# *Consistent cooperative localization*

2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

# Consistent Cooperative Localization

Alexander Bahr Matthew R. Walter John J. Leonard
*Computer Science and Artificial Intelligence Lab, MIT, Cambridge, MA, USA*

***Abstract*— In cooperative navigation, teams of mobile robots obtain range and/or angle measurements to each other and dead-reckoning information to help each other navigate more accurately. One typical approach is moving baseline navigation, in which multiple Autonomous Underwater Vehicles (AUVs) exchange range measurements using acoustic modems to perform mobile trilateration. While the sharing of information between vehicles can be highly beneficial, exchanging measurements and state estimates can also be dangerous because of the risk of measurements being used by a vehicle more than once; such data re-use leads to inconsistent (overconfident) estimates, making data association and outlier rejection more difficult and divergence more likely.**

**In this paper, we present a technique for the consistent cooperative localization of multiple AUVs performing mobile trilateration. Each AUV establishes a bank of filters, performing careful bookkeeping to track the origins of measurements and prevent the use any of the measurements more than once. The multiple estimates are combined in a consistent manner, yielding conservative covariance estimates. The technique is illustrated using simulation results. The new method is compared side-by-side with a naive approach that does not keep track of the origins of measurements, illustrating that the new method keeps conservative covariance bounds whereas state estimates obtained with the naive approach become overconfident and diverge.**

***Index Terms*— Cooperative Localization, Cooperative Navigation, Sensor Fusion**

## I. Motivation

As mobile robots become more affordable and reliable, the use of a groups of many robots becomes more feasible. While the primary reason to use a group of robots is often to speed up tasks such as map building, or add redundancy, more recently the concept cooperative navigation (CN) has been explored. In CN a member A of the group obtains exteroceptive measurements such as range or bearing (or both) to another member B and uses these measurements together with B's own position estimate, obtained through a status broadcast, to improve its own position. Robots often broadcast status information to coordinate with one another or to provide telemetry information to an operator and are equipped with sensors which enable them to locate other members. As a result, adding the CN capability often does not require any extra hardware and represents a cost-effective way to improve the navigation accuracy of the individual vehicles.

With the price and size of Global Positioning System (GPS) units having dropped significantly in the past two decades, it is now possible to outfit each member of large groups of robots with its own unit. Having access to very accurate, absolute position information at rates of several Hz has simplified the robot navigation problem significantly for many areas. There are, however, many environments where GPS signals are either intermittently or not available at all. This mostly affects robots operating indoors or underground, but also those operating underwater such as AUVs. Vehicles operating in these environments may occasionally have access to absolute position information by surfacing for a GPS update in the case of an AUV, or when the camera on an indoor robot recognizes a landmark with a known position [1]. In between these absolute position updates, robots have to rely on dead-reckoning sensors. When relying only on dead-reckoning sensors the uncertainty in their position estimate grows without bound.

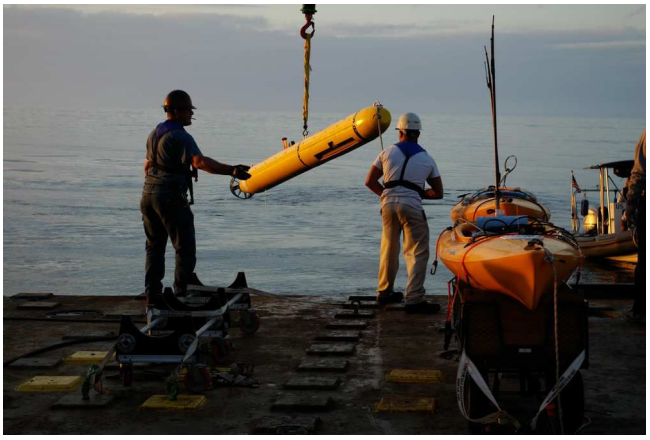

Fig. 1. An Autonomous Underwater Vehicle (AUV) on the crane and two Autonomous Surface Craft (ASC) on deck after a cooperative navigation experiment. The GPS-derived position of the ASCs is obtained by the submerged AUV through an acoustic modem. Using time-stamped messages and globally synchronized clocks on all vehicles the AUV is also able to determine its range to the ASC through time-of-flight measurements.

For scenarios where some members have intermittent access to absolute position information, the uncertainty of their position estimate may be significantly lower than the position uncertainty of those that rely solely on dead-reckoning for a long period of time. By broadcasting their low-uncertainty position estimate, all receiving members of the group, which also obtain a range or bearing measurement to the broadcasting vehicle, are then able to improve their own estimate.

When robot A uses the position estimate of another robot B to update his own, their position estimates become correlated. Not taking these cross-correlations into account can have a negative effect if there is a chain of updates back to B, often leading to an overconfidence in B's position, estimate

which then diverges as a result [2]. Various approaches have been devised to either properly account for the cross-covariance or to use very conservative uncertainty bounds to avoid overconfidence. The resulting failure modes and are outlined in later sections, but many of the algorithms impose additional requirements which make them unfeasible for many CN-scenarios.

Our approach requires that each robot includes additional information within their status broadcast. The receiving robots can then use this information to ensure that the cross-covariances are properly accounted for. The approach does not require centralized data storage and processing as all updates are done locally on each vehicle using only data from the broadcasting vehicle. It does not enforce a particular communication hierarchy or topology and individual members can join and leave the group and do not need any awareness of previous communications or the size of the group. Unlike other methods, broadcasts do not need to be received by all participating vehicles as each transmission contains all the information which is required for a position update which accounts for the cross-correlations.

The main motivation for our approach comes from working with AUVs (figure 1). In the harsh underwater environment, an unreliable navigation estimate may result in the loss of an expensive vehicle. When submerged, AUVs have to rely on dead-reckoning sensors and the number of surfacings for a GPS fix needs to be minimized to save energy or maintain covertness in military applications. Through CN, the surfacing of a single vehicle can then benefit a large number of submerged vehicles. As submerged vehicles can only communicate acoustically over a very slow and unreliable channel [3], the approach can only rely on unacknowledged broadcast-based communication. While the underwater scenario enforces particularly hard constraints, these constraints ensure that our approach works for many other multi-robot applications outside a very controlled lab environment.

## II. Related Work

Roumeliotis *et al.* have contributed a large body of work to the field of CN. Early work relies on a central site for data storage and processing [4]. With this setup, the authors make useful insights into relationship between the number of cooperating robots and the individual position uncertainty. The result is an analytical expression for the upper bound on the positioning uncertainty increase rate for the group [5]. In another experiment, the central filter that keeps track of the state and covariance of all vehicles is replaced by distributed filters that run on the individual members. The exchange of only local data is necessary, but as both vehicles are required to transmit, this approach does not scale as well as others that rely only on one-way broadcasts [6]. Caglioti *et al.* also use a distributed filter approach. While they only require one-way data exchange (broadcast), these broadcasts occur very frequently and their method relies on perfect communication as each vehicle is required to receive every broadcast.

The problem of fusing measurements from several sources while properly keeping track of common information has been addressed by Grime [7] and Nettleton [8]. Unlike the work of Roumeliotis *et al.*, they track the information parametrization of the Gaussian rather than the standard form. In the information form, the update step is simply an addition and joint information, which models co-dependencies among different states, can be subtracted if the communication topology is known.

A general approach to the problem of fusing correlated estimates has been proposed by Julier and Uhlmann [9], [10]. Their Covariance Intersection (CI) algorithm fuses two different estimates for a random variable, each represented by their estimated mean and covariance much like the update step in the Kalman filter. The result is a posterior covariance that guarantees consistency under the assumption of Gaussian noise. Arambel *et al.* present an application of the CI algorithm for a group of space vehicles, where relative position measurements are communicated in a ring topology [2]. Each of these works have examples of how the state estimator can diverge if estimates are fused with a simple Kalman update without accounting for correlation among the estimates. A disadvantage of CI algorithm is that it can only fuse two state estimates. Additionally, unlike the standard Kalman Filter, it cannot perform a partial update such as those that apply to vehicles that only have a range *or* bearing sensor. As a result, robots that only have a bearing sensor, such as a monocular camera, or have only range information from time-of-flight-based techniques cannot participate in a setup which relies on CI for the update of position estimates.

The previous work presented thus far relies on the Kalman Filter in its original and modified (CI) form, or its inverse, the information filter, to compute an estimate. Fox *et al.* [11] use sample-based Markov techniques to perform cooperative localization. They represent the distribution by a large number of Monte Carlo samples rather than as a Gaussian distribution. This representation allows for complex, multi-modal uncertainty distributions and avoids problems with the linearization that is required for Kalman Filter-based methods. Transferring the distribution between vehicles requires a comparably fast communication channel which might not always be available [12]. Their approach also requires information exchange such that broadcast-based approaches cannot be used, which further increases the bandwidth necessary.

The remainder of the paper is structured as follows. In section IV, we outline the assumptions we make for our group of vehicles cooperating for localization. Section III then gives a brief review of Extended Kalman Filter(EKF)-based CN for the special case where only intra-vehicle range measurements are available. Section V describes the Interleaved Update (IU) algorithm that we propose to fuse intra-vehicle range measurements together with status broadcasts from other vehicles to update our own position estimate. Section VI shows an example of the algorithm at work for a group of vehicles over several time steps. Section VII shows

simulated results for a group of cooperating vehicles and shows how the naive approach of not taking proper care of cross-correlations leads to an overconfident (inconsistent) position estimate. Section VIII presents the conclusions and provides an outlook to future work.

## III. Extended Kalman Filter (EKF)-based Cooperative Navigation

EKF-based approaches are the algorithms most commonly used for CN. All EKF-based methods model the estimated random variables as Gaussians, typically parametrized in terms of their mean and covariance, and generally assume that the sensor noise is zero mean, white noise. The shortcomings of these methods, such as the unimodal distribution of the state estimate and the error introduced through the required linearization, have given rise to alternative methods [11],[12],[13]. The EKF outlined below describes a variant that only uses range measurements for its updates. For a more detailed description of the EKF and how to incorporate bearing measurements or range/bearing measurement pairs please refer to [14].

For EKF-based CN, we assume that all $n$ vehicles $V_i, i = [1 \dots n]$ maintain a vector $\boldsymbol{x}_i(k) = [x_i(k), y_i(k), z_i(k)]^T$ that contains an estimate of their position at time $k$, and the covariance matrix

$$\boldsymbol{P}_i(k) = \begin{bmatrix} {\sigma_{xx}}^2(k) & {\sigma_{xy}}^2(k) & {\sigma_{xz}}^2(k) \\ {\sigma_{yx}}^2(k) & {\sigma_{yy}}^2(k) & {\sigma_{yz}}^2(k) \\ {\sigma_{zx}}^2(k) & {\sigma_{zy}}^2(k) & {\sigma_{zz}}^2(k) \end{bmatrix}$$

describing the uncertainty associated with that estimate.

### A. Prediction

Whenever vehicle $i = 1$ obtains proprioceptive measurements $\boldsymbol{u}_1(k)$ from its dead-reckoning sensors, $\boldsymbol{x}_1(k)$ and $\boldsymbol{P}_1(k)$ are propagated

$$\overline{\boldsymbol{x}}_1(k+1) = g(\boldsymbol{u_1}(k), \boldsymbol{x_1}(k)) \quad (1)$$

$$\overline{\boldsymbol{P}}_1(k+1) = \boldsymbol{G}_1(k+1)\boldsymbol{P}_1(k)\boldsymbol{G}_1^T(k+1) + \boldsymbol{Q}_1(k+1) \quad (2)$$

where $\boldsymbol{Q}_1(k+1)$ is a diagonal matrix where the elements contain the variance of the proprioceptive sensor noise, which is modeled as mean-free Gaussian noise and $\boldsymbol{G}_1(k+1)$ is the Jacobian containing the partial derivatives of $g$

$$\frac{\partial g(\boldsymbol{u}_1(k+1), \boldsymbol{x}_1(k))}{\partial \boldsymbol{x}_1(k)}$$

### B. Update

If vehicle 1 at a later time $l$ receives a broadcast from vehicle 2 that contains $\overline{\boldsymbol{x}}_2(l)$ and $\overline{\boldsymbol{P}}_2(l)$ together with an intra-vehicle range measurement $r_{1,2}(l)$, it uses this information to update its estimate of its own position as follows:

First, it computes what the predicted range $z_{1,2}(l)$ between the two vehicles would be, based on their estimated position.

$$z_{1,2}(l) = \|\overline{\boldsymbol{x}}_1(l) - \overline{\boldsymbol{x}}_2(l)\|_2$$

The difference between the predicted measurement and the measured distance $z_{1,2}(l) - r_{1,2}(l)$ represents the innovation.

It then builds the combined covariance matrix

$$\overline{\boldsymbol{P}}_{1,2}(l) = \left[\begin{array}{c|c} \overline{\boldsymbol{P}}_1(l) & 0 \\ \hline 0 & \overline{\boldsymbol{P}}_2(l) \end{array}\right]$$

and computes the Jacobian $\boldsymbol{H}_{1,2}(l)$ that contains the derivatives of the range measurement with respect to the position of vehicle 1 and 2 (time index $l$ omitted on matrix components).

$$\boldsymbol{H}_{1,2}(l) = \begin{bmatrix} \frac{\partial r}{\partial \overline{x}_1} & \frac{\partial r}{\partial \overline{y}_1} & \frac{\partial r}{\partial \overline{z}_1} & \frac{\partial r}{\partial \overline{x}_2} & \frac{\partial r}{\partial \overline{y}_2} & \frac{\partial r}{\partial \overline{z}_2} \end{bmatrix}$$

Using the residual covariance and the variance

$$\boldsymbol{S}_{1,2}(l) = \boldsymbol{H}_{1,2}(l)\overline{\boldsymbol{P}}_{1,2}(l)\boldsymbol{H}_{1,2}^T(l) + \sigma_r$$

and $\sigma_r$ associated with the exteroceptive (range) sensor we compute the Kalman gain

$$\boldsymbol{K}_{1,2}(l) = \overline{\boldsymbol{P}}_{1,2}(l)\boldsymbol{H}_{1,2}^T(l)\boldsymbol{S}_{1,2}^{-1}(l)$$

that represents a weighting factor for how much the measurement will affect the updated position. Using the innovation $z_{1,2}(l) - r_{1,2}(l)$ and the Kalman gain, the updated position estimate is

$$\boldsymbol{x}_1(l) = \overline{\boldsymbol{x}}_1(l) + \boldsymbol{K}_{1,2}(l)\Big(z_{1,2}(l) - r_{1,2}(l)\Big) \quad (3)$$

and the combined covariance is

$$\boldsymbol{P}_{1,2}(l) = \left[\begin{array}{c|c} \boldsymbol{P}_1(l) & 0 \\ \hline 0 & \boldsymbol{P}_2(l) \end{array}\right] = \Big(\boldsymbol{I}_{6\times 6} - \boldsymbol{K}_{1,2}(l)\boldsymbol{H}_{1,2}(l)\Big)\overline{\boldsymbol{P}}_{1,2}(l) \quad (4)$$

from which we can extract the updated covariance estimate for vehicle 1 $\boldsymbol{P}_1(l)$. Note that we also obtain an updated estimate for the position and covariance of vehicle 2 $\boldsymbol{P}_2(l)$ and $\boldsymbol{x}_2(l)$.

## IV. Problem Statement

We make the following assumptions with regard to our group of vehicles.

*Vehicles*

The group consists of an arbitrary number of vehicles. Vehicles are not required to maintain a hierarchy, but each of them has a unique id. No vehicle needs to be aware of the size of the group.

*Sensors*

Each vehicle needs to have proprioceptive sensors to obtain dead-reckoning results and a single or several exteroceptive sensors to obtain a range, bearing or both to another vehicle. While covariance intersection, the standard algorithm for consistent data fusion, requires a range *and* bearing estimate, the particular strength of our approach lies in its

ability to incorporate measurements from only one such sensor.

*Communication*

Vehicle to vehicle communication is based on unacknowledged broadcast only. Our approach does not require any vehicle to receive all messages that were broadcasted.

## V. The Interleaved Update Algorithm

### A. Initialization

For the Interleaved Update algorithm, each vehicle $i$ at time $k$ now maintains a set $\boldsymbol{\mathcal{X}}_i(k)$ of state estimate vectors together with a set $\boldsymbol{\mathcal{P}}_i(k)$ of associated covariance matrices. In the elements of $\boldsymbol{\mathcal{X}}_i(k)$ and $\boldsymbol{\mathcal{P}}_i(k)$ the superscript $q$ indicates the filter consisting of the state $\boldsymbol{x}_i^q(k)$ and the covariance $\boldsymbol{P}_i^q(k)$. As we will explain later, the maximum size of the set is $2^n$ where $n$ is the total number of vehicles cooperating for navigation.

$$\begin{aligned}\boldsymbol{\mathcal{X}}_i(k) &= \left\{\boldsymbol{x}_i^1(k),\ldots,\boldsymbol{x}_i^q(k),\ldots,\boldsymbol{x}_i^{2^n}(k)\right\}\\ \boldsymbol{\mathcal{P}}_i(k) &= \left\{\boldsymbol{P}_i^1(k),\ldots,\boldsymbol{P}_i^q(k),\ldots,\boldsymbol{P}_i^{2^n}(k)\right\}\end{aligned}$$

Before vehicle $i$ receives information from any other vehicle the only contents of $\boldsymbol{\mathcal{X}}_i(k)$ and $\boldsymbol{\mathcal{P}}_i(k)$ are $\boldsymbol{x}_i^1(k)$ and $\boldsymbol{P}_i^1(k)$.

$$\begin{aligned}\boldsymbol{\mathcal{X}}_i(k) &= \left\{\boldsymbol{x}_i^1(k)\right\}\\ \boldsymbol{\mathcal{P}}_i(k) &= \left\{\boldsymbol{P}_i^1(k)\right\}\end{aligned}$$

### B. Prediction

Each time vehicle $i$ receives proprioceptive sensor readings it uses the Kalman Filter prediction steps for state and covariance (eq. (1) and (2)) to propagate $\boldsymbol{x}_i^1(k)$ and $\boldsymbol{P}_i^1(k)$.

$$\begin{aligned}\boldsymbol{x}_i^1(k) &\xrightarrow{(1)} \overline{\boldsymbol{x}}_i^1(k+1)\\ \boldsymbol{P}_i^1(k) &\xrightarrow{(2)} \overline{\boldsymbol{P}}_i^1(k+1)\end{aligned}$$

### C. First Update

When vehicle $i$ receives a broadcast from vehicle $j$ at time $l$, it first instantiates a second filter $\overline{\boldsymbol{x}}_i^2(l)$,$\overline{\boldsymbol{P}}_i^2(l)$ by copying the state and covariance matrix from $\overline{\boldsymbol{x}}_i^1(l)$,$\overline{\boldsymbol{P}}_i^1(l)$ the forward predicted version of the initial filter.

$$\begin{aligned}\overline{\boldsymbol{x}}_i^2(l) &= \overline{\boldsymbol{x}}_i^1(l)\\ \overline{\boldsymbol{P}}_i^2(l) &= \overline{\boldsymbol{P}}_i^1(l)\end{aligned}$$

The vehicle also instantiates a matrix $\boldsymbol{T}_i$ where each row represents a filter and each column represents a vehicle number. The entry in matrix $\boldsymbol{T}_i(q,i)$ is the time when vehicle $i$ was last used to update filter $q$.

Using the Kalman update equations (eq. (3) and (4)), we now only update $\overline{\boldsymbol{x}}_i^2(l)$ and $\overline{\boldsymbol{P}}_i^2(l)$. After this update, our sets $\boldsymbol{\mathcal{X}}_\rangle(l)$, $\boldsymbol{\mathcal{P}}_i(l)$ and the matrix $\boldsymbol{T}_i(l)$ look as follows.

All elements of column $i$ in $\boldsymbol{T}_i(l)$ are $l$ as all filters are forward predicted using (1) and (2) up to the actual time $l$. Row 1 in $\boldsymbol{T}_i(l)$ represents the filter which has never been updated by any other vehicle and contains the initial state forward predicted up to $l$.

$$\begin{aligned}\boldsymbol{\mathcal{X}}_i(l) &= \left\{\boldsymbol{x}_i^1(l),\boldsymbol{x}_i^2(l)\right\}\\ \boldsymbol{\mathcal{P}}_i(l) &= \left\{\boldsymbol{P}_i^1(l),\boldsymbol{P}_i^2(l)\right\}\\ \boldsymbol{T}_i(l) &= \begin{bmatrix}0\ldots0 & l & 0\ldots0 & 0 & 0\ldots0\\ 0\ldots0 & l & 0\ldots0 & l & 0\ldots0\end{bmatrix}\\ &\qquad\qquad\ \uparrow \qquad\qquad \uparrow\\ &\qquad\qquad\ i \qquad\qquad j\end{aligned}$$

### D. Subsequent Predictions

The first prediction for vehicle $i$ after the update, propagates both filters using eq. (1) and (2) to $\overline{\boldsymbol{\mathcal{X}}}_i(l+1)$ and $\overline{\boldsymbol{\mathcal{P}}}_i(l+1)$ and all elements in column $i$ in $\overline{\boldsymbol{T}}_i(l+1)$ are set to $l+1$.

$$\begin{aligned}\overline{\boldsymbol{T}}_i(l+1) &= \begin{bmatrix}0\ldots0 & l+1 & 0\ldots0 & 0 & 0\ldots0\\ 0\ldots0 & l+1 & 0\ldots0 & l & 0\ldots0\end{bmatrix}\\ &\qquad\qquad\ \uparrow \qquad\qquad \uparrow\\ &\qquad\qquad\ i \qquad\qquad j\end{aligned}$$

Matrix $\overline{\boldsymbol{T}}_i$ therefore keeps track of which vehicles have been used to update a particular filter as well as the age of the updates. Predictions after $l+1$ up to the next update are propagated the same way, both filters are propagated and all elements in column $i$ of $\overline{\boldsymbol{T}}_i$ are set to the actual time. All other columns remain unchanged.

### E. Broadcast

Every time vehicle $i$ sends out a broadcast, the transmitted data consists out of $\overline{\boldsymbol{\mathcal{X}}}_i$, $\overline{\boldsymbol{\mathcal{P}}}_i$ and $\overline{\boldsymbol{T}}_i$. By maintaining a state $\overline{\boldsymbol{x}}_i^1$ on vehicle $i$ which is continuously propagated and has not been updated with information from vehicle $j$, we make sure that a future broadcast from vehicle $i$ received by vehicle $j$ contains a state which is not cross-correlated with vehicle $j$ and can therefore be used by vehicle $j$ for an update.

### F. Subsequent Updates

The general update case when vehicle $i$ receives a broadcast from $j$ after both vehicles have received broadcasts from various other vehicles and have incorporated those to update their navigation filters looks as follows.

We define $\mathcal{S}_i$ as the set of all $m$ vehicle ids which vehicle $i$ received updates from. $\mathcal{S}_i$ not only contains the ids of which vehicle $i$ has directly received broadcasts from, but also those ids which have been propagated to it through other vehicles. The power set $2^{S_i}$ than contains all $2^m$ possible subsets of these ids. Each subset

$$\overline{\boldsymbol{\mathcal{A}}}_i^1,\ldots,\overline{\boldsymbol{\mathcal{A}}}_i^q,\ldots,\overline{\boldsymbol{\mathcal{A}}}_i^{2^m} \subseteq (2^{S_i} \cup i) \tag{5}$$

then corresponds to a filter maintained in $\overline{\boldsymbol{x}}_i^q$,$\overline{\boldsymbol{P}}_i^q$ which maintains a state that has been updated by the ids in the corresponding subset $\overline{\boldsymbol{\mathcal{A}}}_i^q$ and therefore has cross-correlations with these vehicles. The information about which ids are in the individual subsets is maintained in line $q$ of $\overline{\boldsymbol{T}}_i$ as each line in $\overline{\boldsymbol{T}}_i$ corresponds to a subset of $\overline{\boldsymbol{\mathcal{A}}}_i$.

Similarly there is a set $\mathcal{S}_j$ for all $o$ ids which vehicle $j$ has received broadcasts from.

$$\overline{\boldsymbol{\mathcal{A}}}_j^1, \ldots, \overline{\boldsymbol{\mathcal{A}}}_j^p, \ldots, \overline{\boldsymbol{\mathcal{A}}}_j^{2^o} \subseteq (2^{S_j} \cup j)$$

When vehicle $i$ receives $\overline{\boldsymbol{\mathcal{X}}}_j$,$\overline{\boldsymbol{\mathcal{P}}}_j$ and $\overline{\boldsymbol{T}}_j$ from vehicle $j$ it first adds entries in $\overline{\boldsymbol{\mathcal{X}}}_i$,$\overline{\boldsymbol{\mathcal{P}}}_i$ and $\overline{\boldsymbol{T}}_i$ for all elements of $\overline{\boldsymbol{\mathcal{A}}}_j$ which are not in $\overline{\boldsymbol{\mathcal{A}}}_i$. As a result vehicle $i$ then maintains filters for a new set $\overline{\boldsymbol{\mathcal{A}}}_i$

$$\overline{\boldsymbol{\mathcal{A}}}_i \cup \overline{\boldsymbol{\mathcal{A}}}_j \rightarrow \overline{\boldsymbol{\mathcal{A}}}_i$$

Each filter $\overline{\boldsymbol{x}}_i^q$,$\overline{\boldsymbol{P}}_i^q$ represented by $\overline{\boldsymbol{\mathcal{A}}}_i^q$ is now updated without introducing any additional cross-correlations. This means that $\boldsymbol{\mathcal{A}}_i^q = \overline{\boldsymbol{\mathcal{A}}}_i^q$. To update $\overline{\boldsymbol{x}}_i^q$,$\overline{\boldsymbol{P}}_i^q$ we now find all possible combinations of sets from $\overline{\boldsymbol{\mathcal{A}}}_i$ and $\overline{\boldsymbol{\mathcal{A}}}_j$ s.t.

$$\overline{\boldsymbol{\mathcal{A}}}_i^g \cup \overline{\boldsymbol{\mathcal{A}}}_j^h \rightarrow \boldsymbol{\mathcal{A}}_i^q \tag{6}$$

Each of these combinations represents a possible update for $\overline{\boldsymbol{x}}_i^q$,$\overline{\boldsymbol{P}}_i^q$

$$\overline{\boldsymbol{x}}_i^g \overset{(3)\ with\ \overline{\boldsymbol{x}}_i^h}{\longrightarrow} \boldsymbol{x}_i^q \tag{7}$$

$$\overline{\boldsymbol{P}}_i^g \overset{(4)\ with\ \overline{\boldsymbol{P}}_i^h}{\longrightarrow} \boldsymbol{P}_i^q \tag{8}$$

We now select $g$ and $h$ s.t. $\boldsymbol{P}_i^q$ has the smallest trace of all possible combinations.

$$(g^*, h^*) = \underset{g,h \text{ s.t. } (6)}{\operatorname{argmin}} \Big( trace\big(\boldsymbol{P}_i^q\big) \Big) \tag{9}$$

Using $g^*$ and $h^*$ determined through eq. (9) we use eq. (7) to update the state.

$$\overline{\boldsymbol{x}}_i^{g*} \overset{(3)\ with\ \overline{\boldsymbol{x}}_i^{h*}}{\longrightarrow} \boldsymbol{x}_i^q$$

Line $q$ in $\boldsymbol{T}_i$ is updated to reflect the age of updates.

$$\begin{aligned} \boldsymbol{T}_i^q(i,u) &= \overline{\boldsymbol{T}}_i^{g*}(g*,u) \qquad \forall u \in \overline{\boldsymbol{\mathcal{A}}}_i^{g*} \\ \boldsymbol{T}_i^q(i,u) &= \overline{\boldsymbol{T}}_j^{h*}(h*,u) \qquad \forall u \in \overline{\boldsymbol{\mathcal{A}}}_j^{h*} \end{aligned}$$

All steps in section (V-F) are repeated for all $2^n$ filters on vehicle $i$ and all other vehicles which overheard the broadcast update their local filters accordingly.

### G. Enforcing Constant Set Size

The amount of information which needs to be transmitted during each broadcast, as well as the number of local prediction and update steps grows with $\mathcal{O}(m^2)$ where $m$ is the size of set $\mathcal{S}_i$ as defined in section V-F. The amount of data which needs to be transmitted per filter however is fairly small ($\approx$ 10 bytes) and the update of each filter only requires 4 $[2 \times 6] \cdot [6 \times 6]$ matrix multiplications for a 3D environment where range and heading measurements are available. Assuming a data packet size of 10 kBytes set sizes up to 30 ids are feasible.

If CN is implemented on a group of robots which is much larger than 30 it is worth noting that Roumeliotis *et al.* show in [15] that for a group of robots with the same level of uncertainty in their proprioceptive measurements the uncertainty growth is inversely proportional to the number of robots thus the contribution of each additional robot follows a law of diminishing return. This suggests that set sizes of 30 and less are sufficient to obtain an improvement of navigation accuracy which is close to the theoretical maximum obtained when broadcasts of all available vehicles are incorporated.

Based on our available communications bandwidth and available processing cycles we can choose an upper bound $b$ for the size of $\mathcal{S}_i$. If our set size grows larger than $b$ we can incorporate the new broadcast according to section V-F and than resize $\mathcal{S}_i$ by eliminating the id which contributes the least amount of information. The resize process consists of two steps. First we determine the vehicle (id) which contributes the least amount of information. Second we remove this id from $\mathcal{S}_i$ and modify $\overline{\boldsymbol{\mathcal{X}}}_i$,$\overline{\boldsymbol{\mathcal{P}}}_i$ and $\overline{\boldsymbol{T}}_i$ accordingly.

*1) Compare:* One method to determine the vehicle with id $q$ which contributes the least amount of information is to compare the trace difference between the filter which was only updated by $\{q, i\}$ with the filter that has the dead reckoning result only $\{i\}$.

$$\begin{aligned} q^* \quad = \quad & \underset{q}{\operatorname{argmin}} \Big( trace\big(\boldsymbol{P}_i^g\big) - trace\big(\boldsymbol{P}_i^h\big) \Big) \\ & \forall q \in \mathcal{S}_i, q \neq i \\ & \boldsymbol{P}_i^g \text{ s.t. } \boldsymbol{\mathcal{A}}_g = \{i, q\} \\ & \boldsymbol{P}_i^h \text{ s.t. } \boldsymbol{\mathcal{A}}_h = \{i\} \end{aligned}$$

*2) Eliminate:* After we determined $q^*$ we remove all filters depending on $q^*$ from our sets $\boldsymbol{\mathcal{X}}_i$,$\boldsymbol{\mathcal{P}}_i$ and obtain our new sets $\boldsymbol{\mathcal{X}}_i^-$ and $\boldsymbol{\mathcal{P}}_i^-$ and our updated matrix $\boldsymbol{T}_i^-$ by removing all lines which have a non-zero entry in column $q^*$.

$$\boldsymbol{\mathcal{X}}_i \overset{\boldsymbol{x}_i^h \text{ if } q^* \notin \boldsymbol{\mathcal{A}}_h}{\longrightarrow} \boldsymbol{\mathcal{X}}_i^- \tag{10}$$

$$\boldsymbol{\mathcal{P}}_i \overset{\boldsymbol{P}_i^h \text{ if } q^* \notin \boldsymbol{\mathcal{A}}_h}{\longrightarrow} \boldsymbol{\mathcal{P}}_i^- \tag{11}$$

$$\boldsymbol{T}_i \overset{\boldsymbol{T}_i(g,h)\ \forall g,\ \text{with}\ \boldsymbol{T}_i(g,q^*)=0}{\longrightarrow} \boldsymbol{T}_i^- \tag{12}$$

## VI. Example

The four frames in figure 2 and the tables I through IV show how the sets $\overline{\boldsymbol{\mathcal{X}}}_i$,$\overline{\boldsymbol{\mathcal{P}}}_i$ and the matrix $\overline{\boldsymbol{T}}_i$ evolve over time.

- *k=1* Up to this point all four vehicles have only used dead-reckoning information so none of their positions are cross-correlated. All sets $\overline{\boldsymbol{\mathcal{X}}}_i$,$\overline{\boldsymbol{\mathcal{P}}}_i$ only contain a single state and covariance matrix.
- *k=2* Vehicle 1 broadcasts its state $\boldsymbol{x}_1(2)$ which is received by vehicle 2 and 3. Both vehicles instantiate a second filter $\boldsymbol{x}_2^2(2)$,$\boldsymbol{P}_2^2(2)$ and $\boldsymbol{x}_3^3(2)$,$\boldsymbol{P}_3^2(2)$ respectively which are updated with the broadcast and range received from vehicle 1, while the other filter in both vehicles are not.
- *k=3* Up to k=3 all filters (filter 1 in vehicle 1 and 4, filter 2 in vehicle 2 and 3) are propagated using the Kalman time prediction step. At k=3 the broadcast from 2 is received at 4. As 2 has been previously updated with 1 the set of filters received by 4 contains 2 new ids (1 and 2). Vehicle 4 therefore

instantiates 3 additional filters, each containing a possible permutation of $\mathcal{S}_4$ as specified in eq. (5).

*k=4* At k=4 vehicle 3 receives a broadcast from vehicle 4. After the update vehicle 3 now maintains the maximum set of 8 filters.

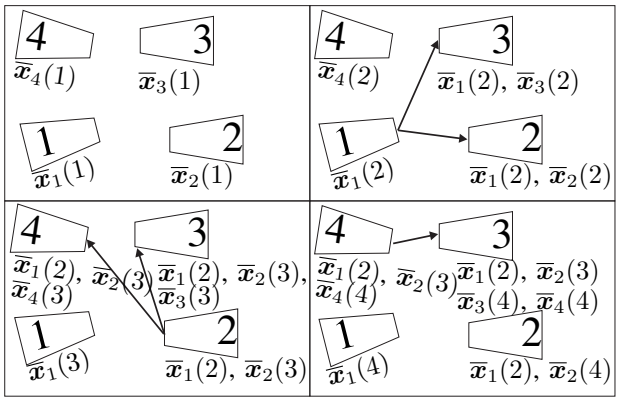


Fig. 2. Four vehicles exchanging navigation information for Cooperative Navigation from time k=1 (top left) to k=4 (bottom right). The arrows indicate which vehicle broadcasts during a particular time step and which vehicles received the broadcast. Below each vehicle are the states which were used to update this vehicle's various position filters.

TABLE I

CONTENTS OF $\boldsymbol{\mathcal{X}}$,$\boldsymbol{\mathcal{P}}$ AND $\boldsymbol{T}$ AT TIME $k = 1$

| $\overline{\boldsymbol{T}}_1(1)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_1(1)$ | $\overline{\boldsymbol{\mathcal{P}}}_1(1)$ |
|---|---|---|---|---|---|
| 1 | | | | $\overline{\boldsymbol{x}}_1^1(1)$ | $\overline{\boldsymbol{P}}_1^1(1)$ |

| $\overline{\boldsymbol{T}}_2(1)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_2(1)$ | $\overline{\boldsymbol{\mathcal{P}}}_2(1)$ |
|---|---|---|---|---|---|
| | 1 | | | $\overline{\boldsymbol{x}}_2^1(1)$ | $\overline{\boldsymbol{P}}_2^1(1)$ |

| $\overline{\boldsymbol{T}}_3(1)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_3(1)$ | $\overline{\boldsymbol{\mathcal{P}}}_3(1)$ |
|---|---|---|---|---|---|
| | | 1 | | $\overline{\boldsymbol{x}}_3^1(1)$ | $\overline{\boldsymbol{P}}_3^1(1)$ |

| $\overline{\boldsymbol{T}}_4(1)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_4(1)$ | $\overline{\boldsymbol{\mathcal{P}}}_4(1)$ |
|---|---|---|---|---|---|
| | | | 1 | $\overline{\boldsymbol{x}}_4^1(1)$ | $\overline{\boldsymbol{P}}_4^1(1)$ |

TABLE II

CONTENTS OF $\boldsymbol{\mathcal{X}}$,$\boldsymbol{\mathcal{P}}$ AND $\boldsymbol{T}$ AT TIME $k = 2$

| $\overline{\boldsymbol{T}}_1(2)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_1(2)$ | $\overline{\boldsymbol{\mathcal{P}}}_1(2)$ |
|---|---|---|---|---|---|
| 2 | | | | $\overline{\boldsymbol{x}}_1^1(2)$ | $\overline{\boldsymbol{P}}_1^1(2)$ |

| $\overline{\boldsymbol{T}}_2(2)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_2(2)$ | $\overline{\boldsymbol{\mathcal{P}}}_2(2)$ |
|---|---|---|---|---|---|
| | 2 | | | $\overline{\boldsymbol{x}}_2^1(2)$ | $\overline{\boldsymbol{P}}_2^1(2)$ |
| 2 | 2 | | | $\overline{\boldsymbol{x}}_2^2(2)$ | $\overline{\boldsymbol{P}}_2^2(2)$ |

| $\overline{\boldsymbol{T}}_3(2)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_3(2)$ | $\overline{\boldsymbol{\mathcal{P}}}_3(2)$ |
|---|---|---|---|---|---|
| | | 2 | | $\overline{\boldsymbol{x}}_3^1(2)$ | $\overline{\boldsymbol{P}}_3^1(2)$ |
| 2 | | 2 | | $\overline{\boldsymbol{x}}_3^2(2)$ | $\overline{\boldsymbol{P}}_3^2(2)$ |

| $\overline{\boldsymbol{T}}_4(2)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_4(2)$ | $\overline{\boldsymbol{\mathcal{P}}}_4(2)$ |
|---|---|---|---|---|---|
| | | | 2 | $\overline{\boldsymbol{x}}_4^1(2)$ | $\overline{\boldsymbol{P}}_4^1(2)$ |

TABLE III

CONTENTS OF $\boldsymbol{\mathcal{X}}$,$\boldsymbol{\mathcal{P}}$ AND $\boldsymbol{T}$ AT TIME $k = 3$

| $\overline{\boldsymbol{T}}_1(3)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_1(3)$ | $\overline{\boldsymbol{\mathcal{P}}}_1(3)$ |
|---|---|---|---|---|---|
| 3 | | | | $\overline{\boldsymbol{x}}_1^1(3)$ | $\overline{\boldsymbol{P}}_1^1(3)$ |

| $\overline{\boldsymbol{T}}_2(3)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_2(3)$ | $\overline{\boldsymbol{\mathcal{P}}}_2(3)$ |
|---|---|---|---|---|---|
| | 3 | | | $\overline{\boldsymbol{x}}_2^1(3)$ | $\overline{\boldsymbol{P}}_2^1(3)$ |
| 2 | 3 | | | $\overline{\boldsymbol{x}}_2^2(3)$ | $\overline{\boldsymbol{P}}_2^2(3)$ |

| $\overline{\boldsymbol{T}}_3(3)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_3(3)$ | $\overline{\boldsymbol{\mathcal{P}}}_3(3)$ |
|---|---|---|---|---|---|
| | | 3 | | $\overline{\boldsymbol{x}}_3^1(3)$ | $\overline{\boldsymbol{P}}_3^1(3)$ |
| 2 | | 3 | | $\overline{\boldsymbol{x}}_3^2(3)$ | $\overline{\boldsymbol{P}}_3^2(3)$ |
| | 3 | 3 | | $\overline{\boldsymbol{x}}_3^3(3)$ | $\overline{\boldsymbol{P}}_3^3(3)$ |
| 2 | 3 | 3 | | $\overline{\boldsymbol{x}}_3^4(3)$ | $\overline{\boldsymbol{P}}_3^4(3)$ |

| $\overline{\boldsymbol{T}}_4(3)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_4(3)$ | $\overline{\boldsymbol{\mathcal{P}}}_4(3)$ |
|---|---|---|---|---|---|
| | | | 3 | $\overline{\boldsymbol{x}}_4^1(3)$ | $\overline{\boldsymbol{P}}_4^1(3)$ |
| 2 | | | 3 | $\overline{\boldsymbol{x}}_4^2(3)$ | $\overline{\boldsymbol{P}}_4^2(3)$ |
| | 3 | | 3 | $\overline{\boldsymbol{x}}_4^3(3)$ | $\overline{\boldsymbol{P}}_4^3(3)$ |
| 2 | 3 | | 3 | $\overline{\boldsymbol{x}}_4^4(3)$ | $\overline{\boldsymbol{P}}_4^4(3)$ |

TABLE IV

CONTENTS OF $\boldsymbol{\mathcal{X}}$,$\boldsymbol{\mathcal{P}}$ AND $\boldsymbol{T}$ AT TIME $k = 4$

| $\overline{\boldsymbol{T}}_1(3)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_1(4)$ | $\overline{\boldsymbol{\mathcal{P}}}_1(4)$ |
|---|---|---|---|---|---|
| 4 | | | | $\overline{\boldsymbol{x}}_1^1(4)$ | $\overline{\boldsymbol{P}}_1^1(4)$ |

| $\overline{\boldsymbol{T}}_2(4)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_2(4)$ | $\overline{\boldsymbol{\mathcal{P}}}_2(4)$ |
|---|---|---|---|---|---|
| | 4 | | | $\overline{\boldsymbol{x}}_2^1(4)$ | $\overline{\boldsymbol{P}}_2^1(4)$ |
| 2 | 4 | | | $\overline{\boldsymbol{x}}_2^2(4)$ | $\overline{\boldsymbol{P}}_2^2(4)$ |

| $\overline{\boldsymbol{T}}_3(4)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_3(4)$ | $\overline{\boldsymbol{\mathcal{P}}}_3(4)$ |
|---|---|---|---|---|---|
| | | 4 | | $\overline{\boldsymbol{x}}_3^1(4)$ | $\overline{\boldsymbol{P}}_3^1(4)$ |
| 2 | | 4 | | $\overline{\boldsymbol{x}}_3^2(4)$ | $\overline{\boldsymbol{P}}_3^2(4)$ |
| | 3 | 4 | | $\overline{\boldsymbol{x}}_3^3(4)$ | $\overline{\boldsymbol{P}}_3^3(4)$ |
| 2 | 3 | 4 | | $\overline{\boldsymbol{x}}_3^4(4)$ | $\overline{\boldsymbol{P}}_3^4(4)$ |
| | | 4 | 4 | $\overline{\boldsymbol{x}}_3^5(4)$ | $\overline{\boldsymbol{P}}_3^5(4)$ |
| 2 | | 4 | 4 | $\overline{\boldsymbol{x}}_3^6(4)$ | $\overline{\boldsymbol{P}}_3^6(4)$ |
| | 3 | 4 | 4 | $\overline{\boldsymbol{x}}_3^7(4)$ | $\overline{\boldsymbol{P}}_3^7(4)$ |
| 2 | 3 | 4 | 4 | $\overline{\boldsymbol{x}}_3^8(4)$ | $\overline{\boldsymbol{P}}_3^8(4)$ |

| $\overline{\boldsymbol{T}}_4(4)$ | | | | $\overline{\boldsymbol{\mathcal{X}}}_4(4)$ | $\overline{\boldsymbol{\mathcal{P}}}_4(4)$ |
|---|---|---|---|---|---|
| | | | 4 | $\overline{\boldsymbol{x}}_4^1(4)$ | $\overline{\boldsymbol{P}}_4^1(4)$ |
| 2 | | | 4 | $\overline{\boldsymbol{x}}_4^2(4)$ | $\overline{\boldsymbol{P}}_4^2(4)$ |
| | 3 | | 4 | $\overline{\boldsymbol{x}}_4^3(4)$ | $\overline{\boldsymbol{P}}_4^3(4)$ |
| 2 | 3 | | 4 | $\overline{\boldsymbol{x}}_4^4(4)$ | $\overline{\boldsymbol{P}}_4^4(4)$ |

## VII. SIMULATION

To validate that the Interleaved Update (IU) algorithm provides consistent estimates we ran a simulation where three moving vehicles took turns exchanging navigation information. Whenever a vehicle did broadcast position information the other two would obtain this information together with a noisy range measurement. No vehicle got a position fix through a simulated GPS update. We ran several simulations where all vehicles used the standard EKF algorithm and naively incorporated every broadcast they obtained, ignoring cross-correlations. We also ran the exact same setup several times with all vehicles using the IU algorithm. Figure 3 and 4 IU show a snap-shot of both runs at $t = 2000\ s$. All vehicles started at $(0, 0)$. Vehicle 1 and 2 ran in concentric squares while vehicle 3 ran on a north-south track. For each vehicle "+" marks the estimated position at this time and centered around it is the 3 $\sigma$-ellipse, while "×" marks the true position. The enlarged sections for each of the vehicles show that the distance between the true and the estimated position in figure 3 is similar to the distance in figure 4 (note the change of scale). However, while the true position for all vehicles is outside the 3 $\sigma_{xx}$-bound (99.6% confidence interval) in figure 3 it is well within it in figure 4. Figure 5 illustrates the evolution of the 1D position error and the associated covariance over time. The top plot shows the results for the EKF estimator. For a long time within the

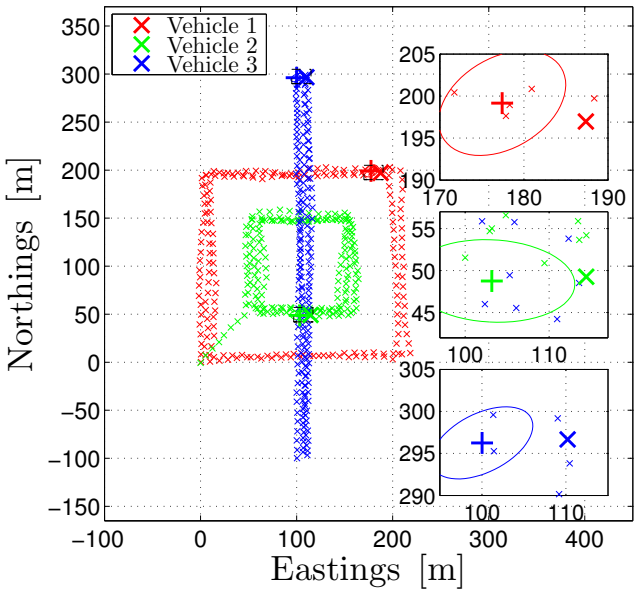


Fig. 3. Tracks of all vehicles and their position at $t = 2000\ s$. All vehicles are incorporating external information using a standard EKF. The enlargements show that for all three vehicles the true position ($\times$) is outside the 3 $\sigma$-ellipse centered around the estimated position ($+$), indicating an overconfident estimate.

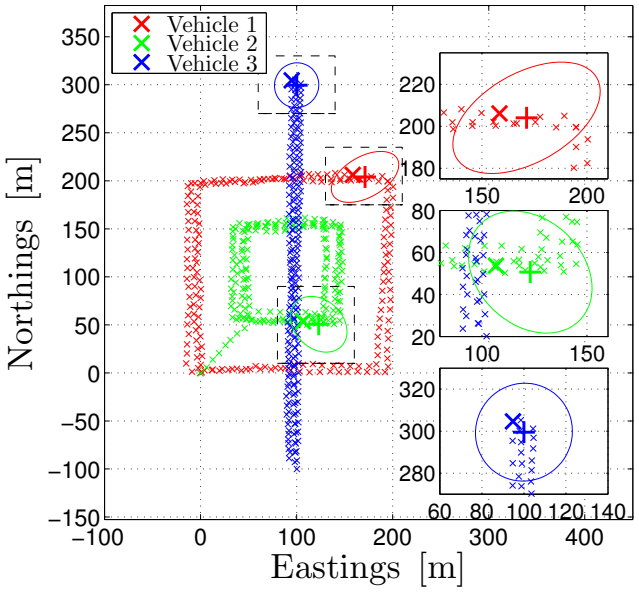


Fig. 4. Tracks of all vehicles and their position at $t = 2000\ s$. All vehicles are incorporating external information using the IU algorithm. The enlargements show that for all three vehicles the true position ($\times$) is within the 3 $\sigma$-ellipse centered around the estimated position ($+$).

observed interval the error in the absolute position estimate for the $x$-component ($|\tilde{x}|$) is outside the 3 $\sigma_{xx}$-bound, a clear indication that the estimator is over confident. The position error grows slowly as no vehicle has access to an absolute position update. This is however not reflected in the 3 $\sigma_{xx}$-bound which stays constant. The bottom plot shows the estimation error for the IU algorithm. Here, the 3 $\sigma_{xx}$-bound grows and the error estimate is well within it. Note that the absolute position error is slightly larger in the IU case. This is due to the fact that the IU algorithm updates very selectively and therefore incorporates less corrective measurements. The slightly larger error is however properly accounted for through a much higher 3 $\sigma_{xx}$-bound and will stay within the predicted bound, while the EKF's overconfidence may cause it to diverge, leading to a rapidly increasing estimation error.

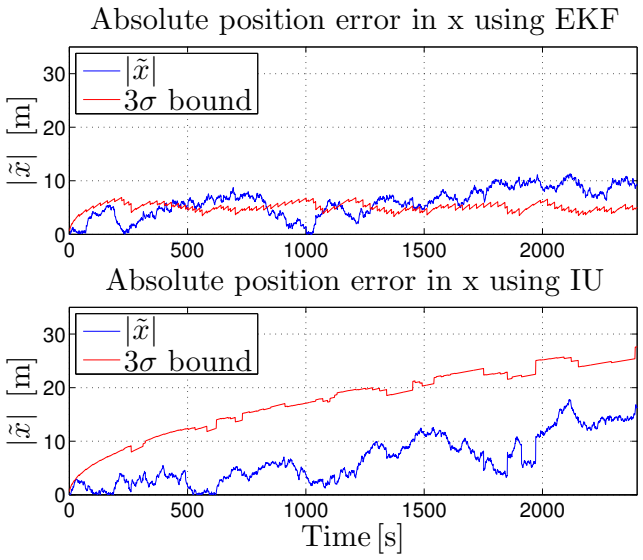


Fig. 5. Position error in the x-direction for one of three vehicles exchanging navigation data as estimated by (top) the standard EKF and (bottom) the Interleaved Update algorithm.

To asses the performance of the standard EKF vs. that of the IU algorithm we computed the Normalized Estimation Error Squared (NEES) as described in [16] for 20 runs (ten standard EKF and ten IU).

$$\epsilon(k) = \tilde{x}(k|k)^T\, P(k|k)^{-1}\, \tilde{x}(k|k)$$

For each time $k$ we compute the $N = 10$ average NEES $\bar{\epsilon}(k)$.

$$\bar{\epsilon}(k) = \frac{1}{N} \sum_{i=1}^{N} \epsilon^i(k) \tag{13}$$

Under the hypothesis $H_0$ that the filter is consistent and under the linear-Gaussian assumption $N\bar{\epsilon}(k)$ will have a chi-square density with $N\,n_x$ degrees of freedom, where $n_x$ is the dimension of $\boldsymbol{x}$. The hypothesis $H_0$, that the state estimation errors are consistent with the filter-calculated covariances, also called *chi-square test*, is accepted if $\bar{\epsilon}(k) \in [r_1, r_2]$ where the acceptance interval is determined such that

$$P\left\{\bar{\epsilon}(k) \in [r_1, r_2]\,|H_0\right\} = 1 - \alpha$$

The two-sided 95 % region for a 20 degree of freedom ($Nn_x = 10 \cdot 2 = 2 = 20$) chi-square distribution is divided by $N$ is

$$\left[\frac{\chi^2_{20}(0.025)}{N}, \frac{\chi^2_{20}(0.975)}{N}\right] = [0.96, 3.42]\,. \tag{14}$$

Figures 6 and 7 show the 10-run average NEES according to (13) and the boundaries determined in (14). For the standard

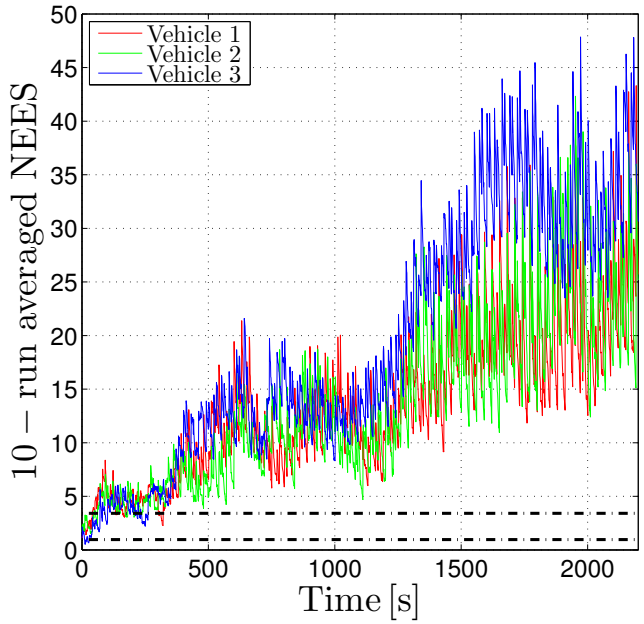


Fig. 6. Averaged NEES for 10 runs using the standard EKF.

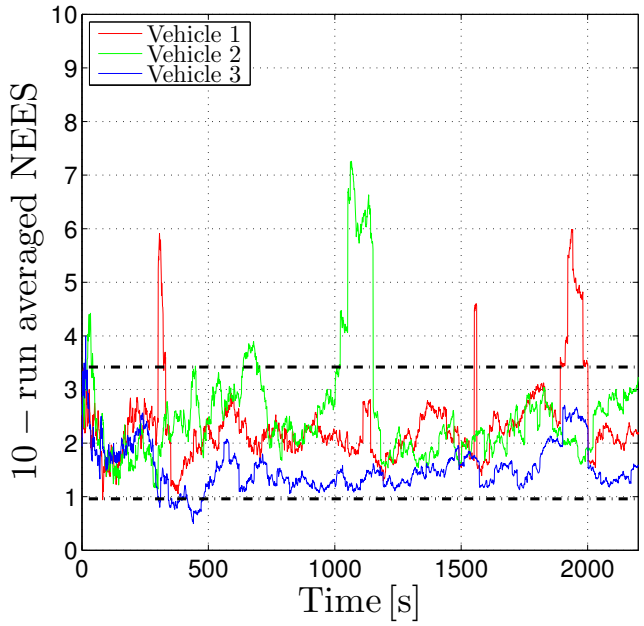


Fig. 7. Averaged NEES for 10 runs as shown using the standard IU.

EKF update the NEES quickly grows above the upper bound (figure 6) and indicates that this approach not only leads to inconsistent results, but that this inconsistency is growing. For the IU algorithm (figure 7) between 5 % and 9 % of the values fall outside the 95 % region which is acceptable [16].

## VIII. Conclusion and Future Work

In this paper we presented an algorithm which addresses the problem of overconfidence in the position estimate of robots exchanging navigation information for cooperative navigation. The assumptions made for our approach are those of real-world robot setups and it allows for vehicles to dynamically join and leave the group of cooperating robots. The algorithm can be adjusted to use a fixed amount of communication overhead and processing cycles such that it can be adapted to the available CPU cycles and communication bandwidth.

Future work will investigate how different set sizes will affect the navigation accuracy, especially in heterogeneous groups where some robots have very accurate dead-reckoning sensors or can often gain access to absolute position information through GPS. We also plan on deploying the algorithm on a fleet of cooperating AUVs.

## References

[1] U. Frese, "Treemap: An o(log n) algorithm for indoor simultaneous localization and mapping," *Autonomous Robots*, vol. 21, no. 2, pp. 103–122, September 2006. [Online]. Available: http://www.springerlink.com/content/b9323114321h3k69/

[2] P. O. Arambel, C. Rago, and R. K. Mehra, "Covariance intersection algorithm for distributed spacecraft state estimation," in *Proc. American Control Conference the 2001*, vol. 6, 25–27 June 2001, pp. 4398–4403.

[3] L. Freitag, M. Johnson, M. Grund, S. Singha, and J. Preisig, "Integrated acoustic communication and navigation for multiple UUVs," in *Proc. MTS/IEEE Oceans 2001*, Honolulu, HI, USA, September 2001, pp. 290–294.

[4] S. Roumeliotis and G. Bekey, "Synergetic localization for groups of mobile robots," in *Proc. 39th IEEE Conference on Decision and Control*, Sydney, Australia, December 2000, pp. 3477–3482.

[5] S. I. Roumeliotis and I. M. Rekleitis, "Propagation of uncertainty in cooperative multirobot localization: Analysis and experimental results," *Autonomous Robots*, vol. 17, no. 1, pp. 1573–7527, July 2004.

[6] S. Roumeliotis and G. Bekey, "Distributed multirobot localization," *Robotics and Automation, IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 781–795, Oct 2002.

[7] S. Grime and H. Durrant-Whyte, "Data fusion in decentralized sensor networks," *Control engineering practice*, vol. 2, p. 849, 1994.

[8] E. Nettleton and H. F. Durrant-Whyte, "Delayed and asequent data in decentralized sensing networks," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, ser. Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference, G. T. McKee and P. S. Schenker, Eds., vol. 4571, Oct. 2001, pp. 1–9.

[9] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proc. American Control Conference*, Albuquerque, New Mexico, USA, June 1997.

[10] ——, "Simultaneous localisation and map building using split covariance intersection," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 29 Oct.–3 Nov. 2001, pp. 1257–1262.

[11] D. Fox, W. Burgard, and S. Thrun, "A probabilistic approach to collaborative multi-robot localization," *Autonomous Robots*, vol. 8, no. 3, pp. 325–344, June 2000. [Online]. Available: http://www.springerlink.com/content/m6505n132qn37561/fulltext.pdf

[12] A. Bahr and J. Leonard, "Cooperative localization for autonomous underwater vehicles," in *Proc. 10th International Symposium on Experimental Robotics (ISER)*, Rio de Janeiro, Brazil, July 2006.

[13] C. Detweiler, J. Leonard, D. Rus, and S. Teller., "Passive mobile robot localization within a fixed beacon field," in *International Workshop on the Algorithmic Foundations of Robotics*, New York, NY, USA, 2006.

[14] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: The MIT Press, 2005.

[15] S. I. Roumeliotis and I. Rekleitis, "Analysis of multirobot localization uncertainty propagation," in *Proc. IEEE Int. Workshop on Intelligent Robots and Systems*, Las Vegas, NV, USA, October 2003, pp. 1763–1770.

[16] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. Wiley Interscience, 2001.