Cooperative Autonomous Underwater Vehicle Localization

Marcelo Borges Nogueira, João Borges Sousa and Fernando Lobo Pereira

***Abstract*—Due to some communication constraints of the underwater environment, cooperative localization for AUVs is a hard task. In this paper, we present the state of the art in this area, analyse the existing solutions to related problems and propose 2 algorithms, based on the EKF, to solve the localization problem with a group of Autonomous Underwater Vehicles equipped with low cost navigation systems in an unknown environment. Without the aid of a global positioning system, the navigation error will grow unbounded. However, navigation performance can be greatly improved with our cooperative navigation scheme, as shown in simulations. We also show that the methods give better results when compared to an existing solution and perform a nonlinear observability analysis of the system to show the boundedness of the error with the aid of a global positioning system.**

I. Introduction

Navigation, which requires some sort of localization, is a central problem of robotics. There are two kinds of sensors used for navigation: proprioceptive sensors (gyros, accelerometers, etc) and external or extereoceptive sensors (GPS, Vision, etc). Independent of the quality of the sensor used, the error in the position estimate based on dead reckoning always grows without bound. Therefore, they must be periodically reset by using information from external sensors. In the underwater environment, if the vehicle is submerged, it cannot use GPS and usually cannot rely on optical instruments (cameras and laser range finders). By surfacing the Autonomous Underwater Vehicle (AUV) is possible to use the GPS to update his position, but it is probably not desirable that it has to surface too often. This way the external sensor usually available are range sensors, based on the time of flight measurements from communications with landmarks.

One way to use range measurements for localization is to employ static beacons in known positions creating a Long Baseline (LBL). But this limits the operation area. To overcome this, instead of static beacons, we can use a group of AUVs that uses each other as landmarks, creating a Moving LBL [1]. The problem of this approach is that, now, the landmarks have some uncertainty associated with their position that may change with time. Another problem with using a group of AUVs that communicate with each other is that the communication between AUVs is unreliable and has a low bandwidth, since it is based on acoustic modems.

In this paper we will introduce two algorithms developed, based on the Extended Kalman Filter, using a group of AUVs all equipped with the same kind of dead reckoning and external sensors. In section II we will present the problem faced in more detail and study some classical approaches to similar problems. In section III we will present the key ideas of the algorithms developed and show some simulations and results in section IV. In section V we perform a observability study of the algorithms proposed and finally some conclusions are draw in section VI.

All the authors are with the Underwater Systems and Technologies Laboratory, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal { marcelo.nogueira@fe.up.pt, jtasso@fe.up.pt, flp@fe.up.pt}

978-1-4244-5222-4/10/$26.00 ©2010 IEEE

II. The Problem and Related Work

When a global positioning system is not available, a classical technique to perform localization is to use a Kalman Filter. If a robot is moving in an environment and there is a map of this environment (known features or landmarks in the environment), you can take measurements (also called observations) like range and/or bearing to map features and use a Kalman Filter to estimate the position of the robot.

The Kalman Filter is a technique that combines the dynamics systems noisy measurements with all other known information to obtain the best state estimates [2]. It is a recursive filter developed by Rudolf Kalman that gives optimal values for linear systems contaminated with Gaussian white noises.

For nonlinear systems, where the state vector in the next time instant is a nonlinear function of the current state vector and of the control signal - the observation can also be a nonlinear function - you can use the Extended Kalman Filter (EKF). A nonlinear systems can be modeled as show in equation (1), where $x(t)$ is the state vector (robot pose - position and orientation - in our case), $u(t)$ is the control signal at time instant t. $y(t)$ is the output measurement, or observation, $q(t)$ and $r(t)$ are the process and the measurement noises (both Gaussian and with zero mean), respectively.

$$\begin{aligned} x(t+1) &= f(x(t), u(t), t) + q(t) \\ y(t) &= h(x(t), u(t), t) + r(t) \end{aligned} \tag{1}$$

The EKF uses linearizations of the functions f (∇F_x and ∇F_u - Jacobians of f w.r.t. x and u respectively) and h (∇H_x

- Jacobian of h w.r.t x), as shown in equation (2):

$$\begin{aligned} e(t) &= y(t) - h(x(t), u(t), t) \\ L(t) &= \nabla H_x P_p(t) \nabla H_x^T + R(t) \\ \text{Prediction:} & \\ \hat{x}_p(t+1) &= f(x(t), u(t), t) \\ P_p(t+1) &= \nabla F_x P(t) \nabla F_x^T + \nabla F_u Q(t) \nabla F_u^T \\ \text{Update:} & \\ k(t) &= P_p(t) \nabla H_x^T L(t)^{-1} \\ \hat{x}(t) &= \hat{x}_p(t) + k(t) e(t) \\ P(t) &= P_p(t) - k(t) L k(t)^T, \end{aligned} \tag{2}$$

where $e(t)$ is called the innovation, $k(t)$ is the Kalman gain , $\hat{x}_p(t)$ and $\hat{x}(t)$ are the predicted and the filtered state estimates, $y(t)$ and $y_p(t)$ are the observation and predicted observation estimate, $P(t)$ and $P_p(t)$ are the filtered and the predicted error covariance matrix. The matrices $Q(t)$, $R(t)$ are given by

$$E\left\{ \begin{bmatrix} q(t) \\ r(t) \end{bmatrix} \begin{bmatrix} q^T(t) & r^T(t) \end{bmatrix} \right\} = \begin{bmatrix} Q(t) & 0 \\ 0 & R(t) \end{bmatrix}.$$

Initial conditions $x_p(0)$ and $p_p(0)$ are needed for the algorithm to work. An update is possible only if an observation is available. Otherwise we make $\hat{x}(t) = \hat{x}_p(t)$ and $P(t) = P_p(t)$.

If a map of the environment is not available it is possible to use a technique called Simultaneous Localization and Mapping or SLAM. A robot can localize itself using a map of landmarks and an onboard sensor which senses these landmarks; conversely, from accurate localization of the robot, it is possible to build a map of the landmarks (determine their spatial coordinates). Performing both tasks simultaneously constitutes the Simultaneous Localization and Mapping (SLAM) problem [3]. This problem has been of much interest in the robotics community since the seminal papers [4] and [5]. A SLAM algorithm builds a consistent estimate of both environment map and vehicle trajectory using noisy proprioceptive and some extereoceptive information [6], [7].

It is possible to use again an EKF to solve a SLAM problem with nonlinear system models [8], [9]. In this case the state vector $x(t)$ is composed of the vehicle pose, $x(t)_V$, and map feature parameters, that is

$$x(t) = [x(t)_V \; x(t)_{F1} \; x(t)_{F2} \; \dots \; x(t)_{Fn}]^T,$$

where $x(t)_{Fi}$ are the parameters of the i^{th} feature at time instant t. The system will be described by equation (3), where f_V describes the motion of the vehicle, and since the features are static, their position remain unchanged over time.

$$\begin{aligned} x(t+1)_V &= f_V(x(t)_V, u(t), t) + q_V(t) \\ x(t+1)_{Fi} &= x(t)_{Fi}, \end{aligned} \tag{3}$$

Notice that the features are not contaminated by noise. The process noise matrix Q and the Jacobian of the function f w.r.t. x are given by equation (4), where Q_V is the process noise of the vehicle motion function f_V and ∇F_{V_x} is the Jacobian of f_V.

$$Q = \begin{bmatrix} Q_V & 0 & \cdots & 0 \\ 0 & 0 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & 0 \end{bmatrix} \quad \nabla F_x = \begin{bmatrix} \nabla F_{V_x} & 0 & \cdots & 0 \\ 0 & I & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & I \end{bmatrix} \tag{4}$$

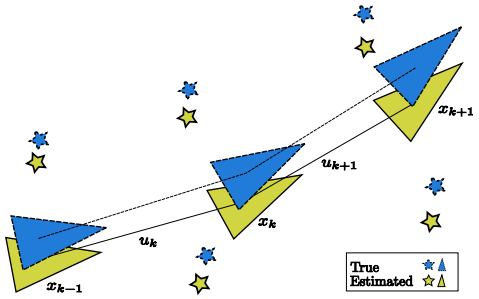


Figure 1. SLAM algorithm result. The final map constructed, and thus the estimated robot trajectory, is an accurate relative map, but there is an error from the true map that depends on the initial uncertainty of the robot.

In figure1 we can see the map and the vehicles trajectory estimated by the SLAM algorithm. Notice that the estimated map/trajectory differs from the true one by some translation and rotation. This is due to the initial uncertainty in vehicles pose as he observes the first features. This way, the error in landmarks are common and highly correlated. This means that even if the true position of the landmarks are not known, the relative location between them may be known with high accuracy. In the EKF implementation this is represented by the off-diagonal terms of the covariance matrix P. As the vehicle observes features, the correlation between them increase monotonically. This means that when a landmark is re-observed and its position updated, all other landmarks that are correlated with it are updated too, even not being observed directly by the vehicle. That is, all landmarks end up forming a network linked by relative locations whose values increase as observations are made. In the limit, a rigid accurate relative map is obtained, with some absolute error depending on the vehicle initial error, as show in figure 1. This also means that the uncertainty of the landmarks (diagonal terms of the covariance matrix P corresponding to landmarks) decrease monotonically to a lower bound as observations are made. This can be seen in equations (2) and (4). Since the features are static and there is no process error Q, the diagonal terms of P corresponding to the landmarks do not increase in the prediction phase of the EKF. In the update phase, if a landmark is observed, its uncertainty is decreased as show in equation (2). Therefore, the error on the estimated position of the vehicle relative to the map is bounded only by the quality of the map and relative measurement sensor [6]. Remind that these results have been proved for the linear case and for Gaussian noises only.

One of the major problems of SLAM is the correct association of the observed features to the ones present in the state vector (current estimated map), also called data association problem. The EKF formulation of SLAM is specially fragile to the correct association of observations to landmarks [10],

and incorrect associations can lead the estimated map to be inconsistent and divergent, causing the completely failure of the algorithm. Data association is particularly important when a vehicle returns to a previously mapped region after a long excursion; the so-called 'loop- closure' problem [7].

According to [11], the problem of position determination has been of considerable interest over the last 4000 years. The basic process of distance measurement, correlation, and triangulation was known to the Phoenicians, who successfully managed to build and maintain quite accurate maps of the Mediterranean area. Maritime and aviation applications require navigation in order to be useful, so navigation is a well-understood quantitative science. So, why robust and reliable autonomous mobile robot navigation remains such a difficult problem? In [11] the authors argue that the reason is clear: it is not the navigation process per se that is a problem, it is the reliable acquisition or extraction of information about navigation landmarks, from sensor information, and the automatic correlation or correspondence of these with some navigation map that makes the autonomous navigation problem so difficult. If you use artificial features, such as beacons, that can communicate to the vehicle and identify themselves, then the association problem is trivially solved.

A special case of SLAM is when only range measurements to landmarks are available. If there are n features in the environment, and the vehicle observes feature i at time t, the observation equation is given by

$$h(x) = \sqrt{(x_V - x_i)^2 + (y_V - y_i)^2},$$

where (x_V, y_V) and (x_i, y_i) are the position of the vehicle and of the i^{th} landmark in the global reference frame, respectively. In this case, a single measurement does not contain enough information to determine the location of a landmark. If there is no a priori information about the position of the landmarks, this partial observability requires the fusion (processing) of several observations from different vehicle positions (in a 2D scenario at least 3 non-collinear measurements) in order to determine the landmark location[12], [13]. But once the positions of the landmarks have been initialized, or prior positions are known, we can use each observation to better estimate the position of both the vehicle and landmark using SLAM.

In the SLAM problem the map usually consist of natural or artificial features found in the environment. In underwater environment it is hard to find natural features, unless the vehicle is close to the sea floor. So often artificial ones are employed, like floating devices or beacons at sea bottom (static features). This limits the operation area and requires a substantial deployment effort before operations, especially in deep water [14]. But what if other AUVs are used as features? In this way we could have several AUVs, each one executing its own mission cooperatively and also use each others navigation information to better estimate their own positions. This scenario, where several AUVs work cooperatively to fulfill a mission is very interesting. For example, searching a particular area can be done more quickly using multiple AUVs. Another good scenario are dangerous missions, where it is unlikely that one AUV will survive long enough, and it may be better to use several inexpensive but prone to failure AUVs instead of one single and much more expensive one.

When a group of robots involved in a mission is composed of different platforms carrying different proprioceptive and extereoceptive sensors, or executing different trajectories through the environment, the quality of the localization estimates will vary significantly across the individual members at a given time. If the members can sense and communicate with its colleagues at all times, then every member of the group would have less uncertainty about its position than the robot with the best localization results [15]. Kurazume et al [16] proposed a method that divides ground operating robots into two groups, A and B. The group A remains stationary and acts as a landmark while group B is in movement. Group B then stops and acts as a landmark for group A. But for an AUVs it is hard to stay stationary, and besides this, we would like that all AUVs to execute their own mission without bothering to help the localization of the others AUVS. Alexander Bahr et al [14] proposed a solution with this scenario. A subgroup of them were called Communication and Navigation Aid-AUVs (CNAs). The CNAs could maintain an accurate estimate of their position through sophisticated DVL, INS sensors or even GPS. Bahrs approach will be discussed in section IV. This is also the idea for the Moving Long Base-Line in [1]. Their methods make use of the good pose estimation by the CNAs and low noise measurements by the AUVs, taking into account that underwater communication is slow. But we would like to devise a method that works without the need of CNAs, that is, all the AUVs involved have similar low cost navigation systems, with relatively high process and measurement noises.

Most properties of SLAM are proved for still map features, but SLAM can still be used for moving features [8], [17]. The error in the pose estimation of the vehicles will not be bounded, but can still be greatly reduced[18]. In the standard SLAM algorithm, to be able to update the state vector, at every step of the algorithm, all the features would have to send their control signal to a central processing unity, to compose the control signal $u(t) = [u(t)_{AUV}\, u(t)_{F1}\, u(t)_{F2} \dots\, u(t)_{Fn}]^T$, where $u(t)_{AUV}$ and $u(t)_{Fi}$ are the control signal of the AUV and of the i^{th} moving feature and at time instant t, respectively. This operation would require a high band width communication between the AUV and the features. This is the idea used in [18][15], since it is applied to land or air vehicles that can rely on a fast communication.

The problem is that underwater communication is slow and unreliable. Radio or optical communication can be used only for distances of a few meters, due to the strong attenuation of electro-magnetic waves underwater. For long range distances, we have to use acoustic modems. Since sound propagation is dependent on temperature and salinity of the water, the acoustic communication channel is unreliable and its performance hard to predict [14]. Another problem is that range calculations based on underwater communication (TOF) is contaminated by several noise sources, some of them not Gaussian. One source of error is the one mentioned just above: the fluctuating speed of sound in water. Multipath is another error source, when the signal, instead of traveling directly from the source to the destination, reflects on the ocean surface or floor first. Both

this sources are related to environmental conditions, which does not change rapidly, and can be non-Gaussian [13]. Since Kalman filters gives optimal solution for Gaussian noise only, the outliers measurements must be removed before the filtering stage. In this work we do not consider those sources of noise on the simulations, but in a real application a technique to remove the outliers, such as the one presented in [13], must be used.

With those considerations in mind, we developed some algorithms using the Extended Kalman Filter and taking the low band width into account, using cooperative localization with n AUVs. These algorithms belong mainly to two classes: based on Localization algorithms and based on SLAM algorithms, which will be better explained in section III.

III. Algorithms Developed

All the algorithms developed consider one AUV as central processor unit (from now on called CAUV), and the others AUVs as features on the environment (called FAUVs), which communicate with the CAUV. The communication between the CAUV and the FAUVs consist of transmitting to each other their pose, pose uncertainty (error covariance matrix $P(t)$) and some additional data (as will be discussed). Using synchronized clocks or calculating the time of flight of the communications, the communicating AUVs can estimate the distance to each other. So the observation consists of range only. Since the FAUVs transmits its current estimated position, there is always a priori knowledge of the "landmark" location. This eliminates the partial observability of the range only sensor, as discussed in section II. While no observation is available, each AUV uses an EKF and proprioceptive information (obtained through a compass or an Inertial Navigation System (INS), for example) to estimate its current position and update its error covariance matrix.

We assume that it may be possible for one of the FAUVs to surface once in a while to update its position and heading trough a GPS (although GPS cannot directly measure heading, it is possible for a sequence of measurements if the AUV is moving, or using some other heading sensor, such as a compass). The chosen FAUV to surface could be the closest to the surface, or the one with the lowest priority task at the moment, for example. The CAUV is considered to be submerged and executing a high priority task, and therefore cannot surface to update his position during the simulations.

In the first algorithm, called Alg1, the position uncertainty of the currently communicating FAUV (denominated from now on as CCFAUV) is added the the sensor error covariance $R(t)$ of the CAUV. This is done by using an Extended Kalman Filter in which the state vector is composed by the pose of the CAUV, $x(t)_{CAUV}$, and by the pose of the currently communicating FAUV (CCFAUV) $x(t)_{CCFAUV}$:

$$x_p(t) = [x(t)_{CAUV} \;\; x(t)_{CCFAUV}]^T.$$

The predicted error covariance matrix $P_p(t)$ is given by

$$P_p(t) = \begin{bmatrix} P(t)_{CAUV} & 0 \\ 0 & P(t)_{CCFAUV} \end{bmatrix},$$

where $P(t)_{CAUV}$ and $P(t)_{CCFAUV}$ are the error covariance matrix of the CAUV position and CCFAUV position, at time instant t, respectively. Notice that we assume that there is no correlation between the positions of the AUVS, since the off-diagonal terms of P_p are zero. Even if the current FAUV communicating has been observed before, the off-diagonal terms are set to zero. This false assumption will lead to a repetitive use of the same evidence and an overconfidence of the robots pose, as discussed in [19]. After acquiring the data from the FAUV, the CAUV uses the EKF to update the position and uncertainty of both the CAUV and FAUV at the same time. In the same way, the FAUV can use the CAUV information to proceed the same calculations and update his position and uncertainty, or his new pose can be sent by the CAUV after the calculations. In between observations, all the AUVs return their state vector and error covariance matrix to the standard mode (its own pose only for the state vector and its pose uncertainty for the error covariance matrix).

Algorithm 1 Alg1 running on CAUV

1) Initialize State Vector and Error Covariance Matrix
2) while true
   a) if an observation is available then
      i) Construct the augmented State Vector $\hat{x}_p(t)$ composed by CAUV position and CCFAUV transmitted position: $\hat{x}_p(t) = [\hat{x}(t)_{CAUV}\, \hat{x}(t)_{CCFAUV}]^T$
      ii) Construct the augmented error cov. matrix $P_p(t)$ composed by CAUV covariance matrix $P(t)_{CAUV}$ and CCFAUV transmitted error cov. matrix $P(t)_{CCFAUV}$
      iii) Update current CAUV and CCFAUV position and their uncertainties, $\hat{x}(t)$ and $P(t)$, using EKF update equations.
   b) else
      i) Predict next CAUV position $\hat{x}_p(t+1)_{CCFAUV}$ and uncertainty $P_p(t+1)_{CCFAUV}$ using EKF predict equations.
   c) endif
   d) t=t+1
3) endwhile

The second algorithm (Alg2) is base on the SLAM algorithm and uses a Extended Kalman Filter, running on the CAUV, in which the state vector is the pose of all AUVs in the system, as show in equation (5):

$$x(t)_{SEKF} = [x(t)_{CAUV}\; x(t)_{FAUV1}\; \ldots\; x(t)_{FAUVn}]^T. \quad (5)$$

We will call this SLAM Extended Kalman Filter present at the CAUV as SEKF. The SEKF takes a step every time an observation is available. The CAUV also runs, in between observations, a regular EKF in which the state vector is the pose of the CAUV, called REKF. The goal of the REKF, also present in all FAUVs, is to uses proprioceptive information to predict the AUV current pose and update its error covariance matrix.

In Alg2, when an observation is available, the CAUV uses his current position in the SEKF $\hat{x}(t)_{CAUV}$ and his current predicted position by the REKF $\hat{x}(t)_{REKF}$ to calculate the control signal $u(t)_{CAUV}$ that took the CAUV from $\hat{x}(t)_{CAUV}$ to $\hat{x}(t)_{REKF}$. Similarly, the CAUV uses the last pose received by the CCFAUV, $\hat{x}(t_{CCFAUV})_{CCFAUV}$, and the current received one, $\hat{x}(t)_{CCFAUV}$, to compute a control signal, $u(t)_{CCFAUV}$ that took the CCFAUV from

$\hat{x}(t_{CCFAUV})_{CCFAUV}$ to $\hat{x}(t)_{CCFAUV}$. In order to compute the control signal of the CCFAUV, the CAUV contains a table with the last communicated pose $\hat{x}(t_i)_i$, $i = 1..n$ for each FAUV. We assume that all others FAUV, other than the one communicating, are not moving, so their control signal are zero. This way, the control signal of the SEKF is given by $u(t)_{SEKF} = [u(t)_{CAUV}\ 0\ 0\ \ldots\ u(t)_{CCFAUV}\ \ldots\ 0]^T$. The ignored motion of the FAUVs will be considered when each one of them becomes the CCFAUV in future observations.

The SEKF process noise covariance matrix is given by the error covariance matrix calculated by the REKF running on the CAUV and on the CCFAUV, as show in equation (6). For all the other FAUV the process noise is zero.

$$Q(t)_{SEKF} = \begin{bmatrix} P(t)_{CAUV} & 0 & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \ldots & 0 \\ 0 & \ldots & P(t)_{CCFAUV} & \ldots & 0 \\ \vdots & \vdots & \ddots & \ldots & 0 \\ 0 & 0 & 0 & \ldots & 0 \end{bmatrix} \quad (6)$$

In order that $P(t)_{REKF}$ in each AUV represents only the noise during the last taken observation and the current one, every time a observation is taken, the error covariance matrix is set to zero both in the CAUV and in the CCFAUV.

According to the Prediction equation of the Extended Kalman Filter we have

$$P_p(t+1) = \nabla F_x P(t) \nabla F_x^T + \nabla F_u Q(t) \nabla F_u^T.$$

Supposing that $Q(t) = Q$ is constant and calling $\nabla F_x = A$, $\nabla F_u = B$ we can conclude that

$$P_p(t) = (A)^t P(0) (A^T)^t + (A)^{t-1} B Q B^T (A^T)^{t-1} + \ldots + B Q B^T.$$

In the REKF that runs in each vehicle, since we set $P(t)_{REKF} = 0$ every time there is an observation, considering this the time $t = 0$ (in the REKF algorithm) we would have

$$P_{REKF}(t) = (A)^{t-1} B Q B^T (A^T)^{t-1} + \ldots + B Q B^T.$$

This way, the SEKF Prediction equation, considering for simplicity that we have only 2 vehicles (one CAUV and one FAUV) in the system, must be

$$P_{SEKF}(t+1) = \begin{bmatrix} (A_1)^{k_1} & 0 \\ 0 & (A_2)^{k_2} \end{bmatrix} P_{SEKF}(t) \begin{bmatrix} (A_1^T)^{k_1} & 0 \\ 0 & (A_2^T)^{k_2} \end{bmatrix} + Q(t)_{SEKF},$$

where $A_i = \nabla F_x$ for the i^{th} vehicle and k_i is the number of time steps executed by the REKF since the i^{th} vehicle has been observed. In the case where there are more vehicles, A_i will be the identity matrix if the i^{th} vehicle is not the CCFAUV. Since A_i depends on $u(t)$ and $x(t)$, which change at every REKF step, $(A_i)^{k_i}$ must be computed by each vehicle during their execution of the REKF and send it to the CAUV, together with $P(t)_{REKF}$, when it is observed. The matrix $(A_i)^{k_i}$ is a n x n matrix, where n is the size of the state vector of the system. But if the vehicles are modeled by a 2D unicycle model, the matrix $(A_i)^{k_i}$ can be fully described by only 2 parameters.

With this information it is possible to use the SEKF to compute the estimated position for all AUVs in the system. After this computation, the CAUV sends to the CCFAUV his new estimated pose and pose uncertainty.

Algorithm 2 Alg2 running on the CAUV

1) Initialize State Vector and Error Covariance Matrix
2) while true
   a) if an observation is available then
      i) Compute the control signal $u(t)_{CAUV}$ that took the CAUV from $\hat{x}(t)_{CAUV}$ to $\hat{x}(t)_{REKF}$
      ii) Compute the control signal $u(t)_{CCFAUV}$ that took the CCFAUV from $\hat{x}(t_{CCFAUV})_{CCFAUV}$ to $\hat{x}(t)_{CCFAUV}$ and compose the augmented control signal $u(t)_{SEKF} = [u(t)_{CAUV}\, 0\, 0 \ldots u(t)_{CCFAUV} \ldots 0]^T$
      iii) Construct the augmented process noise matrix $Q(t)_{SEKF}$ with $P(t)_{CAUV}$ and $P(t)_{CCFAUV}$
      iv) Update the state vector $\hat{x}(t)_{SEKF}$ and uncertainty $P(t)_{SEKF}$ using SEKF predict and update equations
      v) Transmit to the CCFAUV his new estimated pose $\hat{x}(t)_{FAUVi}$ and uncertainty $P(t)_{FAUVi}$
      vi) Reset the error cov. matrix in the CAUV and CCFAUV
   b) else
      i) Predict CAUV next pose $\hat{x}_p(t+1)_{REKF}$ and uncertainty $P_p(t+1)_{REKF}$ using REKF predict equations.
   c) endif
   d) t=t+1
3) endwhile

IV. Simulations and Results

The vehicle model used for all AUVs in the simulations was a 2D model (unicycle model) where the control input $u(t) = [v_x\, v_y\, v_\Theta]^T$, where v_x and v_y are the velocity in the x and y axes and v_Θ is the turn rate of the vehicle at each time step. This would be the dead reckoning information available, for example, by an Inertial Measurement Unit. This way, the process noise have 3 values: $q(t) = [q(t)_{vx}\, q(t)_{vy}\, q(t)_{tr}]^T$, where q_{vx} and q_{vy} are the noise in the velocity in x and y axes, respectively, and q_{tr} is the noise, in degrees, in the turn rate. The matrix Q is a diagonal matrix with 0.02 in the first 2 terms and 0.35 in the third. The external sensor, a range detector, had an error variance $R = [0.5]$. All the noises are consider white noises with zero mean. In all the simulations the time step was 0.1 seconds.

Figure 2 shows a simulation of Alg2 with 3 FAUVs, observation time of 5 seconds and no FAUV GPS update. In the figure we can see the true trajectory described by the AUVs (FAUVs describes circles), the filtered trajectory estimated by the algorithm and a prediction only estimation (prediction equations from 2), using only proprioceptive information. Notice that the prediction only trajectory drifts away from the true trajectory for all AUVs as time increases, as expected, especially for the CAUV, which is moving faster in this simulation. On the other hand, the filtered estimation stays close to the true trajectories and drifts away slower.

The results shown from now on are from simulations where all the AUVs have the same trajectory (identical control signals - the trajectories are similar to the trajectory described by the CAUV in figure 2), but the initial position and orientation are random. All the experiments are repeated 400 times to get the mean error value from all the executions. The error in each execution, err, is the mean of the euclidean error between the the true trajectory, in x and y axes, described by the CAUV (CAUV_{true}) and the filtered one (CAUV_{filt}) for each time

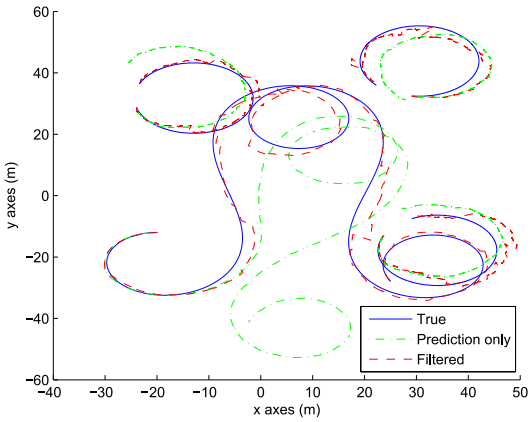


Figure 2. 2D simulation of Alg2 during 320 seconds with 3 FAUVs ploted on x and y axes. FAUVs describes circles at speed of .2 m/s, while CAUV has a longer trajectory at speed of 1 m/s. The observation time is 5 seconds, and there is no FAUV position update by GPS.

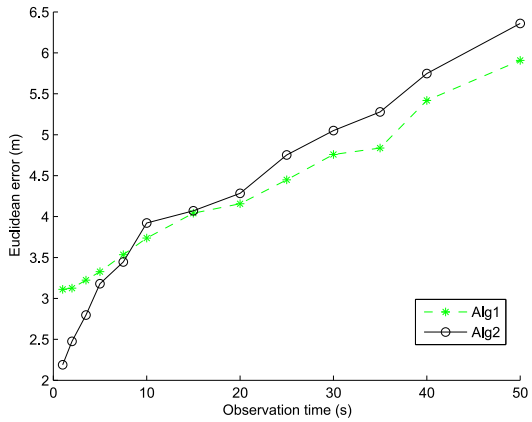


Figure 3. Mean Euclidean CAUV error varying the observation time. Total of 4 AUVs and there is no FAUV position update by GPS

Algorithm	Mean Euclidean Error (m)	Variance
Prediction	8.2	20.5
CONA	11.6	21
Alg1	3.3	2.4
Alg2	3.0	1.6

Table I
MEAN EUCLIDEAN ERROR AND VARIANCE OF THE ALGORITHMS FOR CAUV DURING 300 SECONDS, WITH AN OBSERVATION TIME OF 5S AND NO GPS UPDATE. THE TOTAL DISTANCE TRAVELED BY THE AUVS WAS 300M.

step of the algorithm, as show in equation (7).

$$err = \frac{1}{nsteps} \sum_{k=0}^{nsteps} \| \, CAUV_{true}(k) - CAUV_{filt}(k) \, \|_2 \quad (7)$$

As show in table I, if the observation time is 5s, the mean error for Alg1 is 3.3m and for Alg2 is 3.0m. With this same parameters, the error for the algorithm proposed by Alexander Bahr et al [14], called Cooperative Navigation Algorithm (CONA), that adjusts the position of the CAUV only, has a mean error of 11.6m, even higher than the prediction only mean error (8.1m). This happens because CONA is based on circles intersection to estimate the CAUV position. If the systems noise (process and sensor) is high, the calculated circles positions have some error, and this can causes a high error on the estimated CAUV position depending on the position of the FAUVs involved [20]. If CONA is modified to estimate not only the CAUV position but FAUV positions as well, the mean error gets even bigger (22.3m).

Figure 3 shows the mean CAUV error of the algorithms for several observation times while the number of FAUVs is constant and equals to 3 and there is no GPS update for the FAUVS. You can see that when the observation time is lower than 10 seconds, Alg2 is better, but when the observation time grows above 25s, Alg1 have better results. This may happen because when observation time is low, the problem is closer to a SLAM with moving features. The assumption that, other than the one communicating, all others FAUVs are not moving is not to harmful and the correlation between all the FAUVs is well built. In the other hand, as the observation time increases Alg1 do not loose performance as fast as Alg2. This is due to the no correlation assumption of this algorithm, which makes a repetitive use of the same evidence, causing the robots to decrease their pose uncertainty more than is warranted by data. One way to reduce this effect, as discussed in [19], is to ignore repetitive observations in short time intervals. That is exactly what happens when the observation time gets high.

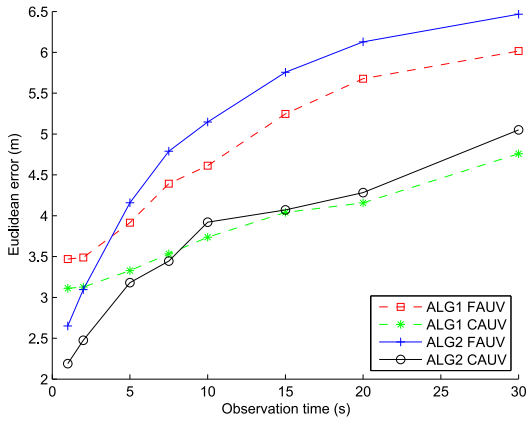
In figure 4(a) we can see the mean error for the FAUVs. You can notice the the FAUV error is higher than the CAUV. This happens because, if there are n FAUVs in the system, the mean observation time for a FAUV is n times longer than for the CAUV. Since in this simulation there are 3 FAUV, the mean observation time for a FAUV is 3 times longer than for the CAUV. So, in a simulation with observation time t_o, the mean error in the pose of the FAUV is approximately equals to the mean error of the CAUV in a simulation with observation time $3t_o$. This was confirmed in 4(b), where the horizontal axis (observation time) of the FAUV error graphic was multiplied by 3 and plotted against the CAUV error.

Notice that with no GPS update, the error is not bounded, and grows as the CAUV moves. The regions where the error gets smaller are curves regions, where for small periods of time the error tends to get smaller. But even if the error is not bounded, you can see that the algorithms proposed greatly reduce the mean error compared with the prediction only approach.

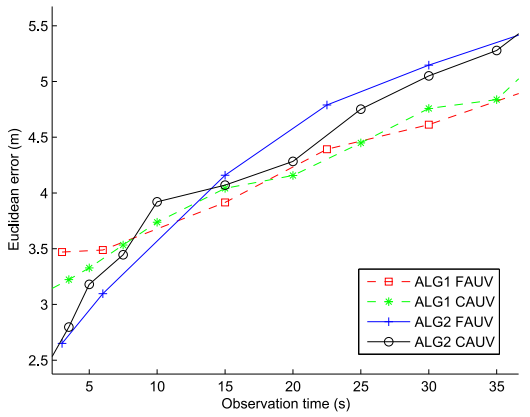
V. OBSERVABILITY ANALYSIS

In this section we perform an observability analysis of the algorithms proposed. To study the observability of the systems, we constructed the observability matrix O. A system is said to be locally weak observable if the rank of O is equal to n (dimension of the state vector).

Since the system is nonlinear, the observability matrix must be constructed using Lie derivatives. Using linearized models of a nonlinear system can lead to wrong results, as discussed in [21].



(a)



(b)

Figure 4. (a) Mean Euclidean CAUV and FAUV error varying the observation time. (b) Same as (a) but dislocating the FAUV graphics by 3 times to the right.

The observability matrix O can be computed by

$$O = [O_0\ O_1\ ...\ O_{n-1}]^T = \frac{\partial l(x)}{\partial x},$$

where

$$l(x) = \begin{bmatrix} L_f^0(h) \\ L_f^1(h) \\ \vdots \\ L_f^{n-1}(h) \end{bmatrix} = \begin{bmatrix} h \\ \dot{h} \\ \vdots \\ h^{n-1} \end{bmatrix}.$$

So

$$L_f^1(h) = \dot{h} = \frac{\partial h}{\partial x}\frac{\partial x}{\partial t} = O_0\dot{x}$$

and

$$L_f^d(h) = \frac{\partial}{\partial x}[h^{d-1}]\frac{\partial x}{\partial t} = \frac{\partial}{\partial x}[L_f^{d-1}(h)]\dot{x} = O_{d-1}\dot{x}.$$

Lets first consider that the system has just 1 FAUV, that is, only 2 AUVs involved. This way, the state vector is given by

$$x = [x_1\ \ x_2]^T = [x_1\ y_1\ \theta_1\ x_2\ y_2\ \theta_2]^T,$$

where x_1 and x_2 are the vector state for the CAUV and FAUV respectively. The differential equations for the unicycle model are given by

$$\dot{x} = [\dot{x}_1\ \ \dot{x}_2]^T u,$$

where

$$\dot{x}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix},\ i = 1, 2 \qquad (8)$$

and $u = [u_1\ u_2]^T$, $u_i = [v_{x_i}\ v_{y_i}\ v_{\Theta_i}]$, is the control signal.

If the system has no GPS update, the observation function, which gives the range between the AUVs, is given by:

$$h(x) = \Delta_{12} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

This way, the row reduced echelon form of the observability matrix will be:

$$O = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & y_1 - y_2 \\ 0 & 1 & 0 & 0 & -1 & x_2 - x_1 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix}.$$

This observability matrix has rank $3 < 6$, and thus the system is not observable, as expected. But even if the covariance error in position grows without bound, the algorithms proposed are able to estimate the states with small error, a lot lower when compared with a prediction only result, as show in figures 5 and 6.

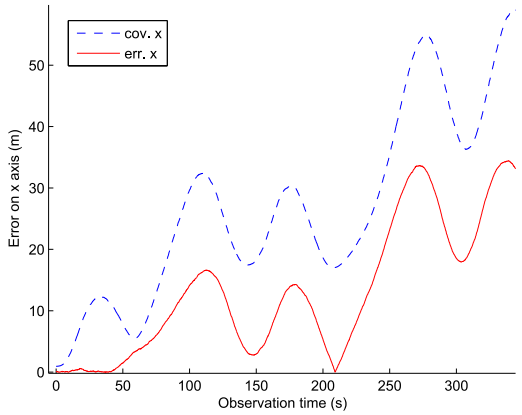


Figure 5. Absolute error on x axis for the CAUV in a prediction only execution and the region of confidence ($3\sigma_x$) calculated based on the covariance matrix $P(t)$

If the GPS updates only the position $(x,\ y)$ of the FAUV, the observability matrix has rank $5 < 6$, and the system is yet not observable. But if the GPS updates the pose (position and heading) of the FAUV, the observation function will be:

$$h(x) = [\Delta_{12}\ \ x_2\ \ y_2\ \ \theta_2]^T.$$

Now the observability matrix is full rank, and the system is locally weak observable. In this case, if the FAUVs have GPS update every time step of the algorithm, the system becomes a simple localization problem with known landmarks. Extending this result for n AUVs, if you consider that at some instant the CAUV has observed all the FAUVs and if only one of them (the k^{th} AUV) has absolute positioning capabilities, as show in equation (9), the system will be locally weak observable.

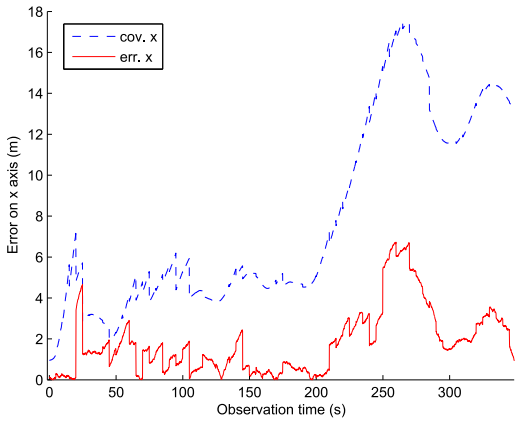


Figure 6. Absolute error on x axis for CAUV in an execution of Alg2, with no GPS update, observation time of 5 seconds and the region of confidence ($3\sigma_{xx}$) calculated based on the covariance matrix $P(t)$

$$h(x) = [\Delta_{12}\ \Delta_{13}\ \cdots\ \Delta_{1n}\ x_k\ y_k\ \theta_k]^T \tag{9}$$

This is show in figure 7. We can observe that during more then 10 minutes of simulation (the AUV moves 700 meters) the error is not increasing (varies from about 1 to 3 meters). Since the observation time is 5 seconds, in between observations the error can grow, but the overall error it is still bounded. In this simulation the FAUV1 updates his pose by GPS constantly, that is, it must be on the surface all the time.

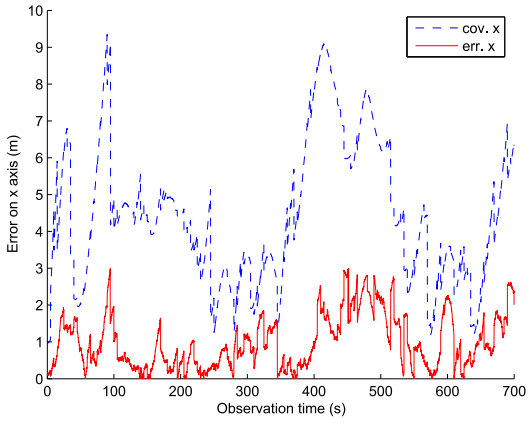
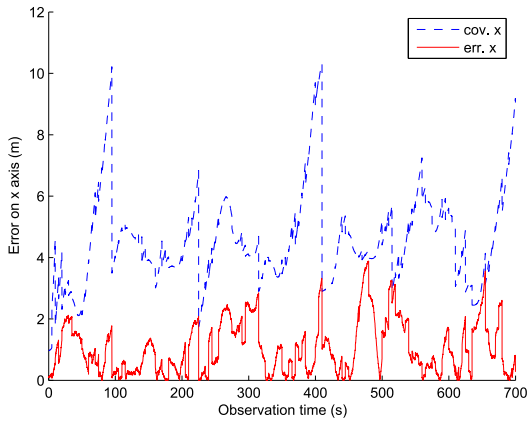


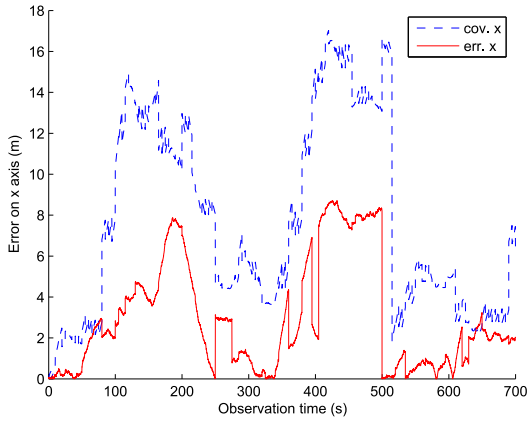
Figure 7. Absolute error on x axis for CAUV in an execution of Alg2 with 3 FAUVs, observation time of 5 seconds, and the region of confidence ($3\sigma_{xx}$) calculated based on the covariance matrix $P(t)$. FAUV1 pose is constantly updated by GPS.

But as proposed previously, the FAUVs will not update their pose constantly, but just once in a while. So what happens to the error in this circumstances?

In figure 8 we show the error on the x axis for the CAUV in an simulation of Alg2 with 3 FAUVs. The observation time is 5 seconds. In this experiment, one FAUV is randomly selected to update its pose by GPS every 100 seconds. This means that each FAUV can stay about 5 minutes underwater, before having to surface in order to update its pose, and the CAUV, of course, can be submerged during the whole experiment. You can notice that once again the error is not increasing during the simulation. It is higher than in figure 7, as expected, but it is still bounded. Whenever a FAUV that has just updated his position is observed, the error in the CAUV pose decreases dastrically. After this step, the CAUV now has a better pose estimate, and can "propagate" this better estimate through the others FAUVs, helping them to better estimate their pose. This result is valid for all four algorithms presented.



(a)



(b)

Figure 8. Absolute error on x axis for the CAUV (a) and for FAUV1 (b) in an execution of Alg2 with 3 FAUVs, observation time of 5 seconds, and the region of confidence ($3\sigma_{xx}$) calculated based on the covariance matrix $P(t)$. One FAUV randomly updates its pose by GPS every 100 seconds.

VI. Conclusion and Future Work

Cooperative localization for AUVs is hard because there are communication constrains in the underwater environment. We proposed 2 algorithms to solve the localization problem with a group of AUVs, all with similar low cost navigation systems. We showed that the algorithms can greatly reduce the mean error of the estimate position, although the error is not bounded.

In future work we will use more realistic models. Since this models are highly nonlinear, we can use filtering techniques that gives better results for nonlinear systems, like the

Unscented Kalman Filter or Particle Filter. There is also a need to study how the SLAM algorithms convergence behaves when the map features are non-stationary and the relationship between GPS position update and the boundedness of the error of the algorithms.

REFERENCES

[1] J. Vaganay, J. Leonard, J. Curcio, and J. Willcox, "Experimental validation of the moving long base-line navigation concept," in *Autonomous Underwater Vehicles, 2004 IEEE/OES*, 2004, pp. 59–65.

[2] P. Maybeck and A. Press, *Stochastic models, estimation and control. Volume I.*, 1979.

[3] J. D. Athanasios Kehagias and S. Singh, "Range-only slam with interpolated range data," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-06-26, May 2006.

[4] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4, 1987, pp. 850, 850.

[5] H. Durrant-Whyte, "Uncertain geometry in robotics," in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4, 1987, pp. 851–856.

[6] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 2, pp. 99–110, 2006.

[7] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 3, pp. 108–117, 2006.

[8] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 3, pp. 229–241, 2001.

[9] P. M. Newman, "EKF based navigation and SLAM," Jul. 2004, SLAM Summer School 2004, Toulouse.

[10] J. Neira and J. Tardos, "Data association in stochastic mapping using the joint compatibility test," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 6, pp. 890–897, 2001.

[11] J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *Robotics and Automation, IEEE Transactions on*, vol. 7, no. 3, pp. 376–382, 1991.

[12] P. Newman and J. Leonard, "Pure range-only sub-sea SLAM," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, 2003, pp. 1921–1926 vol.2.

[13] E. Olson, J. J. Leonard, and S. Teller, "Robust Range-Only beacon localization," *Oceanic Engineering, IEEE Journal of*, vol. 31, no. 4, pp. 949–958, 2006.

[14] A. Bahr and J. J. Leonard, "Cooperative localization for autonomous underwater vehicles," in *ISER*, 2006, pp. 387–395.

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

[16] R. Kurazume, S. Hirose, S. Nagata, and N. Sashida, "Study on cooperative positioning system (basic principle and measurement experiment)," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2, 1996, pp. 1421–1426 vol.2.

[17] C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas," in *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 1, 2003, pp. 842–849 vol.1.

[18] R. Sharma and C. Taylor, "Cooperative navigation of MAVs in GPS denied areas," in *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, 2008, pp. 481–486.

[19] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, "Particle filters for mobile robot localization," 2001.

[20] A. Bahr and J. Leonard, "Minimizing trilateration errors in the presence of uncertain landmark positions," in *Proc. 3rd European Conference on Mobile Robots (ECMR)*, 2007, pp. 48–53.

[21] K. W. Lee, W. Wijesoma, and I. Javier, "On the observability and observability analysis of SLAM," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2006, pp. 3569–3574.