An Origin State Method for Communication Constrained Cooperative Localization with Robustness to Packet Loss

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 broadcast its navigation information to multiple 'client' vehicles over a bandwidth-limited and faulty communication channel. The server's broadcasted pose-graph can be used in conjunction with an estimator on the client to exactly reproduce the corresponding server-client centralized estimate. We present an evaluation over an extensive 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.

1 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 vehicles (servers) to cooperatively aid the navigation of subsea vehicles (clients), which is robust to the packet loss and low-bandwidth that is endemic in underwater acoustic communication networks. Our algorithm is capable of bounding the position 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 (Milne, 1983; Whitcomb et al., 1999). The relative range is measured using two-way time-of-flight (TOF) acoustic broadcasts and assuming a known sound-speed profile. Narrowband LBL beacon networks, however, are limited in their ability to scale to many vehicles because 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. In the same vain, ultra-short-baseline (USBL) systems allow a topside ship to observe the relative range and bearing of a subsea vehicle, but are similarly limited in scalability.

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 (Eustice et al., 2011). 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 broadcast—OWTT networks scale well. The use of OWTT observations to augment vehicle navigation presents 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 (Partan et al., 2007). Acoustic communication is constrained by high 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 $100\ \mathrm{bps}$ (Murphy, 2012), 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 between platforms in the network. All of these challenges amount to a communication framework that enforces small-payloads and infrequent updates between vehicles.

Draft manuscript, July 16, 2015.

Portions of this work have appeared previously in (Walls and Eustice, 2012, 2013).

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

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

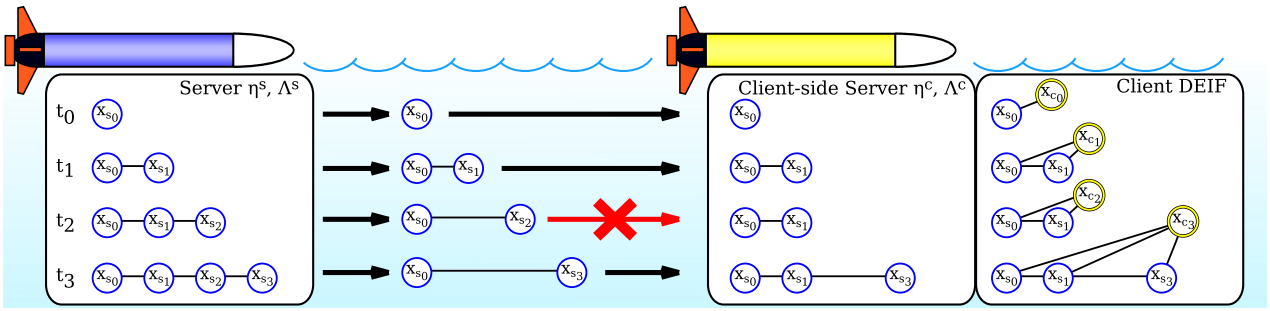


Fig. 1: Origin state method algorithm overview. The server (blue) fuses its local observations and adds delayed-state poses at each time-of-launch (TOL) (the blue circles). The server uses our novel origin state method to incrementally broadcast its pose-graph in a fault-tolerant way. At the time-of-arrival (TOA) 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 still reconstructs the server pose-graph after receiving the origin state packet at t_3, encapsulating all server information accumulated between t_1 and t_3.

A variety of cooperative localization frameworks exist for improving position estimates across a team of robots by sharing local navigation information. Previous methods, however, do not address the severely limited bandwidth and fragility of the underwater acoustic communication channel. In this paper, we consider a solution to the navigation of a client vehicle aided by a server platform, as depicted in Fig. 1. We present the following contributions:

- We present a general algorithm, called the origin state method (OSM), that allows multiple servers to broadcast their pose-graph via a faulty, low bandwidth communication channel that works in real-time for practical, underwater, acoustic networks.
- We use the OSM algorithm to compute a 'delta information' over server TOL poses, which serves as input to a decentralized extended information filter (DEIF) algorithm (Webster et al., 2013) that exactly computes the result of the centralized extended information filter (CEIF) onboard the client up to communication delay.
- We provide extensive evaluation over more than $12\ \mathrm{h}$ of field trials demonstrating the ability of the OSM to robustly broadcast a server pose-graph to multiple clients using a small, fixed-bandwidth data packet.
- We provide a novel factor-based interpretation of the 'delta information' and discuss how it can be used in real-time in other estimation frameworks such as the nonlinear least-squares incremental smoothing and mapping (iSAM) framework (Kaess et al., 2008).

The remainder of this paper proceeds as follows. Section 2 reviews the prior literature within cooperative localization. Section 3 summarizes the proposed OSM algorithm. Section 4 describes the interface between the OSM and the DEIF. Section 5 presents the results of more than $12\ \mathrm{h}$ of field trials performed with multiple autonomous underwater vehicles (AUVs). Finally, Section 7 concludes.

2 Related Work

Cooperative vehicle networks enable robots with the best navigation sensors to localize robots with poorer position estimates. The goal is to augment each platform's local sensing with measurement constraints between the vehicles themselves as depicted in Fig. 2. Prior literature is discussed below and summarized in Table 1.

Simple, real-time algorithms that require minimal bandwidth are within the egocentric class of filters (Fox et al., 2000; Maczka et al., 2007; Vaganay et al., 2004). These algorithms scale by treating each inter-vehicle 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 through relative observations between robot estimates, which can lead to inconsistent (i.e., overconfident) estimates (Maczka et al., 2007). The negative consequences of ignoring correlation have been demonstrated by Roumeliotis and Bekey (2002), Bahr et al. (2009b), and Walls and Eustice (2011).

Covariance intersection (Julier and Uhlmann, 1997) can be used to consistently fuse two estimates with unknown correlation. Recently, it has been applied to the cooperative localization problem by Li and Nashashibi (2013) and Carrillo-Arce et al. (2013) to cope with inconsistency in egocentric algorithms. However, previous work requires a full rank (i.e., range and bearing) relative observation. Bahr et al. (2009b) previously noted the challenge of incorporating partial relative pose information in cooperative frameworks with covariance intersection.

Bahr et al. (2009b) and Fallon et al. (2010a) propose

Table 1: Summary of prior algorithms for multiple platform navigation, specifically within the context of server-client communication topologies. No previous method is able to reproduce the centralized solution while being robust to communication packet loss.

	Literature	Online/ real-time	Packet loss tolerant	Bandwidth conservative [a]	Reproduces centralized	Consistent estimate	Comments
Centralized	Howard et al. (2002)	no	—	—	yes	yes	Centralized MLE.
	Dellaert et al. (2003)	no	—	—	yes	yes	Centralized MLE.
	Webster et al. (2012)	no	—	—	yes	yes	Centralized EKF.
Egocentric	Fox et al. (2000)	yes	yes	yes	no	no	Sampling-based approach.
	Vaganay et al. (2004)	yes	yes	yes	no	no	Moving LBL paradigm.
	Maczka et al. (2007)	yes	yes	yes	no	no	Egocentric KF.
Distributed	Roumeliotis and Bekey (2002)	yes	no	no	yes	yes	Distributed EKF.
	Ribeiro et al. (2006)	yes	no	yes	yes	yes	Quantized innovations.
	Bahr et al. (2009b)	yes	yes	yes	no	yes	Bank of estimators.
	Fallon et al. (2010a)	yes	yes	no	yes	yes	Requires acknowledgements.
	Cunningham et al. (2010, 2012)	yes	yes	no	yes	yes	Transmits reduced pose-graph.
	Kim et al. (2010)	yes	yes	no	yes	yes	Transmits entire server graph.
	Leung et al. (2010)	yes	yes	no	yes	yes	Transmits knowledge sets.
	Webster et al. (2010, 2013)	yes	no	yes	yes	yes	Transmits delta information.
	Nerurkar et al. (2011)	yes	no	yes	yes	yes	Sign-of-innovations.
	Bailey et al. (2011)	yes	no	yes	yes	yes	Transmits delta information.
	Origin State Method (this paper)	**yes**	**yes**	**yes**	**yes**	**yes**	Can be used in conjunction with EIF or iSAM.

[a] We consider an algorithm bandwidth conservative if it employs a fixed-bandwidth data packet and does not require acknowledgement.

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. (2006) and Nerurkar et al. (2011) achieve consistency through a noteworthy approach 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 full packet reception, which is unrealistic for faulty communication channels.

The most general cooperative localization algorithms estimate the full joint distribution over all vehicle poses (Fig. 2), and can generally be realized through centralized estimators in post-process or high-bandwidth systems in real-time (Howard et al., 2002; Dellaert et al., 2003). Roumeliotis and Bekey (2002) developed a distributed extended Kalman filter (EKF)-based method, though it requires moderately high-bandwidth, and two-way information exchange. Cunningham et al. (2010, 2012) and Kim et al. (2010) studied the problem of nonlinear simultaneous localization and mapping (SLAM) in a distributed fashion using smoothing and mapping where each platform *(i)* transmits its full local pose-graph (or a representative subset), *(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 estimator solution at the expense of high communication cost, which grows with the size of the local graph. Leung et al. (2010) exploits the Markov property to reduce the required information exchange within a recursive Bayesian filter.

Webster et al. (2012) presented a post-process centralized EKF specifically designed for synchronous-clock acoustic cooperative localization. They later distributed this centralized filter result exactly (Webster et al., 2013) with a 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 (the 'delta state' information) to the client where the centralized filter solution is reproduced. Bailey et al. (2011) 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 (Walls and Eustice, 2012) a preliminary method toward alleviating the non-faulty communication constraint in distributing local server information, but which still relied upon a client acknowledgment scheme—limiting scalability to multiple clients.

Several other works in acoustic cooperative underwater navigation have emerged for fusing OWTT-based relative ranges in server-client communication topologies (Vaganay et al., 2004; Maczka et al., 2007; McPhail and Pebody, 2009; Bahr et al., 2009a,b; Eustice et al.,

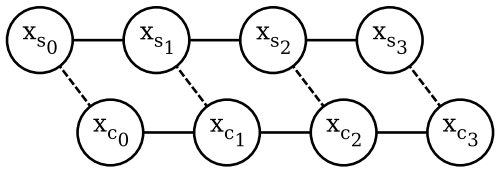


Fig. 2: Joint pose-graph over server, $\mathbf{x}_{s_i}$, and client, $\mathbf{x}_{c_j}$, poses. Dashed lines illustrate edges generated by relative range observations between server TOL poses and client TOA poses. In order to compute the full pose-graph, the client must have access to the server's local pose-graph.

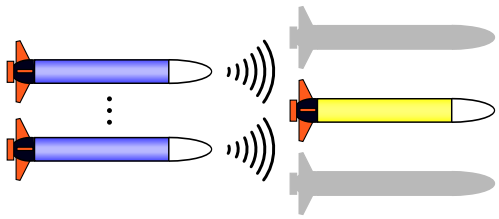


Fig. 3: Supported communication topology. One or more server vehicles (left, blue) broadcast state information to a network of client vehicles (right, yellow). The OSM algorithm allows each client to reproduce the corresponding centralized estimate. Servers can support many clients in parallel.

2011; Fallon et al., 2010b, 2011). Previously considered 'single-beacon' cooperative localization or the use of cooperative navigation aids (CNAs) are equivalent to the server-client scenario explored here. Earlier methods, however, generally compromise between offline and consistent (by estimating the full joint distribution), or real-time and inconsistent (by ignoring correlation between relative range observations). Note that online methods that neglect correlation are, in fact, exact when the server positions are actually independent. While many of these algorithms have been validated in post-process using experimental data, only a few have been presented with real-time field trials including Maczka et al. (2007), Fallon et al. (2010b), and Fallon et al. (2010a).

Our work is closest to Webster et al. (2013), Bailey et al. (2011), and Walls and Eustice (2012) in its effort to distribute local data fusion in an optimal way and to leverage the sparsity of the Gaussian information form to compactly broadcast this information. We build upon the previously reported approach in Walls and Eustice (2012) by *(i)* improving network scalability via a *passive* origin shifting scheme that eliminates the need for client-side acknowledgments, *(ii)* introducing a recovery packet mechanism that enables clients to enter and leave the network or recover after a long period of communication dropout, and *(iii)* presenting several AUV trials demonstrating our algorithm's ability for real-time underwater navigation.

3 Consistent Cooperative Localization

We consider one or more independent server vehicles aiding the navigation of multiple client vehicles, such as is depicted in Fig. 3. For the sake of presentation, we refer to a single server vehicle, although our algorithm can support multiple, (the multiple server scenario is demonstrated in Section 5). The client vehicles are able to passively observe their range to the server vehicle during periodic server broadcasts. Each client updates its pose estimate using its local information, the range observations to the servers, and the information broadcast by each server. A centralized estimator, for example Webster et al. (2012), 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. The centralized solution includes information from all servers and a single client vehicle, but not information from the other client vehicles. Our formulation is able to reproduce this centralized delayed-state filter result onboard each client vehicle in real-time for the server and individual client states.

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. All vehicles synchronize their local clocks to GPS time while at the surface. Low-drift reference clocks enable the vehicles to remain synchronized throughout operation. During our trials we used a SeaScan Inc. temperature compensated crystal oscillator (TXCO), which provides a stable reference pulse with approximately $1\ \mathrm{ms}$ drift over $14\ \mathrm{h}$ (Eustice et al., 2011). Newer commercially available free-running clocks promise several orders of magnitude improved performance, for example, Symmetricomm (2013) provides a $120\ \mathrm{mW}$ chip scale atomic clock with less than $1\ \mathrm{ms}$ drift over $5000\ \mathrm{h}$ ($\sim$208 days).

Previously reported algorithms that are able to compute a consistent estimate track the joint distribution over both client *and* server poses, i.e., the joint pose-graph over the client and server (see Fig. 2). Although the client may only be interested in computing its own state estimate, tracking the server's pose-graph allows the client to track correlation between successive relative range observations. In general, the server must transmit its full local pose-graph to the client at the time of each new relative measurement in order for the client to compute this full solution. Since the size of the server pose-graph continually grows, transmitting the full server information is not feasible in a communication constrained domain. OSM supplies a fixed-bandwidth representation of new server poses that allows each client to asynchronously reconstruct the server pose-graph despite high packet loss. Each server broadcast contains all new local information relative to a server state known by the client, termed the origin state. The client then reconstructs the server pose-graph and can compute the centralized solution. Fig. 1 provides a graphical overview of the OSM algorithm.

3.1 Information Filter

The OSM algorithm relies upon manipulating a Gaussian distribution in the information form to efficiently broadcast the server pose-graph. We assume that the server state evolves with linear Gaussian noise models. Client process and measurement models, however, can be fully nonlinear. The server information will there-

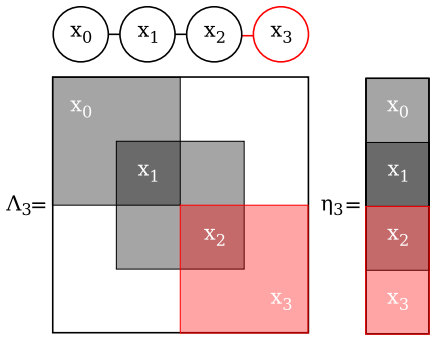


Fig. 4: Server pose-graph example. The red area of the information matrix illustrates that measurements occurring between the previous TOL, $\mathbf{x}_2$, and the current TOL, $\mathbf{x}_3$, contribute additively to a subblock of the information matrix. The information matrix sparsity pattern corresponds to the adjacency matrix of the pose-graph.

fore have a known and predictable structure that we can leverage to robustly broadcast it. Although we may use any estimator that satisfies our assumptions, we employ a delayed-state information filter to initially construct the server pose-graph.

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

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

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

The representation of the delayed-state collection is termed the pose-graph (Fig. 4). Nodes in the graph express delayed-state variables while edges encode dependencies between nodes. The sparsity pattern of the information matrix corresponds exactly to the adjacency matrix of the pose-graph.

The single vehicle navigation problem is framed in terms of estimating the joint distribution over a collection of historic poses (i.e., past vehicle states). In this case, the state vector is composed of these historic poses, termed 'delayed-states',

$$\boldsymbol{x} = \left[\mathbf{x}_1^\top, \ldots, \mathbf{x}_{n-1}^\top, \mathbf{x}_n^\top\right]^\top,$$

corresponding to the distribution $p(\boldsymbol{x})$. The information filter state vector grows over time by performing prediction with augmentation. As noted in Eustice et al. (2006), processes that evolve sequentially with the Markov property, i.e.,

$$p(\boldsymbol{x}) = p(\mathbf{x}_n|\mathbf{x}_{n-1}) \cdots p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_1),$$

admit 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, as depicted in Fig. 4. New odometry inputs

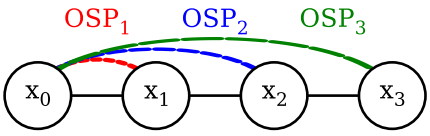


Fig. 5: Pose-graph over server TOL poses. The OSM algorithm allows the client to reconstruct the underlying server pose-graph (horizontal lines) from the set of received OSPs (colored arcs). Each OSP encodes the transition from $\mathbf{x}_0$, which represents the origin state.

and local measurements only modify a block of the information matrix and vector corresponding to the current robot pose and the most recent delayed-state (i.e., only the value of the pose-graph edge between the last delayed-state and the current state is affected). Herein, we assume that new poses are appended onto the state vector.

3.2 Origin State Method

Our goal is to broadcast information that allows the client to reconstruct the server pose-graph. However, the server does not have any knowledge of the information that the client has actually received, because communication is broadcast and unacknowledged. The underlying assumption is that the server pose-graph grows as a Markov chain—the standard model for a dynamical system. Loop closures, popular in SLAM literature, are not supported by this model since their pose-graph breaks this chain assumption. Each origin state broadcast, called an origin state packet (OSP), encodes a server transition from the origin state to the current state (Fig. 5). The OSP represents the relationship between the origin and current state as their joint marginal distribution, i.e., the two-node pose-graph over the origin and current TOL states. We show here that the client can incrementally reconstruct the server's pose-graph from the sequence of received OSPs.

3.2.1 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. 5. The index label of the new TOL state and the origin are also broadcast to the client for reconstruction.

We can partition the set of intermediate server TOL states occurring between the origin state, $\mathbf{x}_o$, and the n^{th} TOL state, $\mathbf{x}_n$, into the set of states received and not received by the client, $\boldsymbol{x}_r$ and $\boldsymbol{x}_{\bar{r}}$, respectively. The server pose-graph at the n^{th} TOL then represents the distribution over the state vector

$$\boldsymbol{x}_n^s = \left[\mathbf{x}_o^\top, \mathbf{x}_{r\cup\bar{r}}^\top, \mathbf{x}_n^\top\right]^\top, \tag{2}$$

where the 's' superscript indicates the distribution as tracked by the server. Note that the server has no knowledge of the partition r and $\bar{r}$. This distribution is expressed

$$p^s(\mathbf{x}_o, \mathbf{x}_{r\cup\bar{r}}, \mathbf{x}_n|\mathbf{Z}^n) = \mathcal{N}^{-1}\big(\boldsymbol{x}_n^s; \boldsymbol{\eta}_n^s, \Lambda_n^s\big),$$

$$\Lambda_n^s = \begin{bmatrix} \Lambda_{o,o} & \Lambda_{o,r\cup\bar{r}} & 0 \\ \Lambda_{r\cup\bar{r},o} & \Lambda_{r\cup\bar{r},r\cup\bar{r}} & \Lambda_{r\cup\bar{r},n} \\ 0 & \Lambda_{n,r\cup\bar{r}} & \Lambda_{n,n} \end{bmatrix}, \tag{3}$$

$$\boldsymbol{\eta}_n^s = \begin{bmatrix} \boldsymbol{\eta}_o \\ \boldsymbol{\eta}_{r\cup\bar{r}} \\ \boldsymbol{\eta}_n \end{bmatrix}, \tag{4}$$

where $\mathbf{Z}^i$ is the set of all observations up to the i^{th} time index.

Under the OSM, the server computes an OSP at every TOL, which is simply the marginal distribution corresponding to the state vector

$$\boldsymbol{x}_{\text{OSP}_n}^s = \big[\mathbf{x}_o^\top, \mathbf{x}_n^\top\big]^\top, \tag{5}$$

computed via the Schur complement of (3) and (4):

$$\begin{aligned} p_{\text{OSP}_n}^s(\mathbf{x}_o, \mathbf{x}_n|\mathbf{Z}^n) &= \int_{\mathbf{x}_{r\cup\bar{r}}} p^s(\mathbf{x}_o, \mathbf{x}_{r\cup\bar{r}}, \mathbf{x}_n|\mathbf{Z}^n) d\mathbf{x}_{r\cup\bar{r}} \\ &= \mathcal{N}^{-1}\big(\boldsymbol{x}_{\text{OSP}_n}^s; \boldsymbol{\eta}_{\text{OSP}_n}^s, \Lambda_{\text{OSP}_n}^s\big), \end{aligned}$$

$$\Lambda_{\text{OSP}_n}^s = \begin{bmatrix} \Lambda_{o,o}^s & \Lambda_{o,n}^s \\ \Lambda_{n,o}^s & \Lambda_{n,n}^s \end{bmatrix}, \tag{6}$$

$$\boldsymbol{\eta}_{\text{OSP}_n}^s = \begin{bmatrix} \boldsymbol{\eta}_o^s \\ \boldsymbol{\eta}_n^s \end{bmatrix}. \tag{7}$$

This formulation allows the server to remain ignorant about states the client has received, r. Moreover, it allows the server to send useful information to multiple clients, where each client has a different set of received server TOL states. In order for the client to reconstruct the server pose-graph, the client must already have the origin state in its representation, (i.e., the origin is a previously received TOL state). The index label of the current TOL, n, and the origin, o_n, are included in the broadcast.

Algorithm 1 summarizes the server-side operation. The server simply maintains an information filter over its pose-graph, augmenting its state vector with each new TOL pose. At each TOL the server computes an OSP to broadcast to the client. Origin shifting and recovery is introduced and discussed below.

Although, in general, the size of the server pose-graph grows with the addition of each new TOL pose, the dimension of the OSP marginal information matrix and vector is fixed. The OSP dimension is twice the state dimension—therefore, a minimal vehicle state size is desirable to reduce acoustic communication packet size.

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**
    $\Lambda_n^s, \eta_n^s, o_n \leftarrow$ `originPacket`$(\Lambda_k, \boldsymbol{\eta}_k)$
    if $o_n \neq o_{n-1}$ **then**
      {origin has been shifted, update backup packet}
      $\Lambda_b, \boldsymbol{\eta}_b, o_b \leftarrow \Lambda_{n-1}^s, \eta_{n-1}^s, o_{n-1}$
    end if
    `broadcastOriginPacket`$(\Lambda_n^s, \eta_n^s, o_n, \Lambda_b, \boldsymbol{\eta}_b, o_b)$
    if `recoveryRequired`() **then**
      `broadcastRecovery`() {Section 3.2.4}
    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

3.2.2 *Client-side Origin State Operation*

The client incrementally reconstructs the server pose-graph from the sequence of successfully received OSPs. Before the n^{th} TOL, the client-side version of the server pose-graph reconstruction contains the server states

$$\begin{aligned} \boldsymbol{x}_{r_m}^c &= \big[\mathbf{x}_o^\top, \mathbf{x}_r^\top\big]^\top \\ &= \big[\mathbf{x}_o^\top, \mathbf{x}_{r'}^\top, \mathbf{x}_{r_m}^\top\big]^\top, \end{aligned}$$

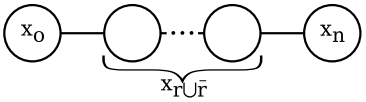
where the 'c' superscript indicates the server distribution as tracked by the client, $r = \{r_1, \ldots, r_m\}$ denotes the set of received server TOL indices, r_m is the previously received TOL index, and $r' = \{r_1, \ldots, r_{m-1}\}$ represents other previously received server TOL indices. To simplify notation, we let $m = r_m$ for the remainder of the discussion. Note that the client has no representation for server states corresponding to missed TOL poses, $\mathbf{x}_{\bar{r}}$. This is equivalent to the server's distribution at m with states in $\mathbf{x}_{\bar{r}}$ marginalized out,

$$p^c(\mathbf{x}_o, \mathbf{x}_{r'}, \mathbf{x}_m|\mathbf{Z}^m) = \mathcal{N}^{-1}\big(\boldsymbol{x}_m^c; \boldsymbol{\eta}_m^c, \Lambda_m^c\big),$$

$$\Lambda_m^c = \begin{bmatrix} \Lambda_{o,o} & \Lambda_{o,r'} & 0 \\ \Lambda_{r',o} & \Lambda_{r',r'} & \Lambda_{r',m} \\ 0 & \Lambda_{m,r'} & \Lambda_{m,m} \end{bmatrix}, \tag{8}$$

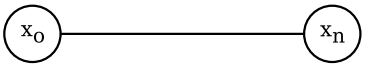
$$\boldsymbol{\eta}_m^c = \begin{bmatrix} \boldsymbol{\eta}_o \\ \boldsymbol{\eta}_{r'} \\ \boldsymbol{\eta}_m \end{bmatrix}. \tag{9}$$

After receiving the n^{th} OSP, the client solves the following problem:



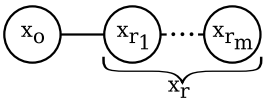
$p^s(\mathbf{x}_o, \mathbf{x}_{r\cup\bar{r}}, \mathbf{x}_n|\mathbf{Z}^n)$

(a) Server-side pose-graph at the current TOL



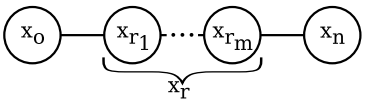
$p^s_{\mathrm{OSP}_n}(\mathbf{x}_o, \mathbf{x}_n|\mathbf{Z}^n)$

(b) Server-side OSP



$p^c(\mathbf{x}_o, \mathbf{x}_r|\mathbf{Z}^m)$

(c) Client-side reconstruction through the last received OSP



$p^c(\mathbf{x}_o, \mathbf{x}_r, \mathbf{x}_n|\mathbf{Z}^n)$

(d) Client-side goal information

Fig. 6: Illustration of OSM operation. (a) represents the server's pose-graph at the n^{th} TOL. The server broadcasts an OSP (b) to the client. The client has already reconstructed a portion of the server graph, (c), having missed TOL poses, $\mathbf{x}_{\bar{r}}$. The client then reconstructs the server goal information illustrated in (d) by fusion of (b) and (c). (a) and (d) are equivalent with the unreceived TOL states, $\mathbf{x}_{\bar{r}}$ marginalized out.

Given the information available to the client,

1. $p^c(\mathbf{x}_o, \mathbf{x}_r|\mathbf{Z}^m)$, the client-side server pose-graph up to the last received TOL and
2. $p^s(\mathbf{x}_o, \mathbf{x}_n|\mathbf{Z}^n)$, the new OSP (computed server-side),

Construct the client goal distribution, i.e., the server pose-graph up to time n, as if unreceived TOL states had been marginalized out,

$$p^c(\mathbf{x}_o, \mathbf{x}_r, \mathbf{x}_n|\mathbf{Z}^n) = \mathcal{N}^{-1}\big(\boldsymbol{x}^c_n; \boldsymbol{\eta}^c_n, \Lambda^c_n\big),$$

$$\Lambda^c_n = \begin{bmatrix} \Lambda_{o,o} & \Lambda_{o,r'} & 0 & 0 \\ \Lambda_{r',o} & \Lambda_{r',r'} & \Lambda_{r',m} & 0 \\ 0 & \Lambda_{m,r'} & \boxed{\Lambda_{m,m}} & \boxed{\Lambda_{m,n}} \\ 0 & 0 & \boxed{\Lambda_{n,m}} & \boxed{\Lambda_{n,n}} \end{bmatrix}, \tag{10}$$

$$\boldsymbol{\eta}^c_n = \begin{bmatrix} \boldsymbol{\eta}_o \\ \boldsymbol{\eta}_{r'} \\ \boxed{\boldsymbol{\eta}_m} \\ \boxed{\boldsymbol{\eta}_n} \end{bmatrix}. \tag{11}$$

The boxed elements in (10) and (11) indicate unknown values in the desired goal reconstruction while the remaining values are known from (8) and (9) because of the assumed Markov structure for the server states (block tri-diagonal information matrix). In other words, all new information since the *last received* OSP only affects a small portion of the information matrix and vector. The client-side reconstruction is illustrated in Fig. 6.

The client-side reconstruction begins by marginalizing out $\mathbf{x}_r$ from the (partially unknown) goal distribution, (10) and (11), producing an expression that can be equated to the (known) received n^{th} OSP, (6) and (7),

$$p^c_{\mathrm{OSP}_n}(\mathbf{x}_o, \mathbf{x}_n|\mathbf{Z}^n) = \int_{\mathbf{x}_r} p^c(\mathbf{x}_o, \mathbf{x}_r, \mathbf{x}_n|\mathbf{Z}^n) d\mathbf{x}_r \tag{12}$$

$$\equiv p^s_{\mathrm{OSP}_n}(\mathbf{x}_o, \mathbf{x}_n|\mathbf{Z}^n). \tag{13}$$

Marginalization of the goal distribution via the Schur complement leads to a set of linear equations in the unknowns. We proceed with a two-step marginalization procedure. First, we marginalize out r' states from (10) and (11) via the Schur complement:

$$\begin{bmatrix} \tilde{\Lambda}_{o,o} & \tilde{\Lambda}_{o,m} & 0 \\ \tilde{\Lambda}_{m,o} & \tilde{\Lambda}_{m,m} & \Lambda_{m,n} \\ 0 & \Lambda_{n,m} & \Lambda_{n,n} \end{bmatrix} = \begin{bmatrix} \Lambda_{o,o} & 0 & 0 \\ 0 & \Lambda_{m,m} & \Lambda_{n,m} \\ 0 & \Lambda_{n,m} & \Lambda_{n,n} \end{bmatrix} - \begin{bmatrix} \Lambda_{o,r'} \\ \Lambda_{m,r'} \\ 0 \end{bmatrix} \Lambda^{-1}_{r',r'} \begin{bmatrix} \Lambda_{r',o} & \Lambda_{r',m} & 0 \end{bmatrix}, \tag{14}$$

$$\begin{bmatrix} \tilde{\boldsymbol{\eta}}_o \\ \tilde{\boldsymbol{\eta}}_m \\ \boldsymbol{\eta}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{\eta}_o \\ \boldsymbol{\eta}_m \\ \boldsymbol{\eta}_n \end{bmatrix} - \begin{bmatrix} \Lambda_{o,r'} \\ \Lambda_{m,r'} \\ 0 \end{bmatrix} \Lambda^{-1}_{r',r'} \boldsymbol{\eta}_{r'}, \tag{15}$$

where the tilde indicates block elements that are changed from (10) and (11). This step results in the following set of expressions,

$$\begin{aligned} \tilde{\Lambda}_{o,o} &= \Lambda_{o,o} - \Lambda_{o,r'}\Lambda^{-1}_{r',r'}\Lambda_{r',o}, \\ \tilde{\Lambda}_{o,m} &= -\Lambda_{o,r'}\Lambda^{-1}_{r',r'}\Lambda_{r',m}, \\ \tilde{\Lambda}_{m,m} &= \Lambda_{m,m} - \Lambda_{m,r'}\Lambda^{-1}_{r',r'}\Lambda_{r',m}, \\ \tilde{\boldsymbol{\eta}}_o &= \boldsymbol{\eta}_o - \Lambda_{o,r'}\Lambda^{-1}_{r',r'}\boldsymbol{\eta}_{r'}, \\ \tilde{\boldsymbol{\eta}}_m &= \boldsymbol{\eta}_m - \Lambda_{m,r'}\Lambda^{-1}_{r',r'}\boldsymbol{\eta}_{r'}. \end{aligned} \tag{16}$$

Second, we marginalize out state m from (14) and (15) and equate to the server-side OSP, (6) and (7),

$$\begin{bmatrix} \Lambda^s_{o,o} & \Lambda^s_{o,n} \\ \Lambda^s_{n,o} & \Lambda^s_{n,n} \end{bmatrix} = \begin{bmatrix} \tilde{\Lambda}_{o,o} & 0 \\ 0 & \Lambda_{n,n} \end{bmatrix} - \begin{bmatrix} \tilde{\Lambda}_{o,m} \\ \Lambda_{n,m} \end{bmatrix} \tilde{\Lambda}^{-1}_{m,m} \begin{bmatrix} \tilde{\Lambda}_{m,o} & \Lambda_{m,n} \end{bmatrix}, \tag{17}$$

$$\begin{bmatrix} \boldsymbol{\eta}^s_o \\ \boldsymbol{\eta}^s_n \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{\eta}}_o \\ \boldsymbol{\eta}_n \end{bmatrix} - \begin{bmatrix} \tilde{\Lambda}_{o,m} \\ \Lambda_{n,m} \end{bmatrix} \tilde{\Lambda}^{-1}_{m,m} \tilde{\boldsymbol{\eta}}_m. \tag{18}$$

The unknown values of the goal client-side server reconstruction, boxed in (10) and (11), are then computed by substituting (16) into the above equality,

$$\begin{aligned} \Lambda_{m,m} &= \tilde{\Lambda}_{m,m} + \Lambda_{m,r'}\Lambda^{-1}_{r',r'}\Lambda_{r',m}, \\ \Lambda_{m,n} &= -\tilde{\Lambda}_{m,m}\tilde{\Lambda}^{-1}_{o,m}\Lambda^s_{o,n}, \\ \Lambda_{n,n} &= \Lambda^s_{n,n} + \Lambda_{n,m}\tilde{\Lambda}^{-1}_{m,m}\Lambda_{m,n}, \\ \boldsymbol{\eta}_m &= \tilde{\boldsymbol{\eta}}_m + \Lambda_{m,r'}\Lambda^{-1}_{r',r'}\boldsymbol{\eta}_{r'}, \\ \boldsymbol{\eta}_n &= \boldsymbol{\eta}^s_n + \Lambda_{n,m}\Lambda^{-1}_{m,m}\tilde{\boldsymbol{\eta}}_m, \end{aligned} \tag{19}$$

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 `havePrimaryOriginIndex`(o_n) **then**
      $\Lambda_n, \boldsymbol{\eta}_n \leftarrow$ `addOriginPacket`$(\Lambda^s_n, \eta^s_n, o_n)$
      `updateDEIF`$(\Lambda_n, \boldsymbol{\eta}_n)$ {(22), (23)}
    else if `haveSecondaryOriginIndex`(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`$(\Lambda_n, \boldsymbol{\eta}_n)$ {(22), (23)}
    else
      `requestRecovery()`
    end if
  end if
end loop

where

$$\begin{aligned} \tilde{\Lambda}_{m,m} &= \tilde{\Lambda}_{m,o}\left(\tilde{\Lambda}_{o,o} - \Lambda^s_{o,o}\right)^{-1}\tilde{\Lambda}_{o,m}\,, \\ \tilde{\boldsymbol{\eta}}_m &= \tilde{\Lambda}_{m,m}\tilde{\Lambda}^{-1}_{o,m}\left(\tilde{\boldsymbol{\eta}}_o - \boldsymbol{\eta}^s_o\right)\,. \end{aligned} \tag{20}$$

When only one TOL state has been received (i.e., $m = 1$, $r = \{r_1\}$, and $r' = \{\emptyset\}$), the derivation proceeds as above, but with only the second marginalization step.

As an implementation aside, at the TOA of the first received OSP, the client does not need to perform any computation to reconstruct the server pose-graph. The initial OSP is simply the two-node server pose-graph consisting of the server origin state and the current TOL state. (Note that this allows any new clients to immediately enter and join the network, i.e., the network can dynamically resize).

3.2.3 *Origin Shifting*

The information difference $\left(\tilde{\Lambda}_{o,o} - \Lambda^s_{o,o}\right)$ in (20) represents the delta information known about the origin state between the client through the last received OSP, $p^c(\mathbf{x}_o|\mathbf{x}_r, \mathbf{Z}^m)$, and the server at the current TOL, $p^s(\mathbf{x}_o|\mathbf{x}_n, \mathbf{Z}^n)$. This difference approaches machine precision as the time between the origin and new TOL state grows (because little additional smoothing of the origin state occurs 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 Walls and Eustice (2012); however, an acknowledgment based scheme does not scale well to many clients. Moreover, numerical instability will continue to plague an acknowledgement driven system if the server does not regularly receive acknowledgments. Instead, we propose a shifting scheme in which the server evaluates a function based upon the numerical stability of the newest OSP—keeping the OSP broadcast *passive* and not requiring any client acknowledgements, such that the method can more easily scale to many client vehicles.

During our real-time experiments (Section 5), the server shifted the origin forward using a threshold, T, on the trace of the difference term in (20),

$$\mathrm{trace}\left(\tilde{\Lambda}_{o,o} - \Lambda^s_{o,o}\right) < \mathrm{T}. \tag{21}$$

The trace is only used as a measure to test the numerical stability of the OSP and is tuned to produce an accurate reconstruction. Note that $\tilde{\Lambda}_{o,o}$ is the $\Lambda^s_{o,o}$ element from the previous OSP, and is therefore readily available without additional computation.

When the shifting function suggests shifting the origin, the new origin is set to the previous 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, in practice each server broadcast encodes at least two OSPs: the primary OSP encoding the transition from the origin to the current TOL, and a secondary OSP encoding the transition from the previous origin to the current origin. Depending upon the available bandwidth, the server could extend the broadcast to include more than two OSPs to increase robustness. From an implementation standpoint, the server does not need to recompute previous OSPs, it simply stores previously broadcast messages. The server-side origin shifting step and corresponding client behavior are outlined in Algorithm 1 and Algorithm 2, respectively.

After the server shifts the origin forward, the client-side reconstruction will contain server TOL states preceding the new origin. The reconstruction rules (19) are unmodified if the client first marginalizes out these earlier server states.

3.2.4 *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 (i.e., reconstruct the current server pose-graph given the state it has already received). Once the client identifies that it has lagged behind, it transmits a recovery request to the server containing the last received TOL index. After receiving a client request, the server computes this special information as an additive 'delta information' (discussed in Section 4) 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 in order to compute a recovery packet, unless it can guarantee that each client has received a more recent TOL. The full client-side operation is summarized in Algorithm 2.

Recovery requests could limit the scalability of the algorithm because each client vehicle requires a slot in the TDMA schedule to transmit. Since recoveries are so rarely necessary, however, only a tiny fraction of the TDMA need be reserved.

4 Online Distributed Estimation

We demonstrate that the incremental reconstruction of the server pose-graph within the OSM framework can be used onboard the client to exactly reproduce the centralized solution to the multiple vehicle localization problem. We couple the OSM algorithm with the DEIF algorithm update (Webster et al., 2013) to compute the client-side state estimate following OWTT range observations. The DEIF algorithm is a method in which a client vehicle can exactly reproduce the centralized delayed-state filter solution 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 DEIF, as originally proposed, is not real-time practical since it relies on an unrealistic communication assumption (perfect packet reception) to build the server information—this is remedied by the OSM representation. The full operation of the OSM and DEIF is illustrated in Fig. 1.

To review the DEIF, 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

$$\begin{aligned} \Delta\Lambda_{s_n} &= \Lambda_{s_n} - \Lambda_{s_{n-1}}, \\ \Delta\boldsymbol{\eta}_{s_n} &= \boldsymbol{\eta}_{s_n} - \boldsymbol{\eta}_{s_{n-1}}, \end{aligned} \tag{22}$$

where the operation conforms for the dimensionality difference and the 's' subscript indicates the server's information. The delta information is illustrated in Fig. 4. 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-lossy) 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 information into its information filter

$$\begin{aligned} \Lambda_{c_n} &= \Lambda'_{c_n} + \Delta\Lambda_{s_n}, \\ \boldsymbol{\eta}_{c_n} &= \boldsymbol{\eta}'_{c_n} + \Delta\boldsymbol{\eta}_{s_n}, \end{aligned} \tag{23}$$

where the 'c' subscript indicates the client-side information including both server TOL states and the current client state. Following the subsequent relative range measurement update, the client-side filter matches the corresponding centralized filter exactly up to communication delay. The client is not required to maintain the full set of server TOL poses in its state vector. Full details of the algorithm as well as extensive comparitve results are provided in Webster et al. (2013).

Fig. 7: Ocean-Server, Inc. Iver2 AUVs used in the field experiments.

5 Field Trials

Seven field trials spanning more than $12\ \mathrm{h}$ of operation with a single server and two client vehicles were performed (Fig. 11). These trials demonstrate the OSM algorithm's ability to incrementally broadcast and reconstruct the server pose-graph, compute the centralized solution, and fuse range-only constraints in a multiple vehicle framework. In addition to the single server to client topology, we provide post-process results demonstrating a two-server support network.

5.1 Experimental Setup

We fielded two Ocean-Server, Inc. Iver2 AUVs (Brown et al., 2009), designated AUV1 and AUV2, (Fig. 7). Each AUV is outfitted with an advanced dead-reckoning sensor suite including a $600\ \mathrm{kHz}$ 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, which is considered to be the client. AUV1 is the only vehicle that observes and fuses GPS when at the surface. To demonstrate the ability of our OSM algorithm to support multiple client vehicles, we also consider a topside support ship (with only GPS

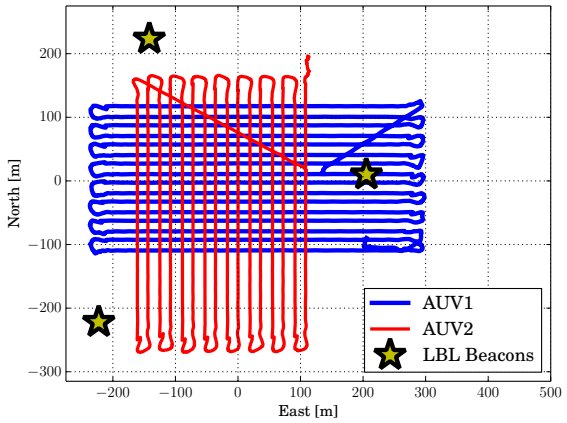


Fig. 8: Experiment B setup with three LBL beacon locations.

reported velocity and heading for input) as a client vehicle. All vehicles were outfitted with a Woods Hole Oceanographic Institution (WHOI) Micro-modem and co-processor board capable of encoding multiple frame, higher bandwidth phase-shift keying (PSK) data packets (Freitag et al., 2005a,b).

For baseline comparision of our OWTT navigation, we deployed a three-beacon 25 kHz LBL network to independently measure underwater vehicle position. Fig. 8 depicts the LBL beacon locations used during our field trials, which were positioned to provide good triangulation observability over the client vehicle survey area (the beacon locations were moored and held fixed for all experiments). Each vehicle interrogated the LBL network roughly twice per minute, resulting in a maximum of six range constraints per minute. We recorded two-way LBL, along with GPS position fixes at the surface, for all vehicles for ground-truth comparison (none of the client vehicles used these measurements during the real-time experiments).

5.2 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 broadcasting pressure depth with each acoustic packet, OWTT 3D range measurements, $z_{3\mathrm{D}}$, can be projected into the horizontal plane,

$$z_r = \sqrt{z_{3\mathrm{D}}^2 - (d_s - d_c)^2},$$

where d_s and d_c are the depth estimates for the server and client, respectively. Moreover, we are motivated to maintain a minimal state size because of the limited acoustic 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 is time-propagated using an odometry-driven plant model,

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

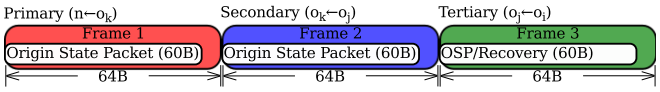


Fig. 9: Acoustic message composition. Each PSK Rate 1 and Rate 2 Micro-modem message contains three 64-byte frames. We use the first two frames to hold two origin state packets and the third frame is reserved for recovery packets, or an additional origin state packet if no recovery has been requested. (Frame colors correspond to OSPs shown in Fig. 5.)

where $\mathbf{u}_{k+1}$ is the delta odometry measurement. 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 Eustice et al. (2011). In the case of the topside surface craft, 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:

$$z_r = \left\| \mathbf{x}_{s_{\mathrm{TOL}}} - \mathbf{x}_{c_{\mathrm{TOA}}} \right\|_2 + v,$$

where $v \sim \mathcal{N}(0, \sigma_r^2)$ represents the range measurement noise. In our experiments, we used $\sigma_r = 30$ cm, to account for noise in timing and depth.

5.3 Acoustic Communication Considerations

Each origin state packet requires 60 bytes to encode. Each double precision element of the origin state information is rounded to a precision of 10^{-5} to reduce the packet size. Moreover, 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 broadcast three 64 byte frames (Fig. 9). We fill the first two frames of Rate 1 and Rate 2 messages with the primary and secondary OSPs as discussed in Section 3.2.3. 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, otherwise we broadcast a tertiary OSP. Acoustic packets were encoded using dynamic compact control language (DCCL) (Schneider and Schmidt, 2010) and transmitted using the Goby-acomms library (Schneider and Schmidt, 2012).

We employed a fixed TDMA cycle, whereby all vehicles were assigned a communication time slot. The server vehicle broadcast an OSP roughly once per minute, while the client transmitted a single data packet over the same time window, used to monitor vehicle health state and to place recovery requests when needed. As noted in Fig. 11h, the average server throughput used for navigation data was ∼25 bps. The

average client-side navigation packet reception rates across experiments varied between 32% and 63%. Our origin state method allowed each client to reliably reconstruct the server pose-graph despite the small bandwidth allotment and low reception rates.

5.4 Results

Fig. 11 summarizes the relative vehicle trajectories over the seven individual field trials. The relative server-client geometries between AUV1 and AUV2 were purposely varied between the different experiments while the topside surface craft had no control and drifted around the survey site, occasionally motoring to stay within the site boundary. During Experiments A and B, the server and client swam on orthogonal lawn-mower trajectories, Experiments C and D, the server-client swam along the same lawn-mower trajectory with the server following at a fixed distance, Experiment E, the server encircled the client via a bounding diamond-shaped trajectory, while during Experiments F and G, the server-client swam along the same lawn-mower trajectory, beginning at different boundaries of the survey area. As seen in Fig. 11h, the different relative geometries and conditions led to varied communication reception performance ranging from 32–63% throughput. During the course of our experiments, each vehicle swam at fixed depth with AUV1 holding a depth of 8 m, AUV2 at 10 m (apart from prescribed surface intervals for GPS ground-truth), and the topside transducer was suspended at 10 m depth.

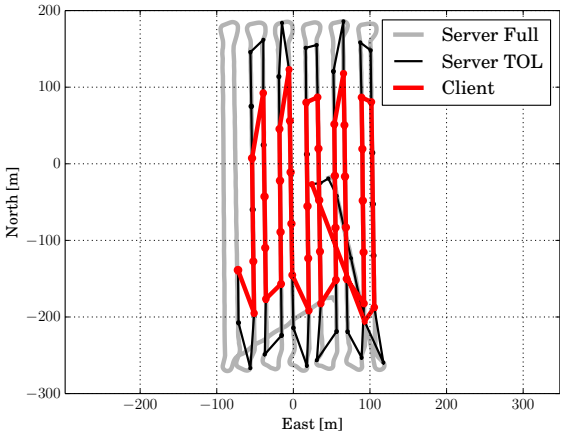


Fig. 10: Pose-graph reconstruction example from Experiment G. Light gray trajectory depicts the server pose-graph over all poses, while the black pose-graph represents server TOL states. The thick red line represents the client-side reconstructed server pose-graph (as if missed server TOL poses had been marginalized out). The two pose-graphs are equal up to communication round-off errors.

5.4.1 *Client-side Server Pose-graph Reconstruction*

The client was able to accurately reconstruct the server pose-graph using the OSM framework throughout all of our field trials. Fig. 10 illustrates an example 'true' server pose-graph with the client-side reconstruction overlaid. Note that the client-side reconstruction does not contain all of the server TOL poses. The client's version of the server pose-graph, however, is equivalent to the server's as if the TOL states corresponding to dropped messages had been marginalized out. As shown in Fig. 13a, each client is able to reconstruct the server pose-graph with small error, on average to the order of 10^{-5} m. Moreover, this error is attributed to the communication round-off of the OSP (using the origin-shifting scheme with full precision OSPs in post-process, both mean and max reconstruction error is on the order of 10^{-12} m).

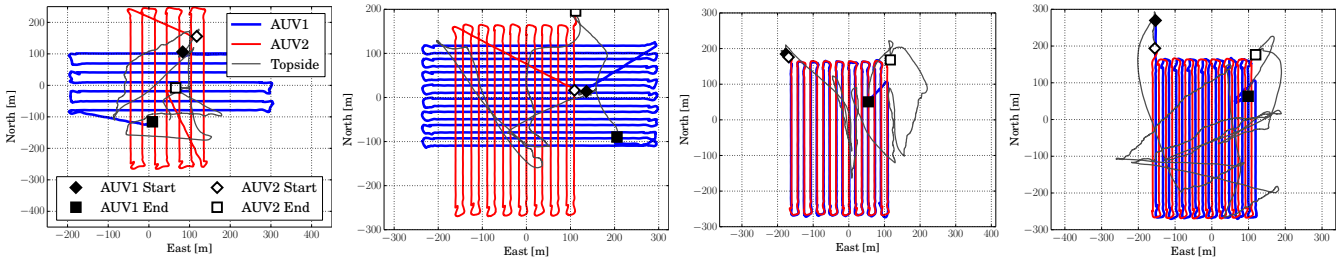
5.4.2 *Server Origin Shifting*

During Experiment A, the server shifted the origin forward at an artificially increased rate (every other TOL) in order to force a client recovery request. During this trial, AUV2 and Topside received one and three recovery packets, respectively, after transmitting requests to the server. These packets were successfully integrated into the server pose-graph reconstruction.

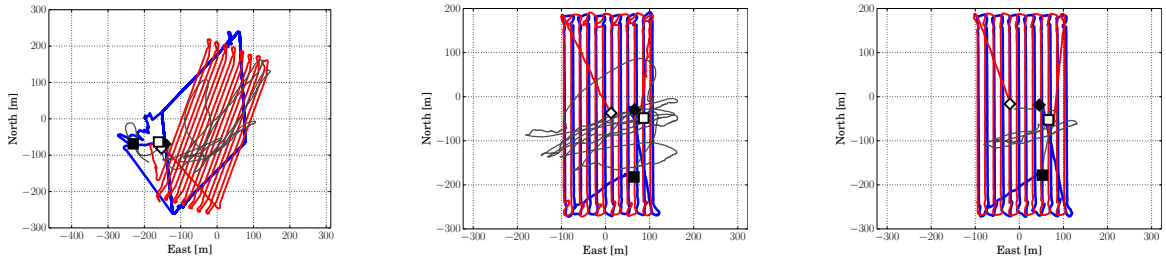
Throughout the remaining 11 h of field trials showcasing normal operation, neither client required a recovery packet. Moreover, during normal operation (Experiments B–G), both clients used a total of 554 Frame 1 OSPs, 62 Frame 2 OSPs, and only 2 Frame 3 OSPs. The server used a trace threshold $T = 5 \cdot 10^{-2}$.

The client will require a secondary (Frame 2) OSP at most once per server origin shift, because the secondary packet 'catches' the client up to the current origin. We expect the client to occasionally require a secondary packet following an origin shift because of the high likelihood of missing any single transmission. However, the client would need to lose communication with the server over an entire period between origin shifts in order to require the tertiary (Frame 3) OSP. In order to require a recovery packet, communication would have to drop out over at least two origin shift periods, depending on the number of broadcast redundant OSPs.

The rate at which the server will shift the origin (based on the trace metric in (21)) depends on the server noise models and available measurements. Absolute position observations, e.g., GPS, and noisy process models will reduce correlation between the current state and the origin state, so that subsequent measurements will not influence the origin as much. Therefore, after receiving absolute position observations or sufficient time given a noisy process model, the server will be forced to shift the origin. Fig. 12 illustrates the server's ori-



(a) Experiment A (0.94 h) (b) Experiment B (2.02 h) (c) Experiment C (1.87 h) (d) Experiment D (1.92 h)



(e) Experiment E (2.01 h) (f) Experiment F (1.63 h) (g) Experiment G (1.65 h)

	A	B	C	D	E	F	G
Server: AUV1 broadcast bps	25.6	25.1	22.4	23.0	23.0	22.8	23.2
Client: AUV2 Reception rate	47.9%	43.2%	32.9%	32.1%	53.0%	34.4%	33.7%
Client: Topside Reception rate	46.1%	55.9%	62.6%	57.1%	60.2%	50.4%	39.9%

(h) Acoustic statistics across Experiments A–G

Fig. 11: Summary of the seven field trials used for experimental evaluation. (a)–(g) Topdown view of the 3-node vehicle trajectories with total trial time indicated in each subcaption. The legend provided in (a) applies to all figures. AUV2 and Topside acted as clients while AUV1 performed the server role. (a) AUV1 shifted the origin forward at an accelerated rate in order to artificially induce recovery requests. (b) AUV1 and AUV2 followed orthogonal lawnmower surveys. (c),(d) AUV1 followed AUV2 along the same lawnmower survey. (e) AUV1 maintained a diamond box path bounding AUV2's lawnmower survey. (f),(g) AUV1 and AUV2 followed similar lawnmower surveys, beginning from opposite ends of the survey area, i.e., AUV1 began on the East boundary while AUV2 began on the West boundary.

gin shifting during Experiment D. The server surfaced at regular intervals, receiving several GPS observations. In post-process, we cut out GPS measurements over a nearly 30 min window. In this case, the server shifted the origin forward less frequently (twice as opposed to four times as seen in Fig. 12b). This demonstrates the ability of the OSM algorithm to automatically adapt origin shifting to varying measurement availability.

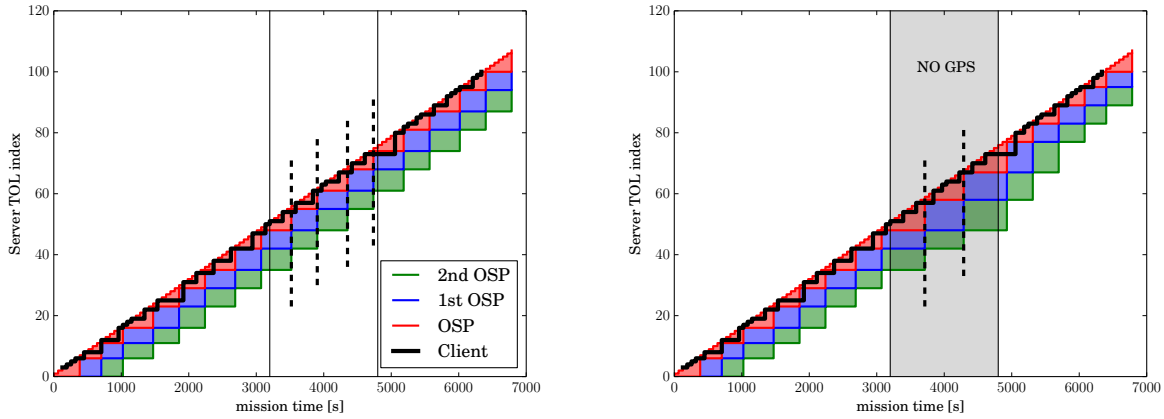
5.4.3 *Client-side DEIF*

Each client employed a DEIF to integrate server and client information with relative range observations. The state estimate of the post-process CEIF agrees with our real-time DEIF with differences commensurate with those reported by Webster et al. (2013) (on the order of 10^{-5} m) and on par with the errors observed in the server pose-graph reconstruction (see Fig. 13b). The two estimates are equal at the TOA of each OSP. The estimates may vary in between TOAs because the CEIF is able to incorporate server information the instant it is received, while the DEIF must wait to incorporate server information until the OSP is received (i.e., up to communication delay).

5.4.4 *Multiple Server Implementation*

It is a relatively simple extension to move from a single to multiple independent server implementation as is depicted in Fig. 3. In this case, the client vehicle reconstructs the local pose-graph from each server and fuses relative range constraints. Range constraints are still measured by the client to each server. The client-side DEIF reproduces the centralized filter (up to communication delay). Communication delay will cause the linearization point used for range observations to differ between the client-based and centralized results. This is because one (or both) of the servers will have made observations that the client has not yet received.

We tested the two server scenario in post-process using the field collected sensor data from Experiment C. In this trial, the Topside vehicle acted as the client with AUV1 and AUV2 acting as servers. The Topside-based DEIF solution differs with the corresponding CEIF on average by 4.5 cm. We can attribute this difference primarily to a difference in range observation linearization



(a) GPS available during surface intervals

(b) No GPS during shaded region

Fig. 12: Server OSP TOL indices and the corresponding index received by the client during Experiment D. The shaded red/green/blue regions represent the OSPs broadcast by the server (color coded according to Fig. 9), while the thick black line plots the client's latest received server index. Large steps in the server indices indicate an origin shift. (a) illustrates the real-time server origin shifting, while (b) demonstrates how the OSM algorithm adapts to varying conditions, waiting longer before shifting the origin when GPS is cut out. Note that the real-time server shifted the origin four times during the same region whereas the post-process origin only shifted twice (indicated by dashed lines).

point.

5.4.5 *Performance Baseline*

OWTT navigation allows AUVs to navigate subsea for extended periods of time with bounded error. Bounded error navigation is usually achieved with the use of networks of stationary acoustic beacons, e.g., LBL. The cost of deploying an LBL system can be prohibitively high, however. Here, we discuss the ability of OWTT relative ranging frameworks to approach the accuracy of LBL systems without the operational and equipment overhead. The accuracy of the post-process CEIF has previously been extensively compared to an LBL navigation solution by Webster et al. (2012).

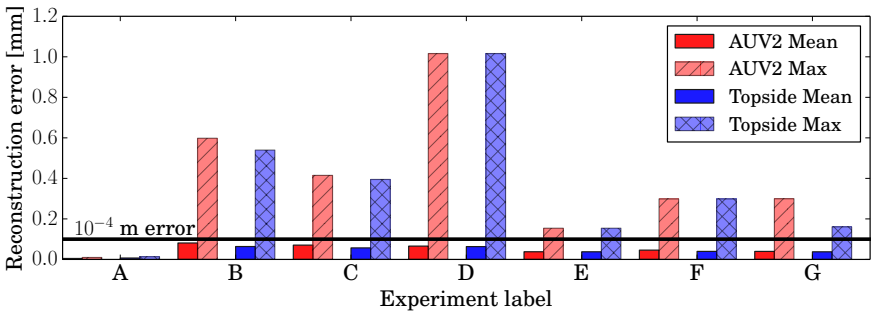
Fig. 14 compares both LBL and OWTT based DEIF navigation solutions to a baseline trajectory computed by fusing GPS and odometry measurements. Since each range observation only adds information in a single direction, the relative geometry between server and client is of utmost importance. Experiment B (Fig. 14a) intuitively has an informative relative geometry as the server continually crosses over the clients path, helping to bound error in both x, y directions. Experiment C (Fig. 14b), however, does not have such a useful relative geometry, as the server largely remains behind the client along the North–South direction. Indeed, we see that the server is able to well bound uncertainty along the East–West direction during Experiment B, but not as well in Experiment C. Moreover, due to an informative relative server trajectory, the client DEIF closely reproduces the full LBL solution in Experiment B.

Each client vehicle received at most one OWTT relative range constraint over each one minute TDMA window, while potentially receiving six LBL ranges over the same period. Using OWTT navigation, however, we can achieve bounded error on the same order as LBL. This solution is only enhanced by selecting appropriate server relative geometries. Advantageously, our method also scales to many client vehicles, whereas the update rate for LBL decreases with new clients. Optimal relative positioning of server vehicles has received recent attention (Tan et al., 2014; Bahr et al., 2012), and is the subject of future and ongoing work.

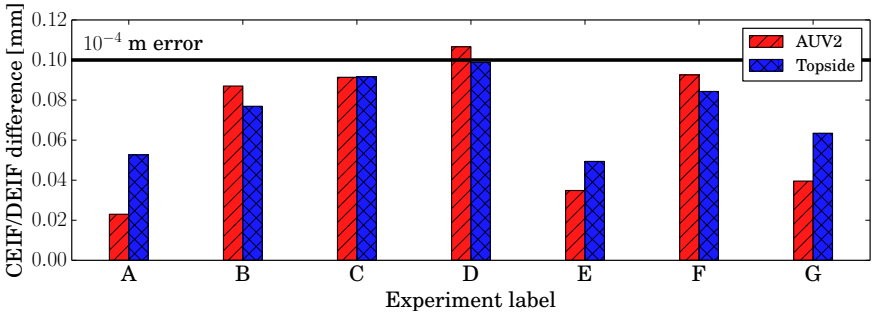
6 Other Estimation Frameworks

The goal of this section is to show that the OSM framework is estimator agnostic and can be applied to other estimation methods. We show *(i)* the OSM framework used within the iSAM algorithm by Kaess et al. (2008), and *(ii)* that the delta information computed from the OSM constitutes a linear factor over server TOL poses. The OSM provides the communication layer for a distributed localization framework. In the previous section, we showed results for the OSM used within a DEIF for online estimation. Here, we apply the OSM within the iSAM algorithm, which constructs a nonlinear optimization problem from a set of constraints or factors.

The OSM algorithm reproduces the linear server information up to communication round-off errors onboard the client. The delta information (22) computed from the client-side reconstructed pose-graph is a linear factor over the previously received and current server TOL states. This factor contains all information due to observations occurring between the TOLs including process predictions, odometry, and absolute position measurements such as GPS. Fig. 15, for example, illustrates a delta factor as summarizing all information that

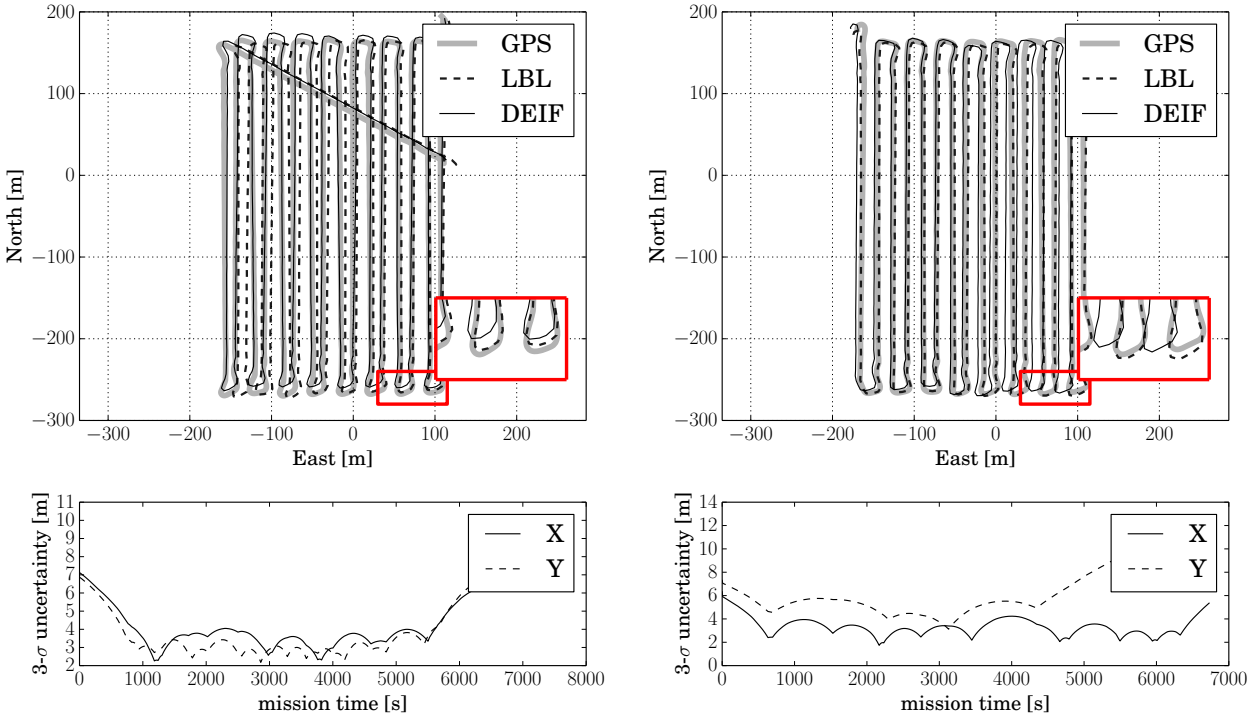


(a) Client-side server pose-graph OSM reconstruction error



(b) Client-side decentralized estimate (DEIF) versus centralized benchmark (CEIF)

Fig. 13: Summarized reconstruction and estimation error across all seven field trials (A–G). (a) The server pose-graph reconstruction error is computed as the norm of the difference between server TOL poses received on the client and their actual value computed on the server. The maximum error for any single pose remains below 1 mm for all trials, while the mean error is on the order 10^{-5} m. (b) Difference in the CEIF versus the DEIF client trajectory estimate at the TOA. Each bar represents the mean norm difference between state estimates for each estimator. Note that the client-side DEIF is able to reproduce the centralized CEIF estimate in real-time to high accuracy.



(a) Experiment B: good server-client geometry

(b) Experiment C: poor server-client geometry

Fig. 14: OWTT and LBL navigation comparison to a GPS baseline. OWTT compares well with LBL in (a), while OWTT has a larger error in (b) (outlined in red in the lower right corner). The server/client relative geometries (Fig. 11), however, greatly influence this result. The lower axis in each subfigure plots the x, y uncertainty estimated by the DEIF, showing that in (b) y, East, uncertainty is not as well bounded.

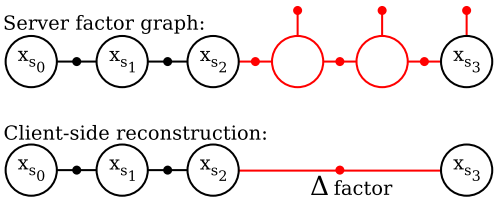


Fig. 15: Server factor graph example. The above factor graph shows the full server factor graph. Red nodes and edges indicate constraints occurring between the previous and current TOL. The lower graph illustrates the reconstructed client-side factor graph. The red delta state factor induces an equivalent factor potential as the above red factors.

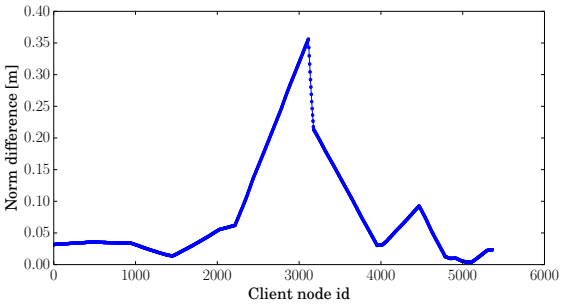


Fig. 16: Difference in estimated pose between the DEIF algorithm and the nonlinear least-squares iSAM during Experiment F. The mean and maximum differences are 0.079 m and 0.35 m, respectively.

occurs between server TOL poses.

We can compute a factor to represent the delta information using the concept of generic linear constraints proposed by Carlevaris-Bianco and Eustice (2013). First, we compute the eigen-decomposition of the delta information

$$\Delta\Lambda = \mathrm{U}\mathrm{D}\mathrm{U}^\top,$$

where $\Delta\Lambda \in \mathbb{R}^{p\times p}$, $q = \mathrm{rank}(\Delta\Lambda)$, $\mathrm{U} \in \mathbb{R}^{p\times q}$, and $\mathrm{D} \in \mathbb{R}^{q\times q}$ is a diagonal matrix. Using the decomposition, we define the delta factor as a Gaussian observation with mean and covariance

$$\mathbf{z}_\Delta = \mathrm{D}^{\frac{1}{2}}\mathrm{U}^\top \boldsymbol{x} \tag{24}$$

$$\Sigma_\Delta = \mathrm{I}_{q\times q}, \tag{25}$$

where $\boldsymbol{x} = [\mathbf{x}_{s_{n-1}}^\top, \mathbf{x}_{s_n}^\top]^\top$ is the state vector over the previous and current server TOL poses. This factorization (while not unique) produces the same information over the server TOL poses as the additive delta information block.

iSAM is equivalent to the DEIF with the exception that measurement constraints may be relinearized after their initialization. This difference is small in our implementation as all odometry constraints are linear, and infrequent range measurements account for the only nonlinearities.

We implemented the factor representation for delta information, (24) and (25), to provide server information to the client. Fig. 16 plots the difference between the DEIF and iSAM for Experiment F. The small difference in DEIF and iSAM estimates is due to differing linearization points. The ability to relinearize range constraints may have additional benefits over the DEIF when a good linearization point is initially not available, for example, when the client dead-reckoned error has grown too large.

7 Conclusion

We presented the first-ever practical real-time algorithm for exact server-client cooperative localization over an extremely faulty and bandwidth-limited communication channel. We validated this distributed estimation algorithm over more than 12 h of field trials deploying three vehicles. We demonstrated how our algorithm is adaptable to support both multiple clients and multiple servers, albeit under a strict server to client communication topology.

Funding

This work was supported the National Science Foundation [award numbers IIS-0746455, ANT-1039951].

References

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

Bahr, A., Leonard, J. J., and Martinoli, A. (2012). Dynamic positioning of beacon vehicles for cooperative underwater navigation. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, pages 3760–3767, Vilamoura.

Bahr, A., Walter, M. R., and Leonard, J. J. (2009b). Consistent cooperative localization. In *Proc. IEEE Int. Conf. Robot. and Automation*, pages 3415–3422, Kobe.

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

Brown, H. C., Kim, A., and Eustice, R. M. (2009). An overview of autonomous underwater vehicle research and testbed at PeRL. *Marine Tech. Soc. J.*, 43(2):33–47.

Carlevaris-Bianco, N. and Eustice, R. M. (2013). Generic factor-based node marginalization and edge sparsification for pose-graph SLAM. In *Proc. IEEE Int. Conf. Robot. and Automation*, pages 5728–5735, Karlsruhe.

Carrillo-Arce, L., Nerurkar, E. D., Gordillo, J., and Roumeliotis, S. I. (2013). Decentralized multi-robot cooperative localization using covariance intersection. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, pages 1412–1417, Tokyo.

Cunningham, A., Paluri, M., and Dellaert, F. (2010). DDF-SAM: Fully distributed SLAM using constrained factor graphs. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, pages 3025–3030, Taipei.

Cunningham, A., Wurm, K. M., Burgard, W., and Dellaert, F. (2012). Fully distributed scalable smoothing and mapping with robust multi-robot data association. In *Proc. IEEE Int. Conf. Robot. and Automation*, pages 1093 –1100, Saint Paul.

Dellaert, F., Alegre, F., and Martinson, E. B. (2003). Intrinsic localization and mapping with 2 applications: Diffusion mapping and marco polo localization. In *Proc. IEEE Int. Conf. Robot. and Automation*, pages 2344–2349, Taipei.

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

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

Fallon, M. F., Kaess, M., Johannsson, H., and Leonard, J. J. (2011). Efficient AUV navigation fusing acoustic ranging and side-scan sonar. In *Proc. IEEE Int. Conf. Robot. and Automation*, pages 2398–2405, Shanghai.

Fallon, M. F., Papadopoulis, G., and Leonard, J. J. (2010a). A measurement distribution framework for cooperative navigation using multiple AUVs. In *Proc. IEEE Int. Conf. Robot. and Automation*, pages 4256–4263, Anchorage.

Fallon, M. F., Papadopoulos, G., Leonard, J. J., and Patrikalakis, N. M. (2010b). Cooperative AUV navigation using a single maneuvering surface craft. *Int. J. Robot. Res.*, 29(12):1461–1474.

Fox, D., Burgard, W., Kruppa, H., and Thrun, S. (2000). A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, 8(3):325–344.

Freitag, L., Grund, M., Partan, J., Ball, K., Singh, S., and Koski, P. (2005a). Multi-band acoustic modem for the communications and navigation aid AUV. In *Proc. IEEE/MTS OCEANS Conf. Exhib.*, pages 1080–1085, Washington.

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

Howard, A., Matari, M. J., and Sukhatme, G. S. (2002). Localization for mobile robot teams using maximum likelihood estimation. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, pages 434–439, Lausanne.

Julier, S. and Uhlmann, J. (1997). A non-divergent estimation algorithm in the presence of unknown correlations. In *Proc. Amer. Control Conf.*, volume 4, pages 2369–2373, Albuquerque.

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

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

Leung, K., Barfoot, T., and Liu, H. (2010). Decentralized localization of sparsely-communicating robot networks: A centralized-equivalent approach. *IEEE Trans. Robot.*, 26(1):62–77.

Li, H. and Nashashibi, F. (2013). Cooperative multi-vehicle localization using split covariance intersection filter. *IEEE Intell. Transp. Syst. Mag.*, 5(2):33–44.

Maczka, D. K., Gadre, A. S., and Stilwell, D. J. (2007). Implementation of a cooperative navigation algorithm on a platoon of autonomous underwater vehicles. In *Proc. IEEE/MTS OCEANS Conf. Exhib.*, pages 1–6, Vancouver.

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

Milne, P. H. (1983). *Underwater acoustic positioning systems*. Gulf Publishing Company, Houston.

Murphy, C. A. (2012). *Progressively Communicating Rich Telemetry from Autonomous Underwater Vehicles via Relays*. PhD thesis, Massachusetts Inst. Tech. / Woods Hole Ocean. Inst. Joint Program, Department of Electrical Engineering and Computer Science.

Nerurkar, E. D., Zhou, K. X., and Roumeliotis, S. I. (2011). A hybrid estimation framework for cooperative localization under communication constraints. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, pages 502–509, San Francisco.

Partan, J., Kurose, J., and Levine, B. N. (2007). A survey of practical issues in underwater networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(4):23–33.

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

Roumeliotis, S. I. and Bekey, G. A. (2002). Distributed multi-robot localization. *IEEE Trans. Robot. Autom.*, 18(5):781–795.

Schneider, T. and Schmidt, H. (2010). The dynamic compact control language: A compact marshalling scheme for acoustic communications. In *Proc. IEEE/MTS OCEANS Conf. Exhib.*, pages 1–10, Sydney.

Schneider, T. and Schmidt, H. (2012). Goby-acomms version 2: Extensible marshalling, queueing, and link-layer interfacing for acoustic telemetry. In *IFAC Conf. Manoeuvring and Cont. Marine Craft*, pages 331–335, Arenzano.

Symmetricomm (2013). SA.45s CSAC—Chip Scale Atomic Clock. Specification sheet and documentations available at http://www.symmetricom.com/products/frequency-references/chip-scale-atomic-clock-csac/SA.45s-CSAC/.

Tan, Y. T., Gao, R., and Chitre, M. (2014). Cooperative path planning for range-only localization using a single moving beacon. *IEEE J. Ocean. Eng.* In Print.

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

Walls, J. M. and Eustice, R. M. (2011). Experimental compar-

ison of synchronous-clock cooperative acoustic navigation algorithms. In *Proc. IEEE/MTS OCEANS Conf. Exhib.*, pages 1–7, Kona.

Walls, J. M. and Eustice, R. M. (2012). An origin state method for lossy synchronous-clock acoustic navigation. In *IFAC Workshop Nav., Guidance, Cont. Underwater Vehicles*, pages 56–62, Porto.

Walls, J. M. and Eustice, R. M. (2013). An exact decentralized cooperative navigation algorithm for acoustically networked underwater vehicles with robustness to faulty communication: Theory and experiment. In *Proc. Robot.: Sci. & Syst. Conf.*, Berlin.

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

Webster, S. E., Walls, J. M., Eustice, R. M., and Whitcomb, L. L. (2013). Decentralized extended information filter for single-beacon cooperative acoustic navigation: Theory and experiments. *IEEE Trans. Robot.*, 29(4):957–974.

Webster, S. E., Whitcomb, L. L., and Eustice, R. M. (2010). Preliminary results in decentralized estimation for single-beacon acoustic underwater navigation. In *Proc. Robot.: Sci. & Syst. Conf.*, Zaragoza.

Whitcomb, L. L., Yoerger, D. R., and Singh, H. (1999). Combined doppler/LBL based navigation of underwater vehicles. In *Proc. Int. Symp. Unmanned Untethered Subm. Tech.*, pages 1–7, Durham.