!º ƒº !º !! º !Œº ƒº ! Œº !º !º ƒ!º !Œº ƒ!º !Œº ! º !Œº º ƒ! º ! º º º ! ! !Œ

République Algérienne Démocratique et Populaire

وزارة التعليم العالي والبحث العلمي

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA DE RECHERCHE SCIENTIFIQUE

جامعة الحاج لخضر باتنة

UNIVERSITE EL-HADJ LAKHDAR-BATNA

Faculté des sciences de l'ingénieur
Département Electronique

Mémoire préparé par :

ALIA KAMEL

Ingénieur d'état en Electronique

Sujet Proposé par :

Dr. S. AOUGHELLANET

Maître de conférence au département d'électronique – université de Batna

En vue de l'obtention du diplôme de :

Magistère en Robotique

Thème

PLANIFICATEUR DE TRAJECTOIRE NEURONAL

Soutenu le : / /2011

Membre du jury :

Dr. Yassine ABDESSEMED	M.C	Université de Batna	Président
Dr. Saïd AOUGHELLANET	M.C	Université de Batna	Rapporteur
Dr. Djamel MELAAB	M.C	Université de Batna	Examinateur
Dr. Abderraouf MESSAI	M.C	Université de Constantine	Examinateur

A la mémoire de mon frère.

REMERCIEMENTS

Je tiens à remercier mon encadreur qui a toujours su être présent lorsque j'avais besoin de lui, pour ses conseils qui mon toujours guidé le long de l'élaboration de ce travail et pour sa patience à mon égard malgré le retard enregistré pour achever le travail.

A cette occasion je tiens à témoigner ma reconnaissance à tous mes amis qui m'ont toujours soutenu et dont le retard de l'achèvement de ce travail dérangeait beaucoup !

Mes remerciements sincères vont aussi aux honorables membres du jury qui ont bien voulu examiné ce modeste travail.

SOMMAIRE

INTRODUCTION GENERALE

Chapitre 1
GENERALITES

Chapitre 2
LE MANIPULATEUR PUMA 560

Chapitre 3
LES RESEAUX DE NEURONES ARTIFICIELS

Chapitre 4
PLANIFICATION DE TRAJECTOIRE

INTRODUCTION GENERALE

La robotique est un domaine relativement nouveau de la technologie moderne allant au-delà de l'ingénierie traditionnelle.
La compréhension de la complexité des robots et de leurs applications impliquele génie électrique, le géniemécanique, les systèmes industriels, l'informatique, l'économie et les mathématiques.
La robotique a grandiénormément cesdernières vingt années grâce à l'avance rapide de l'ordinateur et de la technologie descapteursainsi qu'au progrès théoriques colossaux dans le domaine du contrôle et du traitement de l'image.

Notre sujet traite un brasde robot industriel fonctionnant dans des environnements structurés d'usine.

PLANIFICATION DE TRAJECTOIRE

Le problème de planification de trajectoire peut s'énoncer comme suit :
À partir d'une description des frontières des obstacles dans l'espace cartésien et du modèle géométrique du robot, on doit trouver une trajectoire optimale sans collision amenant le manipulateur d'une configuration de départ à la configuration finale.
Pour résoudre ce problème on peut distinguer deux approches. L'approche dite *Locale* qui est beaucoup moins coûteuse en temps de calcul et ne suppose pas une connaissance complète de l'environnement (des capteurs sont utilisés pour guider la planification au fur et à mesure des déplacements). Cependant, du fait de son caractère local, elle ne garantit pas l'arrivée à la configuration finale.
L'approche dite *Globale* permet une recherche exhaustive de la trajectoire de coût minimum au sens d'un critère donné. Cette méthode trouve toujours la solution lorsqu'elle existe, mais elle est très coûteuse tant en temps de calcul qu'en espace mémoire. Ceci ne présente pas un handicap dans le cas où la trajectoire doit être

calculée dans un environnement fixe, le calcul est fait hors ligne et une fois pour toute.

En pratique, ces deux approches sont utilisées en collaboration, afin de bénéficier des avantages de chacune. Ceci constitue l'approche dite *Mixte*, qui permet de trouver la trajectoire quand elle existe en un temps minimum.
Finalement l'approche neuronale [8] utilisant les réseaux de neurones avec ou sans apprentissage constitue une solution efficace pour ce problème. Cette technique connexionniste qui bénéficie d'une architecture massivement parallèle permet la détermination de la trajectoire optimale en temps réel.

Les algorithmes existants [9]

Dans ce qui suit on va présenter certaines de ces méthodes en expliquant brièvement leur principe ainsi que leurs avantages et leurs inconvénients.

Méthode par décomposition de l'environnement en cellule.

Une première méthode de planification de trajectoire consiste à décomposer l'environnement du robot en cellules .Latombe 1991 [10] (en un ensemble de régions connexes adjacentes). Il suffit ensuite de trouver un algorithme travaillant sur la discrétisation de l'environnement. Là, différentes techniques existent, par exemple, la partition de Voronoï .Choset 1996 [11] ou les graphes de visibilité .Chazzzelle et Guibas 1989 [12].
Dans ces cas, l'environnement discrétisé est représenté dans un graphe et trouver une trajectoire revient à chercher un chemin dans un graphe, problème qui peut être facilement traité informatiquement .Shin et McKay 1986 [13].
D'autres algorithmes de planification travaillent dans un environnement discrétisé. Parmi les plus connus, on peut citer A* .Hart 1968 [14] qui planifie une trajectoire optimale connaissant l'environnement ou D* . Stentz et Anthony 1994 [15] qui planifie une trajectoire dynamiquement, ce qui lui permet de découvrir son environnement au fur et à mesure.
Les avantages de tels algorithmes sont entre autre la facilité d'implémentation et de représentation. On travaille sur une grille et chaque case est en 8-connexité. C'est-à-

dire que d'une case, on a que 8 choix de déplacement possible (26 dans l'espace cartésien).

Par contre, ces algorithmes possèdent de gros défauts. Notamment le fait que les contraintes cinématiques du robot ne sont pas prises en compte (accélération, vitesse, vitesse angulaire maximale, encombrement etc.), ce qui autorise le planificateur à trouver une trajectoire qui ne sera peut-être pas exécutable par le robot.

De plus, le fait de discrétiser l'espace de recherche limite beaucoup le nombre de trajectoire possible.

La méthode des champs de potentiel

Les champs de potentiel .Khatib 1986 [16] est une méthode assez originale qui assimile le robot à une particule soumise à un champ de forces répulsives et attractives.

Un obstacle génère un champ de potentiel répulsif tandis que l'objectif à atteindre génère un champ de potentiel attractif. L'algorithme calcul donc un vecteur résultant qui indiquera au robot comment effectuer son déplacement.

Cet algorithme est totalement réactif et peut donc être très facilement implémenté en temps réel. Cependant, comme l'environnement est très peu souvent totalement convexe, cette méthode entraine facilement le robot dans des minimas locaux. De plus, les problèmes d'oscillation peuvent, dans certain cas, être constatés.

Ici non plus, on ne tient pas compte des contraintes du robot.

La méthode de la fenêtre dynamique

Cette méthode .Fox et al. 1997 [17] travaille dans l'espace de commande du robot. La méthode calcule les vitesses possibles du robot pour que celui-ci ne rentre en collision avec aucun obstacle. Une fois l'espace de recherche des vitesses calculées, l'algorithme trouvera la commande optimale à envoyer au robot en minimisant une fonction de coût sur ce domaine de recherche (minimisation du temps de parcourt, de l'énergie dépensée, maximisation de la vitesse etc.)

Cette méthode un peu plus abstraite permet de prendre en compte les contraintes cinétiques du robot. Par contre, elle manque en flexibilité, ce qui rend difficile son implémentation dans un cadre multi-robot.

La méthode de la bande élastique

Une méthode .Quinlan 1994 [18] qui tend une "bande élastique" entre le robot et l'objectif. Cette bande étant capable de se déformer en présence d'un obstacle, celle-ci génère donc une trajectoire envisageable par le robot.

La planification par logique floue

Cette méthode de planification se base sur la logique floue.Zadeh 1965 [19] : Chaque grandeur physique (par exemple une distance d'un obstacle) est convertie en une variable linguistique (petit, moyen ou grand par exemple).

Alors qu'en logique booléenne classique, une distance serait soit petite, soit grande, la logique floue autorise une distance à être à la fois grande et petite (20% petite et 80% grande) suivant une fonction d'appartenance précise.

La planification floue va donc raisonner non pas sur des grandeurs physiques, mais sur des variables linguistiques. On appliquera des règles précises sur ces variables. Finalement, on obtiendra la commande sous la forme de variables linguistiques que l'on reconvertira en grandeurs physiques (lors de la défuzzification).

Cette approche permet de ne pas résonner au niveau de la grandeur physique, mais plus comme un humain le ferait.

Exemple : Si la distance au feu est faible et le feu est rouge, je freine fort.

Ici, on ne raisonne pas sur la valeur de la distance directement. (On ne va pas se dire, dans la vie de tous les jours : *si je suis à moins de 6mètre du feu, alors, je fais telle action*. Un humain raisonnerait plutôt comme ça : *si je suis moyennement proche, si je suis assez proche, si je suis très proche alors, je fais telle action*)

La logique floue permet donc d'obtenir des trajectoires progressives qui évitent les obstacles.

La planification par juxtaposition de splines polynomiales

L'algorithme de Mickel Defoort .Defoort 2007, 2009 [20] [21] génère des bouts de splines (une fonction définie par morceaux par des polynômes) polynomiales qui

respectent les contraintes cinématiques du robot. On peut simplifier l'algorithme en le forçant à générer non pas des splines, mais simplement des polynômes.La trajectoire du robot est alors assimilée à la juxtaposition de plusieurs polynômes.
Un bout de trajectoire est alors représenté par un système paramétré de deux polynômes du troisième degré. Les coefficients des degrés zéro et un fixent la position initiale et la vitesse initiale du robot (ce qui permet de garder la continuité entre les différents polynômes). Ainsi, il ne reste plus qu'à optimiser les coefficients du second et du troisième degré du polynôme à l'aide d'algorithmes d'optimisation comme CFSQP .Lawrence, Zhou et Tits [22].

Cette méthode permet donc de respecter les contraintes cinématiques du robot. De plus, l'optimisation des paramètres des polynômes peut prendre en compte des fonctions de coût, ce qui permet de choisir selon quelle critère l'on décide qu'une trajectoire est optimale ou non.

DKP : Deterministic Kinodynamic Planning

Cette méthode de planification .Gaillard et al 2009 [23] est assez complexe, mais permet de respecter toutes les contraintes du robot et d'éviter les problèmes des minimas locaux.
DKP peut être décomposé en deux parties. La première planifie localement des bouts de trajectoires polynomiales (d'ordre deux) sur différents temps de parcourt. Pour trouver un bout de trajectoire optimal, il suffit de chercher une solution optimale dans l'espace des paramètres des polynômes via une résolution géométrique des contraintes. Avec ce planificateur local, on construit un arbre de solutions locales.
La seconde partie de l'algorithme est un planificateur global basé sur un A*-like qui va choisir les bouts de trajectoire à adopter pour résoudre le problème.
L'avantage de cet algorithme est qu'il génère une grande diversité de bouts de polynôme, ce qui permet de trouver une solution même dans les environnements très complexes.

Objectif du travail

Utiliser un réseau de neurone non supervisé (sans apprentissage) pour définir une trajectoire optimale en temps réeldans un environnement statique discrétisé.

Organisation du mémoire

Chapitre 1 : Généralités sur la robotique.

Chapitre 2 : Le manipulateur Puma 560.

Chapitre 3 : Les réseaux de neurones artificiels.

Chapitre 4 : Planification de trajectoire

Chapitre 1
INTRODUCTION

1.1 Robotique

La robotique est un domaine relativement nouveau de la technologie moderne allant au-delà de l'ingénierie traditionnelle.

La compréhension de la complexité des robots et de leurs applications impliquele génie électrique, le géniemécanique, les systèmes industriels, l'informatique, l'économie et les mathématiques.

La robotique a grandiénormément cesdernières vingt années grâce à l'avance rapide de l'ordinateur et de la technologie descapteursainsi qu'au progrès théoriques colossaux dans le domaine du contrôle et du traitement de l'image.

Notre sujet traite un brasde robot industriel fonctionnant dans des environnements structurés d'usine.

1.2 Histoire de Robotique

Le terme « ROBOT » a été présenté pour la première fois dans le vocabulaire par l'écrivain tchèque KarelCapek dans son œuvre « les Robots Universels de Rossum » en 1920, le mot « ROBOTA » étant le mot tchèque désignant le travail. Depuis lors le terme a été appliqué pour désigner une grande variété de dispositifs mécaniques, tel que lestéléopérateurs, véhicules sous-marins, explorateurs autonomes, etc.

Pratiquement tout ce qui fonctionne avec un degré d'autonomie, sous le contrôle d'un ordinateur, est appelé robot.

Dans ce mémoire le terme « robot » signifiera un bras manipulateur industriel contrôlé par ordinateur.

Un tel dispositif, quoique loin des robots de science-fiction, est un système électromécanique extrêmement complexes dont la description analytique exige des

méthodes avancées qui présentent un problème intéressant de recherche et de challenge. Figure 1.1.

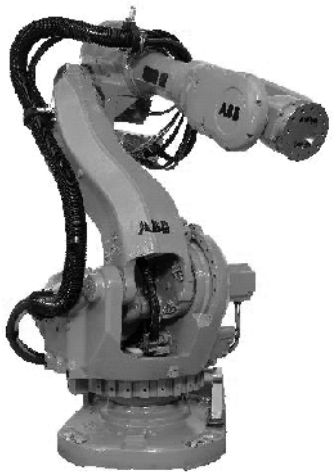


Figure 1.1: Le Robot ABB IRB6600.

Une définition officielle d'un tel robot vient de l'Institut Américain de Robotique (RIA) :

Un robot est un manipulateur reprogrammable multifonctionnel conçu pour déplacer du matériel, des pièces, des outils, ou des dispositifs spécialisés par des mouvements variables programmés pour l'exécution d'une variété de tâches[1].

L'élément clé dans cette définition est lareprogrammabilité des robots. C'est l'ordinateur qui donne au robot son utilité et son adaptabilité.
La révolution prétendue de la robotique n'est, en fait, qu'une partie de la grande révolution de l'ordinateur.

Même la version restreinte des robotsa plusieurs particularités qui en font une solution attirante dans un environnement industriel.
Parmi les avantages souvent cités en faveur de l'introduction des robots sont : une maind'œuvre diminuée, une précision et productivitéaccrues ainsi qu'une bonne flexibilité comparésaux machines spécialisées et aux travailleurs humains.

Le robot, comme nous l'avons défini, est né en fait du mariage de deux technologies précédentes :
Celui destéléopérateurs et celui des fraiseuses contrôlées numériquement.
Les téléopérateurs, ou les dispositifs maître-esclave, ont été développés pendant la deuxième guerre mondiale pour manipuler les matériaux radioactifs.
Le contrôle numérique par ordinateur (CNC) a été développé à cause de la haute précision exigée dans l'usinage de certains articles, comme les composants de haute performance sur les avions.
Les premiers robots ont essentiellement combiné les liens mécaniques du téléopérateur avec l'autonomie et la programmabilité des machines CNC.

Plusieurs faits marquants l'histoire de la technologie des robots sont présentés ci-dessous.

Evénements marquants l'histoire de la Robotique

1947 –Développement du premier téléopérateurasservi alimenté électriquement.

1948 - Développement d'un téléopérateur incluant des réactions de force.

1949 - La recherche sur la fraiseuse numériquement contrôlée est amorcée.

1954 - George Devol conçoit le premier robot programmable.

1956 - Joseph Engelberger, un étudiant de physique de l'université de la Colombie, achète les droits du robot de Devol et fonde la Société « Unimation ».

1961 - Le premier robot « Unimate » est installé à Trenton auNew Jersey dans l'usine deGeneralMotorss pour unepresse métallique.

1961 - le premier robot incorporant des réactions de force est développé.

1963 - le premier système de vision de robot est développé.

1971 - le Bras Stanford est développé à l'université Stanford.

1973 - le premier langage de programmation robotique (WAVE) est développé à Stanford.

1974 - Cincinnati Milacron a présenté le robot T3 avec le contrôle d'ordinateur.

1975 - Unimation Inc. enregistre son premier bénéfice financier.

1976 –« Le système compliant actif à centre éloigné » (Remote Center Compliance RCC) est développé au Laboratoires de Drapier à Boston pour l'assemblage de pièces.

1976 - Les brasde robot sont employés sur le Viking I et II dans la recherche spatiale et l'atterrissage sur de la planète Mars.

1978 - Unimation présente le robot de PUMA, basé sur des conceptions de General Motors

1979 - la conception du robot SCARA est présentée au Japon

1981 - Le premier robot Direct Drive est développé à l'Université Carnegie-Mellon

1982 - FANUC du Japon et le General Motors forment la société GMFANUC pour commercialiser les robots en Amérique du Nord.

1983 –la société AdeptTechnologyest fondée et commercialise avec succès le robot Direct Drive.

1986 - le robot sous-marin, Jason, de l'Institut OcéanographiqueWoods-Hole, explore l'épave du Titanic, trouvé une année plus tôt par le docteur Robert Bernard.
1988 –Le Stäubli-Groupachète Unimation de Westinghouse.

1988 - La Société IEEE Robotics and Automation est formée.

1993 - le robot expérimental, ROTEX, de l'Agence Spatiale allemande (DLR) est emporté à bord de la navette spatiale Colombie et exécute une variété de tâches en mode téléoperéet en modes autonome.

1996 - Honda dévoile son robot Humanoïde; un projet commencé dans secret en 1986

1997 - la première compétition de football de robot, RoboCup-97, est tenue dans Nagoya au Japon. 40 équipes du monde entier sont représentées.

1997 - le robot mobile Sojourner voyage pour la planète Mars dans la mission de la NASAPathFinder.

2001 - Sony commence la production en masse du premier robot de compagnie, un robot chien nommé Aibo qui signifie ami ou compagnon en japonais.

2001 - le Système de commande à distance de station spatiale (SSRMS) est lancé dans l'espace à bord de la navette spatiale Endeavor pour faciliter la construction continue de la Station spatiale.

2001 –la première télé-chirurgie exécutéepardes chirurgiens à New York sur une femme se trouvant à Strasbourg en France, ces derniers ont exécuté unelaparoscopie (déplacement de la vésicule biliaire).

2001 - Les robots sont employés pour chercher des victimes au World Trade Center après l'attentat du 11 septembre 2001.

2002 - Le robot Humanoïde de Honda ASIMO (le premier robot pouvant marcher indépendamment avec des mouvements relativement lisses et pouvant monter à l'escalier) sonne la cloche d'ouverture deBourse aux actions de New York le 15 février.

2003- SONY sort l'AIBO ERS-7, la 3ème génération de l'animal de compagnie robotisé.

2004 - Epson sort le robot le plus petit, pesant 0.35 onces (10 gr) et mesurant 2.8 pouces (70 mil) dans la hauteur, le Robot Micro Volant est dévoilé comme le robot hélicoptère le plus léger et le plus petit au monde. La société espère qu'il sera employé comme "une caméra volante" pendant des catastrophes naturelles.

2005 - Les chercheurs à l'université *Cornell* New YorkUSAprétendent avoir construit le premier robot pouvant se dupliquer.

2005 - Depuis le 6 octobre dans l'Aquarium de Londres trois poissons robotisés nagent dans un réservoir spécialement conçu.

Les premières applications couronnées de succès des robots manipulateurs incluaient généralement des tâches de tri, de transfert de matériels, de moulage par injection ou d'estampage. Dans ces opérations le robot n'avait qu'à suivre simplement une presse pour décharger, transférer ou bien empilerle produit fini.
Ces premiers robots avaient la capacité d'être programmé pour exécuter un ordre de mouvements donnétel que le déplacement d'un emplacement A fermant unepince, vers un emplacement B, etc., mais n'avaient aucune capacité de détection externe !
En raison de l'interaction accrue du robot avec son environnement, des applicationsplus complexes, telles que le soudage, le meulage, l'ébavurage et l'assemblage exigent non seulement des mouvement plus complexe mais aussi une forme de détection externe telle quela vision, le toucher, oupercevoirla force exercée.
Il est bien connu ces derniers temps que les applications importantes de robots ne sont en aucun cas limitées à ces emplois industriels où le robot remplace directement un opérateur humain, les applications robotiques se sont imposées dans beaucoup de secteurs où l'utilisation de gens est peu pratique, indésirable voire impossibles. Parmi ceux-ci on peut citer :

- l'exploration sous-marine et planétaire.

- la récupération de satellites.
- la réparation ou le désamorçage de dispositifs explosifs.
- le travail dans les environnements radioactifs.

Finalement, les prothèses, comme des membres artificiels, sont des dispositifs robotisés exigeant des méthodes d'analyse et de conception semblable à ceux des manipulateurs industriels.

1.3 Composants et structure des Robots

1.3.1 Représentation Symbolique des Robots

Les Robots manipulateurs sont composés de segments connectés par des liaisons dans une chaîne cinématique.

Les liaisons sont la ROTOÏDE typiquement rotative ou la PRISMATIQUE purement linéaires.

Une liaisonrotoïdetelle une charnièrepermet la rotation relative entre deux segments alors qu'une liaison prismatique permet un mouvement linéaireentre deux segments. Nous employons la convention (R) pour la représentation de liaisonsrotoïdes et(P) Pour des liaisons prismatiques.La représentation symbolique est donnée dans la figure 1.2

Chaque liaison représente l'interconnexion entre deux segments notés ' l_i' et ' l_{i+1} '. On note l'axe de rotation d'une liaison rotoïde ou l'axe le long duquel une liaison prismatique glisse par ' z_i'.

Si la liaison est l'interconnexion du segment i et i+1 les variables de liaison notées par ' θ_i' pour une liaison de rotoïde et ' d_i' pour la liaison prismatique, représentent le déplacement relatif entre deux segments adjacents.

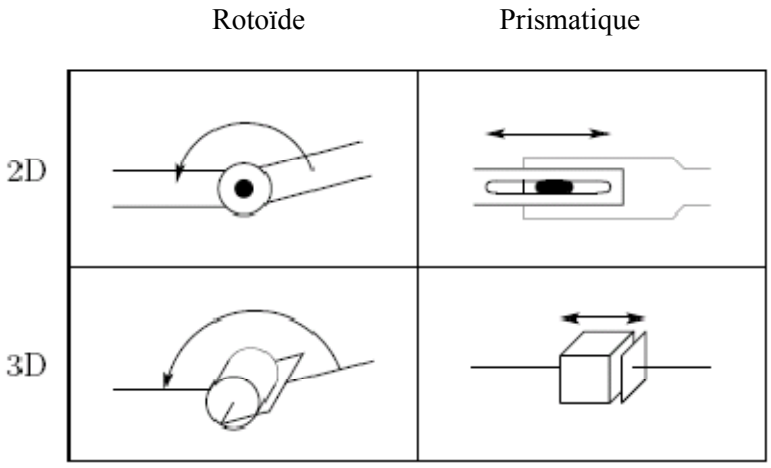


Figure 1.2: Représentation symbolique des articulations du Robot

1.3.2 Degrés de liberté et espace de travail

Le nombre de liaisons détermine « le degré de liberté » (DDL) du manipulateur. Typiquement un manipulateur doit posséder au moins six DDL indépendants : trois pour le positionnement et trois pour l'orientation. Avec moins de six DDL le bras ne peut pas atteindre chaque point dans son environnement de travail avec une orientation donnée.

Certaines applications nécessitant d'atteindre des cibles autour ou derrière des obstacles exigent plus de six DDL.

La difficulté de contrôler un manipulateur augmente rapidement avec le nombre de segments. Un manipulateur ayant plus de six segments et classé cinématiquement redondant.

On convient d'appeler les trois premiers DL d'un robot (partant de la base du robot) « *le Porteur du robot* ».Les DL résiduels forment le poignet, caractérisé par des dimensions beaucoup plus petites et une plus faible masse.Dans la pratique, « le Poignet » de type rotule est très répondu.Le robot obtenu en associant un porteur à 3 DL à un poignet rotule est la structure la plus classique à 6 DL. Elle permet d'assurer un découplage entre la position et l'orientation.

La zone de travail d'un manipulateur est le volume total balayé de par l'organe terminal « *effecteur* ».

La zone de travail est fonctionaussi bien de la géométrie du manipulateur que des contraintes mécaniques des liaisons. Par exemple, une liaison rotoïdepeut être limitée à moins de 360°de mouvement.

La zone de travail est souvent composée d'une zone de travail accessible et une zone de travail adroite.

La zone de travail accessible est l'ensemble entier de points accessibles par le manipulateur, tandis que la zone de travail adroite est l'ensemble des points que le manipulateur peut atteindre avec une orientation arbitraire de l'effecteur.

Evidemment la zone de travail adroite est un sous-ensemble de la zone de travail accessible.

1.3.3 Classification des Robots

Les robots manipulateurs peuvent être classés suivant plusieurs critères, tel que leur source de puissance, ou la façon avec laquelle les liaisons sont actionnées, leur géométrie, la structure cinématique, leur domaine d'application, ou leur méthode de commande.

Une telle classification est utile principalement pour déterminer quel est le robot adéquat pour accomplir une tâche donnée.

Par exemple, un robot hydraulique ne serait pas convenable pour manipuler des produits alimentaires ou pour faire le ménage dans une pièce, alors qu'un robot SCARA(décrit après) ne serait pas convenable pour travaillerdans une fonderie.

La source de puissance

Typiquement, les robots sont soit électriquement, hydrauliquement, ou pneumatiquementalimentés.

Les actionneurs hydrauliques sont imbattables par leur vitesse de réponse et par leur capacité à produire un couple consistant, de cela les robots hydrauliques sont utilisés principalement pour manier les chargeslourdes.

L'inconvénientmajeur des robots hydrauliques estle liquide hydraulique qui requiert plusd'équipements périphériques, tel que lespompes, et donc nécessitent plus d'entretien. D'autre part les robots hydrauliques sont pratiquementles plus bruyants.

Les robots commandés par des servomoteurs DC ou ACsont de plus en pluspopulaires depuis qu'ils sont moins chers, plus propres et plus calmes.
Les robots pneumatiques sont peu coûteux et simples mais ne peuvent pas êtrecontrôlés avec précision et donc limités àleur champ d'application.

Le domaine d'application

Ledomaine leplus large projeté pour les futures applications des robots est dans l'assemblage.Donc, les robots sont souvent classés par domaine d'application: dans l'assemblage ou non.
Les robots d'assemblage ont tendance à être petit, commandé électriquement et sont dans leur conception soit rotoïde ou SCARA (décrit ci-après).
Les principaux domaines d'applications, jusqu'à présent, des robots de *non-assemblage* ont été dans la soudure, la peinture par spray, le maniement de matériels, et la charge et décharge de machines.

La méthode de Commande

Les robots sont classifiés par la méthode de contrôle : en asservi et non-asservi.
Les premiers robots n'étaientpas asservis.
Ces robots sont essentiellement des dispositifs en boucle ouverte dont le mouvement est limité à des points d'arrêt mécaniques déterminés au préalable. Ils sont utilisésprincipalement pour le transfert de matériels.
De fait et conformément à la définition donnée précédemment, ces robots « aux arrêts fixes »ne peuvent être qualifiés de robots.
Les robots asservisutilisent une boucle ferméecommandée par ordinateur afin de déterminerleur mouvement et sont ainsi capables d'être vraiment multifonctionnels et reprogrammables.
Les robots asservis sont plutôt classés suivant lamanière avec laquelle le contrôleur pilotel'organe terminal.
Le type le plus simple de cette classe de robots est le robot point à point.
Un robot point à point peut « apprendre » une série discrète de points mais sans aucun contrôle du chemin de l'organe terminal entre ces points.

On apprend habituellement à de tels robots une série de points avec un « teach pendant ».

Les points sont alors mémoriséset «rejoués »point par point.Ces robots sont très limités dans leur domaine d'application.

Par contredans les robots à trajectoire continue, le chemin tout entier de l'organe terminal peut être contrôlé.

Par exemple,l'organe terminal du robot peut être « enseigné » à suivre une droite entre deux points ou même à suivre un contour tel qu'une trajectoire de soudure.

En outre, la vitesse et/ou l'accélération de l'organe terminal peuvent souventêtre contrôlés.

La géométrie

La plupart des manipulateurs industriels de nos jours ont six ou moins de degrés de liberté.

Ces manipulateurs sont habituellement classés cinématiquement sur la base des trois premières liaisons du bras, le poignet étant traité séparément.

La majorité de ces de manipulateurs coïncide avec un des cinq types géométriques:

Articulé ou anthropomorphes(RRR), Sphérique(RRP), SCARA (RRP), Cylindrique (RPP), ou Cartésien (PPP).

Chacun de ces cinq configurations est un robot série et sera discuté en détails plus bas.

Dans une sixième,distincte et fondamentale classesont les manipulateurs appelés « parallèles ».

Les robots parallèles (basés sur la structure de Stewart) : l'organe terminal est relié à une base mécanique fixée par plusieurs chaînes parallèles.

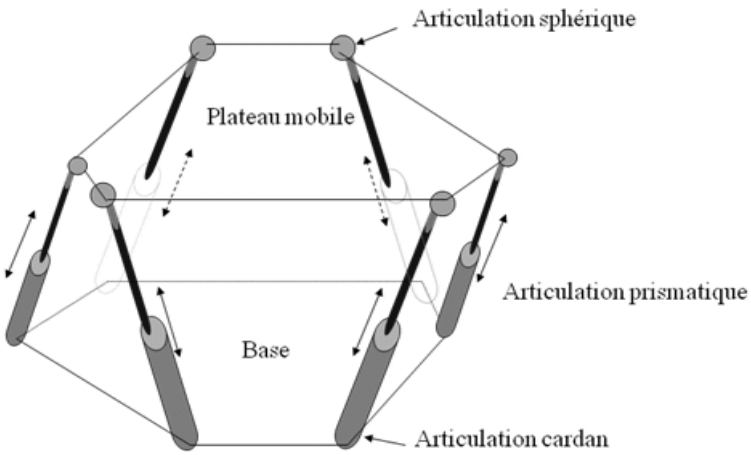


Figure 1.3: Structure de Stewart

1.3.4 Les configuration les plus utilisées

Configuration Articulée (RRR)

Le manipulateur articulé appelé aussi le manipulateur rotoïde, ou anthropomorphe est un design des plus utilisé.

Un exemple de cette configuration est le robot articulé d'ABB IRB1400 montréplusbras (Figure 1.4) et le robot Motoman SK16 (Figure 1.5).

Figure 1.4: LeRobot ABB IRB1400.

Dans ces deux exemplesl'axede l'articulation z2 et parallèleà z1 et les deux (z1 et z2)sont perpendiculaires à z0. La structure et la terminologieassociéesà ce type de manipulateurest montrée dans la Figure 1.5. Son espace de travail dans la Figure 1.6. La configuration rotoïde permet relativement une grande liberté de mouvement dans un espace assez compact.

Figure 1.4: Le Manipulateur Motoman SK16.

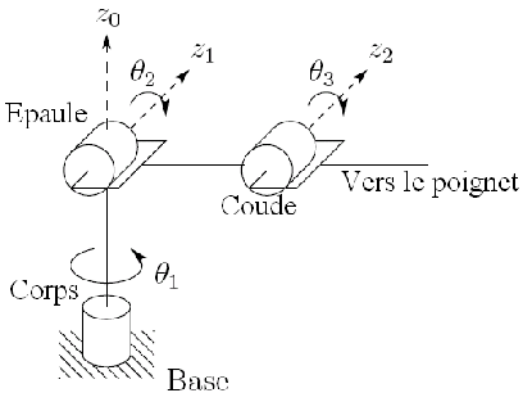


Figure 1.5: Structure du porteur.

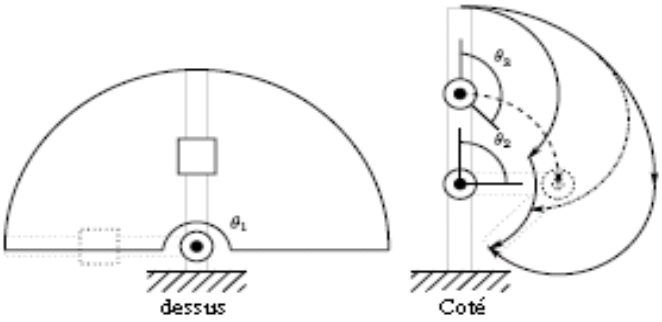


Figure 1.6: Espace de travail du manipulateur.

Configuration Sphérique (RRP)

En remplaçant la troisième articulation (le coude) dans la configuration rotoïde par une prismatique on obtient la configurationsphériquemontrée dans la Figure 1.7. Le termeconfiguration sphériquedérive du fait queles coordonnées sphériquesdéfinissantla position de l'organe terminal,en respectant que l'origine de la base coïncide avec l'intersection des deux axes z1 et z2,sont les mêmes que ceux des trois premières articulations. La figure 1.8 montrele brasStanford, un des robots sphériques les plus connus. L'espace de travail du manipulateur sphériqueest montré dans la Figure 1.9.

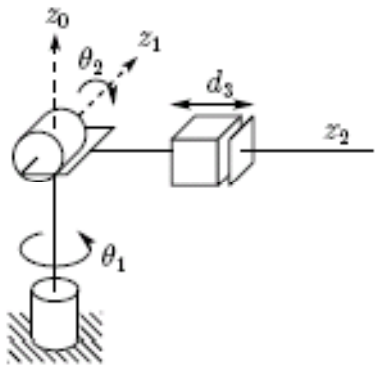


Figure 1.7: Configuration sphérique.

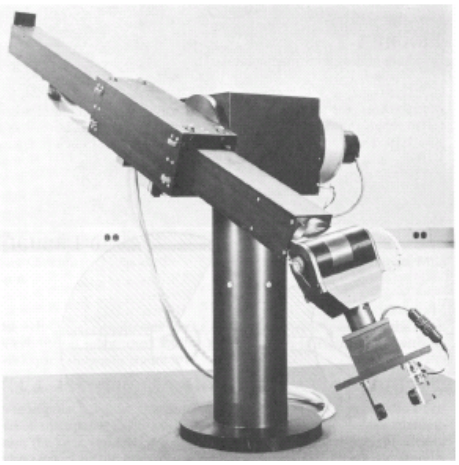


Figure 1.8: le bras Stanford.

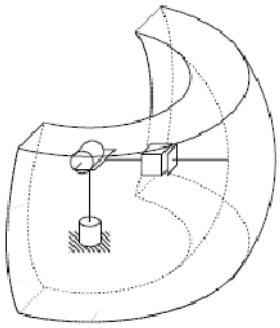


Figure 1.9: Espace de travail du manipulateur sphérique.

Configuration SCARA (RRP)

Le soi-disant. SCARA (pour SelectiveCompliantArticulated Robot for Assembly) montré dans la figure 1.10 est une configuration très connue, comme son nom sous-entend spécialement conçue pour les opérations d'assemblage.Quoique le robot SCARA a une structure RRP, il est tout à fait différentde la configuration sphérique ni dans l'apparence ni dans le domaine d'application.

A la différence du design sphérique qui a z0, z1, z2 mutuellement perpendiculaires, le robot SCARA a z0, z1, z2 parallèles. La figure 1.11 montre le manipulateur Epson E2L653S, unmanipulateur de ce type. L'espace de travail de SCARA est montré dans la figure 1.12.

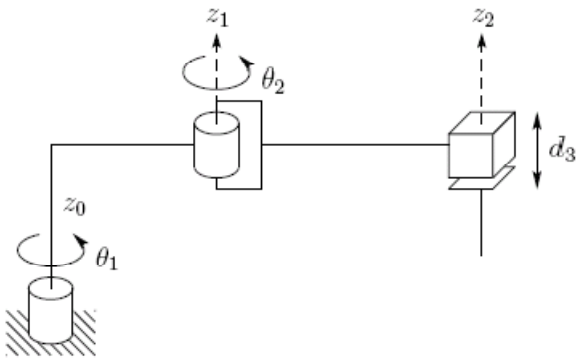


Figure 1.10: le robot SCARA
(SelectiveCompliantArticulated Robot for Assembly).

Figure 1.11 : Le manipulateur SCARA Epson E2L653S.

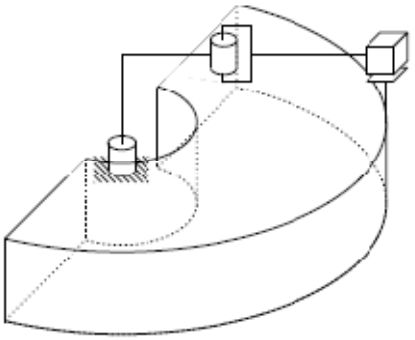


Figure 1.12 : Espace de travail du robot SCARA.

Configuration Cylindrique (RPP)

La configuration cylindrique est montrée dans la figure 1.13. La première articulation est rotoïde et produit une rotation autour de la base, alors que la seconde et la troisième sont prismatiques.Comme le nom le suggère,les variables articulaires sont les coordonnées cylindriques de l'organe terminalen prenant compte de la base.

Le Seiko RT3300, un robot A cylindrique, est montré dans la figure 1.14, avec son espace de travail dans la figure 1.15.

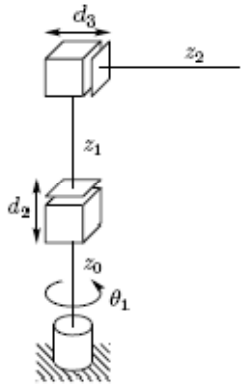


Figure 1.13: La configuration du manipulateur cylindrique.

Figure 1.14: LeRobot Seiko RT3300.

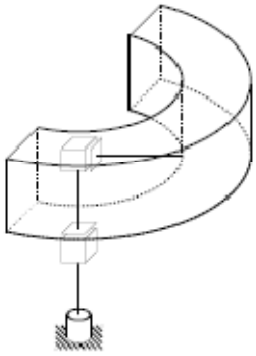


Figure 1.15: Espace de travaildumanipulateur cylindrique.

Configuration Cartésienne (PPP)

Un manipulateur dont les trois premières articulations sont prismatiques est connu sous le nom de manipulateur cartésien. Montré dans la figure 1.16.

Pour le manipulateur cartésien les variables articulaires sont les coordonnées cartésiennes de l'organe terminal en respectant la base.

Comme on peut le deviner la descriptioncinématique de ce manipulateur est la plus simple de toutes les configurations. Les manipulateurs cartésiens sont très utiles dans les applications d'assemblage sur table et, comme robot de chantier, pour le transfert ou le chargement de matériels.

Unexempledurobot cartésien, d'Epson-Seiko, est montré dans la figure 1.17. L'espace de travail est montré dans la figure 1.18.

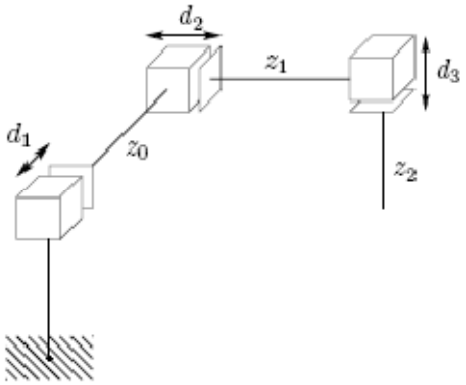


Figure 1.16: La configuration cartésienne.

Figure 1.17: Le Robot Cartésien d'Epson.

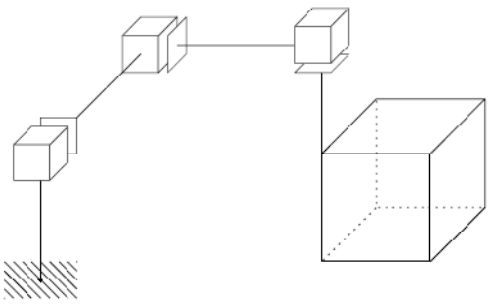


Figure 1.18: Espace de travail du manipulateur cartésien.

Le Manipulateur Parallèle

Un manipulateur parallèle est une configuration dans laquelle les segments forment une chaine fermée.

Plus spécifiquement, ce manipulateur contient deux chaines, ou plus, indépendantes connectant la base à l'organe terminal. La figure 1.19 montrele robot parallèle IRB 940 Tricept de ABB.

La cinématique des robots parallèles à chaîne fermée permet d'obtenir des structures avec une plus grande rigidité, et delà une précision plus grande que les robots à chaînes ouvertes.

La description cinématique des robots parallèles est fondamentalement différente de celle des robots à chaine ouverte et donc requière des méthodes différentes d'analyse.

Figure 1.19: Le Robot Parallèle ABB IRB940 Tricept.

1.3.5 Systèmes robotiques

La conception d'un robot manipulateur ne serait juste si on le voyait comme une simple série de liaisons mécaniques.Le bras mécanique est juste un composant de tout un « Système robotique », montré dans la figure 1.20, lequel se compose du bras, de la source de puissanceextérieure, de l'outillage de l'organe terminal, des capteurs externes et internes,de l'interface avec l'ordinateur, et de l'ordinateur decontrôle.Même le logiciel programmé doit être considéré comme une partie intégrante du système entier, du moment que la manière avec laquelle le robot est

programmé etcontrôlé peut avoir l'impact majeur sur sa performance ainsi que sur son champ d'application.

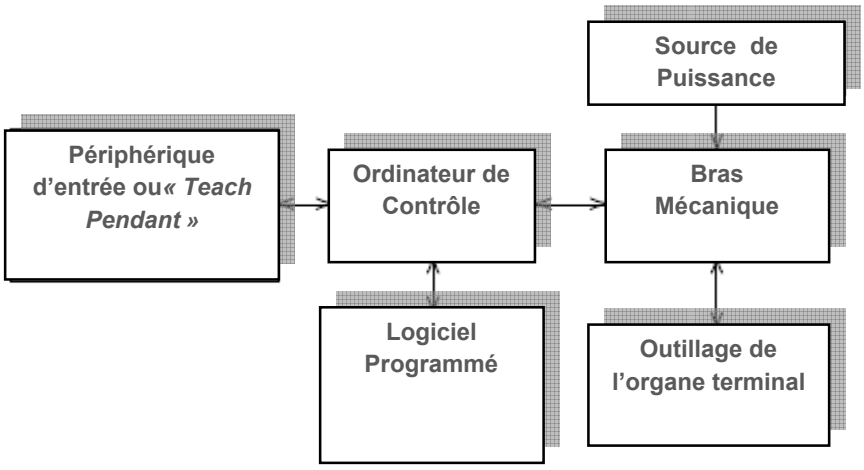


Figure 1.20: Composants du système robotique.

1.3.6 Précision et répétabilité

La précision d'un manipulateur est une mesure de l'erreur de distance à laquelle on peut amener le manipulateur à un point donné dans son espace de travail.

La répétabilité est une mesure de l'erreur maximale de positionnement répété de l'organe terminal en tout point de son espace de travail.La plupart des manipulateurs d'aujourd'hui sont d'une grande répétabilité mais pas très précis.

La méthode principale pour la capture des erreurs de position est d'utiliser, dans la plupart des cas, des encodeurs de position localisés dans les articulations, soit sur l'arbre du moteur qui actionne l'articulation ou carrément sur l'articulation elle-même.

Il n'existe typiquement pas de mesure directe de la position et l'orientation de l'organe terminal.Il faut compter sur la géométrie du manipulateur et sa rigidité pour déduire (i.e., calculer) la position de l'organe terminal à partir de la mesure des angles des différentes articulations. Par conséquent la précision est affectée par les erreurs de calcul, la précision de l'usinage lors de la construction du manipulateur, les effets de flexibilité comme le fléchissement des segments sous l'effet du poids, le jeu entre les engrenages. C'est principalement la raison pour laquelle les robots sont conçus avec une rigidité extrêmement grande sans laquelle la précision ne peut être améliorée que

par une sorte de capture directe de la position de l'organe terminal telle que la vision par exemple.

1.3.7 Poignet etorgane terminal

Le poignet d'un manipulateur se réfère à l'articulation dans la chaine cinématique entre le bras et la main. Les articulations du poignet sont presque toujours rotoïdes. De nos jours on conçoit de plus en plus des manipulateurs avec des poignets sphériques, c'est à dire, poignet dont les axes des trois articulations se coupent en un point commun. Le poignet sphérique est représenté symboliquement dans la figure 1.21.

Le poignet sphérique simplifie considérablement l'analyse cinématique en permettant le découplage de la position et de l'orientation.

Donc,lemanipulateurva posséder troisdegrés de liberté de position, lesquels sont produits par trois articulations ou plus dans le bras.

Le nombre de degrés de liberté d'orientation vont alors dépendre des degrés de liberté du poignet.

Il est commun de trouver des poignets avec un, deux, ou trois degrés de liberté selon le domaine d'application. Par exemple le robot SCARA montré dans la figure 1.11a quatre degrés de liberté : trois pour le bras, et un pour le poignet, qui a seulement un roulis autour de l'axe z final.

On dit qu'un robot est aussi bon que sa main ou organe terminal l'est. Le bras et le poignet ensemble servent essentiellement au positionnement de l'organe terminal ainsi que l'outil qu'il peut porter éventuellement. En fait c'est l'organe terminal ou l'outil qui effectue le travail. Le type le plus simple des organes terminaux sont les pinces comme ceux montrés dans la figure 1.22 qui ne sont généralement capables que de deux actions, s'ouvrir et se fermer. Alors que ceux-ci sont adaptées au transfert de matériaux, manipulation de pièces ou maintenir de simples outils, elles sont loin d'être adéquates pour d'autres tâches comme la soudure ou l'assemblage par exemple. Par conséquent bon nombre de recherches sont dédiées à la conception d'organes terminaux à usage spécifique pouvant être changés rapidement quand la tâche l'impose. Il y a également beaucoup de recherches dédiées au développement de

mains anthropomorphiques. De telles mains sont développées pour les prothèses ainsi que pour la production.

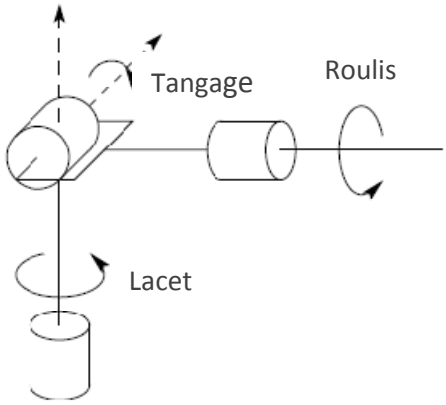


Figure 1.21: Structure d'un poignet sphérique.

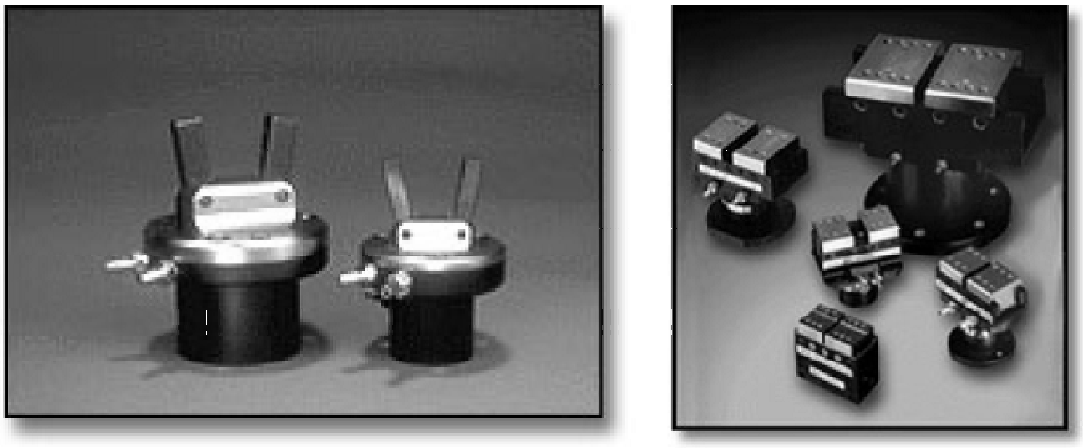


Figure 1.22: pinces mâchoires angulaires et Parallèles

1.4Les grandes lignes de la robotique [2]

Une application typique qui implique un manipulateur industriel est montrée dans la figure 1.23. Le manipulateur est montré avec un outil d'affûteur pour enlever un certain surplus de métal d'une surface.

Dans le présent contexte on peut se poser les questions suivantes :

Quels sont les problèmes fondamentaux à résoudre et qu'est ce qu'on doit connaitre pour être en mesure de programmer un robot pour accomplir des tâches comme celle de la figure 1.23 ?

Pour répondre à ces questions, et pour illustrer quelques problèmes majeurs, on utilise un simple mécanisme plan à deux segments.

Supposons qu'on veuille mouvoir le manipulateur de sa position initial vers une position *A*, à partir de laquelle le robot doit suivre le contour de la surface *S*, à vitesse constante, tout en maintenant une force prescrite *F* sur la surface. En faisant ainsi le robot va couper ou affûter la surface suivant les spécifications prédéterminées.

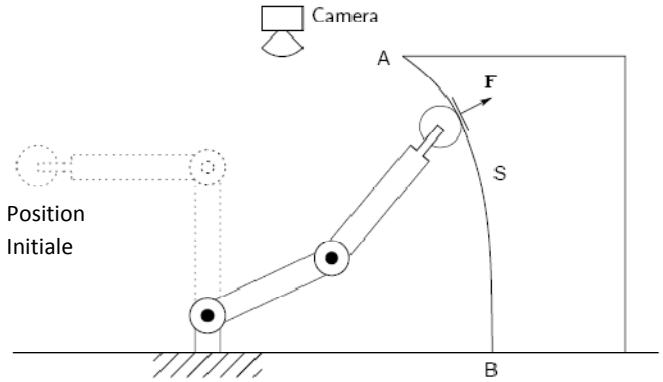


Figure 1.23: Exemple robot plan à deux segments.

Problème 1: Modèle Géométrique Direct

Le premier problème rencontré est de décrire et la position de l'outil et les emplacements A et B (et probablement toute la surface S) en respectant un système de coordonnées commun.

Typiquement, lemanipulateurva être capable de percevoir sa propre position en utilisant des capteurs internes (encodeurs de position) localisés dans les articulations 1 et 2, pouvant mesurer directement les angles θ1 et θ2 des articulations.

Par conséquent on aurait besoins aussi d'exprimer les positions A et B en fonction de ces angles.

Ceci aboutit au Modèle Géométrique Direct dont l'objectif est de « *déterminer la position et l'orientation de l'organe terminal ou l'outil en fonction des variables articulaires* ».

Habituellement on établit un système de coordonnées fixe, appelé repère de base dans lequel tous les objets incluant le manipulateur sont référencés. Dans ce cas précisément on établit le repère de base Ox_0y_0 à la base du robot, comme montré dans la figure 1.24.

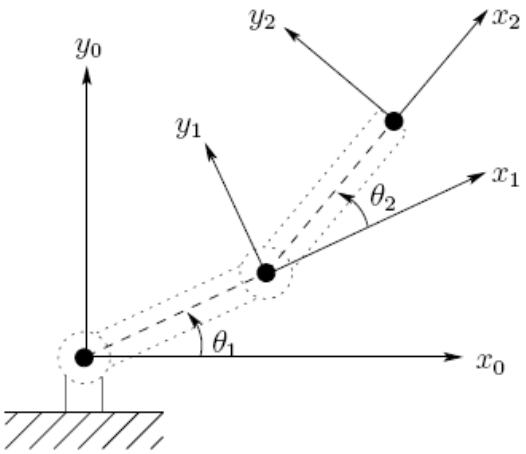


Figure 1.24: Repères pour un robot plan à deux segments.

Les coordonnées (x, y) de l'outil sont exprimées dans ce repère comme :

$$x = x_2 = \alpha_1 \cos\theta_1 + \alpha_2 \cos(\theta_1 + \theta_2) \quad (1.1)$$

$$y = y_2 = \alpha_1 \sin\theta_1 + \alpha_2 \sin(\theta_1 + \theta_2), \quad (1.2)$$

α1etα2 représentent respectivement les longueurs des deux segments.

Aussi l'orientation du repère de l'outil par rapport au repère de base est donnée par les cosinus des axes x2 et y2 relativement aux axes x0 et y0, C'est-à-dire :

$$\begin{aligned} x_2 \cdot x_0 &= \cos(\theta_1 + \theta_2); \quad x_2 \cdot y_0 = \sin(\theta_1 + \theta_2) \\ y_2 \cdot x_0 &= \sin(\theta_1 + \theta_2); \quad y_2 \cdot y_0 = \sin(\theta_1 + \theta_2) \end{aligned} \quad (1.3)$$

Chose qu'on peut écrire sous la forme d'une matrice orientation :

$$\begin{bmatrix} x_2 \cdot x_0 & y_2 \cdot x_0 \\ x_2 \cdot y_0 & y_2 \cdot y_0 \end{bmatrix} = \begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \end{bmatrix}. \quad (1.4)$$

Ces équations (1.1-1.4) sont appelées **les équations du modèle géométrique directe**.

Problème 2: Modèle Géométrique Inverse.

Maintenant, les angles θ1 et θ2 des articulations étant donnéeson peutdéterminerles coordonnées x et y de l'organe terminal.

Afin de commander le robot pour le déplacer à l'emplacement B on a besoin de l'inverse; C'est à dire, on a besoin des d'exprimer les variables articulaires θ1, θ2 en fonction des coordonnées x et y de B.

Ceproblèmeest appelé le "Modèle Géométrique Inverse". En d'autres termes,x et y étant donnés dans les équations (1.1-1.2) du modèle géométrique directe,on désirerésoudrele système d'équations pour trouver les angles des variables articulaires.

Etant donné que les équations du modèle géométrique directe sont non-linéaires, une solution n'est ni évidente ni unique en général. On peut voir, par exemple, dans le cas d'un mécanisme plan à deux segments qu'il n'existe pas de solution si les coordonnées (x, y) données sont hors de portée du manipulateur. Si les coordonnées (x, y) données sont à la portée du manipulateur on aura alors deux solutions comme montré dans la figure 1.25, les fameuses configurations "*coude en haut et coude en bas*", ou une solution unique si le manipulateur doit être étendu au maximum pour attendre un point donné. On peut même, dans certains cas, avoir une infinité de solutions !

Considérons lediagramme de la figure 1.26. En utilisantlaloides Cosinus on peut voir quel'angle θ2 est donné par :

$$\cos\theta_2 = \frac{x^2 + y^2 - \alpha_1^2 - \alpha_2^2}{2\alpha_1\alpha_2} := D. \qquad (1.5)$$

On peut maintenant déduire :

$$\theta_2 = \cos^{-1}(D). \qquad (1.6)$$

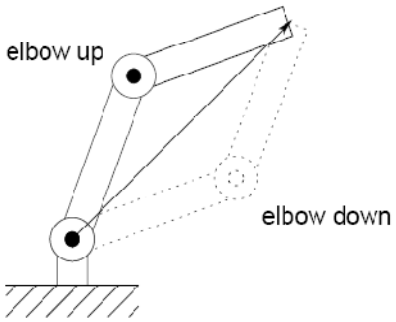


Figure 1.25: Solutions multiples pour le modèle géométrique inverse.

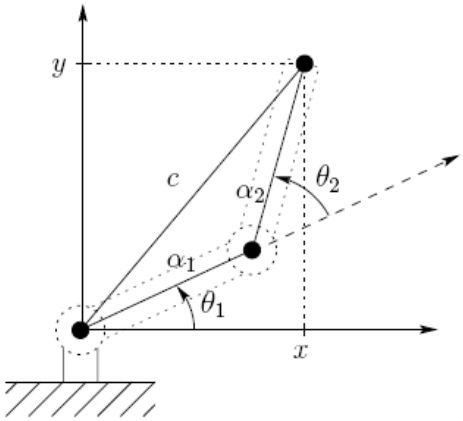


Figure 1.26: Résolution pour les angles articulairespour un bras plan à deux segments.

Toutefois, une bonne méthode pour trouver θ2 et de remarquer que si cos(θ2) est donné par (1.5)alorssin(θ2)sera donné par :

$$\sin(\theta_2) = \pm\sqrt{1 - D^2} \qquad (1.7)$$

Et, donc, θ2 peut être donné par :

$$\theta_2 = \tan^{-1}\frac{\pm\sqrt{1 - D^2}}{D}. \qquad (1.8)$$

L'avantage de cette dernière approche est qu'elle couvre les deux solutions coude en haut et coude en bas en choisissant le signes positif et négatif dans (1.8) respectivement.

$\theta_1$sera donné par :

$$\theta_1 = \tan^{-1}(y/x) - \tan^{-1}\left(\frac{\alpha_2 \sin\theta_2}{\alpha_1 + \alpha_2 \cos\theta_2}\right). \quad (1.9)$$

On remarque que l'angleθ1dépenddeθ2. Ceci est compris physiquement vu qu'on doit donner une valeur différente à θ1 en fonction de la solution choisie pour l'angleθ_2.

Problème 3: Modèle Cinématique Inverse

Afin de suivre le contour à une vitesse constante, ou toute vitesse prescrite, on doit connaitre la relation qui existe entre la vitesse de l'outil est les vitesses des différentes articulations.

Dans ce cas on peut différencier les équations (1.1) et (1.2) pour obtenir :

$$\begin{aligned} \dot{x} &= -\alpha_1 \sin\theta_1 \cdot \dot{\theta}_1 - \alpha_2 \sin(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2) \\ \dot{y} &= \alpha_1 \cos\theta_1 \cdot \dot{\theta}_1 + \alpha_2 \cos(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2). \end{aligned} \quad (1.10)$$

En utilisant la notation vectorielle $x=$ et $\theta=$ $\begin{matrix}1\\2\end{matrix}$ il est possible d'écrire ces équations comme :

$$\begin{aligned} \dot{x} &= \begin{bmatrix} -\alpha_1 \sin\theta_1 - \alpha_2 \sin(\theta_1 + \theta_2) & -\alpha_2 \sin(\theta_1 + \theta_2) \\ \alpha_1 \cos\theta_1 + \alpha_2 \cos(\theta_1 - \theta_2) & \alpha_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \dot{\theta} \\ &= J\dot{\theta}. \end{aligned} \quad (1.11)$$

La matrice J définie dans (1.11) est appelé " *le Jacobien du manipulateur* " dont la détermination est fondamentale pour le manipulateur.

La détermination des vitesses articulaires à partir de la vitesse de l'organe terminal est conceptuellement simple étant donné que la relation de vitesse est linéaire! Ainsi les vitesses articulaires sont trouvées via le *JacobienInverse*.

$$\dot{\theta} = J^{-1}\dot{x} \quad (1.12)$$

Où est donné par :

$$J^{-1} = \frac{1}{\alpha_1 \alpha_2 s\theta_2} \begin{bmatrix} c_2 c_{\theta_1+\theta_2} & c_2 s_{\theta_1+\theta_2} \\ -\alpha_1 c_{\theta_1} - \alpha_2 c_{\theta_1+\theta_2} & -\alpha_1 s_{\theta_1} - \alpha_2 s_{\theta_1+\theta_2} \end{bmatrix} \quad (1.13)$$

Cθ et *Sθ* désignent respectivement *cos θ*et*sin θ*. ledéterminant, det J, duJacobien dans (1.11) est : $\alpha_1\alpha_2$ sin θ_2. Le Jacobienn'a pas d'inverse lorsque $\theta_2 = 0$ ouπ, le cas dans lequel le manipulateur est dit « dans une configuration singulière » comme montrée dans la figure1.27pour$\theta_2 = 0$. Ladéterminationde telles of configurations singulières est importantepour plusieurs raisons. Dans les configurations singulières il y a une infinité de mouvements qui sont irréalisables ; C'est-à-dire que l'organe terminal du manipulateur ne peut se mouvoir dans certaines directions.

Dans le cas ci-dessus l'organe terminal ne peut pas bouger dans la direction parallèle à x2, partant d'une configuration singulière.

Les configurations singulières sont aussi liées à la non- unicité des solutions du modèleinverse. Par exemple, pour une position donnée de l'organe terminal, il y a en général deux solutions possibles pour le modèle inverse.

Notons que la configuration singulière sépare ces deux solutions au sens que le manipulateur ne peut passer d'une configuration à l'autre sans passer par la singularité. Pour beaucoup d'applications il est primordial de planifier les mouvements du manipulateur de telle sorte à éviter les configurations singulières.

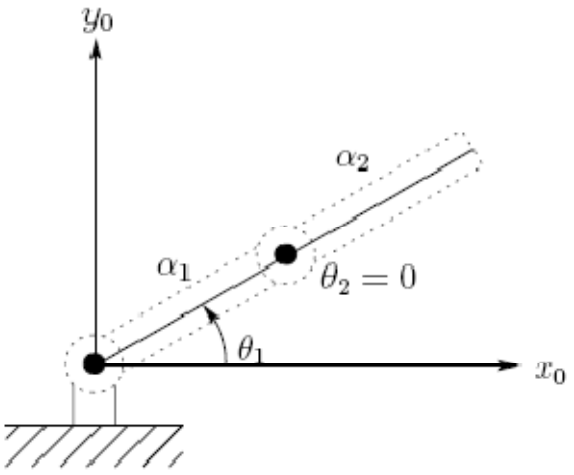


Figure 1.27: Une configuration singulière.

Problème 4: planification de chemin et Génération deTrajectoire

Le problème de la commande du robot est décomposéhiérarchiquementen trois tâches : planification du chemin, génération de la trajectoire, et poursuite de latrajectoire.

Le problème de la planification du chemin est de déterminer dans l'espace des configurations un chemin pour mouvoir le robot vers une position destinée tout en évitant la collision avec des objets dans son espace de travail. Dans ces chemins on ne considère que les informations concernant la position sans considération temporelles, i.e. *sans considérations des vitesses ou des accélérations*le long du chemin planifié.

Le problème de génération de la trajectoire, consiste à générer des trajectoires "références" qui déterminent l'historique temporel du manipulateur le long d'un chemin donné ou entre une configuration initiale et finale.

Problème 5: La Vision

De nos jours, les camérassont devenues des capteurs relativement "bon marché" et fiables utiliséesdans beaucoup d'applications robotiques. A la différence des capteurs articulaires, qui donnent une information sur la configuration interne du robot, les caméras peuvent être utilisées pas seulement pour la mesure de la position du robot mais aussi pour localiser des objets externes au robot dans son espace de travail.

Problème 6: Le Modèle Dynamique.

Un robot manipulateur est essentiellement un dispositif de positionnement. Pour contrôler la position on doit connaitre les propriétés dynamiques du manipulateur afin de dimensionner la force qui doit être exercée pour provoquer son mouvement : une force insuffisante et le manipulateur est trop lent à réagir ; une force trop grande peut faire entrer en collision le robot avec des objets externes ou le faire osciller autour de la position désirée.

Dériver les équations dynamiques du mouvement pour les robots n'est pas une tâche facile à cause du grand nombre de degrés de liberté et des non-linéarités présentes dans le système.

Le modèle dynamique du robot inclue la dynamique des actionneurs qui produisent les forces et les couples pour remuer le robot et la dynamique de la chaine qui transmet la puissance des actionneurs vers les segments.

Problème 7: La commande de Position.

Le problème de commande du mouvement consiste en deux choses : la poursuite et le rejet de perturbation, c'est à dire déterminer les entrées de commande nécessaires pour suivre, ou traquer, une trajectoire désirée planifiée d'avance pour le manipulateur, tout en rejetant simultanément les perturbations dues aux effets non modélisables comme le frottement ou le bruit.

Problème 8: Contrôle de la Force.

Une fois que le manipulateur atteigne l'emplacement A il doit suivre le contour S en maintenant une force constante normale à la surface. Pour se faire, connaissant l'emplacement des objets el la forme du contour, on peut exécuter cette tâcheutilisant seulement la commande de position. Pourtant ceci va être assez difficile à accomplir en pratique,

Vu que le manipulateur possède lui-même une grande rigidité, des erreurs quelconques de positions dues à l'incertitude dans l'emplacement exacte de la surface ou de l'outil vont engendrer des forces extrêmement grandes à la fin de l'organe terminal pouvant endommager l'outil, la surface ou le robot. Une approche meilleure et de mesurer directement les forces d'interaction et d'utiliser un « systèmede contrôle de la force » pour accomplir la tâche.

Chapitre 2

LE MANIPULATEUR PUMA 560

2.1 Introduction

Le Puma est probablement le robot le plus répandu dans les laboratoires universitaires et l'un des robots d'assemblage les plus courants figures 2.1 – 2.2. Conçu par Vic Schienman et financé par GM dans le MIT au milieu des années 70, le Puma (Programmable Universal Machine for Assembly) a été produit pendant de nombreuses années par Unimation (rachetée plus tard par Westinghouse et vendu à perte plus tard pour Staubli, une société suisse). Ces robots et leurs successeurs sont trouvés principalement dans les laboratoires universitaires [3].

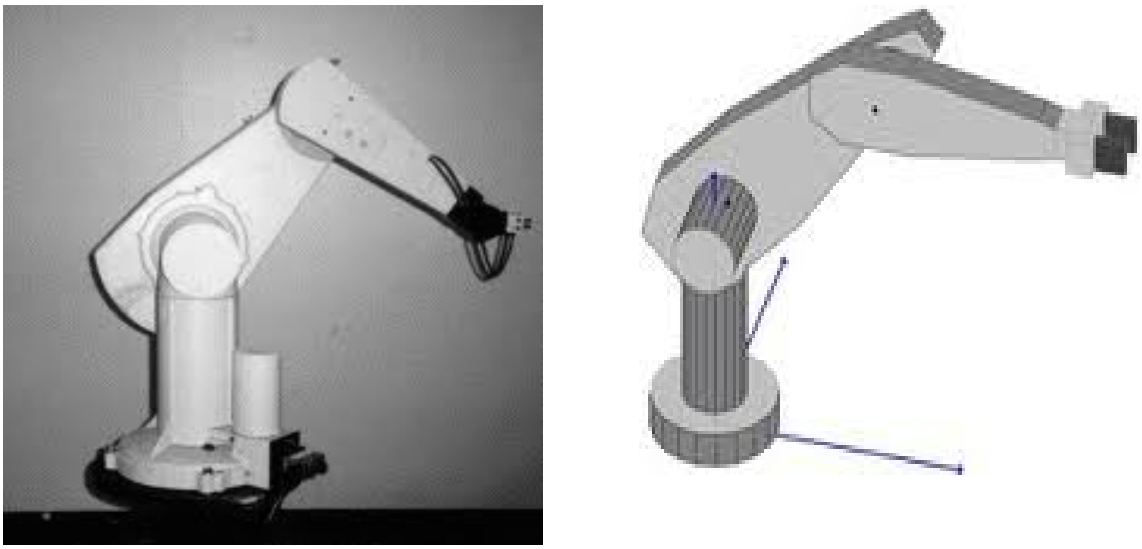


Figure 2.1 le bras manipulateur Puma 560.

Le Puma 560 est un robotmanipulateur à six degrés de liberté. L'effecteur du robot peut atteindre un point donnéà l'intérieur de son espace de travail par n'importe quelle direction. Les six degrés de liberté sont contrôlées par six servomoteurs DC, chacun étant couplé avec un encodeur 500-1000 et un potentiomètre. Le robot peut déterminer sa position globale par l'information retournée par rétroaction. Le contrôleur d'origine du 560 Puma a été reconçu. Le robot peut actuellement être contrôlé à distance via le réseau.

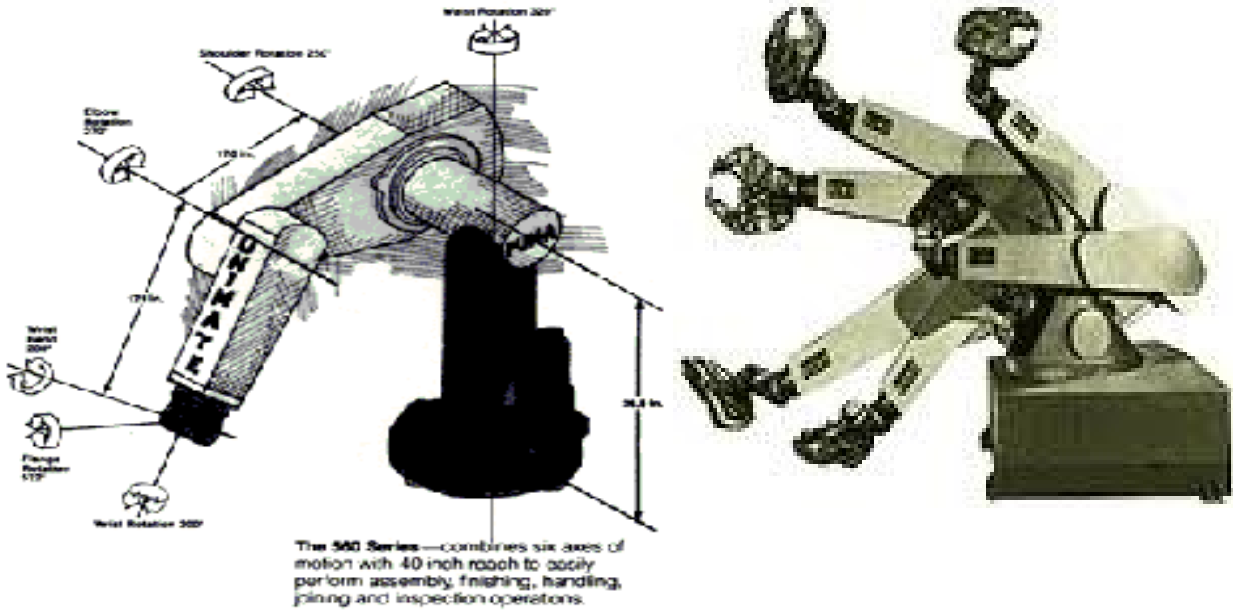


Figure 2.2Illustration du bras manipulateur Puma 560.

2.2 Présentation du PUMA 560

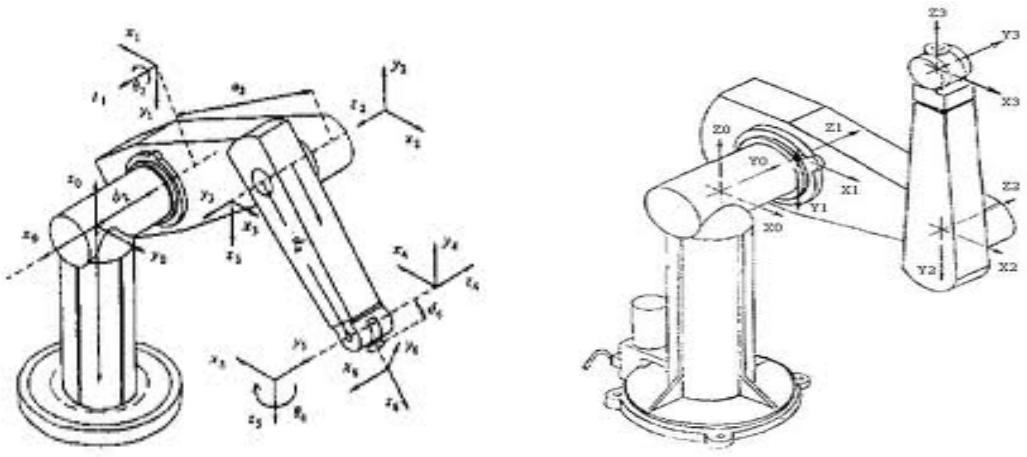


Figure 2.3Choix des repères du bras manipulateur Puma 560.

2.2.1 Les calculs mathématiques

Les calculs mathématiques ont été obtenus principalement à partir d'une seule source, "Robot Dynamics and Control" par Spong [2], mais ont été largement comparés à diverses sources sur Internet.

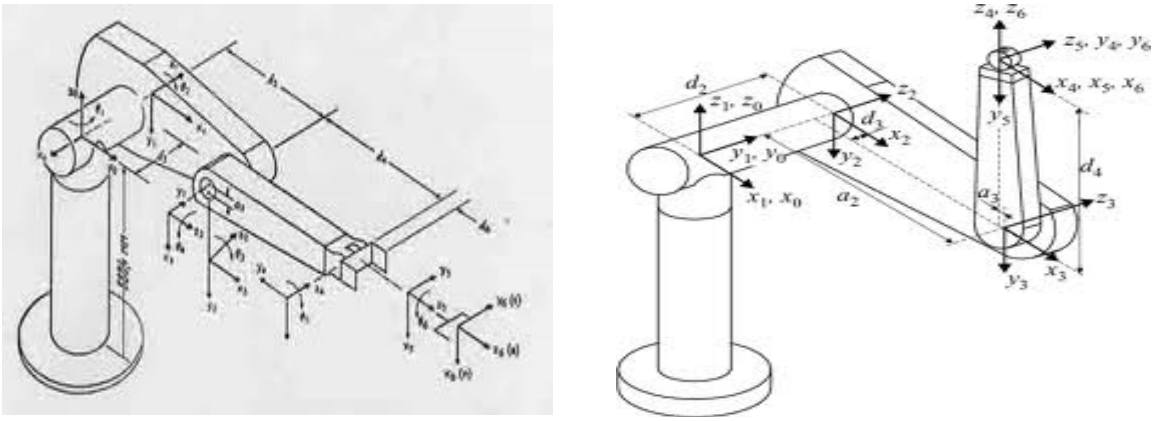


Figure 2.4Repères et Distances du bras manipulateur Puma 560.

Le modèle géométrique direct

Le modèle géométrique direct du Puma 560 n'est qu'un cas trivial de la construction des paramètres Denavit-Hartenberg (DH) de la matrice de transformation.
Avec des paramètres obtenus à partir d'un tableau des paramètres DH (Figure. 2.5)

Link	a (m)	α(deg)	d (m)	θ(deg)
1	0	90	0.67	*
2	0.4318	0	0	*
3	0.4318	-90	0.15005	*
4	0	90	0	*
5	0	-90	0	*
6	0	0	0	*

Figure2.5paramètres D-H pour Puma 560.

Matrices de passage

La matrice finale est obtenue par multiplication successives des matrices de transformation et/ou de rotation définies par l'architecture du manipulateur (Figure 2.6)

$$T_1=\begin{bmatrix}\cos(\theta_1) & 0 & \sin(\theta_1) & 0\\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 0\\ 0 & 1 & 0 & 0.67\\ 0 & 0 & 0 & 1\end{bmatrix}\quad T_2=\begin{bmatrix}\cos(\theta_2) & -\sin(\theta_2) & 0 & 0.4318\cdot\cos(\theta_2)\\ \sin(\theta_2) & \cos(\theta_2) & 0 & 0.4318\cdot\sin(\theta_2)\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\quad T_3=\begin{bmatrix}\cos(\theta_3) & 0 & -\sin(\theta_3) & 0.4318\cdot\cos(\theta_3)\\ \sin(\theta_3) & 0 & \cos(\theta_3) & 0.4318\cdot\sin(\theta_3)\\ 0 & -1 & 0 & 0.15005\\ 0 & 0 & 0 & 1\end{bmatrix}$$

$$T_4=\begin{bmatrix}\cos(\theta_4) & 0 & \sin(\theta_4) & 0\\ \sin(\theta_4) & 0 & -\cos(\theta_4) & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\quad T_5=\begin{bmatrix}\cos(\theta_5) & 0 & -\sin(\theta_5) & 0\\ \sin(\theta_5) & 0 & \cos(\theta_5) & 0\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\quad T_6=\begin{bmatrix}\cos(\theta_6) & -\sin(\theta_6) & 0 & 0\\ \sin(\theta_6) & \cos(\theta_6) & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

$$T=T_1\cdot T_2\cdot T_3\cdot T_4\cdot T_5\cdot T_6$$

Figure 2.6les sixmatrices de transformation du Puma 560.

Le résultat de la multiplication des matrices de transformations nous donne la position du repère 6 (organe terminal) par rapport à la basedans l'espace tridimensionnel (Figure2.7)

$$p_x=\cos(\theta_1)\cdot\left[a_2\cdot\cos(\theta_2)-a_3\cdot\left(\cos(\theta_2)\cdot\sin(\theta_3)+\sin(\theta_2)\cdot\cos(\theta_3)\right)\right]-d_3\cdot\sin(\theta_1)$$

$$p_y=\sin(\theta_1)\cdot\left[a_2\cdot\cos(\theta_2)-a_3\cdot\left(\cos(\theta_2)\cdot\sin(\theta_3)+\sin(\theta_2)\cdot\cos(\theta_3)\right)\right]+d_3\cdot\cos(\theta_1)$$

$$p_z=-a_2\cdot\sin(\theta_2)-a_3\cdot\left(\cos(\theta_2)\cdot\cos(\theta_3)-\sin(\theta_2)\cdot\sin(\theta_3)\right)+d_1$$

Figure2.7 Formule depositiondu repère 6 par rapport à la base.

Le modèle géométrique Inverse

Composé de trois segments et un poignet sphérique, architecturalement parlant, il était évident que le découplage cinématique était la méthode à suivre pour trouver le modèle géométrique inverse (IK).

Le découplage cinématique consiste à séparer les calculs de la position des calculs de l'orientation. Ceci peut être réalisé si l'on considère les trois liens du Puma séparément du poignet sphérique. Compte tenu de l'architecture du Puma (Figure 2.8), on peut voir que le centre de rotation du poignet, coïncide avec le repère du troisième segment (Figure 2.9).

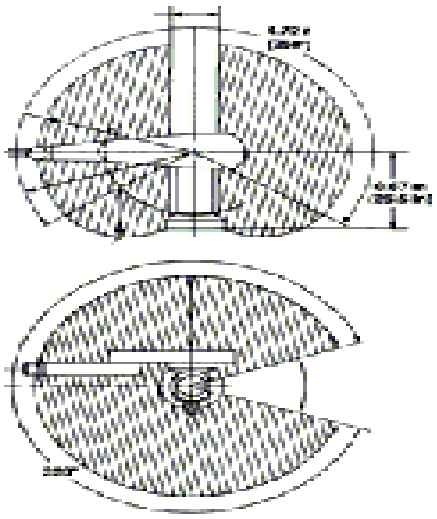


Figure 2.8Espace de travail du Puma 560

En considérant le cas d'un modèle Puma 560 théorique où nous n'avons pas d'outil monté sur le poignet, on aura pas besoin d'inclure le calcul de la position du poignet sphérique dans les formules finales du modèle inverse.

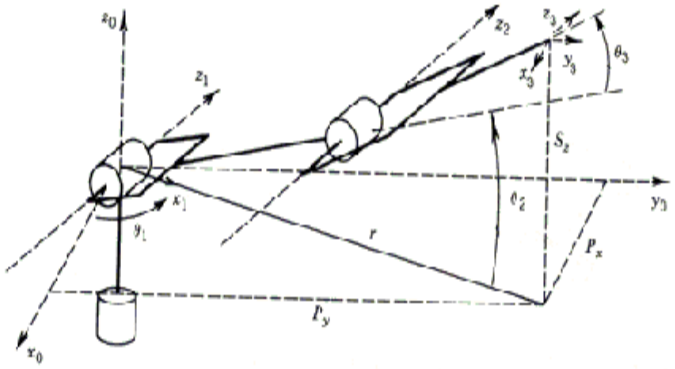


Figure 2.9Schéma simplifié des trois premiers segments du Puma 560
avec repères appropriés

Dans le cas des manipulateurs Puma, où il y a un offset $d_1 \neq 0$ entre le segment un et deux (Figure 2.10), la possibilité d'une configuration singulière n'existe pas. Ça aurait été le cas si et seulement si le centre du poignet coupe l'axeZ_0.

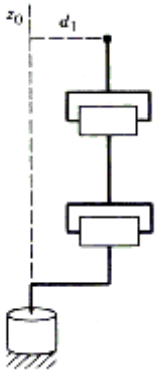


Figure 2.10 manipulateur avec offset.

De cette façon, il n'y aura que deux solutions pourθ_1, correspondant aux configurations bras gauche et bras droit comme le montre la Figure 2.11.

Quand on regarde le robot de ce qui précède et on prétend que les segments deux et trois sont comme s'ils étaient un seul, il est facile d'observer que nous pouvons obtenir la grandeur de θ_1 via simple trigonométrie:

$$\theta_1 = \phi - \alpha$$

$$\phi = \mathrm{Atan}\left(\frac{p_y}{p_x}\right)$$

$$\alpha = \mathrm{Atan}\left(\frac{-\sqrt{r^2 - d_1^2}}{d_1}\right) = \mathrm{Atan}\left(\frac{\sqrt{p_x^2 + p_y^2 - d_1^2}}{d_1}\right)$$

$$\theta_1 = \mathrm{Atan}\left(\frac{p_y}{p_x}\right) - \mathrm{Atan}\left(\frac{\sqrt{p_x^2 + p_y^2 - d_1^2}}{d_1}\right)$$

Equation 2.1

Notons que $d_1$dans ces équations et les Figures 2.10 et 2.11diffère ded_1 dans la table des paramètres DH. Le premier est en fait un déplacement entrelessegments deux et trois combinés et l'axe $\mathbf{Z_0}$ de rotation du premier segment et peut être trouvé dans la table DH sous d_3, plus loin dans ce texte noté L_1.

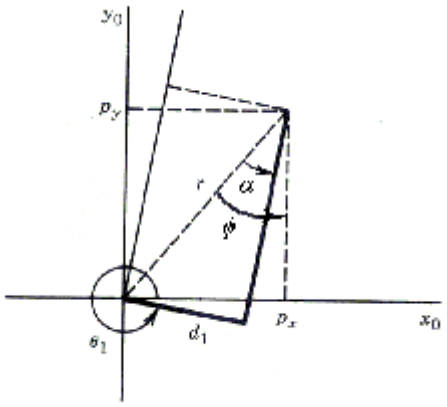


Figure 2.11Configuration bras droit.

En outre, pour trouver les angles$\square_1$ et $\square_2$ pour le manipulateur,$\square_1$ étant donné, nous considérons le plan formé par ledeuxième segment et le troisième comme le montre la Figure 2.12.

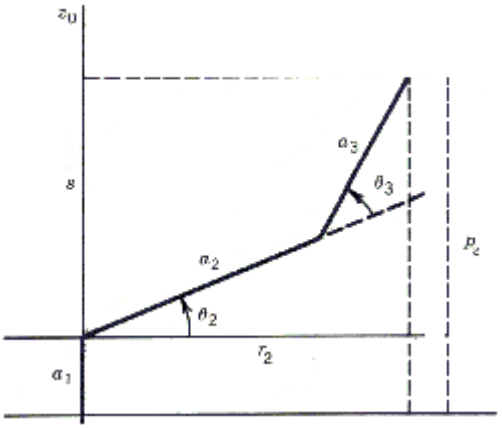


Figure 2.12Projection sur le plan formé par segments 2 et 3

Puisque le mouvement des segments deux et trois est plan, nous pouvons obtenir la valeur de $\theta_3$par application de la loi du cosinus:

$$\mathrm{Cos}(\theta_3)=\frac{r_2^2+s^2-a_2^2-a_3^2}{2\cdot a_2\cdot a_3}$$

$$\mathrm{Cos}(\theta_3)=\frac{p_x^2+p_y^2-d_1^2+p_z^2-a_2^2-a_3^2}{2\cdot a_2}=D$$

et donc $\theta_3$est donné par:

$$\theta_3=\mathrm{Atan}\left(\frac{\sqrt{a_3^2-D^2}}{D}\right)$$

$$\theta_3=\mathrm{Atan}\left[\frac{\sqrt{a_3^2-\left(\frac{p_x^2+p_y^2-d_1^2+p_z^2-a_2^2-a_3^2}{2\cdot a_2}\right)^2}}{\frac{p_x^2+p_y^2-d_1^2+p_z^2-a_2^2-a_3^2}{2\cdot a_2}}\right]$$

Equation 2.2

De la même façon θ_2 est donné par:

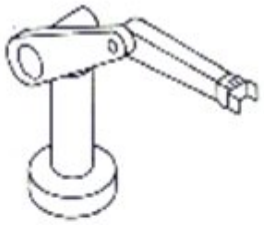
$$\theta_{23}=\text{Atan}\left[\frac{p_z\cdot\left(-a_2\cdot\text{Cos}\left(\theta_3\right)\right)-\left(\text{Cos}\left(\theta_1\right)\cdot p_x+\text{Sin}\left(\theta_1\right)\cdot p_y\right)\cdot\left(a_3-a_2\cdot\text{Sin}\left(\theta_3\right)\right)}{p_z\cdot\left(a_2\cdot\text{Sin}\left(\theta_3\right)-a_3\right)-a_2\cdot\text{Cos}\left(\theta_3\right)\cdot\left(\text{Cos}\left(\theta_1\right)\cdot p_x+\text{Sin}\left(\theta_1\right)\cdot p_y\right)}\right]$$

$$\theta_2=\theta_{23}-\theta_3$$

$$\theta_2=\text{Atan}\left[\frac{p_z\cdot\left(-a_2\cdot\text{Cos}\left(\theta_3\right)\right)-\left(\text{Cos}\left(\theta_1\right)\cdot p_x+\text{Sin}\left(\theta_1\right)\cdot p_y\right)\cdot\left(a_3-a_2\cdot\text{Sin}\left(\theta_3\right)\right)}{p_z\cdot\left(a_2\cdot\text{Sin}\left(\theta_3\right)-a_3\right)-a_2\cdot\text{Cos}\left(\theta_3\right)\cdot\left(\text{Cos}\left(\theta_1\right)\cdot p_x+\text{Sin}\left(\theta_1\right)\cdot p_y\right)}\right]-\theta_3$$

Equation 2.3

Notons que les racines carrées à partir des équations de $\theta_1$et $\theta_3$ont deux solutions possibles qui correspondent aux positions «coude haut » et « coude bas », respectivement (Figure 2.10). Cela nous donne deux solutions possibles pourθ_1 et $\theta_3$ce qui donne un total de quatre solutions possibles pourθ_2. Par conséquent, nous avons un total de quatre combinaisons possibles pour le modèle géométrique inverse pour une position donnée du poignet sphérique.



Coude Haut

Coude Bas

Figure2.13Configuration bras droit Puma 560

L'orientation du poignet sphérique peut être réalisée comme une application des angles de rotation (Figure 2.14) à partir du programme. Aucun calcul n'est nécessaire dans les calculs Le modèle géométrique direct.

.

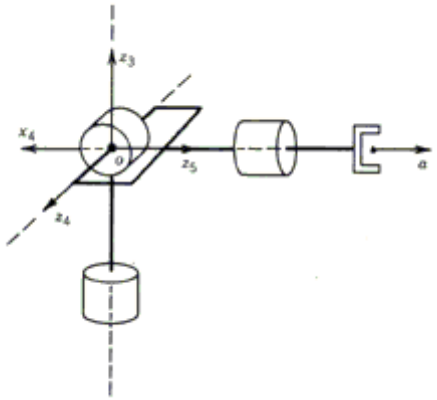


Figure 2.14poignet sphérique

Chapitre 3

LES RESEAUX DE NEURONES ARTIFICIELS

3.1 Introduction

Un réseau de neurones artificiels est un modèle de calcul dont la conception est très schématiquement inspirée du fonctionnement des neuronesbiologiques.

Les réseaux de neurones sont généralement optimisés par des méthodes d'apprentissage de type probabiliste, en particulier bayésiens. Ils sont placés d'une part dans la famille des applications statistiques, qu'ils enrichissent avec un ensemble de paradigmes permettant de générer des classifications rapides (réseaux de Kohonen en particulier), et d'autre part dans la famille des méthodes de l'intelligence artificielle auxquelles ils fournissent un mécanisme perceptif indépendant des idées propres de l'implémenteur, et fournissant des informations d'entrée au raisonnement logique formel.

En modélisation des circuits biologiques, ils permettent de tester quelques hypothèses fonctionnelles issues de la neurophysiologie, ou encore les conséquences de ces hypothèses pour les comparer au réel. [4]

3.2 Modèle d'un neurone

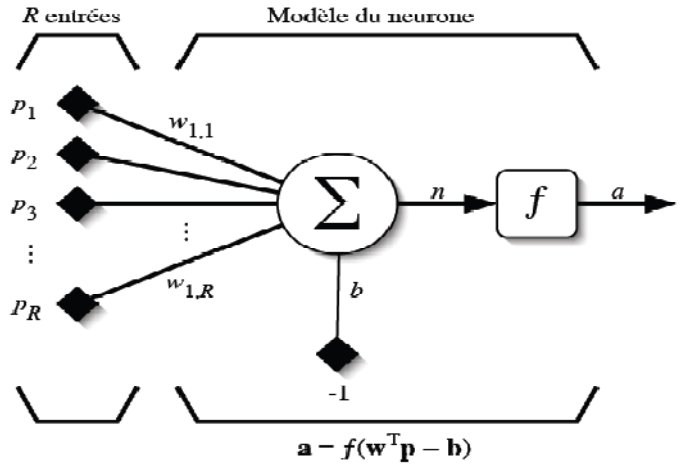


Figure 3.1: Modèle d'un neurone artificiel

Le modèle mathématique d'un neurone artificiel est illustré à la figure 3.1. Un neurone est essentiellement constitué d'un intégrateur qui effectue la somme pondérée de ses entrées. Le

résultat***n*** de cette somme est ensuite transformée par une fonction de transfert***f*** qui produit la sortie ***a*** duneurone. [5]

En suivant les notations présentées à la section précédente, les ***R*** entrées du neurones correspondentau vecteur $\boldsymbol{P} = [p_1 p_2 \cdots p_R]^T$, alors que $\boldsymbol{W} = [w_{1,1} w_{1,2} \cdots w_{1,R}]^T$ représente le vecteurdes poids du neurone. La sortie n de l'intégrateur est donnée par l'équation suivante :

$$\begin{aligned} n &= \sum_{j-1}^{R} w_{1,j}\, p_j - b \\ &= w_{1,1}\, p_1 + w_{1,2}\, p_2 + \cdots + w_{1,R}\, p_R - b, \end{aligned}$$

Que l'on peut aussi écrire sous forme matricielle :

$$n = \mathbf{w}^T \mathbf{p} - b.$$

Cette sortie correspond à une somme pondérée des poids et des entrées moins ce qu'on nomme lebiais ***b*** du neurone. Le résultat ***n*** de la somme pondérée s'appelle « le niveau d'activation du neurone ».Le biais ***b*** s'appelle aussi « le seuil d'activation du neurone ». Lorsque le niveau d'activation atteint oudépasse le seuil ***b***, alors l'argument de ***f*** devient positif (ou nul). Sinon, il est négatif. Ajoutons la fonction d'activation f pour obtenir la sortie du neurone :

$$a = f(n) = f(\mathbf{w}^T \mathbf{p} - b).$$

En remplaçant $\boldsymbol{W}^T$par une matrice $\boldsymbol{W} = \boldsymbol{W}^T$d'une seule ligne, on obtient une forme générale :

$$a = f(\mathbf{W}\mathbf{p} - b).$$

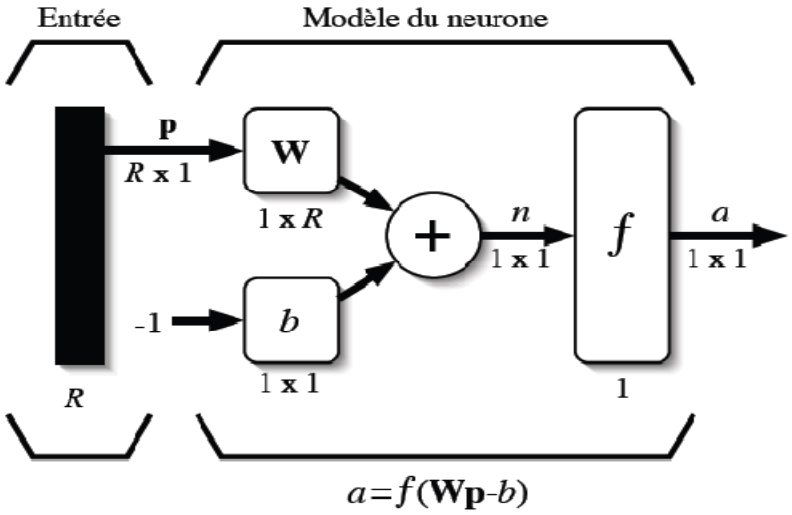


Figure 3.2: Représentation matricielle du modèle d'un neurone artificiel.

3.3 Fonctions de transfert

Différentes fonctions de transfert peuventêtre utilisées comme fonction d'activation du neurone tableau 3.1. Les trois les plusutilisées sont les fonctions «seuil» (en anglais «hard limit»), «linéaire» et «sigmoïde».

TAB. 3.1 – Fonctions de transfert ***a = f(n).***

Nom de la fonction	Relation d'entrée/sortie	Icône	Nom Matlab
seuil	$a = 0 \quad \text{si } n < 0$ $a = 1 \quad \text{si } n \geq 0$		hardlim
seuil symétrique	$a = -1 \quad \text{si } n < 0$ $a = 1 \quad \text{si } n \geq 0$		hardlims
linéaire	$a = n$		purelin
linéaire saturée	$a = 0 \quad \text{si } n < 0$ $a = n \quad \text{si } 0 \leq n \leq 1$ $a = 1 \quad \text{si } n > 1$		satlin
linéaire saturée symétrique	$a = -1 \quad \text{si } n < -1$ $a = n \quad \text{si } -1 \leq n \leq 1$ $a = 1 \quad \text{si } n > 1$		satlins
linéaire positive	$a = 0 \quad \text{si } n < 0$ $a = n \quad \text{si } n \geq 0$		poslin
sigmoïde	$a = \frac{1}{1+\exp^{-n}}$		logsig
tangente hyperbolique	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$		tansig
compétitive	$a = 1 \quad \text{si } n \text{ maximum}$ $a = 0 \quad \text{autrement}$		compet

Comme son nom l'indique, la fonction seuil applique un seuil sur son entrée. Plus précisément,une entrée négative ne passe pas le seuil, la fonction retourne alors la valeur 0 (on peut interpréterce 0 comme signifiant faux), alors qu'une entrée positive ou nulle dépasse le seuil, et la fonction retourne 1 (vrai). Utilisée dans le contexte d'un neurone, cette fonction est illustrée à la figure 3.3a. On remarque alors que le biais ***b***dans l'expression de :

a=hardlim(w^{T}p - b)détermine l'emplacement du seuil sur l'axe wTp, où la fonction passe de 0 à 1. Cette fonction permet de prendre des décisions binaires.

La fonction linéaire est très simple, elle affecte directement son entrée à sa sortie :***a = n***.

Appliquée dans le contexte d'un neurone, cette fonction est illustrée à la figure 3.3b. Dans ce cas,la sortie du neurone correspond à son niveau d'activation dont le passage à zéro se produit lorsque$\boldsymbol{w^T p=b.}$

La fonction de transfert sigmoïde est quant à elle illustrée à la figure 3.3c. Son équation estdonnée par :

$$a = \frac{1}{1+\exp^{-n}}.$$

Elle ressemble soit à la fonction seuil, soit à la fonction linéaire, selon que l'on est loin ou près de b,respectivement. La fonction seuil est très non-linéaire car il y a une discontinuité lorsque $\boldsymbol{w^T p = b.}$De son côté, la fonction linéaire est tout à fait linéaire. Elle ne comporte aucun changement de pente. La sigmoïde est un compromis intéressant entre les deux précédentes.

Notons finalement, que la fonction «tangente hyperbolique» est une version symétrique de la sigmoïde.

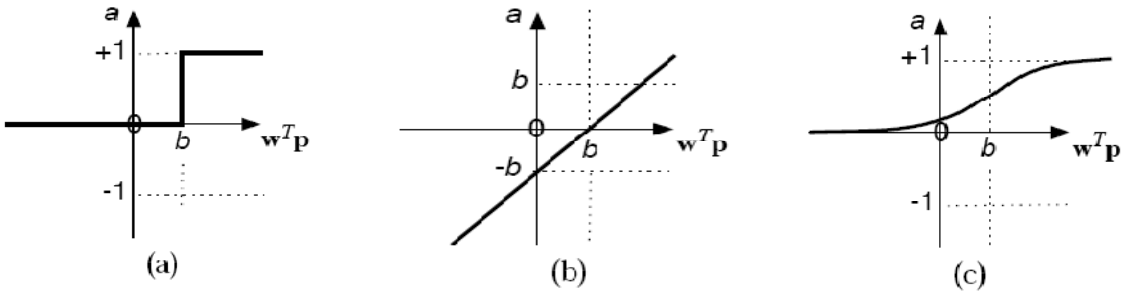


Figure3.3 – Fonction de transfert : (a) du neurone «seuil», (b) du neurone «linéaire»et (c) du neurone «sigmoïde».

3.4 Architecture de réseau

Un réseau de neurones est un maillage de plusieurs neurones, généralement organisée en couches. Pour construire une couche de ***Sneurones***, il s'agit simplement de les assembler comme à la figure3.4. Les ***S*** neurones d'une même couche sont tous branchées aux ***R entrées***. On dit alors que la couche est totalement connectée. Un poids $\boldsymbol{w_{i,j}}$est associé à chacune des

connexions. Nous noterons toujours le premier indice par ***i*** et le deuxième par ***j*** (jamais l'inverse). Le premier indice (rangée) désigne toujours le numéro de neurone sur la couche, alors que le deuxième indice (colonne) spécifie le numéro de l'entrée. Ainsi, $w_{i,j}$désigne le poids de la connexion qui relie le neurone ***i*** à son entrée ***j***. L'ensemble des poids d'une couche forme donc une matrice ***W*** de dimension $S \times R$:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ \vdots & \vdots & \ddots & \vdots \\ w_{S,1} & w_{S,2} & \cdots & w_{S,R} \end{bmatrix}$$

Notez bien que $S \neq R$, dans le cas général (les nombres de neurones et d'entrées sont indépendants).Si l'on considère que les ***S*** neurones forment un vecteur de neurones, alors on peut créer lesvecteurs $b=[b_1 b_2 \cdots b_S]^T$, $n = [n_1 n_2 \cdots n_S]^T$ et $a = [a_1 a_2 \cdots a_S]^T$. Ceci nous amène à lareprésentation graphique simplifiée, illustrée à la figure 3.5. On y retrouve, comme à la figure3.2, les mêmes vecteurs et matrice. La seule différence se situe au niveau de la taille, ou plusprécisément du nombre de rangées (***S***), de ***b***, ***n***, ***a*** et***W***.

Finalement, pour construire un réseau, il ne suffit plus que de combiner des couches comme à la figure 3.6. Cet exemple comporte ***R*** entrées et trois couches de neurones comptant respectivementS^1, S^2 et S^3 neurones. Dans le cas général, de nouveau, $S^1 \neq S^2 \neq S^3$. Chaque couchepossède sa propre matrice de poidsW^k, où***k*** désigne l'indice de couche. Dans le contexte des vecteurset des matrices relatives à une couche, nous emploierons toujours un exposant pour désignercet indice. Ainsi, les vecteurs b^k, n^ket a^k sont aussi associés à la couche ***k***.

Il importe de remarquer dans cet exemple que les couches qui suivent la première ont commeentrée la sortie de la couche précédente. Ainsi, on peut mettre en cascadeautant de couche que l'on veut,du moins en théorie !

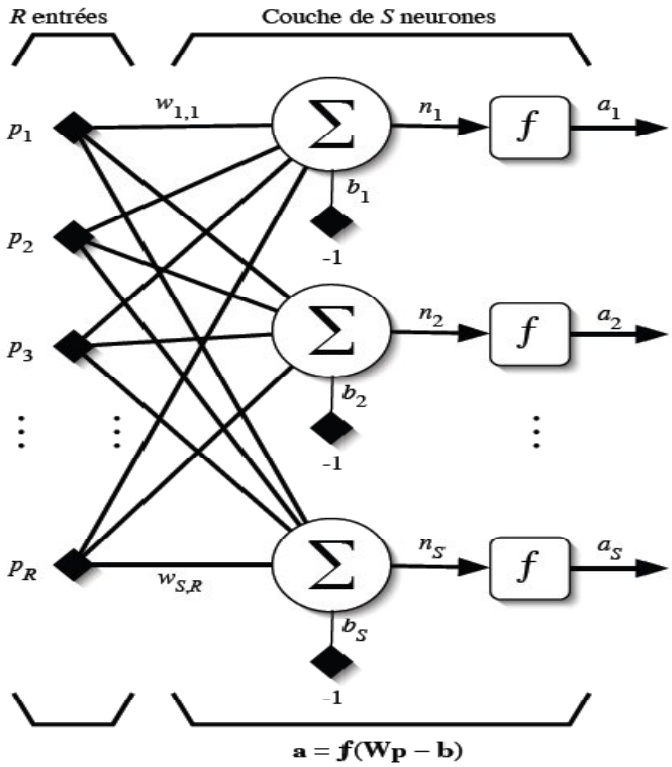


Figure 3.4 – Couche de *S* neurones.

Nous pouvons aussi fixer un nombre quelconque de neurones sur chaque couche. En pratique, il n'est pas souhaitable d'utiliser trop de neurones. Finalement, on note aussi que l'on peut changer de fonction de transfert d'une couche à l'autre. Ainsi, toujours dans le cas général, $f^1 \neq f^2 \neq f^3$.

La dernière couche est nommée «couche de sortie». Les couches qui précédent la couche de sortie sont nommées «couches cachées».

Les réseaux multicouches sont beaucoup plus puissants que les réseaux simples à une seulecouche. En utilisant deux couches (une couche cachée et une couche de sortie), à condition d'employerune fonction d'activation sigmoïde sur la couche cachée, on peut entraîner un réseau à produire une approximation de la plupart des fonctions, avec une précision arbitraire (cela peutcependant requérir un grand nombre de neurones sur la couche cachée). Sauf dans de rares cas, les réseaux de neurones artificiels exploitent deux ou trois couches.

Entrainer un réseau de neurones signifie modifier la valeur de ses poids et de ses biais pourqu'il réalise la fonction entrée/sortie désirée.

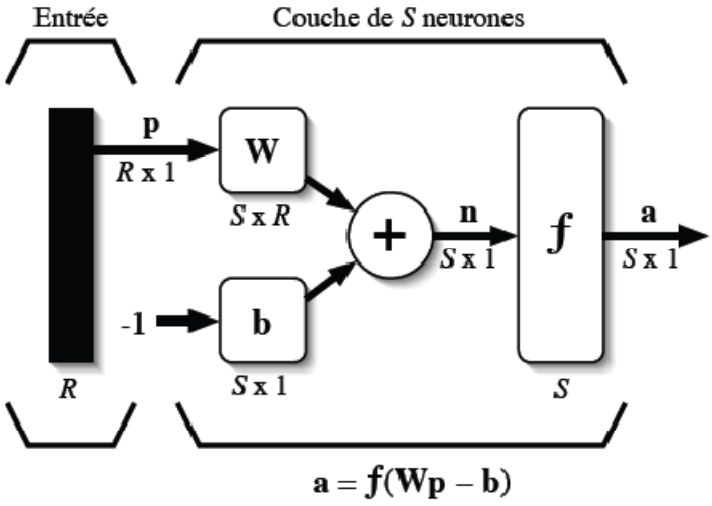


Figure3.5 – Représentation matricielle d'une couche de S neurones.

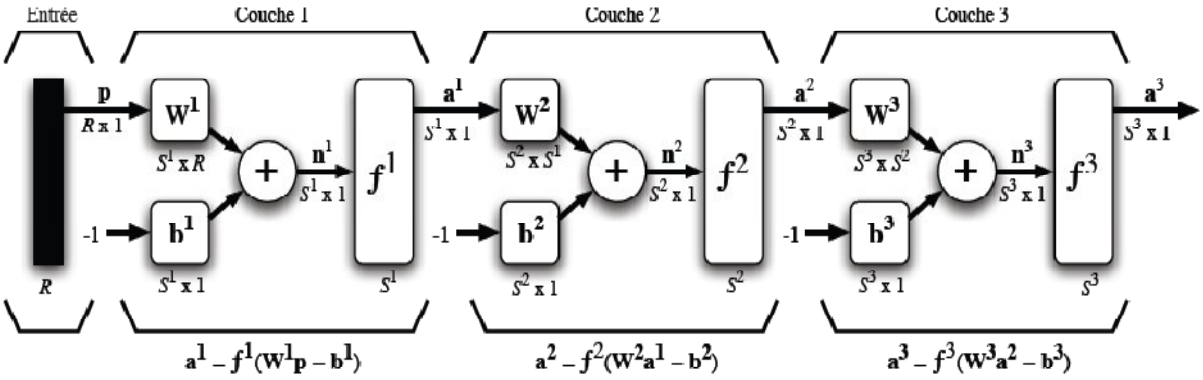


Figure3.6 – Représentation matricielle d'un réseau de trois couches.

Pour spécifier la structure duréseau, il faut aussi choisir le nombre de couches et le nombre de neurones sur chaque couche.

Tout d'abord, rappelons que le nombre d'entrées du réseau (***R***), de même que le nombre de neuronessur la couche de sortie est fixé par les spécifications du problème que l'on veut résoudre avecce réseau. Par exemple, si la donnée du problème comporte quatre variables en entrée et qu'elleexige de produire trois variables en sortie, alors nous aurons simplement $\boldsymbol{R = 4}$ et $\boldsymbol{S^M = 3}$, où***M*** correspond à l'indice de la couche de sortie (ainsi qu'au nombre de couches). Ensuite, la naturedu problème peut aussi nous guider dans le choix des fonctions de transfert. Par exemple, sil'on désire produire des sorties binaires 0 ou 1, alors on choisira probablement une fonction seuil(voir tableau 3.1) pour la couche de sortie. Il reste ensuite

àchoisir le nombre de couchescachées ainsi que le nombre de neurones sur ces couches, et leur fonction de transfert. Il faudraaussi fixer les différentsparamètres de l'algorithme d'apprentissage.

Finalement, la figure 3.7 illustre le dernier élément de construction employé pour bâtir des réseaux dit «récurrents».

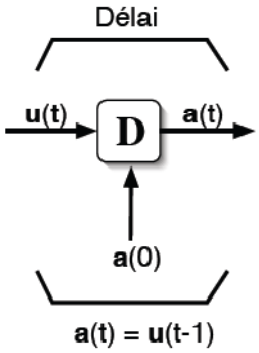


Figure 3.7 – Elément de retard.

Il s'agit d'un registre àdécalage qui permet d'introduireun retard dans une donnée que l'on veut acheminer dans un réseau. La sortie retardée***a(t)*** prendla valeur de l'entrée u au temps ***t−1***. Cet élément de retard présuppose que l'on peut initialiser lasortie au temps ***t = 0*** avec la valeur ***a(0)***. Cette condition initiale est indiquée à la figure 3.7 parune flèche qui entre par le bas de l'élément.

3.5 Processus d'apprentissage

Parmi les propriétés désirables pour un réseau de neurones, la plus fondamentale est sûrement la capacité d'apprendre de son environnement, d'améliorer sa performance à travers un processus d'apprentissage.

L'apprentissage est un processus dynamique et itératif permettant de modifier les paramètres d'un réseau en réaction avec les stimuli qu'il reçoit de son environnement. Le type d'apprentissage est déterminé par la manière dont les changements de paramètre surviennent.

Cette définition implique qu'un réseau se doit d'être stimulé par un environnement, qu'il subisse des changements en réaction avec cette stimulation, et que ceux-ci provoquent dans le futur une réponse nouvelle vis-à-vis de l'environnement. Ainsi, le réseau peut s'améliorer avec le temps. [5]

L'apprentissage est une phase du développement d'un réseau de neurones durant laquelle le comportement du réseau est modifié jusqu'à l'obtention du comportement désiré. [6]

L'apprentissage est la modification des poids du réseau dans l'optique d'accorder la réponse du réseau aux exemples et à l'expérience. Il est souvent impossible de décider à priori des valeurs des poids des connexions d'un réseau pour une application donnée. A l'issu de l'apprentissage, les poids sont fixés : c'est alors la phase d'utilisation. [7]

3.5.1 Supervisé

L'apprentissage dit « supervisé » est caractérisé par la présence d'un «professeur» qui possède une connaissance approfondie de l'environnement dans lequel évolue le réseau de neurones. En pratique, les connaissances de ce professeur prennent la forme d'un ensemble de Q couples de vecteurs d'entrée et de sortie que nous noterons $\{(p_1, d_1), (p_2, d_2), \ldots, (p_Q, d_Q)\}$, où p_i désigne un stimulus (entrée) et d_i la cible pour ce stimulus, c'est-à-dire les sorties désirées du réseau.

Chaque couple (p_i, d_i) correspond donc à un cas d'espèce de ce que le réseau devrait produire(la cible) pour un stimulus donné. Pour cette raison, l'apprentissage supervisé est aussi qualifié d'apprentissage par des exemples.

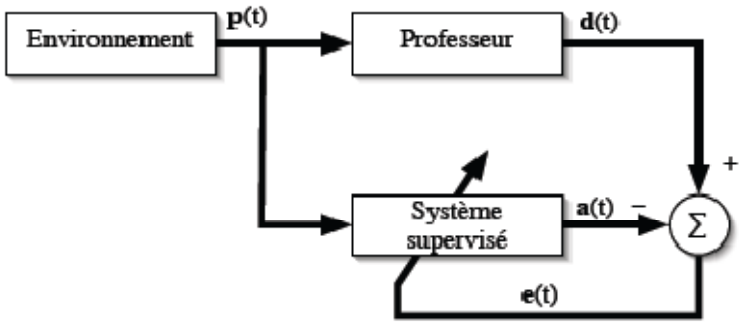


Figure3.8 – Schéma bloc de l'apprentissage supervisé.

3.5.2 Non-supervisé

L'apprentissage est dit «non-supervisé» ou encore«auto-organisé». Il est caractérisé par l'absence complète de professeur, c'est-à-dire qu'onne dispose pas de signal d'erreur, comme dans le cas supervisé. Nous ne disposons donc que d'un environnement qui fournitdes stimuli, et d'un réseau qui doit apprendre sans intervention externe. En assimilant les stimuli del'environnement à une description de son état interne, la tâche du réseau est alors de

modéliser cet état le mieux possible. Pour y arriver, il importe d'abord de définir une mesure de la qualité pource modèle, et de s'en servir par la suite pour optimiser les paramètres libres du réseau, c'est-à-dire ses poids synaptiques. A la fin de l'apprentissage, le réseau aura développé une habilité à former des représentations internes des stimuli de l'environnement permettant d'encoder les caractéristiques de ceux-ci et, par conséquent, de créer automatiquement des classes de stimuli similaires.

3.6 Règles d'apprentissage

3.6.1 Par correction d'erreur

La première règle que l'on peut utiliser est fondée sur la correction de l'erreur observée en sortie. Soit $a_i(t)$ la sortie que l'on obtient pour le neurone i au temps t. Cette sortie résulte d'unstimulus $p(t)$que l'on applique aux entrées du réseau dont un des neurones correspond au neurone$i$. Soit $d_i(t)$ la sortie que l'on désire obtenir pour ce même neurone i au temps t. Alors, $a_i(t)$ et$d_i(t)$ seront généralementdifférents et il est naturel de calculer l'erreur $e_i(t)$ entre ce qu'on obtientet ce qu'on voudrait obtenir :

$$e_i(t) = d_i(t) - a_i(t),$$

et de chercher un moyen de réduire autant que possible cette erreur. Sous forme vectorielle, on obtient:

$$\mathbf{e}(t) - \mathbf{d}(t) - \mathbf{a}(t),$$

avec$\mathbf{e(t) = [e_1(t)e_2(t) \cdots e_i(t) \cdots e_S(t)]}$ qui désigne le vecteur des erreurs observées sur les S neurones de sortie du réseau. L'apprentissage par correction des erreurs consiste à minimiser un indice de performance F basé sur les signaux d'erreur $\mathbf{e_i(t)}$, dans le but de faire converger les sorties du réseau avec ce qu'on voudrait qu'elles soient. Un critère très populaire est la somme des erreurs quadratiques :

$$F(\mathbf{e}(t)) = \sum_{i=1}^{S} e_i^2(t) = \mathbf{e}(t)^T \mathbf{e}(t).$$

Maintenant, il importe de remarquer que les paramètres libres d'un réseau sont ses poids. Prenons l'ensemble de ces poids et assemblons-les sous la forme d'un vecteur $w(t)$ au temps t. Pour minimiser $F(e(t)) = F(w(t)) = F(t)$, nous allons commencer par choisir des poids initiaux ($t = 0$) au hasard, puis nous allons modifier ces poids de la manière suivante :

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta \mathbf{x}(t),$$

où le vecteur ***x(t)*** désigne la direction dans laquelle nous allons chercher le minimum et ***η*** est une constante positive déterminant l'amplitude du pas dans cette direction (la vitesse d'apprentissage).

L'objectif est de faire en sorte que ***F(t + 1) <F(t)***. Mais comment peut-on choisir la direction ***x***pour que la condition précédente soit respectée ? Considérons la série de Taylor de 1er ordre autourde ***w(t)***:

$$F(t+1) - F(t) + \nabla F(t)^T \Delta \mathbf{w}(t),$$

où (désigne le gradient de ***F***par rapport à ses paramètres libres (les poids *w*) au temps *t*, et**Δw(t) = w(t + 1) − w(t)**. Or, pour que ***F(t + 1) < F(t)***, il faut que la condition suivante soit respectée:

$$\nabla F(t)^T \Delta \mathbf{w}(t) = \eta \nabla F(t)^T \mathbf{x}(t) < 0.$$

N'importe quel vecteur x(t) qui respecte l'inégalité précédente pointe donc dans une direction qui diminue ***F***. On parle alors d'une direction de «descente». Pour obtenir une descente maximum, étant donnée **η> 0**, il faut que le vecteur **x(t)**pointe dans le sens opposé au gradientcar c'est dans ce cas que le produit scalaire sera minimum :

$$\mathbf{x}(t) = -\nabla F(t)$$

Ce qui engendre la règle dite de «descente du gradient» :

$$\Delta \mathbf{w}(t) = -\eta \nabla F(t)$$

3.6.2 Par la règle de Hebb

L'apprentissage par la règle de Hebb exprime la variation de poids en fonction de la corrélation entre l'entrée ***p*** et la sortie ***a*** d'un neurone :

$$\Delta \mathbf{w} = \eta \, \mathbf{p} \, a.$$

Cette règle nous dit que plus la réponse du neurone sera forte vis-à-vis d'un stimulus, plus la variation de poids sera grande.

On peut formuler l'énoncé de Hebb sous la forme d'une règle d'apprentissage en deux parties:

1. Si deux neurones de part et d'autre d'une synapse (connexion) sont activés simultanément (d'une manière synchrone), alors la force de cette synapse doit être augmentée ;

2. Si les mêmes deux neurones sont activés d'une manière asynchrone, alors la synapse correspondantdoit être affaiblie ou carrément éliminé.

Mathématiquement, on peut exprimer la règle de Hebb sous sa forme la plus simple par la formule suivante :

$$\Delta w_j(t-1) = \eta p_j(t) a(t),$$

où$\boldsymbol{\eta}$ est une constante positive qui détermine la vitesse de l'apprentissage, $\boldsymbol{p_j(t)}$ correspond à l'activité pré-synaptique (l'entrée ***j*** du neurone) au temps ***t***, et ***a(t)***à l'activité post-synaptique (sortie du neurone) à ce même temps ***t***. Cette formule fait ressortir explicitement la corrélation entre le signal qui entre et celui qui sort. Sous une forme vectorielle, on écrit :

$$\Delta \mathbf{w}(t-1) = \eta \mathbf{p}(t) a(t).$$

3.6.3 Compétitif

L'apprentissage compétitif, comme son nom l'indique, consiste à faire entrer en compétition les neurones d'un réseau pour déterminer lequel sera actif à un instant donné. Contrairement aux autres types d'apprentissage ou, généralement, tous les neurones peuvent apprendre simultanément et de la même manière, l'apprentissage compétitif produit un «vainqueur» ainsi que, parfois, un ensemble de neurones «voisins» du vainqueur, et seuls ce vainqueur et, potentiellement, son voisinage bénéficient d'une adaptation de leur poids. On dit alors que l'apprentissage est local carlimité à un sous-ensemble des neurones du réseau.

Une règle d'apprentissage compétitif comporte les éléments suivants :

– Un ensemble de neurones identiques (même type) sauf pour les valeurs de leurs poids synaptiques;

– Une limite imposée à la «force» d'un neurone;

– Un mécanisme permettant aux neurones d'entrer encompétition pour le droit de répondre à uncertain sous-ensemble des stimuli d'entrée, de manière à ce qu'un seul neurone de sortie soit actif à la fois.

Ainsi, les neurones individuels peuvent apprendre à se spécialiser sur des sous-ensembles de stimulisimilaires pour devenir des détecteurs de caractéristiques.

Dans leur forme la plus simple, les réseaux de neurones qui utilisent l'apprentissage compétitif sont souvent constitués d'une seule couche de neurones de sortie, totalement

connectée sur les entrées. Un neurone vainqueur modifiera ses poids synaptiques en les rapprochant (géométriquement)d'un stimulus d'entrée p pour lequel il a battu tous les autres neurones lors de la compétition :

$$\Delta \mathbf{w} = \begin{cases} \eta(\mathbf{p} - \mathbf{w}) & \text{si le neurone est vainqueur} \\ 0 & \text{autrement} \end{cases},$$

où**0 <η< 1**correspond à un taux d'apprentissage. Un neurone qui ne gagne pas la compétition ne modifiera aucunement ses poids. Il ne sera donc pas affecté par le stimulus en question. Parfois on définit également un voisinage autour du neurone gagnant et on applique une règle similaire surles voisins, mais avec un taux d'apprentissage différent :

$$\Delta \mathbf{w} = \begin{cases} \eta_1(\mathbf{p} - \mathbf{w}) & \textbf{si} \text{ le neurone est vainqueur} \\ \eta_2(\mathbf{p} - \mathbf{w}) & \textbf{si} \text{ le neurone est voisin du vainqueur} \\ 0 & \text{autrement} \end{cases},$$

avec$\eta_2 \leq \eta_1$.

L'apprentissage compétitif est surtout utilisé dans le contexte d'un apprentissage dit non-supervisé, c'est-à-dire lorsqu'on ne connait pas lesvaleurs désirées pour les sorties du réseau.

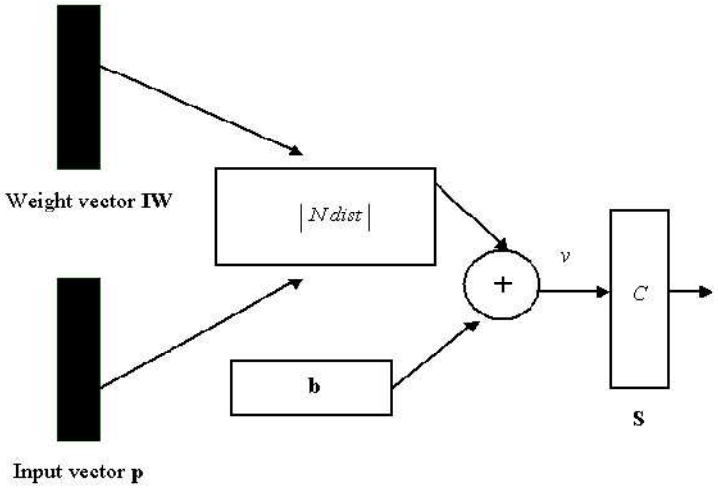


Figure 3.9 : Architecture d'un réseau compétitif

3.7Taches d'apprentissage

Approximation.

Soit la fonction g telle que :

$$d = g(p),$$

oùp est l'argument de la fonction (un vecteur) et d la valeur (un scalaire) de cette fonction évaluée en p. Supposons maintenant que la fonction $g(p)$est inconnue. La tâche d'approximation consiste alors à concevoir un réseau de neurones capable d'associer les éléments des couples entrée-sortie : $\{(p_1, d_1), (p_2, d_2), \ldots, (p_Q, d_Q)\}$. Ce problème peut êtrerésolu à l'aide d'un apprentissage supervisé sur les Q exemples, avec les $p_i$représentant les stimuli,et les $d_i$représentant les sorties désirées pour chacun de ces stimuli, avec $i = 1, 2, \ldots, Q$.

Ou inversement, on peut aussi dire que l'apprentissage supervisé est un problème d'approximationde fonction.

Association.

Il en existe deux types : l'auto-association et l'hétéro-association. Le problème de l'auto-association consiste à mémoriser un ensemble de patrons (vecteurs) en les présentant successivement au réseau. Par la suite, on présente au réseau une version partielle ou déformeéd'un patron original, et la tâche consiste à produire en sortie le patron original correspondant.

Le problème de l'hétéro-association consiste quant à lui à associer des paires de patrons :un patron d'entrée et un patron de sortie. L'auto-association implique un apprentissage non supervisé, alors que l'hétéro-association requiert plutôt un apprentissage supervisé.

Classement.

Pour cette tâche, il existe un nombre fixe de catégories (classes) de stimulid'entrée que le réseau doit apprendre àreconnaitre. Dans un premier temps, le réseau doitentreprendre une phase d'apprentissage supervisé durant laquelle les stimuli sont présentés en entrée et les catégories sont utilisées pour former les sorties désirées, généralement enutilisant une sortie

par catégorie. Ainsi, la sortie 1 est associée à la catégorie 1, la sortie 2 à la catégorie 2, etc. Pour un problème comportant ***Q***catégories, on peut par exemple fixer les sorties désirées ***d =[d1, d2, . . . , dQ]T***à l'aide de l'expression suivante :

$$d_i = \begin{cases} 1 & \text{si le stimulus appartient à la catégorie } i \\ 0 & \text{autrement} \end{cases}, i = 1, \ldots, Q.$$

Par la suite, dans une phase de reconnaissance, il suffira de présenter au réseau n'importequel stimulus inconnu pour pouvoir procéder au classement de celui-ci dans l'une ou l'autredes catégories. Une règle simple de classement consiste, par exemple, à choisir la catégorie associée avec la sortie maximale.

Prédiction.

La notion de prédiction est l'une des plus fondamentales en apprentissage. Ils'agit d'un problème de traitement temporel de signal. En supposant que nous possédons***M*** échantillons passés d'un signal, ***x(t−1), x(t−2), . . . , x(t−M)***, échantillonnés à intervallede temps fixe, la tâche consiste àprédire la valeur de ***x***au temps ***t***. Ce problème de prédiction peut être résolugrâce à un apprentissage par correction des erreurs, mais d'une manière non supervisée (sans professeur), étantdonné que les valeurs de sortie désirée peuvent êtrefixées directement de la série chronologique. Plus précisément, l'´échantillon de ***x(t)*** peut servir de valeur désirée et le signal d'erreur pour l'adaptation des poids se calcule simplement parl'´équation suivante :

$$e(t) = x(t) - \widehat{x}(t \mid t-1, t-2, \ldots, t-M),$$

où x(t) désigne la sortie désirée et ***x^(t | t−1, t−2, . . . , t−M)***représente la sortie observée du réseau étantdonné les***M***échantillonsprécédents. La prédiction s'apparente à la constructiond'un modèle physique de la série chronologique. Dans la mesure où le réseaupossède desneurones dont la fonction de transfert est non-linéaire, le modèle pourra lui-aussi être non-linéaire.

Commande.

La commande d'un processus est une autre tâche d'apprentissage que l'on peutaborder à l'aide d'un réseau de neurones. Considérons un système dynamique non-linéaire***{u(t), y(t)}***où***u(t)***désigne l'entrée du système et ***y(t)***correspond à la réponse de celui-ci.

Dans le cas général, on désire commander ce système de manière à ce qu'il se comporteselon un modèle de référence, souvent un modèlelinéaire, ***{r(t), d(t)},***où pour tout temps***t≥ 0***, on arrive à produire une commande u(t) telle que :

$$\lim_{t\to\infty} |d(t) - y(t)| = 0,$$

demanièreà ce que la sortie du système suive de près celle du modèle de référence. Cecipeut se réalisergrâce à certains types de réseauxsupervisés.

Chapitre 4

PLANIFICATION DE TRAJECTOIRE

4.1Modèle neuronal proposé

La méthode que nous proposons concerne la planification de trajectoire en temps réel dans un milieu statique.

4.1.1 Environnement statique

Dans ce cas le robot est le seul élément en mouvement, les obstacles qui l'entourent et qui se trouvent dans son champ de travail ont des positions fixes.

Afin d'obtenir l'image d'un obstacle dans l'espace des configurations, on utilise la décomposition cellulaire, obtenue par balayage selon les différents Axes Ox, Oy et Oz. Le choix du pas de discrétisation dépend essentiellement de la précision désirée.

Le problème se pose de la manière suivante, le robot se trouve initialement dans une configuration (point initial dans l'espace cartésien) de départ qui correspond à la cellule C_0 (x_0,y_0,z_0) il veut atteindre la cellule $C_t(x_t,y_t,z_t)$ qui correspond à la configuration d'arrivée (point final dans l'espace cartésien). La trajectoire suivit entre la cellule C_0 de départ et la cellule C_t d'arrivée est définie par un ensemble de cellules $\{C_0, C_1 C_2,\ldots C_t\}$.

A chaque point Ci de cet espace cartésien discrétisé, on affecte un neurone $n_i$qui est l'image d'une cellule C_i. Dans un espace à trois dimensions les voisins d'un point C_i sont au nombre de 26, comme indiqué sur la figure 4.1 et donc le nombre de neurones du réseau est égal à 26. Chaque neurone n_i est relié aux 26 neurones voisins qui l'entourent et qui représentent les cellules en contact direct avec la cellule C_i.

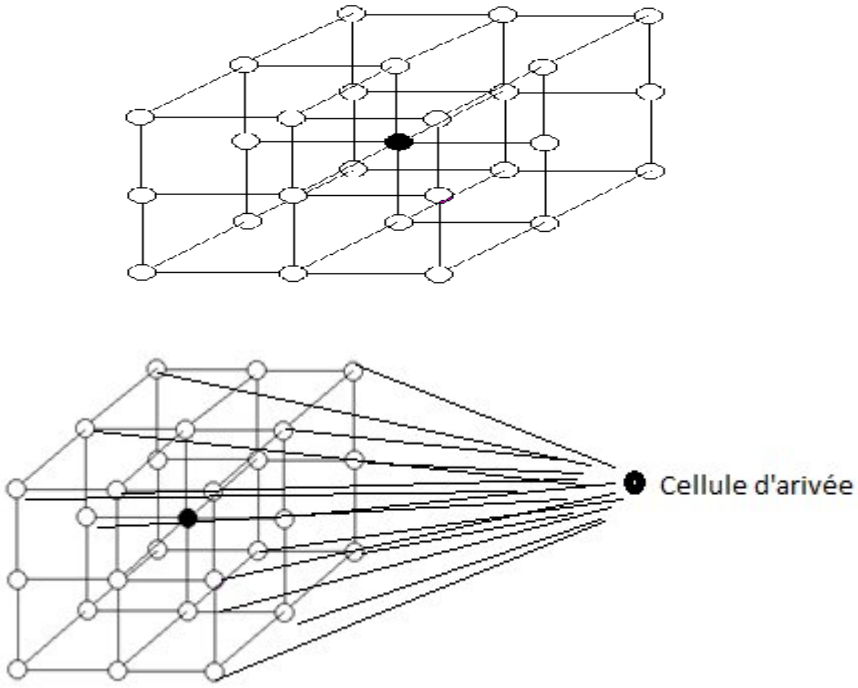


Figure 4.1 Voisins interconnectés d'un neurone.

A chaque itération une combinaison de 26 cellules servira à paramétrer les poids du réseau (IW). Les 26 neurones entrent alors en compétition pour choisir un vainqueur dont la distance euclidienne (Ndist)le séparant de la cellule d'arrivée est la moindre. Ce vainqueur va permettre de choisir la prochaine combinaison des 26 cellules servant à configurer le réseau.

4.1.2 Les biais Ii

Chaque neurone $n_i$reçoit une entrée de biais I_i qui l'informe sur la présence ou non d'un obstacle éventuel. Si le neurone ni correspond à une cellule qui fait partie d'un obstacle, son biais I_i= -Esinon la cellule est libre et le biais est nul I_i=0.

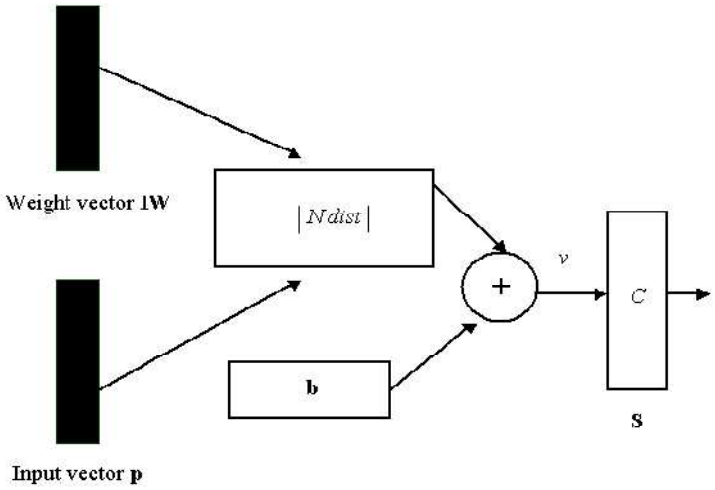


Figure 4.2 : Architecture d'un réseau compétitif

4.1.3 Les poids Wi

Les poids Wi des liaisons reliant l'entréedu réseau (la cellule d'arrivée C_t) aux26 neurones ni {i=1,2…26} sont égales respectivement aux coordonnées cartésiennes x,y et z de chaque cellule image des 26 neurones. Figure 4.3

L'état initial du réseau de neurones est tel que toutes les sorties x_i des neurones sont nulles :

$$x_i=0 \quad \{i=1,2…26\}$$

Les valeurs attribuées aux biais obligent le réseau de se déplacer du neurone n_0 correspondant à la cellule initiale C_0 vers les neurones voisins progressivement et à chaque itération en évitant tout point faisant partie d'un obstacle jusqu'à atteindre le neurone n_t qui correspond à la cellule de configuration finale C_t. Les cellules $\{C_0, C_1 C_2,… C_t\}$ constituent la trajectoire du robot.

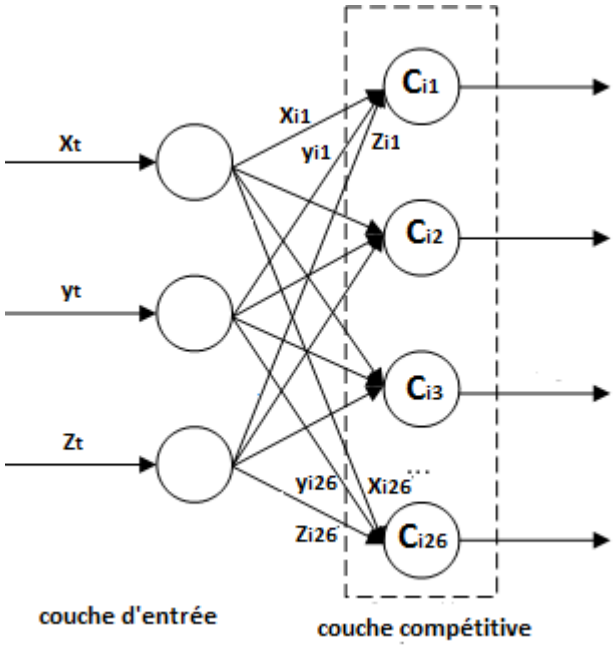


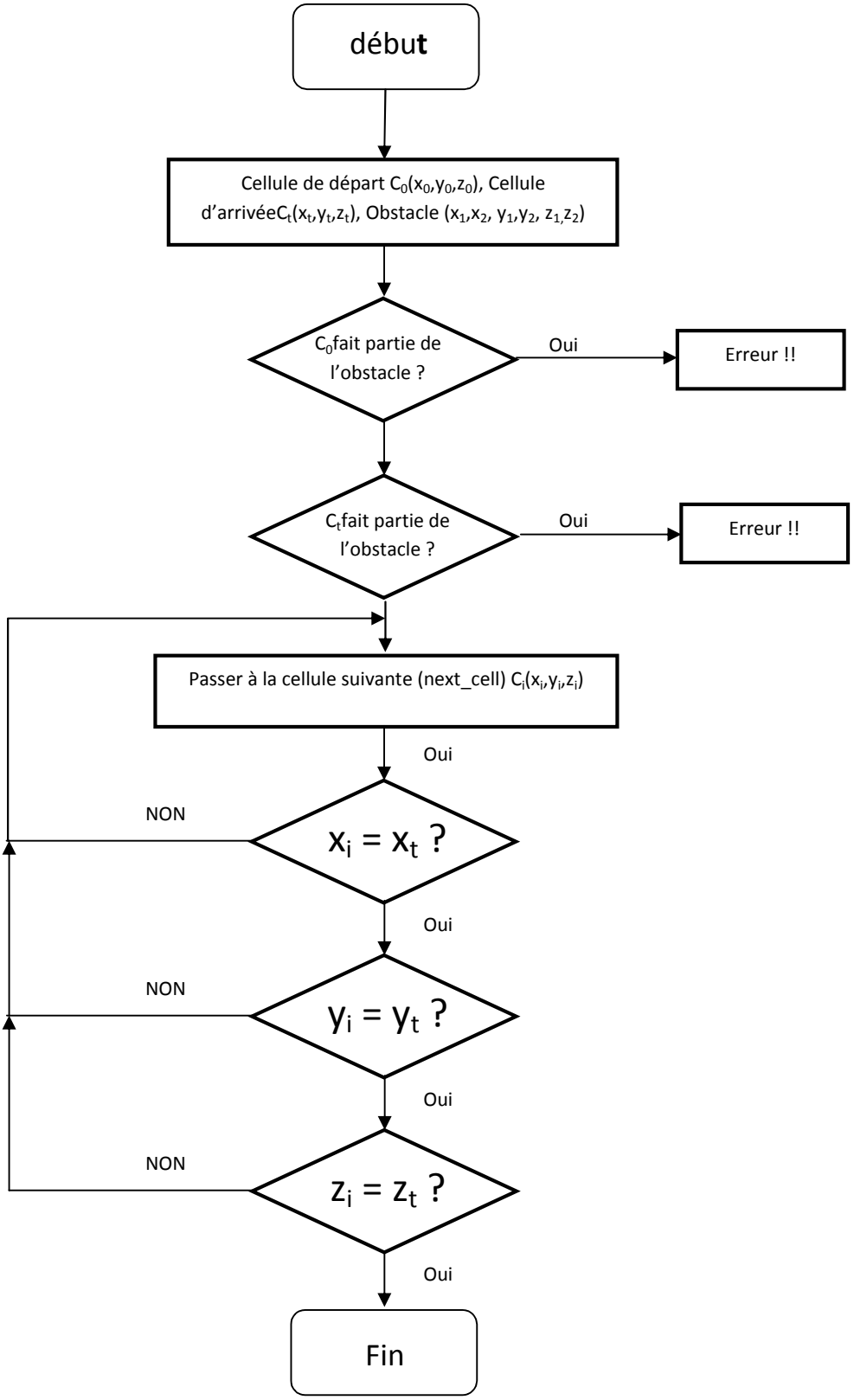
Figure 4.3 le réseau de neurones développé

Dans le cas où le réseau entre dans un minimum local, en présence d'un obstacle (le robot alors oscille entre deux cellules ou tourne en rond!), l'évolution du réseau s'arrête sans atteindre l'objectif. Pour remédier à un tel état on a introduit un déplacement sur l'axe Oz correspondant à la distance minimale séparant la cellule ou se trouve le robot et la limite de l'obstacle.

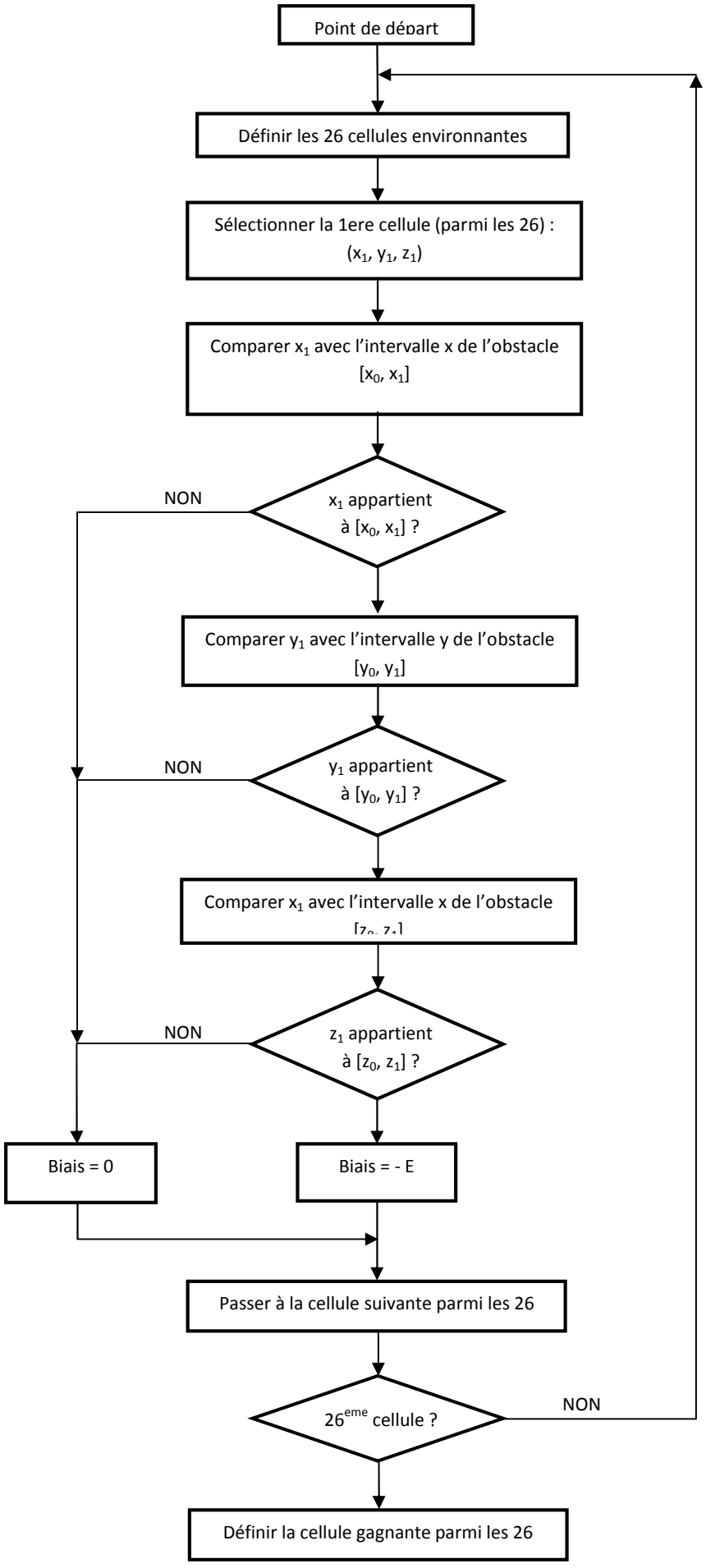
Ceci permet au réseau de neurones de sortir directement de cette situation et de poursuivre son évolution jusqu'à atteindre la cellule cible C_t. Alors le réseau de neurones proposé est donc stable.

4.1.4 Les algorithmes

Traject_obst



Next_cell


Point de départ
Définir les 26 cellules environnantes
Sélectionner la 1ere cellule (parmi les 26) : (x_1, y_1, z_1)
Comparer x_1 avec l'intervalle x de l'obstacle [x_0, x_1]
x_1 appartient à [x_0, x_1] ?
NON
Comparer y_1 avec l'intervalle y de l'obstacle [y_0, y_1]
y_1 appartient à [y_0, y_1] ?
NON
Comparer x_1 avec l'intervalle x de l'obstacle [z_0, z_1]
z_1 appartient à [z_0, z_1] ?
NON
Biais = 0
Biais = - E
Passer à la cellule suivante parmi les 26
26^{eme} cellule ?
NON
Définir la cellule gagnante parmi les 26

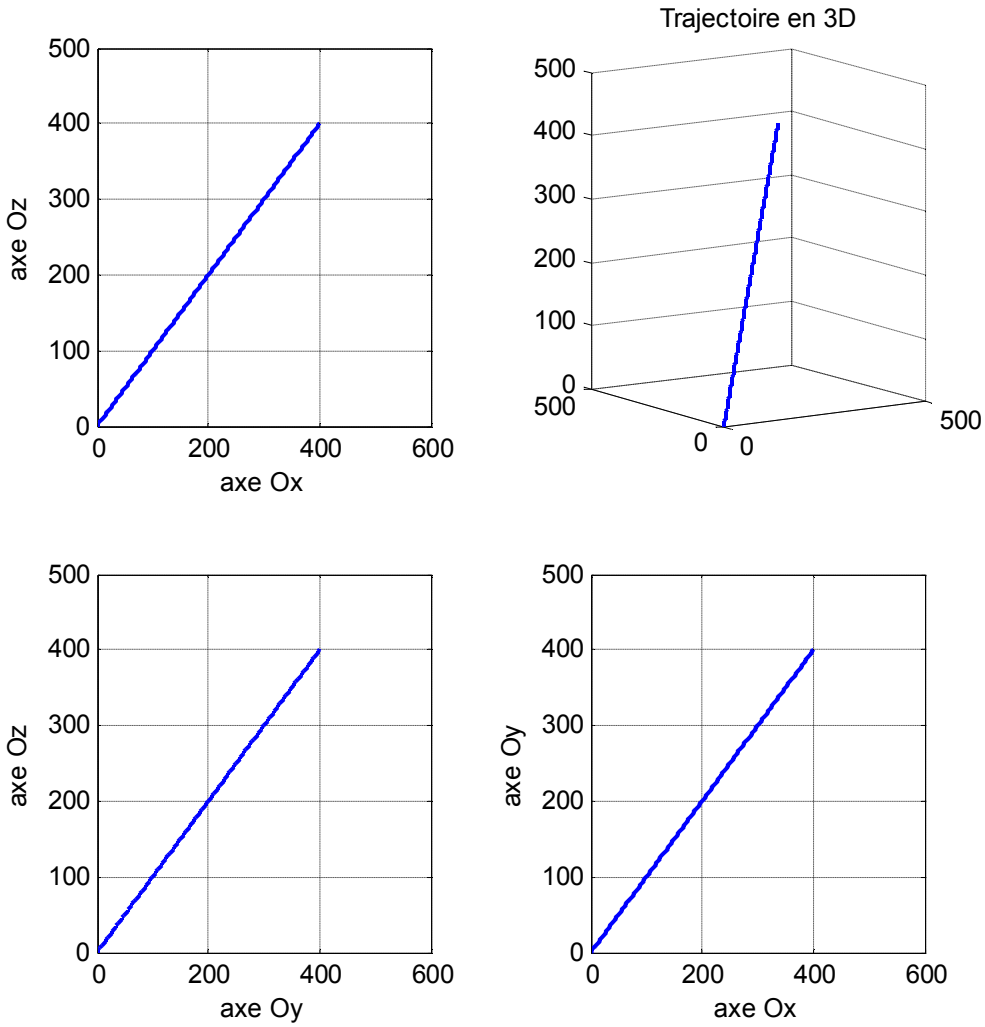
D'après les travaux de simulation effectués avec la méthode neuronale proposée, nous avons constaté une grande efficacité de contournement des obstacles. Nous présentons dans ce qui suit quelques exemples sans et avec la présence d'obstacles (plan, cube, parallelipède).

4.1.5 Exemples d'application du réseau développé.

Exp1 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (sans obstacle)
(0,0,0)	(400,400,400)	Sur l'axe Ox : Néant. Sur l'axe Oy : Néant. Sur l'axe Oz : Néant.

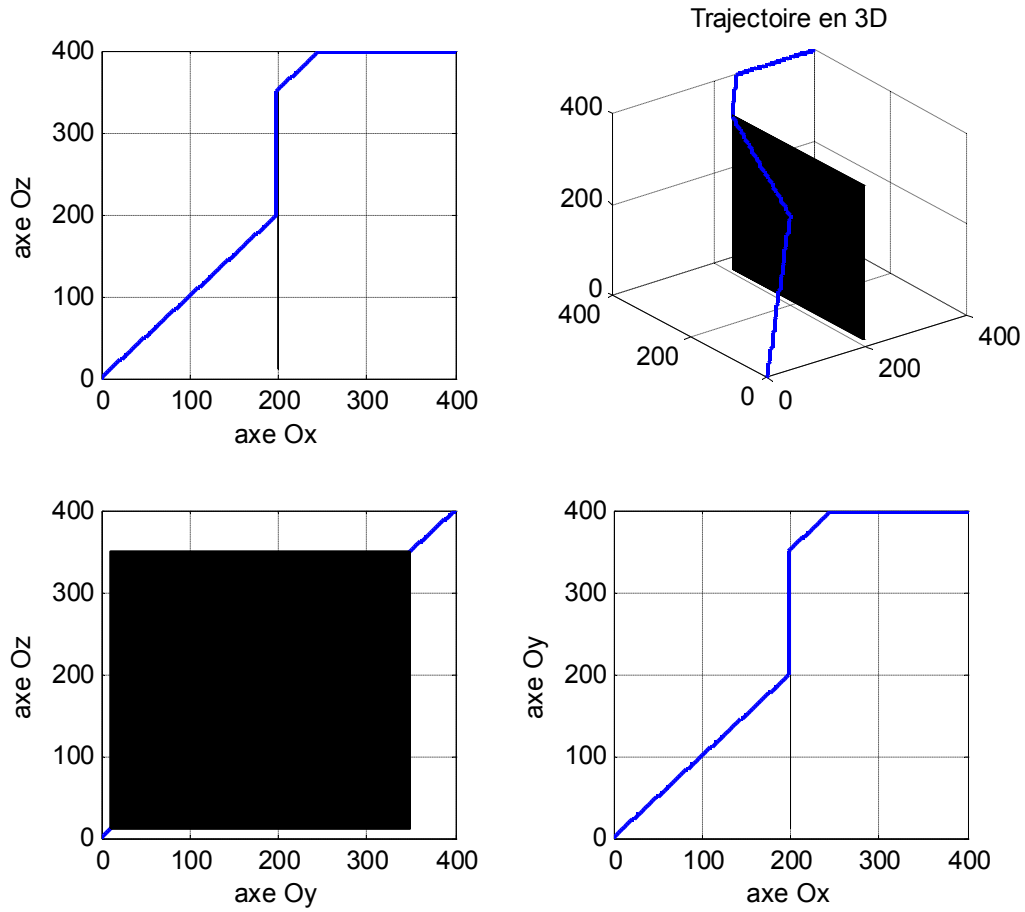
Trajectoire générée en 400 itérations !



Exp2 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (plan X)
(0,0,0)	(400,400,400)	Sur l'axe Ox : [199-200]. Sur l'axe Oy : [10-350]. Sur l'axe Oz : [10-350].

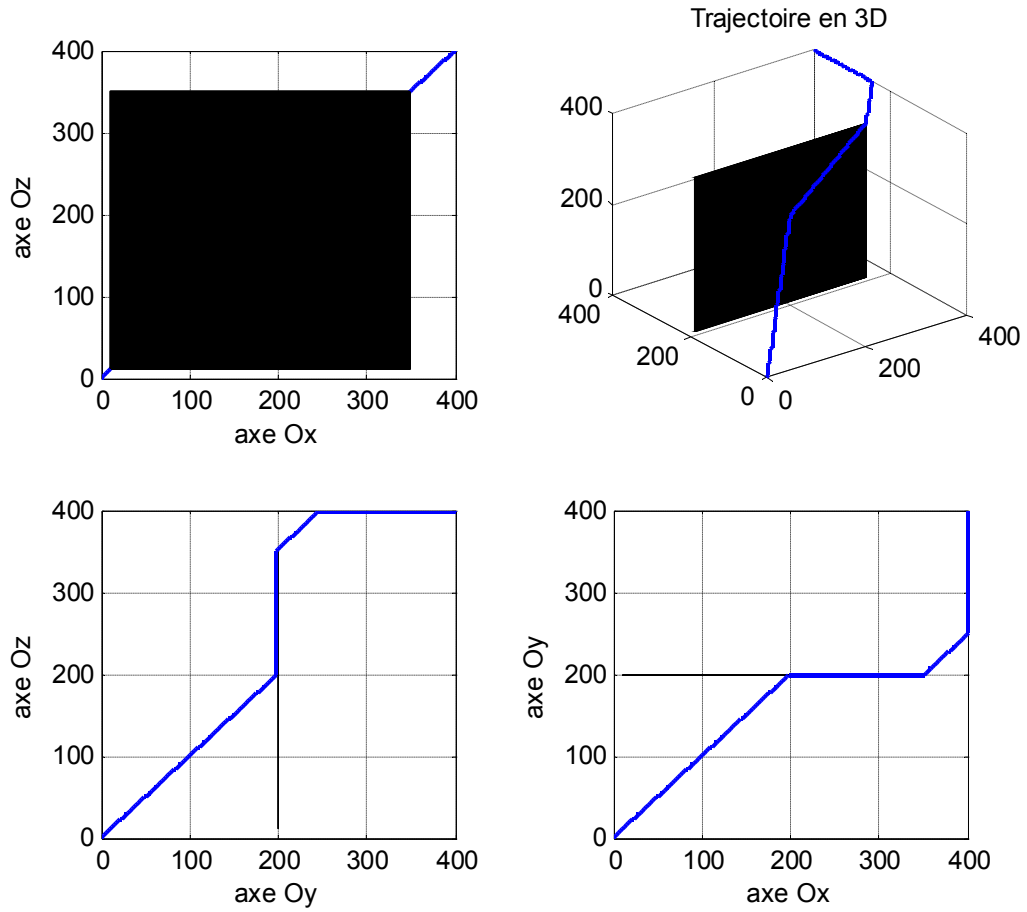
Trajectoire générée en 552 itérations !



Exp 3 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (plan Y)
(0,0,0)	(400,400,400)	Sur l'axe Ox : [10-350]. Sur l'axe Oy : [199-200]. Sur l'axe Oz : [10-350].

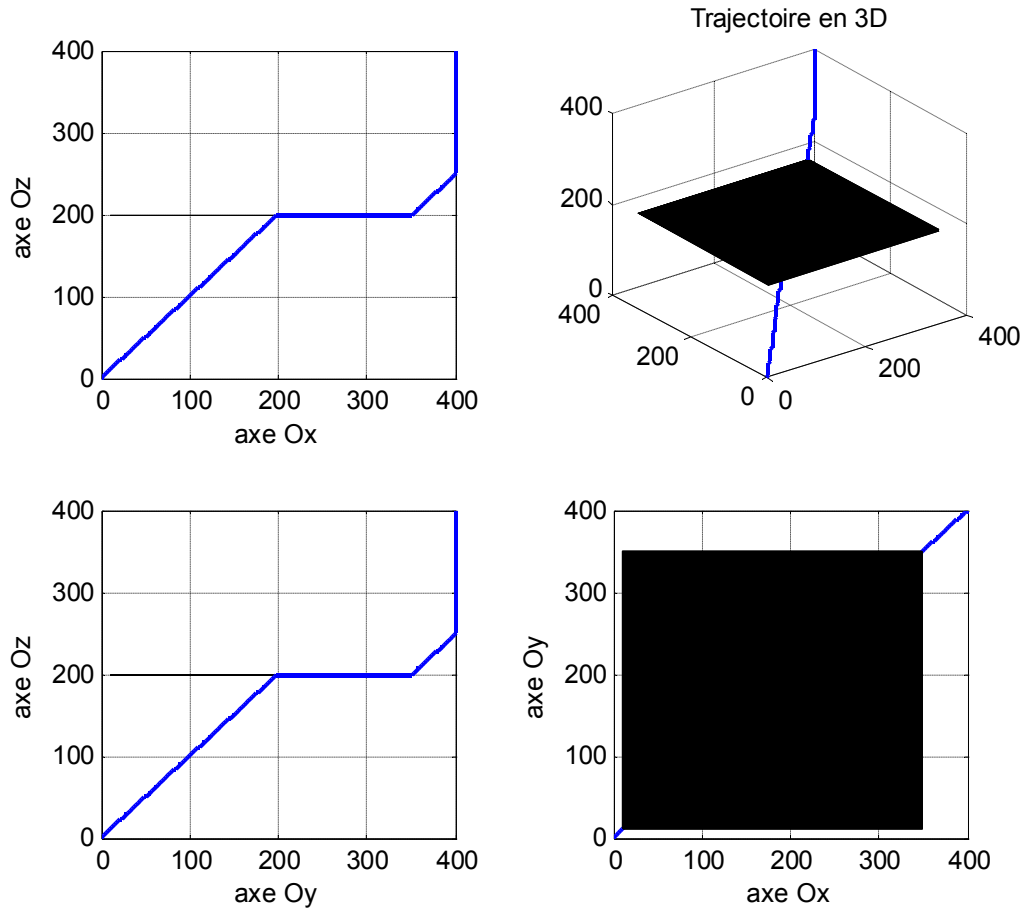
Trajectoire générée en 552 itérations !



Exp 3 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (plan Z)
(0,0,0)	(400,400,400)	Sur l'axe Ox : [199-200]. Sur l'axe Oy : [10-350]. Sur l'axe Oz : [10-350].

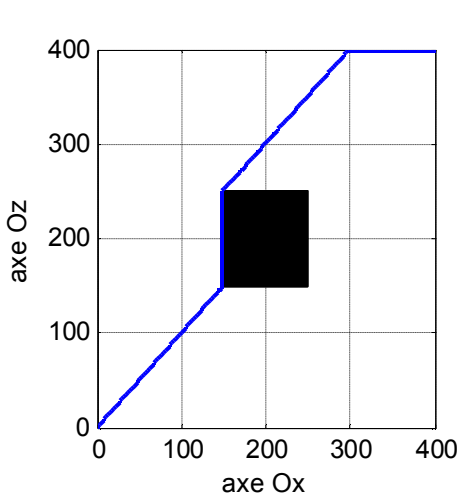
Trajectoire générée en 552 itérations !

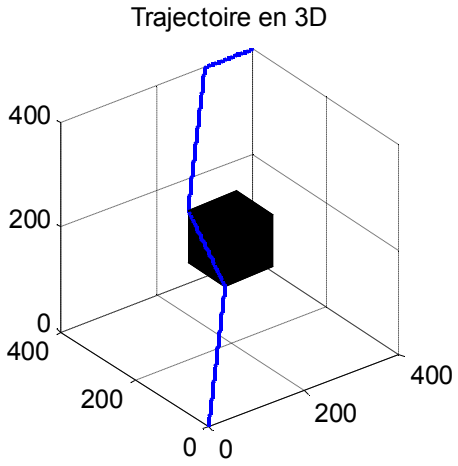


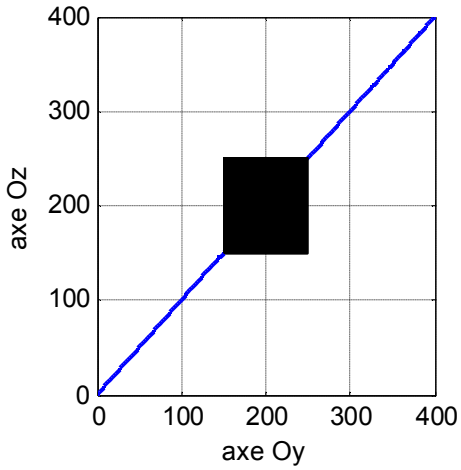
Exp4 :

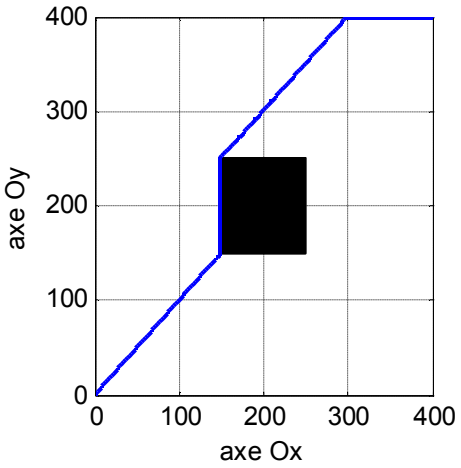
Cellule de départ C_0	Cellule cible C_t	Obstacle (cube)
(0,0,0)	(400,400,400)	Sur l'axe Ox : [150-250]. Sur l'axe Oy : [150-250]. Sur l'axe Oz : [150-250].

Trajectoire générée en 500 itérations !





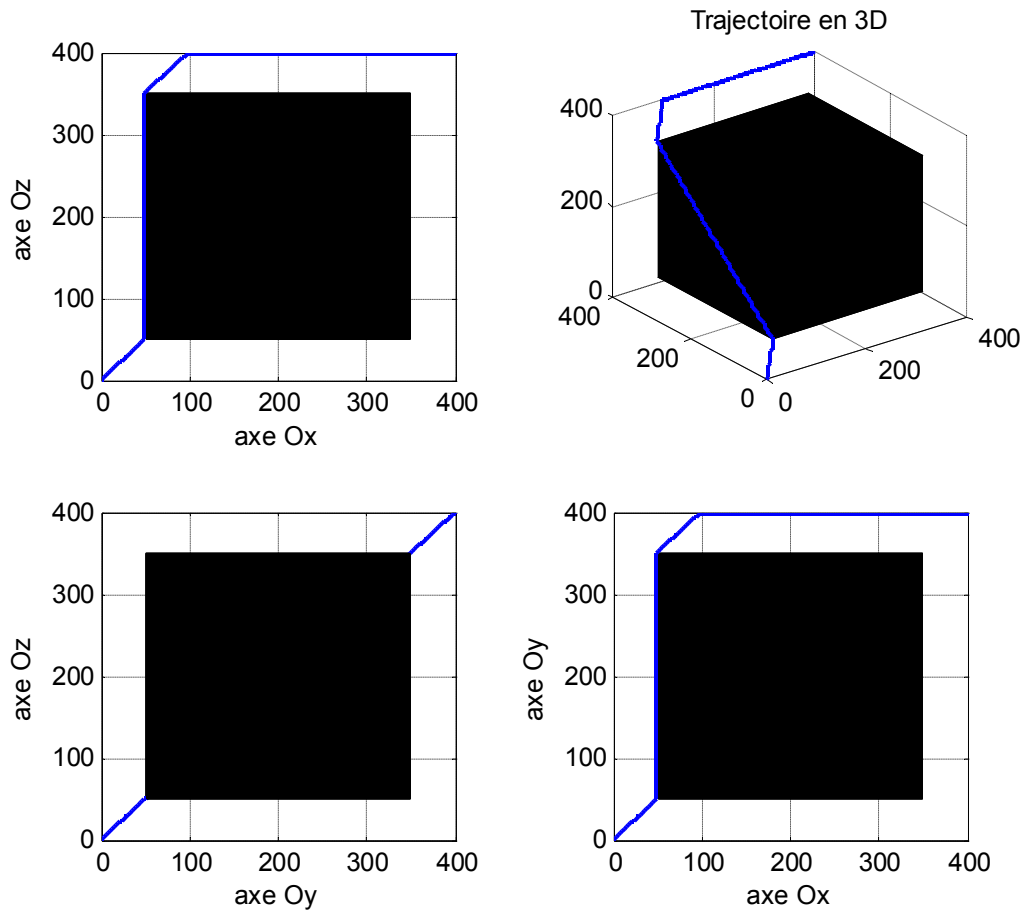




Exp5 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (cube)
(0,0,0)	(400,400,400)	Sur l'axe Ox : [50-350]. Sur l'axe Oy : [50-350]. Sur l'axe Oz : [50-350].

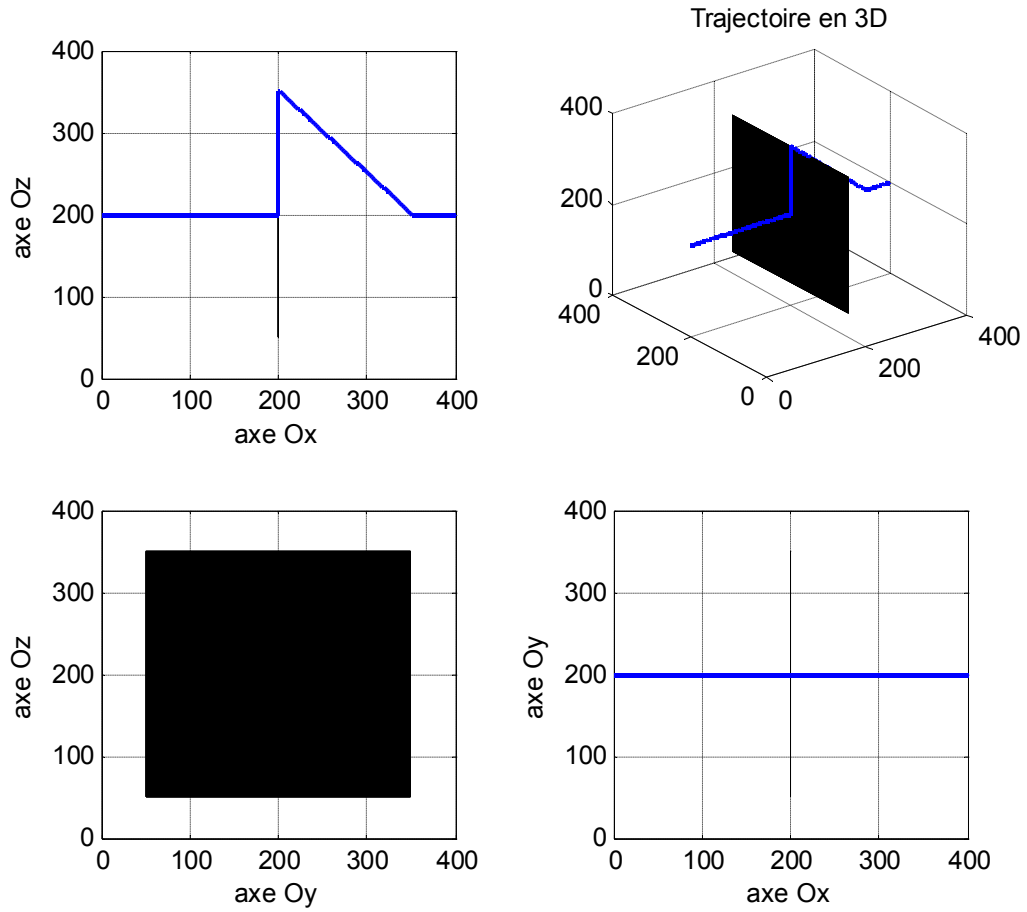
Trajectoire générée en 701 itérations !



Exp6 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (plan X)
(0,200,200)	(400,200,200)	Sur l'axe Ox : [200-200]. Sur l'axe Oy : [50-350]. Sur l'axe Oz : [50-350].

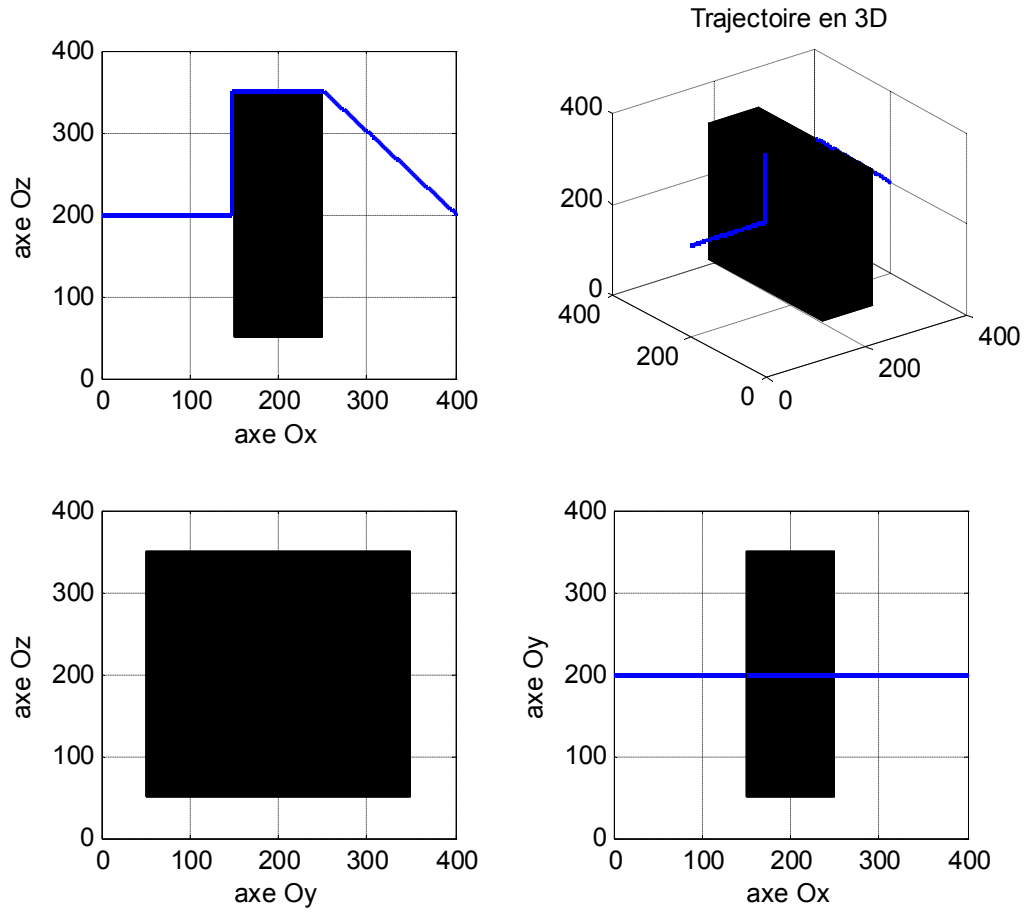
Trajectoire générée en 400 itérations !



Exp7 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (parallelipède)
(0,200,200)	(400,200,200)	Sur l'axe Ox : [150-250]. Sur l'axe Oy : [50-350]. Sur l'axe Oz : [50-350].

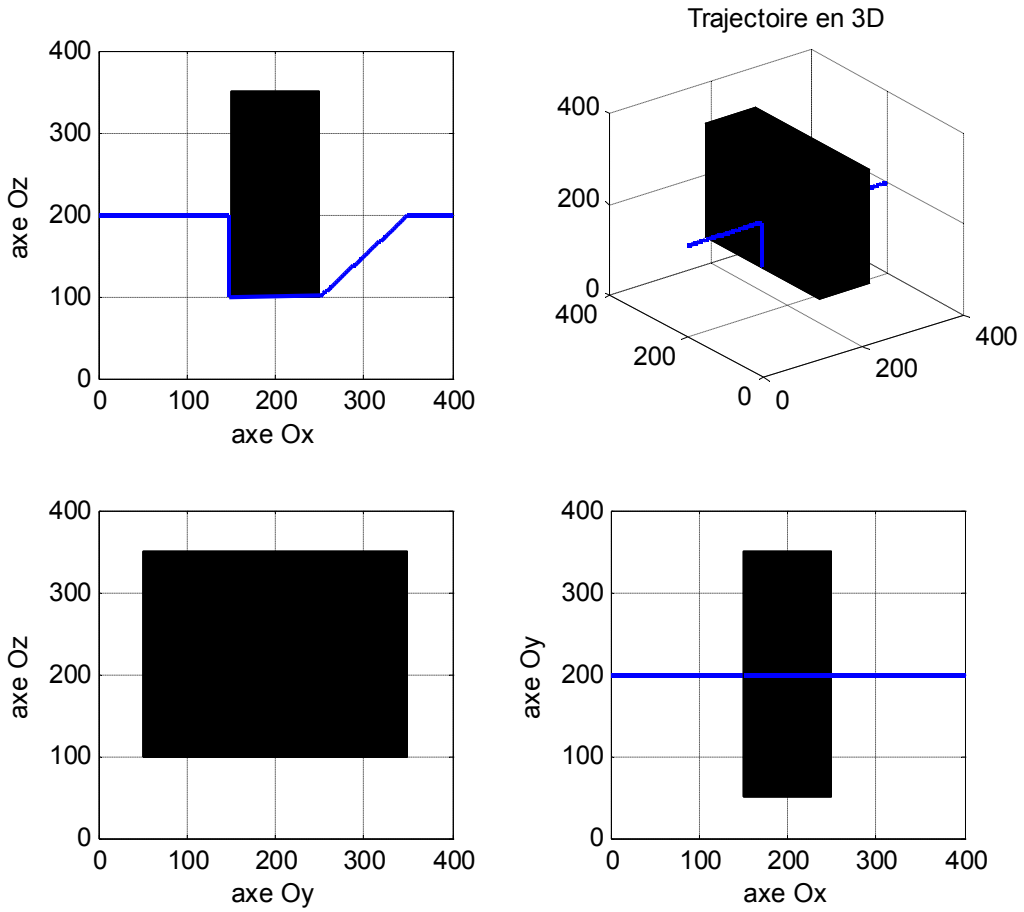
Trajectoire générée en 302 itérations !



Exp 8 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (parallelipède)
(0,200,200)	(400,200,200)	Sur l'axe Ox : [150-250]. Sur l'axe Oy : [50-350]. Sur l'axe Oz : [100-350].

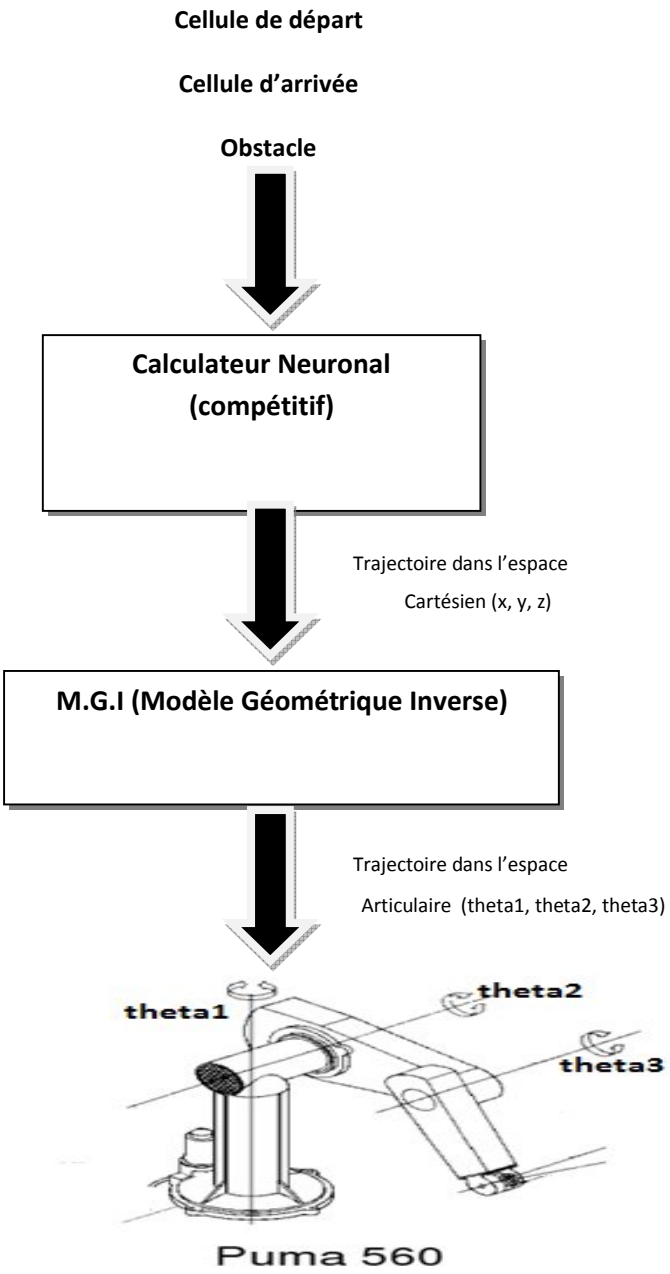
Trajectoire générée en 300 itérations !



4.1.6Application au robot manipulateur PUMA560

La planification de trajectoire précédemment développée est appliquée au bras manipulateur PUMA560. Vu que les trois dernières articulations concernant les mouvements du poignet sont de dimensions réduites, uniquement les trois premiers degrés de liberté sont considérés.

Pour passer de l'espace cartésien à l'espace articulaire (modèle géométrique inverse) on a utilisé l'outil matlab « roboticstoolbox release 9» développé par Peter Corke[24] qui entre autres prend en considération l'espace de travail ainsi que les singularités dues à la géométrie du puma 560 !

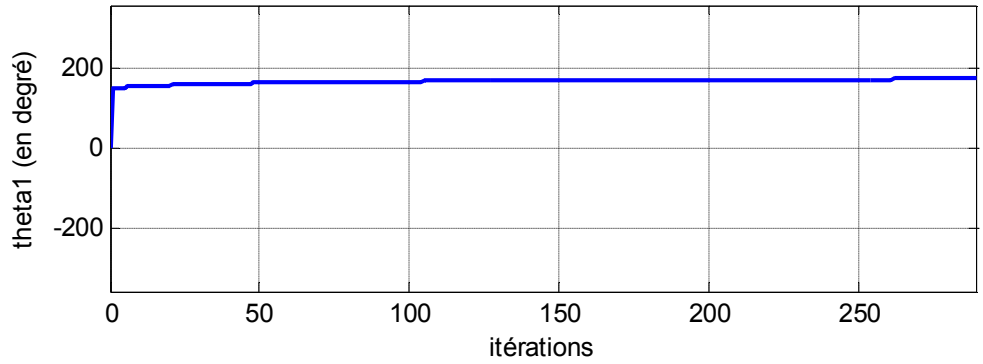


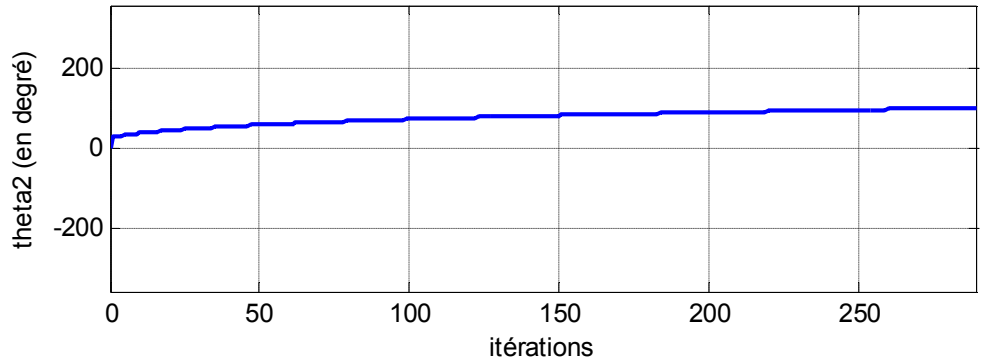
Puma 560

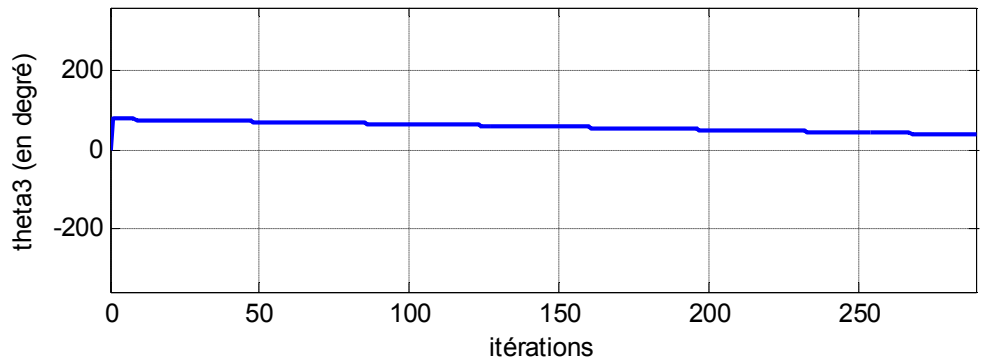
Exp1 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (sans obstacle)
(110,110,110)	(400,110,110)	Sur l'axe Ox : Néant. Sur l'axe Oy : Néant. Sur l'axe Oz : Néant.

Trajectoire générée en 289 itérations !



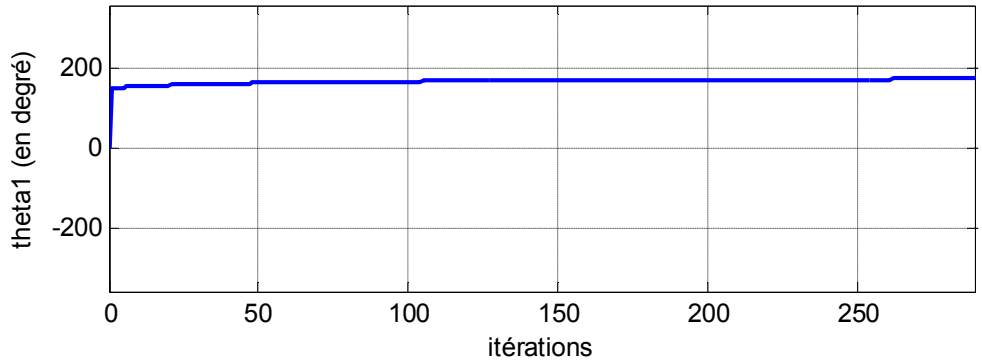


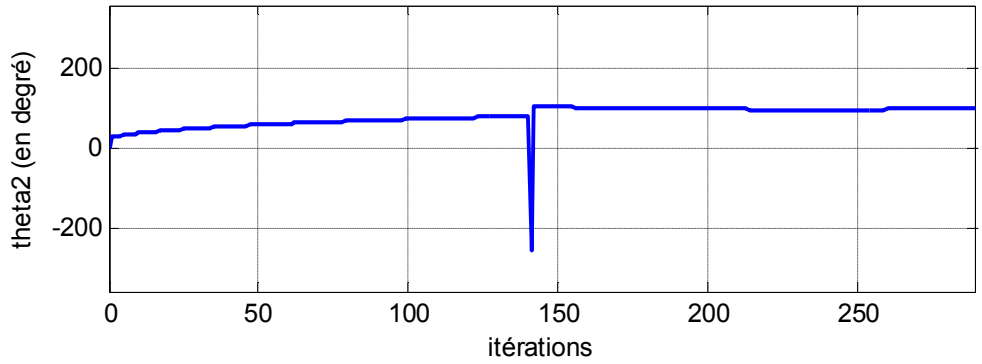


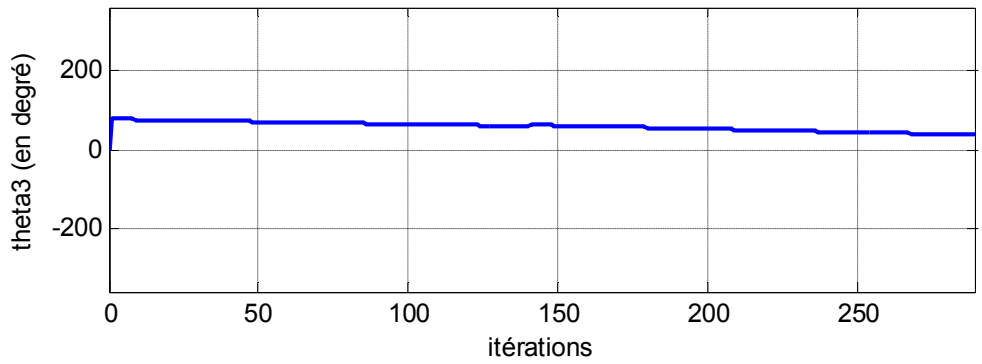
Exp2 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (Plan X)
(110,110,110)	(400,110,110)	Sur l'axe Ox : [250-250]. Sur l'axe Oy : [0-500]. Sur l'axe Oz : [0-500].

Trajectoire générée en 290 itérations !



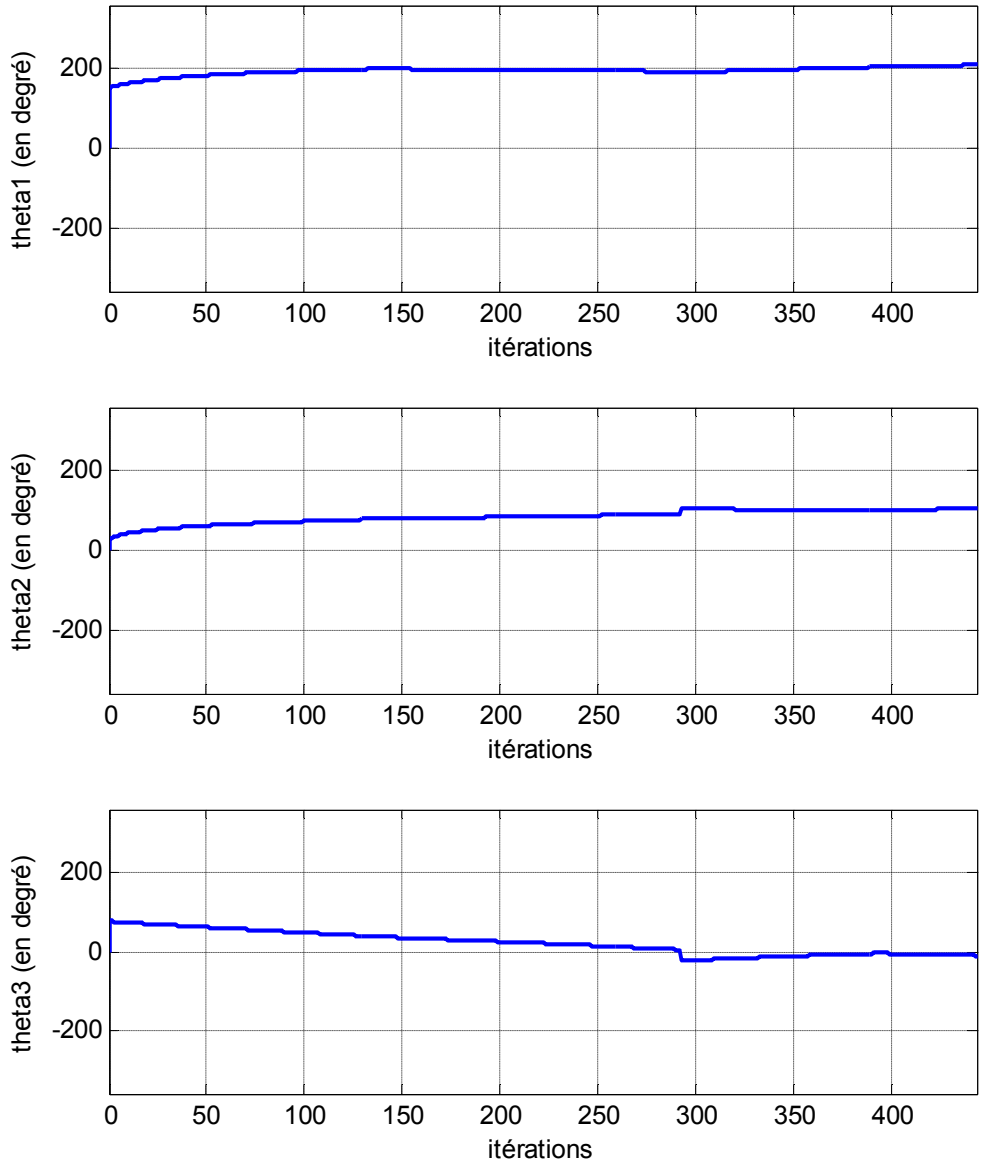




Exp 3 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (Plan Y)
(110,110,110)	(400,400,400)	Sur l'axe Ox : [0-500]. Sur l'axe Oy : [250-250]. Sur l'axe Oz : [0-500].

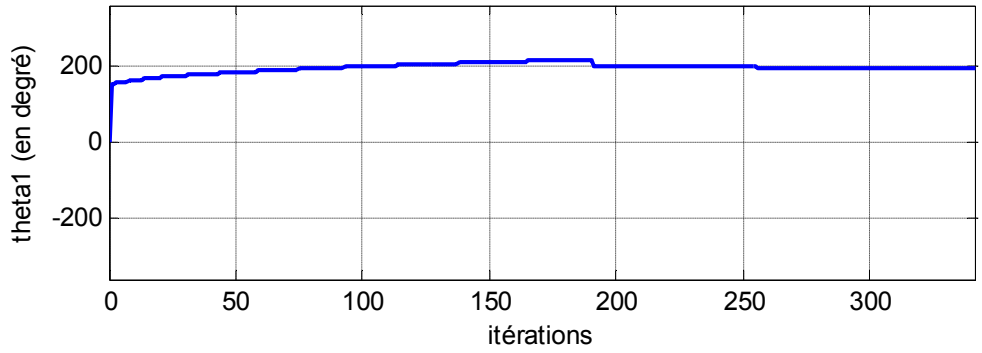
Trajectoire générée en 443 itérations !

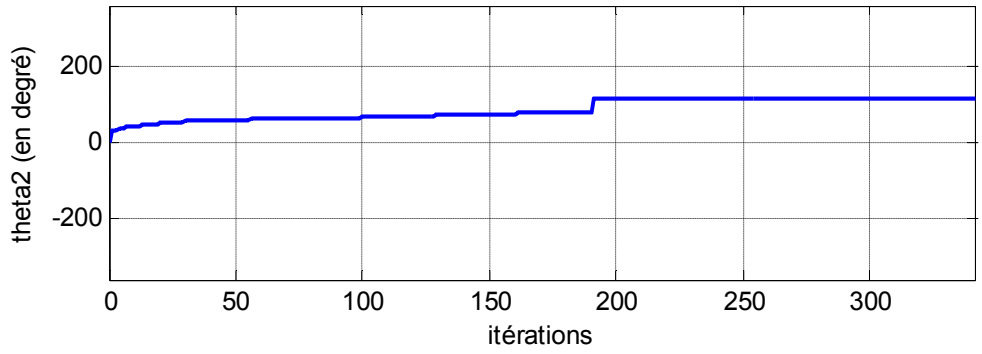


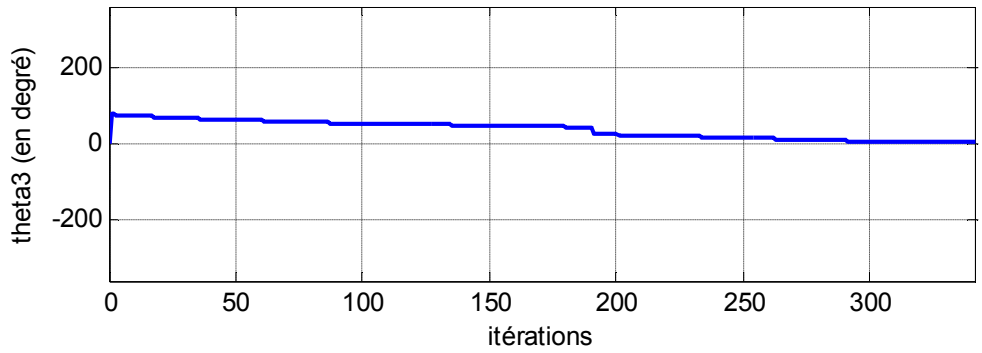
Exp 4 :

Cellule de départ C_0	Cellule cible C_t	Obstacle (Parallelipède)
(110,110,110)	(500,300,200)	Sur l'axe Ox : [150-400]. Sur l'axe Oy : [50-400]. Sur l'axe Oz : [50-400].

Trajectoire générée en 342 itérations !







4.1.7 Conclusion

Nous avons constatéque la méthode neuronale développée est efficace et qu'elle constitue une très bonne solution face au problème de planification de trajectoire en temps réel. La trajectoire obtenue est optimale dans le sens du plus court chemin.

Les avantages de notre méthode sont entre autre la facilité d'implémentation et de représentation. On travaille sur une grille et chaque cellule est en 26-connexité. C'est-à-dire que d'une cellule, on a que 26 choix de déplacement possible.

Par contre, notre méthode possèdeun défaut. Notamment le fait que les contraintes cinématiques du robot ne sont pas prises en compte (accélération, vitesse, vitesse angulaire maximale etc.).

CONCLUSION GENERALE

L'objectif principal de ce travail était l'utilisation des réseaux de neurones non supervisés dans la planification de trajectoire pour un bras de robot. Bien que beaucoup de travaux aient été réalisés dans ce domaine, l'utilisation des réseaux de neurones non supervisés est vraiment rare.

Le problème majeur rencontré était la projection de l'espace cartésien discrétisé sur un réseau de neurones, càd associer à chaque point de l'espace un neurone artificiel !

Nous avons constatéque la méthode neuronale développée est efficace et qu'elle constitue une très bonne solution face au problème de planification de trajectoire en temps réel. La trajectoire obtenue est optimale dans le sens du plus court chemin.

Les avantages de notre méthode sont entre autre la facilité d'implémentation et de représentation. On travaille sur une grille et chaque cellule est en 26-connexité. C'est-à-dire que d'une cellule, on a que 26 choix de déplacement possible.

Par contre, notre méthode possède un défaut. Notamment le fait que les contraintes cinématiques du robot ne sont pas prises en compte (accélération, vitesse, vitesse angulaire maximale etc.).

Bibliographie

[1] **ANSI/RIA Robotic Industries Association.***National Robot Safetystandard, 15Juin1999*

[2] **Mark W. Spong, Seth Hutchinson, and M. Vidyasagar.** *Robot Dynamics and control, Second Edition 28Janvier2004*

[3] **Blog TPE Consacré à L'intelligence artificielle.***La robotique, les prémisses de l'IA,*24 février 2011

[4] **WIKIPEDIA L'encyclopédie libre**. *Réseau de neurones artificiels.*

[5] **Marc Parizeau.***Réseaux de Neurones (Le perceptron multicouche et son algorithme de retropropagation des erreurs), Université Laval, Laval, 2004, 272 p*

[6] **Mouret, J.-B. And doncieux.S.** *Evolvingmodular neural-networks throughexaptation , IEEE Congress on Evolutionary Computation, 2009 (CEC 2009).*

[7] **Claude Touzet.** *Les réseaux de neurones : Introduction connexionnisme, EC2, 1992, 160 p.*

[8] **SaidAoughellanet, TayebMohammedi, Youcef BouterfaDepartement of Electronic Engineering University of Batna.** *Neural network path planning applied to PUMA560 robot arm,* Proceedings of the 6th WSEAS Int. Conf. on NEURAL NETWORKS, Lisbon, Portugal, June 16-18, 2005 (pp211-215*)*

[9] **Ferdinand PIETTE.***Algorithmes de planification de trajectoires : bref état de l'artRobotique,08 mai 2011*

[10] **Latombe 1991.** *Robot Motion Planning (Livre)*

[11] **Choset 1996.***SensorBased Motion Planning : the HierarchicalGeneralizedVoronoi Graph(Thèse de doctorat, California Institute of Technology)*

[12] **Chazzzelle et Guibas 1989**.*Visibiliy and ontersectionproblems in plane geometry(Article, Discrete and ComputationalGeometry)*

[13] **Shin et McKay 1986.** *A dynamicprogrammingapproach to trajectory planning of the roboticmanipulators, (Article, IEEE Transactions on Automatic Control)*

[14] **Hart 1968 .***A Formal Basis for the HeuristicDetermination of Minimum CostPaths (Article, IEEE Transactions on Systems Science and Cybernetics*)

[15] **Stentz et Anthony 1994.***Optimal and Efficient Path Planning for Partially-KnownEnvironments (Article, Proceedings of the International Conference on Robotics and Automation)*

[16] **Khatib 1986.** *Real-time obstacle avoidance for manipulators and mobile robots (Article; International Journal of RoboticsResearch)*

[17] **Fox et al. 1997**.*The dynamicwindowapproach to collision avoidance (Article, IEEE Robotics and Automation Magazine)*

[18] **Quinlan 1994.** *Real-Time PathModofication of Collision-Free Paths (Thèse de doctorat, Université de Stanford)*

[19] **Zadeh 1965**.*Fuzzy sets* (Article, Information and Control)

[20] **Defoort 2007.***Contributions à la planification et à la commande pour les robots mobiles cooperatifs (Thèse de doctorat, Ecole centrale de Lille)*

[21] **Defoort 2009.** *Motion planning for cooperativeunicycle-type mobile robots withlimitedsensing ranges: A distributedreceding horizon approach (Article, Robotics and AutonomousSystems)*

[22] **Lawrence, Zhou et Tits.***User's Guide for CFSQP Version 2.5: A C Code for*

Solving (Large Scale) ConstrainedNonlinear (Minimax) OptimizationProblems,GeneratingIteratesSatisfying All InequalityConstraints

[23] **Gaillard et al. 2010**.*DeterministicKinodynamic Planning (Workshop of the UK Planning and SchedulingSpecialInterest Group, Italy)*

[24] **P.Corke.** *A roboticstoolbox for MATLAB, IEEE Robotics and Automation Magazine,vol.3, pp.24-32, Sept. 1996.*