Towards Fast and Reliable Localization of an Underwater Object: An Interval Approach

Quentin Brefort[1], Luc Jaulin[1],
Martine Ceberio[2], and Vladik Kreinovich[2]
[1]ENSTA-Bretagne, LabSTICC, IHSEV, OSM
2 rue François Verny, 29806 Brest, France
Quentin.Brefort@ensta-bretagne.org
luc.jaulin@ensta-bretagne.fr
[2]Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, Texas 79968, USA
mceberio@utep.edu, vladik@utep.edu

Abstract

To localize an underwater object, we measure the distance to this object from several sonar sensors with known locations. The problem is that the signal sent by some of the sonars is reflected not by the desired object(s), but by some auxiliary object and thus, the values measured by these sensors are drastically different from the distance to the desired object. To solve this problem, currently probabilistic methods are used; however, since we do not know the exact probability distributions, these methods may miss the actual location of the object. There exist interval-based methods which provide guaranteed (reliable) bounds on the object's location, but these methods sometimes require too much computation time. In this paper, we propose a new faster algorithm for reliable localization of underwater objects.

1 Formulation of the Problem

Localizing an underwater object: general idea. In some practical situations, we need to find the spatial location $\vec{x} = (x_1, x_2, x_3)$ of an underwater object: for example, of a mobile underwater robot or an adversary's submarine.

To locate this object, we can use a network of stationary omnidirectional sonars whose locations $\vec{s}_1, \ldots, \vec{s}_n$ are known. A sonar emits an acoustic signal. This signal is reflected by the object, and the reflection is detected by a sensor attached to the sonar. The sensor measures the time that passes from the emission of the original signal to the detection of the reflected signal. Since we know the speed of sound in water, we can thus measure the distance $d(\vec{s}_i, \vec{x})$ from the i-th sensor to the object.

Once we know the distances from the object to several sensors, we can determine the coordinates $\vec{x}$ of this object.

Comment. A similar problem occurs in GPS-based localization; see, e.g., [1].

In principle, this problem is solvable. Once we know the distance $d_i \stackrel{\text{def}}{=} d(\vec{s}_i, \vec{x})$ from the object to the i-th sensor, we thus get an equation with three unknown coordinates x_1, x_2, and x_3.

In general, once the number of equations is larger than or equal to the number of unknowns, this system of equations has a unique solution. Thus, if we use at least three different sensors, we can find all three coordinates and so, locate the object.

In practice, we face challenges. The above argument describes the ideal case, when all the measurements are exact, and all the measurement results are absolutely reliable.

In practice, measurements are never absolutely exact, the measurement result $\widetilde{d}_i$ is, in general, somewhat different from the actual (unknown) distance d_i; see, e.g., [8].

Also, sometimes the signal from some sonars gets reflected not from the desired object, but from some other objects – or from the shore, or from a surface separating two layers of water. In this case, the reading $\widetilde{d}_i$ of this sensor is an outlier, it has nothing to do with the actual distance d_i.

How this problem is solved now: probabilistic approach. We cannot predict the exact values of the measurement error $\Delta d_i \stackrel{\text{def}}{=} \widetilde{d}_i - d_i$, we can, at best, based on our prior experience, predict how frequent are different values of measurement error. In other words, we can, in principle, determine the probabilities of different values Δd_i.

Similarly, we cannot easily determine which measurement results correspond to reflections from the object and which to reflection from other objects. However, in principle, based on the prior experiences, we can determine the probability of a measurement result being an outlier.

Because of this, traditionally, probabilistic methods are used to locate an underwater robot.

Limitations of the probabilistic approach. The probabilistic approach is a perfect way to solve the localization problem in situations when we know all the probabilities. In most practical situations, however, we only have a *partial* knowledge of these probabilities – i.e., we have an *approximate* probabilistic model. We then use this approximate model.

It is worth mentioning that as we perform more and more measurements, we can use the measurement results to update the corresponding probability distributions – e.g., by using the Kalman filter techniques.

The problem with this approach is that it uses a (very) approximate probabilistic model which is, in general, different from the actual (unknown) probabilities. As a result, this method may miss the object.

For example, if we assume that the measurement errors are normally distributed, then, with probability 99.9%, all measurement results are within the 3 standard deviations (3σ) from the measured values; thus, with this high confidence, we conclude that the object is at a distance between $\widetilde{d}_i - 3\sigma$ and $\widetilde{d}_i + 3\sigma$. However, for many measurement procedures, the corresponding probability decreases with Δx as a power law [8], for which deviations larger than 3σ are probable. When such a deviation occurs, the actual distance from the sensor is outside the interval $[\widetilde{d}_i - 3\sigma, \widetilde{d}_i + 3\sigma]$ that our method reports.

Such a miss can be disaster: in research and underwater mineral exploration, we may lose an expensive robots; in military applications, losing track of an adversary's attack-capable submarine may lead to an even more serious disaster.

To avoid such disasters, we need to produce guaranteed (reliable) bounds on the location of the object.

Reliable methods for localizing underwater robots: interval approach. The manufacturer of the measuring instrument always provides us with the upper bound Δ on the measurement error. For measuring distance, this manufacturer-provided upper bound, in general, depends on the distance: usually, shorter distances are measured more accurately than the longer ones. As a result, for each measured value $\widetilde{d}_i$, we know the upper bound Δ_i on the corresponding measurement error $\Delta d_i = \widetilde{d}_i - d_i$: $|\Delta d_i - d_i| \le \Delta_i$.

As a result, once we know the measurement result $\widetilde{d}_i$, we can conclude that the actual (unknown) distance d_i is between the bounds: $\widetilde{d}_i - \Delta_i \le d_i \le \widetilde{d}_i + \Delta_i$. In other words, we conclude that the distance d_i belongs to the interval $[\widetilde{d}_i - \Delta_i, \widetilde{d}_i + \Delta_i]$.

Situations when for each measured quantity, we only know an interval containing its actual value, are ubiquitous; see, e.g., [8]. To process such data, special *interval computations* techniques have been invented; see, e.g., [2, 7]. It is therefore reasonable to use interval methods to get guaranteed (reliable) bounds on the actual (unknown) location of the robot.

Such a scheme is presented, e.g., in [3]. For each sonar i, the robot is located in the ring S_i formed by the two circles centered around this sonar: the ring between the circle corresponding to distance $\widetilde{d}_i - \Delta_i$ and the circle corresponding to the distance $\widetilde{d}_i + \Delta_i$. If all the recorded values $\widetilde{d}_i$ corresponded to the robot, then we could find the set S of possible locations of the robot as the intersection of the sets S_i corresponding to all m sonars. In real life, as we have mentioned, some measurements come from other objects; in this case, some of the sets S_i reflect locations of these other objects, and thus, the overall intersection may be empty. We need to take this fact into account.

The actual location of the robot belongs to the intersection of all the sets S_i for all i for which the i-th sensor detects the reflection from the robot. One or more other sensors may detect reflection from another object(s); thus, the intersection of the corresponding sets S_j contains the location of that other object. Usually, most sensors detect reflection from the sensor, so we can find the actual location of the sensor as a non-empty intersection of a subfamily of the family of all the sets S_i – a subfamily for which the number of intersecting sets with non-empty intersection is the largest possible.

To locate the robot, we therefore use a Guaranteed Outlier Minimal Number Estimator (GOMNE) described in [2, 4, 5]. This algorithm first finds the largest possible value q for which the intersection of q sets S_i is non-empty, then finds the corresponding "q-relaxed intersection", i.e., the union of all non-empty intersections of q sets S_i: $\bigcup\limits_{I:\#(I)=q} \bigcap\limits_{i\in I} S_i$. To compute the corresponding intersections, GOMNE uses SIVIA (Set Inversion via Interval Analysis), an algorithm described in [2].

Simulations show that in more than 90% of the cases, the resulting algorithm finds the correct location of the robot, which is much more efficient than for the previously known reliable methods of locating underwater robots.

Main limitation of the existing interval approach: it is too slow. Off-line, the above interval methods works perfectly well. However, we need to determine the object's coordinates in real time. The object is moving, so we need to know its location before it moved away from this location. As the number of sensors increases, the needed computation time increases drastically – so that it exceeds the time needed to real-time computations.

It is therefore necessary to develop faster algorithms for reliable localization of underwater objects.

What we do in this paper. In this paper, we propose a new interval-based method for fast and reliable localization of underwater objects.

Comment. To make our description clearer, we illustrate our main ideas on the simplified example of a 2-D localization. These ideas can be easily expanded to a more realistic 3-D localization problem – and in our description, we explain how they can be expanded.

2 Analysis of the Problem

Two stages of localization. Once we have located an object, we repeatedly send sonar signals to find its updated location. In this paper, we will denote the time interval between two sequential measurements by Δt.

From the computational viewpoint, it is therefore reasonable to consider two stages of the localization process:

- first, we have no prior information on where the object is, and we need to find its initial coordinates $\vec{x}$;
- on the second stage, we know the approximate location $\vec{x}_0 \approx \vec{x}(t-\Delta t)$ of the object at the previous moment of time $t-\Delta t$ (and we know the accuracy ε_0 of this approximation), and we want to use this information, as

well as the results of the measurements performed at the current moment of time t, to find the object's current location $\vec{x}(t)$.

Which stage is easier? In general, the more information we have, the better: we get more accurate estimates, and we can often use more computationally efficient algorithms. For sure, the additional information cannot worsen the performance: if the new information does not lead to a more accurate or faster estimation, we can simply ignore it.

From this viewpoint, let us compare the two stages that we described in the previous subsection. At the first stage, all we know are measurement results. At the second stage, in addition to the measurement results, we also have an additional information: we know the previous location of the object. Thus, the localization problem corresponding to the second stage is easier to solve.

Because of this comparison, we will start our analysis with this easier-to-solve second stage, and then we will explain how our ideas can be expanded to the more-difficult-to-solve second stage.

Measurements are frequent. To prevent losing track of the object, the existing sonar systems perform measurements very frequently. Thus, the time interval Δt between the two consequent measurements is usually very small.

We know the upper bound v on the velocity of the underwater object. Thus, during the time Δt, the object cannot move further away than the distance $\varepsilon \stackrel{\text{def}}{=} v \cdot \Delta t$: $d(\vec{x}(t), \vec{x}(t-\Delta)) \le \varepsilon$.

We also know the approximate location $\vec{x}_0$ of the object at moment $t - \Delta t$, and we know the accuracy ε_0 of this approximation. Thus, we have $d(\vec{x}(t - \Delta t), \vec{x}_0) \le \varepsilon_0$. By the triangle inequality, we now have

$$d(\vec{x}(t), \vec{x}_0) \le d(\vec{x}(t), \vec{x}(t-\Delta)) + d(\vec{x}(t-\Delta t), \vec{x}_0) \le \varepsilon + \varepsilon_0.$$

Resulting constraints on $\Delta\vec{x} \stackrel{\text{def}}{=} \vec{x}(t) - \vec{x}_0$. The result $\widetilde{d}_i$ of the i-th measurements constraints the actual location $\vec{x}(t)$. Let us reformulate this constraint in terms of the difference $\Delta\vec{x} \stackrel{\text{def}}{=} \vec{x}(t) - \vec{x}_0$.

Once we know this difference, we can easily reconstruct the actual location $\vec{x}$ as $\vec{x} = \vec{x}_0 + \Delta\vec{x}$.

We know that $\widetilde{d}_i - \Delta_i \le d(\vec{x}(t), \vec{s}_i) \le \widetilde{d}_i + \Delta_i$. By squaring all three sides of this double inequality, we get

$$(\widetilde{d}_i - \Delta_i)^2 \le d^2(\vec{x}(t), \vec{s}_i) \le (\widetilde{d}_i + \Delta_i)^2. \tag{1}$$

Here,

$$d^2(\vec{x}(t), \vec{s}_i) = (\vec{x}(t) - \vec{s}_i)^2.$$

In terms of the difference $\Delta\vec{x}$, we have $\vec{x}(t) = \vec{x}_0 + \Delta x$, thus

$$d^2(\vec{x}(t), \vec{s}_i) = (\vec{x}_0 + \Delta\vec{x} - \vec{s}_i)^2 = ((\vec{x}_0 - \vec{s}_i) + \Delta\vec{x})^2 =$$

$$(\vec{x}_0-\vec{s}_i)^2+2\Delta\vec{x}\cdot(\vec{x}_0-\vec{s}_i)+(\Delta\vec{x})^2. \tag{2}$$

Substituting the expression (2) into the formula (1), we get

$$(\widetilde{d}_i-\Delta_i)^2\le(\vec{x}_0-\vec{s}_i)^2+2\Delta\vec{x}\cdot(\vec{x}_0-\vec{s}_i)+(\Delta\vec{x})^2\le(\widetilde{d}_i+\Delta_i)^2.$$

Subtracting $(\vec{x}_0-\vec{s}_i)^2+(\Delta\vec{x})^2$ from all the sides of this inequality, we get

$$(\widetilde{d}_i-\Delta_i)^2-(\vec{x}_0-\vec{s}_i)^2-(\Delta\vec{x})^2\le 2\Delta\vec{x}\cdot(\vec{x}_0-\vec{s}_i)\le$$

$$(\widetilde{d}_i+\Delta_i)^2-(\vec{x}_0-\vec{s}_i)^2-(\Delta\vec{x})^2. \tag{3}$$

We know that $0\le(\Delta\vec{x})^2\le(\varepsilon+\varepsilon_0)^2$, so $-(\varepsilon+\varepsilon_0)^2\le-(\Delta\vec{x})^2\le 0$ and thus, (3) implies that

$$(\widetilde{d}_i-\Delta_i)^2-(\vec{x}_0-\vec{s}_i)^2-(\varepsilon+\varepsilon_0)^2\le 2(\vec{x}_0-\vec{s}_i)\cdot\Delta\vec{x}\le(\widetilde{d}_i+\Delta_i)^2-(\vec{x}_0-\vec{s}_i)^2, \tag{4}$$

or, equivalently,

$$\underline{v}_i\le\vec{a}_i\cdot\Delta\vec{x}\le\overline{v}_i, \tag{5}$$

where we denoted

$$\vec{a}_i\stackrel{\text{def}}{=}2(\vec{x}_0-\vec{s}_i), \tag{6}$$

$$\underline{v}_i\stackrel{\text{def}}{=}(\widetilde{d}_i-\Delta_i)^2-(\vec{x}_0-\vec{s}_i)^2-(\varepsilon+\varepsilon_0)^2, \tag{7}$$

and

$$\overline{v}_i\stackrel{\text{def}}{=}(\widetilde{d}_i+\Delta_i)^2-(\vec{x}_0-\vec{s}_i)^2. \tag{8}$$

For interval computations, it is often convenient to express an interval $[\underline{v}_i,\overline{v}_i]$ by its midpoint $\widetilde{v}_i\stackrel{\text{def}}{=}\dfrac{\underline{v}_i+\overline{v}_i}{2}$ and its *radius* (half-width) $\delta_i\stackrel{\text{def}}{=}\dfrac{\overline{v}_i-\underline{v}_i}{2}$; see, e.g., [2, 7]. In these terms, the interval takes the form $[\underline{v}_i,\overline{v}_i]=[\widetilde{v}_i-\delta_i,\widetilde{v}_i+\delta_i]$, and the double inequality (5) takes the form

$$\vec{a}_i\cdot\Delta\vec{x}\in[\widetilde{v}_i-\delta_i,\widetilde{v}_i-\delta_i]. \tag{9}$$

In our case, from (7) and (8), we conclude that

$$\widetilde{v}_i=(\widetilde{d}_i)^2+\Delta_i^2-(\vec{x}_0-\vec{s}_i)^2-\frac{1}{2}\cdot(\varepsilon+\varepsilon_0)^2; \tag{10}$$

$$\delta_i=2\widetilde{d}_i\cdot\Delta_i+\frac{1}{2}\cdot(\varepsilon+\varepsilon_0)^2; \tag{11}$$

Let us consider all pairs of sensors (triples, in 3-D case). Let us first consider a 2-D case. Let us assume that two sensors i and j both detect the reflection from the object. In this case, the difference $\Delta\vec{x}$ satisfies two conditions:

$$\vec{a}_i\cdot\Delta\vec{x}\in[\widetilde{v}_i-\delta_i,\widetilde{v}_i+\delta_i]; \tag{12a}$$

$$\vec{a}_j\cdot\Delta\vec{x}\in[\widetilde{v}_j-\delta_j,\widetilde{v}_j+\delta_j] \qquad \text{(12b).}$$

The fact that the scalar (dot) product $\Delta\vec{x}\cdot\vec{a}_i$ belongs to the interval $[\widetilde{v}_i-\delta_i,\widetilde{v}_i+\delta_i]$ means that the absolute value of the difference $\Delta v_i\stackrel{\text{def}}{=}\Delta\vec{x}\cdot\vec{a}_i-\widetilde{v}_i$ does not exceed δ_i: $|\Delta v_i|\le\delta_i$. In terms of the values Δv_i and Δv_j, the conditions (12a) and (12b) can be described as

$$\vec{a}_i\cdot\Delta\vec{x}=\widetilde{v}_i+\Delta v_i; \tag{13a}$$

$$\vec{a}_j\cdot\Delta\vec{x}=\widetilde{v}_j+\Delta v_j. \tag{13b}$$

with $|\Delta v_i|\le\delta_i$ and $|\Delta v_j|\le\delta_j$.

In coordinate terms, $\Delta\vec{x}=(\Delta x_1,\Delta x_2)$, $\vec{a}_i=(a_{i1},a_{i2})$, $\vec{a}_j=(a_{j1},a_{j2})$, and the system (13) takes the form

$$a_{i1}\cdot\Delta x_1+a_{i2}\cdot\Delta x_2=\widetilde{v}_i+\Delta v_i; \tag{14a}$$

$$a_{j1}\cdot\Delta x_1+a_{j2}\cdot\Delta x_2=\widetilde{v}_j+\Delta v_j, \tag{14b}$$

i.e., in matrix form,

$$A(\Delta x)=v, \tag{15}$$

where

$$A\stackrel{\text{def}}{=}\begin{pmatrix}a_{i1} & a_{i2}\\ a_{j1} & a_{j2}\end{pmatrix}.$$

Thus, for the matrix $B=A^{-1}$ with components $B=\begin{pmatrix}b_{1i} & b_{1j}\\ b_{2i} & b_{2j}\end{pmatrix}$, we have $\Delta x=B\cdot v$, i.e., we have, for $m=1,2$:

$$\Delta x_m=b_{mi}\cdot(\widetilde{v}_i+\Delta v_i)+b_{mj}\cdot(\widetilde{v}_j+\Delta v_j). \tag{16}$$

From (16), we get

$$\Delta x_m=\widetilde{x}_m+\delta x_m, \tag{17}$$

where

$$\widetilde{x}_m\stackrel{\text{def}}{=}b_{mi}\cdot\widetilde{v}_i+b_{mj}\cdot\widetilde{v}_j; \tag{18}$$

$$\delta x_m\stackrel{\text{def}}{=}b_{mi}\cdot\Delta v_i+b_{mj}\cdot\Delta v_j. \tag{19}$$

We know that $|\Delta v_i|\le\delta_i$ and $|\Delta v_j|\le\delta_j$. In general, by considering two cases $c\ge 0$ and $c\le 0$, one can easily check that the largest value of a linear function $c\cdot x$ for $|x|\le t$ is equal to $|c|\cdot t$. Thus, the largest possible value of the expression (19) is equal to

$$r_m\stackrel{\text{def}}{=}|b_{mi}|\cdot\delta_i+|b_{mj}|\cdot\delta_j. \tag{20}$$

Thus, for each $m=1,2$, if both measurements i and j measure reflections from the desired object (and not from some other object), then we conclude that

$$\Delta x_m\in[\underline{x}_m,\overline{x}_m]\stackrel{\text{def}}{=}[\widetilde{x}_m-r_m,\widetilde{x}_m+r_m]. \tag{21}$$

In the 3-D case, formulas are similar, the only difference is that to find three coordinates, we need to consider triples of sensors (i, j, k), and thus, we need to invert the corresponding 3×3 matrices.

From intervals corresponding to all possible pairs (or triples) to actual location of the underwater object. We assume that out of n sensors, the vast majority q detect the reflection from the actual object. Thus, out of possible $\dfrac{n \cdot (n-1)}{2}$ pairs of sensors, for $\dfrac{q \cdot (q-1)}{2}$ pairs – the majority – the above procedure will lead to an interval containing the actual location of the robot.

For some other pairs of sensors, we will get the location of an auxiliary object (when both sensors detect signals reflected form that object) or just a meaningless interval – when two sensors detect reflections from different objects.

As a result, the intersection of all the intervals $[\underline{x}_m, \overline{x}_m]$ corresponding to all possible pairs is usually empty: the actual locations are contained in many such intervals, while the locations of auxiliary objects are contained in few such intervals.

For each real number, we can find the number of intervals $[\underline{x}_m, \overline{x}_m]$ containing this number. Based on the above analysis, as possible locations of the object, we should select the set of all the points for which the number of containing intervals is the largest possible.

This set can be computed as follows (see, e.g., [6]):

- first, we sort all the endpoints $\underline{x}_m$ and $\overline{x}_m$ corresponding to all the pairs (triples) of sensors, into an increasing sequence

$$x_{(0)} \stackrel{\text{def}}{=} -\infty < x_{(1)} \le x_{(2)} \le \ldots \le x_{(N)} < x_{(N+1)} \stackrel{\text{def}}{=} +\infty;$$

- then, for $k = 0, 1, \ldots, N$, we sequentially compute the number I_k of intervals $[\underline{x}_m, \overline{x}_m]$ that contain values from the interval $[x_{(k)}, x_{(k+1)}]$ as follows:
  - we start with $I_0 = 0$;
  - once we know I_{k-1}, we take $I_k = I_{k-1} + 1$ if $x_{(k)}$ is one of the lower bounds $\underline{x}_m$ (so, a new containing interval is added) and we take $I_k = I_{k-1} - 1$ if $x_{(k)}$ is one of the upper bounds $\underline{x}_m$ (so, one of the containing intervals is deleted).
- Then, we find the largest of the values $I_0, \ldots, I_N$, and we return the interval $[x_{(k)}, x_{(k+1)}]$ for which I_k is equal to this largest value. If there are several such indices k corresponding to different values k, we return the interval $[x_{(\underline{k})}, x_{(\overline{k}+1)}]$, where $\underline{k}$ is the smallest of such indices and $\overline{k}$ is the largest of such indices.

What about the first stage. The above ideas described the second stage, when we already know the location of the robot at the previous moment of time $t - \Delta t$.

What about the first stage, when we have no prior information about the robot's location? On the first stage, we can repeat the same procedure – i.e., consider all pairs (or all triples) of sensors, and find the interval corresponding to the majority of sensors.

The only difference is that we do not know the previous location $\vec{x}_0$ and thus, we cannot use the above linearization technique – when we represented the unknown location $\vec{x}$ as $\vec{x}_0 + \Delta\vec{x}$ and took into account that the difference $\Delta\vec{x}$ is small. Instead, to find a possible location, we have to use the original inequalities

$$(\widetilde{d}_i - \Delta_i)^2 \le (\vec{x} - \vec{s}_i)^2 \le (\widetilde{d}_i - \Delta_i)^2; \tag{22a}$$

$$(\widetilde{d}_j - \Delta_j)^2 \le (\vec{x} - \vec{s}_j)^2 \le (\widetilde{d}_j - \Delta_j)^2; \tag{22b}$$

$$(\widetilde{d}_k - \Delta_k)^2 \le (\vec{x} - \vec{s}_k)^2 \le (\widetilde{d}_k - \Delta_k)^2. \tag{22c}$$

If we knew the exact values d_i, d_j, and d_k of the distances, then we would get a system

$$(\vec{x} - \vec{s}_i)^2 = (\vec{x})^2 - 2\vec{s}_i \cdot \Delta x + (\vec{s}_i)^2 = d_i^2; \tag{23a}$$

$$(\vec{x} - \vec{s}_j)^2 = (\vec{x})^2 - 2\vec{s}_j \cdot \Delta x + (\vec{s}_j)^2 = d_j^2; \tag{23b}$$

$$(\vec{x} - \vec{s}_k)^2 = (\vec{x})^2 - 2\vec{s}_k \cdot \Delta x + (\vec{s}_k)^2 = d_k^2. \tag{23c}$$

If we subtract equation (23a) from each of the equations (23b) and (23c), then we get two equations which are linear in $\vec{x}$, i.e., linear in terms of the three coordinates x_1, x_2, and x_3. We can use these two linear equations to express x_2 and x_3 as linear functions of x_1. Substituting these linear expressions into the equation (23a), we will then get an easy-to-solve quadratic equation for x_1.

The only remaining problem is to take into account that instead of the exact values d_i^2, we only have an interval of possible values $[(\widetilde{d}_i - \Delta_i)^2, (\widetilde{d}_i + \Delta_i)^2]$. This can be taken into account by using standard interval computations techniques such as centered form [2, 7].

Thus, we arrive at the following algorithm.

3 Resulting Algorithm

What is known. Before we start the measurements, we know:

- the time interval Δt between two consequent measurements,
- the upper bound v on the possible velocity of the detected object, and
- for each i from 1 to n, the location $\vec{s}_i$ of the i-th sensor.

Based on these values, we pre-compute the value $\varepsilon = v \cdot \Delta t$.

After the measurements are performed, we know, for each i from 1 to n:

- the result $\widetilde{d}_i$ of the i-th distance measurement, and
- the upper bound Δ_i on the accuracy of this measurement.

At the *first stage*, when we have no prior information about the location of the robot, this is all we know. Once we have detected the object, we reach the *second stage*, at which, at each moment of time t, we also know:

- the estimated location $\vec{x}_0 = (x_{01}, x_{02}, x_{03})$ of the robot at the previous moment of time $t - \Delta t$, and
- the accuracy ε_0 with which we know this location, i.e., an upper bound on the distance $d(\vec{x}(t - \Delta t), \vec{x}_0)$.

Algorithm: general description. First, we consider all possible triples of sensors (pairs in the 2-D case), and we use the measurement results of these three sensors to find, for each of the 3 coordinates $m = 1, 2, 3$, the interval $[\underline{x}_m, \overline{x}_m]$ of possible values of $x_m(t)$ (on the first stage) or $\Delta x_m = x_m(t) - x_{0m}$ (on the second stage).

For each m, we then:

- sort all the endpoints $\underline{x}_m$ and $\overline{x}_m$ into an increasing sequence
$$x_{(0)} \stackrel{\text{def}}{=} -\infty < x_{(1)} \le x_{(2)} \le \ldots \le x_{(N)} < x_{(N+1)} \stackrel{\text{def}}{=} +\infty;$$
- then, for $k = 0, 1, \ldots, N$, we sequentially compute the number I_k as follows:
  - we start with $I_0 = 0$;
  - once we know I_{k-1}, we take $I_k = I_{k-1} + 1$ if $x_{(k)}$ is one of the lower bounds $\underline{x}_m$, and we take $I_k = I_{k-1} - 1$ if $x_{(k)}$ is one of the upper bounds $\underline{x}_m$.
- Then, we find the largest of the values $I_0, \ldots, I_N$, and we return the interval $[x_{(k)}, x_{(k+1)}]$ for which I_k is equal to this largest value. If there are several such indices k corresponding to different values k, we return the interval $[x_{(\underline{k})}, x_{(\overline{k}+1)}]$, where $\underline{k}$ is the smallest of such indices and $\overline{k}$ is the largest of such indices.

These intervals describe the object's location:

- on the first stage, the intervals' midpoints form the approximate location vector $\vec{x}_0$;
- on the second stage, these midpoints, when added to the previous location $\vec{x}_0$, form the new approximate location vector $\vec{x}_0$.

On both stages, the square root of the sum of squares of radii of these intervals is the (new) location accuracy ε_0.

How to compute the intervals $[\underline{x}_m, \overline{x}_m]$? To complete this description, we need to describe how to compute the intervals $[\underline{x}_m, \overline{x}_m]$ corresponding to different triples of sensors.

This computation is different on the first stage, when we do not yet have any prior information about the object, and on the second stage, when we already know the object's previous location. We will describe these two cases one by one.

How to compute an interval $[\underline{x}_m, \overline{x}_m]$ corresponding to sensors i, j, and k: first stage. If we knew the exact values d_i, d_j, and d_k of the distances, then we would get a system

$$(\vec{x})^2 - 2\vec{s}_i \cdot \Delta x + (\vec{s}_i)^2 = d_i^2; \tag{24a}$$

$$(\vec{x})^2 - 2\vec{s}_j \cdot \Delta x + (\vec{s}_j)^2 = d_j^2; \tag{24b}$$

$$(\vec{x})^2 - 2\vec{s}_k \cdot \Delta x + (\vec{s}_k)^2 = d_k^2. \tag{24c}$$

We subtract equation (24a) from each of the equations (24b) and (24c); as a result, we get two equations which are linear in $\vec{x}$, i.e., linear in terms of the three coordinates x_1, x_2, and x_3. We use these two linear equations to express x_2 and x_3 as linear functions of x_1. Substituting these linear expressions into the equation (24a), we get an easy-to-solve quadratic equation for x_1. Once we know x_1, we can use the known linear formulas describing x_2 and x_3 in terms of x_1 to find the values of x_2 and x_3.

To take into account that instead of the exact values d_i^2, we only have an interval of possible values $[(\widetilde{d}_i - \Delta_i)^2, (\widetilde{d}_i - \Delta_i)^2]$, we use standard interval computations techniques such as centered form.

How to compute an interval $[\underline{x}_m, \overline{x}_m]$ corresponding to sensors i, j, and k: second stage. For each sensor, we compute the values

$$\widetilde{v}_i = (\widetilde{d}_i)^2 + \Delta_i^2 - (\vec{x}_0 - \vec{s}_i)^2 - \frac{1}{2} \cdot (\varepsilon + \varepsilon_0)^2; \tag{10}$$

$$\delta_i = 2\widetilde{d}_i \cdot \Delta_i + \frac{1}{2} \cdot (\varepsilon + \varepsilon_0)^2; \tag{11}$$

and the vector $\vec{a}_i = 2(\vec{x}_0 - \vec{s}_i)$ with coordinates a_{i1}, a_{i2}, a_{i3}).

Then, we form a matrix $A = \begin{pmatrix} a_{i1} & a_{i2} & a_{i3} \\ a_{j1} & a_{j2} & a_{j3} \\ a_{k1} & a_{k2} & a_{k3} \end{pmatrix}$, and compute its inverse matrix $B = \begin{pmatrix} b_{1i} & b_{1j} & b_{1k} \\ b_{2i} & b_{2j} & b_{2k} \\ b_{3i} & b_{3j} & b_{3k} \end{pmatrix}$. For each $m = 1, 2, 3$, we compute $\widetilde{x}_m = b_{mi} \cdot \widetilde{v}_i + b_{mj} \cdot \widetilde{v}_j + b_{mk} \cdot \widetilde{v}_k$, $r_m = |b_{mi}| \cdot \delta_i + |b_{mj}| \cdot \delta_j + |b_{mk}| \cdot \delta_k$, $\underline{x}_m = \widetilde{x}_m - r_m$, and $\overline{x}_m = \widetilde{x}_m + r_m$.

Acknowledgment

This work was supported in part by the US National Science Foundation grants 0953339, HRD-0734825 and HRD-1242122 (Cyber-ShARE Center of Excellence) and DUE-0926721. The work was performed when Quentin Brefort was visiting the University of Texas at El Paso. This visit was supported by ENSTA-Bretagne.

References

[1] V. Drevelle and P. Bonnifait, "iGPS: global positioning in urban canyons with road surface maps", *IEEE Intelligent Transportation Systems Magazine*, 2012, Vol. 4, No. 3, pp. 6–18.

[2] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter, *Applied Interval Analysis*, Springer Verlag, London, 2001.

[3] L. Jaulin, A. Stancu, and B. Desrochers, "Inner and outer approximations of probabilistic sets", *Proceedings of the American Society of Civil Engineers (ASCE) Second International Conference on Vulnerability and Risk Analysis and Management ICVRAM'2014 and Sixth International Symposium on Uncertainty Modelling and Analysis ISUMA'2014*, Liverpool, UK, July 13–16, 2014, to appear.

[4] L. Jaulin and E. Walter, "Guaranteed robust nonlinear minimax estimation", *IEEE Transaction on Automatic Control*, 2002, Vol. 47, No. 11, pp. 1857–1864.

[5] L. Jaulin, E. Walter and O. Didrit, "Guaranteed robust nonlinear parameter bounding", *Proceedings of Symposium on Modelling, Analysis and Simulation, part of IMACS Multiconference on IMACS Multiconference, Computational Engineering in Systems Applications CESA'96*, Lille, France, July 9–12, 1996, Vol. 2, pp. 1156–1161.

[6] K. A. Marzullo, *Maintaining the Time in a Distributed System: An Example of a Loosely-Coupled Distributed Service*, Stanford University, PhD Dissertation, 1984.

[7] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis.* SIAM Press, Philadelphia, Pennsylviania, 2009.

[8] S. G. Rabinovich, *Measurement Errors and Uncertainty: Theory and Practice*, Springer Verlag, Berlin, 2005.