GESMI, un logiciel pour l'aide à localisation de mines sous-marines

Luc Jaulin[1], Michel Legris[1], Frédéric Dabe[2]

[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. *Dans cet article, nous présentons le logiciel GESMI (Guaranteed Estimation of Sea Mines with Intervals) qui a été conçu pour traiter les données récupérées par Redermor, robot sous-marin chasseur de mines. Ce logiciel doit aider un opérateur humain à localiser des mines sur une carte afin qu'on puisse y envoyer un robot démineur. Une approche à erreurs bornées a été choisie et les techniques de résolution utilisent l'analyse par intervalles et la propagation de contraintes. Les expériences traitées dans cet article ont été faites avec le GESMA (Groupe d'Etude Sous-Marine de l'Atlantique) dans la baie de Douarnenez en Bretagne.*

1. Introduction

Dans cet article, nous allons présenter le logiciel GESMI (Guaranteed Estimation of Sea Mines with Intervals). Ce logiciel a pour vocation d'aider un opérateur humain à effectuer la localisation de mines, à partir de données récupérées par un robot sous-marin. Cette étude a été proposée par le GESMA et est liée au PEA REA discret et vise à valider le géo-référencement des données acquises. Il semble clair que la localisation des mines ne peut se faire sans la localisation du robot lui même. Le problème que nous devons donc résoudre est un problème de type SLAM (*Simultaneous localization and map building*) (voir par exemple [9]).

Dans cet article, nous allons considérer une approche ensembliste pour le SLAM [11] et nous allons montrer que le problème se ramène à un CSP (*constraints satisfaction problem*) qui peut être résolu efficacement par l'utilisation de techniques de propagation de contraintes sur les intervalles [8], [6]. Nous allons illustrer l'efficacité de l'approche sur des données réelles collectées par le robot *Redermor* (qui signifie *coureur des mers* en Breton) du GESMA. Plus de détails sur cette expérience et sur la méthode de résolution peuvent être trouvées dans un article [7] que nous avons soumis à ICRA.

Pour cette étude, nous nous sommes inspirés de l'article de Drocourt et. al. [4] et de celui de Porta [12] où les auteurs utilisent l'analyse par intervalles dans le contexte du SLAM pour des robots roulants. Ici, l'approche est rendue plus efficace par l'ajout de techniques de propagation qui n'ont jamais été utilisées dans ce contexte. Notons qu'il existe de nombreuses autres applications où les techniques de propagation de contraintes sur les intervalles se sont avérées efficaces dans le contexte de l'estimation (voir par exemple, [1] pour la calibration de robots, [13] pour l'estimation d'état, [10], [14] pour l'estimation dans le contexte de la commande, [3] pour l'étude de la connexité des ensembles de vraisemblance, ...).

2. Le Redermor

Le robot pour lequel le logiciel GESMI a été développé est représenté sur les figures 2.1 et 2.2. Il s'agit d'un robot sous-marin entièrement autonome. Ce robot, conçu et réalisé par le GESMA (Groupe d'Etude Sous-Marine de l'Atlantique), a une longueur de 6 m, un diamètre de 1 m et un poids de 3800 Kg. Il possède un système de propulsion et de commande très performant, dans le but de rechercher des mines dans les fonds de l'océan.

3. Capteurs

Le robot est équipé de nombreux capteurs que nous allons maintenant décrire.

- **Un GPS** (Global Positioning System). Du fait que les ondes électromagnétiques (ici autour de 1.2 MHz), ne se propagent pas sous l'eau, le GPS n'est opérationnel que lorsque le robot est à la surface de l'océan. Pendant une expérience de plongée (typiquement d'une durée de deux heures) du Redermor, le robot n'est capable de se localiser par GPS qu'au moment où il plonge et au moment où il réapparaît à la surface. Pour l'expérience sur laquelle nous allons travailler dans le cadre de cet article, nous avons pu mesurer qu'à l'instant $t_0 = 6000$ sec, le robot avait plongé à une position $\ell^0 = (\ell^0_x, \ell^0_y) = (-4.458227931^{\circ}, 48.212920614^{\circ})$,

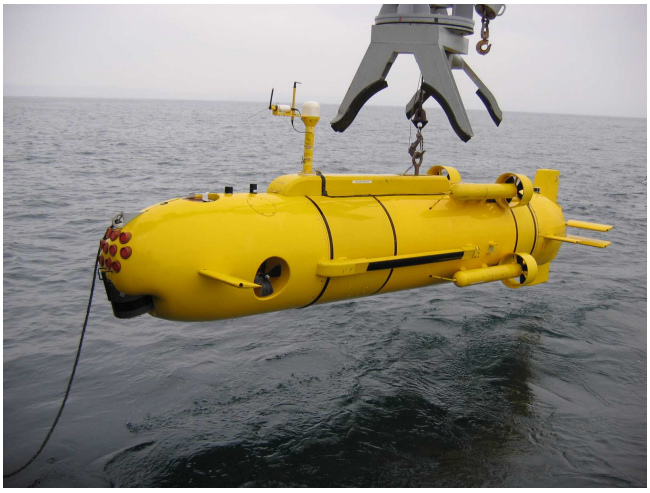

Fig. 2.1. Le *Redermor*, construit par le GESMA (Groupe d'Etude Sous-Marine de l'Atlantique), juste avant d'être mis à l'eau

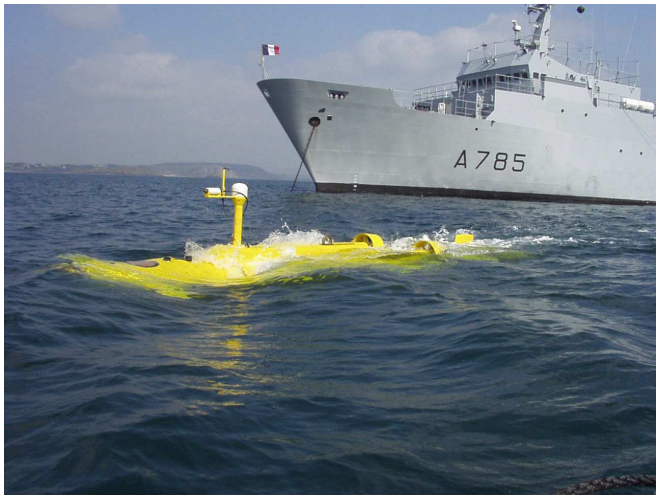

Fig. 2.2. Le *Redermor* à la surface de l'eau, encore proche du bateau d'où il vient d'être lancé.

où ℓ_x^0 est la longitude et ℓ_y^0 la latitude, l'erreur étant inférieure à ± 2.5 m. Lorsque le robot revient à la surface au temps $t_f = 11999.4$ sec, sa position est (toujours avec une précision de 2.5 mètres) donnée par $\ell^f = (\ell_x^f, \ell_y^f) = (-4.454660760^o, 48.219129760^o)$.

– **Un Loch-Doppler.** Ce type de capteur permet de calculer la vitesse du robot $\mathbf{v}_r$. Un Loch-Doppler envoie des ultrasons qui sont réfléchis au fond de l'océan. Le capteur est capable d'estimer la vitesse du robot en utilisant l'effet Doppler. Pour les fréquences que nous avons choisies, (autour de 300 kHz) la vitesse du robot vérifie la relation d'appartenance

$$\mathbf{v}_r \in \tilde{\mathbf{v}}_r + 0.004 * [-1, 1] . \tilde{\mathbf{v}}_r + 0.004 * [-1, 1], \quad (3.1)$$

où $\tilde{\mathbf{v}}_r$ est la vitesse mesurée par le capteur. De plus, le Loch-Doppler est capable de calculer l'altitude a du robot avec une erreur inférieure à 10cm.

– **Une centrale inertielle** (Octans III de la société IXSEA). Ce capteur utilise l'effet Sagnac (dans une fibre optique circulaire tournant sur elle même, le temps que met la lumière pour faire un tour complet dépend du sens de parcours) et la rotation de la terre pour en déduire la direction de l'axe nord-sud. La connaissance de cette direction nous donne deux équations impliquant les angles d'Euler, qui sont le roulis ϕ, le tangage θ et le cap ψ du robot, exprimé dans un repère local. Grâce à l'accéléromètre inclus dans la centrale, il est possible d'en déduire le vecteur de gravité et de générer ainsi une équation supplémentaire qui permettra de calculer les trois angles d'Euler. Si nous appelons $\tilde{\phi}, \tilde{\theta}, \tilde{\psi}$, les trois angles d'Euler retournés par la centrale, les vrais angles d'Euler satisfont

$$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} \in \begin{pmatrix} \tilde{\phi} \\ \tilde{\theta} \\ \tilde{\psi} \end{pmatrix} + \begin{pmatrix} 1.75 \times 10^{-4}.[-1, 1] \\ 1.75 \times 10^{-4}.[-1, 1] \\ 5.27 \times 10^{-3}.[-1, 1] \end{pmatrix}. \quad (3.2)$$

d'après le constructeur de la centrale. Notons que la connaissance du vecteur de gravité et de l'axe de rotation par la centrale permet théoriquement, par un simple produit scalaire, de retrouver la latitude du robot. Cependant, la précision obtenue est trop faible pour que nous puissions la prendre en compte pour notre localisation.

– **Un baromètre**. Il mesure la profondeur du robot. Si $\tilde{d}$ est la profondeur mesurée, alors la profondeur réelle $p_z(t)$ du robot satisfait $p_z(t) \in [-1.5, 1.5] + \tilde{d}.[0.98, 1.02]$. L'intervalle $[-1.5, 1.5]$ dépend de la force des vagues et de celle des marées.

Pour notre expérience, pour chaque

$$t \in \mathcal{T} \stackrel{\text{def}}{=} \{6000.0, 6000.1, 6000.2, \ldots, 11999.4\},$$

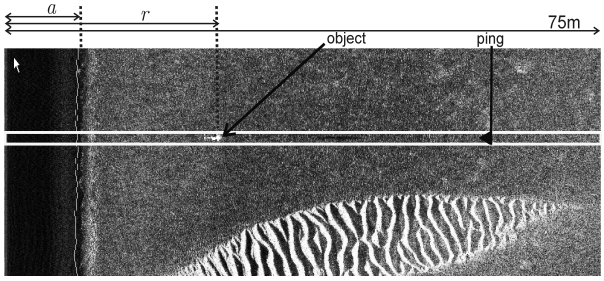

Fig. 3.1. Tranche de la cascade qui permet de détecter une mine et de calculer sa distance au robot

le vecteur des mesures

$$\tilde{\mathbf{u}}(t) = \left(\tilde{\phi}(t), \tilde{\theta}(t), \tilde{\psi}(t), \tilde{v}_r^x(t), \tilde{v}_r^y(t), \tilde{v}_r^z(t), \tilde{a}(t), \tilde{d}(t)\right), \tag{3.3}$$

est disponible. En utilisant les caractéristique des capteurs, nous pouvons obtenir une boite $[\mathbf{u}(t)]$ qui contient les vraies valeurs pour le vecteur

$$\mathbf{u}(t) = (\phi(t), \theta(t), \psi(t), v_r^x(t), v_r^y(t), v_r^z(t), a(t), p_z(t)). \tag{3.4}$$

Nous disposons aussi d'informations d'un autre type que celles que nous venons de décrire. Il s'agit des données récupérées par le sonar (KLEIN 5400 side scan sonar) et placé à tribord du robot. Le sonar envoie des ultrasons toutes les 0.1 seconde dans un plan très fin qui lui est orthogonal. Il récupère alors l'écho (en dessous d'une longueur de 75 mètres) afin de construire une image en forme de ruban dont la figure 3.1 représente une tranche. Cette image est appelée *cascade*. Elle est large de 75m et haute de 10 km (ce qui correspond à la longueur parcourue par le Redermor pendant sa mission). Après la mission la cascade est déroulée et analysée par un opérateur humain qui peut alors faire une estimation $\tilde{r}(t)$ de la distance $r(t)$ entre le robot et la mine à l'instant de détection t. De plus, à partir de la bande verticale noire (noir : car aucun écho n'est détecté) sur la gauche de la cascade (appelée *colonne d'eau*), il est aussi capable d'en déduire une estimation $\tilde{a}(t)$ de l'altitude $a(t)$ du robot. La figure 3.1 représente la détection de la cinquième mine dans le cadre de notre expérience. Le ping associé est représenté par le rectangle blanc fin. Sur la cascade délivrée par le Redermor après sa mission, six mines $\mathbf{m}_1, \ldots, \mathbf{m}_6$ ont été détectées manuellement, à l'aide du logiciel SonarPro, dont la figure 3.2 représente un screenshot.

La table ci-dessous décrit (i) le temps t pour lequel une mine a été detectée à tribord, (ii) le numéro i de la mine associée et (iii) une mesure $\tilde{r}_i(t)$ de la distance entre le robot et la mine. La distance réelle $r_i(t)$ entre le robot et la mine $\mathbf{m}_i$ est censée satisfaire la relation

$$r_i(t) \in [\tilde{r}_i(t) - 1, \tilde{r}_i(t) + 1]. \tag{3.5}$$

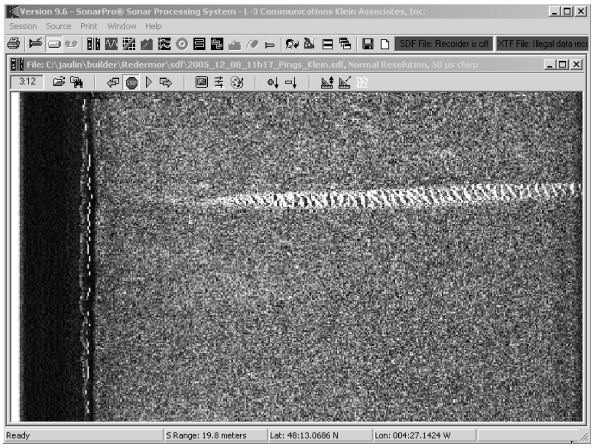

Fig. 3.2. Screenshot du logiciel SonarPro

t	7054	7092	7374	7748	9038	9688	...
i	1	2	1	0	1	5	...
$\tilde{r}_i(t)$	52.42	12.47	54.40	52.68	27.73	26.98	...

t	...	10024	10817	11172	11232	11279	11688
i	...	4	3	3	4	5	1
$\tilde{r}_i(t)$	...	37.90	36.71	37.37	31.03	33.51	15.05

Table de détection des mises. Elle donne une mesure de la distance
entre les mines et le robot. Elle a été obtenue par un opérateur humain
après analyse de l'image générée par le sonar, grâce au logiciel SonarPro

4. Contraintes

Sur la zone parcourue par le robot, construisons un repère local $(\mathbf{O}, \mathbf{i}, \mathbf{j}, \mathbf{k})$ où $\mathbf{O}$ est la position du robot à l'instant initial $t_0 = 6000$s, le vecteur $\mathbf{i}$ indique le nord, $\mathbf{j}$ indique l'est et $\mathbf{k}$ est orienté vers le centre de la terre. Soit $\mathbf{p} = (p_x, p_y, p_z)$ les coordonnées du robot exprimée dans le repère $(\mathbf{O}, \mathbf{i}, \mathbf{j}, \mathbf{k})$. A partir de la latitude et de la longitude, données par le GPS, nous pouvons en déduire les deux premières coordonnées du robot, exprimées dans le repère local, en utilisant la relation suivante :

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

De plus, le mouvement du robot peut être décrit par l'équation d'état

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

où

$$\mathbf{R}(\phi,\theta,\psi) = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} * \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{pmatrix} \quad (4.3)$$

et $\mathbf{v}_r$ représente la vitesse du Redermor mesurée par le Loch-Doppler (voir équation (3.1)). Définissons l'ensemble $\mathcal{M}$ des (t,i) tels que la $i$ième mine ait été détectée à l'instant t (voir la table de détection des mines) par l'opérateur humain. A partir de la table, nous obtenons

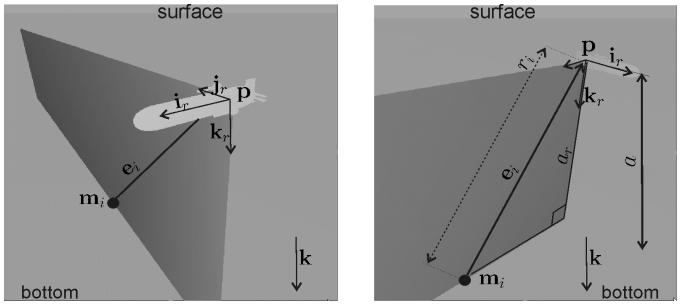

Fig. 4.1. La distance entre le robot et une mine peut être mesurée par un sonar latéral. Le logiciel GESMI a été utilisé pour de rendu 3D de cette figure.

$$\mathcal{M} = \{(7054,1),(7092,2),(7373,1),\ldots,(11279,5),(11688,1)\}.$$

Lorsque la $i$ième mine $\mathbf{m}_i$ a été détectée, elle est forcément à tribord du robot, sous ce dernier et dans un plan qui lui est perpendiculaire (voir figure 4.1). Définissons le vecteur

$$\mathbf{e}_i(t) = \mathbf{p}(t) - \mathbf{m}_i. \quad (4.4)$$

et

$$r_i(t) = \begin{cases} \|\mathbf{e}_i(t)\| \text{ et } \mathbf{R}^{\mathrm{T}}(t).\mathbf{e}_i(t) \in [0,0]\times[-\infty,0]\times[-\infty,0] \\ \infty \text{ sinon} \end{cases} \quad (4.5)$$

Nous avons donc les contraintes suivantes

$$\begin{cases} \text{(i)} & \|\mathbf{e}_i(t)\| = r_i(t) \text{ si } (t,i)\in\mathcal{M} \\ \text{(ii)} & a(t) = \mathbf{k}^{\mathrm{T}}.\mathbf{R}.\mathbf{k}.a_r(t) \\ \text{(iii)} & \mathbf{R}^{\mathrm{T}}(t)\mathbf{e}_i(t) = \left(0, -\sqrt{r_i^2(t)-a_r^2(t)}, -a_r^2(t)\right) \text{ si } (t,i)\in\mathcal{M} \\ \text{(iv)} & \mathbf{e}_i^{\mathrm{T}}(t).\mathbf{k} + a(t) = 0 \text{ si } (t,i)\in\mathcal{M}. \end{cases} \quad (4.6)$$

L'égalité (i) signifie que lorsque la mine $\mathbf{m}_i$ est détectée, la distance mesurée $r_i(t)$ sur la cascade correspond à la norme du vecteur $\mathbf{e}_i(t)$. Dans l'égalité (ii), $a_r(t)$ représente la distance entre le robot et le fond, en suivant la direction $\mathbf{k}_r$ (voir figure 4.1). L'égalité (iii) est obtenue en utilisant le théorème de Pythagore qui permet d'obtenir une expression de $\mathbf{e}_i(t)$ dans le repère du robot $(\mathbf{p},\mathbf{i}_r,\mathbf{j}_r,\mathbf{k}_r)$. La dernière équation est obtenue en supposant que le fond de l'océan est plat (au moins localement).

Puisque $\dot{\mathbf{p}}(t) = \mathbf{R}(t).\mathbf{v}_r(t)$ (voir équation (4.2)), nous avons

$$\dot{\mathbf{e}}_i(t) = \mathbf{R}(t).\mathbf{v}_r(t). \quad (4.7)$$

5. Propagation de contraintes

Notre problème de localisation de mines peut se traduire en un ensemble d'équations données (appelé aussi CSP : Constraint Satisfaction Problem) ci-dessous.

$$\begin{cases} \forall (t,i)\in\mathcal{M} = \{(7054,1),(7092,2),(7373,1),\ldots \\ \qquad\qquad \ldots,(11279,5),(11688,1)\}. \\ \|\mathbf{e}_i(t)\| = r_i(t), \\ a(t) = \mathbf{k}^{\mathrm{T}}.\mathbf{R}(t).\mathbf{k}.a_r(t) \\ \mathbf{R}^{\mathrm{T}}(t)\mathbf{e}_i(t) = \left(0, -\sqrt{r_i^2(t)-a_r^2(t)}, -a_r^2(t)\right) \\ \mathbf{e}_i^{\mathrm{T}}(t).\mathbf{k} + a(t) = 0 \end{cases}$$

$$\begin{cases} \forall t\in[6000.0, 11999.4], \quad \forall i\in\{0,1,\ldots,5\} \\ \begin{pmatrix} p_x(t) \\ p_y(t) \end{pmatrix} = 111120. \begin{pmatrix} 0 & 1 \\ \cos\left(\ell_y(t) * \frac{\pi}{180}\right) & 0 \end{pmatrix} \begin{pmatrix} \ell_x(t)-\ell_x^0 \\ \ell_y(t)-\ell_y^0 \end{pmatrix}, \\ \mathbf{R}_\psi(t) = \begin{pmatrix} \cos\psi(t) & -\sin\psi(t) & 0 \\ \sin\psi(t) & \cos\psi(t) & 0 \\ 0 & 0 & 1 \end{pmatrix}, \\ \mathbf{R}_\theta(t) = \begin{pmatrix} \cos\theta(t) & 0 & \sin\theta(t) \\ 0 & 1 & 0 \\ -\sin\theta(t) & 0 & \cos\theta(t) \end{pmatrix}, \\ \mathbf{R}_\varphi(t) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi(t) & -\sin\varphi(t) \\ 0 & \sin\varphi(t) & \cos\varphi(t) \end{pmatrix}, \\ \mathbf{R}(t) = \mathbf{R}_\psi(t)\mathbf{R}_\theta(t)\mathbf{R}_\varphi(t), \\ \dot{\mathbf{p}}(t) = \mathbf{R}(t).\mathbf{v}_r(t), \quad \dot{\mathbf{e}}_i(t) = \mathbf{R}(t).\mathbf{v}_r(t), \\ r_i(t) = \begin{cases} \|\mathbf{e}_i(t)\| \text{ si } \mathbf{R}^{\mathrm{T}}(t).\mathbf{e}_i(t)\in[0,0]\times[-\infty,0]^{\times 2} \\ \infty \text{ sinon} \end{cases} \end{cases}$$

Expliquons maintenant le principe des techniques de propagation de contraintes. Chaque équation matricielle est tout d'abord décomposée en contraintes scalaires. Par exemple la contrainte $\mathbf{y} = \mathbf{A}\mathbf{x}$, $\mathbf{y}\in\mathbb{R}^2$, $\mathbf{x}\in\mathbb{R}^2$ se décompose en deux contraintes scalaires :

$$\begin{cases} y_1 = a_{11}x_1 + a_{12}x_2 \\ y_2 = a_{21}x_1 + a_{22}x_2 \end{cases} \quad (5.1)$$

Chaque contrainte scalaire est décomposée en contraintes primitives (c'est-à-dire non décomposable). Par exemple, la première contrainte $y_1 = a_{11}x_1 + a_{12}x_2$ se décompose en trois contraintes primitives :

$$z_1 = a_{11}x_1,\ z_2 = a_{12}x_2,\ y_1 = z_1 + z_2. \quad (5.2)$$

Nous affectons à chaque variable un domaine de $\mathbb{R}$ la contenant. Ce domaine peut être égal à $[-\infty, \infty]$ si nous n'avons aucune information de nature ensembliste sur la variable. On considère alors les contraintes primitives les une après les autres et l'on cherche à contracter les domaines pour chacune des variables en utilisant les techniques de l'analyse par intervalles (voir [8], pour plus de détails) et ceci jusqu'à ce qu'aucune contraction ne soit possible sur tout l'ensemble de contraintes primitives. Afin d'illustrer le principe de la contraction d'une contrainte primitive, considérons la contrainte primitive à trois variables $z = x + y$. Supposons que les domaines pour les variables soient $x \in [1, 5], y \in [2, 4], z \in [6, \infty]$, le raisonnement pour la contraction des domaines est décrit ci-dessous

$$\begin{aligned} z = x + y \Rightarrow z &\in [6, \infty] \cap ([1, 5] + [2, 4]) \\ &= [6, \infty] \cap [3, 9] = [6, 9]. \\ x = z - y \Rightarrow x &\in [1, 5] \cap ([6, \infty] - [2, 4]) \\ &= [1, 5] \cap [2, \infty] = [2, 5]. \\ y = z - x \Rightarrow y &\in [2, 4] \cap ([6, \infty] - [1, 5]) \\ &= [2, 4] \cap [1, \infty] = [2, 4]. \end{aligned} \tag{5.3}$$

Ainsi, nous avons donc $x \in [2, 5]$, $y \in [2, 4]$, $z \in [6, 9]$, ce qui est plus précis que les relations d'appartenance initiales : $x \in [1, 5]$, $y \in [2, 4]$, $z \in [6, \infty]$.

6. GESMI

Le logiciel GESMI a été programmé en C++ Builder. Il réalise le traitement de données décrit ci-dessus. Il admet pour entrée un fichier texte qui contient des données des capteurs (GPS, si le robot est à la surface, les vitesses données par le Loch Doppler, l'altitude du robot, sa profondeur, les accélérations et les angles d'Euler). Comme GESMI est dédié au Redermor pour lequel les capteurs sont connus, les précisions sont fixées une fois pour toute dans le programme et n'apparaissent donc pas dans le fichier de données. Dans le fichier texte apparaissent aussi les données de la table de détection des mines.

GESMI propose une interface graphique 3D conviviale (voir figure 6.1) pour visualiser les résultats (trajectoire estimée pour l'AUV, enveloppe pour la trajectoire, pavés pour les mines; ...). La partie graphique a été faite grâce à la bibliothèque OpenGl. Une simulation des déplacements du robot peut être lancée. Le robot est dessiné avec beaucoup de réalisme grâce une exportation vers OpenGl des plans AutoCad qui ont servi à sa construction. GESMI génère deux fichiers images (.bmp). Une première image (figure 6.2) correspond à la carte et permet une visualisation précise de la trajectoire reconstituée ainsi qu'une estimation ensembliste pour la position des mines. Une deuxième image (figure 6.3) représente la cascade reconstruite par GESMI. Elle permet à l'opérateur humain de retrouver facilement sur la cascade des mines qui lui auraient échappé.

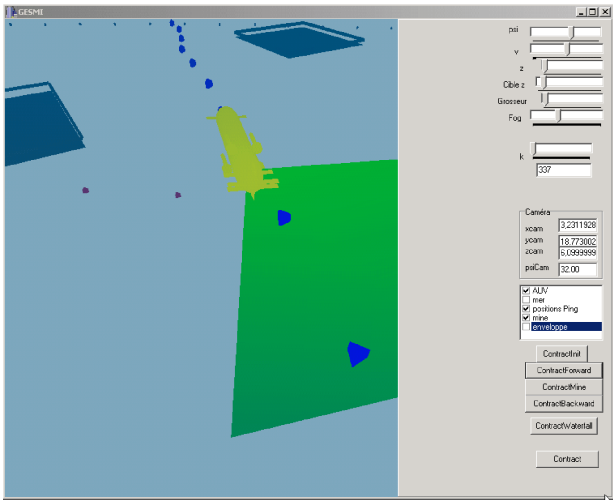

Fig. 6.1. Logiciel GESMI, pour la localisation de mines

7. Résultats

Pour notre expérience, les résultats obtenus par GESMI sont illustrés sur la figure 7.1 et se traduisent mathématiquement par un pavé (ou vecteur d'intervalles) de très grande dimension. La figure (a) représente l'enveloppe enfermant la trajectoire du robot. Cette enveloppe résulte d'une fusion des informations de différents capteurs, à savoir le GPS (disponible seulement en début et en fin de mission), de l'image sonar qui permet le repérage des mines, du Loch Doppler pour la mesure des vitesses, de la centrale inertielle pour la mesure des angles et du baromètre pour la mesure de la profondeur. Nous avons observé que si nous omettions de prendre en compte certaines mines, nous obtenions des enveloppes sensiblement plus larges, ce qui semble, bien sûr, logique. La figure (b) représente six pavés censés contenir les six mines repérées sur la cascade. Comme ces six mines ont été déposées par nos soins, au fond de l'océan, nous savons avec une bonne précision où elles se trouvent. Nous avons ainsi pu vérifier a posteriori qu'effectivement, elles se trouvaient bien dans ces six pavés. Le temps de calcul demandé par GESMI pour effectuer tous ces calculs est de l'ordre de 30 secondes sur un Pentium III. Les pavés censés contenir les mines ont une longueur inférieure à 30m. Cette précision est tout à fait acceptable pour une opération de déminage, qui pourra être faite soit par un plongeur, soit un robot démineur.

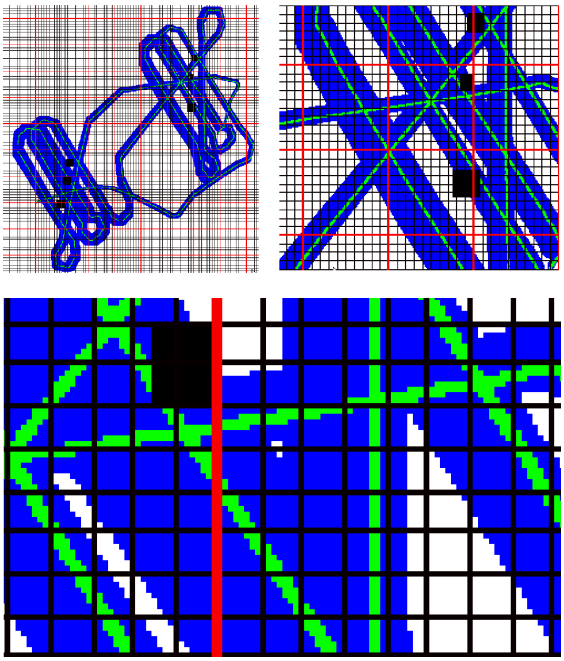

Fig. 6.2. Carte reconstituée par GESMI; chaque petit carré possède une longueur de 10m

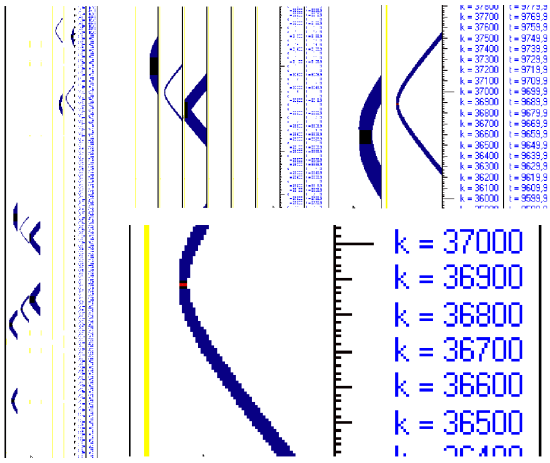

Fig. 6.3. Cascade reconstituée construite par GESMI

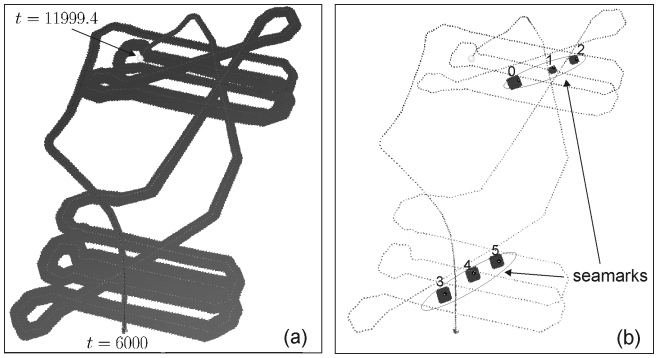

Fig. 7.1. (a) Enveloppe enfermant la trajectoire pour le robot, (b) Pavés contenant les six mines déposées par nos soins

Notons que dans le cas particulier où la position des mines est connue, le problème SLAM que nous venons de traiter se ramène à un problème d'estimation d'état, dans un contexte à erreurs bornées. L'enveloppe devient extrêmement fine et le temps de calcul demandé par GESMI est de l'ordre de 5 secondes. Notons de l'efficacité avec laquelle les méthodes de propagation de contraintes parviennent à résoudre le problème d'estimation d'état avait déjà été observée par de nombreux auteurs (voir par exemple, [5], [2], [1], [8]).

La figure 7.2 représente la cascade (complète en haut, partielle en bas) reconstituée par GESMI dans le cadre de notre expérience. Chaque ligne correspond à une des six mines ($i = 0, \ldots, 5$) qui ont été détectées. Les zones grises contiennent l'ensemble de tous les couples $(t, \|\mathbf{p} - \mathbf{m}_i\|)$ possibles, ensemble que l'on appelle couramment les *hyperboles de migration*. Les douze petits disques noirs correspondent aux mines détectées par l'opérateur humain. A partir de chacun de ces disques, nous pouvons reconstituer l'instant t auquel la mine a été détectée (en abscisse), le numéro de la mine (correspondant au numéro de la ligne), et la distance r_i entre le robot et la mine (en ordonnée). Donc, à partir de tous ces petits disques noirs, nous sommes capables de reconstruire la table de détection des mines. Les zones noires correspondent à tous les couples (t, r_i) possibles. Certaines de ces zones sont minuscules et sont recouvertes par les petits disques noirs. D'autres sont plus larges et ne contiennent aucun disque noir. Il s'agit généralement de mines qui ont été manquées par l'opérateur lors du balayage visuel de la cascade. Ainsi, avec l'aide de la figure 7.2, l'opérateur pourrait retrouver très facilement quelques une de ces mines non détectées. Si tel est le cas, un nouvel appel de GESMI aurait pour conséquence la génération d'une enveloppe encore plus fine pour la trajectoire, une localisation des mines encore plus précise, une cascade avec des zones noires encore plus petite et donc à nouveau une plus forte probabilité de détecter de nouvelles mines sur la cascade, ... Ainsi,

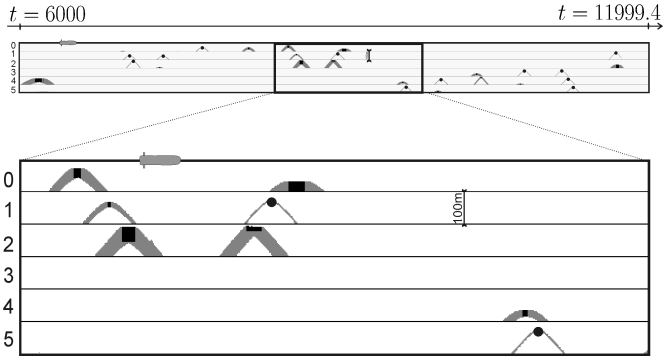

Fig. 7.2. Cascades reconstituées par GESMI dans le but d'aider l'opérateur à détecter manuellement de nouvelles mines

l'opérateur humain peut être considéré comme un contracteur ([8]) à l'intérieur d'un processus de propagation de contraintes.

8. Conclusion

Dans cet article, nous présenté un nouveau logiciel GESMI capable de nous aider pour la localisation de mines sous marines à partir de données récupérées par un robot sous-marin. Nous avons ainsi montré l'efficacité des méthodes de propagation de contraintes sur un problème de type SLAM. Les performances de GESMI ont été validées sur une expérience réelle impliquant un très grand nombre de données obtenues par le Redermor lors d'une mission qui a duré environ deux heures. Le temps de calcul très faible (environ 30 secondes pour deux heures d'expérience), laisse penser une possible application de GESMI pour du traitement en temps réel. Si toutes les hypothèses sur les bornes des capteurs, sur le fond plat, sur la hauteur des vagues, sur le modèle du robot, ... sont correctes alors, il existe toujours au moins une solution à notre problème : celle correspondant à notre expérience.

Lorsque des données aberrantes sont censées pouvoir apparaitre, notre approche n'est plus capable de donner des résultats garantis, non pas que notre traitement ne soit pas garanti, mais parce que les hypothèses de garantie des résultats nécessitent à la fois une garantie du traitement (c'est le cas ici) mais aussi une garantie des hypothèses sur les bornes et les contraintes (cela n'est pas le cas en pratique). Considérons trois situations pessimistes différentes qui doivent être connues de tout utilisateur de GESMI afin de se prémunir contre toute mauvaise interprétation des résultats.

Situation 1. Le pavé (contenant l'ensemble solution) renvoyé par GESMI est vide. GESMI rapporte donc l'existence d'au moins une donnée aberrante mais il n'est pas capable se retourner une enveloppe pour la trajectoire ni une estimation pour la position des mines. Il n'est pas non plus capable de dire quel capteur a été à l'origine de cet échec.

Situation 2. L'ensemble solution est vide, mais GESMI, qui est conservatif, renvoie un ensemble solution non vide qui nous semble précis. GESMI n'est pas assez efficace pour détecter l'existence d'une donnée aberrante et l'utilisateur peut interpréter, à tort, qu'il dispose d'une localisation précise des mines. Des algorithmes plus efficaces auraient peut être pu détecter l'absence de solution, ce qui nous aurait ramené à la situation 1.

Situation 3. L'ensemble solution est non-vide, mais, du fait de quelques données aberrantes, il ne contient pas la trajectoire réelle pour le robot, la position réelle des mines, ... Aucune autre méthode ne pourrait montrer la présence de données aberrantes. Encore une fois, GESMI pourrait nous amener à conclure qu'une estimation garantie pour la localisation des mines a été effectuée alors que le robot a pu suivre une tout autre trajectoire.

Dans le cas de notre expérience faite avec le Redermor, il est clair que des données aberrantes peuvent être présentes. Nous avons observé que lorsque nous modifions volontairement la valeur de certaines données (pour créer des données aberrantes), rapidement GESMI concluait à l'absence de solution pour notre ensemble de contraintes. Pour notre jeu de données retourné par le Redermor lors de sa mission de deux heures, un pavé précis et non vide a été retourné par GESMI. La seule chose que nous pouvons donc conclure est que si toutes nos hypothèses sur les bornes et sur nos équations sont correctes (ce qui n'est pas certain), alors les six pavés obtenus contiennent à coup sûr les six mines qu'il nous fallait détecter.

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. 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).
3. 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).
4. C. Drocourt, L. Delahoche, B. M. E. Brassart, and A. Clerentin. Incremental construction of the robot's environmental map using interval analysis. *Global Opti-*

mization and Constraint Satisfaction: Second International Workshop, COCOS 2003 **3478**, 127–141 (2005).

5. 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).
6. L. Jaulin, M. Kieffer, I. Braems, and E. Walter. Guaranteed nonlinear estimation using constraint propagation on sets. *International Journal of Control* **74**(18), 1772–1782 (2001).
7. L. Jaulin, M. Legris, and F. Dabe. Localization and map building of an underwater robot using interval constraint propagation (submitted). In "Proceedings of the IEEE International Conference on Robotics and Automation", pp. 000–000, Rome, Italy (2007).
8. 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).
9. J. J. Leonard and H. F. Durrant-Whyte. Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research* **11**(4) (1992).
10. 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).
11. M. D. Marco, A. Garulli, S. Lacroix, and A. Vicino. A set theoretic approach to the simultaneous localization and map building problem. In "In Proceedings of the 39th IEEE Conference on Decision and Control", vol. 1, pp. 833–838, Sydney, Australia (2000).
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. T. 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. 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).