An Exact Decentralized Cooperative Navigation Algorithm for Acoustically Networked Underwater Vehicles with Robustness to Faulty Communication: Theory and Experiment

Jeffrey M. Walls and Ryan M. Eustice

***Abstract*—This paper reports on an exact real-time solution for server-client cooperative localization over a faulty and extremely bandwidth-limited underwater communication channel. Our algorithm, termed the origin state method, enables a 'server' vehicle to aid the navigation of multiple 'client' vehicles via a novel representation of the server's pose-graph that is robust to communication packet loss. This transmitted pose-graph can be used in conjunction with a decentralized extended information filter on the client to reproduce the corresponding two-vehicle server-client centralized result exactly. We present a full comparative evaluation for the first-ever real-time field implementation of the proposed algorithm for a multi-agent autonomous underwater vehicle network using underwater acoustic modems to communicate in a synchronous-clock transmission framework.**

I. Introduction

Underwater vehicles typically rely on fusing Doppler velocity log (DVL) body-frame velocities, attitude, and pressure depth observations to compute a dead-reckoned navigation solution. While attitude and depth are well instrumented, there is no easy method to directly observe x, y horizontal position [the global positioning system (GPS) does not work underwater]. In this paper, we report a novel algorithm enabling multiple underwater vehicles (servers) to cooperatively aid the navigation of several other vehicles (clients) that is robust to packet-loss and low-bandwidth that is endemic in underwater acoustic communication networks. Our algorithm is capable of bounding the error growth of the client vehicles to that of the server vehicles.

Typical bounded-error underwater navigation methods, such as long-baseline (LBL), measure the relative range between the vehicle and fixed reference beacons [18, 29]. The relative range is measured using two-way time-of-flight (TOF) acoustic broadcasts and assuming a known sound-speed profile. Narrowband acoustic beacon networks, however, are limited in their ability to scale to many vehicles as only one vehicle can interrogate the network at a time. Moreover, the range of vehicle operations is limited to the acoustic footprint of the beacon network.

The use of synchronous-clock hardware enables a team of vehicles to observe their relative range via the one-way-travel-time (OWTT) of narrowband acoustic broadcasts [7].

This work was supported by a grant from the National Science Foundation under awards IIS-0746455 and ANT-1039951.

J. Walls is with the Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109, jmwalls@umich.edu

R. Eustice is with the Department of Naval Architecture and Marine Engineering, University of Michigan, Ann Arbor, MI 48109, eustice@umich.edu

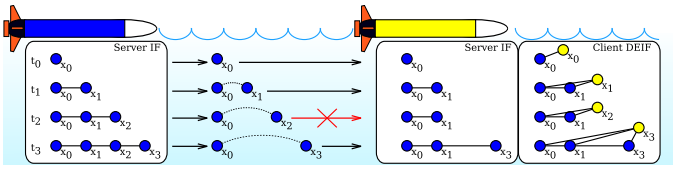


Fig. 1: Origin state method algorithm overview. The server (blue) fuses its local observations and adds delayed-states at time-of-launches. It uses our novel origin state method to incrementally transmit its pose-graph in a fault-tolerant way. At the time-of-arrival of each received origin state packet, the client (yellow) reconstructs the server pose-graph and updates its estimator to fuse all new information. In this example, although the client misses the server transmission at t_2, the client can still reconstruct the server pose-graph after receiving the origin state packet at t_3 and perform a relative range update.

The OWTT relative range is measured between the transmitting vehicle at the time-of-launch (TOL) and the receiving vehicle at the time-of-arrival (TOA). Since ranging is passive—all receiving platforms observe relative range from a single transmission—OWTT networks scale well. OWTTs to augment vehicle navigation present several open questions regarding how to share and incorporate information across the network in a robust and optimal way.

The underwater acoustic communication channel is severely limited by the physical characteristics of seawater [21]. Acoustic communication is constrained by large latency and low bandwidth with packet loss often greater than 50%. The underwater acoustic channel has an upper-bound range rate product of $40\ \mathrm{km} \cdot \mathrm{kbps}$. In practice, underwater vehicle networks are only able to obtain real-world bandwidth on the order of $100\ \mathrm{bps}$ [19], which is several orders of magnitude less than terrestrial communication networks. An unacknowledged broadcast protocol is also commonly employed in conjunction with time division multiple access (TDMA) scheduling, which further limits overall bandwidth by dividing transmission time over the communication network. All of these challenges amount to a communication framework that enforces small-payloads and infrequent updates between vehicles.

A variety of cooperative localization frameworks exist for improving the position estimates across a team of robots via sharing of navigation information. No method, however, currently addresses the severely limited bandwidth and fragility of the underwater acoustic communication channel in a potentially scalable and optimal way. In this paper, we consider the solution for the navigation of a client vehicle aided by a server platform. The contributions of this work are:

1) We present a general algorithm, called the origin state method (OSM), that allows multiple servers to transmit their pose-graphs via a faulty, low bandwidth, communication channel, and to optimally fuse this information

onboard a client within a decentralized extended information filter (DEIF) [28] that works in real-time for practical, underwater, acoustic networks.
2) We present a comparative experimental evaluation in full-scale autonomous underwater vehicle (AUV) trials. Our algorithm performance is compared to other previously reported methods including an egocentric filter [16] and a fully nonlinear smoothing approach (i.e., iSAM) [14].

II. Related Work

Cooperative networks enable robots with the best navigation sensors to localize robots with poorer position estimates. Simple, real-time algorithms that require minimal bandwidth are within the egocentric class of filters [11, 16, 25]. These algorithms scale by treating each relative observation as independent and only require the transmitter's current position estimate. While trivially resistant to communication failure, these methods do not account for the correlation that develops between robot estimates, which eventually leads to inconsistent (i.e., overconfident) estimates.

In response to egocentric approaches, Bahr et al. [2] and Fallon et al. [9] propose distributed bookkeeping strategies to ensure that information is incorporated in a consistent manner. Each of their approaches requires additional bandwidth or use of acknowledgments. Similarly motivated, Ribeiro et al. [22] and Nerurkar et al. [20] achieve consistency through a noteworthy approach to cooperative navigation in which they transmit just a single bit per measurement (representing the sign-of-innovations)—yielding an algorithm that closely mirrors the standard Kalman filter. While reducing overall bandwidth, the algorithm requires 100% packet reception, which is unrealistic for faulty communication channels.

The most general cooperative navigation algorithms estimate the full joint distribution over all vehicle poses [5, 13]. Roumeliotis and Bekey [24] developed a distributed extended Kalman filter (EKF)-based method, though it requires high bandwidth, and two-way information exchange. Cunningham et al. [4] and Kim et al. [15] later studied the problem of nonlinear simultaneous localization and mapping (SLAM) in a distributed fashion where each platform *(i)* transmits its full local pose-graph, *(ii)* collects the local pose-graphs from neighboring platforms, and *(iii)* estimates the full distribution by optimizing over all available graphs. The result is a consistent estimate that matches the centralized omniscient estimator solution at the expense of high communication cost, which grows with the size of the local graph.

Webster et al. [27] presented a post-process centralized EKF specifically designed for synchronous-clock acoustic cooperative localization. They later distributed this centralized filter result exactly [28], called the DEIF, by leveraging the sparse update properties of the delayed-state information filter. Their solution requires a strict server-to-client support topology, as the server transmits representative local information to the client where the centralized filter solution is reproduced. Bailey et al. [3] independently developed an equivalent formulation for sharing locally obtained information, relying on fusion centers to perform relative robot measurement updates. The fusion centers increase complexity, but allow for arbitrary communication topologies. In practice, both of these methods are not realizable in the underwater scenario because they require a non-faulty communication channel. We previously reported [26] a preliminary method toward alleviating the non-faulty communication constraint in distributing local server information that relied upon a client acknowledgment scheme.

Several other works in acoustic cooperative underwater navigation have emerged for fusing OWTT-based relative ranges between teams of vehicles [1, 2, 8, 10, 16, 17, 25]. These methods, however, generally trade off between offline and consistent, or real-time and inconsistent.

Our work is closest to [28], [3], and [26] in its effort to distribute local data fusion in an optimal way and to leverage the sparsity of the Gaussian information form to compactly transmit this information. In this paper we *(i)* build upon the previously reported approach in [26] by improving network scalability via a *passive* origin shifting scheme that eliminates the need for acknowledgments, *(ii)* introduce a recovery packet mechanism that enables clients to enter and leave the network or recover after a long period of communication dropout, and *(iii)* present full-scale comparative AUV trials demonstrating our algorithm's ability for real-time underwater navigation.

III. Consistent Cooperative Navigation

We consider several independent server vehicles aiding the navigation of multiple client vehicles. For the sake of presentation, we refer to a single server vehicle, although our algorithm can support multiple. The client vehicles are able to passively observe their range to the server vehicle during periodic server transmissions. Each client then updates its pose estimate using its local information, the range observations, and the transmitted information. Relative range observations occur between the server at the time-of-launch (TOL) and each client at the time-of-arrival (TOA) by measuring the one-way-travel-time (OWTT) of an acoustic broadcast. A centralized estimator, e.g., [27], which has access to the local and relative observations of all vehicles, but is realizable in post-process only, serves as the gold-standard benchmark solution. Our formulation is able to reproduce this centralized filter result onboard each client vehicle in real-time for the server and client states.

The proposed OSM algorithm provides a means to incrementally communicate the server's pose-graph (represented in the information form) with a small fixed-bandwidth data packet that is robust to packet loss. Each server transmission contains all new local information relative to a server state known by the client, termed the origin. The client then reconstructs the server pose-graph to recreate the solution of the DEIF. Fig. 1 provides an overview of the OSM algorithm.

A. Information Filter

The OSM algorithm relies on manipulating the Gaussian distribution in the information form to efficiently transmit the server pose-graph. We use a delayed-state information filter

to initially construct the server pose-graph. The information filter tracks a Gaussian over its state, $\boldsymbol{x}$, parametrized in the information form; that is $p(\boldsymbol{x}) = \mathcal{N}^{-1}(\boldsymbol{x} : \boldsymbol{\eta}, \Lambda)$, where the information matrix, Λ, and vector, $\boldsymbol{\eta}$, are related to the mean and covariance by

$$\Lambda = \Sigma^{-1}, \quad \boldsymbol{\eta} = \Lambda\boldsymbol{\mu} \tag{1}$$

where Σ and $\boldsymbol{\mu}$ are the covariance matrix and mean vector of $\boldsymbol{x}$, respectively.

The single vehicle navigation problem is framed in terms of estimating the joint distribution over a collection of historic poses (past vehicle states). In this case, the state vector is composed of these historic poses, termed 'delayed-states', $\boldsymbol{x} = \left[\mathbf{x}_n^\top, \mathbf{x}_{n-1}^\top, \ldots, \mathbf{x}_1^\top\right]^\top$. The information filter state vector grows over time by performing prediction with augmentation. As noted in [6], processes that evolve with the Markov property result in a sparse, block tri-diagonal information matrix. This sparsity leads to an update formulation that only affects a small sub-block of the information matrix and vector. We assume that new poses are prepended onto the state vector.

The representation of the delayed-state collection is termed the pose-graph (Fig. 2). Nodes in the graph express delayed-state variables while edges encode the relationships between nodes. The sparsity pattern of the information matrix corresponds exactly to the adjacency matrix of the pose-graph.

A pose-graph constructed from proprioceptive measurements (e.g., DVL body-frame velocities, GPS) creates a Markov chain of vehicle poses. New odometry inputs and local measurements only modify a block of the information matrix corresponding to the current robot pose and the most recent delayed-state; only the value of the pose-graph edge between the last delayed-state and the current state is affected.

B. Origin State Method

The OSM algorithm represents a way for decomposing, transmitting, and later reconstructing the server's pose-graph. The underlying assumption is that the transmitted pose-graph grows as a Markov chain—the standard model for a dynamic system. Each origin state transmission, called an origin state packet (OSP), encodes a server transition from the origin state to the current state (see Fig. 2). The OSP represents the relationship between the origin and newest state as their joint marginal distribution, i.e., the two-node pose-graph over the origin and newest TOL state. Since the newest server measurements continue to smooth the entire pose-graph, the client is able to infer the newest edge in the server pose-graph by observing the difference in information known about the origin state at the current and last received OSP (12).

Conceptually, if the server state were to evolve according to a simple odometry model between TOL states, then the client could ascertain the updated pose-graph via inversion of compounding operations. For example, in Fig. 1, at t_3 the server could transmit the marginal $\left[\mathbf{x}_0, \mathbf{x}_3\right]$ and the client could then solve for the new edge as $\mathbf{x}_{13} = \ominus\mathbf{x}_{01} \oplus \mathbf{x}_{03}$, where $\mathbf{x}_{01}$ is computed from the client's prior information and $\mathbf{x}_{03}$ is obtained from the server's marginal information. In the general case where the server fuses arbitrary proprioceptive measurements (e.g., GPS, LBL), the OSM is a generalization of this concept so that the client can reconstruct the server pose-graph distribution.

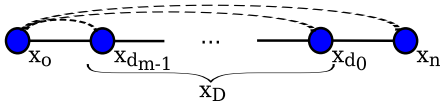


Fig. 2: Pose-graph example illustrating transition encoded in OSM packets.

1) Server-side Origin State Operation: The server vehicle maintains an information filter, augmenting its state vector with a copy of each TOL state. At the TOL, the server broadcasts an OSP containing the marginal information of the current TOL state and a designated previous delayed-state, the origin, as depicted in Fig. 2. The index label of the new TOL state and the origin are also transmitted to the client for reconstruction.

At time k, the n^{th} TOL, the server vehicle's state vector contains the current TOL state in addition to past TOL states,

$$\boldsymbol{x}_k = \left[\mathbf{x}_{t_n}^\top, \mathbf{x}_{t_{n-1}}^\top, \ldots, \mathbf{x}_{t_1}^\top\right]^\top, \tag{2}$$

where the 't' subscript indicates a TOL. The OSP contains the marginal,

$$\boldsymbol{x}_k^s = \left[\mathbf{x}_{t_n}^\top, \mathbf{x}_{t_o}^\top\right]^\top, \tag{3}$$

computed via the Schur complement [6] where the 's' superscript represents the marginal computed by the server and the 'o' subscript indicates some previous TOL state, designated the origin state, such that $o \in \{n-1, \ldots, 1\}$. In order for the client to reconstruct the server pose-graph, the client must have the origin state in its representation. The marginal information contained in the n^{th} OSP is denoted

$$\Lambda_n^s = \begin{bmatrix} \Lambda_{t_n,t_n}^s & \Lambda_{t_n,t_o}^s \\ \Lambda_{t_o,t_n}^s & \Lambda_{t_o,t_o}^s \end{bmatrix}, \quad \boldsymbol{\eta}_n^s = \begin{bmatrix} \boldsymbol{\eta}_{t_n}^s \\ \boldsymbol{\eta}_{t_o}^s \end{bmatrix}. \tag{4}$$

Algorithm 1 summarizes the server-side operation.

2) Client-side Origin State Operation: The client maintains two pose-graphs: the first is simply the reconstructed server pose-graph, the second is the client-side DEIF. The client incrementally reconstructs the server pose-graph from the sequence of successfully received OSPs.

Each OSP represents the two-node server pose-graph consisting of the origin state and the newest TOL state. Prior to the TOA, the client has already reconstructed the server's pose-graph (including the origin state through the last received TOL state). The target distribution is the updated server pose-graph, which now includes the edge to the newest TOL state.

The client essentially solves a reverse marginalization procedure to incorporate the newest OSP. The update rules are derived by starting with the client-side target distribution (the current server pose-graph with the unsuccessfully received TOL states marginalized out), marginalizing out states to obtain the distribution over the origin and newest TOL states, and equating terms with the transmitted OSP.

At the TOA of the first received OSP, the client does not need to perform any computation to reconstruct the server

Algorithm 1 Server-side Origin State Method

Require: $\Lambda_0, \boldsymbol{\eta}_0$ {initial server belief}
$\Lambda_b, \boldsymbol{\eta}_b, o_b \leftarrow 0$ {backup origin state packet}
loop
  if k is TOL_n **then**
    $\Lambda^s_n, \eta^s_n, o_n \leftarrow$ `originPacket`$(\Lambda_k, \boldsymbol{\eta}_k)$
    if $o_n \neq o_{n-1}$ **then**
      {origin has been shifted, update backup packet}
      $\Lambda_b, \boldsymbol{\eta}_b, o_b \leftarrow \Lambda^s_{n-1}, \eta^s_{n-1}, o_{n-1}$
    end if
    `broadcastOriginPacket`$(\Lambda^s_n, \eta^s_n, o_n, \Lambda_b, \boldsymbol{\eta}_b, o_b)$
    if `recoveryRequired`() **then**
      `broadcastRecovery`() {Section III-B4}
    end if
    $\Lambda_k, \eta_k \leftarrow$ `predictAugment`$(\Lambda_k, \boldsymbol{\eta}_k)$
    $n \leftarrow n+1$
  else
    $\Lambda_k, \boldsymbol{\eta}_k \leftarrow$ `predict`$(\Lambda_k, \boldsymbol{\eta}_k)$
  end if
  $\Lambda_k, \boldsymbol{\eta}_k \leftarrow$ `localMeasUpdate`$(\Lambda_k, \boldsymbol{\eta}_k, \mathbf{z}_k)$
  $k \leftarrow k+1$
end loop

pose-graph, (4). The initial OSP is simply the two-node server pose-graph consisting of the server origin state and the TOL state. (Note that this allows any new clients to immediately enter and join the network.) At the TOA of the n^{th} OSP, the client receives the information contained in (4). The client has reconstructed some portion of the server pose-graph with the state vector

$$\boldsymbol{x}_{d_0} = \left[\mathbf{x}^\top_{t_D}, \mathbf{x}^\top_{t_o}, \mathbf{x}^\top_{t_P}\right]^\top, \tag{5}$$

where D is the set of received TOL states occurring after the origin state, $D = \{d_0, \ldots, d_{m-1}\}$, P represents the set of all TOL states before the origin state, and $\mathbf{x}_{t_o}$ is the current origin state. Using (4) and the client's current reconstruction, (5), the client solves for the target distribution, i.e., the updated server pose-graph with state

$$\boldsymbol{x}_n = \left[\mathbf{x}^\top_{t_n}, \mathbf{x}^\top_{t_D}, \mathbf{x}^\top_{t_o}, \mathbf{x}^\top_{t_P}\right]^\top, \tag{6}$$

where $\mathbf{x}_{t_n}$ represents the n^{th} server TOL. Due to the Markovity of the server process, (6) and (5) differ only by a new edge between the previous and the newest TOL states. There are three cases for which the client will incorporate the new packet into its existing reconstruction depending on the number of TOL states occuring after the origin (number of states in $D = \{d_0, \ldots, d_{m-1}\}$ as in Fig. 2).

a) Case I: The origin is the previously received TOL state (i.e., D=$\{\emptyset\}$). The client-side reconstructed pose-graph begins with the corresponding information matrix and vector

$$\Lambda_o = \begin{bmatrix} \Lambda_{t_o,t_o} & \Lambda_{t_o,t_P} \\ \Lambda_{t_P,t_o} & \Lambda_{t_P,t_P} \end{bmatrix}, \quad \boldsymbol{\eta}_o = \begin{bmatrix} \boldsymbol{\eta}_{t_o} \\ \boldsymbol{\eta}_{t_P} \end{bmatrix}. \tag{7}$$

The target distribution the client wants to compute (the current server distribution) has the information matrix and vector

$$\Lambda_n = \begin{bmatrix} \boxed{\begin{matrix} \Lambda_{t_n,t_n} & \Lambda_{t_n,t_o} \\ \Lambda_{t_o,t_n} & \Lambda'_{t_o,t_o} \end{matrix}} & 0 \\ 0 & \begin{matrix} \Lambda_{t_o,t_P} \\ \Lambda_{t_P,t_P} \end{matrix} \end{bmatrix}, \; \boldsymbol{\eta}_n = \begin{bmatrix} \boxed{\begin{matrix} \boldsymbol{\eta}_{t_n} \\ \boldsymbol{\eta}'_{t_o} \end{matrix}} \\ \boldsymbol{\eta}_{t_P} \end{bmatrix}, \tag{8}$$

respectively, where the boxes surround the new or modified elements of the client information matrix and vector. Also, the prime symbols over the blocks corresponding to the origin state are to stress that these are not the same as the information blocks of the previously reconstructed information matrix and vector, (7). These updated terms are computed as

$$\begin{aligned} \Lambda_{t_n,t_n} &= \Lambda^s_{t_n,t_n} \\ \Lambda_{t_n,t_o} &= \Lambda^s_{t_n,t_o} \\ \Lambda'_{t_o,t_o} &= \Lambda^s_{t_o,t_o} + \Lambda_{t_o,t_P}\Lambda^{-1}_{t_P,t_P}\Lambda_{t_P,t_o} \\ \boldsymbol{\eta}_{t_n} &= \boldsymbol{\eta}^s_{t_n} \\ \boldsymbol{\eta}'_{t_o} &= \boldsymbol{\eta}^s_{t_o} + \Lambda_{t_o,t_P}\Lambda^{-1}_{t_P,t_P}\boldsymbol{\eta}_{t_P}. \end{aligned} \tag{9}$$

b) Case II: A single TOL state exists between the origin state and the newest TOL (i.e., D=$\{d\}$). Without loss of generality, we take P=$\{\emptyset\}$ to further simplify the discussion. The client has already reconstructed the following information matrix and vector

$$\Lambda_d = \begin{bmatrix} \Lambda_{t_d,t_d} & \Lambda_{t_d,t_o} \\ \Lambda_{t_o,t_d} & \Lambda_{t_o,t_o} \end{bmatrix}, \quad \boldsymbol{\eta}_d = \begin{bmatrix} \boldsymbol{\eta}_{t_d} \\ \boldsymbol{\eta}_{t_o} \end{bmatrix}. \tag{10}$$

The desired target distribution has the form

$$\Lambda_n = \begin{bmatrix} \boxed{\begin{matrix} \Lambda_{t_n,t_n} & \Lambda_{t_n,t_d} \\ \Lambda_{t_d,t_n} & \Lambda'_{t_d,t_d} \end{matrix}} & 0 \\ 0 & \begin{matrix} \Lambda_{t_d,t_o} \\ \Lambda_{t_o,t_d} \end{matrix} \end{bmatrix}, \; \boldsymbol{\eta}_n = \begin{bmatrix} \boxed{\begin{matrix} \boldsymbol{\eta}_{t_n} \\ \boldsymbol{\eta}'_{t_d} \end{matrix}} \\ \boldsymbol{\eta}_{t_o} \end{bmatrix}, \tag{11}$$

where again the boxes surround new or modified elements. In this case, the updated terms are:

$$\begin{aligned} \Lambda'_{t_d,t_d} &= \left[-\Lambda^{-1}_{t_o,t_d}\left(\Lambda^s_{t_o,t_o} - \Lambda_{t_o,t_o}\right)\Lambda^{-1}_{t_d,t_o}\right]^{-1} \\ \Lambda_{t_n,t_d} &= -\Lambda^s_{t_n,t_o}\Lambda^{-1}_{t_d,t_o}\Lambda'_{t_d,t_d} \\ \Lambda_{t_n,t_n} &= \Lambda^s_{t_n,t_n} + \Lambda_{t_n,t_d}\Lambda'^{-1}_{t_d,t_d}\Lambda_{t_d,t_n} \\ \boldsymbol{\eta}_{t_d} &= \Lambda'_{t_d,t_d}\Lambda^{-1}_{t_o,t_d}\left(\boldsymbol{\eta}_{t_o} - \boldsymbol{\eta}^s_{t_o}\right) \\ \boldsymbol{\eta}_{t_n} &= \boldsymbol{\eta}^s_{t_n} + \Lambda_{t_n,t_d}\Lambda'^{-1}_{t_d,t_d}\boldsymbol{\eta}_{t_d}. \end{aligned} \tag{12}$$

c) Case III: Multiple TOL states exist between the origin state and the new node (i.e., D=$\{d_0, \ldots, d_{m-1}\}$). Prior to incorporating the OSP, the client begins with the information matrix and vector

$$\Lambda_{d_0} = \begin{bmatrix} \Lambda_{t_D,t_D} & \Lambda_{t_D,t_o} \\ \Lambda_{t_o,t_D} & \Lambda_{t_o,t_o} \end{bmatrix}, \quad \boldsymbol{\eta}_{d_0} = \begin{bmatrix} \boldsymbol{\eta}_{t_D} \\ \boldsymbol{\eta}_{t_o} \end{bmatrix}, \tag{13}$$

where d_0 corresponds to the most recently received TOL. This case requires first marginalizing out $D' = D \setminus \{d_0\}$ such that

$$\Lambda^c_{d_0} = \begin{bmatrix} \Lambda^c_{t_{d_0},t_{d_0}} & \Lambda^c_{t_{d_0},t_o} \\ \Lambda^c_{t_o,t_{d_0}} & \Lambda^c_{t_o,t_o} \end{bmatrix}, \quad \boldsymbol{\eta}^c_{d_0} = \begin{bmatrix} \boldsymbol{\eta}^c_{t_{d_0}} \\ \boldsymbol{\eta}^c_{t_o} \end{bmatrix}, \tag{14}$$

where the 'c' superscript indicates the marginal of the server pose-graph computed by the client vehicle. The client can then solve for the server pose-graph with the D' elements marginalized out under the conditions of Case II beginning with $\Lambda^c_{d_0}$ and $\boldsymbol{\eta}^c_{d_0}$. The final target distribution has the following information matrix and vector

$$\Lambda_n = \begin{bmatrix} \boxed{\begin{matrix} \Lambda_{t_n,t_n} & \Lambda_{t_n,t_{d_0}} \\ \Lambda_{t_{d_0},t_n} & \Lambda'_{t_{d_0},t_{d_0}} \end{matrix}} & 0 & 0 \\ 0 & \Lambda_{t_{d_0},t_{D'}} & 0 \\ 0 & \Lambda_{t_{D'},t_{d_0}} & \begin{matrix} \Lambda_{t_{d_0},t_{D'}} & \Lambda_{t_{D'},t_o} \\ \Lambda_{t_{D'},t_{D'}} & \Lambda_{t_{D'},t_o} \\ \Lambda_{t_o,t_{D'}} & \Lambda_{t_o,t_o} \end{matrix} \end{bmatrix}, \; \boldsymbol{\eta}_n = \begin{bmatrix} \boxed{\begin{matrix} \boldsymbol{\eta}_{t_n} \\ \boldsymbol{\eta}'_{t_{d_0}} \end{matrix}} \\ \boldsymbol{\eta}_{t_{D'}} \\ \boldsymbol{\eta}_{t_o} \end{bmatrix}, \tag{15}$$

Algorithm 2 Client-side Origin State Method

$\Lambda_0, \boldsymbol{\eta}_0 \leftarrow 0$
loop
  if $(\Lambda^s_n, \eta^s_n, o_n, \Lambda_b, \eta_b, o_b) \leftarrow$ `receivedPacket()` **then**
    if `haveOriginIndex`(o_n) **then**
      $\Lambda_n, \boldsymbol{\eta}_n \leftarrow$ `addOriginPacket`$(\Lambda^s_n, \eta^s_n, o_n)$
      `updateDEIF`$(\Lambda_n, \boldsymbol{\eta}_n)$ {(18), (19)}
    else if `haveBackupOriginIndex`(o_b) **then**
      $\Lambda_{n-1}, \boldsymbol{\eta}_{n-1} \leftarrow$ `addOriginPacket`(Λ_b, η_b, o_b)
      $\Lambda_n, \boldsymbol{\eta}_n \leftarrow$ `addOriginPacket`$(\Lambda^s_n, \eta^s_n, o_n)$
      `updateDEIF`$(\Lambda_n, \boldsymbol{\eta}_n)$ {(18), (19)}
    else
      `requestRecovery()`
    end if
  end if
end loop

where Λ_{t_n,t_n}, $\Lambda_{t_n,t_{d_0}}$, and $\boldsymbol{\eta}_{t_n}$ are all computed under Case II. The final unknown elements of the information matrix and vector are then computed as

$$\begin{aligned} \Lambda'_{t_{d_0},t_{d_0}} &= \Lambda'^{c}_{t_{d_0},t_{d_0}} + \Lambda_{t_{d_0},t_{D'}}\Lambda^{-1}_{t_{D'},t_{D'}}\Lambda_{t'_D,t_{d_0}} \\ \boldsymbol{\eta}'_{t_{d_0}} &= \boldsymbol{\eta}'^{c}_{t_{d_0}} + \Lambda_{t_{d_0},t_{D'}}\Lambda^{-1}_{t_{D'},t_{D'}}\boldsymbol{\eta}_{t_{D'}} \end{aligned} . \quad (16)$$

3) Origin Shifting: The information difference $\left(\Lambda^s_{t_o,t_o} - \Lambda_{t_o,t_o}\right)$ in (12) represents the delta information known about the origin state between the server and client. This difference approaches machine precision as the time difference between the origin and new TOL state grows (because additional smoothing of the origin state is small after sufficient time). The reconstruction rules require the inversion of this decreasing term, leading to numerical inaccuracies that can cause divergent errors in the reconstruction. A simple solution is to ensure that the origin is periodically shifted forward.

An origin shifting scheme based on acknowledgments from each client was previously proposed in [26]; however, an acknowledgment based scheme does not scale well to many clients, and also nullifies several benefits of OWTT-based relative ranging. Moreover, numerical instability will continue to plague the system if the server does not regularly receive acknowledgments. We propose a shifting scheme in which the server evaluates a criteria based on the numerical stability of the newest OSP—keeping the algorithm *passive*, such that the method can more easily scale to many vehicles.

During our real-time experiments (Section IV), the server shifted the origin forward based on a threshold for the RV-coefficient (a multivariate analog to the correlation coefficient [23]) between the current TOL state, $\mathbf{x}_{t_n}$, and the origin state, $\mathbf{x}_{t_o}$. The motivation being that the unstable difference term, (12), will be small when correlation between the states is small. Unfortunately, this approach is not general enough, as the RV-coefficient can remain large while the difference term decreases when the server is purely dead-reckoning. In post-process we discovered a superior alternative shifting criteria—the server compares the trace of the difference term in (12) to a threshold value, T,

$$\mathrm{trace}\left(\Lambda^s_{t_o,t_o} - \Lambda_{t_o,t_o}\right) < \mathrm{T}. \quad (17)$$

The trace directly compares the numerically unstable term and works equally well for a purely dead-reckoning server.

When the criteria suggests shifting the origin, the new origin is set to the last TOL state. The server is now free to marginalize out TOL states preceding the new origin. To help ensure that each client vehicle can maintain a reconstruction of the server pose-graph that contains the origin state, each server transmission encodes two OSPs: the standard OSP encoding the transition from the origin to the current TOL, and a backup OSP encoding the transition from the previous origin to the current origin. Depending on the available bandwidth, the server could transmit multiple backup packets to increase robustness, although in our implementation we use just one.

4) Recovery Packet: Passively shifting the origin limits the robustness of the OSM algorithm. The server can no longer guarantee that the client has received the origin TOL state (or the previous origin state, as described above). If the client vehicle has not received an update in a sufficiently long period of time, it will require a special information packet in order to recover. After receiving a client request, the server computes this special information as an additive 'delta information' (discussed in Section III-C) from the last TOL state that the client has received up to the current origin state. One implementation detail here, is that now the server must not marginalize out the oldest TOL states from its pose-graph unless it can guarantee that each client has received a more recent TOL in order to compute a recovery packet. Algorithm 2 summarizes the client-side operation.

C. Decentralized Extended Information Filter

The OSM algorithm uses the DEIF algorithm [28] update to fuse the client state estimate following OWTT range observations. The DEIF algorithm is a method in which a client vehicle can exactly reproduce the solution of a centralized filter for server-to-client cooperative networks. Essentially, the DEIF provides an efficient way to incorporate the newest server information in a delayed-state framework. The server vehicle maintains an information filter to fuse its local measurements, augmenting its state vector with each TOL position. Each 'delta information' encompasses all the local information that the server has gained between TOLs, computed as

$$\begin{aligned} \Delta\Lambda_{s_{t_n}} &= \Lambda_{s_{t_n}} - \Lambda_{s_{t_{n-1}}} \\ \Delta\boldsymbol{\eta}_{s_{t_n}} &= \boldsymbol{\eta}_{s_{t_n}} - \boldsymbol{\eta}_{s_{t_{n-1}}} \end{aligned} , \quad (18)$$

where the operation conforms for the dimensionality difference and the 's' subscript indicates the server's information. Delta information packets can be conceptually considered as expressing a transition on the server pose-graph from the previous TOL state to the current TOL state.

The client-side DEIF is driven by its local measurement updates and periodic (assumed non-faulty) delta information packets from the server vehicle, which the fault-tolerant OSM algorithm provides. The client-side DEIF tracks the current client state in addition to the set of server TOL states. Upon packet reception, the client vehicle simply adds the delta

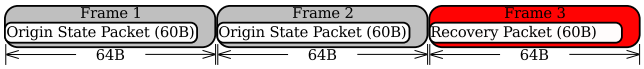


Fig. 3: Acoustic message composition. Each PSK Rate 1 and Rate 2 Micro-modem message contains three 64-byte frames. We pack these frames to hold two origin state packets and a reserved frame for a recovery packet.

information into its information filter

$$\begin{aligned} \Lambda_{c_{t_n}} &= \Lambda'_{c_{t_n}} + \Delta\Lambda_{s_{t_n}} \\ \boldsymbol{\eta}_{c_{t_n}} &= \boldsymbol{\eta}'_{c_{t_n}} + \Delta\boldsymbol{\eta}_{s_{t_n}} \end{aligned}, \tag{19}$$

where the 'c' subscript indicates the client-side information. Following the subsequent relative range measurement update, the client-side filter matches the corresponding centralized filter exactly. Full details of the algorithm are provided in [28].

IV. Field Trials

We present results for two experiments below. First, Experiment A demonstrates the effectiveness of the OSM algorithm for a three-node network including one server and two clients. Second, in Experiment B, we use a two-node network and purposely shift the origin forward at an accelerated rate to demonstrate the ability of the client vehicle to receive a recovery packet after losing communication.

A. Implementation Details

1) Experimental Setup: We fielded two Ocean-Server, Inc. Iver2 AUVs, designated AUV1 and AUV2, in our experiments. Each AUV is outfitted with an advanced dead-reckoning (DR) sensor suite including a 600 kHz RDI DVL, a Microstrain 3DM-GX3-25 attitude heading reference system (AHRS), and a Desert Star Systems SSP-1 digital pressure sensor. Throughout our experiments, AUV1 acts as the server, aiding AUV2. AUV1 is the only vehicle that observes GPS, when at the surface. To demonstrate the ability of our OSM algorithm to support multiple client vehicles, we also treated a topside ship (with only GPS reported velocity for input) as a client vehicle. We recorded two-way 25 kHz LBL, and GPS position fixes at the surface, for all vehicles for ground-truth comparison, although the client vehicles did not fuse these measurements.

2) Vehicle State Description: Since AUV attitude and depth are both instrumented with small bounded error, we focus on world-frame x, y horizontal position estimation. By transmitting pressure depth with each acoustic packet, OWTT range measurements can be projected into the horizontal plane. We are also motivated to maintain a minimal state size because of the limited acoustic modem channel capacity.

The state estimator on each vehicle tracks a state vector composed of its horizontal position; $\mathbf{x}_k = [x_k, y_k]^\top$. The state process is driven forward with an odometry measurement input, $\mathbf{u}_{k+1}$,

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{u}_{k+1}.$$

The odometry input and corresponding input covariance, Q_{k+1}, are obtained by Euler integrating DVL and AHRS measurements and performing a first-order covariance estimate as described in [8]. In the case of the topside client vehicle, world-frame velocity is integrated from GPS reported speed and track direction.

For the server vehicle, GPS reported x, y observations at vehicle surfacings are treated as linear observations of state. OWTT measurements, z_r, provide a range between the server TOL position and the client TOA position, with nonlinear observation model:

$$z_r = \left\| \mathbf{x}_{s_{\mathsf{TOL}}} - \mathbf{x}_{c_{\mathsf{TOA}}} \right\| + v,$$

where $v \sim \mathcal{N}(0, \sigma_r^2)$ represents the range measurement noise.

3) Acoustic Communication Considerations: Each vehicle is outfitted with a WHOI Micro-modem [12] and a co-processor board capable of encoding multiple frame, higher bandwidth phase-shift keying (PSK) data packets. Each origin state packet requires 60 bytes. Each double precision element of the origin state information is rounded to a precision of 10^{-5} to reduce the packet size. Furthermore, since the information matrix is symmetric, only the upper diagonal elements are transmitted. Both Micro-modem PSK Rate 1 and Rate 2 messages allow the user to transmit three 64 byte frames (Fig. 3). We fill the first two frames of Rate 1 and Rate 2 messages with the two OSM packets as discussed in Section III-B3. If a client vehicle has requested a recovery packet, we transmit the custom recovery packet in the remaining third frame, so that normal operation continues for vehicles that do not require a recovery step.

We employed a fixed TDMA cycle, whereby all vehicles were assigned a communication slot. The server vehicle transmitted three OSPs per two minute TDMA period, while the client transmitted a single data packet, used for recovery requests. The average client-side reception rates in different experiments varied between 36.96% and 64.47%.

B. Experiment A

Experiment A was conducted to demonstrate a typical application scenario in which inter-vehicle correlation is high and must be properly tracked to obtain the optimal result—a server AUV that periodically surfaces to receive GPS while aiding another subsea AUV with no GPS. During two trials (A.1 and A.2), AUV1 acted as server and drove in a diamond-box pattern around AUV2's lawn-mower survey pattern (Fig. 4). Each leg of the lawn-mower survey was roughly 500 m in length. Both AUVs maintained a depth of approximately 5 m with AUV1 occasionally coming to the surface at the apex of

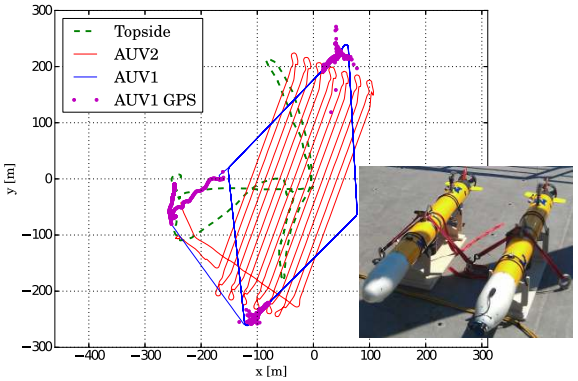
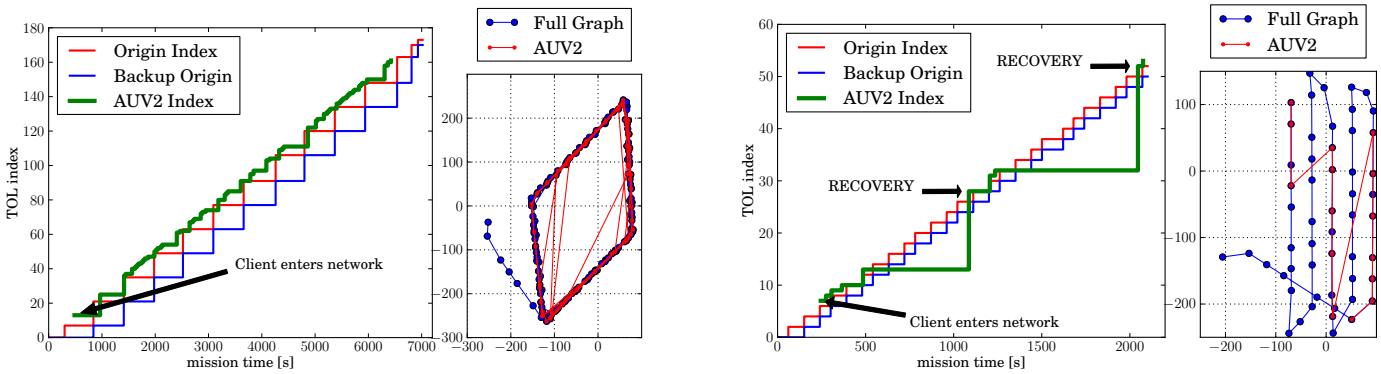


Fig. 4: Experiment A: vehicle trajectories during multi-vehicle field trial.



(a) Experiment A.1

(b) Experiment B

Fig. 5: Pose-graph reconstruction. The left plot in each subfigure shows the origin index from both the current origin state packet and the backup packet. Also plotted is the index of the most recent TOL that the client has received. When the client index dips below the transmitted origin, it indicates that the client used the backup packet. If the client index is less than the backup packet origin, the client would require a recovery packet (this occurs in Experiment B). The right plot of each subfigure illustrates the full server pose-graph with the client-side reconstruction (a subset of the full) overlaid.

the diamond pattern. The topside ship acted as a client in the first trial (A.1) of this experiment, drifting around the survey area and periodically driving to new locations. Since we did not use topside GPS position for state, it serves as a proxy for a vehicle with extremely poor navigation as compared to the server vehicle.

Table I lists the mean norm difference between TOL states in the server pose-graph (constructed onboard the server) and the client's reconstructed server pose-graph. In all cases, the client accurately reconstructs the server pose-graph despite low reception rates. The maximum error during the experiment is roughly $10\,\mathrm{cm}$. This difference is due to compounded numerical inaccuracies in the OSM algorithm and round-off errors in acoustic transmission. Using the alternate trace-based origin shifting criteria (designated 'trace'), both the mean and maximum error in the pose-graph reconstruction are reduced.

During both 2 hour long trials, neither the topside client nor AUV2 required a recovery packet. The origin was shifted forward by the server roughly every 14 TOL states corresponding to a time of about 9 minutes. The transmitted TOL indices as well as the reconstructed server pose-graph for trial A.1 are shown in Fig. 5(a). While on average the server shifted the origin forward more frequently with the trace-based shifting method, AUV2 still would not have required a recovery packet.

Table II summarizes the client position error relative to LBL/GPS. These error values serve as a basis for comparison between DR and the OSM algorithm. Since the LBL/GPS data itself is noisy, the reported errors are not absolute error measures. In trial A.1, topside DR is quite poor as it only fuses GPS-derived speed and heading. The OSM algorithm provides an obvious improvement over pure DR for both topside and subsea clients as it is able to bound the client error growth.

TABLE I: Mean norm diff. Server Pose-Graph and Client Reconstruction

	Exp. A.1	Exp. A.2	Exp. B
AUV2 OSM (RV)	0.013454 m	0.002761 m	0.000010 m
AUV2 OSM (trace)	0.000631 m	0.000984 m	NA
Topside OSM (RV)	0.001880 m	NA	NA

TABLE II: Mean norm diff. Client Estimate and LBL/GPS Fixes

	Exp. A.1		Exp. A.2		Exp. B
	LBL [m]	GPS [m]	LBL [m]	GPS [m]	GPS [m]
AUV2 DR	11.72	9.50	13.36	8.55	16.47
AUV2 OSM (RV)	6.08	7.50	9.82	4.63	7.71
Topside DR	251.36	267.22	NA	NA	NA
Topside OSM (RV)	19.95	19.56	NA	NA	NA

C. Experiment B

Experiment B consisted of two intersecting lawn-mower surveys run by the server and client, AUV1 and AUV2, respectively. Each leg of both lawn-mower surveys is roughly $300\,\mathrm{m}$ in length. We artificially forced the server vehicle to shift the origin forward at a much faster rate (every two transmissions) to demonstrate the ability of the client vehicle to request and receive a recovery packet. LBL was not available for comparison during this experiment.

During the 45 minute experiment, the client received several origin state packets that it could not integrate into its reconstructed server pose-graph because it had not received the server's origin state. The client received two recovery packets during the experiment and successfully added the recovery information. As can be seen in Fig. 5(b) and Table I, the client (AUV2) is able to reliably reconstruct the server pose-graph using the recovery information. Moreover, the position difference of AUV2 relative to GPS is tabulated in Table II, and shows that the OSM algorithm running onboard AUV2 is able to reduce its mean position error by half over DR.

D. Nonlinear Algorithm Comparison

The OSM constitutes a smoothing algorithm built upon a delayed-state information filter. Our algorithm has an obvious advantage over egocentric methods in that it produces a consistent estimate by tracking correlation between the server and client vehicles. Moreover, the OSM reproduces the centralized filter estimate up to communication round-off and pose-graph reconstruction errors.

While an extended information filter is sub-optimal for nonlinear estimation, our solution compares well to nonlinear

TABLE III: Delayed-state filter TOA estimate comparison to iSAM

	Central	OSM (RV)	OSM (trace)	Ego
Mean norm diff. [m]	0.1030	0.1295	0.1229	1.5721
Mean KLD [nats]	0.0162	0.0207	0.0187	2.5700

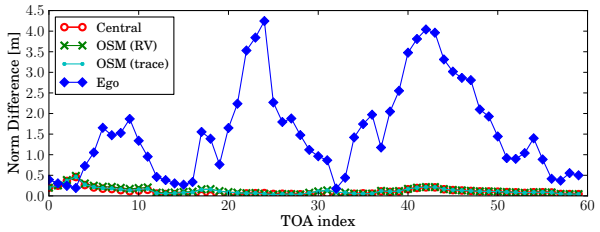


Fig. 6: Norm difference between estimated client TOA states and iSAM result.

smoothing algorithms such as incremental smoothing and mapping (iSAM), [14]. The delayed-state information filter is equivalent to a nonlinear least-squares approach with the exception that measurement constraints are only linearized once. This difference is small in our implementation as all odometry constraints are linear, and infrequent range measurements account for the only nonlinearities.

We treat the fully nonlinear iSAM solution as the gold-standard benchmark. Fig. 6 and Table III summarize the difference between the delayed-state filter client TOA pose estimates and iSAM, as well as illustrate the ability of the centralized EKF [27] and the OSM to reproduce the iSAM distribution over client states at the TOA. The small difference between the centralized filter and OSM is due to small numerical errors during the server pose-graph reconstruction. The OSM performs much better than the egocentric approach in its ability to match both the client mean and covariance estimates. The egocentric approach differs in mean and is also overconfident in its uncertainty estimate.

V. Conclusion

We presented the first-ever practical real-time method allowing a client vehicle to exactly reproduce the centralized filter estimate under an extremely faulty and bandwidth limited underwater acoustic communication channel. We validated this decentralized estimation algorithm with several real-world experiments, and showed it to closely reproduce the fully nonlinear least-squares estimate obtained by iSAM.

References

[1] A. Bahr, J. J. Leonard, and M. F. Fallon. Cooperative localization for autonomous underwater vehicles. *Int. J. Robot. Res.*, 28(6):714–728, 2009.

[2] A. Bahr, M. R. Walter, and J. J. Leonard. Consistent cooperative localization. In *Proc. IEEE Int. Conf. Robot. and Automation*, pp. 3415–3422, May 2009.

[3] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte. Decentralised cooperative localisation for heterogeneous teams of mobile robots. In *Proc. IEEE Int. Conf. Robot. and Automation*, pp. 2859–2865, May 2011.

[4] A. Cunningham, M. Paluri, and F. Dellaert. DDF-SAM: Fully distributed SLAM using constrained factor graphs. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, pp. 3025–3030, Oct. 2010.

[5] F. Dellaert, F. Alegre, and E. B. Martinson. Intrinsic localization and mapping with 2 applications: Diffusion mapping and marco polo localization. In *Proc. IEEE Int. Conf. Robot. and Automation*, pp. 2344–2349, Sept. 2003.

[6] R. M. Eustice, H. Singh, and J. J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Trans. Robot.*, 22(6):1100–1114, 2006.

[7] R. M. Eustice, L. L. Whitcomb, H. Singh, and M. Grund. Recent advances in synchronous-clock one-way-travel-time acoustic navigation. In *Proc. IEEE/MTS OCEANS Conf. Exhib.*, Sept. 2006.

[8] R. M. Eustice, H. Singh, and L. L. Whitcomb. Synchronous-clock one-way-travel-time acoustic navigation for underwater vehicles. *J. Field Robot.*, 28(1):121–136, 2011.

[9] M. F. Fallon, G. Papadopoulis, and J. J. Leonard. A measurement distribution framework for cooperative navigation using multiple AUVs. In *Proc. IEEE Int. Conf. Robot. and Automation*, pp. 4256–4263, May 2010.

[10] M. F. Fallon, M. Kaess, H. Johannsson, and J. J. Leonard. Efficient AUV navigation fusing acoustic ranging and side-scan sonar. In *Proc. IEEE Int. Conf. Robot. and Automation*, pp. 2398–2405, May 2011.

[11] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, pp. 325–344, 2000.

[12] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball. The WHOI micro-modem: An acoustic communications and navigation system for multiple platforms. In *Proc. IEEE/MTS OCEANS Conf. Exhib.*, pp. 1–7, 2005.

[13] A. Howard, M. J. Matari, and G. S. Sukhatme. Localization for mobile robot teams using maximum likelihood estimation. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, volume 1, pp. 434–439, 2002.

[14] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Trans. Robot.*, 24(6):1365–1378, 2008.

[15] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller. Multiple relative pose graphs for robust cooperative mapping. In *Proc. IEEE Int. Conf. Robot. and Automation*, pp. 3185–3192, 2010.

[16] D. K. Maczka, A. S. Gadre, and D. J. Stilwell. Implementation of a cooperative navigation algorithm on a platoon of autonomous underwater vehicles. In *Proc. IEEE/MTS OCEANS Conf. Exhib.*, pp. 1–6, Oct. 2007.

[17] S. McPhail and M. Pebody. Range-only positioning of a deep-diving autonomous underwater vehicle from a surface ship. *IEEE J. Ocean. Eng.*, 34(4):669–677, Oct. 2009.

[18] P. H. Milne. *Underwater acoustic positioning systems*. Gulf Publishing Company, Houston, 1983.

[19] C. A. Murphy. *Progressively Communicating Rich Telemetry from Autonomous Underwater Vehicles via Relays*. PhD thesis, Massachusetts Inst. Tech. / Woods Hole Ocean. Inst. Joint Program, Cambridge, MA, June 2012.

[20] E. D. Nerurkar, K. X. Zhou, and S. I. Roumeliotis. Hybrid estimation framework of multi-robot cooperative localization using quantized measurements. Technical report, Dept. of Comp. Sci. & Eng., University of Minnesota, 2011.

[21] J. Partan, J. Kurose, and B. N. Levine. A survey of practical issues in underwater networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(4):23, Oct. 2007.

[22] A. Ribeiro, G. B. Giannakis, and S. I. Roumeliotis. SOI-KF: Distributed kalman filtering with low-cost communications using the sign of innovations. *IEEE Trans. Signal Process.*, 54(12):4782–4795, 2006.

[23] P. Robert and Y. Escoufier. A unifying tool for linear multivariate statistical methods: The RV-coefficient. *J. Royal Statistical Society. Series C (Applied Statistics)*, 25(3):257–265, 1976.

[24] S. I. Roumeliotis and G. A. Bekey. Distributed multirobot localization. *IEEE Trans. Robot. Autom.*, 18(5):781–795, 2002.

[25] J. Vaganay, J. J. Leonard, J. A. Curcio, and J. S. Wilcox. Experimental validation of the moving long base-line navigation concept. In *Proc. IEEE/OES Autonomous Underwater Vehicles Conf.*, pp. 59–65, 2004.

[26] J. M. Walls and R. M. Eustice. An origin state method for lossy synchronous-clock acoustic navigation. In *IFAC Workshop Nav., Guidance, Cont. Underwater Vehicles*, Apr. 2012.

[27] S. E. Webster, R. M. Eustice, H. Singh, and L. L. Whitcomb. Advances in single-beacon one-way-travel-time acoustic navigation for underwater vehicles. *Int. J. Robot. Res.*, 31(8):935–950, 2012.

[28] S. E. Webster, J. M. Walls, L. L. Whitcomb, and R. M. Eustice. Decentralized extended information filter for single-beacon cooperative acoustic navigation: Theory and experiments. *IEEE Trans. Robot.*, 2013. In Print.

[29] L. L. Whitcomb, D. R. Yoerger, and H. Singh. Combined doppler/LBL based navigation of underwater vehicles. In *Proc. Int. Symp. Unmanned Untethered Subm. Tech.*, 1999.