International Journal of Advanced Robotic Systems

OPEN ACCESS ARTICLE

Swarm Underwater Acoustic 3D Localization: Kalman vs Monte Carlo

Regular Paper

Sergio Taraglio[1]* and Fabio Fratichini[1,2]

1 ENEA Centro Ricerche Casaccia, Rome, Italy
2 Department of Electronic Engineering, University of Tor Vergata, Rome, Italy
*Corresponding author(s) E-mail: taraglio@enea.it

Received 12 November 2014; Accepted 02 April 2015

DOI: 10.5772/61034

Abstract

Two three-dimensional localization algorithms for a swarm of underwater vehicles are presented. The first is grounded on an extended Kalman filter (EKF) scheme used to fuse some proprioceptive data such as the vessel's speed and some exteroceptive measurements such as the time of flight (TOF) sonar distance of the companion vessels. The second is a Monte Carlo particle filter localization processing the same sensory data suite. The results of several simulations using the two approaches are presented, with comparison. The case of a supporting surface vessel is also considered. An analysis of the robustness of the two approaches against some system parameters is given.

Keywords Distributed Control Systems, Mobile Robots and Intelligent Autonomous Systems, Underwater Robots, Sensor Fusion, Swarm Localization

1. Introduction

The exploration of the oceans, both for scientific and economic purposes, is becoming increasingly important. Several application fields can here considered here as examples of the importance of sea study: the search for underwater energy sources or raw materials; oil pipe or underwater cable monitoring; scientific missions collecting biological, physical and chemical data; and sea monitoring against different pollution sources (see e.g., [1]). Because of the limitations of our biological characteristics, underwater robotics have gained an essential role in the study and exploitation of the seas. One of its more promising branches is that of autonomous underwater vehicles (AUV), i.e., those vehicles that are capable of performing the required tasks without human supervision, coping with missions unknown.

In recent years, research on AUVs has broadened towards the simultaneous use of more vessels, i.e., the implementation of multi-robot configurations all the way to full swarms of underwater vehicles, see e.g. [2, 3].

Whether one or more AUVs are considered, one of the focal points of autonomy is the reliable knowledge of the vessel pose. This knowledge is not easily gained, especially when considering several vessels at the same time. Unfortunately an underwater system suffers the limiting characteristics of its environment. Water, especially saltwater, blocks electromagnetic waves, inhibiting the use of positioning systems such as GPS. At the same time this implies difficult communication between the AUVs of a swarm or with a remote operator. The available means to localize a single AUV are thus the exploitation of inertial sensors, velocity sensors, and/or gyroscopes combined in dead reckoning.

However, in the framework of an underwater swarm, the localization of a single vehicle can profit from the existence of the other vessels, gathering information from fellow AUVs. The key issue for all swarm localization methods is the best possible combination of proprioceptive measures (dead reckoning) and exteroceptive sensor readings, the main difference being the employed estimator.

The localization of swarms of robots has been extensively studied for packs of terrestrial surface robots. The problems linked to communication of information in this environment are less demanding with respect to the underwater robots. The possibility of exploiting a GPS localization system, naturally considering all related problems (tall buildings, overcast sky), greatly helps in the task of correctly localizing the robots.

In this work two different approaches for robot localization in a swarm of underwater vehicles are compared: one based on Kalman filtering and the second on a Monte Carlo scheme. The focus is toward the localization of members of a not-too-diluted swarm. Since the means of communication in the swarm is represented by ultrasound links, the swarm's typical dimension should allow the successful transmission of data. Therefore, the overall dimension of the robot group should not grow too large. The typical diameter of the considered swarm is less than 750 *m*, i.e. within 0.25 *s* of flight time for an acoustic ping.

The exteroceptive measure considered here is the distance among the vessels, as measured by the time of flight (TOF) of the ultrasound transmission. An initial synchronization of the vessel clocks is assumed, e.g. with NTP (network time protocol) [4], performed just before the mission at swarm deployment. Each robot, whenever communicating with another (or broadcasting to all), transmits its own location and time, acting as a localization beacon for the listener or listeners.

The main drawback of a Kalman approach in underwater swarm localization is represented by the large quantity of data that must be circulated among the vessels: all the relevant matrices of the algorithm, which, in addition, grow non-linearly with the number of considered robots. Since ultrasound links are generally very limited in bandwidth (a few tens of *kbps*) this can become a very limiting feature. The Monte Carlo localization (MCL) paradigm allows a drastic reduction in the data traffic among the vessels. In MCL it is sufficient for the robots to communicate only their estimated position and time of transmission.

A comparison of the bandwidth needs of the two approaches is provided in section 5, where the dependency among bandwidth, update frequencies and transmission speed is described under the assumption of the use of a transmission protocol such as the Slotted Aloha one [5]. A further aspect of an ultrasound-based localization system is related to the speed of the communication in the water. The limitedness of this velocity affects the two approaches in different ways that are described in section 6.

Some experimental simulations are presented in order to compare the two methods; an analysis of their robustness and their dependence on some algorithm parameters is also presented.

In the simulations a simple kinematic model of the single AUV is considered, capable of measuring its own velocity and attitude and of communicating over an ultrasonic acoustic link with the other vessels. Through the measurement of the time of flight (TOF) of the ultrasound transmission the AUVs can measure their relative distance. All the available information, proprioceptive and exteroceptive, is then combined with either of the two above-mentioned methods in order to compute all vessel poses.

Since the initial poses of all the swarm elements are known with a given initial pose error, it is possible to substitute the term localization with pose tracking throughout the paper.

Considering that an operative swarm of underwater robots will be deployed from some kind of surface vessel, the case of a swarm-supporting boat able to measure its own position via GPS, and able to furnish this data to the swarm, is described and simulated.

In the second section of the paper the related work is presented. In section 3 the multi-robot Kalman-based algorithm is revised. In the fourth section the principles of Monte Carlo localization are presented. In the fifth section data throughput considerations are given, while in the sixth section the influence of non-instantaneous communication on the two approaches is described. In sections 7 three experimental cases are described and simulated, and results given: linear, circular and sinusoidal cases are considered. In section 8 an analysis of the dependence of the approaches on some algorithm parameters is presented. Finally, in the last section conclusions and future work are discussed.

2. Related Work

Localization in the underwater realm is mainly based on ultrasound waves. Classical methods of underwater positioning are Long Baseline (LBL) and Short Baseline (SBL) systems [6]. These exploit a given number of floating buoys with known positions, see also [7], but this means to structure in some way the environment, an option that may be impossible in some operative scenarios.

The localization of a swarm of terrestrial robots has been extensively studied. An approach is based on the subdivision of the swarm in subgroups one of which, in turn, is kept at a fixed position and acts as a set of landmarks for moving others [8, 9, 10]. In [11] and [12] belief functions combined with a Monte Carlo approach and particle filtering optimization have been successfully employed; [13] and [14] employ a Kalman filter where the proprioceptive measures are used to estimate the future state of the system and the exteroceptive ones are used to correct and update the estimate. In [15] this approach is extended by considering the most generic relative observations between two robots. More recently the work of Olfati-Saber, see e.g. [16], has addressed the problem of decentralized Kalman filtering in sensor networks through consensus algorithms for swarm controlling strategies. In [17] the consistency of EKF-based cooperative localization considering observability has been investigated. The typical operative environment considered in these works is a two-dimensional terrestrial one. The MCL ([18]) has been employed in several robotic applications, see e.g. [19], and also in underwater environments for the localization of mobile sensors [20]. It basically behaves by creating a cloud of particles, each representing a possible state of the variable under measurement and a weighting process to measure the fitness of each particle to the actual measure.

3. Kalman Localization

The key point, in the Kalman approach, for cooperative localization in a swarm of robots is viewing the group as a single entity that can access the information of a large number of proprioceptive and exteroceptive sensors.

In the following, all the vessels are described by the same motion equations, and each robot possesses proprioceptive sensors for the motion estimate. Each AUV also encompasses an ultrasound communication link, which can collect the information from the other vehicles in the swarm.

It is important to stress that each time there exists communication between two swarm members, there is also an exteroceptive distance measure. The broadcasted message contains the timestamp relative to its emission, and any receiving robot can therefore compute its distance from the emitting robot position.

Let us consider the global dynamical state X of the whole swarm, it will be a vector composed of Mx6 items where M is the number of robots and 6 is the number of variables describing the single vehicle (x, y, z, roll, pitch, yaw), i.e., a vector composed of the poses $\vec{x}_i$ of all the robots.

$$\vec{X} = \left(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_M\right) \tag{1}$$

The vessel coordinate system is based in the centre of gravity of the vehicle, and its x-axis is longitudinally directed from stern to bow; the y-axis is towards starboard and the z-axis is downward.

The kinematic model of the single robot uses a linear velocity parallel to the x-axis (thrust) and the possibility of changing all three Euler angles through appropriate angular velocities.

Let us describe first the Kalman approach for a single vessel of the swarm. The mathematical model describing the time evolution of the single swarm robot is:

$$\vec{x}_i(t) = f\left(\vec{x}_i(t-1),\quad \vec{u}_i(t-1),\quad \vec{w}_i(t-1)\right)\ i = 1,\ldots,M \tag{2}$$

where f is generally a non-linear function of the state $\vec{x}_i(t-1)$, of the input $\vec{u}_i(t-1)$ and of the noise $\vec{w}_i(t-1)$ at the preceding time step *t-1*. Each vessel can also measure all the other ones, and this can be described by:

$$z_{i(t)} = h\left(\vec{x}_i(t),\quad \vec{x}_j(t),\quad n_i(t)\right)\ \ i = 1,\ldots,M;\ j \neq i \tag{3}$$

here h is the measurement function linking the state of the i-th robot with the state of the measured one (the j-th) and the measure noise $n_i(t)$ at time t.

Kalman filtering is a well-known strategy that yields an estimate of a dynamical process using feedback control. It foresees the process state at a given time and employs a measurement feedback to update the state through a better estimate. It is an iterative process that loops through two different phases: on one side it predicts the state of the system and the error covariance; on the other it computes the so-called Kalman gain to correct both the state estimate and the error covariance on the grounds of some kind of available measure. Since the time evolution function (equation 2) may be not linear, an extended version of the filter has been employed. The EKF basically behaves as the standard procedure, but uses a local linearization of the functions. A very interesting characteristic of this filter is its iterative aspect. The result of an iteration of the filter is used as input for the successive step; in this way the filter retains the memory of the history of the system.

Let us consider the whole swarm and examine in detail the two phases of prediction and update.

3.1 Prediction

Each robot, at any given time step, estimates its state at the successive time step on the grounds of the kinematic model and the available proprioceptive measures (linear and angular velocities) and their null average Gaussian noise. The time evolution of the cross-correlation matrix is also computed:

$$\hat{x}_i^-(t+1) = f\left(\hat{x}_i^+(t), u_i(t), w_i(t)\right) \tag{4}$$

$$\hat{P}_{ii}^-(t+1) = \Phi_i(t)\hat{P}_{ii}^+(t)\Phi_i^T(t) + G_i(t)W_i(t)G_i^T(t) \tag{5}$$

$$\hat{P}_{ij}^-(t+1) = \Phi_i(t)\hat{P}_{ij}^+(t)\Phi_j^T(t) \tag{6}$$

here equation (4) is the state time evolution (as in equation (2)) and equations (5) and (6) describe the time evolution of the covariance matrix P in the diagonal (equation (5)) and off diagonal cross-correlation terms (equation (6)); Φ is the system propagation matrix, G is the system noise input matrix and W is the noise input covariance. The minus sign stands for *a-priori*, and the plus sign for *a-posteriori*. In other words, the state is predicted on the basis of the kinematic model and the input velocity and the new *a-priori* covariance matrix is computed with the system proprioceptive measures for the diagonal terms (the *6N* system variables), and with the system propagation matrix for the off diagonal terms.

In order to perform a distributed EKF it is convenient to process the *a-posteriori* estimated cross-correlation matrix (equation 6) through a singular value decomposition (SVD). Operating in such a way, each robot can compute its own contribution multiplying the SVD term by its dynamical matrix, see e.g. [15].

3.2 Update

Every time a robot measures something, an update can be performed. In order to describe the procedure the availability of the heading (compass) and depth (pressure gauge), roll and pitch angles (inclinometers) of the measuring vessel, and the acoustic TOF distance of another vessel are here considered. The non-linear measuring function h is that shown in equation (3) and the noise is considered as a null average Gaussian one. It is now possible to compute the *a-posteriori* state estimate (i.e., after the measurement):

$$\begin{aligned} \hat{x}_r^+(t) &= \hat{x}_r^-(t) + K_r(t)(z_i(t) - h(\hat{x}_i^-, \hat{x}_j^-)) \\ K_r(t) &= \hat{P}_{ri}^-(t)H_i^T(t) + \hat{P}_{rj}^-(t)H_j^T(t)S^{-1} \end{aligned} \tag{7}$$

where the index $r = 1, ..., M$ describes the vessel, $K_r(k)$ is the so-called Kalman gain with S the residual covariance, and the last term is the residual. The H terms are the Jacobians of the measuring function h with respect to the two state vectors x_i and x_j :

$$H_i = \begin{bmatrix} -\frac{\Delta x}{\sqrt{\Delta r^2}} & -\frac{\Delta y}{\sqrt{\Delta r^2}} & -\frac{\Delta z}{\sqrt{\Delta r^2}} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

$$H_j = \begin{bmatrix} \frac{\Delta x}{\sqrt{\Delta r^2}} & \frac{\Delta y}{\sqrt{\Delta r^2}} & \frac{\Delta z}{\sqrt{\Delta r^2}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{9}$$

and $\Delta r^2 = \Delta x^2 + \Delta y^2 + \Delta z^2$. Clearly equation (8) describes the distance measurement together with the proprioceptive absolute measure of z, roll, pitch and yaw, while equation (9) is relative to the other vessel and the distance measure alone.

Finally the *a-posteriori* covariance matrix estimate is:

$$\hat{P}_{rs}^+ = \hat{P}_{rs}^- - \left(\hat{P}_{ri}^- H_i^T + \hat{P}_{rj}^- H_j^T\right) S^{-1} \left(H_i \hat{P}_{is}^- + H_j \hat{P}_{js}^-\right) \tag{10}$$

with the indexes $r = 1, ..., M$ and $s = 1, ..., M$.

Until now the described process is relative to a single robot; the localization of multiple robots considers a series of multiple relative measurements among the various vessels on the state vector of equation (1).

Such a scenario can be approached in two different ways: in a centralized or in a distributed one. In the first a central supervisor can be considered collecting all the data from all the vehicles and performing the multi-robot system state estimation. The second paradigm can be split into two further classes: uncooperative or cooperative algorithms. The first class simply tries to localize each robot as if it was alone in the world, i.e., counting on its own estimate and measures alone, without gathering additional information from the others in the swarm. The second class can exploit the information coming from the companions and performing a local algorithm for the pose estimation. Regardless of the approach employed, the Kalman scheme makes use of a series of matrices and vectors that must circulate among the robots.

As stated, in an underwater environment an extremely limited bandwidthis is available, and in the following a distributed cooperative approach has been employed. Let us now examine the scheme used in more detail.

In this Kalman scheme, whenever a robot measures its own distance from another, it should broadcast a message with its own ID, that of the measured robot, its state vector, and the vector of measurement; for the update phase an exchange of the predictions of all of the swarm should be distributed between the members. Each robot may then compute the update of its own state and covariance matrix, which should be transmitted back to any other robot in the swarm. In a fully distributed algorithm this would mean an asynchronous and uncontrollable pattern of communication among the robots. Taking into account transmission latencies, communication delays and protocol strategies (who transmits and when), the system would easily saturate the available bandwidth.

In order to limit the communication problem in the swarm as much as possible, a distributed-centralized approach has been tested. The idea is as follows. Whenever a measurement is made, the measuring robot broadcasts its ID and a *measurement done* flag. It then gathers predictions from all the companions, performs the EKF for the whole swarm, and finally broadcasts the new matrices to the fellow robots. It is obviously a centralized algorithm but it can be considered distributed in time: at each time step the measuring robot is the only one computing, and at the next step it will probably be a different one.

The difference with the fully distributed approach is in the limitation of the data exchange due to an ordered communication strategy that can reduce the transmission overheads. The amount of exchanged data will be the same, but since only one robot is actually broadcasting its results and collecting data, there will be fewer communication overheads, collisions and possible multi-paths deriving from multiple robots trying to communicate all at the same time.

The difference with a fully centralized approach with a fixed location for the algorithm computation is in a slightly smaller amount of data circulation: in a fully centralized approach the measuring vessel should communicate its share of data to the computing node.

The amount of exchanged data naturally increases if more than a single robot performs exteroceptive measures. In the following it has been supposed that at a given time step each vessel measures the distance of all the others, i.e., it performs *M-1* exteroceptive measures. This scenario is reasonable since the distance measure can be performed exploiting the communication itself (that contains the timestamp).

The algorithm may consider this scenario in two possible ways: on one side it may compute the EKF considering one measure at a time, and updating the state *M-1* times; on the other it may consider a single measurement (a vector of *M-1* components) and update the state a single time only. Naturally the second approach would certainly be less computationally intensive.

A set of dedicated experiments has shown, as expected, that the results of the EKF in terms of position accuracy does not depend on the chosen method, and thus the second one has been chosen.

4. Monte Carlo localization

In the Monte Carlo approach the idea is as follows. There is a variable of interest (e.g., the underwater robot position) that is represented at a given time by a set of *N* samples (the particles) and a weight function that measures the contribution of each particle to the overall estimate of the variable.

In this work the MCL approach estimates the robot pose, i.e. position and orientation of each single robot considered separately:

$$\vec{x}_k = \left(x_k, y_k, z_k, \psi_k, \theta_k, \varphi_k\right) \tag{11}$$

with *k* the index of the single robot.

Also in this method there are two distinct phases: prediction and update. The first is basically identical to that of the Kalman case: through the use of the robot model, each particle representing a given robot is made to evolve, adding the appropriate noise to the process. In the second, the information coming from a direct or indirect measure of the considered variable is used in order to update the particle weights for a better description of the variable itself through a weight-based particle resampling.

As soon as a measure performed by a given robot is available, the update phase begins. Each of the *N* particles representing the robot pose is weighted with its suitability with the available measure. There are, naturally, several different ways to estimate this suitability. In the following it has been used:

$$W(i) = \frac{1}{\sqrt{2\pi}\sigma_\rho} e^{\frac{-(\rho-\rho_i)^2}{2\sigma_\rho^2}} \tag{12}$$

where ρ_i is the measure relative to the *i*-th particle (with *i* = *1,...,N*) and ρ the actual available measurement, σ_ρ is the standard deviation of the measurement. The quantities computed with equation (12), once normalized, represent the *a-posteriori* distribution function for the particles, on whose grounds the particles are subsequently resampled, creating a new set for the following time step. The new set composition will thus be poor in particles with lower values for *W(i)* and rich in particles with larger weights, thus being in greater accordance with the measurement, nevertheless leaving open different possibilities but with a much lower probability.

In the resampling process it may sometimes happen that the algorithm may become too aggressive, discarding many of the good particles, getting caught in some kind of local minimum and yielding a poor estimate and a large error. In order to limit such adverse behaviour, some additional randomness is usually injected into the particle set under the form of a fraction of the resampled particles, assuming random values within a given distribution. The decision for such an intervention may be performed, for example, monitoring the long and short-term average values of the weights in equation (12) [20]. This monitoring estimates the algorithm ability in following the statistical distribution describing the problem being considered.

Let us now consider a swarm of *M* robots. Each robot pose will be described by a set of *N* samples, and each will predict its next position using its own samples. Whenever a distance measure is available, the robot that has performed the measurement will update its samples according to equation (12) in a fully local way, regardless of the fellow robots. Obviously it will rely on the believed position of the measured robot. This approach thus considers *M* different and independent Monte Carlo localizations with a state vector as in equation (11), one for each of the *M* robots. This is different to the Kalman approach in which there is a global state for the whole swarm (equation 1).

It is evident that in the MCL case the data circulation is more limited; this amount of information will still increase with the number of robots in the swarm, but at a smaller pace with respect to the Kalman one since there are no matrices to circulate. Whenever there exists the transmission of position and time, a robot may measure a fellow one.

5. Data throughput considerations

Let us assume, for the sake of clarity, a swarm composed of three robots only. Figure 1 compares the data flow among the robots in the two algorithms in order to complete a localization action for all the elements in the swarm.

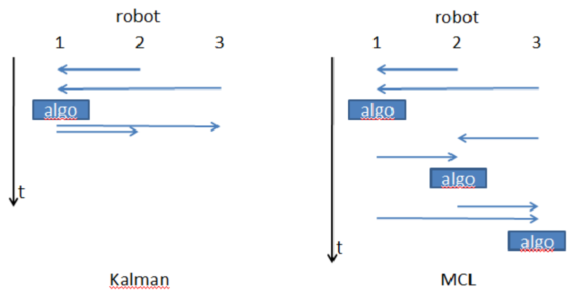

Figure 1. Data circulation among the robots of a swarm of three members to complete a localization for all of the vessels. The image on the left shows the Kalman-based approach; on the right is the MCL approach. *t* is the time.

5.1 Kalman-based Data Flow

In the Kalman-based scenario (left of Figure 1) each of robots 2 and 3 should communicate to robot number 1 its state and ID (six real numbers + 1 byte) and covariance matrix (36 real numbers), plus the cross-correlation matrices between itself and robot 1 (36 + 36 real numbers). In the message should also be the timestamp (three real numbers: minutes, seconds and milliseconds). This is depicted by the topmost two arrows on the left of Figure 1. After communication, robot 1 may compute the update of the whole swarm and then broadcast back to robot 2 and 3 the computed swarm state (3 x 6 real numbers) and the full cross-correlation matrix (36 x 3 x 3 real numbers). This is represented in Figure 1 by the lowermost arrows on the left side.

Generalizing to a swarm composed of M robots, the full amount of data (prediction and update) to be communicated among the robots is: $432M^2 - 368M + 104$ bytes, assuming a real number to be represented in single precision (4 bytes).

These data are those needed to perform a full update of all the vessels in the swarm, with a single robot measuring all others.

5.2 MCL-based Data Flow

In the MCL-based scenario (right of Figure 1) each of robots 2 and 3 communicate to robot 1 their position with covariance (attitude is not needed), timestamp and ID (9 real numbers + 1 byte). At this moment robot 1 may update its own pose according to the incoming data. Subsequently robot 1 and 3 will transmit the same quantities to robot 2, which will in turn update its own pose. Finally robot 1 and 2 will transmit to robot 3 in order to let it compute its update. Thus, to update the complete swarm locations there will be data traffic of 9x6 real numbers plus 6 bytes (robot ID).

Generalizing to an M elements swarm, the data will add up to $37M(M-1)$ bytes.

This amount of data is that needed to enable the entirety of the robots in the swarm to perform an update, under the scenario of an all-to-all communication as in the Kalman approach, for the sake of comparison. In fact the MCL algorithm is fully decentralized and asynchronous: as soon as a robot receives data from another, even a single one, it may update its own state on the grounds of the incoming data.

5.3 Kalman vs MCL Update Frequency and Throughput

A typical underwater ultrasound link allows a bandwidth of about 30 *kbps*, see e.g., [23]. If, for example, a Slotted Aloha protocol is used for the transmission system, the actual bandwidth will be reduced to 36% of that available in the best transmission conditions [5]. With the above computed data flow needs and this limit, it is possible to calculate two quantities: the update frequency allowed and the data throughput.

The first is how often the swarm members may receive an update on their position by the localization algorithm, while the second shows the exchanged amount of data in the communication network; both are a function of the swarm elements number and the communication channel capability. The two are obviously dependent, and one can be deduced from the other.

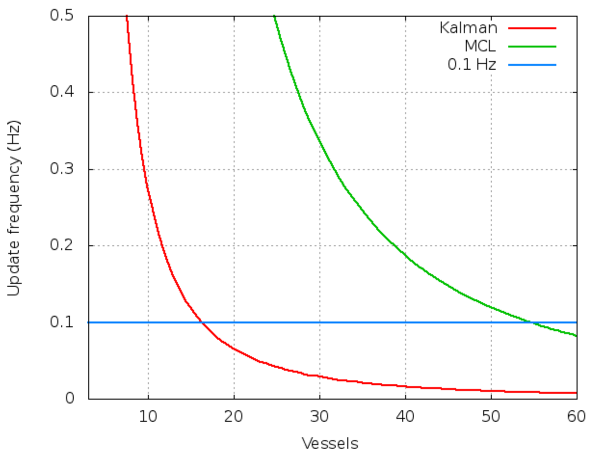

Figure 2. Maximum allowed update frequency for the two approaches as a function of swarm elements number. The blue line shows a viable update frequency of 0.1 *Hz*.

Figure 2 shows the maximum update frequencies of the two approaches as a function of the number of vessels in the swarm. The maximum update frequency represents the update frequency saturating the communication channel capacity. Naturally, as the number of robots increases the higher bandwidth need lowers the frequency. Let us assume, as an example, an update frequency of 0.1 *Hz*, i.e., one full swarm position update every 10 *s*. Using the graph in Figure 2, it is possible to consider a Kalman swarm of up to 16 elements and an MCL one of less than 54.

Figure 3 shows the transmission needs of the two approaches as a function of the swarm elements number. This plot is drawn at the above considered update frequency of 0.1 *Hz*.

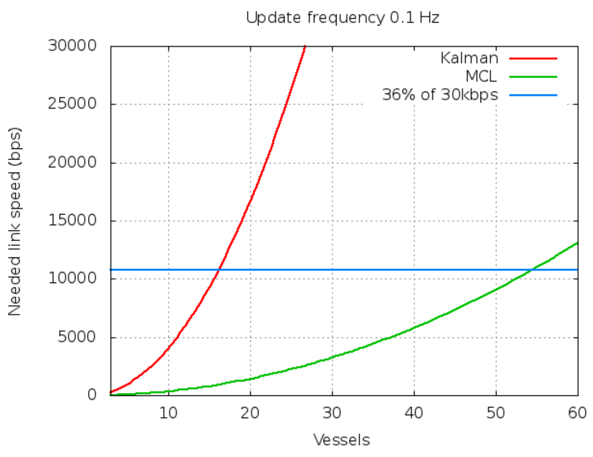

Figure 3. Needed link speed as a function of vessel number at an update frequency of 0.1 *Hz*. The blue line shows the available link speed (30*kbps*) as reduced by the Slotted Aloha protocol.

The plot also shows the considered limit with the use of the Slotted Aloha protocol. As above, the bandwidth saturation will happen in the Kalman case with 16 robots, while the MCL approach can withstand more than 50 vessels communicating.

In addition to the above considerations, it must be taken into account that the communication link cannot be fully dedicated to swarm localization; it should also carry data relative to the mission tasks and other relevant swarm activities. Therefore, the above limits on robot numbers must be considered indicative.

Underwater a different means of communication may also be used: the optical one, based on high power light emitting diodes, see e.g. [24]. With such technology it is possible to attain higher throughputs, even if this comes at the cost of a much narrower range than the ultrasound link. Taking as an example a conservative 200*kbps* LED-based optical link, the maximum number of robots may easily rise to 40 for the Kalman case and to more than 135 for the MCL one, still at 0.1 *Hz*.

In the following section, in order to compare the two approaches an update frequency of 0.1 *Hz*, an ultrasound link at 30 *kbps* and a swarm composed of ten robots are considered.

6. Communication Velocity Considerations

A further aspect should be considered: the speed of sound in the water is limited, being about 1500 *m*/*s*; in other words it takes time to transmit data using an ultrasound link. This affects the considered localization approaches in different ways.

6.1 Kalman Approach

In the Kalman-based approach all the robots need to communicate their prediction to the computing node. This in turn must then broadcast back the update of the whole swarm. This ping-pong implies synchronization, a difficult task with a finite transmission velocity as it is in the underwater case.

Let us examine in some detail the communication characteristics of the prediction phase of the algorithm, the broadcast one being basically similar. For the sake of simplicity let us consider two robots: number 1 transmits its prediction to number 2. Figure 4 shows the evolution of this communication phase with the aid of a space-time diagram of a mono-dimensional space; robot 1 is in a fixed position, while robot 2 is moving. At the beginning (*t*=0), robot 1 transmits its prediction with velocity v_s, when the data reach robot 2 (upper right) some time has passed and robot 1 is in position 1a and robot 2 is in 2a.

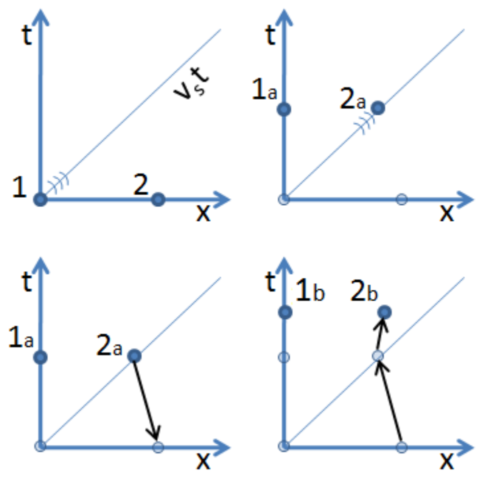

Figure 4. Space-time diagram of the transmission of prediction between two robots. Upper left: robot 1 starts transmission to 2 with sound speed v_s. Upper right: robot 2 receives data, robot 1 is in 1a and robot 2 in 2a. Lower left: robot 2 backtracks itself as if being still in the initial position to update. Lower right: robot 2 with two prediction phases on the grounds of the updated initial position (2) reaches position 2b, robot 1 is in 1b.

Robot 2 should now update the state of the swarm, but with respect to its initial position 2, so it must backtrack its trajectory to the initial position (lower left diagram) and update as if it were still in that location. Once updated some additional time has passed, and robot 2 should evolve its corrected position with a series of prediction phases to 2a and then to 2b. This schema can be implemented while keeping a memory of the past robot course.

A similar scheme applies for the broadcasting phase of the updated state by robot 2 towards 1.

6.2 MCL Approach

The MCL localization algorithm, on the contrary, does not suffer the transmission lag. As stated above it is completely decentralized and asynchronous: as soon as data are received from a fellow robot, the receiver may compute an update of its own state. The received data is composed of the position of the sending robot at the sending time and the sending time itself. The receiving robot, through the TOF, may thus compute the distance from the received position and perform the update. Naturally, in the meantime the sending robot will have moved from the transmitted location, but this does not impair the algorithm, also because the updating robot should not transmit anything back.

7. Simulation characteristics and results

The underwater vehicle considered has proprioceptive and exteroceptive sensors in order to measure itself and its environment. It has been assumed that each vessel is equipped with:

- a tachymeter, for the measurement of the linear velocity due to the thrust of the propeller;
- an inclinometer (or gyro), for the measurement of the pitch and roll angle;
- a pressure gauge, for the depth measurement (*z* absolute coordinate);
- a compass, for the heading measurement (yaw absolute coordinate);
- an ultrasound transducer, for the measure of relative distance of two vessels with time of flight (TOF) and communication.

The introduction of a compass and gyro for the absolute orientation and a pressure gauge for the depth measure are of basic importance since they improve the observability of the system. The measurement noise is assumed Gaussian with a zero mean.

In the result section the considered noise values for the measured data are compatible with low-cost sensors similar to those planned to actually equip a low-cost submarine presently under development.

In the simulations, all the vehicles are considered to be kinematic objects, i.e. without the computation of their dynamics. The vehicles are supposed to exchange information instantaneously, i.e. it is implicitly assumed that the mechanism considered in section 6 can be ignored at least at a first level of approximation, considering the swarm as dense (i.e., less than 0.25 *s* of TOF, that is a swarm radius of circa 750*m*).

All the simulations have been performed in Matlab and a porting in C++ is in progress. The time step is of 0.1 *s*.

The result section is organized by scenario. The missions are stated with the details of the relative parameters and the two approaches are then compared. The reported simulations are: a linear trajectory, a circular one and a sinusoidal one. The last, in the case of a Kalman approach, is also computed with and without the aid of an external ground truth point, i.e. a boat being an element of the swarm, but with the additional possibility of exploiting a GPS system.

Both the Kalman and the MCL localizations share the same parameters in order to allow a meaningful comparison.

The robots move at a speed of 1 *m/s*. The assumed standard deviation on linear and angular speed is 0.1 *m/s* and 0.05 *rad/s* respectively. The standard deviation on the TOF distance measure is 0.3 *m*, on the heading is 2°, on roll and pitch 1° and 0.07 *m* on depth. The initial positions of the swarm elements are uniformly random and known with a standard deviation of 0.05 *m*.

The update frequency for the localization is 0.1 *Hz*, i.e. one full swarm update every 10*s* (following the considerations in section 5). Since for the simulations the used time step is of 0.1 *s*, this implies one update every 100 steps.

The measurement error on the velocity at each time step is considered within 20% of the current value for the modulus and randomly within a cone of 30° of aperture for the direction, both in Kalman and in MCL.

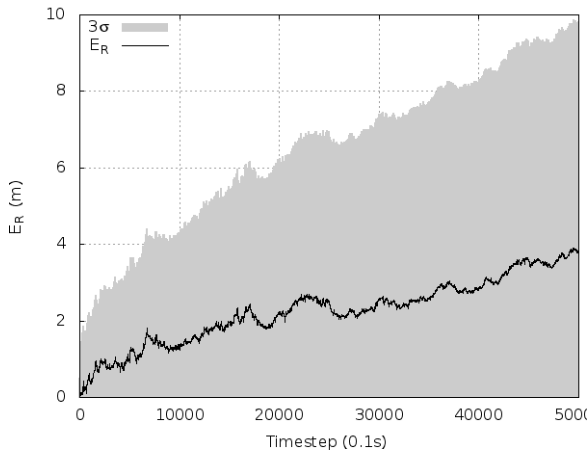

Figure 5. Linear trajectory: average position error (on the ten robots) and 3σ error vs. time (Kalman), average on ten runs

For each method and scenario the simulations have been repeated ten times, and the plots reported here are relative to the averaged results.

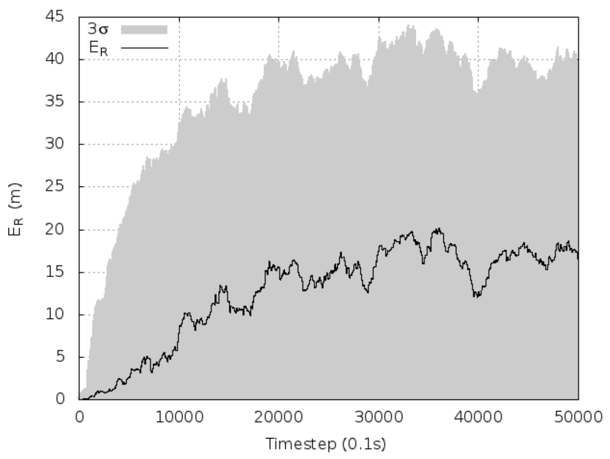

Figure 6. Linear trajectory: same as Figure 5 in the MCL case, average on ten runs

Finally, the Monte Carlo localization is performed using a set of 1000 particles for each vessel.

7.1 Linear Trajectories

In this simulation the ten robots composing the swarm are made to follow linear trajectories parallel to one another.

7.1.1 Kalman Results

Figure 5 shows the average position error (on the whole swarm) and the 3σ error. The position error of the single robot is computed as the distance between the true vessel position as output by the simulator (subscript *s*) and the localization algorithm computed one (subscript *l*):

$$E_R = \sqrt{\left(x_s - x_l\right)^2 + \left(y_s - y_l\right)^2 + \left(z_s - z_l\right)^2} \tag{13}$$

The variance of this quantity is directly computed on the grounds of the variances output by the Kalman filter.

After 5000 seconds of mission, i.e., 5000 metres, the relative average position error is of about 0.1% and the 3σ error is of 0.2%, i.e. the ratio of position error vs. the travelled distance. The same quantities considering dead reckoning alone have been of the order of 8% and 30%, with the same variances on speed and initial position.

7.1.2 MCL Results

In Figure 6 the output of the MCL localization algorithm is presented. In this case the algorithm computed position is the weighted average of the positions of all particles.

The relative 3σ error is computed on the basis of the weighted variance of each of the *N* particles:

$$\sigma_R = \sqrt{\frac{\sum_i w_i\left(r_i - \bar{r}\right)^2}{\sum_i w_i}} \tag{14}$$

where r_i is the position of the *i*-th particle, w_i its weight (equation 12) and $\bar{r}$ the average position. As is evident from Figure 6 the MCL approach yields a much larger average error in the position of the vessels, of about 15 *m*. The percentage values are thus 0.4% and 0.2% for average position error and its 3σ error, respectively.

In Figure 7 a blow-up of the leftmost part of the plot in Figure 5 is shown, to show the saw tooth shape of the plot due to the periodical correction of the position by the algorithm, once every 10 *s*, i.e., once every 100 time steps.

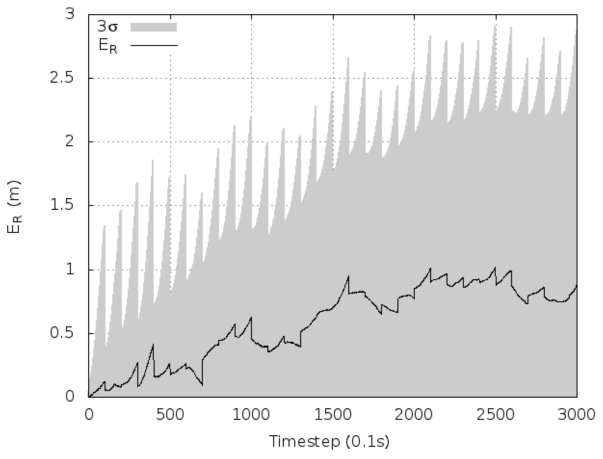

Figure 7. Average and 3σ errors as in Figure 4 zoomed to show the periodical update at 0.1 *Hz*

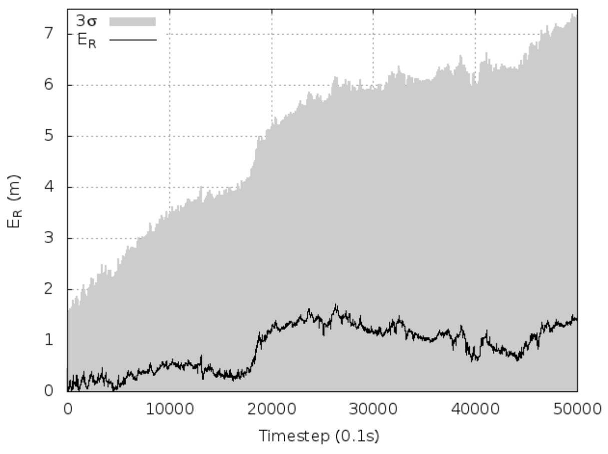

Figure 8. Circular trajectory: average position error (on the ten robots) and 3σ error vs. time (Kalman), average on ten runs

7.2 Circular Trajectories

In this simulation the ten robots composing the swarm are made to follow circular trajectories in a plane. The robots' angular speed is 0.005 *rad/s* in yaw.

7.2.1 Kalman results

In Figure 8 the average position error as a function of time in the Kalman case is shown.

After 5000 *m* of trajectory, the circles have been coursed almost four times and the final relative average position

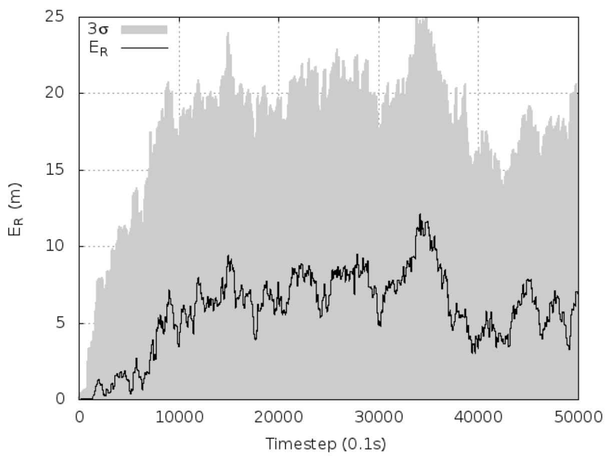

Figure 9. Circular trajectory: same as Figure 8 in the MCL case, average on ten runs

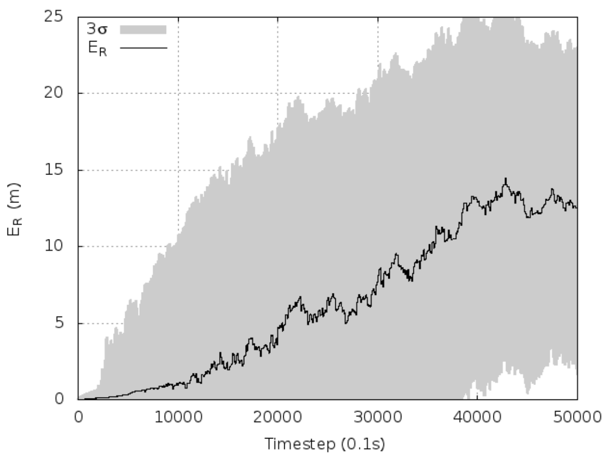

Figure 11. Sinusoidal trajectory: same as Figure 10 in the MCL case, average on ten runs

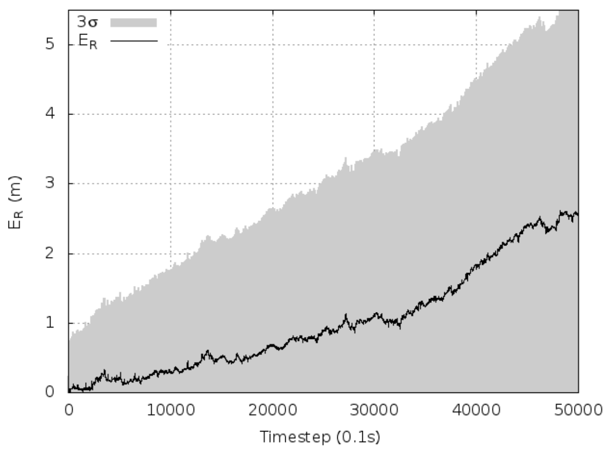

Figure 10. Sinusoidal trajectory: average position error (on the ten robots) and 3σ error as a function of time (Kalman), average on ten runs

error is of about 0.03%, and its 3σ error of about 0.04% (position error vs. trajectory length).

7.2.2 MCL results

The simulation results in the MCL case for the same circular trajectories are presented in Figure 9. The overall behaviour of the error is similar to the Kalman case. The final relative average position error is of about 0.2% and its 3σ error is of about 0.1%. In this simulation the performance of the MCL algorithm is also four to five times worse than the Kalman-based one.

7.3 Sinusoidal trajectories

Here, the ten robots composing the swarm are made to follow sinusoidal trajectories in vertical planes. The idea is to roughly simulate an actual mission in which the swarm starts at the surface, dives to a given depth, and finally re-surfaces. The robots move with a slowly varying pitch angular velocity.

7.3.1 Kalman results

In Figure 10 the average position error as a function of time in the Kalman-based case is shown. After 5000 *s* of simulation, the final relative average position error is of about 0.06% and its 3σ error of about 0.02% (position error vs. trajectory length).

Comparing the results in Figure 5, 8 and 10, the Kalman approach seems to better exploit a more complicated motion pattern than a simply linear one to lower the average position error.

7.3.2 MCL results

In Figure 11 the results in the case of the Monte Carlo localization strategy are presented. The final relative average position error is of about 0.3% and its 3σ error of about 0.2%, always after 5000 *m* of mission. In this simulation the final error values are also around five times larger than in the Kalman case.

7.3.3 Kalman with surface vessel in the loop

In the swarm operative conditions the exploitation of a surface vessel, a boat, to deploy the underwater robots may be foreseen. Once such a boat is available, it may be useful to actively use it in the swarm mission. The boat can enter the swarm as a further member with the added value of the possibility of directly measuring its own position via a GPS sensor.

In Figure 12 such a simulation is shown: a swarm of 9+1 robots has been considered. The nine vessels are the underwater ones following the above sinusoidal trajectory; the 1 is a GPS-enabled surface member.

From the results presented in Figure 13 it is evident that the average position error becomes bounded: it does not diverge with time as in the preceding simulations. This is due to the existence of a absolute position reading incoming from the GPS.

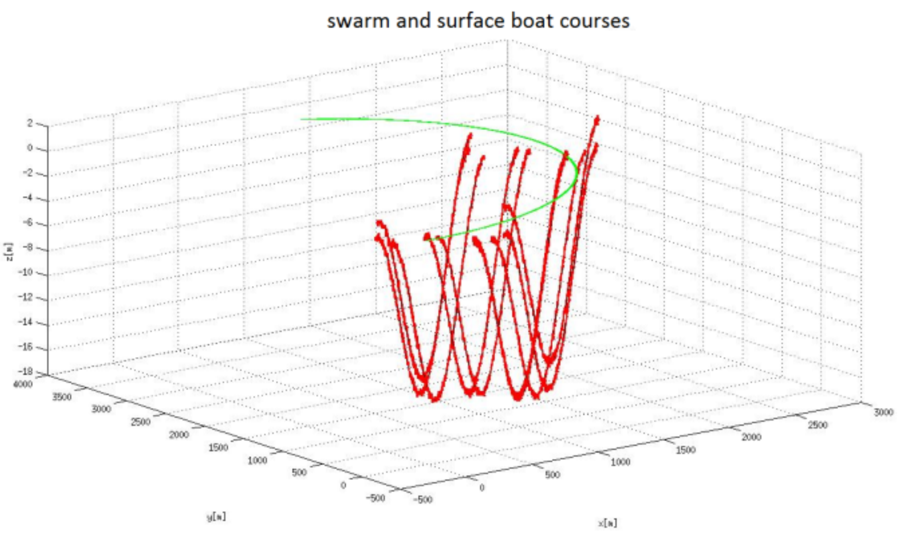

Figure 12. Surface (green) and underwater (red) courses. Units are in metres.

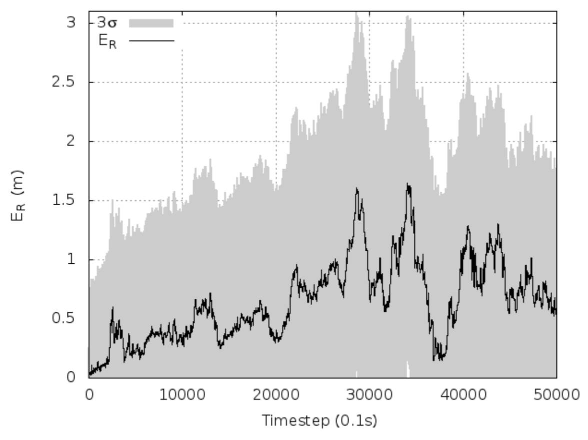

Figure 13. Sinusoidal trajectory case using a GPS-enabled boat: average position error (on the 9+1 robots) and 3σ error as a function of time (Kalman), average on ten runs

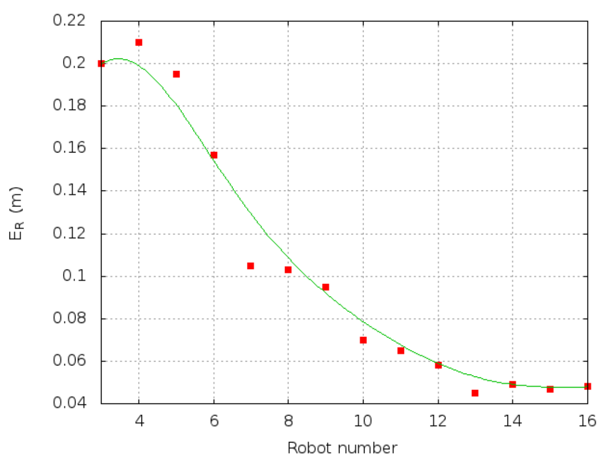

Figure 14. Average position error as a function of the robot number (Kalman case)

8. Parameter Analysis

In Figure 14 the dependence of the average position error at the end of the simulation on the number of vehicles in the swarm is shown, keeping all the other parameters fixed. It is clear that 'union is strength': the more the vessels the better the estimate, until an asymptote is reached. The figure is relative to the Kalman case, but in the MCL one the behaviour is also similar.

In the above simulations the initial positions of all the vessels composing the swarm are known, with the stated standard deviation of 0.05m. Physically, the initial position error is due to the uncertainty of the deployment of each vessel. This may vary in operative conditions due to, e.g. sea conditions; it is thus interesting to change the initial error to study its influence on the final average position error in the localization. In Figure 15 the data are plotted relative to 50 runs of the Kalman algorithm with ten robots and a mission length of 2.5 km following the sinusoidal trajectories.

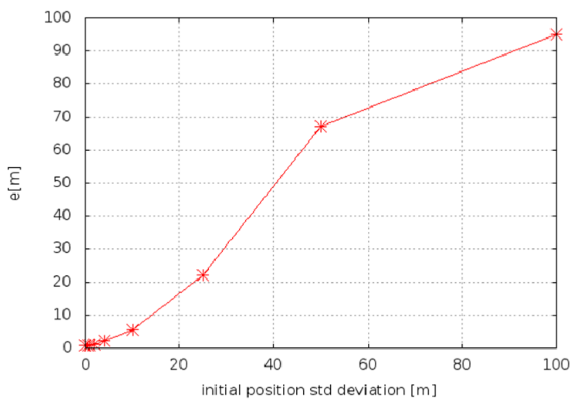

Figure 15. Average final position error as a function of the initial position standard deviation (Kalman case)

Naturally, the final position error increases with the initial uncertainty on position, but for the smaller ones (the reasonably attainable ones in operative situations) the final errors are limited. In this experiment, the standard deviation of the percentage error on the velocity measure is kept at 5%.

The behaviourof the MCL algorithm is similar, but with larger errors.

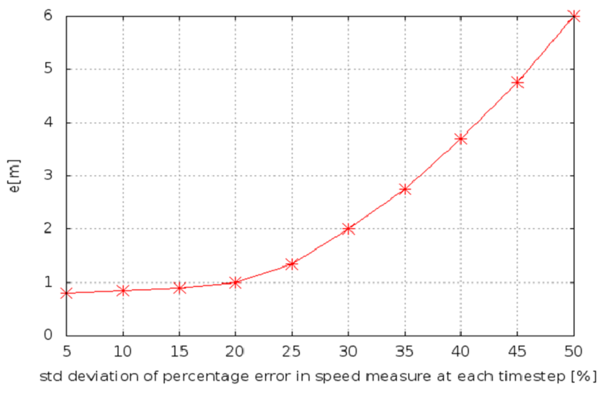

Figure 16. Average final position error as a function of the percentage error in speed at each time step (Kalman case)

At each time step, in the prediction phase the estimation of the vessel position is performed through the proprioceptive measure of the instantaneous robot velocity (dead reckoning). This measure is affected by a measure error whose percentage value can be increased in the simulations to study its influence on the algorithm robustness. In Figure 16 a plot where the final average position error is charted against the velocity measure error in the Kalman case is shown. In this series of simulations the error on the initial position is kept null.

From the plot in Figure 16 it can be deduced that the percentage error in velocity measurement, on which the dead reckoning is grounded, weakly affects the final error for the typical values that can be expected in real-world experiments. This means that the localization algorithm is able to cope with the incorrect proprioceptive speed measure unless the error is very large.

The same kind of test has been performed for the MCL case yielding similar behaviour, but with larger final errors. This reinforces the idea that the strategy of using the relative distance as an exteroceptive measure is able to tame the uncertainty of the velocity measure of the robots. On the contrary, the uncertainty of the initial positions is a more critical parameter, it makes the algorithm get more and more imprecision on the guess of the real vessel position.

9. Discussion and Conclusions

This work has presented two possible approaches to the localization of underwater swarms through the use of ultrasonic communication. The first is a Kalman-based approach, and the second is a particle filter, i.e. Monte Carlo, localization.

The overall scenario exploits some proprioceptive measures (pose, speed) and an exteroceptive one: the distance between the measuring robot and the measured one as computed through the time of flight of the exchanged sonar message. In this sense every vessel of the swarm acts as a beacon for all of the others, but the beacon position is known with a varying amount of error.

In a three-dimensional environment each vessel possesses six degrees of freedom; thus, the overall system is heavily undetermined, i.e. the covariance on the system state quickly diverges. The exploitation of real world measures such as the yaw angle (compass), roll and pitch angles (inclinometers) and the z coordinate (pressure gauge) greatly improves the localization process both for the Kalman and for the Monte Carlo-based localization, enhancing system observability.

The two approaches differ greatly in the amount of data needed to be circulated among the vessels in order to compute the robot positions.

In the Kalman filtering, during the computation the various vehicles must distribute to the others their own estimates and covariance, and all the cross-correlation matrices needed by the algorithm. This heavy communication scheme clashes against the usually available bandwidth in underwater ultrasound links, allowing swarms with a very limited number of elements.

On the contrary, the Monte Carlo-based localization needs a reduced data exchange: essentially time and position. The transmission of these data can even be considered as a header to whatever message a vessel should broadcast to the others, allowing the localization algorithm to work asynchronously and in the background.

The Kalman strategy can be labelled as a mixed distributed-centralized approach. Only one robot, the measuring one, computes the Kalman filter for all of the system elements and distributes its results to the whole community, but at the next time step a different robot will probably be the next to observe and compute the system state.

The Monte Carlo approach, instead, is fully distributed, since each robot computes its own position, in a totally passive way, as soon as a fellow in the swarm has transmitted its position and time. The update phase in a robot can be performed even on the basis of a single incoming transmission and the communication among the robots is one way only, without the need for any ping-pong between them.

In the Kalman approach, the state to be estimated is one encompassing the entire swarm, while for the MCL there are as many independent states as there are robots.

The numerical results of the presented simulations show that the Kalman approach performance, in terms of average position error, is definitely better than the MCL-based one. This is not unexpected, given the characteristics of the two schemes: on one side mathematics and on the other statistics. Besides, it must also be considered that the Kalman approach is based on a single synchronous swarm where covariances and cross-correlations among vessels are taken into account, while the MCL approach is fully decentralized and asynchronous with a weak interaction among the vessels' state variables.

An analysis of the statistical significance of the difference between the results of the two approaches (t-test) has shown that the difference is not significant (5% threshold).

Robustness analysis of the Kalman-based algorithm has shown that the final average position error can be limited even in the presence of noise in the measure of each vessel speed vector, and in the inaccuracy of initial position knowledge. These are the two major error sources: the initial position error is responsible for inaccuracy in relative distances among the robots, and speed measure noise is responsible for rendering unpredictable the future positions of the vessels. The Kalman-based algorithm is able to tame these two sources using the relative distance measures coming from the sonar TOFs.

In the MCL approach, the overall behaviour against these noise sources is similar, but the larger uncertainties, due to the statistical nature of the filtering, cause larger position errors. These worse results may also be due to an inaccurate motion model, taking into account the 10 s interval between updates. However, such a model originates from the noise characteristics of the foreseen on-board low-cost sensors that will equip the vessels.

On the side of the computational load, it is possible to assess that the two approaches are roughly of the same computing weight but with different characteristics. The Kalman one has a heavier computational load, needing e.g. matrix single value decomposition (SVD) and matrix inversion, while the other has a much lighter load but repeated over a large number of particles. It must be also noted that the Kalman approach is monolithic, i.e. the necessary computation amount is fixed, once the swarm parameters are chosen. On the other hand, the MCL approach is more flexible: depending on the on-board computing power and the frequency of incoming data, it is possible to tune the number of particles in the algorithm.

In summary, it can be affirmed that the Kalman approach performs better than the MCL approach, keeping in mind that a more thorough study of the parameters of the latter is yet to be performed. Nevertheless, the MCL-based algorithm seems more appealing when dealing with an underwater swarm, being fully asynchronous and distributed among the vessels. In addition, it is more flexible from the point of view of the computational needs, being possible to tune the particles number representing a vessel depending on the available on board computational power. Finally, it allows a much higher number of vessels, needing a lower bandwidth for communication. This also implies that the MCL-based swarm leaves more room for mission and payload communication among the vessels than a Kalman-based one.

Future work is targeted to a more thorough study of the Monte Carlo algorithm to fine-tune its parameters and characteristics for greater accuracy, and to study its scalability with the swarm element number. In particular, due attention will be placed to the conditions in which resampling takes place and its characteristics in terms of the number of resampled particles. Another aspect is the robustness of the algorithm against the motion model and the update frequency. In addition, a more realistic series of simulations, taking into account the limits on acoustic transmission, outlined in section 6, are foreseen in the near future.

10. Acknowledgements

This work was partially supported by the HARNESS project, funded by the Italian Institute of Technology (IIT) through the SEED initiative.

The authors are indebted to Dr. Marco Paoloni and Dr. Stefano Chiesa for fruitful discussions.

This article is a revised and expanded version of a paper entitled 'Kalman Underwater Acoustic 3D Localisation in Groups of Robots' presented at 'Bio-inpired Robotics' Conference, Frascati, Italy, 14-15 May, 2014.

The authors are grateful to the anonymous referees and their observations, which have greatly helped at making the paper clearer.

11. References

[1] VV.AA., (2014) Proc of the OCEANS'14 MTS/IEEE Conference, St. John's, NL, Canada.

[2] Read M., Möslinger C., Dipper T., Kengyel D., Hilder J., Thenius R., Tyrrell A., Timmis J., Schmickl T., (2014) 'Profiling Underwater Swarm Robotic Shoaling Performance Using Simulation,' Towards Autonomous Robotic Systems: Lecture Notes in Computer Science, pp. 404-416.

[3] http://www.coralbots.org/index.html. Accessed on 3 Feb 2015.

[4] http://en.wikipedia.org/wiki/Network_Time_Protocol. Accessed on 3 Feb 2015.

[5] Kurose J.F., Ross K.W., (2012) 'Computer Networking: A Top-Down Approach,' 6th ed., Pearson.

[6] Leonard J., Bennett A., Smith C., Feder H., (1998) 'Autonomous Underwater Vehicle Navigation,' MIT Marine Robotics Laboratory Technical Memorandum.

[7] Caiti A., Garulli A., Livide F., Prattichizzo D., (2005) 'Localization of Autonomous Underwater Vehicles by Floating Acoustic Buoys: A Set-Membership Approach,' IEEE Journal of Oceanic Engineering, vol. 30, no. 1., pp. 140-152.

[8] Kurazume R., Nagata S., Hirose S., (1994)'Cooperative Positioning with Multiple Robots,' Proc. 1994 IEEE Int. Conf. Robotics and Automation, Los Alamitos, CA, vol. 2, pp. 1250–1257.

[9] Kurazume R., Nagata S., Hirose S., (1996) 'Study on Cooperative Positioning System (Basic Principle and Measurement Experiment),' Proc. 1996 IEEE Int. Conf. Robotics and Automation, Minneapolis, MN,vol. 2, pp. 1421–1426.

[10] Rekleitis I.M., Dudek G., Milios E.E., (2002) 'Multi-Robot Cooperative Localization: A Study of Trade-Offs Between Efficiency and Accuracy,' Proc. of International Conference on Intelligent Robot and Systems (IROS02), Lausanne, Switzerland, vol. 3, pp. 2690-2695.

[11] Fox D., Burgard W., Kruppa H., and Thrun S., (1999) 'Collaborative Multirobot Localization,' Proc. 23rd Annual German Conf. Artificial Intelligence (KI), Bonn, Germany,pp. 255–266.

[12] Fox D., Burgard W., Kruppa H., and Thrun S., (2000) 'A Probabilistic Approach to Collaborative Multi-Robot Localization,' Autonomous Robots, vol. 8(3), pp. 325–344.

[13] Thrun S., Fox D. and Burgard W., (2000) 'Monte Carlo Localization with Mixture Proposal Distribution,' Proc. AAAI Nat. Conf. Artificial Intelligence, Austin, TX, USA, pp. 859–865.

[14] Roumeliotis S.I., (2000) 'Robust Mobile Robot Localization: From Single-Robot Uncertainties to Multi-Robot Interdependencies,' Ph.D. dissertation, Elec. Eng. Dept., Univ. Southern California, Los Angeles, CA.

[15] Roumeliotis S.I. and Bekey G.A., (2002) 'Distributed Multirobot Localization,' IEEE Transaction on Robotics and Automation, vol. 18(5), pp. 781-795.

[16] Martinelli A., Pont F., Siegwart R., (2005) 'Multi-Robot Localization Using Relative Observations,' Proc. 2005 IEEE Int. Conf. Robotics and Automation, Barcelona, Spain, pp. 2798–2802.

[17] Olfati-Saber R., (2007) 'Distributed Kalman filtering for Sensor Networks,' Proc. of the 46th IEEE Conf. on Decision and Control, pp. 5492-5498.

[18] Huang G.P., Trawny N., Mourikis A.I., Roumeliotis S.I., (2011) 'Observability-Based Consistent EKF Estimators For Multi-Robot Cooperative Localization,' Autonomous Robots, vol. 30(1), pp. 99-122.

[19] Fratichini F., Chiesa S., Taraglio S., (2013) 'Three Dimensional Localisation In Underwater Swarms Through A Kalman Approach,' Proc. of 10th International Conference on Informatics in Control, Automation and Robotics, ICINCO 2013, Reykjavik, Iceland, pp. 215-221.

[20] Thrun S., Burgard W., Fox D., (2005) 'Probabilistic Robotics,' MIT Press, Cambridge, Mass.

[21] Thrun S., Fox D., Burgard W., Dellaert F., (2001) 'Robust Monte Carlo localization for Mobile Robots,' Artificial Intelligence, vol. 128 (1–2), pp. 99–141.

[22] Hu L., Evans D., (2004) 'Localization for Mobile Sensor Networks,' Proc. of Tenth Annual International Conference on Mobile Computing and Networking (MobiCom 2004). Philadelphia.

[23] Song H.C., Hodgkiss W.S., Kuperman W.A., Akal T., Stevenson M., (2009) 'High-Frequency Acoustic Communication Achieving High Bandwidth Efficiency (Letter),' J. Acoust. Soc. Am, vol.126 (2), pp. 561-563.

[24] Rao J., Wei W., Wang F., Zhang X., (2011), 'An Underwater Optical Wireless Communication System Based on LED Source,' Proc. SPIE 8331, Photonics and Optoelectronics Meetings (POEM) 2011: Optical Communication Systems and Networking, 83310N.