**THÈSE / UNIVERSITÉ DE BRETAGNE OCCIDENTALE**
*sous le sceau de l'Université européenne de Bretagne*

pour obtenir le titre de
**DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE OCCIDENTALE**
*Mention : Robotique*
**École Doctorale des Sciences de la Mer (EDSM)**


présentée par
Clément Aubry
Préparée à l'Institut de Recherche de l'École Navale (IRENav)


# Analyse par intervalles pour la détection de boucles dans la trajectoire d'un robot mobile

**Manuscrit provisoire du 05/09/2014**

Devant le jury composé de :

**Luc Jaulin**
Professeur des Universités, ENSTA-Bretagne - UBO / *Directeur*

**Rozenn Desmare**
Maître de Conférences, École Navale / *Encadrante*

**Dominique Meizel**
Professeur des Universités, ENSIL / *Rapporteur*

**Gilles Trombettoni**
Professeur des Universités, Université de Montpellier 2 / *Rapporteur*

**Frederic Boyer**
Professeur des Universités, École des Mines de Nantes / *Examinateur*

**Sebastien Lagrange**
Maître de Conférence, ISTIA / *Examinateur*

**Olivier Reynet**
Maître de Conférence, ENSTA-Bretagne – UBO / *Examinateur*

## Résumé

**Analyse par intervalles pour la détection de boucles dans la trajectoire d'un robot mobile**

Le travail de thèse présenté dans ce mémoire a permis le développement d'une nouvelle méthode de détection de boucles dans la trajectoire d'un robot mobile. Celle-ci se base, contrairement à celles existantes, sur l'utilisation de données proprioceptives et est ainsi totalement découplée des problématiques de localisation et cartographie auxquelles elle est généralement associée. L'utilisation de l'analyse par intervalles a permis, dans notre contexte de mesures à erreurs bornées, de calculer deux temps pour lesquels la position du robot est identique alors qu'un mouvement significatif a pu être observé entre ces deux instants. La méthode est automatique et ne nécessite que la connaissance des équations d'état du robot ainsi que les mesures de vitesse effectuées au cours de sa mission. Les résultats de l'algorithme sur données réelles, issues d'une expérience effectuée en milieu sous-marin, sont très satisfaisants et ont permis de prouver l'existence et l'unicité de plus de la moitié des boucles observées dans la trajectoire du robot.



**mots-clé :** analyse par intervalles, détection, fermeture de boucles, trajectoire, robot, sous-marin



## Abstract

**Interval analysis for loop detection in the trajectory of a mobile robot**

The work presented in this thesis deals with a new method for loop detection in the trajectory of a mobile robot. It is based on the use of proprioceptive measures, whereas other methods use exteroceptive measures. This makes this method totally independent from the simultaneous localisation and mapping problems that they are generally coupled with. The use of interval analysis allowed us, in this bounded error context, to compute two instants for which the position of the robot is identical although a significant movement has been observed between these moments. The method is automatic and only requires the knowledge of the state equations of the robot as well as the velocity measures carried out during its mission. The results of the true data algorithm, derived from an experiment carried out underwater, are very satisfactory and made it possible to prove the existence and uniqueness of more than half of the loops observed in the trajectory of the robot.



**keywords :** interval analysis, detection, loop closure, trajectory, robot, underwater

# Table des matières

**Introduction générale** **1**

**1 Robotique mobile** **5**

1.1 Introduction . . . . . 5
1.2 Robotique mobile . . . . . 6
1.3 Localisation et cartographie . . . . . 11
1.4 Détection de fermeture de boucle . . . . . 22
1.5 Contribution principale . . . . . 35

**2 Analyse par intervalles** **37**

2.1 Introduction . . . . . 38
2.2 Ensembles, treillis et intervalles fermés . . . . . 39
2.3 Intervalles réels . . . . . 46
2.4 Intervalles de $\mathbb{R}^n$ . . . . . 50
2.5 Tubes . . . . . 51
2.6 Noyau d'une fonction intervalle . . . . . 60
2.7 Recherche d'existence et d'unicité . . . . . 72
2.8 Conclusion . . . . . 78

**3 Fermeture de boucle** **79**

3.1 Introduction . . . . . 80
3.2 Détection de fermeture de boucle par mesures proprioceptives . . . 83
3.3 Validation de l'existence d'une boucle . . . . . 91
3.4 Application sur données réelles . . . . . 99
3.5 Trajectoire de retour sans boucles . . . . . 103
3.6 Conclusion . . . . . 105

**Conclusion générale** **107**

**Liste des publications et communications** **111**

**A Fonctionnement d'un Sonar à effet Doppler** **113**

**Bibliographie** **115**

# Introduction générale

Lorsqu'un robot doit interagir dans un environnement, il doit le connaître, par exemple au travers d'une carte. Il doit aussi savoir l'état dans lequel il se trouve, par exemple son orientation et sa position. Ces systèmes complexes que sont les robots utilisent pour cela des actionneurs, qui leur permettent d'interagir avec l'environnement, et des capteurs, qui assurent l'observation de l'évolution du robot ou de celle de l'environnement en question. Ces capteurs sont bien souvent eux-mêmes des systèmes très complexes mais ils n'arrivent jamais à rendre une information qui soit des plus fiables. Pour cela, des erreurs sont prises en compte dans les mesures et doivent être modélisées. La localisation en robotique mobile implique l'utilisation de ces informations entachées d'erreurs. Par exemple, une boussole mesurera une information sur le cap que suit le robot avec une erreur dépendant fortement de la qualité du capteur. Quand ces informations sont couplées avec le modèle de déplacement du robot que l'on connaît parfois de façon approximative, l'erreur de positionnement qui en découle augmente au fur et à mesure du temps : plus le robot avance, plus il se perd.

Lorsque le robot se perd, il est possible de réduire cette erreur d'estimation en utilisant les informations de l'environnement dans lequel il évolue. Cette solution est possible lorsque l'on en a une bonne connaissance, par l'intermédiaire d'une carte par exemple. Lorsque la cartographie d'un milieu n'est pas disponible, l'exploration devient plus difficile mais reste possible grâce à la résolution de problèmes de type SLAM (*Simultaneous Localization And Mapping*). Dans un environnement terrestre où la propagation des ondes électromagnétiques est possible, ces problématiques sont grandement simplifiées par la présence d'informations de qualité : par exemple, l'apparition du GPS a permis de résoudre une grande partie des problématiques de localisation. Le développement de toutes ces techniques est aujourd'hui fortement documenté mais des problèmes persistent :

si nous sommes capables de réduire nos erreurs d'estimation quant au positionnement, nous ne sommes pas capables de les supprimer complètement. Aussi des améliorations restent à apporter. Une amélioration possible est de trouver, au cours de la trajectoire effectuée par le robot, des positions déjà connues. En effet, si un robot est amené à se perdre mais qu'il recoupe sa trajectoire passée, pour laquelle l'erreur de positionnement est forcément moins importante, il est capable de quantifier l'erreur accumulée au cours du trajet et ainsi de la corriger. Le problème de **détection de fermeture de boucle** vise à récupérer cette information afin de la transmettre à des algorithmes de localisation ou de SLAM pour aider à un meilleur positionnement.

Cette thèse propose de répondre à la problématique de détection de fermeture de boucle dans la trajectoire d'un robot mobile. Nous nous plaçons dans le contexte de la **robotique sous-marine** pour lequel les mesures de l'environnement sont souvent rendues très difficiles à cause de la turbidité de l'eau ou de la mauvaise propagation des ondes électromagnétiques. A contrario, les mesures de l'évolution du robot sont souvent bien plus fiables que dans l'air : l'environnement terrestre (de la surface de la planète terre jusqu'à l'atmosphère) est perturbé et le modèle d'évolution de ce milieu possède une dynamique très rapide (tels que les changements de luminosité dus aux changements climatiques). L'utilisation des mesures de l'évolution du robot prend alors tout son sens dans l'environnement subaquatique et c'est pourquoi nous proposons une méthode de détection de fermeture de boucle sur la base de mesures proprioceptives uniquement.

Nous verrons dans le chapitre 1 différentes problématiques de localisation et cartographie, de même que la problématique SLAM, qui utilisent cette information de fermeture de boucle, ceci afin de bien comprendre le contexte dans lequel se placera la méthode développée dans cette thèse. Par la suite, un état de l'art des techniques existantes pour la détection de fermeture de boucle sera présenté. Nous verrons alors que toutes ces méthodes utilisent une observation de l'environnement et sont très souvent fortement couplées à des algorithmes SLAM. Nous mettrons alors en évidence les problématiques liées à l'utilisation de telles méthodes et qui peuvent être réduits ou supprimés par le découplage d'une méthode de détection de fermeture de boucle par rapport aux algorithmes SLAM ou de localisation impliqués.

Le deuxième chapitre aborde l'outil mathématique utilisé dans nos travaux : **l'analyse par intervalles**. Bien que les probabilités soient couramment utilisées

dans la robotique pour répondre aux problématiques citées ci-dessus, l'utilisation des méthodes ensemblistes prend tout son sens de par leur caractère garanti. En effet, compte tenu des hypothèses faites lors de la modélisation, aucun résultat n'est perdu. La capacité de traitement de problèmes fortement non linéaires sans approximation particulière donne à l'analyse par intervalles un avantage précieux car la modélisation de nos robots utilise souvent des fonctions non linéaires. De plus, les capteurs que nous utilisons (indépendamment de l'environnement) sont souvent connus pour rendre un résultat avec une précision qui apporte des bornes à nos mesures, celles ci pouvant être représentées par des intervalles. Les autres représentations, telles que les distributions probabilistes ou les nuages de point ne seront pas abordés. Ce chapitre abordera donc l'analyse par intervalles par sa base soit la théorie ensembliste. Nous verrons ensuite comment les relations d'ordre appliquées à ces ensembles nous permettent de définir les intervalles dans plusieurs espaces : l'espace des réels, l'espace des réels en dimension $n$ puis l'espace des fonctions de $\mathbb{R}$ à valeur dans $\mathbb{R}^n$. Des outils mathématiques à ce dernier espace seront développés et nous verrons enfin les méthodes ensemblistes de validation des résultats utilisés dans le cadre de nos travaux.

Le troisième chapitre présente la formalisation de la problématique de détection de fermeture de boucle et de son utilisation dans le cadre d'une application sur données réelles. L'utilisation de l'analyse par intervalles pour la représentation nous permet de fournir un résultat de détection garanti, donnée importante pour l'utilisation ultérieure de cette information. Cette nouvelle méthode apporte de nouveaux outils tels que la représentation temporelle des boucles dans la trajectoire d'un robot mobile. Cette formalisation est confortée par l'utilisation de la méthode à partir de données issues d'une expérience de deux heures (sans remontée) effectuée par un robot sous-marin chasseur de mines.

# Chapitre 1

# Robotique mobile

**Sommaire**


**1.1 Introduction . . . . . 5**
**1.2 Robotique mobile . . . . . 6**
1.2.1 Différents environnements . . . . . 6
1.2.2 Différents modes de perception . . . . . 9
**1.3 Localisation et cartographie . . . . . 11**
1.3.1 Localisation . . . . . 12
1.3.1.1 Localisation locale . . . . . 12
1.3.1.2 Localisation globale . . . . . 14
1.3.2 Cartographie . . . . . 16
1.3.2.1 Représentations métriques . . . . . 16
1.3.2.2 Représentations topologiques . . . . . 17
1.3.2.3 Représentations hybrides . . . . . 19
1.3.3 Problématique SLAM . . . . . 19
**1.4 Détection de fermeture de boucle . . . . . 22**
1.4.1 Comme un problème de localisation globale . . . . . 27
1.4.1.1 De l'observation vers la carte . . . . . 27
1.4.1.2 De la carte vers l'observation . . . . . 30
1.4.1.3 Approche reconnaissance d'image . . . . . 32
1.4.2 Algorithmes dédiés à la détection de fermeture de boucle 33
**1.5 Contribution principale . . . . . 35**


## 1.1 Introduction

Dans le dictionnaire Larousse, un robot est défini comme un *appareil automatique capable de manipuler des objets ou d'exécuter des opérations selon un*

*programme fixe, modifiable ou adaptable*. Nous trouvons, en effet, des robots dans toutes les applications, des robots de toutes les tailles et de toutes les formes. Il est communément admis que la robotique sera la révolution technologique du XXIe siècle, tout comme l'automobile et l'informatique l'ont été au XXe siècle. La robotique englobe tous les domaines d'application : l'industrie (robots manipulateurs ou "manufacturiers"), la médecine où des robots ont permis des chirurgies moins intrusives du fait de leur précision, le domaine spatial avec des robots d'exploration, l'informatique, etc.

La robotique dont il est question dans cette thèse concerne l'ensemble des robots dits mobiles, c'est-à-dire capables de se mouvoir dans un espace. Qu'il s'agisse de robots spatiaux, terrestres, aériens ou bien sous-marins, toutes ces disciplines regroupent des problématiques connexes comme par exemple la locomotion, la localisation et la cartographie.

Nous appellerons robot mobile un système mobile dans son environnement capable d'effectuer une tâche avec un certain degré d'autonomie décisionnel et énergétique.

## 1.2 Robotique mobile

### 1.2.1 Différents environnements

#### Dans l'espace

Une application pour laquelle la robotique est des plus pertinentes est l'exploration spatiale. En effet, dans ce domaine, la connaissance des environnements des planètes qui nous entourent est essentielle. Notre difficulté à explorer cet environnement réside dans les conditions climatiques et atmosphériques nécessaires à la vie humaine. Les robots, qui peuvent facilement s'adapter et être spécialisés pour un type de conditions particulières, permettent d'obtenir des analyses précises *in-situ*. Par exemple, nous citerons les différentes missions des robots de la NASA SPIRIT et CURIOSITY [NASA-JPL 2014] envoyés sur Mars respectivement en 2004 et 2012.

#### Dans l'air

La robotique aérienne est, elle aussi, en plein essor. Ses utilisations sont multiples en allant du simple hobby jusqu'aux applications militaires. Il faut ajouter,

à ces deux domaines extrêmes, l'utilisation de quadrirotors à des fins scientifiques. Plusieurs projets mettent ainsi en œuvre ce type de robot, avec des degrés d'autonomie différents et adaptés à leur utilisation. Le rapport [Derkx, Sorin et al. 2008] dresse un état de l'art des utilisations des quadrirotors avec des projets d'inspection du littoral des Moutiers suite à la tempête Xinthia, des projets de suivi des risques d'éboulement en vallée d'Aspes et d'autres projets d'inspection de piles de ponts de grande hauteur ([Metni et Hamel 2007 ; Derkx, Sorin et al. 2008]). Dans ces domaines, l'apport de la robotique permet de diminuer les coûts et risques liés à ces domaines extrêmes.

#### Sur Terre

La robotique terrestre regroupe les robots ayant la capacité de se déplacer par leurs propres moyens sur le sol avec un certain degré d'autonomie décisionnel et énergétique. Ces robots peuvent être classifiés en plusieurs catégories :

**Robots roulants** Tous ces robots se déplacent par le biais de chenilles ou de roues. Ils sont utilisés dans de multiples domaines : l'armement (drones d'inspection de bâtiments ou drones anti-mines), l'industrie (convoyage de pièces), le transport (par exemple, les mini-bus sans pilote du parc d'attraction Vulcania ou bien les prochaines voitures sans pilote (Google-Car, Volvo DriveMe) qui apparaîtront dans quelques années). Il faut aussi noter les robots de service déjà présents dans les foyers avec les robots aspirateurs, tondeuses et de nettoyage autonomes. À ces domaines déjà bien intégrés dans nos vies courantes, il faut ajouter tous les robots qui se déplacent actuellement dans les laboratoires servant de support à la recherche (B2P2 à l'ISTIA [Paillat 2010], ...).

**Robots bipèdes** Ces robots sont équipés de deux jambes et suivent ainsi un mécanisme de mimétisme en reproduisant la marche humaine. Nous citerons le robot français Nao [Gouaillier et al. 2009] de la société Aldébaran, le HRP-3 [Kaneko et al. 2008] développé par l'AIST au Japon, le robot Qrio [Geppert 2004] de Sony ou encore Asimo [Hirose et Ogawa 2007] de Honda. Ces robots humanoïdes sont bien souvent destinés au service à la personne, avec de nombreux projets pour l'accompagnement de personnes âgées à domicile ou bien pour la recherche sur l'autisme. En effet, l'apparence humaine de ces robots les rend plus conviviaux et ils sont donc plus à même de s'intégrer dans nos environnements. De plus, leurs dimen-

sions sont souvent adaptées pour la navigation dans des environnements humain.

**Robots à pattes** Ces robots regroupent tous les robots qui se déplacent par le biais de pattes, leur nombre peut aller de 4 à ... beaucoup. Parmi ces derniers, le Big Dog de Boston Dynamic [Raibert et al. 2008] est capable de se déplacer en terrain accidenté avec une charge utile importante, Cheetah, un robot créé par la DARPA peut atteindre la vitesse de 45.5km/h en mimant la course du guépard, et plusieurs robots équipés de 6 à une multitude de pattes se déplacent en utilisant des techniques dites de *biomimétisme* (robot araignée par exemple).

### Dans l'eau

La robotique dans le milieu maritime connaît, elle aussi, un fort développement. En effet, les contraintes d'accessibilité que subit l'homme sont une fois encore atténuées lors d'utilisations de robots. Les applications sont nombreuses : la navigation en autonomie, avec des projets comme MicroTransat [MicroTransat 2014], une course transatlantique par des voiliers autonomes ; la surveillance de ports, de coques de bateaux ; l'exploration sous marine dans le cadre de l'archéologie, pour la découverte de pétrole sous les fonds marins ou bien pour l'entretien ou encore pour le suivi de câbles de télécommunication sous-marins (figure 1.1). L'eau étant un très mauvais conducteur pour les ondes électromagnétiques, les outils développés pour et par les humains dans le domaine aérien et le domaine terrestre ne sont plus disponibles. Par exemple, le système GPS et les communications par voies hertziennes ne sont pas accessibles. Si l'on ajoute à ces nouvelles contraintes l'animosité du milieu aquatique (pressions en jeu et salinité en cas d'eau de mer), les développements de machines performantes dans ces domaines deviennent un véritable challenge et accroissent donc l'intérêt porté vers ces applications. Notons, dans ce contexte, l'apparition récente d'un robot anguille ([Boyer et al. 2006]) qui utilise un nouveau sens pour se déplacer et se localiser : l'électrolocation.

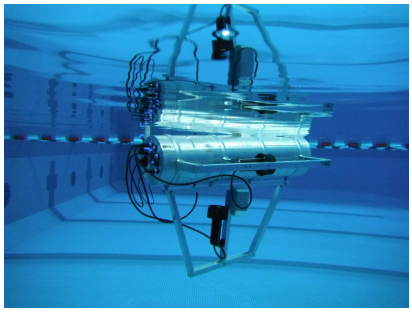
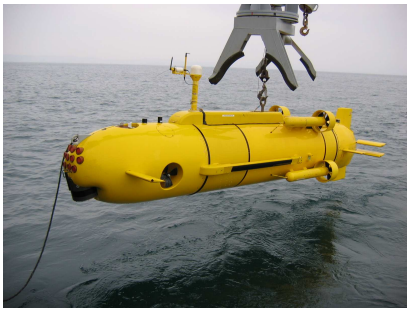



FIGURE 1.1 – Deux robots sous-marins : ROSEN de l'IRENav et le Redermor du GESMA

### 1.2.2 Différents modes de perception

Les robots sont dotés de capteurs qui leur permettent de percevoir, soit leur environnement par des capteurs *extéroceptifs*, soit des informations relatives à leur évolution : nous parlerons alors de capteurs *proprioceptifs*. Certains capteurs peuvent fournir une information bénéficiant des deux approches : ils seront alors appelés capteurs *exproprioceptifs*.

#### Observer son environnement

Les capteurs extéroceptifs permettent au robot d'obtenir une information relative à son environnement global (dans un repère absolu). Par exemple, un capteur GPS reçoit les informations par les satellites afin de donner une estimation de sa position. Une caméra, quant à elle, va utiliser les informations lumineuses en provenance de l'environnement dans lequel le robot évolue : c'est donc aussi un capteur extéroceptif. Ces capteurs sont utilisés quand on a besoin d'informations "à jour" de l'environnement. Par exemple, un robot se déplaçant dans un environnement connu par une carte précise a besoin d'informations extéroceptives qui lui permettront de prendre en compte les éléments mobiles de la carte (la position des éléments mobiles ne pouvant pas être connu, au contraires de celle des éléments fixes).

#### Observer son évolution

Par définition, les informations obtenues par les capteurs proprioceptifs ne prennent pas en compte une référence externe au robot mais une observation

de son évolution. Par exemple, les gyromètres qui mesurent une vitesse de rotation instantanée sont des capteurs proprioceptifs. La roue codeuse qui permet de quantifier la vitesse de rotation d'un moteur l'est aussi. Ces capteurs permettent, par exemple, de connaître les vitesses de rotation des roues d'un robot mobile et donc de pouvoir quantifier le mouvement qu'il effectue connaissant son mode de déplacement. C'est le principe de l'odométrie définie dans [Von Der Hardt et al. 1996 ; Chong et Kleeman 1997 ; Doh et al. 2003] comme l'utilisation de capteurs dans le but d'évaluer le déplacement d'un robot.

Si la définition d'un capteur proprioceptif interdit à ce dernier d'observer son environnement, il est courant de classer certains capteurs extéroceptifs dans la branche des capteurs proprioceptifs. Par exemple, une boussole est bien souvent classée comme un capteur proprioceptif alors qu'elle va mesurer le champ magnétique terrestre, mesure extéroceptive. Le constat est le même pour beaucoup de capteurs utilisés en robotique et il existe donc très peu de capteurs purement proprioceptifs. En robotique mobile, la définition d'un capteur proprioceptif est bien souvent étendue à tout type de capteur permettant de décrire l'évolution du robot.

Plus concrètement, tout capteur permettant de décrire, par utilisation directe des données qu'il perçoit, l'évolution d'un robot sera classé comme capteur proprioceptif. Par exemple, un capteur de vitesse à effet Doppler (Doppler Velocity Log - DVL) est un capteur proprioceptif. Les DVL sont des capteurs qui permettent de mesurer la vitesse d'un mobile : c'est donc une information proprioceptive. Pour cela, le DVL émet quatre faisceaux et capte, dans l'écho de ces derniers contre le fond ou autre interface, les décalages de fréquence induits par l'effet Doppler [1]. De cette manière, le capteur est capable de retourner la vitesse du capteur par rapport à l'interface sur laquelle se sont réfléchies les ondes émises. La mesure de la vitesse par DVL est donc, par définition, une mesure extéroceptive mais elle ne renvoie pas d'informations sur le milieu extérieur. Ce capteur sera donc classé comme un capteur proprioceptif.

Si l'on veut situer ces deux types de mesure dans un exemple concret, imaginons un robot mobile dans une maison dont la carte des différentes pièces est bien connue. Ce robot se localise par odométrie uniquement (information proprioceptive) mais il a besoin, s'il veut pouvoir éviter les obstacles (meubles, jouets,

1. L'effet Doppler est le décalage de fréquence d'une onde entre l'émission et la réception lorsque la distance entre les deux varie au cours du temps.

résidents, ...) d'une information extéroceptive. C'est pourquoi les informations proprioceptives et extéroceptives sont toutes les deux importantes et surtout complémentaires. Formellement, lorsqu'on décrit le fonctionnement d'un robot par ses équations d'état comme dans l'équation 1.2.1, le vecteur $u$ correspond aux mesures proprioceptives et le vecteur $y$ correspond aux mesures extéroceptives.

$$\begin{aligned} \dot{x} &= f(x,u), \\ y &= g(x). \end{aligned} \tag{1.2.1}$$

## 1.3 Localisation et cartographie

Tous les robots présentés précédemment peuvent se mouvoir dans un environnement qui conditionne leur mode de déplacement. La mobilité (ou locomotion) est une compétence acquise en robotique dans de nombreux et différents environnements. Mais à quoi sert cette mobilité si l'on ne sait pas où l'on se trouve relativement à l'objectif que l'on s'est fixé ? Quelles informations dont dispose le robot vont permettre cette localisation ? Si l'on parle de localisation, il faut aussi introduire le concept de carte, puisque ces deux éléments sont indissociables.

Lorsque les robots évoluent dans notre environnement, les informations sont souvent présentes en masse et réutilisables. Par exemple, un robot volant pourra utiliser un capteur GPS pour se localiser alors qu'un robot d'exploration sur Mars ne le pourra pas. Ce même robot volant dispose d'une cartographie précise et mise à jour très rapidement et fréquemment. Le rover envoyé sur Mars possède un plan de la planète acquis par imagerie satellite. Prenons maintenant l'exemple d'un robot sous-marin : si nous connaissons 80% de la surface de la planète Mars, nous ne connaissons que 20% de la surface de nos océans. Nous comprenons alors que la localisation devient difficile puisque l'on ne dispose pas de carte.

Nous allons présenter succinctement les problématiques de localisation ainsi que celle de localisation et cartographie simultanées. Ces paragraphes ne sont que des présentations succinctes de ces concepts largement étudiés par la communauté scientifique depuis plusieurs décennies et qui font encore aujourd'hui l'objet de thèses (par exemple [Guyonneau 2013 ; Bars 2011]). Cette présentation est nécessaire dans la mesure où l'apport de cette thèse se trouve dans l'ajout d'une information capitale à ces problématiques, la fermeture de boucle dans la trajectoire d'un robot mobile.

### 1.3.1 Localisation

La problématique de localisation, au sens large, revient à déterminer la posture d'un robot dans son environnement. La posture, également appelée pose, est composée de la position et de l'orientation du robot (les conventions utilisées dans ce manuscrit pour la représentation d'un robot sont illustrées par la figure 1.2). Nous pouvons observer, sur cette figure, un robot qui évolue dans un plan et dont la posture $\mathbf{p}(t)$ est composée de la position, $(x(t), y(t))$, et de l'orientation $\theta(t)$.

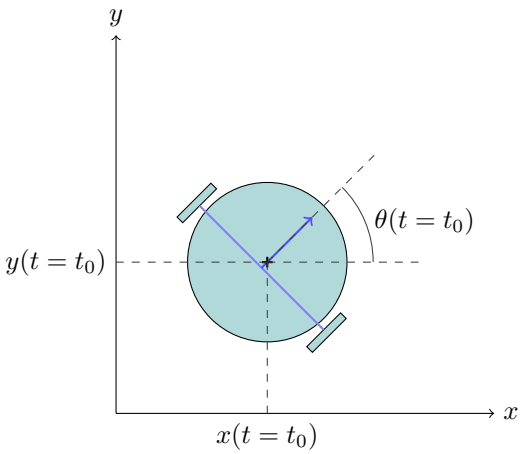



FIGURE 1.2 – Posture d'un robot mobile à l'instant $t = t_0$ : $\mathbf{p}(t_0) = \begin{pmatrix} x(t_0) \\ y(t_0) \\ \theta(t_0) \end{pmatrix}$.

La problématique de localisation est divisée en deux grandes familles : la localisation locale et la localisation globale.

#### 1.3.1.1 Localisation locale

La localisation locale admet la connaissance d'une posture initiale pour notre robot et on cherchera à déterminer, par la connaissance du mode de déplacement du robot (par exemple connaissant ses équations d'état), la posture à un temps $t$ par rapport à sa posture à l'instant $t - 1$. La figure 1.3 illustre le problème de suivi de posture : à l'instant $t = t_0$, la position du robot est connue et les positions aux instants $t_n$ seront estimées à partir de la position à $t_{n-1}$. La principale difficulté pour résoudre ce problème de suivi de posture est d'éviter le phénomène de

dérive. En effet, la modélisation de l'évolution du robot est souvent entachée d'imprécisions qui vont induire des erreurs ou bien des incertitudes dans l'évaluation du mouvement effectué par le robot entre deux instants. Ces erreurs vont alors se répercuter tout au long du processus de localisation. Pour bien comprendre l'origine du phénomène de dérive, imaginons un robot de type char, comme représenté sur la figure 1.3. Nous connaissons bien les équations d'évolution du robot mais il nous est impossible de modéliser le glissement des roues puisqu'il est lui-même fortement dépendant de la nature du terrain sur lequel le robot évolue. Il est donc courant de supposer les glissements nuls lors de la modélisation en admettant cette erreur de dérive incrémentielle. Ce phénomène est représenté sur la figure 1.3 où l'on observe une différence entre la position réelle $\mathbf{p}(t_n)$ du robot (trajectoire en pointillés) et la position estimée $\widetilde{\mathbf{p}}(t_n)$ (trajectoire en traits pleins).

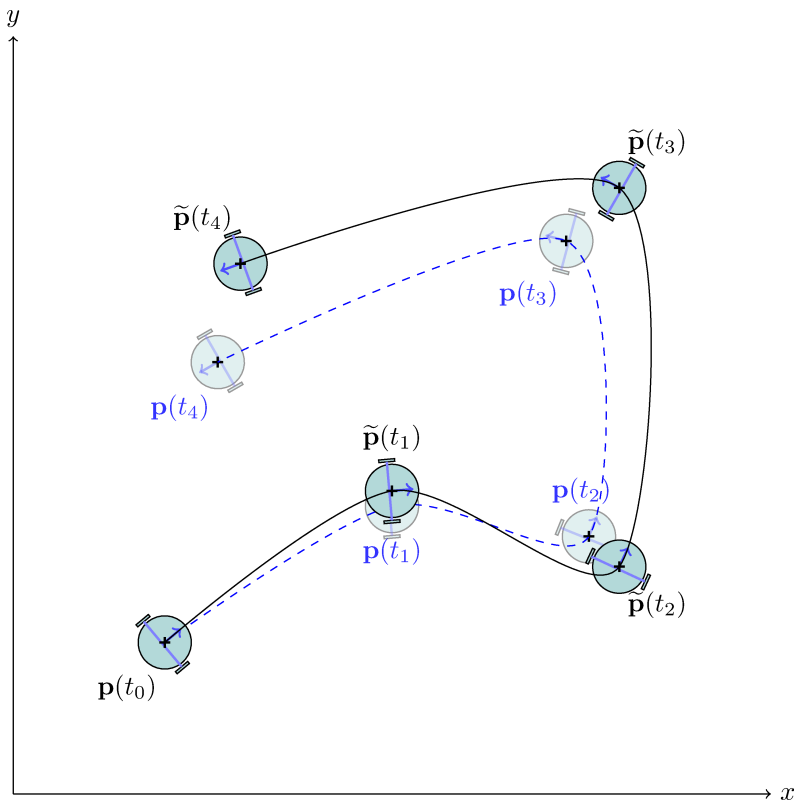



FIGURE 1.3 – Illustration du problème de suivi de posture - $\mathbf{p}(t)$ représente la position réelle et $\widetilde{\mathbf{p}}(t)$ la position estimée.

#### 1.3.1.2 Localisation globale

La localisation globale concerne le cas plus général où le robot se trouve dans une posture inconnue. Il doit alors observer son environnement pour en déduire sa posture, les méthodes de localisation locale pouvant être ensuite appliquées. Ce cas est plus difficile que le suivi de posture puisqu'il nécessite au moins la connaissance d'un référentiel qui permettra l'acquisition d'une première posture "globale". Ce référentiel peut être donné par une carte d'amer.

La localisation sur une carte d'amer vient de la navigation maritime côtière, où on relève l'angle d'observation de trois amers pour en déduire une position. La figure 1.4 illustre le principe de navigation par amers : à un instant donné, le navigateur relève l'angle entre le nord, sa position et l'amer qu'il cible et reporte, sur une carte, cet angle. Si l'opération est renouvelée sur trois amers observés, on obtiendra, à l'intersection des trois droites, la position du bateau. Bien souvent, ces dernières ne forment pas un point d'intersection unique. Dans cette situation, le bateau sera considéré comme étant au centre du cercle inscrit au triangle formé par les trois droites. La localisation par amers a été formalisée [Leonard et H. F. Durrant-Whyte 1992b ; Stevens et al. 1995] et complétée. Ainsi, on en distingue plusieurs types : par mesure de distances uniquement (*range-only localization*), par relèvement d'angles (*bearing-only localization*) ainsi que les méthodes utilisant les deux approches.

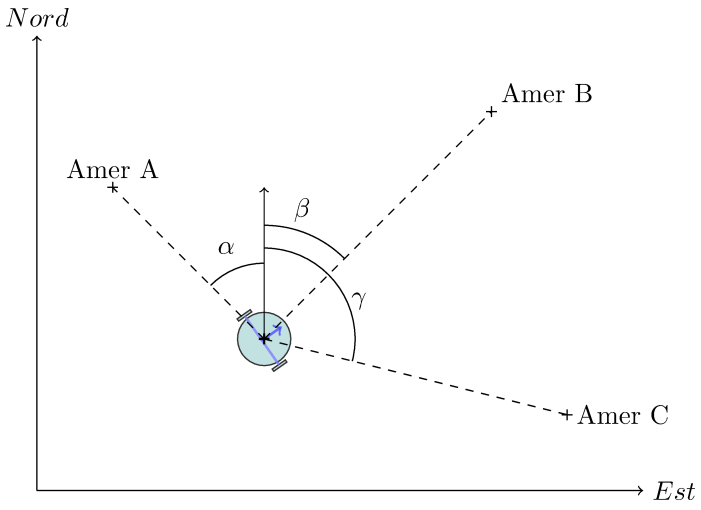



FIGURE 1.4 – Navigation par relèvement d'amers.

Dans le cadre probabiliste, il existe des méthodes de filtrage particulaire qui répondent au problème de localisation globale (Monte Carlo Localization (MCL), [Dellaert et al. 1999], [Fox et al. 1999], [Andreasson et al. 2005], Rao-Blackwellised Particle Filter (RBPF) [Barfoot 2005]). Dans ces méthodes, la pose du robot est approximée par une distribution de probabilité qui utilise un ensemble d'échantillons appelés *particules*. Chacune des particules représente une hypothèse de positionnement du robot. A chaque itération, chaque particule est pondérée par la pertinence qu'elle représente avec les dernières mesures obtenues. En concentrant les particules qui représentent les positions les plus vraisemblables, par ré-échantillonnage puis normalisation, une distribution de probabilité discrète est obtenue pour la position du robot.

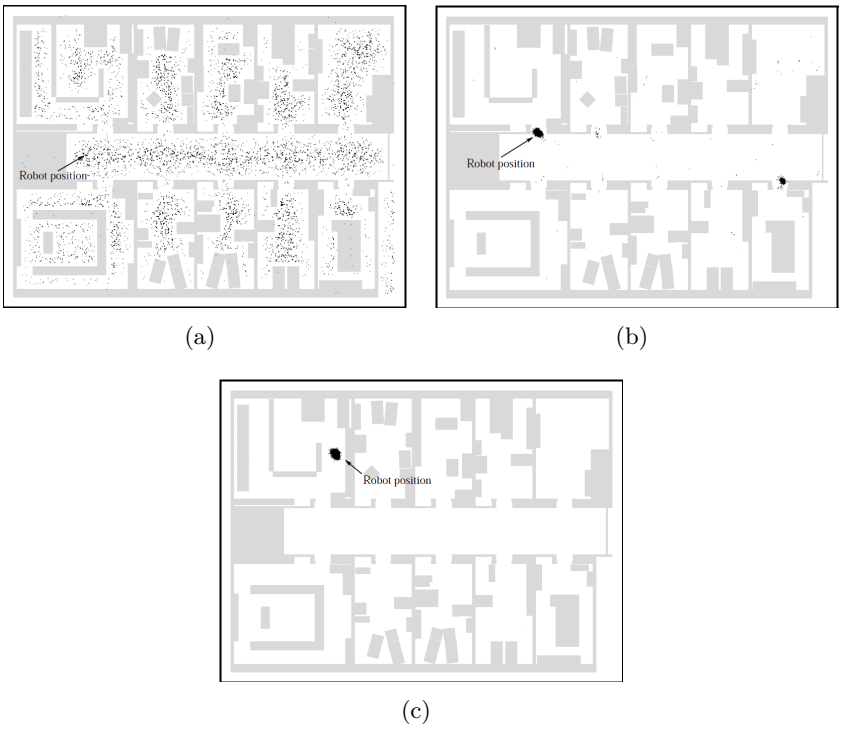



FIGURE 1.5 – Illustration du principe de localisation MCL, extrait de [Fox et al. 1999].

Sur la figure 1.5, lors de l'initialisation de l'algorithme, (figure (a)), les particules sont dispersées sur la carte. Après un mouvement du robot d'approximativement un mètre, sur la figure (b), l'algorithme a concentré les particules en deux lieux quasiment symétriques. Ces deux lieux sont ceux qui répondent de

manière la plus probable à l'estimation de positionnement en prenant en compte les mesures extéroceptives effectuées par le robot. Sur la figure (c), après un autre mouvement d'environ deux mètres, le robot a levé l'ambiguïté et connaît maintenant sa position.

**Kidnapping**

Le kidnapping est un problème de localisation globale. Prenons un robot qui évolue à l'estime dans un environnement connu. Si on kidnappe ce robot et qu'on le place manuellement à un autre endroit de la carte, les informations accumulées par l'odométrie et qui lui permettaient jusque là de se localiser deviennent obsolètes et le robot va devoir retrouver sa position sur la carte avant de continuer sa navigation. Il lui faudra donc utiliser des informations extéroceptives qui lui permettront une nouvelle localisation (souvent appelée re-localisation dans ce cadre). C'est donc un problème de localisation globale car aucune information n'est disponible sur l'état initial du robot.

Si les méthodes probabilistes ont prouvé leur efficacité dans le domaine de la localisation, il faut noter l'utilisation d'autres méthodes de représentation de données incertaines dans ce contexte. L'analyse par intervalles, qui sera développée dans le chapitre 2, est une de ces méthodes. L'utilisation de l'analyse par intervalles a montré son efficacité par l'apport de la robustesse et du caractère garanti apporté aux résultats ([Jaulin, Kieffer, Braems et al. 2001],[Clérentin et al. 2008], [Gning 2006]). La thèse [Guyonneau 2013] dresse un état de l'art complet sur les méthodes ensemblistes de localisation globale.

### 1.3.2 Cartographie

La cartographie est une donnée très importante pour la localisation puisqu'elle va fixer un référentiel nécessaire à la résolution de ces problèmes. Il existe plusieurs types de représentation qui sont adaptés à différents environnements et à différentes utilisations.

#### 1.3.2.1 Représentations métriques

La représentation métrique ([Elfes 1987 ; Moravec 1988]) est celle que nous connaissons tous. Ce type de carte, équipé d'un repère et d'une échelle, permet

d'obtenir une information géométrique de la surface couverte par la carte. Ces cartes donnent une information de distance entre les différents éléments de la carte. Pour citer quelques exemples, les cartes IGN, les cartes marines, un planisphère ou un globe sont des cartes métriques. Les cartes métriques utilisées pour la localisation en robotique peuvent être des cartes *classiques* comme celles citées précédemment, des cartes d'amers qui représentent l'environnement sous forme d'amers - un élément identifiable de l'environnement pouvant servir de repère - ou bien des cartes dites d'occupation ([Elfes 1987 ; Moravec 1988]). Ces dernières divisent l'espace en cellules - sous-ensemble de l'espace représenté - et indiquent une probabilité d'occupation de cette cellule par un obstacle. Une image issue d'un sonar de navigation (imagerie à 360°), comme le montre la figure 1.6, peut être interprétée comme une grille d'occupation. La couleur indique la probabilité de présence d'un obstacle (en bleu une probabilité faible, en blanc une probabilité élevée).

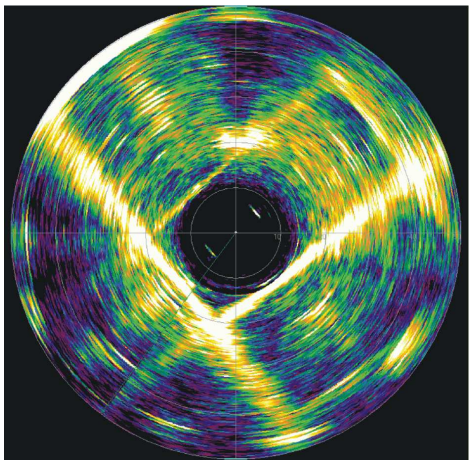


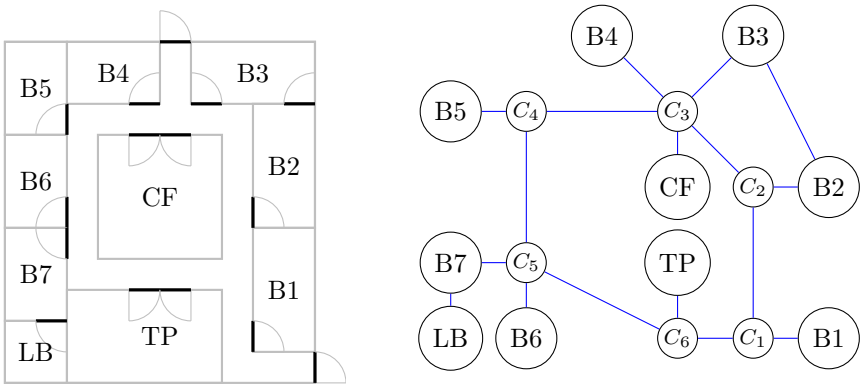
FIGURE 1.6 – Image SONAR d'un robot dans une piscine (source : [Bars 2011]), les segments blancs correspondent aux bords de la piscine qui émettent les plus fort écho. On peut voir apparaître des lignes parallèles à ces échos issus des trajets multiples.

#### 1.3.2.2 Représentations topologiques

La représentation topologique [Kuipers et Byun 1991 ; Kortenkamp et Weymouth 1994 ; Zimmer 1996] apporte une information non géométrique d'un espace. Ce type de cartes vise à étudier les connexions qui existent entre les différents

sous-espaces qui constituent la carte. Ces cartes sont souvent représentées par des graphes. Par exemple, il peut être intéressant d'utiliser une carte topologique pour représenter les connexions entre différentes pièces dans un bâtiment. La localisation sur une carte topologique ainsi construite vise à identifier le nœud du graphe représentant la pièce du bâtiment où l'on se trouve.

Cet outil est souvent utilisé à un degré décisionnel en robotique puisqu'il permet de connaître les étapes nécessaires au passage d'un nœud vers un autre dans un espace. La figure 1.7 confronte la représentation métrique (figure 1.7(a)) et la représentation topologique (figure 1.7(b)) d'un bâtiment.




(a) Carte métrique

(b) Carte topologique, les nœuds $C_n$ appartiennent au couloir entre les différentes pièces.

FIGURE 1.7 – Différences entre carte métrique et carte topologique.

Sur une carte métrique, un robot pourra se localiser n'importe où dans les limites de la carte. Sur la carte topologique, la localisation se fera par rapport aux entités représentées par les différents nœuds : par exemple, le robot se trouve dans la salle de conférence (notée $CF$).

La carte topologique va permettre au robot de connaître les étapes de navigation nécessaires pour aller d'un nœud à un autre. Par exemple, notre robot qui a été localisé dans la salle de conférence ($CF$) doit rejoindre la salle de travaux pratiques (notée $TP$). Pour cela, il doit effectuer le trajet $CF \rightarrow C_3 \rightarrow C_4 \rightarrow C_5 \rightarrow C_6 \rightarrow TP$ ou bien le trajet $CF \rightarrow C_3 \rightarrow C_2 \rightarrow C_1 \rightarrow C_6 \rightarrow TP$. La carte topologique n'apportera pas d'informations sur la longueur des trajets sauf à étiqueter les arêtes par une longueur.

#### 1.3.2.3 Représentations hybrides

Si ces deux types de représentation apportent des informations différentes, il arrive qu'elles soient mélangées ([Yamauchi et Langley 1997 ; Simhon et Dudek 1998 ; Choset et Nagatani 2001]) pour présenter des cartes hybrides comme nous le montre la figure 1.8 qui rassemble les cartes présentées sur la figure 1.7. Ainsi, le robot connaîtra les entités topologiques qui construiront le chemin qu'il doit effectuer pour se rendre d'une entité à une autre tout en ayant accès aux données métriques qui lui permettraient, par exemple, une localisation métrique à l'intérieur d'une pièce.

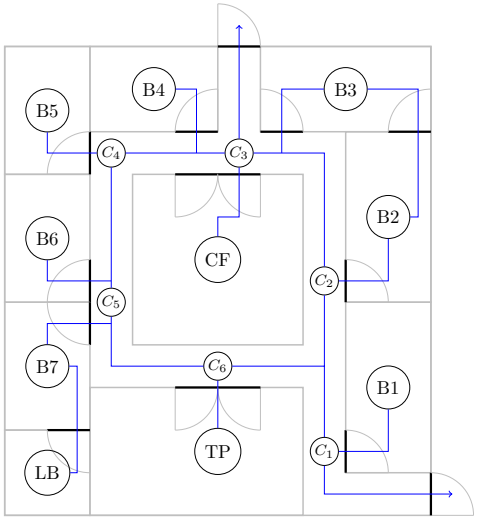



FIGURE 1.8 – Exemple de carte hybride métrique/topologique.

### 1.3.3 Problématique SLAM

Nous avons vu que les problèmes de localisation recherchent une position par rapport à un repère que l'on s'est fixé, soit par rapport à la première position connue pour la localisation locale ou bien par rapport à une carte pour la localisation globale. La cartographie va représenter les lieux utilisés dans ces problèmes. Ceux-ci sont donc liés. Mais comment résoudre l'un ou l'autre lorsque l'on visite un lieu pour la première fois ? C'est dans ce cadre que la problématique SLAM (*Simultaneous Localization And Mapping*, [Leonard et H. F. Durrant-Whyte 1992a]) se place. En effet, en l'absence de référentiel, il faut identifier, à l'aide de capteurs

extéroceptifs, les éléments de l'environnement qui nous entourent et qui pourront servir de repère. Ces éléments seront positionnés afin de construire une carte de l'environnement. Ainsi, lorsque l'on retrouvera ces éléments dans le futur, on sera capable d'identifier la position de l'observateur.

**Exemple 1.3.1** (Exemple introductif au problème SLAM)**.** *Nous pouvons voir, sur la figure 1.9, un robot dans un labyrinthe dont il ne connaît pas le plan. Ce robot est équipé de trois capteurs qui lui donnent une information de distance à courte portée (capteur extéroceptif) ainsi qu'une centrale inertielle capable de lui donner des informations relatives au mouvement qu'il effectue (capteur proprioceptif). La stratégie choisie pour atteindre la sortie du labyrinthe est de toujours tourner à gauche en premier lorsqu'un choix se présente. En (a), le robot est mis en position de départ et voit, grâce à ses capteurs un couloir dont il ne perçoit pas la fin. En (b), le robot a avancé en découvrant une partie du labyrinthe et, se retrouvant face à un choix, tourne à gauche. En (c), le robot a continué son avancée. Tout au long de son trajet, il a enregistré le plan qu'il a découvert et s'y positionne : c'est la configuration du problème SLAM. En (d), le robot est en position finale. On peut voir qu'une partie du plan n'a pas été découverte : ceci est dû au choix de la stratégie qui permet de trouver la sortie du labyrinthe. Si le robot avait pris à droite lorsqu'un choix lui était présenté, le plan final serait différent mais le robot serait arrivé à cette position finale.*

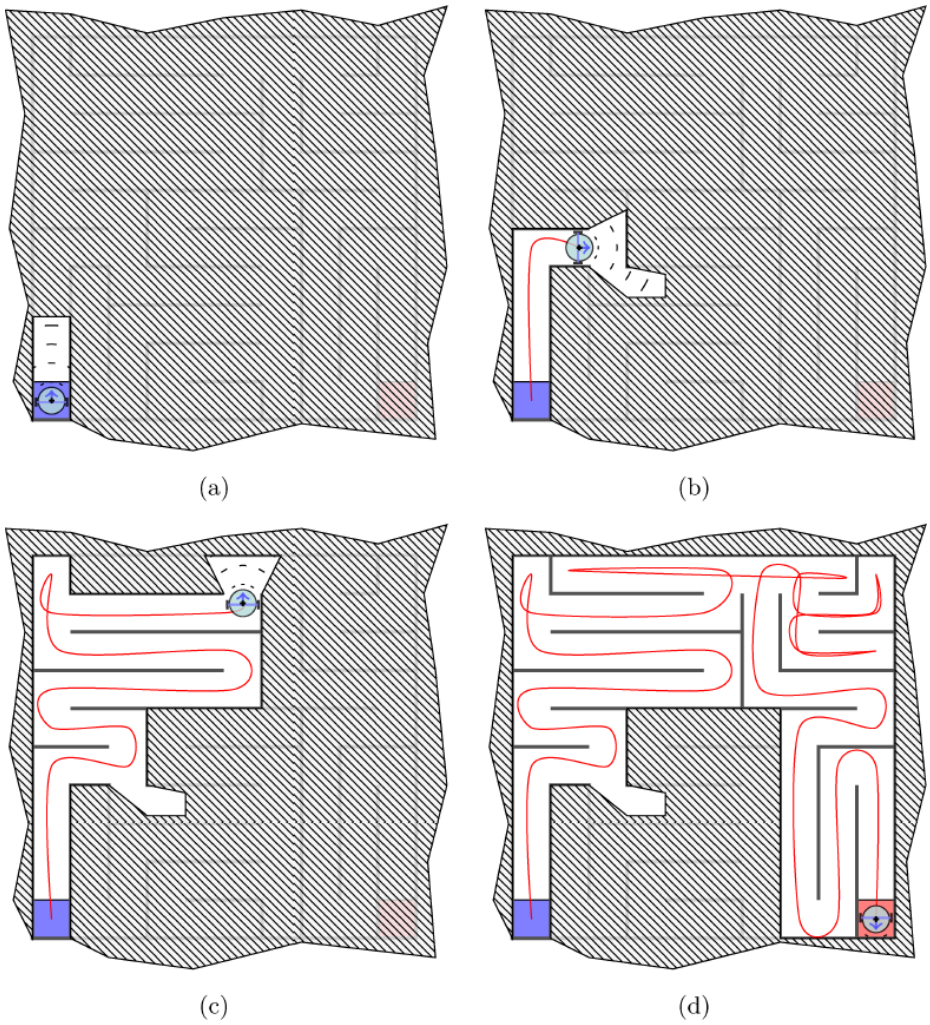



FIGURE 1.9 – Illustration du problème SLAM. Une estimation grossière de la trajectoire du robot est représentée en rouge.

**Remarque 1.3.1.** *Le lecteur averti trouvera cet exemple bien trop parfait, en effet, lors de la résolution d'un problème SLAM, aucun algorithme n'est capable de produire une carte aussi précise. Il faudrait apporter une distorsion sur les images 1.9(b) et 1.9(c) pour intégrer l'erreur de cartographie amenée par le phénomène de dérive. Le robot serait capable de recaler sa carte lors de la sortie du labyrinthe ou bien lors de la détection d'une boucle.*

*Cette stratégie ne fonctionne plus dans un labyrinthe à îlots où l'on chercherait à atteindre le centre. Dans ce cas, le robot ferait le tour entier du labyrinthe*

*pendant un temps infini. Pour se sortir d'une telle situation, il faudrait pouvoir changer de stratégie, par exemple, lors de la détection d'une boucle dans la trajectoire du robot.*

Différentes approches existent pour traiter le problème SLAM : certaines sont offline [2] et peuvent demander un temps de traitement considérable, d'autres sont online [3] et nécessitent des algorithmes temps réels. La plupart de ces approches transforment le problème en problème d'estimation d'état, où la pose ainsi que la position des amers sont les variables à estimer ([Castellanos et Tardós 1999 ; Dissanayake et al. 2001 ; Montemerlo et Wegbreit 2003]). Certaines solutions sont, elles, basées sur des techniques d'estimation probabilistes. Nous citerons les méthodes utilisant le filtrage de Kalman, l'estimation Bayésienne ou bien le filtrage particulaire succinctement expliqué pour résoudre le problème de localisation globale (voir [Thrun et al. 2005] pour le détail de ces méthodes). Le lecteur trouvera dans [H. Durrant-Whyte et Bailey 2006 ; Bailey et H. Durrant-Whyte 2006] un état de l'art sur le SLAM.

Comme pour la localisation, des algorithmes SLAM ont été développés dans le cadre du calcul ensembliste [Jaulin 2009 ; Bars 2011]. L'aspect garanti apporté par le calcul ensembliste (détaillé dans le chapitre 2) apporte une sécurité pour ce type de traitement.

## 1.4 Détection de fermeture de boucle

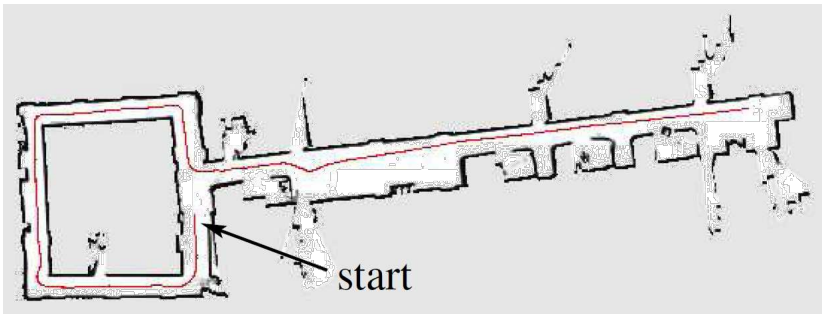
Quand on cherche à construire la carte d'un environnement peu ou mal connu, il est souvent utile de revenir à une position initiale ou bien à une position connue : l'exemple d'un labyrinthe à îlots cité précédemment (exemple 1.3.1) illustre bien ce phénomène. Ce mécanisme de bouclage permettant de mieux appréhender l'environnement, nous connaissons alors un périmètre engendré par cette boucle sans être perdu puisque nous sommes revenus à une position connue. Les problématiques SLAM se servent de cette information de fermeture de boucle. En effet, cette information permet de minimiser le phénomène de dérive lors de la fermeture d'une boucle : quand une boucle est détectée, l'erreur de positionnement à l'ins-

2. offline : traitement effectué une fois que toutes les données du problème sont disponibles. Dans notre exemple, un robot navigue et récolte des données. Après cette mission, les données sont analysées.

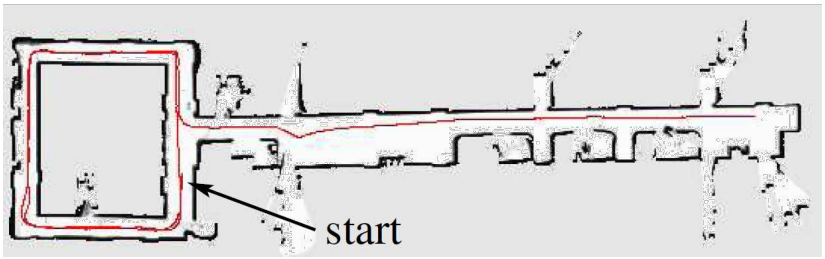
3. online : traitement effectué pendant la mission du robot, l'algorithme traite les données dès qu'elles sont disponibles.

tant ou l'on détecte la boucle peut être ré-évaluée et cette information répercutée sur le trajet passé.

Les figures 1.10 et 1.11 nous montrent deux exemples où l'information de fermeture de boucle est utilisée dans des algorithmes SLAM. Les méthodes utilisées pour produire ces images seront détaillées plus loin dans cette section.



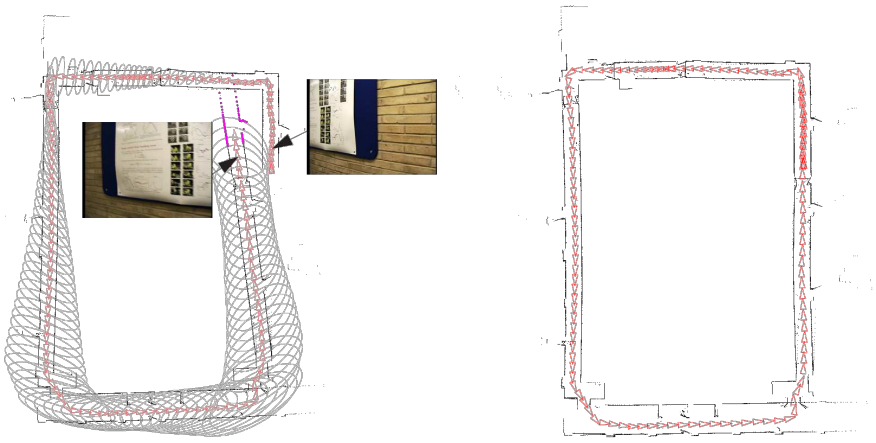

(a)




(b)

FIGURE 1.10 – Exemple d'utilisation de l'information de fermeture de boucle [Stachniss et al. 2004].

Sur la figure 1.10(a), le robot tourne autour d'une pièce sans effectuer un tour complet avant de s'engager dans un corridor. Sur l'image (b), le robot fait deux fois le tour de cette même pièce avant de s'engager dans le corridor. On peut voir la différence entre les deux expériences : quand le robot a effectué une boucle, il a corrigé l'erreur d'estimation de sa pose et est donc capable de produire une carte plus précise de l'environnement parcouru. La précision apportée pour cette expérience est visible dans l'orientation du corridor.



(a) (b)

FIGURE 1.11 – Exemple d'utilisation de l'information de fermeture de boucle [Ho et Newman 2006].

On peut voir, sur la figure 1.11(a), la trajectoire du robot avec des positions estimées représentées par des ellipses. Plus le robot avance, plus l'incertitude augmente (phénomène de dérive). Cette incertitude de positionnement va alors engendrer une erreur de cartographie puisque l'estimation de la pose du robot est erronée. Lors de la détection d'une boucle, on peut réduire l'incertitude sur la position à l'instant où la boucle a été détectée et ainsi répercuter cette information sur l'estimation de la pose au cours du trajet. Avec cette information, une carte plus précise est alors construite, comme le montre l'image (b).

Ces exemples illustrent bien l'importance de la détection de fermeture de boucle dans des algorithmes SLAM : nous obtenons un meilleur positionnement et une meilleure carte. Lorsque la détection de fermeture de boucle n'est pas prise en compte, la carte ainsi que le positionnement, malgré le fait qu'ils soient moins précis, restent viables (i.e. la position est bien donnée dans la carte même si cette dernière est fausse). Le problème le plus important apporté par la détection de fermeture de boucle advient lors d'une fausse détection. Si un faux positif est détecté, la carte devient alors totalement erronée. Ce problème est illustré sur la figure 1.12. Le robot, sur la figure (a), rencontre à deux instants différents la même scène (ici, les arbres sur le bord de la route). Ce problème est celui de l'aliasing

perceptuel [4]. Une façon de garder la carte cohérente est de créer physiquement une boucle dans la trajectoire, comme nous le montre la figure (b). Une autre manière de traiter le problème est de superposer les cartes afin que les deux positions soient identiques, figure (c). Dans ces deux cas, le positionnement ainsi que la carte ne sont plus viables. Cet exemple naïf illustre bien les problèmes qui peuvent être engendrés par l'utilisation d'un faux positif. Imaginons maintenant cette situation rencontrée sur un plan plus complexe : on peut voir apparaître des problèmes de recouvrement de cartes difficiles à gérer pour les problèmes SLAM.

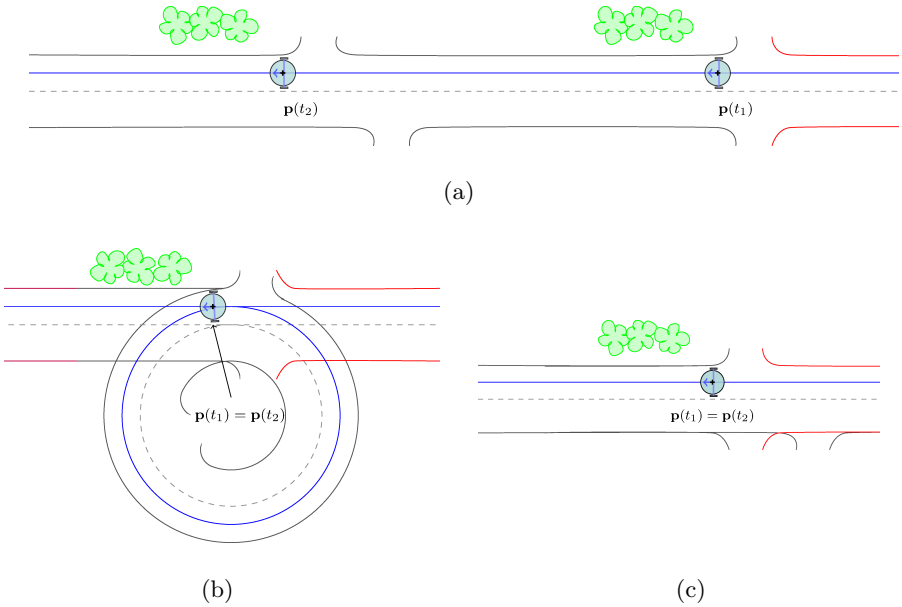



FIGURE 1.12 – Exemple de problème créé par un faux positif dans la détection de fermeture de boucle.

Comme l'information de fermeture de boucle est utilisée par de nombreux algorithmes SLAM, ces problématiques sont très étroitement liées. De ce fait, la détection de fermeture de boucle n'est que peu étudiée en tant que telle. Détecter une boucle est bien souvent classé comme un problème de localisation globale. En effet, les méthodes utilisées pour la détection de fermeture boucle présentées dans les sections suivantes utilisent souvent une cartographie qu'elles vont comparer à l'observation courante. Il s'agit donc d'extraire d'une observation de l'environnement une position pour le robot. La détection de fermeture de boucle peut donc

4. L'aliasing perceptuel est le fait de rencontrer deux observations fortement similaires dans deux endroits différents.

être traitée comme un problème de localisation globale. Comme ces méthodes se basent toutes sur l'observation extérieure, information extéroceptive, cette section présentera tout d'abord du vocabulaire utilisé en traitement d'image (ces images pouvant être issues de caméra, sonar, lidar, ...) pour l'extraction d'informations importantes au traitement du problème concerné. Ensuite nous verrons les méthodes de localisation globale qui répondent le mieux à la détection de fermeture de boucle. Enfin, nous verrons quelques algorithmes dédiés à la détection de fermeture de boucle introduits dans des algorithmes SLAM. Une dernière partie conclura et introduira la contribution principale de cette thèse soit la détection de fermeture de boucle à partir de mesures proprioceptives.

#### Outils de traitement d'image

Les algorithmes présentés dans cette section utilisent l'observation de l'environnement pour connaître la position de l'observateur. Ces mesures extéroceptives peuvent être issues de différents types de capteurs : caméra monoculaire, plusieurs caméras, lidar [5], radar, sonar, télémètre et bien d'autres. Quel que soit le type du capteur, l'information qu'il nous donne est très souvent mise sous la forme d'une image : image de profondeur pour les capteurs retournant une distance, image issue d'une caméra ou bien image en trois dimensions lorsqu'il s'agit de stéréovision. Il apparaît donc important de mettre en place le vocabulaire utilisé en traitement d'image qui nous permettra de résoudre un problème de localisation globale comme celui de la détection de fermeture de boucle.

Une image étant un élément très riche, il convient d'en retirer une forme plus compacte contenant uniquement les informations utiles. Cette information est contenue dans un certain nombre de composantes pertinentes qui constituent les *primitives visuelles* de l'image. Les primitives sont caractérisées par des *descripteurs* d'image. Ces descripteurs sont nombreux et sont calculés sur la base des caractéristiques de l'image : couleur, texture, luminance, etc.

Les plus souvent rencontrés dans le traitement d'image en robotique (pour plus de détail, le lecteur est dirigé vers [Mikolajczyk et Schmid 2005]) sont :

— SIFT : Les descripteurs SIFT (*Scale Invariant Feature Transform*, [D. G. Lowe 2004]) sont très robustes aux changements de point de vue et aux changements d'échelle. Ils sont composés par les gradients aux alentours

5. LIght Detection And Ranging : permet de mesurer une distance à un objet par la mesure du temps d'aller retour d'un signal lumineux.

de différents points d'intérêt de l'image calculés sur plusieurs niveaux de profondeur.

— SURF : Les descripteurs SURF (Speeded Up Robust Feature, [Bay et al. 2006]) sont inspirés des SIFT, avec une vitesse de calcul plus rapide.

— Coins de Harris : Ces descripteurs [Harris et Stephens 1988] permettent de détecter des croisements de lignes francs dans une image. L'exemple le plus connu est celui d'une fenêtre pour laquelle on détecte les angles. Le calcul de ces détecteurs est aussi très rapide.

— Les caractéristiques globales d'une image, tels que les histogrammes de couleur, de teinte ou luminosité peuvent apporter des informations qualitatives d'une image mais peuvent aussi être utilisées en parallèle d'autres descripteurs. Dans ce cas leur utilisation permet d'obtenir ces descripteurs sous plusieurs espaces de représentation, les rendant plus robustes par rapport à telle ou telle caractéristique (i.e. en jouant sur la luminosité d'une image pour calculer des primitives SIFT par exemple, on rendra l'algorithme de détection moins sensible à des variations de luminosité).

Dans la partie suivante, nous avons choisi de remplacer le terme amer par le mot primitive pour des raisons phonétiques, même si cela constitue un abus de langage.

### 1.4.1 Comme un problème de localisation globale

#### 1.4.1.1 De l'observation vers la carte

Les méthodes présentées dans cette section sont des méthodes ascendantes : on extrait des informations de l'observation courante afin de les comparer aux informations connues de la carte.

**Appariement image / carte**

Le concept d'appariement image / carte (*image-to-map* en anglais) implique la connaissance des amers sur une carte. Ces amers sont définis par leurs informations quantitatives et/ou qualitatives sur la carte (par leurs primitives visuelles). C'est sur cette même carte que le robot va devoir se localiser en fonction de ses observations. Il s'agit donc bien d'une problématique de localisation globale.

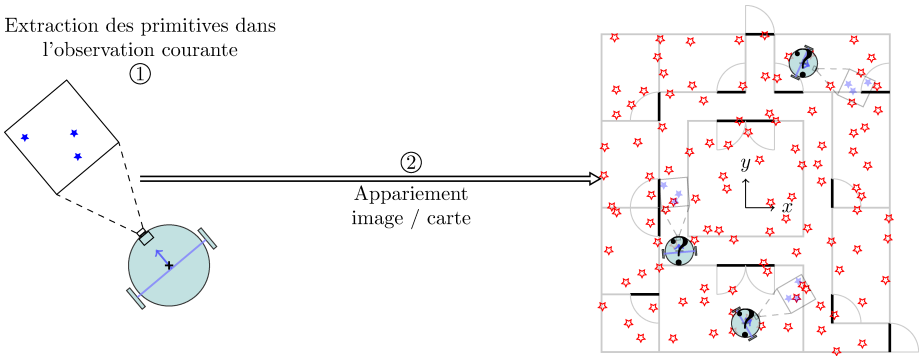



FIGURE 1.13 – Illustration du concept d'appariement image/carte, exemple sur carte métrique.

C'est dans ce cadre que les auteurs de [Se, D. Lowe et al. 2002] implémentent le concept d'appariement image/carte. À partir des primitives observées dans l'image courante, les amers de la carte (décrites par leurs primitives) les plus ressemblants sont utilisés pour obtenir la position du robot par reconstruction 3D. Les primitives de l'image courante sont construits par des descripteurs SIFT. Une procédure RANSAC, [Fischler et Bolles 1981] algorithme itératif d'estimation de paramètres en présence de valeurs aberrantes, est utilisée pour retrouver les amers de la carte globale qui permettent la meilleure reconstruction. Cette méthode de localisation peut être utilisée afin de détecter une boucle. Pour cela, il faut comparer les différentes localisations au cours du trajet et appairer les positions dont la distance métrique est faible lorsque le temps de trajet entre ces deux positions implique un changement de lieu. Détecter une fermeture de boucle en ce sens provoque donc une dépendance à la localisation. Lorsque la localisation est erronée, la détection de fermeture de boucle s'en retrouve perturbée.

Par une approche similaire, [B. P. Williams et al. 2007] cherchent à répondre à la problématique du kidnapping dans le cadre d'un SLAM monoculaire pour lequel le robot n'est équipé que d'une caméra : la stéréovision n'est donc pas possible. L'algorithme SLAM utilisé dans ces travaux [A. Davison 2003] construit une carte globale d'amers puis, lorsque la caméra est occultée ou lorsqu'un mouvement trop rapide de cette dernière est détecté (kidnapping), une phase de relocalisation est abordée. Tout d'abord, les primitives de l'image en cours sont calculées (détection des angles par la méthode de Shi-Tomasi [Shi et Tomasi 1994]) puis, une corrélation est effectuée entre les primitives ainsi obtenues et celles de la carte obtenues par l'algorithme SLAM. Le résultat de cette corrélation va permettre de

retenir les amers de la carte qui permettront une bonne reconstruction 3D de la position du robot. Cette reconstruction utilise une méthode RANSAC similaire à celle employée dans les travaux de [Se, D. Lowe et al. 2002] décrits précédemment. Dans ce cadre, la détection de fermeture de boucle est effectuée en tentant une relocalisation et en la comparant à la position estimée. Ce qui la fait dépendre encore une fois de l'algorithme de localisation. De plus, ces méthodes visuelles ont montré leurs limites lors de modifications importantes de l'environnement (luminosité, changement de point de vue trop important, ...).

#### Appariement carte locale / carte globale

L'appariement carte locale / carte globale (*map-to-map* en anglais) est issu de l'appariement image / carte. Les informations issues de l'observation courante vont être utilisées pour construire une sous-carte locale qui sera à son tour comparée aux sous-cartes composant la carte globale. Lorsque deux sous-cartes locale et globale s'avèrent très proches l'une de l'autre, on peut alors déduire un positionnement sur la carte globale.

La figure 1.14 illustre cette méthode : le robot va tout d'abord construire une sous-carte puis la comparer avec la carte globale elle-même composée de sous-cartes. Sur cette figure, le robot est dans la salle CF.

Les travaux de [Se, D. G. Lowe et al. 2005] prennent en compte plusieurs images successives qui sont utilisées afin de construire une sous-carte locale relative à la position courante. Les amers construits pour cette sous-carte sont alors comparés avec ceux des sous-cartes composant la carte globale. Le fait d'intégrer une plus grande quantité d'informations au voisinage de la position du robot pour construire une carte locale permet de diminuer les problèmes d'aliasing perceptuels.

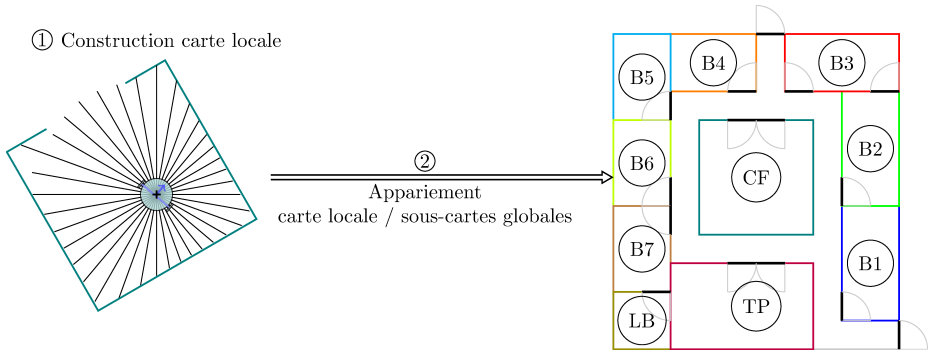



FIGURE 1.14 – Illustration du concept d'appariement carte locale/carte globale.

Dans [Clemente et al. 2007], les auteurs construisent des sous cartes locales par une méthode de SLAM utilisant un Filtre de Kalman Étendu (Extended Kalman Filter - EKF). Ces sous-cartes sont ensuite comparées avec la carte globale afin de trouver des correspondances entres sous-cartes locales et sous-cartes globales.

Dans ces deux cas, une plus grande quantité d'informations est utilisée dans le voisinage de la position estimée. Si l'on utilise ces méthodes, la problématique devient celle du recouvrement entre sous-cartes. De plus, dans [Clemente et al. 2007] la validité de l'association de données en cas de fermeture de boucle est assurée par l'algorithme GCBB (*Geometric Constraint Branch and Bound*, [Neira et al. 2003]) qui cherche à vérifier une contrainte d'une part sur la similarité entre les amers observés et d'autre part sur la distance relative entre les amers avant de valider la cohérence des appariements. Ces contraintes améliorent la robustesse de l'algorithme. En effet, une fausse détection de fermeture de boucle pouvant être catastrophique dans un algorithme SLAM, des vérifications doivent être effectuées avant de valider un appariement. Pour un algorithme SLAM qui vise à obtenir une carte des lieux visités, il sera plus acceptable de ne pas produire de faux positif, cela même au détriment du nombre de vraies boucles détectées (voir l'exemple présenté sur la figure 1.12).

### 1.4.1.2 De la carte vers l'observation

Les méthodes présentées dans cette section sont des méthodes descendantes. En fonction de la position estimée par un algorithme SLAM, les amers que l'on est censé voir doivent être retrouvés dans l'observation courante.

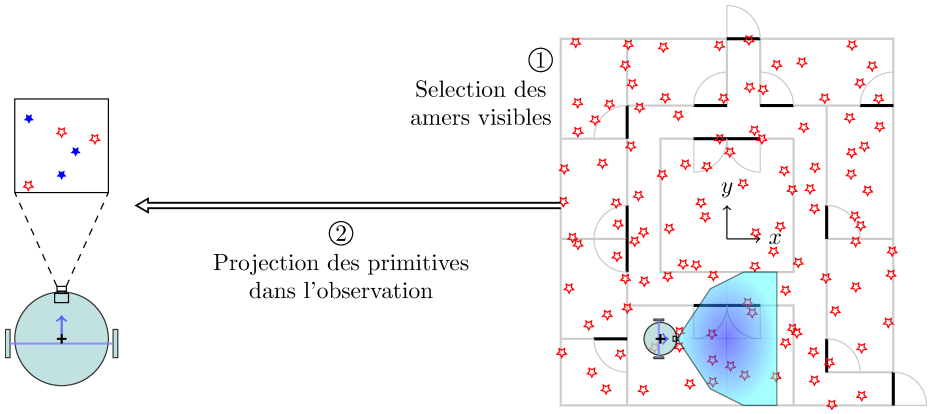



FIGURE 1.15 – Illustration du concept d'attention visuelle. Les étoiles bleues représentent les primitives de l'observation courante, les étoiles rouges les primitives projetées dans l'observation courante. Ces représentations sont abstraites, il n'existe en fait pas d'appariement sur cet exemple.

Sur la figure 1.15, à droite, on voit le robot ainsi qu'une zone colorée qui représente son environnement "visible". Les amers présents dans cette zone sont projetés (à travers les primitives visuelles qui les représentent) dans l'observation courante puis une comparaison est faite pour valider l'appariement.

**Appariement carte globale / amer local**

Dans les travaux de Davison [A. J. Davison et al. 2004], les primitives supposées visibles d'après l'estimation actuelle de la position sont projetées sur l'image courante et mises en concurrence avec les primitives de l'image. Ce mécanisme permet alors d'améliorer l'estimation de la position lorsque des primitives sont bien appairées. Cette méthode est ici appliquée à un algorithme SLAM et une caméra grand angle. D'une capture à l'autre, on retrouvera plus facilement - du fait du grand angle - des amers perçus dans l'image précédente et on pourra ainsi corriger l'estimation de position relativement à l'observation. Dans le cadre de la détection de fermeture de boucle appliquée au SLAM, il est courant de détecter une boucle lorsque deux positions éloignées dans le temps sont proches au sens métrique. Avec cette méthode, il sera plus aisé de discriminer une détection de fermeture de boucle lorsque des amers supposés visibles n'auront pas été repérés dans l'observation courante. Il faut tout de même noter que la dépendance à l'estimation de la position réduit le type de problème que ces approches, qualifiées d'approches *map-to-image*, sont capables de résoudre. Par exemple, que se

passe-t'il lors d'un kidnapping ?

#### 1.4.1.3 Approche reconnaissance d'image

Une autre approche consiste à faire l'appariement directement entre les images (approche *image-to-image*) : la méthode décrite dans [Cummins et Newman 2008b ; Cummins et Newman 2008a] vise à détecter une boucle lorsque l'on visite un lieu que l'on a déjà vu auparavant. Cette méthode fait la comparaison avec une approche type sac de mots. Cet algorithme se base sur une représentation topologique et permet de classer les lieux visités d'après l'observation. Cette approche de localisation topologique peut aussi être utilisée dans un SLAM métrique puisque l'algorithme ne travaille que sur les images. Cette approche nécessite toutefois l'apprentissage offline d'un bon vocabulaire avant de donner des résultats concrets. On peut envisager un SLAM métrique avec cet algorithme utilisé pour la détection de fermeture de boucle. C'est dans ce cadre que les auteurs de [B. Williams et al. 2009] implémentent cet algorithme et le comparent à deux autres approches.

Cette comparaison montre que l'appariement carte locale / carte globale utilisé dans [Clemente et al. 2007] permet de trouver de meilleurs alignements dans la carte (et donc une carte plus précise) lorsque des primitives fortement similaires sont présentes dans les observations. Le principal défaut de cette méthode est de trouver des primitives d'une observation à l'autre sans créer trop de faux positifs. Il apparaît que cet algorithme a besoin d'être adapté à la situation pour donner de bons résultats (*tuning*). L'appariement image / carte utilisé dans [B. Williams et al. 2008] nécessite une carte métrique précise et utilise beaucoup de ressources pour le calcul des primitives et les tests de correspondance avec la carte. Cette méthode donne elle aussi des résultats lorsqu'un bon *tuning* est effectué. Il faut noter la dépendance à l'environnement de ces algorithmes qui doivent tous être adaptés à la situation pour donner des résultats satisfaisants.

Les auteurs de [Ulrich et Nourbakhsh 2000] et [Wang et al. 2006] utilisent aussi une approche reconnaissance d'image dans le cadre de la localisation globale sur carte métrique. L'image courante est comparée aux images qui composent le modèle de l'environnement pour en détecter la provenance. Pour cela, une mesure de similarité est effectuée entre l'observation courante et chacune des images du modèle, un mécanisme de vote élit l'image qui obtiendra la plus grande similarité (le plus grand nombre de votes). Cet appariement requiert cependant

une phase d'apprentissage offline du modèle de l'environnement. La méthode de localisation développée dans [Booij et al. 2007] utilise ce même concept adapté à une représentation topologique.

Les travaux décrits dans [Granstrom et al. 2009] présentent un algorithme SLAM basé sur des observations issues de télémètres laser. À partir d'une observation, les auteurs construisent des primitives invariantes en rotation qui permettent de décrire un scan du laser (aire couverte par un scan, longueurs, courbures, ...). Ces primitives seront ensuite comparées afin d'en sortir un maximum de vraisemblance entre deux scans qui permettront de détecter une fermeture de boucle.

### 1.4.2 Algorithmes dédiés à la détection de fermeture de boucle

Il existe peu de méthodes dédiées à la détection de fermeture de boucle. La plupart des algorithmes SLAM se servent de l'estimation de leur position et vont utiliser une des méthodes de localisation globale présentées précédemment pour détecter une boucle. Les méthodes développées dans cette section cherchent à découpler les deux problèmes afin d'obtenir une information plus fiable.

Les travaux de [Frintrop et Cremers 2007] utilisent une approche *map-to-image* qui utilise le même mécanisme d'*attention visuelle* ([Itti et al. 1998], [Tsotsos et al. 1995], [Frintrop 2006]) que la méthode de [A. J. Davison et al. 2004] présentée en section 1.4.1.2, c'est-à-dire la projection des amers supposés visibles d'après l'estimation de la position dans l'image courante. Cette méthode vise à étudier la relation de composition entre l'image courante et les amers de la carte proches de la position estimée. Lorsqu'un amer est effectivement présent, il en résulte une forte zone de saillance au niveau de sa position. La détection utilisée dans cette méthode met en œuvre deux techniques : la première va sélectionner des régions d'intérêt (ROI - Region Of Interest) par le système d'attention visuelle développé dans [Frintrop 2006] et la seconde va utiliser les détecteurs de Harris-Laplace. Comme dans la méthode développée dans [Angeli 2008], l'utilisation de détecteurs multiples permet d'améliorer les performances globales de l'algorithme. Alors que les régions d'intérêt sont facilement détectables lors d'une observation, les détecteurs de Harris-Laplace vont isoler des points précisément dans une région d'intérêt. Dans ces travaux, les auteurs comparent l'approche décrite précédemment avec une approche ascendante pour la détection des primitives dans une image et concluent que l'approche descendante donne de meilleurs

résultats quant au nombre de bons appariements. Toutefois, les performances ne peuvent être comparées que lorsque l'on prend en compte un taux de faux positifs. Lorsque le taux de faux positifs est très bas, ce qui veut dire que seuls quelques faux positifs (voire aucun) sont acceptés, l'approche ascendante donne de meilleurs résultats alors que lorsque cette contrainte est minorée, c'est la méthode descendante qui prend le dessus (un meilleur taux de détection à taux de détection de faux positifs égal). Bien que cet algorithme soit développé pour la détection de fermeture de boucle, il reste dépendant de l'observation et est donc sujet à de fortes modifications pour la détection des primitives visuelles dans l'image, d'où l'apparition d'un taux de faux positifs significatif. De plus, comme toutes les approches descendantes, la méthode reste fortement couplée à l'algorithme SLAM pour l'estimation de la position qui conditionne les amers supposés visibles dans l'image.

La figure 1.11 présente des résultats issus de [Ho et Newman 2006]. L'algorithme SLAM utilisé ici se base sur une observation spatiale de l'environnement (images de profondeur issues d'un télémètre laser) alors que le processus de détection de fermeture de boucle utilise la caméra du robot (via des descripteurs SIFT). Le découplage de ces deux processus permet de ne pas prendre en compte l'erreur dans l'estimation de la position pour la détection de fermeture de boucle. Ainsi, lorsque le robot parcourt une distance relativement longue, ce qui engendre une erreur de localisation importante, et détecte une fermeture de boucle, il devient plus aisé de corriger le positionnement et d'en retirer une référence précise pour la construction de la carte. De plus, lors d'une détection de fermeture de boucle, les images de profondeur correspondant aux deux positions sont comparées afin de discriminer les cas d'aliasing perceptuels. C'est pour cela que la méthode dit comparer les apparences visuelle et spatiale. Comme dans la plupart des algorithmes dédiés à la détection de fermeture de boucle, on remarque l'utilisation de multiples comparaisons (ici, l'utilisation de plusieurs capteurs) qui améliore les résultats de la détection.

Dans leurs travaux, les auteurs de [Stachniss et al. 2004] (voir figure 1.10) détectent une boucle lorsque l'estimation de deux positions éloignées dans le temps sont très proches physiquement. Pour cela, les auteurs comparent la distance métrique entre deux points de la carte par rapport à la distance topologique entre ces deux points, c'est-à-dire la distance qu'a parcourue le robot pour se rendre d'un point à un autre. Si la distance topologique est grande lorsque la distance

métrique est petite, le robot se trouve dans une situation de fermeture de boucle. Cette information est alors utilisée pour contrôler les mouvements du robot afin qu'il effectue spécialement cette tâche de fermeture de boucle. Cette approche active permet alors d'améliorer l'estimation de position ainsi que la carte produite par l'algorithme FastSLAM [Montemerlo et Wegbreit 2003] utilisé ici.

Dans [Angeli 2008], les auteurs ont choisi une approche ascendante basée sur l'utilisation des sacs de mots visuels ([Filliat 2007], [Csurka et al. 2004]) pour la classification des primitives. Cette approche vise à rassembler des descripteurs d'images pour un lieu donné. La méthode est souple puisqu'elle permet de prendre en compte plusieurs types de primitives visuelles afin d'avoir une représentation la plus complète possible d'un lieu. Dans [Angeli 2008], des histogrammes locaux de teinte sont utilisés en plus de descripteurs SIFT pour obtenir une complémentarité dans la description (des descripteurs SIFT sont calculés dans plusieurs espaces de représentation). Une phase d'apprentissage peut être nécessaire pour la construction du dictionnaire/vocabulaire contenant les descripteurs de lieux mais il est possible d'effectuer cette phase online, permettant alors une utilisation de la méthode dans des lieux sans en avoir la connaissance *a priori*. Cette méthode se place dans le cadre probabiliste du filtrage Bayésien afin d'obtenir une probabilité de fermeture de boucle d'après une observation. La détection de fermeture de boucle est donc effectuée en reconnaissant ou non un lieu déjà visité. Comme la méthode [Cummins et Newman 2008b] décrite précédemment, on utilise ici une approche de classification d'image qui permet une certaine souplesse quant à l'utilisation de l'algorithme. En effet, elle est adaptée dans [Angeli 2008] pour un SLAM métrique aussi bien que pour un SLAM topologique.

## 1.5 Contribution principale

Les méthodes de détection de fermeture de boucle présentées précédemment utilisent toutes des informations extéroceptives, et plus particulièrement des caméras. La principale difficulté vient du phénomène d'aliasing perceptuel. En effet, lorsque deux lieux se ressemblent fortement, et en fonction de la qualité et de la pertinence de l'observation, ces méthodes seront plus ou moins capables d'identifier une fermeture de boucle. Cette dépendance à l'environnement rend cette tâche très difficile car une mauvaise détection de fermeture de boucle peut avoir des conséquences dommageables sur la construction d'une carte et sur la locali-

sation dans des algorithmes SLAM.

Une méthode qui ne se baserait pas sur l'observation extérieure mais bien sur l'évolution du robot pourrait apporter, sous réserve de fiabilité et de garantie des résultats, une réelle avancée dans ce domaine. Cette fiabilité peut être approchée, comme nous l'avons vu dans la méthode décrite dans [Clemente et al. 2007], par un jeu de contraintes qui aideront à la validation des appariements. Les méthodes qui découplent la détection de fermeture de boucle du SLAM vont aussi gagner en fiabilité, sous réserve de l'utilisation de descripteurs fiables et/ou par l'utilisation de multiples descripteurs. Cette fiabilité reste toutefois largement dépendante de l'observation et de la construction des descripteurs. On peut aussi relever la dépendance à l'algorithme SLAM auquel plusieurs de ces méthodes sont soumises. Lors d'une mauvaise estimation de la position, la détection de fermeture de boucle est inopérante.

Il existe des outils d'analyse numérique capables de produire un résultat garanti. Les travaux présentés dans ce manuscrit utilisent un de ces outils appelé analyse par intervalles qui est décrit dans le chapitre 2. Cet outil, utilisé dans une stratégie de détection de fermeture de boucle par mesures proprioceptives nous permettra de détecter la présence d'une boucle dans la trajectoire d'un robot autonome de manière garantie. Cette méthode est l'apport principal de cette thèse et sera décrite dans le chapitre 3 [Clemente et al. 2007 ; Jaulin, Kieffer, Didrit et al. 2001].

# Chapitre 2

# Analyse par intervalles

## Sommaire


**2.1 Introduction . . . . . 38**
**2.2 Ensembles, treillis et intervalles fermés . . . . . 39**
2.2.1 Théorie des ensembles . . . . . 39
2.2.1.1 Opérations ensemblistes . . . . . 40
2.2.1.2 Opérations mathématiques généralisées aux ensembles . . . . . 41
2.2.2 Relations d'ordre sur les ensembles . . . . . 42
2.2.2.1 Ordre total, ordre partiel . . . . . 43
2.2.3 Treillis . . . . . 43
2.2.4 Intervalles fermés . . . . . 45
**2.3 Intervalles réels . . . . . 46**
2.3.1 Notions de base . . . . . 46
2.3.2 Arithmétique des intervalles réels . . . . . 47
2.3.2.1 Opérations élémentaires . . . . . 47
2.3.2.2 Arithmétique et fonctions usuelles . . . . . 47
2.3.2.3 Fonctions d'inclusion . . . . . 48
**2.4 Intervalles de $\mathbb{R}^n$ . . . . . 50**
**2.5 Tubes . . . . . 51**
2.5.1 Définition . . . . . 52
2.5.1.1 Évaluation d'un tube . . . . . 52
2.5.2 Arithmétique des tubes . . . . . 56
2.5.3 Intégrale d'un tube . . . . . 56
2.5.4 Intégrale d'un tube avec bornes intervalles . . . . . 57
**2.6 Noyau d'une fonction intervalle . . . . . 60**
2.6.1 Noyau d'une fonction . . . . . 60
2.6.2 Caractérisation du noyau d'une fonction intervalle . . . 62
2.6.3 Algorithme . . . . . 66

2.6.4 Application à la localisation garantie par mesure de distance . . . . . 68
2.6.4.1 Localisation par mesure de distance . . . . . 68
2.6.4.2 Application . . . . . 70
**2.7 Recherche d'existence et d'unicité . . . . . 72**
2.7.1 Résolution de $f(x) = 0$ par la méthode de Newton . . . 73
2.7.2 Méthode de Newton par intervalles . . . . . 74
2.7.3 Propriété d'existence et unicité . . . . . 76
**2.8 Conclusion . . . . . 78**

## 2.1 Introduction

Tout utilisateur scientifique d'un ordinateur doit être conscient que la représentation d'un nombre dans ce dernier est finie. Se pose alors la question de l'arrondi. Une modélisation mathématique prenant en compte des variables mal connues, des paramètres incertains, des valeurs infiniment petites, des conversions de nombres à virgules flottantes peuvent entacher un calcul d'erreur. A ce moment, le scientifique se posera la question "Mais quelle est mon erreur ?". Alors que ses calculs sont justes, la représentation numérique, elle, ne l'est pas. Pour citer un exemple moderne des plus connus, le 4 juin 1996, une fusée Ariane 5 a explosé lors de son premier vol au bout de 40 secondes. Une commission scientifique a mis en cause une erreur de conversion entre deux calculateurs. D'autres exemples plus ou moins catastrophiques montrent bien que la représentation en virgule flottante n'est pas suffisante puisqu'une simple erreur d'arrondi peut avoir des conséquences désastreuses. Il est donc nécessaire d'utiliser un outil de représentation numérique des données qui garantisse l'exactitude des calculs numériques. Les théories ensemblistes, dont fait partie l'analyse par intervalles, nous permettent, au même titre que les probabilités, de représenter des variables peu ou mal connues, dépendantes entre elles ou non.

L'essor de la robotique autonome amène des questions de fiabilité et sécurité bien plus importantes que celles rencontrées dans le cas de machines téléopérées. En effet, dans le cadre du contrôle à distance d'une machine (que ce soit un robot téléopéré ou un outil sophistiqué), il est plus aisé de réagir à un événement imprévu puisque l'opérateur possède tout le contrôle et l'expérience nécessaires à cette tâche (sous réserve que l'outil soit fiable). Lorsqu'il s'agit d'un robot complètement autonome, ces situations doivent être prises en compte par une réaction

appropriée qui doit être adaptée au maximum de situations. Une autre façon de minimiser ces risques est de rendre le système le plus déterministe possible, par exemple en apportant une garantie mathématique aux calculs que le robot effectue lors de la planification des tâches et réactions qu'il rencontrera lors de sa mission. De cette façon, les risques liés à des résultats erronés, ou à une mauvaise interprétation de ces derniers, sont connus et le robot réagira de toute manière par une réaction prévue et contrôlée. C'est dans ce cadre que l'utilisation de l'analyse par intervalles prend tout son sens dans la robotique : la garantie des résultats diminue les sources d'erreurs (*i.e.* exemple de la fusée Ariane 5) et augmente les critères de fiabilité et sécurité. Dans notre contexte, l'analyse par intervalles sera utilisée pour qualifier de façon ensembliste les boucles possibles dans la trajectoire d'un robot mobile.

L'analyse par intervalles est un outil statistique de représentation des données. L'idée première est de représenter un flottant par un encadrement, un intervalle composé de bornes inférieure et supérieure qui sont représentables en machine. Dès lors, il faut développer une arithmétique propre à ces intervalles. Il est difficile de déterminer la paternité de cet outil, mais on peut aisément attribuer sa démocratisation à R.E. Moore en 1966 [R. E. Moore 1966]. Ce chapitre présente les bases de l'analyse par intervalles en commençant par les théories ensemblistes. Ensuite, nous verrons comment l'introduction de relations d'ordres dans ces ensembles nous amène aux intervalles réels et vers plusieurs extensions de ces derniers, à savoir les intervalles de $\mathbb{R}^n$ et les intervalles de fonctions - ou *tubes*. Une méthode permettant de caractériser le noyau d'une fonction sera ensuite proposée et nous finirons par un test d'existence et d'unicité couramment utilisé en analyse par intervalles.

## 2.2 Ensembles, treillis et intervalles fermés

### 2.2.1 Théorie des ensembles

La théorie des ensembles est une branche des mathématiques créée par le mathématicien allemand Georg Cantor à la fin du XIXe siècle. Cette théorie a unifié les mathématiciens par une idée simple : un ensemble est une *collection* d'objets définis et distincts qui sont les *éléments* de cet ensemble. L'idée maîtresse réside dans le fait qu'un objet appartient - ou non - à un ensemble. Les mathématiques se sont ainsi dotés de sigles particuliers qui désignent des ensembles : $\mathbb{R}$ pour l'en-

semble des réels, $\mathbb{N}$ pour l'ensemble des entiers naturels, etc. Dans notre étude, les ensembles seront un moyen de représenter des données incertaines alors qu'on utilise très souvent la théorie des probabilités pour cela.

Nous allons présenter ici quelques opérations, définies dans le contexte ensembliste ou bien étendues à ce même contexte. Nous nous limiterons aux outils utiles à la compréhension du document.

#### 2.2.1.1 Opérations ensemblistes

Soient deux ensembles $\mathcal{A}$ et $\mathcal{B}$. Les opérations ci-après sont définies dans un contexte ensembliste uniquement mais nous verrons qu'elles sont aisément transposables dans le cas des intervalles.

**L'intersection**

$\mathcal{A} \cap \mathcal{B} \triangleq \{x \mid x \in \mathcal{A} \text{ et } x \in \mathcal{B}\}$ : tous les éléments $x$ qui composent l'intersection de $\mathcal{A}$ et de $\mathcal{B}$ appartiennent à l'ensemble $\mathcal{A}$ et à l'ensemble $\mathcal{B}$.

**L'union**

$\mathcal{A} \cup \mathcal{B} \triangleq \{x \mid x \in \mathcal{A} \text{ ou } x \in \mathcal{B}\}$ : les éléments $x$ qui composent l'union de $\mathcal{A}$ et de $\mathcal{B}$ sont tous les éléments appartenant à l'ensemble $\mathcal{A}$ et tous les éléments appartenant à l'ensemble $\mathcal{B}$.

**La différence**

$\mathcal{A} \backslash \mathcal{B} \triangleq \{x \mid x \in \mathcal{A} \text{ et } x \notin \mathcal{B}\}$ (lire $\mathcal{A}$ *privé de* $\mathcal{B}$) : les éléments $x$ qui forment l'ensemble $\mathcal{A}$ privé de $\mathcal{B}$ sont tous les éléments qui appartiennent à l'ensemble $\mathcal{A}$ sans appartenir à l'ensemble $\mathcal{B}$.

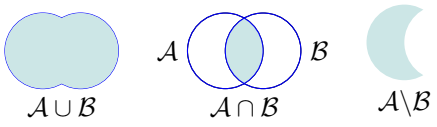



FIGURE 2.1 – Illustration de quelques opérations ensemblistes.

**Le produit cartésien**

$\mathcal{A} \times \mathcal{B} \triangleq \{(x, y) \mid x \in \mathcal{A} \text{ et } y \in \mathcal{B}\}$ : le produit cartésien est composé par les couples $(x, y)$ tels que $x \in \mathcal{A}$ et $y \in \mathcal{B}$.

**L'inclusion**

$\mathcal{A} \subset \mathcal{B} \Leftrightarrow \forall x \in \mathcal{A}\ ,\ x \in \mathcal{B}$ : l'inclusion de l'ensemble $\mathcal{A}$ dans l'ensemble $\mathcal{B}$ implique que tous les éléments appartenant à $\mathcal{A}$ appartiennent aussi à l'ensemble $\mathcal{B}$.

**L'égalité**

$\mathcal{A} = \mathcal{B} \Leftrightarrow (\mathcal{A} \subset \mathcal{B} \text{ et } \mathcal{B} \subset \mathcal{A})$ : l'égalité de deux ensemble $\mathcal{A}$ et $\mathcal{B}$ implique que $\mathcal{A}$ inclut $\mathcal{B}$ et $\mathcal{B}$ inclut $\mathcal{A}$

#### 2.2.1.2 Opérations mathématiques généralisées aux ensembles

Soient deux ensembles $\mathcal{A}$ et $\mathcal{B}$ et une fonction $f : \mathcal{A} \to \mathcal{B}$.

**L'image directe**

Si $\mathcal{A}_1 \subset \mathcal{A}$ alors *l'image directe* de $\mathcal{A}_1$ par la fonction $f$ est

$$f(\mathcal{A}_1) \triangleq \{f(x) \in \mathcal{B} \mid x \in \mathcal{A}_1\}. \tag{2.2.1}$$

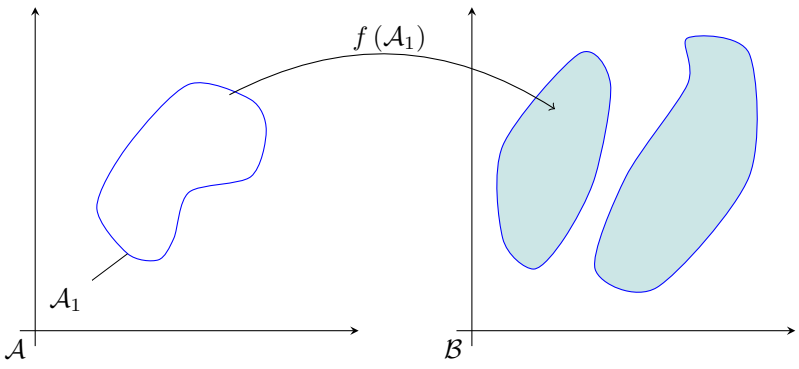



FIGURE 2.2 – Image directe.

La figure 2.2 nous montre l'image directe d' une fonction $f : \mathcal{A} \to \mathcal{B}$. L'ensemble $\mathcal{A}_1$ est inclus dans l'ensemble $\mathcal{A}$, symbolisé par les axes de la figure de

gauche. L'image directe de l'ensemble $\mathcal{A}_1$ par la fonction $f$ est l'ensemble grisé sur la figure de droite, qui est inclus dans l'ensemble $\mathcal{B}$, représenté par les axes de la figure de droite.

**L'image réciproque**

Si $\mathcal{B}_1 \subset \mathcal{B}$ alors *l'image réciproque* de $\mathcal{B}_1$ par la fonction $f$ est

$$f^{-1}(\mathcal{B}_1) \triangleq \{x \in \mathcal{A} \mid f(x) \in \mathcal{B}_1\}. \tag{2.2.2}$$

Ce qui implique que l'image de l'ensemble vide - noté $\emptyset$ - est l'ensemble vide et que son image réciproque est aussi l'ensemble vide :

$$f(\emptyset) = f^{-1}(\emptyset) = \emptyset. \tag{2.2.3}$$

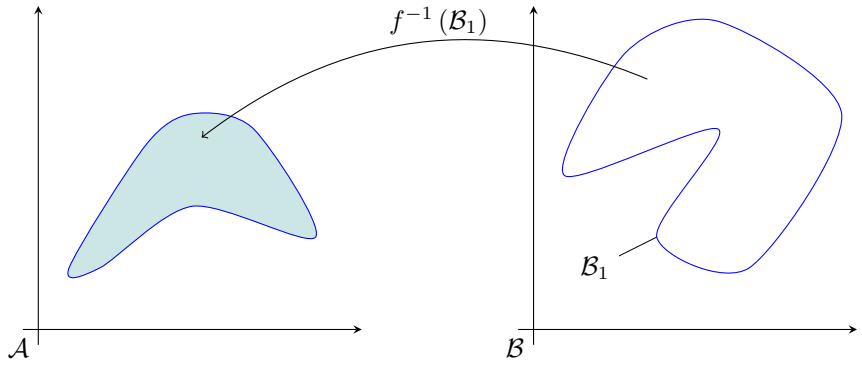



FIGURE 2.3 – Image réciproque.

La figure 2.3 nous montre l'image réciproque d'un ensemble $\mathcal{B}_1 \subset \mathcal{B}$, l'ensemble $\mathcal{B}$ étant représenté par les axes de la figure de droite. Cette image réciproque est composée des éléments $x$ - en bleu - de l'ensemble $\mathcal{A}$ (axes de la figure de gauche) qui donnent $f(x) \in \mathcal{B}_1$.

### 2.2.2 Relations d'ordre sur les ensembles

La section précédente nous rappelle quelques définitions sur les ensembles qui sont à l'origine de l'analyse par intervalles. Avant de passer aux intervalles, il est nécessaire de décrire les relations d'ordre qui régissent ces ensembles [Jaulin, Kieffer, Didrit et al. 2001].

Une relation $\mathbf{R}$ est une *relation d'ordre* définie sur un ensemble $\mathcal{A}$ si pour tout élément $x, y, z \in \mathcal{A}$, on a :

Réflexivité : $x \ \mathbf{R} \ x$.

Antisymétrie : $x \ \mathbf{R} \ y$ et $y \ \mathbf{R} \ x \Rightarrow x = y$.

Transitivité : $x \ \mathbf{R} \ y$ et $y \ \mathbf{R} \ z \Rightarrow x \ \mathbf{R} \ z$.

**Définition 2.2.1.** *Un ensemble $\mathcal{A}$ muni d'une relation d'ordre $\boldsymbol{R}$ est un* ensemble ordonné.

**Exemple 2.2.1.** *L'ensemble $\mathbb{E} = \{1, 2, 3, ...\}$ est un ensemble ordonné s'il est muni de la relation d'ordre $\leq$.*

#### 2.2.2.1 Ordre total, ordre partiel

**Définition 2.2.2.** *Un ensemble $\mathcal{A}$ muni d'une relation d'ordre $\boldsymbol{R}$ est dit* totalement ordonné *si pour tout élément $x, y \in \mathcal{A}$ on a soit $x \ \boldsymbol{R} \ y$, soit $y \ \boldsymbol{R} \ x$.*

Cette définition signifie que dans le cas d'un ensemble totalement ordonné, deux éléments de cet ensemble sont toujours comparables. Une relation d'ordre sera qualifiée de partielle lorsqu'elle n'est pas totale.

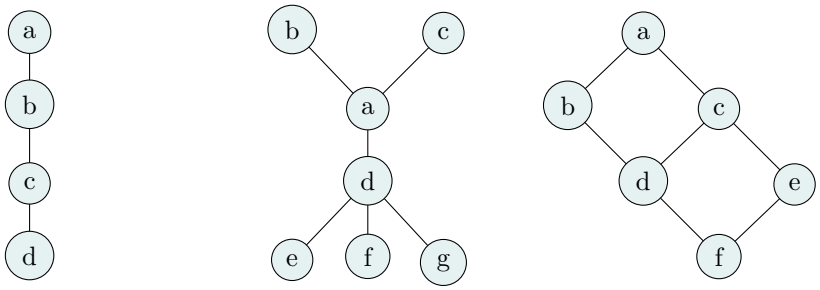
**Exemple 2.2.2.** *Exemples de relations d'ordre total / partiel :*

- *L'ensemble $\mathbb{N}$ des entiers naturels est totalement ordonné par la relation $\leq$. Pour tous les éléments de $\mathbb{N}$, une comparaison par la relation $\leq$ est possible. Pour tout élément $a$, $b$ de $\mathbb{N}$ $b \not\leq a$ implique $a \leq b$.*
- *La relation d'inclusion $\subset$ pour les ensembles est une relation d'ordre partielle puisque $\mathcal{A} \not\subset \mathcal{B}$ ne veut pas nécessairement dire que $\mathcal{B} \subset \mathcal{A}$.*

### 2.2.3 Treillis

**Définition 2.2.3.** *Un treillis $(\mathcal{E}, \leq)$ est un ensemble partiellement ordonné. Pour chaque $(x, y) \in \mathcal{E}$, on admet une borne inférieure $x \wedge y \in \mathcal{E}$ et une borne supérieure $x \vee y \in \mathcal{E}$.*

Un treillis est en fait un ensemble partiellement ordonné avec une structure particulière. Cette particularité tient aux bornes inférieure et supérieure. La figure 2.4 montre la distinction entre ensembles totalement/partiellement ordonnés et la structure particulière d'un treillis. Sur cette figure, les arêtes entre les nœuds représentent les arcs du treillis ; les différents exemples de cette figure doivent être lus de bas en haut.




(a) Ensemble totalement ordonné. (b) Ensemble ordonné. (c) Treillis.

FIGURE 2.4 – Ensembles ordonnés et treillis.

Le lecteur trouvera plus de détails sur les ensembles ordonnés et treillis dans [Davey et Priestley 2002]. Un treillis $\mathcal{E}$ est dit complet si et seulement si pour tout sous-ensemble (fini ou non) $\mathcal{A}$ de $\mathcal{E}$, la borne supérieure notée $\wedge\mathcal{A}$ et la borne inférieure notée $\vee\mathcal{A}$ appartiennent à $\mathcal{E}$. Il est possible d'ajouter de nouveaux éléments à un treillis non borné pour le rendre borné.

**Exemple 2.2.3.** *L'ensemble $\mathbb{R}^n$ est un treillis lorsqu'il est doté de la relation d'ordre $\leq$ (aussi appelée relation d'ordre naturelle et souvent omise dans la notation).*
*Pour tout $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$*

$$\mathbf{x} \leq \mathbf{y} \Leftrightarrow \forall i \in \{1, \ldots, n\}, x_i \leq y_i.$$

*On a*

$$\mathbf{x} \wedge \mathbf{y} = (x_1 \wedge y_1, \ldots, x_n \wedge y_n)$$

*et*

$$\mathbf{x} \vee \mathbf{y} = (x_1 \vee y_1, \ldots, x_n \vee y_n),$$

$$\text{où} \begin{cases} x_i \wedge y_i = \min(x_i, y_i) \\ x_i \vee y_i = \max(x_i, y_i) \end{cases}.$$

**Exemple 2.2.4.** *Le treillis $(\mathbb{R}^n, \leq)$ n'est pas borné alors que $\left(\overline{\mathbb{R}^n}, \leq\right)$ défini par $\overline{\mathbb{R}^n} = \mathbb{R}^n \cup \{-\infty, +\infty\}^n$ l'est.*

**Proposition 2.2.1.** *Soient deux treillis $(\mathcal{E}_1, \leq_1)$ et $(\mathcal{E}_2, \leq_2)$. Le produit $\mathcal{E}_1 \times \mathcal{E}_2$ est le treillis $(\mathcal{E}, \leq)$ avec la relation d'ordre $\leq$ défini par $(a_1, a_2) \leq (b_1, b_2) \Leftrightarrow ((a_1 \leq_1 b_1)$ et $(a_2 \leq_2 b_2))$ pour tous les ensembles $(x_1, x_2) \in \mathcal{E}_1 \times \mathcal{E}_2$.*

**Exemple 2.2.5.** *On s'intéresse ici à l'ensemble $E = \{a, b, c\}$ et $S$ l'ensemble de tous les sous-ensembles de $E$ équipés de l'inclusion. La figure 2.5 représente cet ensemble $S$.*

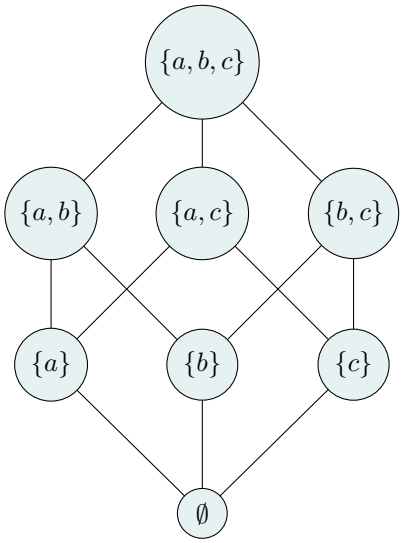



FIGURE 2.5 – L'ensemble $S$

*Chacun des nœuds représente un élément de l'ensemble $S$ et la relation d'inclusion est représentée par les arcs entre les différents nœuds. Si nous choisissons deux éléments du graphe, par exemple $\{a\}$ et $\{b\}$, on a forcément une borne supérieure, ici l'élément $\{a, b\}$ et une borne inférieure, $\{\emptyset\}$. Ces bornes sont amenées par la relation d'ordre qui compose ce treillis : $\{a\} \subset \{a, b\}$, $\{b\} \subset \{a, b\}$, $\{\emptyset\} \subset \{a\}$ et $\{\emptyset\} \subset \{b\}$.*

### 2.2.4 Intervalles fermés

La théorie des ensembles ainsi que celle des treillis nous permettent maintenant de définir les intervalles de façon plus générale. Il faut avant cela définir l'enveloppe intervalle.

**Définition 2.2.4** (Enveloppe intervalle)**.** *L'enveloppe intervalle d'un sous-ensemble $\mathcal{A}$ d'un treillis $\mathcal{E}$ est le plus petit intervalle $[\mathcal{A}]$ contenant $\mathcal{A}$ :*

$$[\mathcal{A}] = [\wedge\mathcal{A}, \vee\mathcal{A}]\,,$$

*et $\mathcal{A}$ est un intervalle si et seulement si $\mathcal{A} = [\mathcal{A}]$.*

Un intervalle $[\mathcal{A}]$ d'un treillis borné $\mathcal{E}$ est un sous-ensemble de $\mathcal{E}$ qui satisfait $[\mathcal{A}] = \{x \in \mathcal{E} \mid \wedge \mathcal{A} \leq x \leq \vee \mathcal{A}\}$. Les ensembles $\emptyset$ et $\mathcal{E}$ sont des intervalles de $\mathcal{E}$. Un intervalle est donc un sous-treillis borné de $\mathcal{E}$.

L'ensemble de tous les intervalles de $\mathcal{E}$ est noté $\mathbb{I}\mathcal{E}$. Un intervalle $[x]$ de $\mathcal{E}$ sera donc noté $[x] = [\wedge [x], \vee [x]]_{\mathcal{E}}$. Ce qui nous amène aux notations plus classiques utilisées en analyse par intervalles où la borne inférieure de $[x]$, $\wedge [x]$, sera notée $\underline{x}$, $x^-$ ou $lb([x])$ (*lb* pour *lower bound*) et la borne supérieure, $\vee [x]$, sera notée $\overline{x}$, $x^+$ ou $ub([x])$ (*ub* pour *upper bound*).

Un intervalle tel que défini ici est donc un sous-ensemble fermé connexe de $\mathcal{E}$. L'appellation *intervalle* utilisée dans cette thèse est un abus de langage. Pour être plus exact, nous devrions parler d'*intervalle fermé* mais par soucis de concision, le terme intervalle est employé.

## 2.3 Intervalles réels

Pour définir l'analyse par intervalles, il est courant de se placer dans le treillis $\left(\overline{\mathbb{R}}, \leq\right)$, avec $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ Cette section présentera donc les intervalles fermés de $\mathbb{R}$ que nous appellerons intervalles réels.

### 2.3.1 Notions de base

Un intervalle réel est un sous-ensemble fermé connexe de $\mathbb{R}$. L'ensemble des intervalles de $\mathbb{R}$ est noté $\mathbb{IR}$. La notation la plus classique, utilisée dans ce manuscrit, pour un intervalle $[x]$ de $\mathbb{R}$ est :

$$[x] = \left[x^-, x^+\right] = \left\{x \in \mathbb{R} \mid x^- \leq x \leq x^+\right\}.$$

**Exemple 2.3.1.** *Quelques exemples*

- $[0, 1]$ *est un intervalle,*
- $[-5, +\infty]$ *est un intervalle,*
- $]-3, 5]$ *n'est pas un intervalle car il n'est pas fermé,*
- $\{2\}$ *est un intervalle.*

Certaines propriétés des intervalles sont définies telles :
le centre $mid([x]) = \frac{x^- + x^-}{2}$,

le rayon $rad([x]) = \frac{x^+ - x^-}{2}$,

la longueur $width([x]) = x^+ - x^-$.

Un intervalle dont la longueur est 0 sera appelé intervalle *dégénéré* : dans l'exemple précédent, l'ensemble $\{2\}$ est un intervalle dégénéré. Plus souvent, on écrira 2 (on trouve parfois la notation $[2]$ et même $[2, 2]$).

### 2.3.2 Arithmétique des intervalles réels

#### 2.3.2.1 Opérations élémentaires

Il est possible d'appliquer les opérations ensemblistes définies à la section 2.2.1.1 aux intervalles.
Soient $[x]$ et $[y]$ deux intervalles de $\mathbb{R}$.

**L'intersection**

$[x] \cap [y] \triangleq \{z \in \mathbb{R} \mid z \in [x] \text{ et } z \in [y]\}$. L'intersection de deux intervalles donnera toujours un intervalle. L'ensemble vide, $\emptyset$, est considéré comme un intervalle.

**L'union**

$[x] \cup [y] \triangleq \{z \in \mathbb{R} \mid z \in [x] \text{ ou } z \in [y]\}$. Contrairement à l'intersection, l'union de deux intervalles peut ne pas être un intervalle.

**Exemple 2.3.2.** *L'union* $[0, 1] \cup [3, +\infty]$ *n'est pas un intervalle alors que l'union* $[0, 3] \cup [1, +\infty] = [0, +\infty]$ *en est un.*

**L'union intervalle**

$[x] \sqcup [y] \triangleq [[x] \cup [y]]$. Cette union intervalle utilise la définition 2.2.4 et permet de garder l'ensemble des intervalles fermés. Elle produit le plus petit intervalle contenant $[x] \cup [y]$.

#### 2.3.2.2 Arithmétique et fonctions usuelles

L'arithmétique des réels est applicable aux intervalles. Si $\diamond$ est un des quatre opérateurs réels (l'addition, la soustraction, la multiplication et la division), l'ex-

tension de ces opérations à deux intervalles $[x]$ et $[y]$ est définie par

$$[x] \diamond [y] = [\{x \diamond y \mid x \in [x], y \in [y]\}]. \tag{2.3.1}$$

Il est aussi possible d'appliquer aux intervalles les fonctions continues élémentaires utilisées pour les réels, telles que $\sin, \cos, \tan, \sqrt{}, \exp, ...$, on définit :

$$f([x]) \overset{2.2.1}{=} \{f(x) | x \in [x]\}. \tag{2.3.2}$$

Le lecteur est invité à consulter les ouvrages [Jaulin, Kieffer, Didrit et al. 2001 ; R. E. Moore 1979 ; Alefeld et Herzberger 1983] pour connaître les bases de l'arithmétique par intervalles.

#### 2.3.2.3 Fonctions d'inclusion

**Définition 2.3.1** (Fonction d'inclusion)**.** *Une fonction* $[f] : \mathbb{IR} \to \mathbb{IR}$ *est une fonction d'inclusion pour la fonction* $f : \mathbb{R} \to \mathbb{R}$ *si et seulement si*

$$f([x]) \subset [f]([x]), \forall [x] \in \mathbb{IR}. \tag{2.3.3}$$

**Définition 2.3.2** (Monotonie)**.** *Une fonction d'inclusion est dite monotone si elle satisfait*

$$[x] \subset [y] \Rightarrow [f]([x]) \subset [f]([y]), \forall [x] \in \mathbb{IR} \text{ et } \forall [y] \in \mathbb{IR}. \tag{2.3.4}$$

**Définition 2.3.3** (Convergence)**.** *Une fonction d'inclusion est dite convergente si elle satisfait*

$$width([x]) \to 0 \Rightarrow width([f]([x])) \to 0, \forall [x] \in \mathbb{IR}. \tag{2.3.5}$$

La façon la plus simple pour construire une fonction d'inclusion est d'y remplacer les occurrences réelles de la variable par un intervalle : les fonctions d'inclusions construites de la sorte sont appelées *fonctions d'inclusion naturelles*. Comme nous pouvons écrire une fonction de plusieurs manières, il est possible d'obtenir plusieurs fonctions d'inclusion pour une même fonction. L'exemple 2.3.3 nous montre plusieurs fonctions d'inclusion naturelles pour une même fonction $f$.

**Exemple 2.3.3.** *Effet de la multi-occurrence sur une fonction d'inclusion*

$$
\begin{aligned}
[f_1]([x]) &= ([x]+4)([x]^2-9)([x]^2-1)+2,\\
[f_2]([x]) &= [x]^5+4[x]^4-10[x]^3-40[x]^2+9.[x]+38,\\
[f_3]([x]) &= ([x]+4)([x][x]-9)([x][x]-1)+2.
\end{aligned}
$$

Parmi celles ci, aucune n'est minimale au sens de l'inclusion. Il est rare que les fonctions d'inclusion soient minimales au sens de l'inclusion, ceci nous permet de définir le pessimisme d'une fonction d'inclusion, plus le pessimisme est grand, plus l'encadrement autour de la valeur réelle sera grand. Une façon bien connue de diminuer le pessimisme d'une fonction d'inclusion est de diminuer le nombre d'occurrences des variables à l'intérieur de celle-ci. La figure 2.6 illustre bien ce phénomène.

Si on évalue chacune des expressions de l'exemple 2.3.3 sur l'intervalle $[x] = [-1.5, 1.5]$, on obtient :

$$
\begin{aligned}
[f_1]([x]) &= [-59.8751, 51.5],\\
[f_2]([x]) &= [-106.8438, 113.0938],\\
[f_3]([x]) &= [-75.3438, 203.0938],
\end{aligned}
$$

comme nous le montre la figure 2.6. Ici, la fonction $[f_1]$ est la moins pessimiste. On peut remarquer que cette fonction d'inclusion est celle qui possède le moins d'occurrences de la variable $[x]$ dans son expression.

**Remarque 2.3.1.** *L'exemple 2.3.3 présenté ci dessus nous montre l'effet de la multi-occurence sur la qualité de l'évaluation d'une fonction d'inclusion. Ceci n'est pas une règle mais une tendance largement utilisé.*

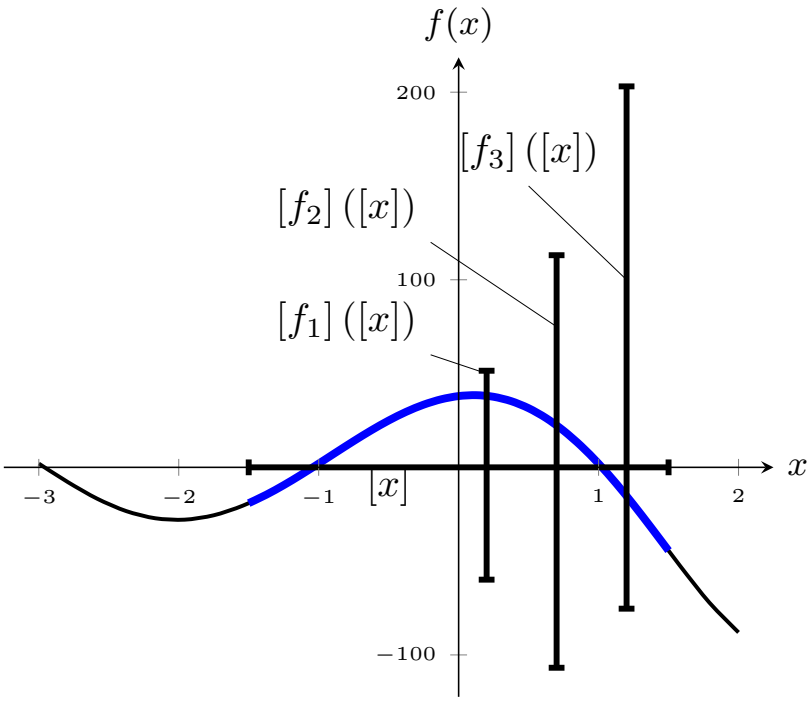



FIGURE 2.6 – Efficacité des fonctions d'inclusion.

## 2.4 Intervalles de $\mathbb{R}^n$

Les intervalles ne se bornent pas aux intervalles réels, on peut définir un intervalle de $\mathbb{R}^n$ comme un vecteur d'intervalles réels. Il faut, pour cela, se placer dans le treillis $(\mathbb{R}^n, \leq^n)$. Ces intervalles vectoriels sont appelés *boîtes*. Ces boîtes sont, en fait, le produit cartésien de $n$ intervalles réels. Par convention, on notera en gras un intervalle de $\mathbb{R}^n$ :

$$[\mathbf{x}] = [x_1] \times [x_2] \times ... \times [x_n]\,, \tag{2.4.1}$$

avec

$$[x_i] = \left[x_i^-, x_i^+\right] \text{ pour } i = \{1, ..., n\}.$$

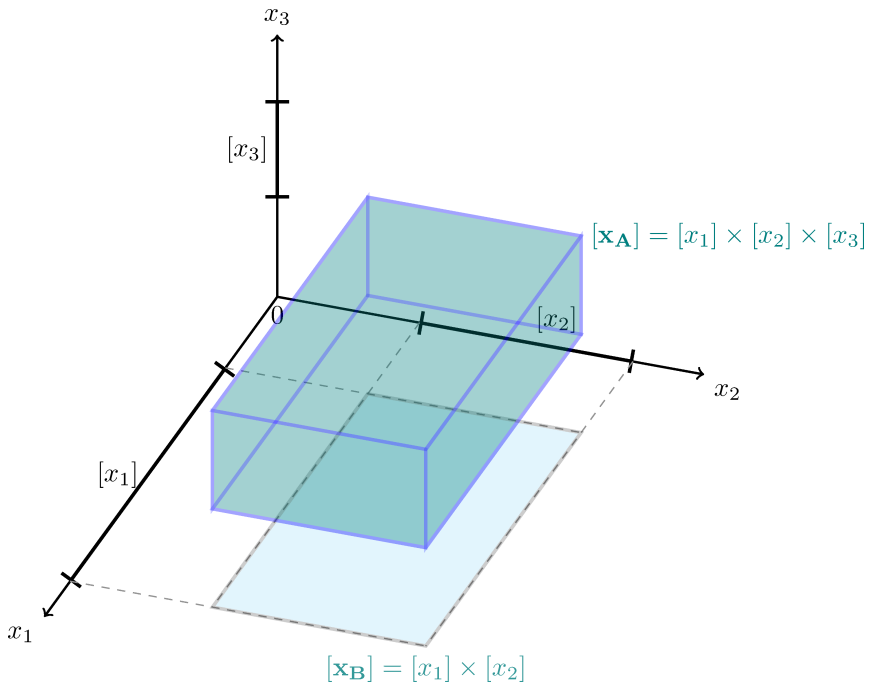



FIGURE 2.7 – Représentation d'un intervalle $[\mathbf{x_A}]$ de $\mathbb{R}^3$ en bleu, et d'un intervalle $[\mathbf{x_B}]$ de $\mathbb{R}^2$ en cyan.

Les intervalles réels permettent de travailler avec des problèmes en dimension $n$. La figure 2.7 montre la représentation d'une boîte de $\mathbb{R}^3$. Il est aussi possible d'adapter l'arithmétique des intervalles réels aux intervalles de $\mathbb{R}^n$. Ainsi, on définit le centre ou le rayon d'un intervalle de $\mathbb{R}^n$ comme des vecteurs de $n$ réels, l'élément $i$ représentant respectivement le centre ou le rayon du composant $i$ de la boîte pour $i \in \{1, ..., n\}$. La taille d'une boîte sera quant à elle interprétée comme la taille maximale parmi les tailles des intervalles qui la composent, plus formellement :

$$width\left([\mathbf{x}]\right) \stackrel{\Delta}{=} \max_{i=1,...,n} \left(width\left([x_i]\right)\right). \tag{2.4.2}$$

## 2.5 Tubes

En nous plaçant dans le treillis $\left(\overline{\mathbb{R}}, \leq\right)$, nous avons vu les intervalles réels. Ensuite, dans le treillis $\left(\overline{\mathbb{R}^n}, \leq^n\right)$, nous avons vu les intervalles vectoriels, les boîtes. Si l'on se place maintenant dans le treillis des fonctions de $\mathbb{R} \to \mathbb{R}^n$, muni

de la relation d'ordre naturelle, nous allons introduire les intervalles de fonctions de $\mathbb{R} \to \mathbb{R}^n$, autrement appelés *tubes*.

### 2.5.1 Définition

L'ensemble $\mathcal{F}^n$ des fonctions de $\mathbb{R} \to \overline{\mathbb{R}^n}$ est un treillis borné muni de l'ordre $\mathbf{f} \leq \mathbf{g} \Leftrightarrow \forall t \in \mathbb{R}, \mathbf{f}(t) \leq \mathbf{g}(t)$.

Un tube $[\mathbf{f}]$ [Kurzhanski et Valyi 1997; LeBars et al. 2012] est un intervalle $\left[\mathbf{f}^-, \mathbf{f}^+\right]$ de $\mathcal{F}^n$. C'est en fait une paire de fonctions $\mathbf{f}^-, \mathbf{f}^+$ telles que $\forall t \in \mathbb{R}, \mathbf{f}^-(t) \leq \mathbf{f}^+(t)$. La figure 2.8 nous montre une fonction $f(t) \in \mathcal{F}^1$ contenue dans un tube $[f](t) = [f^-(t), f^+(t)]$. Un tube est une façon de représenter une fonction incertaine. On peut considérer qu'un tube est l'analogue, dans le contexte ensembliste de ce qu'est un signal aléatoire dans un contexte probabiliste.

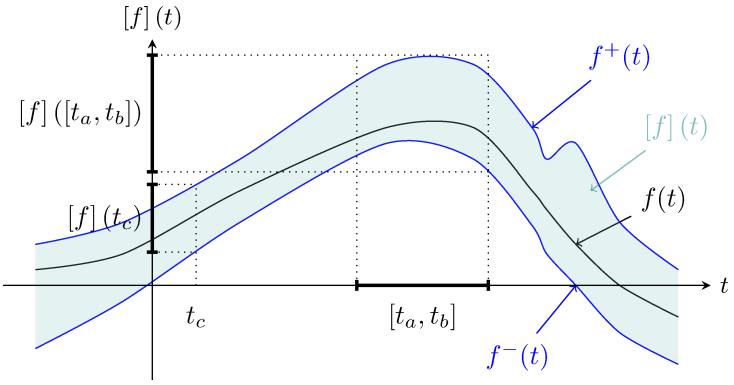



FIGURE 2.8 – Un tube $[f](t)$ pour la fonction $f(t)$

**Remarque 2.5.1.** *Nous présentons ici les tubes, intervalles de fonctions de* $\mathbb{R} \to \mathbb{R}^n$. *Ces tubes sont un cas particuliers des intervalles de fonctions.*

#### 2.5.1.1 Évaluation d'un tube

Pour une fonction $\mathbf{f} : \mathbb{R} \to \mathbb{R}^n$, $\mathbf{f} \in \mathcal{F}^n$, l'évaluation de cette fonction sur l'intervalle $t$ est définit par $\mathbf{f}([t]) = \{\mathbf{f}(t) \,|t \in [t]\}$ (équation 2.3.2.2 page 48). Dans le cadre des tubes, l'évaluation intervalle $[\mathbf{f}]([t])$ d'un tube $[\mathbf{f}]$ est définie par la plus petite boîte qui contient l'ensemble $\{[\mathbf{f}](t), t \in [t]\}$.

L'évaluation intervalle $[\mathbf{f}]([t])$ d'un tube est alors définie par :

$$[\mathbf{f}]([t]) = \bigsqcup_{t\in[t]} [\mathbf{f}](t)\,. \tag{2.5.1}$$

Si $\mathbf{f} \in [\mathbf{f}]$ et $t \in [t]$ alors $\mathbf{f}(t) \in [\mathbf{f}](t)$ et aucune autre boîte plus petite ne satisfait cette condition.

On peut voir sur la figure 2.8 l'évaluation du tube $[f](t)$ sur deux intervalles : $[t_a, t_b]$ et $(t_c)$.

Si un tube peut être représenté par une paire de fonctions $f^-, f^+$, cette représentation nécessite l'expression analytique des fonctions $f^-$ et $f^+$. Lorsque ces expressions ne sont pas - ou peu - connues, il convient d'utiliser une représentation plus "numérique". Cette représentation peut, par exemple, être utilisée pour encadrer les mesures d'un capteur pour lequel on saurait borner l'erreur.

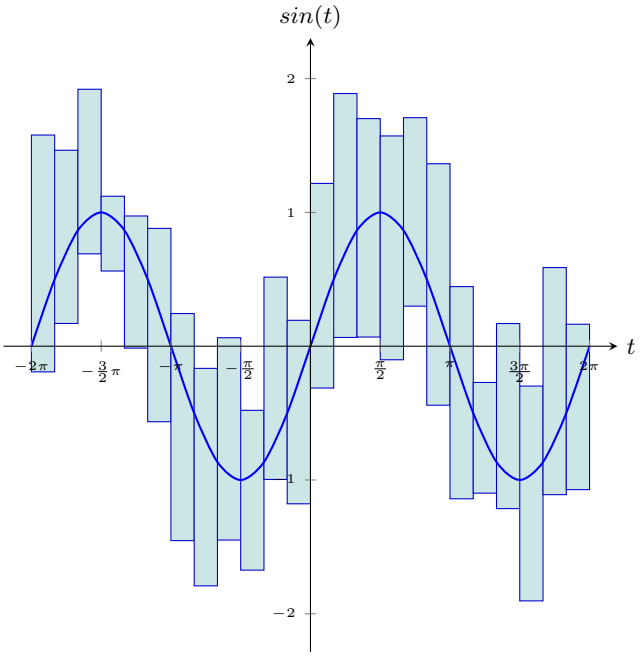



FIGURE 2.9 – Illustration d'un tube

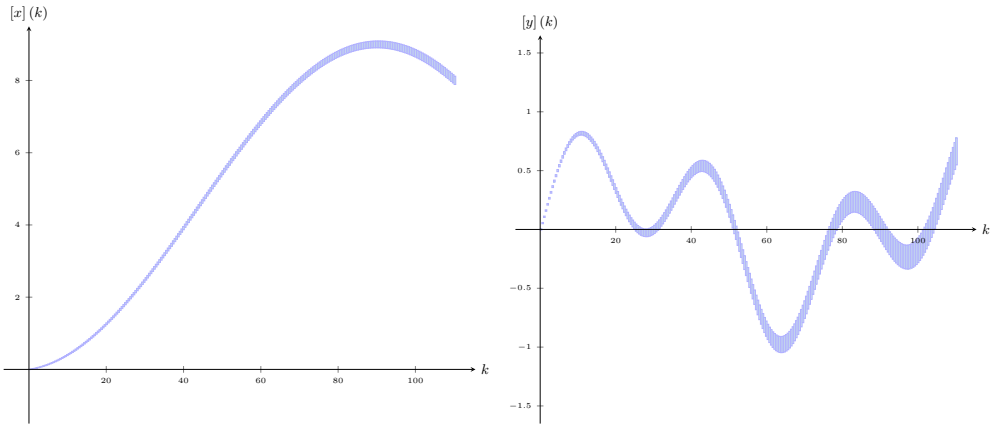
La représentation discrète d'un tube est donnée à la figure 2.9. Un tube discret avec un pas de temps $\delta$ est composé de différentes boîtes appelées *tranches* (*slice* en anglais). La $k^{ième}$ tranche est la boîte $[k\delta, k\delta + \delta] \times [\mathbf{f}](t_k)$ pour $t_k \in [k\delta, k\delta + \delta]$ et sera notée $[\mathbf{f}](k)$.

**Exemple 2.5.1.** *Considérons un robot mobile (type char) évoluant sur un plan et régi par les équations d'état suivantes :*

$$\begin{aligned} \dot{x} &= v\cos(\theta), \\ \dot{y} &= v\sin(\theta), \\ \dot{\theta} &= R.\frac{u_2 - u_1}{l}, \end{aligned} \tag{2.5.2}$$

*où $v$ représente sa vitesse, $\theta$ son cap et $u_1, u_2$ les commandes en accélération de ses roues motrices. Le robot connaît son état d'origine $x = (0, 0, 0)^T$. Nous émettons ici l'hypothèse que les capteurs qui donneront les informations au robot sont à erreurs bornées connues [Meizel et al. 1996].*

*Ce robot peut, par la connaissance de ces équations d'état et de son état d'origine, représenter sa position à l'instant $t$ sous la forme de tubes. Les figures 2.10(a) et 2.10(b) représentent respectivement le tube pour la position selon l'axe $x$ et celui pour la position selon l'axe $y$.*




(a) (b)

FIGURE 2.10 – En (a) le tube pour la position selon l'axe $x$. En (b) le tube pour la position selon l'axe $y$.

*Sur la figure 2.11, la position du robot est affichée. Cette position est une représentation conjointe des tubes $[x](k)$ et $[y](k)$. Chacune des boîtes représente donc la position $(x, y)$ du robot à un instant $k$.*

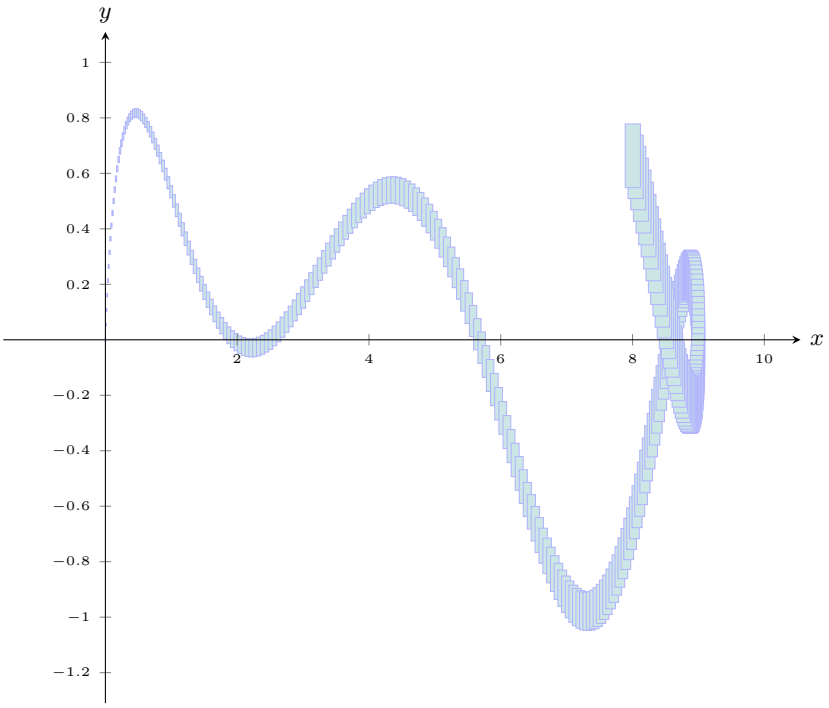



FIGURE 2.11 – Trajectoire d'un robot mobile.

**Exemple 2.5.2.** *Dans le domaine de l'automatique, les tubes nous permettent de caractériser des réponses paramétrées. Par exemple, un système du second ordre nous donne une réponse unitaire :*

$$y(t) = k - \frac{k}{\sqrt{1-m^2}}.\cos(\omega.\sqrt{1-m^2}.t).e^{-t.m.\omega}, \tag{2.5.3}$$

*avec $k$ le gain statique du système, $\omega$ la pulsation de résonance et $m$ le coefficient d'amortissement. Si le coefficient d'amortissement $m$ est incertain mais borné, on peut dire que $m \in [m]$. La représentation sous forme de tube nous permet d'afficher sur la figure 2.12 un tube $[y](t)$ qui contient toutes les réponses unitaires d'un système pour $k = 1$, $\omega = 0.3\ rad.s^{-1}$ et $m \in [0.13, 0.27]$.*

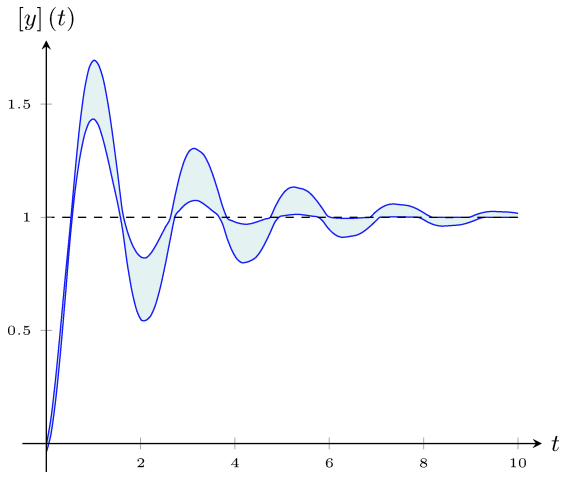



FIGURE 2.12 – Réponse d'un système du deuxième ordre.

### 2.5.2 Arithmétique des tubes

On peut étendre aux tubes les opérations classiques appliquées aux fonctions de $\mathcal{F}^n$, comme la somme, la multiplication, l'image par une fonction. Il suffit pour cela d'utiliser les notions d'arithmétique par intervalle et de fonctions d'inclusion présentées en section 2.2.1.1 pour chaque $t$. L'arithmétique des tubes est donc une extension directe de l'arithmétique des intervalles. Comme dans le cas des intervalles réels ou de $\mathbb{R}^n$, le résultat d'une opération sur les tubes contient les résultats de chacune des opérations effectuées sur l'ensemble des fonctions $\mathcal{F}^n$ qui le compose.

Il est aussi possible d'appliquer certaines techniques pour la satisfaction d'équations d'état aux tubes ( [Raissi et al. 2004 ; A. Goldsztejn, Hayes et al. 2011]).

### 2.5.3 Intégrale d'un tube

Comme nous le faisons pour les fonctions de $\mathcal{F}^n$, il est possible d'appliquer l'opérateur d'intégration aux tubes :

**Définition 2.5.1** (Intégrale d'un tube)**.** *Soient $t_1, t_2 \in \mathbb{R}$ tels que $t_1 \leq t_2$. L'intégrale d'un tube $[\mathbf{f}]$ sur l'intervalle $[t_1, t_2]$ est définie par :*

$$\int_{t_1}^{t_2} [\mathbf{f}](\tau)\, d\tau = \left\{ \int_{t_1}^{t_2} \mathbf{f}(\tau)\, d\tau \mid \mathbf{f} \in [\mathbf{f}] \right\}. \tag{2.5.4}$$

*Comme $t_2 \geq t_1$ et que l'opérateur intégrale est monotone, on peut dire que :*

$$\int_{t_1}^{t_2} [\mathbf{f}](\tau)\, d\tau = \left[\int_{t_1}^{t_2} \mathbf{f}^-(\tau)\, d\tau, \int_{t_1}^{t_2} \mathbf{f}^+(\tau)\, d\tau\right], \tag{2.5.5}$$

et on a, d'après la définition d'un tube :

$$\mathbf{f} \in [\mathbf{f}] \Rightarrow \int_{t_1}^{t_2} \mathbf{f}(\tau)\, d\tau \in \int_{t_1}^{t_2} [\mathbf{f}](\tau)\, d\tau. \tag{2.5.6}$$

Cette définition nous permet aussi de définir la primitive intervalle d'un tube :

**Définition 2.5.2** (Primitive d'un tube)**.** *Le tube*

$$[\mathbf{g}](t) = \int_0^t [\mathbf{f}](\tau)\, d\tau \tag{2.5.7}$$

*est la primitive du tube $[\mathbf{f}](t)$. est le tube primitive du tube $[\mathbf{f}](t)$. La primitive d'un tube est nulle pour $t = 0$.*

### 2.5.4 Intégrale d'un tube avec bornes intervalles

Supposons maintenant que les bornes $t_1, t_2$ de l'intervalle $[t_1, t_2]$ soient incertaines, ou plus précisément, qu'elles appartiennent à des intervalles, $t_1 \in [t_1]$ et $t_2 \in [t_2]$. Le théorème proposé dans [Aubry et al. 2013] nous permet de calculer le plus petit intervalle contenant l'intégrale $\int_{t_1}^{t_2} \mathbf{f}(\tau)\, d\tau$.

**Théorème 2.5.1** (Intégrale d'un tube avec bornes intervalles)**.** *Soit $\mathbf{f}$ une fonction de $\mathbb{R} \to \mathbb{R}^n$ et $t_1, t_2 \in \mathbb{R}$. Soient $t_2 \geq t_1 \geq 0$, $t_1 \in [t_1], t_2 \in [t_2], \mathbf{f} \in [\mathbf{f}]$. Alors la plus petite boîte qui contient toutes les valeurs possibles pour $\int_{t_1}^{t_2} \mathbf{f}(\tau)\, d\tau$ est définie par :*

$$\int_{[t_1]}^{[t_2]} [\mathbf{f}](\tau)\, d\tau = \left[lb\left(\mathbf{g}^-([t_2]) - \mathbf{g}^-([t_1])\right), ub\left(\mathbf{g}^+([t_2]) - \mathbf{g}^+([t_1])\right)\right], \tag{2.5.8}$$

*où $[\mathbf{g}](t) = \int_0^t [\mathbf{f}](\tau)\, d\tau$ est la primitive intervalle du tube $[\mathbf{f}]$.*

*Démonstration.* Prenons la i$^{\text{ème}}$ composante de l'intégrale et essayons d'en calculer la borne inférieure $z_i^-$. Cette borne est donnée par :

$$z_i^- = \min\left\{\int_{t_1}^{t_2} f_i(\tau)\, d\tau \mid t_1 \in [t_1], t_2 \in [t_2], f_i \in [f_i]\right\}.$$

Comme l'opérateur intégrale est monotone, la contribution optimale de la fonction $f_i$ est obtenue pour $f_i = \text{lb}([f_i]) = f_i^-$, on a

$$z_i^- = \min\left\{\int_{t_1}^{t_2} f_i^-(\tau)\, d\tau \mid t_1 \in [t_1], t_2 \in [t_2]\right\}.$$

Maintenant, la i$^{\text{ème}}$ borne inférieure $g_i^-(t)$ de $[\mathbf{g}](t)$ correspond à $\int_0^t f_i^-(\tau)\, d\tau$, la primitive de $f_i^-$ qui est nulle en $t = 0$. On a alors

$$z_i^- = \min\left\{g_i^-(t_2) - g_i^-(t_1) \mid t_1 \in [t_1], t_2 \in [t_2]\right\}.$$

Comme $t_1$ et $t_2$ n'apparaissent qu'une fois dans l'expression $g_i^-(t_2) - g_i^-(t_1)$, on peut écrire :

$$z_i^- = \min\left\{g_i^-([t_2]) - g_i^-([t_1])\right\} = \text{lb}\left(g_i^-([t_2]) - g_i^-([t_1])\right).$$

En utilisant un raisonnement identique pour le calcul de la borne supérieure $z_i^+$ de la i$^{\text{ème}}$ composante de l'intégrale , on a $z_i^+ = \text{ub}\Big(g_i^+([t_2]) - g_i^+([t_1])\Big)$, ce qui termine la preuve. □

**Exemple 2.5.3** (Illustration de l'intégrale d'un tube avec bornes intervalles)**.** *Soit* $[f](t)$ *un tube défini par ses bornes :* $[f](t) = [t-2; 2t+3]$ *avec* $t \geq 0$. *Soient* $[t_1] = [1;4]$ *et* $[t_2] = [5;6]$. *Calculons*

$$I = \int_{[t_1]}^{[t_2]} [f](\tau)\, d\tau = \int_{[1,4]}^{[5,6]} [\tau - 2, 2\tau + 3]\, d\tau.$$

*D'après le théorème 2.5.1, on a*

$$I = \left[lb\left(g^-([t_2]) - g^-([t_1])\right), ub\left(g^+([t_2]) - g^+([t_1])\right)\right].$$

*Où* $[g](t)$*, la primitive du tube* $[f](t)$ *est donnée par*

$$\begin{aligned}
[g](t) &= \int_0^t [f](\tau)\, d\tau = \int_0^t [\tau - 2, 2\tau + 3]\, d\tau, \\
&= \left[\int_0^t (\tau - 2)\, d\tau, \int_0^t (2\tau + 3)\, d\tau\right], \\
&= \left[\tfrac{1}{2}t^2 - 2t, t^2 + 3t\right] = \left[g^-(t), g^+(t)\right].
\end{aligned}$$

*Comme chacune des bornes* $g^{-}(t), g^{+}(t)$ *sont croissantes, on a*

$$\begin{aligned}
g^{-}([t_1]) &= \left[\tfrac{1}{2}\left(t_1^{-}\right)^2 - 2t_1^{-}, \tfrac{1}{2}\left(t_1^{+}\right)^2 - 2t_1^{+}\right] = \left[-\tfrac{3}{2}, 0\right], \\
g^{+}([t_1]) &= \left[\left(t_1^{-}\right)^2 + 3t_1^{-}, \left(t_1^{+}\right)^2 + 3t_1^{+}\right] = [4, 28], \\
g^{-}([t_2]) &= \left[\tfrac{1}{2}\left(t_2^{-}\right)^2 - 2t_2^{-}, \tfrac{1}{2}\left(t_2^{+}\right)^2 - 2t_2^{+}\right] = \left[\tfrac{5}{2}, 6\right], \\
g^{+}([t_2]) &= \left[\left(t_2^{-}\right)^2 + 3t_2^{-}, \left(t_2^{+}\right)^2 + 3t_2^{+}\right] = [40, 54].
\end{aligned}$$

*Alors*

$$\begin{aligned}
g^{-}([t_2]) - g^{-}([t_1]) &= \left[\tfrac{5}{2}, 6\right] - \left[-\tfrac{3}{2}, 0\right] = \left[\tfrac{5}{2}, \tfrac{15}{2}\right], \\
g^{+}([t_2]) - g^{+}([t_1]) &= [40, 54] - [4, 28] = [12, 50].
\end{aligned}$$

*Finalement*

$$\begin{aligned}
I &= \left[lb\left(g^{-}([t_2]) - g^{-}([t_1])\right), ub\left(g^{+}([t_2]) - g^{+}([t_1])\right)\right], \\
&= \left[lb\left(\left[\tfrac{5}{2}, \tfrac{15}{2}\right]\right), ub\left([12, 50]\right)\right] = \left[\tfrac{5}{2}, 50\right].
\end{aligned}$$

*La figure 2.13 nous montre que pour* $t_1, t_2, f$ *inconnus, on a*

$$\left\{\begin{array}{l} t_1 \in [t_1] \\ t_2 \in [t_2] \\ f \in [f] \end{array}\right. \quad \Rightarrow \quad \int_{t_1}^{t_2} f(\tau)\, d\tau \in \int_{[t_1]}^{[t_2]} [f](\tau)\, d\tau.$$

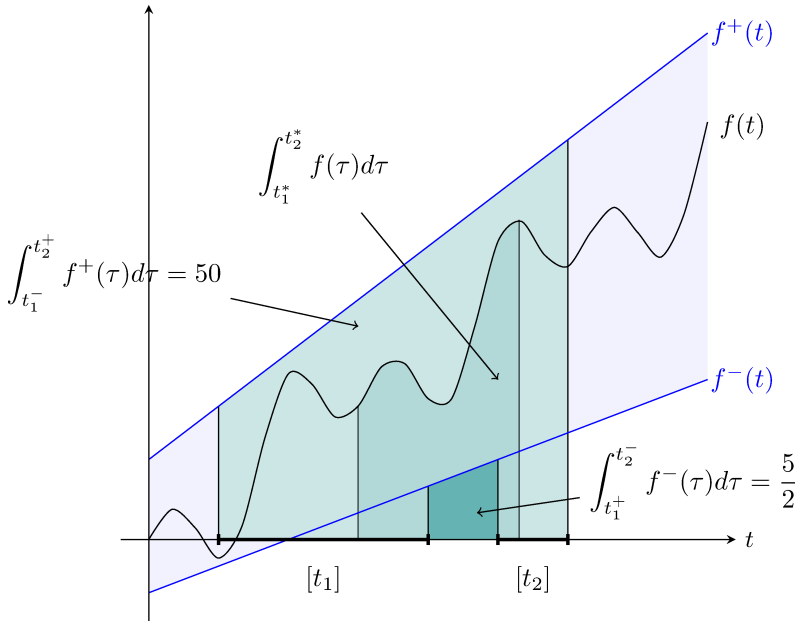



FIGURE 2.13 – Illustration de l'intégrale d'un tube avec bornes intervalles. Ici, on à bien $t_1^* \in [t_1]$, $t_2^* \in [t_2]$ et $f \in [f]$ donc $\int_{t_1^*}^{t_2^*} f(\tau)\, d\tau \in \int_{[t_1]}^{[t_2]} [f](\tau)\, d\tau$.

## 2.6 Noyau d'une fonction intervalle

La notion de noyau d'une fonction est un formalisme relativement peu utilisé dans les mathématiques. Nous verrons dans cette section comment un nouvel algorithme de caractérisation du noyau permet de résoudre un problème de localisation avec amers et mesures incertaines qu'il n'est pas possible de traiter avec les méthodes existantes. Pour notre application, nous verrons dans le chapitre 3 que la recherche de boucles dans la trajectoire d'un robot mobile revient à caractériser le noyau d'une fonction.

### 2.6.1 Noyau d'une fonction

**Définition 2.6.1** (Noyau [Dummit et Foote 2004])**.** *Pour une fonction* $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$*, le* noyau *est défini par :*

$$\ker \mathbf{f} = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{f}(\mathbf{x}) = \mathbf{0}\} = \mathbf{f}^{-1}(0). \tag{2.6.1}$$

En mathématiques, le noyau d'une fonction est un sous ensemble du domaine initial qui permet de mesurer le degré d'injectivité de la fonction. Par exemple,

si le noyau d'une fonction $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ ne possède qu'un seul élément $\mathbf{x_0}$, on aura $\mathbf{f}(\mathbf{x_0}) = \mathbf{0}$ et aucun $\mathbf{x} \neq \mathbf{x_0}$ ne satisfait $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. Dans ce cas, la fonction est injective. Maintenant, si le noyau est composé de $n$ éléments $\mathbf{x}_n$ avec $n \leq 2$, la fonction n'est pas injective. Pour un système d'équations carré, il s'agit en fait de la recherche des solutions, nous verrons par la suite pourquoi l'utilisation du noyau est nécessaire lorsque l'on ne possède pas un tel système et que l'on travaille avec des fonctions incertaines.

**Exemple 2.6.1.** *Soit une fonction* $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^2$ *définie par :*

$$\mathbf{f}(x_1, x_2) = \begin{cases} 2x_1 + 3x_2 - 2 \\ 3x_1 - 2x_2 \end{cases} . \tag{2.6.2}$$

*On peut calculer le noyau de cette fonction* $\mathbf{f}$*, il suffit pour cela de résoudre* $\mathbf{f}(\mathbf{x}) = 0$ *où* $\mathbf{x} = (x_1, x_2)$*. Ce système peut être mis sous forme matricielle :*

$$\begin{bmatrix} 2 & 3 \\ -4 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} . \tag{2.6.3}$$

*En utilisant une méthode de résolution linéaire, on arrive à :*

$$x_1 = \frac{4}{13}, \tag{2.6.4}$$

$$x_2 = \frac{6}{13}, \tag{2.6.5}$$

*qui est le noyau de* $\mathbf{f}$*.*

Quand la fonction $\mathbf{f}$ dépend de paramètres inconnus mais toutefois bornés, le noyau de la fonction $\mathbf{f}$ est appelé *united solution set - ensemble de solutions unitaires* - [Shary 2002] et sa caractérisation a bien été étudiée dans la littérature [Braems, Berthier et al. 2001 ; Braems, Jaulin et al. 2001 ; A. Goldsztejn 2003 ; A. Goldsztejn et Jaulin 2006]. Tous ces auteurs utilisent pour cela l'analyse par intervalles. L'ensemble de solutions unitaires d'une fonction $\mathbf{f} : [\mathbf{x}] \times [\mathbf{v}] \to \mathbb{R}^n$ est défini par :

$$\begin{aligned} \mathbb{X} &= \{\mathbf{x} \in [\mathbf{x}] \subset \mathbb{R}^n \mid \exists \mathbf{v} \in [\mathbf{v}], \ \mathbf{f}(\mathbf{x}, \mathbf{v}) = \mathbf{0}\} \\ &= \{\mathbf{x} \in [\mathbf{x}] \subset \mathbb{R}^n \mid \mathbf{0} \in \mathbf{f}(\mathbf{x}, [\mathbf{v}])\}, \end{aligned} \tag{2.6.6}$$

où $[\mathbf{x}]$ est une boîte de $\mathbb{R}^n$ et $[\mathbf{v}] \subset \mathbb{V}$ une boîte de $\mathbb{R}^m$.

La fonction $\mathbf{f} : [\mathbf{x}] \times [\mathbf{v}] \to \mathbb{R}^n$ est supposée continue par rapport à $\mathbf{x}, \mathbf{v}$ et différentiable par rapport à $\mathbf{x}$. Si l'on souhaite obtenir une approximation précise de l'ensemble $\mathbb{X}$, il faut bien souvent découper l'ensemble $\mathbb{V}$, ce qui limite la résolution à des ensembles $\mathbb{V}$ de faible dimension. En pratique, $\mathbf{v}$ est le vecteur des paramètres incertains et il existe des applications pour lesquelles cet ensemble est de dimension infinie, comme nous le verrons au chapitre 3. Dans ce cas, il faut trouver une méthode de recherche du noyau qui ne nécessite pas le découpage de cet ensemble. Cette section propose une telle méthode.

**Remarque 2.6.1.** *Lorsqu'une fonction* $\mathbf{f}$ *dépend de paramètres connus mais bornés, il est possible de la mettre sous la forme d'un intervalle de fonctions* $\mathbf{f}(\mathbf{x}, [\mathbf{v}])$.

### 2.6.2 Caractérisation du noyau d'une fonction intervalle

Si l'on prend en compte une fonction $\mathbf{f}$ incertaine qui appartient à un intervalle de fonctions $[\mathbf{f}] = \left[\mathbf{f}^-; \mathbf{f}^+\right]$, la valeur réelle de cette fonction sera notée $\mathbf{f}^*$ et nous savons que $\mathbf{f}^* \in [\mathbf{f}]$, ce qui correspond à une hypothèse d'erreurs bornées [Walter et Pronzato 1997]. On peut définir le noyau d'une fonction intervalle par :

$$\mathbb{X} = \ker [\mathbf{f}] = \bigcup_{\mathbf{f} \in [\mathbf{f}]} \ker \mathbf{f} = \{\mathbf{x} \in [\mathbf{x}] \subset \mathbb{R}^n \mid \mathbf{0} \in [\mathbf{f}](x)\}. \tag{2.6.7}$$

Nous cherchons ici à caractériser cet ensemble en trouvant deux ensembles qui l'encadrent, un ensemble intérieur $\mathbb{X}^-$ et un ensemble extérieur $\mathbb{X}^+$ tels que

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+. \tag{2.6.8}$$

Ainsi, nous aurons :

$$\ker \mathbf{f}^* \subset \mathbb{X}^+. \tag{2.6.9}$$

Nous proposons ici une méthode pour la caractérisation du noyau d'une fonction intervalle comme défini par l'équation 2.6.7. La démonstration théorique sera suivie de deux exemples qui illustreront la méthode.

Soient $\mathbf{f}^-(\mathbf{x})$ et $\mathbf{f}^+(\mathbf{x})$ les bornes inférieure et supérieure d'une fonction intervalle $[\mathbf{f}](\mathbf{x})$, ces deux fonctions de $\mathbb{R}^n$ à valeur dans $\mathbb{R}^n$ satisfont

$$\forall \mathbf{x} \in \mathbb{R}^n, \ [\mathbf{f}](\mathbf{x}) = \left[\mathbf{f}^-(\mathbf{x}), \mathbf{f}^+(\mathbf{x})\right]. \tag{2.6.10}$$

Sous l'hypothèse d'avoir deux fonctions d'inclusion convergentes pour $\mathbf{f}^-(\mathbf{x})$

et $\mathbf{f}^{+}(\mathbf{x})$, on peut définir les deux fonctions suivantes :

$$[\mathbf{f}^{\subset}]([\mathbf{x}]) = \left[\mathrm{ub}\left(\mathbf{f}^{-}([\mathbf{x}])\right), \mathrm{lb}\left(\mathbf{f}^{+}([\mathbf{x}])\right)\right], \tag{2.6.11}$$

$$[\mathbf{f}^{\supset}]([\mathbf{x}]) = \left[\mathrm{lb}\left(\mathbf{f}^{-}([\mathbf{x}])\right), \mathrm{ub}\left(\mathbf{f}^{+}([\mathbf{x}])\right)\right]. \tag{2.6.12}$$

**Exemple 2.6.2** (Illustration par un tube de $\mathbb{IR}$ dans $\mathbb{IR}$)**.** *Soit $f^*(t)$ une fonction de $\mathbb{R}$ dans $\mathbb{R}$ et $[f](t)$ un tube pour la fonction $f^*$, nous savons que $f^*(t) \in [f](t)$. Nous connaissons aussi les fonctions d'inclusion $[f^-](t)$ et $[f^+](t)$ pour chacune des bornes du tube $[f](t) = [f^-(t); f^+(t)]$. Cette configuration est représentée sur la figure 2.14 où on peut observer l'évaluation intervalle des fonctions $[f^-](t)$ et $[f^+](t)$ sur les intervalles $[t_a]$ et $[t_b]$.*

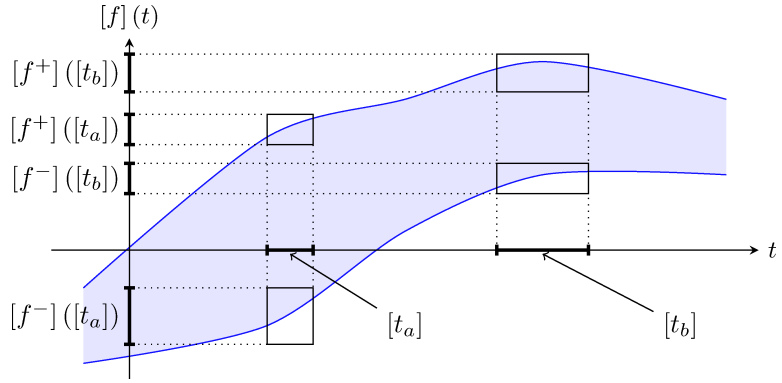



FIGURE 2.14 – Évaluation intervalle de $f^-(t_a)$,$f^+(t_a)$,$f^-(t_b)$ et$f^+(t_b)$.

*Il est possible, à partir de ces évaluations intervalles, de construire les fonctions $f^{\subset}(t)$ et $f^{\supset}(t)$. Cette construction est illustrée sur la figure 2.15.*

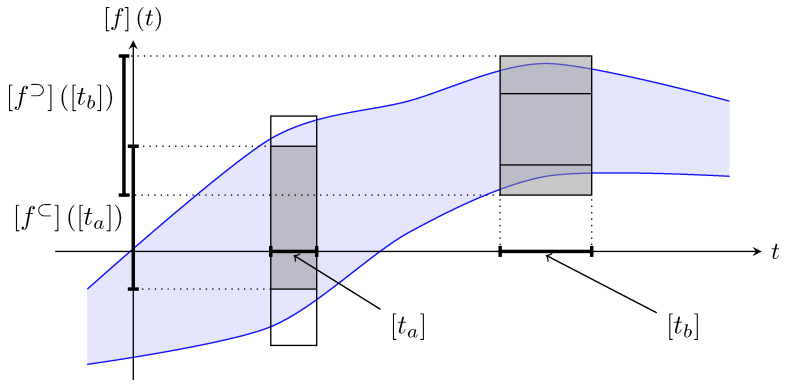



FIGURE 2.15 – Construction des fonctions $f^{\subset}(t)$ et $f^{\supset}(t)$.

**Exemple 2.6.3** (Illustration par un intervalle de fonctions de $\mathbb{IR}^2$ dans $\mathbb{IR}^2$)**.** *Pour cet exemple, nous considérons une fonction incertaine* $\mathbf{f}^*(x)$ *une fonction de* $\mathbb{R}^2$ *dans* $\mathbb{R}^2$ *et* $[\mathbf{f}](x)$ *un intervalle de fonctions pour la fonction* $\mathbf{f}^*$*. On peut voir sur les figures 2.16 et 2.17 l'évaluation intervalle des fonctions* $[\mathbf{f}^-](x)$, $[\mathbf{f}](x)$ *et* $[\mathbf{f}^+](x)$.

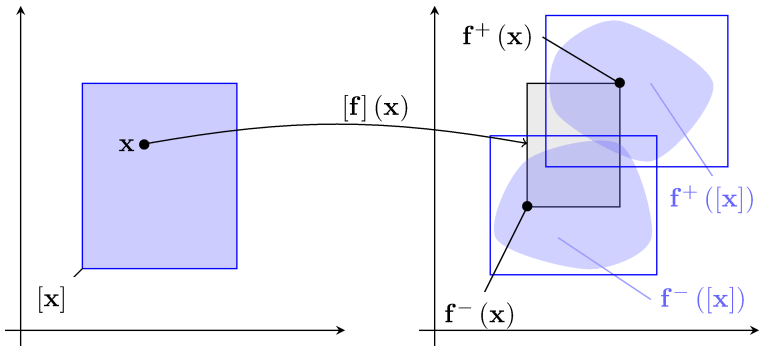



FIGURE 2.16 – Fonctions d'inclusion associées aux fonctions $\mathbf{f}^+(\mathbf{x})$ et $\mathbf{f}^+(\mathbf{x})$

*Le passage de la figure 2.16 à la figure 2.17 montre l'aspect convergent des fonctions* $\mathbf{f}^-(\mathbf{x})$ *et* $\mathbf{f}^+(\mathbf{x})$*. Lorsque la taille de l'intervalle* $[\mathbf{x}]$ *à évaluer diminue, la taille des évaluations intervalles associées* $\mathbf{f}^-([\mathbf{x}])$ *et* $\mathbf{f}^+([\mathbf{x}])$ *diminue aussi. Par le même raisonnement, les fonctions d'inclusion associées aux fonctions* $\mathbf{f}^-$ *et* $\mathbf{f}^+$ *respectent elles aussi cette notion de convergence.*

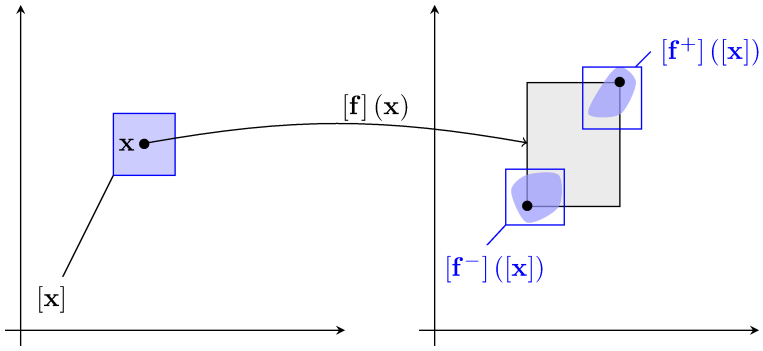



FIGURE 2.17 – Fonctions d'inclusion associées aux fonctions $\mathbf{f}^+(\mathbf{x})$ et $\mathbf{f}^+(\mathbf{x})$

*La construction des fonctions* $[\mathbf{f}^{\subset}]([\mathbf{x}])$ *et* $[\mathbf{f}^{\supset}]([\mathbf{x}])$ *est possible lorsque les fonctions d'inclusion* $[\mathbf{f}^-](x)$ *et* $[\mathbf{f}^+](x)$ *sont assez fines pour permettre* $[\mathbf{f}^-]([\mathbf{x}]) \cap [\mathbf{f}^+]([\mathbf{x}]) = \emptyset$*, voir figure 2.18. Dans le cas contraire,* $[\mathbf{f}^-]([\mathbf{x}]) \cap [\mathbf{f}^+]([\mathbf{x}]) \neq \emptyset$ *(comme sur la figure 2.16) la fonction* $[\mathbf{f}^{\subset}](x)$ *peut se retrouver vide.*

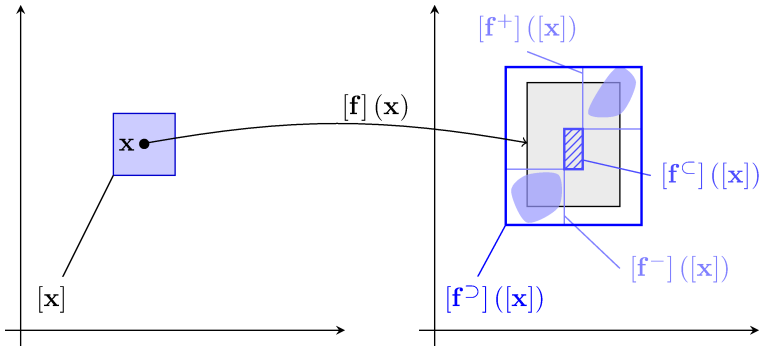



FIGURE 2.18 – Illustration de l'inclusion $[\mathbf{f}^{\subset}]([\mathbf{x}]) \subset [\mathbf{f}](\mathbf{x}) \subset [\mathbf{f}^{\supset}]([\mathbf{x}])$

**Proposition 2.6.1.** *D'après la définition des fonctions* $[\mathbf{f}^{\subset}]$ *et* $[\mathbf{f}^{\supset}]$*, on a :*

$$\mathbf{x} \in [\mathbf{x}] \Rightarrow [\mathbf{f}^{\subset}]([\mathbf{x}]) \subset [\mathbf{f}](\mathbf{x}) \subset [\mathbf{f}^{\supset}]([\mathbf{x}])\,. \tag{2.6.13}$$

*Démonstration.* Comme

$$\mathbf{x} \in [\mathbf{x}] \Rightarrow \left\{ \begin{array}{l} \mathbf{f}^{-}(\mathbf{x}) \geq \mathrm{lb}\left(\mathbf{f}^{-}([\mathbf{x}])\right) \\ \mathbf{f}^{+}(\mathbf{x}) \leq \mathrm{ub}\left(\mathbf{f}^{+}([\mathbf{x}])\right) \end{array} \right. , \tag{2.6.14}$$

on a

$$\mathbf{x} \in [\mathbf{x}] \Rightarrow [\mathbf{f}](\mathbf{x}) \subset [\mathbf{f}^{\supset}]([\mathbf{x}])\,.$$

Et comme

$$\mathbf{x} \in [\mathbf{x}] \Rightarrow \left\{ \begin{array}{l} \mathbf{f}^{-}(\mathbf{x}) \leq \mathrm{ub}\left(\mathbf{f}^{-}([\mathbf{x}])\right) \\ \mathbf{f}^{+}(\mathbf{x}) \geq \mathrm{lb}\left(\mathbf{f}^{+}([\mathbf{x}])\right) \end{array} \right. , \tag{2.6.15}$$

on a

$$\mathbf{x} \in [\mathbf{x}] \Rightarrow [\mathbf{f}^{\subset}]([\mathbf{x}]) \subset [\mathbf{f}](\mathbf{x})\,.$$

□

**Proposition 2.6.2.** *Nous proposons les tests d'inclusion suivants :*

$$\left( \begin{array}{lll} \textit{(i)} & \mathbf{0} \in [\mathbf{f}^{\subset}]([\mathbf{x}]) & \Rightarrow \quad [\mathbf{x}] \subset \mathbb{X} \\ \textit{(ii)} & \mathbf{0} \notin [\mathbf{f}^{\supset}]([\mathbf{x}]) & \Rightarrow \quad [\mathbf{x}] \cap \mathbb{X} = \emptyset \end{array} \right), \tag{2.6.16}$$

*pour la caractérisation du noyau d'une fonction incertaine.*

*Démonstration.* Commençons par démontrer *(i)*. On a

$$\begin{aligned} \mathbf{x} \in \mathbb{X} &\overset{(2.6.7)}{\Rightarrow} \mathbf{x} \in \ker[\mathbf{f}] \Leftrightarrow \exists \mathbf{f} \in [\mathbf{f}],\ \mathbf{f}(\mathbf{x}) = \mathbf{0} \\ &\Leftrightarrow \mathbf{0} \in [\mathbf{f}](\mathbf{x}) \Leftrightarrow \mathbf{0} \in \left[\mathbf{f}^{-}(\mathbf{x}), \mathbf{f}^{+}(\mathbf{x})\right]. \end{aligned} \tag{2.6.17}$$

Maintenant

$$\begin{aligned} \mathbf{0} \in \left[\mathbf{f}^{\subset}\right]([\mathbf{x}]) &\overset{(2.6.13)}{\Rightarrow} \forall \mathbf{x} \in [\mathbf{x}], \mathbf{0} \in [\mathbf{f}](\mathbf{x}) \\ &\Rightarrow \forall \mathbf{x} \in [\mathbf{x}], \mathbf{0} \in \left[\mathbf{f}^{-}(\mathbf{x}), \mathbf{f}^{+}(\mathbf{x})\right] \\ &\Rightarrow \forall \mathbf{x} \in [\mathbf{x}], \mathbf{x} \in \mathbb{X} \Rightarrow [\mathbf{x}] \subset \mathbb{X}. \end{aligned} \tag{2.6.18}$$

Nous allons maintenant démontrer (ii).

$$\begin{aligned} \mathbf{0} \notin \left[\mathbf{f}^{\supset}\right]([\mathbf{x}]) &\overset{(2.6.13)}{\Rightarrow} \forall \mathbf{x} \in [\mathbf{x}], \mathbf{0} \notin [\mathbf{f}](\mathbf{x}) \\ &\Rightarrow \forall \mathbf{x} \in [\mathbf{x}], \mathbf{x} \notin \mathbb{X} \Rightarrow [\mathbf{x}] \cap \mathbb{X} = \emptyset. \end{aligned} \tag{2.6.19}$$

□

**Remarque 2.6.2.** *Les deux tests de la proposition 2.6.2 sont des tests d'inclusion et ils nous permettent de trouver deux sous pavages*[1]*, un intérieur et l'autre extérieur, pour l'ensemble* $\mathbb{X}$ *avec un paveur quelconque.*

### 2.6.3 Algorithme

L'algorithme 1 ([Aubry et al. 2014]) propose la caractérisation du noyau d'un intervalle de fonctions. Cet algorithme de type *branch-and-bound* est similaire à l'algorithme SIVIA[2]. L'idée principale de l'algorithme est de bissecter un domaine initial de recherche en un pavage régulier. Cet algorithme utilise pour cela les deux tests de la proposition 2.6.1. En entrée, l'algorithme nécessite un intervalle de fonctions $[\mathbf{f}]$ décrit par ses bornes $\mathbf{f}^{-}$ et $\mathbf{f}^{+}$, un domaine initial de recherche $[\mathbf{x_0}]$ et une précision $\varepsilon$. L'algorithme retourne trois sous pavages : $\mathbb{X}^{\text{in}}$ qui contient les boîtes $[\mathbf{x}]$ qui sont incluses dans $\mathbb{X}$ ; $\mathbb{X}^{\text{out}}$ qui contient les boîtes $[\mathbf{x}]$ en dehors de l'ensemble $\mathbb{X}$ et $\mathbb{X}^{?}$qui contient des boîtes $[\mathbf{x}]$ de petite taille pour lesquelles

1. Un pavage correspond à la description d'une boîte par plusieurs boîtes de taille plus petite qui ne se chevauchent pas. Un sous pavage est un ensemble de boites d'un pavage.
2. Set Inversion Via Interval Analysis - voir [Jaulin, Kieffer, Didrit et al. 2001]

rien n'est connu. Une boîte est considérée comme petite lorsque $w([\mathbf{x}]) < \varepsilon$ .

**Algorithme 1 :** KER

**Entrées** : $\varepsilon, [\mathbf{x_0}], [\mathbf{f}](x)$

```
// Initialisation
Q ← {[x0]} ; X^in = ∅ ; X^out = ∅ ; X^? = ∅;
// Boucle principale
tant que Q n'est pas vide faire
    [x] ← Q.prendre_premier_élément() ; // L'élément est retiré de
        la liste et rangé dans [x]
    si 0 ∉ [f^⊃]([x]) alors
        X^out ← X^out ∪ [x];
    sinon si 0 ∈ [f^⊂]([x]) alors
        X^in ← X^in ∪ [x];
    sinon si width([x]) < ε alors
        X^? ← X^? ∪ [x];
    sinon
        // La boîte est divisée et le résultat est mis dans la
            liste Q
        ([x1], [x2]) = Bissecter([x]);
        Q ← {Q; [x1] ; [x2]};
    fin
fin
```

**Sorties** : $\mathbb{X}^{\text{in}}, \mathbb{X}^{?}, \mathbb{X}^{\text{out}}$

Lors de l'initialisation, une liste $\mathcal{Q}$ est remplie avec la boîte représentant le domaine initial de recherche et les listes de sortie sont vidées. La boucle principale du programme s'effectue tant que la liste $\mathcal{Q}$ n'est pas vide. Un élément de la liste est retiré de cette dernière et les tests de la proposition 2.6.1 vont permettre de classer cette boîte en fonction de son appartenance à l'ensemble $\mathbb{X}$. Si la taille maximale parmi les tailles des intervalles qui composent la boîte est inférieure au paramètre $\varepsilon$, la boîte est rangée dans la liste $\mathbb{X}^{?}$. Si les tests échouent et que la boîte est encore assez grande, elle est bissectée en deux éléments qui sont à leur tour rangés dans la liste $\mathcal{Q}$. Quand tous les éléments de la liste $\mathcal{Q}$ ont étés triés,

on obtient un encadrement pour $\mathbb{X}$ :

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+.$$

où

$$\mathbb{X}^- = \mathbb{X}^{\text{in}},$$
$$\mathbb{X}^+ = \mathbb{X}^{\text{in}} \cup \mathbb{X}^?.$$

### 2.6.4 Application à la localisation garantie par mesure de distance

Cette section donne un exemple d'utilisation de l'algorithme de caractérisation du noyau sur un problème de localisation par mesure de distance et amer incertain. A notre connaissance, ce problème ne peut pas être résolu par les méthodes existantes, ce qui démontre bien l'intérêt de ce nouvel algorithme. Nous verrons dans le chapitre 3 comment le concept du noyau est utilisé pour résoudre le problème de détection de fermeture de boucle.

#### 2.6.4.1 Localisation par mesure de distance

Considérons $k$ amers $\mathbf{m}(i)$ positionnés en $(m_x(i), m_y(i))$ ainsi qu'un robot mobile localisé à une distance $d(i)$ de l'amer $i$. Lorsque la position des amers est connue, la position $\mathbf{p} = (p_x, p_y)$ du robot est située à l'intersection des cercles de centre $\mathbf{m}(i)$ et de rayon $d(i)$. Plus formellement, la position du robot sera consistante avec la distance $d(i)$ si :

$$(p_x - m_x(i))^2 + (p_y - m_y(i))^2 = d^2(i). \tag{2.6.20}$$

Si le robot se trouve dans un environnement difficile d'accès comme le milieu sous-marin, le positionnement des amers peut être incertain, de même que la mesure de distance. En utilisant l'analyse par intervalles, on peut dire que ces

quantités appartiennent à des intervalles :

$$\mathbf{m}(i) \in [\mathbf{m}](i), \tag{2.6.21}$$

$$\text{avec } [\mathbf{m}](i) = \begin{pmatrix} [m_x](i) \\ [m_y](i) \end{pmatrix}, \tag{2.6.22}$$

$$\text{et } d(i) \in [d](i). \tag{2.6.23}$$

Avec cette *intervalisation*, le problème devient :

$$\exists\, m_x \in [m_x](i)\,,\ \exists m_y\ \in [m_y](i)\,,\ \exists\, d \in [d](i) \tag{2.6.24}$$

$$\text{tels que } (p_x - m_x)^2 + (p_y - m_y)^2 = d^2, \tag{2.6.25}$$

ou, d'une manière équivalente

$$0 \in [f_i](\mathbf{p})\,, \tag{2.6.26}$$

avec

$$[f_i](\mathbf{p}) = (p_x - [m_x](i))^2 + (p_y - [m_y](i))^2 - [d(i)]^2\,. \tag{2.6.27}$$

L'utilisation de l'arithmétique symbolique des intervalles [Jaulin et Chabert 2010] nous permet de mettre cette fonction sous forme d'un tube :

$$[f_i](\mathbf{p}) = \left[f_i^-(\mathbf{p})\,, f_i^+(\mathbf{p})\right], \tag{2.6.28}$$

avec

$$\begin{aligned} f_i^-(\mathbf{p}) = &\max\left(0, \operatorname{sign}\Big((p_x - m_x^+(i))(p_x - m_x^-(i))\Big)\right) \\ &\cdot \min\left(\Big(p_x - m_x^+(i)\Big)^2, (p_x - m_x^-(i))^2\right) \\ &+ \max\left(0, \operatorname{sign}\Big((p_y - m_y^+(i))(p_y - m_y^-(i))\Big)\right) \\ &\cdot \min\left(\Big(p_y - m_y^+(i)\Big)^2, \Big(p_y - m_y^-(i)\Big)^2\right) - \Big(d^+(i)\Big)^2, \end{aligned} \tag{2.6.29}$$

et

$$\begin{aligned} f_i^+(\mathbf{p}) = & \max\left(\left(p_x - m_x^+(i)\right)^2, \left(p_x - m_x^-(i)\right)^2\right) \\ & + \max\left(\left(p_y - m_y^+(i)\right)^2, \left(p_y - m_y^-(i)\right)^2\right) \\ & - \left(d^-(i)\right)^2. \end{aligned} \tag{2.6.30}$$

L'ensemble des positions concordant avec les $k$ mesures de distance est donc décrit par le noyau de la fonction $[\mathbf{f}]$ :

$$\mathbb{P} = \{\mathbf{p} \mid \mathbf{0} \in [\mathbf{f}](\mathbf{p})\} = \ker[\mathbf{f}]. \tag{2.6.31}$$

Comme la fonction de distance, définie par l'équation 2.6.27, peut être mise sous la forme d'un tube et que le problème de localisation par mesure de distance peut être résolu en cherchant le noyau de cette fonction, nous pouvons utiliser l'algorithme 1(KER) pour caractériser l'ensemble $\mathbb{P}$ contenant toutes les positions consistantes avec les $k$ distances et positions des amers (toutes deux incertaines).

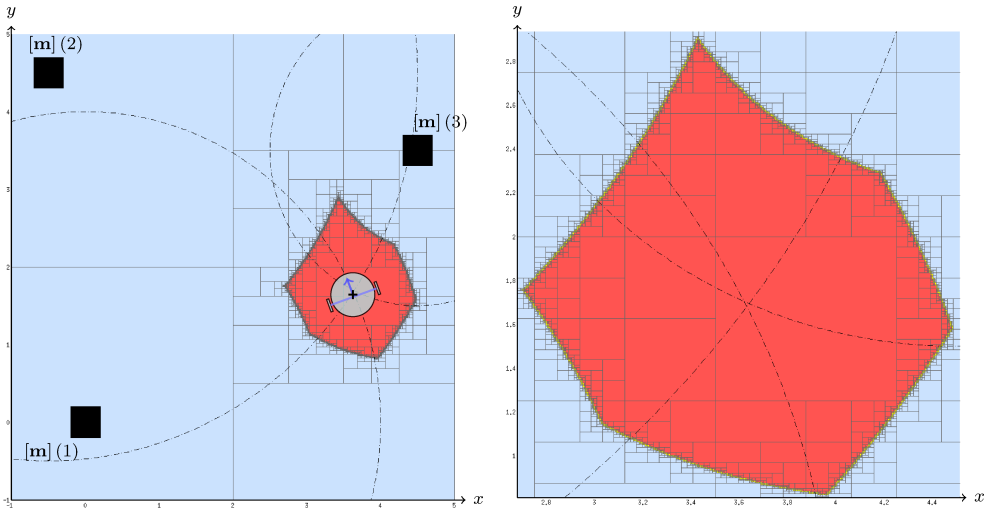
#### 2.6.4.2 Application

Nous considérons un robot se déplaçant sur un plan. Ce robot doit calculer sa position relative à trois amers. Le robot est équipé d'un capteur qui mesure le temps de parcours d'un signal acoustique émis par chaque amer afin de retourner une distance $d(i)$ du robot à l'amer $i$. Le positionnement des amers ainsi que la mesure de distance sont perturbés mais ces perturbations sont supposées bornées. On peut donc dire que $\mathbf{m}(i) \in [\mathbf{m}](i)$ et $d(i) \in [d](i)$. Les conditions de la simulation présentée par la suite sont les suivantes :

$$[\mathbf{m}](1) = \begin{pmatrix} [-0.2; 0.2] \\ [-0.2; 0.2] \end{pmatrix}, [\mathbf{m}](2) = \begin{pmatrix} [-0.7; -0.3] \\ [4.3; 4.7] \end{pmatrix}, [\mathbf{m}](3) = \begin{pmatrix} [4.3; 4.7] \\ [3.3; 3.7] \end{pmatrix},$$

$$[d](1) = [3.5; 4.5], [d](2) = [4.5; 5.5], [d](3) = [1.5; 2.5],$$

$$[\mathbf{x_0}] = \begin{pmatrix} [-1; 5] \\ [-1; 5] \end{pmatrix} \text{ et } \varepsilon = 0.01.$$




(a) (b)

FIGURE 2.19 – Position d'un robot mobile par rapport à trois amers incertains - Zoom sur cette position.

La figure 2.19 illustre le problème de localisation garantie. Les positions incertaines des amers sont représentées par des carrés noirs. Les cercles en pointillé sont présents à caractère informatif. Ils représentent le centre de la mesure $[d]\,(i)$ tracé à partir du centre de l'amer $[\mathbf{m}]\,(i)$. Le positionnement du robot est calculé par l'algorithme 1(KER). Quand l'algorithme est terminé, on obtient un encadrement pour la position du robot :

$$\mathbb{P}^{-} \subset \mathbb{P} \subset \mathbb{P}^{+}.$$

où

$$\mathbb{P}^{-} = \mathbb{X}^{\text{in}},$$
$$\mathbb{P}^{+} = \mathbb{X}^{\text{in}} \cup \mathbb{X}^{?},$$

et $\mathbb{X}^{\text{in}}$, $\mathbb{X}^{?}$ et $\mathbb{X}^{\text{out}}$ sont les sorties de l'algorithme.

Les boîtes rouges représentent l'ensemble $\mathbb{P}^{-}$ des boîtes qui satisfont le test $0 \in [\mathbf{f}^{\subset}]\,([\mathbf{x}])$. Les boîtes bleues qui composent la plus grande partie du domaine initial de recherche représentent les boîtes qui n'appartiennent pas à $\mathbb{P}$, ces boîtes satisfont la condition $0 \notin [\mathbf{f}^{\supset}]\,([\mathbf{x}])$. Enfin, les boîtes jaunes qui forment la frontière entre ces deux zones (voir figure (b)) représentent les boîtes qui n'ont passé aucun

des deux tests de la proposition 2.6.1.

Cette simulation à été effectuée sous Matlab R2010a avec la toolbox Intlab V6. Les temps de calcul observés sont de $8,7$ secondes pour $\varepsilon = 0.1$ et 2 minutes $20,7$ secondes pour $\varepsilon = 0.01$ et de 19 minutes $52,4$ secondes pour $\varepsilon = 0.001$.

## 2.7 Recherche d'existence et d'unicité de la solution à $f(x) = 0$ lorsque $f$ est incertaine

Un problème courant en analyse numérique est la recherche d'une racine d'une application :

$$\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \mathbf{x} \mapsto (f_1(\mathbf{x}), ..., f_n(\mathbf{x})). \tag{2.7.1}$$

C'est à dire rechercher $\mathbf{x}^*$ qui vérifie $\mathbf{f}(\mathbf{x}^*) = 0$.

Les méthodes existantes approchent la solution par itérations successives afin de donner une solution $\mathbf{x}^0$ très proche du zéro $\mathbf{x}^*$. La question se pose alors quant à l'existence de ce zéro autour d'un voisinage proche de la solution estimée $\mathbf{x}^0$. L'arithmétique des intervalles s'est dotée d'outils pour répondre à cette question. Parmi celles-ci, on peut citer les tests d'existence de Moore [R. E. Moore 1977] basés sur la méthode de Newton ou l'opérateur de Krawczyk [Neumaier 1990], ceux basés sur le test d'existence de Miranda [R. Moore et Kioustelidis 1980] ou encore sur le théorème de Borsuk [Borsuk 1933 ; Frommer et Lang 2005]. Nous verrons dans le chapitre 3 que notre application nécessite une preuve de l'existence d'une boucle. En effet, savoir qu'il y a probablement une boucle dans la trajectoire n'est pas suffisant pour la prise en compte des résultats. Aussi, un test d'existence sera utilisé afin de pouvoir répondre avec certitude à cette question. En outre, ce test nous permettra de compter le nombre de boucles présentes dans la trajectoire de façon garantie.

Nous allons présenter dans cette section la méthode de Newton pour la résolution de $f(x) = 0$ qui, dans sa version intervalisée, nous permettra d'introduire les notions d'existence et unicité nécessaires à notre application. Les autres méthodes citées ci-dessus ne seront pas présentées puisqu'elles n'ont pas amélioré les résultats que nous présentons au chapitre 3.

### 2.7.1 Résolution de $f(x) = 0$ par la méthode de Newton

On cherche ici à trouver une racine d'une fonction $f(x) : \mathbb{R} \mapsto \mathbb{R}$ continument dérivable par rapport à $x$. Pour cela, on se base sur son approximation de Taylor au premier ordre en partant d'un point connu $x_0$ et de préférence proche du zéro à déterminer. On a donc : $f(x) \simeq f(x_0) + f'(x_0)(x - x_0)$, ce qui revient à approximer la fonction en $x_0$ par sa tangente. En calculant l'intersection de cette approximation avec l'axe des abscisses, on trouvera un point $x_1$ qui aura une forte probabilité d'être plus proche de la racine que le point $x_0$. A force d'approximations successives, on obtiendra un point $x_n$ très proche d'une racine de la fonction.

Formellement, on construit la suite :

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \tag{2.7.2}$$

et on a bien $x_{k+1}$ le point d'intersection de la tangente en $x_k$ avec l'axe des abscisses.

**Exemple 2.7.1.** *La figure 2.20 illustre cette méthode pour* $f(x) = \exp(x) - 1$. *L'unique racine de cette fonction est* $x = 0$.

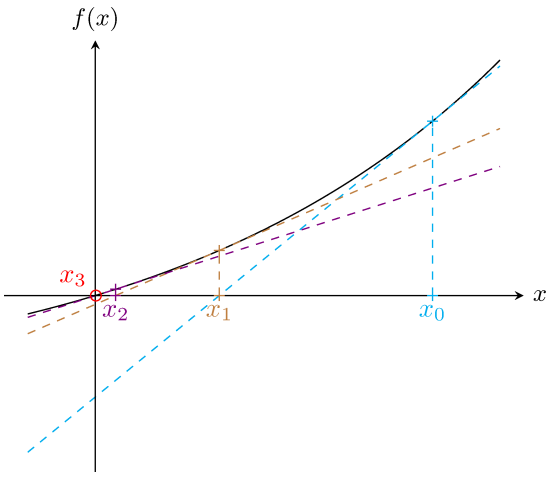



FIGURE 2.20 – Illustration de la méthode de Newton

*Si on construit la suite définie par 2.7.2 à partir du point $x_0 = 1$, on trouvera :*

$$x_1 = 0.367879441171442,$$
$$x_2 = 0.0600800687267887,$$
$$x_3 = 0.00176919944264467,$$
$$x_4 = 1.56411078989843e - 6.$$

*Nous pouvons voir sur cet exemple qu'au bout de quatre itérations, on obtient une estimation très proche de la solution. Si cette solution peut être satisfaisante dans certains cas, il existe des domaines qui demandent plus de garantie quant à l'existence de ce zéro.*

### 2.7.2 Méthode de Newton par intervalles

Moore a décrit, dans [R. E. Moore 1966], une méthode de Newton par intervalles. Pour expliquer cette méthode, nous partons d'une fonction réelle $f(x) : \mathbb{R} \mapsto \mathbb{R}$ continument dérivable par rapport à $x$. Si on possède une fonction d'inclusion $F'([x])$ pour la dérivée $f'(x)$ de $f(x)$, on peut construire la suite :

$$[x_{k+1}] = [x_k] \cap \mathcal{N}([x_k]), \tag{2.7.3}$$

avec (intervalisation de 2.7.2) :

$$\mathcal{N}([x_k]) = \hat{x_k} - \frac{f(\hat{x_k})}{F'([x_k])}, \tag{2.7.4}$$

où $\hat{x_k}$ est un point quelconque de l'intervalle $[x_k]$ (si $\hat{x_k}$ est souvent choisi comme le centre de l'intervalle à évaluer, ce n'est pas une règle et l'on peut prendre n'importe quel point à l'intérieur de l'intervalle $[x_k]$). Lorsque l'intervalle de départ $[x_0]$ contient une racine de $f(x)$ et si $0 \notin F'([x_0])$, alors les intervalles $[x_k]$ construits par la suite 2.7.3 la contiennent aussi et la suite converge vers cette racine.

**Exemple 2.7.2.** *Reprenons l'exemple 2.7.1. La fonction $f(x) = \exp(x) - 1$ est continument dérivable selon $x$. Nous pouvons choisir $F'([x]) = \exp([x])$ comme fonction d'inclusion pour $f'(x)$. Nous allons maintenant chercher une racine de la fonction $f$ dans l'intervalle $[x_0] = [-1; 1.5]$. Dans la méthode de Newton classique, où la suite va construire un nouvel itéré à partir de la tangente à $f(x)$ en ce point,*

*la méthode de Newton par intervalles va construire le nouvel itéré à partir de deux tangentes. Les pentes de ces tangentes correspondent aux bornes inférieure et supérieure de $F'([x])$. Par ces deux tangentes, on décrit donc l'évolution possible de cette tangente pour tout l'intervalle $[x_k]$. Ce phénomène est décrit par la figure 2.21 où l'on peut voir la première itération de la suite.*

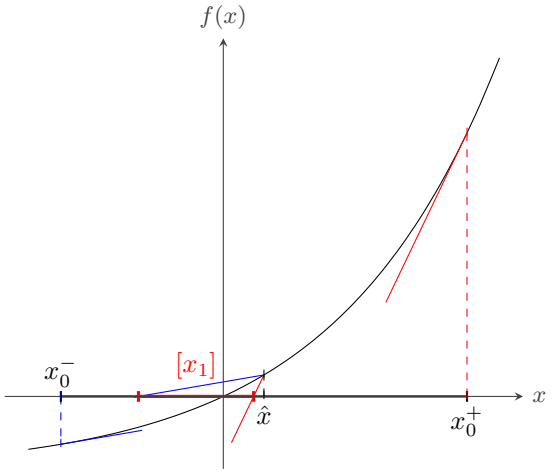



FIGURE 2.21 – Illustration de la méthode de Newton par intervalles. Le point $\hat{x}$ est choisi ici comme le centre de l'intervalle à évaluer. On pourrait prendre n'importe quel point $\hat{x} \in [x]$. Il est courant de voir ce choix utilisé dans la littérature.

*Comme la fonction $f(x)$ est croissante, les bornes inférieure et supérieure de $F'([x])$ sont représentées par les tangentes à $f(x)$ aux bornes inférieure et supérieure de l'intervalle $[x_0]$. Ensuite, ces tangentes sont centrées autour du point $\hat{x}$ et leur intersection avec l'axe des abscisses donne le résultat de l'itération. Cette figure ne présente que la première itération, si on calcule la suite pour quatre itérations, on obtient :*

$$\begin{aligned}
[x_1] &= [-0.5221, 0.1867]\,, \\
[x_2] &= [-0.0396, 0.0926]\,, \\
[x_3] &= [-0.0015, 0.0021]\,, \\
[x_4] &= [-0.4595, 0.5429]\,10^{-6}.
\end{aligned}$$

La méthode de Newton par intervalles a été développée dans le cas de fonctions $\mathbf{f}(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}^n$ dérivables selon $\mathbf{x}$. La dérivée est alors remplacée par la matrice Jacobienne $\mathbf{J_f}$ et on a besoin d'une fonction d'inclusion $[\mathbf{J_f}]$ pour cette matrice.

L'opérateur de Newton devient :

$$\mathcal{N}\left(\mathbf{f}, [\mathbf{J_f}], [\mathbf{x_k}]\right) = \hat{\mathbf{x_k}} - [\mathbf{J_f}]^{-1}([\mathbf{x}]).\mathbf{f}(\hat{\mathbf{x_k}}). \tag{2.7.5}$$

Nous avons vu qu'une fonction incertaine peut être représentée par un tube, c'est le cas par exemple lorsque des mesures sont bruitées. Si on possède une fonction d'inclusion $[\mathbf{f}]$ pour notre fonction incertaine $\mathbf{f}$, on a $\mathbf{f} \in [\mathbf{f}]$ et on peut étendre la définition de l'opérateur de Newton par intervalles aux fonctions incertaines représentées par des tubes :

$$\mathcal{N}\left([\mathbf{f}], [\mathbf{J_f}], [\mathbf{x_k}]\right) = \hat{\mathbf{x_k}} - [\mathbf{J_f}]^{-1}([\mathbf{x}]).[\mathbf{f}](\hat{\mathbf{x_k}}). \tag{2.7.6}$$

**Remarque 2.7.1.** *Dans le cadre de la méthode de Newton par intervalles appliquée à des fonctions à valeurs dans $\mathbb{R}^n$ incertaines, il est préférable de résoudre le système d'équation linéaire $[\mathbf{J_f}]([\mathbf{x}]).[\mathbf{s}] = [\mathbf{f}](\hat{\mathbf{x_k}})$ pour trouver $[\mathbf{s}] = [\mathbf{J_f}]^{-1}([\mathbf{x}]).[\mathbf{f}](\hat{\mathbf{x_k}})$ et de ne pas inverser la matrice. Le lecteur est renvoyé vers [Rump 1980; Alefeld et Herzberger 1983] pour plus de précisions.*

**Remarque 2.7.2.** *Le point faible de cette méthode se trouve effectivement au niveau du terme $[\mathbf{J_f}]^{-1}([\mathbf{x}])$. Le problème d'inversion matricielle invoqué plus haut ne fait pas état du cas ou un des éléments de cette matrice contient $0$. Dans ce cas, il est impossible d'inverser cette matrice et donc de donner un résultat pour l'opérateur de Newton. Il faut donc faire attention quand au choix de l'intervalle à tester. Plus ce dernier sera de petite taille, plus les chances de trouver un zéro dans la jacobienne seront faibles.*

### 2.7.3 Propriété d'existence et unicité

Une propriété intéressante de la méthode de Newton est la propriété d'existence et unicité (pour des exemples d'utilisation, voir [Neveu et al. 2010 ; Tucker 1999]). Par la méthode de Newton, on cherche à évaluer la racine d'une fonction $\mathbf{f}$ dans un intervalle $\mathbf{x_0}$ donné. Les travaux de Moore ([R. E. Moore 1979]) ont mis en évidence le fait que si un itéré de la méthode de Newton est inclus dans son intervalle initial, il existe une solution unique à $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ dans cet intervalle. Plus formellement, on a :

$$\mathcal{N}\left(\mathbf{f}, [\mathbf{J_f}], [\mathbf{x}]\right) \subset [\mathbf{x}] \Rightarrow \exists! \mathbf{x} \in [\mathbf{x}] \mid \mathbf{f}(\mathbf{x}) = \mathbf{0}, \tag{2.7.7}$$

où $\exists!$ veut dire *il existe un unique*. Si la fonction $\mathbf{f}$ est incertaine et représentée par un tube, on a $\mathbf{f} \in [\mathbf{f}]$ et donc

$$\mathcal{N}\left([\mathbf{f}], [\mathbf{J_f}], [\mathbf{x}]\right) \subset [\mathbf{x}] \Rightarrow \exists! \mathbf{x} \in [\mathbf{x}] \mid \mathbf{f}(\mathbf{x}) = \mathbf{0}, \tag{2.7.8}$$

d'une manière équivalente

$$\mathcal{N}\left([\mathbf{f}], [\mathbf{J_f}], [\mathbf{x}]\right) \subset [\mathbf{x}] \Rightarrow \forall \mathbf{f} \in [\mathbf{f}], \exists! \mathbf{x} \in [\mathbf{x}] \mid \mathbf{f}(\mathbf{x}) = \mathbf{0}, \tag{2.7.9}$$

**Exemple 2.7.3.** *Une fonction incertaine peut être représentée par un intervalle de fonctions, mais ce dernier peut aussi être défini par une fonction paramétrée :*

$$\mathbf{f}(x) \in [\mathbf{f}](x) \text{ avec } [\mathbf{f}](x) = \{\varphi(\mathbf{p}, x), \mathbf{p} \in [\mathbf{p}]\}$$

*Soit* $f(x, \mathbf{p}) = \cos(p_1.x + p_2) - p_3.x$ *avec* $[\mathbf{p}] \in [1, 1.1]^3$. *Cette fonction est représentée sur la figure 2.22. On cherche à prouver l'existence et l'unicité d'une racine de cette fonction dans l'intervalle* $[x] = [0.15, 0.4]$ *représenté en noir.*

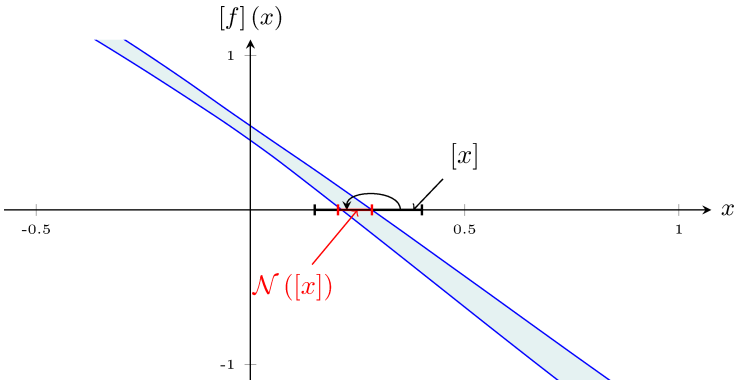



FIGURE 2.22 – Illustration de la méthode de Newton par intervalles sur un tube.

*On peut voir que* $\mathcal{N}([x]) \subset [x]$. *On a donc la garantie qu'il existe un* $x^* \in [x]$ *tel que* $f(x^*, \mathbf{p}) = 0$ *avec* $\mathbf{p} \in [\mathbf{p}]$.

**Remarque 2.7.3.** *Attention, le cas ou* $\mathcal{N}([\mathbf{x}]) \not\subset [\mathbf{x}]$ *ne veut pas dire qu'il n'existe pas de zéro pour la fonction* $\mathbf{f}$ *dans l'intervalle* $[\mathbf{x}]$.

## 2.8 Conclusion

Ce chapitre a présenté les bases de l'analyse par intervalles au travers de la théorie ensembliste et des relations d'ordre. L'arithmétique présentée sur les intervalles réels et étendue aux intervalles de $\mathbb{R}^n$ ainsi qu'aux intervalles de fonctions nous permet d'encadrer un ou plusieurs nombres flottants entachés d'erreurs. Cet encadrement peut être utilisé dans différentes conditions : pour encadrer l'évolution au cours du temps d'une fonction incertaine avec les tubes, une position en plusieurs dimensions ou un ensemble de $n$ paramètres incertains. Dans le cadre de la robotique sous-marine, les capteurs proprioceptifs sont souvent bien plus précis que dans un environnement terrestre. Ceci permet une représentation précise des dites mesures par des tubes qui respectent l'hypothèse d'erreurs bornées présentée sur différents exemples et nécessaire à l'outil ensembliste. Pour notre problème de détection de fermeture de boucles, nous utiliserons l'analyse par intervalles pour représenter l'évolution du robot au cours du temps. Cette représentation implique une méthode de résolution ensembliste comme l'algorithme de caractérisation du noyau. La propriété d'existence et unicité apportée par le test de Newton apporte une fiabilité dans les résultats qui répond totalement au besoin de la robotique pour cette question car, si les résultats sont indiscutablement fiables, les sources d'erreurs sont diminuées.

# Chapitre 3

# Détection de fermeture de boucle par mesures proprioceptives

**Sommaire**


**3.1 Introduction . . . . . 80**
3.1.1 Définition du problème . . . . . 80
3.1.2 Formalisation du problème . . . . . 82
**3.2 Détection de fermeture de boucle par mesures proprioceptives . . . . . 83**
3.2.1 Tests . . . . . 84
3.2.1.1 Condition $t_1 < t_2$ . . . . . 84
3.2.1.2 Test intégrale . . . . . 85
3.2.1.3 Test d'injectivité . . . . . 86
3.2.2 Algorithme . . . . . 88
3.2.3 Application sur données simulées . . . . . 90
**3.3 Validation de l'existence d'une boucle . . . . . 91**
3.3.1 Test de Newton . . . . . 91
3.3.2 $\mathcal{E}$-inflation . . . . . 95
3.3.3 Recherche de boîtes englobantes . . . . . 96
3.3.4 $\mathcal{N}$-inflation . . . . . 97
**3.4 Application sur données réelles . . . . . 99**
3.4.1 Présentation de l'expérimentation . . . . . 99
3.4.2 Résultats . . . . . 100
**3.5 Trajectoire de retour sans boucles . . . . . 103**
**3.6 Conclusion . . . . . 105**

## 3.1 Introduction

Ce chapitre présente une nouvelle formalisation du problème de détection de fermeture de boucle et propose une méthode de résolution qui utilise les outils développés dans le chapitre précédent. En effet, nous verrons comment les mesures effectuées au cours de la mission d'un robot mobile peuvent être représentées sous forme de tubes et comment ces informations peuvent être prises en compte pour la résolution du problème. L'algorithme KER nous servira de base pour le développement d'une méthode permettant de retrouver les temps associés à un événement de fermeture de boucle dans la trajectoire d'un robot mobile. Enfin, nous verrons comment la détection de fermeture de boucle revient à trouver le noyau d'une fonction intervalle de $\mathbb{R}^n \rightarrow \mathbb{R}^n$. L'utilisation du test de Newton sur les résultats produits par l'algorithme nous permettra parfois de garantir la fiabilité des résultats sans possibilité de fausse détection.

### 3.1.1 Définition du problème

Si l'on considère un robot mobile sur un plan, sa trajectoire peut être décrite par les fonctions $x(t)$, $y(t)$ et $\theta(t)$ définies dans la durée de la mission $t \in [0, t_{max}]$. Si on cherche maintenant à détecter les boucles que le robot a faites dans sa trajectoire, il faut trouver deux configurations définies à deux instants différents dans le temps ou la position $x, y$ est identique. De plus, il faut s'assurer que le robot a bien effectué un mouvement significatif entre ces deux instants afin de ne pas détecter une boucle lorsque le robot est immobile ou avance très lentement. Cet énoncé nous amène à définir une boucle. Une boucle dans la trajectoire d'un robot mobile est définie comme deux positions identiques à deux instants éloignés dans le temps, temps pendant lequel le robot aura effectué un mouvement significatif lui permettant de revenir sur sa trajectoire passée.

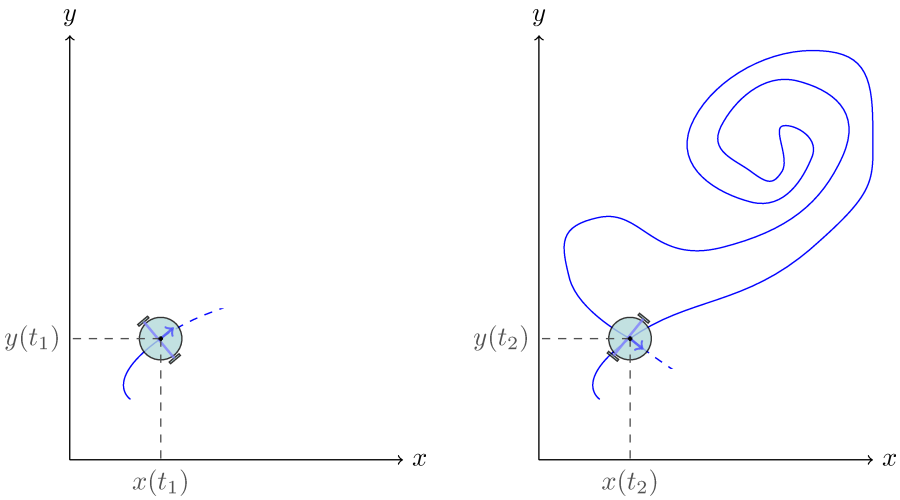



FIGURE 3.1 – En (a), le robot en position $t_1$. En (b), le robot en position $t_2$. La trajectoire effectuée par le robot est représentée en trait plein. La position du robot à l'instant $t_1$ est identique à celle à l'instant $t_2$ : la condition $t_2 > t_1$ est bien respectée et on observe un mouvement significatif entre ces deux instants. Le robot a donc effectué une boucle dans sa trajectoire.

Cette définition implique une détection de fermeture de boucle par une recherche sur les temps composants la trajectoire. En effet, la méthode présentée dans ce chapitre va chercher à répondre formellement à la définition précédente. Cette idée rejoint le concept de comparaison entre distance métrique et distance topologique développée dans [Stachniss et al. 2004]. En effet, la distance topologique se base sur le chemin parcouru entre deux instants. Nous avons vu dans le premier chapitre que la détection de fermeture de boucle s'effectue le plus souvent par des mesures extéroceptives. Dans notre contexte de robotique sous-marine, ces mesures sont peu fiables et souvent fortement contraintes (turbidité de l'eau pour les caméras, objets en suspension, fort taux d'aliasing perceptuel). Nous allons donc montrer comment cette détection de fermeture de boucle peut être effectuée sur la base de mesures proprioceptives, mesures bien plus fiables dans le milieu aquatique que les données extéroceptives.

Nous allons, dans un premier temps, définir le problème de détection de fermeture de boucle dans le contexte de mesures proprioceptives. Nous verrons ensuite comment cette question est formalisée en analyse par intervalles via l'utilisation

des tubes et de différents tests d'inclusion. Enfin, nous présenterons les résultats de l'algorithme sur des données réelles.

### 3.1.2 Formalisation du problème

La définition précédente d'une boucle doit être traduite mathématiquement. Définissons $\mathbf{p}(t) = (x(t), y(t))^T$ la position du robot sur un plan. Le temps $t$ appartient à l'intervalle $[0, t_{max}]$, avec $t_{max}$ la durée de la mission du robot. L'ensemble des boucles de la trajectoire peut être représenté par l'ensemble :

$$\mathbb{T}^* = \left\{ (t_1, t_2) \in [0, t_{\max}]^2 \mid \mathbf{p}(t_1) = \mathbf{p}(t_2),\ t_1 < t_2 \right\}. \tag{3.1.1}$$

Sur l'exemple présenté sur la figure 3.1, l'ensemble $\mathbb{T}^*$ comporte un seul élément $\mathbf{t} = (t_1, t_2)$ alors qu'on en compte trois sur l'exemple de la figure 3.2. Un élément $\mathbf{t} = (t_1, t_2)$ sera appelé *t-paire*. L'ensemble des t-paire sera représenté sur un plan appelé *t-plan* représenté sur la figure 3.2(a). Le t-plan est un outil de représentation des t-paires. Il s'agit d'un plan composé en abscisses d'un temps $t_a$ et en ordonnée $t_b$ ; on lira la coordonée $t_1$ d'une t-paire sur l'axe des abscisses et la coordonée $t_2$ sur l'axe des ordonnées. Pour plus de commodité, les t-paire seront qualifiés de boucles.

Plusieurs catégories de boucles apparaissent : les boucles englobantes, les boucles simples qui n'englobent aucune boucle et les boucles imbriquées, que nous verrons par la suite. Prenons l'exemple de deux boucles $\mathbf{t_a} = (t_{1,a}, t_{2,a})$ et $\mathbf{t_b} = (t_{1,b}, t_{2,b})$. Si $[t_{1,a}, t_{2,a}] \subset [t_{1,b}, t_{2,b}]$, nous dirons que la boucle $\mathbf{t_b}$ englobe la boucle $\mathbf{t_a}$. Nous verrons en section 3.5 comment la qualification de ces boucles peut nous permettre de trouver un chemin de retour sans boucles pour le robot.

Ces nouveaux concepts ayant été introduits, la section suivante peut maintenant présenter notre méthode de détection de boucles par mesures proprioceptives.

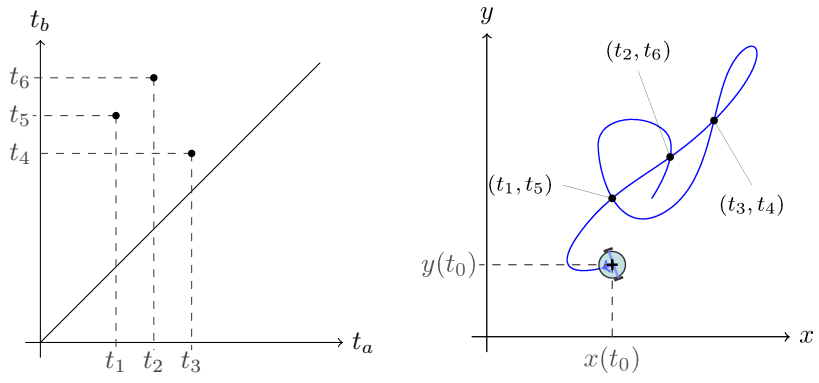



FIGURE 3.2 – T-plane (a) correspondant à une trajectoire (b) comportant plusieurs boucles.

## 3.2 Détection de fermeture de boucle par mesures proprioceptives

Nous considérons le cas d'un robot mobile, dont nous connaissons les équations d'état, équipé de capteurs lui permettant de connaître sa vitesse (i.e. DVL, roues codeuses) et d'un capteur lui permettant de connaître son orientation (i.e. boussole, centrale inertielle). Notons cette vitesse $\mathbf{v}(t) \in \mathbb{R}^2$. Le temps $t$ appartient à l'intervalle $[0, t_{max}]$, avec $t_{max}$ la durée de la mission du robot. La position du robot sur un plan, notée $\mathbf{p}(t) = (x(t), y(t))^T$ est définie par :

$$\mathbf{p}\,(t) = \int_0^t \mathbf{v}\,(\tau)\, d\tau, \tag{3.2.1}$$

et correspond à la position du robot centrée autour de sa position d'origine notée $\mathbf{p}\,(0)$.

L'ensemble des boucles de la trajectoire définie par l'équation (3.1.1) peut être réécrit comme :

$$\mathbb{T}^* = \left\{ (t_1, t_2) \in [0, t_{\max}]^2 \mid \int_{t_1}^{t_2} \mathbf{v}(\tau) d\tau = \mathbf{0},\ t_1 < t_2 \right\}. \tag{3.2.2}$$

Supposons que le robot mesure sa vitesse $\mathbf{v}$ dans un contexte d'erreurs bornées, c'est-à-dire qu'une boîte $[\mathbf{v}]\,(t)$ est connue pour tout $t \in [0, t_{max}]$. On est capable

de définir l'ensemble des t-paire $\mathbf{t} = (t_1, t_2)$ réalisables comme :

$$\mathbb{T} = \left\{ \mathbf{t} \mid 0 \leq t_1 < t_2 \leq t_{max},\ \exists \mathbf{v} \in [\mathbf{v}]\,,\ \int_{t_1}^{t_2} \mathbf{v}(\tau) d\tau = \mathbf{0} \right\}. \tag{3.2.3}$$

où $[\mathbf{v}]$ est un tube pour la vitesse $\mathbf{v}$ et l'ensemble $\mathbb{T}^*$ est inclus dans l'ensemble $\mathbb{T}$. L'algorithme principal de cette thèse propose de trouver un ensemble extérieur et un ensemble intérieur pour l'ensemble $\mathbb{T}$.

### 3.2.1 Tests

Les deux ensembles, intérieur et extérieur qui vont nous permettre de décrire l'ensemble $\mathbb{T}$, vont être représentés par des sous pavages réguliers. Ces sous pavages seront constitués de boîtes qui contiendront des t-paire. Ces boîtes seront issues du découpage d'un domaine initial de recherche, le t-plan, dimensionné par $t_{max}$. Nous appellerons *t-box* une boîte du t-plan définie par $[\mathbf{t}] = [t_1] \times [t_2]$. Les tests décrits dans cette section vont permettre de déterminer l'appartenance de ces boîtes à l'ensemble $\mathbb{T}$. Ainsi, si $[\mathbf{t}] \subset \mathbb{T}$, nous dirons que $[\mathbf{t}]$ est *réalisable*. Si $[\mathbf{t}] \cap \mathbb{T} = \emptyset$, $[\mathbf{t}]$ sera *irréalisable*. Dans tous les autres cas, la boîte sera qualifiée d'*indéterminée*. Nous verrons aussi qu'à l'aide d'un test d'existence et unicité, nous serons capables, dans certaines situations, de prouver qu'une boîte $[\mathbf{t}]$ contient un unique élément de $\mathbb{T}^*$. Ce test nous permettra aussi de compter le nombre d'éléments de $\mathbb{T}^*$ et ainsi de connaître le nombre de boucles dans la trajectoire du robot.

#### 3.2.1.1 Condition $t_1 < t_2$

Les définitions de l'ensemble des boucles réalisables (équations 3.2.2, 3.2.3) impliquent la condition $t_1 < t_2$. Cette condition nous permet de découper le t-plan en deux parties par rapport à la droite $t_1 = t_2$ et de ne travailler que sur un demi plan. Si cette condition n'était pas présente, le t-plan serait symétrique par rapport à cette droite et les résultats seraient doublés. En effet, si on cherche $\mathbf{p}(t_1) = \mathbf{p}(t_2)$ sans la condition $t_1 < t_2$ et qu'une boucle existe pour ces deux temps, deux couples $(t_1, t_2)$ et $(t_2, t_1)$ sont solutions de cette équation et ces solutions sont symétriques par rapport à la droite $t_1 = t_2$. De plus, cette même droite serait elle-même composée de t-box indéterminées (voir remarque 3.2.1).

Dans notre cas, on teste des boîtes $[\mathbf{t}] = [t_1] \times [t_2]$. Une boîte qui ne respecte pas la condition $t_1 < t_2$ n'est pas, pour autant, à déclarer irréalisable. En effet,

les seules boîtes qui peuvent être rejetées à travers cette condition sont celles qui satisfont la condition $t_2 < t_1$ dans sa version intervalisée :

$$[t_1] - [t_2] \subset \mathbb{R}^+ \Rightarrow [\mathbf{t}] \cap \mathbb{T} = \emptyset. \tag{3.2.4}$$

Les boîtes qui répondent à cette contrainte sont celles appartenant au demi t-plan inférieur lorsque ce dernier est découpé selon la droite $t_1 = t_2$. Les autres boîtes composant le t-plan seront considérées comme potentiellement réalisables si elles respectent la condition $t_1 < t_2$ et donc répondent au test :

$$[t_1] - [t_2] \subset \mathbb{R}^-. \tag{3.2.5}$$

Ces deux tests permettent donc d'éliminer plus de la moitié de l'espace de recherche initial et de sélectionner les boîtes qui pourront être testées par la suite.

#### 3.2.1.2 Test intégrale

Ce test se divise en deux parties et va nous permettre d'inclure (ou d'exclure) une boîte de l'ensemble $\mathbb{T}$. La première partie concerne les boîtes irréalisables :

$$\mathbf{0} \notin \int_{[t_1]}^{[t_2]} [\mathbf{v}](\tau) d\tau \Rightarrow [\mathbf{t}] \cap \mathbb{T} = \emptyset. \tag{3.2.6}$$

En effet, nous pouvons écrire que :

$$\mathbf{0} \notin \int_{[t_1]}^{[t_2]} [\mathbf{v}](\tau) d\tau \;\Rightarrow\; \mathbf{0} \notin \Big( [\mathbf{p}]([t_2]) - [\mathbf{p}]([t_1]) \Big), \tag{3.2.7}$$

ce qui veut dire que les positions aux instants $t_1$ et $t_2$, qui sont incertaines et représentées par des boîtes, ne s'intersectent pas. Il est donc impossible de trouver une boucle entre ces deux temps et la boîte est déclarée irréalisable.

La seconde partie concerne les boîtes réalisables. Lorsqu'une boîte $[\mathbf{t}]$ respecte la condition $t_2 > t_1$ (équation 3.2.5) nous pouvons écrire :

**Proposition 3.2.1.** *Pour une boîte* $[\mathbf{t}] = [t_1] \times [t_2]$ *et si* $[t_1] - [t_2] \subset \mathbb{R}^-$ *:*

$$\int_{t_1}^{t_2} \mathbf{v}^-(\tau) d\tau \leq 0 \leq \int_{t_1}^{t_2} \mathbf{v}^+(\tau) d\tau \Rightarrow [\mathbf{t}] \subset \mathbb{T} \tag{3.2.8}$$

*Démonstration.* L'équation 3.2.3 nous dit qu'une boîte $[\mathbf{t}] \subset \mathbb{T}$ si elle satisfait la condition :

$$\exists \mathbf{v} \in [\mathbf{v}], \int_{t_1}^{t_2} \mathbf{v}(\tau)d\tau = \mathbf{0}, \tag{3.2.9}$$

qui est équivalente à :

$$\begin{aligned}
&\begin{cases} \exists v_x \in [v_x], \int_{t_1}^{t_2} v_x(\tau)d\tau = 0 \\ \exists v_y \in [v_y], \int_{t_1}^{t_2} v_y(\tau)d\tau = 0 \end{cases} \\
\Leftrightarrow \quad &\begin{cases} \int_{t_1}^{t_2} v_x^-(\tau)d\tau \leq 0 \text{ et } \int_{t_1}^{t_2} v_x^+(\tau)d\tau \geq 0 \\ \int_{t_1}^{t_2} v_y^-(\tau)d\tau \leq 0 \text{ et } \int_{t_1}^{t_2} v_y^+(\tau)d\tau \geq 0 \end{cases} \\
\Leftrightarrow \quad &\int_{t_1}^{t_2} \mathbf{v}^-(\tau)d\tau \leq \mathbf{0}, \int_{t_1}^{t_2} \mathbf{v}^+(\tau)d\tau \geq \mathbf{0}.
\end{aligned}$$

□

Ce test nous permet donc de caractériser les boîtes appartenant à l'ensemble intérieur qui caractérise l'ensemble $\mathbb{T}$. Il s'agit en fait du noyau de la fonction défini par :

$$[\mathbf{f}]([\mathbf{t}]) = \left[\int_{t_1}^{t_2} \mathbf{v}^-(\tau)d\tau, \int_{t_1}^{t_2} \mathbf{v}^+(\tau)d\tau\right]. \tag{3.2.10}$$

#### 3.2.1.3 Test d'injectivité

Quand les deux intervalles de temps $[t_1]$ et $[t_2]$ formant la boîte $[\mathbf{t}]$ s'intersectent, le test intégrale n'est pas capable de rejeter une t-box, même si elle est de très faible taille. En effet, si les deux temps $[t_1]$ et $[t_2]$ s'intersectent, le test intégrale compare deux parties de trajectoires qui s'intersectent forcément. Dans ce cas, il est toujours possible de rejeter une t-box si la fonction $\mathbf{p}(t)$ définie par l'équation 3.2.1 est injective dans l'intervalle $\left[t_1^-, t_2^+\right]$ (attention, ce test d'injectivité nécessite $t_2 > t_1$ afin que $\left[t_1^-, t_2^+\right]$ ne soit pas impropre). En effet, si l'injectivité de la fonction est prouvée dans ce domaine, nous ne serons pas capables de trouver un $t_1 \in [t_1]$ et un $t_2 \in [t_2]$ tels que $\mathbf{p}(t_1) = \mathbf{p}(t_2)$ et $t_1 \neq t_2$. Dans ce cas, la boîte $[\mathbf{t}]$ sera classée comme irréalisable. Le test d'injectivité nous permet donc d'exclure de l'ensemble $\mathbb{T}$ les boîtes de petite taille qui ne peuvent être écartées par le test intégrale. La proposition suivante formalise cette condition d'injectivité locale :

**Proposition 3.2.2.** *Pour une boîte* $[\mathbf{t}] = [t_1] \times [t_2]$*, nous avons :*

$$\mathbf{0} \notin [\mathbf{v}]\left(\left[t_1^-, t_2^+\right]\right) \;\Rightarrow\; [\mathbf{t}] \cap \mathbb{T} = \emptyset.$$

*Démonstration.* Pour démontrer la proposition par l'absurde, émettons deux hypothèses :

$$\mathbf{0} \notin [\mathbf{v}]\left(\left[t_1^-, t_2^+\right]\right) \text{ et } [\mathbf{t}] \cap \mathbf{T} \neq \emptyset.$$

Si on choisit un couple $(t_1, t_2) \in [\mathbf{t}] \cap \mathbb{T}$, l'équation 3.2.3 nous dit que

$$\exists \mathbf{v} \in [\mathbf{v}], \int_{t_1}^{t_2} \mathbf{v}(\tau) d\tau = \mathbf{0}. \tag{3.2.11}$$

La fonction $\mathbf{q}(t) = \int_0^t \mathbf{v}(\tau)\, d\tau$, (voir équation 3.2.1) est dérivable par rapport à $t$. Nous pouvons donc appliquer le théorème des accroissements finis généralisé [Lagrange et al. 2007] sur l'intervalle $\left[t_1^-, t_2^+\right]$. Et donc si $t_1, t_2 \in \left[t_1^-, t_2^+\right]$, alors :

$$\exists \bar{\mathbf{v}} \in \left[\frac{d\mathbf{q}}{dt}\left(\left[t_1^-, t_2^+\right]\right)\right], \mathbf{q}(t_2) - \mathbf{q}(t_1) = \bar{\mathbf{v}} \cdot (t_2 - t_1), \tag{3.2.12}$$

où $\left[\frac{d\mathbf{q}}{dt}\left(\left[t_1^-, t_2^+\right]\right)\right]$ est une boîte qui inclut l'ensemble $\left\{\frac{d\mathbf{q}}{dt}(t) \mid t \in \left[t_1^-, t_2^+\right]\right\}$. Maintenant, nous savons de l'équation 3.2.11 que $\mathbf{q}(t_2) - \mathbf{q}(t_1) = \mathbf{0}$ et $\frac{d\mathbf{q}}{dt}(t) = \mathbf{v}(t)$. Alors, l'équation 3.2.12 implique que $\exists \bar{\mathbf{v}} \in [\mathbf{v}]\left(\left[t_1^-, t_2^+\right]\right)$, $\mathbf{0} = \bar{\mathbf{v}} \cdot (t_2 - t_1)$, *i.e.*, $\exists \bar{\mathbf{v}} \in [\mathbf{v}]\left(\left[t_1^-, t_2^+\right]\right)$, $\bar{\mathbf{v}} = \mathbf{0}$. Ce qui entre en contradiction avec notre supposition de départ $\mathbf{0} \notin [\mathbf{v}]\left(\left[t_1^-, t_2^+\right]\right)$. □

**Remarque 3.2.1.** *Ce test permet d'exclure les boîtes* $[\mathbf{t}]$ *qui forment la droite* $t_1 = t_2$ *car ces dernières ne peuvent être écartées ni par le test intégrale (qui nécessite* $t_2 > t_1$*) ni par l'une des conditions qui découpent en deux le t-plan (équations 3.2.5 et 3.2.4). Si ce test d'injectivité n'était pas appliqué, les boîtes encadrant la droite* $t_1 = t_2$ *ne pourraient pas être discriminées et seraient bissectées jusqu'à atteindre la taille* $\varepsilon$ *avant d'être qualifiées d'indéterminées, ce qui augmenterait le nombre de bissections et donc le temps de traitement.*

### 3.2.2 Algorithme

L'algorithme 2, issu de [Aubry et al. 2013], propose une caractérisation de l'ensemble $\mathbb{T}$ définie par l'équation 3.2.3. Comme l'algorithme KER (algorithme 1), cet algorithme de type *branch-and-bound* est similaire à l'algorithme SIVIA. L'algorithme va partitionner le t-plan en trois sous pavages réguliers en utilisant les tests développés dans la section 3.2.1. En entrée, l'algorithme nécessite une fonction incertaine $[\mathbf{v}]$, une précision $\varepsilon$ et $t_{max}$ la durée de la mission à analyser. Les sous pavages retournés par l'algorithme vont encadrer l'ensemble $\mathbb{T}$. L'ensemble $\mathbb{T}^{\text{in}}$ contiendra les boîtes $[\mathbf{t}]$ réalisables, l'ensemble $\mathbb{T}^{\text{out}}$ contiendra les boîtes $[\mathbf{t}]$ irréalisables et enfin, l'ensemble $\mathbb{T}^{?}$ contiendra les boîtes $[\mathbf{t}]$ indéterminées. Les boîtes appartenant à ce dernier ensemble sont toutes les boîtes de petite taille qui n'ont passé aucun des tests. En ligne 2, l'algorithme est initialisé, les listes de sortie sont vidées et le domaine initial de recherche est construit à partir de la durée de la mission. La boucle principale commence ligne 4 et sera parcourue tant que la liste n'aura pas été vidée, i.e. tant que des boîtes du t-plan n'auront pas trouvé leur place dans une des listes de sortie. Dans cette boucle principale, on extrait une boîte de la liste pour la soumettre aux tests de la section 3.2.1 et démontrer si elle appartient ou non à l'ensemble $\mathbb{T}$. Si un des tests réussit, l'algorithme continue avec une autre boîte. Les boîtes de petite taille pour lesquelles rien n'a été prouvé sont rangées dans la liste $\mathbb{T}^{?}$ (une boîte est considérée de petite taille si la taille maximale parmi les tailles des intervalles qui la compose est inférieure au paramètre $\varepsilon$). Si toutes ces étapes échouent, c'est-à-dire que la boîte extraite de la liste en ligne 2 n'a passé aucun des tests et que sa taille est supérieure à $\varepsilon$, la boîte est bissectée et le résultat placé dans la liste $\mathcal{Q}$. L'algorithme finit lorsque l'on sort de la boucle principale et on obtient un encadrement pour l'ensemble $\mathbb{T}$ [Jaulin et Walter 1993] :

$$\mathbb{T}^{-} \subset \mathbb{T} \subset \mathbb{T}^{+},$$

ou encore :

$$\mathbb{T}^{\text{in}} \subset \mathbb{T} \subset \left(\mathbb{T}^{\text{in}} \cup \mathbb{T}^{?}\right).$$

**Algorithme 2 :** LOOP

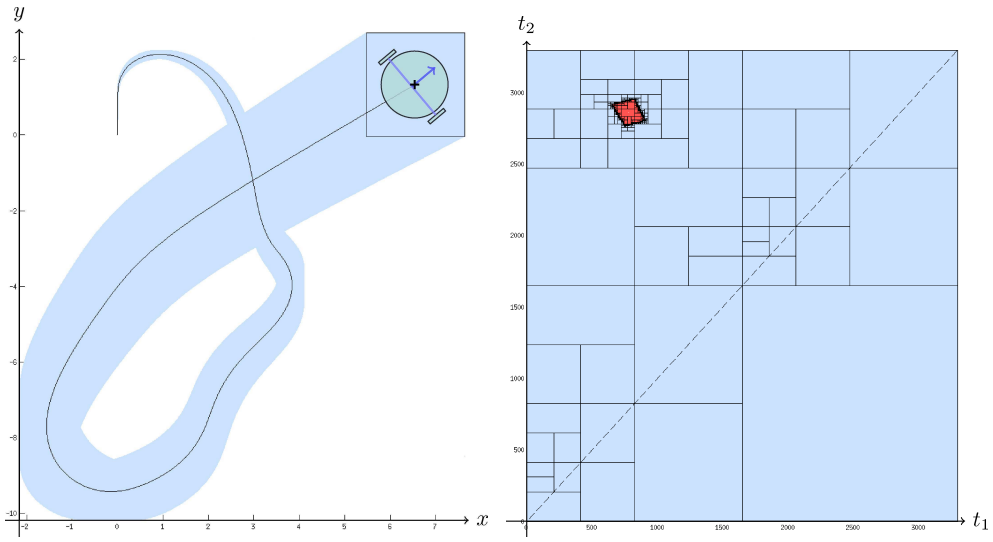
**Entrées** : $\varepsilon, t_{max}, [\mathbf{v}]$

1 // Initialisation
2 $\mathcal{Q} \leftarrow \{[0, t_{max}] \times [0, t_{max}]\}$ ; $\mathbb{T}^{\text{in}} = \emptyset$ ; $\mathbb{T}^{\text{out}} = \emptyset$ ; $\mathbb{T}^{?} = \emptyset$;
3 // Boucle principale
4 **tant que** $\mathcal{Q}$ *n'est pas vide* **faire**
5 $\quad [\mathbf{t}] \leftarrow \mathcal{Q}.prendre\_premier\_élément()$ ; // L'élément est retiré de la liste et rangé dans $[\mathrm{t}]$
6 $\quad$ // Construction du sous pavage extérieur
7 $\quad$ **si** $\left( \begin{array}{l} [t_1] - [t_2] \subset \mathbb{R}^+ \ OU \\ \mathbf{0} \notin \int_{[t_1]}^{[t_2]} [\mathbf{v}](\tau) d\tau \ OU \\ \mathbf{0} \notin [\mathbf{v}]\left(\left[t_1^-, t_2^+\right]\right) \end{array} \right)$ **alors**
8 $\quad\quad \mathbb{T}^{\text{out}} \leftarrow \mathbb{T}^{\text{out}} \cup [\mathbf{t}]$;
9 $\quad$ // Construction du sous pavage intérieur
10 $\quad$ **sinon si** $\left( \ [t_1] - [t_2] \subset \mathbb{R}^- \ ET \ \int_{[t_1]}^{[t_2]} \mathbf{v}^-(\tau) d\tau \leq 0 \leq \int_{[t_1]}^{[t_2]} \mathbf{v}^+(\tau) d\tau \ \right)$ **alors**
11 $\quad\quad \mathbb{T}^{\text{in}} \leftarrow \mathbb{T}^{\text{in}} \cup [\mathbf{t}]$;
12 $\quad$ **sinon si** $width([\mathbf{t}]) < \varepsilon$ **alors**
13 $\quad\quad \mathbb{T}^{?} \leftarrow \mathbb{T}^{?} \cup [\mathbf{t}]$;
14 $\quad$ **sinon**
15 $\quad\quad$ // La boîte est bissectée et le résultat mis dans la liste $\mathcal{Q}$
16 $\quad\quad ([\mathbf{t_A}], [\mathbf{t_B}]) = Bissecter([\mathbf{t}])$;
17 $\quad\quad \mathcal{Q} \leftarrow \{\mathcal{Q}; [\mathbf{t_A}]; [\mathbf{t_B}]\}$;
18 $\quad$ **fin**
19 **fin**

**Sorties** : $\mathbb{T}^{\text{in}}, \mathbb{T}^{?}, \mathbb{T}^{\text{out}}$

### 3.2.3 Application sur données simulées

Un robot mobile effectue une trajectoire en relevant sa vitesse à intervalles de temps réguliers. Ce relevé de vitesse est connu pour être entaché d'erreurs mais ces dernières sont bornées. La vitesse peut donc être représentée sous la forme d'un tube. La figure 3.3(a) nous montre la représentation d'un robot qui exécute une trajectoire contenant une unique boucle. Si on applique l'algorithme LOOP(algorithme 2), on obtient le découpage représenté sur la figure 3.3(b). Sur




(a) Trajectoire du robot (b) T-plane correspondant à la trajectoire

FIGURE 3.3 – Exemple d'application de l'algorithme LOOP sur des données simulées.

cet exemple, on peut voir que le t-plan est bien partitionné en plusieurs ensembles. $\mathbb{T}^{\text{out}}$ est l'ensemble bleu qui occupe la plus grande partie du t-plan. Toutes les coordonnées $[\mathbf{t}] = [t_1] \times [t_2]$ qui appartiennent à cet ensemble correspondent à deux parties de la trajectoire (l'une de $\mathbf{p}\left(t_1^-\right)$ à $\mathbf{p}\left(t_1^+\right)$ et l'autre de $\mathbf{p}\left(t_2^-\right)$ à $\mathbf{p}\left(t_2^+\right)$) sans boucle. Les boîtes rouges en haut à gauche du t-plan forment l'ensemble $\mathbb{T}^{\text{in}}$. La figure 3.4 présente un zoom sur cette partie du t-plan et on peut y voir, à la frontière des ensembles $\mathbb{T}^{\text{in}}$ et $\mathbb{T}^{\text{out}}$, des boîtes jaunes qui forment l'ensemble $\mathbb{T}^?$. Une boucle dans la trajectoire est donc décrite par un sous pavage de l'ensemble $\mathbb{T}^+ = \mathbb{T}^{\text{in}} \cup \mathbb{T}^?$.

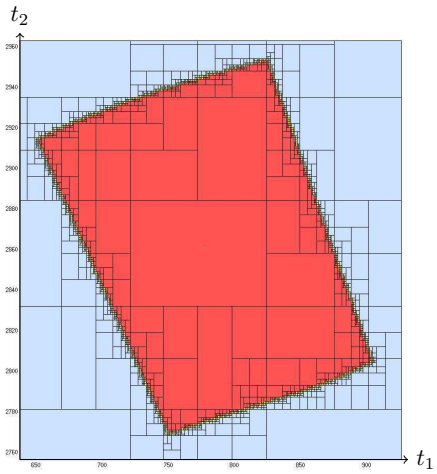



FIGURE 3.4 – Zoom sur le sous pavage représentant la boucle.

## 3.3 Validation de l'existence d'une boucle

La caractérisation de l'ensemble $\mathbb{T}$ ne suffit pas à répondre à la problématique de détection de fermeture de boucle. En effet, l'ensemble $\mathbb{T}^+$ contient tous les couples $(t_1, t_2)$ pour lesquels une boucle est possible dans la trajectoire incertaine du mobile (le statut indéterminé des boîtes appartenant à l'ensemble $\mathbb{T}^?$ nous oblige à l'inclure à l'ensemble des boucles réalisables). Ceci ne veut pas dire qu'une unique boucle existe pour chacune des boîtes qui forment l'ensemble $\mathbb{T}^+$. Le test de Newton intervient alors afin de montrer l'existence et l'unicité d'une boucle dans la trajectoire pour une boîte $[\mathbf{t}] = [t_1] \times [t_2]$.

### 3.3.1 Test de Newton

L'utilisation du test de Newton pour la garantie d'unicité d'une solution dans le contexte de systèmes dynamiques a été proposée pour la première fois dans [Tucker 1999] pour prouver l'existence de l'attracteur de Lorentz (14eme problème de Smale). Nous avons vu, en section 2.7.3, que le test de Newton peut nous apporter la garantie d'existence et unicité, même dans le cas de fonctions incertaines appartenant à un intervalle de fonction. Rappelons tout d'abord l'expression de

cet opérateur : pour une fonction incertaine définie par :

$$\begin{aligned}[\mathbf{f}] : \mathbb{IR}^n &\to \mathbb{IR}^n \\ \mathbf{t} &\mapsto [\mathbf{f}](\mathbf{t}),\end{aligned}$$

si cette fonction est contiûment dérivable par rapport à $\mathbf{t}$ et si $[\mathbf{J_f}]$ désigne une fonction d'inclusion pour $\mathbf{J_f}$, la matrice Jacobienne de la fonction $\mathbf{f} \in [\mathbf{f}]$. Pour $\hat{\mathbf{t}}$ un point de $[\mathbf{t}]$ (souvent choisi comme le centre de $[\mathbf{t}]$), l'opérateur de Newton est défini par :

$$\mathcal{N}([\mathbf{f}], [\mathbf{J_f}], [\mathbf{t}]) = \hat{\mathbf{t}} - [\mathbf{J_f}]^{-1}([\mathbf{t}]) \cdot [\mathbf{f}](\hat{\mathbf{t}}). \tag{3.3.1}$$

De plus, Moore [R. E. Moore 1979] nous dit que

$$\mathcal{N}([\mathbf{f}], [\mathbf{J_f}], [\mathbf{t}]) \subset [\mathbf{t}] \Rightarrow \forall \mathbf{f} \in [\mathbf{f}], \exists! \mathbf{t} \in [\mathbf{t}] \mid \mathbf{f}(\mathbf{t}) = \mathbf{0}. \tag{3.3.2}$$

Ce test va nous permettre de comptabiliser les éléments qui forment l'ensemble $\mathbb{T}^*$.

Dans notre cas, on a :

$$[\mathbf{f}](\mathbf{t}) = \int_{t_1}^{t_2} [\mathbf{v}](\tau) d\tau, \tag{3.3.3}$$

et la Jacobienne $[\mathbf{J_f}]$ est définie par :

$$[\mathbf{J_f}](\mathbf{t}) = \left( \frac{\partial [\mathbf{f}]}{\partial t_1}(\mathbf{t}) \quad \frac{\partial [\mathbf{f}]}{\partial t_2}(\mathbf{t}) \right) = \left( -[\mathbf{v}](t_1) \quad [\mathbf{v}](t_2) \right). \tag{3.3.4}$$

Nous pouvons donc définir, comme suit, un test d'existence et unicité pour notre application.

**Proposition 3.3.1.** *Soit* $[\mathbf{t}] = [t_1] \times [t_2]$ *tel que* $[t_1] - [t_2] \subset \mathbb{R}^-$.

$$\mathcal{N}([\mathbf{v}], [\mathbf{t}]) = \hat{\mathbf{t}} - \left( -[\mathbf{v}]([t_1]) \quad [\mathbf{v}]([t_2]) \right)^{-1}([\mathbf{t}]) \cdot \int_{\hat{t}_1}^{\hat{t}_2} [\mathbf{v}](\tau) d\tau \tag{3.3.5}$$

*et si*

$$\mathcal{N}([\mathbf{v}], [\mathbf{t}]) \subset [\mathbf{t}] \tag{3.3.6}$$

*alors il existe un unique* $\mathbf{t} \in [\mathbf{t}]$ *tel que* $\mathbf{t} \in \mathbb{T}^*$.

**Remarque 3.3.1.** *Quel que soit le résultat du test de Newton, l'ensemble des*

$\mathbf{t} = (t_1, t_2) \in [\mathbf{t}]$ *réalisables est défini par :*

$$\hat{\mathbb{T}}\left([\mathbf{v}], [\mathbf{t}]\right) = \bigcup_{\mathbf{v} \in [\mathbf{v}]} \left\{ (t_1, t_2) \in [\mathbf{t}] \mid \int_{t_1}^{t_2} \mathbf{v}(\tau) d\tau = \mathbf{0} \right\}$$

*et possède un nombre d'éléments infini. Nous pouvons aussi dire que, pour chaque fonction* $\mathbf{v} \in [\mathbf{v}]$*, l'ensemble :*

$$\hat{\mathbb{T}}\left(\mathbf{v}, [\mathbf{t}]\right) = \left\{ (t_1, t_2) \in [\mathbf{t}] \mid \int_{t_1}^{t_2} \mathbf{v}(\tau) d\tau = \mathbf{0} \right\}$$

*possède un élément unique. Ceci est également valable pour la vitesse réelle* $\mathbf{v}^*$ *puisqu'elle appartient aussi à* $[\mathbf{v}]$*. En conséquence, lorsque le test de Newton réussit, nous pouvons dire qu'il existe une unique boucle dans la trajectoire correspondant à* $[\mathbf{t}]$*. Et cela même s'il existe un nombre infini de trajectoires et donc de boucles possibles.*

La difficulté réside ici dans le choix d'une boîte $[\mathbf{t}]$ qui permettra la validation de ce test. En effet, lorsqu'une boucle est présente dans la trajectoire incertaine d'un robot, l'algorithme ne nous rendra pas une unique boîte correspondant à cette boucle mais bien un sous pavage, partie de $\mathbb{T}^+$ qui contiendra toutes les coordonnées $[t_1] \times [t_2]$ qui satisfont l'équation 3.2.3 pour chaque boucle. La simulation présentée en section 3.2.3 illustre ce phénomène. Si on voulait montrer que la trajectoire de la figure 3.3(a) possède une unique boucle, il faudrait exécuter un test d'unicité sur le sous pavage représentant la boucle illustré par la figure 3.4. Comme le test de Newton prend en entrée une boîte et non un sous pavage, il convient d'exécuter ce test sur une boite qui contient l'union intervalle de ce sous pavage et non sur chacune des boîtes qui le composent.

Le choix d'une boîte à tester par l'opérateur de Newton est donc primordial pour obtenir la garantie d'existence et unicité d'une boucle dans la trajectoire. Comprenons bien qu'il est inutile d'effectuer le test de Newton sur chacune des boîtes composant l'ensemble $\mathbb{T}^+$. En effet, le résultat de ce test serait au mieux une boîte qui contiendrait le sous pavage auquel appartient la boîte testée (voir exemple 3.3.1). Si l'on se rapporte à la méthode de Newton par intervalle pour une fonction incertaine, on peut voir ce phénomène comme le fait de calculer l'opérateur de Newton sur un intervalle plus petit que le plus petit résultat que cet opérateur pourrait rendre. Il est donc normal qu'aucune contraction ne puisse être effectuée et donc qu'aucune garantie d'existence et unicité ne puisse être

apportée.

**Exemple 3.3.1.** *Dans cet exemple, une fonction incertaine définie par une fonction paramétrée est encadrée par un tube dont les bornes sont des fonctions linéaires. Ce tube possède une racine en* $[t] = [1; 4]$.

$$f(t, \mathbf{p}) = p_1 t + p_2, \tag{3.3.7}$$

*avec* $p_1 \in [1; 2]$ *et* $p_2 \in [-4; -2]$. *La dérivée de ce tube est donc bornée dans l'intervalle* $[1, 2]$.

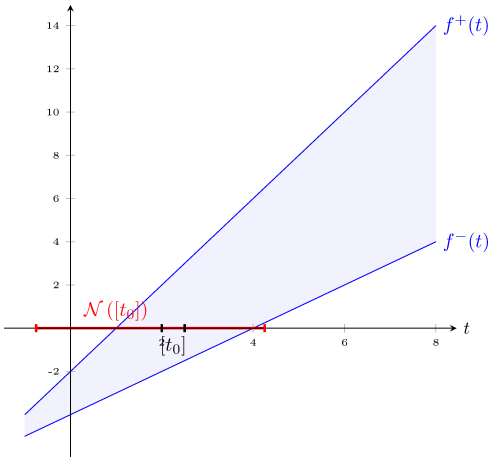



FIGURE 3.5 – Choix d'une boîte de petite taille en entrée du test de Newton.

*Si on applique le test de Newton sur un intervalle contenu dans la solution recherchée, le résultat rendu ne peut pas être de taille inférieur à cette solution. Ici, avec* $[t_0] = [2; 2.5]$, *nous obtenons*

$$\begin{aligned} \mathcal{N}([t_0]) &= \hat{t} - [f]\left(\hat{t}\right) / [f']([t]), \\ \mathcal{N}([t_0]) &= 2.25 - [f](2.25) / [1; 2], \\ \mathcal{N}([t_0]) &= [-0.7501, 4.2501] \end{aligned}$$

Plusieurs alternatives sont possibles pour contrer ce phénomène. Nous allons présenter dans un premier temps la méthode de l'$\mathcal{E}$-inflation, puis nous verrons comment l'encadrement des sous pavages représentant une boucle peut être utilisé et enfin, nous présenterons la méthode dites de $\mathcal{N}$-inflation qui nous permet de rechercher automatiquement une boîte à tester par l'opérateur de Newton.

### 3.3.2 $\mathcal{E}$-inflation

La méthode de l'$\mathcal{E}$-inflation est une heuristique qui consiste à augmenter la taille d'un intervalle d'un coefficient $\mathcal{E}$. La définition originale de l'$\mathcal{E}$-inflation nous vient de S.M. Rump [Rump 1980] :

**Définition 3.3.1.** *Pour un intervalle réel* $[x]$,

$$[x] \circ \mathcal{E} = \begin{cases} [x] + width([x]) \cdot [-\mathcal{E}, +\mathcal{E}] & si\ width([x]) \neq 0 \\ [x] + [-\eta, +\eta] & sinon \end{cases}, \tag{3.3.8}$$

où $\eta$ représente le plus petit nombre représentable en machine et $\mathcal{E}$ le coefficient d'inflation. Lorsqu'il s'agit d'intervalles de $\mathbb{R}^n$ et non d'intervalles réels, l'$\mathcal{E}$-inflation est appliquée à chacun des intervalles qui composent la boîte.

Pour chacune des boîtes composant l'ensemble $\mathbb{T}^+$, nous avons appliqué la méthode de l'$\mathcal{E}$-inflation avant de tester la boîte résultante par l'opérateur de Newton. Lorsque le coefficient $\mathcal{E}$ est bien choisi, nous nous retrouvons à tester des boîtes d'un sous pavage représentant une boucle qui sont plus grandes que l'union intervalle des boîtes composant ce sous pavage. Le test de Newton peut alors rendre un résultat positif et la boîte testée sera ajoutée à une liste qui contient toutes les boites $[\mathbf{t}]$ qui satisfont le test de Newton. Toutefois, les résultats apportés ne sont pas satisfaisants puisque certaines boîtes, pour lesquelles le test a été validé, se chevauchent créant ainsi une détection de fermeture boucle redondante. Cette redondance peut, bien-sûr, être éliminée en remplaçant les boîtes qui se chevauchent par leur intersection. On obtiendrait alors la liste$\mathbb{T}^{\mathcal{N}}$ de boîtes pour lesquelles une unique boucle existe dans la trajectoire. Le nombre d'éléments de cette liste représente alors un nombre minimal de boucles existantes, de manière garantie, dans la trajectoire. Autrement dit, une borne inférieure au nombre de boucles existantes dans la trajectoire du robot.

**Remarque 3.3.2.** *Chaque sous pavage de l'ensemble* $\mathbb{T}^+$ *est composé de boîtes de taille variable. Par exemple, l'ensemble* $\mathbb{T}^+$ *de la simulation présentée en section 3.2.3 contient* 1510 *boîtes et leur taille varie de* 1 *à* 103 *unités de temps. On comprend alors que le choix du coefficient* $\mathcal{E}$ *devient problématique et ne peut pas être systématisé.*

### 3.3.3 Recherche de boîtes englobantes

Une méthode automatique pour trouver une boîte capable de retourner un résultat exploitable par le test de Newton est de prendre, en entrée du test de Newton, l'union intervalle de chacun des sous pavages représentant une boucle. Pour l'exemple présentée en section 3.2.3, cette union intervalle correspond à l'union intervalle de toutes les boîtes qui composent l'ensemble $\mathbb{T}^+$ puisque la trajectoire ne comporte qu'une seule boucle (rappelons que l'ensemble $\mathbb{T}^+$ de cet exemple est composé de 1510 intervalles de $\mathbb{IR}^2$). On peut observer la boite englobante correspondant à ce sous pavage sur la figure 3.6. Lorsque la trajectoire est plus complexe, il faut rechercher les boîtes connexes avant d'effectuer cette union intervalle. Cette recherche peut prendre beaucoup de temps lorsque la trajectoire est composée de multiples boucles.

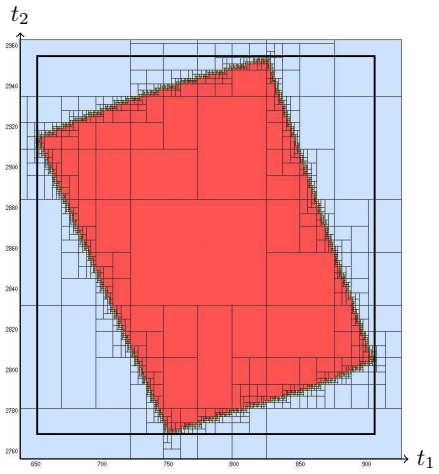



FIGURE 3.6 – Représentation de la boîte englobante (en noir) de l'ensemble $\mathbb{T}^+$ pour l'exemple de la section 3.2.3 (page 90).

**Remarque 3.3.3.** *Il faut tout de même appliquer une $\mathcal{E}$-inflation aux boîtes englobantes créées par ce calcul d'unions puisque ces dernières partagent leurs bornes avec certaines boîtes de l'ensemble $\mathbb{T}^+$. La boîte englobante d'un sous pavage représentant une boucle est donc la plus petite boîte que pourrait retourner l'opérateur de Newton et qui garantirait l'existence d'une boucle pour ce sous pavage.*

### 3.3.4 $\mathcal{N}$-inflation

Nous nous servons de l'opérateur de Newton afin de garantir l'existence et l'unicité. Mais l'origine de cet opérateur est la recherche d'un zéro. Si on applique l'opérateur de Newton à une petite boîte ou un point proche d'un zéro d'une fonction, le résultat, s'il existe et même s'il n'est pas inclus dans lui même, est censé nous rapprocher du zéro en question. Cette stratégie, que nous appellerons $\mathcal{N}$-inflation (lire newton-inflation) consiste à utiliser les capacités de recherche d'un zéro de manière itérative avant d'utiliser les propriétés d'existence et unicité de cet opérateur. Cette méthode est fortement inspirée de [Alexandre Goldsztejn 2008] où l'auteur propose une méthode de recherche similaire basée sur un autre opérateur d'existence.

On recherche à prouver l'existence d'une racine pour une fonction incertaine représentée par une fonction paramétrée : on connaît un $\mathbf{x}$ contenant ces racines tel que :

$$\mathbf{f}(\mathbf{x}, \mathbf{p}) = 0 \text{ avec } \mathbf{p} \in [\mathbf{p}]. \tag{3.3.9}$$

La fonction $\mathbf{f}$ est continument dérivable par rapport à $\mathbf{x}$, $\mathbf{f}$ et $\mathbf{x}$ sont de dimensions $n$, et $[\mathbf{p}]$ est une boîte de petite taille en dimension $m$.

Nous proposons la séquence :

$$[\mathbf{x_{k+1}}] = \mathcal{N}([\mathbf{x_k}] + [\varepsilon], [\mathbf{p}]), \tag{3.3.10}$$

où $\mathcal{N}$ est l'opérateur de newton paramétré et $[\varepsilon]$ est un intervalle de petite taille contenant 0 (il s'agit en fait d'une $\mathcal{E}$-inflation nécéssaire puisque on attend une inclusion stricte pour garantir l'unicité par le test de Newton). Cette séquence est initialisée avec $[\mathbf{x_0}]$ un interval de petite taille de préférence proche de la solution. Si $[\mathbf{x_{k+1}}] \subset [\mathbf{x_k}] + [\varepsilon]$, on obtient :

$$\forall \mathbf{p} \in [\mathbf{p}], \exists! \mathbf{x} \in [\mathbf{x}] \mid \mathbf{f}(\mathbf{x}, \mathbf{p}) = \mathbf{0}. \tag{3.3.11}$$

Ce mécanisme est facilement compréhensible si on applique cette méthode à un tube composé de fonctions linéaires, comme le montre l'exemple suivant.

**Exemple 3.3.2.** *Reprenons l'exemple 3.3.1 présenté en page 94 et appliquons la $\mathcal{N}$-inflation : la boîte initiale $[t_0]$ est une boîte de petite taille, de préférence proche du zéro à trouver. Choisissons $[t_0] = [4.5, 5]$ et définissons $\hat{t}$ comme le milieu de*

*l'intervalle* $[t]$ *à évaluer. L'application de l'opérateur de Newton nous donne :*

$$\begin{aligned}\mathcal{N}([t_0]) &= \hat{t} - [f]\left(\hat{t}\right) / [f']([t]),\\ \mathcal{N}([4.5,5]) &= 4.75 - [0.7499;7.5] / [1;2],\\ \mathcal{N}([4.5,5]) &= [-2.7501, 4.3751].\end{aligned}$$

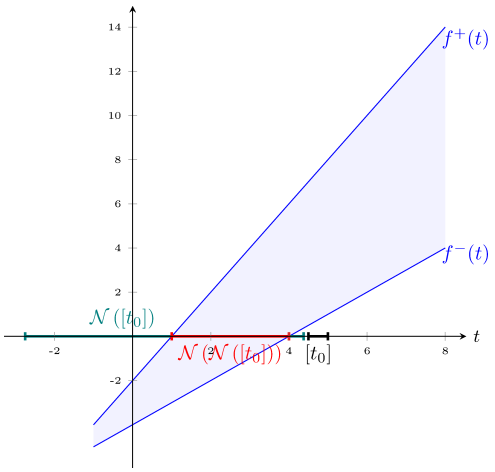



FIGURE 3.7 – Exemple d'utilisation de la $\mathcal{N}$-inflation.

*Comme* $\mathcal{N}([t_0]) \not\subset [t_0]$*, on ne peut rien prouver. Si on continue les itérations de la méthode de Newton pour la recherche de racines, on trouve :*

$$\mathcal{N}(\mathcal{N}([f],[t_0])) = [1,4].$$

*Or* $[1,4] \subset [-2.7501, 4.3751]$ *ou* $\mathcal{N}(\mathcal{N}([t_0])) \subset \mathcal{N}([t_0])$*. Donc,*

$$\forall f \in [f], \exists! t \in [1,4] \quad | \quad f(t) = 0.$$

L'exemple simple présenté ci-dessus utilise des fonctions linéaires pour les bornes du tube $[f](t)$ mais comme la méthode de Newton cherche la linéarisation de la fonction autour d'un point, il est aisé de l'étendre aux fonctions non linéaires à valeurs dans $\mathbb{R}^n$.

**Remarque 3.3.4.** *Comme pour la méthode de recherche de boîtes englobantes, il convient d'effectuer une* $\mathcal{E}$*-inflation à chaque itération de la méthode de Newton afin de garantir l'inclusion stricte du résultat du test de Newton. L'utilisation*

*de cette sécurité est toutefois moins indispensable puisque l'opérateur de Newton ne peut pas rendre de résultat plus petit que l'intervalle minimal contenant les racines des fonctions $f \in [f]$.*

## 3.4 Application sur données réelles

### 3.4.1 Présentation de l'expérimentation

Un robot sous marin évolue sur un plan (profondeur constante) en respectant les équations d'état suivantes :

$$\dot{\mathbf{p}} = \mathbf{v} = \begin{pmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{pmatrix} . \begin{pmatrix} u_x \\ u_y \end{pmatrix},$$

où $(u_x, u_y)^{\mathrm{T}}$ représente le vecteur vitesse du robot et $\psi$ le cap du robot. $\mathbf{p}(0)$ correspond à la position initiale du robot dans son système de coordonnées. Les seules mesures qui nous intéressent sont $u_x, u_y, \psi$ et nous prendrons en compte le fait que des tubes $[u_x], [u_y], [\psi]$ sont disponibles. L'arithmétique par intervalles nous permet de trouver un tube $[\mathbf{v}]$ pour la vitesse $\mathbf{v}$ à partir de ces données. Nous utilisons le jeu de données recueilli pour une expérience décrite dans [Jaulin 2009] effectuée par le robot chasseur de mines *Redermor* du GESMA [1] (une photo de ce robot est présentée sur la figure 1.1). Pendant cette mission, le robot effectue une surveillance de zone sur un plan horizontal à une profondeur de 20 mètres. La mission a été effectuée en baie de Douarnenez (Bretagne, France) et dure environ deux heures. La vitesse réelle du robot est connue pour satisfaire la contrainte :

$$\mathbf{u} \in \begin{pmatrix} [0.996\widetilde{u}_x - 0.004, 1.004\widetilde{u}_x + 0.004] \\ [0.996\widetilde{u}_y - 0.004, 1.004\widetilde{u}_y + 0.004] \end{pmatrix}, \tag{3.4.1}$$

où $(\widetilde{u}_x, \widetilde{u}_y)$ sont les vitesses mesurées par le DVL (*Workhorse Navigator Doppler Velocity Log*). On note $\tilde{\psi}$ le cap du robot en radians mesuré par le gyrocompas (*Octans III de IXSEA*), le cap réel du robot satisfait la condition $\psi \in \tilde{\psi} \pm 0.00527$.

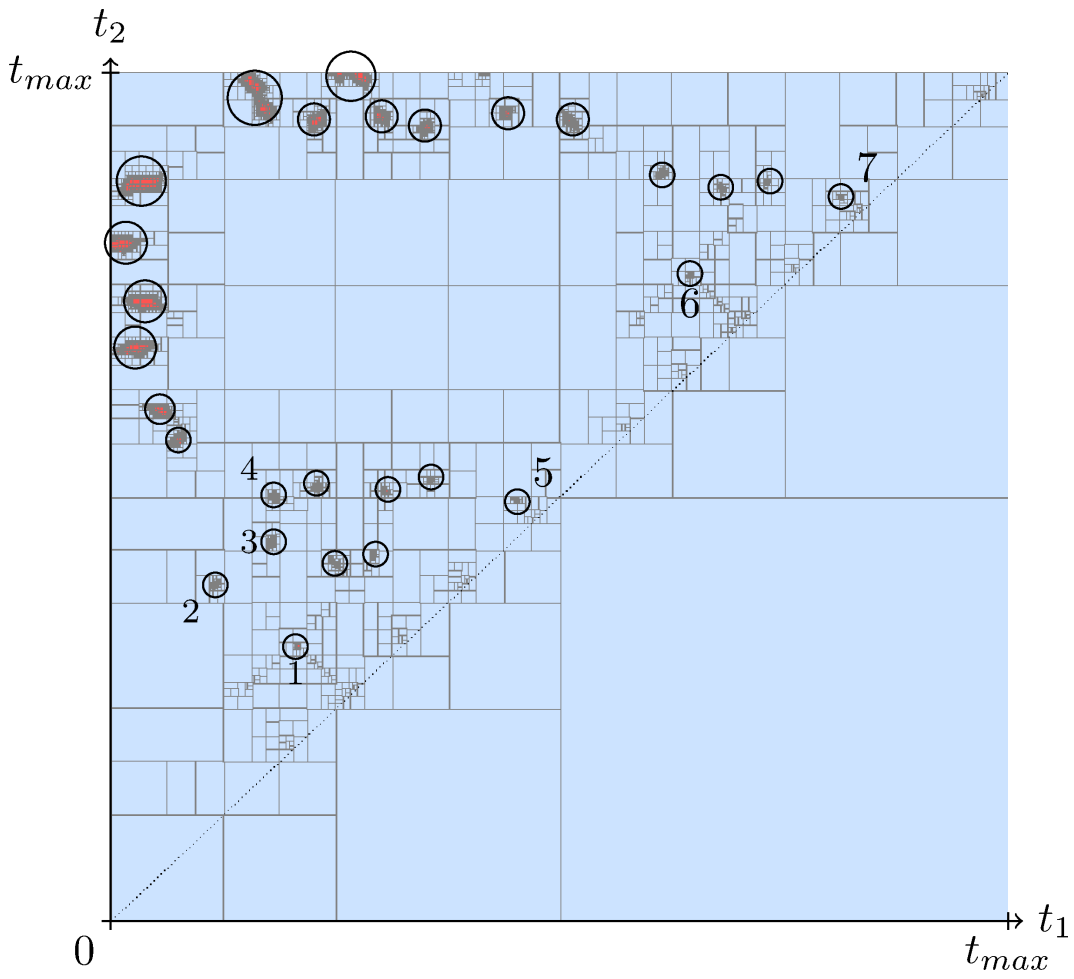
1. Groupe d'Étude Sous-Marine de l'Atlantique

### 3.4.2 Résultats

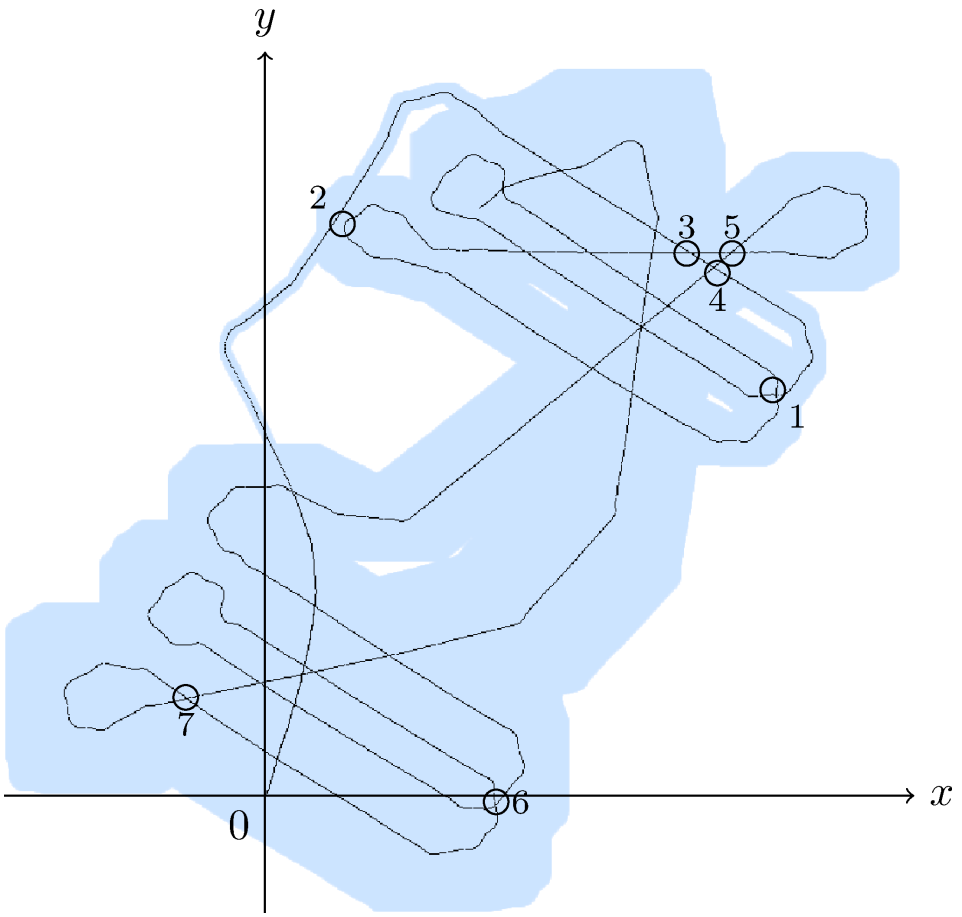
Comme nous possédons un tube $[\mathbf{v}]$ pour la vitesse du sous marin, nous sommes capables de produire un tube $[\mathbf{p}](t) = \int_0^t [\mathbf{v}](\tau)\, d(\tau)$ par une intégration intervalle. Ce tube pour la trajectoire du sous marin est représenté sur la figure 3.8(b) avec $\mathbf{p}(0) = 0$.

L'hypothèse de départ, *i.e.* seules des données proprioceptives sont disponibles, implique le fait que la taille des boîtes représentant la position augmente au fur et à mesure du temps. La courbe noire sur ces figures est le centre de chaque boîte représentant une position à un instant donné. Cette trajectoire ne représente pas la trajectoire réelle mais une trajectoire possible du robot. L'algorithme LOOP (algorithme 2) a produit le t-plan de la figure 3.8(a) en moins de deux minutes pour une mission de deux heures sur un ordinateur portable classique (la précision est fixée à une unité de temps par rapport à l'acquisition des données (100ms)).

On peut voir sur cette figure la ligne pointillée qui correspond à la droite $t_1 = t_2$. Toutes les boucles les plus proches de cette droite correspondent à des boucles simples (les boucles numérotées 1, 5, 6 et 7 sur la figure 3.8(a)). Ces boucles simples sont détectées assez précisément même quand $t_1$ et $t_2$ sont grands. Les cercles présents sur la figure 3.8 ont été ajoutés afin d'aider à la compréhension : sur le t-plan (figure 3.8(a)), ces cercles nous aident à repérer les 28 sous pavages correspondants aux 28 boucles dans la trajectoire (figure 3.8(b)) ; certains de ces sous pavages sont numérotés sur le t-plan et permettent de visualiser sur la trajectoire une estimation de l'emplacement de la boucle correspondante. Toutes les boucles n'ont pas été numérotées afin de ne pas encombrer la figure, le lecteur est invité à essayer le programme de démonstration qui a produit ces images à l'adresse : https://sites.google.com/site/clementaubry/research/loop-detection.

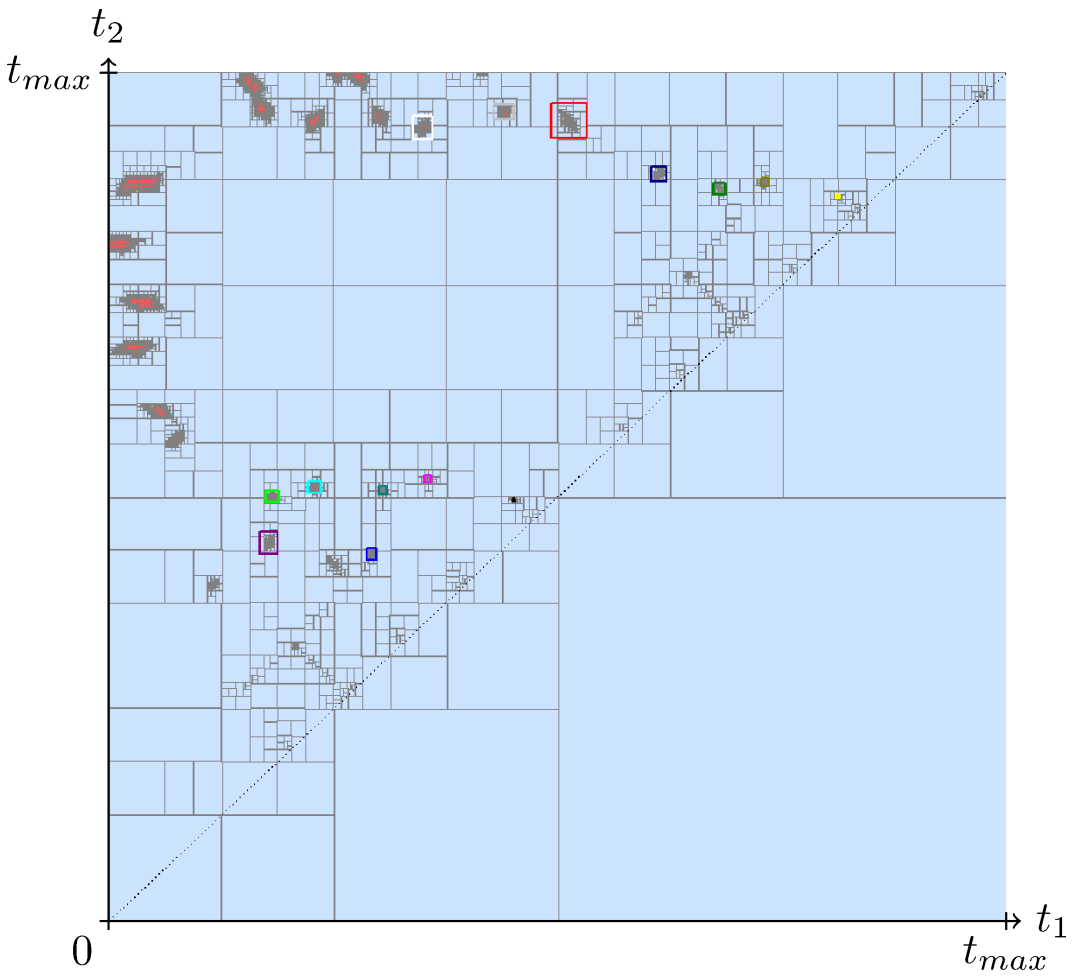



(a) T-plane associé à la mission.

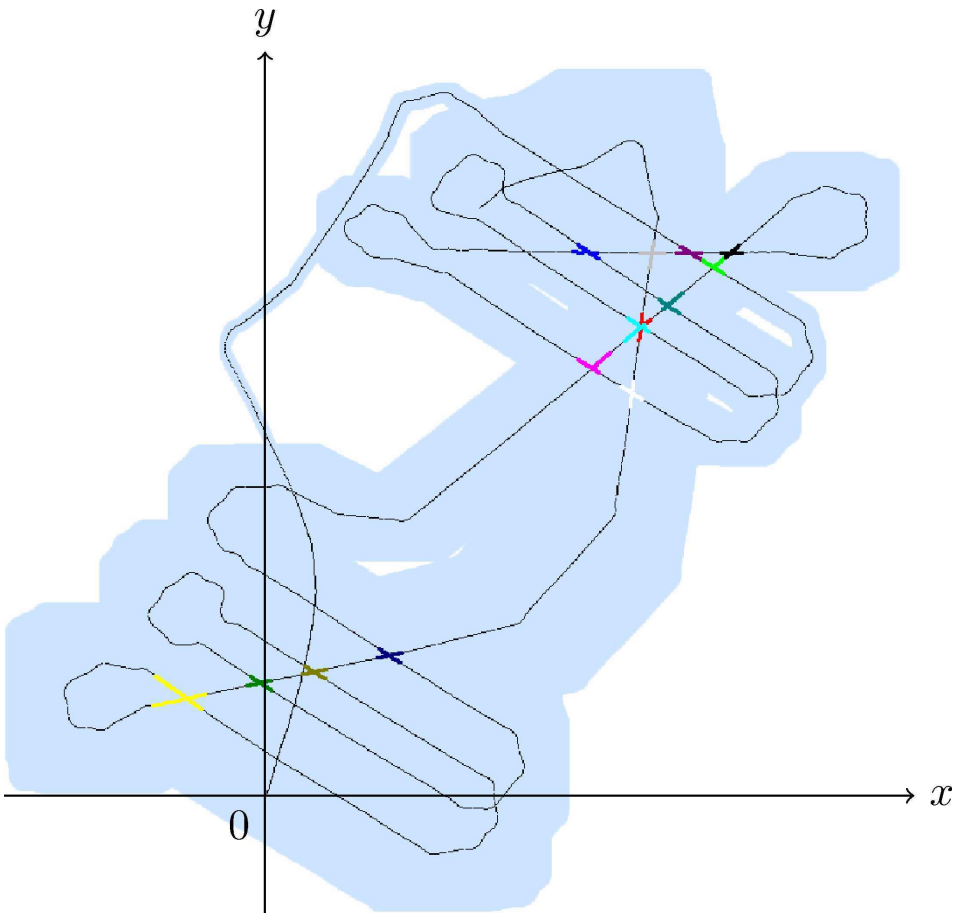



(b) Trajectoire du robot.

FIGURE 3.8 – Résultats de l'algorithme LOOP sur les données du Redermor. A gauche le t-plan, les cercles contiennent des sous pavages qui indique un événement de fermeture de boucle. La numérotation de quelques sous pavages permet de visualiser sur la figure de droite la boucle correspondante. Le lecteur est invité à essayer le programme de démonstration qui a produit ces images à l'adresse : https://sites.google.com/site/clementaubry/research/loop-detection



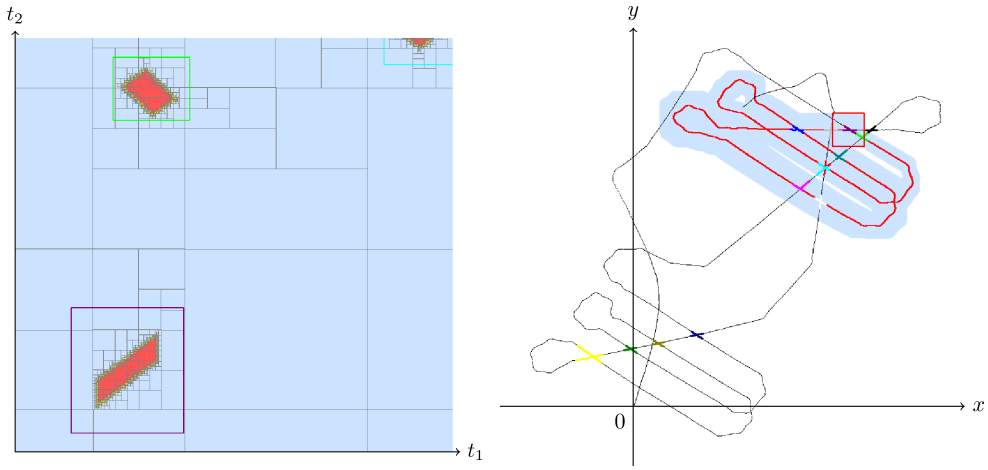

(a) T-plane associé à la mission.




(b) Trajectoire du robot.

FIGURE 3.9 – Résultats du test de Newton. A gauche le t-plan, les boîtes de couleur ont passés avec succès le test de Newton. Les intersections correspondantes ont été mises en couleur sur l'estimation de la trajectoire de la figure de droite. Le lecteur est invité à essayer le programme de démonstration qui a produit ces images à l'adresse : https://sites.google.com/site/clementaubry/research/loop-detection

Le test de Newton, couplé à la $\mathcal{N}$-inflation, a prouvé l'existence de 14 boucles dans la trajectoire du robot parmi les 28 qui la composent potentiellement. Les boîtes qui ont permis de prouver l'existence et l'unicité sont affichées en couleurs sur le t-plan de la figure 3.9(a). Pour se faire une idée de la place de ces boucles dans la trajectoire réelle de l'AUV [2], les intersections correspondantes sont affichées par les mêmes couleurs sur l'estimation de la trajectoire, figure 3.9(b). Ce résultat du test de Newton ne veut pas dire que la trajectoire ne possède que 14 boucles mais bien qu'il existe au moins 14 boucles dans la trajectoire du robot. De plus, la plupart des 14 boîtes qui ont passé avec succès le test de Newton correspondent à des boucles proches de la droite $t_1 = t_2$. La figure 3.10 présente un zoom sur le t-plan et la trajectoire correspondant à un résultat positif du test de Newton.




(a) (b)

FIGURE 3.10 – Zoom sur le t-plan (a). La trajectoire affichée sur la figure (b) correspond à la trajectoire entre deux instants, $t_1$ et $t_2$, choisis dans le sous pavage inférieur gauche de la figure 3.10(a). Cette trajectoire comporte donc une unique boucle et ce résultat est garanti par le test de Newton (boîte mauve encadrant le sous pavage en question).

## 3.5 Trajectoire de retour sans boucles

Ce chapitre nous a permis de formaliser et valider, à travers une expérience sur données réelles, le problème de détection de fermeture de boucles par mesures proprioceptives. Les concepts développés dans ce cadre, le *t-plan* et les *t-paires*

2. Autonomous Underwater Vehicle

nous servent à décrire ces événements de fermeture de boucle. Cette information, très souvent utilisée dans des algorithmes SLAM afin de réduire l'erreur de positionnement, peut aussi nous permettre de trouver un chemin de retour sans boucle pour le robot. Cette utilisation nécessite la qualification des boucles telle qu'elle a été abordée en section 3.1.2.

Cette qualification peut être effectuée graphiquement par l'intermédiaire du triangle englobant d'une boucle. Ce triangle est formé par les coordonnées d'une t-paire $\mathbf{t_i}$ et la droite $t_1 = t_2$. Si ce triangle contient une ou plusieurs t-paire, alors la boucle $\mathbf{t_i}$ englobe les boucles contenues dans ce triangle. Sur la figure 3.11, la boucle $(t_2, t_6)$ englobe la boucle $(t_3, t_4)$. De même, la boucle $(t_1, t_5)$ englobe la boucle $(t_3, t_4)$. Les boucles $(t_2, t_6)$ et $(t_1, t_5)$ sont donc des boucles englobantes alors que la boucle $(t_3, t_4)$ est une boucle simple.

On peut aussi définir la notion de boucles imbriquées. Si, pour deux boucles $\mathbf{t_a} = (t_{1,a}, t_{2,a})$ et $\mathbf{t_b} = (t_{1,b}, t_{2,b})$ on a $[t_{1,a}, t_{2,a}] \cap [t_{1,b}, t_{2,b}] \neq \emptyset$, alors les boucles sont imbriquées. Sur le t-plan, les boucles dont les triangles englobant s'intersectent sont des boucles imbriquées. Sur la figure 3.11, les boucles $(t_2, t_6)$ et $(t_1, t_5)$ sont imbriquées.

La qualification des boucles (simple, englobante, imbriquée) peut nous permettre de trouver rapidement un chemin de retour sans boucle pour notre robot. C'est le problème mythologique du fil d'Ariane. La suppression des boucles imbriquées implique la suppression de toutes les boucles qu'elles encadrent, les boucles simples restantes peuvent aussi être supprimées. De cette manière, on trouve le plus court chemin sans boucle permettant de revenir à la position d'origine empruntant un trajet connu et donc très probablement libre de toute collision avec les objets fixes de l'environnement.

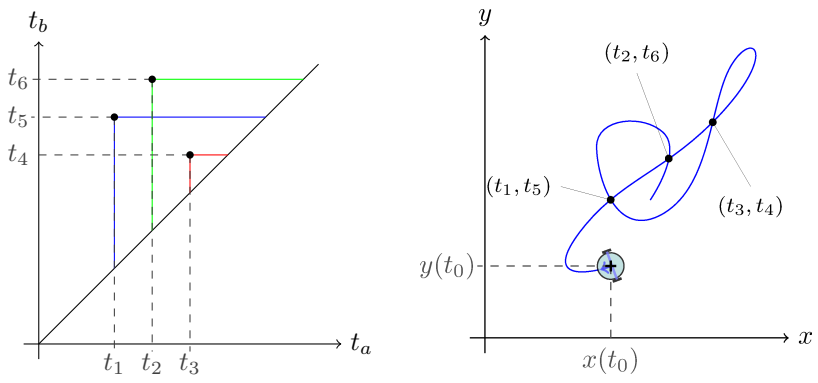



FIGURE 3.11 – T-plane (a) correspondant à une trajectoire (b) comportant plusieurs boucles.

## 3.6 Conclusion

L'algorithme Loop (algorithme 2) est capable, à partir de données proprioceptives, de trouver des coordonnées temporelles correspondant à des temps pour lesquels le robot a effectué une boucle. Contrairement aux méthodes présentées dans le chapitre 1, cette information n'est pas utilisée ici dans un algorithme SLAM ou de localisation. En effet, ne connaissant que le tube pour la vitesse, il nous est impossible de répercuter cette information de fermeture de boucle sur la trajectoire. Toutefois, la légitimité des informations apportées par notre algorithme ne peut être remise en question grâce aux qualités d'existence et unicité apportées par le test de Newton, ce qui n'est pas le cas des autres méthodes de détection de fermeture de boucle. Le couplage de notre méthode à un algorithme SLAM est tout à fait envisageable car le temps de calcul est très faible (deux minutes de calcul pour une mission de deux heures). Ce qui nous laisse penser que l'intégration est possible. Dans ce cadre, on comprend aisément que lorsque l'algorithme SLAM reçoit une information de fermeture de boucle, il est possible de contracter le tube représentant la position et même de répercuter cette information sur le tube pour la vitesse. Alors la réduction de cette erreur de vitesse nous permettrait de produire des sous pavages plus petits pour les boucles suivantes. Cette réduction de l'erreur de vitesse implique aussi une précision plus importante sur la Jacobienne utilisée dans le test de Newton et permettrait de caractériser l'existence et l'unicité d'un nombre de boucles plus important.

# Conclusion générale

Le travail présenté dans ce mémoire de thèse propose une méthode de détection de fermeture de boucle dans la trajectoire d'un robot mobile. Cette méthode est automatisée et rapide. La détection est basée sur l'utilisation des temps pour lesquels une donnée de vitesse est disponible. En effectuant de simples tests sur l'intégrale d'un tube représentant la vitesse et son erreur d'approximation, nous sommes capables de produire, pour chaque boucle de la trajectoire, un encadrement de deux temps entre lesquels le robot a effectué une boucle dans sa trajectoire. L'utilisation de l'opérateur de Newton et de l'analyse par intervalles confère à ces résultats un caractère garanti quant à l'existence et l'unicité d'une solution dans un grand nombre de cas. De ce fait, un algorithme SLAM peut prendre en compte cette information de fermeture de boucle sans aucun doute possible.

Les méthodes existantes pour la détection de fermeture de boucle sont basées sur l'utilisation de données extéroceptives alors que notre méthode se base sur l'observation de l'évolution du robot à travers des données proprioceptives, permise par le domaine d'application qui est le milieu sous-marin, dans lequel ces données sont souvent bien plus fiables que les données extéroceptives.

Nous avons vu dans l'état de l'art que les méthodes existantes sont bien souvent directement couplées à des algorithmes SLAM et dépendent donc fortement de l'estimation de la position effectuée par ces derniers. La méthode de détection de boucle présentée dans cette thèse, découplée de tout traitement antérieur, apporte donc une certaine robustesse puisqu'elle utilise directement les données issues de mesures et la modélisation du robot.

L'outil statistique utilisé dans les autres méthodes admet un pourcentage de fausses détections alors que la nôtre se base sur des outils garantis et tous les résultats qu'elle fournit - les boucles pour lesquelles on a prouvé l'existence et l'unicité - sont directement exploitables. L'application sur données réelles prouve l'existence de quatorze boucles dans la trajectoire alors que l'on peut en compter vingt huit sur l'estimation de la trajectoire.

Ce résultat pourrait toutefois, à mon sens, être amélioré de différentes manières. Parmi celles ci, l'intégration de notre algorithme dans un système complet, c'est-à-dire un robot d'exploration utilisant un algorithme SLAM, permettrait un échange d'informations bénéfique pour les deux applications. En effet, la détec-

tion de boucle apporterait une meilleure précision au positionnement et ce dernier pourrait éventuellement permettre une contraction du tube représentant la vitesse utilisé par l'algorithme de détection. Si le tube pour la vitesse est plus précis, la détection de boucle produira des sous pavages, représentant une intersection, plus fins et améliorerait probablement les résultats du test de Newton. Bien que l'algorithme soit rapide, son implémentation dans un système robotisé n'est pas triviale et devient elle-même une perspective intéressante. L'utilisation de l'analyse par intervalles n'est pas, à mon sens, une barrière quant à l'implémentation de notre méthode en complément d'un algorithme SLAM utilisant un autre outil tel que les distributions de probabilité puisque ces deux résultats sont découplés. Il est donc tout à fait envisageable de fournir à un tel algorithme SLAM deux temps pour lesquels il devra retrouver des observations similaires dans la trajectoire.

Il est par ailleurs intéressant de noter que notre algorithme a déjà trouvé une application concrète. En effet, la société ECA robotics l'utilise afin d'aider au "mosaïcing". Dans cette application, un robot sous-marin équipé d'un DVL et d'un sonar latéral utilise la détection de fermeture de boucle afin de sélectionner les images du sonar latéral sur lesquelles un recouvrement peut être retrouvé. Ainsi, le "mosaïcing" n'est effectué que sur les images correspondant aux temps de la trajectoire pour lesquels le robot a fait une boucle et le temps de traitement global est diminué.

Il est également possible d'implémenter notre algorithme en utilisant des contracteurs, ce qui permettrait d'améliorer significativement les temps de calcul. Dans ce cas, la réduction du temps de traitement devrait permettre d'obtenir, à terme, une version temps réel de l'algorithme qui pourrait alors être implémentée en parallèle d'une méthode SLAM temps réel.

On pourrait aussi envisager un travail plus approfondi sur les tests d'existence et unicité. Des méthodes de résolution ensemblistes existent pour des problèmes d'inversion de matrice intervalle. Celles testées au cours de ces travaux n'ont pas apporté d'amélioration significative. C'est pourquoi nous avons choisi de ne pas les aborder ici. Toutefois, d'autres théorèmes mathématiques, dont certains n'ont pas encore été "intervallisés", devraient permettre d'apporter la garantie d'existence et unicité au même titre que l'opérateur de Newton. Les travaux récents de Goldsztejn à propos d'un nouvel opérateur d'existence basé sur le test de Hansen-Segpunta et utilisant une méthode de recherche de solution similaire à la $\mathcal{N}$-inflation nous laisse à penser que cette piste peut être approfondie. Cette méthode mérite d'être expérimentée pour notre problématique, même si elle ne fournit pas la propriété d'unicité apportée par le test de Newton.

Les travaux présentés dans ce manuscrit ont été appliqués au milieu sous-marin mais ils pourraient être utilement ouverts aux milieux terrestre et spatial, ce qui suppose toutefois de pouvoir disposer d'un capteur de vitesse avec une précision comparable à celle utilisée dans l'expérience présentée dans cette thèse. A titre d'exemple, les centrales inertielles présentes sur le marché ne sont, pour l'instant, pas utilisables dans notre application mais on peut imaginer que les

recherches technologiques dans ce domaine produiront dans un futur proche des capteurs qui permettront d'obtenir le niveau de précision requis.

Les travaux réalisés dans le cadre de cette thèse ont permis une réelle avancée dans la connaissance de la trajectoire d'un robot mobile évoluant en milieu sous-marin. Ils ouvrent de nouvelles perspectives qui méritent d'être approfondies, comme nous l'avons évoqué ci-dessus, afin de permettre à terme de disposer de robots mobiles capables d'évoluer en milieu marin, terrestre ou spatial avec une capacité de localisation toujours plus fiable.

# Liste des publications et communications


Aubry, C. (2013). “Improving Newton Existence Test”. SWIM - Small Workshop on Interval Methods. URL : http://www.ensta-bretagne.fr/swim13/index.php/program/.

Aubry, C., R. Desmare et L. Jaulin (2011). “Loop detection in a mobile robot trajectory using interval analysis”. SWIM - Small Workshop on Interval Methods. URL : https://agora.bourges.univ-orleans.fr/ramdani/swim2011/.

Aubry, C., R. Desmare et L. Jaulin (2013). “Loop detection of mobile robots using interval analysis”. In : *Automatica* 49.2, p. 463–470. DOI : 10.1016/j.automatica.2012.11.009.

Aubry, C., R. Desmare et L. Jaulin (2014). “Kernel characterization of an interval function”. In : *Mathematics in Computer Science*. accepted. DOI : 10.1007/s11786-014-0206-9.

Aubry, C. et L. Jaulin (2012). “Existence and uniqueness tests to solve image evaluation problem”. SWIM - Small Workshop on Interval Methods. URL : http://hs.informatik.uni-oldenburg.de/swim2012/.

# Annexe A

# Fonctionnement d'un Sonar à effet Doppler

Un Sonar à effet Doppler est un capteur qui mesure une vitesse. Pour cela, le sonar émet un train d'ondes de fréquence $f$ appelé *ping* puis écoute l'écho qui lui revient, après une réflexion du train d'ondes sur un objet. Lorsque cet objet est en mouvement, la fréquence de l'écho est décalée. C'est cet effet Doppler que le sonar utilise en mesurant le décalage de fréquence pour le transformer en vitesse. Cette transformation s'effectue en doublant le décalage induit par l'effet Doppler (aller et retour du train d'ondes) :

$$\Delta f = 2fv/C, \tag{A.0.1}$$

avec $\Delta f$ le décalage en fréquence entre l'émission et la réception, $C$ la célérité du milieu et $v$ la vitesse de l'élément réflecteur du train d'ondes dans l'axe du transducteur. En utilisant ce principe avec deux transducteurs décalés en orientation, une vitesse est obtenue dans un plan. La figure A.1, inspirée de [LPO p.d.], illustre ce principe : si on se place dans le plan vertical $x0z$ qui passe par le centre de deux transducteurs opposés, les transducteurs mesurent respectivement un signal affecté par l'effet Doppler, les deux signaux permettent de connaître la composante $V_{mx}$ de la vitesse du courant dans l'axe du capteur. On obtient alors $V_x$ la composante de cette vitesse selon l'axe $0x$.

La plupart des instruments commercialisés utilisent quatre transducteurs : deux faisceaux opposés permettent de calculer une composante horizontale et la composante verticale de la vitesse ; les deux autres faisceaux de transducteur mesurent simultanément l'autre composante horizontale et la même composante verticale. La comparaison des mesures des deux composantes verticales donne une erreur, donnant une information sur la validité des données. Cette validation peut être importante dans un milieu turbulent (par exemple lorsque les vitesses

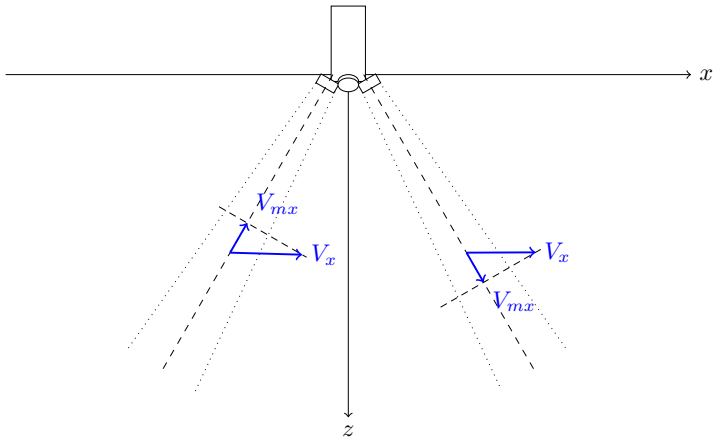



FIGURE A.1 – Obtention de la vitesse dans un plan à partir de deux transducteurs d'un Sonar à effet Doppler.

des particules réfléchissantes ne sont pas homogènes).

Les Sonar à effet Doppler sont divisés en deux catégories : les ADCP (*Acoustic Doppler Current Profiler*) qui sont fixés au sol et permettent de mesurer la vitesse des particules dans un fluide (courantométrie) et les DVL (*Doppler Velocity Log*) qui sont fixés sur un mobile (robot, bateau, etc) et permettent de mesurer la vitesse de ce mobile par rapport au fond si celui ci est à portée du sonar.

# Bibliographie


Alefeld, G. et J. Herzberger (1983). *Introduction to Interval Computation*. New York : Academic Press (cité pages 48, 76).

Andreasson, Henrik, André Treptow et Tom Duckett (2005). "Localization for mobile robots using panoramic vision, local features and particle filter". In : *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, p. 3348–3353 (cité page 15).

Angeli, Adrien (2008). "Détection visuelle de fermeture de boucle et applications à la localisation et cartographie simultanées". Thèse de doct. Université Pierre et Marie Curie-Paris VI (cité pages 33, 35).

Aubry, C., R. Desmare et L. Jaulin (2013). "Loop detection of mobile robots using interval analysis". In : *Automatica* 49.2, p. 463–470. DOI : 10.1016/j.automatica.2012.11.009 (cité pages 57, 88).

Aubry, C., R. Desmare et L. Jaulin (2014). "Kernel characterization of an interval function". In : *Mathematics in Computer Science*. accepted. DOI : 10.1007/s11786-014-0206-9 (cité page 66).

Bailey, T et H.F Durrant-Whyte (2006). "Simultaneous localization and mapping (SLAM) : Part II". In : *Robotics & Automation Magazine, IEEE* 13.3, p. 108–117 (cité page 22).

Barfoot, Timothy D (2005). "Online visual motion estimation using FastSLAM with SIFT features". In : *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, p. 579–585 (cité page 15).

Bars, Fabrice Le (2011). "Analyse par intervalles pour la localisation et la cartographie simultanées ; Application à la robotique sous-marine". PhD dissertation. Brest, France : Université de Bretagne Occidentale (cité pages 11, 17, 22).

Bay, Herbert, Tinne Tuytelaars et Luc Van Gool (2006). “Surf : Speeded up robust features”. In : *Computer Vision–ECCV 2006*. Springer, p. 404–417 (cité page 27).

Booij, Olaf et al. (2007). “Navigation using an appearance based topological map”. In : *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, p. 3927–3932 (cité page 33).

Borsuk, Karol (1933). “Drei Sätze über die n-dimensionale euklidische Sphäre”. In : *Fundamenta Mathematicae* 20.1, p. 177–190 (cité page 72).

Boyer, F. et al. (2006). “Robot anguille sous-marin en 3d”. In : *Techniques de l’Ingénieur* (cité page 8).

Braems, I., F. Berthier et al. (2001). “Guaranteed Estimation of Electrochemical Parameters by Set Inversion Using Interval Analysis”. In : *Journal of Electroanalytical Chemistry* 495.1, p. 1–9 (cité page 61).

Braems, I., L. Jaulin et al. (2001). “Guaranteed numerical alternatives to structural identifiability testing”. In : *In Proceedings of the 40th IEEE Conference on Decision and Control*. T. 4. Orlando, p. 3122–3127 (cité page 61).

Castellanos, J. A. et J.D. Tardós (1999). “Mobile Robot Localization and Map Building : A Multisensor Fusion Approach”. In : *Kluwer* (cité page 22).

Chong, Kok Seng et Lindsay Kleeman (1997). “Accurate odometry and error modelling for a mobile robot”. In : *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*. T. 4. IEEE, p. 2783–2788 (cité page 10).

Choset, Howie et Keiji Nagatani (2001). “Topological simultaneous localization and mapping (SLAM) : toward exact localization without explicit localization”. In : *Robotics and Automation, IEEE Transactions on* 17.2, p. 125–137 (cité page 19).

Clemente, Laura A et al. (2007). “Mapping Large Loops with a Single Hand-Held Camera.” In : *Robotics : Science and Systems* (cité pages 30, 32, 36).

Clérentin, A. et al. (2008). “Uncertainty and imprecision modeling for the mobile robot localization problem”. In : *Autonomous Robots* 24.3, p. 1573–7527 (cité page 16).

Csurka, Gabriella et al. (2004). “Visual categorization with bags of keypoints”. In : *Workshop on statistical learning in computer vision, ECCV*. T. 1. 1-22, p. 1–2 (cité page 35).

Cummins, Mark et Paul Newman (2008a). “Accelerated appearance-only SLAM”. In : *Robotics and automation, 2008. ICRA 2008. IEEE international conference on*. IEEE, p. 1828–1833 (cité page 32).

Cummins, Mark et Paul Newman (2008b). “FAB-MAP : Probabilistic localization and mapping in the space of appearance”. In : *The International Journal of Robotics Research* 27.6, p. 647–665 (cité pages 32, 35).

Davey, B. A. et H. A. Priestley (2002). *Introduction to Lattices and Order*. (ISBN 0521784514) : Cambridge University Press (cité page 44).

Davison, A.J. (2003). “Real-time simultaneous localisation and mapping with a single camera”. In : *International Conference on Computer Vision* (cité page 28).

Davison, Andrew J, Y González Cid et Nobuyuki Kita (2004). “Real-time 3D SLAM with wide-angle vision”. In : *Proc. IFAC/EURON Symp. Intelligent Autonomous Vehicles* (cité pages 31, 33).

Dellaert, Frank et al. (1999). “Monte carlo localization for mobile robots”. In : *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*. T. 2. IEEE, p. 1322–1328 (cité page 15).

Derkx, François, Jean Luc Sorin et al. (2008). “Inspection des ouvrages d’art par drone : Bilan et perspectives des travaux du LCPC”. In : *Bulletin des laboratoires des ponts et chaussées* 273 (cité page 7).

Dissanayake, G. et al. (2001). “A solution to the simultaneous localization and map building (SLAM) problem”. In : *IEEE Transactions Robotics and Automation* 3.17, p. 229–241 (cité page 22).

Doh, Nakju, Howie Choset et Wan Kyun Chung (2003). “Accurate relative localization using odometry”. In : *Robotics and Automation, 2003. Proceedings. ICRA’03. IEEE International Conference on*. T. 2. IEEE, p. 1606–1612 (cité page 10).

Dummit, D.S. et R.M. Foote (2004). *Abstract Algebra*. John Wiley et Sons (cité page 60).

Durrant-Whyte, H.F et T Bailey (2006). “Simultaneous localization and mapping : part I”. In : *Robotics & Automation Magazine, IEEE* 13.2, p. 99–110 (cité page 22).

Elfes, A. (1987). “Sonar-based real world mapping and navigation”. In : *IEEE Transactions on Robotics and Automation*, p. 249–265 (cité pages 16, 17).

Filliat, David (2007). “A visual bag of words method for interactive qualitative localization and mapping”. In : *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, p. 3921–3926 (cité page 35).

Fischler, Martin A et Robert C Bolles (1981). “Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography”. In : *Communications of the ACM* 24.6, p. 381–395 (cité page 28).

Fox, Dieter et al. (1999). “Monte carlo localization : Efficient position estimation for mobile robots”. In : *AAAI/IAAI* 1999, p. 343–349 (cité page 15).

Frintrop, Simone (2006). *VOCUS : A visual attention system for object detection and goal-directed search*. T. 2. Springer (cité page 33).

Frintrop, Simone et Armin B Cremers (2007). “Top-down Attention Supports Visual Loop Closing.” In : *EMCR*. Citeseer (cité page 33).

Frommer, Andreas et Bruno Lang (2005). “Existence tests for solutions of nonlinear equations using Borsuk’s theorem”. In : *SIAM journal on numerical analysis* 43.3, p. 1348–1361 (cité page 72).

Geppert, Linda (2004). “Qrio, the robot that could”. In : *Ieee Spectrum* 41.5, p. 34–37 (cité page 7).

Gning, A. (2006). “Localisation garantie d’automobiles. Contribution aux techniques de satisfaction de contraintes sur les intervalles”. PhD dissertation. Compiègne, France : Université de Technologie de Compiègne (cité page 16).

Goldsztejn, A. (2003). “Verified projection of the solution set of parametric real systems”. In : *Proceedings of the 2nd International Workshop on Global Constrained Optimization and Constraint Satisfaction, Lausanne, Switzerland, 2003 (COCOS 2003)* (cité page 61).

Goldsztejn, A., W. Hayes et P. Collins (2011). “Tinkerbell Is Chaotic”. In : *SIAM Journal on Applied Dynamical Systems* 10.4, p. 1480–1501 (cité page 56).

Goldsztejn, A. et L. Jaulin (2006). “Inner and Outer Approximations of Existentially Quantified Equality Constraints”. In : *Proceedings of the Twelfth International Conference on Principles and Practice of Constraint Programming, (CP 2006)*. Nantes (France) (cité page 61).

Goldsztejn, Alexandre (2008). “Sensitivity Analysis Using a Fixed Point Interval Iteration”. English. URL : http://hal.archives-ouvertes.fr/hal-00339377 (cité page 97).

Gouaillier, David et al. (2009). “Mechatronic design of NAO humanoid”. In : *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, p. 769–774 (cité page 7).

Granstrom, K et al. (2009). “Learning to detect loop closure from range data”. In : *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, p. 15–22 (cité page 33).

Guyonneau, R. (2013). “ Méthodes ensemblistes pour la localisation en robotique mobile”. Thèse de doct. Université d'Angers (cité pages 11, 16).

Harris, Chris et Mike Stephens (1988). “A combined corner and edge detector.” In : *Alvey vision conference*. T. 15. Manchester, UK, p. 50 (cité page 27).

Hirose, Masato et Kenichi Ogawa (2007). “Honda humanoid robots development”. In : *Philosophical Transactions of the Royal Society A : Mathematical, Physical and Engineering Sciences* 365.1850, p. 11–19 (cité page 7).

Ho, Kin Leong et Paul Newman (2006). “Loop closure detection in SLAM by combining visual and spatial appearance”. In : *Robotics and Autonomous Systems* 54.9, p. 740–749 (cité pages 24, 34).

Itti, Laurent, Christof Koch, Ernst Niebur et al. (1998). “A model of saliency-based visual attention for rapid scene analysis”. In : *IEEE Transactions on pattern analysis and machine intelligence* 20.11, p. 1254–1259 (cité page 33).

Jaulin, L. (2009). “A Nonlinear Set-membership Approach for the Localization and Map Building of an Underwater Robot using Interval Constraint Propagation”. In : *IEEE Transaction on Robotics* 25.1, p. 88–98 (cité pages 22, 99).

Jaulin, L. et G. Chabert (2010). “Resolution of nonlinear interval problems using symbolic interval arithmetic”. In : *Engineering Applications of Artificial Intelligence* 23.6, p. 1035–1049 (cité page 69).

Jaulin, L., M. Kieffer, I. Braems et al. (2001). “Guaranteed Nonlinear Estimation Using Constraint Propagation on Sets”. In : *International Journal of Control* 74.18, p. 1772–1782 (cité page 16).

Jaulin, L., M. Kieffer, O. Didrit et al. (2001). *Applied Interval Analysis*. Springer (cité pages 36, 42, 48, 66).

Jaulin, L. et E. Walter (1993). “Guaranteed nonlinear parameter estimation from bounded-error data via interval analysis”. In : *Mathematics and Computers in Simulation* 35.2, p. 123–137 (cité page 88).

Kaneko, Kenji et al. (2008). “Humanoid robot HRP-3”. In : *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, p. 2471–2478 (cité page 7).

Kortenkamp, David et Terry Weymouth (1994). “Topological mapping for mobile robots using a combination of sonar and vision sensing”. In : *AAAI*. T. 94, p. 979–984 (cité page 17).

Kuipers, Benjamin et Yung-Tai Byun (1991). “A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations”. In : *Robotics and autonomous systems* 8.1, p. 47–63 (cité page 17).

Kurzhanski, A. et I. Valyi (1997). *Ellipsoidal Calculus for Estimation and Control*. Boston, MA : Birkhäuser (cité page 52).

Lagrange, S., N. Delanoue et L. Jaulin (2007). “On sufficient conditions of injectivity, development of a numerical test via interval analysis”. In : *Reliable computing* 13.5, p. 409–421 (cité page 87).

LeBars, F. et al. (2012). “State estimation with fleeting data”. In : *Automatica* 48.2, p. 381–387 (cité page 52).

Leonard, J. J. et H. F. Durrant-Whyte (1992a). *Directed Sonar Sensing for Mobile Robot Navigation*. Boston : Kluwer (cité page 19).

Leonard, J. J. et H. F. Durrant-Whyte (1992b). “Dynamic map building for an autonomous mobile robot”. In : *International Journal of Robotics Research* 11.4 (cité page 14).

Lowe, David G (2004). “Distinctive image features from scale-invariant keypoints”. In : *International journal of computer vision* 60.2, p. 91–110 (cité page 26).

LPO, Brest. URL : http://www.univ-brest.fr/lpo/instrumentation/152b.htm (cité page 113).

Meizel, D., A. Preciado-Ruiz et E. Halbwachs (1996). “Estimation of mobile robot localization : geometric approaches”. In : *Bounding Approaches to System Identification*. Sous la dir. de M. Milanese et al. New York, NY : Plenum Press, p. 463–489 (cité page 54).

Metni, Najib et Tarek Hamel (2007). “A UAV for bridge inspection : Visual servoing control law with orientation limits”. In : *Automation in construction* 17.1, p. 3–10 (cité page 7).

MicroTransat (2014). *http ://www.microtransat.org/* (cité page 8).

Mikolajczyk, Krystian et Cordelia Schmid (2005). “A performance evaluation of local descriptors”. In : *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 27.10, p. 1615–1630 (cité page 26).

Montemerlo S. Thrun, D. Koller et B. Wegbreit (2003). “FastSLAM 2.0 : An improved particle filtering algorithm for simultaneous localization and mapping that provably converges”. In : *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)* (cité pages 22, 35).

Moore, Ramon E (1977). “A test for existence of solutions to nonlinear systems”. In : *SIAM Journal on Numerical Analysis* 14.4, p. 611–615 (cité page 72).

Moore, RE et JB Kioustelidis (1980). “A simple test for accuracy of approximate solutions to nonlinear (or linear) systems”. In : *SIAM Journal on Numerical Analysis* 17.4, p. 521–529 (cité page 72).

Moravec, Hans P (1988). “Sensor fusion in certainty grids for mobile robots”. In : *AI magazine* 9.2, p. 61 (cité pages 16, 17).

NASA-JPL (2014). *http ://marsrover.nasa.gov/home/index.html* (cité page 6).

Neira, José, Juan D Tardós et José A Castellanos (2003). “Linear time vehicle relocation in SLAM”. In : *ICRA*. Citeseer, p. 427–433 (cité page 30).

Neumaier, A. (1990). *Interval Methods for Systems of Equations*. Cambridge, UK : Cambridge University Press (cité page 72).

Neveu, Bertrand, Gilles Trombettoni et Gilles Chabert (2010). “Improving inter-block backtracking with interval Newton”. In : *Constraints* 15.1, p. 93–116 (cité page 76).

Paillat, Jean-Luc (2010). “Conception et contrôle de robots à géométrie variable : applications au franchissement d’obstacles autonome”. Thèse de doct. Université d’Angers (cité page 7).

R. E. Moore (1966). *Interval Analysis*. Englewood Cliffs, NJ : Prentice-Hall (cité pages 39, 74).

R. E. Moore (1979). *Methods and Applications of Interval Analysis*. Philadelphia, PA : SIAM (cité pages 48, 76, 92).

Raibert, Marc et al. (2008). “Bigdog, the rough-terrain quadruped robot”. In : *Proceedings of the 17th World Congress*, p. 10823–10825 (cité page 8).

Raissi, T., N. Ramdani et Y. Candau (2004). “Set membership state and parameter estimation for systems described by nonlinear differential equations”. In : *Automatica* 40, p. 1771–1777 (cité page 56).

Rump, Siegfried M (1980). “Kleine fehlerschranken bei matrixproblemen”. In : *Universität Karlsruhe* (cité pages 76, 95).

Se, Stephen, David G Lowe et James J Little (2005). “Vision-based global localization and mapping for mobile robots”. In : *Robotics, IEEE Transactions on* 21.3, p. 364–375 (cité page 29).

Se, Stephen, David Lowe et Jim Little (2002). “Global localization using distinctive visual features”. In : *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*. T. 1. IEEE, p. 226–231 (cité pages 28, 29).

Shary, S. (2002). “A new technique in systems analysis under interval uncertainty and ambiguity”. In : *Reliable Computing* 8, p. 321–418 (cité page 61).

Shi, Jianbo et Carlo Tomasi (1994). “Good features to track”. In : *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*. IEEE, p. 593–600 (cité page 28).

Simhon, Saul et Gregory Dudek (1998). “A global topological map formed by local metric maps”. In : *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*. T. 3. IEEE, p. 1708–1714 (cité page 19).

Stachniss, Cyrill, Dirk Hahnel et Wolfram Burgard (2004). “Exploration with active loop-closing for FastSLAM”. In : *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. T. 2. IEEE, p. 1505–1510 (cité pages 23, 34, 81).

Stevens, A., M. Stevens et H. Durrant-Whyte (1995). “ldquo ;OXNAV rdquo ; : reliable autonomous navigation”. In : *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*. T. 3, 2607–2612 vol.3 (cité page 14).

Thrun, S., W. Bugard et D. Fox (2005). *Probabilistic Robotics*. Cambridge, M.A. : MIT Press (cité page 22).

Tsotsos, John K et al. (1995). “Modeling visual attention via selective tuning”. In : *Artificial intelligence* 78.1, p. 507–545 (cité page 33).

Tucker, W. (1999). “The Lorenz attractor exists”. In : *Comptes Rendus de l'académie des Sciences*. I 328.12, p. 1197–1202 (cité pages 76, 91).

Ulrich, Iwan et Illah Nourbakhsh (2000). “Appearance-based place recognition for topological localization”. In : *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*. T. 2. Ieee, p. 1023–1029 (cité page 32).

Von Der Hardt, H-J, Didier Wolf et René Husson (1996). "The dead reckoning localization system of the wheeled mobile robot ROMANE". In : *Multisensor Fusion and Integration for Intelligent Systems, 1996. IEEE/SICE/RSJ International Conference on*. IEEE, p. 603–610 (cité page 10).

Walter, E. et L. Pronzato (1997). *Identification of Parametric Models from Experimental Data*. London, UK : Springer-Verlag (cité page 62).

Wang, Junqiu, Hongbin Zha et Roberto Cipolla (2006). "Coarse-to-fine vision-based localization by indexing scale-invariant features". In : *Systems, Man, and Cybernetics, Part B : Cybernetics, IEEE Transactions on* 36.2, p. 413–422 (cité page 32).

Williams, Brian Patrick, Paul Smith et Ian D Reid (2007). "Automatic Relocalisation for a Single-Camera Simultaneous Localisation and Mapping System." In : *ICRA*, p. 2784–2790 (cité page 28).

Williams, Brian et al. (2008). "An image-to-map loop closing method for monocular SLAM". In : *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, p. 2053–2059 (cité page 32).

Williams, Brian et al. (2009). "A comparison of loop closing techniques in monocular SLAM". In : *Robotics and Autonomous Systems* 57.12, p. 1188–1197 (cité page 32).

Yamauchi, Brian et Pat Langley (1997). "Place recognition in dynamic environments". In : *Journal of Robotic Systems* 14.2, p. 107–120 (cité page 19).

Zimmer, Uwe R (1996). "Robust world-modelling and navigation in a real world". In : *Neurocomputing* 13.2, p. 247–260 (cité page 17).

## Résumé

**Analyse par intervalles pour la détection de boucles dans la trajectoire d'un robot mobile**

Le travail de thèse présenté dans ce mémoire a permis le développement d'une nouvelle méthode de détection de boucles dans la trajectoire d'un robot mobile. Celle-ci se base, contrairement à celles existantes, sur l'utilisation de données proprioceptives et est ainsi totalement découplée des problématiques de localisation et cartographie auxquelles elle est généralement associée. L'utilisation de l'analyse par intervalles a permis, dans notre contexte de mesures à erreurs bornées, de calculer deux temps pour lesquels la position du robot est identique alors qu'un mouvement significatif a pu être observé entre ces deux instants. La méthode est automatique et ne nécessite que la connaissance des équations d'état du robot ainsi que les mesures de vitesse effectuées au cours de sa mission. Les résultats de l'algorithme sur données réelles, issues d'une expérience effectuée en milieu sous-marin, sont très satisfaisants et ont permis de prouver l'existence et l'unicité de plus de la moitié des boucles observées dans la trajectoire du robot.



**mots-clé :** analyse par intervalles, détection, fermeture de boucles, trajectoire, robot, sous-marin



## Abstract

**Interval analysis for loop detection in the trajectory of a mobile robot**

The work presented in this thesis deals with a new method for loop detection in the trajectory of a mobile robot. It is based on the use of proprioceptive measures, whereas other methods use exteroceptive measures. This makes this method totally independent from the simultaneous localisation and mapping problems that they are generally coupled with. The use of interval analysis allowed us, in this bounded error context, to compute two instants for which the position of the robot is identical although a significant movement has been observed between these moments. The method is automatic and only requires the knowledge of the state equations of the robot as well as the velocity measures carried out during its mission. The results of the true data algorithm, derived from an experiment carried out underwater, are very satisfactory and made it possible to prove the existence and uniqueness of more than half of the loops observed in the trajectory of the robot.



**keywords :** interval analysis, detection, loop closure, trajectory, robot, underwater