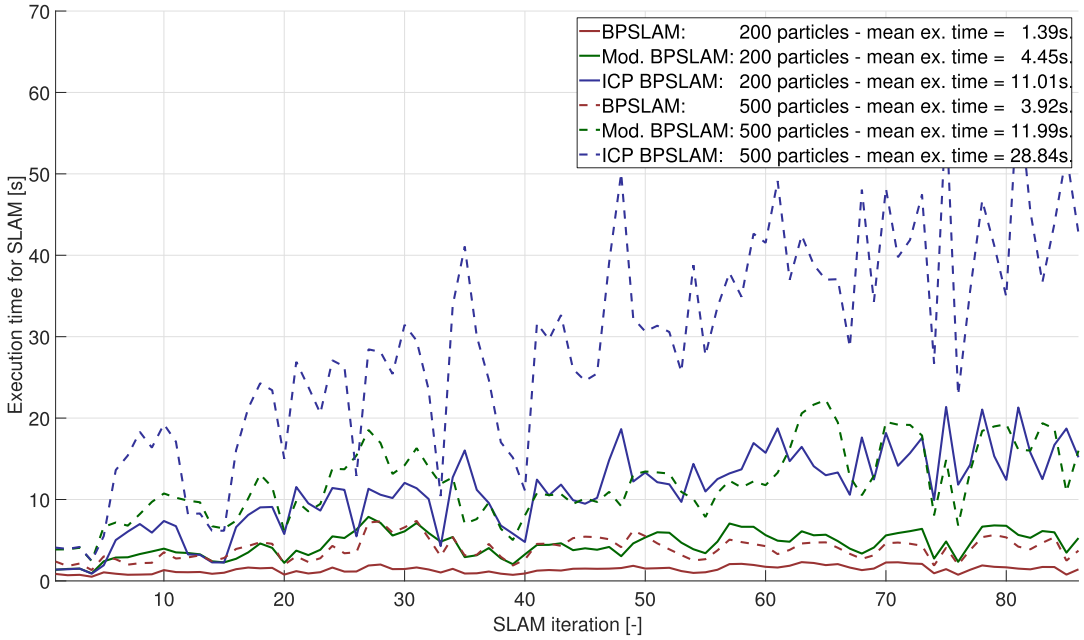

FIGURE 11. Comparison of execution time for the different weighting methods.

shows a run-time comparison with the three different methods studied in this paper, for two different number of particles. The execution time of a particle filter clearly depends on the number of particles, as can be seen from Figure 11. The run-time also depends on several other parameters, such as number of measurements and number of grids covered by the measurements, map size, and map resolution (see [36], [37] for a more detailed analysis of computational complexity for the distributed particle mapping algorithm). From the run-time results shown in Figure 11 we see that the run-time of the original BPSLAM algorithm is several times faster than the other two methods. This is misleading since the BPSLAM only uses $\frac{1}{150}$ of the measurements in the simulations presented here. The original and the modified version will have similar run-time complexities for the same amount of information. The modified BPSLAM method has an average run-time of 4.45 and 11.99 seconds for 200 and 500 particles, respectively. Both are below the limit of 15 seconds (which is the execution period of the SLAM algorithm in our simulations). The maximum execution time is above 20 seconds for

the run using 500 particles, so adaptive execution time would have to be employed in this case for real-time constraints to be met.

The situation is different for the ICP method, which is significantly more computationally expensive. For 200 particles, the mean run-time is just above 11 seconds, and maximum run-time of above 20 seconds, making this comparable to the modified version's run-time with 500 particles. Using 500 particles and the ICP method gives us an average run-time of almost the double of the target run-time. It should be noted that the focus of this study has been on the overall concept, rather than the ICP method itself. A custom designed ICP method for the BPSLAM method could potentially improve both speed and convergence properties. For example by limiting the ICP method to search for 3 DOF transformations, the performance would likely be improved.

The simulations presented have been performed on a PC with Intel Core i7 3770 @ 3.4 GHz using 8 logical processor cores in parallel execution. The algorithm presented in Algorithm 1 is suitable for parallel execution, especially the first for-loop (line 3-7), since each particle is independent. The use of modern graphical processing units (GPUs) with several thousand of cores could ease the load on the processor and speed up the algorithm. In recent years, single board computers with more than 250 GPU cores have been developed, which is suitable for use in unmanned vehicles (e.g. NVIDIA Jetson TX2).

VII. CONCLUSION AND FUTURE WORK

Through this paper we have detailed an estimator designed for mapping of drifting and rotating icebergs. The estimator consists of a top level EKF estimating the relative position and orientation between the AUV and the iceberg, as well as the relative velocity. The iceberg position, orientation, and velocity is also estimated, using inputs from a bottom-level estimator. The bottom-level estimator is based on the BPSLAM algorithm, and two new methods for weighting of multibeam range measurements are proposed. An additional output of the SLAM algorithm is a gridded map containing the topography of the iceberg, which can be extracted in real-time or upon completion of the survey. This can be an important input to the decision-making process in an IM operation.

The results show particle depletion when using the original BPSLAM algorithm, believed to stem from using MBE swaths with partial overlap. The modified version of the BPSLAM weighting scheme is more robust with respect to areas with less information, and can better handle multiple measurements. This opens the possibility for adaptive sample time for the SLAM algorithm – which can help to ensure real-time execution as long as the average execution time is faster than real-time. The third method that is discussed in this paper is an ICP method for BPSLAM, and it shows improved accuracy in the simulated cases, with the cost of significantly increased computational complexity. For the actual bathymetry, on the other hand, the ICP perform worse. This is believed to stem from ICP convergence problems due to a more complex optimization problem in the bathymetric case, due to more than three times the number of beams.

Further work include development of a custom ICP method for the BPSLAM, testing the iceberg SLAM algorithm using real iceberg drift data, and evaluation of the algorithm using real under ice data. Further, the use of GPU programming should be investigated to improve real-time performance, and to reduce the load on the CPU.

The MBE is set to have a constant sample rate of 10 Hz, which may be optimistic, depending on the range. Reduced MBE sample rate will reduce the computational complexity, but will also reduce the amount of information available to the SLAM algorithm. This is not believed to be a problem, since the original BPSLAM method performed similar to the modified version in the bathymetric case, using only $\frac{1}{150}$ of the measurements for resampling. The map will, however, be more sparsely populated, which may require the resolution of the map to be decreased. Regardless, extending the MBE simulator to be range dependent and investigating how this affects the SLAM estimation are interesting points of future work. Similarly, how DVL drop-outs affect the algorithm should be investigated, since this may occur frequently in challenging acoustic environments like under the ice.

To extend the algorithm towards complete iceberg mapping, the realm of active SLAM in conjunction with the iceberg SLAM method should be considered.

REFERENCES

[1] K. Eik, "Review of experiences within ice and iceberg management," *J. Navigat.*, vol. 61, no. 4, pp. 557–572, 2008.

[2] I. Kubat, M. Sayed, S. B. Savage, and T. Carrieres, "An operational model of iceberg drift," *Int. J. Offshore Polar Eng.*, vol. 15, no. 2, pp. 125–131, 2005.

[3] A. Barker, M. Sayed, and T. Carrieres, "Determination of iceberg draft, mass and cross-sectional areas," in *Proc. Int. Offshore Polar Eng. Conf. (ISOPE)*, Toulon, France, 2004, pp. 899–904.

[4] D. L. Murphy and T. Carrieres, "CIS-IIP iceberg model inter-comparison," North American Ice Service, Alexandria, VA, USA, Tech. Rep., 2010.

[5] R. E. Francois and W. Nodland, "Unmanned arctic research submersible (UARS) system development and test report," Appl. Phys. Lab., Univ. Washington, Seattle, WA, USA, Tech. Rep. APL-UW 7219, 1972.

[6] A. L. Forrest *et al.*, "Digital terrain mapping of Petermann ice island fragments in the Canadian high arctic," in *Proc. IAHR Int. Symp. Ice*, Dalian, China, 2012, pp. 1–12.

[7] J.-A. Fernández-Madrigal and J. L. B. Claraco, *Simultaneous Localization and Mapping for Mobile Robots: Introduction and Methods* (Advances in Computational Intelligence and Robotics), 1st ed. Hershey, PA, USA: IGI Global, 2012.

[8] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 99–110, Jun. 2006.

[9] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robot. Autom. Mag.*, vol. 13, no. 3, pp. 108–117, Sep. 2006.

[10] C. Cadena *et al.* (2016). "Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age." [Online]. Available: https://arxiv.org/abs/1606.05830

[11] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, "An efficient approach to bathymetric SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Oct. 2009, pp. 219–224.

[12] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, "Incorporating prior maps with bathymetric distributed particle SLAM for improved AUV navigation and mapping," in *Proc. MTS/IEEE OCEANS*, Biloxi, MS, USA, Oct. 2009, pp. 1–7.

[13] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, "A featureless approach to efficient bathymetric SLAM using distributed particle mapping," *J. Field Robot.*, vol. 28, no. 1, pp. 19–39, 2011.
[14] P. Norgren and R. Skjetne, "A particle filter SLAM approach to online iceberg drift estimation from an AUV," in *Proc. Int. Conf. Ocean, Offshore Arctic Eng.*, 2017, p. V008T07A011.
[15] M. Ludvigsen *et al.*, "Network of heterogeneous autonomous vehicles for marine research and management," in *Proc. MTS/IEEE OCEANS*, Monterey, CA, USA, Sep. 2016, pp. 1–7.
[16] J. M. Thorleifson *et al.*, "The Theseus autonomous underwater vehicle. A Canadian success story," in *Proc. MTS/IEEE OCEANS*, Halifax, NS, Canada, Oct. 1997, pp. 1001–1006.
[17] J. S. Ferguson, "The Theseus autonomous underwater vehicle. Two successful missions," in *Proc. Int. Symp. Underwater Technol.*, Tokyo, Japan, Apr. 1998, pp. 109–114.
[18] P. Wadhams, J. P. Wilkinson, and A. Kaletzky, "Sidescan sonar imagery of the winter marginal ice zone obtained from an AUV," *J. Atmos. Ocean. Technol.*, vol. 21, no. 9, pp. 1462–1470, 2004.
[19] P. Wadhams, J. P. Wilkinson, and S. D. McPhail, "A new view of the underside of Arctic sea ice," *Geophys. Res. Lett.*, vol. 33, no. 4, p. L04501, 2006.
[20] J. G. Bellingham, E. D. Cokelet, and W. J. Kirkwood, "Observation of warm water transport and mixing in the arctic basin with the ALTEX AUV," in *Proc. IEEE/OES AUV*, Woods Hole, MA, USA, Oct. 2008, pp. 1–5.
[21] C. Kaminski *et al.*, "12 days under ice—An historic AUV deployment in the Canadian high Arctic," in *Proc. IEEE/OES AUV*, Monterey, CA, USA, Sep. 2010, pp. 1–11.
[22] J. Ferguson, "1000 km under ice with an AUV—Setting the stage for future achievement," in *Proc. IEEE Symp. Workshop Sci. Use Submarine Cables Related Technol. Underwater Technol.*, Tokyo, Japan, Apr. 2011, pp. 1–5.
[23] P. Norgren, R. Lubbad, and R. Skjetne, "Unmanned underwater vehicles in Arctic operations," in *Proc. IAHR Int. Symp. Ice*, Singapore, 2014, pp. 89–101.
[24] S. Carreno, P. Wilson, P. Ridao, and Y. Petillot, "A survey on terrain based navigation for AUVs," in *Proc. MTS/IEEE OCEANS*, Seattle, WA, USA, Sep. 2010, pp. 1–7.
[25] O. K. Hagen, K. B. Ånonsen, and T. O. Sæbø, "Toward autonomous mapping with AUVs—Line-to-line terrain navigation," in *Proc. MTS/IEEE OCEANS*, Washington, DC, USA, Oct. 2015, pp. 1–8.
[26] C. Roman and H. Singh, "Improved vehicle based multibeam bathymetry using sub-maps and SLAM," in *Proc. Intell. Robot. Syst.*, Edmonton, AB, Canada, Aug. 2005, pp. 3662–3669.
[27] S. Zandara, P. Ridao, D. Ribas, A. Mallios, and A. Palomer, "Probabilistic surface matching for bathymetry based SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, Karlsruhe, Germany, May 2013, pp. 40–45.
[28] A. Palomer, P. Ridao, D. Ribas, A. Mallios, N. Gracias, and G. Vallicrosa, "Bathymetry-based SLAM with difference of normals point-cloud subsampling and probabilistic ICP registration," in *Proc. MTS/IEEE OCEANS*, Bergen, Norway, Jun. 2013, pp. 1–8.
[29] A. Palomer, P. Ridao, D. Ribas, A. Mallios, and G. Vallicrosa, "Octree-based subsampling criteria for bathymetric SLAM," in *Proc. Jornadas De Autom.*, Valencia, Spain, 2014, pp. 1–6.
[30] A. Palomer, P. Ridao, and D. Ribas, "Multibeam 3D underwater SLAM with probabilistic registration," *Sensors*, vol. 16, no. 4, p. 560, 2016.
[31] V. Bichucher, J. M. Walls, P. Ozog, K. A. Skinner, and R. M. Eustice, "Bathymetric factor graph SLAM with sparse point cloud alignment," in *Proc. MTS/IEEE OCEANS*, Washington, DC, USA, Oct. 2015, pp. 1–7.
[32] N. Fairfield, G. Kantor, and D. Wettergreen, "Towards particle filter SLAM with three dimensional evidence grids in a flooded subterranean environment," in *Proc. IEEE Int. Conf. Robot. Autom.*, Orlando, FL, USA, May 2006, pp. 3575–3580.
[33] N. Fairfield, G. Kantor, and D. Wettergreen, "Real-time SLAM with octree evidence grids for exploration in underwater tunnels," *J. Field Robot.*, vol. 24, nos. 1–2, pp. 3–21, 2007.
[34] K. P. Murphy, "Bayesian map learning in dynamic environments," in *Proc. Adv. Neural Inf. Process. Syst.*, 1999, pp. 1015–1021.
[35] A. Doucet, N. de Freitas, K. Murphy, and S. Russell, "Rao-Blackwellised particle filtering for dynamic Bayesian networks," in *Proc. Uncertainty Artif. Intell.*, Stanford, CA, USA, 2000, pp. 176–183.
[36] A. I. Eliazar and R. Parr, "DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks," in *Proc. Int. Joint Conf. Artif. Intell.*, Acapulco, Mexico, 2003, pp. 1135–1142.
[37] A. I. Eliazar and R. Parr, "DP-SLAM 2.0," in *Proc. IEEE Int. Conf. Robot. Autom.*, New Orleans, LA, USA, vol. 2. 2004, pp. 1314–1320.
[38] A. I. Eliazar and R. Parr, "Hierarchical linear/constant time SLAM using particle filters for dense maps," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 339–346.
[39] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, "Bathymetric SLAM with no map overlap using Gaussian processes," in *Proc. Intell. Robot. Syst.*, San Fransisco, CA, USA, Sep. 2011, pp. 1242–1248.
[40] S. Barkby, S. B. Williams, O. Pizarro, and M. V. Jakuba, "Bathymetric particle filter SLAM using trajectory maps," *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1409–1430, 2012.
[41] P. W. Kimball and S. M. Rock, "Sonar-based iceberg-relative AUV navigation," in *Proc. IEEE/OES AUV*, Woods Hole, MA, USA, Oct. 2008, pp. 1–6.
[42] P. W. Kimball and S. M. Rock, "Estimation of iceberg motion for mapping by AUVs," in *Proc. IEEE/OES AUV*, Monterey, CA, USA, Sep. 2010, pp. 1–9.
[43] P. W. Kimball and S. M. Rock, "Sonar-based iceberg-relative navigation for autonomous underwater vehicles," *Deep Sea Res. II, Top. Stud. Oceanograp.*, vol. 58, nos. 11–12, pp. 1301–1310, 2011.
[44] P. W. Kimball and S. M. Rock, "Mapping of translating, rotating icebergs with an autonomous underwater vehicle," *IEEE J. Ocean. Eng.*, vol. 40, no. 1, pp. 196–208, Jan. 2015.
[45] M. Hammond and S. M. Rock, "Enabling AUV mapping of free-drifting icebergs without external navigation aids," in *Proc. MTS/IEEE OCEANS*, St. John's, NL, Canada, Sep. 2014, pp. 1–7.
[46] M. Hammond and S. M. Rock, "A SLAM-based approach for underwater mapping using AUVs with poor inertial information," in *Proc. IEEE/OES AUV*, Oxford, MS, USA, Oct. 2014, pp. 1–8.
[47] C. G. Kunz, "Autonomous underwater vehicle navigation and mapping in dynamic, unstructured environments," Ph.D. dissertation, Joint Program Appl. Ocean Sci. Eng., Massachusetts Inst. Technol., Cambridge, MA, USA, 2012.
[48] P. Norgren and R. Skjetne, "Iceberg detection and edge-following using auv with multibeam sonar," in *Proc. Int. Conf. Port Ocean Eng. Arctic Conditions*, Trondheim, Norway, 2015, pp. 1–13.
[49] P. Norgren and R. Skjetne, "Line-of-sight iceberg edge-following using an AUV equipped with multibeam sonar," in *Proc. IFAC-PapersOnline*, 2015, vol. 48. no. 16.
[50] R. Yulmetov, A. Marchenko, and S. Løset, "Iceberg and sea ice drift tracking and analysis off north-east Greenland," *Ocean Eng.*, vol. 123, pp. 223–237, Sep. 2016.
[51] D. P. Williams, F. Baralli, M. Micheli, and S. Vasoli, "Adaptive underwater sonar surveys in the presence of strong currents," in *Proc. Int. Conf. Robot. Autom.*, May 2016, pp. 2604–2611.
[52] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. Hoboken, NJ, USA: Wiley, 2011.
[53] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point," *Comput. Graph. Forum*, vol. 32, no. 5, pp. 113–123, Aug. 2013.
[54] R. Sandhu, S. Dambreville, and A. Tannenbaum, "Particle filtering for registration of 2D and 3D point sets with stochastic dynamics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[55] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 34–46, Feb. 2007.
[56] J. D. Hol, "Resampling in particle filters," M.S. thesis, Dept. Electr. Eng., Linköpings Univ., Linköpings, Sweden, 2004.
[57] P. Norgren and R. Skjetne, "Using autonomous underwater vehicles as sensor platforms for ice-monitoring," *Model. Identificat. Control*, vol. 35, no. 4, pp. 263–277, 2014.
[58] T. Prestero, "Verification of a six-degree of freedom simulation model for the REMUS autonomous underwater vehicle," M.S. thesis, Dept. Ocean Eng., Massachusetts Inst. Technol., Cambridge, MA, USA, 1994.
[59] S. Holsen, "DUNE: Unified navigation environment for the REMUS 100 AUV," M.S. thesis, Dept. Marine Technol., Norwegian Univ. Sci. Technol., Trondheim, Norway, 2015.
[60] G. Timco, "Compilation of iceberg shape and geometry data for the grand banks region," Program Energy Res. Develop., Canadian Hydraulics Center, Nat. Res. Council Canada, Ottawa, ON, Canada, Tech. Rep. PERD/CHC 20-43, 1999.
[61] K. Gade, "NavLab, a generic simulation and post-processing tool for navigation," *Eur. J. Navigat.*, vol. 2, no. 4, pp. 1–9, 2004.
[62] *Imagenex Model 837B 'Delta T' 300 m Multibeam Profiling Sonar*, Imagenex, Port Coquitlam, BC, Canada, May 2017.

PETTER NORGREN received the M.Sc. degree in engineering cybernetics from the Norwegian University of Science and Technology (NTNU) in 2011. From 2012 to 2013, he was with Marine Cybernetics AS involved in hardware-in-the-loop testing of marine control systems. He begun a Ph.D. Scholarship with the Department of Marine Technology, NTNU, in 2013, involved in autonomous underwater vehicles for arctic marine operations. His research interests include AUVs for arctic marine operations and research, simultaneous localization and mapping, autonomous exploration, and tracking and monitoring of AUVs using USVs.

ROGER SKJETNE received the M.Sc. degree from the University of California at Santa Barbara in 2000 and the Ph.D. degree in control engineering from the Norwegian University of Science and Technology (NTNU) in 2005. He holds an Exxon Mobil Prize for his Ph.D. thesis at NTNU. Prior to his studies, he was a certified Electrician for Aker Elektro AS on numerous oil installations for the North Sea, and from 2004 to 2009, he was with Marine Cybernetics AS, involved in hardware-in-the-loop simulation for testing marine control systems. Since 2009, he has been the Kongsberg Maritime Chair of Professor in marine control engineering with the Department of Marine Technology, NTNU. His research interests include dynamic positioning of marine vessels, arctic stationkeeping and ice management systems, control of shipboard hybrid electric power systems, nonlinear motion control of marine vehicles, and autonomous ships and marine robots.