REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE.
Ministère de l'Enseignement Superieur et de la Recherche Scientifique.

Université Mouloud Mammeri de Tizi-Ouzou
- Faculté des Sciences -
Département de Mathématiques

Mémoire en vue de l'obtention du diplôme de master

THÉME :

Commande d'un robot mobile par vision artificielle

PAR : **Celia Ait Ali**

Sous la direction de Pr. Mellah Rabah

Domaine : Mathématiques et Informatique
Filière : Mathèmatiques
Specialité : Recherche Oerationnelle - Méthodes et modèles de décision

Promotion : 2016 - 2017

Remerciements

Mes remerciements vont naturellement à M.Mellah qui m'a encadrée et guidée tout au long de ce travail.
J'adresse également toute ma reconnaissance à M.Aiden pour ses conseils et l'intérêt qu'il a porté à ce mémoire.
Enfin, je tiens à remercier ma famille et mes amis pour leur soutien durant tout mon cursus.

Table des matières

Table des matières ii

Table des figures iii

Introduction v

I Généralités sur la robotique 1

1 Généralités sur la robotique 3

1.1 Définition d'un robot . . . 3
1.2 Définition d'un robot mobile . . . 3
1.3 Roues . . . 3
1.4 Capteurs . . . 4
1.4.1 Les Capteurs logiques . . . 4
1.4.2 Les Capteurs analogiques . . . 4
1.5 Actionneurs . . . 4

II Navigation par asservissement visuel en environnement sans obstacles 5

2 Navigation par asservissement visuel en environnement sans obstacles 7

2.1 Introduction . . . 7
2.2 Modélisation d'un robot mobile . . . 7
2.2.1 Modèle mathématique du robot . . . 7
2.2.2 Mouvement de la camera et vecteur de commande . . . 10
2.2.3 La jacobienne du robot . . . 11
2.3 Modélisation de la caméra . . . 12
2.4 Informations visuelles . . . 13
2.4.1 Perception . . . 13
2.5 Commande d'un robot mobile par asservissement visuel 2D . . . 14
2.5.1 Planification de trajectoire . . . 14
2.5.2 Résultat de simulation . . . 17
2.5.3 Conclusion . . . 18

III Navigation par asservissement visuel en environnement avec obstacles 19

3 Navigation par asservissement visuel en environnement avec obstacles 21
3.1 Asservissement visuel 3D 21
3.2 Évitement d'obstacles 25
3.3 Carte topologique 26
3.3.1 Définition supervisée des nœuds 27
3.3.2 Définition non supervisée des nœuds 27
3.3.3 Définition des arêtes 28
3.4 Navigation au long cours 28
3.5 Résultats de simulation 30

A Nomenclature 35

B Calcul du torseur Cinématique 37

Bibliographie 41

Table des figures

1.1 robots mobiles 4

2.1 Modélisation d'un robot mobile 8
2.2 Modélisation graphique de la caméra 12
2.3 MATLAB : Simulation d'un robot mobile en 2D 17
2.4 MATLAB : Résultat de simulation d'un robot mobile en 2D 18

3.1 Schéma de commande 22
3.2 Modèle physique du robot sur SIMULINK 23
3.3 Robot Pioneer 3-DX 23
3.4 Carte topologique d'un environnement 3D 26
3.5 Schéma Final de la simulation (SIMULINK) 30
3.6 Robot en environnement virtuel 31

Première partie

Généralités sur la robotique

Chapitre 1

Généralités sur la robotique

1.1 Définition d'un robot

Un robot est un dispositif mécatronique [1] (alliant mécanique, électronique et informatique) possédant des capteurs, un système logique et des actionneurs. Il exécute une ou plusieurs tâches de manière autonome conformément à un programme préétablie et modifiable.

Le terme «robot» est apparu pour la première fois vers 1920 dans une pièce de théâtre de science-fiction : R. U. R. (Rossum's Universal Robots) du tchèque K. Tschapek. Il désignait de petits êtres artificiels anthropomorphes [2] répondant parfaitement aux ordres de leur maître ("robota" signifie travail ou corvée en tchèque)

1.2 Définition d'un robot mobile

On désigne le plus souvent par le terme «robot mobile» des robots à roues avec une liberté de mouvements qui leurs confèrent une autonomie, L'objectif principal d'un robot mobile consiste à réaliser un mouvement à partir d'un point source vers un point destination et peut être doté de moyens de perception et de raisonnement.

1.3 Roues

Afin d'assurer l'acheminement du robot, les roues peuvent être disposées de différentes manières en fonction des caractéristiques recherchées.

Des roues peuvent, ainsi, assurer la traction ou la propulsion pendant que d'autres s'occupent de la direction de la machine. Les différentes combinaisons des roues assurent une diversité de commande et de la maniabilité. Les rangs de celles-ci sont disposés sur les sommets soit d'un rectangle. Soit d'un triangle.

La figure suivante nous présente quelques types de disposition et de fonctionnement :

1. **La mécatronique** : Domaine d'ingénierie interdisciplinaire combinant de la mécanique, de l'électronique, de l'automatisme et de l'informatique en temps réel.
2. **Anthropomorphe** : Qui, par sa forme, s'apparente à un être humain (corps ou apparence).

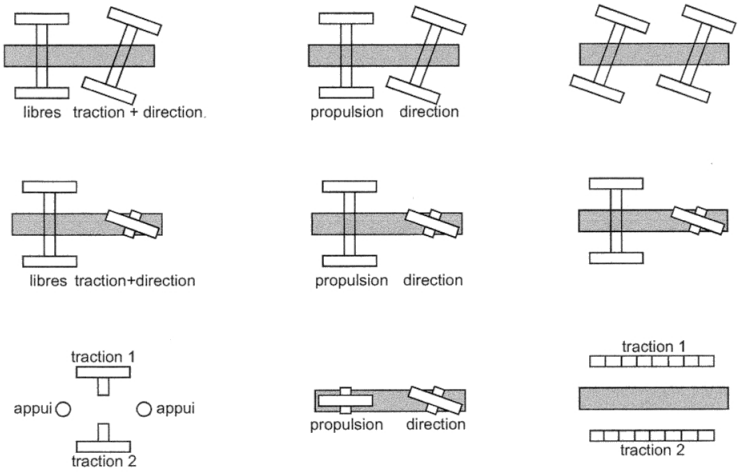

FIGURE 1.1 – robots mobiles

1.4 Capteurs

Pour réaliser un système de robotique mobile adroit, il est nécessaire d'utiliser des capteurs qui fourniront la perception requise de l'environnement pour une prise de décision intelligente.

On peut les classer en 2 catégories :

1.4.1 Les Capteurs logiques

Ils ne peuvent interpréter que 2 valeurs : '`oui` ou `non`', 'absence ou présence du phénomène'. Ces capteurs sont aussi appelés T.O.R (Tout Ou Rien). On peut, notamment, citer les capteurs infra-rouge en guise d'exemple.

1.4.2 Les Capteurs analogiques

Ils peuvent prendre une infinité de valeurs en continue, comme la luminosité ou la température.

On caractérise un capteur ou une combinaison de capteurs par sa sensibilité, sa précision, l'étendue de ses mesures et sa vitesse de détection.

1.5 Actionneurs

Pour qu'un robot puisse se déplacer à l'intérieur de son environnement et interagir avec celui-ci, il doit être équipé d'actionneurs. Dans le cas d'un robot mobile il s'agira de moteurs pouvant faire tourner les roues afin d'effectuer des déplacements. Ces moteurs contrôlent des commandes motrices représentées par la vitesse d'avancement (mètres par seconde m/s) et le taux de rotation (rotation par seconde deg/s).

Deuxième partie

Navigation par asservissement visuel en environnement sans obstacles

---

Chapitre 2

Navigation par asservissement visuel en environnement sans obstacles

2.1 Introduction

Pour réaliser la planification du mouvement et le contrôle des robots mobiles non-holonomes on adopte habituellement des approches d'asservissement visuel.

Un robot mobile non-holonome est une catégorie de robots qui dispose seulement de 2 degrés de liberté sur un plan, puisque les translations latérales sont impossibles à réaliser.

— Une translation : avance ou recule.

— Une rotation : tourne vers la droite ou vers la gauche.

Degré de liberté : Le nombre de degré de liberté d'un robot mobile est défini comme étant le nombre de mouvements indépendants que ce robot peut faire par rapport à un système de coordonnées déterminé.

Un schéma de contrôle de suivi visuel permet de réaliser une convergence exponentielle du système en boucle fermée par une méthode de placement de pôles.

Dans ce qui va suivre, nous considérons un problème de contrôle de suivi visuel de sorte qu'un robot mobile modélisé avec une caméra fixée sur le dessus puisse suivre une cible mobile. On le supposera comme étant un objet reconnaissable avec des dimensions appropriées (un espace d'état tridimensionnel), dans le plan de l'image.

La cinématique du robot mobile ainsi que celle de la cible peuvent être décrites, respectivement, par la figure (2.1)

2.2 Modélisation d'un robot mobile

2.2.1 Modèle mathématique du robot

Dans un premier temps, nous allons déterminer les coordonnées généralisées qui constituent l'ensemble des variables permettant de construire un modèle de configuration associé à un système que l'on va nommer $\boldsymbol{S}$

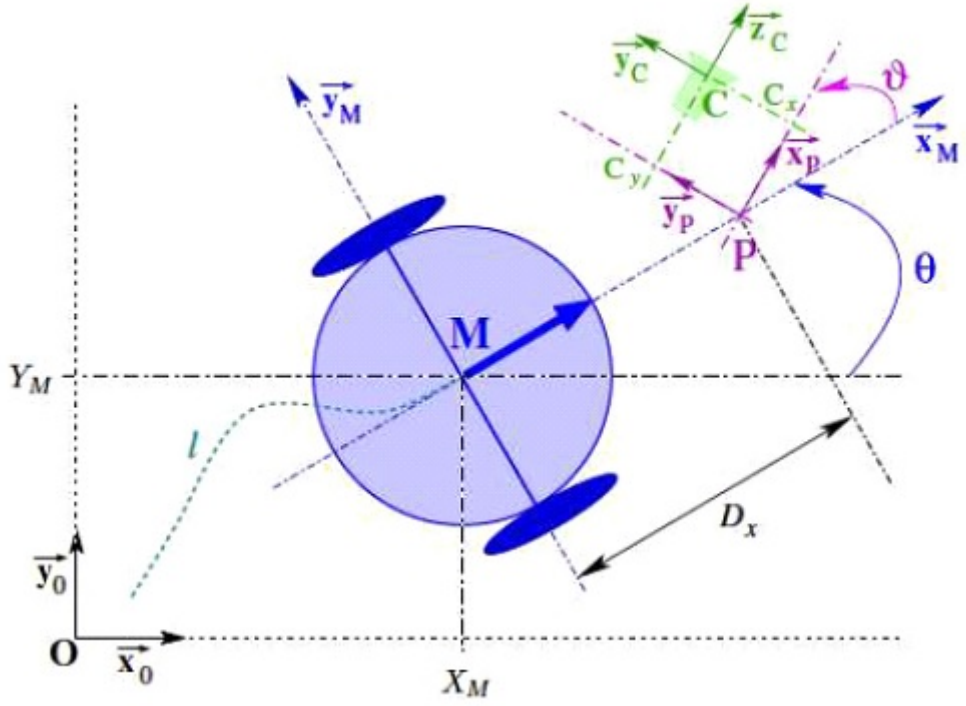

FIGURE 2.1 – Modélisation d'un robot mobile

Le but est de présenter la loi de commande réalisant une navigation.

La représentation graphique ci-dessous représente les cadres de références, global et local :

La modélisation du robot repose sur de multiples repères définis comme suit :

— $R_O(O, \sim x_O, \sim y_O, \sim z_O)$: repère de base.
— $R_M(M, \sim x_M, \sim y_M, \sim z_M)$: repère du robot.
— $R_R(P, \sim x_P, \sim y_P, \sim z_P)$: repère d'un point (cible).
— $R_C(C, \sim x_C, \sim y_C, \sim z_C)$

: repère de la caméra.

D_x : longueur de l'entre-axe [MP]

Pour la base mobile, la position est représentée par les coordonnées (X_M, Y_M) du point M dans le repère R_O.

L'orientation est donnée par l'angle $\theta = (\sim x_O, \sim x_M)$.

La position de la camera dans le repère R_P est décrite par le vecteur $P_C = (C_X, C_Y, C_Z)_T$. L'orientation de la platine est donnée par l'angle $V = (\sim x_M, \sim x_P)$.

Il est nécessaire de cartographier le mouvement dans la trame de référence globale. Pour cela, nous introduisons ici les matrices de passage homogènes.

Rappelons qu'une matrice de passage homogène du repère R_1 vers le repère R_2, est

définie de la manière suivante :

$$\mathcal{H}_{R_2/R_1} = \begin{pmatrix} \mathcal{R}_{R_2/R_1} & \mathcal{T}_{R_2/R_1} \\ 0_{(1,3)} & 1 \end{pmatrix} \tag{2.1}$$

ou $\mathcal{R}_{R_2/R_1}$ correspond à la matrice de rotation de R_1 vers R_2 et $\mathcal{T}_{R_2/R_1}$ correspond aux coordonnées de l'origine de R_1 exprimées dans R_2. De plus, $0_{(1,3)}$ est un vecteur nul de dimensions $(1, 3)$.

Les matrices de passage homogènes entre les différents repères sont définies comme suit :

$$\mathcal{H}_{R_O/R_M} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & X_M \\ \sin(\theta) & \cos(\theta) & 0 & Y_M \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

$$\mathcal{H}_{R_P/R_C} = \begin{bmatrix} 0 & 0 & 1 & C_x \\ 0 & 1 & 0 & C_y \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$\mathcal{H}_{R_M/R_P} = \begin{bmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 & D_x \\ \sin(\vartheta) & \cos(\vartheta) & 0 & 0 \\ 0 & 0 & 1 & h_r \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

Vecteur d'état

Le vecteur d'état du robot à l'instant t est défini comme suit :

$$\chi(t) = [X_M(t) \quad Y_M(t) \quad \theta(t) \quad \vartheta(t)]^T \tag{2.5}$$

Ce vecteur va nous permettre de caractériser de manière unique la configuration du système robotique à tout instant.

En suivant le modèle précédent, le vecteur d'état de la caméra est représenté comme suit :

$$\chi_C(t) = [X_C(t) \quad Y_C(t) \quad \theta_T(t)]^T \tag{2.6}$$

$X_C(t)$ et $Y_C(t)$: Les coordonnées à l'instant t du point C dans le repère R_O.
$\theta_T(t) = \theta(t) + \vartheta(t)$.

Vecteur de commande

Afin de contrôler le système robotique par asservissement visuel, il est nécessaire de définir un vecteur de commande en vitesse. Dans cette optique, et à partir des capteurs, nous introduisons le vecteur q qui dépend du temps $\mathbf{t}$:

$$q(t) = [l(t) \quad \theta(t) \quad \vartheta(t)]^T \tag{2.7}$$

où $l(t)$ représente l'abscisse curviligne du point O' par rapport au repère $\mathbf{R_1}$. La dérivée de $q(t)$ par rapport au temps donne :

$$\dot{q}(t) = \frac{\mathrm{d}q(t)}{\mathrm{d}t} = [v(t) \quad \omega(t) \quad \varpi(t)]^T \tag{2.8}$$

où $v(t)$ et $\omega(t)$ désignent, respectivement, les vitesses linéaire et angulaire de la base mobile et $\varpi(t)$ correspond à la vitesse de rotation de la platine par rapport au robot. Ce dernier doit être commandé en vitesse par le vecteur de commande représenté par $\dot{q}(t)$.

2.2.2 Mouvement de la camera et vecteur de commande

Lors de la réalisation d'un asservissement visuel, il est nécessaire d'étudier le lien entre le mouvement de la caméra et le vecteur de commande. Pour cela nous utilisons le torseur cinématique de la caméra, noté $T^{R_C}_{C/R_O}$ par rapport au repère R_O et exprimé dans le repère R_C.

Il est à préciser que l'expression et le calcul de $T^{R_C}_{C/R_O}$ sont issus de [Pissard-Gibollet et Rives 1995] et [Cadenat 1999].

$$T^{R_C}_{C/R_O} = \left[{V^{R_C}_{C/R_O}}^T \quad {\Omega^{R_C}_{R_C/R_O}}^T \right]^T \tag{2.9}$$

$$V^{R_C}_{C/R_O} = \left[V^{R_C}_{x_C} \quad V^{R_C}_{y_C} \quad V^{R_C}_{z_C} \right]^T \tag{2.10}$$

$$\Omega^{R_C}_{R_C/R_O} = \left[\Omega^{R_C}_{x_C} \quad \Omega^{R_C}_{y_C} \quad \Omega^{R_C}_{z_C} \right]^T \tag{2.11}$$

Les vecteurs V et Ω définissent, respectivement, les vitesses linéaires et angulaires de la caméra par rapport au repère de la scène R_O.

L'expression finale du torseur cinématique pour notre système robotique après calculs,

est donnée comme suit :

$$\mathcal{T}_{C/R_O}^{R_C} = \begin{bmatrix} 0 & 0 & 0 \\ -\sin(\vartheta) & C_x + D_x\cos(\vartheta) & C_x \\ \cos(\vartheta) & -C_x + D_x\sin(\vartheta) & -C_y \\ 0 & -1 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \dot{q} \tag{2.12}$$

Dans ce qui va suivre, le torseur cinématique sera noté T. ce dernier nous décrit le comportement de la caméra par rapport au repère de la scène.

Les vecteurs horizontaux nuls (lignes de zéros) représentent les mouvements non réalisables (la translation selon $\sim x_C$ et les rotations selon $\sim y_C$ et $\sim z_C$).

En les supprimant, le torseur cinématique ne comportera que les degrés de liberté de la caméra réellement commandables.

Ainsi, nous obtenons T_r :

$$\mathcal{T}_r = \begin{bmatrix} V_{y_C}^{R_C} \\ V_{z_C}^{R_C} \\ \Omega_{x_C}^{R_C} \end{bmatrix} = \begin{bmatrix} -\sin(\vartheta) & C_x + D_x\cos(\vartheta) & C_x \\ \cos(\vartheta) & -C_x + D_x\sin(\vartheta) & -C_y \\ 0 & -1 & -1 \end{bmatrix} \dot{q} \tag{2.13}$$

2.2.3 La jacobienne du robot

La matrice jacobienne du robot relie le torseur cinématique de la caméra au vecteur de commande du robot comme suit :

$$\mathcal{J}_r = \begin{bmatrix} -\sin(\vartheta) & C_x + D_x\cos(\vartheta) & C_x \\ \cos(\vartheta) & -C_x + D_x\sin(\vartheta) & -C_y \\ 0 & -1 & -1 \end{bmatrix} \tag{2.14}$$

Avec $det[J_r(v(t), t)] = -D_x$.

Donc la matrice $J_r(q(t), t)$ est régulière et inversible pour $D_x \neq 0$.

Tous les éléments en rapport avec la base mobile sont désormais décrits.

Passons à la modélisation de la caméra qui nous aidera dans la synthèse de lois de commande réalisant un asservissement visuel.

2.3 Modélisation de la caméra

La modélisation est basée sur l'hypothèse selon laquelle tous les rayons passent par un seul point, appelé centre optique. Comme nous pouvons le voir sur la figure ci-dessous, les points sont projetés sur le plan image selon une projection perspective en utilisant le point C comme centre optique.

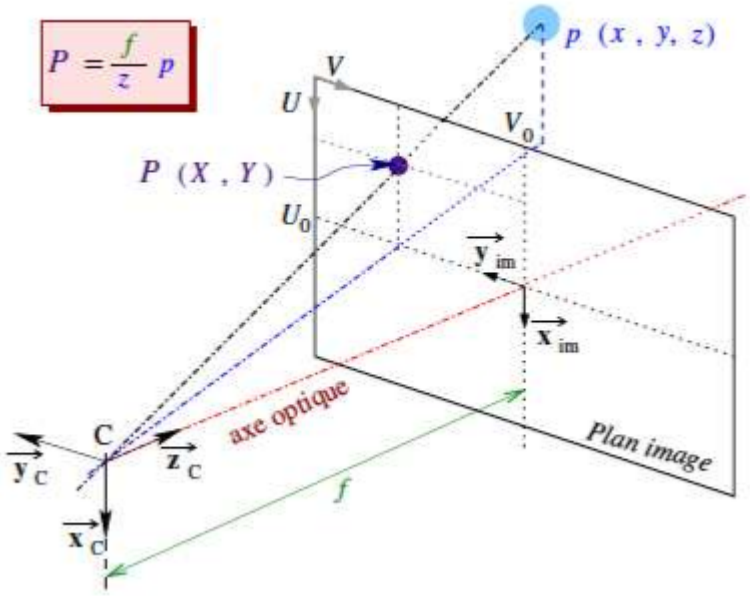

FIGURE 2.2 – Modélisation graphique de la caméra

Ainsi, un point p est projeté sur le plan image en un point P de coordonnées métriques (X, Y), selon la relation suivante :

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \frac{f}{z} & 0 & 0 \\ 0 & \frac{f}{z} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.15}$$

f : distance focale de la caméra.

Les coordonnées de P peuvent être exprimées en pixels à partir des coordonnées métriques en utilisant la matrice des paramètres intrinsèques :

$$\begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} \alpha_U & 0 & U_0 \\ 0 & \alpha_V & V_0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{2.16}$$

Avec $\alpha_U = \alpha_V = \frac{1}{pix}$ et pix la taille d'un pixel.

2.4 Informations visuelles

Les systèmes robotiques peuvent être décomposés en trois grandes parties :

- La couche de perception qui analyse l'environnement autour du robot (elle dépend de la vitesse à laquelle les capteurs du robot peuvent fonctionner).
- La couche de décision qui va tenter de résoudre la tâche donnée au système tout en prenant en compte l'environnement du robot, elle nécessite une grande puissance de calcul pour pouvoir assurer une réactivité suffisante.
- La couche action qui va utiliser les capacités du robot pour réaliser la tâche pour impacter son environnement (dans notre cas : rouler en suivant une trajectoire). Cette couche doit être extrêmement réactive et impose des contraintes importantes en terme de temps réel.

Dans ce qui va suivre, nous allons nous focaliser sur la couche de perception.

2.4.1 Perception

La vision est l'élément central de la perception sur un robot, Le traitement des images est souvent réalisé en série : on acquiert une image, on la convertit, on la rectifie, puis on cherche à en tirer des informations de plus haut niveau.

La caméra fixée sur le robot est un instrument de mesure de la luminance, elle est capable de capter trois luminances avec un filtre rouge vert et bleu. En combinant ces filtres on obtient la couleur.

Les capteurs de la caméra sont constitués de matrices de pixels grâce à des semi-conducteurs construits de puits de charge qui seront convertis en tensions.

Quand un rayon lumineux arrive à la surface du capteur il est attiré par le puits de charge le plus proche (chaque puits de charge correspond à un pixel).

On mesure la luminance de chaque carré de pixel par échantillonnage spatial, qui, combiné à un échantillonnage temporel forme un flux vidéo.

Après la conversion en pixels, on corrige la déformation de l'image liée à l'utilisation de l'objectif grand angle et à la conception de la caméra.

Un fois l'image corrigée, on passe à l'évaluation de l'erreur de positionnement en comparant la position voulue du robot dans le plan à un instant t à la position perçue au même instant.

2.5 Commande d'un robot mobile par asservissement visuel 2D

Le but d'un asservissement 2D est de faire converger l'image courante vers une image de référence. Nous allons donc créer un espace en deux dimensions qui nous permettra de suivre l'évolution du robot en le commandant.

Cette opération permet de s'affranchir du modèle de la scène, et, est donc adaptée aux cas des environnements peu connus. C'est là son principal avantage. Son inconvénient majeur vient des trajectoires non maîtrisées dans l'espace cité plus haut.

Le robot est fondamentalement un dispositif de déplacement opérationnel. Afin de le commander à partir de données purement visuelles, il est nécessaire de planifier une trajectoire à suivre dans l'espace 2D afin que le robot puisse se déplacer dans un environnement sans obstacle, dans un premier temps.

2.5.1 Planification de trajectoire

Position du problème

Soit un système à temps continu de représentation d'état :

$$\dot{x} = f(x, u, t) \tag{2.17}$$

et de condition initiale $x(t_0) = x_0$, où $t \in \mathbb{R}$, $u \in \mathbb{R}^{\triangleright}$ et $x \in \mathbb{R}^{\kappa}$. u et x qui représenteront des signaux sont des fonctions de $\mathbb{R}$ vers respectivement $\mathbb{R}^{\triangleright}$ et $\mathbb{R}^{\kappa}$. Une trajectoire unique x est définie pour l'état sur $[t_0, t_f]$. Elle représente une fonction de la condition initiale x_0 et de la commande u sur $[t_0, t_f]$.

Soit un critère :

$$J(x_0, t_0, u) = f_0(x_f, t_f) + \int_{t_0}^{t_f} \Psi(x, u, t)dt \tag{2.18}$$

avec $x_f = x(t_f)$. Les fonctions f et Ψ ainsi que les instants t_0 et t_f étant donnés, ce critère ne dépend que de x_0 et de u sur $[t_0, t_f]$. Un critère scalaire $J(x_0, t_0, u)$ est associé à l'application. Celle-ci est une fonctionnelle. Plusieurs présentations peuvent exister, notamment la présentation de Mayer et de Lagrange.

Principe de Mayer et de Lagrange

Ce principe est donnée sous la forme d'un système :

$$\dot{x} = f(x(t), u(t), t) \tag{2.19}$$

Avec $x(t_0) = x_0$, $x_f = x(t_f)$, $u \in U$ et $t \in [t_0, t_f]$.

Le coût est à minimiser, et est donné comme suit :

$$J(u(t)) = g(x(t_f)) + \int_{t_0}^{t_f} f_0(x(t), u(t), t)dt \tag{2.20}$$

On parlera de principe de Lagrange lorsque $g(x(t_f) = 0$ Et donc :

$$J(u(t)) = \int_{t_0}^{t_f} f^0(x(t), u(t), t)dt \tag{2.21}$$

Et du principe de Mayer lorsque $f^0(x(t), u(t), t) = 0$, la fonctionnelle vaudra alors :

$$J(u(t)) = g(x(t_f)) \tag{2.22}$$

Classement de la commande optimale

Les fonctions objectives peuvent être classées en deux critères physiques de performance.

— Temps optimal :

On parle d'un problème en temps optimal lorsque $f^0(x, u, t) = 1$ et $g(x(tf)) = 0$ le temps final t_f sera libre on aura alors :

$$J(u(t)) = min \int_{t_0}^{t_f} 1dt \tag{2.23}$$

— Coût optimal :

On parle d'un problème en coût optimal lorsque le temps final t_f est fixé, c'est à dire :

$$J(u(t)) = g(x(t_f)) + min \int_{t_0}^{t_f} f^0(x, u, t)dt \tag{2.24}$$

On peut aussi trouver des problèmes qui combinent les deux critères physiques de performance, et on parlera dans ce cas d'un problème de contrôle en temps et en coût optimal.

Dans ce cas il est intéressant de s'intéresser au problème de minimisation du temps de transfert dans une première phase, puis traiter correctement la minimisation du coût.

Le principe d'optimalité de Pontriaguine

Soient le système à temps continu (2.1) et le critère de performance associé (2.2) avec $x(t_0) = x_0$. L'Hamiltonien correspondant est défini par :

$$H(x, \eta, u, t) = \eta_0 f^0(x, u) + \eta(t) f(x, u, t) \tag{2.25}$$

où η est appelé vecteur d'état adjoint.

Le principe du minimum de Pontriaguine stipule que si la commande u^o est optimale, la fonction hamiltonienne de celle-ci (en temps optimal) sera définie comme suit :

$$H(x, \eta, u^o, t) = max_{u \in U}[H(x, \eta, u, t)] \tag{2.26}$$

En faisant appel au calcul des variations, on définit un certain nombre d'équations permettant de résoudre le problème de commande optimale. Ces équations correspondent aux équations canoniques de Hamilton qui régissent la dynamique de l'état donnée par :

$$\dot{x} = \frac{\partial H}{\partial \eta} \tag{2.27}$$

Et celle de l'état adjoint donnée par :

$$\dot{\eta} = -\frac{\partial H}{\partial x} \tag{2.28}$$

On leur ajoute les conditions aux limites (en t_0 et en t_f), dites équations de transversalité. En t_0 :

$$(H(x_0, \eta, u, t_0) + \frac{\partial g}{\partial t_0})\sigma t_0 + (-\eta(t_0) + \frac{\partial g}{\partial x_0})^T \sigma x_0 = 0 \tag{2.29}$$

En t_f :

$$(H(x_f, \eta, u, t_f) + \frac{\partial g}{\partial t_f})\sigma t_f + (-\eta(t_f) + \frac{\partial g}{\partial x_f})^T \sigma x_f = 0 \tag{2.30}$$

En fonction de la nature du problème considéré, des relations additionnelles peuvent apparaître :

— si, sur $u(t)$, aucune contrainte (de type saturation) n'est imposée à l'instant t :

$$\frac{\partial H}{\partial u}(t) = 0 \tag{2.31}$$

— Si H ne dépend pas de t :

$$\frac{\partial H}{\partial t} = 0 \tag{2.32}$$

2.5.2 Résultat de simulation

Nous passons à l'expérimentation. Dans un premier temps, nous allons implémenter un générateur de trajectoire pour un robot mobile quelconque.

Le travail se fait grâce au logiciel MATLAB selon l'algorithme suivant :

1. Initialiser un environnement sans obstacles (Carte 2D).
2. Initialiser l'ensemble des points de cheminement pour la trajectoire souhaitée.
3. Initialiser l'orientation et la position de départ du robot.

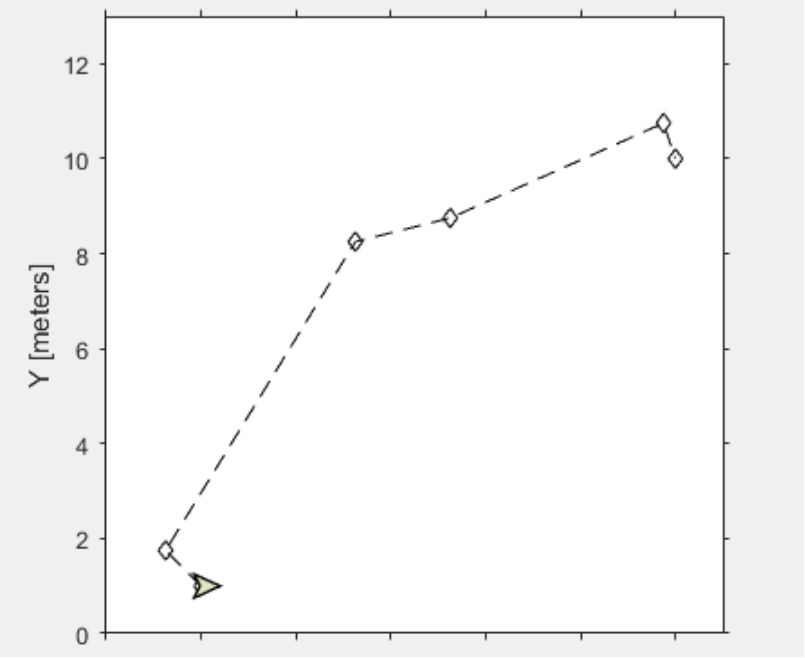

FIGURE 2.3 – MATLAB : Simulation d'un robot mobile en 2D

4. Créer le contrôleur de trajectoire.
5. Définir le chemin suivant les paramètres du contrôleur.
6. Définir le point d'arrêt du robot.
7. Démarrer la simulation à l'aide de la sortie du contrôleur sur la carte donnée jusqu'à ce qu'elle atteigne le point final avec un régulateur fonctionnant à 10 Hz.

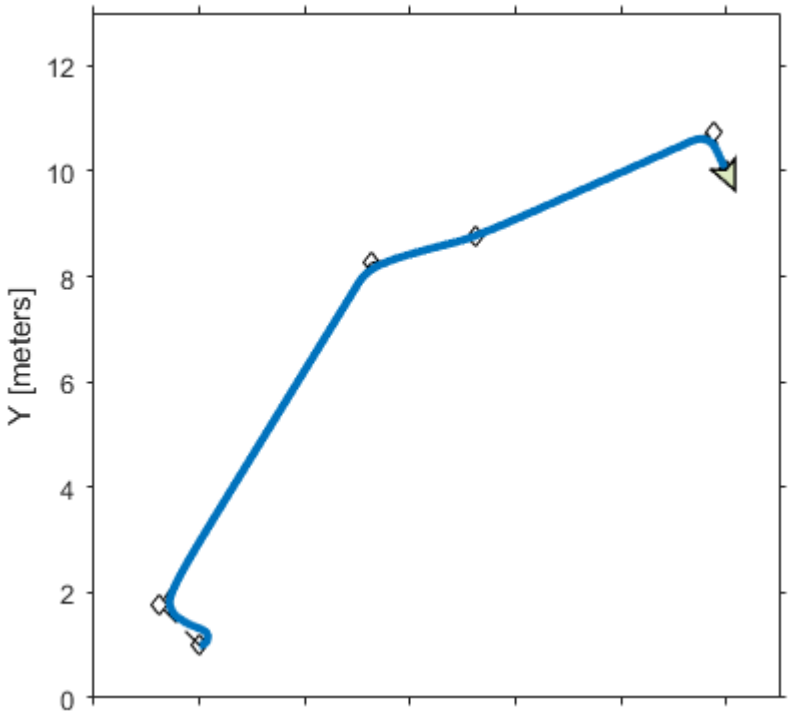

FIGURE 2.4 – MATLAB : Résultat de simulation d'un robot mobile en 2D

2.5.3 Conclusion

Dans ce chapitre, un système robotique a été modélisé afin de réaliser une navigation. Nous avons introduit les commandes de contrôle nécessaires avant de nous intéresser à la synthèse d'un asservissement visuel 2D qui contrôle, par essence, l'effecteur du robot.

Cependant, cette simulation n'est pas réaliste dans le cas d'une navigation réelle où la présence d'obstacles est inévitable. De plus, un mouvement libre du robot n'est pas envisageable dans ce type d'environnement minimaliste (2D).

Avec cette nouvelle contrainte, deux problèmes pouvant entraîner l'échec de la navigation apparaissent : le risque de collision du robot en cas de présence d'obstacles et son manque d'autonomie lorsqu'il se déplace dans son environnement.

Dans le prochain chapitre, nous nous focaliserons d'abord sur le manque d'autonomie du robot. Afin de répondre à cette problématique, nous passerons à un asservissement visuel 3D qui proposera une immersion en réalité virtuelle. Nous pourrons estimer l'évolution des indices visuels à partir des commandes appliquées au robot et du modèle d'évolution lorsque le suivi d'images est possible, en temps réel. Dans un second temps, nous allons nous attacher à résoudre le problème de collision dans un environnement avec obstacles.

Troisième partie

Navigation par asservissement visuel en environnement avec obstacles

Chapitre 3

Navigation par asservissement visuel en environnement avec obstacles

3.1 Asservissement visuel 3D

L'approche 3D est la plus intuitive, elle a pour but la commande en génération de trajectoires du robot grâce à une caméra vidéo afin de l'amener d'un état initial à un état final.

Elle suppose que le modèle de la scène est connu par le système de vision. L'avantage de cette pratique est une parfaite maîtrise du robot dans l'espace opérationnel.

La loi de commande utilise les informations 3D issues d'une chaîne de localisation 3D et commande le robot en vitesse.

Comme expliqué plus haut, le problème d'asservissement est vu comme un problème de poursuite de trajectoire qui nécessite le calcul d'un chemin à suivre. Dans le cas parfait où tout est connu, le rafraîchissement de la trajectoire est nécessaire que si l'on considère que la scène peut évoluer.

Le robot évolue dans son environnement selon le modèle suivant :

$$\dot{x} = \frac{v_g + v_d}{2} \cos\theta \tag{3.1}$$

$$\dot{y} = \frac{v_g + v_d}{2} \sin\theta \tag{3.2}$$

$$\dot{\theta} = \frac{v_d + v_g}{l} \tag{3.3}$$

Avec :

— θ : Système d'orientation.
— (x, y) : Système de translation.
— l : Distance entre deux roues.
— v_g et v_d : Vitesses des deux roues, respectivement, gauche et droite.

On pose :

$$\dot{\theta} = \frac{u_\theta}{l} \tag{3.4}$$

$$\dot{x} = u_\sigma \cos(\theta) \tag{3.5}$$

$$\dot{y} = u_\sigma \sin(\theta) \tag{3.6}$$

Avec :

$$\begin{cases} u_\sigma = \frac{v_d + v_g}{2} \\ u_\theta = v_d - v_g \end{cases}$$

A partir de l'équation (3.4), nous avons $\theta = \frac{u_\theta}{l}$. Ce qui donne en transformée de Laplace :

$$s\theta(s) = \frac{u_\theta}{l} \tag{3.7}$$

En supposant que les conditions initiales sont nulles. La fonction de transfert de θ par rapport à l'entrée u_θ est alors :

$$H_\theta(s) = \frac{\theta(s)}{u_\theta(s)} = \frac{1}{ls} \tag{3.8}$$

Le schéma de commande sera représenté comme suit :

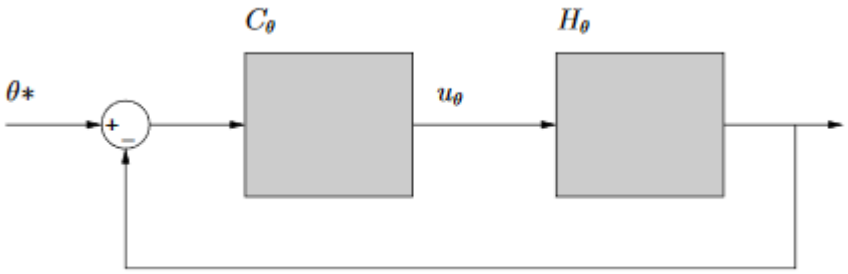

FIGURE 3.1 – Schéma de commande

La sortie u_θ du correcteur $C_\theta(s)$ est connectée a l'entrée du bloc $H_\theta(s)$ (correspondant a la dynamique de l'orientation θ du robot).

L'entrée du correcteur $C_\theta(s)$ est l'écart entre la sortie θ de $H_\theta(s)$ et l'orientation désirée θ^*.

Le modèle SIMULINK sera représenté comme suit :

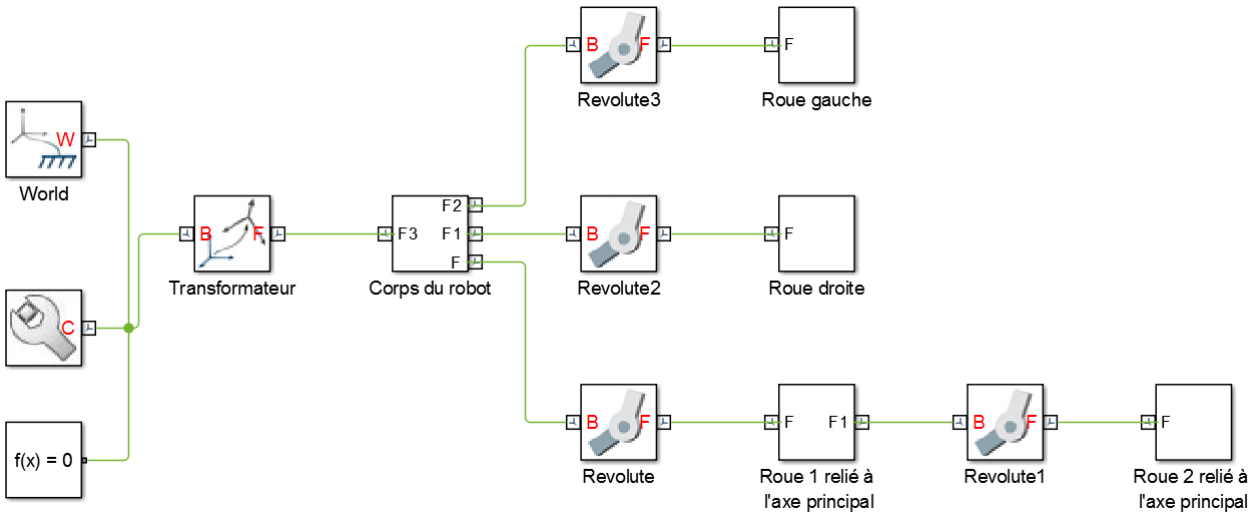

FIGURE 3.2 – Modèle physique du robot sur SIMULINK

Afin de superviser le robot dans un univers de simulation, nous allons utiliser un modèle de conception assistée par ordinateur (CAO) [1] existant, puis, développer un modèle Simulink et simMechanics pour créer des cas de tests, ce qui nous permettra de valider les algorithmes.

Le modèle CAO est un assemblage du robot Pioneer 3-DX qui est une plateforme à conduite différentielle disposant d'une liberté de mouvement suffisante pour éviter les obstacles en environnement encombré (On abordera ce point dans la partie évitement d'obstacles).

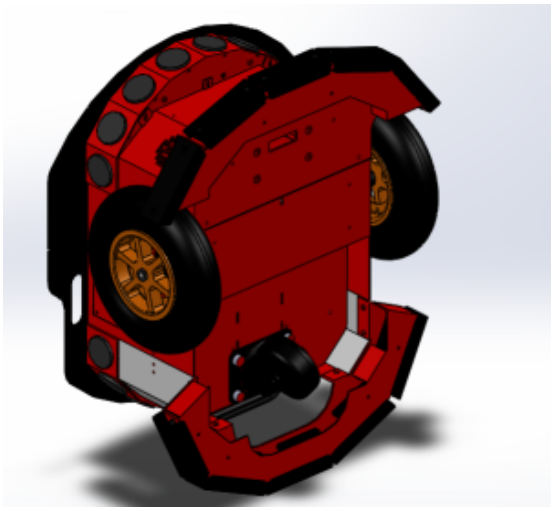

FIGURE 3.3 – Robot Pioneer 3-DX

1. **CAO** :La conception assistée par ordinateur (CAO) comprend l'ensemble des logiciels et des techniques de modélisation géométrique permettant de concevoir, de tester virtuellement à l'aide d'un ordinateur et des techniques de simulation numérique.

Les fichiers PAO sont exportés vers des fichiers XML [2] L'exportation se fait sous MATLAB grâce à la commande "smimport" qui translate le fichier en modèle SIMULINK. Ce dernier est formé de blocs représentant, entre autres, le corps du robot, les roues et l'axe reliant le corps aux roues sous forme d'arborescence.

Le comportement du robot est suivi en créant les cas de test suivant :

1. Valider le fonctionnement du robot sous contraintes gravitationnelles.
2. Ajouter des actionneurs afin de voir comment le robot se comporte.
3. Valider les mouvements du robot sous différentes vitesses de roues, ici, on attend du robot des rotations sur place, en inclinaison et en ligne droite.

— Test 1 : Observer le vecteur de gravité afin qu'il prenne un sens physique, on oriente la gravité dans le sens des z négatifs
— Test 2 : Afin d'actionner le robot, des blocs sont ajouté :
  — En guise d'actionneur, une constante est ajouté au bloc représentant l'attache reliant la roue au corps du robot, ce qui permettra au robot de se déplacer ou de s'arrêter.
  — Un intégrateur qui régulera la vitesse. Il est relié au bloc précédant afin qu'une rampe (pour définir cet angle) puisse être créée.

  Le robot reçoit une position angulaire sous forme de rampe qui permettra à la roue de tourner avec une vitesse constante.

  La différence entre les vitesses linéaires des deux roues permettront la rotation.

On intégrera dans le masque, la modélisation faite en chapitre 1 grâce à un sous système cinématique.

Les vitesses angulaires des roues sont récupérées et implémentées dans ce sous système. Ce qui nous permettra le calcul de la position des deux roues (P_x, P_y), et l'orientation θ (les paramètres linéaires du robot).

2. **XML** : eXtensible Markup Language est un langage qui permet de décrire des données à l'aide de balises et de règles que l'on peut personnaliser.

3.2 Évitement d'obstacles

La méthode consistera a réaliser un suivi de trajectoire sur la seule base d'informations fournies par la caméra. Les lois de commande étant utilisées pour la navigation, elle ne requièrent pas de connaissances à priori sur l'environnement, et sont calculées sur la base de données acquises au cours de la navigation.

Afin d'accroître la portée de cette dernière, nous ajouterons une carte de l'environnement qui, couplée aux méthodes précédemment présentées, offrira une portée globale à la navigation.

De plus, nous verrons que l'ajout de cette carte nécessite de définir une stratégie de navigation au long cours. Pour cela, nous introduirons un algorithme de supervision garantissant le bon déroulement des déplacements. L'algorithme de déplacement qui dépendra des capteurs de la caméra suit la logique suivante :

— **Initialisation**

On supposons que les capteurs sont calibrés sur une échelle de 0 (noir) a 100 (blanc).

— **Déroulement**

— La zone est complètement noire, le capteur retourne 0.
— La zone est complètement blanche, le capteur retourne 100.
— Si un capteur retourne une valeur $c < 100$, le robot n'est plus aligné.

— **Loi de commande**

Faire tourner le robot d'un angle proportionnel à la différence $(C_d - C_g)$.

A l'aide des correcteurs réalisant l'asservissement visuel 2D et l'algorithme du contournement d'obstacles, il est possible de réaliser une navigation référencée multi-capteurs dans un environnement encombré.

Cependant l'amer [3] doit être visible par la caméra dès le début de la navigation pour que cette dernière puisse démarrer. La portée de la caméra limitera la navigation.

En prenant en compte cette contrainte, on utilisera une carte topologique qui nous permettra de représenter la scène ou le modèle de l'environnement à l'aide de données de notre choix. La portée de la navigation sera ainsi augmentée et la réalisation de longs déplacements par asservissement visuel possible.

3. **Amer** : Terme utilisé pour définir un référentiel visuel fixe et identifiable qui sera considéré comme le point de repère du robot

3.3 Carte topologique

La modélisation de la carte topologique ne se base pas sur la géométrie exacte de l'environnement.

L'espace est découpé en lieux qui forment les nœuds d'un graphe. Chaque arête reliant deux nœuds peut être associée à une relation de connectivité entre ces lieux, à une action du robot ou une particularité topologique de l'espace, comme une porte marquant la limite entre deux zones. De vastes zones peuvent n'être représentées que par un unique nœud dans le graphe, permettant d'obtenir un modèle très compact. Aussi, elles offrent de sérieux avantages en terme de planification de chemin, la complexité algorithmique de la recherche dans un graphe étant moindre que dans un espace continu.

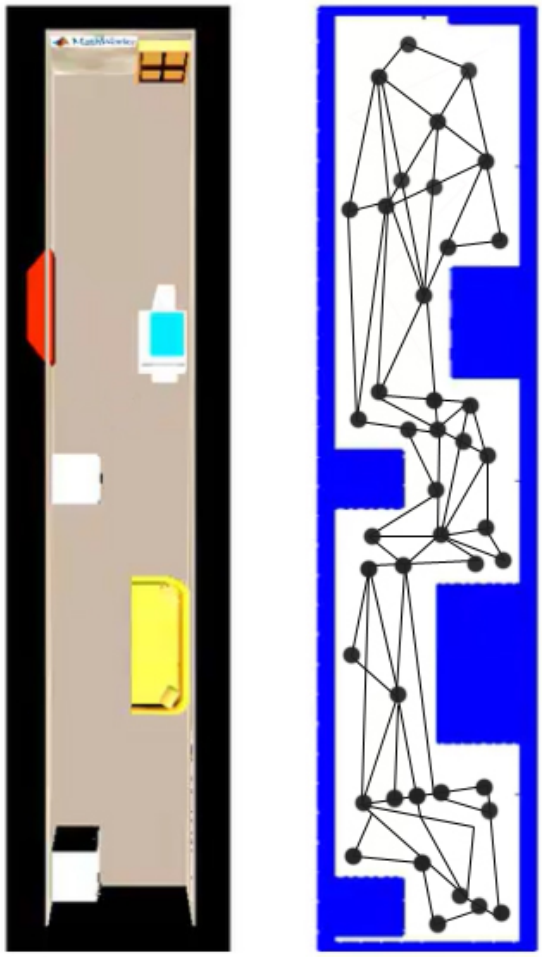

FIGURE 3.4 – Carte topologique d'un environnement 3D

On peut répertorier de nombreuses formes de cartes topologiques qui peuvent être regroupées en fonction de la manière dont sont définis les nœuds et les arêtes. On peut distinguer les techniques qui définissent des lieux de manière supervisée de celles où les lieux sont définis automatiquement par le robot.

Les différents modèles peuvent aussi être classés en fonction de l'information codée

par les arêtes qui peuvent, tour à tour, matérialiser des relations métriques d'adjacence ou les actions du robot pour passer de l'une à l'autre.

3.3.1 Définition supervisée des nœuds

Un exemple de définition supervisée de lieux peut être trouvé dans [Kunz 1997] et [Dedeoglu 1999]. Les auteurs définissent les nœuds du graphe comme les zones de transition de l'environnement, c'est à dire les intersections entre le couloir et une pièce donnée.

D'autres approches proposant une définition de plus haut niveau du concept de lieu ont été proposées [Ullah 2008, Ulrich 2000]. Elles ne reposent plus sur la reconnaissance d'un lieu basée sur une définition formelle, mais sur la classification du lieu défini par l'apprentissage automatique à partir d'exemples.

Le processus de construction de la carte topologique consiste alors à connecter les différents ensemble identifiés. Le plus souvent ces approches sont bien adaptées à des scènes intérieures pour lesquels la notion de lieux est clairement définie. Cependant elles s'adaptent assez mal au domaine extérieur pour lequel il n'existe, le plus souvent, pas de séparation claire entre des espaces successifs.

3.3.2 Définition non supervisée des nœuds

La seconde famille d'approches pour la définition des nœuds consiste à laisser le robot décider lui-même ce qui constitue un lieu sans définition préalable [Wichert 1998, Yamauchi 1996].

Deux approches ont été proposées. Dans le premier cas, le robot regroupe ses perceptions en fonction de leur similarité et utilise des critères propres aux données pour estimer s'il est arrivé dans un nouveau lieu [Chapoulie 2012, Gaussier 2000, Bailey 1999]. Il doit donc disposer d'un moyen efficace de comparer les données. Ce n'est pas une tâche triviale, notamment lors que la taille de l'environnement croît et avec elle la quantité de données.

La seconde approche consiste simplement à considérer que le robot a changé de lieux lorsqu'il a parcouru une distance donnée depuis le dernier lieu enregistré [Yamauchi 1996, Wichert 1998]. Cette approche est d'un intérêt moindre puisqu'elle utilise uniquement l'odométrie[4] pour la définition d'un lieu à la place de critères intrinsèques à l'environnement, et, est à utiliser lorsque les ressources du système ne permettent pas d'autres techniques.

4. **Odométrie :** L'odométrie est la mesure du déplacement. Elle nous permet de déterminer la position d'un robot : le déplacement est intégré au fur et à mesure pour définir la position relative à la position de départ.

3.3.3 Définition des arêtes

Les différents modèles de carte topologique se différencient aussi par la définition des arêtes reliant deux nœuds. On peut citer le cas des modèles utilisant les arêtes pour coder les relations métriques entre les nœuds [Engelson 1992, Bailey 1999, Landsiedel 2013]. L'avantage de ces méthodes est qu'elles souffrent moins de l'imprécision de l'odométrie que les cartes métriques.

Ce type de carte fournit une description de haut niveau directement utilisable pour la commande du robot. Un des avantages de cette représentation est qu'elle offre une description directement accessible à l'homme.

Dans le cas d'une navigation au long cours dans un environnement encombre, voici, ci-dessous les différentes phases nécessaires.

3.4 Navigation au long cours

Perception

Afin de réaliser une navigation au long cours, l'utilisation d'une caméra comme capteur principal est essentiel lors des phases d'évitement d'obstacles.

Modélisation

La scène dans laquelle se déroule la navigation est modélisée à l'aide d'une carte topologique.

Chaque nœud représente la zone de visibilité d'un point de repère fixe et un couple de nœuds est relié s'il existe une zone de visibilité commune pour les deux repères.

Des données sensorielles sont associées à chacun des nœuds. Elles seront exploitées par les processus de localisation et de commande.

Localisation

La localisation consiste à identifier la cible qui se situe dans le champ de vue de la caméra. Ainsi, il est possible de connaître la situation du robot à l'intérieur du graphe. La localisation est réalisée à l'aide des données associées à chacun des nœuds de la carte topologique.

Comme les informations sensorielles correspondent aux descripteurs issus d'images obtenues pour différents points de vue, la localisation consiste à réaliser un test de similitude entre les images. Pour cela, les descripteurs de l'image courante et ceux de la base de données sont comparées.

L'image issue de la base de données ayant le plus grand nombre de points communs avec l'image courante est sélectionnée. Le nœud correspondant est alors considéré comme celui ou se situe le robot.

Planification

Lors d'une première phase de localisation, les cibles sont identifiées dans le champ de vue de la caméra pour la situation initiale $[X_M(0)Y_M(0)]$.

Dans un second temps, la cible finale étant estimée par l'algorithme du modèle de capteurs, le chemin à parcourir est ainsi tracé. Aucune information relative à la distance n'est nécessaire dans la carte topologique.

Action

On peut distinguer trois phases d'action nécessaires à la réalisation d'une navigation au long cours :

— Premièrement : Définir un algorithme permettant au robot de réaliser des navigations sur une courte distance.

— Deuxièmement : Définir un algorithme permettant d'éviter des obstacles. Celui-ci essaie de stabiliser le robot sur un chemin défini à partir des données fournies par les capteurs de la caméra.

— Troisièmement : Synthétiser l'orientation en direction de la cible d'intérêt afin que le robot puisse parcourir une longue distance.

Décision

Le processus de décision doit permettre au robot d'activer ou de désactiver les outils présentés précédemment afin de garantir le succès de la navigation au long cours. Un algorithme de supervision pourrait, par exemple, prendre en charge le processus de décision selon une stratégie de navigation :

1. Identifier les amers qu'il peut voir depuis sa position $[X_M(0)Y_M(0)]$.
2. Comme l'environnement de navigation est considéré comme étant un graphe connexe, les calcules peuvent se faire à l'aide de l'algorithme de Dijkstra. Si plusieurs chemins sont disponibles, alors celui contenant le moins d'éléments est sélectionné (concept du plus court chemin). Ainsi, le point final à atteindre est désormais déterminé.
3. De petites rotations sont alors réalisées par le robot. Cela nous permet d'estimer la profondeur des indices visuels de la cible.
4. Test d'arrêt : Lorsque les indices visuels de référence estimés sont égaux à ceux mesurés lors de la pré-navigation (convergence du processus d'estimation), l'algorithme s'arrête.

3.5 Résultats de simulation

Le modèle physique du robot (figure 3.2) a été relié au modèle 3D (le fichier CAO, utilisé précédemment, a été converti en fichier VRML [5]).

Ceci, en utilisant des informations de transfert de coordonnées et des transformations géométriques basiques grâce aux métriques de rotation et aux vecteurs de transformation. Ainsi, le masque de réalité virtuelle présent sous Simulink active la fenêtre simulant notre robot dans son environnement.

Le schéma final est donné comme suit :

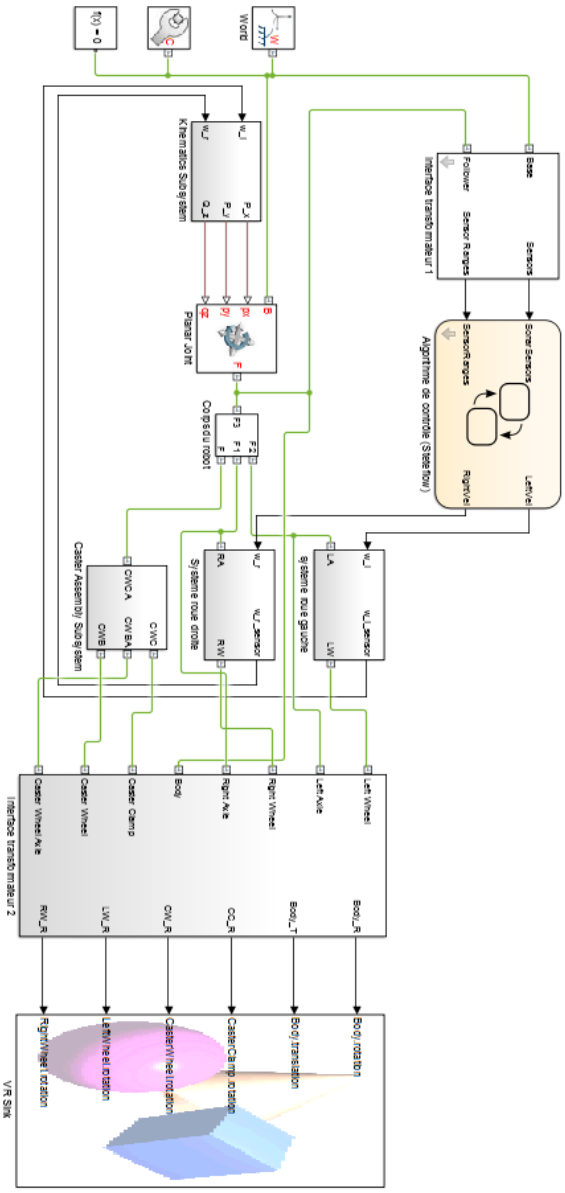

FIGURE 3.5 – Schéma Final de la simulation (SIMULINK)

5. **VRML :** Virtual Reality Markup Language est un langage de description d'univers virtuels en 3 dimensions qui ont habituellement ".wrl" pour extension.

Après exécution, une fenêtre VRML apparaît. Le robot se déplace en évitant les obstacles présents dans la pièce sous les directives de l'algorithme de contrôle programmé dans un environnement Steteflow [6].

FIGURE 3.6 – Robot en environnement virtuel

6. **Steteflow :** environnement de modélisation et de simulation de logique de décision combinatoire et séquentielle à partir de machines d'état et de diagrammes de flux.

Conclusion

Dans ce mémoire, nous nous sommes intéressés à la navigation d'un robot mobile par vision artificielle dans un environnement possiblement encombré.

Le système robotique étant muni d'une caméra et d'une loi de commande, nous avons exploité les techniques d'asservissement visuel 2D pour guider le robot vers son objectif. Celui-ci est défini par une mesure dans l'image relative à un amer d'intérêt et correspond à la situation devant être atteinte par la caméra.

Cependant, ces techniques ne permettent de réaliser la tâche considérée que si le robot évolue dans un environnement libre. En effet, la présence d'obstacles peut conduire à des collisions, des occultations ou à la non visibilité de l'amer d'intérêt dès le début de la tâche.

En se focalisant sur ces contraintes, nous avons abordé un ensemble de procédés en utilisant un asservissement visuel 3D afin d'éviter les obstacles en environnement encombré.

Aussi, une loi de commande a été créée afin de planifier la trajectoire du robot, ce qui nous a conduit à déduire que le système est linéaire et que la commande est quadratique.

Sur ces bases, une simulation a été proposée en utilisant les logiciels MATLAB et SIMULINK, ceci, en implémentant l'algorithme de contrôle sous Steteflow.

En synthèse, le robot évolue par asservissement visuel, dans un environnement immersif (réalité virtuelle) avec obstacles , en totale autonomie et sans risque de collisions.

Annexe A

Nomenclature

Symbole	Description
S	Système du modèle mathématique du robot
R_O	Repère de base
R_M	Repère du robot
R_P	Repère d'un point (cible)
R_C	Repère de la caméra
D_x	Longueur de l'entre-axe $[MP]$
θ	Angle d'orientation du robot
P_C	Vecteur décrivant la position de la caméra
V	Angle d'orientation de la platine
$\mathcal{H}_{R_2/R_1}$	Matrice de passage de R_1 vers R_2
$\mathcal{R}_{R_2/R_1}$	Matrice de rotation de R_1 vers R_2
$\mathcal{T}_{R_2/R_1}$	Coordonnées de l'origine de R_1 exprimées dans R_2
$\mathcal{H}_{R_O/R_M}$	Matrice de passage de R_M vers R_O
$\mathcal{H}_{R_P/R_C}$	Matrice de passage de R_C vers R_P
$\mathcal{H}_{R_M/R_P}$	Matrice de passage de R_P vers R_M
$\mathcal{X}(t)$	Vecteur d'état du robot
$X_C(t)$, $Y_C(t)$	Coordonnées à l'instant t du point C dans le repère R_O
$q(t)$	Vecteur de commande
$l(t)$	Abscisse curviligne du point O'
$\dot{q}(t)$	Dérivée du vecteur de commande
$\vartheta(t)$	Vitesse linéaire de la base mobile
$\omega(t)$	Vitesse angulaire de la base mobile
$\varpi(t)$	Vitesse de rotation de la platine par rapport au robot
$T^{R_C}_{C/R_O}$	Torseur cinématique de la caméra
V	Vecteur des vitesses linéaires de la caméra
Ω	Vecteur des vitesses angulaires de la caméra
$\mathcal{J}_r$	Matrice jacobienne du robot
f	Distance focale de la caméra (partie modélisation de la caméra)
$\dot{x}$	Système à temps continu de représentation d'état
$J(x,t,u)$	Critère scalaire d'optimalité
$H(x,\eta,u,t)$	Fonction hamiltonienne
η	Vecteur d'état adjoint

Annexe B

Calcul du torseur Cinématique

Les calculs permettant d'établir l'expression du torseur cinématique de la caméra pour le système robotique modélisé dans la figure 2.1 sont les suivants :

Le torseur est déterminé par rapport au repère de la scène R_O et exprimé dans le repère caméra R_C.

Il faut d'abord déterminer son expression dans le repère de la base mobile R_M puis le projeter dans R_C.

Pour rappel, les expressions de $\mathcal{H}_{R_P/R_C}$ et $\mathcal{H}_{R_M/R_P}$ sont respectivement les matrices de passage entreR_C et R_P, et R_P et R_M :

$$\mathcal{H}_{R_P/R_C} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \tag{B.1}$$

$$\mathcal{H}_{R_M/R_P} = \begin{bmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{B.2}$$

De plus, le torseur $T^C_{R_O}$ s'écrit comme suit :

$$\mathcal{T}_{C/R_O} = \begin{bmatrix} \overrightarrow{V}_{C/R_O} \\ \overrightarrow{\Omega}_{R_C/R_O} \end{bmatrix} = \begin{bmatrix} \overrightarrow{V}_{C/R_M} + \overrightarrow{V}_{M/R_O} + \overrightarrow{\Omega}_{R_M/R_O} \wedge \overrightarrow{MC} \\ \overrightarrow{\Omega}_{R_C/R_M} + \overrightarrow{\Omega}_{R_M/R_O} \end{bmatrix} \tag{B.3}$$

Il est nécessaire de déterminer $\overrightarrow{MC}$, $\mathcal{T}_{C/R_M}$ et $\mathcal{T}_{M/R_O}$.

Le calcul de $\mathcal{T}_C/R_O$ exprimé dans R_C sera décomposé comme suit :

— Calculer T_M/R_O exprimé dans R_M.
— Calculer T_C/R_M donnée dans R_M.
— Calculer $\overrightarrow{MC}$ exprimé dans R_M.
— Déterminer T_C/R_O donné dans R_M à partir des trois résultats précédents et de l'équation (B.2)
— Déterminer T_C/R_O exprimé dans R_C à partir de T_C/R_O donné dans R_M et de la matrice de passage entre les repères R_M et R_C.

Calcul de T_{M/R_O}

A partir de la géométrie du robot, T_{M/R_O} est donné par :

$$T_{M/R_O} = \begin{bmatrix} V_{M/R_O} \\ \Omega_{R_M/R_O} \end{bmatrix} = \begin{bmatrix} v \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ w \end{bmatrix} \tag{B.4}$$

Calcul de $T_{C/RM}$

Les points P et C sont respectivement liés aux repères R_M et R_P, les vitesses $\overrightarrow{V}_{P/R_M}$ et $\overrightarrow{V}_{C/R_P}$ sont nulles. De plus, R_C est fixe par rapport à R_P, $\overrightarrow{\Omega}_{R_C/R_P} = \overrightarrow{0}$, $T_{C/RM}$ sera donc donné par :

$$\mathcal{T}_{C/R_M} = \begin{bmatrix} \overrightarrow{V}_{C/R_M} \\ \overrightarrow{\Omega}_{R_C/R_M} \end{bmatrix} = \begin{bmatrix} \overrightarrow{V}_{C/R_P} + \overrightarrow{V}_{P/R_M} + \overrightarrow{\Omega}_{R_P/R_M} \wedge \overrightarrow{PC} \\ \overrightarrow{\Omega}_{R_C/R_P} + \overrightarrow{\Omega}_{R_P/R_M} \end{bmatrix} = \begin{bmatrix} \overrightarrow{\Omega}_{R_P/R_M} \wedge \overrightarrow{PC} \\ \overrightarrow{\Omega}_{R_P/R_M} \end{bmatrix} \tag{B.5}$$

Le calcul de T_{C/R_M} dans R_M nécessite donc la détermination de $\overrightarrow{\Omega}_{R_P/R_M}$ et de $\overrightarrow{PC}$ dans ce repère.

A l'aide de la géométrie du robot, PC_P^R est directement donné par le vecteur $[C_x \quad C_y \quad C_z]^T$. Pour obtenir PC^{R_M} à partir de PC^{R_P} , nous utilisons la matrice de passage $\mathcal{R}_{R_M/R_P}$ comme suit :

$$PC^{R_M} = \mathcal{R}_{R_M/R_P} PC^{R_P} \tag{B.6}$$

Nous obtenons alors :

$$PC^{R_M} = \begin{bmatrix} C_x \cos(\vartheta) - C_y \sin(\vartheta) \\ C_x \sin(\vartheta) + C_y \cos(\vartheta) \\ C_z \end{bmatrix} \tag{B.7}$$

A présent, il nous reste à calculer la vitesse de rotation de R_P par rapport à R_M. Compte tenu de la géométrie du robot, Ω_{R_P/R_M} est donné par le vecteur $[0 \quad 0 \quad \varpi]^T$. A

l'aide de cette dernière équation et des relations précédentes, il est possible de déduire l'expression du torseur cinématique de la caméra dans R_M :

$$\mathcal{T}_{C/R_O}^{R_C} = \begin{bmatrix} V_{C/R_M} \\ \Omega_{R_C/R_M} \end{bmatrix} = \begin{bmatrix} -C_x \sin(\vartheta) - C_y \cos(\vartheta) \\ C_x \cos(\vartheta) - C_y \sin(\vartheta) \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \varpi \tag{B.8}$$

Calcul de $\overrightarrow{MC}$ dans R_M

nous allons décomposer $\overrightarrow{MC}$ à l'aide de la relation de Chasles de la manière suivante :

$$\overrightarrow{MC} = \overrightarrow{MP} + \overrightarrow{PC} \tag{B.9}$$

A l'aide de la géométrie du robot, nous obtenons le résultat suivant :

$$MC = \begin{bmatrix} D_x + C_x \cos(\vartheta) - C_y \sin(\vartheta) \\ C_x \sin(\vartheta) + C_y \cos(\vartheta) \\ h_r + C_z \end{bmatrix} \tag{B.10}$$

Calcul de $T_{C/RO}$ dans R_M

A l'aide des équations (B.3), (B.8) et (B.10), le torseur cinématique de la caméra par rapport au repère de la scène s'exprime dans R_M de la manière suivante :

$$T_{C/RO} = \begin{bmatrix} 1 & -(C_x \sin(\vartheta) + C_y \cos(\vartheta)) & -(C_x \sin(\vartheta) + C_y \cos(\vartheta)) \\ 0 & D_x + C_x \cos(\vartheta) - C_y \sin(\vartheta) & C_x \cos(\vartheta) - C_y \sin(\vartheta) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \dot{q} \tag{B.11}$$

Calcul de $T_{C/RO}$ dans R_C

Il reste maintenant à projeter le torseur dans RC. La matrice de passage entre R_M et R_C est classiquement définie par la relation suivante :

$$\mathcal{R}_{R_M/R_C} = \mathcal{R}_{R_M/R_P}\mathcal{R}_{R_P/R_C} \tag{B.12}$$

Le torseur cinématique de la caméra par rapport à R_O et exprimé dans R_C est donc donné par :

$$\mathcal{T}^{R_C}_{C/R_O} = \begin{bmatrix} \mathcal{R}_{R_M/R_C} & 0_{(3,3)} \\ 0_{(3,3)} & \mathcal{R}_{R_M/R_C} \end{bmatrix} T^{R_M}_{C/R_O} \tag{B.13}$$

où $0_(3,3)$ est une matrice carrée et nulle de dimensions 3. Nous obtenons finalement :

$$\mathcal{T}^{R_C}_{C/R_O} = \begin{bmatrix} V_{\vec{x}_C} \\ V_{\vec{y}_C} \\ V_{\vec{z}_C} \\ \Omega_{\vec{x}_C} \\ \Omega_{\vec{y}_C} \\ \Omega_{\vec{z}_C} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ -\sin(\vartheta) & C_x + D_x\cos(\vartheta) & C_x \\ \cos(\vartheta) & -C_x + D_x\sin(\vartheta) & -C_y \\ 0 & -1 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \dot{q} \tag{B.14}$$

Bibliographie

Sources primaires

A.Angelova L.Matthies, D.Helmick et P.Perona, *Slip prediction using visual information. In Robotics : Science and Systems*, The MIT Press, 2006.

A.DeLuca et Giuseppe Oriolo, *Modeling and control of nonholonomic mechanical systems*, Dipartimento di Informatica e Sistemistica Università degli Studi di Roma "La Sapienza" Via Eudossiana 18, 00184 Roma, Italy.

A.Gerevini et I.Serina, *LPG : A planner based on local search for planning graphs with action costs*, International Conference on Artificial Intelligence Planning Systems, pages : 13–20, 2002.

A.Pruski, *Robotique mobile - La planification de trajectoire*, Hermes, 1996.

B.BAYLE, *Robotique mobile*, Ecole Nationale Supérieure de Physique de Strasbourg Université Louis Pasteur. Année 2008–2009.

D.Bellot, *Contribution à l'analyse et à la synthèse de schémas de commande référencée vision*, PhD thesis, Université Paul Sabatier - Toulouse III, 2002.

D.S.Naidu, *Optimal Control Systems*, Idaho State University Pocatello. Idaho, USA.June 2002.

F.Ben-Amar, *Modèles de comportement des véhicules tout terrain pour la planification physico-géométrique de trajectoires*, Thèse de doctorat, Université Pierre et Marie Curie, Paris, juillet 1994.

F.Chaumette, *Potential problems of stability and convergence in imagebased and position-based visual servoing*, The Confluence of Vision et Control, pages : 66–78. LNCIS Series, No 237, Springer-Verlag, 1998.

J.Gaspar, N.Winters et J.Santos-Victor, *Vision-based navigation and environmental representations with an omni-directional camera*, IEEE transactions on robotics et automation, vol. 6, no. 6, pages : 890–898, 2000.

K.Iagnemma, C.Ward, *Classification-based wheel slip detection and detector fusion for outdoor mobile robots*, In IEEE International Conference on Robotics et Automation, pages :2730–2735, Rome, Italie, avril 2007.

M.Beetz et al., *Integrating active localization into high-level robot control systems*, Robotics et Autonomous Systems, Pages : 206–220, 1998.

N.J.Nilson, *A mobile automation : An application of artificial intelligence techniques*, In IJCAI, pages : 509–520. IJCAI, 1969.

O.Booij B.Terwijn, Z.Zivkovic et B.Krose, *Navigation using an appearance based topological map*, In IEEE Int. Conf. on Robotics et Automation, pages : 3927– 3932, Rome, Italy, 2007.

R.Fierro, *Control of a Nonholonomic Mobile Robot : Backstepping Kinematics into Dynamics*, Automation et Robotics Research Institute The University of Texas. September 20, 1996.

Rives, Pissard-Gibollet et, *Applying visual servoing techniques to control a mobile hand-eye system*, In IEEE Int., Conf. on Robotics et Automation, Nagoya, Japan, 1995. (cité en page 10).

S.A.Wadoo, *Feedback Control and Nonlinear Controllability of Nonholonomic Systems*, Thesis submitted to the Faculty of the Virginia Polytechnic Institute et State University in partial fulfillment of the requirements for the degree of Masters of Science in Electrical Engineering. Jan 9, 2003.

S.Thrun, *Learning metric-topological maps for indoor mobile robot and navigation*, Artifcial Intelligence, pages : 59-71, 1998.

T.Peynot, *Sélection et contrôle de modes de déplacement pour un robot mobile autonome en environnements naturels*, Thèse de doctorat, Institut National Polytechnique de Toulouse, juillet 2006.

V.Cadenat, *Commande référencée multi-capteurs pour la navigation d'un robot mobile*, PhD thesis, Université Paul Sabatier - Toulouse III, 1999.

V.Kucera, *The Discrete Riccati Equation of Optimal Control*, Ústav teorie informace a automatizace ČSA V (Institute of Information Theory et Automation — Czechoslovak Academy of Sciences), Vyšehradská 49, Praha 2. Année 1972.

V.Sangveraphunsiri, M.Thianwiboon et, *Traction control of a rocker-bogie field mobile robot*, Thammasat International Journal of Science et Technology, pages : 48–59, 2005.

Sources secondaires

[Bailey1999]-T.Bailey, J. K.Rosenblatt et H. F.Durrant-Whyte, *Robust Distinctive Place Recognition for Topological Maps*, International Conference on Field et Service Robotics, 1999. (cité en pages 27 et 28).

[Chapoulie2012]-Alexandre Chapoulie, Patrick Rives et David Filliat, *Topological segmentation of indoors/outdoors sequences of spherical views*, IEEE Conference on Intelligent Robots et Systems, IROS, 2012. (cité en page 27).

[Dedeoglu1999] G.Dedeoglu, M.Mataric et G.S.Sukhatme.Incremental, *Online topological map building with a mobile robot*, Mobile Robots XIV - SPIE, pages : 129–139, 1999. (cité en page 27).

[Engelson1992]-S.P.Engelson et D.V.McDermott, *Error correction in mobile robot map learning*, IEEE International Conference on Robotics et Automation, 1992. (cité en page 28).

E.W.Dijkstra, *A note on two problems in connexion with graphs*, Numerische Mathematik, vol. 1, pages : 269–271, 1959.

[Gaussier2000]-P.Gaussier C.Joulain, J.P.Banquet et S.Lepretre, *Topological segmentation of indoors/outdoors sequences of spherical views*, IEEE Conference on Intelligent Robots et Systems, IROS, 2012. (cité en page 27).

[Kunz1997] C.Kunz, T.Willeke et I.Nourbakhsh, *Automatic mapping of dynamic office environments*, IEEE International Conference on Robotics et Automation (ICRA), 1997. (cité en page 27).

R.T.Vaughan, B.P. Gerkey et A. Howard, *On device abstraction for portable, reusable robot code*, Proceedings IEEE/RSJ International Conference on Intellignet Robots et Systems, pages : 2421-2427, 2003.

[Ullah2008] M.M.Ullah A.Pronobis, B.Caputo et J.Luo, *Towards robust place recognition for robot localization Robotics and Automation*, ICRA 2008. (cité en page 27).

[Ulrich2000] I.Ulrich, I.Nourbakhsh, *Appearance-based place recognition for topological localization*, International Conference on Robotics et Automation, 2000. (cité en page 27).

[Wichert1998]-G.VonWichert, *Mobile robot localization using a self-organised visual environment representation*, Robotics et Autonomous Systems, 1998. (cité en page 27).

[Yamauchi1996]-B.Yamauchi et R. Beer, *Spatial learning for navigation in dynamic environments*, Systems, Man, et Cybernetics, Special Issue on Learning Autonomous Robots, 1996. (cité en page 27).