2022 American Control Conference (ACC)
Atlanta, USA, June 8-10, 2022

Distributed Event-Triggered Localization for High Latency Communication*

Luke Barbier[1], Luke Morrissey[1], Nisar Ahmed[1], Eric Frew [1], Sonia Martinez[2], Kenneth Center[3]

***Abstract*— This paper presents a Kalman Filter-based solution to the problem of decentralized cooperative robot localization and tracking with high latency and low bandwidth communication using an event-triggered fusion algorithm. Event-triggering is a compression technique for measurement sharing whereby only 'surprising' measurements are exchanged between agents. For unsurprising measurements, the lack of a measurement transmission is itself information about the value of a measurement that can be fused by other agents. This relies on consistency between 'common' estimates of agents, but, in high latency communication, consistency is not guaranteed and is difficult to achieve with bandwidth restrictions. We present a novel 'Delta-Tier' algorithm to address these issues, using token-passing to preserve consistency and auto-selection of event trigger thresholds δ to fit the available bandwidth. We demonstrate the success of our approach in achieving decentralized performance that is on par with idealized centralized fusion for an autonomous underwater robotics application in a simulated nonlinear environment, where robots must localize each other and track other non-cooperative agents.**

I. Introduction

Autonomous underwater vehicle (AUV) teams are suited for many tasks too dangerous or mundane for humans to complete, including underwater reconnaissance and ocean floor mapping [1]–[3]. Within these teams, it is critical to know locations of other team members so as to not collide and to cooperate on shared mission tasks. Cooperative localization is the task of estimating shared vehicle states through inter-team communication. The underwater environment presents a unique challenge for cooperative localization in that communication is multicast, low bandwidth and scheduled, which leads to delays between transmissions. This is due to the use of acoustic communication methods, because typical radio-frequency methods do not work well underwater. In addition, for the same reasons, GPS is unavailable, which necessitates new filtering methods that can fully use information from available sensors. These sensors typically include a doppler velocity loggers (DVL), inertial-measurement unit (IMU), compass, barometer, sonar and an ultra-short baseline acoustic positioning system (USBL). USBLs often come built into the acoustic modem (AM) communication device, allowing the AM to operate in either measurement mode or communication mode.

Event-triggered (ET) Kalman filtering is an approach to cooperative localization that utilizes intelligent measurement sharing [4]–[6]. The core idea is that data is only exchanged between communicating agents if some mutually known event threshold is satisfied at the transmitter. The selective exclusion of data in a communicated packet can be interpreted as compressed information that supports data fusion, with relatively small losses versus optimal performance yet significantly lower data payloads. Since decentralized approaches to ET cooperative localization were previously developed for asynchronous, low latency communication scenarios, it is challenging to adapt these approaches to the underwater environment, where scheduled communication strategies for single-channel data transmission necessarily entail high latency information sharing.

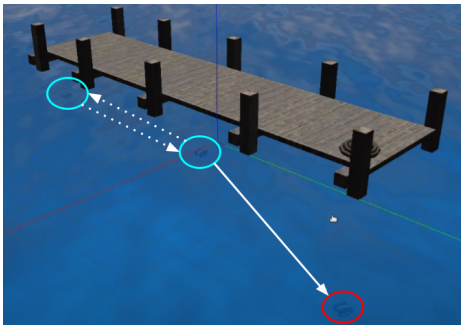


Fig. 1: Simulated AUVs (blue) patrolling a harbor zone and exchanging tracking and localization information: one AUV collects sonar measurements of an intruder (red) along with measurements of itself and the other blue AUV, and exchanges information with the other blue AUV.

To address this gap, we present Delta-Tier, a new algorithm that leverages the bandwidth savings of decentralized ET Kalman filtering by auto-selecting the δ thresholds to adapt to the available payload space for cooperative tracking and localization, while adhering to the scheduled high latency and ultra-low bandwidth constraints of the subsea environment. To operate in such ultra-low bandwidth environments, we also discuss how quantization can be applied in unison with Delta-Tier for effective measurement compression and sharing. In summary, the main contributions of this work are: (i) a method for auto-selection of δ-bands for ET; (ii) adaptation of ET, a traditionally asynchronous method, to scheduled, single-channel communication; and (iii) evaluation of the proposed method in a simulation environment with nonlinear motion and measurement models.

In the remainder of the paper, we present the problem statement and background for decentralized event-triggered estimation (Sec. II); discusses the Delta-Tier algorithm and

*This work was supported by ONR STTR Contract #N17A-029-0114.

[1] Smead Aerospace Engineering Sciences, University of Colorado Boulder, Boulder, CO, USA: nisar.ahmed@colorado.edu

[2] Dept. of Mechanical and Aerospace Engineering, University of California San Diego, La Jolla, CA USA

[3] OrbitLogic, Inc., Greenbelt, MD USA

quantization along with their application through acoustic modems to underwater cooperative localization (Sec. III); present simulations results to evaluate Delta-Tier (Sec. IV) and conclusions and future work (Sec. V).

II. Background and Related Work

A. Problem Statement

We adopt the same notation as in [6]. Let a quantity, S, pertaining to an agent, i, at time k be S_k^i, and let j's ($j \neq i$) estimate of that same quantity be $S_k^{j,i}$. We assume there are N agents in an underwater semi-cooperative localization scenario, l of which are team members and the remainder of which are non-cooperative intruders. Agents participating in cooperative localization will be denoted *network agents* while other agents will be called *intruder agents*. Network agents attempt to estimate the state of all other network agents and intruders. Agent i's inertial position and velocity dynamics are modeled by

$$x_{k+1}^i = f(x_k^i, u_k^i) + w_k^i, \tag{1}$$

where x_k^i is agent i's n-dimensional, ownship state vector, control inputs are u_k^i and $w_k^i \sim \mathcal{N}(0, Q_k^i)$ is white process noise with $Q_k^i \in S_+^n$. Measurements are modeled as

$$y_{k+1}^i = h(x_{k+1}^i) + v_{k+1}^i, \tag{2}$$

where $h(\cdot)$ is the measurement function, $v_{k+1} \sim \mathcal{N}(0, R_k^i)$ is white measurement noise with $R_k^i \in S_+$, and y_k^i is a measurement taken by i at time k with dimension n_q.

In addition to serving as a communication device, AMs provide range, azimuth and elevation measurements to another beacon. Throughout this work, we assume the communication throughput and sensor characteristics of the BluePrint Subsea SeaTrac system[1]: 32 byte packet with 4s transmission (and ring-down) time. One AM is located at the surface in a static, known location (e.g. dock, boat maintaining position) and serves as the source of absolute positioning. Along with receiving absolute positioning from the surface, network agents can take relative position measurements of each other within a bounded range.

Latency, $\mathcal{L}_m$, in the communication network is defined as

$$\mathcal{L}_m = t_r - t_c,$$

where t_c is the time when network agent i collects measurement m, and t_r is the time when information contained in measurement m is received (and fused) by another network agent j ($j \neq i$). This latency arises from the half-duplicity of acoustic underwater communication. Communication schedules must be agreed upon to determine when agents can share information with the network and avoid packet collisions.

A communication schedule is defined that determines the ordering of AM measurement collections and transmissions for each agent, so as to avoid packet collision. All transmissions are multicast and therefore whose contents are available to all network agents. Because the communication scheme is multicast and has high latency, we assume that agents have sufficient CPU power (and computational time) to estimate the joint state of all agents in the environment. That is, the state vector to be estimated by network agents i and j is $s^i = s^j = [x^1, \ldots, x^l, x^{l+1}, \ldots, x^N]^T$ with l network agents and $N - l$ intruders. This communication and measurement topology is fixed throughout time. The measurements i has collected onboard up until time k are denoted $Y_{1:k}^i$. The measurements agent i has received from other agents are denoted ${}^cY_{1:k}^i$. Let X_k denote the state of all agents in the environment, and i's minimum mean-squared error (MMSE) estimate of all agents in the environment be $\hat{X}_k^i$. The corresponding MSE/covariance matrix of the estimate is denoted as P_k^i. These quantities are given by

$$\hat{X}_k^i = \mathbb{E}[X_k^i | Y_{1:k}^i, {}^cY_{1:k}^i], \tag{3}$$

$$P_k^i = \mathbb{E}[(X_k - \hat{X}_k^i)(X_k - \hat{X}_k^i)^T | Y_{1:k}^i, {}^cY_{1:k}^i]. \tag{4}$$

Each agent produces estimates locally and online. Our problem is to design a cooperative localization algorithm that can utilize information shared between agents to improve $\hat{X}^i$ while adhering to the constraints of a scheduled, low bandwidth and high latency environment. An illustration of this scenario is shown for the underwater environment in Fig. 1 with two network agents and one intruder agent. We denote an estimate as $\hat{\mathcal{X}}_k^{(\cdot)} = (\hat{X}_k^{(\cdot)}, P_k^{(\cdot)})$.

B. Related Work

Various approaches have been proposed for aiding autonomous underwater robotic teams in cooperatively localizing. These approaches broadly fall under two categories: sharing estimates and sharing measurements.

When estimates are shared, they can be combined through Covariance Intersection (CI) [7], or in the case of heterogeneous states among agents, partial State CI [8]. This approach is known as Decentralized Data Fusion (DDF). CI has the important property that estimates can be combined in the presence of unknown cross correlations. This comes with the trade-off that CI can result in overly conservative fused state estimates. Split-CI is a modified CI approach that allows for some cross correlations between estimates to be utilized [9]–[11]. In certain applications the computational cost of tracking other agents in order to perform CI for the exchange of information can be too large. Carillo-Arce, et al. [12] propose an approach where agents only estimate their ownship states and opportunistically create 'pseudo-state estimates' using their ownship estimate projected through a measurements of other agents. This pseudo-state estimate can then be fused through CI. This computationally cheaper approach, however, can be also overly conservative and lose much of the information carried in the cross correlation of measurements. In addition, this approach is not readily applicable to the underwater domain due to the communication latency defined above. Once a measurement is collected, information must be stored until the communication window opens for this agent, necessitating some memory of the other agent which the approach does not natively support.

Funk, et al. [13] propose Quantized Covariance Intersection (QCI) which is able to represent the mean and

[1] https://blueprintsubsea.com/seatrac/products.php

covariance in a small packet size through quantization for the purpose of fusion. The diagonals of the covariance matrix are inflated to account for the uncertainty introduced through quantization of the mean vector. In addition, this approach is able to represent off-diagonal information in the covariance by again inflating the diagonals to create a diagonal dominant positive semi-definite covariance matrix that provides a conservative estimate for fusion. This approach is theoretically applicable to our bandwidth and latency constraints, but again can suffer from excessive conservative information loss. Nevertheless, QCI is used as a comparison point in the evaluation of our proposed algorithm in Sec. IV.

To combat loss of information through solely CI-based methods, measurement sharing approaches have been proposed that maintain the cross-correlations between states. Paull et al. [14] propose a cooperative-SLAM approach where agents share measurements of landmarks and other agents and solve for trajectories in a graph SLAM framework. The approach, however, does not perform well in environments with infrequent landmark encounters to anchor position uncertainty.

ET methods have also been proposed that can greatly improve upon the conservativeness of solely DDF approaches. Trimpe and D'Andrea [15] utilized such a method in a decentralized control problem where agents broadcasted measurements if their local estimate deviated past a threshold from the common estimate. Shi et al. introduced a fusion algorithm that quantified the omission of measurement sharing as a set-valued update approximated by a truncated Gaussian [16]. Ouimet et al. [5] propose a hybrid ET-DDF approach, where agents share measurements in an ET framework but 'resync' estimates between agents when the uncertainty between the local and common estimate grows beyond a threshold through DDF. Loefgren et al. [6] adapts this approach for the heterogeneous states case to increase scalability. While all of the above approaches are able to balance localization accuracy with communication cost and maintain cross correlations between states, they are not natively applicable to the subsea environment due to their asynchronous, low latency communication assumptions.

C. Decentralized Event-Triggered Estimation

Ouimet et al. [5], [17] proposed a hybrid approach to cooperative localization whereby: (i) measurements between agents were exchanged via ET using a *common information filter*, and (ii) every so often DDF was used to completely synchronize local estimates with the common information shared between pairs of network agents. The later step was used to correct for 'drift' among neighboring agents because a non-uniform communication and measurement topology was assumed. This would lead to absolute positioning information not being shared to some agents, and resync steps are thus needed to share that information. Since in our problem scenario all communication is multicast, and all agents have access to absolute positioning, the DDF estimate synchronization step is not necessary.

In the proposed decentralized ET estimation scheme, each network agent i and j still maintains a local estimate at time k, $\hat{\mathcal{X}}_k^i$ and $\hat{\mathcal{X}}_k^j$, respectively. Each member of a pair of communicating network agents also still maintains a *common estimate* estimate that is conditioned only on measurements exchanged with the other member of the pair, respectively denoted for i and j as ${}^c\hat{\mathcal{X}}_k^{i,j}$ and ${}^c\hat{\mathcal{X}}_k^{j,i}$, where we now require ${}^c\hat{\mathcal{X}}_k^{i,j} = {}^c\hat{\mathcal{X}}_k^{j,i}\ \forall k$. Hence, whereas step (i) of the approach developed in [5], [17] still applies, the removal of step (ii) from this process requires the development of a different strategy maintain the correctness of decentralized ET fusion.

Assuming for now that such a strategy is in place, then agent i can decide whether to share measurement y_k^i with agent j by triggering on the innovation (residual) of the measurement. That is, for a set event trigger threshold δ, if $|y_k^i - h_y({}^c\hat{X}_k^{i,j})| > \delta$, then the measurement is *surprising* and transmitted to the other agent. If this is not the case, the measurement is omitted from transmission, and agent j is able to infer that $y_k^i \in [h_y({}^c\hat{X}_k^{i,j}) - \delta, h_y({}^c\hat{X}_k^{i,j}) + \delta]$.

A measurement that falls within this bound can be fused in the KF framework as a set-valued measurement approximated as a truncated Gaussian [16]. With mean, μ, and variance, σ^2, the Gaussian variable y_k has a truncated Gaussian representation, $\bar{y}_k$, with $\mathbb{E}[\bar{y}_k] = \mu + \bar{z}$ and $\mathbb{E}[(\bar{y}_k - \mathbb{E}[\bar{y}_k])^2] = (1-\vartheta)\sigma^2$ where $\bar{z}$ is the bias introduced through truncation and $\vartheta \in [0,1]$ is a scaling term of the measurement variance. Agent j can approximately fuse $\bar{y}_k$ within a modified KF measurement update by first defining the following quantities, where $\hat{X}_k^j$ is agent j's current estimate (post-prediction step) and $\bar{X}_k^j$ is agent j's estimate before the prediction step was executed in the KF update,

$$\mu = h(\hat{X}_k^j) - h(\bar{X}_k^j), \tag{5}$$

$$q_e = H_k \bar{P}_k^j H_k^T + r_k, \tag{6}$$

$$\alpha = h({}^c\hat{X}_k^{j,i}) - h(\bar{X}_k^j), \tag{7}$$

$$\nu^- = \frac{-\delta + \alpha - \mu}{\sqrt{q_e}}, \tag{8}$$

$$\nu^+ = \frac{\delta + \alpha - \mu}{\sqrt{q_e}}, \tag{9}$$

where H_k is the Jacobian row vector of $h(\hat{X}_k^j)$, r_k is the diagonal element of R_k^j corresponding to measurement y_k, and α is the innovation of the common estimate compared to agent j's estimate prior to the prediction step.

Let $\phi(x)$ be the standard Gaussian pdf, and $Q(x) = 1 - \Phi(x)$ where $\Phi(x)$ is the standard Gaussian cdf. Agent j can then fuse measurement $\bar{y}_k^i$ through the following modified Kalman update

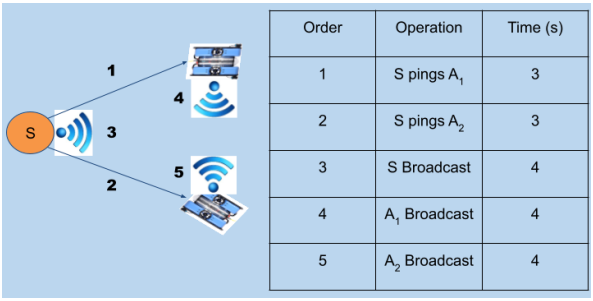
$$\bar{z} = \frac{\phi(\nu^-) - \phi(\nu^+)}{Q(\nu^-) - Q(\nu^+)}\sqrt{q_e}, \tag{10}$$

$$\vartheta = \left(\frac{\phi(\nu^-) - \phi(\nu^+)}{Q(\nu^-) - Q(\nu^+)}\right)^2 - \frac{\nu^-\phi(\nu^-) - \nu^+\phi(\nu^+)}{Q(\nu^-) - Q(\nu^+)}, \tag{11}$$

$$K_k = P_k^j H_k^T (H_k P_k^j H_k^T + R_k)^{-1}, \tag{12}$$

$$\hat{X}_{k+1}^j = \hat{X}_k^j + K_k \bar{z}, \tag{13}$$

$$P_{k+1}^j = P_k^j - \vartheta P_k^j H_k^T (H_k P_k^j H_k^T + R_k)^{-1} C_k P_k^j. \tag{14}$$



Order	Operation	Time (s)
1	S pings A_1	3
2	S pings A_2	3
3	S Broadcast	4
4	A_1 Broadcast	4
5	A_2 Broadcast	4

Fig. 2: Example modem schedule and time table for 3 beacons: one modem at the surface and one modem on each agent A_1 and A_2. *ping* indicates a USBL measurement which provides range, azimuth and elevation values from the sending beacon to the collected beacon.

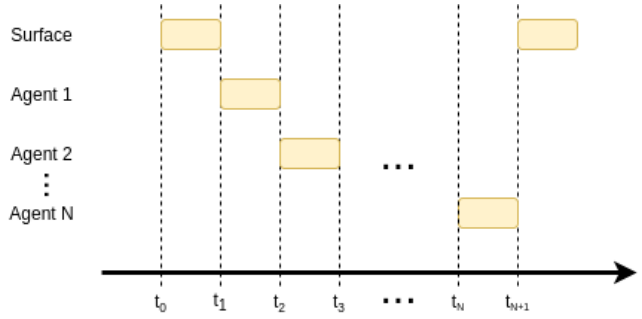


Fig. 3: Common token ownership cycle. Only one agent has the token to modify the common estimate in order to ensure consistency across the network. The token is passed to the next agent in the schedule after a broadcast.

The next section describes how only one common filter is needed for all agents due to multi-cast communication.

D. Acoustic Modem Properties

The decentralized ET approaches in [5], [6], [17] made two assumptions that do not hold up well in the underwater environment with typical AM systems. The first is the notion of asynchronous communication. Measurements cannot be shared solely based on events because of the chance for packet collision. Only one AM can transmit at a single time or packets may become corrupted. This naturally introduces a defined schedule of transmission. An example schedule is shown in Fig. 2 where a surface beacon first measures both agents, then broadcasts those measurements. Then agent A_1 and A_2 both broadcast their locally collected measurements.

The second assumption that does not hold is the notion of an incomplete graph communication network. That is, some network agents only communicate (and estimate) with a proper subset of other network agents. Yet, realistic AMs are inherently based on multicast tranmission. Unicast mode is supported at the driver layer, but not the physical layer. If in range, an AM will receive a packet intended for another AM and drop it after inspecting its intended recipient. Given this fact and the high latency for scenarios where all agents are within the operating range of the AM (typically a few kilometers), it is inefficient to utilize the AM in unicast mode. If information is available that is beneficial for fusion purposes, it is logical to use that information whenever possible. This is why broadcast communication (and thus a complete-graph network) where every agent estimates and communicates with all other network agents suits the underwater environment, and is considered here.

III. Delta-Tier Algorithm

We now present the Delta-Tier (DT) algorithm by first explaining a necessary condition for ET in scheduled communication systems: common estimate consistency. We then present the algorithm in terms of the receiving and transmission steps. Lastly, we explain how to apply low-level quantization for message compression, which is necessary to maximize the number of measurements that can be shared.

A. Common Estimate Consistency

We would like to share measurements through ET-based compression but adhere to the scheduled communication constraints. Instead of immediately sharing a measurement implicitly or explicitly, agent i will buffer it. When the communication window opens for agent i, it will transmit this buffer, and the receiving agent j will retroactively fuse the buffered (implicit and explicit) measurements.

It is important to maintain the following condition in order to properly fuse the shared implicit measurement in the ET framework:

$$ {}^{c}\hat{X}_k^{i,j} = {}^{c}\hat{X}_k^{j,i}, {}^{c}P_k^{i,j} = {}^{c}P_k^{j,i} \quad \forall i,j \in \{1,...,n\}. \tag{15}$$

Since we are using broadcast communication and $s^i = s^j \ \forall i,j \in \{1,...,n\}$, we only need to track one common filter for all agents and can simplify the above equation to

$$ {}^{c}\hat{X}_k^{i} = {}^{c}\hat{X}_k^{j}, {}^{c}P_k^{i} = {}^{c}P_k^{j} \quad \forall i,j \in \{1,...,n\}. \tag{16}$$

Consistency can be achieved in the distributed network by defining a *common token* needed to modify the common estimate, ${}^{c}\hat{\mathcal{X}}_k^{(\cdot)} = ({}^{c}\hat{X}_k^{(\cdot)}, {}^{c}P_k^{(\cdot)})$. When agent i broadcasts its measurements at time t_l, it passes the common token to the next agent in the schedule, agent j. Visualized in Fig. 3, if agent j broadcasts its measurements at time t_m, it owns the token from time t_l to t_m and can modify ${}^{c}\hat{\mathcal{X}}_k^{j}$ with all measurements it has collected during this time frame, $Y_{l:m}^j$.

Agent i then receives the buffer of measurements containing $Y_{l:m}^j = \{I_{l:m}^j, E_{l:m}^j\}$ at t_m. It loads the stored common estimate ${}^{c}\hat{\mathcal{X}}_l^{i}$, its main estimate $\hat{\mathcal{X}}_l^i$, and a ledger of measurements it has stored from t_l to t_m, $Y_{l:m}^i$. $I_{l:m}^i$ is the set of implicit measurements and $E_{l:m}^i$ is the set of explicit measurements taken by agent i from times t_l to t_m. It first merges $Y_{l:m}^j$ with $Y_{l:m}^i$:

$$ Y_{l:m}^i = Y_{l:m}^i \cup Y_{l:m}^j. \tag{17}$$

It then replays the filter from time t_l to t_m, fusing all of $Y_{l:m}^i$ at the correct times. During the replay ${}^{c}\hat{\mathcal{X}}_l^{i}$ is only updated with $Y_{l:m}^j$ to ensure consistency of this common estimate across all agents. More precisely, (16) will not be true for whichever agent has the token at time k but will be true

during the replay when all network agents are actually fusing the shared measurements.

B. Delta-Tier Algorithm

Let t_l^i and t_m^i be the start and end of agent i's token period respectively. We want to then share measurements $Y_{l:m}^i = \{I_{l:m}^i, E_{l:m}^i\}$ with all other network agents $j \neq i$.

Because we do not send values for implicit measurements, we need to somehow represent them in the buffer that they were collected in. To do so, we create a header that describes the sequence of measurements taken. The header, ${}^qH_{l:m}^i$, contains the number of measurements collected and the average latency between measurements for measurement type q from times l to m for agent i. ${}^qH_{l:m}^i$ can be used to fully reconstruct the sequence of implicit measurements taken of type $q \in \{1, ..., n_l\}$ for n_l measurement types. We then define the set of headers for all measurements types taken by agent i from times l to m below,

$$\mathcal{H}_{l:m}^i = \{{}^qH_{l:m}^i\} \quad \forall q \in \{1, ..., n_l\}\,.$$

As visualized in Fig. 4, the buffer $B_{l:m}^i$, that agent i will transmit is then

$$B_{l:m}^i = \{\mathcal{H}_{l:m}^i, E_{l:m}^i\}.$$

So far, we have assumed a given trigger, δ, used to determine if measurements are filtered implicitly or explicitly. In reality, δ must be selected for the application. The choice of δ is important because of the constrained buffer size. If δ is too small, too many measurements will be filtered explicitly, and the buffer will overflow. If δ is too large, too few explicit measurements will be shared, and we will be under-utilizing the bandwidth available. This motivates the use of a set of n_b buffers, $\mathcal{B} = \{{}^1B, ..., {}^{n_b}B\}$ and a corresponding set of triggers, $\Delta = \{{}^1\delta, ..., {}^{n_b}\delta\}$ and estimates ${}^c\mathcal{X} = \{{}^1\mathcal{X}, ..., {}^{n_b}\mathcal{X}\}$ resulting from fusing measurements implicitly or explicitly for each ${}^j\delta$. Let $\overline{\mathcal{B}_{l:m}^i}$ be the set of non-overflowed buffers for agent i for times l to m. That is, if we have a maximum bandwidth, λ, for transmission,

$$\overline{\mathcal{B}_{l:m}^i} = \{\, {}^jB \in \mathcal{B}_{l:m}^i \mid \text{len}({}^jB) \leq \lambda \,\}, \tag{18}$$

where len(B) is the bandwidth of buffer B, the buffer with the maximum amount of information, ${}^{\max}B_{l:m}^i$ that we would like to transmit at time m is then

$${}^{\max}B_{l:m}^i = \underset{{}^jB \in \overline{\mathcal{B}_{l:m}^i}}{\arg\min} \; {}^j\delta. \tag{19}$$

This is visualized in Fig. 5, where we see that delta-tiers with higher δ triggers do not overflow as quickly as lower δ triggers, thus providing a near optimal (with increasing n_b) choice of δ. The selected trigger ${}^{\max}\delta$ must also be included in the message header $\mathcal{H}_{l:m}^i$, so n_b must be selected to be small enough so that each transmission's choice of δ can be represented in the buffer.

Algorithm 1 summarizes the overall Delta-Tier algorithm. It is presented in terms execution at the end of a token period t_m, but steps 4-15 can be done online from t_l to t_m to spread out computation overhead.

Algorithm 1 Delta-Tier Algorithm

1: **procedure** DELTATIER($\mathcal{B}$, ${}^c\mathcal{X}$, Δ, ${}^c\hat{\mathcal{X}}_l^{(\cdot)}$, $Y_{l:m}$)
2: $B \leftarrow \emptyset \quad \forall\, B \in \mathcal{B}$ ▷ Initialize all buffers to empty
3: $\mathcal{X} \leftarrow {}^c\hat{\mathcal{X}}_l^{(\cdot)} \quad \forall\, \mathcal{X} \in {}^c\mathcal{X}$ ▷ Initialize all delta-tiers to common estimate of network
4: **for** $t \in \{t_l, ..., t_m\}$ **do**
5: **for** ${}^jB \in \mathcal{B}$ **do**
6: **for** $y \in Y_t$ **do**
7: **if** innovation $({}^j\hat{X}, y) > {}^j\delta$ **then**
8: ${}^jB$.append(y)
9: **else**
10: ${}^jB$.modify_header(y)
11: **end if**
12: **end for**
13: Kalman_Update(${}^j\mathcal{X}$, ${}^jB$)
14: **end for**
15: **end for**
16: $\overline{\mathcal{B}_{l:m}^i} = \{\, {}^jB \in \mathcal{B}_{l:m}^i \mid \text{len}({}^jB) \leq \lambda \,\}$
17:
18: ${}^{\max}B_{l:m}^i = \underset{{}^jB \in \overline{\mathcal{B}_{l:m}^i}}{\arg\min} \; {}^j\delta$
19: **return** ${}^{\max}B_{l:m}^i$
20: **end procedure**

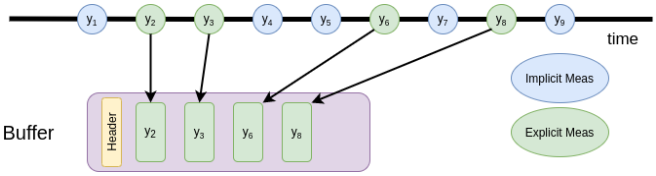


Fig. 4: Explicit measurements are added to the buffer while implicit measurements are omitted. A header message is included that can be used to reconstruct the implicit measurements on the receiver's end.

C. Quantization

Measurements considered here have floating point values, represented by a Float32 or a Float64. In the underwater environment where throughput through an AM is on the order of bytes, these floating point representations quickly fill the bandwidth. This motivates the use of a compression technique that utilizes the known ranges of values of measurements. For a measurement range $[a, b]$ to be compressed using $n < 8$ bytes (Float64), we can define 2^{8n} evenly sized bins that cover the range $[a, b]$. This gives a resolution of $\frac{b-a}{2^{8n}}$.The integer index of this bin is then transmitted using n bytes. This method allows for lossy compression of any continuous value in $[a, b]$ into a codebook value.

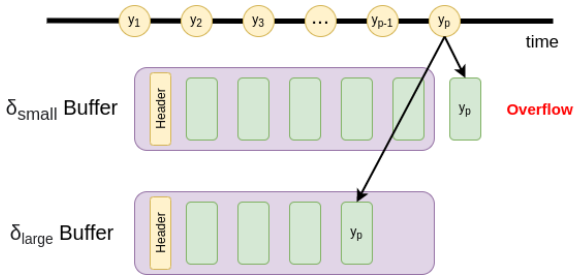


Fig. 5: Two delta-tiers: δ_s and δ_l with $\delta_l > \delta_s$. At time p the δ_s buffer has no room left to include y_p and overflows. Because buffer δ_l has a higher trigger, it has stored (fused) more messages implicitly and has room for explicit measurement y_p.

IV. Simulation Results

Evaluation of the proposed algorithm was done through 500 Monte Carlo (MC) simulations in MATLAB. The DT algorithm was compared against a centralized, ideal solution called the Omniscient Filter and Quantized Covariance Intersection (QCI) [13], adapted for this application. The simulations were conducted in a 2D 20m x 20m environment with three agents: 2 blue team agents and 1 red team agent. The blue team agents cooperated in tracking all position and velocities states of each agent in the network. The red agent was non-cooperative. Scenarios were run for 2000 time steps.

All agents dynamics were nonlinear and given by a Dubin's unicycle model. Forward velocity and angle rate were controlled to guide the blue agents to uniform random waypoints in the environment. The red agent remained a fixed distance from the first blue agent so as to remain in sensor scanning range.

In an effort to increase simulation fidelity when compared with real systems and support the applicability of the proposed algorithm to such systems, blue team agents operated a dual-filter fusion architecture. Because hardware systems often feature a flight controller which runs an optimized filter estimating the ownship states of the vehicle, one possible fusion architecture is to share information between the navigation filter running on the flight controller and a lower-rate tracking filter running onboard a separate general purpose computing system, e.g. by means of covariance intersection [11]. In the underwater application, the navigation filter has access to high rate owship measurements from sensors such as IMU, gyro, compass and Doppler-Velocity Logger. The tracking filter has access to lower rate sensor data measuring other agents in the environment such as side scan sonar and AM measurements. Information passing between the filters is typically done by fusing the estimates through covariance intersection of the overlapping states. Utilizing cross-correlations in the estimate, non-overlapping states that were not subject to direct fusion updates are subsequently updated by each filter via conditional Gaussian updating for Kalman filters, as done in the partial-state covariance intersection approach [6], [8].

Utilizing this dual navigation-tracking filter architecture described above, each blue agent ran a navigation filter which estimated 2D position and orientation along with their respective velocities. This filter had access to angle rate measurements representing a gyro, global orientation representing a compass and body frame velocities representing a DVL. The tracking filter onboard each agent was the primary focus of this experiment and was the filter used in the cooperative algorithms being examined. It estimated the inertial position and velocities of all agents. Sensor noise characteristics followed the model given by (2).

The tracking filter had access to range and azimuth measurements from two sources. The first was inertial position fixes from a static beacon with known position, thereby constraining estimate position uncertainty. Range and azimuth measurements from this source were shared with the blue agents at intervals according to the schedule defined in Fig. 2. These position fixes included measurements of each blue agent but did not include the non-cooperative red agent. In addition, each blue agent fused relative range and azimuth measurements to other agents in range modeling a side scan sonar. Measurements would be taken with 80% probability if another agent was within a 20m radius.

The Omniscient Filter had access to all sonar, modem and navigation filter estimates onboard each agent. This filter served as an upper bound on the performance that could be attained from the sensor and dynamics configurations of the simulation environment.

DT event triggered range and azimuth measurements from the sonar. Because the tracking filters were not estimating orientation of the other agent, relative azimuth measurements were anchored to a global orientation. This was done simply by subtracting the orientation estimate of the navigation filter from the azimuth measurement to get an x-axis referenced azimuth measurement. It was assumed that the navigation filter's estimate was sufficiently accurate that the correlations between the anchored azimuth measurement and the navigation filter estimate could be ignored. Hand tuning was done to find a delta-band ratio of range to azimuth measurements that would split the buffer space evenly between azimuth and range measurements. This generally lead to the better performance of the algorithm. The buffer has space for 20 explicit measurements with 1 byte of compression for each measurement. Shared measurement values were quantized according to the approach in Sec. III-C.

QCI quantizes the mean and covariance of an estimate according to input parameters b, the number of bits for a quantized element, and x_{max}, the maximum value that an element could take on. Given a set packet size of 32 bytes (256 bits), the ratio of bits given to an element in the state vector compared to an element in the covariance is a tuning parameter. The simplest choice is to allocate the same number of bits to elements from either quantity. For a 12 state vector estimate (3 agents, 4 states each) sending the mean requires 12 elements and the upper triangle of the covariance $\frac{12^2-12}{2} + 12 = 78$, totaling 90 elements to be transmitted. $\frac{256}{90} = 2.84$ bits/element. Even though this is not feasible for the packet size of the AM considered without additional compression, we set the number of bits to 4 per element for comparison with the other algorithms. x_{max} was set to 11 for the mean and 40 for the off diagonals of the covariance. Regular CI with a full fusion parameter search was used in this implementation instead of the fast-CI fusion parameter approximation used in [13].

A. Results

Fig. 6 demonstrates the behavior of the DT algorithm for a single MC instance. The high frequency volatility of the covariance bound is due to the periodic nature of the inertial range and azimuth fixes. The lower frequency increases in uncertainty occur when the agents are not in sonar range of each other, but error is still constrained through the sharing of measurements and the inertial fixes. Agent 2's estimates

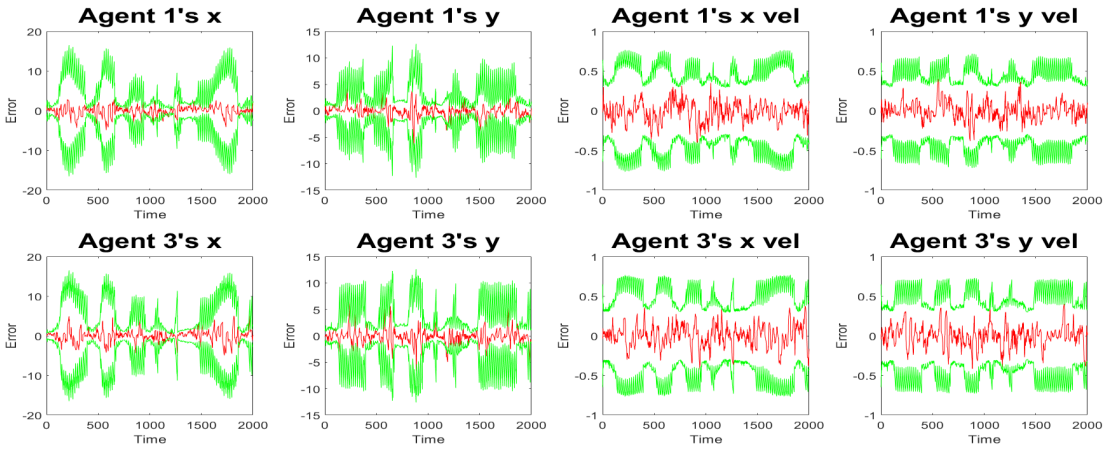


Fig. 6: Agent 2's 2σ tracking filter's error plots of agents 1 and 3's states in a DT MC simulation instance. The red line indicates the error in the estimate of that state while the green lines show the 2σ covariance bound on the error.

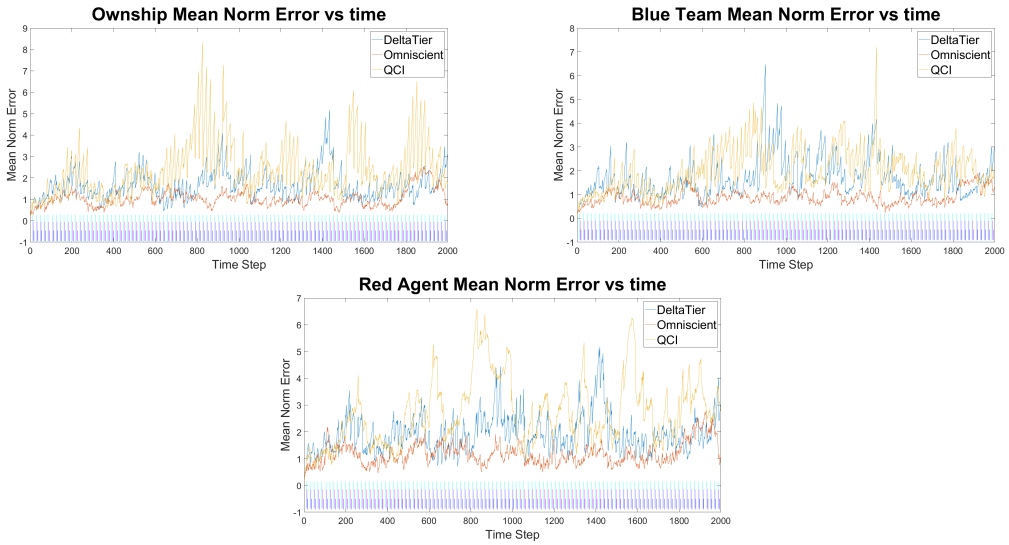


Fig. 7: Average ownship, blue team and red agent estimate norm error for the Omniscient, Delta-Tiering and QCI algorithms for each time step across 500 MC runs. The vertical lines at the bottom of the plots represent the acoustic modem activity. The tall cyan lines are the surface beacon's broadcast while the shorter magenta line and the shortest blue line are agent 1 and agent 2's triggering events respectively.

of agent 1 and 3 are correlated because agent 3 (the red agent) was ensured to always be in sonar range of the first agent so that large errors and uncertainties resulting from no information being collected about the red agent did not dominate the results. We see that even though much of Agent 2's information about agent 3 is shared through agent 1, it is able to use cross-correlations from its estimate of agent 1 to get a nearly as accurate estimate of the red agent.

Fig. 7 shows the results of the Monte Carlo sims for different types of states in the state vector. The ownship states included each blue agent's own position and velocity. The 'blue team' states include each blue agent's estimate of the other blue agent; the 'red agent' results included each blue agent's estimate of the red agent. The third plot demonstrates most of the differences between the algorithms. Both DT and QCI perform worse than the centralized, ideal Omniscient Filter. The figure shows that DT generally has better performance than QCI throughout the simulations in tracking the red agent. Since the red agent does not have its inertial range and bearing reported, all tracking information is collected locally or shared between agents. This results in an increased reliance on the sharing of information between agents for localizing this agent. DT outperforms QCI in this instance because it is more able to take advantage of cross-correlations between states in an estimate through the sharing of measurements. QCI is able to transmit off diagonal information, but its effectiveness is diminished by the large quantization loss of only $2^4 = 16$ quantization bins.

The ownship and blue team mean norm error plots show a similar result. Of course, the Omniscient Filter outperforms the other algorithms because it has complete access to both blue agent's navigation filters which have good ownship

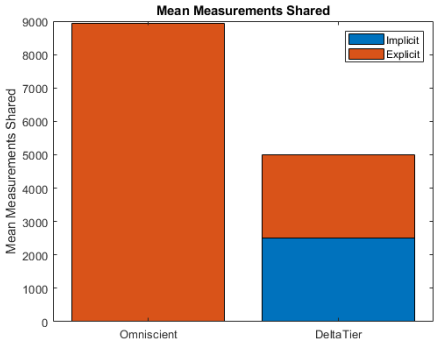


Fig. 8: Average number of measurements shared in each MC run for the two algorithms that share measurements.

tracking. DT and QCI have similar performance because these states are more heavily influenced by the agent's onboard navigation filters and inertial fixes. We do see DT achieve a slightly lower mean norm error in both instances as the measurements shared by each agent are used to update ownship information and tracking of the other blue agent.

The noisy averages seen in the plots in Fig. 7 is a result of the inertial range and azimuth fixes occurring at a regular frequency which aids greatly in reducing the error and constraining uncertainty.

In all of the plots in Fig. 7 we see DT achieving comparable performance to the centralized solution and in Fig. 8 we see it does so while sharing 1/2 of the number of measurements and only 1/4 the measurement values on average. The reason DT does not share all of the measurements either explicitly and implicitly is because of the token passing algorithm described in Sec. III-A.

V. Conclusions and Future Work

This paper presented a cooperative localization algorithm called Delta-Tier which utilizes event-triggered bandwidth savings for high latency in underwater communication. Delta-Tier auto-selects event-triggering thresholds to fit the available packet size. Details of the Delta-Tier algorithm were presented at the packet level along with the byte-level, quantization of measurements. We showed through Monte Carlo simulations with nonlinear dynamics and measurement models for underwater cooperative localization and tracking that DT performs similarly to the ideal centralized algorithm and performs better than another possible state of the art approach (QCI).

One important characteristic of underwater communication that we did not explicitly address in this work is the issue of packet loss. Depending on the mission objectives and how dynamic the agents are, this may not be an issue. In certain dynamic scenarios such as the one simulated in this work, measurements from previous communication periods are unlikely to be of greater tracking value in the limited buffer size than more up-to-date measurements taken recently.

Future work will also investigate ways to transmit all measurements an agent takes, not just the measurements that happen to fall within an agent's token period. The AM sensor naturally leads to a batch processing framework, so perhaps this sharing of a greater number of measurements could be accomplished through a new graph-SLAM based method that is compatible with ET compression. Future work will also evaluate the Delta-Tier algorithm on actual AUV hardware with the sensors described in this work as well as cooperative coverage control algorithms for harbor surveillance.

References

[1] S. B. Williams, O. Pizarro, M. Jakuba, and N. Barrett, "Auv benthic habitat mapping in south eastern tasmania," in *Field and Service Robotics*. Springer, 2010, pp. 275–284.

[2] P. Wadhams and M. Doble, "Digital terrain mapping of the underside of sea ice from a small auv," *Geophysical Research Letters*, vol. 35, no. 1, 2008.

[3] M. J. Oliver, M. W. Breece, D. A. Fox, D. E. Haulsee, J. T. Kohut, J. Manderson, and T. Savoy, "Shrinking the haystack: using an auv in an integrated ocean observatory to map atlantic sturgeon in the coastal ocean," *Fisheries*, vol. 38, no. 5, pp. 210–216, 2013.

[4] W. Song, J. Wang, S. Zhao, and J. Shan, "Event-triggered cooperative unscented kalman filtering and its application in multi-uav systems," *Automatica*, vol. 105, pp. 264–273, 2019.

[5] M. Ouimet, N. Ahmed, and S. Martínez, "Event-based cooperative localization using implicit and explicit measurements," in *2015 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2015, pp. 246–251.

[6] I. Loefgren, N. Ahmed, E. Frew, C. Heckman, and S. Humbert, "Scalable event-triggered data fusion for autonomous cooperative swarm localization," in *2019 22th International Conference on Information Fusion (FUSION)*. IEEE, 2019, pp. 1–8.

[7] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, vol. 4. IEEE, 1997, pp. 2369–2373.

[8] N. R. Ahmed, W. W. Whitacre, S. Moon, and E. W. Frew, "Factorized covariance intersection for scalable partial state decentralized data fusion," in *2016 19th International Conference on Information Fusion (FUSION)*. IEEE, 2016, pp. 1049–1056.

[9] S. Julier and J. Uhlmann, "Generalizations of covariance intersection," Internal Report, Tech. Rep., 1998.

[10] S. J. Julier and J. K. Uhlmann, "Simultaneous localisation and map building using split covariance intersection," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, vol. 3. IEEE, 2001, pp. 1257–1262.

[11] H. Li and F. Nashashibi, "Cooperative multi-vehicle localization using split covariance intersection filter," *IEEE Intelligent transportation systems magazine*, vol. 5, no. 2, pp. 33–44, 2013.

[12] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo, and S. I. Roumeliotis, "Decentralized multi-robot cooperative localization using covariance intersection," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1412–1417.

[13] C. Funk, B. Noack, and U. D. Hanebeck, "Conservative quantization of fast covariance intersection," in *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2020, pp. 68–74.

[14] L. Paull, G. Huang, M. Seto, and J. J. Leonard, "Communication-constrained multi-auv cooperative slam," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 509–516.

[15] S. Trimpe and R. D'Andrea, "An experimental demonstration of a distributed and event-based state estimation algorithm," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 8811–8818, 2011.

[16] D. Shi, T. Chen, and L. Shi, "An event-triggered approach to state estimation with multiple point-and set-valued measurements," *Automatica*, vol. 50, no. 6, pp. 1641–1648, 2014.

[17] M. Ouimet, D. Iglesias, N. Ahmed, and S. Martínez, "Cooperative robot localization using event-triggered estimation," *Journal of Aerospace Information Systems*, vol. 15, no. 7, pp. 427–449, 2018.