



HAL
open science

Méthodes ensemblistes pour la localisation en robotique mobile

Rémy Guyonneau

► **To cite this version:**

Rémy Guyonneau. Méthodes ensemblistes pour la localisation en robotique mobile. Automatique / Robotique. Université d'Angers, 2013. Français. NNT: . tel-00961501

HAL Id: tel-00961501

<https://theses.hal.science/tel-00961501v1>

Submitted on 20 Mar 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse de Doctorat

Rémy GUYONNEAU

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université d'Angers
sous le label de l'Université de Nantes Angers Le Mans

Discipline : Automatique et Génie Informatique
Laboratoire : Laboratoire d'Ingénierie des Systèmes Automatisés (LISA)

Soutenue le 19 novembre 2013

École doctorale : 503 (STIM)
Thèse n° : 1379

Méthodes ensemblistes pour la localisation en robotique mobile

JURY

Rapporteurs : **M. François CHARPILLET**, Directeur de Recherche INRIA, LORIA, Nancy
M. Luc JAULIN, Professeur des Universités à l'ENSTA, Laboratoire DTN, Brest

Examineurs : **M. Sébastien LAGRANGE**, Maître de Conférences à l'Université d'Angers, LISA
M. Gilles CHABERT, Maître de Conférences à l'École des Mines de Nantes, LINA
M. Olivier SIMONIN, Professeur des Universités à l'INSA de Lyon, CITI-Inria Lab

Directeur de thèse : **M. Laurent HARDOUIN**, Professeur des Universités à l'Université d'Angers, LISA

M thodes ensemblistes pour la localisation en robotique mobile

TH SE DE DOCTORAT

Sp cialit  : Automatique et G nie Informatique

 cole doctorale : STIM (Science et Technologies de l'Information et Math matiques)

Pr sent e et soutenue publiquement

le : 19 Novembre 2013

  : l'ISTIA,  cole d'Ing nieur de l'Universit  d'Angers

par : R my GUYONNEAU

Devant le jury ci-dessous :

Rapporteurs :

Fran ois CHARPILLET - Directeur de Recherche INRIA (LORIA - Nancy)
Luc JAULIN - Professeur des Universit s   l'ENSTA (Laboratoire DTN - Brest)

Examineurs :

Gilles CHABERT - Ma tre de Conf rences   l' cole des Mines de Nantes (LINA)
Olivier SIMONIN - Professeur des Universit s   l'INSA de Lyon (CITI-Inria Lab)

Directeur de th se :

Laurent HARDOUIN - Professeur des Universit s   l'Universit  d'Angers (LISA)

Co-encadrant :

S bastien LAGRANGE - Ma tre de Conf rences   l'Universit  d'Angers (LISA)

Laboratoire : LISA (Laboratoire d'Ing nierie des Syst mes Automatis s)
EA4094 - Universit  d'Angers - 62, avenue Notre Dame du Lac - 49000 ANGERS

ED (N ) : 503

Remerciements

Le travail présenté dans ce mémoire a été réalisé au sein du Laboratoire d'Ingénierie des Systèmes Automatisés (LISA) de l'École d'Ingénieur ISTIA, à l'Université d'Angers.

Mes premiers remerciements vont à Laurent Hardouin et Sébastien Lagrange pour leur encadrement, leur disponibilité et pour tout ce qu'ils m'ont appris. Je remercie Laurent pour m'avoir fait confiance et pour m'avoir permis de faire cette thèse. Je remercie Sébastien pour sa présence et ses conseils qui m'ont fait avancer et progresser.

Je tiens à remercier Jean-Louis Boimond pour m'avoir accueilli au LISA et, dans un autre registre, pour tous ses conseils verticaux.

Merci à Gilles Chabert, François Charpillet, Luc Jaulin et Olivier Simonin d'avoir accepté de juger ce travail écrit et oral. Merci à Luc pour son suivi et toutes ses remarques constructives.

Parce-que travailler dans une bonne ambiance est tout de suite plus agréable, je tiens à remercier mes collègues et amis du bureau E37 (Rabah, Euriell, Jean-Luc, Sara, Julien, Oumar, Agnès, Alban, Karl, Yann...) pour tous les bons moments, les pauses cafés, les astuces LaTeX, les conseils administratifs, les Beaussier...

Mes remerciements vont aussi aux membres du LISA pour leur accueil et leurs conseils avisés du monde de la recherche et de l'enseignement. Je tiens en particulier à remercier Philippe Lucidarme pour m'avoir permis de participer à l'aventure Cart-O-matic. Je n'oublie pas le personnel de l'ISTIA pour leur disponibilité et, entre autre, les midis en salle de convivialité.

J'adresse évidemment mes remerciements à ma famille et mes amis qui m'ont aidé et encouragé aux moments opportuns. Merci à mes grands parents qui sont toujours là pour moi.

Un grand merci à mes parents et mon p'tit frère qui m'ont soutenu, et me sou-

tiennent encore, dans chacun de mes choix.

Je tiens aussi à remercier toutes les personnes que j'ai croisé durant ces huit années universitaires, qui ont cru en moi et m'ont donné envie d'aller plus loin que ce que j'avais initialement prévu.

Enfin et surtout, merci à Pauline.

Table des matières

Liste des notations	vii
Liste des termes Anglais	xi
Introduction	1
1 Présentation générale du problème de localisation	3
1.1 Localisation locale ou globale	3
1.2 Environnement statique ou dynamique	4
1.3 Localisation passive ou active	4
1.4 Robot seul ou en meute	5
2 Problèmes de localisation considérés	5
2.1 Présentation des robots	5
2.2 Présentation de l'environnement	7
2.3 Présentation des capteurs	7
2.4 Présentation des problèmes de localisations traités	8
3 Aperçu des chapitres	9
Chapitre 1 – État de l'art de la localisation en robotique mobile	11
1 Considération de l'environnement	12
1.1 Différents types d'environnements	12
1.2 Différentes représentations	14
1.3 Deux approches métriques classiques	15
2 Différents capteurs	16
2.1 Odométrie	18
2.2 Capteurs de distances	18
2.3 Caméras	21
3 Différentes approches aux problèmes de localisations	24
3.1 Approches probabilistes	24
3.2 Approches ensemblistes	28

3.3	Conclusion	33
Chapitre 2	– Méthode ensembliste de localisation globale	37
1	Présentation du problème	38
1.1	Robot	38
1.2	Environnement	39
1.3	Objectif	41
2	Présentation de la méthode	43
2.1	Formalisation en problème de satisfaction de contraintes	43
2.2	Présentation des contracteurs	48
2.3	Algorithme IAL	58
3	Expérimentations et comparaisons	63
3.1	Expérimentations avec un ordinateur distant	63
3.2	Expérimentations avec un MiniRex	66
3.3	Comparaison MCL/IAL	71
3.4	Conclusion	76
Chapitre 3	– Contracteurs de visibilité	79
1	Définitions générales	81
1.1	Visibilité entre deux points	81
1.2	Espaces visible/non-visible d'un point	82
1.3	Espaces visible/non-visible/partiellement-visible d'un ensemble	83
1.4	Visibilité considérant un ensemble d'obstacles	86
2	Cas particuliers de la visibilité	93
2.1	Visibilité d'un point	95
2.2	Visibilité d'un segment	98
2.3	Visibilité d'un polygone	105
3	Les contracteurs	112
3.1	Contracteurs de visibilité d'un point	114
3.2	Contracteurs de visibilité d'un segment	117
3.3	Contracteurs de visibilité d'un polygone	120
Chapitre 4	– Applications de la visibilité	125
1	Suivi de posture à l'aide d'une information booléenne	126
1.1	Présentation du problème	126
1.2	Présentation de la méthode	130
1.3	Expérimentations et résultats	133
1.4	Conclusion sur cette application de la visibilité	138

2	Visibilité pour l'algorithme IAL	139
2.1	Formalisation de la contrainte	141
2.2	Méthode de traitement de la contrainte	141
2.3	Résultats	143
2.4	Conclusion sur cette application	144
	Conclusion générale	147
	Productions scientifiques	151
	Annexe A – Introduction à l'analyse par intervalles	153
1	Présentation générale	153
1.1	Manipulation d'ensembles	154
1.2	Intervalles	157
1.3	Arithmétiques des intervalles	164
2	Deux outils ensemblistes.	173
2.1	Inversion ensembliste	173
2.2	Problèmes de satisfaction de contraintes	176
	Annexe B – Présentation des outils probabilistes	183
1	Localisation probabiliste.	184
1.1	Introduction aux probabilités	184
1.2	Approche probabiliste de la localisation	188
1.3	Filtre Bayésien	191
2	Filtres Gaussiens	194
2.1	Présentation générale.	194
2.2	Filtre de Kalman	196
2.3	Filtre de Kalman étendu	198
3	Filtres Non-paramétriques.	205
3.1	Présentation générale.	205
3.2	Filtre à particules	205
	Annexe C – Outils pour la visibilité	211
1	Manipulation de segments.	212
1.1	Équation paramétrique d'un segment	212
1.2	Intersection de deux segments.	212
1.3	Segments et polygones convexes	220
2	Quelques propriétés sur le calcul de déterminant.	223
2.1	Première proposition	224

2.2	Deuxième proposition	225
2.3	Troisième proposition.	226
3	Quelques démonstrations	229
3.1	Démonstration pour la Proposition 3.6.	229
3.2	Démonstration pour la Proposition 3.7.	231
3.3	Démonstration de la Proposition 3.11	233

Liste des notations

Notations Générales

\triangleq	:	Égal à, par définition.
\emptyset	:	Ensemble vide.
\wedge	:	Opérateur logique ET.
\vee	:	Opérateur logique OU.
\mathbb{R}	:	Ensemble des réels.
$ $:	Définition d'ensemble par compréhension (tel que).
\forall	:	Quantificateur universel (pour tout).
\exists	:	Quantificateur existentiel (il existe).
$\{x\}$:	Ensemble des x .

Visibilité

$(\mathbf{x}_1 \vee \mathbf{x}_2)_{\varepsilon_j}$:	\mathbf{x}_1 voit \mathbf{x}_2 par rapport à l'obstacle ε_j .
$(\mathbf{x}_1 \bar{\vee} \mathbf{x}_2)_{\varepsilon_j}$:	\mathbf{x}_1 ne voit pas \mathbf{x}_2 par rapport à l'obstacle ε_j .
$Seg(\mathbf{x}_1, \mathbf{x}_2)$:	Segment défini par les points \mathbf{x}_1 et \mathbf{x}_2 .
$E_{\varepsilon_j}(\mathbf{x})$:	Espace visible de \mathbf{x} par rapport à l'obstacle ε_j .
$\widehat{E}_{\varepsilon_j}(\mathbf{x})$:	Espace non-visible de \mathbf{x} par rapport à l'obstacle ε_j .
$E_{\varepsilon_j}(\mathbb{X})$:	Espace visible de l'ensemble \mathbb{X} par rapport à l'obstacle ε_j .
$\widehat{E}_{\varepsilon_j}(\mathbb{X})$:	Espace non-visible de l'ensemble \mathbb{X} par rapport à l'obstacle ε_j .
$\widetilde{E}_{\varepsilon_j}(\mathbb{X})$:	Espace partiellement-visible de l'ensemble \mathbb{X} par rapport à l'obstacle ε_j .

Problèmes de localisation

r_i	: $i^{\text{ème}}$ robot.
t	: Temps discret.
$\mathbf{q}_{i,t}$: Posture du robot r_i à l'instant t .
$\mathbf{x}_{i,t} = (x_{1,i,t}, x_{2,i,t})$: Position du robot r_i à l'instant t .
$\theta_{i,t}$: Orientation (direction) du robot r_i à l'instant t .
$\mathbf{u}_{i,t}$: Vecteur de contrôles appliqué sur le robot r_i entre les instants t et $t + 1$.
$\Delta_{\mathbf{x}_{i,t}}$: Distance parcourue entre $\mathbf{x}_{i,t}$ et $\mathbf{x}_{i,t+1}$.
$\Delta_{\theta_{i,t}}$: Différence d'orientation entre $\theta_{i,t}$ et $\theta_{i,t+1}$.
f	: Fonction dynamique correspondant à la dynamique du robot.
$\mathbf{z}_{i,t}$: Mesures effectuées par le robot r_i à l'instant t .
$\mathbf{w}_{i,t}$: $i^{\text{ème}}$ mesure effectuée à l'instant t .
$y_{i,t}$: Distance de la $i^{\text{ème}}$ mesure effectuée à l'instant t .
$\gamma_{i,t}$: Angle de la $i^{\text{ème}}$ mesure effectuée à l'instant t .
\mathcal{E}	: Environnement dans lequel évolue(nt) le(s) robot(s).
ε_j	: $j^{\text{ème}}$ obstacle de l'environnement \mathcal{E} .
\mathcal{C}	: Carte connue de l'environnement.
\mathcal{E}_G	: Grille d'occupation caractérisant l'environnement \mathcal{E} .
$c_{i,j}$: Cellule de la grille aux coordonnées (i, j) .

Analyse par Intervalles

\cap	: Intersection.
\cup	: Union ensembliste.
\sqcup	: Union intervalle.
\subset	: Inclusion.
$\mathbb{X} \times \mathbb{Y}$: Produit Cartésien.
$\mathbb{X} \setminus \mathbb{Y}$: $\{\mathbf{x} \mid (\mathbf{x} \in \mathbb{X}) \wedge (\mathbf{x} \notin \mathbb{Y})\}$.
\mathbb{X}^-	: Sous-approximation de l'ensemble \mathbb{X} .
\mathbb{X}^+	: Sur-approximation de l'ensemble \mathbb{X} .
$[\cdot]$: Enveloppe intervalle.
$[\mathbf{x}]$: Intervalle.
\underline{x}	: Borne inférieure de $[\mathbf{x}]$.
\bar{x}	: Borne supérieure de $[\mathbf{x}]$.
\mathbb{IR}	: Ensemble des intervalles.
$w([\mathbf{x}])$: Taille de $[\mathbf{x}]$.
$\text{mid}([\mathbf{x}])$: Centre de $[\mathbf{x}]$.
$[\mathbf{x}]$: Vecteur d'intervalles (pavé, boîte).
\mathbb{IR}^n	: Ensemble des vecteurs d'intervalles.
$([\mathbf{x}_1], [\mathbf{x}_2]) = \text{Bissect}([\mathbf{x}])$: Bisection de $[\mathbf{x}]$ en deux pavés $[\mathbf{x}_1]$ et $[\mathbf{x}_2]$.
$[f]$: Fonction d'inclusion de la fonction f .

$[f]^*$: Fonction d'inclusion minimale de la fonction f .

Approche probabiliste

X	: Variable aléatoire X .
$p(x)$: Probabilité d'avoir $X = x$.
$X \sim f_X$: Variable aléatoire X distribuée selon une densité f_X .
$X \sim \mathcal{N}(\mu_X, \sigma_X^2)$: Variable aléatoire X distribuée selon une densité Gaussienne de moyenne μ_X et de covariance σ_X^2 .
$p(x, y)$: Probabilité d'avoir $X = x$ et $X = y$.
$p(x y)$: Probabilité d'avoir $X = x$ sachant qu'on a $X = y$.
$E[X]$: Espérance de la variable aléatoire X .
$V(X)$: Variance de la variable aléatoire X .
$\text{cov}(X)$: Covariance de la variable aléatoire X .
Σ_X	: Matrice de covariance du vecteur de variables aléatoires X .
μ_X	: Moyenne.
σ^2	: Covariance.
$bel(\mathbf{x}_t)$: Croyance de l'état \mathbf{x} à l'instant t .
$\overline{bel}(\mathbf{x}_t)$: Prédiction de l'état \mathbf{x} à l'instant t (avant correction).
ε_t	: Bruit affectant le changement d'état.
R_t	: Matrice de covariance du bruit ε_t .
γ_t	: Bruit affectant le vecteur de mesures.
Q_t	: Matrice de covariance du bruit γ_t .
\mathcal{X}_t	: Ensemble des particules à l'instant t .
$\mathbf{x}_t^{[M]}$: $M^{\text{ème}}$ particule à l'instant t .
$w_t^{[M]}$: Poids de la $M^{\text{ème}}$ particule à l'instant t .

Liste des termes Anglais

Au sein de ce document tous les termes scientifiques, usuellement utilisés en anglais, ont été traduits en français par soucis d'homogénéité. Pour le lecteur intéressé, ci-suit la correspondance anglaise de certains termes utilisés afin de faciliter la recherche de documents et d'informations sur ces notions.

Suivi de posture	-	Pose tracking
Posture	-	Pose
Environnement intérieur	-	Indoor environment
Environnement extérieur	-	Outdoor environment
Carte d'amers	-	Landmark map
Carte de caractéristiques	-	Feature map
Grille d'occupation	-	Occupancy grid map
Capteur de distance	-	Range sensor
Télémètre laser	-	Laser Range Finder (LIDAR)
Diaphonie	-	Cross talk
Diviseur de rayon	-	Beam splitter
Caméra de profondeur	-	Depth camera

Introduction

Sommaire

1	Présentation générale du problème de localisation	3
1.1	Localisation locale ou globale	3
1.2	Environnement statique ou dynamique	4
1.3	Localisation passive ou active	4
1.4	Robot seul ou en meute	5
2	Problèmes de localisation considérés	5
2.1	Présentation des robots	5
2.2	Présentation de l'environnement	7
2.3	Présentation des capteurs	7
2.4	Présentation des problèmes de localisations traités	8
3	Aperçu des chapitres	9

La robotique est un domaine en plein essor, amenant son lot d'innovations et de défis technologiques et académiques. Les robots sont déjà présents dans notre environnement (sur les chaînes de montages dans l'automobile, chez les particuliers sous forme de jouets ou d'aspirateurs...) et la tendance est à une augmentation de cette présence. Les tâches à réaliser et les environnements à affronter par les futurs robots seront de plus en plus complexes.

Les systèmes robotiques peuvent être classifiés selon cinq catégories, comme précisé dans [Paillat 2002] :

- Les *robots marcheurs* désignent l'ensemble des robots équipés de *pattes* (par opposition aux robots équipés de roues ou encore de chenilles). On peut notamment noter les robots humanoïdes (de forme humaine) qui sont des robots marcheurs bipèdes.
- Les *robots manipulateurs* sont généralement des bras mécaniques fixés au sol ou encore sur des rails. Ils sont notamment utilisés en industrie afin d'effectuer des tâches répétitives rapidement et de façon précise.

- Les *robots aériens*, aussi appelés *drones* ou UAV¹ désignent l'ensemble des robots volants sans pilote. On peut par exemple noter les avions militaires photographiant des zones dangereuses.
- Les *robots sous-marins*, ou UUV² désignent l'ensemble des robots sous-marins évoluant sans pilote. Ils sont par exemple utilisés pour la détection de mines sous-marines.
- Les *robots mobiles terrestres*, ou UGV³ correspondent à l'ensemble des véhicules terrestres se déplaçant sans pilote, à l'exception des robots marcheurs. Les robots aspirateurs sont des exemples d'UGV.

Il est important de noter que le terme "robots mobiles" bien que désignant l'ensemble des robots à bases mobiles (par opposition aux robots manipulateurs), est généralement employé pour désigner les robots mobiles terrestres. Ce sont ces robots qui sont considérés dans ce manuscrit. La Figure 1 présente un exemple de robot mobile tel que considéré ici.



Figure 1 – Exemple de robot mobile : le robot MiniRex, conçu au LISA (université d'Angers).

Le problème de localisation est un problème majeur en robotique mobile. Afin de pouvoir effectuer un certain nombre de tâches de façon autonome, un robot doit être capable de se localiser dans son environnement. Prenons par exemple une tâche de navigation : un robot doit se déplacer d'un point A (sa position courante) à un point B (sa position cible). Ce qui en prenant un exemple concret peut se traduire par : un robot aspirateur doit rejoindre sa station d'accueil afin de pouvoir se recharger. Pour pouvoir effectuer cette opération, il est indispensable que le robot connaisse sa position courante afin de calculer la trajectoire à emprunter pour aller de sa position courante à la position cible. De plus, le robot se doit de calculer sa position tout

1. Unmanned Aerial Vehicles - véhicule aérien sans pilote.
2. Unmanned Undersea Vehicles - véhicule sous-marin sans pilote.
3. Unmanned Ground Vehicles - véhicule terrestre sans pilote.

au long de son déplacement, pour pouvoir par exemple mettre à jour sa trajectoire, mais surtout pour pouvoir déterminer quand il est arrivé à destination.

Cette thèse s'intéresse au problème de localisation en robotique mobile.

1 Présentation générale du problème de localisation

Le problème de localisation en robotique mobile consiste à déterminer la posture (position et orientation) d'un robot mobile dans son environnement. Malheureusement, la posture d'un robot ne peut pas être mesurée directement. En effet, les robots ne possèdent pas de capteurs parfaits permettant de déterminer directement leur posture. On notera que le *GPS*⁴ est un système permettant de mesurer directement la position. Cependant ce système n'est pas parfait (précision de l'ordre du mètre) et est difficilement exploitable en milieu clos (intérieur de bâtiments, forêts...). Partant du principe qu'une seule donnée capteur ne peut pas permettre une localisation efficace, le problème de localisation consiste alors à traiter des données de natures différentes (mesures de différents capteurs, différentes cartes...) afin de déterminer la posture d'un robot.

Il existe une multitude de problèmes de localisation, plus ou moins difficiles. Ci-suivent quelques uns de ces problèmes [Thrun 2005]. Cette liste, loin d'être exhaustive, a pour objectif de présenter un aperçu de la variété des difficultés que présente la localisation en robotique mobile.

1.1 Localisation locale ou globale

Les problèmes de localisation peuvent être caractérisés par le type des informations disponibles, à l'initialisation et pendant le processus de localisation.

Suivi de posture. Dans un contexte de suivi de posture on suppose que la posture initiale du robot est connue. La localisation du robot se fait alors de façon itérative : connaissant sa posture précédente, en évaluant son déplacement un robot peut estimer sa posture courante. On procède ainsi à une localisation locale : la posture courante du robot est localement évaluée en fonction de la posture précédente. La difficulté de ce type de problème consiste à éviter la *dérive* du robot. En effet, l'évaluation du déplacement du robot est soumise à une certaine imprécision, qui va se répercuter et s'amplifier tout au long du processus itératif de localisation. Pour éviter cette dérive, on considère généralement des mesures extéroceptives (type sonar, télémètre laser...) ainsi que la connaissance de l'environnement (une carte) pour *recadrer* l'évaluation de la posture du robot.

4. Global Positioning System – système de localisation globale. La position est obtenue en triangulant les informations provenant d'au moins quatre satellites.

Localisation globale. Cette fois la posture initiale du robot n'est pas connue. Dans ce cas il n'est pas possible d'évaluer la posture courante du robot en fonction de sa précédente. Ce problème est plus difficile que le précédent car la posture du robot doit être évaluée de façon globale dans l'environnement.

Kidnapping. Ce problème correspond à un *mélange* des deux premiers. Alors que le robot se déplace dans son environnement (en effectuant un suivi de posture par exemple) il est *kidnappé et déplacé* de façon aléatoire dans l'environnement (sans qu'il le sache). En d'autres termes, le robot effectue un déplacement imprévu qu'il n'est pas capable d'évaluer. Toute la difficulté de cette situation consiste à détecter ce kidnapping (identifier le fait que le robot ne soit plus où il croit être). Une fois le kidnapping identifié, on se retrouve dans un contexte de localisation globale.

1.2 Environnement statique ou dynamique

La difficulté d'un problème de localisation dépend de l'environnement dans lequel évolue le robot.

Environnement statique. Un environnement statique correspond à un environnement qui ne change pas au cours du temps (les murs d'un bâtiment par exemple). La seule donnée variable du problème de localisation correspond alors à la posture du robot.

Environnement dynamique. Un environnement dynamique est un environnement dont la configuration est modifiée au cours du temps. En d'autres termes, il y a des entités, en plus du robot, qui se déplacent dans l'environnement pendant que le robot se localise (des personnes qui marchent, des portes qui s'ouvrent et se ferment...). On peut noter que la plupart des environnements réels sont dynamiques. Le processus de localisation dans un environnement dynamique est bien évidemment plus difficile que dans un environnement statique.

1.3 Localisation passive ou active

Il est possible de caractériser les algorithmes de localisation en tenant compte de l'influence du processus de localisation sur les déplacements du robots.

Localisation passive. Lors d'une localisation passive, l'algorithme de localisation se contente *d'observer*. Les déplacements du robot sont gérés par un autre module et n'ont pas pour objectif de faciliter la localisation du robot. C'est comme si le robot avait deux pilotes : le premier conduit et le deuxième regarde par la fenêtre pour essayer de savoir où il est, sans communication entre les deux.

Localisation active. Pendant une localisation active, le processus de localisation influence les déplacements du robot afin de faciliter l'évaluation de la posture du

robot et de minimiser l'erreur de localisation (en évitant les parties symétriques d'un environnement par exemple). En reprenant l'analogie précédente c'est comme si le robot n'avait cette fois qu'un seul pilote, qui conduit tout en regardant autour de lui pour se localiser.

1 .4 Robot seul ou en meute

Les algorithmes de localisation peuvent être caractérisés en fonction du nombre de robots considéré.

Localisation mono-robot. La plupart du temps le problème de localisation s'attache à localisation d'un seul robot évoluant dans son environnement. En ne considérant qu'un seul robot, les données récoltées ont l'avantage d'être centralisées sur la même plateforme. De plus il n'y a pas de problème lié à la communication inter robots.

Localisation multi-robots. En considérant une meute de robots (plusieurs robots évoluant dans le même environnement au même moment), il y a deux approches possibles. La première consiste à considérer les robots de façon individuelle, se ramenant ainsi à plusieurs localisations mono-robot. Cependant si les robots sont capables de se détecter mutuellement et de communiquer entre eux, il est possible d'améliorer les résultats de la localisation. En effet en fusionnant l'estimation qu'un robot fait de sa propre posture avec l'évaluation faite par un autre robot de la meute, il est possible d'améliorer la localisation du premier robot. Ceci permet notamment d'améliorer la robustesse du processus de localisation (deux informations de deux sources différentes valent mieux qu'une seule).

2 Problèmes de localisation considérés

Le problème de localisation en robotique mobile soulève une multitude de sous-problèmes. Il est donc important de définir le périmètre de cette thèse et d'introduire différentes notations. Pour traiter un problème de localisation il est nécessaire de considérer : des robots, des capteurs associés à ces robots, un environnement dans lequel les robots évoluent et évidemment un problème de localisation.

2 .1 Présentation des robots

Nous nous intéressons ici aux robots mobiles terrestres, tel qu'illustré sur la Figure 1. Sur les différentes figures présentes dans ce document, les robots sont schématisés comme présentés Figure 2.

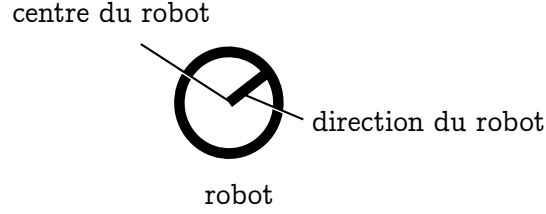
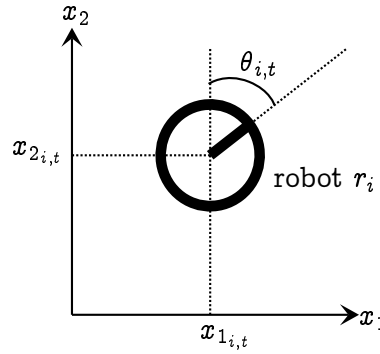


Figure 2 – Schématisation d'un robot.

Figure 3 – Exemple de posture pour un robot r_i à l'instant t .

La posture $\mathbf{q}_{i,t}$ d'un robot r_i à l'instant t est définie par

$$\mathbf{q}_{i,t} = (x_{1,i,t}, x_{2,i,t}, \theta_{i,t})^T, \quad (1)$$

avec $\mathbf{x}_{i,t} = (x_{1,i,t}, x_{2,i,t})^T$ la position du robot dans l'environnement et $\theta_{i,t}$ son orientation. Cette posture est donnée dans un repère global attaché à l'environnement dans lequel le robot évolue. On notera que sur l'ensemble des schémas le centre du robot correspond au point de référence pour sa posture (Figure 3). En pratique, le point de référence peut par exemple correspondre au centre de l'arbre des roues motrices du robot ou encore à la position d'un des capteurs utilisés pour localiser le robot (un télémètre laser par exemple).

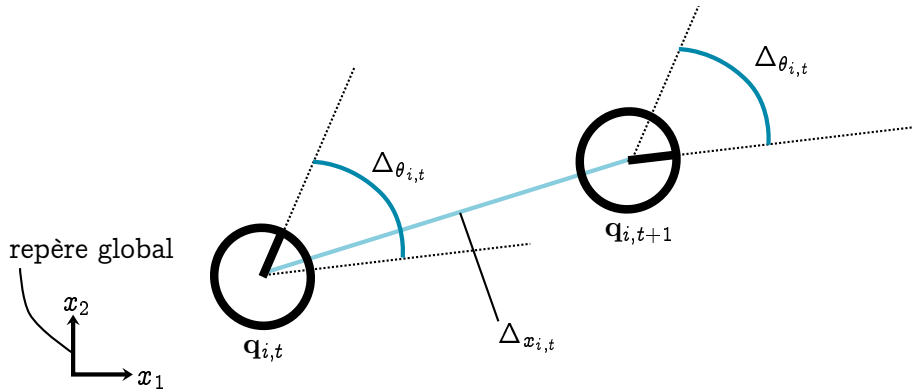
La dynamique d'un robot r_i est caractérisée par

$$\mathbf{q}_{i,t+1} = f(\mathbf{q}_{i,t}, \mathbf{u}_{i,t}), \quad (2)$$

avec $\mathbf{u}_{i,t}$ le vecteur de contrôles appliqué sur le robot entre les instants t et $t + 1$ et f la fonction modélisant la dynamique du robot.

Dans ce manuscrit la fonction f considérée est la suivante

$$\begin{pmatrix} x_{1,i,t+1} \\ x_{2,i,t+1} \\ \theta_{i,t+1} \end{pmatrix} = \begin{pmatrix} x_{1,i,t} \\ x_{2,i,t} \\ \theta_{i,t} \end{pmatrix} + \begin{pmatrix} \sin(\theta_{i,t})\Delta x_{i,t} \\ \cos(\theta_{i,t})\Delta x_{i,t} \\ \Delta\theta_{i,t} \end{pmatrix}, \quad (3)$$

Figure 4 – Illustration de la dynamique d'un robot r_i .

avec $\mathbf{u}_{i,t} = (\Delta x_{i,t}, \Delta \theta_{i,t})^T$, $\Delta x_{i,t}$ la distance parcourue entre $\mathbf{x}_{i,t}$ et $\mathbf{x}_{i,t+1}$, et $\Delta \theta_{i,t}$ la différence d'orientation entre $\theta_{i,t}$ et $\theta_{i,t+1}$ (Figure 4).

On peut noter que le Chapitre 2 s'intéresse à une localisation mono-robot tandis que le Chapitre 4 présente notamment une méthode de localisation multi-robots.

2.2 Présentation de l'environnement

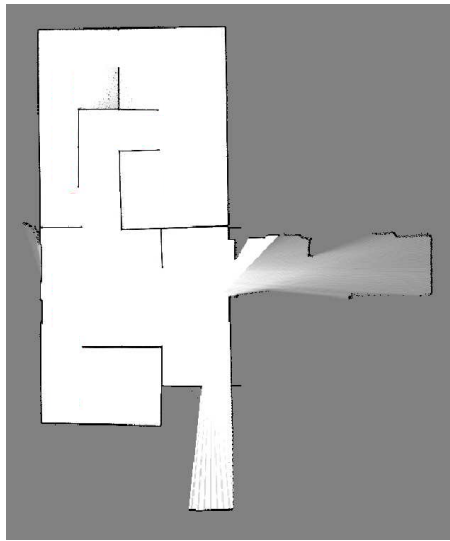
On considère tout au long de ce document un environnement intérieur (Chapitre 1, Section 1.1.1) statique. Un environnement intérieur correspond à l'intérieur d'un bâtiment (un bureau, un immeuble, une usine...). L'environnement, nommé \mathcal{E} , est considéré connu et est modélisé à l'aide d'une approche métrique (Chapitre 1, Section 1.2.2). En d'autres termes la carte connue de l'environnement, notée \mathcal{C} , est basée sur les distances séparant les différents obstacles ainsi que sur leurs positions respectives les uns vis à vis des autres.

Deux approches métriques sont manipulées dans ce document. Le Chapitre 2 considère une grille d'occupation booléenne (Figure 5a), tandis que le Chapitre 4 utilise des cartes de caractéristiques approximant les obstacles par des polygones convexes (Figure 5b).

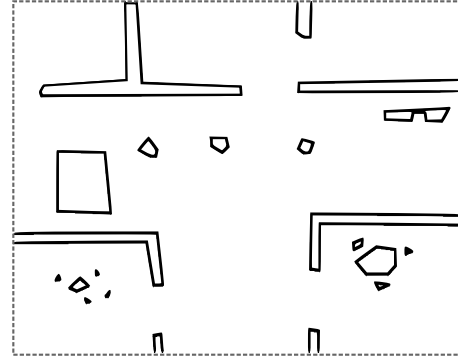
2.3 Présentation des capteurs

Tous les robots considérés disposent d'une technique d'odométrie. En d'autres termes ils sont capables d'approximer leur trajectoire entre deux instants t et $t + 1$ (Chapitre 1, Section 2.1). Dans notre cas l'odométrie permet d'approximer $\Delta \theta_{i,t}$ et $\Delta x_{i,t}$.

La supposition d'un environnement intérieur rendant l'utilisation d'un GPS difficile, il est nécessaire d'utiliser d'autres méthodes, d'autres capteurs extéroceptifs



(a) Une grille d'occupation. Les cellules blanches sont vides, les cellules noires contiennent au moins un obstacle et les cellules grises sont inexplorées.



(b) Une carte de caractéristiques. Les obstacles sont représentés sous la forme de polygones.

Figure 5 – Exemple de cartes considérées.

afin de localiser les robots. Le jeu de mesures fourni par le capteur du robot r_i à l'instant t est noté $z_{i,t}$.

Le Chapitre 2 considère un robot équipé d'un télémètre laser à balayage (Chapitre 1, Section 2.2.2). Chaque mesure fournie par le capteur correspond alors à la distance séparant le robot de l'obstacle le plus proche dans une direction donnée.

Le Chapitre 4 présente quand à lui l'utilisation d'une mesure originale : la visibilité entre deux robots. Cette information booléenne (deux robots se voient ou non) peut par exemple être fournie par une caméra ou encore un capteur infrarouge. Cette information est utilisée afin de localiser une meute de robots évoluant dans un même environnement.

2.4 Présentation des problèmes de localisations traités

Comme précisé précédemment, il existe une multitude de problèmes de localisation. Cette thèse s'intéresse particulièrement à deux problèmes : le problème de localisation globale et le problème de suivi de posture dans des environnements intérieurs statiques. Ces deux problèmes sont traités de façon passive (le processus de localisation n'a pas d'influence sur le déplacement des robots).

Le Chapitre 2 s'intéresse au problème de localisation globale dans un contexte

ensembliste. La plupart des méthodes classiques de localisation sont probabilistes : elles utilisent des densités de probabilités (Annexe B) afin de caractériser les différentes incertitudes. Dans un contexte ensembliste, ces incertitudes sont caractérisées à l'aide d'ensembles, et plus particulièrement d'intervalles (Annexe A) dans notre cas.

Dans le Chapitre 4, nous nous intéressons au problème de suivi de posture en présence d'une information booléenne. A priori, les méthodes probabilistes classiques ne peuvent pas traiter une telle information. Une approche ensembliste du problème permet en revanche de tenir compte de cette information et de s'intéresser au problème de suivi de posture. Afin de pouvoir traiter cette information de visibilité une meute de robots est considérée (avec un seul robot, il n'y aurait pas d'information...).

3 Aperçu des chapitres

Afin de conclure cette introduction, nous présentons ici un rapide aperçu des chapitres qui suivent et mettons en avant les deux contributions majeures de cette thèse.

Chapitre 1. Dans ce chapitre nous présentons un état de l'art de la localisation en robotique mobile. Nous introduisons les différentes problématiques existantes ainsi que les solutions classiquement utilisées pour les résoudre. Ce chapitre permet de situer les contributions de cette thèse dans le paysage de la robotique mobile actuelle.

Chapitre 2. Ce chapitre présente la première contribution de cette thèse : un algorithme ensembliste efficace permettant de traiter les problèmes de localisation globale et de kidnapping. Il introduit le problème considéré puis la méthode mise en place afin de le résoudre. Plusieurs expérimentations sont présentées afin de démontrer l'intérêt et l'efficacité de la méthode. Ces travaux ont permis les publications suivantes [Guyonneau 2011, Guyonneau 2012c, Guyonneau 2013b].

Chapitre 3. Ce chapitre présente la deuxième contribution de cette thèse : le développement d'une méthode permettant de traiter une information de visibilité entre deux points. Ce chapitre est essentiellement théorique et permet de mettre en place un ensemble de notions qui ont des applications directes aux problèmes de localisation en robotique mobile.

Chapitre 4. Dans ce chapitre nous illustrons l'intérêt des notions présentées dans le Chapitre 3 pour les problèmes de localisation en robotique mobile. Nous présentons deux cas d'applications de ces notions : le suivi de posture d'une meute de robots à l'aide d'une information booléenne et une amélioration de l'algorithme de localisation globale présenté au Chapitre 2. Ces travaux ont permis les publications suivantes [Guyonneau 2012a, Guyonneau 2012b, Guyonneau 2013a].

État de l'art de la localisation en robotique mobile

Sommaire

1	Considération de l'environnement	12
1.1	Différents types d'environnements	12
1.2	Différentes représentations	14
1.3	Deux approches métriques classiques	15
2	Différents capteurs	16
2.1	Odométrie	18
2.2	Capteurs de distances	18
2.3	Caméras	21
3	Différentes approches aux problèmes de localisations	24
3.1	Approches probabilistes	24
3.2	Approches ensemblistes	28
3.3	Conclusion	33

Ce chapitre a pour objectif de dresser un état des lieux du problème de localisation en robotique mobile. Sans pour autant être exhaustif, l'objectif est de présenter différentes méthodes et outils classiquement utilisés quand on s'intéresse aux problèmes de localisation. Ce chapitre est divisé en trois grandes sections. Premièrement, différentes approches concernant la considération de l'environnement sont présentées : pour qu'un robot puisse se localiser, il a besoin de détenir des informations sur l'environnement dans lequel il évolue. Ensuite, les capteurs classiquement considérés sont introduits : pour qu'un robot puisse se localiser il a besoin d'effectuer des mesures sur son environnement et/ou sur son déplacement. Pour finir, les différents problèmes de localisation sont présentés avec leurs approches de résolution classiques.

Pour le lecteur intéressé, un complément d'informations sur les problèmes de localisation en robotique mobile peut se trouver dans [[Jensfelt 2001a](#), [Thrun 2005](#)].

1 Considération de l'environnement

Le problème de localisation en robotique mobile suppose connu l'environnement dans lequel se déplace(nt) le(s) robot(s). Les environnements considérés peuvent être divers et variés, avec des problématiques de localisation différentes. De plus l'environnement n'est jamais connu parfaitement, mais est généralement caractérisé à l'aide d'une carte. Là encore il existe plusieurs types de représentation possibles, chacune ayant des avantages et des inconvénients. Cette section présente différentes catégories d'environnements et de cartes classiquement étudiées.

1.1 Différents types d'environnements

Comme précisé précédemment plusieurs types d'environnements peuvent être considérés, chacun avec des problématiques et des difficultés différentes. Il existe plusieurs critères afin de regrouper ces environnements en catégories partageant des problématiques communes. Nous présentons ici deux catégorisations classiques en robotique mobile.

1.1.1 Environnements intérieurs ou extérieurs

Il est possible de différencier les environnements intérieurs des environnements extérieurs.

Les environnements extérieurs représentent l'ensemble des environnements se trouvant à l'extérieur de bâtiments. Ce sont des environnements complexes, possiblement non structurés et généralement de très grande taille. En considérant ce type d'environnement on peut par exemple s'intéresser à la localisation d'un robot mobile (voiture) dans une ville. Le GPS est un capteur très utilisé dans ces conditions. On notera cependant que sa précision n'est pas garantie et qu'il devient difficile à utiliser en milieux denses (forêts, rues bordées par de hauts immeubles, ...). Les télémètres lasers ainsi que les caméras sont aussi des capteurs classiquement utilisés. À l'inverse les capteurs de type sonar ne sont pas particulièrement utiles (à cause de la taille de l'environnement) et l'odométrie est généralement considérée mauvaise, notamment à cause de la nature possiblement hostile du terrain (terrain accidenté, pente, terre molle,...). [[Bailey 2002](#), [Adams 2004](#), [Agrawal 2006](#)] s'intéressent par exemple à la localisation de robots dans des environnements extérieurs.

À l'inverse, la localisation intérieure s'intéresse à la localisation de robots à l'intérieur de bâtiments. En considérant ce type d'environnement on peut par exemple

s'intéresser à la localisation d'un robot mobile (aspirateur) dans un appartement. Ces environnements sont plus structurés que les précédents, de moins grande taille et n'entraînent généralement pas de problèmes de franchissement d'obstacles (les sols sont plats et peu encombrés dans la plupart des cas). Les sonars, télémètres lasers et caméras sont les capteurs les plus utilisés pour localiser les robots dans ce genre d'environnement (avec l'odométrie). On notera que l'utilisation du GPS est difficilement envisageable en milieu clos. [Dulimart 1997, Jensfelt 2001a, Zhou 2007] s'intéressent par exemple à la localisation de robots en environnement intérieur. Les travaux présentés dans cette thèse s'intéressent à ce type d'environnement. Par la suite, les suppositions et notions présentées considéreront des environnements intérieurs, et peuvent ne pas être applicables pour d'autres types de milieux.

1 .1.2 Environnements statiques ou dynamiques

Une autre catégorisation classique consiste à identifier les environnements comme étant statiques ou dynamiques.

Un environnement dynamique est un environnement dont l'état change au cours du temps. Une rue avec des voitures se déplaçant, on encore un bureau avec des personnes le traversant sont des exemples d'environnements dynamiques. La difficulté de ce genre d'environnement est double :

- Pour la cartographie : il ne faut pas tenir compte des éléments mouvants. Lors de la cartographie d'un bureau par exemple, il n'est pas intéressant de placer dans la carte la position d'une personne à un instant t , sachant qu'à l'instant d'après cette donnée sera devenue obsolète, la personne s'étant déplacée.
- Pour la localisation : certains obstacles de l'environnement ne sont pas présents sur la carte (les obstacles dynamiques), ce qui entraîne des disparités entre ce que le robot voit et la carte qu'il utilise pour se localiser.

[Yamauchi 1996, Hahnel 2003, Wolf 2005] s'intéressent aux problèmes de localisation dans des environnements dynamiques.

À l'inverse un environnement statique est un environnement qui ne change pas au cours du temps. Un bâtiment vide est un exemple d'environnement statique. C'est le type d'environnement le plus considéré pour le développement de méthodes de localisation. En effet ce sont des contextes plus simples (comparé aux environnements dynamiques) et maîtrisés (il n'y a pas de déplacement aléatoire d'obstacles). Les environnements statiques sont donc peu réalistes (dans la plupart des cas un robot mobile évolue dans un environnement dynamique) mais représentent un contexte théorique intéressant pour le développement de méthodes de localisation. [Leonard 1991, Thrun 2002] s'intéressent par exemple aux problèmes de localisation dans des environnements statiques.

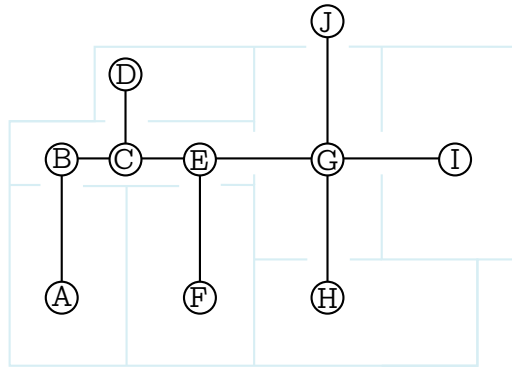


Figure 1.1 – Exemple de carte topologique (en noir).

1.2 Différentes représentations

En plus d'être variés, les environnements peuvent être caractérisés de différentes façons. En effet un environnement n'est jamais connu parfaitement, il est approximé (par l'intermédiaire d'une carte par exemple). Il existe une multitude de méthodes afin de représenter un environnement. [Thrun 1998b] divise les représentations de l'environnement en deux parties : les approches métriques et les approches topologiques. Pour [Chatila 1985] il existe trois modèles différents : géométrique, topologique et sémantique. Le modèle sémantique étant destiné à des décisions de haut niveau, il n'est pas pertinent pour le problème de localisation. Ce qui nous laisse les modèles géométriques et topologiques, rejoignant [Engelson 1992] et son approche métrique/topologique.

1.2.1 Modélisation topologique

L'approche topologique [Kuipers 1991, Kortenkamp 1994, Zimmer 1996] caractérise l'environnement par un graphe s'intéressant aux liens entre différentes régions de l'environnement. Un plan de métro peut par exemple être une carte topologique : il représente les liens entre les différentes stations sans tenir compte de la distance les séparant. La Figure 1.1 présente un exemple de carte topologique. Un bâtiment composé de pièces et de couloirs s'adapte particulièrement bien à ce genre de représentation. L'un des avantages de cette représentation réside dans l'abstraction qui est faite sur la géométrie de l'environnement.

Se localiser à l'aide d'une carte topologique revient à identifier le nœud du graphe correspondant à sa position. Il est donc important que les régions choisies de l'environnement pour la construction du graphe soient facilement identifiables, et ce, en limitant les ambiguïtés. [Ulrich 2000, Blaer 2002] présentent des exemples de localisation manipulant des cartes topologiques.

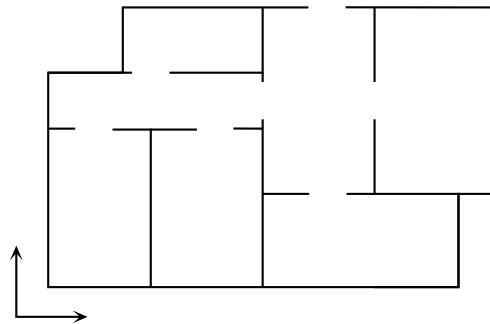


Figure 1.2 – Exemple de carte métrique.

1 .2.2 Modélisation métrique

L'approche métrique s'intéresse quant à elle à la distance séparant les différents éléments de l'environnement [Elfes 1987, Moravec 1988]. Une carte indiquant les sorties de secours d'un bâtiment ainsi que la position des extincteurs correspond par exemple à une représentation métrique. La Figure 1.2 présente un exemple de carte métrique.

1 .2.3 Modélisation hybride

Les deux modélisations présentées précédemment sont complémentaires [Thrun 1996]. Il est possible de considérer conjointement ces deux approches, ce qui se traduit par des modélisations hybrides [Yamauchi 1997, Simhon 1998, Choset 2001]. Ces dernières sont généralement des modélisations topologiques auxquelles on ajoute des données métriques. On pourra cependant noter que [Thrun 1998b] présente une carte hybride essentiellement métrique à laquelle il ajoute des données topologiques afin de faciliter la planification de trajectoires.

1 .3 Deux approches métriques classiques

Les travaux présentés dans ce document s'appuient sur des représentations métriques de l'environnement. Nous présentons ici deux cartes métriques classiquement utilisées.

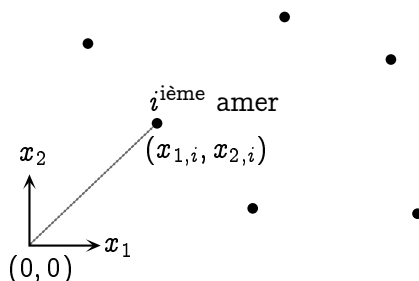


Figure 1.3 – Exemple de carte d'amers. Les amers correspondent ici à des points définis par leurs coordonnées (x_1, x_2) .

1 .3.1 Carte de caractéristiques (carte d'amers)

Une *carte de caractéristiques* (ou *carte d'amers*¹) [Leonard 1992, Stevens 1995] représente un environnement par les coordonnées globales d'amers (points, lignes, polygones...). Le problème de localisation à l'aide d'une carte d'amers revient à un problème d'identification de paramètres : calculer la posture du robot en considérant une liste d'amers (avec leurs coordonnées) ainsi qu'un vecteur d'observations de ces amers (observations réalisées par le robot). La Figure 1.3 illustre le principe d'une carte d'amers et la Figure 1.4 présente un exemple de localisation à l'aide d'une telle carte. Cette méthode de caractérisation est dite paramétrique, le paramètre étant le type d'amer considéré couplé au type de capteur.

1 .3.2 Grille d'occupation

L'inconvénient des méthodes paramétriques réside dans la nécessité d'avoir un modèle explicite pour toutes les informations considérées. Une autre approche métrique, appelée grille d'occupation [Moravec 1985, Elfes 1987], caractérise l'environnement par un ensemble de sous-régions, appelées cellules. Chaque cellule indique la probabilité (entre 0 et 1) de la présence d'obstacles dans la sous-région correspondante. La Figure 1.5 présente un exemple de grille d'occupation.

L'avantage de cette représentation, bien qu'approximative, réside dans son indépendance vis à vis de la nature des obstacles ainsi que du capteur utilisé.

1. En robotique mobile, un *amer* correspond à un élément identifiable de l'environnement servant de repère (murs, objets, angles...).

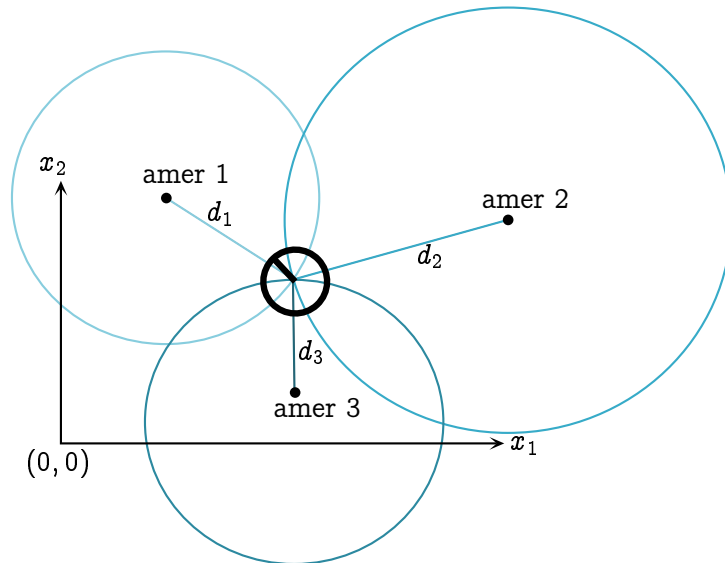


Figure 1.4 – Exemple de localisation à l'aide d'une carte d'amers. Considérons un robot détectant un amer 1 à une distance d_1 . On peut alors en déduire que le robot est sur le cercle de rayon d_1 et ayant pour centre l'amer 1. Le robot détecte aussi l'amer 2 à une distance d_2 . On peut alors restreindre l'estimation de la position du robot aux deux points d'intersection des deux cercles. Si maintenant le robot détecte un troisième amer à une distance d_3 , alors on peut en déduire la position du robot par rapport aux trois amers. C'est le principe de la triangulation. Connaissant les coordonnées globales des amers, et connaissant les coordonnées relatives du robot vis à vis des amers, il est aisé d'en déduire les coordonnées globales du robot.

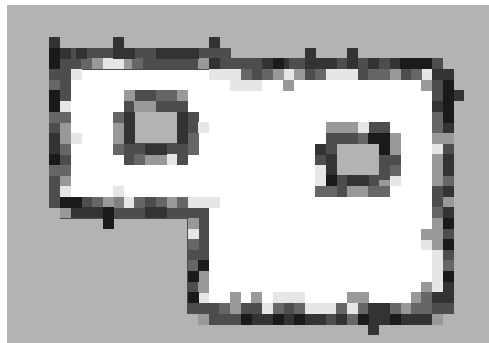


Figure 1.5 – Exemple de grille d'occupation. Les cellules blanches sont des cellules avec une probabilité nulle de contenir un obstacle. Plus la cellule est foncée plus elle a de chance de contenir un obstacle. Les cellules inexplorées sont initialisées avec une probabilité de 0.5, ce qui est représenté par la partie grise sur la figure.

2 Différents capteurs

Afin de pouvoir se localiser un robot doit être capable de mesurer l'environnement qui l'entoure et/ou d'évaluer son déplacement. Pour se faire il doit être doté de capteurs. Il est possible de classer ces capteurs en deux grandes familles : les capteurs *proprioceptifs* et les capteurs *extéroceptifs*.

Les capteurs proprioceptifs sont des capteurs qui vont mesurer l'état interne du robot (le nombre de tours des roues motrices par exemple). En général ces capteurs sont utilisés afin d'évaluer le déplacement du robot. Les capteurs extéroceptifs représentent l'ensemble des capteurs qui mesurent l'environnement dans lequel se déplace le robot. Il existe une multitude de capteurs extéroceptifs : sonars, lasers, caméras...

Cette section présente différents capteurs classiquement utilisés pour les problèmes de localisation en robotique mobile.

2.1 Odométrie

L'odométrie correspond à l'utilisation de capteurs dans le but d'évaluer le déplacement d'un robot [Hardt 1996, Chong 1997, Doh 2003]. L'odométrie permet une évaluation relative du déplacement du robot entre deux instants. C'est une technique très utilisée pour la localisation en robotique mobile, généralement couplée avec des capteurs extéroceptifs [Chenavier 1992, Suksakulchai 2000, Burguera 2005]. Bien que l'odométrie d'un robot soit généralement construite à l'aide de capteurs proprioceptifs (roues codeuses...), on notera qu'il existe aussi des techniques d'odométrie basées sur des capteurs extéroceptifs, comme des caméras [Nister 2004, Agrawal 2006].

2.2 Capteurs de distances

Les capteurs de distances (télémètres²) sont des capteurs très utilisés en robotique mobile. Ils permettent d'avoir une information de distance entre le robot et l'obstacle le plus proche (dans une direction donnée). On peut distinguer deux technologies de capteurs : les capteurs utilisant les ultrasons et les capteurs utilisant la lumière.

2.2.1 Capteurs ultrasons

Principe : le capteur est composé d'un émetteur et d'un récepteur. L'émetteur envoie des ultrasons qui sont réfléchis par l'obstacle puis récupérés par le récepteur.

2. La télémétrie correspond au fait de mesurer la distance d'un objet lointain (à l'aide de lasers, ultrasons, ...). Un capteur permettant de faire de la télémétrie est appelé télémètre. On notera qu'un capteur ultrasons est un télémètre.

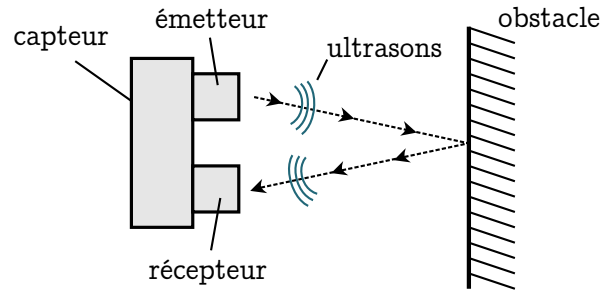


Figure 1.6 – Principe du capteur ultrasons.

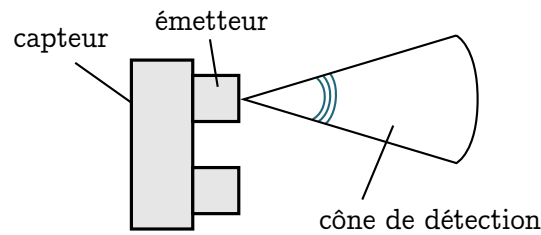


Figure 1.7 – Cône de détection.

En évaluant le temps de propagation de l'onde (temps entre l'émission et la réception), il est possible d'évaluer la distance séparant le capteur de l'obstacle. La Figure 1.6 illustre le principe des capteurs ultrasons.

Une des caractéristiques de ces capteurs est que l'onde émise par l'émetteur n'est pas unidirectionnelle mais prend la forme d'un cône d'environ 30 degrés (Figure 1.7), ce qui représente à la fois un avantage et un inconvénient.

L'avantage correspond à la taille de la zone de détection. En effet pour détecter un obstacle il n'est pas nécessaire qu'il soit exactement en face du capteur mais dans le cône de détection. Ce qui permet notamment de détecter de petits obstacles, comme des pieds de chaise.

Mais inversement la détection de l'obstacle n'est pas précise. En effet l'obstacle est détecté sur un arc de cercle et non pas en un point (Figure 1.8).

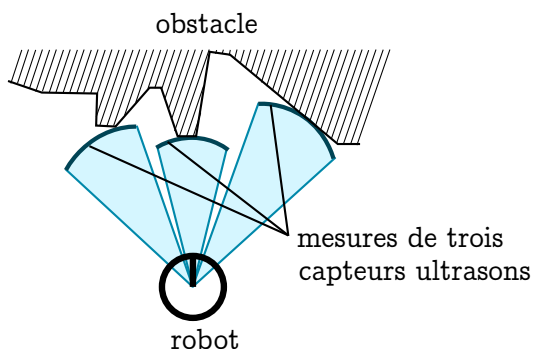


Figure 1.8 – Exemple de jeu de mesures avec trois capteurs ultrasons. Les arcs de cercles foncés illustrent les obstacles détectés.

Ultrasons

Les ultrasons sont des ondes sonores inaudibles pour l'homme. Les sons audibles par l'homme ont des fréquences comprises approximativement entre 20 Hz et 20 KHz. Les ultrasons correspondent aux sons qui ont une fréquence supérieure à 20 KHz. Remarque : les sons ayant une fréquence inférieure à 20 Hz sont appelés infrasons.

Voici différents avantages et inconvénients concernant ce type de capteurs :

Avantages	Inconvénients
<ul style="list-style-type: none"> – Faible coût – Possibilité de détecter de petits obstacles – Indépendant de la lumière ambiante – Permet de détecter les vitres, les miroirs... 	<ul style="list-style-type: none"> – Sensible à la température et à la pression (modification du temps de déplacement de l'onde) – Sensible à la forme et la texture des obstacles – Problème de diaphonie ^a <p><small>^a. La diaphonie correspond à l'interférence d'un premier signal avec un deuxième. Si deux capteurs ultrasons ayant la même fréquence sont utilisés côte à côte, il n'est pas possible de déterminer, lors de la réception d'une onde, lequel en est l'émetteur.</small></p>

[Drumheller 1987, Leonard 1991, Moreno 2002] utilisent par exemple des capteurs ultrasons.

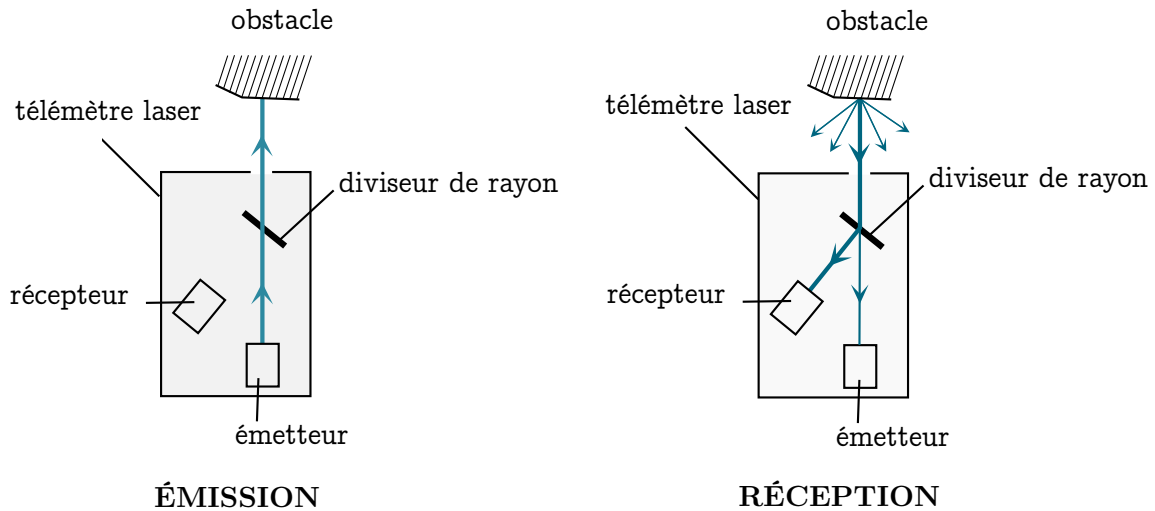


Figure 1.9 – Principe de fonctionnement d'un télémètre laser.

2 .2.2 Télémètres lasers

Le principe d'un télémètre laser est sensiblement le même que celui d'un capteur ultrasons, sauf que cette fois le capteur émet un rayon laser (une onde lumineuse). Le télémètre laser est donc lui aussi composé d'un émetteur et d'un récepteur, et il évalue le déphasage entre l'émission et la réception.

La Figure 1.9 illustre le principe de fonctionnement d'un télémètre laser. On remarque qu'une partie du rayon laser réfléchi emprunte le *même chemin* que lors de l'émission. C'est cette partie qui est interceptée et redirigée vers le récepteur à l'aide d'un diviseur de rayon. La mesure fournie alors par le télémètre laser est ponctuelle. Ce qui signifie que pour détecter un obstacle il faut que ce dernier soit exactement devant le capteur (sur la trajectoire du rayon laser). C'est pourquoi en robotique mobile la plupart des télémètres lasers sont à *balayages*. Ils sont équipés d'un miroir pivotant permettant de prendre plusieurs mesures dans un plan donné (Figure 1.10). La Figure 1.11 propose un exemple de jeu de mesures fourni par un télémètre laser à balayage.

Les télémètres lasers ont généralement une portée plus importante que les capteurs ultrasons³ tout en restant relativement précis (erreurs de quelques centimètres pour une mesure de plusieurs mètres). À l'inverse, ils sont généralement chers et ne peuvent détecter ni les objets transparents (vitres...), ni les objets réfléchissants (miroirs...).

[Borthwick 1993, Arras 1998, Fox 1999, Gutmann 2002] considèrent des télémètres lasers.

3. On notera que sous l'eau, c'est l'inverse.

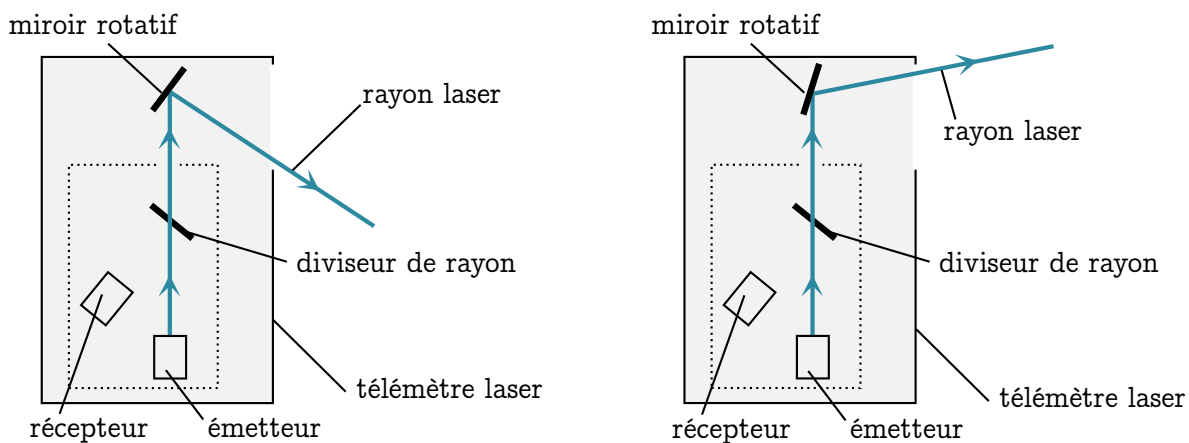


Figure 1.10 – Télémètre laser à balayage. La direction du rayon laser dépend de l'inclinaison du miroir rotatif. Les pointillés correspondent au capteur présenté Figure 1.9.

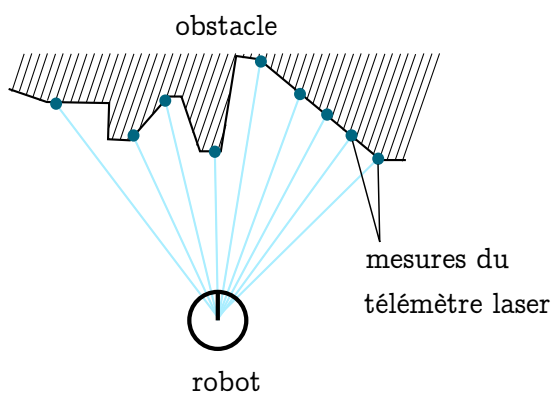


Figure 1.11 – Exemple de jeu de mesures fourni par un télémètre laser à balayage.

2 .3 Caméras

La localisation visuelle (localisation à l'aide d'images) comprend une multitude de problématiques et de méthodes attachées à ces problématiques. Généralement la localisation visuelle consiste à identifier des amers et à se localiser en utilisant par exemple une carte de caractéristiques (Section 1 .3.1) ou encore une carte topologique (Section 1 .2.1). Il est donc possible de diviser la localisation visuelle (dans un paradigme de localisation à l'aide d'amers) en deux tâches principales : l'identification d'amers et la localisation.

2 .3.1 Deux types d'amers

Classiquement, deux types d'amers sont utilisés : les amers artificiels et les amers naturels.

Les amers artificiels correspondent à des points d'intérêts ajoutés à l'environnement afin de faciliter la localisation du robot (Satellites GPS, phares...). Ils correspondent à des *cibles* facilement identifiables par le robot. L'avantage de ce type d'amer réside dans la simplicité d'utilisation. Cependant ces amers doivent être ajoutés à l'environnement ce qui représente une contrainte non négligeable. [Briggs 2000, Jang 2002] utilisent des amers artificiels.

Les amers naturels correspondent à des points d'intérêts naturellement présents dans l'environnement. Pour un environnement intérieur on pourra par exemple considérer les portes, les fenêtres, les coins de pièces... L'utilisation d'amers naturels permet de localiser un robot sans modifier l'environnement. Cependant, l'identification de ces amers dans des environnements complexes reste un problème compliqué. [Sim 1998, Hayet 2002, Dao 2003] s'intéressent au problème de localisation visuelle basée sur des amers naturels. Si certains travaux considèrent des amers bien définis, d'autres approches laissent au robot le soin de choisir les amers naturels les plus intéressants au regard de l'environnement [Thrun 1998a, Ouerhani 2005].

2 .3.2 Avantages et inconvénients

Les caméras sont des capteurs bon marché qui fournissent une quantité d'informations importante. La vision est considérée comme une des sources d'information les plus importantes pour l'homme, et il en va de même pour les robots mobiles qui doivent interagir avec leur environnement.

Le principal problème de ces capteurs réside dans le fait que l'information fournie ne soit pas directement exploitable. À l'inverse des télémètres, dont les informations peuvent être utilisées sans lourd traitement, les images données par une ou plusieurs caméras doivent être traitées, notamment afin d'identifier des amers.

On notera qu'il existe différents types de caméras, chaque caméra permettant de "voir" différents éléments. On peut par exemple citer les caméras classiques RGB⁴ mais aussi les caméras infrarouges [Lee 2007] qui permettent de détecter la lumière infrarouge ou encore les caméras de profondeur [Biswas 2012] qui permettent d'avoir une information de distance. Dans ce dernier cas on se retrouve dans une situation de télémétrie.

3 Différentes approches aux problèmes de localisations

Après avoir présenté les différentes caractérisations de l'environnement ainsi que plusieurs capteurs classiquement utilisés, nous pouvons nous intéresser aux différents problèmes de localisation en robotique mobile. Il existe deux problèmes principaux : le problème de suivi de posture et le problème de localisation globale. Il est aussi possible de différencier un troisième problème, le kidnapping, qui correspond sensiblement à la *fusion* des deux premiers.

Rappelons ici que ce manuscrit s'intéresse aux problèmes de localisation dans le cas d'une modélisation métrique de l'environnement. C'est pourquoi dans cette section nous ne nous intéressons pas aux méthodes utilisant des modélisations alternatives, comme l'approche topologique par exemple.

3.1 Approches probabilistes

La majorité des algorithmes de localisation probabilistes correspondent à des variantes du filtre Bayésien. Les notions théoriques de ce filtre sont présentées Annexe B. L'application directe de ces filtres pour le problème de localisation est appelée *localisation Markovienne*. L'algorithme basique de cette approche est présenté Algorithme 1. Ce dernier est directement dérivé du filtre Bayésien (Annexe B, Algorithme 32). On remarquera cependant que l'algorithme de localisation Markovienne a besoin d'une carte \mathcal{C} en entrée, qui influe plus particulièrement sur l'étape de correction (Ligne 3). Certaines méthodes tiennent aussi compte de la carte lors de l'étape de prédiction (Ligne 2). Tout comme le filtre Bayésien, la localisation Markovienne calcule la croyance courante $bel(\mathbf{q}_t)$ en fonction de la croyance précédente $bel(\mathbf{q}_{t-1})$. On notera que $bel(\mathbf{q}_t)$ est une densité de probabilité qui correspond à la connaissance de la posture du robot \mathbf{q}_t à l'instant t .

3.1.1 Suivi de posture

Pour beaucoup d'applications en robotique mobile, la posture initiale du robot est connue. Pendant sa mission, le robot se déplace et doit alors tenir à jour l'évalua-

4. Red Green Blue - Rouge Vert Bleu

Algorithme 1: Localisation Markovienne

Données : $bel(\mathbf{q}_{t-1}), \mathbf{u}_t, \mathbf{z}_t, \mathcal{C}$

- 1 pour tous les \mathbf{q}_t faire
- 2 // Prédiction
- 3 $\bar{bel}(\mathbf{q}_t) = \int p(\mathbf{q}_t | \mathbf{u}_t, \mathbf{q}_{t-1}, \mathcal{C}) bel(\mathbf{q}_{t-1}) d\mathbf{q};$
- 4 // Correction
- 5 $bel(\mathbf{q}_t) = \eta p(\mathbf{z}_t | \mathbf{q}_t, \mathcal{C}) \bar{bel}(\mathbf{q}_t);$
- 6 fin

Résultat : $bel(\mathbf{q}_t)$.

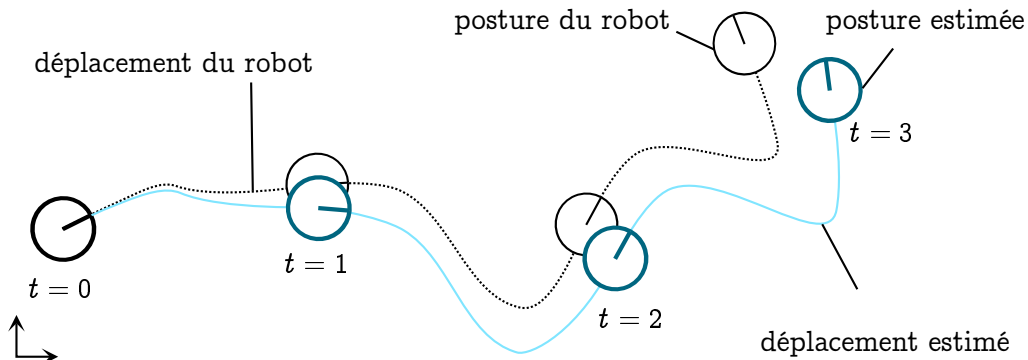


Figure 1.12 – Illustration de la dérive d'un robot.

tion de sa position. En utilisant son odométrie, un robot est capable d'évaluer son déplacement. Seulement plus le robot se déplace, plus les erreurs liées à l'odométrie se propagent et s'intensifient (incertitudes sur la position initiale, incertitudes sur les capteurs utilisés pour l'odométrie...). On dit dans ce cas que le robot dérive. La Figure 1.12 illustre le phénomène de dérive pour un robot mobile.

Afin d'éviter cette dérive les robots sont généralement équipés de capteurs extéroceptifs. Le problème de suivi de posture consiste alors à traiter les données extéroceptives afin de compenser les erreurs de l'odométrie et ainsi éviter la dérive du robot. Pour effectuer cette compensation toute la difficulté réside dans la mise en correspondance du jeu de mesures extéroceptives et de la connaissance de l'environnement (la carte).

En pratique la posture initiale du robot n'est jamais connue parfaitement mais est approximée. Pour la plupart des approches probabilistes cette approximation se caractérise par une Gaussienne centrée sur une posture estimée $\bar{\mathbf{q}}_0$. La croyance de la posture initiale du robot se traduit alors par

$$bel(\mathbf{q}_0) = \det(2\pi\Sigma_{\mathbf{q}_0})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{q}_0 - \bar{\mathbf{q}}_0)^T \Sigma_{\mathbf{q}_0}^{-1} (\mathbf{q}_0 - \bar{\mathbf{q}}_0)\right), \quad (1.1)$$

ou encore

$$bel(\mathbf{q}_0) \sim \mathcal{N}(\bar{\mathbf{q}}_0, \Sigma_{\mathbf{q}_0}), \quad (1.2)$$

avec $\Sigma_{\mathbf{q}_0}$ la matrice de covariance de l'incertitude sur la posture initiale. $bel(\mathbf{q}_0)$ est une densité de probabilité qui correspond à la connaissance sur la posture initiale du robot \mathbf{q}_0 .

Le filtre de Kalman, et plus particulièrement le filtre de Kalman étendu, correspond à la base de la plupart des implémentations de la localisation Markovienne dans un contexte de suivi de posture. Les notions théoriques du filtre de Kalman et son extension sont présentées Annexe B. Le lecteur intéressé trouvera dans [Thrun 2005] un exemple ainsi que des explications plus approfondies concernant la localisation à l'aide du filtre de Kalman étendu.

On pourra par exemple noter que [Jensfelt 1999, Wijk 2000, Jensfelt 2001b] ou encore [Ivanjko 2004, Lingemann 2005] utilisent des méthodes basées sur le filtre de Kalman étendu pour traiter le problème de suivi de posture en robotique mobile.

3.1.2 Localisation globale

Les techniques de suivi de posture peuvent être utilisées à condition de connaître la posture initiale du robot. Cependant dans certaines situations, ou afin de rendre un robot complètement autonome, la posture initiale du robot n'est pas connue. Estimer la posture courante d'un robot sans connaissance a priori sur sa posture initiale correspond au problème de localisation globale (la posture du robot doit être trouvée de façon globale dans l'environnement, à la différence du suivi de posture ou la posture courante du robot est estimée localement, en tenant compte de la posture précédente).

Le problème de localisation globale est plus compliqué que celui du suivi de posture. La complexité augmente notamment avec la taille de l'environnement ainsi qu'avec le nombre de symétries présentes dans l'environnement. Là encore les solutions probabilistes se sont imposées.

Si pour le problème de suivi de posture le filtre de Kalman représente la base de la plupart des solutions classiques, pour le problème de localisation globale il existe deux approches probabilistes répandues : la localisation de type *Monte Carlo* (MCL⁵) et la localisation par grille.

Localisation par grille : l'idée de la localisation par grille est de décomposer l'espace d'états (les postures possibles pour le robot) en une grille et d'affecter à chaque cellules de cette grille la probabilité que le robot soit dans la configuration correspondante. La croyance $bel(\mathbf{q}_t)$ est alors définie par

$$bel(\mathbf{q}_t) = \{p_{k,t}\} \quad (1.3)$$

5. Monte Carlo Localization - Localisation Monte Carlo

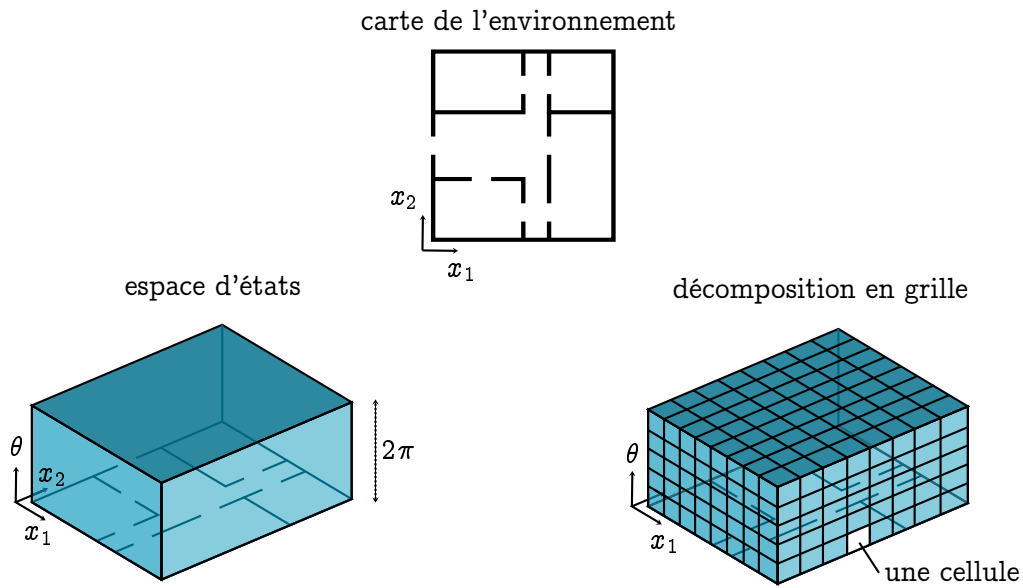


Figure 1.13 – Exemple de décomposition en grille. À chaque cellule on affecte la probabilité que le robot soit dans la configuration correspondante (position (x_1, x_2) et orientation θ). On notera que l'espace d'états correspond à l'ensemble des positions et orientations potentielles pour le robot.

avec $p_{k,t}$ la probabilité que le robot soit dans la configuration correspondant à la $k^{\text{ième}}$ cellule à l'instant t . L'algorithme basique de la localisation par grille décompose l'espace d'états en une grille statique (qui ne change pas au cours du temps) composée de cellules de mêmes tailles. La figure 1.13 présente un exemple de grille. On notera que la précision de la grille (la taille des cellules) a un impact majeur sur le résultat de la localisation : une grille trop grossière ne permettra pas de localiser le robot correctement, à l'inverse une grille trop fine sera gourmande en temps de calcul. On notera que certains utilisent des grilles adaptatives afin d'avoir un compromis entre précision et temps de calcul.

Pour un problème de localisation globale la posture initiale du robot n'est pas connue. La croyance $bel(q_0)$ est alors initialisée en donnant la même probabilité à toutes les cellules $\left(\frac{1}{\text{nombre de cellules}}\right)$. Il est même possible d'affiner cette initialisation en ne considérant que les cellules physiquement possibles (une cellule impossible étant une cellule correspondant par exemple à un obstacle dans l'environnement : le robot ne peut pas être *dans* un mur.)

[Burgard 1996, Burgard 1999, Fox 1999] utilisent des méthodes de localisations à l'aide de grilles.

Localisation Monte Carlo (MCL) : l'idée de la localisation Monte Carlo est de

représenter la croyance de la posture du robot par un ensemble de particules. Cette méthode, parmi les plus utilisées en robotique mobile, est basée sur une approche de type filtre particulaire (Annexe B). L'algorithme de base représente la croyance $bel(\mathbf{q}_t)$ par un ensemble de M particules $\mathcal{X}_t = \{\mathbf{x}_t^{[1]}, \mathbf{x}_t^{[2]}, \dots, \mathbf{x}_t^{[M]}\}$, chaque particule correspondant à une posture possible pour le robot. À chaque particule est affecté un poids, qui correspond à la vraisemblance que la posture du robot corresponde à la particule. L'Algorithme 2 présente la méthode de localisation Monte Carlo. On peut noter qu'il est basé sur l'algorithme des filtres à particules (Annexe B, Algorithme 36), auquel on ajoute la prise en compte du modèle de l'environnement (la carte C).

Algorithme 2: Localisation Monte Carlo

Données : $\mathcal{X}_{t-1}, \mathbf{u}_t, \mathbf{z}_t, C$

- 1 $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset;$
- 2 **pour** $m = 1$ à M **faire**
- 3 générer $\mathbf{x}_t^{[m]} \sim p(\mathbf{q}_t \mid \mathbf{u}_t, \mathbf{x}_{t-1}^{[m]});$
- 4 $w_t^{[m]} = p(\mathbf{z}_t \mid \mathbf{x}_t^{[m]}, C);$
- 5 $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle \mathbf{x}_t^{[m]}, w_t^{[m]} \rangle;$
- 6 **fin**
- 7 **pour** $m = 1$ à M **faire**
- 8 générer $\mathbf{x}_t^{[i]}$ en fonction de $\bar{\mathcal{X}}_t;$
- 9 $w_t^{[i]} = p(\mathbf{z}_t \mid \mathbf{x}_t^{[i]});$
- 10 ajouter $\mathbf{x}_t^{[i]}$ à $\mathcal{X}_t;$
- 11 **fin**

Résultat : $\mathcal{X}_t.$

La croyance initiale $bel(\mathbf{q}_0)$ est cette fois initialisée par un jeu de particules générées aléatoirement dans l'environnement. À chacune de ces particules initiales, on affecte un facteur d'importance uniforme $\left(\frac{1}{M}\right)$, avec M le nombre de particules générées.

Il existe plusieurs variantes et améliorations pour cet algorithme (Mixture-MCL [Thrun 2001], Adaptive-MCL [Gutmann 2002], Self-adaptive MCL [Zhang 2009]...).

[Dellaert 1999, Fox 1999, Thrun 2001, Fu 2011] utilisent des méthodes basées sur la localisation Monte Carlo. On peut notamment noter qu'en plus de traiter le problème de localisation globale, la plupart des méthodes MCL permettent de s'intéresser au problème du kidnapping [Thrun 2000, Burchardt 2011].

3.2 Approches ensemblistes

Les approches ensemblistes du problème de localisation représentent une alternative intéressante aux approches probabilistes. Bien que, pour la plupart, plus lentes

que les méthodes probabilistes, les méthodes ensemblistes ont l'avantage d'être robustes et garanties.

3 .2.1 Présentation

Pour les approches ensemblistes, les différentes variables (posture du robot, incertitudes des capteurs...) ne sont non plus estimées par des densités de probabilités (comme c'était le cas pour les approches probabilistes) mais par des ensembles bornés. L'analyse par intervalles (Annexe A) a notamment déjà été démontrée comme étant particulièrement adaptée pour traiter le problème d'estimation d'état (de localisation) dans un contexte ensembliste [Jaulin 2001a, Delafosse 2005, Gning 2006]. C'est cette approche qui va nous intéresser dans la suite de ce document.

Pour l'approche ensembliste du problème de localisation l'objectif est de calculer une boîte $[\mathbf{q}_t]$ (la plus petite possible) contenant la posture \mathbf{q}_t du robot à l'instant t . Pour ce faire, toutes les incertitudes et erreurs sont supposées bornées. Ce qui en d'autres termes permet d'associer une boîte $[\mathbf{z}_t]$ au vecteur de mesures \mathbf{z}_t et une boîte $[\mathbf{u}_t]$ au vecteur de contrôles \mathbf{u}_t , avec $\mathbf{z}_t \in [\mathbf{z}_t]$ et $\mathbf{u}_t \in [\mathbf{u}_t]$. Il est alors possible d'en déduire la fonction dynamique ensembliste suivante

$$[\mathbf{q}_{t+1}] = [f]([\mathbf{q}_t], [\mathbf{u}_t]), \quad (1.4)$$

avec $[f]$ une fonction d'inclusion pour f , la fonction caractérisant la dynamique du robot (Introduction, Section 2.1). Les fonctions d'inclusion sont présentées Annexe A Section 1.3.4. En utilisant l'arithmétique des intervalles, connaissant une boîte $[\mathbf{q}_t]$ contenant la posture à l'instant t , il est alors possible de calculer une boîte $[\mathbf{q}_{t+1}]$ contenant la posture à l'instant $t + 1$. Le contexte ensembliste n'échappe pas au phénomène de dérive. En effet en n'utilisant que l'équation précédente (ce qui revient à n'utiliser que les données odométriques), les boîtes résultantes seront de plus en plus volumineuses au fil des déplacements du robot. Le principe est le même que pour l'approche probabiliste, les incertitudes sur la posture du robot vont se propager et s'amplifier, ce qui va entraîner l'évaluation de boîtes plus importantes. La Figure 1.14 illustre la version ensembliste du phénomène de dérive.

Il est important de noter ici qu'il n'y a pas de sens à privilégier une posture plutôt qu'une autre au sein d'une boîte (comme solution au problème de localisation). Si pour l'approche probabiliste le maximum de la densité de probabilité de la croyance correspond à la posture la plus probable pour le robot, il n'est pas possible d'identifier une posture particulière au sein d'une boîte dans un contexte ensembliste.

3 .2.2 Différentes méthodes ensemblistes

On présente ici deux approches ensemblistes (basées sur l'analyse par intervalles), la première s'intéresse au problème de suivi de posture et la deuxième traite le problème de localisation globale.

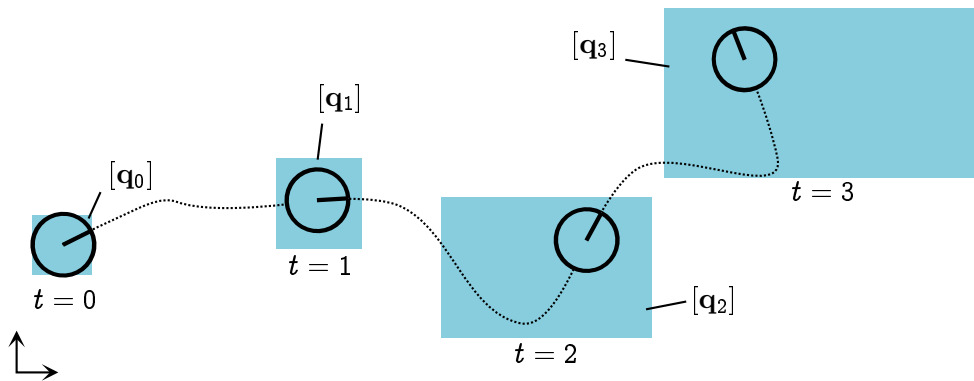


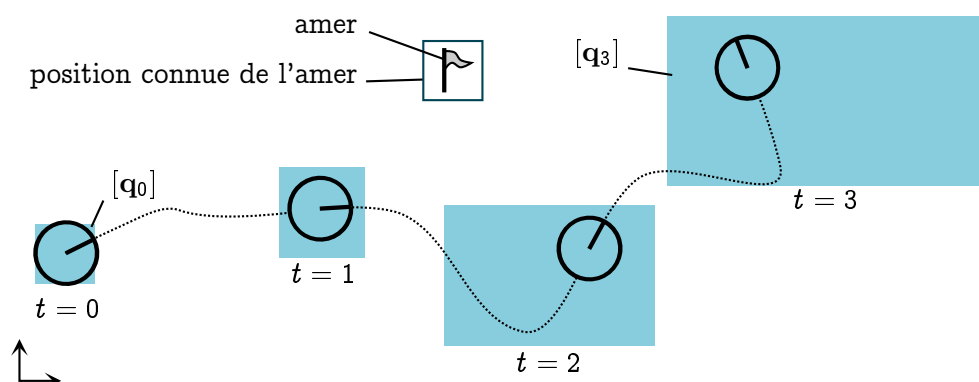
Figure 1.14 – Version ensembliste de la dérive.

Problème de suivi de posture ensembliste. L'idée de la méthode présentée ici est basée sur le principe de la propagation de contraintes (Annexe A, Section 2.2.4). Schématiquement l'objectif est de réussir à contracter une boîte sur trajectoire du robot (réduire sa taille) à un instant t et de propager cette contraction sur les autres boîtes de la trajectoire.

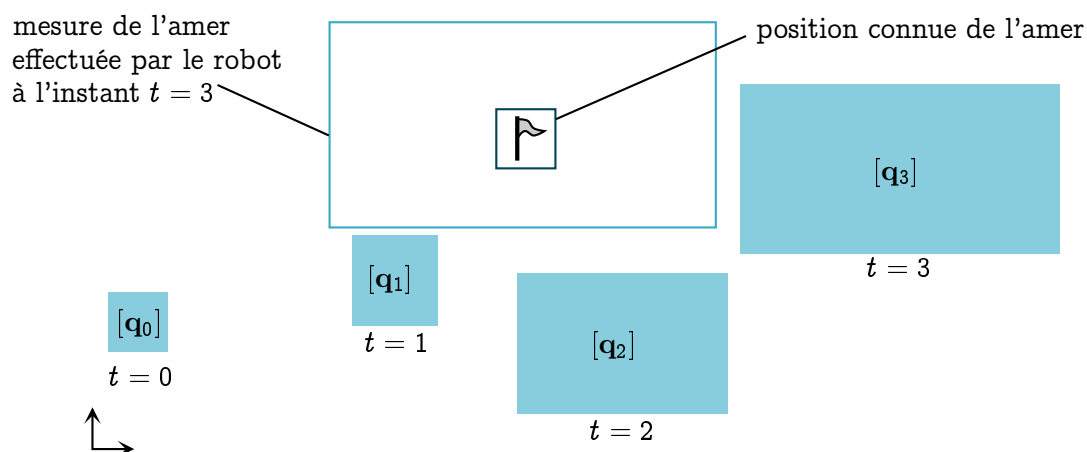
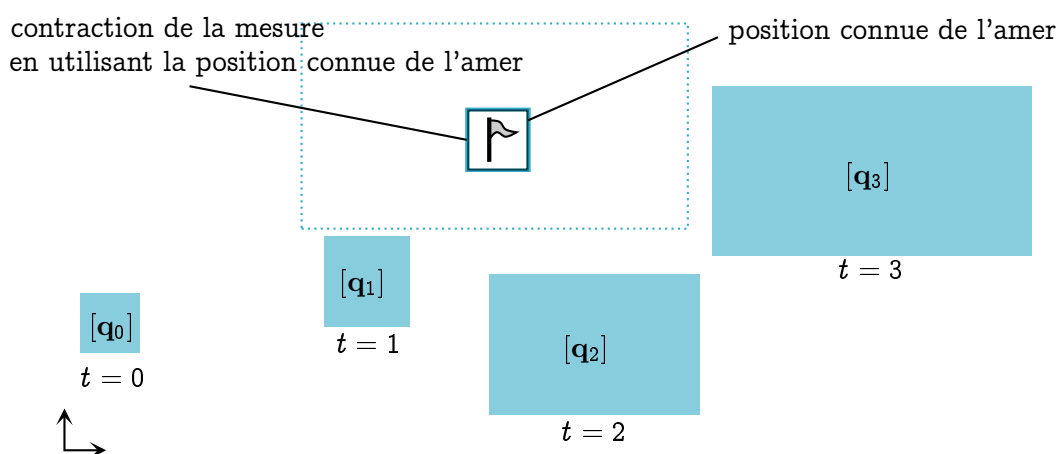
La Figure 1.15 illustre le principe du suivi de posture ensembliste. Figure 1.15a le robot se déplace, évalue sa posture actuelle en fonction de la précédente à l'aide de l'odométrie. Il est en situation de dérive, les postures évaluées sont de moins en moins précises (les boîtes sont de plus en plus grosses). À l'instant $t = 3$ le robot détecte un amer (Figure 1.15b). L'incertitude sur la posture du robot influence la précision de la mesure, ce qui justifie la taille de la boîte résultante de la mesure (Figure 1.15b). La position de l'amer est connue (aux incertitudes près), il est donc possible de contracter la mesure faite par le robot sur la connaissance de la position de l'amer. En d'autres termes on veut rendre la mesure du robot cohérente avec la connaissance de l'environnement. C'est ce qui est illustré Figure 1.15c. La contraction de la mesure correspond à un gain d'information et, à l'aide de l'analyse par intervalles, il est possible de propager cette information sur la connaissance de la posture du robot pour contracter la boîte courante (Figure 1.15d). Cette contraction entraîne une amélioration de la connaissance de la posture du robot à l'instant $t = 3$. Les différentes postures de la trajectoire du robot étant reliées entre elles à l'aide de la fonction itérative, il est possible de propager l'amélioration de la boîte à l'instant $t = 3$, sur les autres boîtes de la trajectoire (Figure 1.15e). On arrive ainsi à maintenir une certaine précision sur la connaissance de la posture du robot et donc à éviter la dérive du robot (Figure 1.15f).

C'est notamment ce principe qui est utilisé dans [Jaulin 2009].

Problème de localisation globale ensembliste. Un des atouts de l'analyse par intervalle est qu'elle permet de s'intéresser au problème de localisation globale de façon garantie (sans approximation comme c'est le cas pour la localisation par grille ou pour les méthodes de Monte Carlo par exemple).

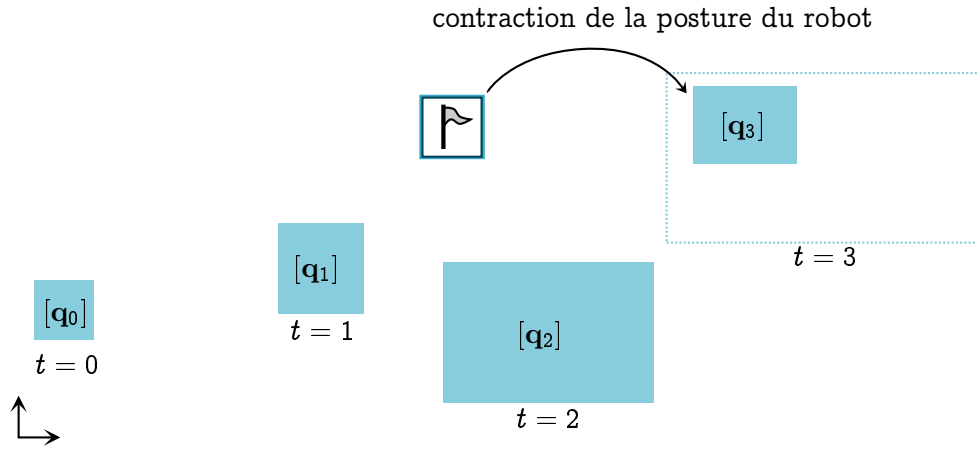


(a) Dérive du robot.

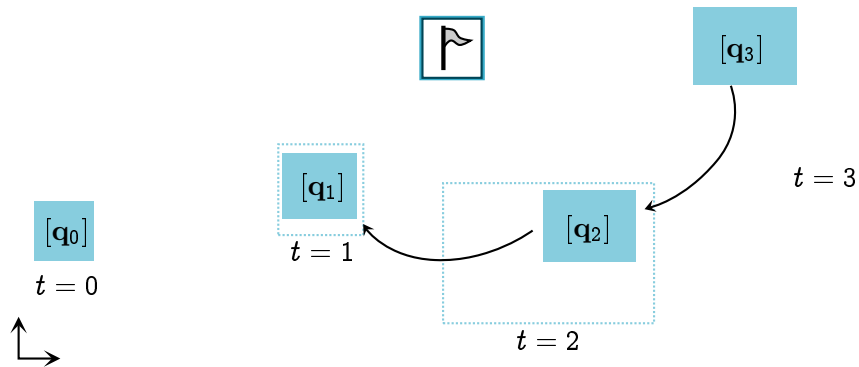
(b) Détection de l'amer à l'instant $t=3$.

(c) Contraction de la mesure de l'amer sur la position connue de l'amer.

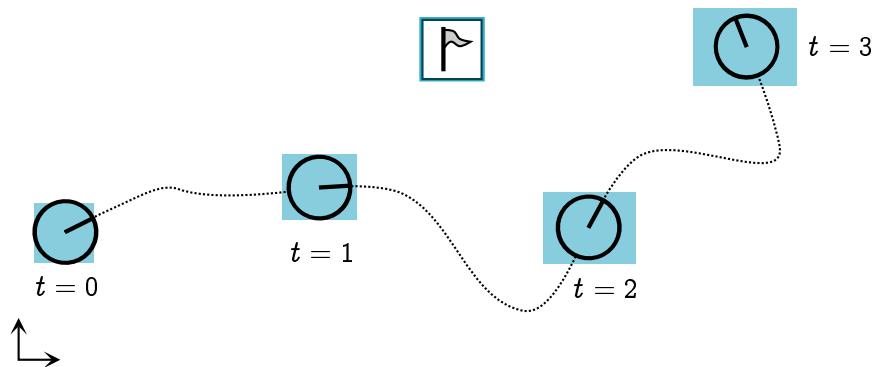
Figure 1.15 – Exemple de suivi de posture ensembliste.



(d) Contraction de la posture du robot en utilisant l'information gagnée lors de la contraction de la mesure.



(e) Propagation de la contraction sur les postures précédentes.



(f) On évite ainsi la dérive du robot en maintenant une connaissance *précise* de sa posture.

Figure 1.15 – Exemple de suivi de posture ensembliste (suite).

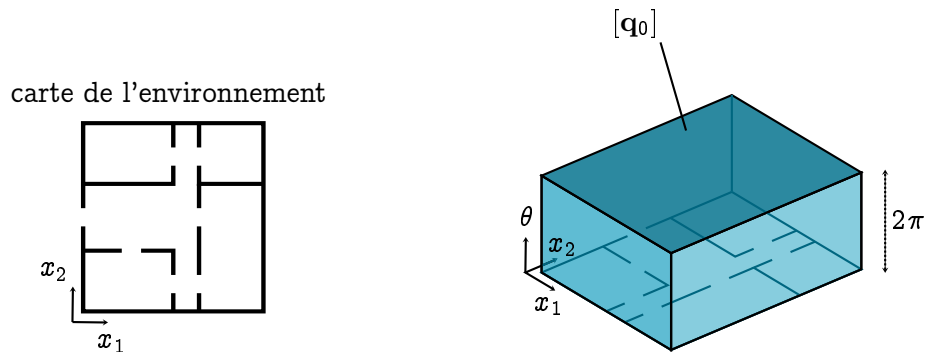


Figure 1.16 – Initialisation ensembliste pour un problème de localisation globale.

Pour un problème de localisation globale, on n'a aucune connaissance à priori sur la posture initiale du robot. La posture q_0 est donc initialement caractérisée par une boîte $[q_0]$ englobant toutes les postures possibles pour le robot (L'ensemble de l'environnement pour la position et $[0, 2\pi]$ pour l'orientation, Figure 1.16). L'idée est alors de supprimer des régions de $[q_0]$ non compatibles avec le jeu de mesures extéroceptif.

La Figure 1.17 présente un exemple simple de localisation globale. L'environnement considéré est présenté Figure 1.17a. Il est composé de deux amers dont les positions sont connues, aux incertitudes bornées près. La première chose à faire est d'initialiser la posture initiale du robot. N'ayant aucune information à priori, la posture initiale q_0 est caractérisée par une boîte $[q_0]$ englobant toutes les postures possibles (Figure 1.17b). Supposons maintenant que le robot fasse un jeu de mesure : il détecte l'amer 1 à une distance maximale d_1 et l'amer 2 à une distance maximale d_2 . En d'autres termes le robot ne peut pas être au delà d'une distance d_1 de l'amer 1 ni au delà d'une distance d_2 de l'amer 2. On peut alors créer des pavés compatibles avec ces deux observations (Figure 1.17c). La dernière étape revient alors à rendre $[q_0]$ compatible avec ces deux mesures (Figure 1.17d).

Il existe différentes techniques pour supprimer les valeurs d'un pavé non compatibles avec les mesures. [Seigneur 2005] utilise par exemple une méthode basée sur l'algorithme SIVIA (Annexe A, Section 2.1.2).

On pourra aussi noter le développement de méthodes hybrides entre les approches probabilistes et ensemblistes comme [Ashokaraj 2004, Abdallah 2008], ou encore [Nassreddine 2010, Jaulin 2011b].

Dans les chapitres qui suivent nous présentons des approches ensemblistes originales au problème de localisation en robotique mobile.

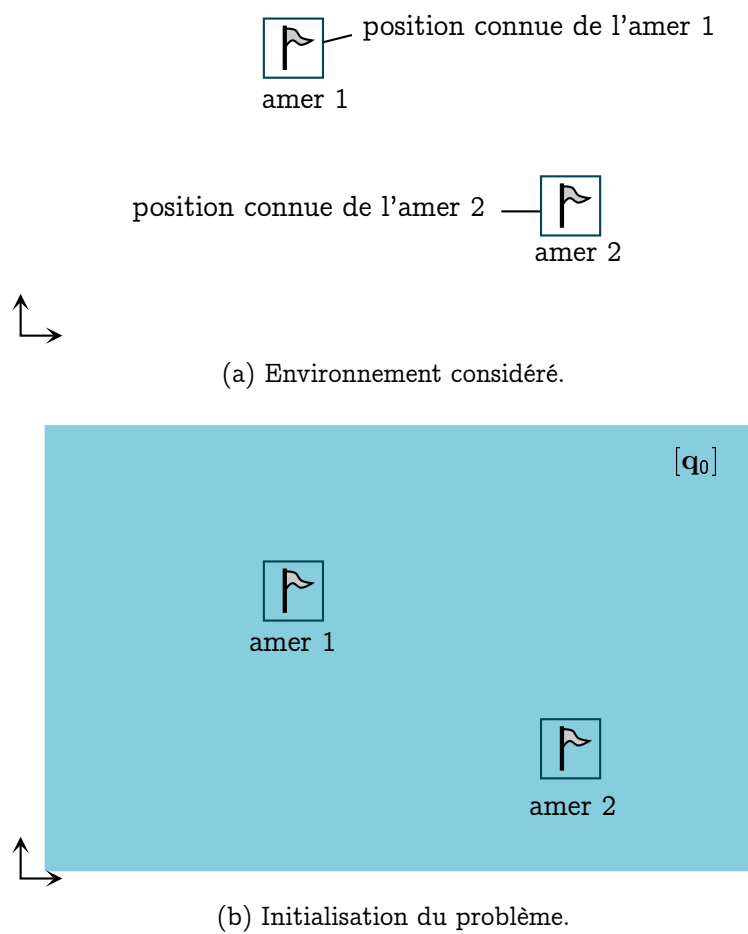
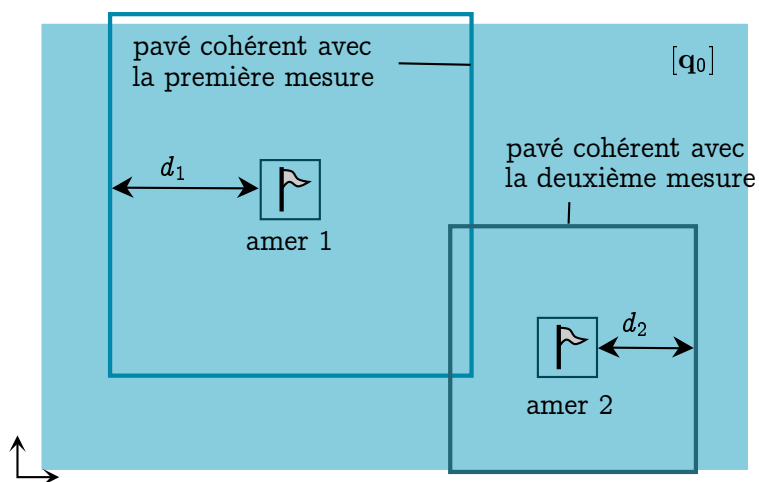
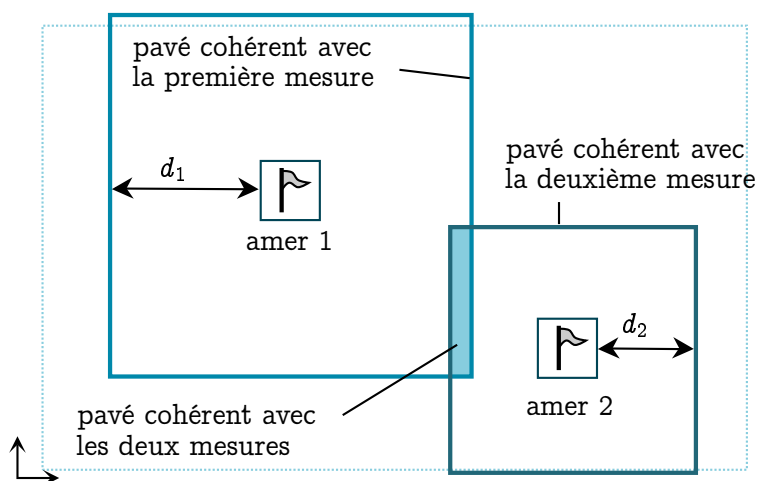


Figure 1.17 – Exemple simple de localisation globale.



(c) Détection de l'amer 1 à une distance d_1 et de l'amer 2 à une distance d_2 .



(d) Contraction de $[q_0]$ pour rendre le pavé compatible avec les deux mesures.

Figure 1.17 – Exemple simple de localisation globale (suite).

3.3 Conclusion

Comme nous avons pu le voir dans ce chapitre il existe une multitude de problèmes de localisation en robotique mobile, et encore plus de solutions à ces problèmes. Toutes les solutions existantes ont leurs avantages et leurs inconvénients. Dans ce document nous nous intéressons particulièrement à la localisation de robots mobiles en environnement clos (à l'intérieur de bâtiments). Afin de traiter ce problème, nous présentons des approches ensemblistes originales. Au fil de ces pages, nous nous intéresserons aux problèmes de localisation globale, du suivi de posture et du kidnapping. À chaque fois nous manipulerons des caractérisations métriques de l'environnement. Il est supposé ici que tous les robots possèdent une technique d'odométrie, plus ou moins précise. Cette supposition n'est pas contraignante car dans la plupart des cas, un robot peut évaluer son déplacement maximal entre deux pas de temps en fonction de sa vitesse maximale. L'objectif de ces travaux est d'identifier les intérêts d'une approche ensembliste dans un tel contexte de localisation, et plus particulièrement, les avantages que présente l'analyse par intervalles et ses outils afin de résoudre ces problèmes.

Méthode ensembliste de localisation globale

Sommaire

1	Présentation du problème	38
1.1	Robot	38
1.2	Environnement	39
1.3	Objectif	41
2	Présentation de la méthode	43
2.1	Formalisation en problème de satisfaction de contraintes	43
2.2	Présentation des contracteurs	48
2.3	Algorithme IAL	58
3	Expérimentations et comparaisons	63
3.1	Expérimentations avec un ordinateur distant	63
3.2	Expérimentations avec un MiniRex	66
3.3	Comparaison MCL/IAL	71
3.4	Conclusion	76

Dans ce chapitre, nous présentons une méthode de localisation ensembliste, appelée IAL (Interval Analysis Localization - Localisation à l'aide de l'Analyse par Intervalles), qui s'intéresse au problème de localisation globale en robotique mobile.

Il existe déjà des méthodes de localisation ensemblistes. Nous pouvons par exemple noter [Jaulin 2009] utilisant une approche de type SIVIA¹ (Annexe A, Section 2.1.2) pour la localisation de sous-marins, ou encore [Seigneur 2005] pour les robots mobiles terrestres. D'autres optent pour des solutions mixtes : combinaison d'outils probabilistes et ensemblistes [Ashokaraj 2004].

1. Set Inversion Via Interval Analysis - Inversion ensembliste à l'aide de l'analyse par intervalles.

L'algorithme proposé ici se démarque des méthodes existantes notamment par le type de carte considérée. En effet il considère une grille d'occupation obtenue à l'aide d'une méthode de SLAM, ne nécessitant ni vectorisation (à la différence de [Seigneur 2005]) ni modification/traitement supplémentaire (par un opérateur par exemple, comme cela peut être fait dans [Jaulin 2009] pour la détection d'amers). Notre approche consiste en une méthode ensembliste entièrement déterministe et répétable, ne manipulant aucun outil probabiliste. Ainsi les résultats obtenus sont garantis, relativement aux incertitudes des capteurs et au nombre de valeurs aberrantes.

Ces travaux ont été réalisés lors du défi CAROTTE (CARTographie par RO-bot d'un TErritoire), organisé par la Direction Générale de l'Armement (DGA) et l'Agence Nationale de la Recherche (ANR). Nous avons en effet utilisé les ressources et données du consortium Cart-O-matic (composé du LISA² et du LORIA³), lauréat en 2012 de la troisième et dernière édition du défi. L'objectif de ce défi était de créer une plate-forme robotique capable d'évoluer dans un environnement inconnu, de cartographier cet environnement, d'identifier un certain nombre d'objets/textures et de les localiser dans la carte réalisée. C'est dans ce contexte que le consortium a, entre autre, mis au point le robot MiniRex⁴ (Figure 2.1) ainsi que la méthode de cartographie et localisation simultanées nommée *Slam-O-matic*. On notera que les données utilisées dans la partie expérimentation (Section 3) proviennent d'un robot MiniRex, et que les cartes manipulées ont été réalisées en utilisant l'algorithme Slam-O-matic.

Dans un premier temps nous présentons le problème de localisation considéré Section 1, puis nous introduisons la méthode développée Section 2 pour finalement présenter un certain nombre d'expérimentations menées dans le but de valider la méthode, Section 3.

1 Présentation du problème

1.1 Robot

Comme précisé précédemment nous nous intéressons aux robots mobiles terrestres (Figure 2.1). Un robot de ce type est caractérisé par les équations dynamiques discrètes suivantes

$$\mathbf{q}_{t+1} = f(\mathbf{q}_t, \mathbf{u}_t), \quad (2.1)$$

$$\mathbf{z}_t = g_{\mathcal{E}}(\mathbf{q}_t). \quad (2.2)$$

2. Laboratoire d'Ingénierie des Systèmes Automatisés.

3. Laboratoire lorrain de recherche en informatique et ses applications.

4. MINI Robot d'EXploration.



Figure 2.1 – Robot MiniRex.

La posture du robot, notée $\mathbf{q}_t = (x_{1_t}, x_{2_t}, \theta_t)$, est définie par sa position $\mathbf{x}_t = (x_{1_t}, x_{2_t})$ et son orientation θ_t dans l'environnement \mathcal{E} à l'instant t . L'environnement $\mathcal{E} \in \mathbb{R}^2$ est un espace à deux dimensions à l'intérieur duquel le robot se déplace. La fonction f caractérise la dynamique du robot et le vecteur \mathbf{u}_t correspond au vecteur de contrôles appliqué entre les instants $t - 1$ et t . Nous rappelons ici la fonction dynamique f considérée :

$$\begin{pmatrix} x_{1_{t+1}} \\ x_{2_{t+1}} \\ \theta_{t+1} \end{pmatrix} = \begin{pmatrix} x_{1_t} \\ x_{2_t} \\ \theta_t \end{pmatrix} + \begin{pmatrix} \sin(\theta_t)\Delta x_t \\ \cos(\theta_t)\Delta x_t \\ \Delta \theta_t \end{pmatrix}, \quad (2.3)$$

avec $\mathbf{u}_t = (\Delta x_t, \Delta \theta_t)^T$, Δx_t modélisant la distance parcourue entre \mathbf{x}_t et \mathbf{x}_{t+1} , et $\Delta \theta_t$ modélisant la différence d'orientation entre θ_t et θ_{t+1} (Introduction, Figure 2). On notera que Δx_t et $\Delta \theta_t$ sont estimées par l'odométrie.

Le vecteur $\mathbf{z}_t = \{\mathbf{w}_{1,t}, \dots, \mathbf{w}_{i,t}, \dots, \mathbf{w}_{n_w,t}\}$ correspond aux mesures faites par le robot à l'instant t . Le robot effectue n_w mesures autour de lui. Ces mesures dépendent directement de la posture du robot \mathbf{q}_t ainsi que de l'environnement \mathcal{E} . Nous considérons ici un télémètre laser comme capteur extéroceptif. Une mesure \mathbf{w}_i est alors caractérisée par une distance y_i et un angle γ_i . C'est ce qui est présenté Figure 2.2.

1 .2 Environnement

Nous considérons ici un environnement intérieur (Figure 2.3), noté \mathcal{E} , caractérisé par une grille d'occupation binaire. La grille d'occupation caractérisant l'environnement est dite binaire, car à l'inverse des grilles d'occupation présentées précé-

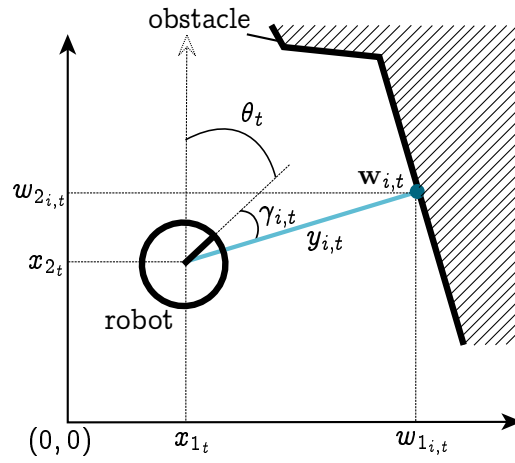
Figure 2.2 – Illustration d'une mesure $w_{i,t}$.

Figure 2.3 – Exemple d'environnement : les arènes du défi CAROTTE.

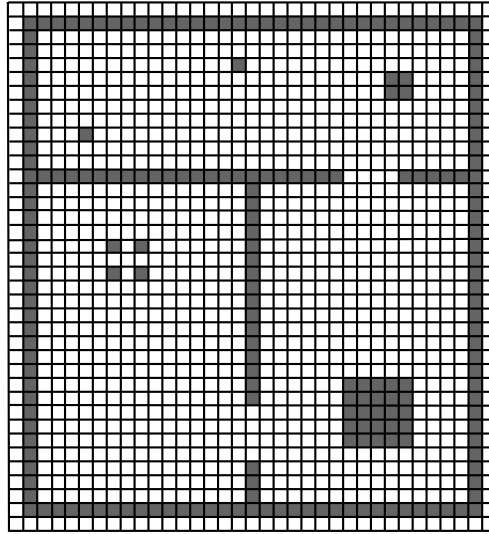


Figure 2.4 – Exemple de grille d’occupation binaire. Les cellules blanches correspondent à des zones de l’environnement ne contenant aucun obstacle.

demment (Chapitre 1), les cellules de la grille considérée ici n’ont que deux valeurs possibles : 0 ou 1.

Soit \mathcal{E}_G une grille composée de $n \times m$ cellules $c_{i,j}$, $i = 1, \dots, n$ et $j = 1, \dots, m$. À chaque cellule on associe un poids $g_{i,j}$ avec

$$g_{i,j} = \begin{cases} 0 & \text{si la cellule correspond à une zone ne contenant aucun obstacle,} \\ 1 & \text{sinon.} \end{cases} \quad (2.4)$$

On notera que la grille \mathcal{E}_G correspond à une version discrète de \mathcal{E} . La Figure 2.4 présente un exemple de grille d’occupation binaire composée de 35×38 cellules.

1 .3 Objectif

Nous nous intéressons ici au problème de localisation globale : le robot doit être capable d’évaluer sa posture courante \mathbf{q}_t sans aucune information à priori sur sa posture initiale \mathbf{q}_0 .

Nous nous plaçons dans un contexte à erreurs bornées. Ce qui permet d’associer à chaque donnée capteur un intervalle contenant la valeur mesurée de manière garantie. Considérons par exemple une mesure $w_{i,t}$. Cette mesure est définie par une distance $y_{i,t}$ et un angle $\gamma_{i,t}$ (les données brutes fournies par le télémètre laser). On note $\hat{y}_{i,t}$ et $\hat{\gamma}_{i,t}$ les vérités terrain correspondants à la mesure (on fait la distinction entre les mesures -les données du capteur- et les vérités terrains). Dans le meilleur des

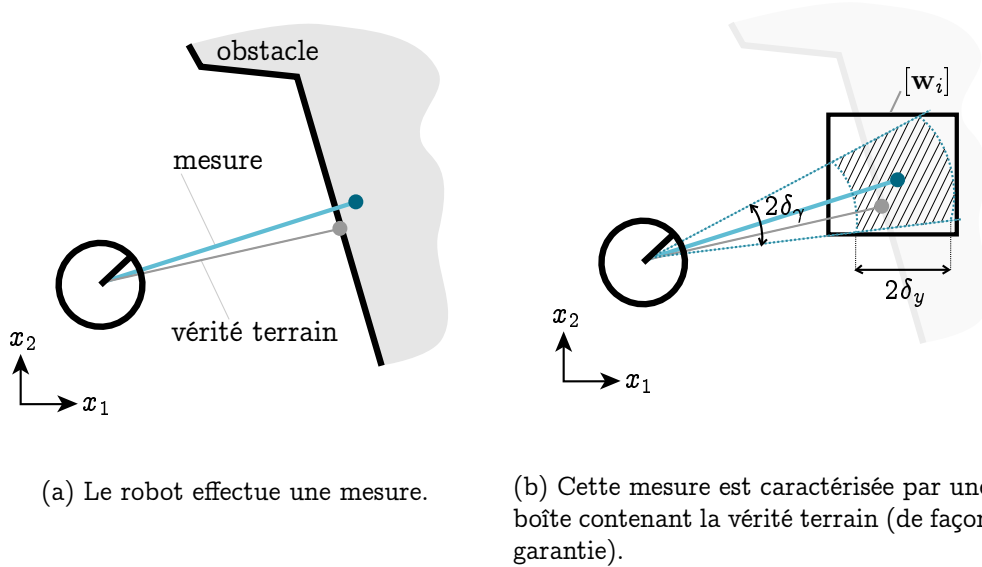


Figure 2.5 – Évaluation garantie d'une mesure sous l'hypothèse d'erreurs bornées.

cas, si le capteur est performant, les mesures effectuées et les vérités terrain sont sensiblement égales : $\hat{\gamma}_{i,t} \approx \gamma_{i,t}$ et $\hat{y}_{i,t} \approx y_{i,t}$. Cependant, on n'a pas la relation d'égalité $\hat{\gamma}_{i,t} = \gamma_{i,t}$ ni $\hat{y}_{i,t} = y_{i,t}$! Cette différence entre les valeurs mesurées et les vérités terrain s'explique par le fait que le capteur ne soit pas parfait.

Dans un contexte à erreurs bornées, il est possible de définir une erreur maximale de distance δ_y et d'angle δ_γ , relatives au télémètre considéré. Là où les approches probabilistes mettent en place des densités de probabilités, en considérant des erreurs bornées, il devient possible de déduire deux intervalles $[y_{i,t}] = [y_{i,t} - \delta_y, y_{i,t} + \delta_y]$ et $[\gamma_{i,t}] = [\gamma_{i,t} - \delta_\gamma, \gamma_{i,t} + \delta_\gamma]$ de façon à ce que $y_{i,t} \in [y_{i,t}]$ et $\gamma_{i,t} \in [\gamma_{i,t}]$. Avec ces deux intervalles, il est alors possible de définir une boîte $[\mathbf{w}_i]$ de façon à ce que $\hat{\mathbf{w}}_i \in [\mathbf{w}_i]$, avec $\hat{\mathbf{w}}_i$ la position de l'obstacle détecté dans l'environnement. La Figure 2.5 illustre la construction d'une telle boîte. On généralise cette démarche pour tous les jeux de mesures du télémètre, mais aussi pour les mesures odométriques.

Les données disponibles afin de localiser le robot sont donc les suivantes :

- les données odométriques $[\mathbf{u}_0], \dots, [\mathbf{u}_{t-1}]$,
- les données extéroceptives (télémètre laser) $[\mathbf{z}_0] = \{[\mathbf{w}_{0,i}]\}, \dots, [\mathbf{z}_t] = \{[\mathbf{w}_{t,i}]\}$,
- la grille d'occupation \mathcal{E}_G .

Afin de traiter ce problème de localisation globale nous proposons une approche ensembliste. L'objectif est alors d'évaluer à chaque instant t une boîte $[\mathbf{q}_t]$, aussi petite que possible, contenant de façon garantie, la posture \mathbf{q}_t du robot.

2 Présentation de la méthode

Comme précisé précédemment, nous proposons une approche ensembliste à ce problème de localisation. Les équations de notre système peuvent alors s'écrire sous la forme

$$\mathbf{q}_{t+1} = f(\mathbf{q}_t, \mathbf{u}_t), \quad \mathbf{q}_{t+1} \in [\mathbf{q}_{t+1}], \mathbf{q}_t \in [\mathbf{q}_t], \mathbf{u}_t \in [\mathbf{u}_t] \quad (2.5)$$

$$\mathbf{z}_t = g_{\mathcal{E}}(\mathbf{q}_t), \quad \mathbf{z}_t \in [\mathbf{z}_t], \mathbf{q}_t \in [\mathbf{q}_t]. \quad (2.6)$$

On remarque que, connaissant $[\mathbf{q}_t]$, il est possible d'évaluer $[\mathbf{q}_{t+1}]$ en utilisant les outils de l'analyse par intervalles (on rappelle que f est connue, il est donc possible d'établir une fonction d'inclusion $[f]$, et que $[\mathbf{u}_t]$ est fourni par l'odométrie).

Le gros du problème consiste à calculer $[\mathbf{q}_t]$ en utilisant les données du télémètre $[\mathbf{z}_t]$ ainsi que la carte \mathcal{E}_G . En d'autres termes, on voudrait pouvoir calculer " $[\mathbf{q}_t] = g_{\mathcal{E}}^{-1}([\mathbf{z}_t])$ ". On aurait ainsi une étape de prédiction (le calcul de $[\mathbf{q}_{t+1}]$) et une étape de correction utilisant les données extéroceptives ($[\mathbf{q}_{t+1}]^* = [\mathbf{q}_{t+1}] \cap g_{\mathcal{E}}^{-1}([\mathbf{z}_{t+1}])$).

La difficulté c'est que nous ne connaissons pas $g_{\mathcal{E}}$. Cette fonction caractérise le fonctionnement du capteur et dépend du capteur, de la posture du robot et de l'environnement. Il n'est donc pas possible de calculer directement $[\mathbf{q}_t] = g_{\mathcal{E}}^{-1}([\mathbf{z}_t])$. Notre approche consiste alors à considérer cette étape de correction comme un problème de satisfaction de contraintes (CSP⁵).

2.1 Formalisation en problème de satisfaction de contraintes

Notre approche repose sur la considération, à l'instant t , du problème de correction comme un problème de satisfaction de contraintes. Nous rappelons qu'un CSP se définit par un ensemble de variables \mathcal{V} , un ensemble de domaines \mathcal{D} pour ces variables et un ensemble de contraintes \mathcal{C} reliant les variables entre-elles (Annexe A, Section 2.2).

Remarque. Ce CSP étant valide à tout instant t , pour plus de lisibilité nous ne noterons pas les indices de temps. On notera abusivement $\mathbf{q}_t \equiv \mathbf{q}$, $\mathbf{z}_t \equiv \mathbf{z}$...

2.1.1 Variables du CSP

La première variable de notre problème correspond à la posture du robot $\mathbf{q} = (x_1, x_2, \theta)$. Nous avons ensuite le jeu de mesures \mathbf{z} : chaque mesure \mathbf{w}_i est caractérisée par une distance y_i et un angle γ_i . La notation \mathbf{w}_i correspond aux coordonnées, dans

5. Constraint Satisfaction Problem.

le repère global (le repère de l'environnement), de l'obstacle détecté (Figure 2.2). Ces coordonnées sont calculées de la façon suivante

$$\mathbf{w}_i = \begin{pmatrix} w_{1_i} \\ w_{2_i} \end{pmatrix} = \begin{pmatrix} y_i \sin(\theta + \gamma_i) + x_1 \\ y_i \cos(\theta + \gamma_i) + x_2 \end{pmatrix}. \quad (2.7)$$

Nous ajoutons donc toutes ces variables à notre problème de satisfaction de contraintes. Ce qui nous donne

$$\mathcal{V} = \left\{ \mathbf{q}, \{\gamma_i, y_i, \mathbf{w}_i\}_{\forall i} \right\}. \quad (2.8)$$

Détails de l'Équation 2.7. Nous rappelons qu'une mesure est caractérisée par une distance y_i et un angle γ_i . Ces données sont relatives à la position et à la direction du robot (Figure 2.6). Il est alors possible de définir (y_{1_i}, y_{2_i}) la position de l'obstacle détecté dans le repère du robot (Figure 2.6) :

$$y_{1_i} = y_i \sin(\gamma_i), \quad (2.9)$$

$$y_{2_i} = y_i \cos(\gamma_i). \quad (2.10)$$

Afin d'avoir les coordonnées de l'obstacle dans le repère de l'environnement, $\mathbf{w}_i = (w_{1_i}, w_{2_i})$, nous appliquons la matrice de transformation (rotation + translation) correspondant à la position (x_1, x_2) et la direction θ du robot dans l'environnement :

$$\begin{pmatrix} w_{1_i} \\ w_{2_i} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} y_i \sin(\gamma_i) \\ y_i \cos(\gamma_i) \end{pmatrix} + \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}. \quad (2.11)$$

En développant l'équation précédente on obtient

$$\begin{aligned} w_{1_i} &= y_i \cos(\theta) \sin(\gamma_i) + y_i \sin(\theta) \cos(\gamma_i) + x_1, \\ &= y_i \sin(\theta + \gamma_i) + x_1. \end{aligned} \quad (2.12)$$

$$\begin{aligned} w_{2_i} &= -y_i \sin(\theta) \sin(\gamma_i) + y_i \cos(\theta) \cos(\gamma_i) + x_2, \\ &= y_i \cos(\theta + \gamma_i) + x_2. \end{aligned} \quad (2.13)$$

2.1.2 Domaines des variables

Il est possible d'associer des domaines à ces variables.

Nous avons $[\mathbf{q}] = ([x_1], [x_2], [\theta])$, qui correspond à l'évaluation courante de la posture \mathbf{q} . Ensuite, comme précisé précédemment, il est possible d'associer des intervalles aux mesures. Nous avons donc $[\gamma_i] = [\gamma_i - \delta_\gamma, \gamma_i + \delta_\gamma]$ et $[y_i] = [y_i - \delta_y, y_i + \delta_y]$. Pour finir nous définissons le domaine $[\mathbf{w}_i]$ comme étant

$$[\mathbf{w}_i] = \begin{pmatrix} [w_{1_i}] \\ [w_{2_i}] \end{pmatrix} = \begin{pmatrix} [y_i] \sin([\theta] + [\gamma_i]) + [x_1] \\ [y_i] \cos([\theta] + [\gamma_i]) + [x_2] \end{pmatrix}. \quad (2.14)$$

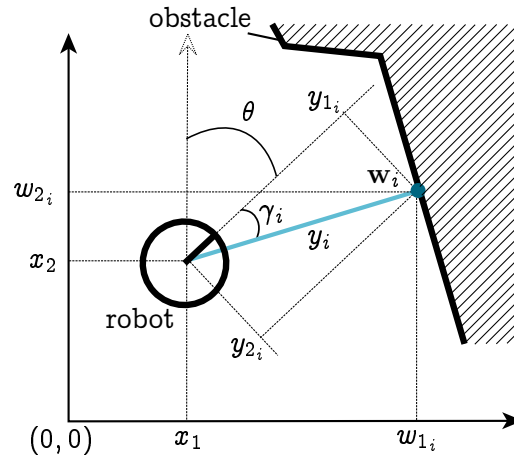


Figure 2.6 – Évaluation de la position de l'obstacle détecté, dans le repère de l'environnement.

Nous avons alors

$$\mathcal{D} = \{[\mathbf{q}], \{[\gamma_i], [y_i], [\mathbf{w}_i]\}_{\forall i}\}. \quad (2.15)$$

2 .1.3 Contraintes du CSP

Nous définissons maintenant les différentes contraintes que nous allons considérer ici. Nous rappelons que les contraintes correspondent à des relations qui relient les variables entre elles.

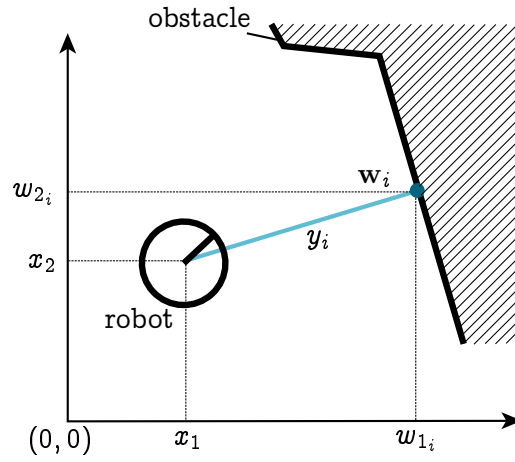
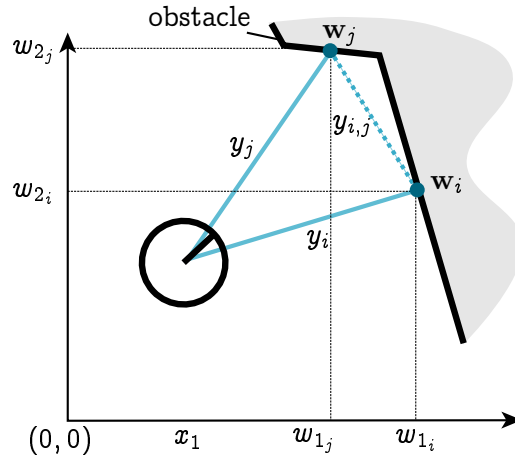
Construction de la mesure. La première contrainte est directement déduite de la construction de la boîte \mathbf{w}_i , reliant les mesures à la posture du robot (Équation 2.7). Cette contrainte, notée c_w est définie comme étant

$$c_w : \mathbf{w}_i = \begin{pmatrix} y_i \sin(\theta + \gamma_i) + x_1 \\ y_i \cos(\theta + \gamma_i) + x_2 \end{pmatrix}. \quad (2.16)$$

Distance robot-obstacle. Ensuite nous pouvons définir une contrainte sur la distance séparant l'obstacle du robot (Figure 2.7). Nous savons que l'obstacle détecté par la $i^{\text{ème}}$ mesure doit être à une distance y_i du robot. Nous avons alors la contrainte c_{y_i}

$$c_{y_i} : y_i^2 = (x_1 - w_{1_i})^2 + (x_2 - w_{2_i})^2. \quad (2.17)$$

Distance entre deux mesures. Comme précisé précédemment nous considérons un ensemble de mesures prises autour du robot. Il est alors possible de considérer les mesures par paires, et de définir une contrainte sur la distance séparant les mesures deux à deux (Figure 2.8). Soient deux mesures \mathbf{w}_i et \mathbf{w}_j , nous définissons $y_{i,j}$

Figure 2.7 – Distance séparant l'obstacle du robot (y_i).Figure 2.8 – Distance $y_{i,j}$ séparant les deux mesures w_i et w_j .

comme étant la distance séparant ces deux mesures. On peut alors écrire la contrainte $c_{y_{i,j}}$

$$c_{y_{i,j}} : y_{i,j}^2 = (w_{1_i} - w_{1_j})^2 + (w_{2_i} - w_{2_j})^2. \quad (2.18)$$

Correspondance mesure-obstacle. La dernière contrainte que nous considérons, précise que la mesure doit correspondre à un obstacle dans l'environnement. En d'autres termes, le capteur ne peut pas détecter des obstacles qui n'existent pas. Cette contrainte, notée

$$c_{\mathcal{E}_G} : w_i \in \mathcal{E}_G, \quad (2.19)$$

est illustrée sur la Figure 2.9. Elle sera présentée plus en détail Section 2.2.3.

Nous avons alors

$$\mathcal{C} = \{c_w, c_{y_i}, c_{y_{i,j}}, c_{\mathcal{E}_G}\}. \quad (2.20)$$

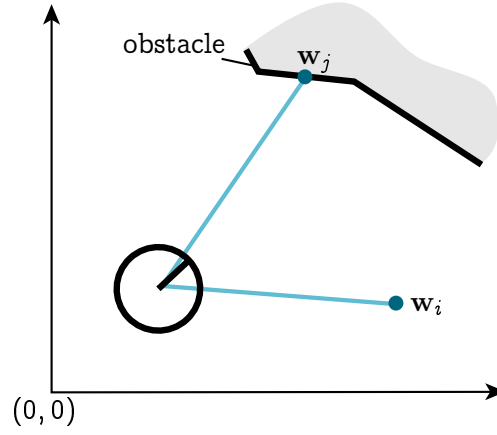


Figure 2.9 – Les mesures doivent correspondre à un obstacle dans l’environnement. La mesure w_j est cohérente avec cette contrainte tandis que la mesure w_i ne l’est pas.

2 .1.4 Problème de satisfaction de contraintes

Nous pouvons maintenant écrire notre étape de correction comme le problème de satisfaction de contraintes suivant

$$\begin{aligned}
 \mathcal{V} &= \{ \mathbf{q}, \{y_i, \gamma_i, \mathbf{w}_i\}_{\forall i} \} \\
 \mathcal{D} &= \{ [\mathbf{q}], \{[y_i], [\gamma_i], [\mathbf{w}_i]\}_{\forall i} \} \\
 \mathcal{C} &= \begin{cases} c_{\mathbf{w}} & : \mathbf{w}_i = \begin{pmatrix} y_i \sin(\theta + \gamma_i) + x_1 \\ y_i \cos(\theta + \gamma_i) + x_2 \end{pmatrix}, \\ c_{y_i} & : y_i^2 = (x_1 - w_{1_i})^2 + (x_2 - w_{2_i})^2, \\ c_{y_{i,j}} & : y_{i,j}^2 = (w_{1_i} - w_{1_j})^2 + (w_{2_i} - w_{2_j})^2, \\ c_{\mathcal{E}_G} & : \mathbf{w}_i \in \mathcal{E}_G. \end{cases} \quad (2.21)
 \end{aligned}$$

Les contraintes considérées ici ne sont pas exhaustives. Il est possible d’en ajouter d’autres en fonction du problème de localisation étudié. Nous verrons par la suite (Chapitre 4) une nouvelle contrainte pouvant être ajoutée à ce CSP afin d’en améliorer les performances.

On notera que la contrainte $c_{y_{i,j}}$ est une contrainte redondante. Elle est définie afin d’améliorer l’efficacité des étapes de contraction et n’est pas nécessaire à la définition du problème.

2.2 Présentation des contracteurs

Afin de pouvoir traiter le CSP présenté précédemment il est nécessaire d'attacher un contracteur (Annexe A, Section 2.2.2) à chacune des contraintes. On rappelle qu'un contracteur est un algorithme associé à une contrainte, permettant de réduire les domaines en *supprimant* les valeurs non-consistantes avec la contrainte.

2.2.1 Contracteur associé à la contrainte c_w

Nous notons $C_w([\mathbf{q}], [\mathbf{w}_i], [y_i], [\gamma_i])$ le contracteur associé à la contrainte c_w . Afin de faciliter la lecture de ce contracteur nous divisons la contrainte c_w en deux sous-contraintes :

$$c_{w_1} : w_{1_i} = y_i \sin(\theta + \gamma_i) + x_1 \quad (2.22)$$

$$c_{w_2} : w_{2_i} = y_i \cos(\theta + \gamma_i) + x_2 \quad (2.23)$$

Nous développons alors deux contracteurs $C_{w_1}([w_{1_i}], [y_i], [\gamma_i], [\mathbf{q}])$ et $C_{w_2}([w_{2_i}], [y_i], [\gamma_i], [\mathbf{q}])$ respectivement attachés aux contraintes c_{w_1} et c_{w_2} . Ces contracteurs, présentés Algorithmes 3 et 4, utilisent le principe de propagation avant-arrière (Annexe A, Section 2.2.4).

En se basant sur ces deux contracteurs, il devient possible de créer le contracteur $C_w([\mathbf{q}], [\mathbf{w}_i], [y_i], [\gamma_i])$, associé à la contrainte c_w . Ce contracteur est présenté Algorithme 5. On notera que Δ (Ligne 4) correspond au critère d'arrêt de l'algorithme. Ce critère peut avoir plusieurs formes : un nombre maximal d'itérations, un seuil de contraction minimum, une combinaison de plusieurs critères...

2.2.2 Contracteur associé aux contraintes c_{y_i} et $c_{y_{i,j}}$

On remarque que les contraintes c_{y_i} et $c_{y_{i,j}}$ sont de la même forme : elles correspondent toutes les deux à une contrainte de distance entre deux points. On peut alors définir un *contracteur de distance* qui sera associé aux deux contraintes. Ce contracteur, noté $C_{dst}([d], [\mathbf{a}], [\mathbf{b}])$, peut être construit comme une propagation avant-arrière sur l'équation suivante

$$d^2 = (a_1 - b_1)^2 + (a_2 - b_2)^2, \quad (2.24)$$

avec $\mathbf{a} = (a_1, b_1)$ et $\mathbf{b} = (b_1, b_2)$. Ce contracteur est présenté Algorithme 6.

2.2.3 Contracteur associé à la contrainte $c_{\mathcal{E}_G}$

Intéressons nous maintenant à la dernière contrainte $c_{\mathcal{E}_G}$, qui précise qu'une mesure w_i doit correspondre à un obstacle dans l'environnement. Dans notre cas, la

Algorithme 3: Contracteur $C_{w_1}([w_{1_i}], [\gamma_i], [y_i], [\mathbf{q}])$

Données : $[w_{1_i}], [y_i], [\gamma_i], [\mathbf{q}] = ([x_1], [x_2], [\theta])$

- 1 // Initialisations
- 2 $[i_1] = [\theta] + [\gamma_i]; [i_2] = \sin([i_1]);$
- 3 $[i_3] = [y_i] \times [i_2]; [i_4] = [i_3] + [x_1];$
- 4 // Propagation arrière
- 5 $[i_4]^* = [i_4] \cap [w_{1_i}];$
- 6 $[i_3]^* = [i_3] \cap ([i_4]^* - [x_1]);$
- 7 $[x_1]^* = [x_1] \cap ([i_4]^* - [i_3]^*);$
- 8 $[y_i]^* = [y_i] \cap \frac{[i_3]^*}{[i_2]};$
- 9 $[i_2]^* = [i_2] \cap \frac{[i_3]^*}{[y_i]^*};$
- 10 $[i_1]^* = [i_1] \cap \sin^{-1}([i_2]^*);$
- 11 $[\theta]^* = [\theta] \cap ([i_1]^* - [\gamma_i]);$
- 12 $[\gamma_i]^* = [\gamma_i] \cap ([i_1]^* - [\theta]);$
- 13 // Propagation avant
- 14 $[i_1]^* = [i_1]^* \cap ([\theta]^* + [\gamma_i]^*);$
- 15 $[i_2]^* = [i_2]^* \cap \sin([i_1]^*);$
- 16 $[i_3]^* = [i_3]^* \cap ([y_i]^* \times [i_2]^*);$
- 17 $[i_4]^* = [i_4]^* \cap ([i_3]^* + [x_1]^*);$
- 18 $[w_{1_i}]^* = [w_{1_i}] \cap [i_4]^*;$

Résultat : $[w_{1_i}]^*, [\gamma_i]^*, [y_i]^*, [\mathbf{q}]^* = ([x_1]^*, [x_2], [\theta]^*)$

Algorithme 4: Contracteur $C_{w_2}([w_{2_i}], [\gamma_i], [y_i], [q])$

Données : $[w_{2_i}], [y_i], [\gamma_i], [q] = ([x_1], [x_2], [\theta])$

- 1 // Initialisations
- 2 $[i_1] = [\theta] + [\gamma_i]; [i_2] = \cos([i_1]);$
- 3 $[i_3] = [y_i] \times [i_2]; [i_4] = [i_3] + [x_2];$
- 4 // Propagation arrière
- 5 $[i_4]^* = [i_4] \cap [w_{2_i}];$
- 6 $[i_3]^* = [i_3] \cap ([i_4]^* - [x_2]);$
- 7 $[x_2]^* = [x_2] \cap ([i_4]^* - [i_3]^*);$
- 8 $[y_i]^* = [y_i] \cap \frac{[i_3]^*}{[i_2]};$
- 9 $[i_2]^* = [i_2] \cap \frac{[i_3]^*}{[y_i]^*};$
- 10 $[i_1]^* = [i_1] \cap \cos^{-1}([i_2]^*);$
- 11 $[\theta]^* = [\theta] \cap ([i_1]^* - [\gamma_i]);$
- 12 $[\gamma_i]^* = [\gamma_i] \cap ([i_1]^* - [\theta]);$
- 13 // Propagation avant
- 14 $[i_1]^* = [i_1]^* \cap ([\theta]^* + [\gamma_i]^*);$
- 15 $[i_2]^* = [i_2]^* \cap \cos([i_1]^*);$
- 16 $[i_3]^* = [i_3]^* \cap ([y_i]^* \times [i_2]^*);$
- 17 $[i_4]^* = [i_4]^* \cap ([i_3]^* + [x_2]^*);$
- 18 $[w_{2_i}]^* = [w_{2_i}] \cap [i_4]^*;$

Résultat : $[w_{2_i}]^*, [\gamma_i]^*, [y_i]^*, [q]^* = ([x_1], [x_2]^*, [\theta]^*)$

Algorithme 5: Contracteur $C_w([q], [w_i], [y_i], [\gamma_i])$

Données : $[q] = ([x_1], [x_2], [\theta]), [w_i] = ([w_{1_i}], [w_{2_i}], [y_i], [\gamma_i])$

- 1 // Initialisations
- 2 $[q]^* = [q]; [w_{1_i}]^* = [w_{1_i}]; [w_{2_i}]^* = [w_{2_i}]; [y_i]^* = [y_i]; [\gamma_i]^* = [\gamma_i];$
- 3 // Boucle de contraction
- 4 **tant que** Δ *n'est pas vérifiée* **faire**
- 5 $([w_{1_i}]^*, [\gamma_i]^*, [y_i]^*, [q]^*) = C_{w_1}([w_{1_i}]^*, [\gamma_i]^*, [y_i]^*, [q]^*);$
- 6 $([w_{2_i}]^*, [\gamma_i]^*, [y_i]^*, [q]^*) = C_{w_2}([w_{2_i}]^*, [\gamma_i]^*, [y_i]^*, [q]^*);$
- 7 **fin**

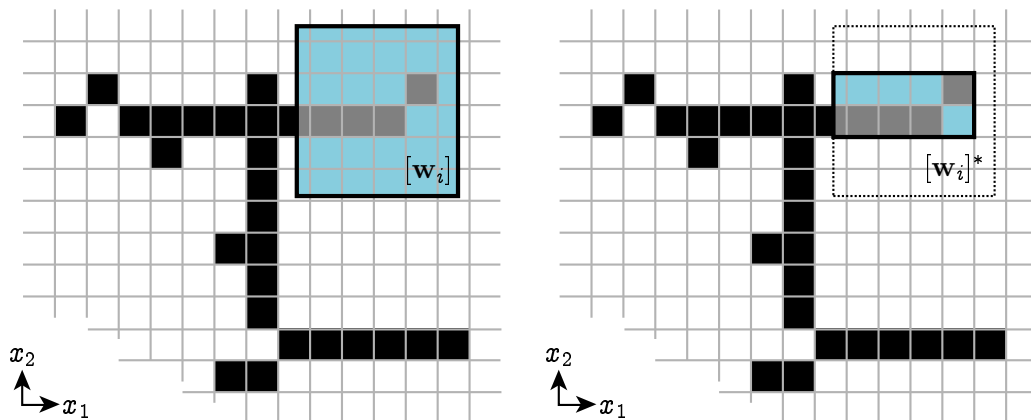
Résultat : $[q]^*, [w_i]^* = ([w_{1_i}]^*, [w_{2_i}]^*), [y_i]^*, [\gamma_i]^*$

Algorithme 6: Contracteur $C_{dst}([d], [a], [b])$

Données : $[d], [a] = ([a_1], [a_2]), [b] = ([b_1], [b_2])$

- 1 // Initialisations
- 2 $[i_1] = [a_1] - [b_1]; [i_2] = ([i_1])^2;$
- 3 $[i_3] = [a_2] - [b_2]; [i_4] = ([i_3])^2;$
- 4 $[i_5] = [i_2] + [i_4];$
- 5 // Propagation (arrière)
- 6 $[i_5]^* = [i_5] \cap [d];$
- 7 $[i_2]^* = [i_2] \cap ([i_5]^* - [i_4]);$
- 8 $[i_4]^* = [i_4] \cap ([i_5]^* - [i_2]^*);$
- 9 $[i_3]^* = [i_3] \cap (\sqrt{[i_4]^*} \cup -\sqrt{[i_4]^*});$
- 10 $[a_2]^* = [a_2] \cap ([i_3]^* + [b_2]);$
- 11 $[b_2]^* = [b_2] \cap ([a_2]^* - [i_3]^*);$
- 12 $[i_1]^* = [i_1] \cap (\sqrt{[i_2]^*} \cup -\sqrt{[i_2]^*});$
- 13 $[a_1]^* = [a_1] \cap ([i_1]^* + [b_1]);$
- 14 $[b_1]^* = [b_1] \cap ([a_1]^* - [i_1]^*);$
- 15 // Propagation (avant)
- 16 $[i_1]^* = [i_1]^* \cap ([a_1]^* - [b_1]^*);$
- 17 $[i_2]^* = [i_2]^* \cap ([i_1]^*)^2;$
- 18 $[i_3]^* = [i_3]^* \cap ([a_2]^* - [b_2]^*);$
- 19 $[i_4]^* = [i_4]^* \cap ([i_3]^*)^2;$
- 20 $[d]^* = [d] \cap ([i_2]^* + [i_4]^*);$

Résultat : $[d]^*, [a]^* = ([a_1]^*, [a_2]^*), [b]^* = ([b_1]^*, [b_2]^*)$



(a) La mesure doit correspondre à un obstacle dans l'environnement (les cellules noires sur la carte).

(b) Il est alors possible de la contracter pour la rendre consistante avec cette contrainte.

Figure 2.10 – Illustration d'une contraction associée à la contrainte $c_{\mathcal{E}_G}$.

mesure w_i est caractérisée par une boîte $[w_i]$ et l'environnement \mathcal{E} par une grille \mathcal{E}_G . On peut donc reformuler la contrainte en précisant qu'il faut que la boîte $[w_i]$ intersecte au moins un obstacle dans la carte \mathcal{E}_G . Contracter la mesure $[w_i]$ revient dans ce cas à supprimer les valeurs qui n'intersectent pas d'obstacle (de cellule à 1 dans \mathcal{E}_G). La Figure 2.10 illustre cette contraction.

Nous voulons donc un algorithme nous permettant de calculer efficacement la plus petite boîte $[w_i]^* \subseteq [w_i]$ contenant l'ensemble des obstacles intersectés par $[w_i]$. La méthode présentée ici s'inspire du pré-calcul présenté dans [Sliwka 2011].

Afin de disposer d'un test d'intersection mesure/obstacle efficace, nous ajoutons un pré-calcul à la carte \mathcal{E}_G : nous enregistrerons, pour chaque cellule, le nombre de cellules à 1 (contenant un obstacle) dans son quadrant Sud-Ouest (Figure 2.11).

Soit une carte \mathcal{E}_G composée de $n \times m$ cellules $c_{i,j}$, $i = 1, \dots, n$ et $j = 1, \dots, m$. À chaque cellule $c_{i,j}$ est associée un poids $g_{i,j} = \{0, 1\}$ évaluant la présence d'un obstacle dans la cellule (ceci correspond à la définition de notre grille d'occupation booléenne). En plus de ce poids nous associons maintenant un entier $\eta_{i,j}$ correspondant au nombre de cellules, du quadrant Sud-Ouest de $c_{i,j}$, ayant un poids à 1. L'algorithme du calcul de ces entiers est présenté Algorithme 7.

En se basant sur ce pré-calcul il est possible de définir une fonction $\phi([w_i])$ qui calcule le nombre de cellules, contenant un obstacle, intersectés par la mesure $[w_i]$:

$$\phi([w_i]) = \eta_{\overline{w_1}, \overline{w_2}} + \eta_{\underline{w_1}-1, \underline{w_2}-1} - \eta_{\overline{w_1}, \underline{w_2}-1} - \eta_{\underline{w_1}-1, \overline{w_2}}, \quad (2.25)$$

avec $c_{\overline{w_1}, \overline{w_2}}$ la cellule contenant le point supérieur droit de la mesure, $c_{\underline{w_1}, \underline{w_2}}$ la cellule

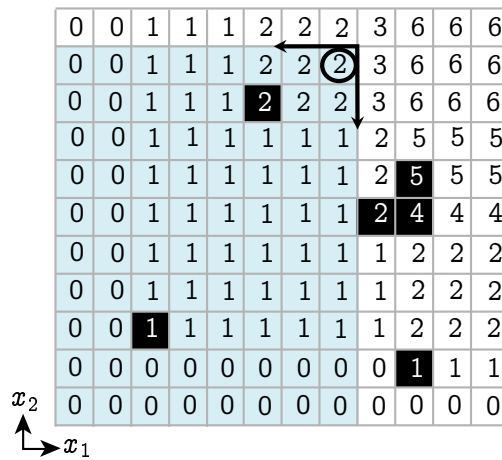


Figure 2.11 – Exemple de pré-calcul. Dans chaque cellule on note le nombre de cellules contenant un obstacle (de cellules noires) dans le quadrant Sud-Ouest. En considérant la cellule entourée, son quadrant Sud-Ouest correspond aux cellules colorées : on remarque qu’il y a bien deux cellules contenant des obstacles (deux cellules noires).

Algorithme 7: Pré-calcul de la grille

Données : $\{g_{1,1}, \dots, g_{n,m}\}$

```

1 // Évaluation pour les autres cellules
2 pour i = 1 à n faire
3   pour j = 1 à m faire
4     si i > 1 et j > 1 alors
5       // La cellule n'est pas aux frontières de la grille
6        $\eta_{i,j} = \eta_{i,j-1} - \eta_{i-1,j-1} + \eta_{i-1,j} + g_{i,j};$ 
7     sinon
8       si i = 1 et j > 1 alors
9         // La cellule est sur la première colonne
10         $\eta_{i,j} = \eta_{i,j-1} + g_{i,j};$ 
11      sinon
12        si i > 1 et j = 1 alors
13          // La cellule est sur la première ligne
14           $\eta_{i,j} = \eta_{i-1,j} + g_{i,j};$ 
15        sinon
16          // Première cellule, i = 1 et j = 1
17           $\eta_{i,j} = g_{i,j};$ 
18        fin
19      fin
20    fin
21  fin
22 fin

```

Résultat : $\{\eta_{1,1}, \dots, \eta_{n,m}\}$

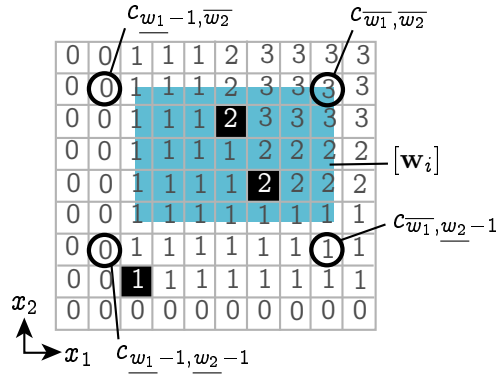


Figure 2.12 – Sur cet exemple nous avons $\phi([w_i]) = 3 + 0 - 1 - 0 = 2$. La boîte $[w_i]$ intersecte donc deux cellules ayant un poids à 1 (elle intersecte deux obstacles dans la carte).

contenant le point inférieur gauche de la mesure et ainsi de suite (Figure 2.12)...

Nous disposons maintenant d'une fonction qui permet de déterminer rapidement combien d'obstacles sont intersectés par une boîte. Contracter une mesure $[w_i]$ sur la grille revient à trouver la plus petite boîte $[w_i]^* \subseteq [w_i]$ de façon à ce que

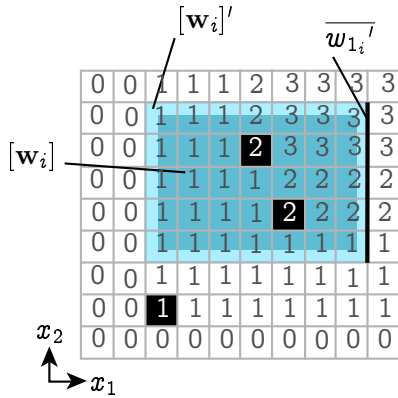
$$\phi([w_i]) = \phi([w_i]^*). \quad (2.26)$$

Afin d'évaluer la boîte $[w_i]^*$ nous effectuons une recherche dichotomique sur chacune des bornes de $[w_i]$. L'idée est alors d'*augmenter* les bornes inférieures et de *diminuer* les bornes supérieures sans changer la valeur de $\phi([w_i])$. La recherche de la boîte $[w_i]^*$ est illustrée Figure 2.13. Le lecteur notera que la recherche se fait de façon discrète *sur la grille* : on ne s'autorise pas à chercher une borne *en plein milieu* d'une cellule. On en déduit le contracteur $C_{\mathcal{E}_G}([a], \mathcal{E}_G)$, Algorithme 8, qui permet de contracter une boîte $[a] = ([a_1], [a_2])$ en fonction d'une grille \mathcal{E}_G (on remarquera la considération d'une boîte $[a]$ et non plus d'une mesure $[w_i]$ par soucis de généralité et de lisibilité).

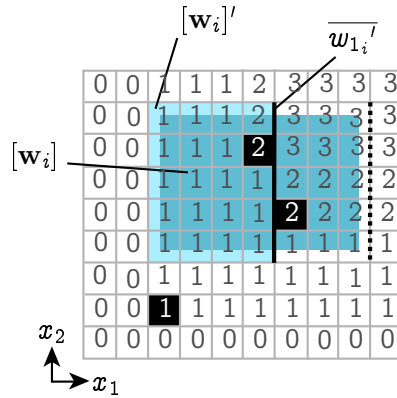
2.2.4 Résolution du CSP

L'approche choisie pour résoudre ce problème de satisfaction de contraintes, consiste en un appel successif des différents contracteurs associés aux contraintes. Cet algorithme, noté $CSP([q], \{[y_i], [\gamma_i], [w_i]\}_{V_i}, \mathcal{E}_G)$ est présenté Algorithme 9. Ci-suivent deux remarques au sujet de cet algorithme :

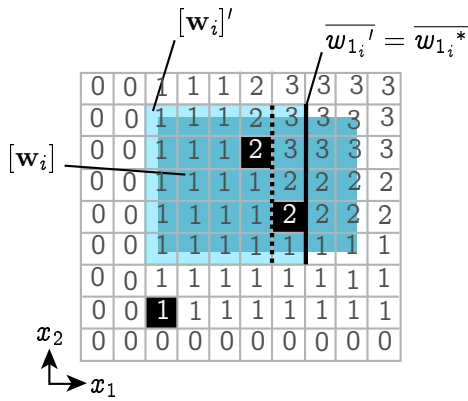
- Afin de pouvoir manipuler la contrainte $c_{y_i, j}$ (contrainte sur la distance séparant deux mesures entre elles), il est nécessaire de coupler les mesures deux à deux. Nous considérons alors qu'au sein du jeu de mesures z , les mesures sont associées deux à deux.



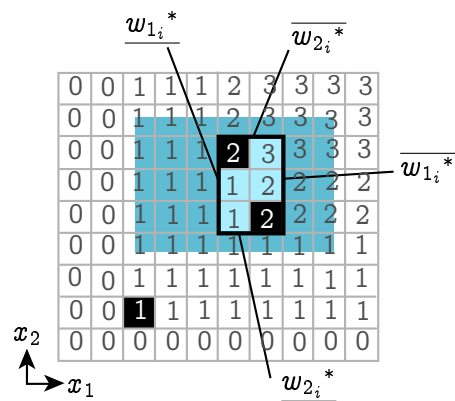
(a) Nous initialisons la recherche avec une boîte $[w_i]'$ correspondant à la plus petite boîte contenant toutes les cellules de la grille intersectées par $[w_i]$. Nous nous intéressons ici à l'obtention de la borne $w_{1_i}^*$. On notera que $\phi([w_i]) = \phi([w_i]') = 2$.



(b) On effectue une recherche dichotomique afin de trouver la plus petite borne w_{1_i}' pour $[w_i]'$ de façon à ce que $\phi([w_i]') = \phi([w_i]) = 2$. Pour cette première itération nous avons $\phi([w_i]') = 1$, la borne $w_{1_i}^*$ est donc plus à droite.



(c) Nous continuons les itérations de notre recherche dichotomique jusqu'à l'obtention de la borne $w_{1_i}^*$ qui correspond à la plus petite valeur possible pour w_{1_i}'



(d) Le même raisonnement est appliqué pour les autres bornes : on cherche à minimiser w_{2_i}' et à maximiser w_{1_i}' et w_{2_i}' .

Figure 2.13 – Illustration d'une contraction associée à la contrainte $c_{\mathcal{E}_G}$.

Algorithme 8: $C_{\mathcal{E}_G}([a], \mathcal{E}_G)$

Données : $[a] = ([a_1], [a_2]) = ([\underline{a}_1, \overline{a}_1], [\underline{a}_2, \overline{a}_2]), \mathcal{E}_G$

```

1 si  $\phi([x]) = 0$  alors
2   // Si la boîte n'intersecte aucun obstacle, il n'y a pas de solution
3    $[a]^* = \emptyset$ ;
4 sinon
5   //  $[a]$  est plaqué sur la grille
6    $[a]'$  = la plus petite boîte contenant toutes les cellules intersectées par  $[a]$ ;
7   // Contraction de  $\overline{a}_1'$ 
8   // Initialisation de la recherche dichotomique pour  $\overline{a}_1^*$ 
9    $a_{1_{mid}} = \overline{a}_1'$ ;  $a_{1_{inf}} = \underline{a}_1'$ ;  $\overline{a}_1^* = \overline{a}_1'$ ;
10  tant que il y a plus deux cellules qui séparent  $a_{1_{mid}}$  de  $a_{1_{inf}}$  faire
11     $a_{1_{mid}}$  =valeur médiane de  $[a_{1_{inf}}, a_{1_{mid}}]$  (sur la grille!);
12    si  $\phi([\underline{a}_1, a_{1_{mid}}], [a_2]) \neq \phi([x])$  alors
13      // Le nombre de cellules a changé, la borne supérieure est plus à droite
14       $a_{1_{inf}} = a_{1_{mid}}$ ;  $a_{1_{mid}} = \overline{a}_1^*$ ;
15    sinon
16      // Le nombre de cellules est inchangé, on peut continuer à essayer de diminuer la borne supérieure
17       $\overline{a}_1^* = a_{1_{mid}}$ ;
18    fin
19  fin
20  // Il y a deux cellules ou moins qui séparent  $a_{1_{mid}}$  de  $a_{1_{inf}}$  et donc trois valeurs possibles pour  $\overline{a}_1^*$  :  $a_{1_{mid}}$ ,  $a_{1_{inf}}$  ou la valeur médiane.
21  si  $\phi([\underline{a}_1, a_{1_{inf}}], [a_2])$  alors
22     $\overline{a}_1^* = a_{1_{inf}}$ ;
23  sinon
24     $a_{1_{med}}$  =valeur médiane de  $[a_{1_{inf}}, a_{1_{mid}}]$ ;
25    si  $\phi([\underline{a}_1, a_{1_{med}}], [a_2])$  alors
26       $\overline{a}_1^* = a_{1_{med}}$ ;
27    sinon
28       $\overline{a}_1^* = a_{1_{mid}}$ ;
29    fin
30  fin
31  // La même méthode est employée pour trouver  $\underline{a}_1^*$ ,  $\overline{a}_2^*$  et  $\underline{a}_2^*$ 
32 fin

```

Résultat : $[a]^* = ([a_1]^*, [a_2]^*) = ([\underline{a}_1^*, \overline{a}_1^*], [\underline{a}_2^*, \overline{a}_2^*])$

- Pour calculer la distance séparant deux mesures $[y_{i,j}]$, Ligne 14, nous utilisons le Théorème d'Al-Kashi (aussi connu comme étant la généralisation du Théorème de Pythagore).

Δ (Ligne 3) correspond toujours au critère d'arrêt de l'algorithme. Le critère le plus simple à mettre en place étant un nombre maximum d'itérations.

Algorithme 9: $CSP(\mathbf{q}, \{[y_i], [\gamma_i], [\mathbf{w}_i]\}_{\forall i}, \mathcal{E}_G)$

Données : $\mathbf{q} = ([x_1], [x_2], [\theta]), \{[y_i], [\gamma_i], [\mathbf{w}_i] = ([w_{1_i}], [w_{2_i}])\}_{\forall i}, \mathcal{E}_G$

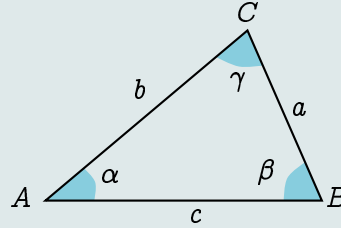
```

1 // Initialisations
2  $\mathbf{q}^* = \mathbf{q}; \{[y_i]^*, [\gamma_i]^*, [\mathbf{w}_i]^*\}_{\forall i} = \{[y_i], [\gamma_i], [\mathbf{w}_i]\}_{\forall i};$ 
3 tant que  $\Delta$  n'est pas vérifié faire
4   pour chaque couple de mesures  $(\mathbf{w}_i, \mathbf{w}_j)$  faire
5     // Contraction des mesures sur la grille
6      $[\mathbf{w}_i]^* = C_{\mathcal{E}_G}([\mathbf{w}_i]^*, \mathcal{E}_G);$ 
7      $[\mathbf{w}_j]^* = C_{\mathcal{E}_G}([\mathbf{w}_j]^*, \mathcal{E}_G);$ 
8     // Calcul de l'angle entre les deux mesures
9      $[\gamma_{i,j}] = \max([\gamma_i]^*, [\gamma_j]^*) - \min([\gamma_i]^*, [\gamma_j]^*);$ 
10    // Calcul de la distance entre les deux mesures (Théorème d'Al-Kashi)
11     $[y_{i,j}] = \sqrt{([y_i]^*)^2 + ([y_j]^*)^2 - 2 \times [y_i]^* \times [y_j]^* \times \cos([\gamma_{i,j}])};$ 
12    // Appel des autres contracteurs
13     $([y_{i,j}]^*, [\mathbf{w}_i]^*, [\mathbf{w}_j]^*) = C_{dst}([y_{i,j}], [\mathbf{w}_i]^*, [\mathbf{w}_j]^*);$ 
14     $([\mathbf{q}]^*, [\mathbf{w}_i]^*, [y_i]^*, [\gamma_i]^*) = C_w([\mathbf{q}]^*, [\mathbf{w}_i]^*, [y_i]^*, [\gamma_i]^*);$ 
15     $([\mathbf{q}]^*, [\mathbf{w}_j]^*, [y_j]^*, [\gamma_j]^*) = C_w([\mathbf{q}]^*, [\mathbf{w}_j]^*, [y_j]^*, [\gamma_j]^*);$ 
16     $([y_i]^*, [\mathbf{q}]^*, [\mathbf{w}_i]^*) = C_{dst}([y_i]^*, [\mathbf{q}]^*, [\mathbf{w}_i]^*);$ 
17     $([y_j]^*, [\mathbf{q}]^*, [\mathbf{w}_j]^*) = C_{dst}([y_j]^*, [\mathbf{q}]^*, [\mathbf{w}_j]^*);$ 
18  fin
19 fin
Résultat :  $\mathbf{q}^* = ([x_1]^*, [x_2]^*, [\theta]^*)$ 

```

Nous disposons maintenant d'un algorithme nous permettant de contracter une posture en utilisant un jeu de mesures, ainsi que la connaissance de l'environnement (la carte).

Théorème d'Al-Kashi - Théorème de Pythagore généralisé



En considérant le triangle ci-dessus, d'après le Théorème d'Al-Kashi nous avons

$$a^2 = b^2 + c^2 - 2bc \cos(\alpha)$$

$$b^2 = a^2 + c^2 - 2ac \cos(\beta)$$

$$c^2 = a^2 + b^2 - 2ab \cos(\gamma)$$

2.3 Algorithme IAL

Nous présentons maintenant notre algorithme de localisation. La première version de l'algorithme IAL est présentée Algorithm 10. Cette version est l'application directe du concept de prédiction/correction que l'on retrouve dans les filtres Bayésiens (Annexe B, Section 1.3). En utilisant l'arithmétique des intervalles, on est capable de prédire la posture courante en fonction de l'évaluation du déplacement et de la connaissance de la posture précédente. Et en utilisant le CSP présenté précédemment, on est capable de contracter (d'améliorer) l'évaluation de la posture courante à l'aide d'un jeu de mesures du télémètre. Dans un contexte de localisation globale, on initialisera $[q_0]$ par toutes les postures possibles pour le robot :

$$[q_0] = ([x_0], [\theta_0]) = [[\mathcal{E}], [0, 2\pi]]. \quad (2.27)$$

L'ensemble des positions possibles correspond à l'environnement dans lequel le robot évolue, et l'ensemble des orientations possibles correspond à $[0, 2\pi]$.

En pratique cet algorithme ne fonctionne presque jamais. En effet, avec un domaine $[q_t]$ trop important, les contracteurs du CSP ne parviennent pas à contracter efficacement. Sans contraction sur $[q_t]$ on propage alors les erreurs d'odométries sans jamais pouvoir améliorer la connaissance de la posture du robot (sans jamais contacter) : le robot dérive. Le cas typique est celui de la localisation globale : avec $[q_0]$ correspondant à toutes les postures possibles, nous ne pouvons généralement pas contracter. Une technique pour *débloquer* un contracteur, quand il n'arrive plus à contracter, est de bissecter le domaine à contracter (le découper en deux sous-domaines, Annexe A, Section 1.2.4). L'idée est alors de coupler des phases de

bissections aux phases de contractions.

Algorithme 10: $IAL([\mathbf{q}_0], \mathcal{E}_G)$

Données : $[\mathbf{q}_0], \mathcal{E}_G$

```

1 // Initialisation
2  $t = 0$ ;
3 tant que  $\Delta$  n'est pas vérifié faire
4   récupérer jeu de mesures  $\mathbf{z}_t$ ;
5   // Correction
6    $[\mathbf{q}_t]^* = CSP([\mathbf{q}_t], [\mathbf{y}_{i,t}], [\gamma_{i,t}], [\mathbf{w}_{i,t}]_{\forall i}, \mathcal{E}_G)$ ;
7   déplacer le robot ; évaluer  $[\mathbf{u}_t]$ ;
8   // Prédiction
9    $[\mathbf{q}_{t+1}] = f([\mathbf{q}_t]^*, [\mathbf{u}_t])$ ;
10   $t = t + 1$ ;
11 fin
12 // Dernière correction
13 récupérer jeu de mesures  $\mathbf{z}_t$ ;
14  $[\mathbf{q}_t]^* = CSP([\mathbf{q}_t], [\mathbf{y}_{i,t}], [\gamma_{i,t}], [\mathbf{w}_{i,t}]_{\forall i}, \mathcal{E}_G)$ ;
Résultat :  $[\mathbf{q}_t]^*$ 

```

2 .3.1 Ajout de bissections

Nous ajoutons donc des étapes de bisection à l'algorithme IAL (Algorithme 10). Le nouvel algorithme est présenté Algorithme 11.

Cet algorithme manipule non plus un seul vecteur d'intervalles mais une liste de boîtes. Chaque boîte est contractée à l'aide du CSP, supprimée si elle n'est pas solution, bissectée si elle est trop volumineuse et conservée si elle contient des solutions et ne peut plus être bissectée.

L'ajout d'une étape de bisection permet d'affiner la solution trouvée par l'algorithme en *cassant* les points d'équilibres atteints par les contractions du CSP. On notera que le critère de bisection, Ligne 10, peut être de différentes formes :

- une taille de boîte limite en dessous de laquelle on ne s'autorise plus à bissecter,
- un nombre de bissections maximal à chaque itération,
- un nombre limite de boîtes à ne pas dépasser,
- une combinaison des trois critères précédents,
- ...

Cet algorithme, tel quel, a encore un inconvénient : il ne permet pas de gérer d'éventuelles valeurs aberrantes dans le jeu de mesures. En effet en admettant qu'une seule valeur du jeu de mesure ne satisfasse pas les contraintes (à cause d'une erreur du capteur par exemple), alors le CSP se retrouve sans solution et retourne l'ensemble

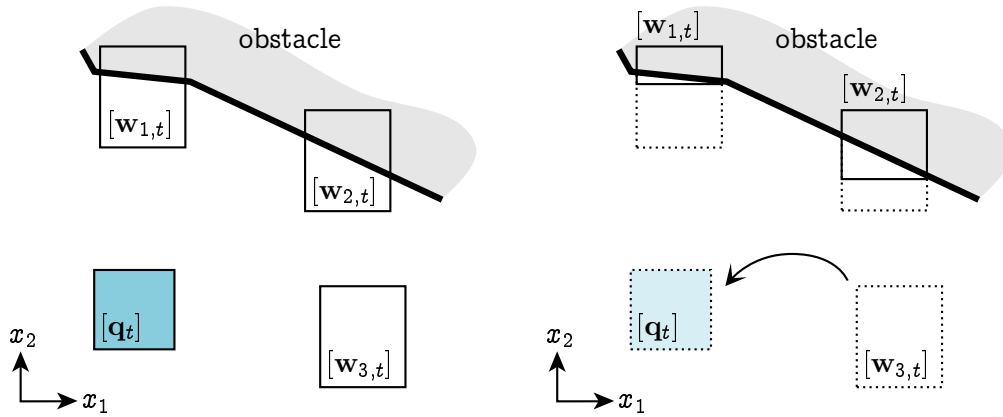
Algorithme 11: $IAL([q_0], \mathcal{E}_G)$

Données : $[q_0], \mathcal{E}_G$

```

1 // Initialisation
2  $t = 0; Q_0 = \{[q_0]\};$ 
3 tant que  $\Delta$  n'est pas vérifié faire
4   récupérer jeu de mesures  $z_t$ ;
5   tant que  $Q_t$  n'est pas vide faire
6     // Récupération de la dernière boîte de  $Q_t$ 
7      $[q] = Q_t.retirer\_derniere\_boite();$ 
8     // Correction
9      $[q_t]^* = CSP([q_t], [y_{i,t}], [\gamma_{i,t}], [w_{i,t}]_{\forall i}, \mathcal{E}_G);$ 
10    si  $[q_t]^*$  peut être bissectée alors
11       $([q_{1,t}], [q_{2,t}]) = Bissecter([q_t]^*);$ 
12       $Q_t.ajouter([q_{1,t}]); Q_t.ajouter([q_{2,t}]);$ 
13    sinon
14       $Q_t.ajouter([q_t]^*);$ 
15    fin
16  fin
17  déplacer le robot ; évaluer  $[u_t]$ ;
18  // Prédiction sur l'ensemble des boîtes de  $Q_t^*$ 
19  pour chaque  $[q_t]^* \in Q_t^*$  faire
20     $[q_{t+1}] = f([q_t]^*, [u_t]);$ 
21     $Q_{t+1}.ajouter([q_{t+1}]);$ 
22  fin
23   $t = t + 1;$ 
24 fin
25 // Dernière correction
26 récupérer jeu de mesures  $z_t$ ;
27 Répéter les Lignes 5 à 16;
Résultat :  $Q_t^*$ 

```



(a) Supposons un jeu de mesures comprenant une valeur aberrante : $[w_{3,t}]$.

(b) Quand on contracte les mesures sur les obstacles, on obtient l'ensemble vide pour $[w_{3,t}]$, et en propageant cette contraction dans notre CSP, nous obtenons l'ensemble vide pour $[q_t]$.

Figure 2.14 – Non-gestion des valeurs aberrantes.

vide. En l'état, l'algorithme manque de robustesse face à d'éventuelles erreurs dans le jeu de mesures.

Afin de pouvoir prendre en compte ces valeurs aberrantes et ainsi rendre notre algorithme plus robuste, nous utilisons le principe de l'intersection relaxée.

2 .3.2 Gestion des valeurs aberrantes

Comme précisé précédemment, en l'état notre algorithme de localisation ne nous permet pas de considérer de valeurs aberrantes dans le jeu de mesures. La Figure 2.14 présente un exemple du résultat de la non-gestion d'une valeur aberrante.

Afin de pouvoir prendre en compte ces valeurs aberrantes, nous ajoutons une étape d'intersection relaxée à notre CSP. L'intersection relaxée d'un ensemble \mathbb{Q} , notée $\cap^{\{q\}} \mathbb{Q}$, correspond à tous les points de l'ensemble \mathbb{Q} qui intersectent au moins $n_{\mathbb{Q}} - q$ éléments de \mathbb{Q} , avec $n_{\mathbb{Q}}$ le nombre d'éléments de \mathbb{Q} . Cette intersection particulière est présentée plus en détail Annexe A, Section 1.1.3. Le nouvel algorithme ainsi obtenu, noté $CSP_{\text{relaxé}}()$, est présenté Algorithme 12. On notera que le terme q dans l'algorithme correspond au nombre de valeurs aberrantes considérées, et n'a aucun rapport avec la posture q du robot.

Il suffit maintenant de remplacer la fonction $CSP()$ par la fonction $CSP_{\text{relaxé}}()$ au sein de l'algorithme IAL pour le rendre robuste en présence de valeurs aberrantes.

Algorithme 12: $CSP_{\text{relaxé}}(\mathbf{q}, \{[y_i], [\gamma_i], [\mathbf{w}_i]\}_{\forall i}, \mathcal{E}_G, q)$

Données : $\mathbf{q} = ([x_1], [x_2], [\theta]), \{[y_i], [\gamma_i], [\mathbf{w}_i] = ([w_{1,i}], [w_{2,i}])\}_{\forall i}, \mathcal{E}_G, q$

```

1 // Initialisations
2  $\mathbf{q}^* = \mathbf{q}; \{[y_i]^*, [\gamma_i]^*, [\mathbf{w}_i]^*\}_{\forall i} = \{[y_i], [\gamma_i], [\mathbf{w}_i]\}_{\forall i};$ 
3 tant que  $\Delta$  n'est pas vérifié faire
4     // Création d'une liste de boîtes pour la q-intersection
5      $\mathbb{Q}^* = \emptyset;$ 
6     pour chaque couple de mesures  $(\mathbf{w}_i, \mathbf{w}_j)$  faire
7          $[\mathbf{q}]_{tmp}^* = [\mathbf{q}]^*;$ 
8         // Contraction des mesures sur la grille
9          $[\mathbf{w}_i]^* = C_{\mathcal{E}_G}([\mathbf{w}_i]^*, \mathcal{E}_G);$ 
10         $[\mathbf{w}_j]^* = C_{\mathcal{E}_G}([\mathbf{w}_j]^*, \mathcal{E}_G);$ 
11        // Calcul de l'angle entre les deux mesures
12         $[\gamma_{i,j}] = \max([\gamma_i]^*, [\gamma_j]^*) - \min([\gamma_i]^*, [\gamma_j]^*);$ 
13        // Calcul de la distance entre les deux mesures (Théorème d'Al-Kashi)
14         $[y_{i,j}] = \sqrt{([y_i]^*)^2 + ([y_j]^*)^2 - 2 \times [y_i]^* \times [y_j]^* \times \cos([\gamma_{i,j}])};$ 
15        // Appel du reste des contracteurs
16         $([y_{i,j}]^*, [\mathbf{w}_i]^*, [\mathbf{w}_j]^*) = C_{dst}([y_{i,j}], [\mathbf{w}_i]^*, [\mathbf{w}_j]^*);$ 
17         $([\mathbf{q}]_{tmp}^*, [\mathbf{w}_i]^*, [y_i]^*, [\gamma_i]^*) = C_w([\mathbf{q}]_{tmp}^*, [\mathbf{w}_i]^*, [y_i]^*, [\gamma_i]^*);$ 
18         $([\mathbf{q}]_{tmp}^*, [\mathbf{w}_j]^*, [y_j]^*, [\gamma_j]^*) = C_w([\mathbf{q}]_{tmp}^*, [\mathbf{w}_j]^*, [y_j]^*, [\gamma_j]^*);$ 
19         $([y_i]^*, [\mathbf{q}]_{tmp}^*, [\mathbf{w}_i]^*) = C_{dst}([y_i]^*, [\mathbf{q}]_{tmp}^*, [\mathbf{w}_i]^*);$ 
20         $([y_j]^*, [\mathbf{q}]_{tmp}^*, [\mathbf{w}_j]^*) = C_{dst}([y_j]^*, [\mathbf{q}]_{tmp}^*, [\mathbf{w}_j]^*);$ 
21        // Ajout de la boîte courante à la liste
22         $\mathbb{Q}^*.ajouter([\mathbf{q}]_{tmp}^*);$ 
23     fin
24     // Intersection relaxée de toutes les boîtes
25      $[\mathbf{q}]^* = [\mathbf{q}]^* \cap (\bigcap^{\{q\}} \mathbb{Q}^*);$ 
26 fin
Résultat :  $[\mathbf{q}]^*$ 

```

3 Expérimentations et comparaisons

Afin de tester notre approche, différentes expérimentations ont été réalisées. Nous présentons ici les résultats obtenus.

3.1 Expérimentations avec un ordinateur distant

Dans un premier temps, nous avons utilisé un ordinateur distant afin de traiter les données issues d'un robot MiniRex. Toutes ces expérimentations ont été réalisées sur un ordinateur ayant les caractéristiques suivantes

- Mémoire : 2,0 GiB,
- Deux Processeurs : Interl(R) Core(TM) CPU 6420 @ 2.13GHz.

Il est important de noter que les cartes utilisées sont directement issues d'un algorithme de SLAM⁶ (sans traitement à posteriori) et que les mesures du télémètre n'ont subi aucun traitement particulier (pour essayer d'identifier les valeurs aberrantes par exemple). Ces tests ont été réalisés dans le cadre du défi CAROTTE.

3.1.1 Arène du LORIA

Pour ces tests, M. Antoine Bautin⁷ et M. Nicolas Beaufort, du LORIA⁸, nous ont fourni la carte d'une arène construite dans le cadre de la participation au défi CAROTTE. La carte fournie a été réalisée à l'aide de l'algorithme de cartographie Slam-O-matic, développé au LISA [Lucidarme 2011]. L'environnement considéré, ainsi que la carte disponible, sont représentés Figure 2.15.

Au sein de cet environnement nous avons procédé à 33 localisations globales *statiques*. Pour chaque localisation, nous n'avons aucune connaissance sur la posture initiale du robot et nous ne nous autorisons aucun déplacement : le robot doit se localiser sans se déplacer (il doit se localiser statiquement). On a donc utilisé 33 jeux de mesures LIDAR, indépendants les uns des autres.

Les résultats des 33 localisations sont illustrés sur la Figure 2.15 et résumés dans le tableau suivant :

6. Simultaneous Localization And Mapping - Localisation et cartographie Simultanées.

7. www.loria.fr/~bautin/

8. Laboratoire Lorrain de Recherche en Informatique et ses Applications, Nancy (France)

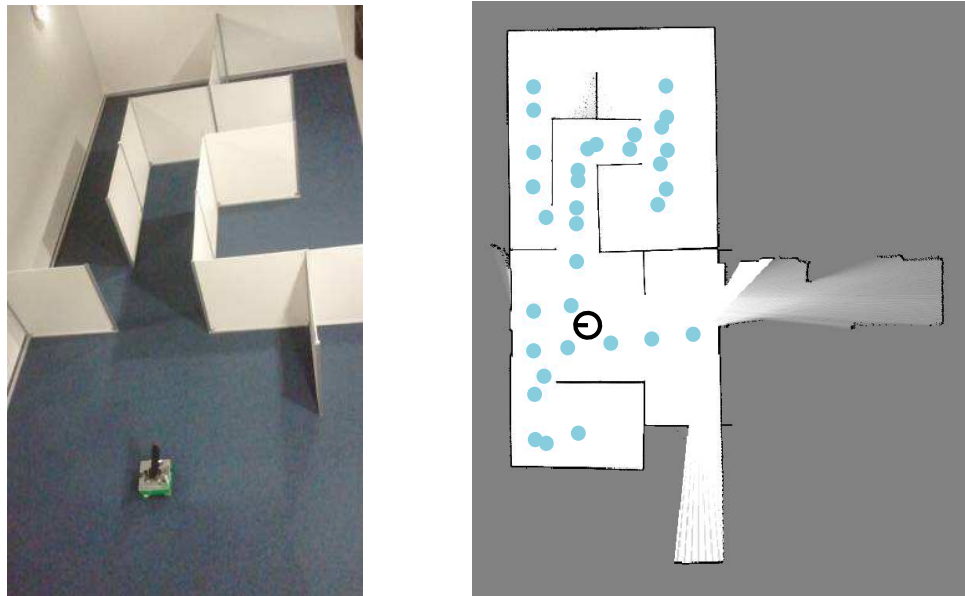


Figure 2.15 – L'image de gauche présente l'arène considérée. L'image de droite présente la carte utilisée ainsi que les différentes localisations du robots (représentées par les 33 points). On notera que la posture indiquée en noire sur l'image de droite correspond à la posture du robot présent sur l'image de gauche.

	Moyenne	Meilleur	Pire
Temps de calcul (s)	33.4	21	45.5
Précision sur x_1 (mm)	± 37.2	± 18.1	± 112.1
Précision sur x_2 (mm)	± 32.7	± 16.5	± 51.5
Précision sur θ (deg)	± 2.75	± 0.55	± 12.2

On rappelle que, bien qu'utilisant des données réelles, ces calculs ont été réalisés sur une machine distante et non pas sur le robot.

Nous retenons de cette manipulation le fait que l'algorithme IAL soit capable d'effectuer des localisations globales (sans connaissance à priori sur la posture du robot) et statiques (sans déplacement du robot). On remarquera qu'il faut moins d'une minute pour localiser le robot, statiquement et de façon globale, dans l'environnement. Au regard d'une mission de plusieurs dizaines de minutes, ce temps de calcul est négligeable. De plus, lorsque le robot est suffisamment bien localisé (que l'espace de recherche est petit), le temps de calcul pour la mise à jour de la position dans cet environnement est inférieur à la seconde. Ce qui laisse penser que l'algorithme est utilisable en conditions réelles.

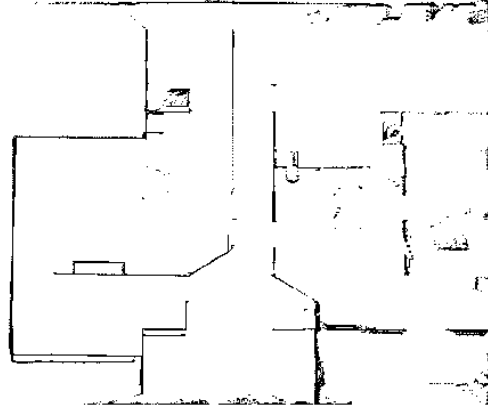


Figure 2.16 – Carte réalisée pendant la première manche du défi CAROTTE - Juin 2011.

3.1.2 Arène du défi CAROTTE

Afin de tester notre algorithme dans un environnement plus hostile, nous avons utilisé des données enregistrées pendant la première édition du défi CAROTTE⁹, par le consortium Cart-O-matic. L'environnement étudié correspond à une arène intérieure de $15\text{m} \times 15\text{m}$. L'intérêt de cette expérimentation est de pouvoir tester l'algorithme en utilisant une carte de moins bonne qualité. Cette carte est présentée Figure 2.16.

Comme pour l'expérimentation présentée précédemment, nous procédons ici à une localisation globale statique (sans déplacement du robot). On s'aperçoit que même dans ces conditions, l'algorithme IAL est assez robuste pour permettre de localiser, statiquement et globalement, le robot. La Figure 2.17 présente un exemple de localisation calculée par l'algorithme. Voici les caractéristiques de cette solution :

- Temps de calcul : 28s,
- Précision en x_1 : $\pm 114.2\text{mm}$,
- Précision en x_2 : $\pm 105.8\text{mm}$,
- Précision en θ : $\pm 2.87\text{deg}$.

L'algorithme apparaît donc comme étant robuste, parvenant à localiser le robot même en présence de valeurs aberrantes et d'une carte imparfaite.

9. CARTographie par RObot d'un TErritoire

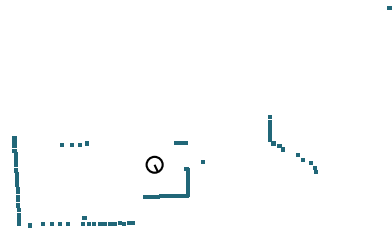


Figure 2.17 – Une des solutions proposée par l’algorithme pour le problème de localisation.

3.2 Expérimentations avec un MiniRex

Nous avons ensuite implémenté notre algorithme de localisation sur un robot MiniRex. Ce travail a été réalisé pendant le stage de master de M. Antoine Durand. Trois expérimentations ont alors été réalisées :

- une mission de localisation globale,
- une mission de suivi de posture,
- une mission de suivi de posture dans des conditions difficiles.

On notera qu’à l’inverse des résultats présentés précédemment, tous les calculs ont cette fois été réalisés sur le robot en *temps réel* (pendant les expérimentations).

L’intérêt de ces expérimentations est de valider l’algorithme dans des conditions réelles, en considérant cette fois le déplacement du robot pour la localisation (on rappelle que les localisations présentées précédemment étaient statiques, sans déplacement du robot). Toutes les cartes manipulées ici sont le résultat d’une mission de SLAM préalablement effectuée par le robot à l’aide de l’algorithme Slam-O-matic. Elles n’ont bénéficié d’aucune modification à posteriori. De plus, le robot MiniRex ne possède pas de capteur odométrique (de roues codeuses par exemple). L’odométrie est donc estimée en fonction du temps d’actionnement des roues. Cette technique, très imprécise, entraîne un fort pessimisme sur l’évaluation de la posture du robot après chaque déplacement. Pour finir on notera que le jeu du télémètre n’est pas traité à priori (pour essayer de déterminer les valeurs aberrantes par exemple), et est utilisé tel quel.

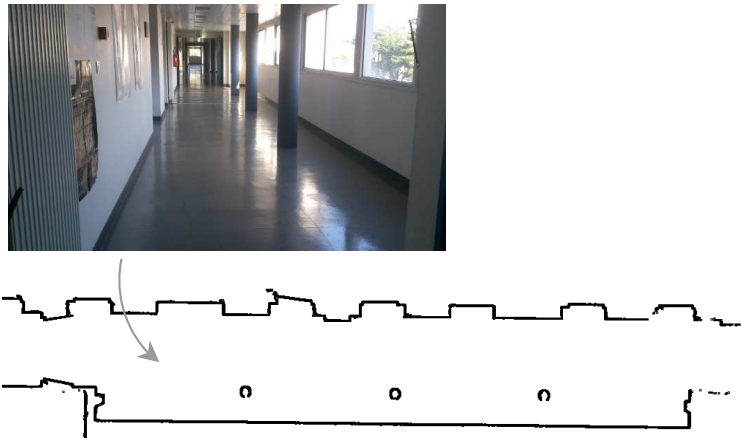


Figure 2.18 – L’environnement ainsi que la carte considérés pour la mission de localisation globale.

Les trois expérimentations ci-dessous ont été réalisées au troisième étage de l’IS-TIA¹⁰, école d’ingénieurs de l’université d’Angers.

3 .2.1 Mission de localisation globale

Une localisation globale dans un environnement symétrique (Figure 2.18) a été réalisée. Le robot a volontairement été placé dans une posture ambiguë afin d’observer le comportement de l’algorithme dans cette situation.

La carte considérée est présentée Figure 2.18. La connaissance sur la posture initiale q_0 du robot correspond à une boîte de $26\text{m} \times 4\text{m} \times 360\text{deg}$. Cette boîte apparaît sur la première image de la Figure 2.19.

La Figure 2.19 présente les données initiales de notre problème de localisation, ainsi que les résultats des sept itérations de l’algorithme effectuées pendant la mission. On remarque bien qu’à cause des symétries présentes dans l’environnement, lors de la première itération, l’algorithme présente deux boîtes distinctes comme solution du problème de localisation. Cependant cette ambiguïté est levée dès la deuxième itération, alors que le robot se déplace dans le couloir, apportant ainsi une nouvelle information permettant de statuer sur sa posture.

Le résultat final de la mission (septième itération) correspond à une boîte d’une taille de $42.1\text{cm} \times 14.9\text{cm} \times 7\text{deg}$ pour une distance parcourue d’approximativement 12m. La première itération, la plus coûteuse en temps de calcul, prend 26 secondes. On remarque que le résultat de cette localisation est du même ordre de grandeur que ceux obtenus lors des expérimentations présentées précédemment, avec un ordinateur distant.

10. Institut des Sciences et Techniques de l’Ingénieur d’Angers

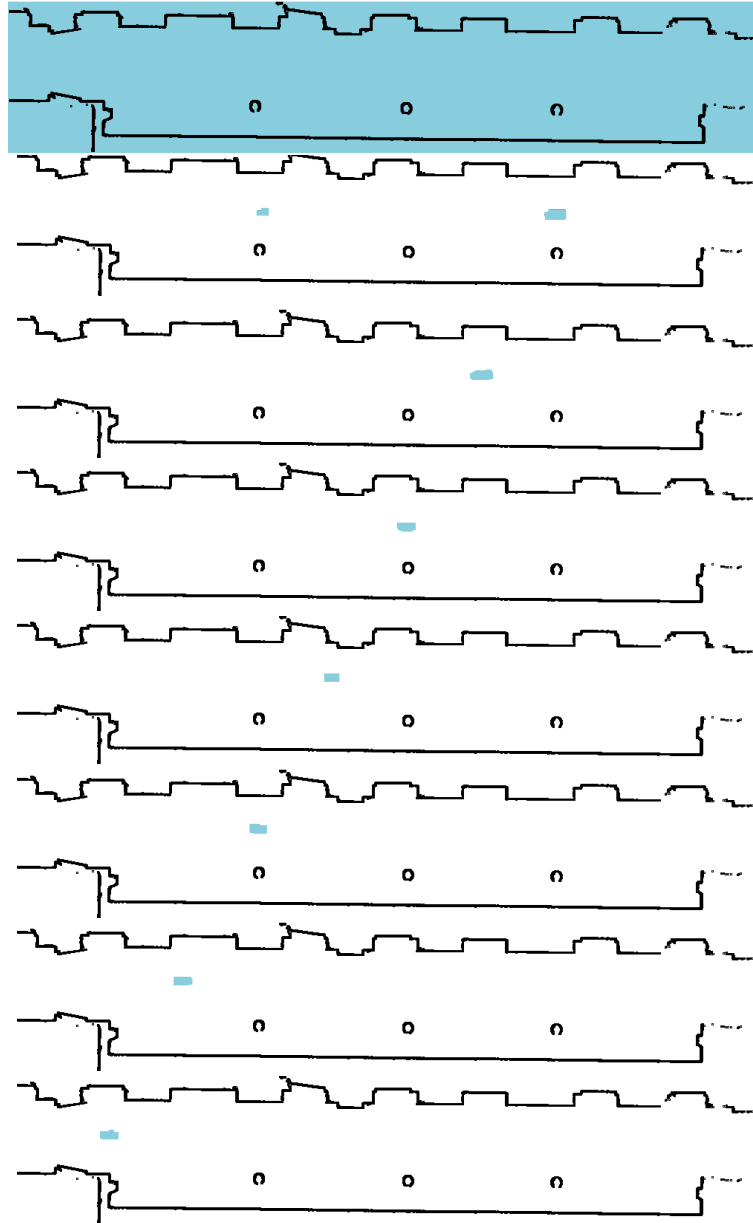
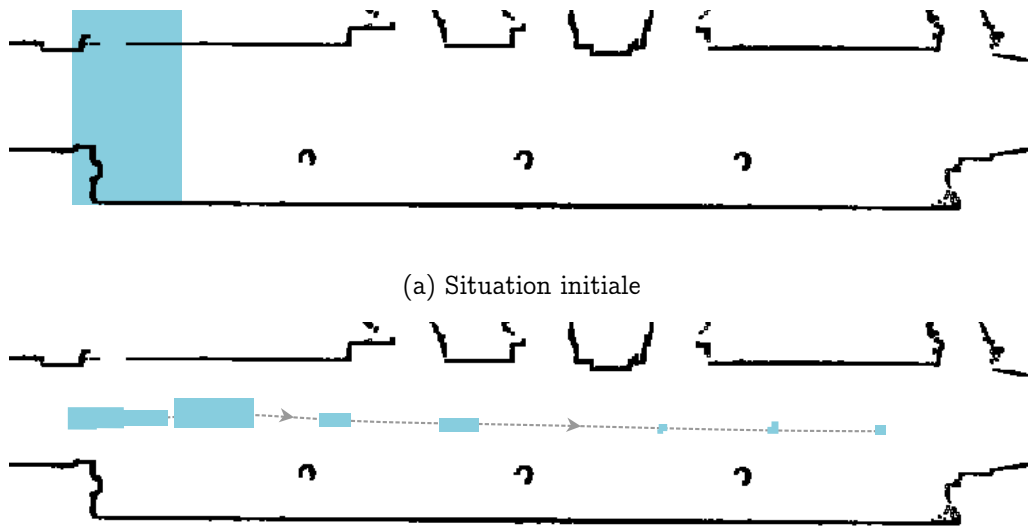


Figure 2.19 – Données initiales de notre problème de localisation (première image), ainsi que les sept itérations effectuées par l’algorithme IAL (la première image correspond à $t = 0$ et la dernière à $t = 7$). Une itération correspond à un pas de temps t . On notera que le robot ne se déplace pas entre les instants $t = 0$ et $t = 1$.



(b) Résultats des différentes itérations. La ligne pointillée approxime le déplacement du robot pendant la mission.

Figure 2.20 – Suivi de posture.

3 .2.2 Mission de suivi de posture

Nous effectuons maintenant une mission de suivi de posture : le robot a une connaissance (approximative) de sa posture initiale et doit se re-localiser après chaque déplacement. Nous effectuons cette localisation en considérant une carte moins *propre* que précédemment. En effet il est possible de remarquer sur la Figure 2.20 que les portes étaient ouvertes lors de la cartographie de la zone considérée (ces portes correspondent aux ouvertures situées au nord de la carte). Ces portes étant fermées lors de la mission de localisation, nous pouvons ainsi tester la robustesse de la méthode en présence de valeurs aberrantes.

La connaissance sur la posture initiale q_0 du robot correspond à une boîte de $1.6\text{m} \times 2.9\text{m} \times 180\text{deg}$ (Figure 2.20a). Les résultats des différentes itérations sont résumés sur la Figure 2.20b. On remarque que malgré la présence de valeurs aberrantes/d'incohérences entre la carte et l'environnement, le robot reste localisé pendant toute la durée de la mission.

Le résultat final de la mission (dernière itération) correspond à une boîte d'une taille de $7\text{cm} \times 10.3\text{cm} \times 4.8\text{deg}$ pour une distance parcourue d'approximativement 18m. Dans ce cas, la première itération dure 1 seconde. On remarque bien que le temps de calcul diminue considérablement quand la taille de la boîte initiale diminue.



Figure 2.21 – L’environnement et la carte considérés pour la mission de suivi de posture.

3 .2.3 Mission de suivi de posture avec virage

Pour finir nous avons effectué une mission de suivi de posture plus longue que la précédente : le robot se déplace plus longtemps dans un environnement plus grand. L’environnement et la carte considérés sont présentés Figure 2.21. On remarque que cette fois encore, la carte comporte un certain nombre d’erreurs. Les disparités entre la carte et l’environnement sont cette fois dues à la présence de baies vitrées dans l’environnement considéré. En effet, la carte a été réalisée en utilisant les données d’un télémètre laser, capteur ne permettant pas de détecter les obstacles transparents (vitres...). De plus le soleil présent lors de la cartographie a affecté les données du télémètre ce qui explique la présence d’obstacles en plein milieu du couloir sur la carte.

Une fois encore nous voulons mettre en évidence la robustesse de l’algorithme dans ces conditions défavorables. La connaissance sur la posture initiale q_0 du robot correspond à une boîte de $1.6\text{m} \times 2.9\text{m} \times 180\text{deg}$ (Figure 2.22a). La Figure 2.22b résume les résultats des différentes étapes du suivi de posture pendant le déplacement du robot. Nous pouvons remarquer que l’algorithme IAL parvient à suivre la posture du robot, et ce même lorsque le robot traverse la zone de baies vitrées.

Le résultat final de la mission (dernière itération) correspond à une boîte d'une taille de $47.9\text{cm} \times 18.9\text{cm} \times 11\text{deg}$ pour une distance parcourue d'approximativement 22m.

3.3 Comparaison MCL/IAL

Les précédentes expérimentations nous permettent de présenter l'algorithme IAL comme étant fonctionnel, robuste et utilisable en conditions réelles.

Nous voulons maintenant comparer cette méthode avec un algorithme classiquement utilisé pour le problème de localisation globale. Nous avons décidé de comparer notre méthode avec la méthode MCL¹¹, filtre particulière très utilisé en robotique mobile [Dellaert 1999, Fox 1999, Thrun 2001, Burchardt 2011, Fu 2011]. Cet algorithme est présenté Chapitre 1, Section 2. Premièrement, nous nous intéressons aux temps de calcul des deux méthodes, puis aux solutions proposées par ces dernières, pour finalement présenter un tableau récapitulatif des avantages et inconvénients des deux approches.

Le lecteur notera que les résultats présentés dans cette section ont été obtenus à l'aide d'un simulateur. Les environnements, cartes et mesures sont donc des données simulées.

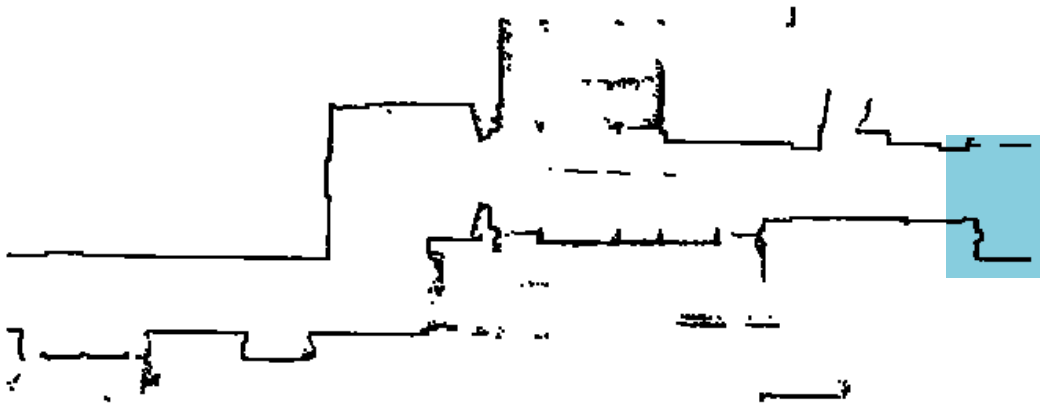
3.3.1 Temps de calcul

Afin de comparer les temps de calcul des deux algorithmes dans des conditions équitables, nous avons créé (en simulation) un environnement de $10\text{m} \times 10\text{m}$, présenté Figure 2.23. Nous supposons que le robot fait une cinquantaine de mesures, dont 10% de valeurs aberrantes. Dans ces conditions, en considérant la posture présente sur la Figure 2.23, il faut 20s à l'algorithme IAL pour localiser le robot, globalement et statiquement.

Dans ces mêmes conditions, l'algorithme MCL a besoin de 10000 particules pour localiser le robot efficacement. Pour une localisation globale statique (sans déplacer le robot), l'algorithme MCL effectue une cinquantaine d'itérations avant de lever toutes les ambiguïtés sur la posture du robot. Ce processus prend environ 8s. On note alors que dans ce cas, la méthode MCL est deux fois plus rapide que IAL.

Le problème, c'est qu'en faisant de la localisation statique avec l'algorithme MCL (en redessinant les particules sans déplacer le robot), le premier jeu de particules est déterminant pour le succès de la localisation. En effet, si aucune particule initiale n'est assez proche de la posture du robot, l'algorithme ne convergera pas vers la bonne solution. Nous avons effectué 10 localisations statiques (MCL), dans les

11. Monte Carlo Localization - Localisation Monte Carlo.



(a) Situation initiale



(b) Résultats des différentes itérations. La ligne pointillé approxime le déplacement du robot pendant la mission.

Figure 2.22 – Suivi de posture avec virage et baie vitrée.

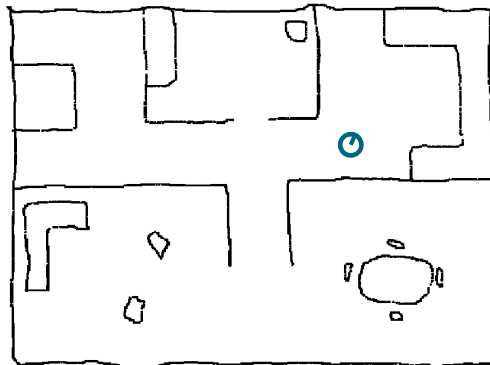


Figure 2.23 – Environnement simulé.

même conditions que précédemment, et nous avons observé 4 échecs (le robot a été mal localisé 4 fois). En effectuant 10 nouvelles localisations, avec cette fois un déplacement du robot après chaque itération, l'algorithme MCL identifie la posture du robot dans 100% des cas. Il apparaît donc que le déplacement du robot est une donnée importante pour MCL. Afin de comparer équitablement les temps de calculs des deux algorithmes il serait donc normal de rajouter le temps de déplacement du robot à la Localisation Monte Carlo. Dans nos expérimentations, en moyenne 5 déplacements (de 20cm) sont nécessaires avant que la meilleure particule ne corresponde à la posture du robot. En admettant raisonnablement qu'il faut 2s au robot pour effectuer ce déplacement, il est possible de rajouter $5 \times 2 = 10$ s au temps de localisation MCL. On arrive ainsi à un temps de localisation similaire pour les deux méthodes IAL et MCL.

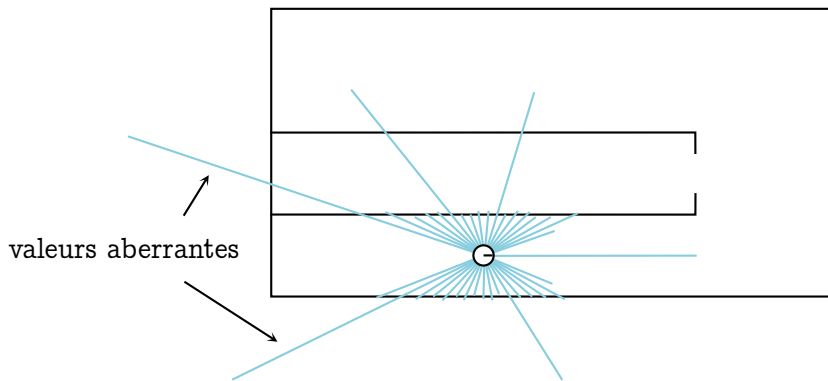
Cette première comparaison porte sur l'étape de localisation globale. À l'échelle d'une mission il est possible d'admettre que le robot est plus souvent bien localisé, que perdu dans son environnement. En effet la localisation globale se pose au début de la mission, et après chaque kidnapping. Le reste du temps, le robot effectue un suivi de posture. Il est donc intéressant de comparer aussi le temps de calcul des deux méthodes pendant ces phases de suivi de posture.

En gardant le même environnement que précédemment, nous avons effectué un suivi de la posture du robot à l'aide des deux méthodes. Une fois bien localisé, le robot se déplace et met à jour la connaissance de sa posture après chaque déplacement. Cette mise à jour prend en moyenne 0.07s avec IAL et 0.15s pour MCL (avec 10000 particules). Si la première itération de l'algorithme IAL est coûteuse en temps de calcul, une fois le domaine de recherche restreint la méthode est beaucoup plus rapide. À l'inverse, la méthode MCL compte un temps de calcul constant pour chaque itération (pour un nombre de particules constant). On notera cependant qu'il existe des versions de la Localisation Monte Carlo pour lesquelles le nombre de particules est variable. En divisant le nombre de particules par deux, on obtient par exemple un temps de calcul de 0.07s pour chaque itération. Une fois encore on observe que les temps de localisation des deux méthodes sont comparables.

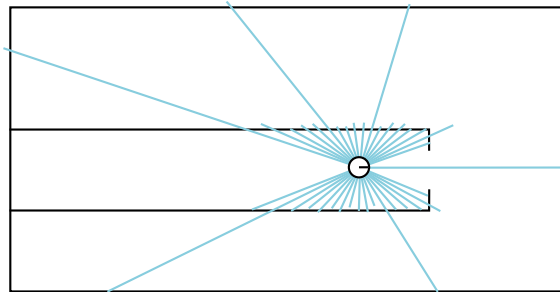
En conclusion, si la méthode MCL semble être plus rapide au premier coup d'œil, il s'avère que cette différence est moins évidente à l'échelle d'une mission complète. La méthode IAL se trouve être compétitive en terme de temps de calcul.

3.3.2 Solutions proposées par les algorithmes

Intéressons nous maintenant aux résultats fournis par les deux méthodes. L'algorithme IAL cherche une boîte qui contient de manière garantie la posture du robot. Cette garantie, est obtenue à l'aide du contexte à erreurs bornés et en supposant un nombre limite de valeurs aberrantes. L'algorithme MCL, quand à lui, cherche la particule avec le plus grand score possible. Ce score est évalué en fonction du nombre de mesures intersectant les obstacles de la carte. En d'autre terme, l'algorithme MCL



(a) Le robot fait un jeu de mesures comprenant six valeurs aberrantes.



(b) La posture présentée ici possède un meilleur score (et est donc plus probable) que la posture du robot.

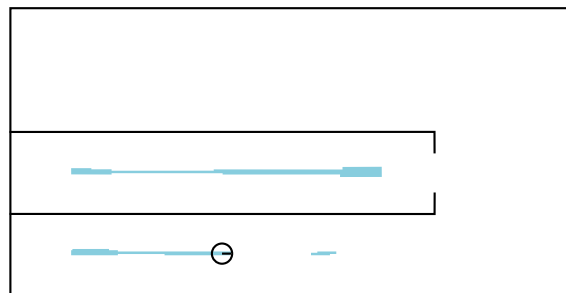
Figure 2.24 – Exemple de minimum global non solution du problème de localisation.

cherche la posture pour laquelle un maximum de mesures correspondent à des obstacles de la carte. Si avec un nombre suffisant de particules, la méthode parvient généralement à identifier cette posture particulière, une question peut toutefois être posée : la particule avec le meilleur score possible (la plus probable) correspond-elle bien à la posture du robot ?

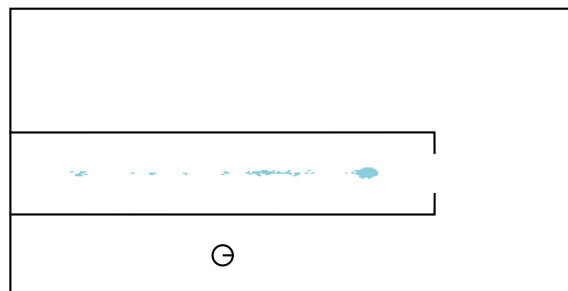
Dans un monde parfait, avec un capteur parfait, la réponse serait oui : la posture la plus probable associée à un jeu de mesures serait la posture génératrice du jeu de mesures (aux symétries près). Mais dès lors que des valeurs aberrantes peuvent faire partie du jeu de mesures, cette supposition n'est plus vérifiée... Un exemple est donné sur la Figure 2.24.

Dans ces conditions l'algorithme MCL converge vers la meilleure particule (qui ne correspond pas à la posture du robot), alors que l'algorithme IAL conserve quand à lui toutes les valeurs consistantes avec le problème, et donc la posture du robot. Ces résultats sont présentés Figure 2.25.

Ce cas (très particulier) permet de mettre en évidence une faiblesse de la méthode probabiliste : MCL recherche la meilleure posture possible et non pas la posture du



(a) Résultat pour l'algorithme IAL.



(b) Résultat pour l'algorithme MCL.

Figure 2.25 – Résultats des deux algorithmes de localisation après une localisation globale effectuée dans les conditions présentées sur la Figure 2.24a. On remarque que MCL a éliminé la posture du robot de ses solutions, alors que IAL a bien conservé cette dernière.

robot (même si ces deux postures sont équivalentes dans la plupart des cas). IAL ne souffre pas de cette approximation et permet de conserver toutes les solutions compatibles avec le problème de localisation. La méthode IAL apparaît donc comme étant plus rigoureuse que son homologue probabiliste.

3.3.3 Tableau récapitulatif

Pour finir cette comparaison, voici un tableau récapitulatif des différents avantages et inconvénients des deux méthodes.

IAL	MCL
Contexte à erreurs bornées.	Erreurs Gaussiennes.
Considère un nombre maximum de valeurs aberrantes.	Nécessite un nombre suffisant de particules.
Cherche toutes les postures cohérentes avec le problème.	Cherche la meilleure posture par rapport aux mesures extéroceptives.
Retourne une ou plusieurs boîte(s) contenant de manière garantie la posture du robot.	Retourne la particule avec le meilleur score (la plus probable).
Plus l'espace de recherche est grand plus les temps de calculs sont importants.	Temps de calcul constant en fonction du nombre de particules.
S'intéresse nativement aux problèmes de localisation globale, kidnapping et suivi de posture.	Il existe des variantes qui permettent de s'intéresser au kidnapping.
Permet d'effectuer des localisations statiques.	Utilise les déplacements du robot pour gagner en robustesse.
Méthode déterministe.	Méthode probabiliste.

3.4 Conclusion

Les différentes expérimentations effectuées permettent de valider l'algorithme de localisation globale proposé dans ce chapitre. Cette validation se fait à plusieurs niveaux.

Premièrement, il s'avère que les temps de calculs de la méthode proposée sont généralement comparables à ceux des méthodes classiquement utilisées en robotique mobile (MCL). L'algorithme ensembliste proposé ici n'est donc pas juste une approche théorique séduisante, mais s'avère intéressant dans un contexte pratique.

On notera que l'algorithme IAL est polyvalent, dans un sens où il permet nativement de s'intéresser aux problèmes de suivi de posture et de localisation globale. Ces deux problèmes ont été considérés lors des expérimentations et l'on remarque

que l'algorithme permet la localisation du robot dans les deux cas, sans avoir à y apporter de modification.

De plus, la méthode s'avère robuste et performante même dans des contextes difficiles. Les différentes expérimentations ont permis de valider l'algorithme :

- en présence de valeurs aberrantes dans le jeu de mesures,
- en présence d'erreurs sur la carte considérée,
- en présence de fortes incertitudes sur l'odométrie.

Cette robustesse rend l'algorithme utilisable en conditions réelles.

Pour conclure sur ces résultats, on notera que l'implémentation utilisée pour les expérimentations présentées ici, n'est pas optimale. En effet l'implémentation a été réalisée afin d'être fonctionnelle, sans contrainte d'efficacité. Tout laisse à penser que ces résultats peuvent être améliorés de façon significative juste en travaillant le code de l'implémentation.

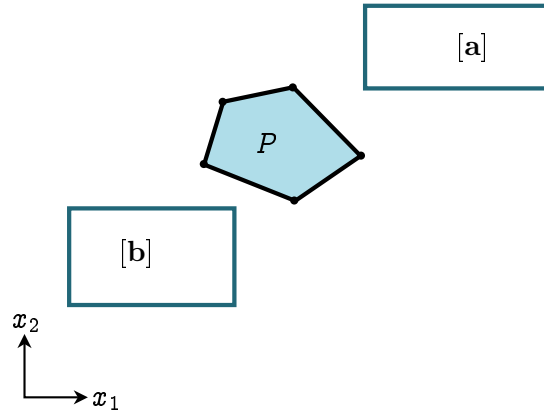
Contracteurs de visibilité

Sommaire

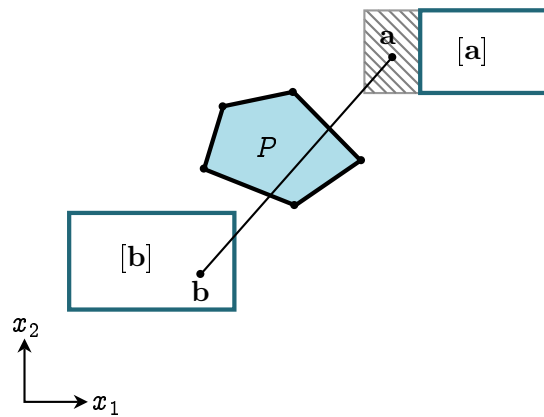
1	Définitions générales	81
1.1	Visibilité entre deux points	81
1.2	Espaces visible/non-visible d'un point	82
1.3	Espaces visible/non-visible/partiellement-visible d'un ensemble 83	
1.4	Visibilité considérant un ensemble d'obstacles	86
2	Cas particuliers de la visibilité	93
2.1	Visibilité d'un point	95
2.2	Visibilité d'un segment	98
2.3	Visibilité d'un polygone	105
3	Les contracteurs	112
3.1	Contracteurs de visibilité d'un point	114
3.2	Contracteurs de visibilité d'un segment	117
3.3	Contracteurs de visibilité d'un polygone	120

La *visibilité* est étudiée et utilisée dans une multitude de domaines : infographie, télécommunication, robotique... [Bittner 2003, Durand 2010]. En dessin assisté par ordinateur par exemple, les images de synthèses sont réalisées en simulant la propagation de la lumière au sein d'une scène. Les notions de visibilités sont alors primordiales pour le calcul des objets visibles depuis un point de vue, ou encore les calculs d'ombres et de pénombres.

En robotique mobile la visibilité est utilisée pour les problèmes de planification de trajectoires (graphe de visibilité) [Lozano-Pérez 1979, Huang 2004], ou encore pour les problèmes de localisation [Rekleitis 2000, Dellaert 2003, Giguere 2012]. C'est dans ce cadre que nous nous sommes intéressés à la visibilité.



(a) On ne veut pas que le segment défini par $a \in [a]$ et $b \in [b]$ intersecte le polygone P .



(b) Il est alors possible de contracter la boîtes $[a]$ (en retirant la partie hachurée).

Figure 3.1 – Nous voulons contracter les boîtes $[a]$ et $[b]$ pour que les segments définis par $a \in [a]$ et $b \in [b]$ n'intersectent pas le polygone P . Nous pouvons alors retirer la partie hachurée du pavé $[a]$ car il n'existe pas de segment avec une extrémité dans cette partie qui n'intersecte pas le polygone.

L'objectif de ce chapitre est de présenter des outils (contracteurs) permettant de résoudre le problème suivant : Soient deux boîtes $[a] \in \mathbb{IR}^2$ et $[b] \in \mathbb{IR}^2$, ainsi qu'un polygone convexe P (Figure 3.1a), on veut contracter les deux boîtes (les réduire) de façon à supprimer les zones pour lesquelles le segment défini par les points $a \in [a]$ et $b \in [b]$ intersecte (ou n'intersecte pas) le polygone P (Figure 3.1b).

Dans le chapitre suivant, ces outils seront utilisés dans deux applications, s'intéressant à des problèmes de localisation en robotique mobile.

1 Définitions générales

Avant de détailler ces contracteurs, il est nécessaire de définir un certain nombre de notions. Dans cette section nous introduisons la visibilité telle que considérée par la suite.

Nous présentons dans un premier temps, la relation de visibilité entre deux points par rapport à un obstacle. Un obstacle ε_j est défini comme étant un sous-ensemble connexe de \mathbb{R}^n .

1.1 Visibilité entre deux points

Définition 3.1 Soient deux points $x_1 \in \mathbb{R}^n$, $x_2 \in \mathbb{R}^n$ et un obstacle ε_j . La relation de visibilité entre les deux points par rapport à l'obstacle est définie par

$$(x_1 V x_2)_{\varepsilon_j} \Leftrightarrow \text{Seg}(x_1, x_2) \cap \varepsilon_j = \emptyset, \quad (3.1)$$

avec $\text{Seg}(x_1, x_2)$ le segment défini par les points x_1 et x_2 .

Le complémentaire de cette relation, la relation de non-visibilité, est notée

$$(x_1 V x_2)_{\varepsilon_j}^c = (x_1 \bar{V} x_2)_{\varepsilon_j}. \quad (3.2)$$

Remarque 3.1 Quelques remarques sur la relation de visibilité :

- la relation de visibilité est réflexive

$$(x_1 V x_2)_{\varepsilon_j} \Leftrightarrow (x_2 V x_1)_{\varepsilon_j}. \quad (3.3)$$

- la relation de visibilité est symétrique

$$(x_1 V x_1)_{\varepsilon_j}. \quad (3.4)$$

- la relation de visibilité n'est pas transitive (Figure 3.2)

$$(x_1 V x_2)_{\varepsilon_j} \wedge (x_2 V x_3)_{\varepsilon_j} \not\Rightarrow (x_1 V x_3)_{\varepsilon_j}. \quad (3.5)$$

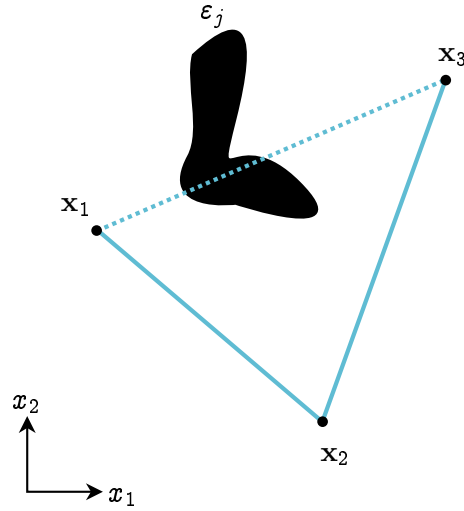


Figure 3.2 – Sur cet exemple : $(x_1 V x_2)_{\varepsilon_j}$, $(x_2 V x_3)_{\varepsilon_j}$ et $(x_1 \bar{V} x_3)_{\varepsilon_j}$.

- La relation de non-visibilité peut s'écrire de la forme

$$(x_1 \bar{V} x_2)_{\varepsilon_j} \Leftrightarrow \text{Seg}(x_1, x_2) \cap \varepsilon_j \neq \emptyset. \quad (3.6)$$

La Figure 3.2 présente des exemples de visibilité et non-visibilité entre deux points.

En utilisant ces deux relations, il est possible de définir les espaces visible et non-visible d'un point.

Relations

- Une relation \mathcal{R} dans \mathcal{X} est un sous ensemble de $\mathcal{X} \times \mathcal{X}$. En d'autre termes c'est une proposition qui relie deux éléments de \mathcal{X} .
- Une relation réflexive \mathcal{R} de l'ensemble \mathcal{X} est une relation pour laquelle

$$\forall a \in \mathcal{X}, a \mathcal{R} a$$

- Une relation symétrique \mathcal{R} de l'ensemble \mathcal{X} est une relation pour laquelle

$$\forall (a, b) \in \mathcal{X}^2, a \mathcal{R} b \Rightarrow b \mathcal{R} a$$

- Une relation transitive \mathcal{R} de l'ensemble \mathcal{X} est une relation pour laquelle

$$\forall (a, b, c) \in \mathcal{X}^3, a \mathcal{R} b \wedge b \mathcal{R} c \Rightarrow a \mathcal{R} c$$

1 .2 Espaces visible/non-visible d'un point

Définition 3.2 Soient un point $\mathbf{x} \in \mathbb{R}^n$ et un obstacle ε_j , avec $\mathbf{x} \notin \varepsilon_j$.

L'espace visible du point \mathbf{x} par rapport à l'obstacle ε_j est défini par

$$\mathbb{E}_{\varepsilon_j}(\mathbf{x}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid (\mathbf{x} \vee \mathbf{x}_i)_{\varepsilon_j}\}. \quad (3.7)$$

L'espace non-visible du point \mathbf{x} par rapport à l'obstacle ε_j est défini par

$$\widehat{\mathbb{E}}_{\varepsilon_j}(\mathbf{x}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid (\mathbf{x} \bar{\vee} \mathbf{x}_i)_{\varepsilon_j}\}. \quad (3.8)$$

Remarque 3.2

$$\mathbb{E}_{\varepsilon_j}^c(\mathbf{x}) = \widehat{\mathbb{E}}_{\varepsilon_j}(\mathbf{x}). \quad (3.9)$$

Remarque 3.3

$$(\mathbf{x}_1 \vee \mathbf{x}_2)_{\varepsilon_j} \Leftrightarrow \mathbf{x}_1 \in \mathbb{E}_{\varepsilon_j}(\mathbf{x}_2), \quad (3.10)$$

$$(\mathbf{x}_1 \vee \mathbf{x}_2)_{\varepsilon_j} \Leftrightarrow \mathbf{x}_2 \in \mathbb{E}_{\varepsilon_j}(\mathbf{x}_1), \quad (3.11)$$

$$\mathbf{x}_2 \in \mathbb{E}_{\varepsilon_j}(\mathbf{x}_1) \Leftrightarrow \mathbf{x}_1 \in \mathbb{E}_{\varepsilon_j}(\mathbf{x}_2). \quad (3.12)$$

Remarque 3.4

$$(\mathbf{x}_1 \bar{\vee} \mathbf{x}_2)_{\varepsilon_j} \Leftrightarrow \mathbf{x}_1 \in \widehat{\mathbb{E}}_{\varepsilon_j}(\mathbf{x}_2), \quad (3.13)$$

$$(\mathbf{x}_1 \bar{\vee} \mathbf{x}_2)_{\varepsilon_j} \Leftrightarrow \mathbf{x}_2 \in \widehat{\mathbb{E}}_{\varepsilon_j}(\mathbf{x}_1), \quad (3.14)$$

$$\mathbf{x}_2 \in \widehat{\mathbb{E}}_{\varepsilon_j}(\mathbf{x}_1) \Leftrightarrow \mathbf{x}_1 \in \widehat{\mathbb{E}}_{\varepsilon_j}(\mathbf{x}_2). \quad (3.15)$$

Les Figures 3.3 et 3.4 présentent des exemples d'espaces visibles et non-visibles.

Ces espaces, définis pour les points, peuvent être généralisés aux ensembles connexes de \mathbb{R}^n .

1 .3 Espaces visible/non-visible/partiellement-visible d'un ensemble

Définition 3.3 Soient un ensemble connexe $\mathbb{X} \subset \mathbb{R}^n$ et un obstacle ε_j , avec $\mathbb{X} \cap \varepsilon_j = \emptyset$.

L'espace visible de \mathbb{X} par rapport à ε_j est défini par

$$\mathbb{E}_{\varepsilon_j}(\mathbb{X}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \vee \mathbf{x})_{\varepsilon_j}\}. \quad (3.16)$$

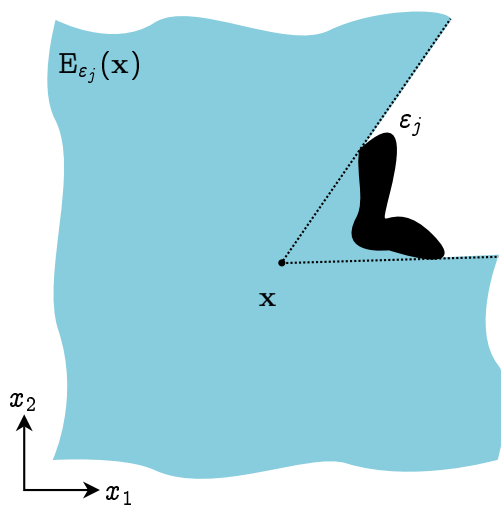


Figure 3.3 – Exemple d'espace visible.

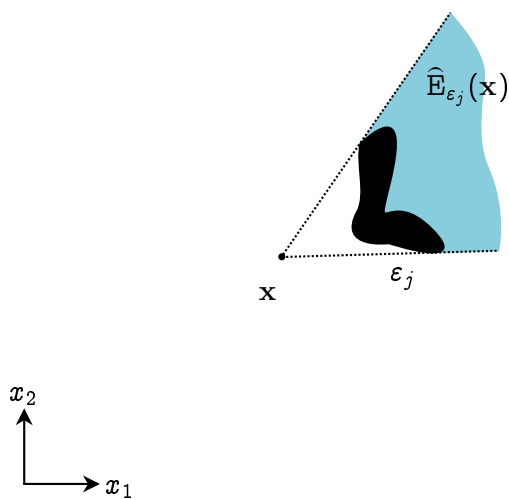


Figure 3.4 – Exemple d'espace non-visible.

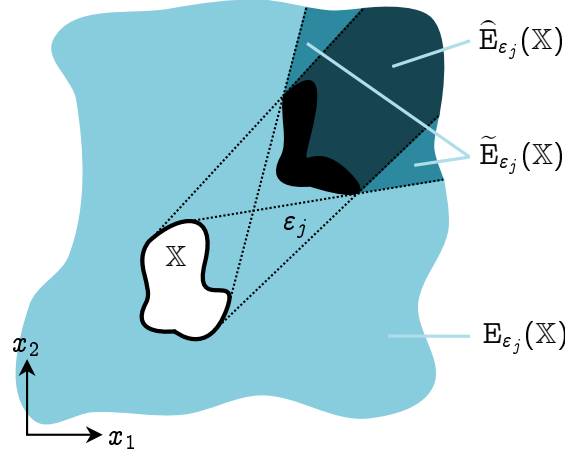


Figure 3.5 – Exemple d’espaces visible, non-visible et partiellement visible.

L’espace non-visible de \mathbb{X} par rapport à ε_j est défini par

$$\widehat{\mathbb{E}}_{\varepsilon_j}(\mathbb{X}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \bar{\mathbf{V}}\mathbf{x})_{\varepsilon_j}\}. \quad (3.17)$$

L’espace partiellement-visible de \mathbb{X} par rapport à ε_j est défini par

$$\widetilde{\mathbb{E}}_{\varepsilon_j}(\mathbb{X}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid \exists \mathbf{x}_1 \in \mathbb{X}, \exists \mathbf{x}_2 \in \mathbb{X}, (\mathbf{x}_i \mathbf{V}\mathbf{x}_1)_{\varepsilon_j} \wedge (\mathbf{x}_i \bar{\mathbf{V}}\mathbf{x}_2)_{\varepsilon_j}\}. \quad (3.18)$$

Il est important de noter que la relation de visibilité, binaire quand on s’intéresse à des points, devient ternaire quand on manipule des ensembles. La relation partiellement-visible correspond classiquement à la pénombre (seulement visible par une partie de la lumière). La Figure 3.5 présente un exemple d’espaces visible, non-visible et partiellement visible.

La Proposition 3.1 permet de lier la visibilité d’un ensemble à la visibilité des points qui le compose. C’est sur cette proposition que reposent les différents contracteurs présentés Section 3 .

Proposition 3.1 Soient un ensemble connexe $\mathbb{X} \subset \mathbb{R}^n$, un obstacle ε_j , avec $\mathbb{X} \cap \varepsilon_j = \emptyset$, un point $\mathbf{x} \in \mathbb{X}$ et un point $\mathbf{x}_i \in \mathbb{R}^n$, de telle sorte que $\mathbf{x}_i \notin \mathbb{X}$. On a

$$(\mathbf{x}_i \mathbf{V}\mathbf{x})_{\varepsilon_j} \Rightarrow \mathbf{x}_i \in \mathbb{E}_{\varepsilon_j}(\mathbb{X}) \cup \widetilde{\mathbb{E}}_{\varepsilon_j}(\mathbb{X}), \quad (3.19)$$

$$(\mathbf{x}_i \bar{\mathbf{V}}\mathbf{x})_{\varepsilon_j} \Rightarrow \mathbf{x}_i \in \widehat{\mathbb{E}}_{\varepsilon_j}(\mathbb{X}) \cup \widetilde{\mathbb{E}}_{\varepsilon_j}(\mathbb{X}). \quad (3.20)$$

Démonstration

$$\begin{aligned}
(\mathbf{x}_i \mathbf{V} \mathbf{x})_{\varepsilon_j} &\Rightarrow \mathbf{x}_i \notin \widehat{\mathbf{E}}_{\varepsilon_j}(\mathbb{X}) \text{ (Eq. 3.17)} \\
(\mathbf{x}_i \mathbf{V} \mathbf{x})_{\varepsilon_j} &\Rightarrow \mathbf{x}_i \in \widetilde{\mathbf{E}}_{\varepsilon_j}(\mathbb{X}) \cup \mathbf{E}_{\varepsilon_j}(\mathbb{X}) \\
(\mathbf{x}_i \overline{\mathbf{V}} \mathbf{x})_{\varepsilon_j} &\Rightarrow \mathbf{x}_i \notin \mathbf{E}_{\varepsilon_j}(\mathbb{X}) \text{ (Eq. 3.16)} \\
(\mathbf{x}_i \overline{\mathbf{V}} \mathbf{x})_{\varepsilon_j} &\Rightarrow \mathbf{x}_i \in \widetilde{\mathbf{E}}_{\varepsilon_j}(\mathbb{X}) \cup \widehat{\mathbf{E}}_{\varepsilon_j}(\mathbb{X})
\end{aligned}$$

Les notions présentées jusqu'à présent ne considèrent qu'un seul obstacle. Nous nous intéressons maintenant à la visibilité par rapport à un ensemble d'obstacles, aussi appelé environnement.

1.4 Visibilité considérant un ensemble d'obstacles

On appelle *environnement* de \mathbb{R}^n , dénoté \mathcal{E} , un ensemble de $n_{\mathcal{O}}$ obstacles

$$\mathcal{E} = \bigcup_{j=1}^{n_{\mathcal{O}}} \varepsilon_j, \quad (3.21)$$

avec $\varepsilon_1, \dots, \varepsilon_j, \dots, \varepsilon_{n_{\mathcal{O}}}$ des obstacles de \mathbb{R}^n .

Il est possible d'étendre les définitions précédentes à un environnement :

$$(\mathbf{x}_1 \mathbf{V} \mathbf{x}_2)_{\mathcal{E}} \Leftrightarrow \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \mathcal{E} = \emptyset, \quad (3.22)$$

$$(\mathbf{x}_1 \overline{\mathbf{V}} \mathbf{x}_2)_{\mathcal{E}} \Leftrightarrow \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \mathcal{E} \neq \emptyset, \quad (3.23)$$

$$\mathbf{E}_{\mathcal{E}}(\mathbf{x}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid (\mathbf{x}_i \mathbf{V} \mathbf{x})_{\mathcal{E}}\}, \quad (3.24)$$

$$\mathbf{E}_{\mathcal{E}}^c(\mathbf{x}) = \widehat{\mathbf{E}}_{\mathcal{E}}(\mathbf{x}), \quad (3.25)$$

$$\mathbf{E}_{\mathcal{E}}(\mathbb{X}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x} \mathbf{V} \mathbf{x}_i)_{\mathcal{E}}\}, \quad (3.26)$$

$$\widehat{\mathbf{E}}_{\mathcal{E}}(\mathbb{X}) = \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x} \overline{\mathbf{V}} \mathbf{x}_i)_{\mathcal{E}}\}. \quad (3.27)$$

Nous voulons maintenant relier les relations de visibilité d'un point dans un environnement, aux relations de visibilité du point par rapports aux différents obstacles de l'environnement.

Proposition 3.2 Soient deux points distincts $\mathbf{x}_1 \in \mathbb{R}^n$, $\mathbf{x}_2 \in \mathbb{R}^n$ et un environnement \mathcal{E} de \mathbb{R}^n composé de $n_{\mathcal{O}}$ obstacles, avec $\mathbf{x}_1 \notin \mathcal{E}$ et $\mathbf{x}_2 \notin \mathcal{E}$. On a

$$(\mathbf{x}_1 \mathbf{V} \mathbf{x}_2)_{\mathcal{E}} \Leftrightarrow \bigwedge_{j=1}^{n_{\mathcal{O}}} (\mathbf{x}_1 \mathbf{V} \mathbf{x}_2)_{\varepsilon_j}, \quad (3.28)$$

$$(\mathbf{x}_1 \overline{\mathbf{V}} \mathbf{x}_2)_{\mathcal{E}} \Leftrightarrow \bigvee_{j=1}^{n_{\mathcal{O}}} (\mathbf{x}_1 \overline{\mathbf{V}} \mathbf{x}_2)_{\varepsilon_j}. \quad (3.29)$$

Démonstration (Eq. 3.28)

$$\begin{aligned}
(\mathbf{x}_1 \mathbf{V} \mathbf{x}_2)_{\mathcal{E}} &\Leftrightarrow \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \mathcal{E} = \emptyset \text{ (Eq. 3.22),} \\
&\Leftrightarrow \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \left(\bigcup_{j=1}^{n_o} \varepsilon_j \right) = \emptyset \text{ (Eq. 3.21),} \\
&\Leftrightarrow \bigcup_{j=1}^{n_o} (\text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j) = \emptyset, \\
&\Leftrightarrow \forall \varepsilon_j \in \mathcal{E}, \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j = \emptyset, \\
&\Leftrightarrow \forall \varepsilon_j \in \mathcal{E}, (\mathbf{x}_1 \mathbf{V} \mathbf{x}_2)_{\varepsilon_j} \text{ (Eq. 3.1),} \\
(\mathbf{x}_1 \mathbf{V} \mathbf{x}_2)_{\mathcal{E}} &\Leftrightarrow \bigwedge_{j=1}^{n_o} (\mathbf{x}_1 \mathbf{V} \mathbf{x}_2)_{\varepsilon_j}.
\end{aligned}$$

Démonstration (Eq. 3.29)

$$\begin{aligned}
(\mathbf{x}_1 \bar{\mathbf{V}} \mathbf{x}_2)_{\mathcal{E}} &\Leftrightarrow \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \mathcal{E} \neq \emptyset \text{ (Eq. 3.23),} \\
&\Leftrightarrow \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \left(\bigcup_{j=1}^{n_o} \varepsilon_j \right) \neq \emptyset \text{ (Eq. 3.21),} \\
&\Leftrightarrow \bigcup_{j=1}^{n_o} (\text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j) \neq \emptyset, \\
&\Leftrightarrow \exists \varepsilon_j \in \mathcal{E} | \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j \neq \emptyset, \\
&\Leftrightarrow \exists \varepsilon_j \in \mathcal{E} | (\mathbf{x}_1 \bar{\mathbf{V}} \mathbf{x}_2)_{\varepsilon_j} \text{ (Eq. 3.6),} \\
(\mathbf{x}_1 \bar{\mathbf{V}} \mathbf{x}_2)_{\mathcal{E}} &\Leftrightarrow \bigvee_{j=1}^{n_o} (\mathbf{x}_1 \bar{\mathbf{V}} \mathbf{x}_2)_{\varepsilon_j}.
\end{aligned}$$

La Figure 3.6 illustre la Propositions 3.2.

Il est aussi possible de relier les espaces de visibilité d'un point dans un environnement aux espaces de visibilité du point par rapport aux différents obstacles de l'environnement.

Proposition 3.3 Soient un point $\mathbf{x} \in \mathbb{R}^n$ et un environnement \mathcal{E} composé de n_o obstacles, avec $\mathbf{x} \notin \mathcal{E}$. On a alors

$$E_{\mathcal{E}}(\mathbf{x}) = \bigcap_{j=1}^{n_o} E_{\varepsilon_j}(\mathbf{x}), \quad (3.30)$$

$$\hat{E}_{\mathcal{E}}(\mathbf{x}) = \bigcup_{j=1}^{n_o} \hat{E}_{\varepsilon_j}(\mathbf{x}). \quad (3.31)$$

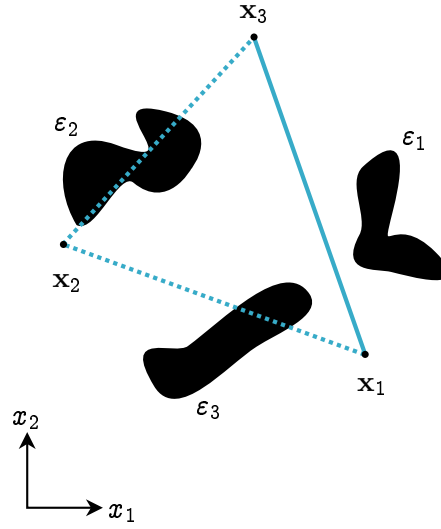


Figure 3.6 – Sur cet exemple nous avons $(x_1 V x_3)_{\varepsilon_1}$ et $(x_1 V x_3)_{\varepsilon_2}$ et $(x_1 V x_3)_{\varepsilon_3}$, on peut donc en conclure que $(x_1 V x_3)_{\mathcal{E}}$. On peut aussi remarquer que $(x_1 V x_2)_{\varepsilon_1}$ et $(x_1 V x_2)_{\varepsilon_2}$ et $(x_1 \bar{V} x_2)_{\varepsilon_3}$, on peut alors conclure que $(x_1 \bar{V} x_2)_{\mathcal{E}}$. Le même raisonnement s'applique pour x_2 et x_3 : $(x_2 V x_3)_{\varepsilon_1}$ et $(x_2 \bar{V} x_3)_{\varepsilon_2}$ et $(x_2 V x_3)_{\varepsilon_3}$, donc $(x_2 \bar{V} x_3)_{\mathcal{E}}$.

Démonstration (Eq. 3.30)

$$\begin{aligned}
 E_{\mathcal{E}}(\mathbf{x}) &= \{x_i \in \mathbb{R}^n \mid (x V x_i)_{\mathcal{E}}\} \text{ (Eq. 3.24),} \\
 &= \{x_i \in \mathbb{R}^n \mid \bigwedge_{j=1}^{n_O} (x V x_i)_{\varepsilon_j}\} \text{ (Eq. 3.28),} \\
 &= \{x_i \in \mathbb{R}^n \mid (x V x_i)_{\varepsilon_1}\} \cap \dots \cap \{x_i \in \mathbb{R}^n \mid (x V x_i)_{\varepsilon_j}\} \cap \dots \cap \{x_i \in \mathbb{R}^n \mid (x V x_i)_{\varepsilon_{n_O}}\}, \\
 &= E_{\varepsilon_1}(\mathbf{x}) \cap \dots \cap E_{\varepsilon_j}(\mathbf{x}) \cap \dots \cap E_{\varepsilon_{n_O}}(\mathbf{x}) \text{ (Eq. 3.7),} \\
 E_{\mathcal{E}}(\mathbf{x}) &= \bigcap_{j=1}^{n_O} E_{\varepsilon_j}(\mathbf{x}).
 \end{aligned}$$

Démonstration (Eq. 3.31)

$$\begin{aligned}
 \widehat{E}_{\mathcal{E}}(\mathbf{x}) &= E_{\mathcal{E}}^c(\mathbf{x}) \text{ (Eq. 3.25),} \\
 &= \left(\bigcap_{j=1}^{n_O} E_{\varepsilon_j}(\mathbf{x}) \right)^c \text{ (Eq. 3.30),} \\
 &= \bigcup_{j=1}^{n_O} E_{\varepsilon_j}^c(\mathbf{x}), \\
 \widehat{E}_{\mathcal{E}}(\mathbf{x}) &= \bigcup_{j=1}^{n_O} \widehat{E}_{\varepsilon_j}(\mathbf{x}) \text{ (Eq. 3.9).}
 \end{aligned}$$

Les Figures 3.7 et 3.8 illustrent la Proposition 3.3.

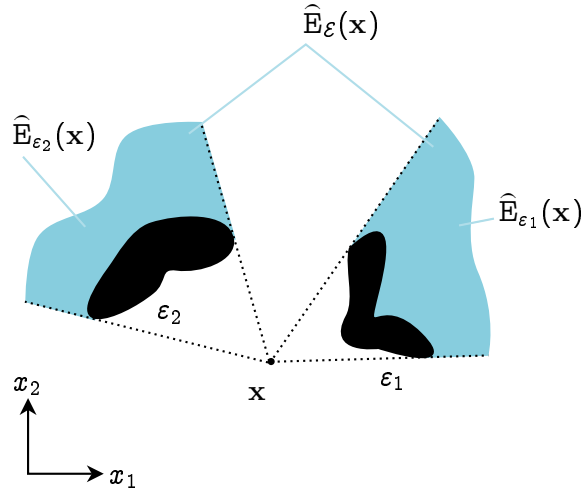


Figure 3.7 – Sur cet exemple nous avons bien $\widehat{E}_{\mathcal{E}}(\mathbf{x}) = \widehat{E}_{\varepsilon_1}(\mathbf{x}) \cup \widehat{E}_{\varepsilon_2}(\mathbf{x})$.

Ce travail de relation fait sur les points, peut aussi se faire en considérant les ensembles connexes. Il est possible de relier les espaces de visibilité d'un ensemble dans un environnement, aux espaces de visibilité de l'ensemble par rapports aux différents obstacles de l'environnement.

Proposition 3.4 Soient un ensemble connexe $\mathbb{X} \subset \mathbb{R}^n$ et un environnement \mathcal{E} composé de n_O obstacles, avec $\mathbb{X} \cap \mathcal{E} = \emptyset$. On a

$$E_{\mathcal{E}}(\mathbb{X}) = \bigcap_{j=1}^{n_O} E_{\varepsilon_j}(\mathbb{X}), \quad (3.32)$$

$$\widehat{E}_{\mathcal{E}}(\mathbb{X}) \supseteq \bigcup_{j=1}^{n_O} \widehat{E}_{\varepsilon_j}(\mathbb{X}). \quad (3.33)$$

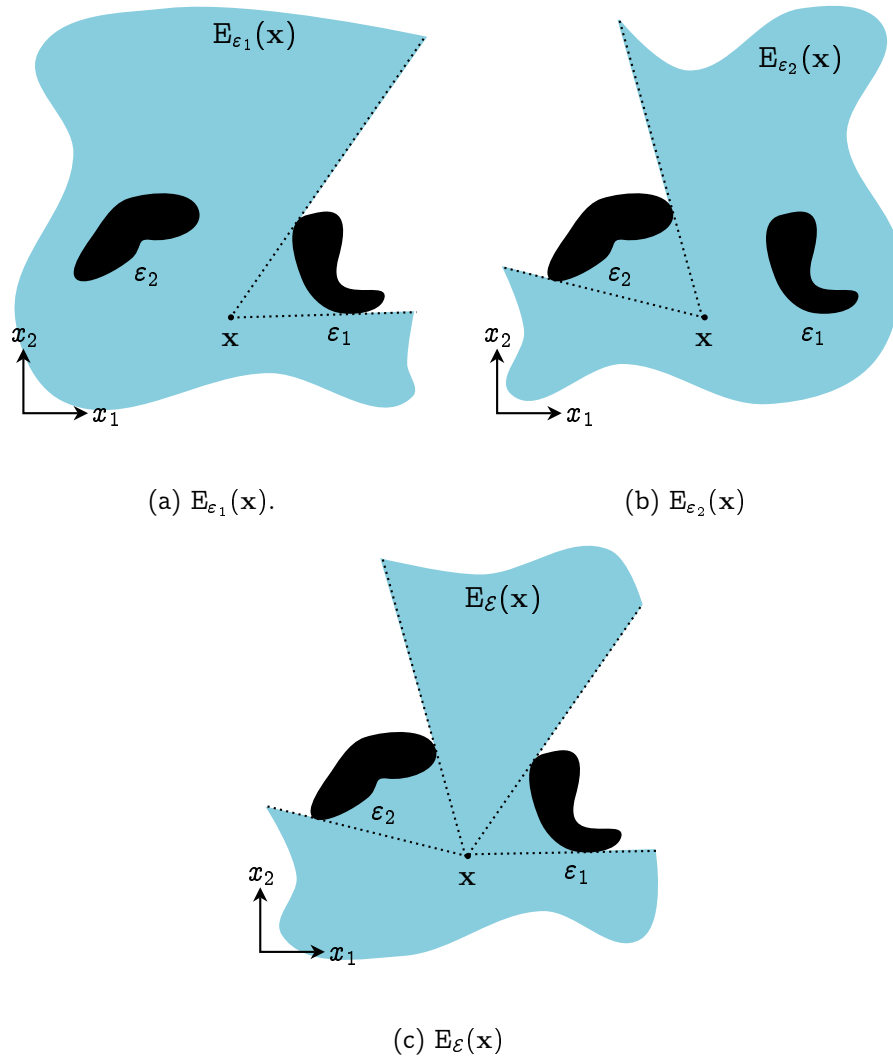


Figure 3.8 – Sur cet exemple nous avons bien $E_{\mathcal{L}}(x) = E_{\epsilon_1}(x) \cap E_{\epsilon_2}(x)$.

Démonstration (Eq. 3.32)

$$\begin{aligned}
E_{\mathcal{E}}(\mathbb{X}) &= \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x} V \mathbf{x}_i)_{\mathcal{E}}\} \text{ (Eq. 3.26),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, \bigwedge_{j=1}^{n_o} (\mathbf{x}_i V \mathbf{x})_{\varepsilon_j}\} \text{ (Eq. 3.28),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i V \mathbf{x})_{\varepsilon_1}\} \cap \dots \cap \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i V \mathbf{x})_{\varepsilon_j}\} \cap \dots \\
&\quad \dots \cap \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i V \mathbf{x})_{\varepsilon_{n_o}}\}, \\
&= \bigcap_{j=1}^{n_o} \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i V \mathbf{x})_{\varepsilon_j}\}, \\
E_{\mathcal{E}}(\mathbb{X}) &= \bigcap_{j=1}^m E_{\mathcal{E}_j}(\mathbb{X}) \text{ (Eq. 3.16).}
\end{aligned}$$

Démonstration (Eq. 3.33)

$$\begin{aligned}
\widehat{E}_{\mathcal{E}}(\mathbb{X}) &= \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \bar{V} \mathbf{x})_{\mathcal{E}}\} \text{ (Eq. 3.27),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, \bigvee_{j=1}^{n_o} (\mathbf{x}_i \bar{V} \mathbf{x})_{\varepsilon_j}\} \text{ (Eq. 3.1).} \\
\bigcup_{j=1}^{n_o} \widehat{E}_{\varepsilon_j}(\mathbb{X}) &= \bigcup_{j=1}^{n_o} \{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \bar{V} \mathbf{x})_{\varepsilon_j}\} \text{ (Eq. 3.17),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^n \mid \bigvee_{j=1}^{n_o} [\forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \bar{V} \mathbf{x})_{\varepsilon_j}]\}. \\
\{\mathbf{x}_i \in \mathbb{R}^n \mid \forall \mathbf{x} \in \mathbb{X}, \bigvee_{j=1}^{n_o} (\mathbf{x}_i \bar{V} \mathbf{x})_{\varepsilon_j}\} &\supseteq \{\mathbf{x}_i \in \mathbb{R}^n \mid \bigvee_{j=1}^{n_o} [\forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \bar{V} \mathbf{x})_{\varepsilon_j}]\}, \\
\Rightarrow \widehat{E}_{\mathcal{E}}(\mathbb{X}) &\supseteq \bigcup_{j=1}^{n_o} \widehat{E}_{\varepsilon_j}(\mathbb{X}).
\end{aligned}$$

Le lecteur remarquera que pour l'espace non visible d'un ensemble, Équation 3.33, nous n'avons plus une égalité, comme c'était le cas Équation 3.31, mais une inclusion. La Figure 3.9 illustre cette inclusion.

Ceci termine les définitions générales concernant les notions de visibilité. Afin de pouvoir résoudre le problème présenté en introduction de ce chapitre, nous allons maintenant nous intéresser à deux types d'obstacles particuliers de \mathbb{R}^2 : les segments et les polygones. Notre objectif final étant de pouvoir contracter au regard d'un obstacle de type polygone, il est nécessaire de s'intéresser à ce type d'obstacle. De plus les polygones peuvent être associés à des ensembles de segments, c'est pourquoi nous nous intéressons aussi aux obstacles de type segment.

Comme présenté précédemment, notre objectif est de pouvoir contracter des boîtes. Ces boîtes peuvent être associées à des polygones, ces polygones peuvent être

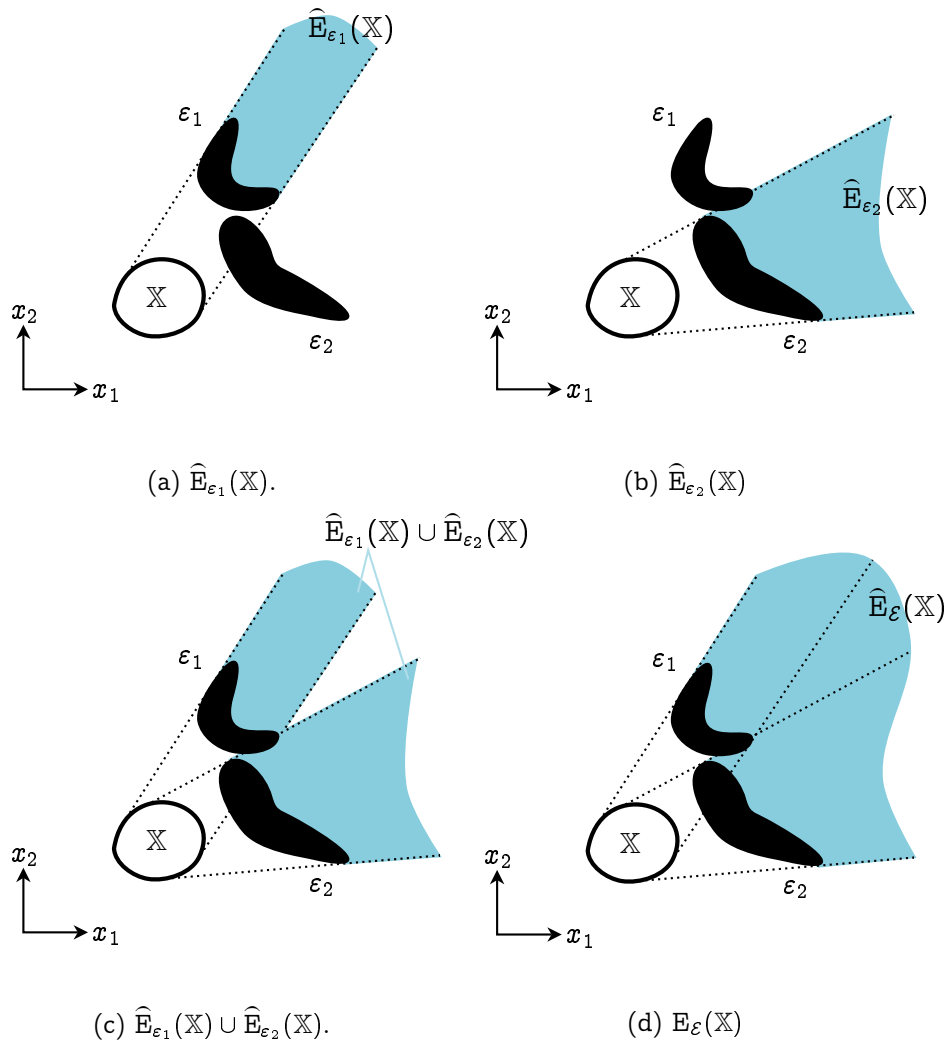


Figure 3.9 – Sur cet exemple on remarque bien que $\widehat{E}_{\mathcal{E}}(\mathbb{X}) \supseteq \widehat{E}_{\varepsilon_1}(\mathbb{X}) \cup \widehat{E}_{\varepsilon_2}(\mathbb{X})$.

définis par des ensemble de segments et ces segments peuvent être caractérisés par des points (les extrémités). Nous allons donc nous intéresser à la visibilité de trois *type de sources* différents : la visibilité d'un point, d'un segment et finalement d'un polygone.

2 Cas particuliers de la visibilité

Dans cette section nous nous intéressons à la visibilité de trois types de sources de \mathbb{R}^2 (point, segment et polygone) par rapport à deux types d'obstacles de \mathbb{R}^2 (segment et polygone). L'intérêt de l'étude de ces cas particuliers, c'est que pour ces derniers les espaces de visibilité peuvent tous être décrits par des ensembles d'inégalités. On notera que cette section utilise des outils présentés dans l'Annexe C.

Pour un obstacle de type segment, nous utilisons la notation

$$\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j}), \quad (3.34)$$

avec $\mathbf{e}_{1_j} \in \mathbb{R}^2$ et $\mathbf{e}_{2_j} \in \mathbb{R}^2$ deux points distincts qui représentent les extrémités du segment ε_j^s .

On appelle ε_j^p l'obstacle correspondant au polygone convexe (Annexe C, Section 1.3.1) composé de n_{P_j} sommets, nommés $\mathbf{e}_1, \dots, \mathbf{e}_k, \dots, \mathbf{e}_{n_{P_j}}$, dans le sens trigonométrique, avec $\mathbf{e}_k = (e_{1_k}, e_{2_k}) \in \mathbb{R}^2$.

ε_j^p peut aussi être vue comme un environnement fermé composé de n_{P_j} obstacles :

$$\varepsilon_j^p \equiv \bigcup_{k=1}^{n_{P_j}} \text{Seg}(\mathbf{e}_k, \mathbf{e}_{k+1}) \quad (3.35)$$

Pour la suite on notera

$$\text{Seg}(\mathbf{e}_k, \mathbf{e}_{k+1}) = \varepsilon_k^s. \quad (3.36)$$

L'intérêt de manipuler des polygones convexes tels que définis Annexe C, réside dans la Proposition 3.5 (On se rappelle que pour les obstacles quelconques on a l'inclusion et non pas l'égalité, Équation 3.33).

Proposition 3.5 Soient un ensemble connexe $\mathbb{X} \in \mathbb{R}^2$ et un obstacle ε_j^p composé de n_{P_j} sommets avec $\mathbb{X} \cap \varepsilon_j^p = \emptyset$. On a

$$\widehat{\mathbb{E}}_{\varepsilon_j^p}(\mathbb{X}) = \bigcup_{k=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_k^s}(\mathbb{X}). \quad (3.37)$$

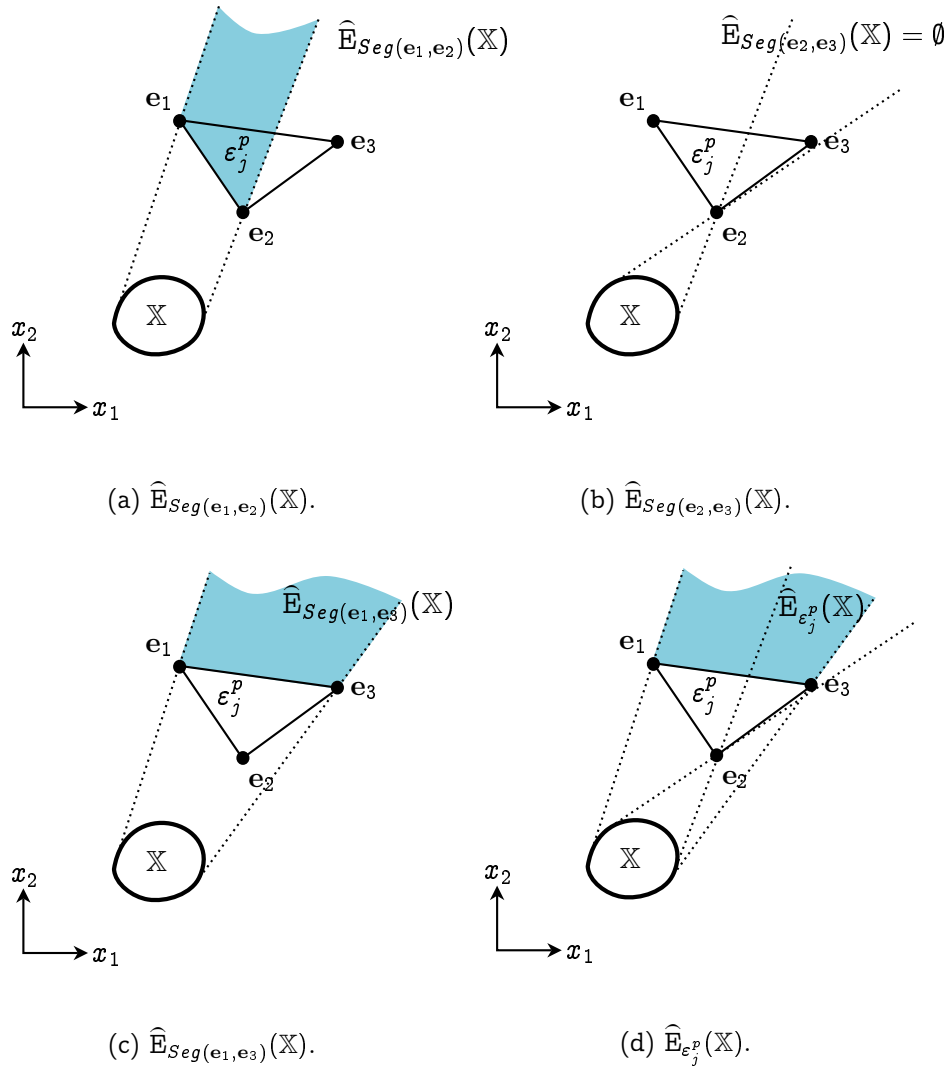


Figure 3.10 – Espace non-visible d'un ensemble connexe \mathbb{X} par rapport à un polygone ε_j^p .

Démonstration

$$\begin{aligned}
\widehat{E}_{\varepsilon_j^p}(\mathbb{X}) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \bar{V} \mathbf{x})_{\varepsilon_j^p}\} \text{ (Eq. 3.17),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x} \in \mathbb{X}, \text{Seg}(\mathbf{x}_i, \mathbf{x}) \cap \varepsilon_j^p \neq \emptyset\} \text{ (Eq. 3.6),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x} \in \mathbb{X}, \text{Seg}(\mathbf{x}_i, \mathbf{x}) \cap \left(\bigcup_{k=1}^{n_{P_j}} \varepsilon_k^s \right) \neq \emptyset\} \text{ (Eq. C.25),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x} \in \mathbb{X}, \bigcup_{k=1}^{n_{P_j}} (\text{Seg}(\mathbf{x}_i, \mathbf{x}) \cap \varepsilon_k^s) \neq \emptyset\}, \\
&= \bigcup_{k=1}^{n_{P_j}} \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x} \in \mathbb{X}, \text{Seg}(\mathbf{x}_i, \mathbf{x}) \cap \varepsilon_k^s \neq \emptyset\}, \\
&= \bigcup_{k=1}^{n_{P_j}} \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x} \in \mathbb{X}, (\mathbf{x}_i \bar{V} \mathbf{x})_{\varepsilon_k^s}\} \text{ (Eq. 3.6),} \\
\widehat{E}_{\varepsilon_j^p}(\mathbb{X}) &= \bigcup_{k=1}^{n_{P_j}} \widehat{E}_{\varepsilon_k^s}(\mathbb{X}) \text{ (Eq. 3.17).}
\end{aligned}$$

La Figure 3.10 illustre la Proposition 3.5.

2 .1 Visibilité d'un point

Nous nous intéressons dans un premier temps à la visibilité d'un point.

2 .1.1 Considérant un segment comme obstacle

La proposition qui suit caractérise l'espace visible d'un point par rapport à un obstacle du type segment.

Proposition 3.6 *Soient un point $\mathbf{x} \in \mathbb{R}^2$ et un obstacle $\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$ avec $\mathbf{x} \notin \varepsilon_j^s$. On a*

$$\begin{aligned}
E_{\varepsilon_j^s}(\mathbf{x}) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid [\mathbf{x}_i \cup \mathbf{x}] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \vee \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\
&\quad \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x} - \mathbf{e}_{1_j}) > 0 \vee \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x} - \mathbf{e}_{2_j}) < 0\}, \quad (3.38)
\end{aligned}$$

avec

$$\zeta_{\mathbf{x}} = \begin{cases} 1 & \text{si } \det(\mathbf{x} - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0, \\ -1 & \text{sinon.} \end{cases} \quad (3.39)$$

Démonstration

$$\begin{aligned}
\mathbb{E}_{\varepsilon_j^s}(\mathbf{x}) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid (\mathbf{x}V\mathbf{x}_i)_{\varepsilon_j^s}\} \text{ (Eq. 3.7),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \text{Seg}(\mathbf{x}, \mathbf{x}_i) \cap \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j}) = \emptyset\} \text{ (Eq. 3.1),} \\
&= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \begin{aligned} &\det(\mathbf{x} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \cdot \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \quad \vee \\ &\det(\mathbf{e}_{1_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) \cdot \det(\mathbf{e}_{2_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) > 0 \quad \vee \\ &[\mathbf{x} \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \end{aligned} \text{ (Eq. C.5).} \end{aligned}$$

D'après la Proposition C.4 (Annexe C) :

$$\begin{aligned}
&\det(\mathbf{x} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \cdot \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\
&\det(\mathbf{e}_{1_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) \cdot \det(\mathbf{e}_{2_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) > 0 \\
\Leftrightarrow &\zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) > 0 \vee \\
&\zeta_x \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) < 0. \tag{3.40}
\end{aligned}$$

Nous avons donc

$$\begin{aligned}
\mathbb{E}_{\varepsilon_j^s}(\mathbf{x}) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid [\mathbf{x}_i \cup \mathbf{x}] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \vee \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\
&\zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) > 0 \vee \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) < 0\},
\end{aligned}$$

La proposition qui suit caractérise l'espace non-visible d'un point par rapport à un obstacle du type segment.

Proposition 3.7 Soient un point $\mathbf{x} \in \mathbb{R}^2$ et un obstacle $\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$ avec $\mathbf{x} \notin \varepsilon_j^s$. On a

$$\begin{aligned}
\widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \begin{aligned} &\zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \quad \wedge \\ &\zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) \leq 0 \quad \wedge \\ &\zeta_x \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) \geq 0 \quad \wedge \\ &[\mathbf{x} \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] \neq \emptyset \end{aligned} \}, \tag{3.41}
\end{aligned}$$

avec

$$\zeta_x = \begin{cases} 1 & \text{si } \det(\mathbf{x} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0, \\ -1 & \text{sinon.} \end{cases}$$

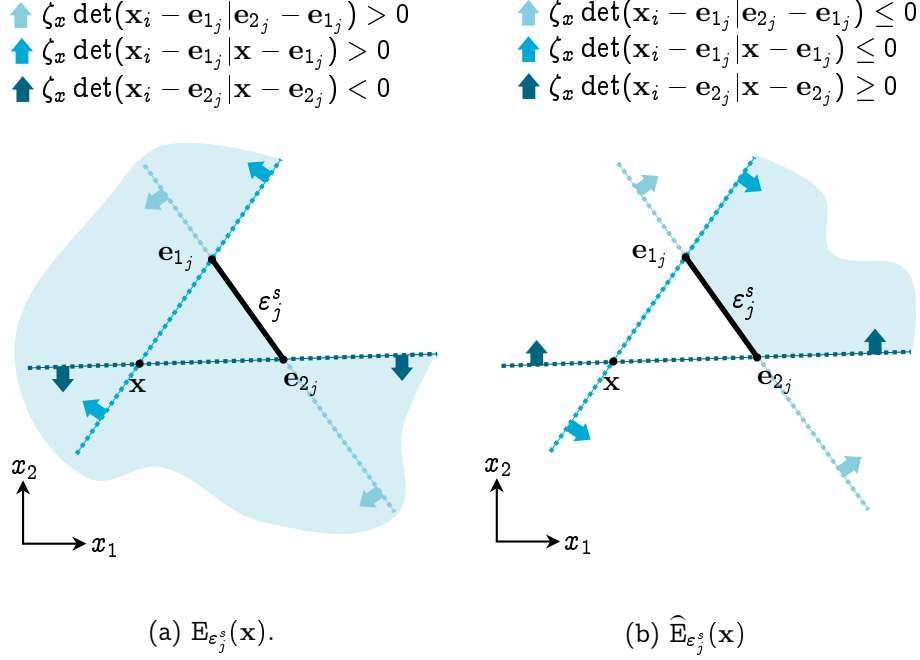


Figure 3.11 – Exemple de caractérisation des espaces visible et non-visible d'un point par rapport à un segment.

Démonstration

$$\begin{aligned}
 \widehat{E}_{\varepsilon_j^s}(\mathbf{x}) &= (E_{\varepsilon_j^s}(\mathbf{x}))^c \text{ (Eq. 3.9),} \\
 &= \left(\{ \mathbf{x}_i \in \mathbb{R}^2 \mid [\mathbf{x} \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \vee \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \right. \\
 &\quad \left. \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) > 0 \vee \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) < 0 \} \right)^c \text{ (Eq. 3.38),} \\
 &= \{ \mathbf{x}_i \in \mathbb{R}^2 \mid ([\mathbf{x} \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}]) \neq \emptyset \vee \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\
 &\quad \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) > 0 \vee \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) < 0 \}^c, \\
 \widehat{E}_{\varepsilon_j^s}(\mathbf{x}) &= \{ \mathbf{x}_i \in \mathbb{R}^2 \mid [\mathbf{x} \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] \neq \emptyset \wedge \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
 &\quad \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_x \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) \geq 0 \}.
 \end{aligned}$$

La Figure 3.11 illustre les Propositions 3.6 et 3.7.

2 .1.2 Considérant un polygone comme obstacle

Nous nous intéressons maintenant aux espaces de visibilité d'un point par rapport à un polygone, tel que défini précédemment.

La proposition qui suit permet de caractériser l'espace visible d'un point par rapport à un polygone.

Proposition 3.8 *Soient un point $\mathbf{x} \in \mathbb{R}^2$ et un polygone convexe ε_j^p composé de n_{P_j} sommets, avec $\mathbf{x} \notin \varepsilon_j^p$. On a*

$$\mathbb{E}_{\varepsilon_j^p}(\mathbf{x}) = \bigcap_{k=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_k^s}(\mathbf{x}). \quad (3.42)$$

Démonstration

$$\mathbb{E}_{\varepsilon_j^p}(\mathbf{x}) = \bigcap_{k=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_k^s}(\mathbf{x}) \text{ (Eq. 3.30 et 3.35).}$$

La proposition qui suit permet de caractériser l'espace non-visible d'un point par rapport à un polygone.

Proposition 3.9 *Soient un point $\mathbf{x} \in \mathbb{R}^2$ et un obstacle ε_j^p composé de n_{P_j} sommets, avec $\mathbf{x} \notin \varepsilon_j^p$. On a*

$$\widehat{\mathbb{E}}_{\varepsilon_j^p}(\mathbf{x}) = \bigcup_{k=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_k^s}(\mathbf{x}). \quad (3.43)$$

Démonstration

$$\widehat{\mathbb{E}}_{\varepsilon_j^p}(\mathbf{x}) = \bigcup_{k=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_k^s}(\mathbf{x}) \text{ (Eq. 3.31 et 3.35).}$$

La Figure 3.12 illustre les Propositions 3.8 et 3.9.

La caractérisation des espaces visible et non-visible sont maintenant disponibles pour un point. Nous pouvons à présent nous intéresser aux espaces de visibilité d'un segment.

2.2 Visibilité d'un segment

Un segment peut être considéré comme un ensemble connexe particulier de \mathbb{R}^2 . Nous allons alors pouvoir utiliser les propositions de la Section 1.3.

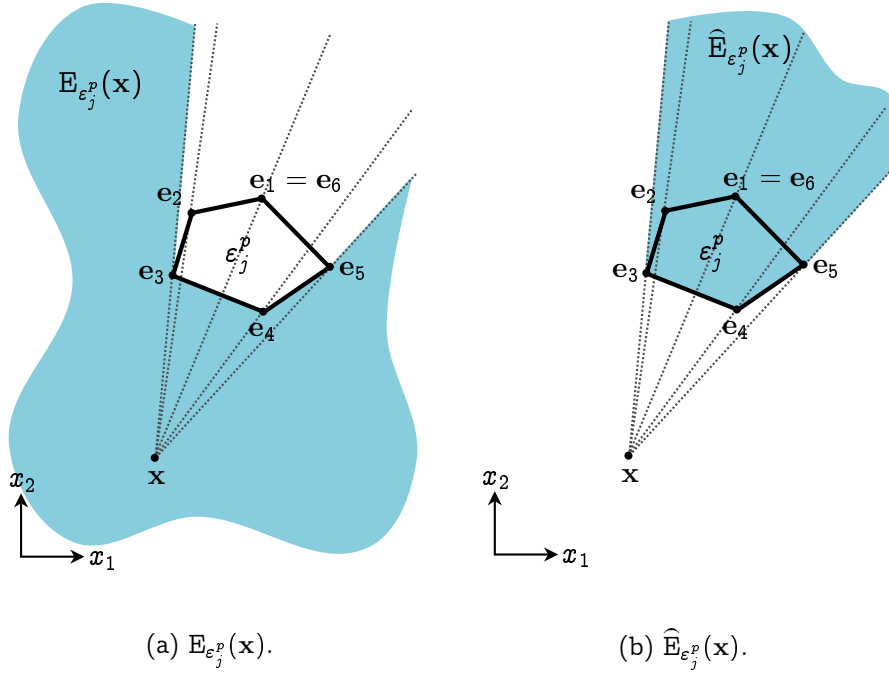


Figure 3.12 – Exemple d'espaces visible et non-visible d'un point vis à vis d'un polygone convexe.

2 .2.1 Considérant un segment comme obstacle

Nous commençons par la caractérisation de l'espace non-visible d'un segment par rapport à un autre segment. Nous remarquons que cet espace peut se caractériser en fonction des espaces non-visibles des extrémités du segment.

Proposition 3.10 Soient un segment $Seg(x_1, x_2)$, avec $x_1 \in \mathbb{R}^2$ et $x_2 \in \mathbb{R}^2$, et un obstacle $\epsilon_j^s = Seg(e_{1_j}, e_{2_j})$, avec $Seg(x_1, x_2) \cap \epsilon_j^s = \emptyset$. On a

$$\widehat{E}_{\epsilon_j^s}(Seg(x_1, x_2)) = \widehat{E}_{\epsilon_j^s}(x_1) \cap \widehat{E}_{\epsilon_j^s}(x_2). \quad (3.44)$$

Démonstration

$$\begin{aligned} \widehat{E}_{\epsilon_j^s}(Seg(x_1, x_2)) &= \{x_i \in \mathbb{R}^2 \mid \forall x \in Seg(x_1, x_2), (x_i \vee x)_{\epsilon_j^s}\} \text{ (Eq. 3.16),} \\ &= \{x_i \in \mathbb{R}^2 \mid \forall x \in Seg(x_1, x_2), [x \cup x_i] \cap [e_{1_j} \cup e_{2_j}] \neq \emptyset \wedge \\ &\quad \zeta_x \det(x_i - e_{1_j} \mid e_{2_j} - e_{1_j}) \leq 0 \wedge \zeta_x \det(x_i - e_{1_j} \mid x - e_{1_j}) \leq 0 \wedge \\ &\quad \zeta_x \det(x_i - e_{2_j} \mid x - e_{2_j}) \geq 0\} \text{ (Eq. 3.41),} \end{aligned}$$

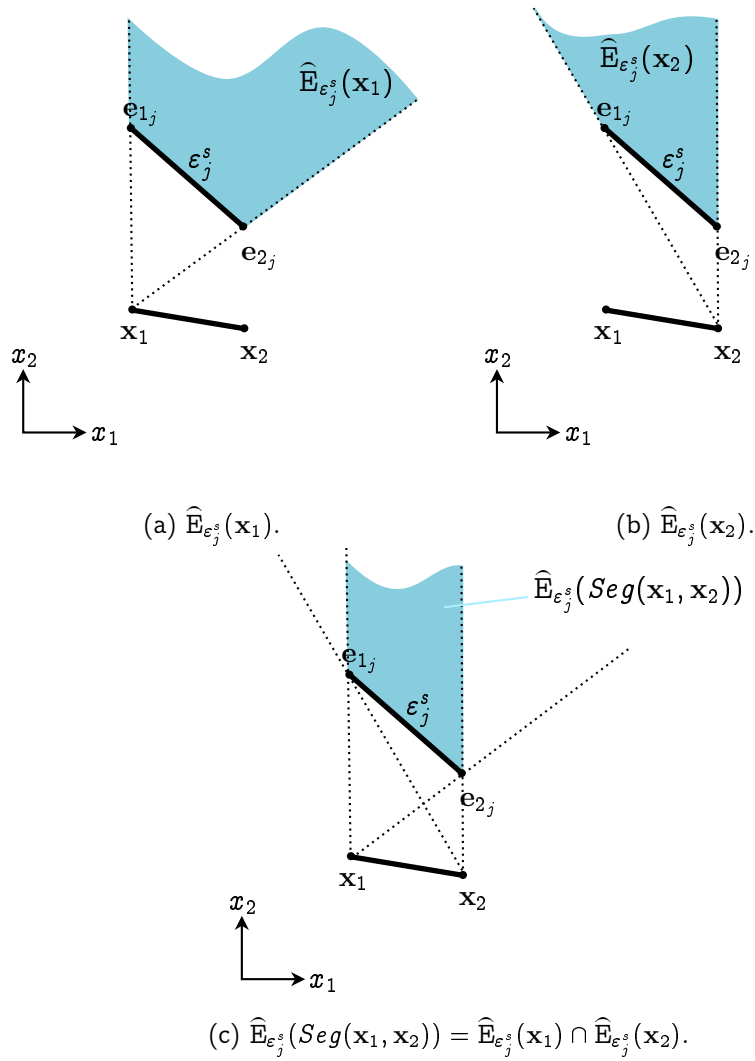


Figure 3.13 – Espace non-visible d'un segment par rapport à un segment.

$$\begin{aligned}
\widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}_1) \cap \widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}_2) = & \{\mathbf{x}_i \in \mathbb{R}^2 \mid \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \\
& [\mathbf{x}_1 \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] \neq \emptyset\} \cap \\
& \{\mathbf{x}_i \in \mathbb{R}^2 \mid \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0 \wedge \\
& [\mathbf{x}_2 \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] \neq \emptyset\}, \text{ (Eq. 3.41)}
\end{aligned}$$

$$\begin{aligned}
\widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}_1) \cap \widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}_2) = & \{\mathbf{x}_i \in \mathbb{R}^2 \mid \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \\
& [\mathbf{x}_1 \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] \neq \emptyset \wedge \\
& \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0 \wedge \\
& [\mathbf{x}_2 \cup \mathbf{x}_i] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] \neq \emptyset\}.
\end{aligned}$$

D'après Proposition C.5 (Annexe C)

$$\begin{aligned}
\forall \mathbf{x} \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
\zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x} - \mathbf{e}_{2_j}) \geq 0 \\
\Leftrightarrow \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
\zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} \mid \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0 \wedge \\
\zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} \mid \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0.
\end{aligned}$$

Nous avons donc

$$\widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{x}_1, \mathbf{x}_2)) = \widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}_1) \cap \widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}_2).$$

La Figure 3.13 illustre la Proposition 3.10.

L'espace visible d'un segment par rapport à un autre segment est plus compliqué à mettre en place. En effet, cette fois il n'est pas possible de le caractériser par les espaces visibles deux extrémités du segments. Il est même nécessaire de considérer deux cas : le cas où $\zeta_{x_1} = \zeta_{x_2}$ et le cas où $\zeta_{x_1} = -\zeta_{x_2}$.

Proposition 3.11 Soient un segment $\text{Seg}(\mathbf{x}_1, \mathbf{x}_2)$, avec $\mathbf{x}_1 \in \mathbb{R}^2$ et $\mathbf{x}_2 \in \mathbb{R}^2$, et

un obstacle $\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$, avec $\text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j^s = \emptyset$. On a

$$\begin{aligned} E_{\varepsilon_j^s}(\text{Seg}(\mathbf{x}_1, \mathbf{x}_2)) = \{ & \mathbf{x}_i \in \mathbb{R}^2 \mid \\ & (\zeta_{\mathbf{x}_1} = \zeta_{\mathbf{x}_2}) \wedge \left(\zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \right. \\ & \quad \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) > 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) > 0 \vee \\ & \quad \left. \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) < 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) < 0 \right) \vee \\ & (\zeta_{\mathbf{x}_1} = -\zeta_{\mathbf{x}_2}) \wedge \left(\right. \\ & \quad \left(\zeta_{\mathbf{e}_{1_j}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) > 0 \vee \zeta_{\mathbf{e}_{1_j}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) < 0 \right) \wedge \\ & \quad \left. \left(\zeta_{\mathbf{e}_{2_j}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) > 0 \vee \zeta_{\mathbf{e}_{2_j}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) < 0 \right) \right) \vee \\ & \left. \left([\mathbf{x}_i \cup \mathbf{x}_1 \cup \mathbf{x}_2] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \right) \right\}. \end{aligned} \quad (3.45)$$

avec

$$\begin{aligned} \zeta_{\mathbf{x}_1} &= \begin{cases} 1 & \text{si } \det(\mathbf{x}_1 - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0, \\ -1 & \text{sinon.} \end{cases} \\ \zeta_{\mathbf{x}_2} &= \begin{cases} 1 & \text{si } \det(\mathbf{x}_2 - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0, \\ -1 & \text{sinon.} \end{cases} \\ \zeta_{\mathbf{e}_{1_j}} &= \begin{cases} 1 & \text{si } \det(\mathbf{e}_{1_j} - \mathbf{x}_1 | \mathbf{x}_2 - \mathbf{x}_1) > 0, \\ -1 & \text{sinon.} \end{cases} \\ \zeta_{\mathbf{e}_{2_j}} &= \begin{cases} 1 & \text{si } \det(\mathbf{e}_{2_j} - \mathbf{x}_1 | \mathbf{x}_2 - \mathbf{x}_1) > 0, \\ -1 & \text{sinon.} \end{cases} \end{aligned}$$

Démonstration L'explication de cette proposition est présente Annexe C, Section 3.3.

Les Figures 3.14 et 3.15 illustrent la Proposition 3.11.

Nous pouvons maintenant utiliser ces propositions pour nous intéresser aux obstacles de type polygone.

2.2.2 Considérant un polygone comme obstacle

Comme il a déjà été précisé, un polygone peut être considéré comme un ensemble de segments. Il est donc possible de caractériser les espaces de visibilité d'un segment par rapport un polygone, en tenant compte des espaces de visibilité du segment par rapport à tous les segments qui délimitent le polygone.

Ce qui nous donne, pour l'espace visible du segment par rapport au polygone :

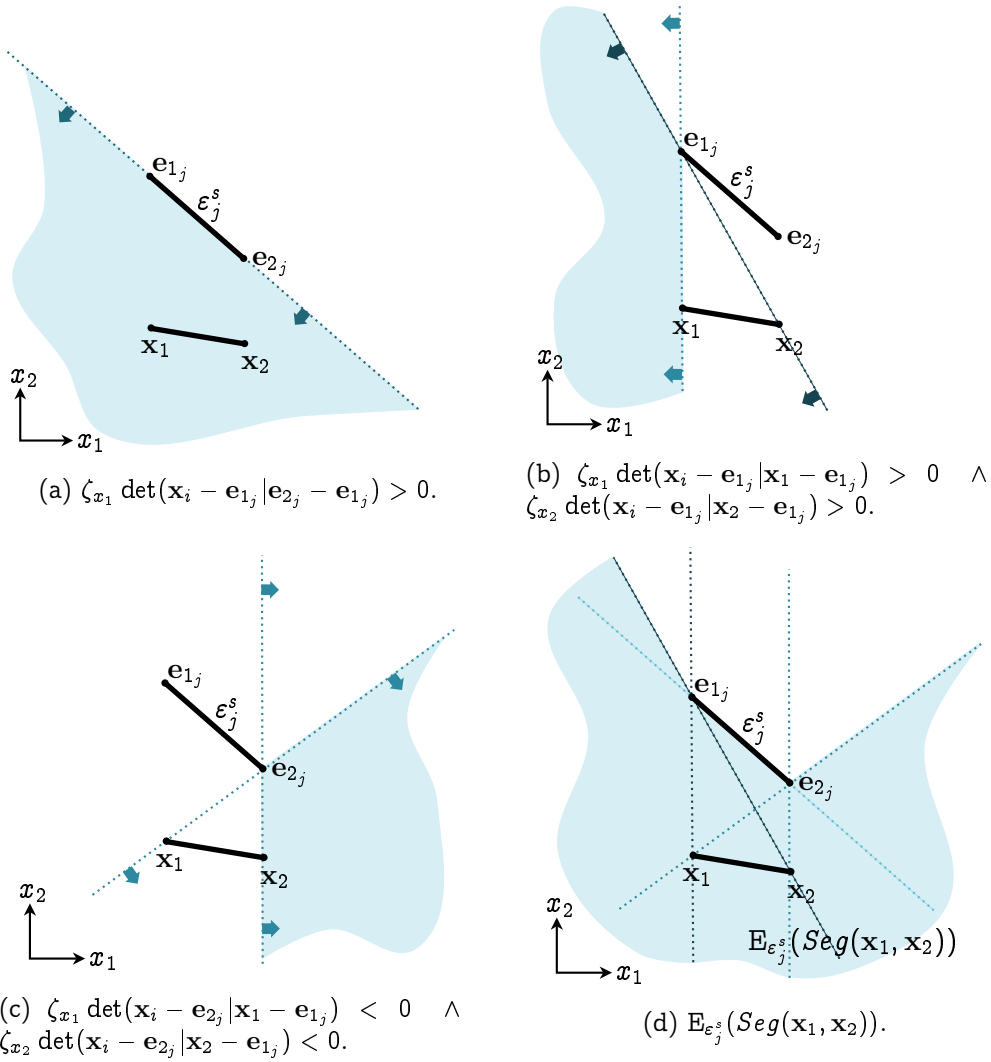


Figure 3.14 – Espace visible d’un segment par rapport à un segment dans le cas où $\zeta_{x_1} = \zeta_{x_2}$.

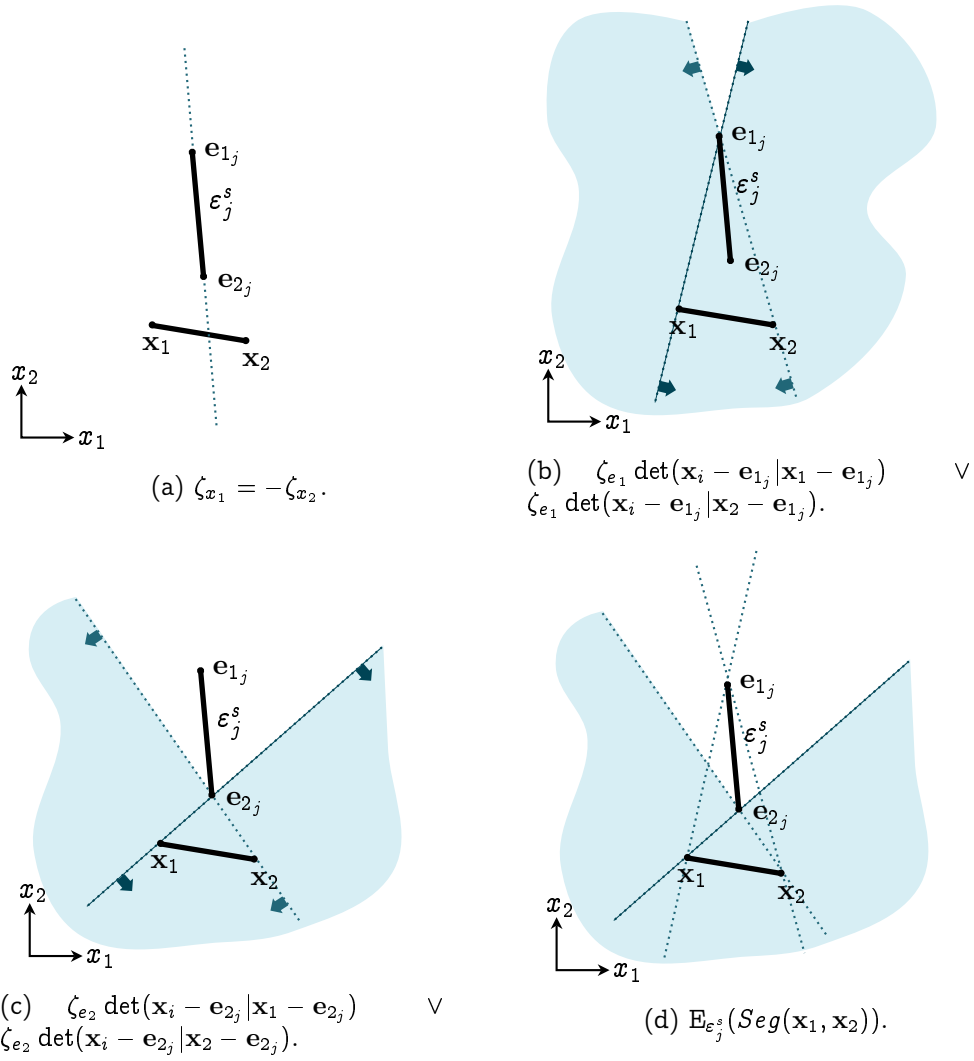


Figure 3.15 – Espace visible d’un segment par rapport à un segment dans le cas où $\zeta_{x_1} = -\zeta_{x_2}$.

Proposition 3.12 *Soient un segment $Seg(\mathbf{x}_1, \mathbf{x}_2)$, $\mathbf{x}_1 \in \mathbb{R}^2$ et $\mathbf{x}_2 \in \mathbb{R}^2$, et un obstacle ε_j^p composé de n_{P_j} sommets, avec $Seg(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j^p = \emptyset$. On a*

$$\mathbb{E}_{\varepsilon_j^p}(Seg(\mathbf{x}_1, \mathbf{x}_2)) = \bigcap_{k=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_k^s}(Seg(\mathbf{x}_1, \mathbf{x}_2)). \quad (3.46)$$

Démonstration

$$\mathbb{E}_{\varepsilon_j^p}(Seg(\mathbf{x}_1, \mathbf{x}_2)) = \bigcap_{k=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_k^s}(Seg(\mathbf{x}_1, \mathbf{x}_2)) \text{ (Eq. 3.32 et 3.35).}$$

Et pour l'espace non-visible du segment par rapport au polygone :

Proposition 3.13 *Soient un segment $Seg(\mathbf{x}_1, \mathbf{x}_2)$, $\mathbf{x}_1 \in \mathbb{R}^2$ et $\mathbf{x}_2 \in \mathbb{R}^2$, et un polygone convexe ε_j^p composé de n_{P_j} sommets, avec $Seg(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j^p = \emptyset$. On a*

$$\widehat{\mathbb{E}}_{\varepsilon_j^p}(Seg(\mathbf{x}_1, \mathbf{x}_2)) = \left(\bigcup_{k=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_k^s}(\mathbf{x}_1) \right) \cap \left(\bigcup_{k=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_k^s}(\mathbf{x}_2) \right). \quad (3.47)$$

Démonstration

$$\begin{aligned} \widehat{\mathbb{E}}_{\varepsilon_j^p}(Seg(\mathbf{x}_1, \mathbf{x}_2)) &= \bigcup_{k=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_k^s}(Seg(\mathbf{x}_1, \mathbf{x}_2)) \text{ (Eq. 3.37),} \\ &= \bigcup_{k=1}^{n_{P_j}} (\widehat{\mathbb{E}}_{\varepsilon_k^s}(\mathbf{x}_1) \cap \widehat{\mathbb{E}}_{\varepsilon_k^s}(\mathbf{x}_2)) \text{ (Eq. 3.44),} \\ \widehat{\mathbb{E}}_{\varepsilon_j^p}(Seg(\mathbf{x}_1, \mathbf{x}_2)) &= \left(\bigcup_{k=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_k^s}(\mathbf{x}_1) \right) \cap \left(\bigcup_{k=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_k^s}(\mathbf{x}_2) \right). \end{aligned}$$

La Figure 3.16 illustre les Propositions 3.12 et 3.13.

Pour finir ces cas particuliers, il nous reste à étudier la visibilité d'un polygone (les boîtes seront associées à des polygone).

2 .3 Visibilité d'un polygone

Un polygone peut être associé à un ensemble de segments, caractérisant ses frontières. Nous pouvons donc utiliser les propositions de la Section précédentes afin de caractériser les espaces de visibilité d'un polygone. Dans un premier temps nous caractérisons les espaces de visibilité d'un polygone par rapport à un obstacle de type segment.

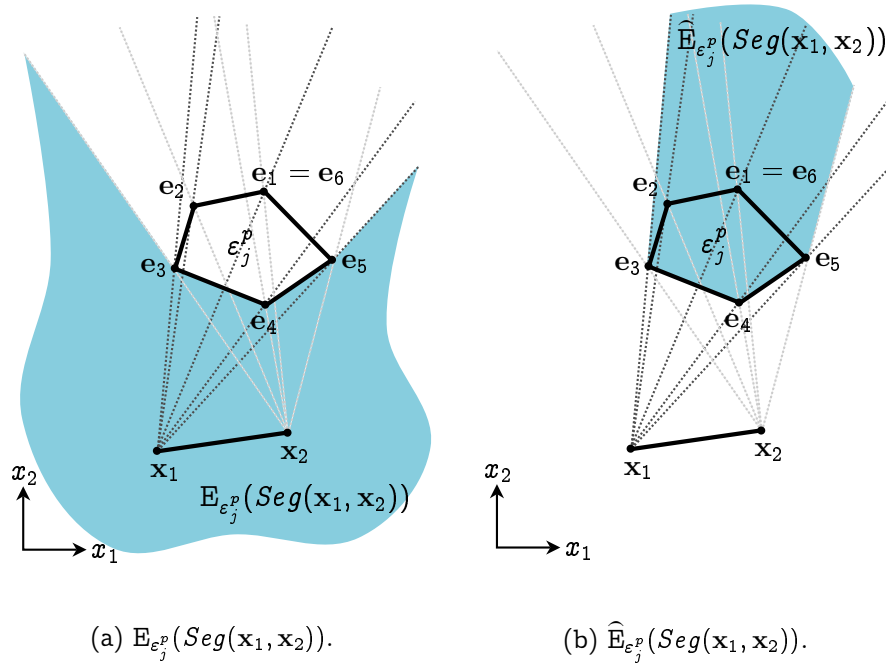


Figure 3.16 – Exemple d’espaces visible et non-visible d’un segment par rapport à un polygone convexe.

2 .3.1 Considérant un segment comme obstacle

Comme précisé précédemment, les espaces de visibilité d’un polygone peuvent être caractérisés par les espaces de visibilité des segments délimitant ce polygone.

Pour l’espace visible d’un polygone par rapport à un segment (Figure 3.17) nous avons :

Proposition 3.14 Soient un polygone convexe P composé de n_P sommets, et un obstacle $\epsilon_j^s = Seg(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$ avec $P \cap \epsilon_j^s = \emptyset$. On a

$$\widehat{E}_{\epsilon_j^s}(P) = \bigcap_{k=1}^{n_P} \widehat{E}_{\epsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})). \quad (3.48)$$

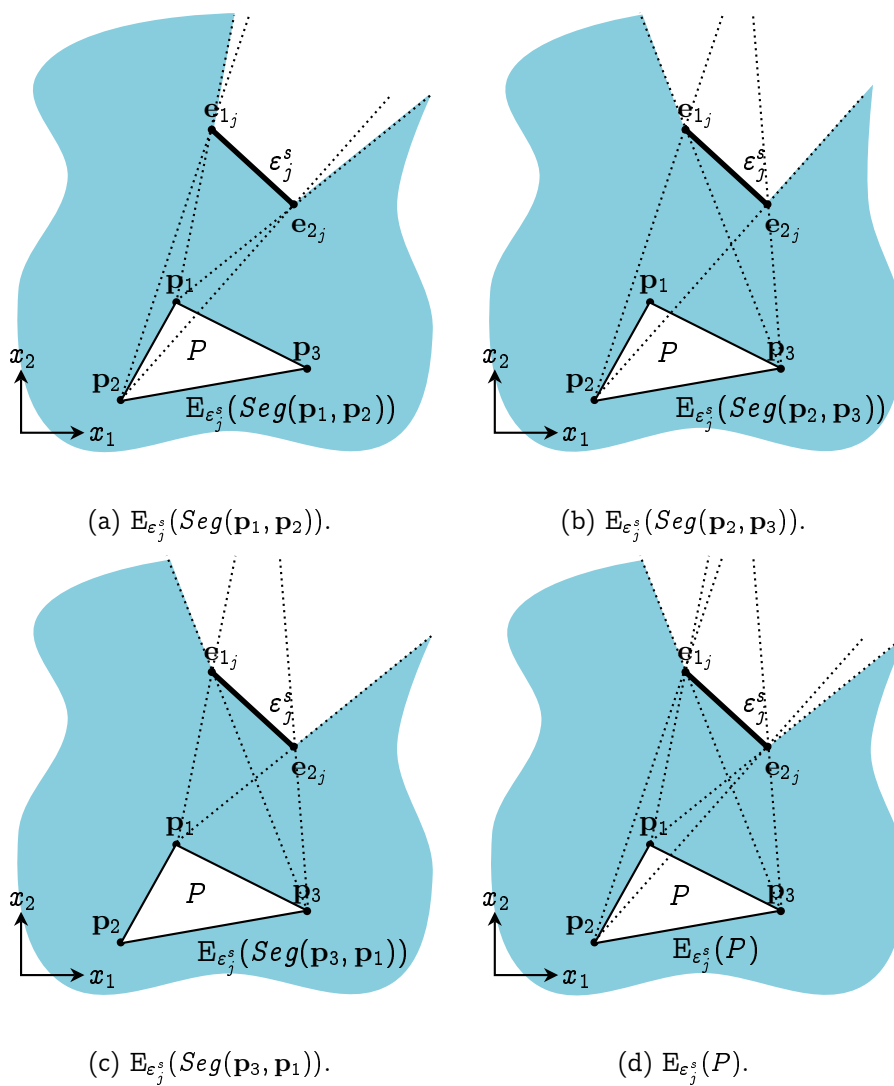


Figure 3.17 – Espace visible d'un polygone P par rapport à un segment.

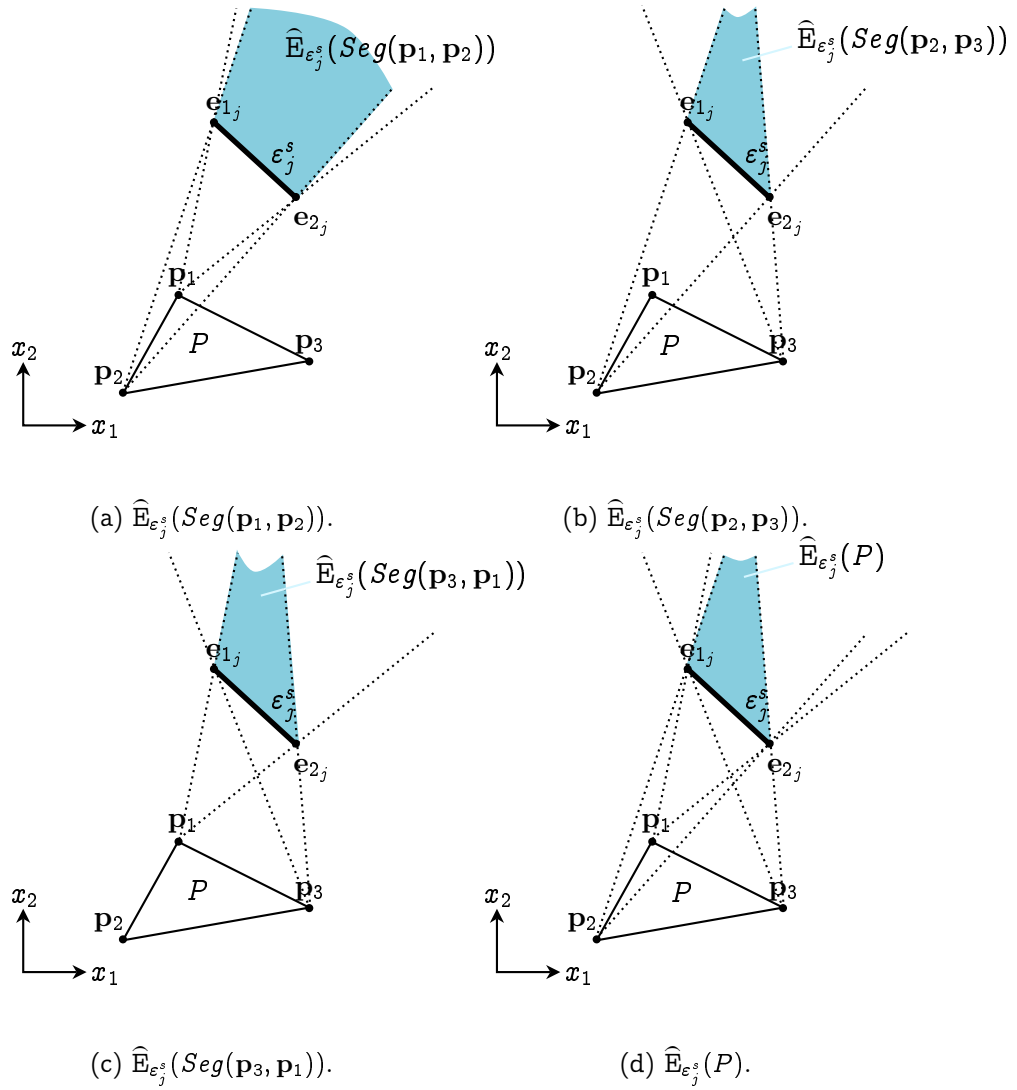


Figure 3.18 – Espace non-visible d'un polygone P par rapport à un segment.

Démonstration

$$\begin{aligned}
\mathbf{x}_i \in \widehat{\mathbb{E}}_{\varepsilon_j^s}(P) &\Leftrightarrow \forall \mathbf{x}_p \in P, (\mathbf{x}_i \overline{\mathbf{V}} \mathbf{x}_p)_{\varepsilon_j^s} \text{ (Eq. 3.17),} \\
&\Leftrightarrow \forall \mathbf{x}_p \in P, \mathbf{x}_i \in \widehat{\mathbb{E}}_{\varepsilon_j^s}(\mathbf{x}_p) \text{ (Eq. 3.13),} \\
&\Rightarrow \forall \text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1}) \in P, \mathbf{x}_i \in \widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})), \\
\mathbf{x}_i \in \widehat{\mathbb{E}}_{\varepsilon_j^s}(P) &\Rightarrow \mathbf{x}_i \in \bigcap_{k=1}^{n_P} \widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})).
\end{aligned}$$

On a donc

$$\widehat{\mathbb{E}}_{\varepsilon_j^s}(P) \supseteq \bigcap_{k=1}^{n_P} \widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})). \quad (3.49)$$

$$\begin{aligned}
\mathbf{x}_i \in \bigcap_{k=1}^{n_P} \widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})) &\Leftrightarrow \forall \text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1}) \in P, \forall \mathbf{x}_s \in \text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1}), \\
&(\mathbf{x}_i \overline{\mathbf{V}} \mathbf{x}_s)_{\varepsilon_j^s} \text{ (Eq. 3.17),} \\
&\Rightarrow \forall \mathbf{x}_p \in P, (\mathbf{x}_i \overline{\mathbf{V}} \mathbf{x}_p)_{\varepsilon_j^s}, \\
\mathbf{x}_i \in \bigcap_{k=1}^{n_P} \widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})) &\Rightarrow \mathbf{x}_i \in \widehat{\mathbb{E}}_{\varepsilon_j^s}(P) \text{ (Eq. 3.13).}
\end{aligned}$$

On a alors

$$\bigcap_{k=1}^{n_P} \widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})) \supseteq \widehat{\mathbb{E}}_{\varepsilon_j^s}(P). \quad (3.50)$$

On peut en conclure

$$\text{(Eq. 3.49 et 3.50)} \Rightarrow \widehat{\mathbb{E}}_{\varepsilon_j^s}(P) = \bigcap_{k=1}^{n_P} \widehat{\mathbb{E}}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})).$$

Et pour l'espace non-visible d'un polygone par rapport à un segment (Figure 3.18), nous avons :

Proposition 3.15 *Soient un polygone convexe P composé de n_P sommets, et un obstacle $\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$ avec $P \cap \varepsilon_j^s = \emptyset$. On a alors*

$$\mathbb{E}_{\varepsilon_j^s}(P) = \bigcap_{k=1}^{n_P} \mathbb{E}_{\varepsilon_j^s}(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})). \quad (3.51)$$

Démonstration

$$\begin{aligned}
\mathbf{x}_i \in E_{\varepsilon_j^s}(P) &\Leftrightarrow \forall \mathbf{x}_p \in P, (\mathbf{x}_i \vee \mathbf{x}_p)_{\varepsilon_j^s} \text{ (Eq. 3.16),} \\
&\Leftrightarrow \forall \mathbf{x}_p \in P, \mathbf{x}_i \in E_{\varepsilon_j^s}(\mathbf{x}_p) \text{ (Eq. 3.10),} \\
&\Rightarrow \forall Seg(\mathbf{p}_k, \mathbf{p}_{k+1}) \in P, \mathbf{x}_i \in E_{\varepsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})), \\
\mathbf{x}_i \in E_{\varepsilon_j^s}(P) &\Rightarrow \mathbf{x}_i \in \bigcap_{k=1}^{n_P} E_{\varepsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})).
\end{aligned}$$

On a donc

$$E_{\varepsilon_j^s}(P) \supseteq \bigcap_{k=1}^{n_P} E_{\varepsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})). \quad (3.52)$$

$$\begin{aligned}
\mathbf{x}_i \in \bigcap_{k=1}^{n_P} E_{\varepsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})) &\Leftrightarrow \forall Seg(\mathbf{p}_k, \mathbf{p}_{k+1}) \in P, \forall \mathbf{x}_s \in Seg(\mathbf{p}_k, \mathbf{p}_{k+1}), \\
&(\mathbf{x}_i \vee \mathbf{x}_s)_{\varepsilon_j^s} \text{ (Eq. 3.16),} \\
&\Rightarrow \forall \mathbf{x}_p \in P, (\mathbf{x}_i \vee \mathbf{x}_p)_{\varepsilon_j^s}, \\
\mathbf{x}_i \in \bigcap_{k=1}^{n_P} E_{\varepsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})) &\Rightarrow \mathbf{x}_i \in E_{\varepsilon_j^s}(P) \text{ (Eq. 3.10).}
\end{aligned}$$

On a donc

$$\bigcap_{k=1}^{n_P} E_{\varepsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})) \supseteq E_{\varepsilon_j^s}(P). \quad (3.53)$$

On peut en conclure

$$\text{(Eq. 3.52 et 3.53)} \Rightarrow E_{\varepsilon_j^s}(P) = \bigcap_{k=1}^{n_P} E_{\varepsilon_j^s}(Seg(\mathbf{p}_k, \mathbf{p}_{k+1})).$$

La dernière étape consiste à calculer les espaces de visibilité d'un polygone en considérant un autre polygone comme obstacle.

2.3.2 Considérant un polygone comme obstacle

Comme pour le cas précédent, les espaces de visibilité d'un polygone en considérant un polygone comme obstacle, peuvent être caractérisés par les espaces de visibilité des segments qui délimitent le polygone. Nous avons alors tous les éléments pour définir ces espaces.

Pour l'espace visible nous avons :

Proposition 3.16 *Soient un polygone convexe P avec n_P sommets et un obstacle ε_j^p composé de n_{P_j} sommets avec $P \cap \varepsilon_j^p = \emptyset$. On a*

$$\mathbb{E}_{\varepsilon_j^p}(P) = \bigcap_{k=1}^{n_P} \left(\bigcap_{k'=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_{k'}}^s(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})) \right). \quad (3.54)$$

Démonstration

$$\begin{aligned} \mathbb{E}_{\varepsilon_j^p}(P) &= \bigcap_{k'=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_{k'}}^s(P) \text{ (Eq. 3.32 et 3.35),} \\ \mathbb{E}_{\varepsilon_j^p}(P) &= \bigcap_{k'=1}^{n_{P_j}} \left(\bigcap_{k=1}^{n_P} \mathbb{E}_{\varepsilon_{k'}}^s(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})) \right) \text{ (Eq. 3.51),} \\ \mathbb{E}_{\varepsilon_j^p}(P) &= \bigcap_{k=1}^{n_P} \left(\bigcap_{k'=1}^{n_{P_j}} \mathbb{E}_{\varepsilon_{k'}}^s(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})) \right). \end{aligned}$$

Et pour l'espace non-visible nous pouvons définir :

Proposition 3.17 *Soient un polygone convexe P avec n_P sommets et un obstacle ε_j^p composé de n_{P_j} sommets avec $P \cap \varepsilon_j^p = \emptyset$. On a*

$$\widehat{\mathbb{E}}_{\varepsilon_j^p}(P) = \bigcap_{k=1}^{n_P} \left(\bigcup_{k'=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_{k'}}^s(\mathbf{p}_k) \right). \quad (3.55)$$

Démonstration

$$\begin{aligned} \widehat{\mathbb{E}}_{\varepsilon_j^p}(P) &= \bigcup_{k'=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_{k'}}^s(P) \text{ (Eq. 3.37),} \\ &= \bigcup_{k'=1}^{n_{P_j}} \left(\bigcap_k \widehat{\mathbb{E}}_{\varepsilon_{k'}}^s(\text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1})) \right) \text{ (Eq. 3.48),} \\ &= \bigcup_{k'=1}^{n_{P_j}} \left(\bigcap_{k=1}^{n_P} (\widehat{\mathbb{E}}_{\varepsilon_{k'}}^s(\mathbf{p}_k) \cap \widehat{\mathbb{E}}_{\varepsilon_{k'}}^s(\mathbf{p}_{k+1})) \right) \text{ (Eq. 3.44),} \\ &= \bigcup_{k'=1}^{n_{P_j}} \left(\bigcap_{k=1}^{n_P} \widehat{\mathbb{E}}_{\varepsilon_k}^s(\mathbf{p}_k) \right), \\ \widehat{\mathbb{E}}_{\varepsilon_j^p}(P) &= \bigcap_{k=1}^{n_P} \left(\bigcup_{k'=1}^{n_{P_j}} \widehat{\mathbb{E}}_{\varepsilon_{k'}}^s(\mathbf{p}_k) \right). \end{aligned}$$

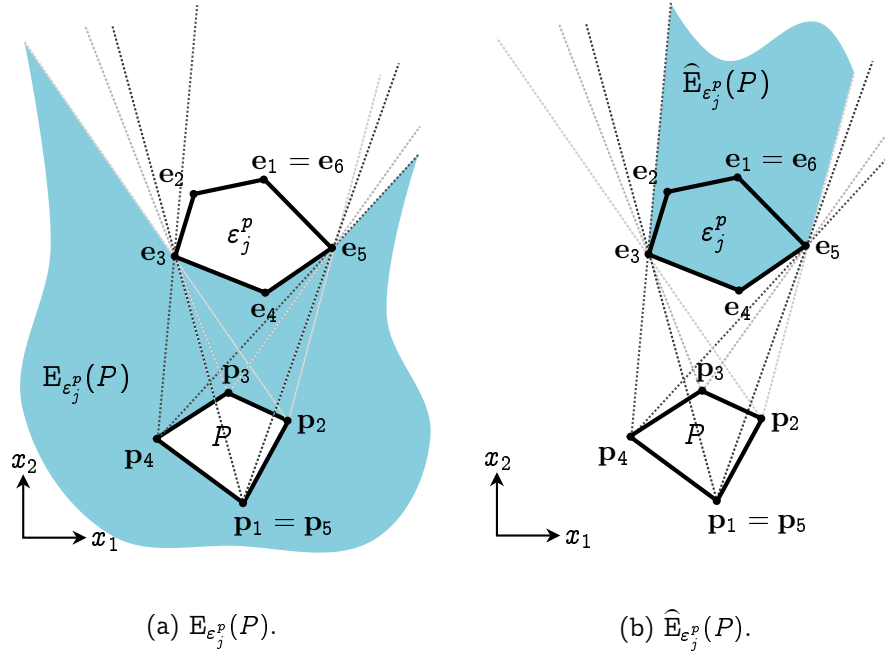


Figure 3.19 – Exemple d’espaces visible et non-visible d’un polygone convexe vis à vis d’un polygone convexe.

La Figure 3.19 illustre les Propositions 3.16 et 3.17.

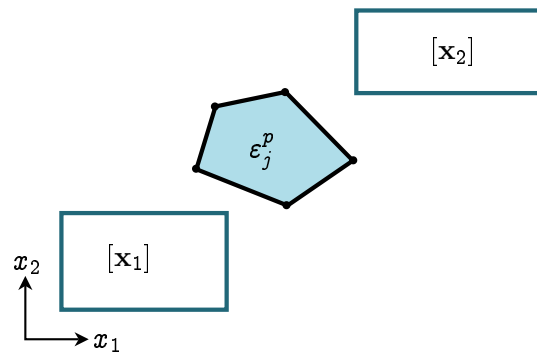
Nous avons maintenant tous les outils nécessaires pour le développement de nos contracteurs, qui vont nous permettre de réduire les boîtes selon la contrainte d’intersection/non-intersection (visibilité/non-visibilité) comme présenté en introduction de ce chapitre.

3 Les contracteurs

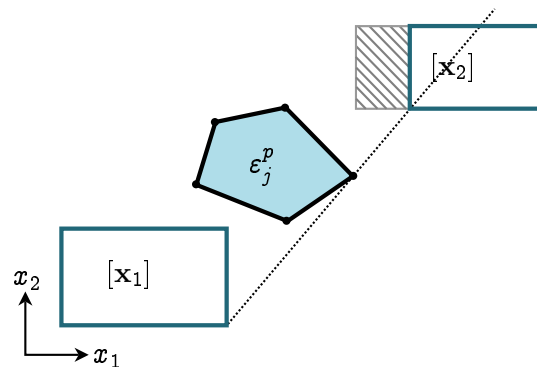
Soient un point x_1 caractérisé par une boîte $x_1 \in [x_1]$, un point x_2 caractérisé par une boîte $x_2 \in [x_2]$ et un obstacle de type polygone ε_j^p . Nous voulons pouvoir contracter les domaines $[x_1]$ et $[x_2]$ selon une contrainte de visibilité $((x_1 V x_2)_{\varepsilon_j^p})$ ou de non-visibilité $((x_1 \bar{V} x_2)_{\varepsilon_j^p})$ (Figure 3.20).

Pour ce faire nous créons, en nous inspirant des propositions précédentes, des contracteurs (des fonctions) qui nous permettent de contracter les boîtes en fonction des contraintes de visibilité. Ces contracteurs se basent sur la proposition suivante :

Proposition 3.18 Soient un point $a \in \mathbb{A}$ appartenant à un ensemble connexe



(a) Soient un point \mathbf{x}_1 caractérisé par une boîte $\mathbf{x}_1 \in [x_1]$, un point \mathbf{x}_2 caractérisé par une boîte $\mathbf{x}_2 \in [x_2]$ et un obstacle de type polygone ε_j^p .



(b) Sachant que $(\mathbf{x}_1 V \mathbf{x}_2)_{\varepsilon_j^p}$ nous voulons contracter les domaines $[x_1]$ et $[x_2]$ pour les rendre consistant avec la contrainte de visibilité : la partie hachurée fait partie de l'espace non-visible de $[x_1]$ par rapport à ε_j^p , elle n'est donc pas consistante avec la contrainte de visibilité et peut être *retirée* du domaine $[x_2]$.

Figure 3.20 – Exemple de contraction selon la contrainte $(\mathbf{x}_1 V \mathbf{x}_2)_{\varepsilon_j^p}$.

$\mathbb{A} \subset \mathbb{R}^n$, une variable $\mathbf{x} \in [\mathbf{x}]$ et un environnement \mathcal{E} de \mathbb{R}^n

$$(\mathbf{aVx})_{\mathcal{E}} \Rightarrow \mathbf{x} \in [\mathbf{x}] \cap \left(\widehat{\mathbf{E}}_{\mathcal{E}}(\mathbb{A})\right)^c \quad (3.56)$$

$$(\mathbf{a}\overline{\mathbf{V}}\mathbf{x})_{\mathcal{E}} \Rightarrow \mathbf{x} \in [\mathbf{x}] \cap \left(\mathbf{E}_{\mathcal{E}}(\mathbb{A})\right)^c \quad (3.57)$$

Démonstration Cette proposition se déduit de la Proposition 3.1.

L'idée pour la construction des contracteurs va alors être de considérer une des deux boîtes comme un polygone et d'utiliser la caractérisation des espaces de visibilité pour contracter la deuxième boîte.

Comme pour la caractérisation des espaces de visibilité, nous allons définir les contracteurs progressivement, en commençant par les contracteurs de visibilité d'un point, puis d'un segment pour finir aux contracteurs de visibilité d'un polygone.

Le lecteur notera que le contracteur C_{det} , utilisé pour nos contracteurs de visibilité, est présenté Annexe C, Section 1.2.1.

3.1 Contracteurs de visibilité d'un point

Pour la suite nous n'avons pas besoin de définir les contracteurs de visibilité d'un point pour un obstacle de type polygone. Nous nous contentons donc de définir les contracteurs de visibilité d'un point pour un obstacle de type segment.

3.1.1 Contracteur de visibilité d'un point avec un obstacle segment

Ce premier contracteur, nommé $C_V([\mathbf{x}], \mathbf{a}, \varepsilon_j^s)$, est associé à la contrainte

$$(\mathbf{xV}\mathbf{a})_{\varepsilon_j^s}, \quad (3.58)$$

avec $\mathbf{a} \in \mathbb{R}^2$, $\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1j}, \mathbf{e}_{2j})$, $\mathbf{e}_{1j} \in \mathbb{R}^2$, $\mathbf{e}_{2j} \in \mathbb{R}^2$ et $\mathbf{x} \in [\mathbf{x}]$. La Figure 3.21 illustre son principe.

Ce contracteur est présenté Algorithme 13. Il est basé sur le complément de l'Équation 3.41.

3.1.2 Contracteur de non-visibilité d'un point avec un obstacle segment

Le dual du contracteur de visibilité, le contracteur de non-visibilité, nommé $C_{\overline{V}}([\mathbf{x}], \mathbf{a}, \varepsilon_j^s)$, est associé à la contrainte

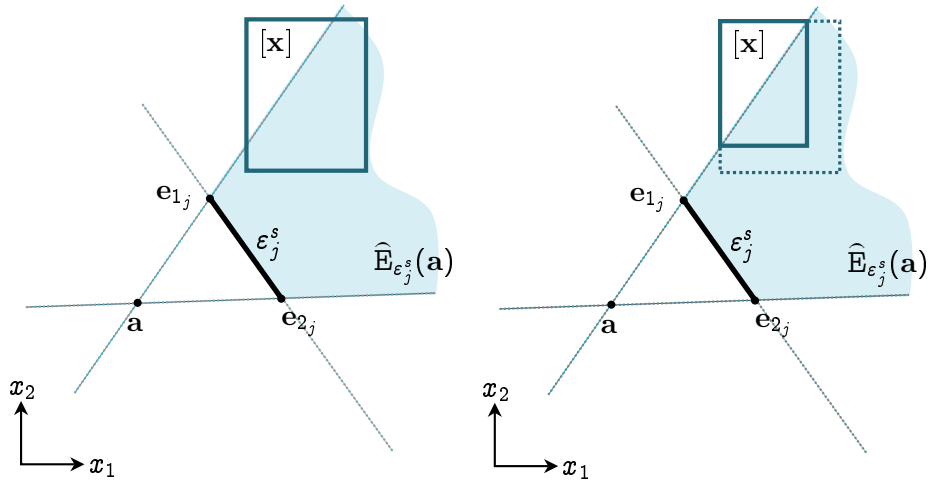
$$(\mathbf{x}\overline{\mathbf{V}}\mathbf{a})_{\varepsilon_j^s}, \quad (3.59)$$

Algorithme 13: Contracteur $C_V([\mathbf{x}], \mathbf{a}, \varepsilon_j^s)$

Données : $[\mathbf{x}], \mathbf{a}, \varepsilon_j^s = \text{Seg}(\mathbf{e}_{1j}, \mathbf{e}_{2j})$

- 1 si $\det(\mathbf{a} - \mathbf{e}_{1j} | \mathbf{e}_{2j} - \mathbf{e}_{1j}) > 0$ alors
- 2 | $\zeta_a = 1;$
- 3 sinon
- 4 | $\zeta_a = -1;$
- 5 fin
- 6 $[\mathbf{i}_1] = C_{det}([\mathbf{x}], \mathbf{e}_{1j}, \mathbf{e}_{2j}, \zeta_a);$
- 7 $[\mathbf{i}_2] = C_{det}([\mathbf{x}], \mathbf{e}_{1j}, \mathbf{a}, \zeta_a);$
- 8 $[\mathbf{i}_3] = C_{det}([\mathbf{x}], \mathbf{e}_{2j}, \mathbf{a}, -\zeta_a);$
- 9 $[\mathbf{i}_4] = C_{\cap=\emptyset}([\mathbf{x}], \mathbf{a}, \mathbf{e}_{1j}, \mathbf{e}_{2j});$

Résultat : $[\mathbf{x}]^* = [\mathbf{i}_1] \cup [\mathbf{i}_2] \cup [\mathbf{i}_3] \cup [\mathbf{i}_4]$

(a) $(\mathbf{x} \vee \mathbf{a})_{\varepsilon_j^s}$, avec $\mathbf{x} \in [\mathbf{x}]$.(b) Contraction de $[\mathbf{x}]$.Figure 3.21 – Illustration du contracteur $C_V([\mathbf{x}], \mathbf{a}, \varepsilon_j^s)$.

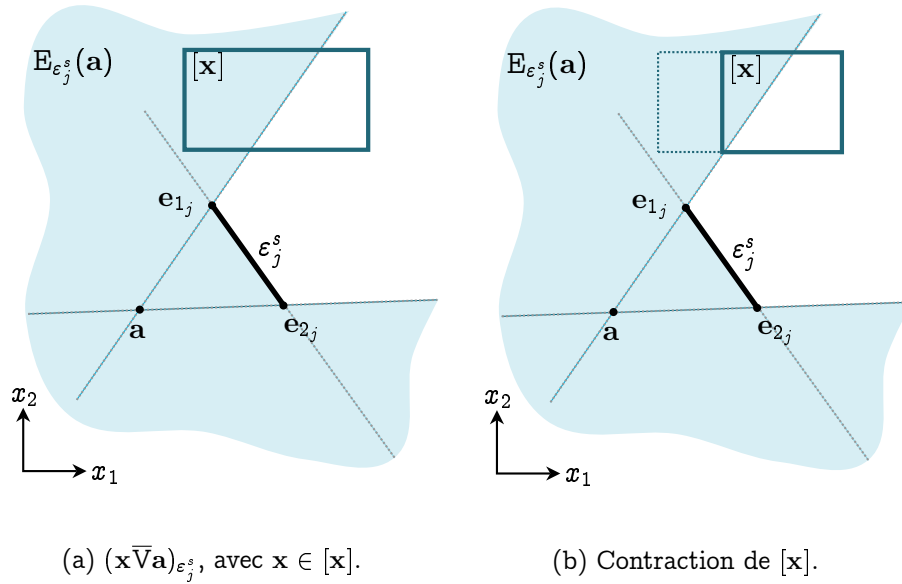


Figure 3.22 – Illustration du contracteur $C_{\bar{V}}([x], a, \varepsilon_j^s)$.

avec $a \in \mathbb{R}^2$, $\varepsilon_j^s = \text{Seg}(e_{1j}, e_{2j})$, $e_{1j} \in \mathbb{R}^2$, $e_{2j} \in \mathbb{R}^2$ et $x \in [x]$.

Ce contracteur est présenté Algorithme 14. Il est basé sur le complément de l'Équation 3.38. La Figure 3.22 illustre son principe.

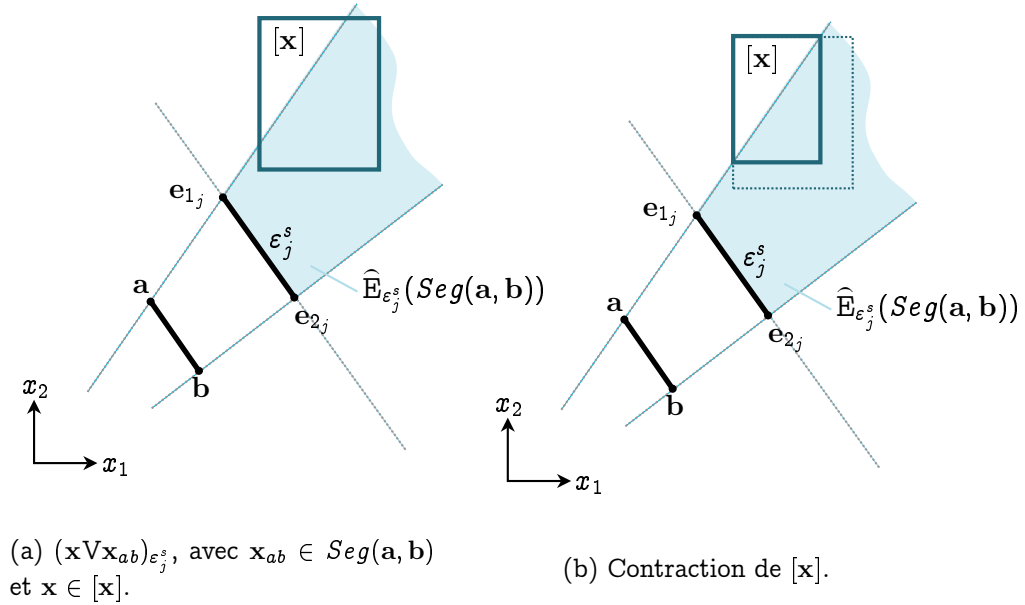
Algorithme 14: Contracteur $C_{\bar{V}}([x], a, \varepsilon_j^s)$

Données : $[x], a, \varepsilon_j^s = \text{Seg}(e_{1j}, e_{2j})$

- 1 **si** $\det(a - e_{1j} | e_{2j} - e_{1j}) > 0$ **alors**
- 2 | $\zeta_a = 1$;
- 3 **sinon**
- 4 | $\zeta_a = -1$;
- 5 **fin**
- 6 $[i_1] = C_{\det}([x], e_{1j}, e_{2j}, -\zeta_a)$;
- 7 $[i_2] = C_{\det}([x], e_{1j}, a, -\zeta_a)$;
- 8 $[i_3] = C_{\det}([x], e_{2j}, a, \zeta_a)$;
- 9 $[i_4] = C_{\cap \neq \emptyset}([x], a, e_{1j}, e_2)$;

Résultat : $[x]^* = [i_1] \cap [i_2] \cap [i_3] \cap [i_4]$

Une fois ces deux contracteurs définis, nous pouvons nous intéresser aux contracteurs de visibilité d'un segment.

Figure 3.23 – Illustration du contracteur $C_V([x], a, b, \epsilon_j^s)$.

3.2 Contracteurs de visibilité d'un segment

Comme pour les contracteurs de visibilité d'un point, nous n'avons pas besoin de définir les contracteurs de visibilité d'un segment pour un obstacle de type polygone. Nous nous contentons donc de définir les contracteurs de visibilité d'un segment pour un obstacle de type segment.

3.2.1 Contracteur de visibilité d'un segment avec un obstacle segment

Ce contracteur, nommé $C_V([x], a, b, \epsilon_j^s)$, est associé à la contrainte

$$(xVx_{ab})_{\epsilon_j^s}, \quad (3.60)$$

avec $x_{ab} \in Seg(a, b)$, $a \in \mathbb{R}^2$, $b \in \mathbb{R}^2$, $\epsilon_j^s = Seg(e_{1j}, e_{2j})$, $e_{1j} \in \mathbb{R}^2$, $e_{2j} \in \mathbb{R}^2$ et $x \in [x]$.

Il est présenté Algorithme 15. Il est basé sur le complément de l'Équation 3.44. La Figure 3.23 illustre son principe.

Algorithme 15: Contracteur $C_V([\mathbf{x}], \mathbf{a}, \mathbf{b}, \varepsilon_j^s)$

Données : $[\mathbf{x}], \mathbf{a}, \mathbf{b}, \varepsilon_j^s = \text{Seg}(\mathbf{e}_{1j}, \mathbf{e}_{2j})$

1 $[\mathbf{i}_1] = C_V([\mathbf{x}], \mathbf{a}, \varepsilon_j^s)$;

2 $[\mathbf{i}_2] = C_V([\mathbf{x}], \mathbf{b}, \varepsilon_j^s)$;

Résultat : $[\mathbf{x}]^* = [\mathbf{i}_1] \cup [\mathbf{i}_2]$

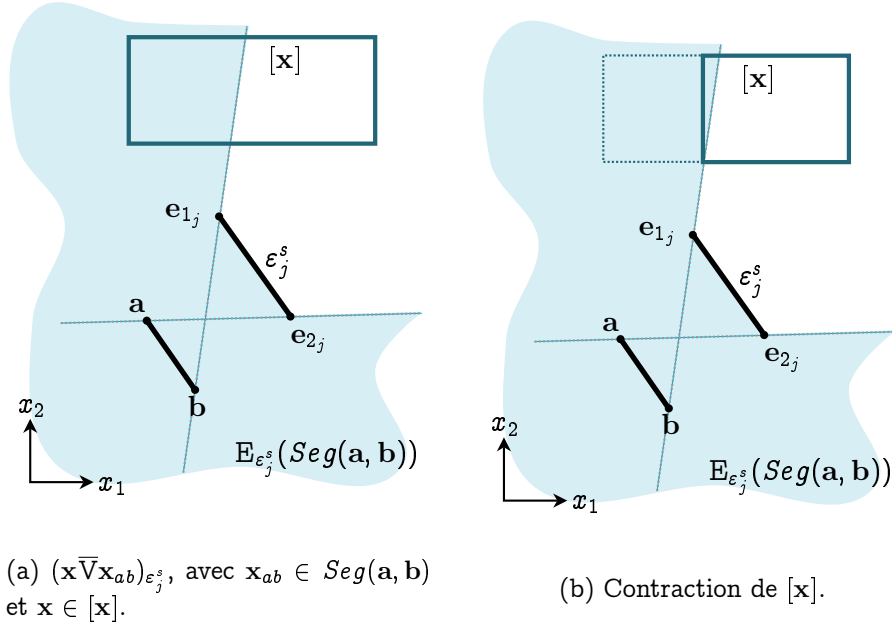


Figure 3.24 – Illustration du contracteur $C_{\bar{V}}([\mathbf{x}], \mathbf{a}, \mathbf{b}, \varepsilon_j^s)$.

3.2.2 Contracteur de non-visibilité d'un segment avec un obstacle segment

Le dual du contracteur de visibilité, le contracteur de non-visibilité, nommé $C_{\bar{V}}([\mathbf{x}], \mathbf{a}, \mathbf{b}, \varepsilon_j^s)$, est associé à la contrainte

$$(\bar{\mathbf{x}} \mathbf{x}_{ab})_{\varepsilon_j^s}, \quad (3.61)$$

avec $\mathbf{x}_{ab} \in \text{Seg}(\mathbf{a}, \mathbf{b})$, $\mathbf{a} \in \mathbb{R}^2$, $\mathbf{b} \in \mathbb{R}^2$, $\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1j}, \mathbf{e}_{2j})$, $\mathbf{e}_{1j} \in \mathbb{R}^2$, $\mathbf{e}_{2j} \in \mathbb{R}^2$ et $\mathbf{x} \in [\mathbf{x}]$.

Ce contracteur est présenté Algorithme 16. Il est basé sur le complément de l'Équation 3.45. La Figure 3.24 illustre son principe.

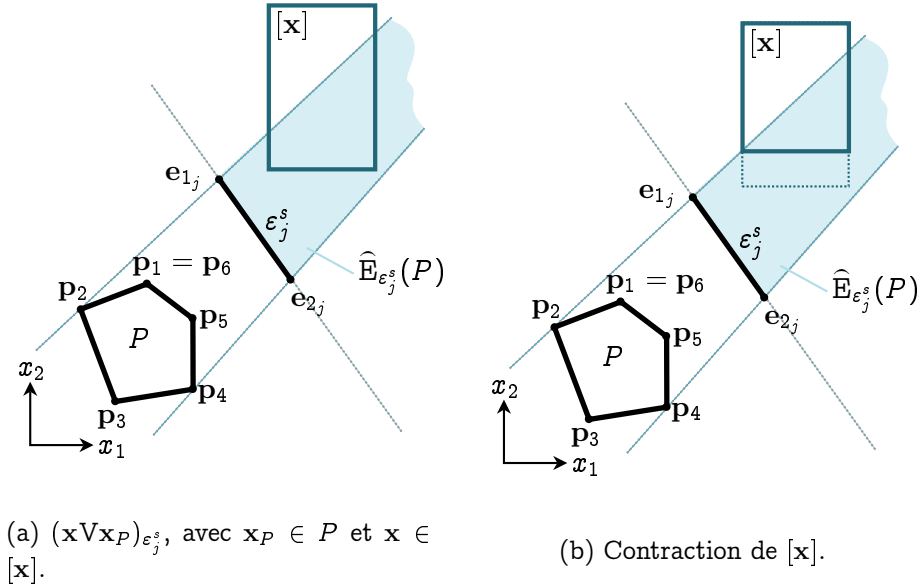
Il ne nous reste plus qu'à écrire les contracteurs de visibilité pour un polygone. Nous aurons ainsi créé les contracteurs qui nous intéressent et nous serons capable de contracter des pavés selon les contraintes de visibilité et non-visibilité.

Algorithme 16: Contracteur $C_{\overline{V}}([\mathbf{x}], \mathbf{a}, \mathbf{b}, \varepsilon_j^s)$

Données : $[\mathbf{x}], \mathbf{a}, \mathbf{b}, \varepsilon_j^s = \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$

- 1 si $\det(\mathbf{a} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0$ alors
- 2 | $\zeta_a = 1;$
- 3 sinon
- 4 | $\zeta_a = -1;$
- 5 fin
- 6 si $\det(\mathbf{b} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0$ alors
- 7 | $\zeta_b = 1;$
- 8 sinon
- 9 | $\zeta_b = -1;$
- 10 fin
- 11 si $\det(\mathbf{e}_{1_j} - \mathbf{a} | \mathbf{b} - \mathbf{a}) > 0$ alors
- 12 | $\zeta_{e_1} = 1;$
- 13 sinon
- 14 | $\zeta_{e_1} = -1;$
- 15 fin
- 16 si $\det(\mathbf{e}_{2_j} - \mathbf{a} | \mathbf{b} - \mathbf{a}) > 0$ alors
- 17 | $\zeta_{e_2} = 1;$
- 18 sinon
- 19 | $\zeta_{e_2} = -1;$
- 20 fin
- 21 // Deux cas possibles
- 22 si $\zeta_a = \zeta_b$ alors
- 23 | $[\mathbf{i}_{11}] = C_{det}([\mathbf{x}], \mathbf{e}_{1_j}, \mathbf{e}_{2_j}, -\zeta_a);$
- 24 | $[\mathbf{i}_{12}] = C_{det}([\mathbf{x}], \mathbf{e}_{1_j}, \mathbf{a}, -\zeta_a);$
- 25 | $[\mathbf{i}_{13}] = C_{det}([\mathbf{x}], \mathbf{e}_{1_j}, \mathbf{b}, -\zeta_b);$
- 26 | $[\mathbf{i}_{14}] = C_{det}([\mathbf{x}], \mathbf{e}_{2_j}, \mathbf{a}, \zeta_a);$
- 27 | $[\mathbf{i}_{15}] = C_{det}([\mathbf{x}], \mathbf{e}_{2_j}, \mathbf{b}, \zeta_b);$
- 28 | $[\mathbf{i}_{retour}] = [\mathbf{i}_{11}] \cap ([\mathbf{i}_{12}] \cup [\mathbf{i}_{13}]) \cap ([\mathbf{i}_{14}] \cup [\mathbf{i}_{15}])$
- 29 sinon
- 30 | // $\zeta_a = -\zeta_b$
- 31 | $[\mathbf{i}_{21}] = C_{det}([\mathbf{x}], \mathbf{e}_{1_j}, \mathbf{a}, -\zeta_{e_1});$
- 32 | $[\mathbf{i}_{22}] = C_{det}([\mathbf{x}], \mathbf{e}_{1_j}, \mathbf{b}, \zeta_{e_1});$
- 33 | $[\mathbf{i}_{23}] = C_{det}([\mathbf{x}], \mathbf{e}_{2_j}, \mathbf{a}, -\zeta_{e_2});$
- 34 | $[\mathbf{i}_{24}] = C_{det}([\mathbf{x}], \mathbf{e}_{2_j}, \mathbf{b}, \zeta_{e_2});$
- 35 | $[\mathbf{i}_{retour}] = ([\mathbf{i}_{21}] \cap [\mathbf{i}_{22}]) \cup ([\mathbf{i}_{23}] \cap [\mathbf{i}_{24}])$
- 36 fin
- 37 $[\mathbf{i}_0] = C_{\cap \neq \emptyset}([\mathbf{x}], \mathbf{a}, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) \cup C_{\cap \neq \emptyset}([\mathbf{x}], \mathbf{b}, \mathbf{e}_{1_j}, \mathbf{e}_{2_j});$

Résultat : $[\mathbf{x}]^* = [\mathbf{i}_{retour}] \cap [\mathbf{i}_0]$

Figure 3.25 – Illustration du contracteur $C_V([x], P, \varepsilon_j^s)$.

3.3 Contracteurs de visibilité d'un polygone

Cette fois nous allons devoir définir les contracteurs pour des obstacles de type segment, mais aussi pour des obstacles de type polygone.

3.3.1 Contracteur de visibilité d'un polygone avec un obstacle segment

Le contracteur $C_V([x], P, \varepsilon_j^s)$ est associé à la contrainte

$$(xVx_P)_{\varepsilon_j^s}, \quad (3.62)$$

avec $x_P \in P$, P un polygone composé de n_P sommets $p_k \in \mathbb{R}^2$, $\varepsilon_j^s = \text{Seg}(e_{1j}, e_{2j})$, $e_{1j} \in \mathbb{R}^2$, $e_{2j} \in \mathbb{R}^2$ et $x \in [x]$. La Figure 3.25 illustre son principe.

Ce contracteur est présenté Algorithm 17. Il est basé sur le complément de l'Équation 3.48.

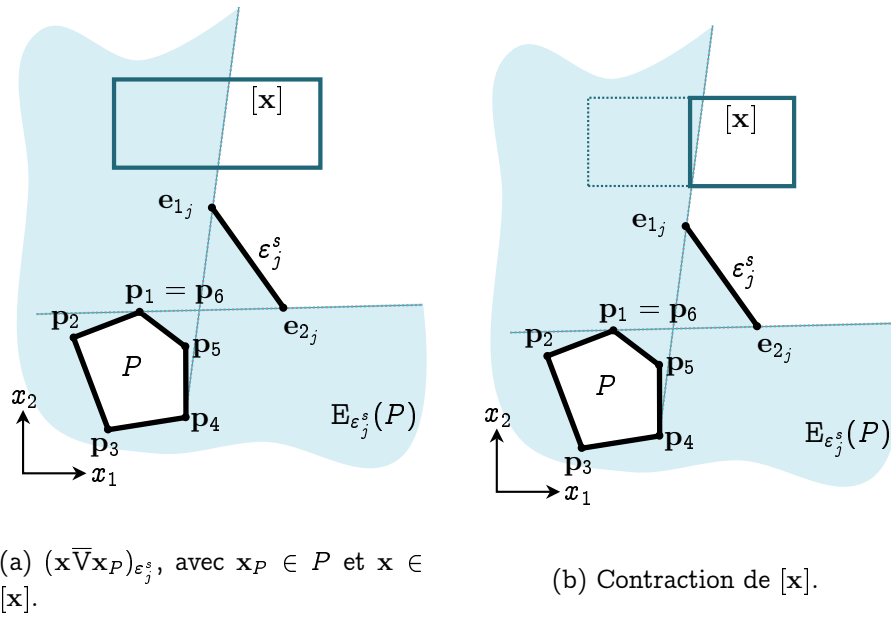
3.3.2 Contracteur de non-visibilité d'un polygone avec un obstacle segment

Le contracteur $C_{\overline{V}}([x], P, \varepsilon_j^s)$ est associé à la contrainte

$$(x\overline{V}x_P)_{\varepsilon_j^s}, \quad (3.63)$$

Algorithme 17: Contracteur $C_V([\mathbf{x}], P, \varepsilon_j^s)$

Données : $[\mathbf{x}], P, \varepsilon_j^s = \text{Seg}(\mathbf{e}_{1j}, \mathbf{e}_{2j})$
 1 pour tout $\mathbf{p}_k \in P$ faire
 2 | $[\mathbf{i}_k] = C_V([\mathbf{x}], \mathbf{p}_k, \mathbf{p}_{k+1}, \varepsilon_j^s)$;
 3 fin
 Résultat : $[\mathbf{x}]^* = \bigcup_{k=1}^{n_P} [\mathbf{i}_k]$

Figure 3.26 – Illustration du contracteur $C_{\bar{V}}([\mathbf{x}], P, \varepsilon_j^s)$.

avec $\mathbf{x}_P \in P$, P un polygone composé de n_P sommets $\mathbf{p}_k \in \mathbb{R}^2$, $\varepsilon_j^s = \text{Seg}(\mathbf{e}_{1j}, \mathbf{e}_{2j})$, $\mathbf{e}_{1j} \in \mathbb{R}^2$, $\mathbf{e}_{2j} \in \mathbb{R}^2$ et $\mathbf{x} \in [\mathbf{x}]$.

Ce contracteur est présenté Algorithme 18. Il est basé sur le complément de l'Équation 3.51. La Figure 3.26 illustre son principe.

Il ne nous reste plus qu'à définir les contracteurs de visibilité d'un polygone avec un obstacle de type polygone.

3.3.3 Contracteur de visibilité d'un polygone avec un obstacle polygone

Le contracteur $C_V([\mathbf{x}], P, \varepsilon_j^p)$ est associé à la contrainte

$$(\mathbf{x} V \mathbf{x}_P)_{\varepsilon_j^p}, \quad (3.64)$$

Algorithme 18: Contracteur $C_{\bar{V}}([x], P, \varepsilon_j^s)$

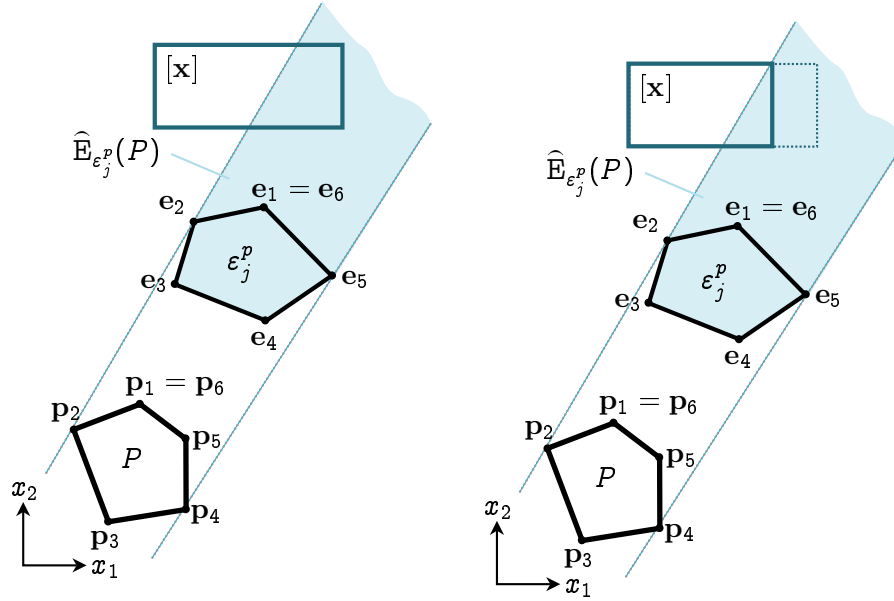
 Données : $[x], P, \varepsilon_j^s = \text{Seg}(e_{1j}, e_{2j})$

 1 pour tout $p_k \in P$ faire

 2 | $[i_k] = C_{\bar{V}}([x], p_k, \varepsilon_j^s)$;

3 fin

 Résultat : $[x]^* = \bigcup_{k=1}^{n_P} [i_k]$


 (a) $(x \bar{V} x_P)_{\varepsilon_j^p}$, avec $x_P \in P$ et $x \in [x]$.

 (b) Contraction de $[x]$.

 Figure 3.27 – Illustration du contracteur $C_{\bar{V}}([x], P, \varepsilon_j^p)$.

avec $x_P \in P$, P un polygone composé de n_P sommets $p_k \in \mathbb{R}^2$, ε_j^p un polygone composé de $n_{\varepsilon_j^p}$ sommets $e_k \in \mathbb{R}^2$ et $x \in [x]$.

Ce contracteur est présenté Algorithme 19. Il est basé sur le complément de l'Équation 3.55. La Figure 3.27 illustre son principe.

3.3.4 Contracteur de non-visibilité d'un polygone avec un obstacle polygone

Le contracteur $C_{\bar{V}}([x], P, \varepsilon_j^p)$ est associé à la contrainte

$$(x \bar{V} x_P)_{\varepsilon_j^p}, \quad (3.65)$$

Algorithme 19: Contracteur $C_V([\mathbf{x}], P, \varepsilon_j^p)$

Données : $[\mathbf{x}], P, \varepsilon_j^p$

- 1 **pour tout** $\mathbf{p}_k \in P$ **faire**
- 2 **pour tout** $\mathbf{e}_{k'} \in \varepsilon_j^p$ **faire**
- 3 $[\mathbf{j}_{k'}] = C_V([\mathbf{x}], \mathbf{p}_k, \text{Seg}(\mathbf{e}_{k'}, \mathbf{e}_{k'+1}))$;
- 4 **fin**
- 5 $[\mathbf{i}_k] = \bigcap_{k'=1}^{n_{P_j}} [\mathbf{j}_{k'}]$;
- 6 **fin**

Résultat : $[\mathbf{x}]^* = \bigcup_{k=1}^{n_P} [\mathbf{i}_k]$

avec $\mathbf{x}_P \in P$, P un polygone composé de n_P sommets $\mathbf{p}_k \in \mathbb{R}^2$, ε_j^p un polygone composé de n_{P_j} sommets $\mathbf{e}_k \in \mathbb{R}^2$ et $\mathbf{x} \in [\mathbf{x}]$.

Ce contracteur est présenté Algorithme 20. Il est basé sur le complément de l'Équation 3.54. La Figure 3.28 illustre son principe.

Algorithme 20: Contracteur $C_{\bar{V}}([\mathbf{x}], P, \varepsilon_j^p)$

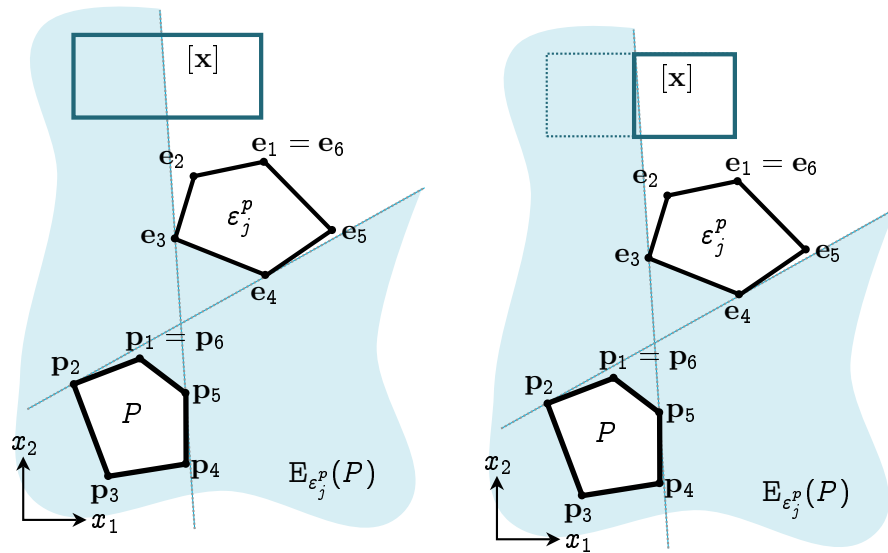
Données : $[\mathbf{x}], P, \varepsilon_j^p$

- 1 **pour tout** $\mathbf{p}_k \in P$ **faire**
- 2 **pour tout** $\mathbf{e}_{k'} \in \varepsilon_j^p$ **faire**
- 3 $[\mathbf{j}_{k'}] = C_{\bar{V}}([\mathbf{x}], \mathbf{p}_k, \mathbf{p}_{k+1}, \text{Seg}(\mathbf{e}_{k'}, \mathbf{e}_{k'+1}))$;
- 4 **fin**
- 5 $[\mathbf{i}_k] = \bigcup_{k'=1}^{n_{P_j}} [\mathbf{j}_{k'}]$;
- 6 **fin**

Résultat : $[\mathbf{x}]^* = \bigcup_{k=1}^{n_P} [\mathbf{i}_k]$

Nous avons maintenant tous les contracteurs nécessaires pour contracter des pavés sous une contrainte de (non-)visibilité. Il est possible de noter que tous ces contracteurs sont minimaux, ils permettent tous d'obtenir la plus petite boîte consistante avec les contraintes de visibilité. Le chapitre suivant présente deux applications de ces contracteurs, applications qui s'articulent autour du problème de localisation en robotique mobile.

Le lecteur notera cependant que ces deux applications ne sont pas exhaustives, les contracteurs présentés ici peuvent être utilisés dès qu'il est question de contraintes d'intersection ou de non-intersection (visibilité/non-visibilité).



(a) $(x \bar{\vee} x_P)_{\varepsilon_j^p}$, avec $x_P \in P$ et $x \in [x]$.

(b) Contraction de $[x]$.

Figure 3.28 – Illustration du contracteur $C_{\bar{\vee}}([x], P, \varepsilon_j^p)$.

Applications de la visibilité

Sommaire

1	Suivi de posture à l'aide d'une information booléenne. . .	126
1.1	Présentation du problème	126
1.2	Présentation de la méthode	130
1.3	Expérimentations et résultats	133
1.4	Conclusion sur cette application de la visibilité.	138
2	Visibilité pour l'algorithme IAL	139
2.1	Formalisation de la contrainte.	141
2.2	Méthode de traitement de la contrainte	141
2.3	Résultats	143
2.4	Conclusion sur cette application	144

Ce chapitre présente deux utilisations des contracteurs de visibilité introduits au chapitre précédent. Nous présentons ici deux applications de ces notions au problème de localisation en robotique mobile.

La première application s'intéresse au problème de suivi de posture dans un contexte extrême. L'objectif est de localiser une meute de robots en ne s'accordant qu'une mesure extéroceptive booléenne : la visibilité entre les robots. On s'intéresse ici au traitement de cette information afin d'éviter la dérive des robots au fil du temps.

La deuxième application s'interroge sur le principe des mesures télémétriques et propose une nouvelle contrainte, afin d'améliorer l'algorithme de localisation globale présenté au Chapitre 2. Cette nouvelle contrainte est prise en compte à l'aide des contracteurs de visibilité.

1 Suivi de posture à l'aide d'une information booléenne

La plupart des méthodes de localisation s'intéressent à la localisation d'un seul robot dans son environnement. Cependant le développement de techniques de communications inter-robots, permettant à des robots d'échanger des informations entre eux, a permis la mise en place de méthodes de localisation multi-robots. L'idée est alors de fusionner les informations des différents robots afin d'avoir une estimation plus précise et plus robuste de leurs positions. Il est notamment possible de noter des méthodes de localisation multi-robots basées sur le filtre de Kalman [Roumeliotis 2002, Roumeliotis 2004, Karam 2006] ou encore utilisant des filtres à particules [Fox 2000, Howard 2003]. Les travaux présentés ici ont plusieurs intérêts. Le premier est de proposer une méthode originale de localisation multi-robots. Ensuite, nous démontrerons qu'une information *faible* peut aussi, dans un certain contexte, permettre de résoudre le problème de localisation. En effet, la plupart des méthodes de localisation actuelles considèrent des informations plus ou moins *riches*, comme des images caméras ou des données télémètres (fournissant une distance et un angle)... Ici nous nous intéressons à une donnée booléenne (0 ou 1) qui, de ce fait, contient moins d'information que les exemples précisés ci-dessus. Pour finir, ces travaux proposent un outil qui peut être intégré à des méthodes de localisation plus classiques, afin par exemple d'en améliorer la robustesse en permettant le traitement de données difficilement exploitables avec ces méthodes.

L'information de visibilité a déjà été considérée pour s'intéresser au problème de localisation en robotique mobile [Rekleitis 2000, Dellaert 2003, Giguere 2012]. Mais ces approches associent l'information de visibilité entre les robots à des données de distance et/ou d'angle séparant les robots. A notre connaissance, il n'existe actuellement aucune méthode, autre que celle présentée dans ce document, pour traiter une information de visibilité exclusivement booléenne.

1.1 Présentation du problème

Comme introduit précédemment, nous nous intéressons ici à la localisation d'une meute de robots à l'aide d'une information booléenne : la visibilité inter robots.

1.1.1 Présentation des robots

Une meute de robots, notée \mathcal{R} , correspond à un ensemble de n_R robots r_i

$$\mathcal{R} = \{r_i\}, \quad i = 1, \dots, n_R. \quad (4.1)$$

On note \mathcal{R}_t l'état de la meute à l'instant t . \mathcal{R}_t correspond à l'état de chacun des robots composant la meute, à l'instant t .

$$\mathcal{R}_t = \{r_{i,t}\}, \quad i = 1, \dots, n_R. \quad (4.2)$$

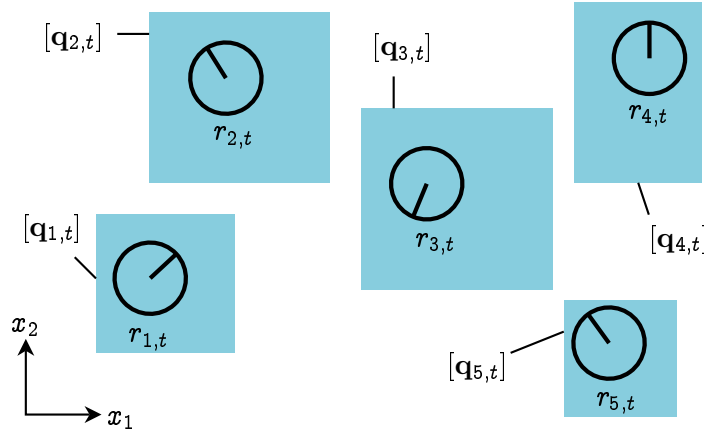


Figure 4.1 – Une meute de cinq robots. L'ensemble des pavés correspond à l'estimation de l'état de la meute à l'instant t . À l'aide de la communication inter-robots, tous les robots ont accès à cette information.

Par la suite on associera un robot $r_{i,t}$ à sa posture $\mathbf{q}_{i,t}$. On rappelle ici que la posture d'un robot correspond à sa position et son orientation dans l'environnement

$$\mathbf{q}_{i,t} = (\mathbf{x}_{i,t}, \theta_{i,t})^T = (x_{1,i,t}, x_{2,i,t}, \theta_{i,t})^T. \quad (4.3)$$

On considère les robots comme étant capables de communiquer entre eux. Un robot r_i est donc capable de transmettre l'estimation de sa posture à l'instant t aux autres robots. En d'autres termes tous les robots ont accès à l'estimation de \mathcal{R}_t . La Figure 4.1 présente une meute composée de cinq robots.

Tous les robots de la meute disposent aussi d'une technique d'odométrie : ils sont capables d'estimer leur déplacement entre deux instants t et $t + 1$. Afin de réduire la taille du problème de suivi de posture, les robots sont supposés équipés de boussole. Les orientations $\theta_{i,t}$ des robots sont donc connues, aux incertitudes près, à tout instant t . Le problème de localisation revient alors à estimer la position $\mathbf{x}_{i,t}$ (et non plus la posture $\mathbf{q}_{i,t}$) des robots à chaque instant t .

Pour terminer cette présentation, on introduit la donnée extéroceptive considérée pour localiser les robots : la visibilité entre les robots. Les capteurs extéroceptifs des robots permettent de définir si deux robots se voient ou non. Cette notion de visibilité est calquée sur celle présentée dans le Chapitre 3 :

$$r_1 \text{ voit } r_2 \text{ dans } \mathcal{E} \text{ à l'instant } t \Leftrightarrow (\mathbf{x}_{1,t} \mathbf{V} \mathbf{x}_{2,t})_{\mathcal{E}}, \quad (4.4)$$

$$r_1 \text{ ne voit pas } r_2 \text{ dans } \mathcal{E} \text{ à l'instant } t \Leftrightarrow (\mathbf{x}_{1,t} \bar{\mathbf{V}} \mathbf{x}_{2,t})_{\mathcal{E}}, \quad (4.5)$$

avec $\mathbf{x}_{1,t}$ la position de r_1 , $\mathbf{x}_{2,t}$ la position du robot r_2 et \mathcal{E} l'environnement dans lequel évoluent les deux robots.

L'information fournie par le capteur est booléenne : considérant deux robots, le capteur précise si oui ou non les robots se voient. On nomme $z_{i,t}$ le jeu de mesures

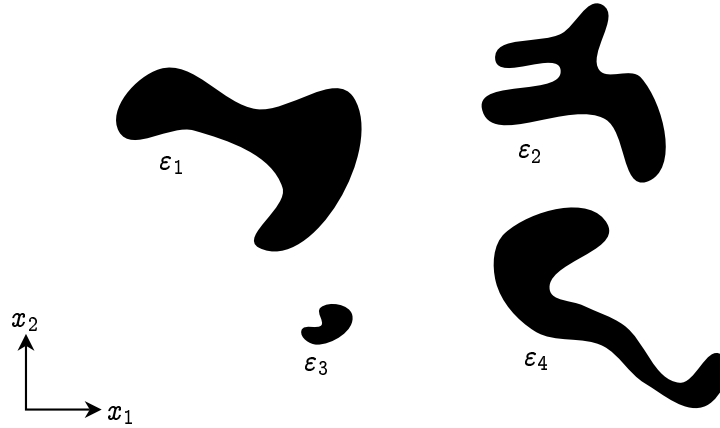


Figure 4.2 – Exemple d'environnement modélisé sous la forme d'ensembles connexes. Sur cet exemple $\mathcal{E} = \varepsilon_1 \cup \varepsilon_2 \cup \varepsilon_3 \cup \varepsilon_4$.

fait par le robot r_i à l'instant t . Ci-suit un exemple de jeu de mesures

$$\mathbf{z}_{i,t} = \{0, 1, 0, \dots, 1\}. \quad (4.6)$$

Ce dernier se traduit par : le robot r_i ne voit pas le robot r_1 à l'instant t (première mesure à 0), le robot r_i voit le robot r_2 à l'instant t (deuxième mesure à 1), le robot r_i ne voit pas le robot r_3 à l'instant t (troisième mesure à 0),..., le robot r_i voit le robot r_{n_R} à l'instant t (dernière mesure à 1).

1 .1.2 Présentation de l'environnement

L'environnement, noté \mathcal{E} , correspond à ensemble de n_O obstacles $\varepsilon_j, j = 1, \dots, n_O$.

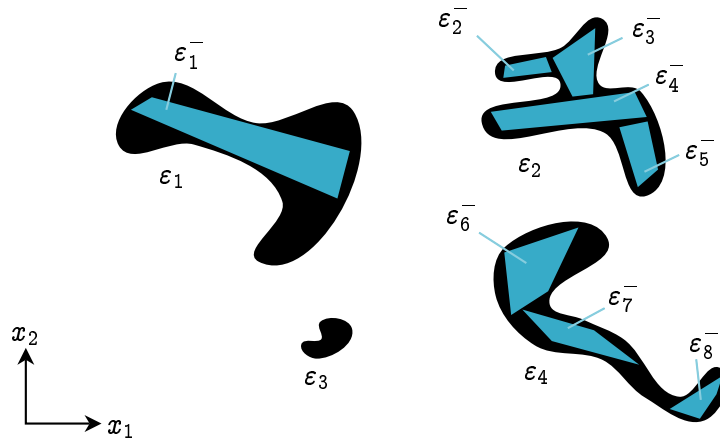
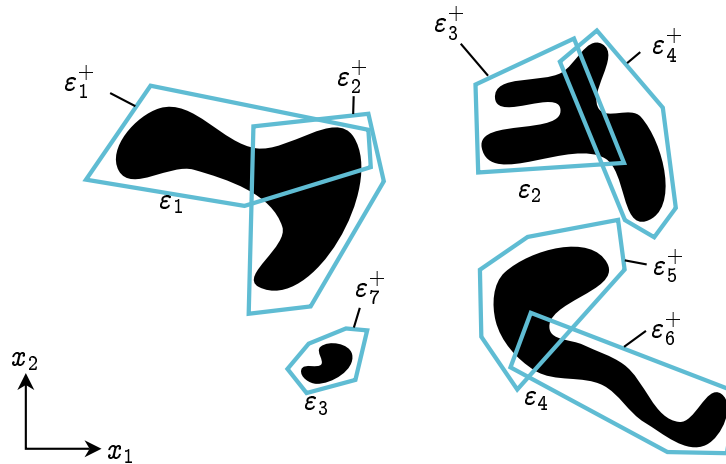
$$\mathcal{E} = \bigcup_{j=1}^{n_O} \varepsilon_j, \quad (4.7)$$

avec ε_j un sous-ensemble connexe de \mathbb{R}^2 . La Figure 4.2 présente un exemple d'environnement.

Comme pour tous les problèmes de localisation, l'environnement n'est pas connu parfaitement mais est caractérisé. Nous avons choisi ici de caractériser l'environnement de façon garantie, en utilisant le principe de la caractérisation intérieure et extérieure. L'environnement \mathcal{E} est donc estimé à l'aide d'une enveloppe extérieure \mathcal{E}^+ et une enveloppe intérieure \mathcal{E}^- , avec

$$\mathcal{E}^- \subseteq \mathcal{E} \subseteq \mathcal{E}^+. \quad (4.8)$$

Il est supposé qu'à tout instant t , l'Équation 4.8 est vérifiée. Ce qui veut dire que les obstacles mouvants (s'ils existent) sont actualisés dans les caractérisations, intérieure et extérieure, à chaque instant t .

Figure 4.3 – Exemple de caractérisation intérieure : $\mathcal{E}^- = \bigcup_{j=1}^8 \varepsilon_j^-$.Figure 4.4 – Exemple de caractérisation extérieure : $\mathcal{E}^+ = \bigcup_{j=1}^8 \varepsilon_j^+$.

Nous considérons ici des ensembles particuliers pour construire les caractérisations intérieures et extérieures. En effet, afin de réutiliser les contracteurs présentés dans le Chapitre 3, les caractérisations sont construites comme étant des ensembles de polygones convexes. Les Figures 4.3 et 4.4 présentent des exemples de caractérisations intérieures et extérieures telles que considérées ici. Ce qui, pour l'environnement présenté Figure 4.2 nous donne la carte présentée Figure 4.5.

1 .1.3 Présentation du problème de localisation

Nous nous intéressons ici au problème de suivi de posture. Par conséquent, les postures initiales des robots sont supposées connues, aux incertitudes près. L'objectif est alors d'estimer la position des robots au cours du temps en utilisant l'information de visibilité, associée à l'odométrie, pour éviter la dérive des robots.

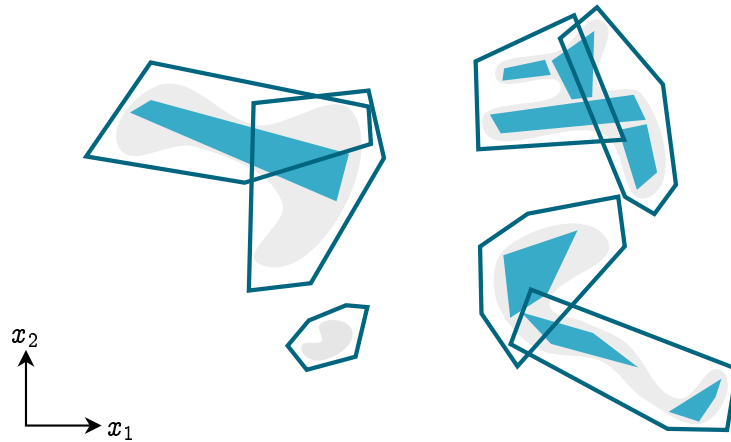


Figure 4.5 – Exemple de carte telle que considérée dans ce chapitre. Le gris clair correspond à l’environnement, les polygones pleins correspondent à la caractérisation intérieure et les polygones vides correspondent à la caractérisation extérieure.

Ce problème d’estimation de posture est considéré d’un point de vue ensembliste, en considérant un contexte à erreurs bornées. Les postures initiales des robots sont caractérisées par des boîtes, tout comme les erreurs d’odométrie ainsi que les données boussoles. Les robots se déplacent dans l’environnement, en estimant leur posture courante en fonction de leur posture précédente. Plus ils se déplacent, plus ils dérivent (plus la taille des estimations est importante, Figure 1.14 du Chapitre 1). L’objectif est alors d’utiliser l’information de visibilité afin de contracter et maintenir la précision sur la connaissance de la position des robots. La Figure 4.6 présente une illustration de ce problème de contraction.

1.2 Présentation de la méthode

Nous présentons maintenant la méthode mise en place afin de résoudre ce problème. Comme indiqué sur la Figure 4.6, il est nécessaire de pouvoir contracter deux boîtes en utilisant l’information de visibilité. En se basant sur la Remarque C.4 de l’Annexe C (stipulant qu’une boîte correspond à un polygone particulier) ainsi que les contracteurs présentés Algorithmes 20 et 19, il est possible de développer des contracteurs de visibilité et non visibilité entre deux boîtes, Algorithme 21 et 22. On notera que ces algorithmes utilisent la fonction *Boite2Polygone()* présentée Annexe C, Algorithme 44.

Les robots évoluant dans un environnement potentiellement composé de plusieurs obstacles, il est nécessaire d’étendre les deux algorithmes précédents. En s’appuyant sur les Propositions 3.4 et 3.4 (Chapitre 3, Section 1.4), il est possible de déduire les algorithmes 23 et 24.

Nous disposons donc maintenant de contracteurs permettant de contracter les

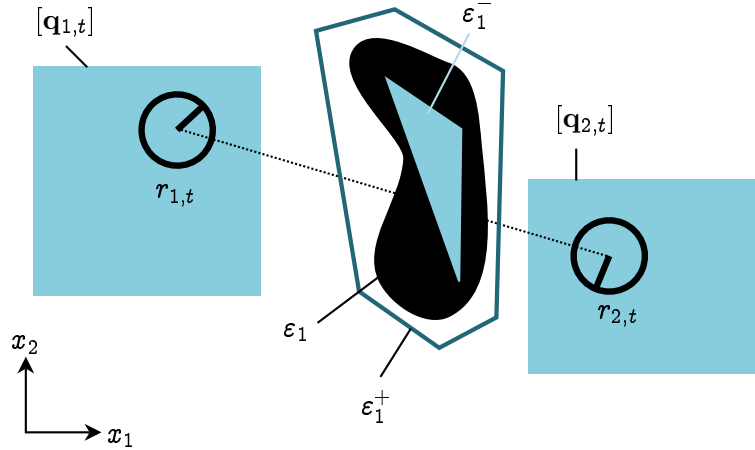


Figure 4.6 – Illustration simple du problème considéré : comment contracter les boîtes $[q_{1,t}]$ et $[q_{2,t}]$ sachant que les robots ne se voient pas? On notera que les éléments représentés en noir (obstacle et posture des robots), sont représentés à titre informatif et ne font pas parties des données du problème.

Algorithme 21: $C_V([x_1], [x_2], \varepsilon_j^p)$

Données : $[x_1], [x_2], \varepsilon_j^p$

- 1 // Contraction du premier pavé
- 2 $P_{x_2} = \text{Boite2Polygone}([x_2]);$
- 3 $[x_1]^* = C_V([x_1], P_{x_2}, \varepsilon_j^p);$
- 4 // Contraction du deuxième pavé
- 5 $P_{x_1} = \text{Boite2Polygone}([x_1]^*);$
- 6 $[x_2]^* = C_V([x_2], P_{x_1}, \varepsilon_j^p);$

Résultat : $[x_1]^*, [x_2]^*$.

Algorithme 22: $C_{\bar{V}}([x_1], [x_2], \varepsilon_j^p)$

Données : $[x_1], [x_2], \varepsilon_j^p$

- 1 // Contraction du premier pavé
- 2 $P_{x_2} = \text{Boite2Polygone}([x_2]);$
- 3 $[x_1]^* = C_{\bar{V}}([x_1], P_{x_2}, \varepsilon_j^p);$
- 4 // Contraction du deuxième pavé
- 5 $P_{x_1} = \text{Boite2Polygone}([x_1]^*);$
- 6 $[x_2]^* = C_{\bar{V}}([x_2], P_{x_1}, \varepsilon_j^p);$

Résultat : $[x_1]^*, [x_2]^*$.

Algorithme 23: $C_V([\mathbf{x}_1], [\mathbf{x}_2], \mathcal{E})$

Données : $[\mathbf{x}_1], [\mathbf{x}_2], \mathcal{E} = \{\varepsilon_j^p\}$

- 1 **pour** chaque $\varepsilon_j^p \in \mathcal{E}$ **faire**
- 2 | $([\mathbf{i}_{1,j}], [\mathbf{i}_{2,j}]) = C_V([\mathbf{x}_1], [\mathbf{x}_2], \varepsilon_j^p);$
- 3 **fin**

Résultat : $[\mathbf{x}_1]^* = \bigcap_j \{[\mathbf{i}_{1,j}]\}, [\mathbf{x}_2]^* = \bigcap_j \{[\mathbf{i}_{2,j}]\}.$

Algorithme 24: $C_{\overline{V}}([\mathbf{x}_1], [\mathbf{x}_2], \mathcal{E})$

Données : $[\mathbf{x}_1], [\mathbf{x}_2], \mathcal{E} = \{\varepsilon_j^p\}$

- 1 **pour** chaque $\varepsilon_j^p \in \mathcal{E}$ **faire**
- 2 | $([\mathbf{i}_{1,j}], [\mathbf{i}_{2,j}]) = C_{\overline{V}}([\mathbf{x}_1], [\mathbf{x}_2], \varepsilon_j^p);$
- 3 **fin**

Résultat : $[\mathbf{x}_1]^* = \bigcup_j \{[\mathbf{i}_{1,j}]\}, [\mathbf{x}_2]^* = \bigcup_j \{[\mathbf{i}_{2,j}]\}.$

boîtes sous une contraintes de visibilité.

La deuxième étape consiste à relier les informations de visibilité obtenues dans l'environnement aux caractérisations intérieures et extérieures. En effet les données fournies par les capteurs extéroceptifs correspondent aux informations de visibilité vis à vis de l'environnement, par exemple

$$(r_1 \vee r_2)_{\mathcal{E}}.$$

Malheureusement l'environnement \mathcal{E} ne fait pas partie des données disponibles pour résoudre notre problème. En revanche, les robots disposent des caractérisations \mathcal{E}^- et \mathcal{E}^+ . La Proposition 4.1 permet de faire le lien entre l'environnement et ses caractérisations, considérant l'information de visibilité.

Proposition 4.1 *Soient deux robots r_1 et r_2 évoluant dans un environnement \mathcal{E} caractérisé par un sous-ensemble \mathcal{E}^- et un sur-ensemble \mathcal{E}^+ , alors*

$$r_1 \text{ voit } r_2 \text{ dans } \mathcal{E} \Rightarrow r_1 \text{ voit } r_2 \text{ dans } \mathcal{E}^- \quad (4.9)$$

$$r_1 \text{ ne voit pas } r_2 \text{ dans } \mathcal{E} \Rightarrow r_1 \text{ ne voit pas } r_2 \text{ dans } \mathcal{E}^+ \quad (4.10)$$

Démonstration

$$\begin{aligned}
r_1 \text{ voit } r_2 \text{ dans } \mathcal{E} &\equiv (\mathbf{x}_1 \vee \mathbf{x}_2)_{\mathcal{E}} \text{ (Eq. 4.4),} \\
&\Rightarrow \forall \varepsilon_j \in \mathcal{E}, \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j = \emptyset \text{ (Eq. 3.1),} \\
\mathcal{E}^- \subseteq \mathcal{E} &\Rightarrow \forall \varepsilon_j^- \in \mathcal{E}^-, \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j^- = \emptyset, \\
&\Rightarrow (\mathbf{x}_1 \vee \mathbf{x}_2)_{\mathcal{E}^-}, \\
&\Rightarrow r_1 \text{ voit } r_2 \text{ dans } \mathcal{E}^-. \\
r_1 \text{ ne voit pas } r_2 \text{ dans } \mathcal{E} &\equiv (\mathbf{x}_1 \bar{\vee} \mathbf{x}_2)_{\mathcal{E}} \text{ (Eq. 4.5),} \\
&\Rightarrow \exists \varepsilon_j \in \mathcal{E} \mid \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j \neq \emptyset \text{ (Eq. 3.6),} \\
\mathcal{E}^+ \supseteq \mathcal{E} &\Rightarrow \exists \varepsilon_j^+ \in \mathcal{E}^+ \mid \text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \varepsilon_j^+ \neq \emptyset, \\
&\Rightarrow (\mathbf{x}_1 \bar{\vee} \mathbf{x}_2)_{\mathcal{E}^+}, \\
&\Rightarrow r_1 \text{ ne voit pas } r_2 \text{ dans } \mathcal{E}^+.
\end{aligned}$$

La Figure 4.7 illustre la Proposition 4.1.

Il est maintenant possible de définir l'algorithme de suivi de posture utilisant l'information de visibilité. Cet algorithme est présenté Algorithme 25. Il comprend deux étapes principales :

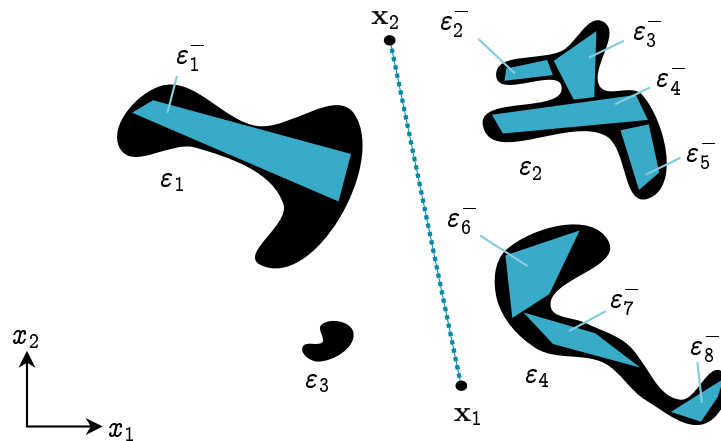
- L'initialisation des postures de chacun des robots, Ligne 2. Les boîtes $[\mathbf{q}_{i,0}]$ sont initialisées de façon à ce que $\mathbf{q}_{i,0} \in [\mathbf{q}_{i,0}]$, avec $\mathbf{q}_{i,0}$ la posture du robot r_i à l'instant initial $t = 0$.
- Les Lignes 3-16 correspondent au suivi de posture. L'instant t_{fin} symbolise la fin de la mission. Il peut théoriquement être infini. À chaque itération, trois actions sont réalisées
 - Ligne 6, Les robots calculent leur posture courante en fonction de leur posture précédente et des données odométriques.
 - Ligne 7, les robots partagent l'estimation de leur posture avec la meute.
 - Ligne 9-15, les postures des robots sont contractées (plus précisément leurs positions) en utilisant les informations de visibilité et les contracteurs.

Nous disposons donc maintenant d'un algorithme permettant d'éviter la dérive d'une meute de robots à l'aide d'une information booléenne.

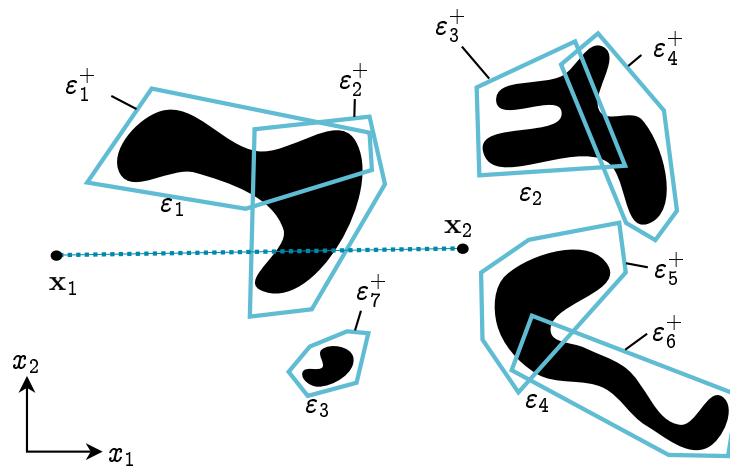
1.3 Expérimentations et résultats

L'efficacité de l'algorithme de localisation a été testé dans trois environnements simulés différents : \mathcal{E}^1 , \mathcal{E}^2 et \mathcal{E}^3 . Ces environnements sont représentés sur la Figure 4.8.

Chacun de ces environnements a une surface de $10\text{m} \times 10\text{m}$. Le tableau ci-dessous présente le nombre de segments de chacune de leurs caractérisations :



(a) Si $(x_1 \vee x_2)_{\mathcal{E}}$ alors, étant donné que $\mathcal{E}^- \subseteq \mathcal{E}$, on en déduit que $(x_1 \vee x_2)_{\mathcal{E}^-}$. Formulé autrement : si le segment $Seg(x_1, x_2)$ n'intersecte aucun obstacle de \mathcal{E} (en noir sur la figure) alors il n'intersecte aucun obstacle de \mathcal{E}^- (représenté par les polygones).



(b) Si $(x_1 \bar{\vee} x_2)_{\mathcal{E}}$ alors, étant donné que $\mathcal{E}^+ \supseteq \mathcal{E}$, on en déduit que $(x_1 \bar{\vee} x_2)_{\mathcal{E}^+}$. Formulé autrement : si le segment $Seg(x_1, x_2)$ intersecte au moins un obstacle de \mathcal{E} (en noir sur la figure) alors il intersecte forcément au moins un obstacle de \mathcal{E}^+ (représenté par les polygones).

Figure 4.7 – Illustration de la Proposition 4.1.

Algorithme 25: Estimation de posture avec l'information de visibilité

Données : $\mathcal{R}, \mathcal{E}^-, \mathcal{E}^+$

```

1 // Initialisation de l'état initial  $\mathcal{R}_0$ 
2 pour tous  $r_i \in \mathcal{R}$ , on initialise  $[\mathbf{q}_{i,0}]$ ;
3 pour  $t = 1$  à  $t_{fin}$  faire
4   // Évaluation de  $\mathcal{R}_t$  en fonction de  $\mathcal{R}_{t-1}$ 
5   //  $[\mathbf{u}_{i,t-1}]$  correspond aux données odométriques
6   pour tous  $r_i \in \mathcal{R}$ ,  $[\mathbf{q}_{i,t}] = f([\mathbf{q}_{i,t-1}], [\mathbf{u}_{i,t-1}])$ ;
7   pour tous  $r_i \in \mathcal{R}$ , partager  $[\mathbf{q}_{i,t}]$  avec la meute;
8   // Contraction à l'aide des informations de visibilité
9   pour chaque  $r_i \in \mathcal{R}$ ,  $r_{i'} \in \mathcal{R}$ ,  $r_i \neq r_{i'}$  faire
10    si  $r_i$  voit  $r_{i'}$  alors
11      |  $([\mathbf{x}_{i,t}], [\mathbf{x}_{i',t}]) = C_V([\mathbf{x}_{i,t}], [\mathbf{x}_{i',t}], \mathcal{E}^-)$ ;
12    sinon
13      |  $([\mathbf{x}_{i,t}], [\mathbf{x}_{i',t}]) = C_{\bar{V}}([\mathbf{x}_{i,t}], [\mathbf{x}_{i',t}], \mathcal{E}^+)$ ;
14    fin
15  fin
16 fin

```

Résultat : $\mathcal{R}_{t_{fin}}$.

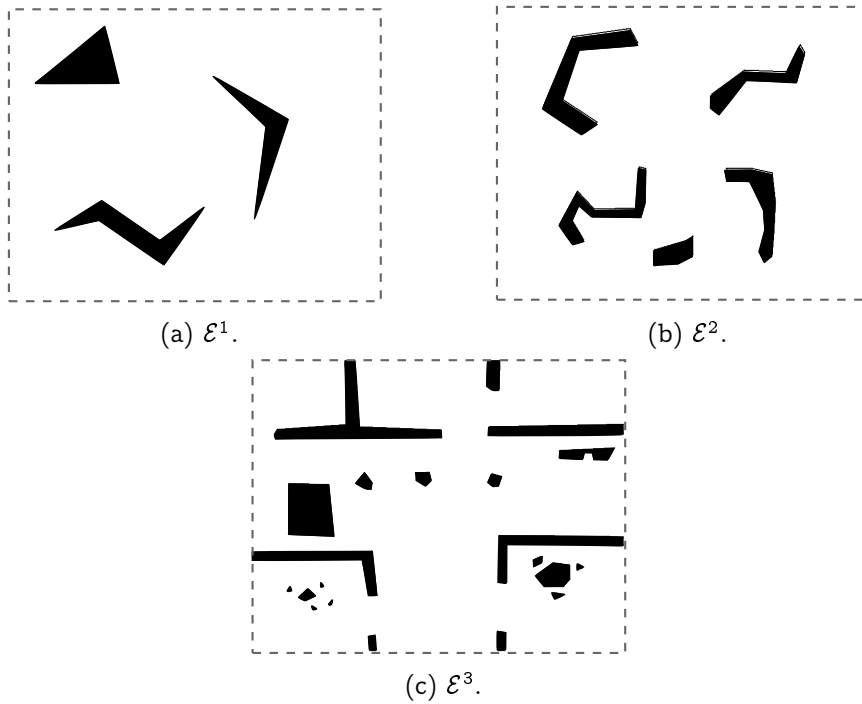


Figure 4.8 – Trois environnements expérimentaux.

	\mathcal{E}^1	\mathcal{E}^2	\mathcal{E}^3
\mathcal{E}^-	19	59	89
\mathcal{E}^+	26	69	101

Nous présentons ici le nombre de segments et non pas le nombre de polygones, car les polygones sont composés d'un nombre de sommets variés. De plus les polygones étant au final considérés comme des ensembles de segments (au sein des contracteurs), le temps de calcul dépend directement du nombre de segments.

Entre chaque pas de temps, les robots se déplacent de 20cm avec une erreur bornée de $\pm 1\%$. La boussole de chacun des robots est supposée exacte à ± 1 deg près.

À l'instant initial $t = 0$, nous avons

$$\forall r_i \in \mathcal{R}, [\mathbf{q}_{i,0}] = ([\mathbf{x}_{i,0}], [\theta_{i,0}]), \quad (4.11)$$

avec

$$[\mathbf{x}_{i,0}] = [\mathbf{x}_{i,0} - 50\text{cm}, \mathbf{x}_{i,0} + 50\text{cm}], \quad (4.12)$$

et

$$[\theta_{i,0}] = [\theta_{i,0} - 1 \text{ deg}, \theta_{i,0} + 1 \text{ deg}]. \quad (4.13)$$

Le processeur utilisé pour les simulations a les caractéristiques suivantes :

Intel(R) Core(TM) CPU - 6420 @ 2.13GHz.

Pendant les tests, les robots évoluent de façon pseudo-aléatoire dans l'environnement, de l'instant $t = 0$ à l'instant $t_{fin} = 1500$. En d'autres termes, nous effectuons 1500 itérations de notre algorithme de localisation. Les résultats de ces expérimentations sont présentés Tableau 4.1. On notera que *moyenne des $w([x_{2,i,t}])$* correspond à la taille moyenne de tous les intervalles $w([x_{2,i,t}])$ pour l'ensemble des itérations, alors que *moyenne des $w([x_{2,i,1500}])$* correspond à la taille moyenne des intervalles $w([x_{2,i,1500}])$ de la dernière itération. Les données notées en gras, Tableau 4.1, correspondent aux cas pour lesquels les robots ne dérivent pas lors du processus de localisation. Pour les autres cas, la localisation des robots a échouées, l'imprécision sur la posture augmente et n'est pas contenue : les robots finissent par dériver. Les robots sont considérés comme ne dérivant pas quand

- la taille moyenne des estimations finales des postures des robots est inférieure ou égale à la taille moyenne des estimations initiales,
- la taille moyenne des estimations finales des postures des robots correspond à la taille moyenne des estimations tout au long du processus de localisation.

Il est possible de remarquer qu'il existe deux facteurs importants au succès de la localisation : la topologie de l'environnement et le nombre de robots considérés.

		nombre de robots					
		4	8	12	16	20	24
\mathcal{E}^1	moyenne des w ($[x_{1_i,t}]$)	439	310	196	131	127	115
	moyenne des w ($[x_{2_i,t}]$)	460	334	176	147	121	117
	moyenne des w ($[x_{1_i,1500}]$)	897	668	172	147	121	117
	moyenne des w ($[x_{2_i,1500}]$)	919	674	193	167	159	133
	temps d'itération moyen	3	28	92	204	351	589
\mathcal{E}^2	moyenne des w ($[x_{1_i,t}]$)	545	355	183	102	93	84
	moyenne des w ($[x_{2_i,t}]$)	552	375	185	115	108	97
	moyenne des w ($[x_{1_i,1500}]$)	1029	780	440	107	100	62
	moyenne des w ($[x_{2_i,1500}]$)	1038	811	435	110	97	95
	temps d'itération moyen	11	65	191	443	745	1160
\mathcal{E}^3	moyenne des w ($[x_{1_i,t}]$)	404	120	75	62	63	56
	moyenne des w ($[x_{2_i,t}]$)	325	77	59	50	47	46
	moyenne des w ($[x_{1_i,1500}]$)	817	126	81	69	87	47
	moyenne des w ($[x_{2_i,1500}]$)	688	65	58	50	57	45
	temps d'itération moyen	15	116	313	646	1058	1727
Sans l'information de visibilité - seulement l'odométrie et la boussole -							
moyenne des w ($[x_{1_i,t}]$)		588	moyenne des w ($[x_{1_i,1500}]$)		1075		
moyenne des w ($[x_{2_i,t}]$)		595	moyenne des w ($[x_{2_i,1500}]$)		1084		

Tableau 4.1 – Résultats expérimentaux (les données sont en cm et ms).

Il apparaît que pour un environnement donné il existe un nombre minimum de robots nécessaires pour permettre d'éviter la dérive de ces derniers. Ceci s'explique par le fait qu'avec un nombre de robots insuffisant, le vecteur de mesures contient trop peu d'informations. Dans nos expérimentations, il est possible de noter que 8 robots sont nécessaires pour une localisation efficace dans l'environnement \mathcal{E}^3 , 12 robots pour l'environnement \mathcal{E}^1 et 16 robots pour l'environnement \mathcal{E}^2 . On notera cependant que trop de robots peut aussi nuire à la localisation de ces derniers. En effet, on peut facilement imaginer qu'un environnement surpeuplé va gêner les déplacements des robots mais aussi limiter leurs champs de visions : en supposant que les robots ne soient pas transparents, un robot placé entre deux autres bloque la détection mutuelle de ces derniers.

On remarque aussi que pour un nombre donné de robots, la topologie de l'environnement est déterminante pour le succès de la localisation. Avec 8 robots par exemple, l'algorithme permet aux robots de se localiser dans l'environnement \mathcal{E}^3 , mais ne permet pas d'éviter la dérive de ces derniers dans les environnements \mathcal{E}^1 et \mathcal{E}^2 . La quantité d'obstacles présents dans l'environnement, leurs tailles et leurs dispositions influencent le résultat de la localisation. Ceci s'explique par le fait qu'une information de visibilité "utile" est une information inattendue : le fait que deux robots se voient alors qu'ils ont plus de chance de ne pas se voir va permettre de contracter efficacement. À l'inverse savoir que deux robots se voient alors qu'il n'y a aucun obstacle dans l'environnement n'est pas une information utile (dans un sens où elle ne permet pas de contracter la posture des robots). En partant de ce principe, on en conclut qu'un environnement avec trop d'obstacles, au sein duquel les robots ne se verraient presque jamais, ne va pas permettre à l'algorithme de localiser efficacement les robots. Le même raisonnement s'applique pour les environnements avec trop peu d'obstacles : les robots se verraient constamment. Mais la quantité des obstacles n'est pas un critère suffisant, leur répartition dans l'environnement est aussi une donnée importante : beaucoup d'obstacles localisés au même endroit sera moins efficace que moins d'obstacle bien répartis.

1.4 Conclusion sur cette application de la visibilité

Dans cette section nous avons montré que la visibilité, associée à l'analyse par intervalles, permet de localiser une meute de robots (d'éviter leur dérive). L'algorithme présenté ici permet d'exploiter une donnée booléenne afin de localiser une meute de robots, ce qui, à notre connaissance, n'a pas été fait avec les méthodes classiques comme les filtres particulaires (Monte Carlo). On notera notamment que les résultats de l'algorithme sont garantis (sous l'hypothèse d'erreurs bornées et considérant des caractérisations garanties).

Cette section présente un cas extrême pour la localisation, dans un sens où seule l'information de visibilité entre les robots est utilisée afin de compenser les erreurs d'odométrie. En pratique, les robots disposent de capteurs fournissant plus d'infor-

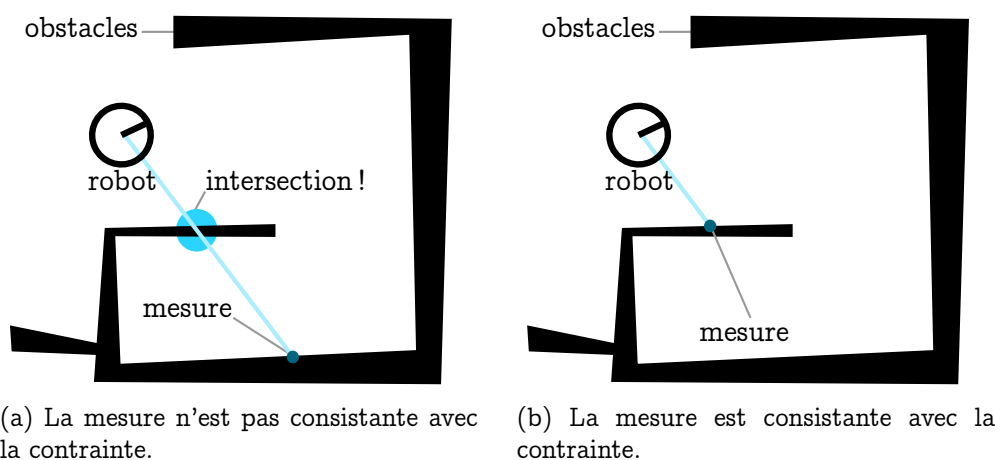


Figure 4.9 – Illustration de la contrainte considérée.

mations (capteurs ultrasons, caméras, ...). L'intérêt de la méthode dans ce cas n'est pas de localiser les robots seulement avec l'information de visibilité mais d'ajouter le traitement de cette information à un algorithme de localisation classique afin, par exemple, d'améliorer sa robustesse.

Dans la prochaine section nous présentons une nouvelle application de la visibilité qui intègre cette dernière à l'algorithme de localisation IAL, présenté au Chapitre 2. Cette fois la visibilité n'est pas utilisée comme unique moyen de localisation mais est intégrée à un algorithme existant pour en améliorer l'efficacité et la robustesse.

2 Visibilité pour l'algorithme IAL

Un télémètre (laser ou sonar) renvoie la distance séparant le capteur de l'obstacle le plus proche dans une direction donnée. C'est le principe même de fonctionnement de ces capteurs. Il n'est donc pas possible pour ces capteurs de détecter un obstacle *derrière* un autre obstacle.

En considérant un télémètre laser, cela revient à dire que le rayon laser de la mesure ne doit pas intersecter d'obstacle (en plus de celui intersecté à l'extrémité du rayon, et qui donc, a généré la mesure). On a donc une relation de visibilité entre le robot et l'obstacle détecté par le capteur.

C'est ce qui est présenté Figure 4.9. Cette relation de visibilité (non-intersection) peut être traitée comme une contrainte sur le jeu de mesures : le robot doit avoir une relation de visibilité avec tous les obstacles détectés. L'intérêt de la mise en place d'une telle contrainte est présenté Figure 4.10. On remarque alors qu'elle permet d'éviter la considération de *fausses symétries*.

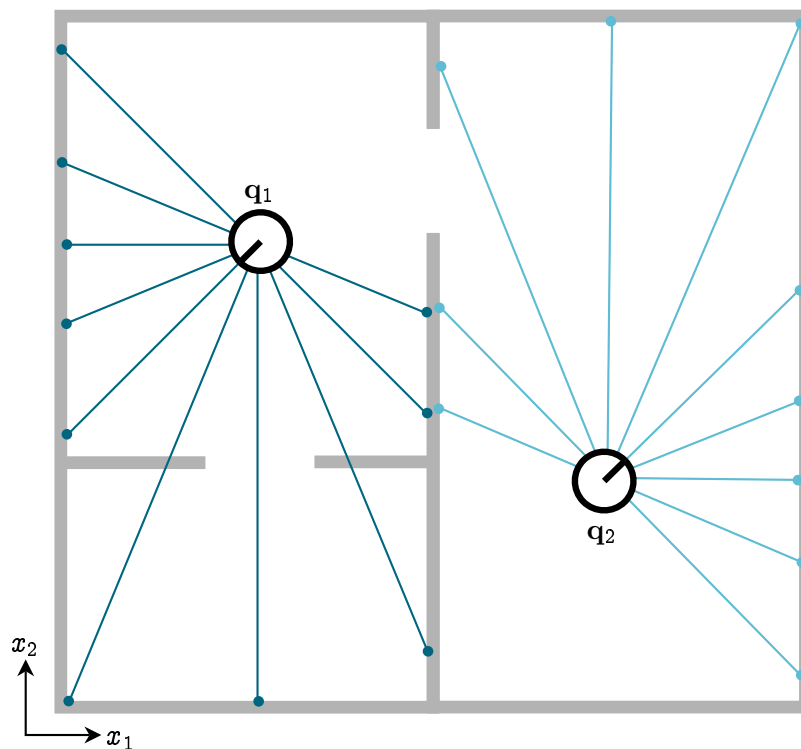
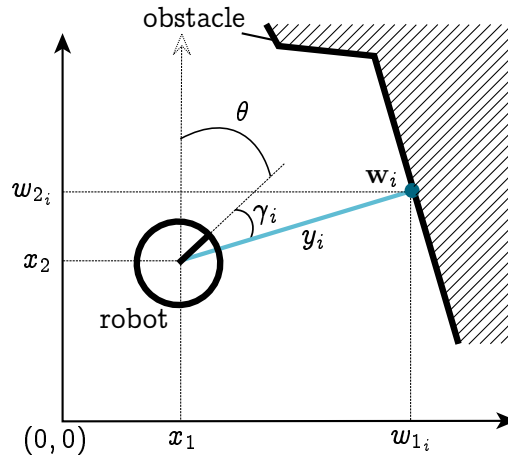


Figure 4.10 – Sans tenir compte de la contrainte proposée ici, on identifie les deux postures q_1 et q_2 comme étant compatibles avec le jeu de mesures. En revanche, en ajoutant la contrainte spécifiant que le robot doit *voir* les obstacles détectés, il est possible d'identifier q_1 comme étant non compatible avec le jeu de mesures (en effet pour q_1 deux mesures ne sont pas compatibles avec la contrainte).

Figure 4.11 – Le robot effectue une mesure w_i .

2.1 Formalisation de la contrainte

Soit un robot r effectuant un jeu de n_w mesures (à l'aide d'un télémètre laser). On note $w_i = (w_{1_i}, w_{2_i})$ les coordonnées de l'obstacle détecté par la $i^{\text{ème}}$ mesure, $i = 1, \dots, n_w$ (Figure 4.11). En notant $\mathbf{x} = (x_1, x_2)$ la position du robot, la contrainte présentée précédemment peut s'écrire :

$$\forall \varepsilon_j \in \mathcal{E}, w_i \in \varepsilon_j \vee \text{Seg}(\mathbf{x}, w_i) \cap \varepsilon_j = \emptyset, \quad (4.14)$$

avec \mathcal{E} l'environnement dans lequel le robot effectue ses mesures. En d'autres termes, l'Équation 4.14 signifie que pour tous les obstacles qui composent l'environnement, soit la mesure appartient à l'obstacle, soit le rayon laser de la mesure n'intersecte pas l'obstacle.

En reprenant la notation introduite au chapitre précédent, nous pouvons donc écrire la contrainte sous la forme

$$\forall \varepsilon_j \in \mathcal{E}, w_i \in \varepsilon_j \vee (\mathbf{x} \vee w_i)_{\varepsilon_j}, \quad (4.15)$$

ou encore

$$\forall \varepsilon_j \in \mathcal{E}, w_i \in \varepsilon_j \vee \left(\mathbf{x} \in \mathbb{E}_{\varepsilon_j}(w_i) \wedge w_i \in \mathbb{E}_{\varepsilon_j}(\varepsilon_j) \right). \quad (4.16)$$

2.2 Méthode de traitement de la contrainte

Nous voulons maintenant être capable de traiter cette nouvelle information afin de l'ajouter à l'algorithme de localisation globale.

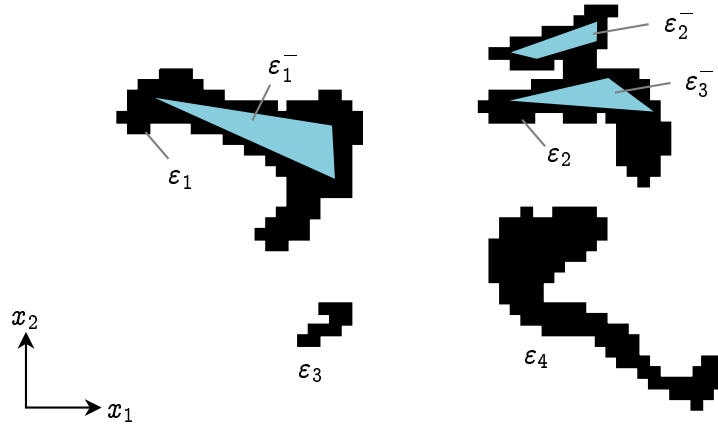


Figure 4.12 – Exemple de caractérisation intérieure : on remarque que tous les obstacles ne sont pas caractérisés.

2.2.1 Caractérisation de l'environnement

L'algorithme de localisation globale présenté au Chapitre 2 caractérise l'environnement à l'aide d'une grille d'occupation. Cependant, afin de pouvoir manipuler les notions de visibilité présentées au chapitre précédent, nous allons ajouter à cette grille d'occupation une caractérisation supplémentaire. Comme pour la section précédente, l'environnement \mathcal{E} est considéré comme étant un ensemble de n_O obstacles, ε_j , $j = 1, \dots, n_O$:

$$\mathcal{E} = \bigcup_{j=1}^{n_O} \varepsilon_j, \quad (4.17)$$

avec ε_j un sous-ensemble connexe de \mathbb{R}^2 .

La contrainte correspond à une relation de visibilité (Équation 4.15). Comme précisé Proposition 4.1, pour contracter sur une relation de visibilité il est possible d'utiliser une caractérisation intérieure de l'environnement. A chaque obstacle $\varepsilon_j \in \mathcal{E}$, nous associons alors une (ou plusieurs) caractérisation(s) intérieure(s) $\varepsilon_{j'}^-$, de façon à ce que

$$\varepsilon_{j'}^- \subseteq \varepsilon_j. \quad (4.18)$$

On notera que $\varepsilon_{j'}^-$ peut correspondre à l'ensemble vide \emptyset .

La Figure 4.12 présente un exemple de caractérisation. A priori, le robot aura toujours une relation de visibilité avec l'obstacle détecté, il n'est donc pas nécessaire d'ajouter une caractérisation extérieure (utilisée pour contracter sur une relation de non-visibilité, Proposition 4.1).

Le lecteur remarquera que la caractérisation intérieure peut s'obtenir à l'aide d'une technique de vectorisation de grille [Delchev 2006]. Ces techniques de vectori-

sation sont déjà utilisées en robotique mobile, par exemple [Kieffer 2000] utilise une vectorisation de l'environnement pour pouvoir localiser le robot.

2.2.2 Algorithme de contraction

Afin de pouvoir intégrer cette contrainte à notre algorithme de localisation globale il est nécessaire de développer un contracteur qui permet de la prendre en compte. Ce contracteur est présenté Algorithme 26. Il permet de contracter la posture (et la mesure) en fonction de la contrainte qui stipule que le robot doit "voir" la mesure. On notera que cet algorithme fait appel au contracteur de visibilité entre deux boîtes, Algorithme 21.

Algorithme 26: $C_V([\mathbf{q}], [\mathbf{w}], \mathcal{E}^-)$

Données : $[\mathbf{q}], [\mathbf{w}], \mathcal{E}^- = \{\varepsilon_{j'}^-\}$

1 // Initialisations

2 $[\mathbf{q}]^* = [\mathbf{q}];$

3 $[\mathbf{w}]^* = [\mathbf{w}];$

4 // Contractions

5 **pour** chaque $\varepsilon_{j'}^-$ **faire**

6 | $([\mathbf{q}]^*, [\mathbf{w}]^*) = C_V([\mathbf{q}]^*, [\mathbf{w}]^*, \varepsilon_{j'}^-);$

7 **fin**

Résultat : $([\mathbf{q}]^*, [\mathbf{w}]^*).$

Maintenant que nous disposons d'un contracteur associé à notre nouvelle contrainte, il est possible de l'ajouter au CSP présenté dans le Chapitre 2. Ainsi, on est capable de prendre en compte cette nouvelle contrainte.

2.3 Résultats

Afin de tester l'intérêt de ce nouveau contracteur nous l'avons ajouté à l'algorithme de localisation globale présenté au Chapitre 2. L'environnement considéré pour notre expérimentation est présenté Figure 4.13a. À la grille d'occupation utilisée par l'algorithme de localisation nous avons ajouté une caractérisation intérieure (Figure 4.13b).

Le robot effectue un jeu de 30 mesures, représenté Figure 4.13c. L'objectif de l'expérimentation est de localiser globalement le robot dans l'environnement à l'aide du jeu de mesures. En utilisant l'algorithme original présenté au Chapitre 2, nous obtenons deux postures distinctes compatibles avec les mesures. Ces postures solutions sont représentées Figure 4.13d. La présence de ces deux solutions distinctes s'explique par la présence de symétries dans l'environnement.

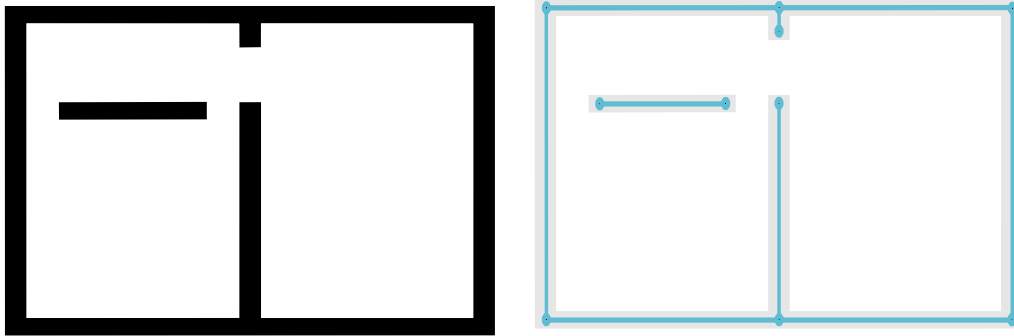
En ajoutant cette fois notre nouveau contracteur de visibilité d'obstacle, associé à la contrainte définie par l'Équation 4.15, nous obtenons le résultat présenté Figure 4.13e. Il est possible de remarquer que l'ambiguïté, présente sur le résultat précédent, est maintenant levée. L'algorithme est capable de ne proposer qu'une seule solution à ce problème de localisation. Nous avons donc bien réussi à prendre en compte cette nouvelle information, ce qui nous permet d'améliorer le résultat de la localisation.

On notera que l'ajout de cette nouvelle contrainte n'augmente pas considérablement le temps de calcul. En effet, il faut 1s pour calculer la posture du robot sans cette contrainte (Figure 4.13d) et 1.5s pour calculer la posture du robot en incluant la contrainte, et donc en levant l'ambiguïté (Figure 4.13e).

2.4 Conclusion sur cette application

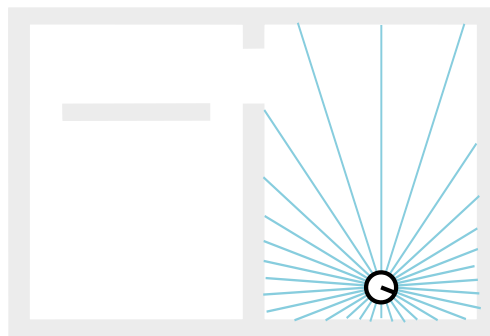
L'ajout de cette nouvelle contrainte à notre algorithme de localisation permet d'améliorer les résultats en limitant les symétries possibles dans l'environnement. En utilisant les notions de visibilité, il est possible de développer un contracteur associé à cette contrainte, et ainsi de pouvoir en tenir compte lors du processus de localisation.

Certains pourront soulever la nécessité de posséder une caractérisation intérieure de l'environnement. Ceci représente un faux problème : comme précisé précédemment, ces caractérisations peuvent être réalisées en s'appuyant sur des techniques de vectorisation. De plus, on notera qu'il n'est pas nécessaire de caractériser tous les obstacles de l'environnement. En effet, en reprenant l'exemple de notre expérimentation nous remarquons que la caractérisation représentée sur la Figure 4.14 permet d'obtenir le même résultat qu'avec une caractérisation complète de l'environnement (Figure 4.13e).

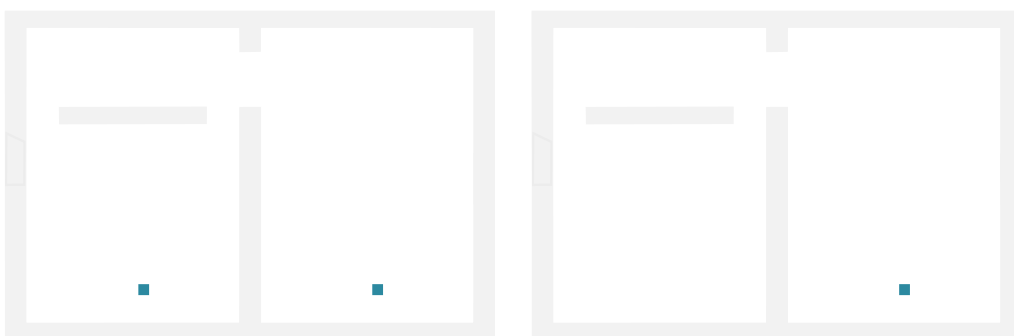


(a) L'environnement considéré.

(b) La caractérisation intérieure.



(c) Le jeu de mesures considéré.



(d) Localisation sans visibilité.

(e) Localisation avec visibilité.

Figure 4.13 – Présentation de l'expérimentation.

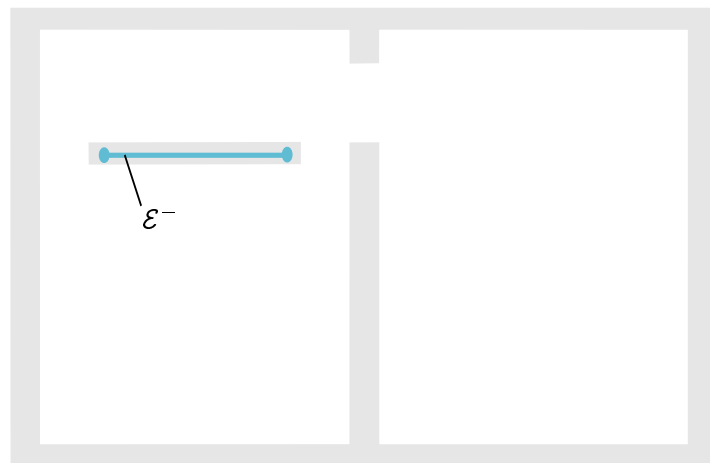


Figure 4.14 – Caractérisation partielle de l'environnement.

Conclusion générale

Cette thèse s'inscrit dans l'étude des problèmes de localisation en robotique mobile, et plus particulièrement aux problèmes de localisation de robots mobiles terrestres (UGV) à l'intérieur de bâtiments. Nous nous sommes penchés sur l'intérêt de l'utilisation de l'analyse par intervalles pour le développement de méthodes permettant la résolution de tels problèmes. En effet à l'heure actuelle, les méthodes probabilistes sont les plus utilisées, et connues comme étant les plus performantes. Que peut apporter l'approche ensembliste face aux solutions déjà en place ?

Dans ce contexte nous nous sommes intéressés à différents problèmes de localisation, nous avons développé des méthodes de résolutions ensemblistes et nous les avons comparées aux méthodes probabilistes classiquement utilisées.

La première contribution de cette thèse repose sur la présentation d'un algorithme de localisation ensembliste original permettant de s'intéresser au problème de localisation globale en robotique mobile. Ce problème, considéré comme le plus difficile parmi les problèmes de localisation, consiste à localiser un robot dans son environnement, sans n'avoir aucune connaissance a priori sur sa posture initiale (position et orientation).

Pour résoudre ce problème nous avons développé une méthode ensembliste, baptisée IAL (Interval Analysis Localization - Localisation à l'aide de l'Analyse par Intervalles). Cette méthode repose principalement sur trois outils de l'analyse par intervalle :

- la résolution de problèmes de satisfaction de contraintes à l'aide de contracteurs,
- la bisection de vecteurs d'intervalles,
- et l'intersection relaxée d'ensembles.

La méthode consiste à représenter le problème de localisation comme un problème de satisfaction de contraintes, en utilisant la bisection pour affiner les résultats et l'intersection relaxée pour améliorer la robustesse de l'algorithme (traiter les valeurs aberrantes).

En considérant un contexte à erreurs bornées, on est alors capable de localiser

un robot globalement dans son environnement et ce, de manière garantie.

Afin de valider cette approche nous avons effectué plusieurs expérimentations. Elles permettent d'apprécier l'intérêt de la méthode dans un contexte réel, et de la comparer avec l'approche probabiliste classique : MCL (Monte Carlo Localization - Localisation de Monte Carlo). L'algorithme IAL apparaît comme étant efficace et ce, même dans des conditions de localisation difficiles :

- en présence de valeurs aberrantes dans le jeu de mesures,
- en présence d'erreurs dans la carte considérée.

La méthode s'avère être utilisable dans un contexte réel, ce qui a été démontré par différentes manipulations expérimentales. Les temps de calculs de notre approche, bien que pouvant être plus élevés que ceux de l'approche probabiliste MCL, sont généralement comparables avec ceux de cette dernière. De plus l'analyse par intervalles permet une garantie des résultats, ce que ne permet pas les outils probabilistes. Cet algorithme met en avant les avantages non négligeables de l'analyse par intervalles, quand on s'intéresse aux problèmes de localisation en robotique mobile.

La deuxième contribution de cette thèse correspond à la mise en place de contracteurs de visibilité : deux points sont considérés visibles dans un environnement si le segment défini par ces deux points n'intersecte aucun obstacle de l'environnement (sinon les points sont considérés non-visibles). Ces notions de visibilité, d'abord présentées de façon théorique, sont ensuite appliquées aux problèmes de localisation au travers de deux applications.

Dans un premier temps, nous utilisons ces outils de visibilité afin de localiser une meute de robots dans un environnement intérieur. L'objectif est de déterminer s'il est possible d'éviter la dérive d'une meute de robot en ne considérant qu'une donnée extéroceptive booléenne : la visibilité entre deux robots (deux robots sont considérés visibles si le segment défini par leurs positions respectives n'intersecte pas d'obstacle dans l'environnement). Ce qui revient à considérer un problème de suivi de posture multi-robots en ne s'autorisant qu'une donnée booléenne : la visibilité entre deux robots (deux robots se voient ou non).

Ce problème est mis sous la forme d'un problème de satisfaction de contraintes et est résolu à l'aide des contracteurs de visibilité (et donc de l'analyse par intervalles). Différentes simulations permettent de valider l'efficacité de la méthode : il est possible de maintenir la précision sur la connaissance de la posture des robots d'une meute en ne manipulant qu'une donnée extéroceptive booléenne. À notre connaissance il n'existe aucune autre méthode permettant le traitement d'une telle donnée à des fins de localisation.

Finalement nous présentons une deuxième application de ces contracteurs de visibilité. En partant du principe qu'un robot ne peut pas *voir* derrière les murs, nous utilisons les notions de visibilité pour ajouter une nouvelle contrainte au CSP de l'algorithme IAL. La contrainte précise que le robot doit *voir* l'obstacle détecté par le capteur. En d'autres termes, le segment défini par la position du robot et la posi-

tion de la mesure ne doit pas intersecter d'obstacle. En utilisant les contracteurs de visibilité nous sommes capables de traiter une telle contrainte. Des tests expérimentaux permettent de valider l'intérêt de cette nouvelle contrainte et son intégration au sein de l'algorithme IAL.

Au travers de tous ces travaux, nous remarquons que l'analyse par intervalles est un outil intéressant quand on s'intéresse aux problèmes de localisation en robotique mobile. C'est une alternative viable aux outils probabilistes qui mérite d'être considérée et développée.

Suite à ces travaux plusieurs perspectives apparaissent. Pour continuer le travail commencé avec l'algorithme IAL il serait intéressant de revoir l'implémentation de l'algorithme afin d'en améliorer l'efficacité. Il est raisonnable de penser qu'une meilleure implémentation permettra d'avoir des résultats encore plus compétitifs face aux approches probabilistes. Une autre perspective aux travaux ayant aboutis à l'algorithme IAL, serait de s'intéresser cette fois à une méthode de planification de trajectoires ensembliste. L'objectif final sous-jacent étant de développer un robot mobile autonome entièrement ensembliste, de la localisation à la navigation.

Pour ce qui est de la suite des travaux sur la visibilité, nous nous interrogeons sur d'autres applications possibles de cette notion, notamment en robotique mobile. Nous envisageons par exemple de nous intéresser au problème du déménageur de piano en considérant nos contracteurs d'intersection (de visibilité). Il est aussi envisageable de développer une technique de SLAM¹ basée sur les notions de visibilité : l'information de visibilité entre deux robot permettrait par exemple de définir des zones *sans obstacles* dans l'environnement, reprenant ainsi l'idée présentée dans [Jaulin 2011a].

1. Simultaneous Localization And Mapping - Cartographie et localisation simultanées

Productions scientifiques

Voici la liste des productions scientifiques issues des travaux présentés dans ce manuscrit ainsi que des travaux périphériques auxquels j'ai pu participer pendant ces trois ans.

Colloques internationaux

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin,
A Visibility Information for Multi-Robot Localization, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)* - Tokyo, Japon.

Antoine Bautin, Philippe Lucidarme, **Rémy Guyonneau**, Olivier Simonin, Sébastien Lagrange, Nicolas Delanoue, Francois Charpillet,
Cart-O-matic project* : autonomous and collaborative multi-robot localization, exploration and mapping, *Workshop of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)* - Tokyo, Japon.

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin, Mehdi Lhommeau,
Set-Membership Method for Discrete Optimal Control, *10th International Conference on Informatics in Control, Automation and Robotics (ICINCO) (2013)* - Reykjavik, Islande.

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin,
Mobile Robots Pose Tracking : a Set-Membership Approach Using a Visibility Information, *9th International Conference on Informatics in Control, Automation and Robotics (ICINCO) (2012)* - Rome, Italie.

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin, Philippe Lucidarme,
Interval Analysis for Global Localization Problem using Range Sensors, *Small Workshop on Interval Methods (SWIM) (2011)* - Bourges, France.

Colloques nationaux

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin, Philippe Lucidarme, Mobile Robot Localization : A Set-Membership Approach, *Journées des Jeunes Chercheurs en Robotique (JJCR) (2013)* - Rennes, France.

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin, Méthode ensembliste pour l'estimation d'état d'un système non linéaire particulier, *Journées Scientifiques Robotique et Automatique (JRA) (2012)* - Nantes, France.

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin, Philippe Lucidarme, The kidnapping Problem of Mobile Robots : A Set Membership Approach, *7th National Conference on Control Architectures of Robots (CAR) (2012)* - Nancy, France.

En attente de réponse

Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin, Philippe Lucidarme, Guaranteed Interval Analysis Localization for Mobile Robots, *Advanced Robotics*, Taylor & Francis (Révision)

Introduction à l'analyse par intervalles

Sommaire

1	Présentation générale	153
1.1	Manipulation d'ensembles	154
1.2	Intervalles	157
1.3	Arithmétiques des intervalles	164
2	Deux outils ensemblistes.	173
2.1	Inversion ensembliste	173
2.2	Problèmes de satisfaction de contraintes	176

Cette annexe présente différentes notions de l'analyse par intervalles. Elle contient toute la théorie ensembliste nécessaire à la compréhension des approches présentées dans ce manuscrit. La Section 1 correspond à une présentation générale de l'analyse par intervalles tandis que la Section 2 présente deux outils ensemblistes utilisés dans ce manuscrit. Cette annexe a été écrite en se basant sur [Jaulin 2001b, Lagrange 2005].

1 Présentation générale

Cette section fait une présentation générale de l'analyse par intervalle. Les intervalles étant des ensembles particuliers, il est nécessaire d'introduire en premier lieu la manipulation d'ensembles, avant de s'intéresser par la suite à la manipulation d'intervalles.

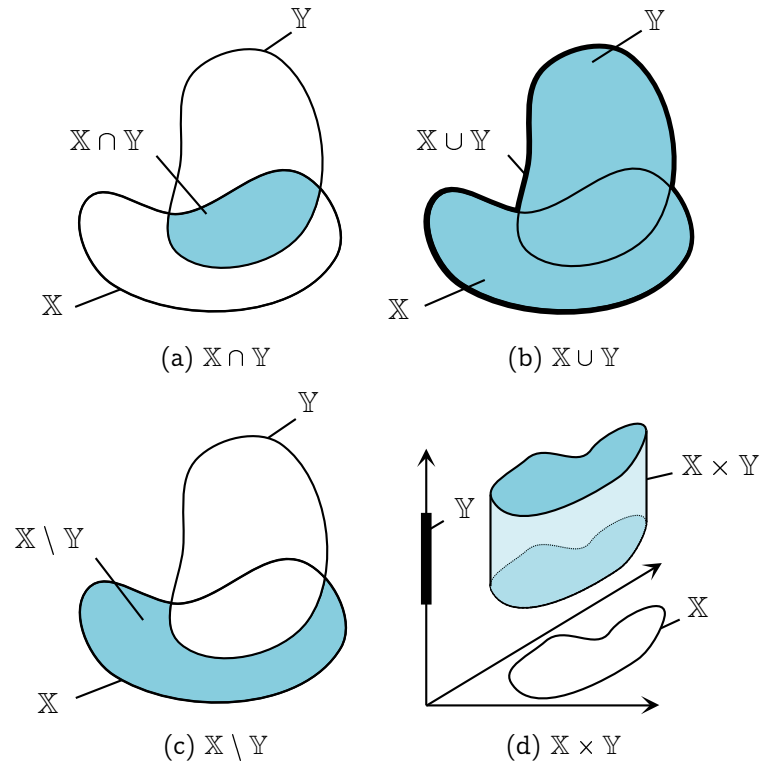


Figure A.1 – Exemple d'opérations sur des ensembles.

Ensemble

Un ensemble correspond à une collection d'éléments. Par exemple l'alphabet correspond à l'ensemble des lettres (c'est un ensemble discret car il est possible de dénombrer le nombre d'éléments - de lettres - de l'ensemble) et \mathbb{R} correspond à l'ensemble de tous les nombres réels (c'est un ensemble continu car il contient une infinité d'éléments - il existe une infinité de nombres réels).

1.1 Manipulation d'ensembles

1.1.1 Opérations classiques

Ci-suivent quelques opérations classiques sur les ensembles.

Soient deux ensembles \mathbb{X} et \mathbb{Y} . Leur intersection est définie par

$$\mathbb{X} \cap \mathbb{Y} \triangleq \{x \mid x \in \mathbb{X} \text{ et } x \in \mathbb{Y}\}, \quad (\text{A.1})$$

et leur union est définie par

$$\mathbb{X} \cup \mathbb{Y} \triangleq \{x \mid x \in \mathbb{X} \text{ ou } x \in \mathbb{Y}\}. \quad (\text{A.2})$$

\mathbb{X} privé de \mathbb{Y} est définie par

$$\mathbb{X} \setminus \mathbb{Y} \triangleq \{x \mid x \in \mathbb{X} \text{ et } x \notin \mathbb{Y}\}. \quad (\text{A.3})$$

Le produit Cartésien de deux ensembles \mathbb{X} et \mathbb{Y} est défini par

$$\mathbb{X} \times \mathbb{Y} \triangleq \{(x, y) \mid x \in \mathbb{X} \text{ et } y \in \mathbb{Y}\}. \quad (\text{A.4})$$

La Figure A.1 illustre ces opérations.

L'inclusion de \mathbb{X} dans \mathbb{Y} est définie par

$$\mathbb{X} \subset \mathbb{Y} \Leftrightarrow \forall x \in \mathbb{X}, x \in \mathbb{Y}, \quad (\text{A.5})$$

et l'égalité entre les deux ensemble est définie par

$$\mathbb{X} = \mathbb{Y} \Leftrightarrow \mathbb{X} \subset \mathbb{Y} \text{ et } \mathbb{Y} \subset \mathbb{X}. \quad (\text{A.6})$$

1 .1.2 Manipulation de fonctions sur des ensembles

En considérant maintenant deux ensembles \mathbb{X} et \mathbb{Y} ainsi qu'une fonction $f : \mathbb{X} \rightarrow \mathbb{Y}$. Si $\mathbb{X}_1 \subset \mathbb{X}$, l'*image directe* de \mathbb{X}_1 par f est définie par

$$f(\mathbb{X}_1) \triangleq \{f(x) \mid x \in \mathbb{X}_1\}. \quad (\text{A.7})$$

En supposant que $\mathbb{Y}_1 \subset \mathbb{Y}$ alors l'*image réciproque* de \mathbb{Y}_1 par f est définie par

$$f^{-1}(\mathbb{Y}_1) \triangleq \{x \in \mathbb{X} \mid f(x) \in \mathbb{Y}_1\}. \quad (\text{A.8})$$

La Figure A.2 illustre la notion d'images directe et réciproque.

En notant \emptyset l'ensemble vide, d'après les définitions précédentes il est possible de remarquer que

$$f(\emptyset) = f^{-1}(\emptyset) = \emptyset. \quad (\text{A.9})$$

Avec \mathbb{X}_1 et \mathbb{X}_2 deux sous-ensembles de \mathbb{X} , et \mathbb{Y}_1 et \mathbb{Y}_2 deux sous-ensembles de \mathbb{Y} , il est facile de montrer que

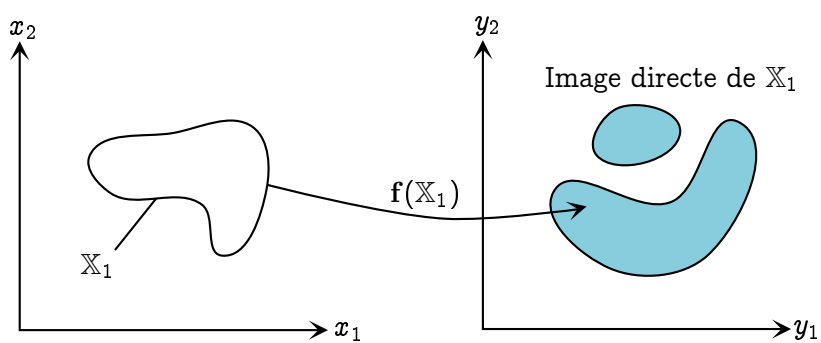
$$f(\mathbb{X}_1 \cap \mathbb{X}_2) \subset f(\mathbb{X}_1) \cap f(\mathbb{X}_2), \quad (\text{A.10})$$

$$f(\mathbb{X}_1 \cup \mathbb{X}_2) = f(\mathbb{X}_1) \cup f(\mathbb{X}_2), \quad (\text{A.11})$$

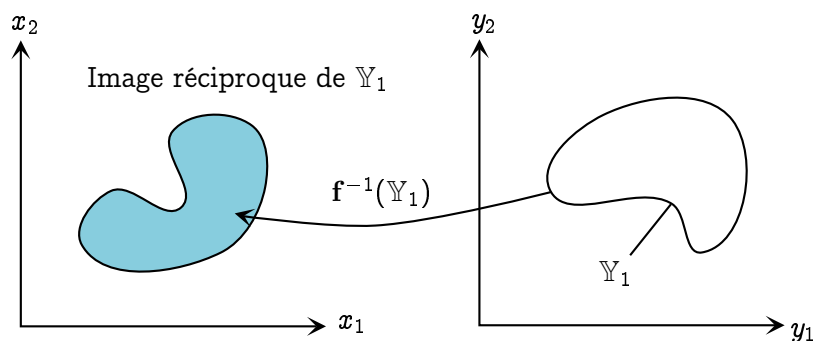
$$f^{-1}(\mathbb{Y}_1 \cap \mathbb{Y}_2) = f^{-1}(\mathbb{Y}_1) \cap f^{-1}(\mathbb{Y}_2), \quad (\text{A.12})$$

$$f^{-1}(\mathbb{Y}_1 \cup \mathbb{Y}_2) = f^{-1}(\mathbb{Y}_1) \cup f^{-1}(\mathbb{Y}_2), \quad (\text{A.13})$$

$$f(f^{-1}(\mathbb{Y})) \subset \mathbb{Y}. \quad (\text{A.14})$$



(a) Exemple d'image directe



(b) Exemple d'image réciproque

Figure A.2 – Exemple d'images directe et réciproque.

1 .1.3 Intersection relaxée

L'intersection relaxée, aussi appelée *q-intersection* correspond à une intersection particulière pour laquelle les points solutions peuvent ne pas intersecter certains des ensembles considérés. Soient m ensembles connexes $\mathbb{X}_i \subset \mathbb{R}^n$, $i = 1, \dots, m$. La q -intersection, notée $\bigcap_{i=1, \dots, m}^{\{q\}} \mathbb{X}_i$, correspond à l'ensemble des points $x \in \mathbb{R}^n$ qui intersectent au moins $m - q$ ensembles \mathbb{X}_i . La Figure A.3 présente des exemples de q -intersections.

Remarque A.1

$$\bigcap_{i=1, \dots, m}^{\{0\}} \mathbb{X}_i \equiv \bigcap_{i=1, \dots, m} \mathbb{X}_i, \quad (\text{A.15})$$

$$\bigcap_{i=1, \dots, m}^{\{m\}} \mathbb{X}_i \equiv \bigcup_{i=1, \dots, m} \mathbb{X}_i. \quad (\text{A.16})$$

1 .2 Intervalles

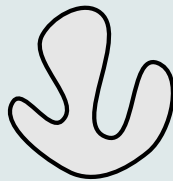
Maintenant que la manipulation d'ensembles a été introduite, il est possible de présenter la manipulation d'intervalles.

1 .2.1 Définitions générales

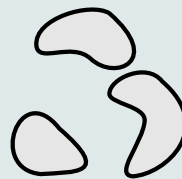
Un intervalle, noté $[x]$, correspond à un sous ensemble connexe fermé de \mathbb{R} . L'ensemble des intervalles de \mathbb{R} est noté $\mathbb{I}\mathbb{R}$.

Ensemble connexe

Un ensemble connexe correspond à un ensemble *d'un seul tenant*.



Ensemble connexe.



Ensemble non-connexe.

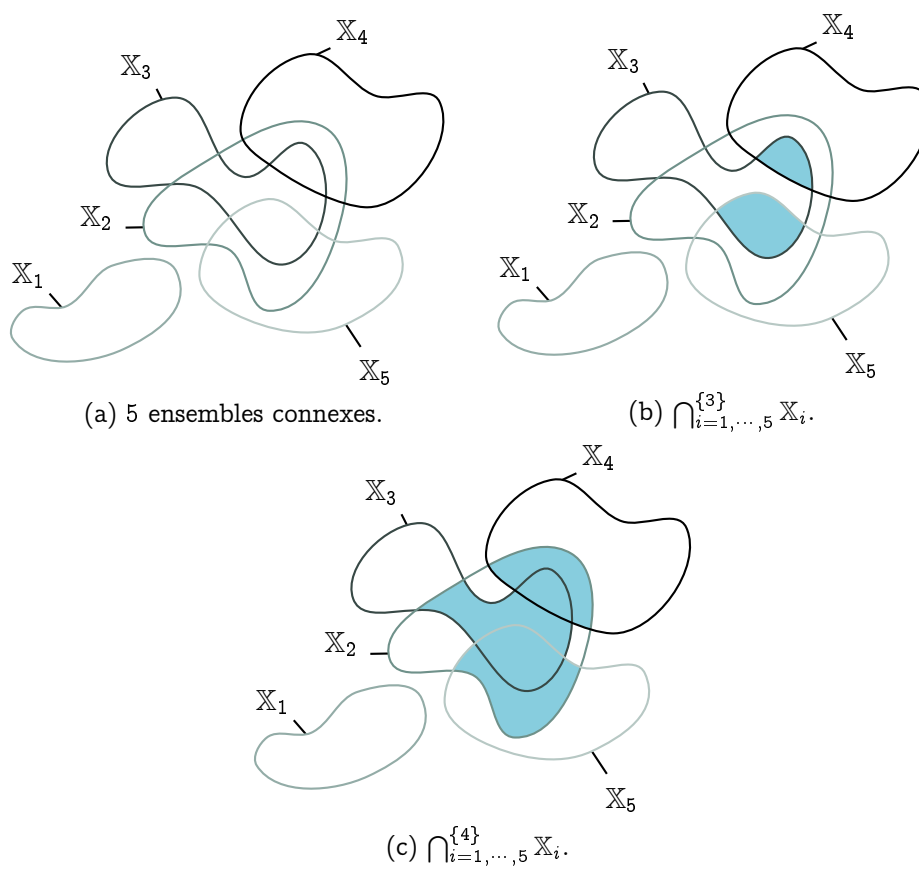
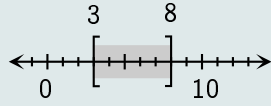


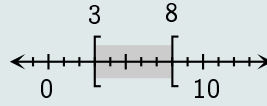
Figure A.3 – Exemple d'intersections relaxées. Dans cet exemple $\bigcap_{i=1, \dots, 5}^{\{0\}} X_i = \bigcap_{i=1, \dots, 5}^{\{1\}} X_i = \bigcap_{i=1, \dots, 5}^{\{2\}} X_i = \emptyset$.

Ensemble fermé

Un ensemble est fermé si tous les points limites de cet ensemble lui appartiennent.



Ensemble fermé, 3 et 8 appartiennent à l'ensemble.



Ensemble non-fermé car 8 n'appartient pas à l'ensemble considéré.

Remarque A.2 On considérera l'ensemble vide \emptyset comme étant un intervalle, \emptyset représentant l'absence de solution à un problème donné.

Remarque A.3 Les bornes infinies ($-\infty$ et $+\infty$) seront considérées comme étant fermées. L'ensemble $[x] = [3, +\infty)$, pourra alors être défini comme un intervalle.

Un intervalle est caractérisé par une borne inférieure et une borne supérieure. La borne inférieure, notée \underline{x} , d'un intervalle $[x]$ est définie par

$$\underline{x} \triangleq \inf\{x \mid x \in [x]\}. \quad (\text{A.17})$$

La borne supérieure, notées \bar{x} , d'un intervalle $[x]$ est définie par

$$\bar{x} \triangleq \sup\{x \mid x \in [x]\}. \quad (\text{A.18})$$

La taille d'un intervalle $[x]$ non vide, notée $w([x])$, est définie par

$$w([x]) \triangleq \bar{x} - \underline{x}. \quad (\text{A.19})$$

On peut noter que $w([3, +\infty]) = +\infty$.

Le centre d'un intervalle $[x]$ non vide est défini par

$$\text{mid}([x]) \triangleq \frac{\underline{x} + \bar{x}}{2}. \quad (\text{A.20})$$

1 .2.2 Opérations ensemblistes sur les intervalles

Les opérations ensemblistes présentées précédemment (Section 1 .1.1) s'appliquent naturellement aux intervalles.

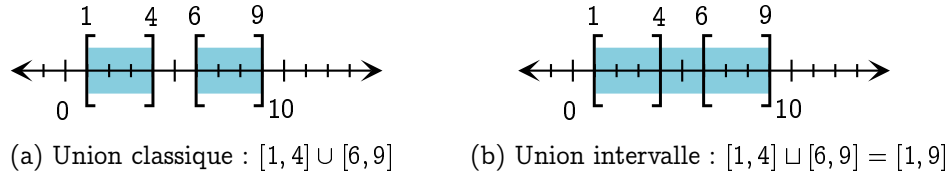


Figure A.4 – Exemple d'union intervalle.

L'intersection de deux intervalles $[x]$ et $[y]$ est définie par

$$[x] \cap [y] \triangleq \{z \in \mathbb{R} \mid z \in [x] \text{ et } z \in [y]\}. \quad (\text{A.21})$$

Il est possible de remarquer que l'intersection de deux intervalles sera toujours un intervalle (au pire, l'ensemble vide) :

$$[x] \cap [y] = \begin{cases} \emptyset & \text{si } \underline{x} > \bar{y} \text{ ou } \bar{x} < \underline{y}, \\ [\max(\underline{x}, \underline{y}), \min(\bar{x}, \bar{y})] & \text{sinon.} \end{cases} \quad (\text{A.22})$$

En considérant l'union ensembliste telle que présentée précédemment, l'union de deux intervalles ne correspond pas obligatoirement à un intervalle. En effet l'union classique de deux intervalles disjoints n'est pas connexe (Figure A.4). Pour pouvoir définir l'union intervalle, il est nécessaire d'introduire la notion d'*enveloppe intervalle*. L'enveloppe intervalle d'un sous-ensemble \mathbb{X} de \mathbb{R} , notée $[\mathbb{X}]$ correspond au plus petit intervalle contenant \mathbb{X} . Par exemple l'enveloppe intervalle de $[1, 4] \cup [6, 9]$ correspond à l'intervalle $[1, 9]$. L'union intervalle se définit alors comme l'enveloppe intervalle de l'union classique de deux intervalles (Figure A.4) :

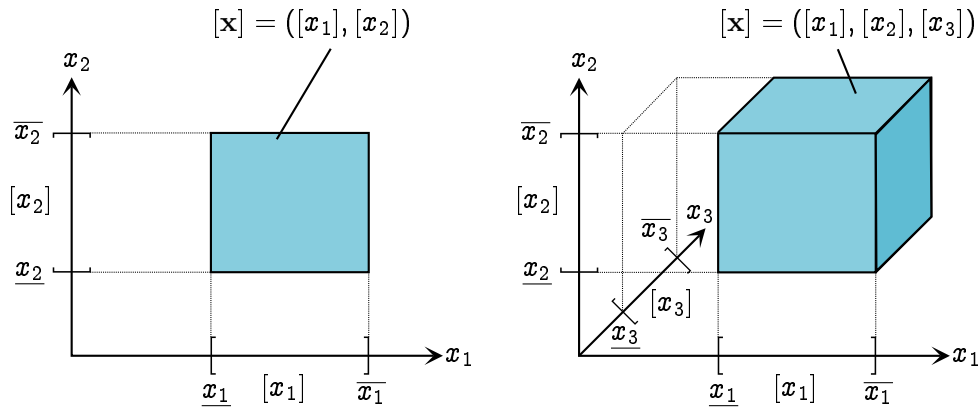
$$[x] \sqcup [y] \triangleq [x \cup y]. \quad (\text{A.23})$$

On remarquera que l'union intervalle de deux intervalles correspond bien à un intervalle.

1.2.3 Les vecteurs d'intervalles

Un *vecteur d'intervalles* (aussi appelé *boîte* ou *pavé*) est un sous-ensemble de \mathbb{R}^n correspondant au produit Cartésien de n intervalles (un vecteur composé de n intervalles) :

$$[\mathbf{x}] = [x_1] \times \cdots \times [x_n] = ([x_1], \cdots, [x_n]). \quad (\text{A.24})$$



(a) Un vecteur d'intervalles à deux dimensions $[x] = ([x_1], [x_2])$.

(b) Un vecteur d'intervalles à trois dimensions $[x] = ([x_1], [x_2], [x_3])$.

Figure A.5 – Exemples de vecteurs d'intervalles

Produit Cartésien

Le produit Cartésien de deux ensembles \mathbb{X} et \mathbb{Y} , noté $\mathbb{X} \times \mathbb{Y}$, correspond à l'ensemble des points ayant une composante dans \mathbb{X} et une composante dans \mathbb{Y} . On a par exemple \mathbb{R} qui correspond à l'ensemble des réels, $\mathbb{R}^2 = \mathbb{R} \times \mathbb{R}$ qui correspond au plan à deux dimensions et $\mathbb{R}^3 = \mathbb{R} \times \mathbb{R} \times \mathbb{R}$ qui correspond à l'espace à trois dimensions.

L'ensemble des vecteurs d'intervalles est noté \mathbb{IR}^n . La Figure A.5 présente des exemples de vecteurs d'intervalles.

Par la suite les termes boîtes, pavés et vecteurs d'intervalles seront utilisés.

La taille d'un pavé $[x]$ non vide, noté $w([x])$, est définie par

$$w([x]) \triangleq \max(w([x_i])), \quad i = 1, \dots, n. \quad (\text{A.25})$$

Le centre d'un pavé non vide $[x]$ est défini par

$$\text{mid}([x]) \triangleq (\text{mid}([x_1]), \dots, \text{mid}([x_n]))^T. \quad (\text{A.26})$$

1.2.4 Opérations ensemblistes sur les vecteurs d'intervalles

Il est possible d'étendre les opérations présentées précédemment pour les intervalles (Section 1.2.2), aux vecteurs d'intervalles.

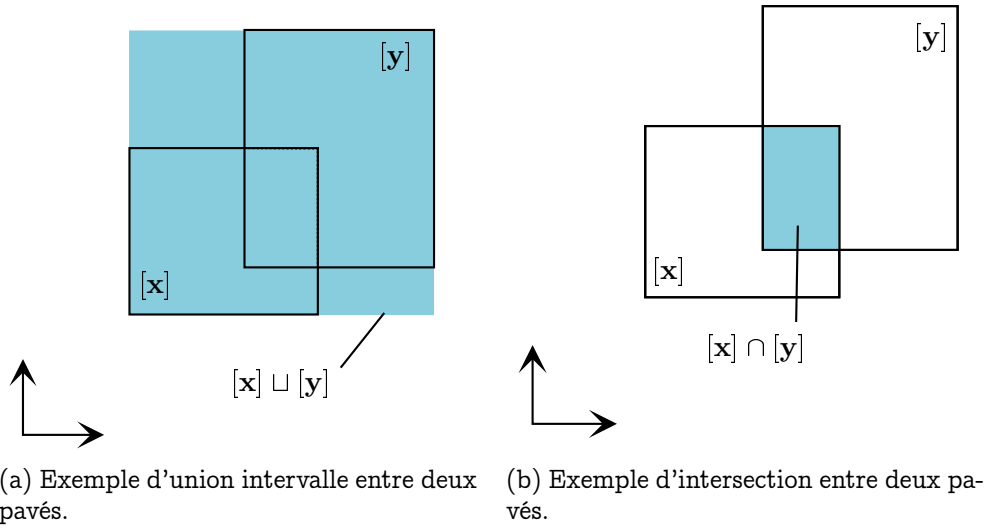


Figure A.6 – Exemples d'union intervalle et d'intersection entre deux pavés.

L'intersection de deux pavés $[x]$ et $[y]$ de $\mathbb{I}\mathbb{R}^n$ est définie par

$$[x] \cap [y] \triangleq ([x_1] \cap [y_1]) \times \cdots \times ([x_n] \cap [y_n]), \quad (\text{A.27})$$

et l'union intervalle des deux pavés est définie par

$$[x] \cup [y] \triangleq ([x_1] \cup [y_1]) \times \cdots \times ([x_n] \cup [y_n]). \quad (\text{A.28})$$

On peut noter que le résultat de ces deux opérations est bien un vecteur d'intervalles (Figure A.6).

L'inclusion d'un pavé $[x] \in \mathbb{I}\mathbb{R}^n$ dans un pavé $[y] \in \mathbb{I}\mathbb{R}^n$ est définie par

$$[x] \subset [y] \Leftrightarrow [x_1] \subset [y_1] \text{ et } \cdots \text{ et } [x_n] \subset [y_n]. \quad (\text{A.29})$$

L'opération de bisection consiste à diviser une boîte $[x]$ en deux boîtes symétriques $[x_1]$ et $[x_2]$. Cette division se fait généralement par rapport au plus grand côté de $[x]$. Considérons un pavé

$$[x] = [x_1, \bar{x}_1] \times \cdots \times [x_n, \bar{x}_n]. \quad (\text{A.30})$$

On note i_b l'indice de la première plus grande dimension de $[x]$:

$$i_b = \min(\{i \mid w([x_i]) = w([x])\}). \quad (\text{A.31})$$

Le résultat de la bisection de $[x]$ est alors

$$\begin{aligned} [x_1] &= [x_1, \bar{x}_1] \times \cdots \times \left[x_{i_b}, \frac{(x_{i_b} + \bar{x}_{i_b})}{2} \right] \times \cdots \times [x_n, \bar{x}_n] \\ [x_2] &= [x_1, \bar{x}_1] \times \cdots \times \left[\frac{(x_{i_b} + \bar{x}_{i_b})}{2}, x_{i_b} \right] \times \cdots \times [x_n, \bar{x}_n] \end{aligned} \quad (\text{A.32})$$

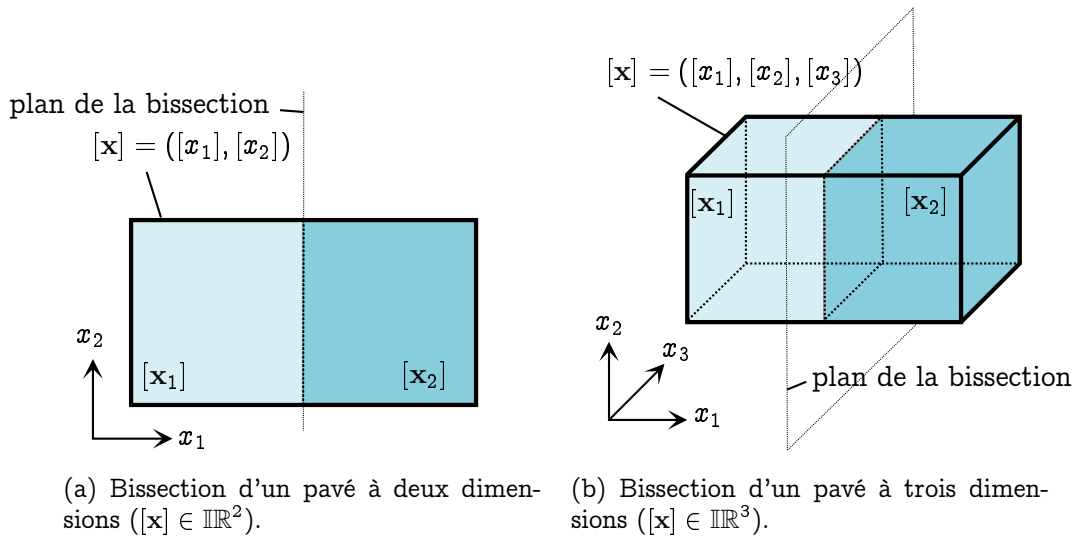


Figure A.7 – Exemples de bisections

La bissection de $[\mathbf{x}] = [0, 10] \times [3, 7]$ conduit par exemple à deux boîtes $[\mathbf{x}_1] = [0, 5] \times [3, 7]$ et $[\mathbf{x}_2] = [5, 10] \times [3, 7]$. On peut noter que $w([\mathbf{x}_1]) = w([\mathbf{x}_2])$ et $[\mathbf{x}_1] \cup [\mathbf{x}_2] = [\mathbf{x}_1] \sqcup [\mathbf{x}_2] = [\mathbf{x}]$. La Figure A.7 présente deux exemples de bissection.

Par la suite on notera $([\mathbf{x}_1], [\mathbf{x}_2]) = \text{Bissect}([\mathbf{x}])$, la fonction qui bissecte $[\mathbf{x}]$ en deux vecteurs d'intervalles $[\mathbf{x}_1]$ et $[\mathbf{x}_2]$.

1.2.5 Notion de sous-pavages

Un sous-pavage d'un ensemble $\mathbb{X} \subset \mathbb{R}^n$ correspond à un ensemble de pavés $\{[\mathbf{x}_i]\}_{i \in \mathbb{N}}$ de \mathbb{R}^n vérifiant

$$\begin{cases} \bigcup_{i \in \mathbb{N}} [\mathbf{x}_i] = \mathbb{X}, \\ \forall i, j \in \mathbb{N}, i \neq j, w([\mathbf{x}_i] \cap [\mathbf{x}_j]) = 0. \end{cases} \quad (\text{A.33})$$

La Figure A.8 présente un exemple de sous-pavage

Sur le même principe, à tout ensemble $\mathbb{X} \subset \mathbb{R}^n$, on peut associer une sur-approximation et une sous-approximation. Une sous-approximation d'un ensemble $\mathbb{X} \subset \mathbb{R}^n$, notées \mathbb{X}^- , correspond à un ensemble de boîtes $\{[\mathbf{x}_i]\}_{i \in \mathbb{N}}$ de \mathbb{R}^n vérifiant

$$\begin{cases} \bigcup_{i \in \mathbb{N}} [\mathbf{x}_i] \subset \mathbb{X}, \\ \forall i, j \in \mathbb{N}, i \neq j, w([\mathbf{x}_i] \cap [\mathbf{x}_j]) = 0. \end{cases} \quad (\text{A.34})$$

Une sur-approximation d'un ensemble $\mathbb{X} \subset \mathbb{R}^n$, notées \mathbb{X}^+ , correspond à un ensemble de boîtes $\{[\mathbf{x}_i]\}_{i \in \mathbb{N}}$ de \mathbb{R}^n vérifiant

$$\begin{cases} \bigcup_{i \in \mathbb{N}} [\mathbf{x}_i] \supset \mathbb{X}, \\ \forall i, j \in \mathbb{N}, i \neq j, w([\mathbf{x}_i] \cap [\mathbf{x}_j]) = 0. \end{cases} \quad (\text{A.35})$$

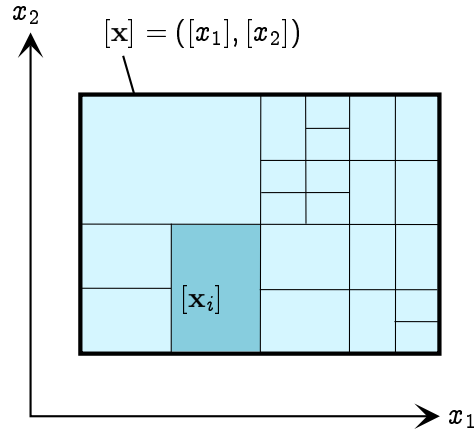


Figure A.8 – Exemple de sous-pavage d'une boîte $[x]$. $[x_i]$ correspond à un sous-pavé du sous-pavage de $[x]$

On a donc

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+ \quad (\text{A.36})$$

La Figure A.9 présente un exemple de sous-approximation et sur-approximation.

1.3 Arithmétiques des intervalles

L'intérêt de la manipulation d'intervalles (et de vecteur d'intervalles) se trouve notamment dans la possibilité d'étendre relativement facilement toutes les relations, opérations et fonctions des réels sur les intervalles.

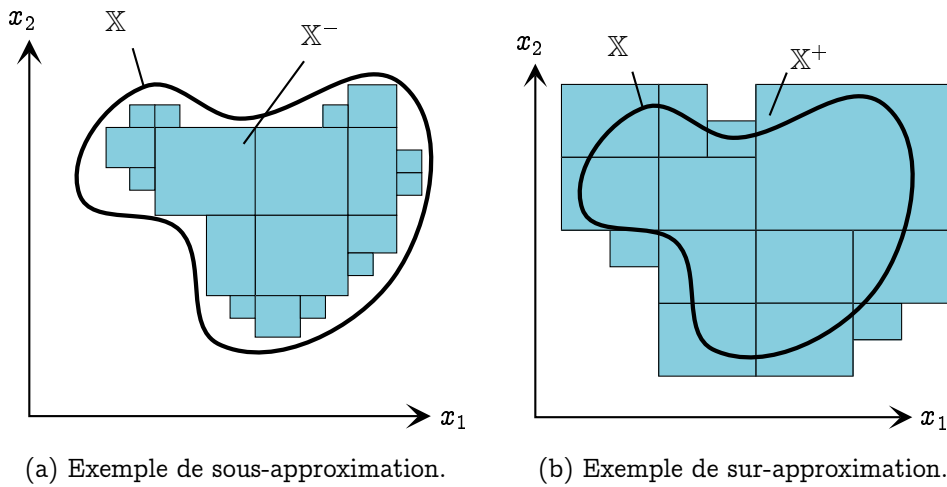
1.3.1 Relations

L'égalité entre deux intervalles $[x]$ et $[y]$ est définie par

$$[x] = [y] \Leftrightarrow \underline{x} = \underline{y} \text{ et } \bar{x} = \bar{y}. \quad (\text{A.37})$$

Cette égalité peut se généraliser aux pavés de \mathbb{IR}^n . Soient $[x]$ et $[y]$ deux vecteurs d'intervalles

$$[x] = [y] \Leftrightarrow \forall i = 1, \dots, n, [x_i] = [y_i]. \quad (\text{A.38})$$

Figure A.9 – Sur et sous-approximations de \mathbb{X} . On retrouve bien $\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+$.

Les relations d'ordres sont définies par

$$[x] < [y] \Leftrightarrow \bar{x} < \underline{y}, \quad (\text{A.39})$$

$$[x] \leq [y] \Leftrightarrow \bar{x} \leq \underline{y}, \quad (\text{A.40})$$

$$[x] > [y] \Leftrightarrow \underline{x} > \bar{y}, \quad (\text{A.41})$$

$$[x] \geq [y] \Leftrightarrow \underline{x} \geq \bar{y}. \quad (\text{A.42})$$

Remarque A.4 Les relations \leq et \geq sont partielles. Si deux intervalles $[x]$ et $[y]$ ont une intersection non nulle, alors $[x] \not\leq [y]$ et $[x] \not\geq [y]$.

Relation totale / relation partielle

Une relation \diamond est totale sur un ensemble \mathbb{E} si tous les éléments de \mathbb{E} sont comparables avec tous les éléments de \mathbb{E} , suivant \diamond . Par exemple, les relations d'ordres (\leq et \geq) sur l'ensemble des lettres de l'alphabet (ordonnées par l'ordre alphabétique) sont totales. En prenant deux lettres au hasard, il sera toujours possible de définir une relation d'ordre entre ces deux lettres : $a \leq d, c \geq a, \dots$.

Une relation est partielle sur un ensemble \mathbb{E} quand elle n'est pas totale (quand il existe des éléments de \mathbb{E} qui ne sont pas comparables entre eux suivant cette relation).

1.3.2 Opérations classiques

Les opérations classiques de l'arithmétique des réels peuvent être étendues aux intervalles. Soient $\star = \{+, -, \times, /\}$ et deux intervalles $[x]$ et $[y]$ de \mathbb{IR} ,

$$[x] \star [y] = [\{x \star y \mid x \in [x], y \in [y]\}]. \quad (\text{A.43})$$

Le résultat d'une opération classique entre deux intervalles correspond au plus petit intervalle contenant toutes les valeurs possibles de cette opération sur tous les points de chacun des deux intervalles.

On a donc, considérant deux intervalles $[x] = [\underline{x}, \bar{x}]$ et $[y] = [\underline{y}, \bar{y}]$

$$[x] + [y] = [\underline{x} + \underline{y}, \bar{x} + \bar{y}], \quad (\text{A.44})$$

$$[x] - [y] = [\underline{x} - \bar{y}, \bar{x} - \underline{y}], \quad (\text{A.45})$$

$$[x] \times [y] = [\min(\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y}), \max(\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y})], \quad (\text{A.46})$$

$$\frac{1}{[y]} = \begin{cases} \left[\min\left(\frac{1}{\underline{y}}, \frac{1}{\bar{y}}\right), \max\left(\frac{1}{\underline{y}}, \frac{1}{\bar{y}}\right) \right] & \text{si } 0 \notin [y] \\ [-\infty, +\infty] & \text{sinon} \end{cases}, \quad (\text{A.47})$$

$$\frac{[x]}{[y]} = [x] \times \frac{1}{[y]}. \quad (\text{A.48})$$

Exemples :

$$[3, 5] \times [4, 8] + [-1, 3] = [12, 40] + [-1, 3] = [11, 43],$$

$$[7, 9]/[-3, 5] = [-\infty, +\infty],$$

$$[4, 8]/[2, 4] = [4, 8] \times \left[\frac{1}{4}, \frac{1}{2}\right] = [1, 4].$$

Il est cependant important de noter que les opérations présentées ci-dessus ne possèdent pas les mêmes propriétés algébriques que les opérations classiques sur les réels. Par exemple :

- La soustraction n'est pas la réciproque de l'addition : $[x] - [x] + [x]$ n'est pas équivalent à $[x]$. Par exemple

$$[2, 5] - [2, 5] + [2, 5] = [-3, 3] + [2, 5] = [-1, 8] \neq [2, 5].$$

On peut remarquer que $[x] - [x] = [0]$ dans le seul cas particulier où $[x]$ est un singleton ($\underline{x} = \bar{x}$).

- La division n'est pas non plus la réciproque de la multiplication :

$$\frac{[2, 5]}{[2, 5]} \times [2, 5] = \left[\frac{2}{5}, \frac{5}{2}\right] \times [2, 5] = \left[\frac{4}{5}, \frac{25}{2}\right] \neq [2, 5].$$

Ici aussi $[x]/[x] = [1]$ dans le seul cas particulier où $[x]$ est un singleton.

- La multiplication d'un intervalle par lui même n'est pas équivalent à l'élever au carré :

$$\begin{aligned}[-1, 3] \times [-1, 3] &= [-3, 9], \\ [-1, 3]^2 &= [0, 9].\end{aligned}$$

- La multiplication n'est pas distributive sur l'addition : $[x] \times ([y_1] + [y_2])$ n'est pas équivalent à $[x] \times [y_1] + [x] \times [y_2]$. Par exemple

$$\begin{aligned}[2, 5] \times ([-1, 2] + [3, 4]) &= [2, 5] \times [2, 6] = [4, 30], \\ [2, 5] \times [-1, 2] + [2, 5] \times [3, 4] &= [-5, 10] + [6, 20] = [1, 30].\end{aligned}$$

Singleton

Un ensemble \mathbb{E} est un singleton s'il ne contient qu'un seul élément. Par exemple $\mathbb{E} = \{3\}$ est un singleton (l'ensemble \mathbb{E} ne contient qu'une valeur : 3) alors que $\mathbb{E}' = \{3, 4\}$ n'en est pas un.

Remarque A.5 *Les points précédents peuvent s'expliquer par le fait que quand un intervalle apparaît plusieurs fois au sein d'une même opération, on "oublie" que l'intervalle correspond à la même variable. C'est flagrant sur l'exemple suivant : soit un intervalle $[x] = [-1, 2]$, $[x] \times [x] = [-1, 2] \times [-1, 2] = [-2, 4]$, ici on a bien "oublié" qu'on multipliait une variable par elle même (d'où la présence de valeurs négatives). Il y a donc un "pessimisme" sur la solution en présence plusieurs occurrences de la même variable. On peut cependant noter que*

$$\begin{aligned}[x] \times [x] &\supset [x]^2, \\ [x] - [x] + [x] &\supset [x], \\ [x] - [x] &\supset \{0\}, \\ [x]/[x] &\supset \{1\}, \\ [x] \times [y_1] + [x] \times [y_2] &\supset [x] \times ([y_1] + [y_2]).\end{aligned}$$

On a donc un pessimisme mais en aucun cas une perte de solution.

Il est possible d'étendre ces opérations aux vecteurs d'intervalles. Pour cela il suffit de reprendre les définitions sur les réels et de remplacer les variables et les opérations par leurs équivalents intervalles.

1 .3.3 Fonctions élémentaires

Tout comme nous l'avons fait pour les opérations classiques, les fonctions élémentaires des réels (cos, sin, exp...) peuvent être étendues aux intervalles.

Soient f une fonction élémentaire définie sur \mathbb{R} à valeurs dans \mathbb{R} et $[x]$ un intervalle de \mathbb{R} ,

$$f([x]) \triangleq \{f(x) \mid x \in [x]\}. \quad (\text{A.49})$$

L'évaluation d'un intervalle par une fonction élémentaire revient donc à calculer le plus petit intervalle contenant l'évaluation de tous les points de l'intervalle par la fonction.

Par exemple

$$\begin{aligned} \cos\left(\left[\frac{\pi}{6}, \pi\right]\right) &= [0, 1], \\ ([-3, 2])^2 &= [0, 9], \\ \sqrt{[4, 9]} &= [2, 3], \\ \text{abs}([-5, 3]) &= [0, 5]. \end{aligned}$$

Maintenant que nous savons utiliser les opérateurs classiques et les fonctions élémentaires sur les intervalles, il devient possible d'évaluer des expressions plus complexes. Ci-suivent deux exemples d'expressions mélangeant opérateurs et fonctions, évaluées pour des variables intervalles.

- Considérons l'expression suivante $y = 3x^3 - x^2 + 7$. Nous voulons évaluer y pour $x \in [-1, 3]$:

$$\begin{aligned} [y] &= 3[-1, 3]^3 - [-1, 3]^2 + 7, \\ &= 3[-1, 27] - [0, 9] + 7, \\ &= [-3, 81] - [-7, 2] = [-3, 81] + [-2, 7], \\ [y] &= [-5, 88]. \end{aligned}$$

- Considérons l'expression suivante $z = \cos(x)^2 + 2x \sin(y) + \sqrt{y}$. Nous voulons évaluer z pour $x \in \left[\frac{\pi}{3}, \frac{\pi}{2}\right]$ et $y \in [0, \pi]$:

$$\begin{aligned} [z] &= \cos\left(\left[\frac{\pi}{3}, \frac{\pi}{2}\right]^2\right) + 2\left[\frac{\pi}{3}, \frac{\pi}{2}\right] \sin([0, \pi]) + \sqrt{[0, \pi]}, \\ &= \left[0, \frac{1}{2}\right]^2 + \left[\frac{2\pi}{3}, \pi\right] \times [0, 1] + [0, \sqrt{\pi}], \\ &= \left[0, \frac{1}{4}\right] + [0, \pi] + [0, \sqrt{\pi}], \\ [z] &= \left[0, \frac{1}{4} + \pi + \sqrt{\pi}\right]. \end{aligned}$$

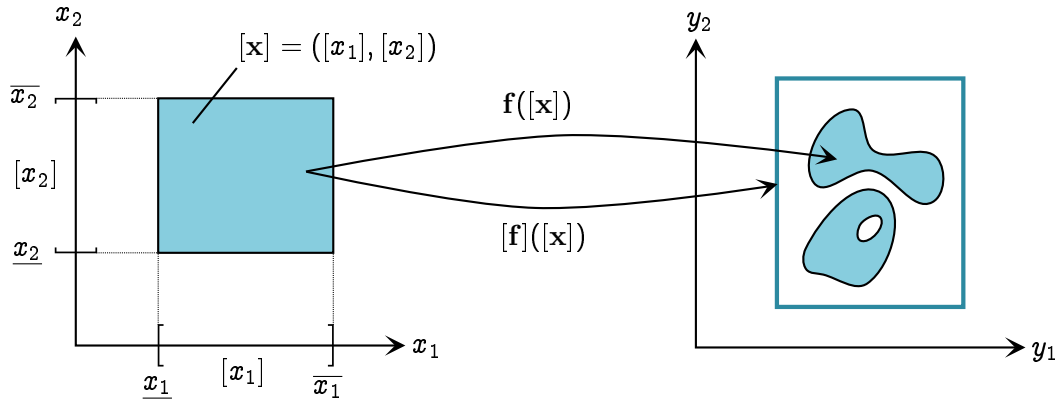


Figure A.10 – Exemple de fonction d'inclusion. On retrouve bien $f([x]) \subset [f]([x])$.

1 .3.4 Fonctions d'inclusions

Considérant une fonction f définie sur \mathbb{R}^n et à valeurs dans \mathbb{R}^m , la fonction ensembliste $[f] : \mathbb{I}\mathbb{R}^n \rightarrow \mathbb{I}\mathbb{R}^m$ est une *fonction d'inclusion* de f si

$$\forall [x] \in \mathbb{I}\mathbb{R}^n, f([x]) \subset [f]([x]). \quad (\text{A.50})$$

La Figure A.10 illustre le principe d'une fonction d'inclusion.

On notera que pour toutes fonctions f il est possible de trouver une fonction d'inclusion $[f]$. La plus évidente consiste à remplacer toutes les opérations de f par les opérations intervalles correspondantes. La fonction d'inclusion ainsi obtenue est appelée *fonction d'inclusion naturelle*. Ci-suivent deux exemples de fonctions d'inclusions naturelles.

- Soit une fonction $f(x) = x^2 + 4x + 4$, il est possible de construire la fonction d'inclusion suivante :

$$[f] : \begin{cases} \mathbb{I}\mathbb{R} \rightarrow \mathbb{I}\mathbb{R} \\ [x] \rightarrow [x]^2 + 4[x] + 4 \end{cases} .$$

Pour $[x] = [-2, 1]$ on obtient alors

$$\begin{aligned} [f]([-2, 1]) &= [-2, 1]^2 + 4[-2, 1] + 4, \\ &= [0, 4] + [-8, 4] + 4, \\ [f]([-2, 1]) &= [-8, 8] + 4 = [-4, 12]. \end{aligned}$$

On peut remarquer que $f([-2, 1]) = [0, 9] \subset [f]([-2, 1]) = [-4, 12]$ (Figure A.11).

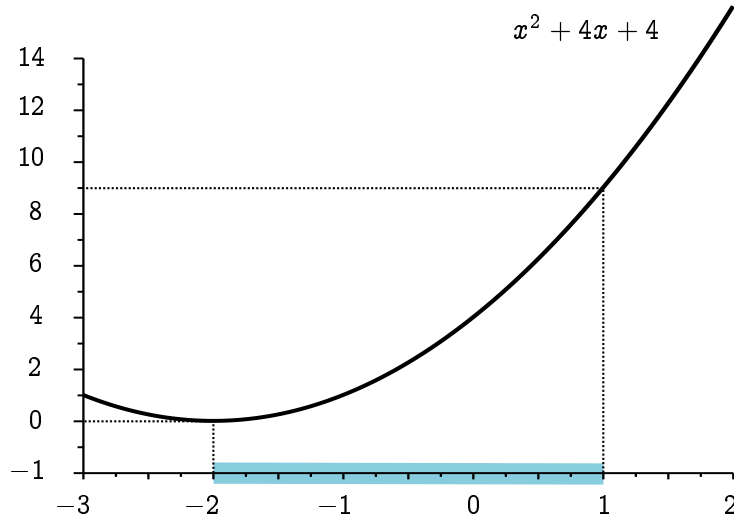


Figure A.11 – Évaluation de $f([-2, 1]) = [0, 9]$, avec $f(x) = x^2 + 4x + 4$.

– Soit une fonction f définie par

$$f : \begin{cases} \mathbb{R}^2 \rightarrow \mathbb{R}^3 \\ (x_1, x_2) \rightarrow (x_1 x_2, x_1 - x_2, x_1 + x_2) \end{cases},$$

il est possible de construire la fonction d'inclusion suivante :

$$[f] : \begin{cases} \mathbb{IR}^2 \rightarrow \mathbb{IR}^3 \\ ([x_1], [x_2]) \rightarrow ([x_1][x_2], [x_1] - [x_2], [x_1] + [x_2]) \end{cases}.$$

Pour $[x_1] = [-2, 1]$ et $[x_2] = [1, 3]$ on obtient alors

$$\begin{aligned} [f]([-2, 1], [1, 3]) &= ([-2, 1] \times [1, 3], [-2, 1] - [1, 3], [-2, 1] + [1, 3]), \\ [f]([-2, 1], [1, 3]) &= ([-6, 3], [-5, -2], [-1, 4]). \end{aligned}$$

Sur le dernier exemple on peut remarquer que construire une fonction d'inclusion $[f]$ pour une fonction $f = (f_1, \dots, f_m)$ revient à construire m fonctions d'inclusions $[f_i]$, $i = 1, \dots, m$. La fonction d'inclusion de f correspond alors à

$$[f] = ([f_1], \dots, [f_m]). \quad (\text{A.51})$$

Bien que la fonction d'inclusion naturelle soit la plus facile à mettre en place (il suffit de remplacer les variables scalaires par des variables intervalles), on remarque qu'elle n'est généralement pas minimale au sens de l'inclusion. En effet le pavé résultant de la fonction d'inclusion naturelle n'est généralement pas le plus petit pavé contenant toutes les solutions. En reprenant l'exemple illustré dans la Figure A.11, on remarque que le plus petit pavé solution correspond à $[0, 9]$ alors que la fonction d'inclusion naturelle retourne $[-4, 12]$.

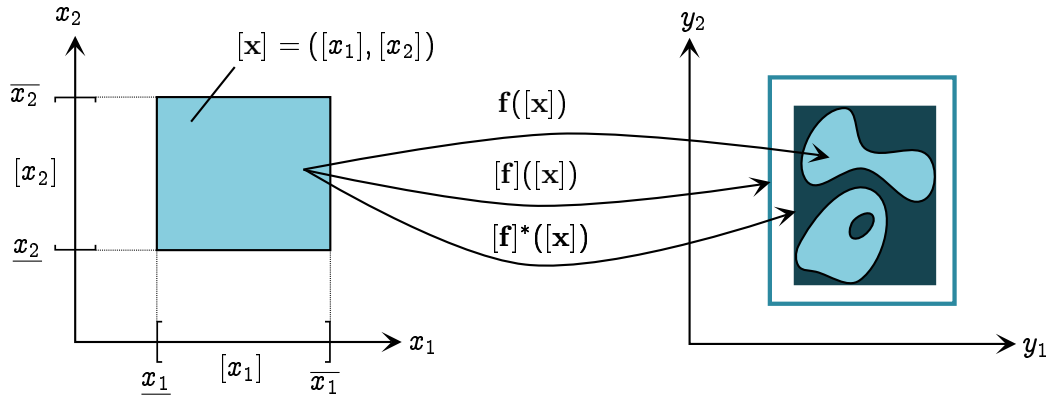


Figure A.12 – Exemple de fonction d'inclusion minimale : $[f]^*([x]) = [f]([x])$.

On définit donc la *fonction d'inclusion minimale* $[f]^*$ d'une fonction f comme étant la fonction d'inclusion vérifiant

$$[f]^* = [f]([x]). \quad (\text{A.52})$$

La Figure A.12 présente un exemple de fonction d'inclusion minimale.

Ci-suit un exemple extrait de [Jaulin 2001b] qui illustre l'efficacité de différentes fonctions d'inclusions pour une même fonction. Considérons les quatre écritures suivantes de la même équations

$$\begin{aligned} f_1(x) &= x(x+1), \\ f_2(x) &= x \times x + x, \\ f_3(x) &= x^2 + x, \\ f_4(x) &= \left(x + \frac{1}{2}\right)^2 - \frac{1}{4}. \end{aligned}$$

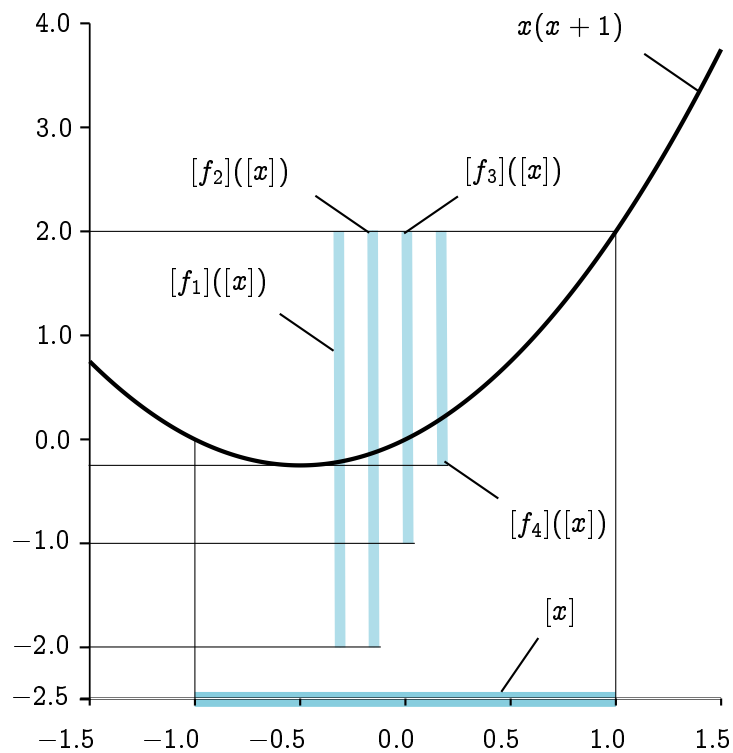


Figure A.13 – Résultat des différentes fonctions d'inclusions.

En considérant leurs fonctions d'inclusions, avec $[x] = [-1, 1]$ on obtient

$$[f_1(x)] = [x]([x] + 1) = [-1, 1]([-1, 1] + 1) = [-1, 1] \times [0, 2],$$

$$[f_1(x)] = [-2, 2].$$

$$[f_2(x)] = [x] \times [x] + [x] = [-1, 1] \times [-1, 1] + [-1, 1] = [-1, 1] + [-1, 1],$$

$$[f_2(x)] = [-2, 2].$$

$$[f_3(x)] = [x]^2 + [x] = [-1, 1]^2 + [-1, 1] = [0, 1] + [-1, 1],$$

$$[f_3(x)] = [-1, 2].$$

$$[f_4(x)] = \left([x] + \frac{1}{2}\right)^2 - \frac{1}{4} = \left([-1, 1] + \frac{1}{2}\right)^2 - \frac{1}{4} = \left[-\frac{1}{2}, \frac{3}{2}\right]^2 - \frac{1}{4} = \left[0, \frac{9}{4}\right] - \frac{1}{4},$$

$$[f_4(x)] = \left[-\frac{1}{4}, 2\right].$$

La Figure A.13 présente les résultats des différentes fonctions d'inclusion. On remarque que la précision de la fonction d'inclusion dépend de sa forme. Comme indiqué précédemment, la présence de plusieurs occurrences d'une même variable augmente le pessimisme du résultat obtenu : la fonction d'inclusion $[f_4]$, qui est la seule à n'avoir qu'une seule occurrence de la variable $[x]$, s'avère être la plus précise. Dans cet exemple, $[f_4]$ correspond à la fonction d'inclusion minimale de f .

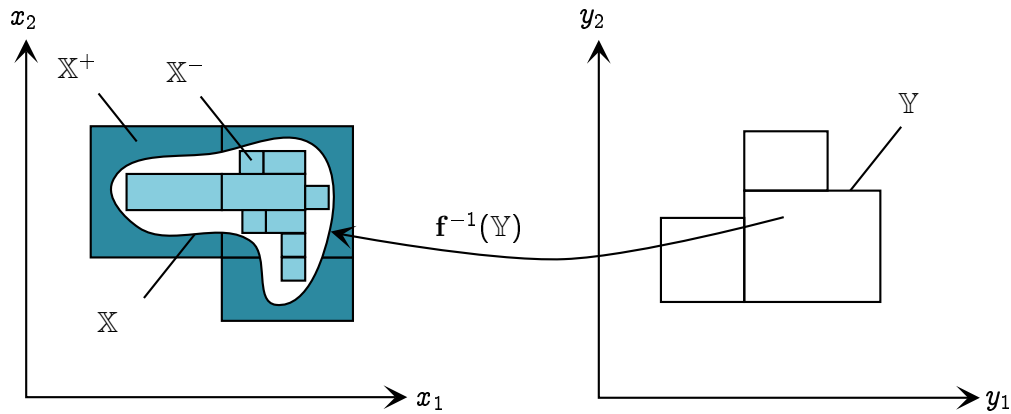


Figure A.14 – Exemple d'inversion ensembliste à l'aide de l'analyse par intervalles. L'objectif est alors de définir les deux ensembles \mathbb{X}^- et \mathbb{X}^+ .

2 Deux outils ensemblistes

Cette section présente deux outils ensemblistes, reposant sur l'analyse par intervalles, qui sont utilisés dans ce manuscrit. Le premier outils à être présenté correspond à l'algorithme SIVIA, qui permet de faire de l'inversion ensembliste. Puis les principes des problèmes de satisfaction de contraintes (CSP) ainsi que leurs résolutions à l'aide de l'analyse par intervalles seront détaillés.

2.1 Inversion ensembliste

2.1.1 Présentation

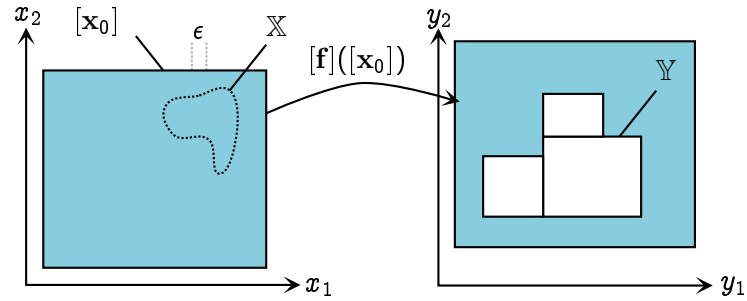
Soient f une fonction (potentiellement non-linaire) de \mathbb{R}^n à valeurs dans \mathbb{R}^m et \mathbb{Y} un sous-ensemble de \mathbb{R}^m . L'inversion ensembliste consiste à calculer

$$\mathbb{X} = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \in \mathbb{Y}\} = f^{-1}(\mathbb{Y}). \quad (\text{A.53})$$

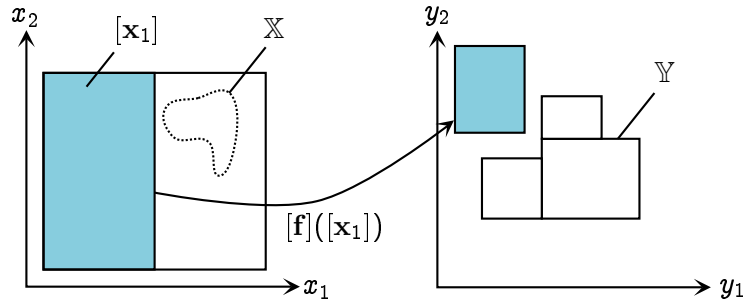
À l'aide de l'analyse par intervalles, il est possible de caractériser l'ensemble \mathbb{X} par une sous et une sur-approximations

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+. \quad (\text{A.54})$$

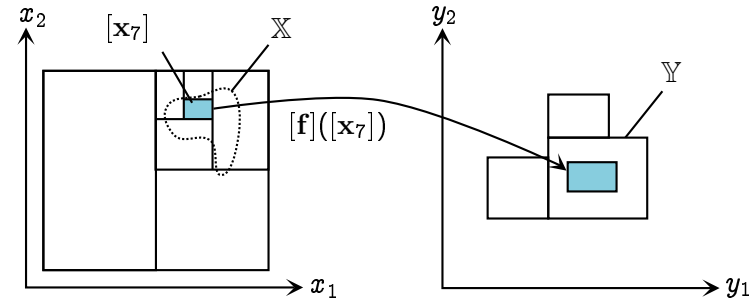
La Figure A.14 présente un exemple d'inversion ensembliste.



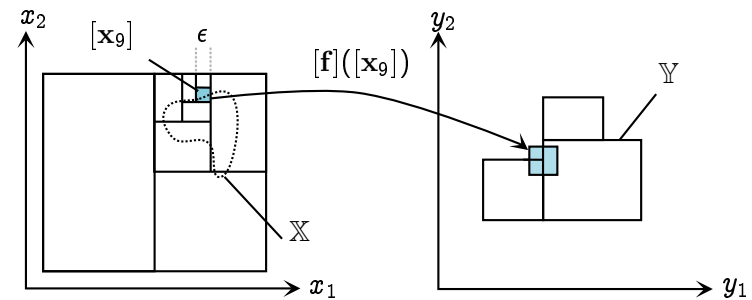
(a) Le pavé $[x_0]$ est indéterminé. Comme $w([x_0]) > \epsilon$, la boîte ne peut pas être gardée telle quelle, elle doit être bissectée. Les boîtes résultantes seront testées à leur tour.



(b) Le pavé $[x_1]$ n'est pas solution : $[f]([x_1]) \cap Y = \emptyset$.



(c) Le pavé $[x_7]$ est solution.



(d) Le pavé $[x_9]$ est indéterminé et est trop petit pour être bissecté (il est suffisamment petit pour être gardé). Ce pavé est donc définitivement classé indéterminé.

Figure A.15 – Les différents cas de l'algorithme SIVIA.

2 .1.2 Algorithme SIVIA

L'algorithme SIVIA (*Set Inverter Via Interval Analysis*¹) est un algorithme d'inversion ensembliste [Jaulin 1993]. Il permet d'obtenir une sous et une sur-approximations de \mathbb{X} , avec une précision aussi fine que désirée. Pour ce faire, l'algorithme a besoin des éléments suivants :

- un ensemble \mathbb{Y} approprié,
- un pavé de recherche $[\mathbf{x}_0]$ (potentiellement très grand) contenant l'ensemble \mathbb{X} ($\mathbb{X} \subset [\mathbf{x}_0]$),
- une fonction d'inclusion $[\mathbf{f}]$ de la fonction \mathbf{f} ,
- une précision ϵ .

Le principe est le suivant, on veut créer un sous-pavage de $[\mathbf{x}_0]$, en classant les sous-pavés en trois catégories :

- Les pavés solutions : un pavé $[\mathbf{x}_i]$ est solution si $[\mathbf{x}_i] \subset \mathbb{X}$, ou en d'autres termes si $[\mathbf{f}]([\mathbf{x}_i]) \subset \mathbb{Y}$.
- Les pavés non-solutions : un pavé $[\mathbf{x}_i]$ n'est pas solution si $[\mathbf{x}_i] \cap \mathbb{X} = \emptyset$ ou encore si $[\mathbf{f}]([\mathbf{x}_i]) \cap \mathbb{Y} = \emptyset$.
- Les pavés indéterminés : les pavés qui sont ni solutions ni non-solutions. Un pavé est indéterminé si $[\mathbf{f}]([\mathbf{x}_i]) \not\subset \mathbb{Y}$ et $[\mathbf{f}]([\mathbf{x}_i]) \cap \mathbb{Y} \neq \emptyset$.

Le tout en ajoutant une contrainte sur la taille maximale possible pour les pavés indéterminés : soit $[\mathbf{x}_i]$ un pavé indéterminé, $w([\mathbf{x}_i]) \leq \epsilon$. La Figure A.15 illustre le principe de l'algorithme.

L'algorithme SIVIA est présenté Algorithme 27. C'est un algorithme récursif qui, comme précisé au dessus, crée une sous et une sur-approximations de \mathbb{X} . Les approximations \mathbb{X}^- et \mathbb{X}^+ sont initialisées par des ensembles vides, puis remplies au fur et à mesure. À la fin on récupère les deux ensembles suivants :

$$\mathbb{X}^- = \{\text{pavés solutions}\}, \quad (\text{A.55})$$

$$\mathbb{X}^+ = \{\text{pavés solutions}\} \cup \{\text{pavés indéterminés}\}. \quad (\text{A.56})$$

On remarquera que \mathbb{X}^- peut être vide.

Dans l'Algorithme 27, lors du test de la boîte $[\mathbf{x}]$, on retrouve quatre cas possibles :

- Ligne 2, la boîte en cours ne contient aucune solution au problème d'inversion. Elle peut donc être supprimée (elle n'est enregistrée dans aucune des deux approximations) (Figure A.15b).
- Ligne 5-6, la boîte en cours ne contient que des solutions, elle doit donc faire partie des deux approximations (Figure A.15c).
- Ligne 9, la boîte en cours est indéterminée et est trop petite pour être bissectée (elle a atteint la précision voulue). La boîte reste donc indéterminée (il se peut

1. Inversion ensembliste à l'aide de l'analyse par intervalles

Algorithme 27: SIVIA

Données : $[\mathbf{x}], [\mathbf{f}], \mathbb{Y}, \epsilon, \mathbb{X}^-, \mathbb{X}^+$

- 1 **si** $[\mathbf{f}]([\mathbf{x}]) \cap \mathbb{Y} = \emptyset$ **alors**
- 2 | retour // Le pavé n'est pas solution, il n'est pas gardé
- 3 **sinon si** $[\mathbf{f}]([\mathbf{x}]) \subset \mathbb{Y}$ **alors**
- 4 | // Le pavé est solution, il est ajouté aux deux approximations
- 5 | $\mathbb{X}^- = \mathbb{X}^- \cup [\mathbf{x}];$
- 6 | $\mathbb{X}^+ = \mathbb{X}^+ \cup [\mathbf{x}];$
- 7 **sinon si** $w([\mathbf{x}]) < \epsilon$ **alors**
- 8 | // Le pavé est indéterminé et trop petit pour être bissecté
- 9 | // Il est alors ajouté à la sur-approximation
- 10 | $\mathbb{X}^+ = \mathbb{X}^+ \cup [\mathbf{x}];$
- 11 **sinon**
- 12 | // Le pavé est indéterminé et plus grand que la précision désirée
- 13 | // On bissecte alors le pavé et on teste les pavés résultants
- 14 | $([\mathbf{x}]', [\mathbf{x}]'') = \text{Bissect}([\mathbf{x}]);$
- 15 | $(\mathbb{X}^-, \mathbb{X}^+) = \text{SIVIA}([\mathbf{x}]', [\mathbf{f}], \mathbb{Y}, \epsilon, \mathbb{X}^-, \mathbb{X}^+);$
- 16 | $(\mathbb{X}^-, \mathbb{X}^+) = \text{SIVIA}([\mathbf{x}]'', [\mathbf{f}], \mathbb{Y}, \epsilon, \mathbb{X}^-, \mathbb{X}^+);$
- 17 **fin**

Résultat : $\mathbb{X}^-, \mathbb{X}^+.$

qu'elle contienne des solutions), et doit être ajoutée à la sur-approximation (Figure A.15d).

- Ligne 13-15, la boîte en cours est indéterminée, mais n'a pas atteint la précision voulue, elle doit donc être bissectée. Les boîtes résultantes de la bisection sont testées à leur tour (Figure A.15a).

2.2 Problèmes de satisfaction de contraintes

2.2.1 Présentation

Un problème de satisfaction de contraintes (CSP²) est défini par trois ensembles :

- un ensemble de variables $\mathbb{V} = \{v_1, v_2, \dots, v_k\}$,
- un ensemble de domaines pour ces variables $\mathbb{D} = \{d_1, d_2, \dots, d_k\}$, avec $v_i \in d_i, i = 1, \dots, k$,
- et un ensemble de contraintes (fonctions, relations...) $\mathbb{C} = \{c_1, c_2, \dots, c_m\}$ reliant les variables entre elles.

L'objectif est alors de réduire les différents domaines en supprimant les valeurs qui ne sont pas compatibles avec les contraintes.

2. Constraint Satisfaction Problem

Voici un exemple de CSP :

$$\begin{cases} \mathbb{V} = \{x_1, x_2\} \\ \mathbb{D} = \{\mathbb{X}_1, \mathbb{X}_2\} \\ \mathbb{C} = \{c_1 : x_2 = x_1^2, c_2 : x_2 = -2x_1 + 1\} \end{cases} . \quad (\text{A.57})$$

L'analyse par intervalles peut permettre de résoudre ce type de problèmes. En reprenant l'exemple de l'Équation A.57, on suppose cette fois que les domaines $\mathbb{X}_1 = [x_1] = [-10, 10]$ et $\mathbb{X}_2 = [x_2] = [-20, 100]$ sont des intervalles. L'objectif est alors de *contracter* $[x_1]$ et $[x_2]$ en supprimant les valeurs non-consistantes avec les contraintes c_1 et c_2 . Pour ce faire, on va associer un *contracteur* à chacune des contraintes.

2 .2.2 Contracteurs

Un contracteur correspond à un opérateur (algorithme) capable de réduire des domaines en fonction de la contrainte à laquelle il est associé. On notera qu'un contracteur ne doit jamais avoir recours à la bisection.

En considérant toujours l'exemple A.57, les Algorithmes 28 et 29 correspondent respectivement aux contracteurs des contraintes c_1 et c_2 . Les contracteurs présentés ici sont des contracteurs simples, on notera que pour des contraintes plus complexes on pourra toujours se ramener à des combinaisons de contracteurs élémentaires (Section 2 .2.4).

Calcul du résultat $([x_1]^*, [x_2]^*)$ du contracteur C_{c_1} sur le CSP A.57 (Figure A.16b) :

$$\begin{aligned} [x_2]^* &= [x_2] \cap [x_1]^2, & [x_1]^* &= [x_1] \cap (\sqrt{[x_2]^*} \sqcup -\sqrt{[x_2]^*}), \\ &= [-20, 100] \cap [-10, 10]^2, & &= [-10, 10] \cap (\sqrt{[0, 100]^*} \sqcup -\sqrt{[0, 100]^*}), \\ &= [-20, 100] \cap [0, 100], & &= [-10, 10] \cap [-10, 10], \\ [x_2]^* &= [0, 100]. & [x_1]^* &= [-10, 10]. \end{aligned}$$

On remarque que le contracteur réduit les domaines en supprimant les valeurs non-consistantes avec la contrainte associée (les valeurs négatives de $[x_2]$ dans ce cas). L'idée maintenant, est de propager cette information en utilisant le deuxième contracteur avec les nouvelles valeurs $[x_1] = [x_1]^*$ et $[x_2] = [x_2]^*$.

2 .2.3 Propagation de contraintes

Comme précisé Section 2 .2.2 les contracteurs permettent de réduire les domaines des variables au sein d'une contrainte. Pour un problème de satisfaction de

Algorithme 28: Contracteur C_{c_1} associé à $c_1 : x_2 = x_1^2$

Données : $[x_1], [x_2]$
1 $[x_2]^* = [x_2] \cap [x_1]^2$;
2 $[x_1]^* = [x_1] \cap (\sqrt{[x_2]^*} \sqcup -\sqrt{[x_2]^*})$;
Résultat : $[x_1]^*, [x_2]^*$.

Algorithme 29: Contracteur C_{c_2} associé à $c_2 : x_2 = -2x_1 + 1$

Données : $[x_1], [x_2]$
1 $[x_2]^* = [x_2] \cap -2[x_1] + 1$;
2 $[x_1]^* = [x_1] \cap \frac{1 - [x_2]^*}{2}$;
Résultat : $[x_1]^*, [x_2]^*$.

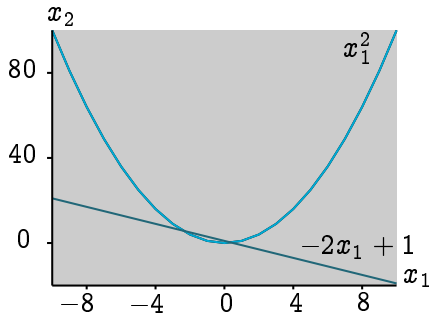
contraintes, plusieurs contraintes sont généralement considérées. La technique de résolution du CSP consiste alors à utiliser successivement les différents contracteurs associés aux contraintes, en conservant les résultats des contractions d'un appel à l'autre, et ainsi de suite jusqu'à l'obtention d'un point fixe : c'est à dire dès qu'aucun contracteur n'arrive à contracter les domaines. On notera qu'un autre critère d'arrêt peut être considéré : le nombre maximal d'appels des contracteurs.

L'Algorithme 30 présente une méthode de résolution d'un CSP à l'aide de contracteurs associés aux contraintes. On peut remarquer que l'appel des contracteurs se fait tant que le nombre d'itérations z_{max} n'a pas été atteint et tant que l'on contracte. Ligne 6, si $\mathbb{D} = \mathbb{D}^*$ alors aucun domaine n'a été contracté pendant l'itération courante, un point fixe a donc été atteint.

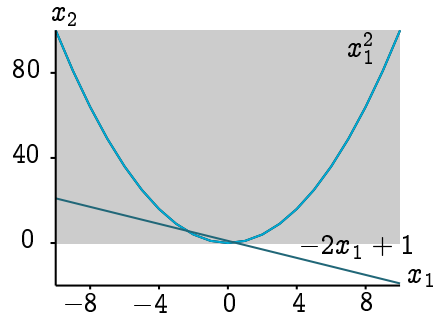
Algorithme 30: Résolution d'un CSP

Données : $\mathbb{V} = \{x_1, \dots, x_k\}, \mathbb{D} = \{[x_1], \dots, [x_k]\}, \mathbb{C} = \{c_1, \dots, c_m\}, z_{max}$
1 $z = 0$; $\mathbb{D}^* = \mathbb{D}$ // Initialisations;
2 répéter
3 $\mathbb{D} = \mathbb{D}^*$;
4 **pour chaque** $c_i, i = 1, \dots, m$ **faire**
5 $\mathbb{D}^* = C_{c_i}(\mathbb{D})$ // À chaque itération on appelle tous les contracteurs;
6 **fin**
7 $z = z + 1$ // On calcule le nombre d'itérations;
8 **tant que** $z < z_{max}$ **et** $\mathbb{D}^* \neq \mathbb{D}$;
Résultat : \mathbb{D}^* .

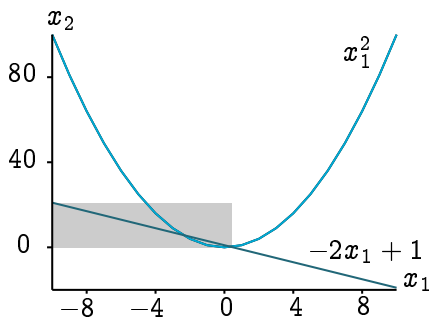
La Figure A.16 présente les premières contractions ainsi que le point fixe du CSP A.57. Ci-suit le détail des différentes contractions présentées. Calcul du résultat $([x_1]^*, [x_2]^*)$ du contracteur C_{c_2} en considérant le résultat obtenu Section 2.2.2



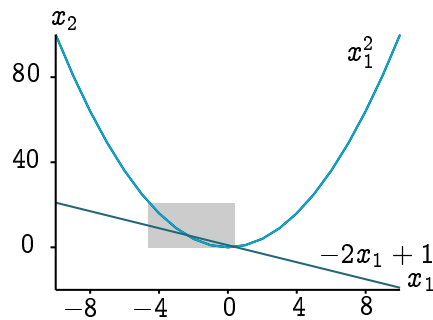
(a) Le CSP considéré. Les solutions correspondent aux points d'intersections des deux droites (contraintes).



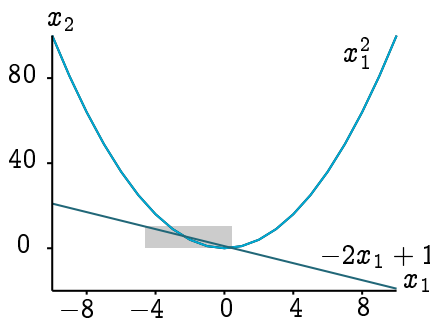
(b) Appel du premier contracteur : $([x_1], [x_2]) = C_{c_1}([x_1], [x_2])$.



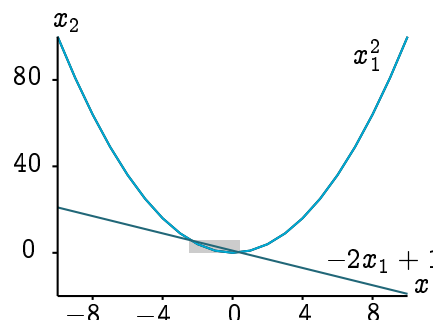
(c) Appel du deuxième contracteur : $([x_1], [x_2]) = C_{c_2}([x_1], [x_2])$.



(d) On propage l'information en rappelant le premier contracteur : $([x_1], [x_2]) = C_{c_1}([x_1], [x_2])$.



(e) Puis on rappelle le deuxième contracteur : $([x_1], [x_2]) = C_{c_2}([x_1], [x_2])$.



(f) Et ainsi de suite jusqu'à l'obtention d'un point fixe (les contracteurs ne peuvent plus contracter).

Figure A.16 – Exemple de propagation de contraintes sur un CSP.

(Figure A.16c) :

$$\begin{aligned}
[x_2]^* &= [x_2] \cap -2[x_1] + 1, & [x_1]^* &= [x_1] \cap \frac{1 - [x_2]^*}{2}, \\
&= [0, 100] \cap -2[-10, 10] + 1, & &= [-10, 10] \cap \frac{1 - [0, 21]}{2}, \\
&= [0, 100] \cap [-19, 21], & &= [-10, 10] \cap \left[-10, \frac{1}{2}\right], \\
[x_2]^* &= [0, 21]. & [x_1]^* &= \left[-10, \frac{1}{2}\right].
\end{aligned}$$

Calcul du résultat $([x_1]^*, [x_2]^*)$ lors du rappel de C_{c_1} (Figure A.16d) :

$$\begin{aligned}
[x_2]^* &= [x_2] \cap [x_1]^2, & [x_1]^* &= [x_1] \cap (\sqrt{[x_2]^*} \sqcup -\sqrt{[x_2]^*}), \\
&= [0, 21] \cap \left[-10, \frac{1}{2}\right]^2, & &= [-10, \frac{1}{2}] \cap (\sqrt{[0, 21]^*} \sqcup -\sqrt{[0, 21]^*}), \\
&= [0, 21] \cap [0, 100], & &= [-10, \frac{1}{2}] \cap [-\sqrt{21}, \sqrt{21}], \\
[x_2]^* &= [0, 21]. & [x_1]^* &= \left[-\sqrt{21}, \frac{1}{2}\right].
\end{aligned}$$

Calcul du résultat $([x_1]^*, [x_2]^*)$ lors du rappel de C_{c_2} (Figure A.16e) :

$$\begin{aligned}
[x_2]^* &= [x_2] \cap -2[x_1] + 1, & [x_1]^* &= [x_1] \cap \frac{1 - [x_2]^*}{2}, \\
&= [0, 21] \cap -2\left[-\sqrt{21}, \frac{1}{2}\right] + 1, & &= [-\sqrt{21}, \frac{1}{2}] \cap \frac{1 - [0, 2\sqrt{21} + 1]}{2}, \\
&= [0, 21] \cap [0, 2\sqrt{21} + 1], & &= [-\sqrt{21}, \frac{1}{2}] \cap \left[-10, \frac{1}{2}\right], \\
[x_2]^* &= [0, 2\sqrt{21} + 1]. & [x_1]^* &= \left[-\sqrt{21}, \frac{1}{2}\right].
\end{aligned}$$

Remarque A.6 *Quand un point fixe est atteint, il est possible de bissecter le domaine courant et de recommencer le processus de contraction sur les deux pavés résultants de la bissection. Ainsi on pourrait identifier plus précisément les deux solutions de l'exemple présent sur la Figure A.16f.*

2.2.4 Propagation avant-arrière

Comme précisé précédemment, les contracteurs présentés sur l'exemple A.57 sont des contracteurs simples, associés à des contraintes simples. Pour réaliser les contracteurs associés à des contraintes plus complexes, il est possible d'utiliser le principe de *propagation avant-arrière* (*forward-backward propagation*). L'objectif va alors être de décomposer la contrainte complexe en contraintes élémentaires, créant ainsi

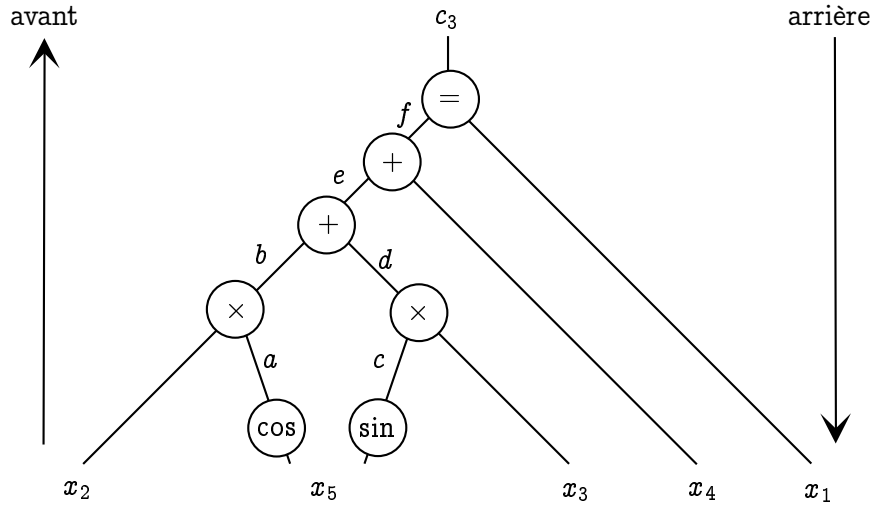


Figure A.17 – Exemple d'arbre généré par la décomposition d'une contrainte complexe en contraintes élémentaires.

un *arbre* ayant pour racine la contrainte complexe et pour feuilles les contraintes élémentaires. La propagation avant-arrière va ensuite consister à effectuer des contractions élémentaires depuis les feuilles vers la racine (avant), puis depuis la racine vers les feuilles (arrière).

Considérons par exemple la contrainte suivante :

$$c_3 : x_1 = x_2 \cos(x_5) + x_3 \sin(x_5) + x_4. \quad (\text{A.58})$$

Il est possible de la décomposer en contraintes élémentaires :

$$\begin{aligned} a &= \cos(x_5), & b &= a \times x_2, \\ c &= \sin(x_5), & d &= c \times x_3, \\ e &= b + d, & f &= e + x_4. \end{aligned}$$

On a alors $c_3 : x_1 = f$. Cette décomposition génère l'arbre présenté Figure A.17.

En s'appuyant sur cette décomposition, il est possible de construire un contracteur C_{c_3} associé à la contrainte c_3 . Ce contracteur (Algorithme 31) est basé sur le principe de contraction avant-arrière et a trois étapes principales :

- Lignes 2-3 : Les variables intermédiaires créées lors de la décomposition en contraintes élémentaires, sont initialisées.
- Lignes 6-12 : Contraction *en avant*. Les contraintes élémentaires sont *retournées* afin de contracter les domaines des variables qui sont à *droite* du signe égal. On remarquera que pour chaque contraction un contracteur élémentaire est appelé (division, soustraction, ...).

- Lignes 13-17 : Contraction *en arrière*. Retour sur les formes initiales des contraintes élémentaires (comme pour l'étape d'initialisation), et contraction des domaines, toujours en manipulant des contracteurs élémentaires.

Les deux dernières étapes sont répétées tant qu'on n'a pas atteint de point stable (tant que les domaines peuvent encore être contractés). On peut cependant considérer d'autres critères d'arrêt :

- Un nombre maximal d'itérations z_{max} : après z_{max} passages dans la boucle **répéter** - tant que on arrête, même si on contracte encore les différents domaines.
- Un pourcentage de contraction minimal : on arrête dès qu'on ne contracte plus assez vite (la réduction des domaines entre chaque itération n'est pas assez significative).

Algorithme 31: Contracteur C_{c_3} associé à $c_3 : x_1 = x_2 \cos(x_5) + x_3 \sin(x_5) + x_4$

Données : $[x_1], [x_2], [x_3], [x_4], [x_5]$

```

1 // Initialisations
2  $[a] = \cos([x_5]); [b] = [a] \times [x_2]; [c] = \sin([x_5]); [d] = [c] \times [x_3];$ 
3  $[e] = [b] + [d]; [f] = [e] + [x_4];$ 
4 répéter
5   // Contraction avant
6    $[x_2] = [x_2] \cap \frac{[b]}{[a]}; [a] = [a] \cap \frac{[b]}{[x_2]};$ 
7    $[x_5] = [x_5] \cap \cos^{-1}([a]);$ 
8    $[x_3] = [x_3] \cap \frac{[d]}{[c]}; [c] = [c] \cap \frac{[d]}{[x_3]};$ 
9    $[x_5] = [x_5] \cap \sin^{-1}([c]);$ 
10   $[f] = [f] \cap [x_1];$ 
11   $[e] = [e] \cap ([f] - [x_4]); [x_4] = [x_4] \cap ([f] - [e]);$ 
12   $[b] = [b] \cap ([e] - [d]); [d] = [d] \cap ([e] - [b]);$ 
13  // Contraction arrière
14   $[a] = [a] \cap \cos([x_5]); [b] = [b] \cap ([a] \times [x_2]);$ 
15   $[c] = [c] \cap \sin([x_5]); [d] = [d] \cap ([c] \times [x_3]);$ 
16   $[e] = [e] \cap ([b] + [d]); [f] = [f] \cap ([e] + [x_4]);$ 
17   $[x_1] = [x_1] \cap [f];$ 
18 tant que que l'on contracte;
Résultat :  $[x_1], [x_2], [x_3], [x_4], [x_5]$ .

```

Présentation des outils probabilistes

Sommaire

1	Localisation probabiliste	184
1.1	Introduction aux probabilités	184
1.2	Approche probabiliste de la localisation	188
1.3	Filtre Bayésien	191
2	Filtres Gaussiens	194
2.1	Présentation générale	194
2.2	Filtre de Kalman	196
2.3	Filtre de Kalman étendu	198
3	Filtres Non-paramétriques	205
3.1	Présentation générale	205
3.2	Filtre à particules	205

Cette annexe présente différents outils probabilistes couramment utilisés pour traiter les problèmes de localisation en robotique mobile. L'objectif de cette présentation est de donner au lecteur les notions théoriques nécessaires pour comprendre les algorithmes classiquement utilisés pour la localisation en robotique mobile. Cette annexe se découpe en trois parties. La première partie fait une présentation générale de quelques notions de probabilités et présente l'approche probabiliste du problème de localisation. La deuxième partie s'intéresse aux filtres Gaussiens notamment utilisés pour résoudre les problèmes d'estimation de posture tandis que la dernière partie s'intéresse aux filtres non-paramétriques, généralement utilisés pour les problèmes de localisation globale. Cette annexe a été écrite en se basant, entre autres, sur [Thrun 2005].

1 Localisation probabiliste

Cette section a deux objectifs. Le premier est de présenter les outils probabilistes classiquement utilisés en robotique mobile et le deuxième est de présenter l'approche probabiliste du problème de localisation.

1.1 Introduction aux probabilités

Soit X une variable aléatoire et x une valeur particulière possible pour X . On peut par exemple considérer X comme correspondant à la face visible d'une pièce après un *pile ou face*. La valeur x peut alors avoir deux valeurs, $x = \text{pile}$ ou $x = \text{face}$. Si les valeurs possibles pour X sont discrètes, alors

$$p(X = x) \tag{B.1}$$

désigne la probabilité que X aie pour valeur x . En reprenant l'exemple de la pièce (et en supposant qu'elle soit bien équilibrée), on a $p(X = \text{pile}) = p(X = \text{face}) = \frac{1}{2}$. On notera que la somme des probabilités de toutes les valeurs possibles pour X est égale à 1.

$$\sum_{\forall x} p(X = x) = 1. \tag{B.2}$$

Ce qui revient à dire qu'il y a 100% de chance que X vaille une valeur de X ... On notera aussi qu'il est impossible d'avoir une probabilité négative.

Par la suite, pour alléger les notations, $p(X = x)$ sera notée $p(x)$.

Les variables aléatoires considérés pour la localisation en robotique mobile sont pour la plupart continues. À partir de maintenant, on supposera que toutes les variables considérées possèdent une fonction de *densité de probabilité* (PDF¹).

Densité de probabilité

Une fonction de densité de probabilité correspond à une fonction qui pour un x donné, donne la probabilité que la valeur aléatoire X soit égale à x .

Un exemple de PDF classique : la *distribution normale* de moyenne μ et de variance σ^2 :

$$p(x) = (2\pi\sigma^2)^{-1/2} \exp\left(-\frac{1}{2} \frac{(x - \mu)^2}{\sigma^2}\right). \tag{B.3}$$

1. Probability Density Function

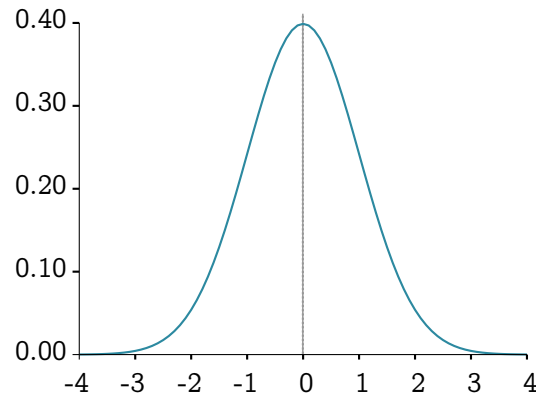


Figure B.1 – Densité de probabilité Gaussienne avec $\mu = 0$ et $\sigma^2 = 1$. Cette distribution particulière est aussi appelée *loi normale centrée réduite*.

Cette distribution est aussi appelée *Loi normale*, *Distribution Gaussienne* ou encore *Loi Gaussienne* (Figure B.1). On notera $X \sim \mathcal{N}(\mu, \sigma^2)$ pour signifier que la variable aléatoire X est distribuée selon une densité de probabilité Gaussienne de moyenne μ et de variance σ^2 . Telle que présentée dans l'Équation B.3, la loi normale suppose que x est un scalaire. La distribution normale considérant un vecteur \mathbf{x} est appelée *Distribution normale multivariée* et correspond à

$$p(\mathbf{x}) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu})\right), \quad (\text{B.4})$$

avec $\boldsymbol{\mu}$ le vecteur des moyennes et Σ une matrice symétrique (semi-définie positive) appelée la *matrice de covariance*.

Matrice symétrique

Une matrice A est symétrique si elle est égale à sa propre transposée : $A = A^T$. En d'autres termes, plus intuitivement, une matrice est symétrique si ses coefficients sont symétriques par rapport à sa diagonale Nord-Ouest/Sud-Est. Exemple de matrice symétrique :

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 0 & 4 \\ 3 & 4 & 5 \end{pmatrix}.$$

Transposée d'une matrice

La transposée d'une matrice A , notée A^T , correspond à la matrice obtenue en inversant les lignes et les colonnes de A . Par exemple

$$\text{si } A = \begin{pmatrix} 1 & 2 \\ 5 & 0 \\ 3 & 4 \end{pmatrix} \text{ alors } A^T = \begin{pmatrix} 1 & 5 & 3 \\ 2 & 0 & 4 \end{pmatrix}.$$

Matrice semi-définie positive

Une matrice A est semi-définie positive si, pour tout vecteur x non nul,

$$x^T A x \geq 0. \quad (\text{B.5})$$

Comme pour le cas discret, la somme de toutes les probabilités des valeurs possibles de X est égale à 1

$$\int p(x) dx = 1. \quad (\text{B.6})$$

Par la suite on utilisera les termes *probabilité*, *densité de probabilité* et *fonction de densité de probabilité* de façons équivalentes.

La *probabilité jointe* entre deux variables aléatoires X et Y est définie par

$$p(x, y) = p(X = x \text{ et } Y = y). \quad (\text{B.7})$$

Elle décrit la probabilité que la variable X vaille x et que la variable Y vaille y . On peut noter que si X et Y sont indépendantes (la valeur de X n'a pas d'influence sur la valeur de Y) alors

$$p(x, y) = p(x)p(y). \quad (\text{B.8})$$

Il arrive souvent que des variables aléatoires conditionnent l'état d'autres variables aléatoires. Note alors la *probabilité conditionnelle*

$$p(x | y) = p(X = x | Y = y), \quad (\text{B.9})$$

comme étant la probabilité d'avoir $X = x$ sachant qu'on a $Y = y$. Si $p(y) \neq 0$ alors on peut écrire la probabilité conditionnelle précédente comme étant

$$p(x | y) = \frac{p(x, y)}{p(y)}. \quad (\text{B.10})$$

On remarque que si X et Y sont indépendantes alors

$$p(x | y) = \frac{p(x)p(y)}{p(y)} = p(x). \quad (\text{B.11})$$

En d'autres termes, si X et Y sont indépendantes alors connaître Y ne donne aucune information sur X ...

Ci-suivent deux théorèmes fondamentaux pour l'approche statistique du problème de localisation en robotique mobile.

Théorème de probabilité totale :

$$p(x) = \sum_{\forall y} p(x | y)p(y) \quad (\text{cas discret}) \quad (\text{B.12})$$

$$p(x) = \int p(x | y)p(y)dy \quad (\text{cas continu}) \quad (\text{B.13})$$

Théorème de Bayes :

$$p(x | y) = \frac{p(y | x)p(x)}{p(y)} = \frac{p(y | x)p(x)}{\sum_{\forall x'} p(y | x')p(x')} \quad (\text{cas discret}) \quad (\text{B.14})$$

$$p(x | y) = \frac{p(y | x)p(x)}{p(y)} = \frac{p(y | x)p(x)}{\int p(y | x')p(x')dx'} \quad (\text{cas continu}) \quad (\text{B.15})$$

Le théorème de Bayes joue un rôle prépondérant en robotique probabiliste. En supposant que x soit une valeur qui dépende de y , la probabilité $p(x)$ est appelée la *probabilité antérieure* et y correspond aux données (mesures). La probabilité $p(x)$ correspond à la connaissance que nous avons de X avant de prendre en compte les mesures y . La probabilité $p(x | y)$ est quand à elle appelée la *probabilité postérieure*. Le théorème de Bayes permet donc d'évaluer $p(x | y)$ en utilisant la probabilité conditionnelle inverse $p(y | x)$.

Il est possible de remarquer que le dénominateur du théorème de Bayes $p(y)$, Équations B.14 et B.15, ne dépend pas de x . C'est pourquoi $p(y)^{-1}$ est souvent défini comme une normalisation et est notée η . Ce qui nous donne

$$p(x | y) = \eta p(y | x)p(x). \quad (\text{B.16})$$

On précisera qu'il est possible de conditionner les relations présentées précédemment avec une troisième variable aléatoire Z . En conditionnant le théorème de Bayes avec $Z = z$ on obtient

$$p(x | y, z) = \frac{p(y | x, z)p(x | z)}{p(y | z)}. \quad (\text{B.17})$$

En considérant des variables X et Y indépendantes, on a (Équation B.8)

$$p(x, y | z) = p(x | z)p(y | z) \quad (\text{B.18})$$

Cette relation est connue comme étant la relation d'*indépendance conditionnelle*, et est équivalente à

$$p(x | z) = p(x | z, y), \quad (\text{B.19})$$

$$p(y | z) = p(y | z, x). \quad (\text{B.20})$$

L'espérance d'une variable aléatoire X est donnée par

$$E[X] = \sum_{\forall x} xp(x) \quad (\text{cas discret}), \quad (\text{B.21})$$

$$E[X] = \int xp(x)dx \quad (\text{cas continu}). \quad (\text{B.22})$$

La variance d'une variable aléatoire X est définie par

$$V(X) = E[(X - E[X])^2]. \quad (\text{B.23})$$

Dans cette annexe, la variance est aussi notée σ^2 .

La covariance entre deux variables aléatoires X et Y correspond à

$$\text{cov}(X, Y) = E[(X - E[X])(Y - E[Y])]. \quad (\text{B.24})$$

Espérance, variance et covariance

L'espérance correspond à la tendance centrale d'une variable aléatoire. Elle est équivalente à la moyenne de la densité de probabilité de cette variable. La variance qualifie la dispersion des valeurs d'une variable aléatoire autour de sa tendance centrale. On peut noter que la variance correspond au carré de l'écart type.

La covariance permet d'évaluer simultanément la variation de deux variables aléatoires par rapport à leurs moyennes respectives. Elle donne une indication sur le *lien* qui unit ces deux variables.

La Figure B.2 illustre les notions de moyenne et de variance.

1 .2 Approche probabiliste de la localisation

1 .2.1 Définitions générales

Dans la majorité des problèmes en robotique mobile (dont le problème de localisation), l'état du système change au cours du temps. Ce dernier est généralement

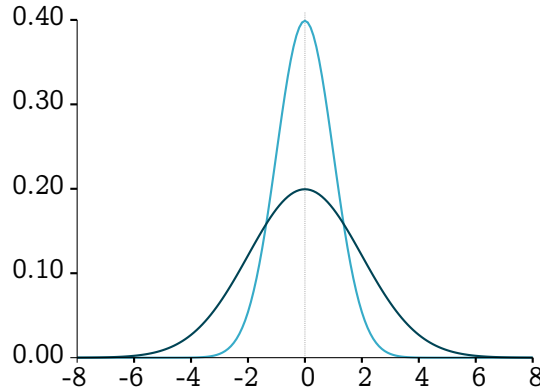


Figure B.2 – Deux densités de probabilités avec des moyennes égales mais des dispersions (variances) différentes. La densité foncée correspond à une loi normale avec $\mu = 0$ et $\sigma^2 = 2$, tandis que la courbe claire correspond à une loi normale de moyenne nulle et de variance $\sigma^2 = 1$.

considéré comme étant discret. En d'autres termes, tous les événements intéressants se passent aux instants $t = 0, 1, 2, \dots$. L'instant particulier $t = 0$ est associé à la situation initiale du problème considéré.

Pour se localiser, un robot est généralement capable de *voir* son environnement. Le vecteur de mesures fournit des informations sur l'environnement à un instant donné. Ces mesures peuvent être réalisées par différents types de capteurs (camera, capteur de distance, ...). Le vecteur de mesures effectué à l'instant t est noté \mathbf{z}_t , et

$$\mathbf{z}_{t_1:t_2} = \mathbf{z}_{t_1}, \mathbf{z}_{t_1+1}, \mathbf{z}_{t_1+2}, \dots, \mathbf{z}_{t_2} \quad (\text{B.25})$$

correspond à l'ensemble des vecteurs de mesures effectués entre les deux instants t_1 et t_2 .

Un robot est aussi généralement capable de se déplacer dans son environnement. Le vecteur de contrôles permet d'évaluer les différents déplacements (changements d'état) du robot. Classiquement, ce vecteur est associé à l'odométrie du robot. Bien que l'odométrie corresponde en réalité à des mesures de capteurs, elle est traitée comme un vecteur de contrôles étant donné qu'elle fournit des informations sur la posture du robot. Le vecteur de contrôles noté \mathbf{u}_t correspond au déplacement du robot entre l'instant $t - 1$ et l'instant t . Comme pour le vecteur de mesures

$$\mathbf{u}_{t_1:t_2} = \mathbf{u}_{t_1}, \mathbf{u}_{t_1+1}, \mathbf{u}_{t_1+2}, \dots, \mathbf{u}_{t_2} \quad (\text{B.26})$$

correspond à l'ensemble des vecteurs de contrôles effectués entre les instants t_1 et t_2 .

Pour une approche probabiliste du problème de localisation, on considère que l'évolution de l'état ainsi que les mesures sont régies par des lois probabilistes. Dans un premier temps on peut définir la loi de probabilité de l'état courant \mathbf{x}_t comme étant

$$p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}). \quad (\text{B.27})$$

Remarque B.1 Dans cette annexe on considère l'état \mathbf{x} comme étant l'état de l'environnement. Le robot faisant parti de son environnement, quand le robot se déplace, il modifie l'état de son environnement.

En supposant que \mathbf{x} soit un état complet, on peut en déduire que \mathbf{x}_{t-1} est statistiquement représentatif de tous les contrôles et mesures précédents, c'est à dire $\mathbf{u}_{1:t-1}$ et $\mathbf{z}_{1:t-1}$. En d'autres termes, les contrôles et mesures précédents n'apportent aucune information supplémentaire à l'état \mathbf{x}_{t-1} . On peut donc en conclure que

$$p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}) = p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_t). \quad (\text{B.28})$$

Cette équation signifie que la posture courante du robot dépend de sa posture précédente et de son déplacement.

État complet

Un état \mathbf{x}_t est dit complet s'il correspond au *meilleur estimateur* de l'état futur \mathbf{x}_{t+1} . En d'autres termes, si les états précédents $(\mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \dots)$ n'apportent aucune information supplémentaire utile à la prédiction de l'état futur.

De façon similaire on peut modéliser les vecteurs de mesures

$$p(\mathbf{z}_t \mid \mathbf{x}_{0:t}, \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}) = p(\mathbf{z}_t \mid \mathbf{x}_t). \quad (\text{B.29})$$

Cette équation signifie que les mesures réalisées à un instant t ne dépendent que de l'état de l'environnement (incluant la posture du robot) à ce même instant.

Les Équations B.28 et B.29 forment le système dynamique stochastique du robot.

1.2.2 Notion de croyance

Un autre concept majeur de la localisation probabiliste : la notion de *croyance*. La croyance illustre la connaissance qu'a le robot sur l'état de son environnement. C'est une densité de probabilité qui caractérise la *confiance* qu'un robot peut avoir sur son estimation de l'état de l'environnement. La croyance d'un état \mathbf{x} est notée $bel(\mathbf{x})$ et correspond à

$$bel(\mathbf{x}_t) = p(\mathbf{x}_t \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t}). \quad (\text{B.30})$$

Dans la formulation précédente, on remarque que les dernières mesures \mathbf{z}_t ont déjà été intégrées pour l'estimation de l'état. Par la suite, il va être nécessaire d'évaluer la croyance de l'état courant \mathbf{x}_t avant l'intégration du dernier jeu de mesures. Cette densité, souvent appelée *prédiction*, est notée $\overline{bel}(\mathbf{x}_t)$ et correspond à

$$\overline{bel}(\mathbf{x}_t) = p(\mathbf{x}_t \mid \mathbf{z}_{1:t-1}, \mathbf{u}_{1:t}). \quad (\text{B.31})$$

On notera que calculer $bel(\mathbf{x}_t)$ à partir de $\overline{bel}(\mathbf{x}_t)$ est appelée *correction*. En effet en calculant $\overline{bel}(\mathbf{x}_t)$ on évalue l'état courant en fonction de l'état précédent et du vecteur de contrôles (déplacement), puis en rajoutant le dernier jeu de mesures on va corriger cette prédiction pour obtenir $bel(\mathbf{x}_t)$.

1 .3 Filtre Bayésien

1 .3.1 Principe

L'algorithme le plus général pour calculer les croyances correspond au *filtre Bayésien*.

L'Algorithme 32 présente le filtre Bayésien basique. C'est un algorithme récursif qui calcule la croyance de l'état courant en fonction de la croyance de l'état précédent.

Algorithme 32: Filtre Bayésien

Données : $bel(\mathbf{x}_{t-1}), \mathbf{u}_t, \mathbf{z}_t$
 1 pour tous les \mathbf{x}_t faire
 2 $\overline{bel}(\mathbf{x}_t) = \int p(\mathbf{x}_t | \mathbf{u}_t, \mathbf{x}_{t-1}) bel(\mathbf{x}_{t-1}) d\mathbf{x}$;
 3 $bel(\mathbf{x}_t) = \eta p(\mathbf{z}_t | \mathbf{x}_t) \overline{bel}(\mathbf{x}_t)$;
 4 fin
 Résultat : $bel(\mathbf{x}_t)$.

1 .3.2 Exemple

Cet exemple est inspiré de [Thrun 2005]. Supposons qu'un robot n'ait que deux positions possibles dans son environnement, soit il est dans la pièce 1, soit dans la pièce 2 (Figure B.3). Afin de détecter la pièce dans laquelle il se trouve, le robot possède une caméra. Il peut aussi se déplacer (reculer) afin de changer de pièce. On notera X la variable aléatoire associée à la position du robot, Z la variable aléatoire associée à la mesure de la caméra et U la variable aléatoire associée au déplacement du robot.

À l'instant initial $t = 0$, le robot ne sait pas dans quelle pièce il se trouve. On assigne donc la même probabilité aux deux possibilités :

$$bel(X_0 = \text{pièce_1}) = 0.5, \quad (\text{B.32})$$

$$bel(X_0 = \text{pièce_2}) = 0.5. \quad (\text{B.33})$$

Les erreurs d'identifications des pièces par la caméra sont caractérisées par les

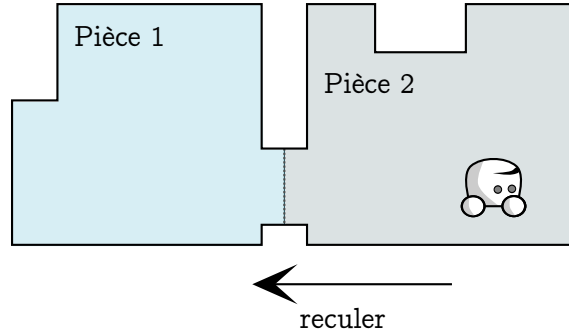


Figure B.3 – Exemple considéré pour illustrer le filtre Bayésien.

probabilités conditionnelles suivantes

$$p(Z_t = \text{pièce_1} \mid X_t = \text{pièce_1}) = 0.6, \quad (\text{B.34})$$

$$p(Z_t = \text{pièce_2} \mid X_t = \text{pièce_1}) = 0.4, \quad (\text{B.35})$$

et

$$p(Z_t = \text{pièce_1} \mid X_t = \text{pièce_2}) = 0.2, \quad (\text{B.36})$$

$$p(Z_t = \text{pièce_2} \mid X_t = \text{pièce_2}) = 0.8. \quad (\text{B.37})$$

En d'autres termes, la caméra identifie relativement bien la pièce 2, par contre quand le robot est dans la pièce 1 la probabilité d'identifier la mauvaise pièce est de 0.4.

Comme précisé précédemment le robot peut se déplacer (reculer). Ainsi s'il est dans la pièce 1, il restera dans la pièce 1, par contre si le robot est dans la pièce 2 il a une probabilité 0.8 de changer de pièce. Ce qui se traduit par :

$$p(X_t = \text{pièce_1} \mid U_t = \text{reculer}, X_{t-1} = \text{pièce_1}) = 1, \quad (\text{B.38})$$

$$p(X_t = \text{pièce_2} \mid U_t = \text{reculer}, X_{t-1} = \text{pièce_1}) = 0, \quad (\text{B.39})$$

$$p(X_t = \text{pièce_1} \mid U_t = \text{reculer}, X_{t-1} = \text{pièce_2}) = 0.8, \quad (\text{B.40})$$

$$p(X_t = \text{pièce_2} \mid U_t = \text{reculer}, X_{t-1} = \text{pièce_2}) = 0.2. \quad (\text{B.41})$$

Le robot peut aussi décider de ne pas reculer, et dans ce cas on a

$$p(X_t = \text{pièce_1} \mid U_t = \text{ne_rien_faire}, X_{t-1} = \text{pièce_1}) = 1, \quad (\text{B.42})$$

$$p(X_t = \text{pièce_2} \mid U_t = \text{ne_rien_faire}, X_{t-1} = \text{pièce_1}) = 0, \quad (\text{B.43})$$

$$p(X_t = \text{pièce_1} \mid U_t = \text{ne_rien_faire}, X_{t-1} = \text{pièce_2}) = 0, \quad (\text{B.44})$$

$$p(X_t = \text{pièce_2} \mid U_t = \text{ne_rien_faire}, X_{t-1} = \text{pièce_2}) = 1. \quad (\text{B.45})$$

Supposons maintenant qu'entre $t = 0$ et $t = 1$ le robot ne s'est pas déplacé, et qu'à l'instant $t = 1$ il estime qu'il est dans la pièce 2. Dans ce cas l'étape de

prédiction (Ligne 2 de l'Algorithme 32) s'écrit

$$\begin{aligned}\overline{bel}(\mathbf{x}_1) &= \int p(\mathbf{x}_1 | \mathbf{u}_1, \mathbf{x}_0) bel(\mathbf{x}_0) d\mathbf{x}, \\ &= \sum_{\forall \mathbf{x}_0} p(\mathbf{x}_1 | \mathbf{u}_1, \mathbf{x}_0) bel(\mathbf{x}_0), \\ &= p(\mathbf{x}_1 | U_1 = \text{ne_rien_faire}, X_0 = \text{pièce_1}) bel(X_0 = \text{pièce_1}) \\ &\quad + p(\mathbf{x}_1 | U_1 = \text{ne_rien_faire}, X_0 = \text{pièce_2}) bel(X_0 = \text{pièce_2}).\end{aligned}$$

Il faut alors remplacer \mathbf{x}_1 pas les deux valeurs possibles pour la variables X_1 .
Pour $X_1 = \text{pièce_1}$ on a

$$\begin{aligned}\overline{bel}(X_1 = \text{pièce_1}) &= \\ & p(X_1 = \text{pièce_1} | U_1 = \text{ne_rien_faire}, X_0 = \text{pièce_1}) bel(X_0 = \text{pièce_1}) + \\ & p(X_1 = \text{pièce_1} | U_1 = \text{ne_rien_faire}, X_0 = \text{pièce_2}) bel(X_0 = \text{pièce_2}) \\ \overline{bel}(X_1 = \text{pièce_1}) &= 1 \times 0.5 + 0 \times 0.5 = 0.5\end{aligned}$$

Et pour $X_1 = \text{pièce_2}$

$$\begin{aligned}\overline{bel}(X_1 = \text{pièce_2}) &= \\ & p(X_1 = \text{pièce_2} | U_1 = \text{ne_rien_faire}, X_0 = \text{pièce_1}) bel(X_0 = \text{pièce_1}) + \\ & p(X_1 = \text{pièce_2} | U_1 = \text{ne_rien_faire}, X_0 = \text{pièce_2}) bel(X_0 = \text{pièce_2}) \\ \overline{bel}(X_1 = \text{pièce_2}) &= 0 \times 0.5 + 1 \times 0.5 = 0.5\end{aligned}$$

On peut remarquer que $\overline{bel}(\mathbf{x}_1) = bel(\mathbf{x}_0)$ ce qui est normal étant donné que le contrôle `ne_rien_faire` ne modifie pas l'environnement (la position du robot).

Il reste maintenant à tenir compte de la mesure "pièce 2" du capteur (Ligne 3 de l'Algorithme 32)

$$bel(\mathbf{x}_1) = \eta p(Z_1 = \text{pièce_2} | \mathbf{x}_1) \overline{bel}(\mathbf{x}_1).$$

En remplaçant \mathbf{x}_1 par les deux valeurs possibles de X_1 on obtient :

$$\begin{aligned}bel(X_1 = \text{pièce_1}) &= \eta p(Z_1 = \text{pièce_2} | X_1 = \text{pièce_1}) \overline{bel}(X_1 = \text{pièce_1}) \\ &= \eta 0.4 \times 0.5 = \eta 0.2\end{aligned}$$

et

$$\begin{aligned}bel(X_1 = \text{pièce_2}) &= \eta p(Z_1 = \text{pièce_2} | X_1 = \text{pièce_2}) \overline{bel}(X_1 = \text{pièce_2}) \\ &= \eta 0.8 \times 0.5 = \eta 0.4\end{aligned}$$

Il est alors possible de calculer la constante de normalisation η

$$\eta = \frac{1}{0.2 + 0.4} = \frac{1}{0.6}.$$

Finalement on obtient les probabilités suivantes :

$$\begin{aligned} \text{bel}(X_1 = \text{pièce_1}) &= \frac{1}{3} \approx 0.33, \\ \text{bel}(X_1 = \text{pièce_2}) &= \frac{2}{3} \approx 0.67. \end{aligned}$$

Ces calculs peuvent facilement être étendus au pas de temps suivant. En supposant que $\mathbf{u}_2 = \text{reculer}$ et que $\mathbf{z}_2 = \text{pièce_1}$, on peut calculer les croyances suivantes :

$$\begin{aligned} \text{bel}(X_1 = \text{pièce_1}) &= \frac{39}{41} \approx 0.95 \\ \text{bel}(X_1 = \text{pièce_2}) &= \frac{2}{41} \approx 0.05 \end{aligned}$$

À l'instant $t = 2$ le robot a une probabilité de 0.95 d'être dans la pièce 1.

Nous allons maintenant présenter différents filtres classiquement utilisés pour la localisation en robotique mobile.

2 Filtres Gaussiens

2.1 Présentation générale

Les filtres Gaussiens correspondent aux premières implémentations des filtres Bayésiens pour des états continus. Encore aujourd'hui ils représentent une grande partie des implémentations utilisées.

Les filtres Gaussiens partagent l'idée que les croyances correspondent à des distributions Gaussiennes (normales) multivariées. L'équation d'une Gaussienne multivariée (Équation B.4) est redonnée ici :

$$p(\mathbf{x}) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu})\right). \quad (\text{B.46})$$

On peut remarquer que cette distribution est complètement caractérisée par deux variables : le vecteur des moyennes $\boldsymbol{\mu}$ et la matrice de covariance Σ . $\boldsymbol{\mu}$ est un vecteur de même dimension que l'état \mathbf{x} , et comme précisé précédemment, la matrice de covariance est une matrice symétrique définie semi-positive. La dimension de Σ correspond à la dimension de \mathbf{x} au carré.

La représentation d'une Gaussienne par sa moyenne et sa covariance est appelée *représentation par moments*. En effet la moyenne et la covariance correspondent respectivement aux premier et deuxième moments d'une densité de probabilité. On pourra noter qu'en ce qui concerne les distributions normales, tous les autres moments, mis à part le premier et le deuxième, sont nuls. On peut noter qu'il existe d'autres types de représentations (représentation canonique par exemple).

Moment

En statistique le moment d'une variable aléatoire X est un indicateur sur la dispersion de cette variable. On note le moment d'ordre k de la variable aléatoire X comme étant l'espérance de la variable X^k :

$$E [X^k] .$$

On peut remarquer que le moment d'ordre 1 correspond à l'espérance de la variable aléatoire (la moyenne de sa densité de probabilité).

Le fait de restreindre les croyances à des distributions normales a des influences conséquentes sur les problèmes considérés. En effet, les Gaussiennes sont unimodales, et du coup ne possèdent qu'un seul maximum. Ce qui caractérise très bien le fait que l'état suivant soit relativement proche de l'état précédent mais permet difficilement de suivre plusieurs hypothèses distinctes. En d'autres termes, les filtres Gaussiens sont particulièrement bien adaptés au problème de suivi de posture (quand la posture courante du robot est proche de sa posture précédente) mais ne conviennent pas pour le problème de localisation globale (quand plusieurs postures distinctes sont probables, dues aux symétries de l'environnement par exemple).

Fonction unimodale

Une fonction unimodale sur \mathbb{I} correspond à une fonction f définie sur \mathbb{I} de telle sorte que

- f n'admet qu'un seul maximum sur \mathbb{I} ,
- f est strictement croissante avant son maximum et est strictement décroissante après.

Dans cette section nous allons présenter deux filtres Gaussiens particuliers : le très utilisé *filtre de Kalman* (KF²) pour les systèmes linéaires, et son extension aux systèmes non-linéaires le *filtre de Kalman étendu* (EKF³).

2. Kalman Filter

3. Extended Kalman Filter

Système linéaire/non-linéaire

Un système linéaire est un système qui ne contient que des équations linéaires. Une équation linéaire est une équation qui peut s'écrire de la forme

$$\mathbf{y} = M \cdot \mathbf{x},$$

avec M une matrice de coefficients connue.

Un système non-linéaire est un système qui contient au moins une équation non-linéaire. Une équation non-linéaire est une équation qui ne peut pas s'écrire de la forme $\mathbf{y} = M \cdot \mathbf{x}$.

Par exemple

$$y = 2x \quad \text{et} \quad y = 3x_1 + x_2 = \begin{pmatrix} 3 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

sont des équations linéaires, alors que

$$y = x^2 \quad \text{et} \quad y = \cos(x)$$

sont des équations non-linéaires.

2.2 Filtre de Kalman

2.2.1 Présentation

Le filtre de Kalman est l'une des techniques les plus utilisées pour implémenter les filtres Bayésiens. Il permet de calculer la croyance d'états continus. Ce filtre est basé sur la caractérisation de variables aléatoires Gaussiennes par les moments de leurs densités de probabilités : à l'instant t la croyance est représentée par sa moyenne μ_t et sa covariance Σ_t .

Le filtre de Kalman peut être utilisé si les trois conditions suivantes sont réunies.

- La densité de probabilité de l'état suivant $p(\mathbf{x}_t \mid \mathbf{u}_t, \mathbf{x}_{t-1})$ doit être une fonction linéaire. Ce qui se traduit par

$$\mathbf{x}_t = A_t \mathbf{x}_{t-1} + B_t \mathbf{u}_t + \varepsilon_t, \tag{B.47}$$

avec A_t et B_t deux matrices connues, et ε_t une variable aléatoire Gaussienne (de moyenne nulle et de covariance R_t) qui modélise le bruit sur le changement

d'état. On notera que les vecteurs \mathbf{x} et \mathbf{u} sont des vecteurs colonnes de la forme

$$\mathbf{x}_t = \begin{pmatrix} x_{1,t} \\ x_{2,t} \\ \vdots \\ x_{n,t} \end{pmatrix} \quad \text{et} \quad \mathbf{u}_t = \begin{pmatrix} u_{1,t} \\ u_{2,t} \\ \vdots \\ u_{m,t} \end{pmatrix}. \quad (\text{B.48})$$

A_t est alors une matrice carrée de taille $n \times n$, avec n la taille du vecteur \mathbf{x} et B_t est une matrice de taille $n \times m$, m étant la taille du vecteur \mathbf{u} . En considérant $\varepsilon_t \sim \mathcal{N}(0, R_t)$ on peut écrire $p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)$ comme étant

$$p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) = \det(2\pi R_t)^{-\frac{1}{2}} \exp \left(-\frac{1}{2} (\mathbf{x}_t - A_t \mathbf{x}_{t-1} - B_t \mathbf{u}_t)^T R_t^{-1} (\mathbf{x}_t - A_t \mathbf{x}_{t-1} - B_t \mathbf{u}_t) \right). \quad (\text{B.49})$$

Ce résultat correspond à l'injection de l'Équation B.47 dans l'Équation B.46 en tenant compte de la règle de la transformation linéaire d'une Gaussienne (Équation B.56) et du fait que les variables \mathbf{x}_{t-1} et \mathbf{u}_t soient considérées constantes car fixées par la probabilité $p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)$ (probabilité d'avoir x_t sachant \mathbf{x}_{t-1} et \mathbf{u}_t). En d'autres termes, le calcul de $p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)$ consiste à ajouter la constante $A_t \mathbf{x}_{t-1} + B_t \mathbf{u}_t$ à la Gaussienne multivariée ε_t .

- La densité de probabilité des mesures $p(\mathbf{z}_t | \mathbf{x}_t)$ doit aussi être une fonction linéaire :

$$\mathbf{z}_t = C_t \mathbf{x}_t + \delta_t, \quad (\text{B.50})$$

avec C_t une matrice connue et δ_t une variable aléatoire Gaussienne (de moyenne nulle et de covariance Q_t) symbolisant le bruit des mesures. C_t est une matrice de taille $k \times n$ avec k la dimension du vecteur de mesures \mathbf{z} . Sur le modèle de l'Équation B.49, avec $\delta_t \sim \mathcal{N}(0, Q_t)$ on peut définir $p(\mathbf{z}_t | \mathbf{x}_t)$ comme étant

$$p(\mathbf{z}_t | \mathbf{x}_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \exp \left(-\frac{1}{2} (\mathbf{z}_t - C_t \mathbf{x}_t)^T Q_t^{-1} (\mathbf{z}_t - C_t \mathbf{x}_t) \right). \quad (\text{B.51})$$

- La croyance initiale $bel(\mathbf{x}_0)$ doit suivre une distribution Gaussienne. Soient μ_0 et Σ_0 la moyenne et la covariance de la croyance initiale alors :

$$p(\mathbf{x}_0) = bel(\mathbf{x}_0) = \det(2\pi \Sigma_0)^{-\frac{1}{2}} \exp \left(-\frac{1}{2} (\mathbf{x}_0 - \mu_0)^T \Sigma_0^{-1} (\mathbf{x}_0 - \mu_0) \right) \quad (\text{B.52})$$

2.2.2 Algorithme

L'algorithme du filtre de Kalman est présenté Algorithme 33. Comme précisé précédemment le filtre de Kalman représente la croyance $bel(\mathbf{x}_t)$ à l'instant t par la moyenne μ_t et la covariance Σ_t . L'entrée de l'algorithme correspond à la croyance à

l'instant $t - 1$, représentée par μ_{t-1} et Σ_{t-1} , ainsi que les vecteurs de contrôles \mathbf{u}_t et de mesures \mathbf{z}_t .

Les lignes 1 et 2 correspondent à la phase de prédiction (au calcul de $\overline{bel}(\mathbf{x}_t)$) en calculant $\overline{\mu}_t$ et $\overline{\Sigma}_t$. Les lignes 3 à 5 correspondent à la phase de correction : le vecteur de mesures \mathbf{z}_t est utilisé afin de calculer $bel(\mathbf{x}_t)$ en fonction de $\overline{bel}(\mathbf{x}_t)$. La variable K_t , calculée Ligne 3 est appelée le *gain de Kalman*. Il indique l'importance à accorder aux mesures pour l'estimation du nouvel état. Ligne 5, I correspond à la matrice identité.

Algorithme 33: Filtre de Kalman

Données : $\mu_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t$

- 1 $\overline{\mu}_t = A_t \mu_{t-1} + B_t \mathbf{u}_t;$
- 2 $\overline{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t;$
- 3 $K_t = \overline{\Sigma}_t C_t^T (C_t \overline{\Sigma}_t C_t^T + Q_t)^{-1};$
- 4 $\mu_t = \overline{\mu}_t + K_t (\mathbf{z}_t - C_t \overline{\mu}_t);$
- 5 $\Sigma_t = (I - K_t C_t) \overline{\Sigma}_t;$

Résultat : μ_t, Σ_t .

Matrice identité

Une matrice identité est une matrice carrée (même nombre de lignes que de colonnes) dont les coefficients sont égaux à 1 sur la diagonale Nord-Ouest/Sud-Est et à 0 partout ailleurs :

$$I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}.$$

2.3 Filtre de Kalman étendu

2.3.1 Présentation

La supposition d'un système linéaire n'est pas réellement applicable au problème de localisation en robotique mobile. En effet les modèles dynamiques des robots sont pour la plupart des modèles non-linéaires.

Le filtre de Kalman étendu permet de s'affranchir de la supposition de linéarité. Supposons maintenant que l'évolution de l'état ainsi que les mesures soient modélisés

par des fonctions non-linéaires, respectivement g et h :

$$\mathbf{x}_t = g(\mathbf{u}_t, \mathbf{x}_{t-1}) + \varepsilon_t, \quad (\text{B.53})$$

$$\mathbf{z}_t = h(\mathbf{x}_t) + \delta_t. \quad (\text{B.54})$$

On obtient alors une généralisation des Équations B.47 et B.50. Seulement les non-linéarités des fonctions g et h ne permettent plus de garantir le fait que la croyance soit distribuée selon une densité Gaussienne, comme c'était le cas pour le cas linéaire. Il s'avère même qu'en pratique il soit parfois impossible de calculer la croyance pour des fonctions g et h non-linéaires. L'idée du filtre de Kalman étendu est alors d'approximer les fonctions non-linéaires par des fonctions linéaires. En d'autres termes, l'EKF est basé sur le même principe que le filtre de Kalman à la différence que cette fois la croyance n'est pas calculée exactement mais est approximée.

Parenthèse sur la transformée d'une loi Gaussienne

- Transformée linéaire d'une variable Gaussienne (Figure B.4a). Considérant une variable aléatoire $X \sim \mathcal{N}(\mu, \sigma^2)$ et une variable aléatoire Y de sorte que $Y = aX + b$, avec a et b deux nombres réels, alors

$$Y \sim \mathcal{N}(a\mu + b, a^2\sigma^2). \quad (\text{B.55})$$

Cette définition peut s'étendre aux vecteurs, soient une variable aléatoire $X \sim \mathcal{N}(\mu, \Sigma)$ et une variable aléatoire Y de sorte que $Y = AX + B$, avec A et B deux matrices, alors

$$Y \sim \mathcal{N}(A\mu + B, A\Sigma A^T). \quad (\text{B.56})$$

- Transformée non-linéaire d'une variable aléatoire (dans le cas simple d'une fonction monotone et dérivable). Soient X une variable aléatoire distribuée selon une densité f_X et une variable aléatoire $Y = g(X)$, où g est une fonction non-linéaire, strictement monotone et dérivable, de dérivée qui ne s'annule nulle part. Alors la densité f_Y de la variable Y correspond à

$$f_Y(y) = \left| \frac{1}{g'(g^{-1}(y))} \right| f_X(g^{-1}(y)),$$

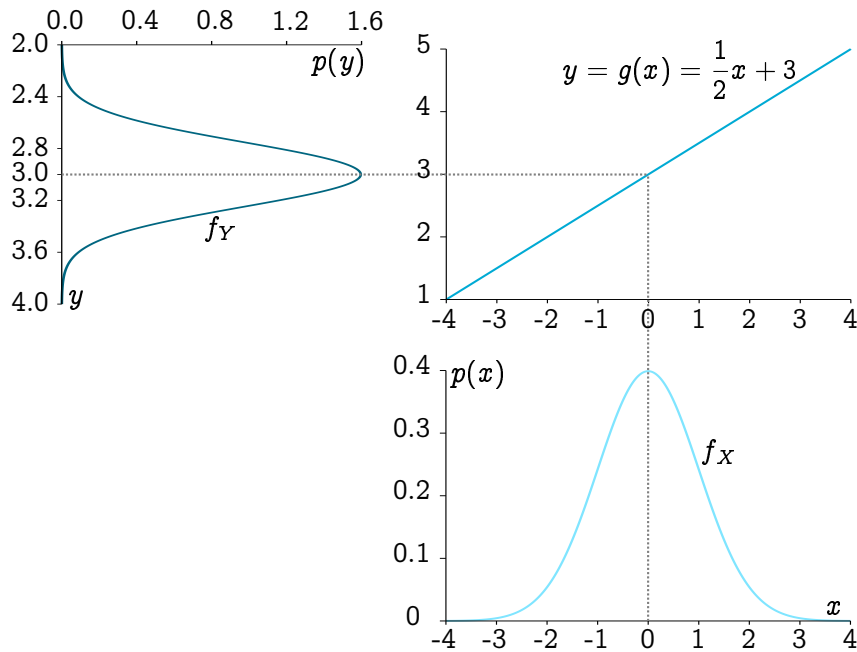
avec g' et g^{-1} respectivement la dérivée et la réciproque de la fonction g . Considérons par exemple

$$Y = g(X) = \exp(X).$$

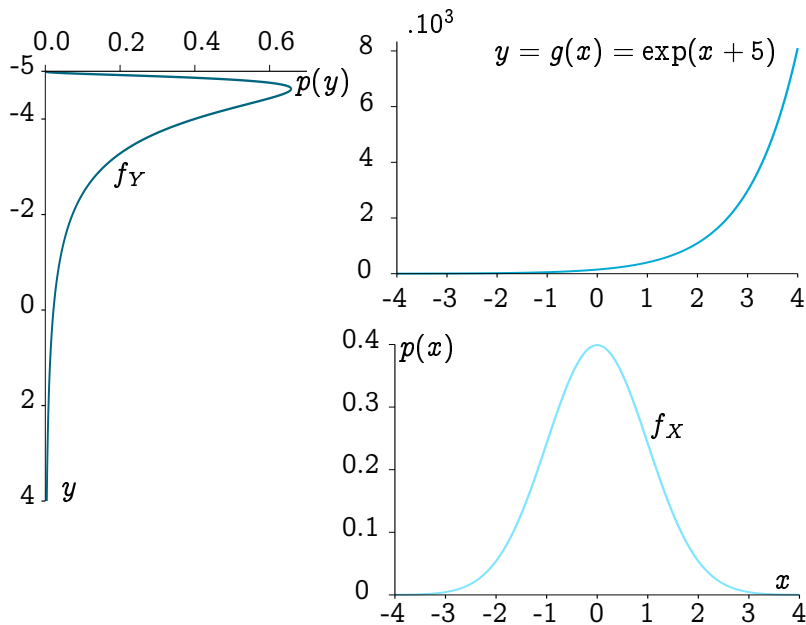
On peut calculer $g'(y) = \exp(y)$ et $g^{-1}(y) = \log(y)$. f_Y peut ainsi s'écrire de la forme

$$f_Y(y) = \left| \frac{1}{\exp(\log(y))} \right| f_X(\log(y)) = \left| \frac{1}{y} \right| f_X(\log(y)).$$

On remarque que f_Y n'est plus une Gaussienne (Figure B.4b).



(a) Soient $X \sim f_X$ et $Y \sim f_Y$ avec $Y = g(X) = \frac{1}{2}X + 3$. Si $X \sim \mathcal{N}(0, 1)$ alors $Y \sim \mathcal{N}(3, \frac{1}{4})$.



(b) Soient $X \sim f_X$ et $Y \sim f_Y$ avec $Y = g(X) = \exp(x + 5)$. On remarque que si f_X est une Gaussienne, f_Y n'est pas une Gaussienne pour autant.

Figure B.4 – Exemple de transformations linéaire et non-linéaire d'une Gaussienne.

2 .3.2 Algorithme

Le concept clé du filtre de Kalman étendu correspond à l'étape de *linéarisation*. Supposons une fonction g non linéaire. Le résultat de cette fonction appliquée à une Gaussienne ne sera classiquement pas Gaussienne : les non-linéarités de la fonction vont "déformer" la forme de la Gaussienne (Figure B.4b). Pour éviter cette déformation, on va approximer g par une fonction linéaire, en faisant en sorte que cette nouvelle fonction linéaire soit tangente à la fonction g pour la moyenne de la Gaussienne.

Il existe plusieurs techniques de linéarisations. Les filtres de Kalman étendus utilisent une méthode appelée le *développement de Taylor* du premier ordre (Figure B.5).

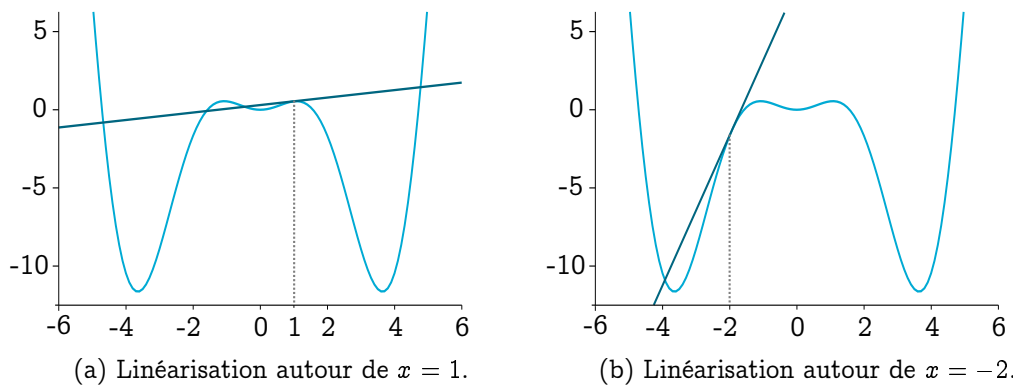


Figure B.5 – Exemple de linéarisation. On considère ici la fonction $f(x) = x^2 \cos(x)$. Le développement de Taylor d'ordre 1 donne $f(x + h) \approx x^2 \cos(x) + h(2x \cos(x) - x^2 \sin(x))$. La Figure B.5a illustre cette linéarisation pour $x = 1$ et la Figure B.5b pour $x = -2$. On remarque que le résultat de la linéarisation est dépendant du point autour duquel on linéarise.

Développement de Taylor

Le développement de Taylor permet l'approximation d'une fonction, plusieurs fois dérivable au voisinage d'un point, par un polynôme dont les coefficients dépendent uniquement des dérivées de la fonction en ce point.

Soit f une fonction n fois dérivable sur \mathbb{I} , et une constante réelle h telle que $x + h \in \mathbb{I}$, alors

$$f(x + h) = f(x) + hf'(x) + \frac{h^2}{2!}f^{(2)}(x) + \dots + \frac{h^n}{n!}f^{(n)}(x) + h^n\epsilon(h),$$

avec $\epsilon(h)$ une fonction qui tend vers 0 quand h tend vers 0, $f^{(n)}$ la $n^{\text{ième}}$ dérivée de f et $n! = 1 \times 2 \times \dots \times (n - 1) \times n$.

D'après le développement de Taylor (au premier ordre) :

– Soit une fonction $f : \mathbb{R} \rightarrow \mathbb{R}$, une variable $x \in \mathbb{R}$ et une constante $h \in \mathbb{R}$

$$f(x + h) \approx f(x) + hf'(x). \quad (\text{B.57})$$

– Soit une fonction $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, une variable $\mathbf{x} \in \mathbb{R}^n$ et une constante $\mathbf{h} \in \mathbb{R}^n$

$$\mathbf{f}(\mathbf{x} + \mathbf{h}) \approx \mathbf{f}(\mathbf{x}) + \mathbf{h} \nabla_{\mathbf{f}}(\mathbf{x}), \quad (\text{B.58})$$

avec $\nabla_{\mathbf{f}}$ le Jacobien de \mathbf{f} .

Jacobien

Soient une variable $\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ et une fonction $\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{pmatrix}$ alors

alors le Jacobien de \mathbf{f} correspond à

$$\nabla_{\mathbf{f}}(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \dots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}.$$

Considérons par exemple la variable $\mathbf{x} = \begin{pmatrix} r \\ \theta \end{pmatrix}$ et la fonction $\mathbf{f}(\mathbf{x}) =$

$$\begin{pmatrix} r \cos(\theta) \\ r \sin(\theta) \end{pmatrix}, \text{ alors } \nabla_{\mathbf{f}}(\mathbf{x}) = \begin{pmatrix} \cos(\theta) & -r \sin(\theta) \\ \sin(\theta) & r \cos(\theta) \end{pmatrix}.$$

L'idée du filtre de Kalman étendu est de linéariser les équations $g(\mathbf{u}_t, \mathbf{x}_{t-1})$ (Équation B.53) et $h(\mathbf{x}_t)$ (Équation B.54) en utilisant le développement de Taylor du

premier ordre. Pour cela il faut choisir le point autour duquel on va linéariser (la variable x de l'Équation B.57 ou encore \mathbf{x} de l'Équation B.58). La logique veut que l'on choisisse l'état le plus probable pour \mathbf{x}_{t-1} . L'état précédent étant caractérisé par une Gaussienne, la valeur la plus probable correspond à la moyenne de cette Gaussienne μ_{t-1} . On décide donc de linéariser les fonction g et h autour de la moyenne μ_{t-1} (Figure B.6). D'après B.58 on a

$$g(\mathbf{x}_{t-1}, \mathbf{u}_t) \approx g(\mu_{t-1}, \mathbf{u}_t) + (\mathbf{x}_{t-1} - \mu_{t-1}) \times \nabla g(\mathbf{x}_{t-1}, \mathbf{u}_t), \quad (\text{B.59})$$

avec $(\mathbf{x}_{t-1} - \mu_{t-1})$ correspondant à la constante h de l'Équation B.58. Pour la suite, par soucis de lisibilité, on notera $G_t := \nabla g(\mathbf{x}_{t-1}, \mathbf{u}_t)$:

$$g(\mathbf{x}_{t-1}, \mathbf{u}_t) \approx g(\mu_{t-1}, \mathbf{u}_t) + G_t(\mathbf{x}_{t-1} - \mu_{t-1}). \quad (\text{B.60})$$

On dispose maintenant de l'équation d'état linéarisée suivante (Équations B.53 et B.60)

$$\mathbf{x}_t \approx g(\mu_{t-1}, \mathbf{u}_t) + G_t(\mathbf{x}_{t-1} - \mu_{t-1}) + \varepsilon_t. \quad (\text{B.61})$$

Sur le même principe que pour l'Équation B.49, avec $\varepsilon_t \sim \mathcal{N}(0, R_t)$ on peut définir

$$p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) \approx \det(2\pi R_t)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{x}_t - g(\mu_{t-1}, \mathbf{u}_t) - G_t(\mathbf{x}_{t-1} - \mu_{t-1}))^T R_t^{-1}(\mathbf{x}_t - g(\mu_{t-1}, \mathbf{u}_t) - G_t(\mathbf{x}_{t-1} - \mu_{t-1}))\right). \quad (\text{B.62})$$

La linéarisation de l'équation h se fait sur le même principe que pour la linéarisation de g . La différence c'est que cette fois, on va linéariser autour de $\bar{\mu}_t$, la valeur la plus probable prédite pour l'état \mathbf{x}_t (la fonction h est utilisée pour la phase de correction, à l'instant t). On a donc

$$h(\mathbf{x}_t) \approx h(\bar{\mu}_t) + (\mathbf{x}_t - \bar{\mu}_t) \nabla h(\bar{\mu}_t). \quad (\text{B.63})$$

En notant $H_t := \nabla h(\bar{\mu}_t)$, on obtient l'équation de mesures linéaire suivante

$$\mathbf{z}_t \approx h(\bar{\mu}_t) + H_t(\mathbf{x}_t - \bar{\mu}_t) + \delta_t. \quad (\text{B.64})$$

On peut finalement s'intéresser à $p(\mathbf{z}_t | \mathbf{x}_t)$, en considérant $\delta_t \sim \mathcal{N}(0, Q_t)$,

$$p(\mathbf{z}_t | \mathbf{x}_t) \approx \det(2\pi Q_t)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{z}_t - h(\bar{\mu}_t) - H_t(\mathbf{x}_t - \bar{\mu}_t))^T Q_t^{-1}(\mathbf{z}_t - h(\bar{\mu}_t) - H_t(\mathbf{x}_t - \bar{\mu}_t))\right). \quad (\text{B.65})$$

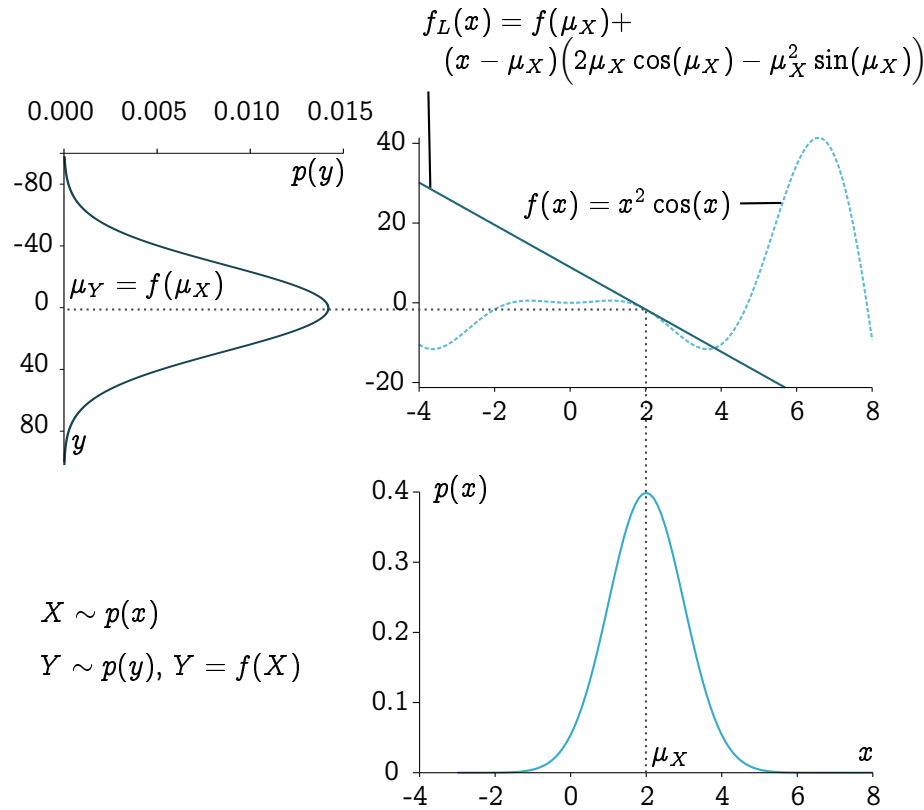


Figure B.6 – Principe de la linéarisation pour le filtre de Kalman étendu. Soient une variable aléatoire $X \sim \mathcal{N}(\mu_X, \sigma_X^2)$ et une variable aléatoire $Y = f(X)$, avec $f(x) = x^2 \cos(x)$. f n'étant pas linéaire, la densité de Y ne sera classiquement pas une Gaussienne. L'idée du filtre de Kalman étendu est alors de linéariser f autour de μ_X (la valeur la plus probable de X), ce qui donne la fonction f_L , puis d'approximer $Y \approx f_L(X)$. La fonction f_L étant linéaire, $p(y)$ devient alors une Gaussienne de moyenne $\mu_Y = f(\mu_X)$ et, sur cet exemple, de covariance $\sigma_Y^2 = \sigma_X^2 (2\mu_X \cos(\mu_X) - \mu_X^2 \sin(\mu_X))^2$.

Ce qui nous amène à l'algorithme du filtre de Kalman étendu, présenté Algorithme 34. On peut remarquer que cet algorithme est très similaire à celui du filtre de Kalman classique : on a la phase de prédiction (lignes 1 et 2), le calcul du gain de Kalman ligne 3 et enfin la phase de correction (lignes 4 et 5). La principale différence entre les deux filtres c'est que pour le filtre de Kalman étendu le système est approximé par un système linéaire.

Algorithme 34: Filtre de Kalman étendu

Données : $\mu_{t-1}, \Sigma_{t-1}, \mathbf{u}_t, \mathbf{z}_t$

- 1 $\bar{\mu}_t = g(\mu_{t-1}, \mathbf{u}_t)$;
- 2 $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$;
- 3 $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$;
- 4 $\mu_t = \bar{\mu}_t + K_t (\mathbf{z}_t - h(\bar{\mu}_t))$;
- 5 $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$;

Résultat : μ_t, Σ_t .

3 Filtres Non-paramétriques

3.1 Présentation générale

Les filtres non-paramétriques représentent une alternative populaire aux filtres Gaussiens. À l'inverse de ces derniers, les filtres non-paramétriques ne caractérisent pas l'état précédent à l'aide d'une forme fixe (une Gaussienne par exemple). Ils utilisent à la place un ensemble fini de valeurs, chacune correspondante à une région de l'espace d'état.

Comme précisé précédemment, les filtres Gaussiens sont particulièrement bien adaptés au problème de suivi de posture en robotique mobile. Cependant ils deviennent inappropriés quand la posture initiale du robot est inconnue (problème de localisation globale). Les filtres non-paramétriques deviennent alors une alternative intéressante. En effet ces derniers sont particulièrement bien adaptés pour la représentation de croyances multimodales complexes. C'est pourquoi ils sont classiquement utilisés pour les problèmes de localisation globale et de kidnapping.

Il existe plusieurs types de filtres non-paramétriques, on s'intéressera ici au *filtre à particules*.

3.2 Filtre à particules

3.2.1 Algorithme basique

Le filtre à particules est une implémentation non-paramétrique du filtre Bayésien. Il caractérise la croyance par un ensemble fini d'états. L'idée principale du filtre à particules est de représenter la croyance $bel(x_t)$ par un ensemble d'états aléatoires (particules) répartis selon la croyance. La Figure B.7 illustre la caractérisation d'une densité à l'aide de particules. Au lieu de caractériser la densité par une fonction paramétrique (Équation B.3 pour une Gaussienne par exemple), les filtres à particules utilisent un ensemble de particules. Une telle représentation est approximative mais a l'avantage de pouvoir représenter des distributions plus complexes.

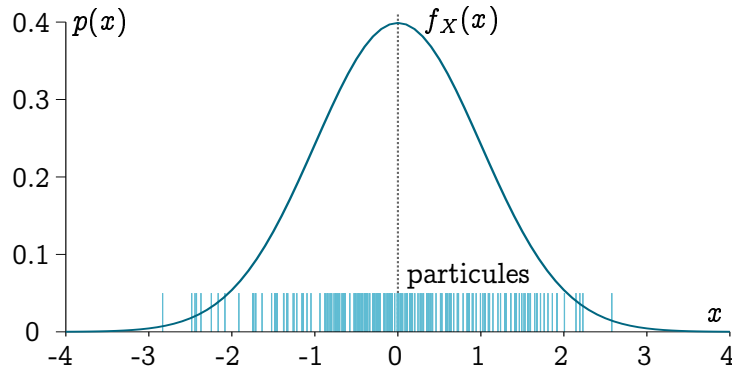


Figure B.7 – Exemple d’approximation à l’aide de particules. Ici on veut caractériser f_X , une Gaussienne de moyenne nulle et de covariance $\sigma^2 = 1$, par un ensemble de particules. Une particule représente une valeur de x possible. Les particules (200 sur cet exemple) sont réparties aléatoirement selon la densité f_X (on peut remarquer qu’il y a plus de particules autour de la moyenne qu’aux extrémités de f_X).

L’ensemble des particules à l’instant t est noté

$$\mathcal{X}_t := \{\mathbf{x}_t^{[1]}, \dots, \mathbf{x}_t^{[m]}, \dots, \mathbf{x}_t^{[M]}\}, \quad (\text{B.66})$$

avec M le nombre de particules. Chaque particule $\mathbf{x}_t^{[m]}$, $1 \leq m \leq M$, correspond à un état probable à l’instant t . Il arrive que le nombre de particules (généralement grand) soit fonction du temps ou encore relié à la croyance $bel(\mathbf{x}_t)$.

Le filtre à particules étant une implémentation particulière du filtre Bayésien, il construit la croyance de l’état courant $bel(\mathbf{x}_t)$ en fonction de la croyance de l’état précédent $bel(\mathbf{x}_{t-1})$. La croyance étant caractérisée par un ensemble de particules, cela revient à calculer un ensemble \mathcal{X}_t en fonction de l’ensemble \mathcal{X}_{t-1} . L’Algorithme 36 présente l’algorithme basique d’un filtre à particules :

- Ligne 3, on actualise la particule $\mathbf{x}_{t-1}^{[m]}$ à l’aide du contrôle \mathbf{u}_t . Cette étape correspond à la phase de prédiction. À la fin de la première boucle for, l’ensemble $\overline{\mathcal{X}}_t$ caractérise $\overline{bel}(t)$.
- Ligne 4, on calcule un *facteur d’importance* (*poids*) pour chaque particules. Le poids de la $m^{\text{ième}}$ particule, noté $w_t^{[m]}$, correspond à la probabilité d’avoir les mesures \mathbf{z}_t depuis l’état $\mathbf{x}_t^{[m]}$. Le facteur d’importance permet de prendre en compte les mesures dans l’ensemble de particules. On peut voir cette étape comme la phase de correction. On notera que les poids des particules sont normalisés de façon à ce que la somme de tous les poids soit égale à 1.
- Les lignes 8 à 10 correspondent à la phase de *ré-échantillonnage*. Durant cette phase l’algorithme crée un nouvel ensemble de particules \mathcal{X}_t en fonction de la prédiction corrigée $\overline{\mathcal{X}}_t$. L’idée de ce ré-échantillonnage est de supprimer les particules de $\overline{\mathcal{X}}_t$ avec un faible facteur d’importance, et de dupliquer celles qui ont un fort facteur d’importance. Pour cela on va redistribuer les particules

de $\bar{\mathcal{X}}_t$ en fonction de leur facteur d'importance. En d'autres termes, avant cette étape, les particules sont approximativement distribuées selon $\bar{bel}(\mathbf{x}_t)$ et après le ré-échantillonnage elles sont approximativement distribuées selon $bel(\mathbf{x}_t) = \eta p(\mathbf{z}_t | \mathbf{x}_t^{[m]}) \bar{bel}(t)$. Cette étape a pour résultat de dupliquer et de supprimer certaines particules de $\bar{\mathcal{X}}_t$.

Algorithme 35: Filtre à particules

Données : $\mathcal{X}_{t-1}, \mathbf{u}_t, \mathbf{z}_t$

- 1 $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset;$
- 2 **pour** $m = 1$ à M **faire**
- 3 générer $\mathbf{x}_t^{[m]} \sim p(\mathbf{x}_t | \mathbf{u}_t, \mathbf{x}_{t-1}^{[m]});$
- 4 $w_t^{[m]} = p(\mathbf{z}_t | \mathbf{x}_t^{[m]});$
- 5 $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle \mathbf{x}_t^{[m]}, w_t^{[m]} \rangle;$
- 6 **fin**
- 7 **pour** $m = 1$ à M **faire**
- 8 générer $\mathbf{x}_t^{[i]}$ en fonction de $\bar{\mathcal{X}}_t;$
- 9 $w_t^{[i]} = p(\mathbf{z}_t | \mathbf{x}_t^{[i]});$
- 10 ajouter $\mathbf{x}_t^{[i]}$ à $\mathcal{X}_t;$
- 11 **fin**

Résultat : $\mathcal{X}_t.$

En résumé le filtre à particules classique a trois étapes importantes :

- Premièrement, il génère de nouvelles particules à l'instant t en fonction des particules à l'instant $t - 1$ et des contrôles appliqués entre $t - 1$ et t .
- Ensuite, il affecte un poids à chaque particule en fonction des mesures faites à l'instant t . Ce poids illustre la probabilité de faire les mesures considérant l'état correspondant à la particule.
- Troisième et dernière étape de l'algorithme, il redessine les particules en privilégiant celles qui ont un poids important. L'idée étant de recentrer les particules sur les régions à forte probabilité et de supprimer les particules des régions peu probables. Un algorithme de ré-échantillonnage est présenté Section 3.2.2.

3.2.2 Ré-échantillonnage systématique

Il existe plusieurs méthodes de ré-échantillonnage qui permettent de tenir compte du facteur d'importance des particules. Cependant le ré-échantillonnage systématique correspond à l'une des méthodes les plus utilisées car elle correspond à la plus simple à implémenter [Douc 2005].

L'Algorithme 36 présente la méthode de ré-échantillonnage systématique et la Figure B.8 illustre son fonctionnement en utilisant les données présentées Tableau B.1. L'idée générale est de créer une fonction cumulative des poids c_i au sein de

laquelle plus une particule a un poids important plus elle est représentée. Le Tableau B.1 présente un exemple de fonction cumulative des poids. Ensuite on va utiliser cette fonction afin de générer les nouvelles particules. On notera que la seule donnée aléatoire de cette méthode correspond au décalage d_1 .

Le ré-échantillonnage permet de se recentrer sur les régions les plus probables en éliminant les régions les moins probables pour la posture du robot. On dit alors qu'il permet de limiter la dégénérescence du jeu de particule. Cependant en contre partie il génère un appauvrissement du jeu de particule dans le sens où le résultat du ré-échantillonnage contiendra plusieurs particules identiques, réduisant ainsi la diversité du jeu.

Algorithme 36: Ré-échantillonnage systématique

Données : $\bar{\mathcal{X}}_t, M$

- 1 $\mathcal{X}_t = \emptyset$;
- 2 $c_1 = w_t^{[1]}$;
- 3 **pour** $i = 2$ à M **faire**
- 4 $c_i = c_{i-1} + w_t^{[i]}$;
- 5 **fin**
- 6 $d_1 \sim U(0, \frac{1}{M})$;
- 7 $i = 1$;
- 8 **pour** $j = 1$ à M **faire**
- 9 **tant que** $d_j > c_i$ **faire**
- 10 $i = i + 1$;
- 11 **fin**
- 12 $\mathcal{X}_t = \mathcal{X}_t \cup \langle \mathbf{x}_t^{[i]}, \frac{1}{N} \rangle$;
- 13 $d_{j+1} = d_j + \frac{1}{N}$;
- 14 **fin**

Résultat : \mathcal{X}_t .

i	1	2	3	4	5	6	7	8	9	10
$w_t^{[i]}$	0.025	0.05	0.1	0.15	0.025	0.05	0.2	0.3	0.05	0.05
c_i	0.025	0.075	0.175	0.325	0.350	0.4	0.6	0.9	0.95	1

Tableau B.1 – Exemple de construction d'une fonction cumulative de poids, considérant 10 particules, chacune ayant un poids $w_t^{[i]}$, $i = 1, \dots, 10$.

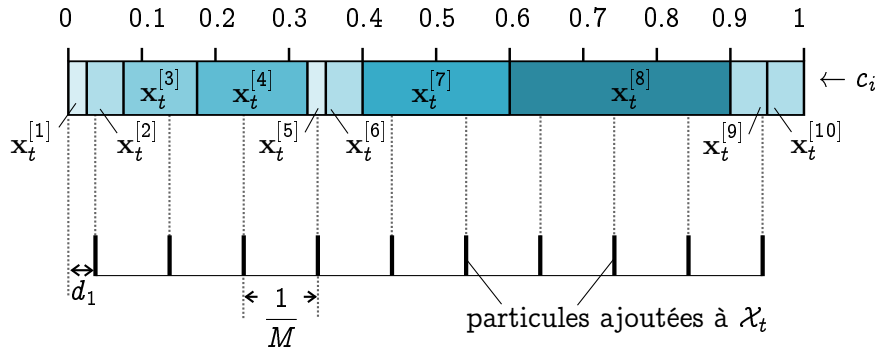


Figure B.8 – Le principe du ré-échantillonnage systématique. En reprenant la fonction cumulative c_i présentée Tableau B.1, on remarque que les particules sont plus ou moins représentées en fonction de leur facteur d'importance (poids). L'idée du ré-échantillonnage systématique est de prendre des particules à intervalles réguliers $\left(\frac{1}{M}\right)$ dans cette fonction cumulative à partir d'un décalage d_1 . Il est donc visible que plus une particule a un facteur d'importance élevé, plus elle a de chance d'être sélectionnée. Sur cet exemple le jeu de particules après ré-échantillonnage correspond à $\mathcal{X}_t = \{x_t^{[2]}, x_t^{[3]}, x_t^{[4]}, x_t^{[5]}, x_t^{[7]}, x_t^{[7]}, x_t^{[8]}, x_t^{[8]}, x_t^{[8]}, x_t^{[9]}\}$.

Outils pour la visibilité

Sommaire

1	Manipulation de segments.	212
1.1	Équation paramétrique d'un segment	212
1.2	Intersection de deux segments.	212
1.3	Segments et polygones convexes	220
2	Quelques propriétés sur le calcul de déterminant	223
2.1	Première proposition	224
2.2	Deuxième proposition	225
2.3	Troisième proposition.	226
3	Quelques démonstrations	229
3.1	Démonstration pour la Proposition 3.6.	229
3.2	Démonstration pour la Proposition 3.7.	231
3.3	Démonstration de la Proposition 3.11	233

Cette annexe présente des outils et démonstrations nécessaires à la visibilité présentée Chapitre 3.

1 Manipulation de segments

1.1 Équation paramétrique d'un segment

Soit un segment $Seg(\mathbf{a}, \mathbf{b})$, $\mathbf{a} = (a_1, a_2) \in \mathbb{R}^2$ et $\mathbf{b} = (b_1, b_2) \in \mathbb{R}^2$. Tout point $\mathbf{s} = (s_1, s_2) \in Seg(\mathbf{a}, \mathbf{b})$ peut s'écrire de la forme

$$\begin{cases} s_1 = (1-t)a_1 + tb_1, \\ s_2 = (1-t)a_2 + tb_2, \\ t \in [0, 1]. \end{cases} \quad (\text{C.1})$$

Cette notation correspond à l'équation paramétrique du segment $Seg(\mathbf{a}, \mathbf{b})$. Elle est utilisée dans certaines démonstrations présentes dans cette annexe.

1.2 Intersection de deux segments

Dans cette section nous présentons un test permettant de tester l'intersection de deux segments. L'objectif est de déterminer si oui ou non il y a intersection. On notera que l'on ne cherche pas ici le point d'intersection, juste à savoir s'il existe.

1.2.1 Position d'un point par rapport à un segment

Soient trois points $\mathbf{a} = (a_1, a_2) \in \mathbb{R}^2$, $\mathbf{b} = (b_1, b_2) \in \mathbb{R}^2$ et $\mathbf{c} = (c_1, c_2) \in \mathbb{R}^2$, on note

$$\begin{aligned} \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= \det \begin{pmatrix} a_1 - b_1 & c_1 - b_1 \\ a_2 - b_2 & c_2 - b_2 \end{pmatrix}, \\ \Rightarrow \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (a_1 - b_1)(c_2 - b_2) - (a_2 - b_2)(c_1 - b_1). \end{aligned} \quad (\text{C.2})$$

Le signe de $\det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b})$ permet de déterminer la *position* du point \mathbf{a} relativement à la droite portée par le vecteur $\overrightarrow{\mathbf{bc}}$. La Figure C.1 illustre les différents cas.

En prenant l'exemple de la Figure C.2 on peut calculer :

$$\begin{aligned} \det(\mathbf{a}_1 - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (4 - 7)(6 - 12) - (4 - 12)(13 - 7), \\ &= 66 > 0. \end{aligned}$$

$$\begin{aligned} \det(\mathbf{a}_2 - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (3 - 7)(6 - 12) - (16 - 12)(13 - 7), \\ &= 0. \end{aligned}$$

$$\begin{aligned} \det(\mathbf{a}_3 - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (15 - 7)(6 - 12) - (9 - 12)(13 - 7), \\ &= -30 < 0. \end{aligned}$$

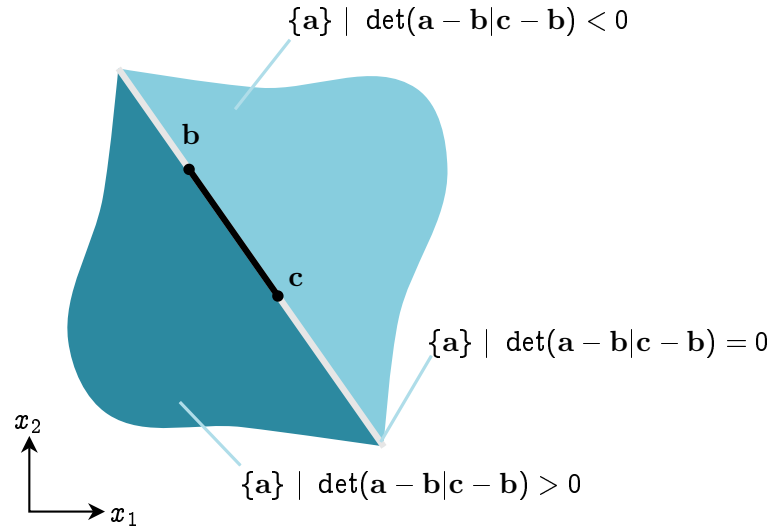


Figure C.1 – Les différents cas pour le signe de $\det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b})$.

Ces résultats sont cohérents avec la Figure C.1.

Remarque C.1 L'équation $\det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b})$ (plus précisément sa valeur absolue) peut s'interpréter comme une mesure de la plus courte distance séparant le point \mathbf{a} de la droite portée par le vecteur $\overrightarrow{\mathbf{bc}}$. En prenant l'exemple Figure C.3 nous avons :

$$\begin{aligned} \det(\mathbf{a}_1 - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (4 - 7)(6 - 12) - (9 - 12)(13 - 7), \\ &= 36. \end{aligned}$$

$$\begin{aligned} \det(\mathbf{a}_2 - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (14 - 7)(6 - 12) - (11 - 12)(13 - 7), \\ &= -36. \end{aligned}$$

$$\begin{aligned} \det(\mathbf{a}_3 - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (11 - 7)(6 - 12) - (2 - 12)(13 - 7), \\ &= 36. \end{aligned}$$

Les trois points \mathbf{a}_1 , \mathbf{a}_2 et \mathbf{a}_3 sont à égale distance de la droite portée par le vecteur $\overrightarrow{\mathbf{bc}}$.

L'Algorithme 37 présente le contracteur associé à la contrainte

$$\zeta \det(\mathbf{x} - \mathbf{a} | \mathbf{b} - \mathbf{a}) \geq 0 \quad (\text{C.3})$$

avec $\mathbf{a} = (a_1, a_2)$ et $\mathbf{b} = (b_1, b_2)$ deux points connus et $\zeta = \{-1, 1\}$. En développant Équation C.3 on obtient

$$x_1 \zeta (b_2 - a_2) - x_2 \zeta (b_1 - a_1) - \zeta (a_2 (b_1 - a_1) - a_1 (b_2 - a_2)) \geq 0 \quad (\text{C.4})$$

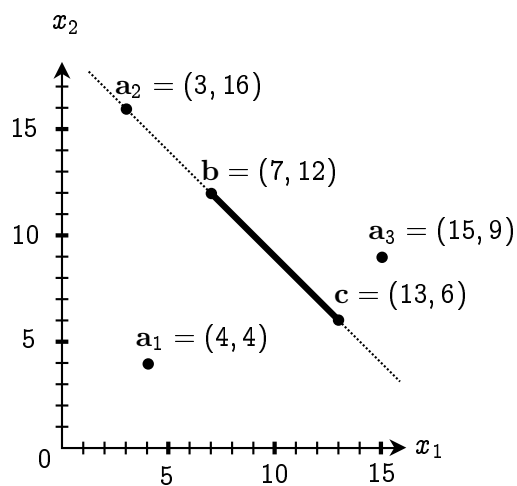


Figure C.2 – Étude de cas.

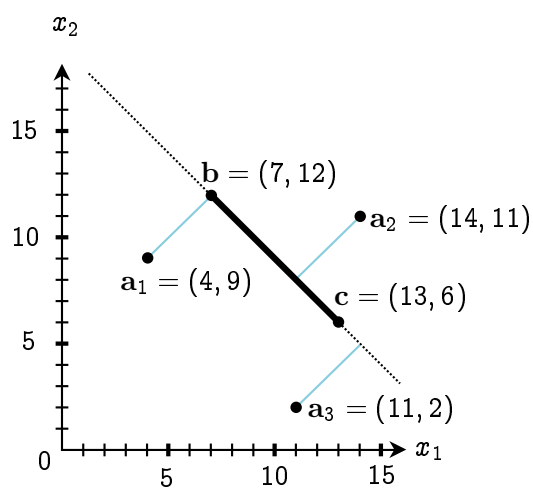


Figure C.3 – Notion de distance.

Algorithme 37: Contracteur $C_{det}([x], \mathbf{a}, \mathbf{b}, \zeta)$

Données : $[x] = ([x_1], [x_2])$, $\mathbf{a} = (a_1, a_2)$, $\mathbf{b} = (b_1, b_2)$, ζ

1 // Variables intermédiaires pour faciliter la lecture

2 $cst_1 = \zeta(b_1 - a_1)$, $cst_2 = \zeta(b_2 - a_2)$, $cst_3 = \zeta(a_2(b_1 - a_1) - a_1(b_2 - a_2))$;

3 // Initialisations

4 $[i_1] = [x_1]cst_2$, $[i_2] = [x_2]cst_1$, $[i_3] = [i_1] - [i_2] + cst_3$;

5 // Contractions

6 $[i_3]^* = [i_3] \cap \mathbb{R}^+$;

7 $[i_1]^* = [i_1] \cap ([i_3]^* - cst_3 + [i_2])$;

8 $[i_2]^* = [i_2] \cap ([i_1]^* + cst_3 - [i_3]^*)$;

9 **si** $cst_2 \neq 0$ **alors**

10 | $[x_1]^* = [x_1] \cap ([i_1]^* / cst_2)$;

11 **sinon**

12 | $[x_1]^* = [x_1]$;

13 **fin**

14 **si** $cst_1 \neq 0$ **alors**

15 | $[x_2]^* = [x_2] \cap ([i_2]^* / cst_1)$;

16 **sinon**

17 | $[x_2]^* = [x_2]$;

18 **fin**

Résultat : $[x]^* = ([x_1]^*, [x_2]^*)$.

1.2.2 Test d'intersection

L'idée générale pour tester l'intersection de deux segments, c'est de tester si les points de chacun des segments sont bien de part et d'autre de l'autre segment [Jaulin 2001b].

Soient quatre points distincts de \mathbb{R}^2 , $\mathbf{a} = (a_1, a_2)$, $\mathbf{b} = (b_1, b_2)$, $\mathbf{c} = (c_1, c_2)$ et $\mathbf{d} = (d_1, d_2)$. On a

$$\begin{aligned} Seg(\mathbf{a}, \mathbf{b}) \cap Seg(\mathbf{c}, \mathbf{d}) = \emptyset &\Leftrightarrow \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \cdot \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) > 0 \vee \\ &\det(\mathbf{c} - \mathbf{a} | \mathbf{b} - \mathbf{a}) \cdot \det(\mathbf{d} - \mathbf{a} | \mathbf{b} - \mathbf{a}) > 0 \vee \\ &[\mathbf{a} \cup \mathbf{b}] \cap [\mathbf{c} \cup \mathbf{d}] = \emptyset, \end{aligned} \quad (\text{C.5})$$

$$\begin{aligned} Seg(\mathbf{a}, \mathbf{b}) \cap Seg(\mathbf{c}, \mathbf{d}) \neq \emptyset &\Leftrightarrow \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \cdot \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq 0 \wedge \\ &\det(\mathbf{c} - \mathbf{a} | \mathbf{b} - \mathbf{a}) \cdot \det(\mathbf{d} - \mathbf{a} | \mathbf{b} - \mathbf{a}) \leq 0 \wedge \\ &[\mathbf{a} \cup \mathbf{b}] \cap [\mathbf{c} \cup \mathbf{d}] \neq \emptyset. \end{aligned} \quad (\text{C.6})$$

On notera qu'il est nécessaire de tester $[\mathbf{a} \cup \mathbf{b}] \cap [\mathbf{c} \cup \mathbf{d}]$ pour éviter le cas particulier présenté Figure C.4.

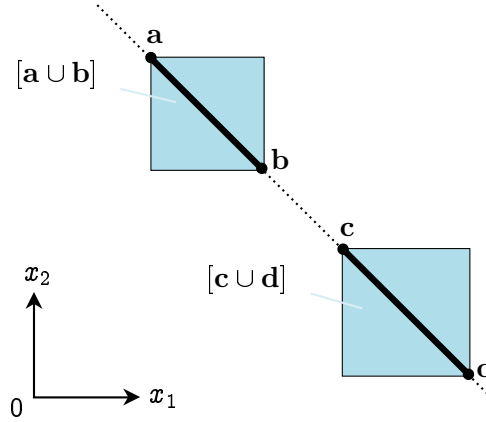


Figure C.4 – Cas particulier pour l'intersection : les deux segments sont alignés. Dans ce cas $\det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \cdot \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) = 0$ et $\det(\mathbf{c} - \mathbf{a} | \mathbf{b} - \mathbf{a}) \cdot \det(\mathbf{d} - \mathbf{a} | \mathbf{b} - \mathbf{a}) = 0$ pourtant les deux segments ne s'intersectent pas.

1 .2.3 Contracteur pour $[\mathbf{a} \cup \mathbf{x}] \cap [\mathbf{c} \cup \mathbf{d}]$

Nous voulons créer le contracteur associé à la contrainte

$$[\mathbf{a} \cup [\mathbf{x}]] \cap [\mathbf{c} \cup \mathbf{d}] \neq \emptyset, \quad (\text{C.7})$$

avec $\mathbf{a} \in \mathbb{R}^2$, $\mathbf{b} \in \mathbb{R}^2$ et $\mathbf{c} \in \mathbb{R}^2$ trois points connus et $[\mathbf{x}] \in \mathbb{I}\mathbb{R}^2$.

En développant l'Équation C.7 on obtient :

$$\begin{aligned} & [\mathbf{a} \cup [\mathbf{x}]] \cap [\mathbf{c} \cup \mathbf{d}] \neq \emptyset \\ & \Leftrightarrow [\min(\mathbf{a}, [\mathbf{x}]), \max(\mathbf{a}, [\mathbf{x}])] \cap [\min(\mathbf{c}, \mathbf{d}), \max(\mathbf{c}, \mathbf{d})] \neq \emptyset \\ & \Leftrightarrow [\max(\min(\mathbf{a}, [\mathbf{x}]), \min(\mathbf{c}, \mathbf{d})), \min(\max(\mathbf{a}, [\mathbf{x}]), \max(\mathbf{c}, \mathbf{d}))] \neq \emptyset \\ & \Leftrightarrow \max(\min(\mathbf{a}, [\mathbf{x}]), \min(\mathbf{c}, \mathbf{d})) \leq \min(\max(\mathbf{a}, [\mathbf{x}]), \max(\mathbf{c}, \mathbf{d})) \\ & \Leftrightarrow \max(\min(\mathbf{a}, [\mathbf{x}]), \min(\mathbf{c}, \mathbf{d})) - \min(\max(\mathbf{a}, [\mathbf{x}]), \max(\mathbf{c}, \mathbf{d})) \leq 0 \\ & \Leftrightarrow \begin{pmatrix} \max(\min(a_1, [x_1]), \min(c_1, d_1)) - \min(\max(a_1, [x_1]), \max(c_1, d_1)) \\ \max(\min(a_2, [x_2]), \min(c_2, d_2)) - \min(\max(a_2, [x_2]), \max(c_2, d_2)) \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ & \Leftrightarrow (\max(\min(a_1, [x_1]), \min(c_1, d_1)) - \min(\max(a_1, [x_1]), \max(c_1, d_1))) \leq 0 \wedge \\ & \quad (\max(\min(a_2, [x_2]), \min(c_2, d_2)) - \min(\max(a_2, [x_2]), \max(c_2, d_2))) \leq 0 \end{aligned} \quad (\text{C.8})$$

On remarque ici la présence des opérateurs $\min()$ et $\max()$. On notera que

$$\min(a, [x]) = \min([x], a) = [\min(a, \underline{x}), \min(a, \bar{x})], \quad (\text{C.9})$$

$$\max(a, [x]) = \max([x], a) = [\max(a, \underline{x}), \max(a, \bar{x})]. \quad (\text{C.10})$$

Il est possible de créer des contracteurs associés à ces opérateurs. L'Algorithme 38 correspond au contracteur associé à la contrainte

$$[y] = \min([x], a), \quad (\text{C.11})$$

et l'Algorithme 39 correspond au contracteur associé à la contrainte

$$[y] = \max([x], a). \quad (\text{C.12})$$

Algorithme 38: $C_{\min}([y], [x], a)$

Données : $[x] = [\underline{x}, \bar{x}], [y] = [\underline{y}, \bar{y}], a$

```

1 // Contraction de [y]
2 [y]* = [y] ∩ min([x], a);
3 // Contraction de [x]
4 si a ∉ [y]* alors
5 | [x]* = [x] ∩ [y]*;
6 sinon
7 | [x]* = [x] ∩ [y*, +∞];
8 fin
Résultat : [x]*, [y]*.
```

Algorithme 39: $C_{\max}([y], [x], a)$

Données : $[x] = [\underline{x}, \bar{x}], [y] = [\underline{y}, \bar{y}], a$

```

1 // Contraction de [y]
2 [y]* = [y] ∩ max([x], a);
3 // Contraction de [x]
4 si a ∉ [y]* alors
5 | [x]* = [x] ∩ [y]*;
6 sinon
7 | [x]* = [x] ∩ [-∞, y*];
8 fin
Résultat : [x]*, [y]*.
```

Les deux algorithmes cités précédemment sont inspirés de [Jaulin 2001c].

En utilisant ces deux contracteurs il devient possible de créer un contracteur associé à la contrainte

$$\max(\min(a, [x]), \min(c, d)) - \min(\max(a, [x]), \max(c, d)) \leq 0. \quad (\text{C.13})$$

Ce nouveau contracteur est présenté Algorithme 40. On remarquera qu'il repose sur une décomposition en contraintes élémentaires et une contraction avant/arrière.

En s'appuyant sur l'Algorithme 40 et sur l'Équation C.8, on peut finalement développer le contracteur (Algorithme 41) associé à la contrainte

$$[a \cup [x]] \cap [c \cup d] \neq \emptyset \quad (\text{C.14})$$

Algorithme 40: $C_{\cap \neq \emptyset}([x], a, c, d)$

Données : $[x] = [\underline{x}, \overline{x}], a, c, d$

- 1 // Initialisations
- 2 $[i_1] = \min(a, [x]);$
- 3 $[i_2] = \max([i_1], \min(c, d));$
- 4 $[i_3] = \max(a, [x]);$
- 5 $[i_4] = \min([i_3], \max(c, d));$
- 6 $[i_5] = [i_2] - [i_4];$
- 7 // Contractions
- 8 $[i_5]^* = [i_5] \cap [-\infty, 0];$
- 9 $[i_2]^* = [i_5]^* + [i_4];$
- 10 $[i_4]^* = [i_2]^* - [i_5];$
- 11 $([i_3]^*, [i_4]^*) = C_{min}([i_4]^*, [i_3], \max(c, d));$
- 12 $([x]^*, [i_3]^*) = C_{max}([i_3]^*, [x], a);$
- 13 $([i_1]^*, [i_2]^*) = C_{max}([i_2]^*, [i_1], \min(c, d));$
- 14 $([x]^*, [i_1]^*) = C_{min}([i_1]^*, [x]^*, a);$

Résultat : $[x]^*$.

Algorithme 41: $C_{\cap \neq \emptyset}([x], \mathbf{a}, \mathbf{c}, \mathbf{d})$

Données : $[x] = ([x_1], [x_2]), \mathbf{a} = (a_1, a_2), \mathbf{c} = (c_1, c_2), \mathbf{d} = (d_1, d_2)$

- 1 // Contraction sur chaque composante
- 2 $[x_1]^* = C_{\cap \neq \emptyset}([x_1], a_1, c_1, d_1);$
- 3 $[x_2]^* = C_{\cap \neq \emptyset}([x_2], a_2, c_2, d_2);$

Résultat : $[x]^* = ([x_1]^*, [x_2]^*)$.

Nous voulons maintenant créer le contracteur associé à la contrainte

$$[\mathbf{a} \cup [\mathbf{x}]] \cap [\mathbf{c} \cup \mathbf{d}] = \emptyset, \quad (\text{C.15})$$

avec $\mathbf{a} \in \mathbb{R}^2$, $\mathbf{b} \in \mathbb{R}^2$ et $\mathbf{c} \in \mathbb{R}^2$ trois points connus et $[\mathbf{x}] \in \mathbb{IR}^2$.

En se basant sur Équation C.8 on en déduit que :

$$\begin{aligned} [\mathbf{a} \cup [\mathbf{x}]] \cap [\mathbf{c} \cup \mathbf{d}] &= \emptyset \\ \Leftrightarrow (\max(\min(a_1, [x_1]), \min(c_1, d_1)) - \min(\max(a_1, [x_1]), \max(c_1, d_1))) &> 0 \wedge \\ (\max(\min(a_2, [x_2]), \min(c_2, d_2)) - \min(\max(a_2, [x_2]), \max(c_2, d_2))) &> 0 \end{aligned} \quad (\text{C.16})$$

Ce qui permet de définir le contracteur (Algorithme 42) associé à la contrainte

$$\max(\min(a, [x]), \min(c, d)) - \min(\max(a, [x]), \max(c, d)) > 0, \quad (\text{C.17})$$

ainsi que le contracteur (Algorithme 43) associé à la contrainte

$$[\mathbf{a} \cup [\mathbf{x}]] \cap [\mathbf{c} \cup \mathbf{d}] = \emptyset. \quad (\text{C.18})$$

Algorithme 42: $C_{\cap=\emptyset}([x], a, c, d)$

Données : $[x] = [\underline{x}, \bar{x}], a, c, d$

```

1 // Initialisations
2  $[i_1] = \min(a, [x]);$ 
3  $[i_2] = \max([i_1], \min(c, d));$ 
4  $[i_3] = \max(a, [x]);$ 
5  $[i_4] = \min([i_3], \max(c, d));$ 
6  $[i_5] = [i_2] - [i_4];$ 
7 // Contractions
8  $[i_5]^* = [i_5] \cap [0, +\infty];$ 
9  $[i_2]^* = [i_5]^* + [i_4];$ 
10  $[i_4]^* = [i_2]^* - [i_5];$ 
11  $([i_3]^*, [i_4]^*) = C_{\min}([i_4]^*, [i_3], \max(c, d));$ 
12  $([x]^*, [i_3]^*) = C_{\max}([i_3]^*, [x], a);$ 
13  $([i_1]^*, [i_2]^*) = C_{\max}([i_2]^*, [i_1], \min(c, d));$ 
14  $([x]^*, [i_1]^*) = C_{\min}([i_1]^*, [x]^*, a);$ 

```

Résultat : $[x]$.

Remarque C.2

$$[\mathbf{x}] \cup \mathbf{a} \cup \mathbf{b} \cap [\mathbf{c} \cup \mathbf{d}] = ([\mathbf{x}] \cup \mathbf{a}) \cap [\mathbf{c} \cup \mathbf{d}] \cup ([\mathbf{x}] \cup \mathbf{b}) \cap [\mathbf{c} \cup \mathbf{d}]. \quad (\text{C.19})$$

Démonstration

$$\begin{aligned}
& \left[[x] \cup a \cup b \right] \cap [c \cup d] \\
&= [\min([x], a, b), \max([x], a, b)] \cap [\min(c, d), \max(c, d)]. \quad (\text{C.20}) \\
& \left[[x] \cup a \right] \cap [c \cup d] \cup \left[[x] \cup b \right] \cap [c \cup d], \\
&= [\min([x], a), \max([x], a)] \cap [\min(c, d), \max(c, d)] \cup \\
&\quad [\min([x], b), \max([x], b)] \cap [\min(c, d), \max(c, d)], \\
&= [\min(c, d), \max(c, d)] \cap \\
&\quad \left([\min([x], a), \max([x], a)] \cup [\min([x], b), \max([x], b)] \right), \\
&= [\min(c, d), \max(c, d)] \cap \\
&\quad [\min(\min([x], b), \min([x], a)), \max(\max([x], b), \max([x], a))], \\
&= [\min(c, d), \max(c, d)] \cap [\min([x], a, b), \max([x], a, b)], \\
&= \left[[x] \cup a \cup b \right] \cap [c \cup d]. \quad (\text{Eq. C.20})
\end{aligned}$$

Algorithme 43: $C_{\cap=\emptyset}([x], a, c, d)$

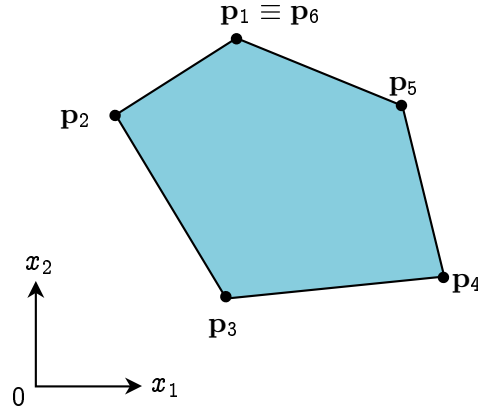
Données : $[x] = ([x_1], [x_2]), a = (a_1, a_2), c = (c_1, c_2), d = (d_1, d_2)$
1 // Contraction sur chaque composante
2 $[x_1]^* = C_{\cap=\emptyset}([x_1], a_1, c_1, d_1);$
3 $[x_2]^* = C_{\cap=\emptyset}([x_2], a_2, c_2, d_2);$
Résultat : $[x]^* = ([x_1]^*, [x_2]^*).$

1.3 Segments et polygones convexes**1.3.1 Définitions**

Cette section présente les polygones tels que considérés dans le Chapitre 3.

Un Polygone (convexe) P correspond ici à un ensemble convexe de \mathbb{R}^2 , délimité par au moins trois segments. On note n_P le nombre de sommets (au moins trois) du polygone P . Les sommets de P sont notés $\mathbf{p}_k, k = 1, \dots, n_P$. Les sommets d'un polygone sont numérotés selon le sens **trigonométrique** tout au long de ce document.

Remarque C.3 *Un polygone P comprenant n_P sommets, est délimité par n_P segments. Afin de simplifier les notations, le premier sommet \mathbf{p}_1 d'un polygone correspond aussi au sommet \mathbf{p}_{n_P+1} . Ce qui permet de délimiter un polygone P*

Figure C.5 – Exemple de polygone P composé de $n_P = 5$ sommets.

par

$$\bigcup_{k=1}^{n_P} \text{Seg}(\mathbf{p}_k, \mathbf{p}_{k+1}), \text{ avec } \mathbf{p}_{n_P+1} \equiv \mathbf{p}_1. \quad (\text{C.21})$$

La Figure C.5 présente un exemple de polygone tel que considéré dans ce manuscrit.

Définition C.1 Soit P un polygone composé de n_P sommets \mathbf{p}_k ($k = 1, \dots, n_P$), on a

$$\forall \mathbf{x} \in P, \forall \mathbf{p}_k \in P, \det(\mathbf{x} - \mathbf{p}_k | \mathbf{p}_{k+1} - \mathbf{p}_k) \leq 0. \quad (\text{C.22})$$

En d'autres termes, l'Équation C.22 décrit tous les points à l'intérieur du polygone (frontières comprises). On notera que cette équation n'est vraie que si les sommets du polygone sont ordonnés dans le sens trigonométrique ! Il est donc possible de définir un polygone P comme

$$P = \{\mathbf{x}_i \in \mathbb{R}^2 \mid \bigwedge_{k=1}^{n_P} \det(\mathbf{x}_i - \mathbf{p}_k | \mathbf{p}_{k+1} - \mathbf{p}_k) \leq 0\}. \quad (\text{C.23})$$

Remarque C.4 Un vecteur d'intervalles à deux dimensions correspond à un polygone particulier. Soit la boîte $[\mathbf{x}] = ([x_1], [x_2]) = ([\underline{x}_1, \overline{x}_1], [\underline{x}_2, \overline{x}_2])$:

$$\begin{aligned} [\mathbf{x}] = \{\mathbf{x}_i \in \mathbb{R}^2 \mid & \det(\mathbf{x}_i - (\underline{x}_1, \underline{x}_2) | (\overline{x}_1, \underline{x}_2) - (\underline{x}_1, \underline{x}_2)) \leq 0 \wedge \\ & \det(\mathbf{x}_i - (\overline{x}_1, \underline{x}_2) | (\overline{x}_1, \overline{x}_2) - (\overline{x}_1, \underline{x}_2)) \leq 0 \wedge \\ & \det(\mathbf{x}_i - (\overline{x}_1, \overline{x}_2) | (\underline{x}_1, \overline{x}_2) - (\overline{x}_1, \overline{x}_2)) \leq 0 \wedge \\ & \det(\mathbf{x}_i - (\underline{x}_1, \overline{x}_2) | (\underline{x}_1, \underline{x}_2) - (\underline{x}_1, \overline{x}_2)) \leq 0\}. \end{aligned} \quad (\text{C.24})$$

La Figure C.6 illustre cette correspondance. L'Algorithme 44 permet de faire la correspondance entre une boîte de \mathbb{R}^2 et un polygone convexe.

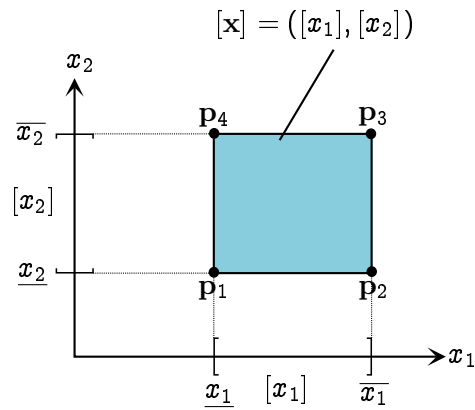


Figure C.6 – Le pavé : un polygone particulier.

Algorithme 44: *Boite2Polygone*($[x]$)

Données : $[x] = ([x_1], [x_2])$

1 $p_1 = (\underline{x}_1, \underline{x}_2)$;

2 $p_2 = (\overline{x}_1, \underline{x}_2)$;

3 $p_3 = (\overline{x}_1, \overline{x}_2)$;

4 $p_4 = (\underline{x}_1, \overline{x}_2)$;

5 $P_x =$ polygone défini par les quatre sommets p_1, p_2, p_3 et p_4 ;

Résultat : P_x .

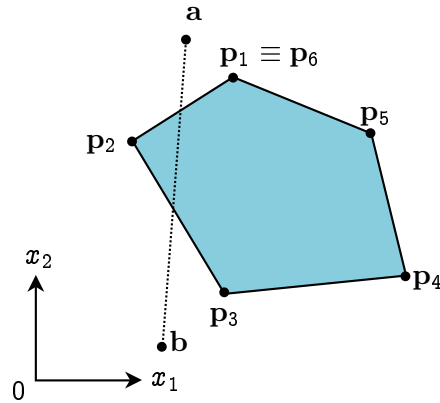


Figure C.7 – La remarque C.5 précise que le segment $Seg(a, b)$ intersecte le polygone seulement s'il intersecte un des segments le délimitant (sachant que a et b n'appartiennent pas au polygone). Ici on a bien $Seg(a, b) \cap Seg(p_1, p_2) \neq \emptyset$ et $Seg(a, b) \cap Seg(p_2, p_3) \neq \emptyset$.

1 .3.2 Intersection entre un polygone et un segment

Remarque C.5 Soient un polygone P composé de n_P sommets et deux points distincts $a \in \mathbb{R}^2$, $b \in \mathbb{R}^2$, avec $a \notin P$ et $b \notin P$. On a

$$Seg(a, b) \cap P \neq \emptyset \Leftrightarrow Seg(a, b) \cap \left(\bigcup_{k=1}^{n_P} Seg(p_k, p_{k+1}) \right) \neq \emptyset. \quad (C.25)$$

La Figure C.7 illustre la Remarque C.5.

2 Quelques propriétés sur le calcul de déterminant

L'objectif de cette section est d'établir trois propriétés sur le calcul de déterminant tel que présenté précédemment. Ces propriétés sont utilisées lors des démonstrations présentes dans cette annexe.

2.1 Première proposition

Proposition C.1 *Soient trois points distincts de \mathbb{R}^2 , $\mathbf{a} = (a_1, a_2)$, $\mathbf{b} = (b_1, b_2)$ et $\mathbf{c} = (c_1, c_2)$. On a*

$$\det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}) = \det(\mathbf{c} - \mathbf{a} | \mathbf{b} - \mathbf{a}) = \det(\mathbf{b} - \mathbf{c} | \mathbf{a} - \mathbf{c}), \quad (\text{C.26})$$

$$\det(\mathbf{a} - \mathbf{c} | \mathbf{b} - \mathbf{c}) = \det(\mathbf{c} - \mathbf{b} | \mathbf{a} - \mathbf{b}) = \det(\mathbf{b} - \mathbf{a} | \mathbf{c} - \mathbf{a}), \quad (\text{C.27})$$

$$\det(\mathbf{a} - \mathbf{c} | \mathbf{b} - \mathbf{c}) = -\det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}). \quad (\text{C.28})$$

Démonstration

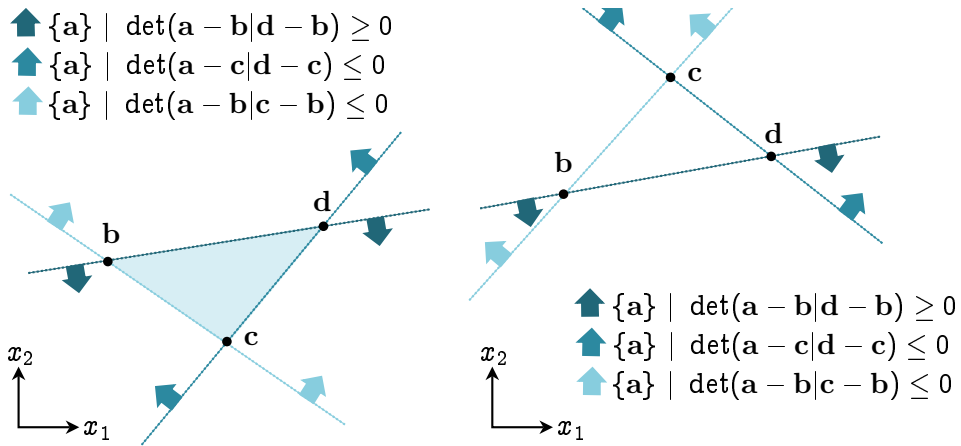
$$\begin{aligned} \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= (a_1 - b_1)(c_2 - b_2) - (a_2 - b_2)(c_1 - b_1) \quad (\text{Eq. C.2}), \\ &= (a_1c_2 - a_1b_2 - b_1c_2 + b_1b_2) - (a_2c_1 - a_2b_1 - b_2c_1 + b_2b_1), \\ \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}) &= a_1c_2 - a_1b_2 - b_1c_2 - a_2c_1 + a_2b_1 + b_2c_1. \\ \det(\mathbf{c} - \mathbf{a} | \mathbf{b} - \mathbf{a}) &= (c_1 - a_1)(b_2 - a_2) - (c_2 - a_2)(b_1 - a_1) \quad (\text{Eq. C.2}), \\ &= (c_1b_2 - c_1a_2 - a_1b_2 + a_1a_2) - (c_2b_1 - c_2a_1 - a_2b_1 + a_2a_1), \\ \det(\mathbf{c} - \mathbf{a} | \mathbf{b} - \mathbf{a}) &= c_1b_2 - c_1a_2 - a_1b_2 - c_2b_1 + c_2a_1 + a_2b_1, \\ \det(\mathbf{c} - \mathbf{a} | \mathbf{b} - \mathbf{a}) &= \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}). \end{aligned}$$

$$\begin{aligned} \det(\mathbf{b} - \mathbf{c} | \mathbf{a} - \mathbf{c}) &= (b_1 - c_1)(a_2 - c_2) - (b_2 - c_2)(a_1 - c_1) \quad (\text{Eq. C.2}), \\ &= (b_1a_2 - b_1c_2 - c_1a_2 + c_1c_2) - (b_2a_1 - b_2c_1 - c_2a_1 + c_2c_1), \\ \det(\mathbf{b} - \mathbf{c} | \mathbf{a} - \mathbf{c}) &= b_1a_2 - b_1c_2 - c_1a_2 - b_2a_1 + b_2c_1 + c_2a_1, \\ \det(\mathbf{b} - \mathbf{c} | \mathbf{a} - \mathbf{c}) &= \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}). \end{aligned}$$

$$\begin{aligned} \det(\mathbf{a} - \mathbf{c} | \mathbf{b} - \mathbf{c}) &= (a_1 - c_1)(b_2 - c_2) - (a_2 - c_2)(b_1 - c_1) \quad (\text{Eq. C.2}), \\ &= (a_1b_2 - a_1c_2 - c_1b_2 + c_1c_2) - (a_2b_1 - a_2c_1 - c_2b_1 + c_2c_1), \\ \det(\mathbf{a} - \mathbf{c} | \mathbf{b} - \mathbf{c}) &= a_1b_2 - a_1c_2 - c_1b_2 - a_2b_1 + a_2c_1 + c_2b_1, \\ \det(\mathbf{a} - \mathbf{c} | \mathbf{b} - \mathbf{c}) &= -\det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}). \end{aligned}$$

$$\begin{aligned} \det(\mathbf{c} - \mathbf{b} | \mathbf{a} - \mathbf{b}) &= (c_1 - b_1)(a_2 - b_2) - (c_1 - b_2)(a_1 - b_1) \quad (\text{Eq. C.2}), \\ &= (c_1a_2 - b_1a_2 - c_1b_2 + b_1b_2) - (c_2a_1 - c_2b_1 - b_2a_1 + b_2b_1), \\ \det(\mathbf{c} - \mathbf{b} | \mathbf{a} - \mathbf{b}) &= c_1a_2 - b_1a_2 - c_1b_2 - c_2a_1 + c_2b_1 + b_2a_1, \\ \det(\mathbf{c} - \mathbf{b} | \mathbf{a} - \mathbf{b}) &= \det(\mathbf{a} - \mathbf{c} | \mathbf{b} - \mathbf{c}). \end{aligned}$$

$$\begin{aligned} \det(\mathbf{b} - \mathbf{a} | \mathbf{c} - \mathbf{a}) &= (b_1 - a_1)(c_2 - a_2) - (b_2 - a_2)(c_1 - a_1) \quad (\text{Eq. C.2}), \\ &= (b_1c_2 - b_1a_2 - a_1c_2 + a_1a_2) - (b_2c_1 - b_2a_1 - a_2c_1 + a_2a_1), \\ \det(\mathbf{b} - \mathbf{a} | \mathbf{c} - \mathbf{a}) &= b_1c_2 - b_1a_2 - a_1c_2 - b_2c_1 + b_2a_1 + a_2c_1, \\ \det(\mathbf{b} - \mathbf{a} | \mathbf{c} - \mathbf{a}) &= \det(\mathbf{a} - \mathbf{c} | \mathbf{b} - \mathbf{c}). \end{aligned}$$



(a) L'intérieur du triangle \mathbf{bcd} correspond aux points \mathbf{a} satisfaisant les trois contraintes. On remarque alors que $\det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq 0$.

(b) Dans ce cas on remarque que $\det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) > 0$, mais il n'y a pas de point \mathbf{a} satisfaisant les trois contraintes.

Figure C.8 – Illustration de la Proposition C.2.

2.2 Deuxième proposition

Proposition C.2 Soient quatre points distincts de \mathbb{R}^2 , $\mathbf{a} = (a_1, a_2)$, $\mathbf{b} = (b_1, b_2)$, $\mathbf{c} = (c_1, c_2)$ et $\mathbf{d} = (d_1, d_2)$. On a

$$\begin{aligned}
 \det(\mathbf{a} - \mathbf{b} | \mathbf{d} - \mathbf{b}) \geq 0 \wedge \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}) \leq 0 \wedge \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq 0 \\
 \Leftrightarrow \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq 0.
 \end{aligned}
 \tag{C.29}$$

La Figure C.8 illustre cette proposition.

Démonstration

$$\begin{aligned}
& \begin{cases} \det(\mathbf{a} - \mathbf{b} | \mathbf{d} - \mathbf{b}) \geq 0 \\ \det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b}) \leq 0 \\ \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq 0 \end{cases} , \\
& \Leftrightarrow \begin{cases} (a_1 - b_1)(d_2 - b_2) - (a_2 - b_2)(d_1 - b_1) \geq 0 \text{ (Eq. C.2)} \\ (a_1 - b_1)(c_2 - b_2) - (a_2 - b_2)(c_1 - b_1) \leq 0 \text{ (Eq. C.2)} \\ (a_1 - c_1)(a_2 - c_2) - (a_2 - c_2)(d_1 - c_1) \leq 0 \text{ (Eq. C.2)} \end{cases} , \\
& \Leftrightarrow \begin{cases} a_1 d_2 - a_1 b_2 - b_1 d_2 - a_2 d_1 + a_2 b_1 + b_2 d_1 \geq 0 \\ a_1 c_2 - a_1 b_2 - b_1 c_2 - a_2 c_1 + a_2 b_1 + b_2 c_1 \leq 0 \\ a_1 d_2 - a_1 c_2 - c_1 d_2 - a_2 d_1 + a_2 c_1 + c_2 d_1 \leq 0 \end{cases} , \\
& \Leftrightarrow \begin{cases} a_1 d_2 - b_1 d_2 - a_2 d_1 + b_2 d_1 \geq a_1 b_2 - a_2 b_1 \\ a_1 c_2 - b_1 c_2 - a_2 c_1 + b_2 c_1 \leq a_1 b_2 - a_2 b_1 \\ a_1 d_2 - a_1 c_2 - c_1 d_2 - a_2 d_1 + a_2 c_1 + c_2 d_1 \leq 0 \end{cases} , \\
& \Leftrightarrow \begin{cases} a_1 d_2 - b_1 d_2 - a_2 d_1 + b_2 d_1 \geq a_1 b_2 - a_2 b_1 \geq a_1 c_2 - b_1 c_2 - a_2 c_1 + b_2 c_1 \\ a_1 d_2 - a_1 c_2 - c_1 d_2 - a_2 d_1 + a_2 c_1 + c_2 d_1 \leq 0 \end{cases} , \\
& \Leftrightarrow \begin{cases} a_1 d_2 - a_2 d_1 - a_1 c_2 + a_2 c_1 \geq b_1 d_2 - b_2 d_1 - b_1 c_2 + b_2 c_1 \\ a_1 d_2 - a_1 c_2 - a_2 d_1 + a_2 c_1 \leq c_1 d_2 - c_2 d_1 \end{cases} , \\
& \Leftrightarrow c_1 d_2 - c_2 d_1 \geq a_1 d_2 - a_2 d_1 - a_1 c_2 + a_2 c_1 \geq b_1 d_2 - b_2 d_1 - b_1 c_2 + b_2 c_1, \\
& \Leftrightarrow c_1 d_2 - c_2 d_1 \geq b_1 d_2 - b_2 d_1 - b_1 c_2 + b_2 c_1, \\
& \Leftrightarrow b_1 d_2 - b_1 c_2 - c_1 d_2 - b_2 d_1 + b_2 c_1 + c_2 d_1 \leq 0, \\
& \Leftrightarrow (b_1 - c_1)(d_1 - c_2) - (b_2 - c_2)(d_1 - c_1) \leq 0, \\
& \Leftrightarrow \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq 0 \text{ (Eq. C.2)}.
\end{aligned}$$

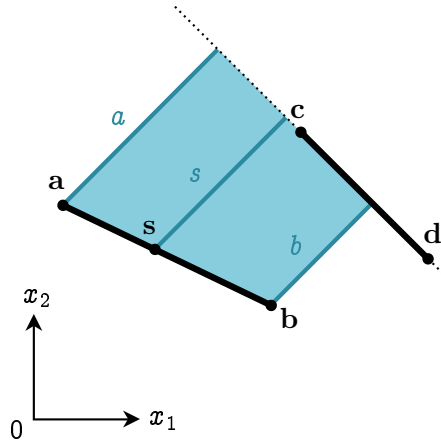
2.3 Troisième proposition

Proposition C.3 Soient un segment $\text{Seg}(\mathbf{a}, \mathbf{b})$, $\mathbf{a} = (a_1, a_2) \in \mathbb{R}^2$, $\mathbf{b} = (b_1, b_2) \in \mathbb{R}^2$, un point $\mathbf{s} \in \text{Seg}(\mathbf{a}, \mathbf{b})$ et deux points $\mathbf{c} = (c_1, c_2) \in \mathbb{R}^2$ et $\mathbf{d} = (d_1, d_2) \in \mathbb{R}^2$, avec $\text{Seg}(\mathbf{a}, \mathbf{b}) \cap \text{Seg}(\mathbf{c}, \mathbf{d}) = \emptyset$. On a

$$\begin{aligned}
& \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \geq \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \\
& \Leftrightarrow \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \geq \det(\mathbf{s} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \geq \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}), \quad (\text{C.30})
\end{aligned}$$

$$\begin{aligned}
& \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \\
& \Leftrightarrow \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq \det(\mathbf{s} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \leq \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}). \quad (\text{C.31})
\end{aligned}$$

En d'autres termes, la Proposition C.3 précise que la distance entre le point \mathbf{s} et la droite portée par le vecteur $\overrightarrow{\mathbf{cd}}$ est comprise entre les distances séparant les deux

Figure C.9 – Illustration de la Proposition C.3 : $a \geq b \Rightarrow a \geq s \geq b$

extrémités du segment ab et cette même droite. C'est ce qui est illustré Figure C.9.

Démonstration

$$\begin{aligned} \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) &= (a_1 - c_1)(d_2 - c_2) - (a_2 - c_2)(d_1 - c_1) \text{ (Eq. C.2),} \\ \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) &= a_1 d_2 - a_1 c_2 - c_1 d_2 - a_2 d_1 + a_2 c_1 + c_2 d_1, \end{aligned} \quad (\text{C.32})$$

$$\begin{aligned} \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) &= (b_1 - c_1)(d_2 - c_2) - (b_2 - c_2)(d_1 - c_1) \text{ (Eq. C.2),} \\ \det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) &= b_1 d_2 - b_1 c_2 - c_1 d_2 - b_2 d_1 + b_2 c_1 + c_2 d_1. \end{aligned} \quad (\text{C.33})$$

$$\det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) - \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) = b_1 d_2 - b_1 c_2 - b_2 d_1 + b_2 c_1 - a_1 d_2 + a_1 c_2 + a_2 d_1 - a_2 c_1. \quad (\text{C.34})$$

$$\begin{aligned} \det(\mathbf{s} - \mathbf{c} | \mathbf{d} - \mathbf{c}) &= (s_1 - c_1)(d_2 - c_2) - (s_2 - c_2)(d_1 - c_1) \text{ (Eq. C.2),} \\ &= s_1 d_2 - s_1 c_2 - c_1 d_2 - s_2 d_1 + s_2 c_1 + c_2 d_1, \\ &= s_1(d_2 - c_2) + s_2(c_1 - d_1) + c_2 d_1 - c_1 d_2, \\ &= \left((1-t)a_1 + tb_1 \right) (d_2 - c_2) + \left((1-t)a_2 + tb_2 \right) (c_1 - d_1) \\ &\quad + c_2 d_1 - c_1 d_2 \text{ (Eq. C.1),} \\ &= a_1 d_2 - a_1 c_2 - ta_1 d_2 + ta_1 c_2 + tb_1 d_2 - tb_1 c_2 + a_2 c_1 - a_2 d_1, \\ &\quad - ta_2 c_1 + ta_2 d_1 + tb_2 c_1 - tb_2 d_1 + c_2 d_1 - c_1 d_2, \\ &= t(a_1 c_2 - a_1 d_2 + b_1 d_2 - b_1 c_2 - a_2 c_1 + a_2 d_1 + b_2 c_1 - b_2 d_1), \\ &\quad + a_1 d_2 - a_1 c_2 + a_2 c_1 - a_2 d_1 - c_1 d_2 + c_2 d_1, \\ \det(\mathbf{s} - \mathbf{c} | \mathbf{d} - \mathbf{c}) &= t \left(\det(\mathbf{b} - \mathbf{c} | \mathbf{d} - \mathbf{c}) - \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \right) \\ &\quad + \det(\mathbf{a} - \mathbf{c} | \mathbf{d} - \mathbf{c}) \text{ (Eq. C.34 et C.32).} \end{aligned} \quad (\text{C.35})$$

3 Quelques démonstrations

Par soucis de lisibilité, certaines démonstrations du Chapitre 3 sont présentées ici. Ces démonstrations utilisent les propriétés sur le calcul de déterminant présenté précédemment.

3.1 Démonstration pour la Proposition 3.6

Proposition C.4 Soient un point $\mathbf{x} \in \mathbb{R}^2$ et un segment $Seg(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$ avec $\mathbf{x} \notin Seg(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$. On a

$$\begin{aligned} & \det(\mathbf{x} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \cdot \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\ & \det(\mathbf{e}_{1_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) \cdot \det(\mathbf{e}_{2_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) > 0 \\ \Leftrightarrow & \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) > 0 \vee \\ & \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) < 0. \end{aligned} \quad (\text{C.40})$$

avec

$$\zeta_{\mathbf{x}} = \begin{cases} 1 & \text{si } \det(\mathbf{x} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0, \\ -1 & \text{sinon.} \end{cases} \quad (\text{C.41})$$

Démonstration Nous voulons montrer que :

$$\begin{aligned} & \det(\mathbf{x} - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \cdot \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\ & \det(\mathbf{e}_{1_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) \cdot \det(\mathbf{e}_{2_j} - \mathbf{x} | \mathbf{x}_i - \mathbf{x}) > 0 \\ \Leftrightarrow & \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) > 0 \vee \\ & \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) < 0. \end{aligned} \quad (\text{C.42})$$

Par soucis de lisibilité, pour la suite de la démonstration, $\det(\mathbf{a} - \mathbf{b} | \mathbf{c} - \mathbf{b})$ sera noté $d(\mathbf{a}, \mathbf{b}, \mathbf{c})$.

Il y a deux cas possibles : $\zeta_{\mathbf{x}} = -1$ et $\zeta_{\mathbf{x}} = 1$ (Eq. C.41).

Cas numéro 1 :

$$\zeta_{\mathbf{x}} = -1 \Leftrightarrow d(\mathbf{x}, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \quad (\text{Eq : C.41}). \quad (\text{C.43})$$

Nous devons montrer que :

$$\begin{aligned} & d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{e}_{1_j}, \mathbf{x}, \mathbf{x}_i) \cdot d(\mathbf{e}_{2_j}, \mathbf{x}, \mathbf{x}_i) > 0 \Leftrightarrow \\ & d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \vee \det(\mathbf{x}_i | \mathbf{e}_{2_j}) \mathbf{x} > 0 \quad (\text{Eq. C.42 et C.43}). \end{aligned}$$

$$\begin{aligned}
& d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \text{ (Eq. C.43)}, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \text{ (Eq. C.29)}, \\
& d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0. \tag{C.44}
\end{aligned}$$

$$\begin{aligned}
& d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \text{ (Eq. C.43)}, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \text{ (Eq. C.29)}, \\
& d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0. \tag{C.45}
\end{aligned}$$

$$\begin{aligned}
& d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \text{ (Eq. C.44 et C.45)}, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) \cdot d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0.
\end{aligned}$$

La Proposition C.4 est donc vérifiée dans le cas où $\zeta_x = -1$.

Cas numéro 2 :

$$\zeta_x = 1 \Leftrightarrow d(\mathbf{x}, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) > 0 \Leftrightarrow d(\mathbf{x}, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \text{ (Eq : C.41 et C.28)}. \tag{C.46}$$

Nous devons montrer que :

$$\begin{aligned}
& d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) > 0 \vee d(\mathbf{e}_{1_j}, \mathbf{x}, \mathbf{x}_i) \cdot d(\mathbf{e}_{2_j}, \mathbf{x}, \mathbf{x}_i) > 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) > 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \text{ (Eq. C.42 et C.46)} \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \text{ (Eq. C.28)}.
\end{aligned}$$

$$\begin{aligned}
& d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \text{ (Eq. C.46)}, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \text{ (Eq. C.29)}, \\
& d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0. \tag{C.47}
\end{aligned}$$

$$\begin{aligned}
& d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \text{ (Eq. C.46)}, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \\
& \quad \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \text{ (Eq. C.29)}, \\
& d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{e}_{2_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0. \tag{C.48}
\end{aligned}$$

$$\begin{aligned}
& d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \vee d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) > 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0 \vee \\
& \quad d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) < 0 \wedge d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) < 0 \text{ (Eq. C.47 et C.48)}, \\
& \Leftrightarrow d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{e}_{1_j}) < 0 \vee d(\mathbf{x}_i, \mathbf{e}_{1_j}, \mathbf{x}) \cdot d(\mathbf{x}_i, \mathbf{e}_{2_j}, \mathbf{x}) > 0.
\end{aligned}$$

La Proposition C.4 est donc aussi vérifiée dans le cas où $\zeta_x = 1$.

3.2 Démonstration pour la Proposition 3.7

Proposition C.5 Soient un segment $Seg(\mathbf{x}_1, \mathbf{x}_2)$, avec $\mathbf{x}_1 \in \mathbb{R}^2$ et $\mathbf{x}_2 \in \mathbb{R}^2$, et un segment $Seg(\mathbf{e}_{1_j}, \mathbf{e}_{2_j})$, avec $Seg(\mathbf{x}_1, \mathbf{x}_2) \cap Seg(\mathbf{e}_{1_j}, \mathbf{e}_{2_j}) = \emptyset$. On a

$$\begin{aligned}
& \forall \mathbf{x} \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) \geq 0 \\
& \Leftrightarrow \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \geq 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) \geq 0 \wedge \\
& \quad \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0.
\end{aligned}$$

Démonstration D'après la Proposition C.3 on déduit que

$$\begin{aligned}
& \zeta_{\mathbf{x}} \in [\zeta_{\mathbf{x}_1}, \zeta_{\mathbf{x}_2}], \\
& \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) \in [\det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}), \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j})], \\
& \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) \in [\det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}), \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j})].
\end{aligned}$$

Nous avons donc,

$$\begin{aligned}
& \forall \mathbf{x} \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \forall \mathbf{x} \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \forall \mathbf{x} \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) \geq 0 \\
& \Leftrightarrow [\zeta_{\mathbf{x}_1}, \zeta_{\mathbf{x}_2}] \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad [\zeta_{\mathbf{x}_1}, \zeta_{\mathbf{x}_2}] \cdot [\det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}), \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j})] \leq 0 \wedge \\
& \quad [\zeta_{\mathbf{x}_1}, \zeta_{\mathbf{x}_2}] \cdot [\det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}), \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j})] \geq 0 \\
& \Leftrightarrow \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0 \wedge \\
& \quad \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0. \quad (\text{C.49})
\end{aligned}$$

On remarque que

$$\zeta_{\mathbf{x}_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \Leftrightarrow \zeta_{\mathbf{x}_1} = \zeta_{\mathbf{x}_2}.$$

On peut alors en déduire que

$$\begin{aligned}
& \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0 \wedge \\
& \quad \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0 \\
\Leftrightarrow & \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) \geq 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) \geq 0 \\
\Leftrightarrow & \forall \mathbf{x} \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) \leq 0 \wedge \\
& \quad \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x} - \mathbf{e}_{1_j}) \leq 0 \wedge \zeta_{\mathbf{x}} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x} - \mathbf{e}_{2_j}) \geq 0 \text{ (Eq. C.49)}.
\end{aligned}$$

3 .3 Démonstration de la Proposition 3.11

$$\begin{aligned}
E_{\varepsilon_j^s}(\text{Seg}(\mathbf{x}_1, \mathbf{x}_2)) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x}_s \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), (\mathbf{x}_i \forall \mathbf{x}_s)_{\varepsilon_j^s}\} \\
&= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x}_s \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\
& \quad \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_s - \mathbf{e}_{1_j}) > 0 \vee \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_s - \mathbf{e}_{2_j}) < 0 \vee \\
& \quad [\mathbf{x}_i \cup \mathbf{x}_s] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset\} \tag{C.50}
\end{aligned}$$

avec,

$$\zeta_s = \begin{cases} 1 & \text{si } \det(\mathbf{x}_s - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0, \\ -1 & \text{sinon.} \end{cases} \tag{C.51}$$

On notera que

$$\forall \mathbf{x}_s \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), [\mathbf{x}_i \cup \mathbf{x}_s] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \Leftrightarrow [\mathbf{x}_i \cup \mathbf{x}_1 \cup \mathbf{x}_2] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \tag{C.52}$$

Il y a alors deux cas possible d'écrire l'Équation C.50 comme étant

$$\begin{aligned}
E_{\varepsilon_j^s}(\text{Seg}(\mathbf{x}_1, \mathbf{x}_2)) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x}_s \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), (\mathbf{x}_i \forall \mathbf{x}_s)_{\varepsilon_j^s}\} \\
&= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \forall \mathbf{x}_s \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\
& \quad \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_s - \mathbf{e}_{1_j}) > 0 \vee \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_s - \mathbf{e}_{2_j}) < 0\} \cup \\
& \quad \{\mathbf{x}_i \in \mathbb{R}^2 \mid [\mathbf{x}_i \cup \mathbf{x}_s] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset\}. \tag{C.53}
\end{aligned}$$

Il y a deux cas possibles : $\zeta_{x_1} = \zeta_{x_2}$ et $\zeta_{x_1} = -\zeta_{x_2}$.

3.3.1 1^{er} cas : $\zeta_{x_1} = \zeta_{x_2}$

Dans ce cas on remarque que

$$\begin{aligned} \zeta_{x_1} = \zeta_{x_2} &\Leftrightarrow \zeta_s = \zeta_{x_1} = \zeta_{x_2} \\ \Rightarrow \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) &= \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) = \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}), \end{aligned} \quad (\text{C.54})$$

et que

$$\det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_s - \mathbf{e}_{1_j}) \in [\det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) \cup \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j})], \quad (\text{C.55})$$

$$\det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_s - \mathbf{e}_{2_j}) \in [\det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) \cup \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j})]. \quad (\text{C.56})$$

On peut alors déduire

$$\begin{aligned} \forall \mathbf{x}_s \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) > 0 &\Leftrightarrow \\ \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) > 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) > 0, &\quad (\text{C.57}) \end{aligned}$$

$$\begin{aligned} \forall \mathbf{x}_s \in \text{Seg}(\mathbf{x}_1, \mathbf{x}_2), \zeta_s \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) > 0 &\Leftrightarrow \\ \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) > 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) > 0. &\quad (\text{C.58}) \end{aligned}$$

D'après les Équations C.57, C.58 et C.53, on peut écrire

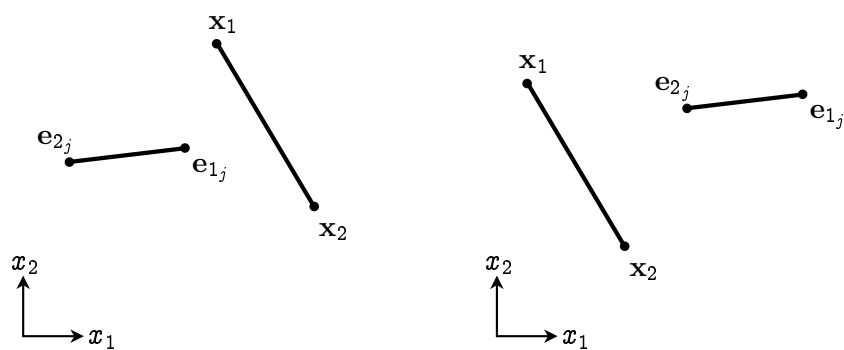
$$\begin{aligned} E_{\varepsilon_j^s}(\text{Seg}(\mathbf{x}_1, \mathbf{x}_2)) &= \{\mathbf{x}_i \in \mathbb{R}^2 \mid \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{e}_{2_j} - \mathbf{e}_{1_j}) > 0 \vee \\ &\quad \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) > 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) > 0 \vee \\ &\quad \zeta_{x_1} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) > 0 \wedge \zeta_{x_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) > 0 \vee \\ &\quad [\mathbf{x}_i \cup \mathbf{x}_s] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset\}. \end{aligned} \quad (\text{C.59})$$

3.3.2 2^{ème} cas : $\zeta_{x_1} = -\zeta_{x_2}$

On rappelle que $\text{Seg}(\mathbf{x}_1, \mathbf{x}_2) \cap \text{Seg}(\mathbf{e}_{1_j}, \mathbf{e}_{2_j}) = \emptyset$. On en déduit que

$$\zeta_{x_1} = -\zeta_{x_2} \Rightarrow \zeta_{e_1} = \zeta_{e_2}. \quad (\text{C.60})$$

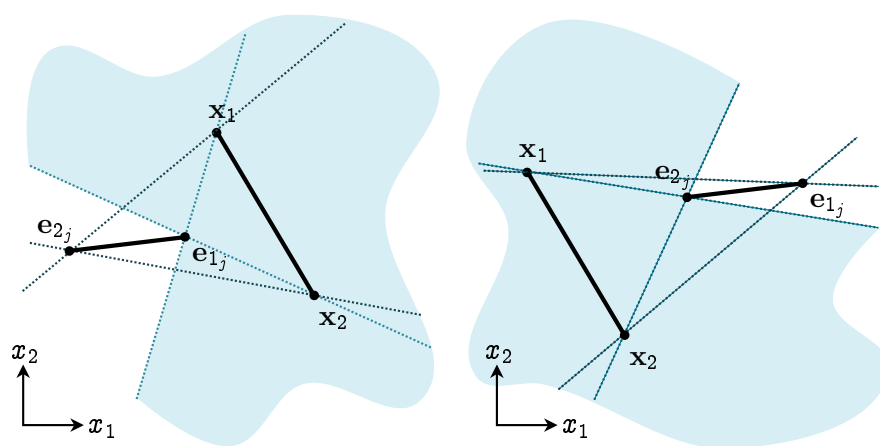
Il se trouve qu'il y a donc deux configurations : $\zeta_{e_1} = \zeta_{e_2} = 1$ (Figure C.10a) et $\zeta_{e_1} = \zeta_{e_2} = -1$ (Figure C.10b).



(a) $\zeta_{e_1} = \zeta_{e_2} = 1$.

(b) $\zeta_{e_1} = \zeta_{e_2} = -1$.

Figure C.10 – Les deux configurations possibles.



(a) $\zeta_{e_1} = \zeta_{e_2} = 1$.

(b) $\zeta_{e_1} = \zeta_{e_2} = -1$.

Figure C.11 – Espaces visibles dans les deux configurations.

Dans la première configuration ($\zeta_{e_1} = \zeta_{e_2} = 1$) on peut identifier l'espace visible (Figure C.11a) comme étant :

$$\begin{aligned} E_{\varepsilon_j^s}(Seg(\mathbf{x}_1, \mathbf{x}_2)) = \{ \mathbf{x}_i \in \mathbb{R}^2 \mid \\ & (\det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) > 0 \vee \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) < 0) \wedge \\ & (\det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) > 0 \vee \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) < 0) \vee \\ & [\mathbf{x}_i \cup \mathbf{x}_1 \cup \mathbf{x}_2] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \}. \end{aligned} \quad (C.61)$$

Alors que dans la deuxième configuration ($\zeta_{e_1} = \zeta_{e_2} = -1$) nous avons (Figure C.11b) :

$$\begin{aligned} E_{\varepsilon_j^s}(Seg(\mathbf{x}_1, \mathbf{x}_2)) = \{ \mathbf{x}_i \in \mathbb{R}^2 \mid \\ & (\det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) < 0 \vee \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) > 0) \wedge \\ & (\det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) < 0 \vee \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) > 0) \vee \\ & [\mathbf{x}_i \cup \mathbf{x}_1 \cup \mathbf{x}_2] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \}. \end{aligned} \quad (C.62)$$

De ces deux remarques, on en déduit que quand $\zeta_{x_1} = -\zeta_{x_2}$,

$$\begin{aligned} E_{\varepsilon_j^s}(Seg(\mathbf{x}_1, \mathbf{x}_2)) = \{ \mathbf{x}_i \in \mathbb{R}^2 \mid \\ & (\zeta_{e_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_1 - \mathbf{e}_{1_j}) > 0 \vee \zeta_{e_1} \det(\mathbf{x}_i - \mathbf{e}_{1_j} | \mathbf{x}_2 - \mathbf{e}_{1_j}) < 0) \wedge \\ & (\zeta_{e_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_1 - \mathbf{e}_{2_j}) > 0 \vee \zeta_{e_2} \det(\mathbf{x}_i - \mathbf{e}_{2_j} | \mathbf{x}_2 - \mathbf{e}_{2_j}) < 0) \vee \\ & [\mathbf{x}_i \cup \mathbf{x}_1 \cup \mathbf{x}_2] \cap [\mathbf{e}_{1_j} \cup \mathbf{e}_{2_j}] = \emptyset \}. \end{aligned} \quad (C.63)$$

Nous pouvons donc caractériser les espaces visibles dans les deux cas : quand $\zeta_{x_1} = \zeta_{x_2}$ et quand $\zeta_{x_1} = -\zeta_{x_2}$.

Bibliographie

- [Abdallah 2008] Fahed Abdallah, Amadou Gning et Philippe Bonnifait. *Box particle filtering for nonlinear state estimation using interval analysis*. Automatica, vol. 44, no. 3, pages 807 – 815, 2008.
- [Adams 2004] M. Adams, Sen Zhang et Lihua Xie. *Particle filter based outdoor robot localization using natural features extracted from laser scanners*. In Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on, volume 2, pages 1493–1498, 2004.
- [Agrawal 2006] M. Agrawal et K. Konolige. *Real-time Localization in Outdoor Environments using Stereo Vision and Inexpensive GPS*. In Pattern Recognition, 2006. ICPR 2006. 18th International Conference on, volume 3, pages 1063–1068, 2006.
- [Arras 1998] K.O. Arras et S.J. Vestli. *Hybrid, high-precision localisation for the mail distributing mobile robot system MOPS*. In Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on, volume 4, pages 3129 –3134 vol.4, may 1998.
- [Ashokaraj 2004] I. Ashokaraj, A. Tsourdos, P. Silson et B. White. *Sensor based robot localisation and navigation : using interval analysis and extended Kalman filter*. In Control Conference, 2004. 5th Asian, volume 2, pages 1086 –1093 Vol.2, 2004.
- [Bailey 2002] Tim Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, 2002.
- [Biswas 2012] J. Biswas et M. Veloso. *Depth camera based indoor mobile robot localization and navigation*. In Robotics and Automation (ICRA), 2012 IEEE International Conference on, pages 1697–1702, 2012.
- [Bittner 2003] Jiri Bittner et Peter Wonka. *Visibility in Computer Graphics*. JOURNAL OF ENVIRONMENTAL PLANNING, vol. 30, pages 729–756, 2003.
- [Blaer 2002] P. Blaer et P. Allen. *Topological mobile robot localization using fast vision techniques*. In Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on, volume 1, pages 1031–1036, 2002.

- [Borthwick 1993] S. Borthwick, M. Stevens et H. Durrant-Whyte. *Position estimation and tracking using optical range data*. In Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on, volume 3, pages 2172–2177, 1993.
- [Briggs 2000] A.J. Briggs, D. Scharstein, D. Braziunas, C. Dima et P. Wall. *Mobile robot navigation using self-similar landmarks*. In Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on, volume 2, pages 1428–1434, 2000.
- [Burchardt 2011] Armin Burchardt, Tim Laue et Thomas Röfer. *Optimizing Particle Filter Parameters for Self-localization*. In Javier Ruiz-del Solar, Eric Chown et PaulG. Plöger, éditeurs, RoboCup 2010 : Robot Soccer World Cup XIV, volume 6556 of *Lecture Notes in Computer Science*, pages 145–156. Springer Berlin Heidelberg, 2011.
- [Burgard 1996] Wolfram Burgard, Dieter Fox, Daniel Hennig et Timo Schmidt. *Estimating the Absolute Position of a Mobile Robot Using Position Probability Grids*. In In Proceedings of the Thirteenth National Conference on Artificial Intelligence, Menlo Park, pages 896–901. AAAI, AAAI Press/MIT Press, 1996.
- [Burgard 1999] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk Hänel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner et Sebastian Thrun. *Experiences with an interactive museum tour-guide robot*. Artificial Intelligence, vol. 114, no. 1–2, pages 3 – 55, 1999.
- [Burguera 2005] A. Burguera, G. Oliver et J.D. Tardos. *Robust scan matching localization using ultrasonic range finders*. In Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on, pages 1367–1372, 2005.
- [Chatila 1985] R. Chatila et J. Laumond. *Position referencing and consistent world modeling for mobile robots*. In Robotics and Automation. Proceedings. 1985 IEEE International Conference on, volume 2, pages 138–145, 1985.
- [Chenavier 1992] F. Chenavier et J.L. Crowley. *Position estimation for a mobile robot using vision and odometry*. In Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on, pages 2588–2593 vol.3, 1992.
- [Chong 1997] Kok Seng Chong et Chong Lindsay Kleeman. *Accurate Odometry and Error Modelling for a Mobile Robot*. In In IEEE International Conference on Robotics and Automation, pages 2783–2788, 1997.
- [Choset 2001] Howie Choset et Keiji Nagatani. *Topological simultaneous localization and mapping (SLAM) : toward exact localization without explicit localization*. Robotics and Automation, IEEE Transactions on, vol. 17, no. 2, pages 125–137, 2001.
- [Dao 2003] Nguyen Xuan Dao, Bum-Jae You, Sang-Rok Oh et Myung Hwangbo. *Visual self-localization for indoor mobile robots using natural lines*. In Intel-

- ligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, volume 2, pages 1252–1257, 2003.
- [Delafosse 2005] M. Delafosse, A. Clerentin, L. Delahoche et E. Brassart. *Uncertainty and Imprecision Modeling for the Mobile Robot Localization Problem*. In Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on, pages 4550–4555, 2005.
- [Delchev 2006] Ivan Delchev et Andreas Birk. *Vectorization of Grid Maps by an Evolutionary Algorithm*. In Gerhard Lakemeyer, Elizabeth Sklar, Domenico G. Sorrenti et Tomoichi Takahashi, editeurs, RoboCup, volume 4434 of *Lecture Notes in Computer Science*, pages 458–465. Springer, 2006.
- [Dellaert 1999] F. Dellaert, D. Fox, W. Burgard et S. Thrun. *Monte Carlo localization for mobile robots*. Proceedings 1999 IEEE International Conference on Robotics and Automation (ICRA), vol. 128, pages 1322–1328, 1999.
- [Dellaert 2003] Frank Dellaert, Fernando Alegre et Eric Beowulf Martinson. *Intrinsic Localization and Mapping with 2 Applications : Diffusion Mapping And Marco Polo Localization*. In in Proceedings of the IEEE International Conference on Robotics and Automation, pages 2344–2349, 2003.
- [Doh 2003] Nakju Doh, H. Choset et Wan-Kyun Chung. *Accurate relative localization using odometry*. In Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on, volume 2, pages 1606–1612, 2003.
- [Douc 2005] Randal Douc, Olivier Cappé et Eric Moulines. *Comparison of Resampling Schemes for Particle Filtering*. CoRR, vol. abs/cs/0507025, 2005.
- [Drumheller 1987] M. Drumheller. *Mobile Robot Localization Using Sonar*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 9, no. 2, pages 325–332, 1987.
- [Dulimart 1997] Hansye S. Dulimart et Anil K. Jain. *Mobile robot localization in indoor environment*. Pattern Recognition, vol. 30, no. 1, pages 99 – 111, 1997.
- [Durand 2010] Frédo Durand. *Visibilité tridimensionnelle : étude analytique et applications*. PhD thesis, 2010.
- [Elfes 1987] A. Elfes. *Sonar-based real-world mapping and navigation*. Robotics and Automation, IEEE Journal of, vol. 3, no. 3, pages 249–265, 1987.
- [Engelson 1992] S.P. Engelson et D.V. McDermott. *Error correction in mobile robot map learning*. In Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on, pages 2555–2560, 1992.
- [Fox 1999] D. Fox, W. Burgard, F. Dellaert et S. Thrun. *Monte Carlo Localization : Efficient Position Estimation for Mobile Robots*. In AAAI/IAAI, pages 343–349, 1999.
- [Fox 2000] Dieter Fox, Wolfram Burgard, Hannes Kruppa et Sebastian Thrun. *A Probabilistic Approach to Collaborative Multi-Robot Localization*. Autonomous Robots, vol. 8, no. 3, pages 325–344, 2000.

- [Fu 2011] Y. Fu, S. Tully, G. Kantor et H. Choset. *Monte Carlo Localization using 3D texture maps*. In Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, pages 482–487, sept. 2011.
- [Giguere 2012] P. Giguere, I. Rekleitis et M. Latulippe. *I see you, you see me : Cooperative localization through bearing-only mutually observing robots*. In Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, pages 863–869, 2012.
- [Gning 2006] A. Gning et Ph. Bonnifait. *Constraints propagation techniques on intervals for a guaranteed localization using redundant data*. Automatica, vol. 42, no. 7, pages 1167–1175, 2006.
- [Gutmann 2002] J.-S. Gutmann et D. Fox. *An experimental comparison of localization methods continued*. In Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on, volume 1, pages 454–459, 2002.
- [Guyonneau 2011] Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin et Philippe Lucidarme. *Interval Analysis for Global Localization Problem using Range Sensors*. In Small Workshop on Interval Methods, 2011.
- [Guyonneau 2012a] Rémy Guyonneau, Sébastien Lagrange et Laurent Hardouin. *Mobile Robots Pose Tracking : a Set-Membership Approach Using a Visibility Information*. In 9th International Conference on Informatics in Control, Automation and Robotics, 2012.
- [Guyonneau 2012b] Rémy Guyonneau, Sébastien Lagrange et Laurent Hardouin. *Méthode ensembliste pour l'estimation d'état d'un système non linéaire particulier*. In Journées Scientifiques Robotique et Automatique, 2012.
- [Guyonneau 2012c] Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin et Philippe Lucidarme. *The kidnapping Problem of Mobile Robots : A Set Membership Approach*. In 7th National Conference on Control Architectures of Robots, 2012.
- [Guyonneau 2013a] Rémy Guyonneau, Sébastien Lagrange et Laurent Hardouin. *A Visibility Information for Multi-Robot Localization*. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2013.
- [Guyonneau 2013b] Rémy Guyonneau, Sébastien Lagrange, Laurent Hardouin et Philippe Lucidarme. *Mobile Robot Localization : A Set-Membership Approach*. In Journées des Jeunes Chercheurs en Robotique, 2013.
- [Hahnel 2003] D. Hahnel, R. Triebel, W. Burgard et S. Thrun. *Map building with mobile robots in dynamic environments*. In Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on, volume 2, pages 1557–1563, 2003.
- [Hardt 1996] H.-J. Von Der Hardt, D. Wolf et R. Husson. *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, pages 603–610, 1996.

- [Hayet 2002] J.-B. Hayet, F. Lerasle et M. Devy. *A visual landmark framework for indoor mobile robot navigation*. In Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on, volume 4, pages 3942–3947, 2002.
- [Howard 2003] Andrew Howard, Maja J Mataric et Gaurav S. Sukhatme. *Putting the 'I' in 'Team' : an Ego-Centric Approach to Cooperative Localization*. In in Proceedings of the IEEE International Conference on Robotics and Automation, pages 868–892, 2003.
- [Huang 2004] Han-Pang Huang et Shu-Yun Chung. *Dynamic visibility graph for path planning*. In Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on, volume 3, 2004.
- [Ivanjko 2004] E. Ivanjko et I. Petrovic. *Extended Kalman filter based mobile robot pose tracking using occupancy grid maps*. In Electrotechnical Conference, 2004. MELECON 2004. Proceedings of the 12th IEEE Mediterranean, volume 1, 2004.
- [Jang 2002] Gijeong Jang, Sungho Lee et Inso Kweon. *Color landmark based self-localization for indoor mobile robots*. In Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on, volume 1, pages 1037–1042, 2002.
- [Jaulin 1993] L. Jaulin et E. Walter. *Set inversion via interval analysis for non-linear bounded-error estimation*. Automatica, 1993.
- [Jaulin 2001a] L. Jaulin, M. Kieffer, I. Braems et E. Walter. *Guaranteed non-linear estimation using constraint propagation on sets*. International Journal of Control, vol. 74, no. 18, pages 1772–1782, 2001.
- [Jaulin 2001b] L. Jaulin, M. Kieffer, O. Didrit et E. Walter. Applied interval analysis. Springer, 2001.
- [Jaulin 2001c] Luc Jaulin. *Path Planning Using Intervals and Graphs*. Reliable Computing, vol. 7, no. 1, pages 1–15, 2001.
- [Jaulin 2009] L. Jaulin. *A Nonlinear Set Membership Approach for the Localization and Map Building of Underwater Robots*. Robotics, IEEE Transactions on, vol. 25, no. 1, pages 88–98, feb. 2009.
- [Jaulin 2011a] Luc Jaulin. *Range-Only SLAM With Occupancy Maps : A Set-Membership Approach*. IEEE Transactions on Robotics, vol. 27, no. 5, pages 1004–1010, 2011.
- [Jaulin 2011b] Luc Jaulin. *Set-membership localization with probabilistic errors*. Robotics and Autonomous Systems, vol. 59, no. 6, pages 489–495, 2011.
- [Jensfelt 1999] P. Jensfelt et H.I. Christensen. *Laser based pose tracking*. In Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on, volume 4, pages 2994–3000, 1999.
- [Jensfelt 2001a] P. Jensfelt. *Approaches to Mobile Robot Localization in Indoor Environments*. PhD thesis, 2001.

- [Jensfelt 2001b] P. Jensfelt et H.I. Christensen. *Pose tracking using laser scanning and minimalistic environmental models*. Robotics and Automation, IEEE Transactions on, vol. 17, no. 2, pages 138–147, apr 2001.
- [Karam 2006] N. Karam, F. Chausse, R. Aufrere et R. Chapuis. *Cooperative Multi-Vehicle Localization*. In Intelligent Vehicles Symposium, 2006 IEEE, pages 564–570, 2006.
- [Kieffer 2000] Michel Kieffer, Luc Jaulin, Éric Walter et Dominique Meizel. *Robust Autonomous Robot Localization Using Interval Analysis*. Reliable Computing, vol. 6, pages 337–362, 2000.
- [Kortenkamp 1994] David Kortenkamp et Terry Weymouth Terry. *Topological mapping for mobile robots using a combination of sonar and vision sensing*. In AAAI'94 : Proceedings of the twelfth national conference on Artificial intelligence (vol. 2), pages 979–984. American Association for Artificial Intelligence, 1994.
- [Kuipers 1991] Benjamin Kuipers et Yung-Tai Byun. *A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations*. Journal of Robotics and Autonomous Systems, vol. 8, pages 47–63, 1991.
- [Lagrange 2005] S. Lagrange. *Contributions aux méthodes d'estimation en aveugle*. PhD thesis, 2005.
- [Lee 2007] Sooyong Lee et Jae-Bok Song. *Mobile robot localization using infrared light reflecting landmarks*. In Control, Automation and Systems, 2007. ICCAS '07. International Conference on, pages 674–677, 2007.
- [Leonard 1991] J.J Leonard et D. H. Whyte. *Mobile Robot Localization by Tracking Geometric Beacons*. IEEE Transactions on Robotics and Automation, vol. 7, 1991.
- [Leonard 1992] John J. Leonard, Hugh F. Durrant-Whyte et Ingemar J. Cox. *Dynamic map building for an autonomous mobile robot*. Int. J. Rob. Res., vol. 11, no. 4, pages 286–298, 1992.
- [Lingemann 2005] K. Lingemann, A. Nüchter, J. Hertzberg et H. Surmann. *High-speed laser localization for mobile robots*. Robotics and Autonomous Systems, vol. 51, no. 4, pages 275 – 296, 2005.
- [Lozano-Pérez 1979] Tomás Lozano-Pérez et Michael A. Wesley. *An algorithm for planning collision-free paths among polyhedral obstacles*. Commun. ACM, vol. 22, no. 10, pages 560–570, 1979.
- [Lucidarme 2011] P. Lucidarme et S. Lagrange. *Slam-O-matic : SLAM algorithm based on global search of local minima*. FR1155625 filed on June 24, 2011.
- [Moravec 1985] Hans Moravec et A. Elfes. *High resolution maps from wide angle sonar*. In Robotics and Automation. Proceedings. 1985 IEEE International Conference on, volume 2, pages 116–121, 1985.
- [Moravec 1988] Hans Moravec. *Sensor fusion in certainty grids for mobile robots*. AI Mag., vol. 9, no. 2, pages 61–74, jul 1988.

- [Moreno 2002] Luis Moreno, Jose M. Armingol, Santiago Garrido, Arturo De La Escalera et Miguel A. Salichs. *A Genetic Algorithm for Mobile Robot Localization Using Ultrasonic Sensors*. J. Intell. Robotics Syst., vol. 34, no. 2, pages 135–154, jun 2002.
- [Nassreddine 2010] G. Nassreddine, F. Abdallah et T. Denoux. *State Estimation Using Interval Analysis and Belief-Function Theory : Application to Dynamic Vehicle Localization*. Systems, Man, and Cybernetics, Part B : Cybernetics, IEEE Transactions on, vol. 40, no. 5, pages 1205–1218, 2010.
- [Nister 2004] D. Nister, O. Naroditsky et J. Bergen. *Visual odometry*. In Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, volume 1, pages I–652–I–659, 2004.
- [Ouerhani 2005] N. Ouerhani et H. Hugli. *Robot self-localization using visual attention*. In Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on, pages 309–314, 2005.
- [Paillat 2002] Jean-Luc Paillat. *Conception et contrôle de robots à géométrie variable : application au franchissement d'obstacles autonome*. PhD thesis, 2002.
- [Rekleitis 2000] I.M. Rekleitis, G. Dudek et E.E. Milios. *Multi-robot collaboration for robust exploration*. In Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on, volume 4, pages 3164–3169, 2000.
- [Roumeliotis 2002] S.I. Roumeliotis et George A. Bekey. *Distributed multirobot localization*. Robotics and Automation, IEEE Transactions on, vol. 18, no. 5, pages 781–795, 2002.
- [Roumeliotis 2004] S.I. Roumeliotis et Ioannis M. Rekleitis. *Propagation of Uncertainty in Cooperative Multirobot Localization : Analysis and Experimental Results*. Autonomous Robots, vol. 17, no. 1, pages 41–54, 2004.
- [Seigneur 2005] E. Seigneur, M. Kieffer, A. Lambert, E. Walter et T. Maurin. *Experimental vehicle localization by bounded-error state estimation using interval analysis*. In Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on, pages 1084 – 1089, aug. 2005.
- [Sim 1998] R. Sim et G. Dudek. *Mobile robot localization from learned landmarks*. In Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on, volume 2, pages 1060–1065, 1998.
- [Simhon 1998] S. Simhon et G. Dudek. *A global topological map formed by local metric maps*. In Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on, volume 3, pages 1708–1714 vol.3, 1998.
- [Sliwka 2011] J. Sliwka, F. Le Bars, O. Reynet et L. Jaulin. *Using interval methods in the context of robust localization of underwater robots*. In Fuzzy Information Processing Society (NAFIPS), 2011 Annual Meeting of the North American, pages 1–6, 2011.

- [Stevens 1995] A. Stevens, M. Stevens et H. Durrant-Whyte. *OXNAV : reliable autonomous navigation*. In Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on, volume 3, pages 2607–2612, 1995.
- [Suksakulchai 2000] S. Suksakulchai, S. Thongchai, D. M. Wilkes et K. Kawamura. *Mobile robot localization using an electronic compass for corridor environment*. In In Proceedings 2000 IEEE International Conference on Systems Man and Cybernetics, pages 8–11, 2000.
- [Thrun 1996] S. Thrun et A. Bucken. *Integrating Grid-Based and Topological Maps for Mobile Robot Navigation*. In Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence, 1996.
- [Thrun 1998a] S. Thrun. *Finding landmarks for mobile robot navigation*. In Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on, volume 2, pages 958–963, 1998.
- [Thrun 1998b] Sebastian Thrun. *Learning metric-topological maps for indoor mobile robot navigation*. Artificial Intelligence, vol. 99, no. 1, pages 21 – 71, 1998.
- [Thrun 2000] Sebastian Thrun, Dieter Fox et Wolfram Burgard. *Monte Carlo localization with mixture proposal distribution*. In in Proc. 17th National Conf. on Artificial Intelligence (AAAI-2000). AAAI Press/The, pages 859–865. MIT Press, 2000.
- [Thrun 2001] S. Thrun, D. Fox, W. Burgard et F. Dellaert. *Robust Monte Carlo localization for mobile robots*. Artificial Intelligence, vol. 128, no. 1–2, pages 99 – 141, 2001.
- [Thrun 2002] Sebastian Thrun. *Robotic Mapping : A Survey*. In Exploring Artificial Intelligence in the New Millenium. Morgan Kaufmann, 2002.
- [Thrun 2005] S. Thrun, W. Burgard et D. Fox. Probabilistic robotics. Intelligent robotics and autonomous agents. MIT Press, 2005.
- [Ulrich 2000] I. Ulrich et I. Nourbakhsh. *Appearance-based place recognition for topological localization*. In Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on, volume 2, pages 1023–1029 vol.2, 2000.
- [Wijk 2000] O. Wijk et H.I. Christensen. *Triangulation-based fusion of sonar data with application in robot pose tracking*. Robotics and Automation, IEEE Transactions on, vol. 16, no. 6, pages 740–752, 2000.
- [Wolf 2005] Denis F. Wolf et Gaurav S. Sukhatme. *Mobile robot simultaneous localization and mapping in dynamic environments*. Autonomous Robots, vol. 19, pages 53–65, 2005.
- [Yamauchi 1996] Brian Yamauchi. *Mobile robot localization in dynamic environments using dead reckoning and evidence grids*. In Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on, volume 2, pages 1401–1406, 1996.

-
- [Yamauchi 1997] Brian Yamauchi et Pat Langley. *Place Recognition in Dynamic Environments*. Journal of Robotic Systems, vol. 14, pages 107–120, 1997.
- [Zhang 2009] L. Zhang, R. Zapata et P. Lépinay. *Self-adaptive Monte Carlo localization for mobile robots using range sensors*. In Proceedings of the International Conference on Intelligent robots and systems (IROS), pages 1541–1546, 2009.
- [Zhou 2007] Yu Zhou, Wenfei Liu et Peisen Huang. *Laser-activated RFID-based Indoor Localization System for Mobile Robots*. In Robotics and Automation, 2007 IEEE International Conference on, pages 4600–4605, 2007.
- [Zimmer 1996] Uwe R. Zimmer. *Robust world-modelling and navigation in a real world*. Neurocomputing, vol. 13, no. 2–4, pages 247 – 260, 1996.

Index

A	
Amer (en robotique mobile).....	15
Amer artificiel.....	23
Amer naturel.....	23
B	
Bissection.....	159
Boîte.....	158
Borne inférieure.....	156
Borne supérieure.....	156
C	
Capteur de distance.....	18
Capteur ultrasons.....	18
Carte d'amers.....	15
Carte de caractéristiques.....	15
Centre d'un intervalle.....	157
Contracteur.....	174
Covariance.....	185
Croyance.....	188
CSP.....	173
D	
Défi CAROTTE.....	37
Densité de probabilité.....	182
Dérive.....	3
Développement de Taylor.....	199
Diaphonie.....	20
Distribution Gaussienne.....	182
Distribution normale.....	182
Distribution normale multivariée..	183
Drone.....	1
E	
Ensemble connexe.....	156
Ensemble fermé.....	156
Enveloppe intervalle.....	157
Environnement dynamique.....	4, 13
Environnement extérieur.....	12
Environnement intérieur.....	12
Environnement statique.....	4, 13
Espérance.....	185
État complet.....	187
Extéroceptif.....	17
F	
Facteur d'importance.....	204
Filtre Bayésien.....	188
Filtre de Kalman.....	194
Filtre non-paramétrique.....	202
Fonction d'inclusion.....	166
Fonction d'inclusion minimale....	168
Fonction d'inclusion naturelle....	166
Fonction unimodale.....	193
Forward-backward propagation....	177
G	
GPS.....	2
Grille d'occupation.....	16
I	
IAL.....	37
Inclusion.....	153
Indépendance conditionnelle.....	185
Intersection.....	152
Intersection relaxée.....	154
Intervalle.....	154
Inversion ensembliste.....	170

- J**
 Jacobien 200
- K**
 Kidnapping 3
- L**
 Localisation active 4
 Localisation globale 3
 Localisation Markovienne 24
 Localisation mono-robot 4
 Localisation Monte Carlo 27
 Localisation multi-robots 5
 Localisation par grille 26
 Localisation passive 4
 Loi Gaussienne 182
 Loi normale 182
- M**
 Matrice de covariance 183
 Modélisation métrique 14
 Modélisation topologique 14
- O**
 Odométrie 17
- P**
 Pavé 158
 Pavé indéterminé 172
 Pavé non-solution 172
 Pavé solution 172
 Poids (d'une particule) 204
 Posture 2
 Prédiction 188
 Probabilité conditionnelle 184
 Probabilité jointe 184
 Produit Cartésien 152
 Propagation avant-arrière 177
 Propagation de contraintes 174
 Proprioceptif 17
- Q**
 Q-intersection 154
- R**
 Ré-échantillonnage systématique .. 205
 Relation partielle 162
- Relation réflexive 79
 Relation symétrique 79
 Relation totale 162
 Relation transitive 79
 Représentation par moments 192
 Robot aérien 1
 Robot humanoïde 1
 Robot manipulateur 1
 Robot marcheur 1
 Robot mobile 1
- S**
 Satisfaction de contraintes 173
 Singleton 164
 SIVIA 172
 Sous-approximation 160
 Sous-pavage 160
 Suivi de posture 3
 Sur-approximation 161
 Système linéaire 193
 Système non-linéaire 193
- T**
 Taille d'un intervalle 156
 Télémètre 18
 Télémètre laser 20
 Télémétrie 18
- U**
 UAV 1
 UGV 1
 Ultrasons 20
 Union 152
 Union intervalle 157, 159
 UUV 1
- V**
 Variance 185
 Vecteur d'intervalles 158

Thèse de Doctorat

Rémy GUYONNEAU

Méthodes ensemblistes pour la localisation en robotique mobile

Set-Membership Approaches for Mobile Robot Localization

Résumé

Cette thèse s'intéresse aux problèmes de localisation en robotique mobile, et plus particulièrement à l'intérêt d'une approche ensembliste pour ces problèmes.

Actuellement les méthodes probabilistes sont les plus utilisées pour localiser un robot dans son environnement, cette thèse propose des approches alternatives basées sur l'analyse par intervalles.

Dans un premier temps, une méthode ensembliste s'intéressant au problème de localisation globale est proposée. Le problème de localisation globale correspond à la localisation d'un robot dans son environnement, sans connaissance a priori sur sa posture (position et orientation) initiale. La méthode proposée associe le problème de localisation à un problème de satisfaction de contraintes (CSP). Elle permet de localiser le robot en utilisant la connaissance de l'environnement, ainsi qu'un jeu de mesures LIDAR. Cette méthode est validée à l'aide de différentes expérimentations et est comparée à une approche probabiliste classique : la Localisation Monte Carlo (MCL).

Dans un second temps une notion de visibilité est étudiée. Deux points sont supposés visibles, si le segment défini par ces deux points n'intersecte pas d'obstacle, autrement ils sont dit non-visibles. À l'aide de l'analyse par intervalles, des contracteurs associés à cette notion de visibilité sont développés. Après une présentation théorique de la visibilité, deux applications de ces contracteurs à la localisation en robotique mobile sont présentées : le suivi de posture d'une meute de robots à l'aide d'une information booléenne, et la prise en compte d'une contrainte supplémentaire dans le CSP associé à la localisation globale.

Mots clés

Robotique mobile, Localisation globale, Suivi de posture, Analyse par intervalles, Problème de satisfaction de contraintes (CSP), Contracteurs, Visibilité

Abstract

This thesis deals with mobile robot localization problems, and more precisely with the interest of a set-membership approach to those problems. Nowadays most of the approaches to localize a robot are probabilistic. This document proposes alternatives based on interval analysis.

First a set-membership approach dealing with the global localization problem is presented. The global localization problem aims to localize a robot without any knowledge of its initial pose (position and orientation). The presented method casts the localization problem into a constraint satisfaction problem (CSP). It allows to localize the robot by using the knowledge of the environment and LIDAR sensor measurements. This approach is validated by several experimentations and is compared to the well known Monte Carlo Localization (MCL).

Then a visibility relation is studied. Two points are visible if the segment defined by those two points does not intersect any obstacle, otherwise they are non-visible. By using interval analysis, contractors associated to the visibility and non-visibility relations are presented. After a theoretical presentation of the visibility, two applications of those contractors are illustrated: the pose tracking of a team of robots using a boolean information, and a new constraint for the global localization algorithm.

Key Words

Mobile robotics, Global localization, Pose tracking, Interval Analysis, Constraint Satisfaction Problem (CSP), Contractors, Visibility