Localization of an underwater robot using interval constraint propagation

Luc Jaulin
E3I2, ENSIETA, 2 rue François Verny, 29806 Brest Cédex 09

[1] E3I2, ENSIETA, France
luc.jaulin@ensieta.fr,
http://www.ensieta.fr/e3i2/Jaulin/
[2] GESMA (Groupe d'Etude Sous-Marine de l'Atlantique), Brest, France

Abstract. Since electromagnetic waves are strongly attenuated inside the water, the satellite based global positioning system (GPS) cannot be used by submarine robots except at the surface of the water. This paper shows that the localization problem in deep water can often be cast into a continuous constraints satisfaction problem where interval constraints propagation algorithms are particularly efficient. The efficiency of the resulting propagation methods is illustrated on the localization of a submarine robot, named *Redermor*. The experiments have been collected by the GESMA (Groupe d'Etude Sous-Marine de l'Atlantique) in the Douarnenez bay, in Brittany.

1. Introduction

This paper deals with the *simultaneous localization and map building* problem (SLAM) in a submarine context (see [10] for the general SLAM problem). The SLAM problem asks if it is possible for an autonomous robot to move in an unknown environment and build a map of this environment while simultaneously using this map to compute its location.

In this paper, we will show that the SLAM problem can be seen as a *continuous constraints satisfaction problem (CCSP)* (see e.g., [16], [2], [14], [15] for notions related CCSP and applications). Then, we will propose to use a basic constraints propagation algorithm (2B-consistency) to solve the CCSP. The efficiency of the approach will be illustrated on an experiment where an actual underwater vehicle is involved. In this problem, we will try to find an envelope for the trajectory of the robot and to compute sets which contain some detected objects.

Many ideas presented here can be found in [5] and [12] where interval analysis has already been used in the context of SLAM for wheeled robots. But the approach is here made more efficient by the addition of constraints propagation techniques, that have never been used in this context. Note that there exist many other robotics applications where interval constraints propagation methods have been successful (see e.g., [1] for the calibration of robots, [9], [13] for state estimation, [11], [17] for control of robots, [4] for topology analysis of configuration spaces, ...).

The paper is organized as follows. The robot to be considered will first be presented in Section 2. Then, in Section 3, a brief description of the available sensors will be given. By taking into account the state equations of the robot and the interpretation of the sensors, Section 4 will provide the constraints that will make it possible to cast our SLAM problem into a CCSP.

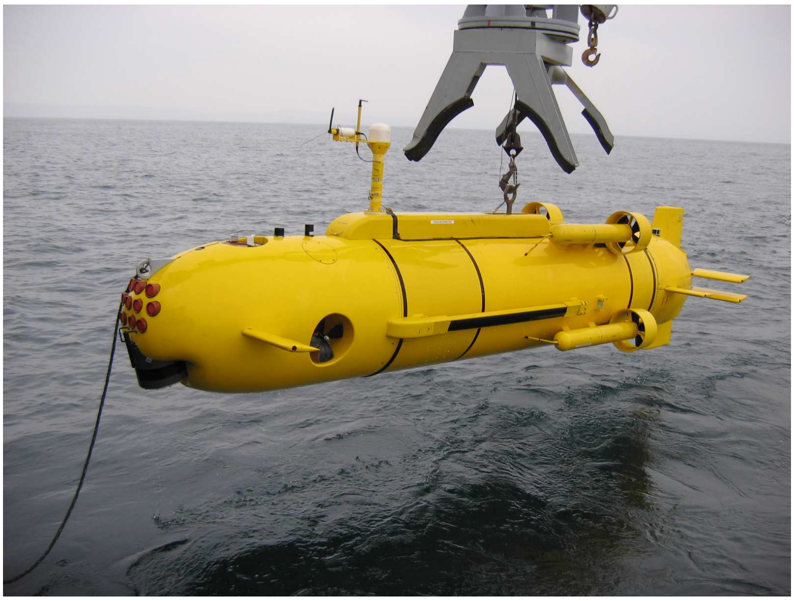

Fig. 2.1. The autonomous underwater vehicle, *Redermor*, built by the GESMA (Groupe d'Etude Sous-Marine de l'Atlantique)

The efficiency of our approach will be illustrated on an actual expriment in Section 5.1. Section 6 will then conclude the paper.

2. Robot

The robot to be considered in our application (see Figure 2.1) is an autonomous underwater vehicle (AUV), named *Redermor* (means *greyhound of the sea*, in the Breton language). This robot, developed by the GESMA (Groupe d'Etude Sous-Marine de l'Atlantique), has a length of 6 m, a diameter of 1 m and a weight of 3800 Kg. It has powerful propulsion and control system able to provide hovering capabilities. The main purpose of the *Redermor* is to evaluate improved navigation by the use of sonar information. It is equipped with a KLEIN 5400 side scan sonar which makes it possible to localize objects such as rocks or mines. It also encloses other sophisticated sensors such as a Lock-Doppler to estimate its speed and a gyrocompass to get its three Euler angles (i.e., its orientation).

3. Measurements

3.1. Sensors

The robot is equipped with the following sensors

- **A GPS** (Global Positioning System). A constellation of 24 satellites broadcasts precise timing signals by radio to GPS receivers, allowing them to accurately determine their location (longitude, latitude and altitude) in any weather, day or night, anywhere on the surface of the Earth. However, since electromagnetic waves (here around 1.2 MHz), do not propagate through the water, this sensor is operational only when the robot is at the surface of the ocean, but not when it is inside the water. During our two-hours experiment, using the GPS, the robot is only able to measure the location where it is dropped and the location where it comes back to the surface. Thus, we know that at time $t_0 = 6000$ s, the robot has been dropped approximately around the position

  $$\ell^0 = (\ell_x^0, \ell_y^0) = (-4.458227931^{\text{o}}, 48.212920614^{\text{o}})\,, \tag{3.1}$$

  where ℓ_x^0 is the west/east longitude and ℓ_y^0 is the south/north latitude. The error related to this position is less than 2.5 meters. When the robot returns to the surface, at time $t_f = 11999.4$ sec, its position is approximately (*i.e.*, again with an error less than 2.5 meters) given by

  $$\ell^f = (\ell_x^f, \ell_y^f) = (-4.454660760^{\text{o}}, 48.219129760^{\text{o}})\,. \tag{3.2}$$

- **A sonar** (KLEIN 5400 side scan sonar). During its mission, the robot detects objects using a sonar located starboard (*i.e.* on its right-hand side). This sonar emits ultrasonic waves to build images such as that represented on Figure 3.1. This image, also called a *waterfall,* is about 75m large for more than 10 km high (corresponding to the length covered by the robot during its mission). After the mission, a scrolling of the waterfall is performed by a human operator which is then able to perform an estimation $\tilde{r}(t)$ of the distance $r(t)$ from the robot to an object detected at time t. Recall that the positions of the objects are assumed to be unknown. From the width of the black vertical band on the left of the picture (called the *water column*), we are also able to compute an estimation $\tilde{a}(t)$ of the altitude $a(t)$ of the robot (distance between the robot and the bottom). Figure 3.1 is related to the detection of the 5th object in the case of the mission made by the robot. The associated ping is represented by the thin white rectangle. Up to now, the detection of an object and the matching between objects are performed manually, from a scrolling of the waterfall, once the robot has accomplished its mission. But we are planning to develop an automatic and reliable procedure for this task.
- **A Loch-Doppler.** This sensor makes it possible to compute the speed of the robot $\mathbf{v}_r$ and returns it in the robot frame. The Lock-Doppler emits ultrasonic waves which are reflected on the bottom of the ocean. Since the bottom is immobile, this sensor is able to compute an estimation of its speed using the Doppler effect. When the frequency of the waves is around 300 kHz, then the actual speed is known to satisfy

  $$\mathbf{v}_r \in \tilde{\mathbf{v}}_r + 0.004 * [-1, 1]\,.\tilde{\mathbf{v}}_r + 0.004 * [-1, 1]. \tag{3.3}$$

  where $\tilde{\mathbf{v}}_r$ denotes the three dimensional speed vector returned by the sensor. The Loch-Doppler is also able to provide the altitude a of the robot with an error less than 10cm.
- **A Gyrocompass** (Octans III from IXSEA). This sensor uses the Sagnac effect and the rotation of the earth to compute the three Euler angles (the roll ϕ, the pitch θ, and the head ψ) of the robot with a high accuracy. If we denote by $\tilde{\phi}, \tilde{\theta}, \tilde{\psi}$, the angles returned by our

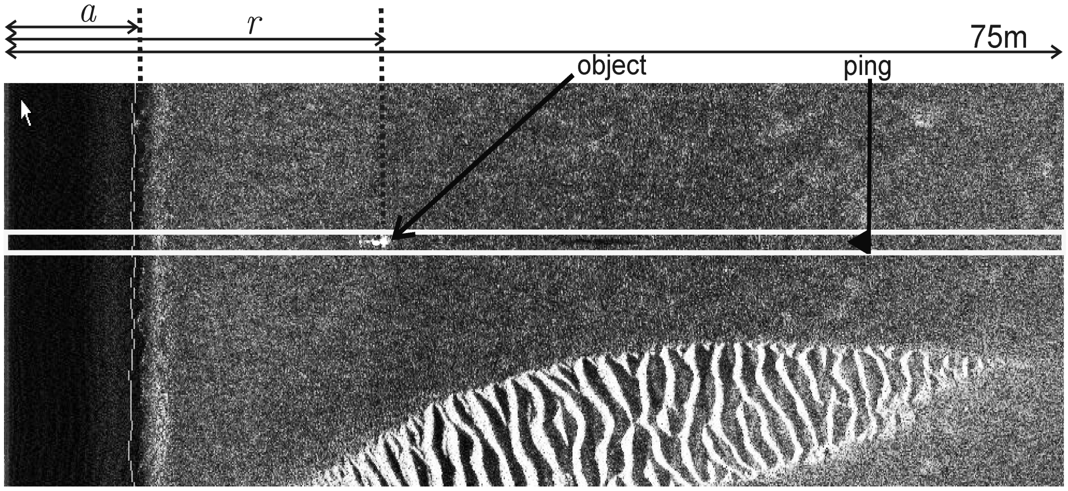

Fig. 3.1. The sonar image makes it possible to detect an object, to compute the distance r between the object and the robot, and the altitude a of the robot

gyrocompass, then the actual Euler angles for our robot should satisfy

$$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} \in \begin{pmatrix} \tilde{\phi} \\ \tilde{\theta} \\ \tilde{\psi} \end{pmatrix} + \begin{pmatrix} 1.75 \times 10^{-4}.\,[-1,1] \\ 1.75 \times 10^{-4}.\,[-1,1] \\ 5.27 \times 10^{-3}.\,[-1,1] \end{pmatrix}. \tag{3.4}$$

- **A barometer** is used to compute the depth of the robot (i.e., the distance between the robot and the surface of the ocean). If $\tilde{d}$ is the depth collected by the sensor, then the actual depth $p_z(t)$ of the robot satisfies $p_z(t) \in [-1.5, 1.5] + \tilde{d}.[0.98, 1.02]$. The interval $[-1.5, 1.5]$ may change depending on the strength of waves and tides.

3.2. Measurements

For each time $t \in \mathcal{T} \overset{\text{def}}{=} \{6000.0, 6000.1, 6000.2, \ldots, 11999.4\}$, the vector of measurements

$$\tilde{\mathbf{u}}(t) = \left(\tilde{\phi}(t), \tilde{\theta}(t), \tilde{\psi}(t), \tilde{v}_r^x(t), \tilde{v}_r^y(t), \tilde{v}_r^z(t), \tilde{a}(t), \tilde{d}(t)\right), \tag{3.5}$$

is collected. Using the characteristics of the sensors, it is possible to get a box $[\mathbf{u}(t)]$ which contains the actual value for the vector

$$\mathbf{u}(t) = \left(\phi(t), \theta(t), \psi(t), v_r^x(t), v_r^y(t), v_r^z(t), a(t), p_z(t)\right), \tag{3.6}$$

for each $t \in \mathcal{T}$.

Moreover, six objects have been detected manually from the sonar waterfall (i.e. the sonar image) collected by the robot. Table 3.1, provides (i) the number i of the ping where an object has been detected starboard, (ii) the corresponding time $\tau(i)$, (iii) the number $\sigma(i)$ of the detected object, and (iv) a measure $\tilde{r}(i)$ of the distance between the robot and the object. The actual distance $r(i)$ between the robot and the object for the ith ping is supposed to satisfy the relation

$$r(i) \in [\tilde{r}(i) - 1, \tilde{r}(i) + 1]. \tag{3.7}$$

Table 3.1. Measurements related to the objects detected by the sonar

i	0	1	2	3	4	5	6	7	8	9	10	11
$\tau(i)$	7054	7092	7374	7748	9038	9688	10024	10817	11172	11232	11279	11688
$\sigma(i)$	1	2	1	0	1	5	4	3	3	4	5	1
$\tilde{r}(i)$	52.42	12.47	54.40	52.68	27.73	26.98	37.90	36.71	37.37	31.03	33.51	15.05

4. Constraints

Around the zone covered by the robot, let us build the frame $(\mathbf{O}, \overrightarrow{\mathbf{i}}, \overrightarrow{\mathbf{j}}, \overrightarrow{\mathbf{k}})$ where $\mathbf{O}$ is the location of the robot at time $t_0 = 6000$s, the vector $\overrightarrow{\mathbf{i}}$ indicates the north, $\overrightarrow{\mathbf{j}}$ indicates the east and $\overrightarrow{\mathbf{k}}$ is oriented toward the center of the earth. Denote by $\mathbf{p} = (p_x, p_y, p_z)$ the coordinates of the robot expressed in the frame $(\mathbf{O}, \overrightarrow{\mathbf{i}}, \overrightarrow{\mathbf{j}}, \overrightarrow{\mathbf{k}})$. From the latitude and the longitude, given by the GPS, we can deduce the two first coordinates of the robot using the following relation:

$$\begin{pmatrix} p_x \\ p_y \end{pmatrix} = 111120 * \begin{pmatrix} 0 & 1 \\ \cos\left(\ell_y * \frac{\pi}{180}\right) & 0 \end{pmatrix} \begin{pmatrix} \ell_x - \ell_x^0 \\ \ell_y - \ell_y^0 \end{pmatrix}. \tag{4.1}$$

Moreover, the robot motion can be described by the following differential equation (also called state equation)

$$\dot{\mathbf{p}}(t) = \mathbf{R}(\phi(t), \theta(t), \psi(t)).\mathbf{v}_r(t), \tag{4.2}$$

where

$$\mathbf{R}(\phi, \theta, \psi) = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{pmatrix}$$

and $\mathbf{v}_r$ represents the speed of the robot measured by the Loch-Doppler sensor (see Equation (3.3)). Figure 4.1 gives an illustration of the meaning of the angles ψ, θ, ϕ. From the right to the left, we have (ψ, θ, ϕ) equal to $(0, 0, 0)$, $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$.

The state equation (4.2) can be interpreted as a constraint between the five functions $\dot{\mathbf{p}}(.), \phi(.), \theta(.), \psi(.)$ and $\mathbf{v}_r(.)$. Although this type of constraints could be handled inside a constraint propagation formalism [7, 8], for simplicity, we shall approximate this constraint between functions by a constraint between variables by resorting to a discretrization. This operation makes it possible to cast our problem into a classical CSP over continuous domains, but it removes the compleness of our approach. Since the sampling time is given by $\delta = 0.1$s, an Euler discretization of the state equation (4.2) yields

$$\mathbf{p}(t + 0.1) = \mathbf{p}(t) + 0.1 * \mathbf{R}(\phi(t), \theta(t), \psi(t)).\mathbf{v}_r(t). \tag{4.3}$$

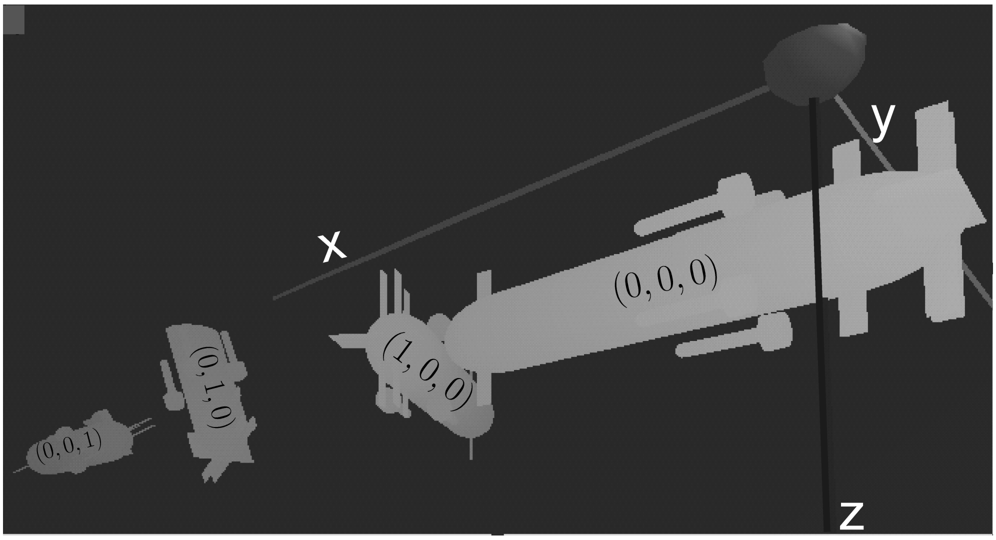

Fig. 4.1. The *Redermor* for different configurations (ψ, θ, ϕ)

When the ith object is detected at time $t = \tau(i)$ (see Table 3.1), it is located starboard of the robot and on a plane which is perpendicular to the robot axis (see Figure 4.2). The associated constraints are

$$\begin{cases} \text{(i)} & ||\mathbf{m}(\sigma(i)) - \mathbf{p}(t)|| \ = r(i) \\ \text{(ii)} & \mathbf{R}^{\mathrm{T}}(\phi(t), \theta(t), \psi(t)).\,(\mathbf{m}(\sigma(i)t) - \mathbf{p}(t)) \in [0,0] \times [0,\infty] \times [0,\infty] \\ \text{(iii)} & m_z(\sigma(i)) - p_z(t) - a(t) \in [-0.5, 0.5] \end{cases} \tag{4.4}$$

where, $\mathbf{m}(\sigma(i))$ represents the location of the ith object and $\mathbf{R}^{\mathrm{T}}(\phi, \theta, \psi).\,(\mathbf{m} - \mathbf{p})$ represents the vector $\mathbf{m} - \mathbf{p}$ expressed in the robot frame. In the constraint (ii), the first interval $[0, 0]$ means that the vector $\mathbf{m} - \mathbf{p}$ is perpendicular to the main axis of the robot, the second interval $[0, \infty]$ indicates that the object is starboard and the third interval $[0, \infty]$ indicated that the object is deeper than the robot itself. If we assume that the bottom of the ocean is flat, then we should have $m_z(\sigma(i)) = p_z(t) + a(t)$ (i.e., the depth m_z of the object lying on the bottom is equal to the altitude a of the robot plus the depth p_z of the robot). The constraint (iii) translates this relation with a small uncertainty represented by the interval $[-0.5, 0.5]$. This assumption is true if the slope of the (almost flat) bottom is limited to $\frac{0.5}{75} = 0.7\%$, which is true in the bottom the Douarnenez bay.

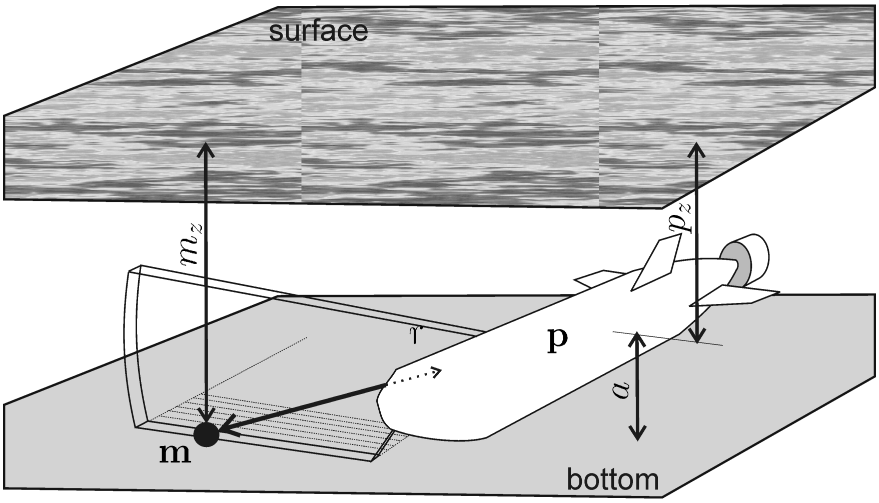

Fig. 4.2. The distance from the robot to the detected object can be obtained using a lateral sonar

5. Results

5.1. Constraints satisfaction problem

Our SLAM problem can be cast into the following constraints satisfaction problem.

$$\left\{\begin{array}{l}
t \in \{6000.0, 6000.1, 6000.2, \dots, 11999.4\}, \quad i \in \{0, 1, \dots, 11\}, \\[2ex]
\begin{pmatrix} p_x(t) \\ p_y(t) \end{pmatrix} = 111120. \begin{pmatrix} 0 & 1 \\ \cos\left(\ell_y(t) * \frac{\pi}{180}\right) & 0 \end{pmatrix} \begin{pmatrix} \ell_x(t) - \ell_x^0 \\ \ell_y(t) - \ell_y^0 \end{pmatrix}, \\[2ex]
\mathbf{p}(t) = (p_x(t), p_y(t), p_z(t)), \\[2ex]
\mathbf{R}_\psi(t) = \begin{pmatrix} \cos\psi(t) & -\sin\psi(t) & 0 \\ \sin\psi(t) & \cos\psi(t) & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{R}_\theta(t) = \begin{pmatrix} \cos\theta(t) & 0 & \sin\theta(t) \\ 0 & 1 & 0 \\ -\sin\theta(t) & 0 & \cos\theta(t) \end{pmatrix}, \\[2ex]
\mathbf{R}_\varphi(t) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi(t) & -\sin\varphi(t) \\ 0 & \sin\varphi(t) & \cos\varphi(t) \end{pmatrix}, \quad \mathbf{R}(t) = \mathbf{R}_\psi(t)\mathbf{R}_\theta(t)\mathbf{R}_\varphi(t), \\[2ex]
\mathbf{p}(t + 0.1) = \mathbf{p}(t) + 0.1 * \mathbf{R}(t).\mathbf{v}_r(t), \\[2ex]
||\mathbf{m}(\sigma(i)) - \mathbf{p}(\tau(i))|| \; = r(i), \\[2ex]
\mathbf{R}^{\mathrm{T}}(\tau(i))\,(\mathbf{m}(\sigma(i)) - \mathbf{p}(\tau(i))) \in [0, 0] \times [0, \infty] \times [0, \infty], \\[2ex]
m_z(\sigma(i)) - p_z(\tau(i)) - a(\tau(i)) \in [-0.5, 0.5]
\end{array}\right.$$

These constraints involve more than 300 000 variables (if a scalar decomposition of the vectors in performed). The sensors (GPS, sonar, ...) make it possible to get some accurate domains for the variables $\phi(t)$, $\theta(t)$, $\psi(t)$, $\mathbf{v}_r(t)$, $a(t)$, $p_z(t)$, $\ell_x(6000)$, $\ell_y(6000)$, $\ell_x(11999.4)$, $\ell_y(11999.4)$. The other variables $\ell_x(6000.1), \dots, \ell_x(11999.3)$, $\ell_y(6000.1), \dots, \ell_y(11999.3)$, $p_x(6000), \dots, p_x(11999.4)$, $p_y(6000), \dots, p_y(11999.4)$, $\mathbf{m}(0), \dots, \mathbf{m}(5)$ are unknown and the domains for their components should initially be instantiated to $[-\infty, \infty]$.

A constraints propagation procedure could thus be thought to contract all domains of our CCSP. Since we want to get accurate results, a scalar decomposition of the matrix constraints involved in our CSP is not recommended. Instead, we have developed efficient contraction algorithms associated to all our matrix constraints, such as $\mathbf{A} = \mathbf{B} * \mathbf{C}$, $||\mathbf{v}|| = r, \dots$. An illustration is given by the following example.

Example: To contract the constraint $\mathbf{R}(t) = \mathbf{R}_\psi(t)\mathbf{R}_\theta(t)\mathbf{R}_\varphi(t)$ involved in our CSP, we can take into account the fact that the matrices are all rotation matrices (i.e., their inverse is equal to their transpose). From this constraint, we can built other matrix constraints, as follows

$$\begin{aligned}\mathbf{R}(t) &= \mathbf{R}_\psi(t)\mathbf{R}_\theta(t)\mathbf{R}_\varphi(t) \Leftrightarrow \mathbf{R}(t)\mathbf{R}_\varphi^{\mathrm{T}}(t) = \mathbf{R}_\psi(t)\mathbf{R}_\theta(t) \\ &\Leftrightarrow \mathbf{R}(t)\mathbf{R}_\varphi^{\mathrm{T}}(t)\mathbf{R}_\theta^{\mathrm{T}}(t) = \mathbf{R}_\psi(t) \Leftrightarrow \dots .\end{aligned}$$

From all these generated redundant constraints; one can built contractors by decomposing them into scalar constraints and by using a hull consistency procedure. The resulting procedure constitutes an efficient contractor for the constraint $\mathbf{R}(t) = \mathbf{R}_\psi(t)\mathbf{R}_\theta(t)\mathbf{R}_\varphi(t)$.

5.2. Propagation

The results obtained by an elementary constraints propagation algorithm (similar to hull consistency) are illustrated by Figure 5.1. Subfigure (a) represents a punctual estimation of the trajectory of the robot. This estimation has been obtained by integrating the state equations (4.2) from the initial point (represented on lower part). We have also represented the 6 objects that have been dropped manually at the bottom of the ocean during the experiments. Note that we are not supposed to know the location of these six object. When we dropped them, we measured their location, but we used this information only to check the consistency of results obtained by the propagation. Subfigure (b) represents an envelope of the trajectory obtained using an interval integration, from a small initial box, obtained by the GPS at the beginning of the mission. In Subfigure (c) a final GPS point has also been considered and a forward-backward propagation has been performed up to equilibrium. In Figure (d) the constraints involving the object have been considered for the propagation. The envelope is now thinner and enveloping boxes containing the objects have also been obtained (see Subfigure (e)). We have checked that the actual positions for the objects (that have been measured independently during the experiments) all belong to the associated box, painted black. In Subfigure (f), a zooming perspective of the trajectory and the enveloping boxes for the detected objects have been represented. The computing time to get all these envelopes is less than one minute with a Pentium III. About ten forward-backward interval propagations have been performed to get the steady box of the CCSP. The C++ code associated with this example as well as a windows executable program can be downloaded at

`http://www.ensieta.fr/e3i2/Jaulin/redermorcp06.zip`

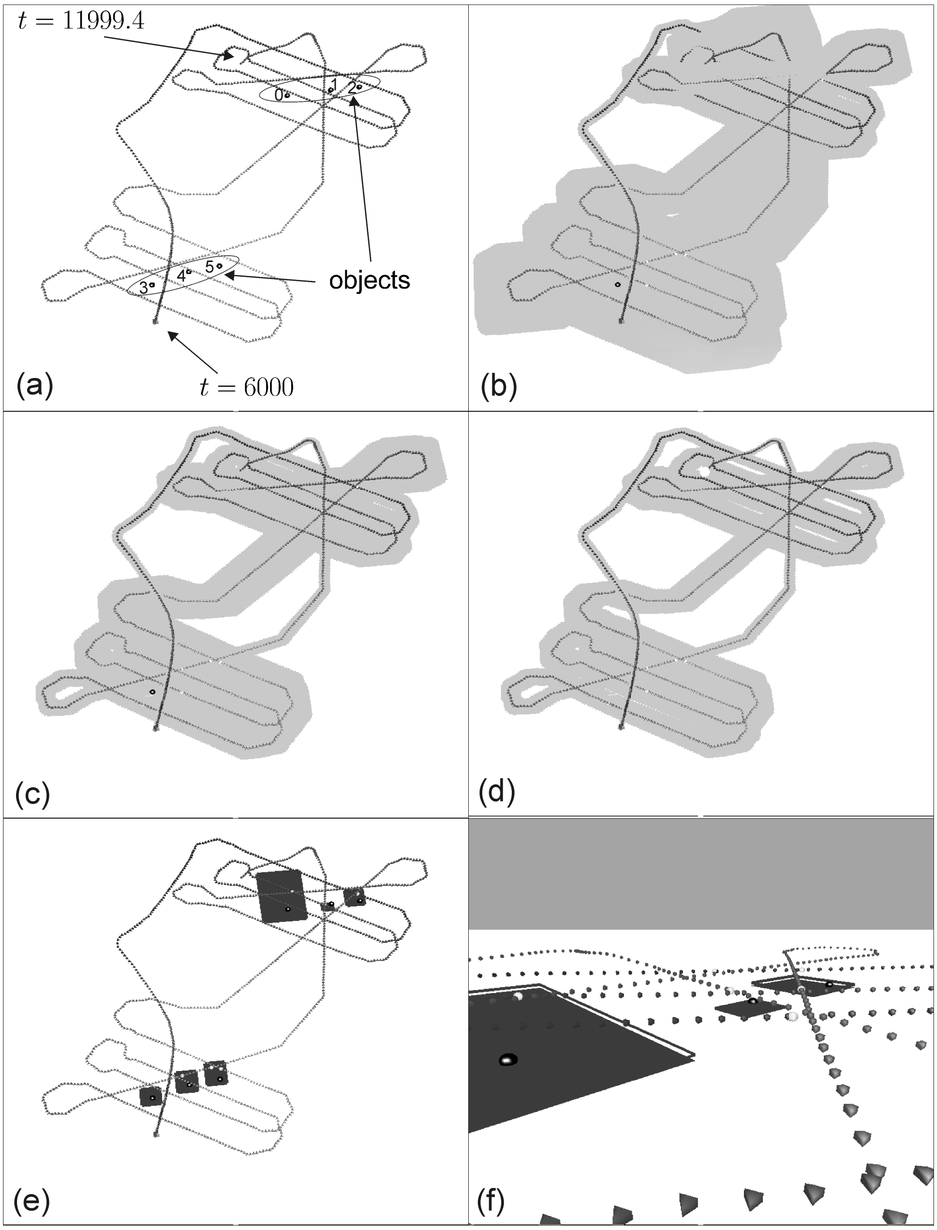

Fig. 5.1. Results obtained by our constraints propagation method

In the case where the position of the objects is approximately known, the SLAM problem translates into a state estimation problem. The structure of the CSP becomes a tree [9] and it is possible to get the global consistency with only one forward and one backward propagation. The envelope for the trajectory becomes very thin and a short computation time is needed. The capabilities of interval propagation methods for state estimation in a bounded error context have already been demonstrated in several applications (see e.g., [6], [3], [1] [9]). For the SLAM problem, the graph of the CSP is not a tree anymore. Of course, the number of cycles of the graph is rather limited, and a large part of the graph is made with one huge tree resulting from the state space equations. Because of these cycles, the global consistency cannot be reached without any bissection. Now, the number of variables of our CSP is huge and we should give up the idea of reaching the global consistency via bissections. On the other hand, redundant constraints can easily be obtained by adding other sensors or by handling the existing constraints, in a symbolic way. Adding redundant constraints could thus be a realistic way to control the accuracy of an interval contraints propagation method for the SLAM problem.

6. Conclusion

In this paper, we have shown that interval constraints propagation could be applied to solve SLAM problems. The efficiency of the approach has been demonstrated on an experiment made with an actual underwater robot (the *Redermor*). The experiment lasted two hours and involved thousands of data. if all assumptions on the bounds of the sensors, on the flat bottom, on the model of the robot, . . . are satisfied, then their exists always at least one solution of the CSP: that corresponding to the actual trajectory of the robot. Thus their is no need to prove the existence of a solution. Since the CSP has more variables than equations, the solution set contains generally a continuum of points.

When outliers occur during the experiment, our approach is not reliable anymore and one should take care about any false interpretation of the results. Consider now three different situation that should be known by any user of our approach for SLAM.

Situation 1. The solution set is empty and an empty set is returned by the propagation procedure. Our approach detects that their exists at least one outlier but it is not able to return any estimation of the trajectory and the positions of the objects. It is also not able to detect which sensor is responsible for the failure.

Situation 2. The solution set is empty but nonempty thin intervals for the variables are returned by the propagation. Our approach is not efficient enough to detect that outliers exist and we can wronly interpret that an accurate and guaranteed estimation of the trajectory of the robot has been done. Other more efficient algorithms could be able to prove that no solution exists which would lead us to the situation 1.

Situation 3. The solution set is not empty but it does not contain the actual trajectory of the robot. No method could be able to prove that outliers occur. Again, our approach could lead us to the false conclusion that a guaranteed estimation of the trajectory of the robot has been done, whereas, the robot might be somewhere else.

Now, for our experiment made on the Redermor, it is clear that outliers might be present. We have observed that when we corrupt some data volontarily (to create ouliers), the propagation method usually returns rapidly that no solution exists for our set of constraints. For our experiment with the data collected, we did not obtain an empty set. The only thing that we can conclude is that no outliers have been detected. The constraints propagation method can thus be seen a tool to validate (or unvalidate) reliability on models and sensor bounds.

References

1. X. Baguenard. "Propagation de contraintes sur les intervalles. Application à l'étalonnage des robots". PhD dissertation, Université d'Angers, Angers, France (2005). Available at: www.istia.univ-angers.fr/~baguenar/.
2. F. Benhamou, F. Goualard, L. Granvilliers, and J. F. Puget. Revising hull and box consistency. In "Proceedings of the International Conference on Logic Programming", pp. 230–244, Las Cruces, NM (1999).
3. P. Bouron. "Méthodes ensemblistes pour le diagnostic, l'estimation d'état et la fusion de données temporelles." PhD dissertation, Université de Compiègne, Compiègne, France (Juillet 2002).
4. N. Delanoue, L. Jaulin, and B. Cottenceau. Using interval arithmetic to prove that a set is path-connected. *Theoretical Computer Science, Special issue: Real Numbers and Computers* **351**(1), 119–128 (2006).
5. C. Drocourt, L. Delahoche, B. M. E. Brassart, and A. Clerentin. Incremental construction of the robot's environmental map using interval analysis. *Global Optimization and Constraint Satisfaction: Second International Workshop, COCOS 2003* **3478**, 127–141 (2005).
6. A. Gning. "Localisation garantie d'automobiles. Contribution aux techniques de satisfaction de contraintes sur les intervalles". PhD dissertation, Université de Technologie de Compiègne, Compiègne, France (2006).
7. M. Janssen, P. V. Hentenryck, and Y. Deville. A constraint satisfaction approach for enclosing solutions to parametric ordinary differential equations. *SIAM Journal on Numerical Analysis* **40**(5), 1896–1939 (2002).
8. L. Jaulin. Nonlinear bounded-error state estimation of continuous-time systems. *Automatica* **38**, 1079–1082 (2002).
9. L. Jaulin, M. Kieffer, O. Didrit, E. Walter. "Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics". Springer-Verlag, London (2001).
10. J. J. Leonard and H. F. Durrant-Whyte. Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research* **11**(4) (1992).
11. F. Lydoire and P. Poignet. Nonlinear predictive control using constraint satisfaction. In "In 2nd International Workshop on Global Constrained Optimization and Constraint Satisfaction (COCOS)", pp. 179–188 (2003).
12. J. Porta. Cuikslam: A kinematics-based approach to slam. In "Proceedings of the 2005 IEEE International Conference on Robotics and Automation", pp. 2436–2442, Barcelona (Spain) (2005).
13. R. Raissi, N. Ramdani, and Y. Candau. Set membership state and parameter estimation for systems described by nonlinear differential equations. *Automatica* **40**, 1771–1777 (2004).
14. D. Sam-Haroud. "Constraint consistency techniques for continuous domains". PhD dissertation 1423, Swiss Federal Institute of Technology in Lausanne, Switzerland (1995).
15. M. van Emden. Algorithmic power from declarative use of redundant constraints. *Constraints* **4**(4), 363–381 (1999).
16. P. van Hentenryck, Y. Deville, and L. Michel. "Numerica: A Modeling Language for Global Optimization". MIT Press, Boston, MA (1997).
17. P. H. Vinas, M. A. Sainz, J. Vehi, and L. Jaulin. Quantified set inversion algorithm with applications to control. *Reliable computing* **11**(5), 369–382 (2006).