الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DJILLALI LIABES DE SIDI BEL ABBES
FACULTE DE TECHNOLOGIE
DÉPARTEMENT DE GENIE MECANIQUE

THESE

Pour l'obtention du diplôme de

Doctorat en Sciences

Spécialité : Génie Mécanique

Option : ROBOTIQUE

Présentée Par**: DAHMANE Sid Ahmed**

Analyse et Compensation des erreurs de Positionnement des Systèmes Robotiques

Soutenue le 25 Juillet /2019 devant la commission d'examen :

BELABBES Baghdad	Professeur	UDL-SBA	Président
MEGUENI Abdelkader	Professeur	UDL.SBA	Directeur de thèse
Melle FEKIRINI Hamida	Professeure	UDL-SBA	Examinateur
ZIADI Abdelkader	Professeur	C.U-Ain Témouchent	Examinateur
BOUZIANE Mohamed Mokhtar	M.C.A	U.Mascara	Examinateur
GHAZI Abdelkader	M.C.A	U.Mascara	Examinateur
AZZEDINE Abdelwahab	M.A.A	UDL.SBA	Invité d'Honneur

2018-2019

Remerciements

Je tiens à remercier tout d'abord Allah de m'avoir donné la foi, le courage et la confiance en moi-même pour pouvoir continuer mes études et attendre mon objective.

Je tiens à exprimer ma gratitude et mes profonds remerciements à mon directeur de thèse, Mr. Megueni Abdelkader qui a accepté de diriger cette thèse, pour son aide tout au long de ce travail.

Mes plus sincères remerciements sont adresser à Mr. AZZEDINE Abdelwahab, pour avoir Co-encadré cette thèse, pour ses conseils avisés, Je lui en suis très reconnaissant car non seulement il a m'a aidé pour concrétiser ce travail, mais aussi il était toujours là pour me soutenir et me motiver.

J'exprime, de même, mes remerciements à Mr. BELABBES Baghdad pour avoir accepté de présider le jury de cette thèse.

J'adresse ma reconnaissance à Mr.ZIADI Abdelkader, BOUZIANE Mohamed Mokhtar, ainsi à Melle FEKIRINI Hamida, qui m'ont fait l'honneur d'être membres de mon jury et ont accepté de consacrer du temps pour la lecture et le jugement de ce travail.

Je remercie Mr. SLIMANE Abdelkader et LOUSDAD Abdelkader pour ses hospitalités, ces conseils et ses contributions considérables pour la réalisation de ce travail,

Je ne peux pas faire abstraction de soutien de Mr. Nekrouf Attou Abdelkader, merci de m'avoir aidé lorsque j'avais besoin d'aide.

Merci à tous mes amis pour leur soutien, leur aide et leur amitié,Je remercie ma famille, ma femme et mais enfants pour son continuel support et son encouragement.

Résumé :

L'objectif principal de notre travail est d'optimiser l'erreur de position et d'orientation de la plate-forme d'un robot manipulateur parallèle plan à l'aide du modèle géométrique direct. Il est bien connu que le principal inconvénient des manipulateurs parallèles est l'existence de singularités au sein de son espace de travail, des méthodes intermédiaires ont étés utilisées pour déterminer la solution optimale. La première méthode est une méthode graphique qui détermine toutes les positions possibles de la plate-forme à partir d'intersections de cercles. La deuxième méthode étant polynomiale destinée a calculée les coordonnées du centre de gravité et l'orientation de la plate-forme. La solution adaptative neuro-floue est proposée dans cette étude constitue la troisième méthode. L'implémentation sous l'environnement Matlab de chacune de ces méthodes a permit l'obtention de résultat de simulation, ces résultat ont était sujet a un une étude comparative qui a montrer que la méthode polynômiale est la solution optimale.

Mots-clés : robot manipulateur parallèle plan, Erreur, Modèle géométrique direct, Singularités, ANFIS, Méthode polynomiale, Optimisation.

ملخص :

الهدف الرئيسي لعملنا هو تحسين وضعية وخطأ توجيه النظام الأساسي باستخدام الهندسة المباشرة لربوت مناور المتوازي المحور. من المعروف جيداً أن العيب الرئيسي في ربوت ذو المحور المتوازي هو وجود مفردات في مساحة العمل الخاصة به، ويقترح الحل المتكيف العصبي المتضائل في هذه الدراسة. تم استخدام طرق وسيطة لتحديد الحل الأمثل. الطريقة الأولى هي طريقة رسومية تحدد جميع المواضع المحتملة للمنصة وفقًا لتقاطع الدوائر. الطريقة الثانية هي طريقة متعددة الحدود المستخدمة حساب من هذه الأساليب العثور على جميع الحلول استنتاج من هذه ماتلاب إحداثيات مركز الثقل واتجاه المنصة. يمكن لمحاكاة البرمجة الأساليب. يظهر التحليل أن طريقة متعددة الحدود هي الطريقة التي توفر الحل الأمثل

المفتاح الكلمات: ربوت ذو المحور المتوازي، خطأ، نموذج هندسي مباشر، مفردات، أنفيس طريقة متعددة الحدود، التحسين

Abstract

The main objective of our work is to optimize the positional and orientation error of a platform using the direct geometric model of a parallel plane manipulator robot. It is well known that the main disadvantage of parallel manipulators is the existence of singularities within its workspace, the adaptive neuro-fuzzy solution is proposed in this study. Intermediate methods have been used to determine the optimal solution. The first method is a graphical method which determines all possible positions of the platform based on the intersection of circles. The second method is the polynomial method used to calculate the coordinates of the center of gravity and the orientation of the platform. Matlab programming simulation of these methods makes it possible to find all the solutions deduced from these methods. The analysis shows that the polynomial method is the one that provides the optimal solution.

Keywords Parallel plane manipulator, Error, Direct geometric model, Singularities, ANFIS, Polynomial method ·

Sommaire

Introduction

INTRODUCTION GENERALE

Les applications récentes de la robotique dans le domaine médical (opérations chirurgicales assistées par ordinateur, télécommande) exigent une précision dans l'exécution des manœuvres de plus en plus poussées particulièrement dans les interventions lourdes (neurochirurgie, chirurgie cardiaque etc.…). Les spécifications en matière de performances dans les applications robotiques deviennent de plus en plus sévères, et des robots de plus en plus précis sont requis. On doit –cependant- au préalable définir ce qu'on entend par précision pour les systèmes robotiques. Plusieurs aspects sont en effet à considérer :

-aspect matériel (montage, choix des matériaux)

- aspect métrologique (résolution des capteurs, précision de la mesure, tolérance des éléments constitutifs du robot).

- aspect de commande (stratégie de commande, qualité du correcteur)

Pour améliorer la précision, c'est à dire réduire les erreurs de pose, il s'avère nécessaire d'évaluer les « vraies » valeurs des paramètres géométriques du robot et de les insérer, par la suite, dans les équations du modèle géométrique inverse (MGI). Cette opération est appelée l'étalonnage géométrique (EG) des robots. Etant donné, qu'il n'est pas toujours possible d'effectuer des mesures directes pour obtenir les valeurs réelles des paramètres, les approches d'étalonnage proposées dans la littérature sont basées principalement sur des modèles d'optimisation. Les valeurs des paramètres identifiés par ces méthodes d'étalonnage ne correspondent pas nécessairement aux vraies valeurs des paramètres du robot. Elles représentent plutôt les valeurs qui permettent de satisfaire les fonctions objectives des modèles d'optimisation utilisés dans la procédure d'étalonnage. Ces fonctions consistent généralement à minimiser les erreurs résiduelles des poses ou des mouvements articulaires.

De nombreux travaux de recherche se sont penchés sur la correction de l'erreur sur la précision et la répétabilité.[1-2]

On citera également les travaux portant sur la correction en ligne des défauts de positionnement.[3]

Les défauts de chaines cinématiques d'un robot industriel sont pourtant, pour une grande part, responsables de la dégradation de la précision de positionnement en statique et en dynamique [4]

Les robots manipulateurs parallèles, en raison de leur boucle fermée possèdent un certain nombre d'avantages par rapport aux systèmes traditionnels (manipulateurs séries) tels que rigidité élevée, puissance élevée densité, hautes fréquences propres, vitesses et haute précision [5]. Cependant, ils ont aussi quelques inconvénients, comme un espace de travail relativement réduit, une géométrie directe relativement complexe ainsi que l'existence de singularités à l'intérieur de l'espace de travail. [6]. L'analyse du modèle géométrique du manipulateur parallèle est analysée suivant dans ses deux aspects : modèles direct et inverse. Le modèle géométrique inverse qui cartographie l'espace opérationnel dans l'espace articulaire, est aisé à résoudre contrairement au modèle géométrique direct qui fait surgir des difficultés dans sa résolution. En outre, l'existence de solutions multiples à la géométrie directe (solutions ou modes de travail) est un autre problème de l'analyse géométrique [7]. L'analyse de la singularité de manipulateurs parallèles planaires a été étudiée par de nombreux chercheurs [8, 9]. Les efforts pour résoudre la géométrie directe des manipulateurs parallèles planaires se sont concentrés sur le manipulateur 3RPR en raison de sa simplicité inhérente. Il est établi que la solution géométrique directe de ce type de manipulateur conduit à un polynôme [10, 11] de degré six. Son étude approfondie montre qu'on est conduit à un maximum de six solutions réelles. Une approche basée sur un réseau de neurones a été développée pour surmonter le problème de la résolution numérique de cette équation polynomiale. De nombreux efforts ont été déployés sur les applications de la technologie adaptative, système d'inférence neuro-floue (ANFIS) dans différents types de machines-outils parallèles en raison de leur extrême flexibilité et capacité d'approximation non linéaire de la fonction de correspondance sortie / entrée [12–13]. Plusieurs réseaux neurones peuvent résoudre le problème des solutions multiples de la géométrique directe. Cette approche surmonte également le problème des singularités et des incertitudes découlant de la planification de trajectoire. Dans cette approche, un réseau est formé en utilisant les données générées par le modèle géométrique inverse. Une analyse et une comparaison des trois méthodes utilisées sont conçues pour déterminer la solution optimale concernant l'erreur minimale avec une durée d'exécution minimale. L'analyse montre que la solution optimale est obtenue par la méthode polynomiale.

Chapitre I

Dans ce chapitre, nous commençons tout d'abord par présenter les robots parallèles en énonçant quelques définitions et donnons une liste non exhaustive des différentes architectures qu'on trouve dans la littérature. Nous introduisons des notions nécessaires pour l'analyse et la manipulation des robots parallèles. Puis nous présentons les méthodes de planification de trajectoire les plus courantes.

I. Généralités sur les robots

I.1.Introduction

Les robots parallèles sont une catégorie de robots présentant des chaînes cinématiques fermées. Le champ des architectures cinématiques possibles est très vaste, ainsi que le nombre de domaines d'applications qui s'étendent de la micromanipulation à la manipulation de charges élevées, en passant par des applications industrielles plus classiques telles que le *pick-and-place* à haute cadence (jusqu'à 3 ou 4 produits déplacés par seconde). Les performances des robots parallèles sont complémentaires et généralement à l'opposé de ceux des robots série. Ils ouvrent donc de nouvelles pistes pour la robotisation de certaines tâches. Ils représentent aujourd'hui une faible part de marché pour les robots industriels commercialisés, mais ils s'imposent naturellement pour certaines applications. Pour d'autres, ils sont en concurrence avec leurs homologues série. Une fois leur architecture cinématique choisie, leur étude demande de la méthodologie et une modélisation fine afin d'évaluer leurs performances. Cet aspect ne doit pas être négligé car les performances sont extrêmement sensibles à leur géométrie, en particulier à cause de la présence de nouveaux types de singularités par rapport aux robots série. Un robot parallèle est constitué d'un organe terminal à n degrés de liberté et d'une base fixe, reliés entre eux par au moins deux chaînes cinématiques indépendantes, la motorisation s'effectue par n actionneurs simples [13]. La première véritable invention d'un robot parallèle daterait de 1928 [14]. C'était un simulateur de mouvement destiné pour l'industrie de divertissement (Figure I.1).Ce robot n'a jamais été construit, à cause de la complexité de sa commande [15]. Dix ans plus tard, et peu après la naissance du terme « *robot* », le premier robot industriel destiné à la peinture automatisée a été breveté **[**16**].** La structure mécanique de ce robot représentée en Figure I.2, consiste en trois bras, très similaires au robot Delta. Dans le restant de cette thèse, nous décrirons d'abord brièvement les origines des robots parallèles. Ensuite, nous présenterons plusieurs exemples de transferts technologiques bien réussis. Enfin, nous essayerons de comprendre la bonne recette de la commercialisation.

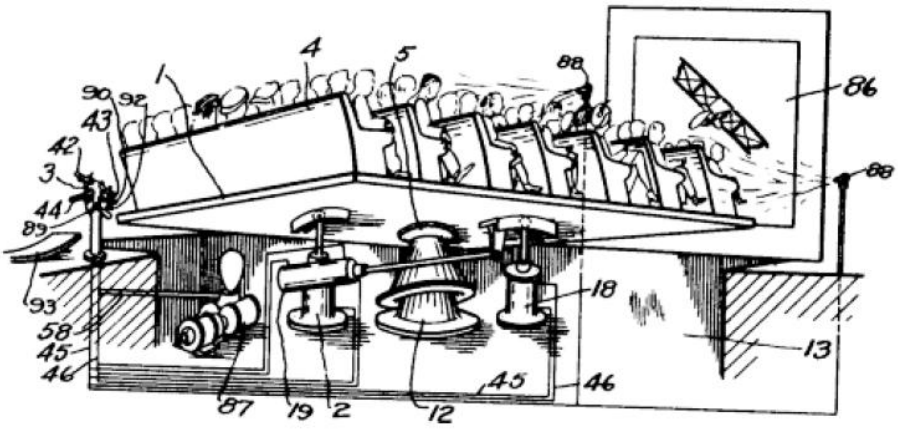


Figure I.1 : le mécanisme sphérique proposé en 1928 par J.E. Gwinnett

Les premiers manipulateurs parallèles construits sont un simulateur de vol [17] et une machine pour tester les pneus [18]. En 1947, Gough a établi les principes de base d'un mécanisme (Figure I.3) permettant de positionner et d'orienter une plate-forme mobile dans le but de tester l'usure de pneumatiques, machine dont il a construit un prototype en 1955 **[**18**]**.

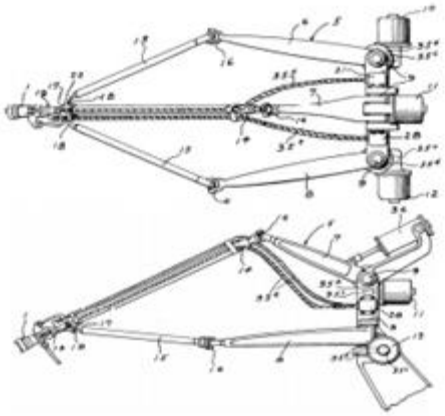


Figure I.2: Mécanisme de Pollard (1942)

Figure I.3 : Plate-forme de Gough-Stewart

La première machine-outil à architecture cinématique parallèle (Parallel Kinematic Machine) date de 1994 avec la Variax de Giddings & Lewis (Figure I..4). C'est une fraiseuse qui possède une structure parallèle du même type que celle de la plate-forme de Gough. Nous appellerons les machines de ce type « Hexapodes ». D'après son constructeur, outre le fait que la machine possède 6 degrés de liberté, elle serait 5 fois plus rigide qu'une machine classique et aurait des vitesses d'avance bien supérieures.

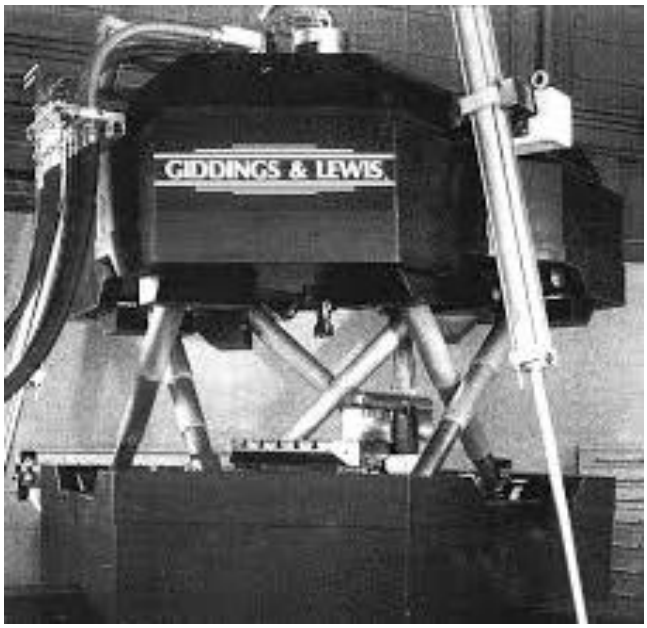


Figure I.4 : Machine Variax

Outre l'hexapode, une autre architecture célèbre est celle du robot Delta [19].C'est au début des années 1980 que ce robot a été inventé par Clavel. Ce robot consiste en trois parallélogrammes liant la base à la nacelle (la plate-forme mobile) qui ne se déplace que selon trois translations (Figure I.5). Plusieurs sociétés ont construit ce robot et plus de 4000 robots Delta ont été vendus [20]. Depuis plusieurs manipulateurs ayant une architecture proche du Delta ont été construites et commercialisées comme par exemple le robot Par4. Ce robot destiné à la manutention a été conçu récemment par l'équipe de François Pierrot du LIRMM qui s'est associé à Fatronik, un centre de recherche appliquée du Pays Basque. Ce robot est basé sur le principe des parallélogrammes, et est caractérisé par l'utilisation d'une nacelle articulée et de quatre pattes identiques à celles du robot Delta. Baptisé Quattro, ce robot est destiné à toutes les applications de manutention (Figure I.6). Aujourd'hui, on trouve de nombreuses utilisations des mécanismes parallèles dans l'industrie : pick-and-place, simulateurs de vol, machines-outils, micro-positionnement, etc. Dans la pratique, la majorité des applications commerciales reposent sur deux architectures célèbres : la plate-forme de Gough-Stewart et le robot Delta. Enfin, il est important de mentionner que l'état de l'art sur la présentation et l'étude des manipulateurs parallèles est un sujet vaste qui nécessite plus qu'un chapitre. Ce sujet est bien approfondi dans [13, 20 et 21].

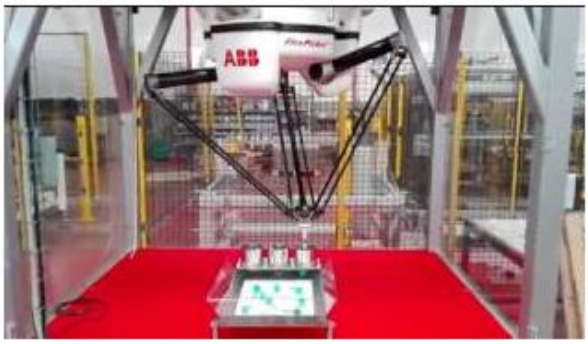


Figure I.5 : Le FlexPicker (ABB) **Figure I.6 :** Le Quattro (Adept Technology)

I.2. CLASSIFICATION DES ARCHITECTURES PARALLELES

Parmi les multiples configurations possibles des architectures des machines à architecture cinématique parallèle (PKMs), une première distinction entre architectures pleinement parallèles, hybrides et redondantes doit être faite.

I.2.1. MANIPULATEURS PLEINEMENT PARALLELES

Robots parallèles à six degrés de liberté

Plusieurs formes d'architectures parallèles à six degré de liberté sont possibles et la plus répandue est la plate-forme de Gough-Stewart (Figure I.7). Il s'agit d'un manipulateur à six degrés de liberté, dont la plate-forme est déplacée par l'élongation de six actionneurs linéaires. Son application la plus connue est le simulateur de vol (Figure I.8) en raison, entre autres, de la masse élevée de la cabine (jusqu'à 15 000 kg) à laquelle on doit faire subir des mouvements. Les simulateurs de vol sont utilisés pour l'entraînement des pilotes au sol.

.

Figure I.7: Plate-forme de Gough **Figure I.8 :** Simulateur de vol **Figure I.9 :** Hexapode M850-11

Les hexapodes sont aussi utilisés afin de positionner et orienter précisément des objets de masse importante. Dans ce type d'applications, la dynamique de la structure n'est pas un critère de première importance, ce qui importe, c'est de donner une position précise à un objet pouvant être massique. Dans l'exemple de la Figure I.9, l'hexapode présenté permet de positionner une charge de 200 *kg* suivant l'axe *z* avec une répétabilité de ±1µm dans un volume de travail de 100×100×50 mm^3.

I.2.2. MANIPULATEURS HYBRIDES

La définition exacte d'un mécanisme hybride est évasive. Dans cette classification nous considérerons comme hybride un mécanisme qui comporte plusieurs chaînes cinématiques reliant le bâti à l'organe terminal et dont une des chaînes au moins comporte plus d'un actionneur. Parmi les machines hybrides que l'on a cataloguées, on peut distinguer trois catégories :

- les mécanismes à porteur parallèle et poignet série,
- les mécanismes à porteur série et poignet parallèle,
- les autres mécanismes hybrides.

I.2.2.1 Porteur parallèle – poignet série

Le robot Tricept de Neos (Figure I.10) est une machine hybride : un mécanisme porteur d'architecture parallèle porte un poignet série. Une particularité de ce robot est la présence d'une patte passive (non-actionnée) qui contraint la cinématique du robot.

Ce robot a connu un réel succès commercial mais il semble que dans le domaine de la machine-outil les performances obtenues soient en deçà de celles espérées.

Figure I.10: Tricept 845 (Neos Robotics)

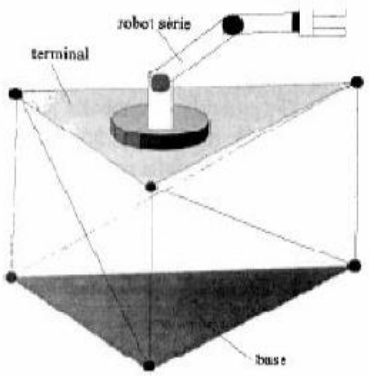


Figure I.11: Robot hybride

Un autre exemple de ce type est le robot de chen [22] (Figure I.11). L'avantage de ce type de robot est l'augmentation de l'espace de travail grâce à la partie série et l'augmentation de la précision grâce à la structure parallèle.

Une autre architecture intéressante est celle de la famille de machines Tricept proposée par Neos, ce robot à 3 ddl est composé d'une partie parallèle comportant trois jambes de type RRP**S,** une 4ième jambe passive de type RRP (qui contraint l'organe terminal à un mouvement selon une sphère a rayon variable), et d'une partie série qui est un poignet de type RRR (Figure I.12).

I.2.2.2.Porteur série – poignet parallèle

La machine-outil Sprint Z3 de DS Technologies possède un mécanisme porteur *x-y* et un poignet d'architecture parallèle *z-A-B*. Destinée à l'industrie de l'aéronautique, cette machine comprend un portique d'architecture classique d'une course de 60 mètre, et une tête d'architecture parallèle (Figure I.13).

Figure I.12 : Le robot Tricept

Figure I.13 : Sprint Z3

- **Le robot hybride Dumbo**

La machine-outil Dumbo de l'IFW à l'université de Hanovre est un mécanisme hybride 3T-2R (Figure I.14). Le porteur constitue un mécanisme hybride à lui tout seul puisqu'une colonne d'axe *z* supporte un mécanisme parallèle constitué de deux vérins et d'une patte passive. Le poignet d'architecture série confère à la machine des débattements angulaires importants.

Figure I.14 : Dumbo (IFW)

I.2.2.3 .Autres mécanismes hybrides

Un exemple des robots série-parallèles, est le robot de l'INRIA [23] constitué d'un robot parallèle à six degrés de liberté appelé le poignet actif monté sur un robot SCARA (Figure I.15). Un autre robot utilisant le même concept à été décrit dans [24] (Figure I.16), ce robot constitue un macro/mini manipulateur où le robot DELTA est utilisé comme poignet d'un robot SCARA, l'avantage de cette configuration est de dépasser la limitation de l'espace de travail. Ce concept de combinaison série-parallèle a aussi été utilisé dans le robot ARTISAN [25] avec une structure parallèle type 3RPS.

Figure I.15: Le SCARA/Poignet actif

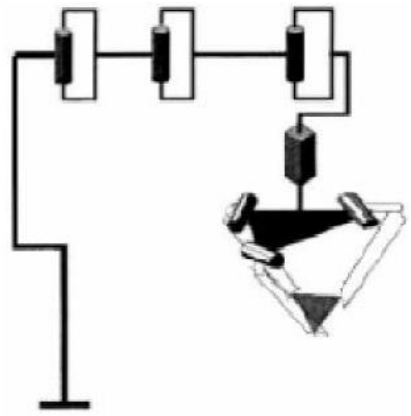


Figure I.16: Le SCARA/DELTA

Parmi les réalisations de ce genre de structures, on trouve le robot hybride de Zhang [26] qui est constitué de deux manipulateurs parallèles mis en série (Figure I.17), chaque manipulateur a 3 degrés de liberté, la plate-forme inférieure et supérieure contrôlent respectivement la position et l'orientation de l'effecteur. Ce type de manipulateur permet de découpler la position et l'orientation de l'organe terminal. On trouve une autre structure similaire dans [27] (Figure I.18).

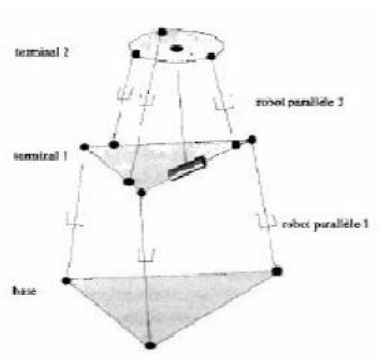


Figure I.17 : Un robot hybride découplé parallèles / série

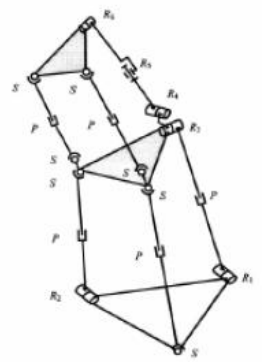


Figure I.18 : Robot hybride à 6 dl

Il existe d'autre robot hybride: comme celui de la société de Lagabex (Figure I.20) [28], qui se compose de quatre modules identiques (quatre plate-forme de Stewart), chaque module possède six vérins électriques identiques. Ces derniers sont actionnés par des moteurs à courant continu. La vitesse de sortie de la tige de vérin étant de 6 mm/s.

Figure I.19 : Robot Logabex- modèle LX4

I.2.3. MANIPULATEURS REDONDANTS

Il existe trois sortes de redondance pour les mécanismes parallèles [29] :

- La redondance d'actionnement correspond aux manipulateurs qui possèdent plus de moteurs que de degrés de liberté. Les moteurs sont agencés de manière à pouvoir générer des efforts internes au mécanisme, et ne produisent pas d'effet sur l'organe terminal, comme pour le robot Archi représenté sur la Figure I.20. Ce robot sur actionné possède quatre moteurs pour trois degrés de liberté. Ce concept de redondance est utile pour éliminer les positions singulières.

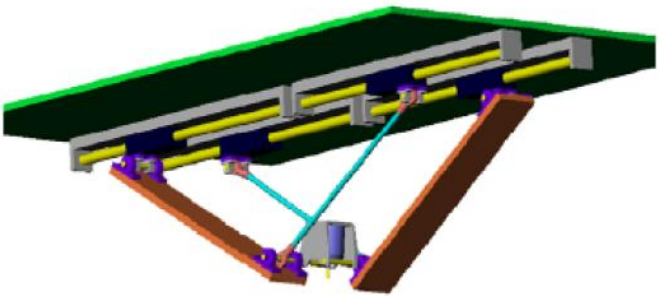


Figure I.20 : Le robot Archi

- La redondance cinématique signifie qu'à une vitesse donnée de la plate-forme mobile correspond une infinité de vitesses pour les moteurs. Ainsi au moins une des jambes du robot est un générateur de mouvement avec un nombre de degré de libertés plus grand que nécessaire, comme pour le robot double Tripod de Merkle représenté sur la Figure I. 21.

Chacune de ces deux jambes possèdent trois actionneurs déplaçant les points P_1 et P_2 dans l'espace et contrôlant ainsi la rotation de l'organe terminal autour de son axe. Ce concept de redondance peut être utilisé pour augmenter l'espace de travail et éviter le passage par des singularités [13].

- La redondance de mesure est obtenue lorsque le nombre des capteurs est plus grand que les nombres des liaisons actionnées. Ce concept de redondance joue un rôle important dans la résolution du modèle géométrique direct, la réduction de l'erreur de positionnement et le calibrage des robots.

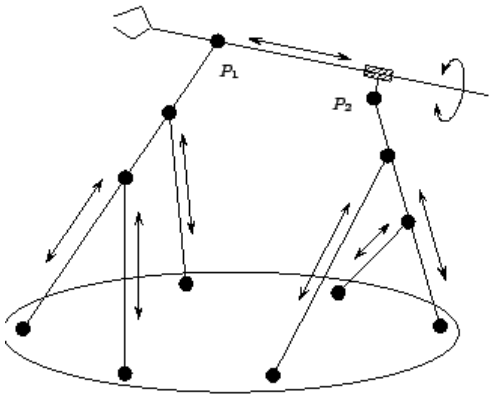


Figure I. 21: Le robot double Tripod de Merkle [13]

I.3. LES DIFFERENTS TYPES DE MOUVEMENT DES MANIPULATEURS PARALLELES

Un grand nombre de manipulateurs d'architecture parallèle peut être obtenu en faisant varier le nombre et la topologie des chaînes cinématiques du manipulateur. Nous distinguons dans cette thèse les manipulateurs plans et les manipulateurs sphériques fonctionnant dans un espace à trois ddls et les manipulateurs spatiaux fonctionnant dans un espace à six ddls [30].

I.3.1. Manipulateurs plans

Nous ne considérons dans cette thèse que le cas des robots plans pleinement parallèles. Comme il y a autant de degré de libertés (ddl) de l'organe terminal que de jambes dans un robot pleinement parallèle et comme il n'y a qu'un seul actionneur par jambe, les robots pleinement parallèles plans sont soit à deux ddl soit à trois ddl.

I.3.1.1. Robots plans à deux degré de libertés

Dans le cas où l'on ne considère que les robots constitués d'articulations de types rotoïdes et prismatiques, McCloy [31] a démontré qu'il pouvait y avoir vingt architectures différentes. Ce nombre se réduit à six si l'on suppose que les actionneurs sont attachés au sol, qu'il n'y a pas d'articulation prismatique passive et qu'aucun actionneur ne supporte le poids d'un autre actionneur. Ces architectures sont montrées sur la figure I.22. Les articulations rotoïdes actives sont représentées par des cercles blancs à l'intérieur tandis que les articulations rotoïdes passives sont représentées par des cercles noirs pleins.

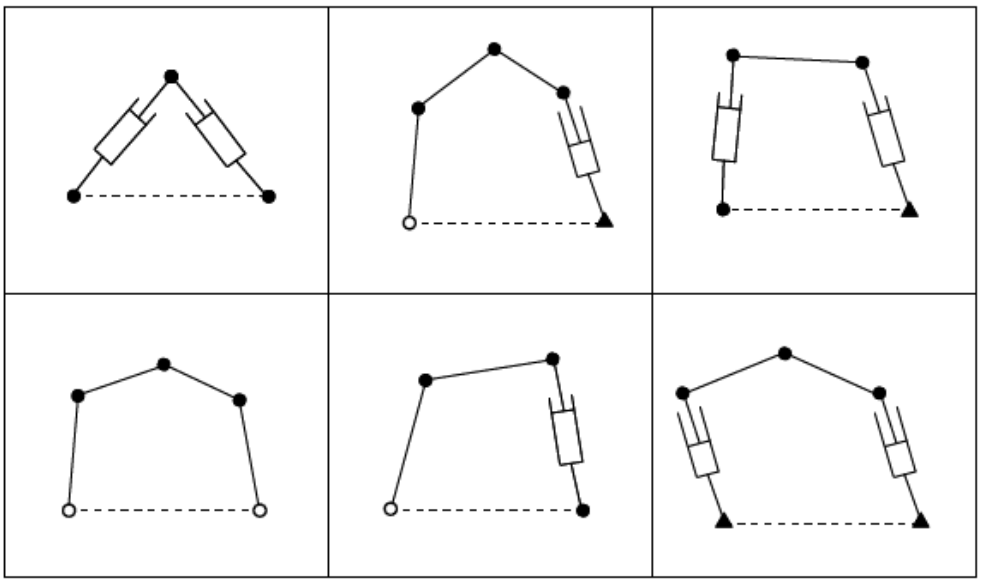


Figure I.22 : Robots plans à 2 ddl

I.3.1.2 Robots plans à 3 ddl

Hayes et Husty [32] ont étudié les plateformes généralisées planes à trois chaînes. Ces robots sont composés d'une plateforme mobile reliée par trois chaînes cinématiques. Chaque chaîne contient trois articulations qui peuvent être, soit de type rotoïde (R), soit de type prismatique (P). Ainsi, les types de chaînes possibles sont les séquences suivantes : RRR, RPR, RRP, RPP, PRR, PPR, PRP. Les chaînes de type PPP sont à exclure puisqu'elles imposent une orientation particulière à la plateforme mobile. En d'autres termes, aucune combinaison de translations dans le plan ne permet d'avoir un changement de l'orientation.

Il y a alors sept types de chaînes utilisés. Dans chaque chaîne il n'y a qu'une seule liaison actionnée. Le tableau I.1 montre les 18 chaînes possibles. Les liaisons actionnées sont soulignées. Les chaînes passives sont de type RR, PR, RP, ou bien PP. Les chaînes passives de type PP sont à rejeter, puisqu'une plateforme ayant une chaîne pareille soit n'est pas contrôlable, soit elle est difficilement assemblable quand les liaisons actives sont spécifiées [33]. Parmi ces 18 chaînes, il y a 8 paires de chaînes symétriques. Ce qui amène certains auteurs à enlever les chaînes marquées ~ dans le tableau I.1, lors de l'étude des différentes architectures possibles [34]. Ceci rend le nombre de chaînes à étudier égal à 10. Si on considère que le sens de montage des chaînes entre la base et l'organe terminal est important nous considérons les 18 chaînes pour calculer le nombre total de plateformes possibles. Ce nombre correspond à la somme du cas où toutes les chaînes sont identiques plus le cas où il y a deux chaînes identiques plus le cas où il y a trois chaînes différentes :

$$C_{18}^{1} + 2 \times C_{18}^{2} + C_{18}^{3} = 1140 \quad \textbf{(I.1)}$$

Table I.1 : Structures possibles des jambes d'un robot plan à 3 ddl

Chaîne passive RR	**Chaîne passive PR**	**Chaîne passive RP**
RRR	RPR	RRP
RRR	PRR	RRP~
RRR~	PRR~	RPR~
PRR	PPR	PRP
RPR	PPR~	RPP
RRP~	PRP~	RPP~

On montre dans la figure I.23, deux exemples de robots plans à trois ddl ayant trois chaînes identiques.

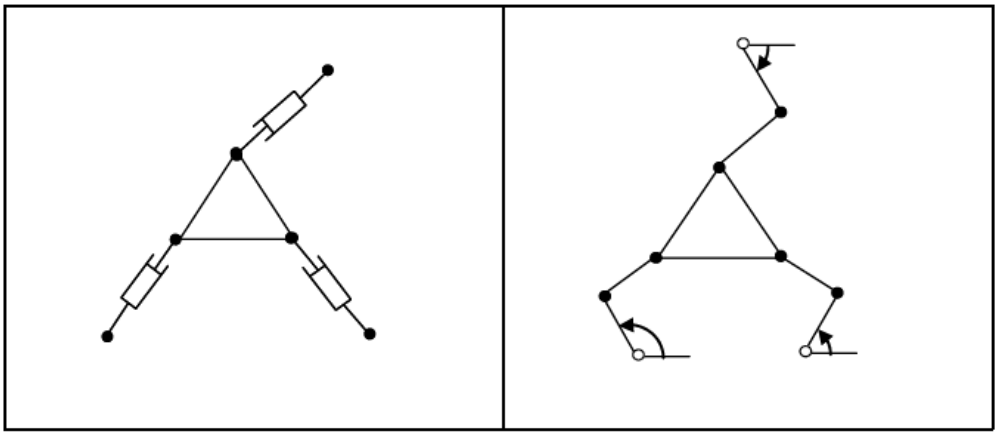


Figure I.23 : Exemples de robots plans à 3 ddl le 3-RPR et le 3-RRR

I.3.2. Manipulateurs sphériques

Les manipulateurs sphériques permettent trois rotations autour d'un point fixe comme par exemple le module parallèle du robot anguille et le manipulateur Agile Eye développé par l'Université de Laval (Figure I.24). Il existe de nombreuses autres façons d'obtenir un manipulateur sphérique, comme par exemple en utilisant une jambe passive contraignant le mouvement [13]. Le robot anguille développé par l'IRCCyN est constitué de 12 modules parallèles, chaque module a une structure sphérique avec trois ddls en orientation [35]; la Figure I.24 montre deux modules de la structure.

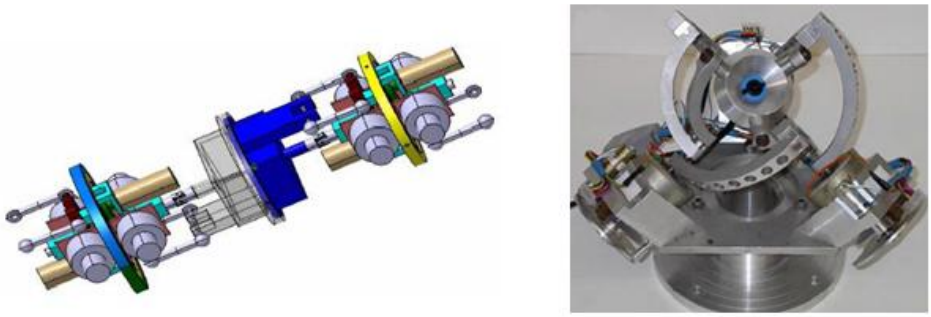


Figure 1.24 : Manipulateurs sphériques (à gauche) deux modules parallèles du robot Anguille et (à droite) le robot Agile Eye

I.3.3. Manipulateurs spatiaux

Les manipulateurs spatiaux sont constitués d'une plate forme et d'une base qui sont reliées entre elles par des chaînes cinématiques transmettant de trois à six degrés de liberté à la plateforme. Nous pouvons distinguer les manipulateurs à trois ddls de translation, les manipulateurs à ddls complexes, les manipulateurs couplant l'orientation à la translation et les manipulateurs découplés.

I.3.3.1. Manipulateurs à trois ddls de translation

Un exemple de manipulateur à trois ddls de translation est le manipulateur 3-UPU proposé par Tsai (Figure I.25). Ce manipulateur est composé d'une plateforme mobile et d'une base fixe, les deux sont reliées par trois jambes identiques composées chacune d'une articulation cardan (U), d'une articulation prismatique (P) motorisée et d'une autre articulation cardan (U) attachée à la plate-forme. Les axes des liaisons cardan liées à la base sont parallèles à ceux liées à la plate-forme de la même chaîne.

Figure 1.25 : le manipulateur 3-UPU à 3 ddls de translations

I.3.3.2. Manipulateurs à ddl complexes

Les manipulateurs à degrés de liberté complexes possèdent des degrés de liberté en orientation et en translation dépendant les uns des autres. Un exemple de ce type de manipulateurs est le robot 3-RPS proposé par Hunt [**36**]. Les paramètres d'orientation et de position de ce robot sont au nombre de six, cependant seulement trois sont indépendants (Figure I.26).

Figure I.26 : le robot 3 RPS

I.4. Les machines à architecture parallèles

Les machines à architecture cinématique parallèle (en anglais Parallel Kinematic Machines, PKMs) sont souvent reconnues pour leur grande rigidité structurelle, leur meilleure rapport charge utile / poids, leurs performances dynamiques élevées et leur meilleure précision [13, 37 et 38]. Ainsi, ils sont prudemment considérés comme des architectures alternatives attirantes pour des tâches exigeantes telles que l'usinage à grande vitesse [39]. La plupart des PKMs existantes peuvent être classées en deux familles principales : Les PKMs de la première famille sont munies de vérins reliant la base à la plate-forme mobile tandis que les PKMs de la deuxième famille se caractérisent par des jambes de longueur fixe qui glissent sur des rails [40,41]. Dans la première famille, on distingue généralement les PKMs à six degrés de libertés appelées Hexapodes et les PKMs à trois ddls appelées Tripodes [42**,**43].Beaucoup de prototypes et de PKMs de type Hexapode existent déjà, comme la machine Hexabot de la société Hexel. Cette machine qui sert de support mobile de pièce est présentée dans la Figure I.27. Nous pouvons aussi trouver des architectures hybrides telles que la machine Tricept de Neos Robotics [44]. Cette PKM possède un poignet sériel à deux rotations monté sur un tripode à cinématique parallèle à jambes de longueur variable, avec une jambe passive (Figure I.28).

Figure I.27 : La machine Hexabot

Figure I.28 : Le Tricept de Neos Robotics

Dans la deuxième famille, nous trouvons l'Hexaglide de l'ETH de Zürich dont les liaisons prismatiques sont toutes coplanaires et parallèles (Figure I.29). L'avantage de cette architecture réside dans le fait que les moteurs sont fixes, ce qui diminue les inerties et permet l'emploi des moteurs linéaires. L'HexaM proposé par Toyoda [45] est un autre exemple avec trois paires de liaisons prismatiques parallèles montées sur un cône vertical (Figure I.30).

Figure I.29 : Machine Hexaglide

Figure I.30 : Machine HexaM

Beaucoup de PKMs à trois axes de translation appartiennent à cette deuxième famille et reprennent une architecture proche du robot Delta linéaire [46]. L'UraneSX de Renault Automation (Figure I.31) et le Quickstep de Krause and Mauser (Figure I.32) possèdent trois liaisons prismatiques parallèles et non-coplanaires [47]. Une PKM hybride avec trois liaisons linéaires inclinées et un poignet de deux axes est le GeorgV de IFW (Figure I.33).

Figure I.31 : UraneSX

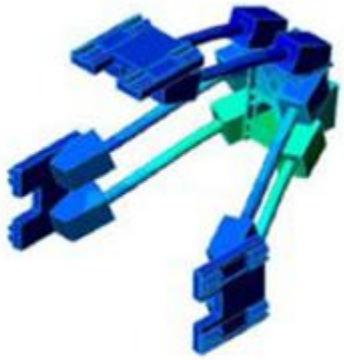


Figure I.32 : Quickstep

Une autre PKM hybride est L'Orthoglide 5-axes présenté dans la Figure I.34. Ce robot utilise un porteur parallèle de type Delta linéaire sur lequel est monté un poignet parallèle sphérique à 2 ddls de type Agile Eye [40]. Le module parallèle assure des performances cinétostatiques homogènes dans l'espace de travail grâce à l'existence d'une configuration isotrope tandis que le poignet permet un grand espace de rotation. Ses applications principales sont l'usinage 5 axes, le soudage laser et le prototypage rapide **[**48**]**. Dans une configuration isotrope, la vitesse et la rigidité statique de l'outil sont égales dans toutes les directions [49**,**50].

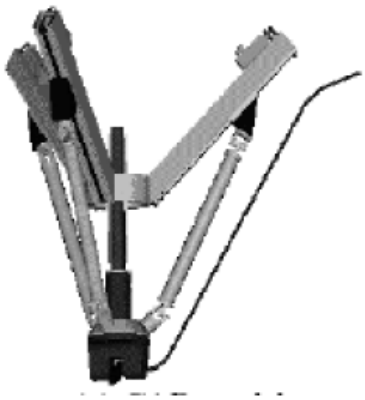


Figure I.33 : le robot GeorgV développé par IFW

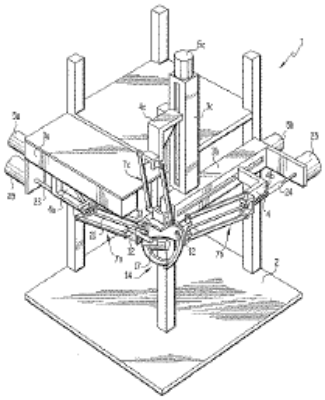


Figure I.34 : L'Orthoglide 5 axes

Plus récemment, plusieurs équipes de recherche ont produit une famille de manipulateurs à trois degrés de liberté, isotropes dans tout l'espace de travail [51**,** 52**,** 53,54]. L'isotropie est liée au conditionnement de la matrice jacobienne. Ce conditionnement est défini comme étant le rapport entre la plus grande et la plus petite valeur propre de la matrice jacobienne. Son domaine de variation est entre 1 et l'infini. Lorsqu'un mécanisme est en configuration singulière, le conditionnement tend vers l'infini. En configuration isotrope, il est égal à un.

Ainsi, le conditionnement mesure l'uniformité de la distribution des vitesses et des efforts autour d'une configuration de l'outil mais ne permet pas d'en mesurer l'importance

I.5. COMPARAISON DES PERFORMANCES DES ROBOTS SERIE ET DES PKMS

Les principaux critères de comparaison sont le volume de travail, le rapport masse transportable / masse du robot, la précision et le comportement dynamique:

- Volume de travail : Le principal défaut d'une machine à architecture cinématique parallèle (PKM) est que son domaine accessible est relativement restreint : il est dû à l'architecture parallèle, qui fait que les jambes d'une PKM se retiennent, en quelque sorte, les unes les autres, empêchant ainsi la plateforme mobile d'accéder à un grand volume de travail par rapport à son encombrement [49]. L'empreinte au sol est un autre critère prépondérant pour une PKM destinée à l'industrialisation [55]. Il dépend en partie de l'orientation spatiale du mécanisme sur lequel elle est basée. Cette orientation est choisie telle que la plate-forme mobile soit accessible par l'opérateur.

- Rapport masse transportable / masse robot : Dans l'architecture série classique, l'effecteur et l'objet manipulé se situent à l'extrémité de la chaîne mécanique articulée. Chaque actionneur doit avoir la puissance nécessaire pour mettre en mouvement non seulement l'objet, mais aussi les corps et actionneurs aval. Chaque segment du robot doit donc être dimensionné en conséquence, ce qui conduit à un faible rapport masse transportable / masse robot. Les actionneurs des PKMs se situent le plus souvent au voisinage des points d'articulation de la base, si ce n'est sur la base elle-même. Dans tous les cas, ces actionneurs agissent sans mettre en œuvre de transmissions complexes. Les segments peuvent être considérablement allégés.

- Précision et répétabilité : L'architecture même des robots de type série pose le problème de l'accumulation des erreurs, une erreur de positionnement sur chaque articulation ayant une répercussion sur la position de l'effecteur, répercussion d'autant plus importante que l'articulation est proche de la base. La mise en série des articulations implique aussi la mise en série de leurs défauts (jeux, frottements, flexion des corps...). Les PKMs ne présentent pas cet inconvénient et leur morphologie leur assure une rigidité remarquable même avec des structures mobiles très allégées.

- Comportement dynamique : Compte tenu de leur faible rapport masse transportable / masse robot et de leur structure massive, les robots de type série ont des performances dynamiques limitées. Avec les PKMs, le fait que les actionneurs soient ramenés au voisinage des points d'articulation de la base donne des performances dynamiques appréciables. Par ailleurs, le fait de diminuer l'inertie du robot contribue à diminuer les effets de couplage dynamique entre les articulations, la commande pouvant éventuellement se contenter d'un modèle dynamique

simplifié. Mentionnons enfin que le mode de fonctionnement des robots série les rend peu aptes à fonctionner à des échelles de taille faibles [56]. En effet leur principe moteur est la force inertielle qui doit dominer les forces de friction. Or cette force inertielle varie avec le carré des longueurs alors que les forces de friction sont beaucoup moins sensibles aux dimensions. Ainsi pour réaliser un micro-robot série, il ne suffit pas de diviser les dimensions d'un robot série fonctionnant convenablement par le facteur de taille approprié, puisqu'à partir d'une certaine échelle les forces de friction deviendront prépondérantes. Le Tableau 1.1 synthétise les principales différences entre les manipulateurs parallèles et leurs homologues séries, ce qui revient à comparer leurs avantages et inconvénients.

Manipulateur sériel	**Robot parallèle**
- Succession de segments en série de la base vers l'effecteur	- Tout les segments sont au contact de la base et de l'effecteur
- Chaîne cinématique ouverte	- Chaîne cinématique fermée
- Important espace de travail	- Espace de travail restreint
- Faible précision	- Grande précision
- Faible charge transportable	- Lourde charge transportable
- Faible rigidité	- Très rigide

Tableau I.2 : Comparatif entre Les robots séries et les PKMs

I.6. NOTIONS IMPORTANTES EN ROBOTIQUE PARALLELE

Des notions sur l'espace de travail, les modèles géométriques direct et inverse, et les singularités sont rappelées dans les parties suivantes. Il existe sur ce sujet plusieurs livres de référence dont : [13], [30], [21].

I.6.1. MODELISATION GEOMETRIQUE

La conception et la commande des robots nécessitent le calcul de certains modèles mathématiques. Nous distinguons parmi ces modèles : les modèles géométriques direct et inverse (MGD et MGI), qui expriment la situation de l'organe terminal en fonction des variables articulaires du mécanisme et inversement, les modèles cinématiques direct et inverse (MCD et MCI), qui expriment la vitesse de l'organe terminal en fonction des vitesses articulaires et inversement, les modèles dynamiques définissant les équations du mouvement du robot, qui permettent d'établir les relations entre les couples ou forces exercés par les actionneurs et les positions, vitesses et accélérations des articulations.

I.6.1.1. Espace articulaire et espace opérationnelle

Pour un manipulateur à n degrés de liberté, l'ensemble des variables articulaires motorisées (les entrées) notées **q** définissent la configuration articulaire tandis que l'ensemble des coordonnées opérationnelles de position et d'orientation de l'effecteur (les sorties) notées **X** définissent la configuration de la plate-forme. On définit une relation entre les variables articulaires motorisées et les coordonnées opérationnelles. Cette relation est définie par un opérateur géométrique F tel que :

$$F(\mathbf{X},\mathbf{q})=0 \qquad (I.2)$$

- EA^n l'espace articulaire lié aux articulations motorisées **q** (n est le nombre d'articulations motorisées).
- EO^m l'espace opérationnel de dimension m lié à la position et à l'orientation de l'effecteur.

I.6.1.2. Modèle géométrique inverse

Le modèle géométrique inverse permet de trouver l'ensemble des configurations articulaires possibles pour une configuration donnée de la plateforme mobile. Parfois, il existe plusieurs solutions au modèle géométrique inverse. Ainsi, nous associons plusieurs postures pour la même configuration de la plate-forme mobile du manipulateur. Un changement de posture revient à changer de solution du modèle géométrique inverse La résolution du MGI ne pose généralement pas de problème. Pour calculer le MGI, nous écrivons un système d'équations non linéaires dont chaque équation est associée à une jambe du manipulateur. Chaque jambe est caractérisée par une origine A_i et une extrémité B_i. La configuration **X** de la plate-forme permet de définir la position des points extrêmes de chaque jambe. Nous pouvons ainsi écrire le MGI de chaque jambe : $A_iB_i = H(\mathbf{X})$. Ce modèle est parfois difficile à résoudre, notamment pour les manipulateurs spatiaux et lorsque la structure de leurs jambes est complexe, nous citons à titre d'exemple le cas de la machine Verne dont l'une des jambes est différente des deux autres.

I.6.1.3. Modèle géométrique direct

Le modèle géométrique direct (MGD) permet de trouver l'ensemble des configurations de la plate-forme mobile en fonction d'une configuration articulaire donnée. Il existe généralement plusieurs solutions au MGD. Il est ainsi possible d'associer plusieurs modes d'assemblage pour les mêmes configurations articulaires. Un changement de mode d'assemblage revient à changer de solution du modèle géométrique direct.

Dans la plupart des cas, l'obtention d'une solution au MGD correspond à la résolution d'un système d'équations non-linéaires ce qui est un problème complexe. La résolution du MGD est nécessaire pour compléter la boucle d'asservissement pour le système du contrôle du mouvement, c.-à-d., le MGD est nécessaire pour indiquer si le point requis dans l'espace de travail a été atteint ou pas. Les méthodes de résolution du modèle géométrique direct des manipulateurs parallèles diffèrent suivant l'architecture du manipulateur et le type de ses articulations. Nous distinguons quatre méthodes, les méthodes analytiques, les méthodes itératives, les méthodes reposant sur l'ajout de capteurs additionnels et les méthodes utilisant les réseaux de neurones. Il existe trois approches analytiques principales pour résoudre le MGD. La première approche consiste à formuler un système d'équations de contraintes non linéaires, puis à convertir ce système en un polynôme à une inconnue afin de le résoudre simultanément en utilisant une technique numérique comme la méthode de Newton-Raphson [57], [58]; la deuxième approche consiste à découpler la position et l'orientation, ce qui permet de réduire la complexité du problème et d'accélérer le processus d'obtention de la solution [59]; la troisième approche utilise les quaternions pour obtenir une solution analytique pour une certaine classe de manipulateurs parallèles de type Hexapode [60]. Ces approches analytiques sont limitées à des architectures spéciales de manipulateurs parallèles et ne peuvent pas être appliquées à toutes sortes de manipulateurs.Dans les approches itératives, le modèle géométrique est formulé afin qu'on puisse le résoudre en utilisant toutes les techniques numériques disponibles, comme par exemple, la méthode de Newton-Raphson [61,62]; cependant, ces techniques numériques sont coûteuses en temps de calcul et ne garantissent pas l'obtention une solution dans un temps borné.

Une autre approche consiste à installer des capteurs supplémentaires sur le manipulateur [63,64] dans le but d'obtenir plus d'information sur l'état du manipulateur. Ces capteurs nous permettent de résoudre facilement et rapidement le MGD mais leur coût supplémentaire limite l'intérêt de cette approche. Une quatrième approche utilise les réseaux de neurones inspirés de la structure du cerveau humain [65]. C'est un outil mathématique qui permet d'approximer les relations non –linéaires entre les vecteurs d'entrés et de sorties.

I.6.2. CONFIGURATIONS SINGULIERES

Les manipulateurs parallèles comportent des configurations dites singulières pour lesquelles le comportement du manipulateur se dégrade. Ces configurations peuvent se situer aussi bien à l'intérieur de l'espace de travail que sur ses frontières. Celles situées à l'intérieur de l'espace de travail sont les plus gênantes pour la génération de trajectoires puisque le manipulateur peut se bloquer dans de telles configurations. Les problèmes suivants peuvent aussi survenir au voisinage de ces configurations singulières :

- Une augmentation importante des efforts dans les articulations qui peut endommager la structure du manipulateur;
- Une perte de rigidité du manipulateur qui peut se traduire par une instabilité de son organe terminal lorsque les articulations motorisées sont bloquées.

Pour déterminer les configurations singulières des manipulateurs parallèles, deux approches existent. La première est une méthode analytique fondée sur l'étude des matrices jacobiennes du manipulateur. La seconde est une méthode géométrique utilisant des outils de géométrie tels que la théorie des vis réciproques (Reciprocal Screw Theory), la géométrie de Grassmann et l'algèbre de Grassmann-Cayley.

I.6.2.1. Approche analytique

L'approche analytique repose sur le travail de Gosselin et Angeles [**66**], qui consiste à l'étude de l'Eq. (I.3) obtenue en différentiant l'Eq. (I.3) par rapport au temps :

$$At + B\dot{q} = 0 \qquad \text{(I.3)}$$

Où $t = \left[\omega, \dot{c}\right]^T$ représente le torseur cinématique de la plate-forme du manipulateur (avec ω est le vecteur des vitesses de rotation et $\dot{c}$ le vecteur des vitesses de déplacement), $\dot{q}$ représente le vecteur des vitesses articulaires et **A** et **B** sont deux matrices jacobiennes. La matrice **A** est nommée matrice *jacobienne parallèle* et la matrice **B** matrice *jacobienne sérielle.* A partir de l'étude de ces deux matrices, Gosselin a défini trois types de configurations singulières **[66]**:

- Les singularités parallèles qui sont dues à la perte de rang de la matrice jacobienne parallèle **A** (lorsque det(**A**)= 0). Dans ce cas, l'effecteur peut bouger alors que les articulations motorisées sont bloquées. Le manipulateur gagne ainsi un ou plusieurs degré(s) de liberté;
- Les singularités sérielles qui sont dues à la perte du rang de la matrice jacobienne sérielle **B**. Dans ce cas, certains déplacements de l'effecteur ne peuvent pas être réalisés et le manipulateur perd un ou plusieurs degré(s) de liberté. Les singularités sérielles représentent aussi les limites de l'espace de travail du manipulateur;
- Les singularités parallèles/sérielles qui sont dues à la perte de rang simultanée de **A** et **B**. Dans ce cas, il est possible de déplacer de manière infinitésimale l'effecteur alors que les articulations motorisées sont bloquées et inversement; Il existe un autre type de singularités, les *singularités structurelles*, qui apparaissent pour des dimensions particulières des manipulateurs. Dans ce cas, pour des configurations articulaires particulières, le modèle géométrique direct admet une infinité de solutions c-à-d une infinité de configurations pour la

plateforme mobile. La Figure I.36 présente respectivement de gauche à droite des configurations singulières parallèle, sérielle et structurelle d'un manipulateur à 5 barres.

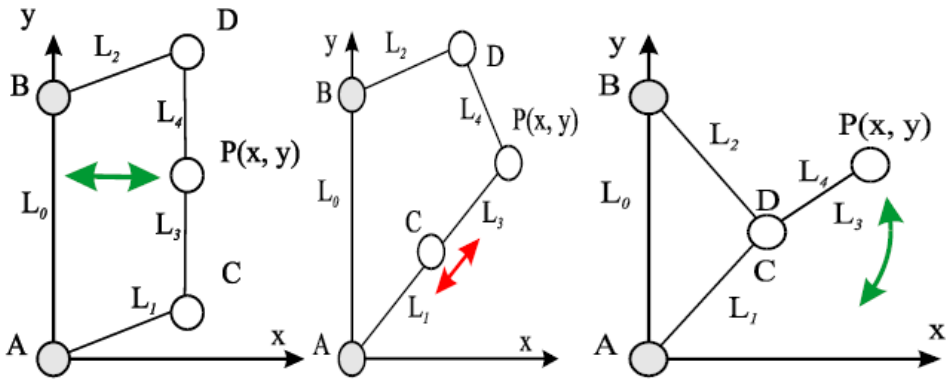


Figure I.35 : Mécanisme à 5 barres en configurations singulières parallèle, sérielle et structurelle respectivement de gauche à droite.

De nombreuses études utilisent les matrices jacobiennes cinématiques pour déterminer les singularités des manipulateurs parallèles. Parmi celles-ci, nous pouvons citer [67], [68] pour les manipulateurs plans et [69], [70], [71] pour les manipulateurs spatiaux.

Ce formalisme [66] est souvent utilisé aujourd'hui mais, contrairement à ce qui a longtemps été admis dans le passé, il ne s'applique pas à tous les robots parallèles. En effet, certaines singularités ne peuvent pas être détectées par ce formalisme (en particulier les singularités internes et les « singularités de contraintes » pour les robots à moins de six degrés de liberté). Ces singularités se produisent quand les jambes des manipulateurs à moins de six ddls sont incapables de contraindre la plateforme à exécuter le mouvement souhaité et en conséquence la plateforme gagne des degrés de liberté supplémentaires rendant le manipulateur incontrôlable. D'autre part, l'écriture symbolique des matrices jacobiennes peut être très difficile en raison de la complexité des expressions. Un important travail d'analyse des singularités a été réalisé par Zlatanov [72], où il a introduit la notion de *« Cspace singularity »* (singularité de l'espace des configurations) qui a permis par la suite de bien comprendre le phénomène de singularité de contrainte [73]. Plus récemment, S. Krut a proposé une méthode d'analyse des singularités adaptée aux robots à trois translations et une rotation, comme le robot H4. Cette méthode permet de recenser toutes les singularités de ce type de robot [74] mais son extension à d'autres architectures de robots n'est pas évidente. Ces travaux ont clairement mis en évidence les limitations du formalisme de Gosselin.

I.6.2.2. Approche géométrique

Les limites de l'approche analytique dans la détermination des singularités ont conduit les chercheurs à utiliser des approches géométriques pour résoudre ce problème. Ainsi d'une part la théorie de vis réciproques a été utilisée pour obtenir une matrice jacobienne de dimension 6×6 permettant de détecter toutes sortes de singularités, y compris les singularités de contrainte. D'autre part, des travaux ont été menés pour identifier géométriquement les singularités parallèles en se servant de la matrice jacobienne du robot. On distingue deux méthodes : la méthode basée sur la géométrie de Grassmann et celle basée sur l'algèbre de Grassmann-Cayley. La méthode géométrique la plus utilisée pour l'identification des singularités parallèles des manipulateurs est basée sur la géométrie de Grassmann. Cette approche fonctionne par inspection des sous-ensembles de droites, appelées variétés. Ces variétés sont associées aux différentes configurations singulières d'une structure consistant de barres liant deux corps. L'application de la géométrie de Grassmann fut commencée par le travail de Dandurand [75] en examinant la dépendance linéaire de droites. Ainsi, en représentant les jambes liant deux corps d'une structure spatiale par des droites, il a présenté une méthodologie qui permet de vérifier la rigidité de cette structure. Toutes les variétés de droites ont été spécifiées ainsi que les conditions géométriques de dégénérescence de ces variétés. Merlet a repris le travail de Dandurand et a introduit la géométrie de Grassmann dans l'étude des singularités des manipulateurs parallèles **[**76**]**. Les colonnes de la transposée de la matrice jacobienne inverse des manipulateurs de type Gough-Stewart sont des vecteurs de Plücker des droites finies portées par les jambes de ces manipulateurs. Ensuite d'autres chercheurs ont défini les conditions pour lesquelles les singularités peuvent être associées à la dépendance des droites finies dans l'espace [77], permettant ainsi d'appliquer la géométrie de Grassmann. Ces droites sont des torseurs statiques d'actionnement qui décrivent les forces instantanées appliquées par les actionneurs sur la plate-forme mobile. En conséquence une matrice jacobienne inverse formée de six vecteurs de Plücker dégénère lorsqu'un sous ensemble de n vecteurs de Plücker engendre une variété de dimension $m< n$. Les variétés dégénérées de six vecteurs de Plücker, $\mathbf{F_i}$ (i=1..6) sont présentées en détail dans [75, 76, 77]. La méthode basée sur l'algèbre de Grassmann-Cayley fonctionne d'une manière similaire à celle basée sur la géométrie de Grassmann mais en traduisant des conditions géométriques spécifiques aux classes des manipulateurs étudiés. Dans cette méthode, le déterminant d'une matrice jacobienne constituée des vecteurs de Plücker est décomposé en une expression algébrique invariante. Cette expression est simplifiée selon l'architecture de la classe des manipulateurs étudiés en utilisant les propriétés du déterminant. L'expression finale est factorisée en utilisant les opérateurs de l'algèbre de Grassmann-Cayley pour obtenir une

condition géométrique d'existence des singularités parallèles. Cette méthode présentée en détail dans le chapitre 4 est utilisée pour l'étude des singularités parallèles des manipulateurs à mobilités restreintes.

I.7. ESPACE DE TRAVAIL

L'espace de travail *W* définit l'ensemble des configurations accessibles de la plateforme mobile. Il se caractérise par l'ensemble des positions et orientations accessibles par un repère lié à la plate-forme mobile.

L'espace de travail est un outil précieux pour différentes applications en robotique, il est utilisé pour analyser les performances des robots manipulateurs, pour la conception optimale de manipulateurs, pour le choix de morphologie en fonction d'une tâche déterminée, pour la préparation hors ligne des trajectoires, etc…

I.7.1. Espace de travail et représentation

L'espace de travail dépend de la morphologie et des paramètres géométriques du manipulateur parallèle. Ainsi il peut avoir des formes très variées et en général complexes. Il peut être restreint par les singularités d'une part et les butées articulaires d'autre part. Pour un manipulateur parallèle à six degrés de liberté. La position et l'orientation du repère lié à la plateforme mobile sont généralement dépendantes. Ainsi, la représentation de l'espace de travail est un volume de dimension six pour lequel il n'existe pas d'illustration possible. On ne peut donc que représenter des sous ensembles de l'espace de travail.

I.7.1.1.Les différents type d'espace de travail

Les différents types d'espace de travail repris de [13] sont :

- Espace de travail à orientation constante ou espace de travail à translation est l'ensemble des positions du point de référence du manipulateur atteignable lorsque l'orientation de la plateforme est fixe;
- Espace de travail à orientation ou espace de travail à centre fixe est l'ensemble des rotations possibles autour du point de référence lorsque celui occupe une position fixe dans le repère absolu.
- Espace de travail maximal, défini comme l'ensemble des positions du point de référence qui peuvent être atteintes avec au moins une orientation de la plate-forme.
- Espace de travail pour un intervalle d'orientation, défini comme l'ensemble des positions du point de référence qui peuvent être atteintes avec au moins une orientation dans un intervalle

donné. L'espace maximal est un cas particulier de l'espace pour un intervalle d'orientation défini entre 0 et 2 π .

- Espace de travail total pour un intervalle d'orientation est l'ensemble des positions du point de référence qui peuvent être atteintes avec toutes les orientations dans un intervalle donné.
- L'espace de travail dextre est l'ensemble des positions du point de référence pour lesquelles toutes les orientations sont permises. L'espace de travail dextre est un cas particulier de l'espace total pour un intervalle d'orientation défini entre 0 et 2 π .
- L'espace de travail à orientation réduite est l'ensemble des positions du point de référence qui peuvent être atteintes pour un sous-ensemble des orientations définies dans un intervalle donné, et où les autres orientations peuvent avoir des valeurs arbitraires. Cet espace de travail est important pour les applications qui n'impliquent pas tous les degrés de liberté du robot, comme l'utilisation d'un robot à six ddls pour une application d'usinage 5-axes. Un autre type d'espace de travail s'avère intéressant surtout dans le domaine de l'usinage:
- L'espace de travail 3-axes est l'ensemble des positions du point de référence défini dans le repère de la pièce à usiner pour lesquelles l'outil est considéré toujours perpendiculaire à la pièce. Cet espace est utile pour montrer la capacité des machines d'usinage 5-axes à usiner les mêmes pièces que celles de leurs homologues 3-axes.

I.7.1.2.Méthodes utilisées pour le calcul de l'espace de travail

Plusieurs méthodes ont été utilisées pour le calcul de l'espace de travail des manipulateurs parallèles, citons :

- Les méthodes algébriques [78 ,79],
- les méthodes algébriques par discrétisation **[**80**]**,
- les méthodes géométriques [81 et 82].

I.7.1.2.1. **Méthodes Algébriques**

Les méthodes algébriques sont plus difficiles à appliquer car elles augmentent la dimension du problème en introduisant des variables supplémentaires. Elles consistent à résoudre un problème d'optimisation en introduisant des pénalités aux frontières [78].

I.7.1.2.2.Méthode Algébriques par discrétisation

Les méthodes algébriques par discrétisation utilisent les modèles géométriques direct et inverse pour calculer l'ensemble des configurations que le manipulateur peut atteindre. Ces données sont enregistrées dans une structure hiérarchique. Nous citons à titre d'exemple les quadtrees et les octrees utilisés dans **[**80**]** pour le calcul des domaines d'unicité des manipulateurs planaires et la méthode d'analyse par intervalles utilisée pour le calcul de l'espace de travail et la détermination des singularités des manipulateurs parallèles **[**83,48**]**.

Cette méthode est coûteuse numériquement et gourmande en place mémoire mais assez facile à programmer.

I.7.1.2.3.Méthodes Géométriques

Les méthodes géométriques permettent de calculer rapidement la frontière de l'espace de travail. Elles peuvent intégrer les contraintes liées aux limites articulaires et des collisions entre les segments [13]. Cependant, la reconstruction de l'espace de travail total est difficile. Pour certains cas comme pour le robot Delta, le calcul de l'espace de travail peut être fait directement en utilisant des logiciels de CAO (conception assisté par ordinateur) puisque l'espace de travail est équivalent à l'intersection entre des volumes simples.

I.8. CONCLUSION

Dans ce chapitre nous avons présenté un bref historique des robots à structures parallèles et avec un ordre chronologique de leur développement depuis le premier hexapode octaédrique de Gaugh-Stewart en 1947. Ensuite on a donné quelques définitions utiles à l'état de l'art de la robotique parallèle, tout en citant quelques exemples de robots parallèles classés selon le nombre de degrés de liberté. Différentes propriétés caractérisant ces manipulateurs et des notions importantes en robotique ont été évoquées puis suivies par la présentation de quelques exemples de robots hybrides. Par la suite, nous avons classifié les architectures parallèles et présenté les avantages respectifs des alternatives parallèles vis-à-vis de leurs homologues sériels. La modélisation géométrique, l'espace de travail et les singularités des manipulateurs parallèles closent ce chapitre.

Chapitre II

II.1.Introduction

De nos jours, de plus en plus d'industries ont recours aux robots industriels pour accomplir diverses opérations, notamment parce que ces machines peuvent opérer dans des milieux rudes et parfois difficiles d'accès pour les humains. Les robots peuvent aussi exécuter de multiples tâches à répétition sur de longues durées. La répétition de tâches étant l'une de leurs principales caractéristiques, on retrouve couramment les valeurs de répétabilité sur les fiches techniques des robots, ce qui n'est toutefois pas le cas pour la précision. Le mode de programmation des robots le plus utilisé dans l'industrie est la programmation par enseignement. Un autre mode appelé programmation hors ligne offre davantage de flexibilité et peut permettre de gagner du temps. Le robot programmé hors ligne peut reproduire les mouvements avec précision à condition que le modèle mathématique programmé dans son contrôleur soit le plus proche possible de la réalité. Les équations des modèles cinématiques sont basées sur la géométrie du robot et utilisent les valeurs des paramètres (valeurs fixes) qui caractérisent cette géométrie. Ces modèles sont classés en deux catégories, le modèle géométrique direct (MGD) et le modèle géométrique inverse (MGI). Le MGI permet de calculer la position articulaire θ_i de chacune des articulations motorisées en se basant sur la pose ou posture (position et orientation) commandée de l'organe terminal ou effecteur. En connaissant les variables articulaires θ_i, le MGD permet de déterminer la position et l'orientation de l'effecteur. Le programme du contrôleur du robot utilise les équations du MGI. Ainsi, pour déplacer l'effecteur à une pose désirée X_d, le contrôleur doit calculer le mouvement θ_i pour chacune des articulations motorisées en utilisant les valeurs de la position et de l'orientation de X_d ainsi que les valeurs des paramètres géométriques. Mais dans les faits, il y'a toujours des différences (erreurs de pose) entre ces valeurs. Ces erreurs sont dues aux différences entre les modèles théorique et réel du robot. En fait, les équations du MGI, qui calculent les mouvements articulaires du robot, utilisent les valeurs nominales des paramètres qui sont différentes de leurs valeurs réelles; principales conséquences des tolérances d'usinage et d'assemblage des composantes de la structure mécanique du robot. Pour améliorer la précision (c.-à-d, réduire les erreurs de pose), il s'avère nécessaire d'évaluer les « vraies » valeurs des paramètres géométriques du robot et de les insérer, par la suite, dans les équations de son MGI. Cette opération est appelée l'étalonnage géométrique (ÉG) des robots. Étant donnée, qu'il n'est pas toujours possible d'effectuer des mesures directes pour obtenir les valeurs réelles des paramètres, les approches d'étalonnage proposées dans la

littérature sont basées principalement sur des modèles d'optimisation. Les valeurs des paramètres identifiés par ces méthodes d'étalonnage ne correspondent pas nécessairement aux vraies valeurs des paramètres du robot. Elles représentent plutôt les valeurs qui permettent de satisfaire les fonctions objectives des modèles d'optimisation utilisés dans la procédure d'étalonnage.

Ces fonctions consistent généralement à minimiser les erreurs résiduelles des poses ou des mouvements articulaires, l'étalonnage des robots manipulateurs est un sujet traité par un grand nombre de travaux de recherche. Les robots sériels sont les plus étudiés, suivis des robots parallèles à six degrés de liberté (ddl) et, notamment, le mécanisme appelé plateforme de Stewart (aussi appelé hexapode). L'étalonnage des manipulateurs parallèles à moins de six degrés de liberté est beaucoup moins étudié, malgré que leur utilisation dans l'industrie soit de plus en plus populaire. Ceci étant dit, l'objectif de plusieurs thèses est de contribuer à l'amélioration de la précision absolue des robots parallèles à moins de six ddl, en utilisant des approches d'ÉG. Les méthodes proposées dans notre travail de recherche consistent à améliorer la précision tout en respectant les balises suivantes :

- améliorer la précision absolue de toute la cellule robotisée, en déterminant le positionnement du référentiel de la base du robot par rapport à celui de la cellule, tout en identifiant le reste des paramètres géométriques;
- apporter une attention particulière à la clarté et à la simplicité des méthodes proposées, l'objectif étant d'améliorer le plus possible la précision absolue et de réduire la complexité des approche proposées;
- proposer des méthodes adaptées à l'industrie, en proposant des approches axées sur la pratique et en utilisant des instruments de mesure modernes, de haute précision et largement disponible sur le marché.

Les approches d'étalonnage proposées dans le présent travail sont testées par simulations et appliquées sur robots parallèles 3RPR. Notons que ces méthodes d'étalonnage peuvent être utilisées pour d'autres robots parallèles. Ceci nous donne une idée de la précision que nous pourrions espérer atteindre après l'étalonnage dans le présent chapitre, suivie de la problématique de recherche, des objectifs de cette thèse et d'une brève description de la précision du modèle géométrique d'un robot, puis une revue de littérature qui couvrira les protocoles de mesures et l'étalonnage des robots industriels en se référant à des travaux de recherches relatifs aux manipulateurs parallèles. Porteront sur les travaux de recherche effectués dans le cadre de notre étude. Ainsi, le traitement et l'amélioration de la précision d'un robot parallèle à trois ddl, en adoptant une méthode d'étalonnage basée sur une analyse

géométrique. Ont présentés les résultats de l'étalonnage de ce même robot. L'approche utilisée est basée sur la réduction du nombre de positions d'étalonnage, tout en garantissant une amélioration significative de la précision absolue. Cet objectif est atteint par l'exploitation des notions des modes d'assemblage et des modes de travail du robot

Contexte et Problématique

Les premiers robots ont été développés dans les années 70 pour réaliser des opérations pénibles, dangereuses et surtout extrêmement répétitives qui ne requéraient qu'une bonne répétabilité et aucune exigence en termes de précision absolue. C'était le cas notamment des opérations de peinture de carrosseries automobiles ou de manutention. Ces opérations ayant longtemps constituées une part significative du marché, les fabricants de robots ont logiquement focalisé leurs efforts de recherche et développement sur la création de gammes de robots dédiées aux besoins de ces opérations ne nécessitant qu'une bonne répétabilité de pose. L'ordre de grandeur de cette répétabilite étant de quelques centièmes de millimètre pour les petits robots et de l'ordre de quelques dixièmes de millimètre pour les gros porteurs de plus de deux mètres de rayon d'action. La compétition mondialisée, l'augmentation de la fréquence des changements de production en réponse aux demandes de personnalisation des produits, amènent d'année en année les industriels à élargir le spectre des applications susceptibles d'être robotisées à l'instar des procèdes plus complexes en terme de trajectoires et plus exigeants en terme de précision absolue comme l'assemblage ou la mesure. Ainsi, le faible cout, la grande accessibilité et la facilite de reconfiguration des robots en font aujourd'hui le moyen matériel essentiel permettant de répondre a cette demande d'agilité. Dans ce contexte, l'adaptation des robots en vue de réaliser des opérations continues comme l'usinage représente aujourd'hui un besoin croissant de l'industrie manufacturière. De nombreuses applications d'usinage robotisé ont été développées par des intégrateurs sur la base de robots standards 6 axes sur lesquels ont été intégrés des éléctrobroches d'usinage. Mais l'exactitude et la faible rigidité des robots industriels en comparaison avec les machines outils néanmoins limitent les applications d'usinage robotisé aux matériaux généralement tendres et aux opérations ne nécessitant pas de précision importante. Le verrou technologique principal réside dans l'amélioration des performances statiques et dynamiques des robots de façon a atteindre le niveau de précision requis par le processus d'usinage dans l'espace de travail des robots.

II.2.Définitions

II.2.1.Précision

Elle est définie comme l'aptitude d'un robot à positionner son organe terminal (Pince) à une position fixée (cible) située dans le volume de travail. En terme de commande, la précision peut être définie comme la moitié de la résolution de commande. Cette définition de la précision implique la situation la plus défavorable on le point cible se trouve entre deux points de commande. En effet un déplacement inférieur à l'incrément de résolution de base ne peut être ni programmé ni mesuré.

La précision d'un robot dépend de plusieurs facteurs, par exemple quand le bras du robot est en pleine extension, les incertitudes tendent à devenir plus élevées, car la charge utile et les masses des différentes liaisons produisent une flexion.

II.2.2.Répétabilité

C'est la capacité d'un robot a positionner son organe terminal à la même position pour une même entrée de commande .Elle se traduit par un nuage de points autour du point cible dans le cas d'une manœuvre répétée n fois.

II.2.3.Résolution spatiale

La résolution spatiale d'un robot est le plus petit incrément de mouvement pouvant être discerné par le capteur de position et avec lequel le robot peut diviser son volume de travail. Elle dépend des jeux et imprécisions mécaniques Ces concepts peuvent être illustrés graphiquement (Figure II-1), en décrivant les quatre situations pouvant se produire suivant la qualité de la précision (bonne ou mauvaise)

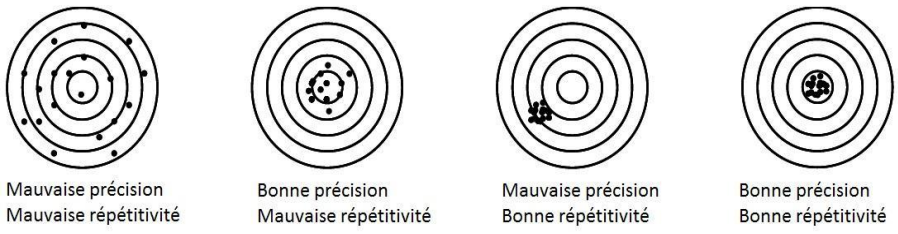


Figure II-1: Précision et répétabilité.[84]

La répétabilité de position (ε) peut être définie, comme étant le rayon de la plus petite sphère qui englobe toutes les positions atteintes pour une même valeur commandée (voir figure II.2). Selon la norme ISO 9283[84], la répétabilité de positionnement est plutôt évaluée en utilisant l'équation 1.1 présentée ci-dessous :

$$\varepsilon = \bar{r} + 3\sigma_r \tag{II.1}$$

Dans cette équation, σ_r représente l'écart type des erreurs composées de position (r) et est estimé comme suit

$$\sigma_r = \sqrt{\frac{\sum_{j=1}^{n}(r_j - \bar{r})^2}{n-1}} \tag{II.2}$$

Où $\bar{r}$ correspond à la moyenne estimée des erreurs composées de position r_j, dont l'évaluation s'effectue n fois pour la même position commandée, soit

$$\bar{r} = \frac{1}{n}\sum_{j=1}^{n} r_j \tag{II.3}$$

Avec

$$r_j = \sqrt{(\bar{x} - x_j)^2 + (\bar{y} - y_j)^2 + (\bar{z} - z_j)^2} \tag{II.4}$$

En sachant que $\bar{x}$, $\bar{y}$ et $\bar{z}$ représentent les moyennes des coordonnées x, y et z mesurées, pour la même position commandée n fois.

La répétabilité d'orientation pour une pose donnée, se calcule séparément pour chacune des orientations angulaires a, b et c:

$$\varepsilon_a = \pm 3\sqrt{\frac{\sum_{j=1}^{n}(a_j - \bar{a})^2}{n-1}} \tag{II.5}$$

$$\varepsilon_b = \pm 3\sqrt{\frac{\sum_{j=1}^{n}(b_j - \bar{b})^2}{n-1}} \tag{II.6}$$

$$\varepsilon_c = \pm 3\sqrt{\frac{\sum_{j=1}^{n}(c_j - \bar{c})^2}{n-1}} \tag{II.7}$$

Avec

$$\bar{a} = \frac{1}{n}\sum_{j=1}^{n} a_j \,, \bar{b} = \frac{1}{n}\sum_{j=1}^{n} b_j \,, \; \bar{c} = \frac{1}{n}\sum_{j=1}^{n} c_j \tag{II.8}$$

$\bar{a}$, $\bar{b}$ et $\bar{c}$ correspondent au moyennes des orientations angulaires obtenues pour la même pose répétée n fois.

a_j , b_j et c_j sont les coordonnées angulaires de la j^{ieme} pose atteinte.

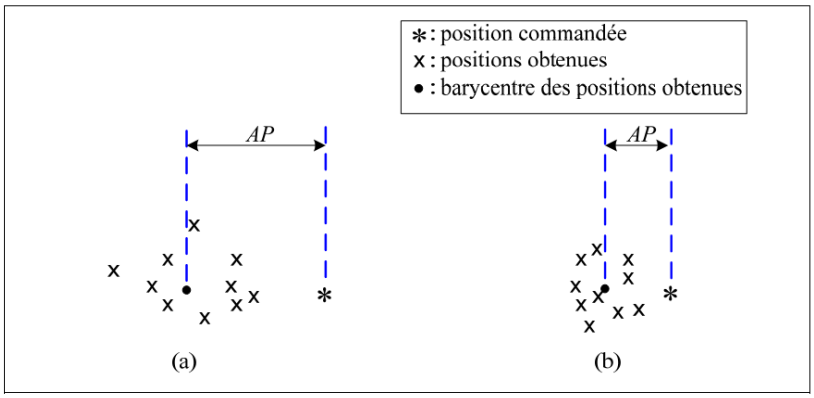


Figure II.2 : Illustration 2D (a) d'une mauvaise et (b) une bonne valeur de répétabilité de position. [85].

II.2.3.1 Précision absolue de pose

La précision absolue en positionnement correspond à la capacité du robot d'atteindre, avec le maximum d'exactitude, une pose (position et orientation) commandée. Notons qu'ici nous parlons de la précision en mode statique (i.e. évaluation de la pose après un arrêt complet du mouvement de l'effecteur) et indépendamment de la trajectoire prise pour se rendre à la pose désirée.

Géométriquement, la précision du robot dans une position donnée peut être définie comme étant la distance entre la position commandée (x_c , y_c , z_c) et le barycentre des positions effectivement atteintes ($\bar{x}$, $\bar{y}$,$\bar{z}$), après des mouvements répétitifs de l'effecteur à cette position. La figure (II.2) illustre géométriquement la notion de précision absolue.

Mathématiquement, selon la norme ISO9283 [86], le calcul de la précision A_P à une position donnée consiste à calculer l'erreur composée selon les coordonnées cartésiennes *x*, *y* et *z* :

$$Ap = \sqrt{(\bar{x} - x_c)^2 + (\bar{y} - y_c)^2 + (\bar{z} - z_c)^2} \qquad \text{(II.9)}$$

avec, $\bar{x}$, $\bar{y}$ *et* $\bar{z}$ correspondent au moyennes des valeurs mesurées en *x*, *y* et *z*.

La précision d'orientation pour une pose donnée, se calcule séparément pour chacune des orientations angulaires :

$$AP_a = \overline{a} - a_c \quad \text{(II.10)}$$

$$AP_b = \overline{b} - b_c \quad \text{(II.11)}$$

$$AP_c = \overline{c} - c_c \quad \text{(II.12)}$$

a_c , b_c et c_c représentent les coordonnées angulaires de la pose commandée.

La qualité de précision des robots est un critère important pour plusieurs applications (le domaine médical, l'usinage de précision…etc.). Toutefois, selon une recherche que nous avons effectuée auprès de 13 des plus populaires fabricants des robots, ceux-ci insistent sur la répétabilité dans leurs documents techniques, sans mentionner les valeurs de la précision. En fait, c'est considéré qu'avec une bonne répétabilité des robots, le manque de précision pourrait être compensé par l'étalonnage.

Notons que depuis quelques années, certains fabricants de robots manipulateurs développent et proposent sur le marché des options préprogrammées destinées à l'étalonnage de leurs robots. A notre connaissance, et sans vouloir mettre en doute l'efficacité de ces options, il n'y a pas encore de travaux de recherche qui aient fait état des résultats obtenus. Les fabricants des instruments de métrologie commencent, eux aussi, à démontrer une attention particulière à l'amélioration de la précision des robots, en proposant des outils de plus en plus adaptés aux processus d'étalonnage. Cet intérêt croissant pour l'étalonnage va de pair avec l'augmentation de la demande pour des robots offrant une précision de qualité.

II.2.3-2 Domaines nécessitant une bonne précision absolue

Un robot doté d'une bonne répétabilité n'est pas nécessairement efficace dans toutes les tâches qu'on lui demande d'exécuter, s'il n'a pas une bonne précision. En fait, les robots industriels, par rapport à leur répétabilité, ont une précision faible [87]. Une telle situation ne cause pas d'inconvénients pour les cas où la programmation est effectuée selon la méthode traditionnelle, basée sur l'enseignement des poses. Par ailleurs, il existe une multitude d'applications (Greenway, 2000; THESAME, 2012) où la précision des robots a autant d'importance que leur répétabilité :

- la programmation hors ligne;
- l'interchangeabilité des robots sans devoir refaire l'enseignement des positions;
- l'utilisation d'un robot comme machine à mesurer;
- l'utilisation des robots dans les interventions chirurgicales;
- l'utilisation des robots dans l'usinage de précision;
- l'utilisation des robots dans des manipulations de haute précision ;

• la production cellulaire, où plusieurs robots se trouvent dans le même espace de travail. Une bonne précision permet d'éviter les collisions et de rendre l'ensemble des robots plus efficace.

II.2.4. Causes de manque de précision des robots industriels et approches d'étalonnages appropriés

Plusieurs travaux de recherche ont discuté de l'origine des erreurs de poses des robots industriels (Schroer, 1994; Greenway, 2000). Celles-ci sont attribuées aux erreurs des articulations actives, à la précision des modèles cinématiques et aux erreurs occasionnées par les aspects mécaniques des robots.

Ces sources d'erreurs, présentées à la Figure II.3, sont appelées respectivement, les facteurs articulaires, les facteurs géométriques et les facteurs non géométriques.

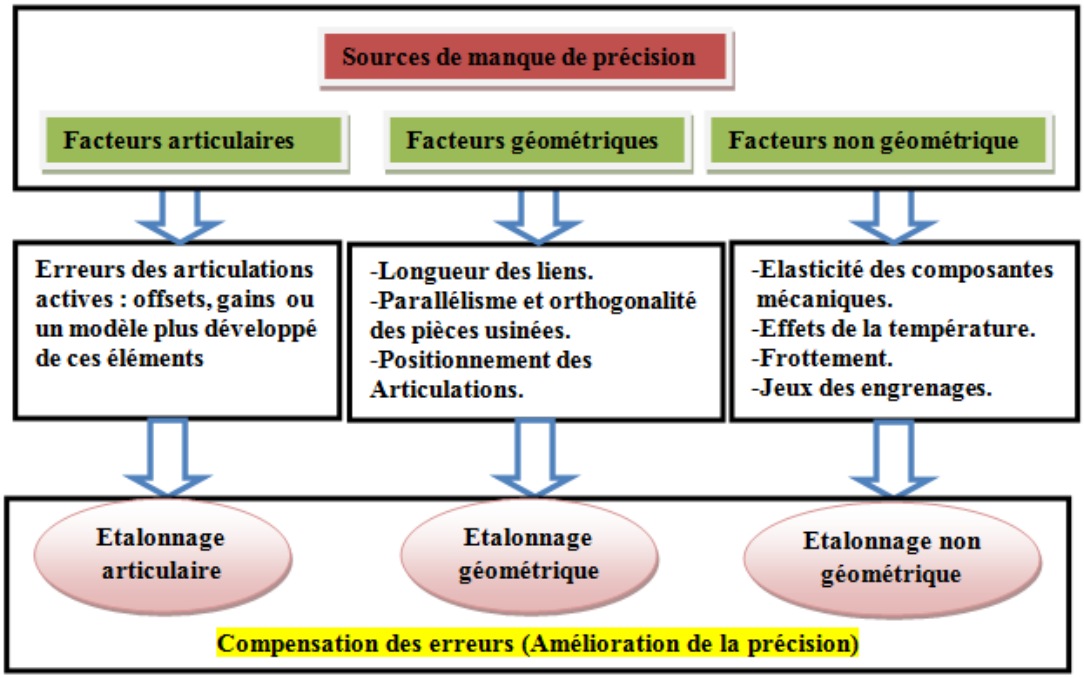


Figure II.3 : Sources de manque de précision des robots et approches d'étalonnage correspondante.

II.2.4.1 Facteurs articulaires

Ces facteurs de manque de précision correspondent aux erreurs des capteurs des articulations actives du robot. Elles représentent les différences entre les valeurs fournies par les encodeurs des actionneurs et les mouvements articulaires réellement effectués.

II.2.4.2 Facteurs géométriques

Les facteurs géométriques sont liés à la précision du modèle cinématique du robot, utilisé par son contrôleur pour calculer les mouvements de l'effecteur. Un modèle qui ne représente pas exactement la géométrie réelle du robot est une source fondamentale de manque de précision.

Les principales causes d'erreurs géométriques sont :

- les longueurs nominales des liens différentes des longueurs réelles. Ceci est causé principalement par les tolérances de fabrication et d'assemblage;
- les caractéristiques géométriques des pièces usinées (parallélisme, orthogonalité, planéité, etc.);
- le positionnement des articulations liées à la base et à l'élément terminal du robot;
- les erreurs de localisation du référentiel de la base du robot par rapport à celui de la cellule;
- les erreurs de localisation du référentiel de l'outil sur l'organe terminal.

II.2.4.3 Facteurs non géométriques

Les facteurs non géométriques sont attribués à la qualité des composantes mécaniques du robot. Ils comptent plusieurs éléments dont voici les principaux :

- L'élasticité des composantes mécaniques;
- Les jeux mécaniques, notamment ceux des engrenages;
- Les effets de la température.

Bien que tous ces éléments (articulaires, géométriques et non géométriques) contribuent à dégrader la qualité de la précision des robots, leurs degrés d'impact varient. Plusieurs recherches démontrent que ce sont les erreurs articulaires et celles d'origine géométrique qui affectent le plus la précision des robots. Ainsi, selon Judd et Knasinski (1990) et Damak (1996), les sources de manque de précision sont classées par ordre d'importance comme suit :

1. les erreurs articulaires (offsets de la position zéro et gains);
2. les erreurs causées par les tolérances d'usinage et d'assemblages (erreurs géométriques);
3. les erreurs non géométriques. Notons que l'impact des erreurs non géométriques peut augmenter selon les caractéristiques mécaniques, l'environnement et les conditions d'utilisation du robot en question. Parmi ces facteurs influents, on peut citer : l'état

mécanique des réducteurs de vitesse, les charges manipulées et leur emplacement dans l'espace de travail, la vitesse de fonctionnement et les configurations adoptées par le robot pour atteindre les poses commandées.

II.3.Les Défauts De Précision

II.3.1 Les Défauts

II.3.1.1. Origines des défauts

a. Les défauts au niveau de la partie opérative

-La structure: défauts géométriques, tolérances de fabrication, jeux….

-Les actionneurs et les capteurs : défauts dimensionnels, de résolution, temps de réponse, dérive dans le temps….

-Les transmissions des mouvements et les retours d'information s: jeux, points durs, déformations, frottement qui demandent un effort supplémentaire au démarrage que les asservissements doivent gérer. Le frottement visqueux, stabilisateur est proportionnel à la vitesse, il correspond à une contre–réaction de la vitesse. Le frottement sec introduit un seuil d'effort et élimine les perturbations de faibles amplitudes. En l'absence de tout frottement, le système sensible aux petites perturbations tend à devenir instable. Dans un asservissement de position le frottement est déstabilisateur, surtout à basse vitesse, au démarrage, au freinage et en présence de jeux. Les variations brutales entre les coefficients d'adhérence et de frottement se traduisent par une accélération, un dépassement du point cible programmé. Les valeurs des frottements sont évaluées par observation du courant : + 5 à 10% qui génère une instabilité des moteurs .La diminutions du gain de l'action intégrale permet souvent de remédier à cette instabilité au détriment de la précision.[88]

-Les jeux de ± 2 µm dans les réducteurs entrainent une erreur sur le positionnement de ± 5 µm et de ± 15 µm en suivi de trajectoire.

-Les jeux fonctionnels dans les paliers et entre les dentures.

-Le manque d'homocinétisme dans les accouplements et les transmissions.

-Les diverses élasticités et déformations sous charges et les transmissions.

-Les déformations dues aux variations de températures du système et de l'environnement.

b. Les défauts de la partie commande

-Les défauts liés à la non linéarité des équations, aux inversions, aux approximations des calculs, au nombre de termes après la virgule.

-Les défauts qui résultent du traitement des informations et du signal.

-L'imprécision des asservissements, un modèle trop éloigné de la réalité, le temps de réponse, la dérive, le manque de performances pour les mouvements rapides, le mauvais réglage des gains, le traitement des boucles de retour trop long et le manque de précision.

-L'intervalle d'échantillonnage mal adapté.

c. Les problèmes provenant de l'environnement

-Le robot qui évolue dans des milieux liquides est soumis à des freinages visqueux.

-Les évolutions dans des milieux pollués abrasifs, à grandes variations de température, dans des champs magnétiques intenses créent des perturbations sur les actionneurs, les capteurs, la transmission des signaux et le traitement des informations …

d. Les vibrations : quatre cas

- Les vibrations provoquées par des sollicitations dynamiques excitatrices: mauvais équilibrages, des couplages à d'autres systèmes mécaniques.

- Les vibrations qui résultent d'une instabilité de la commande, c'est un phénomène de pompage. On y remédie en en améliorant la stabilité par un procédé de compensation.

- Les vibrations causées par une excitation de la commande, d'un mode propre d'un pré-actionneur, d'une commande à fréquence élevée. La vibration du pré-actionneur est convertie en ondulations sur les couples des actionneurs.[88]

e. Prise en compte de ces défauts –Améliorations

-Faire le bilan des qualités, défauts, performances, caractéristiques de chaque composant.

-Affiner le meilleur choix par rapport aux compromis acceptables.

-Modélisations précises des tâches, de l'environnement en tenant compte des contraintes.

-Définir la gestuelle du robot avec précision : lois cinématiques et amplitudes, points d'arrêts, dépassements acceptables, contournements, coopérations…

-Réaliser des procédures performantes.

-Définir les sécurités et la durée de vie des matériels.

f. Quantification des erreurs et des défauts.

Elle est complexe, beaucoup de paramètres connus avec plus ou moins de précision interviennent, il faut leur attribuer des coefficients aléatoires .Se référer aux documents de l'A.F.R.I (Association Française de Robotique Industrielle), de l'A.F.N.O.R (Association Française de Normalisation) qui présentent les définitions, se définissent par rapport à une référence ou un repère.

Ces causes génèrent des déformations tridimensionnelles qu'il faut modéliser et mettre en équations, celles-ci sont fortement non linéaires. Comme les déformations restent faibles, on peut les linéariser pour faciliter la résolution, il en est de même pour les mouvements vibratoires analysés dans les chapitres suivants.

II.3.1.2. Erreurs de déformations d'un segment-Modélisations-Mises en équations

En considérant des mouvements lents, on fait une modélisation géométrique des erreurs de chaque élément. Les mouvements rapides utilisent le modèle dynamique.

a. Les déformations d'un segment-Flexion-Torsion-Allongement

-Les liaisons peuvent être d'orientation quelconque et les segments coudés.

-pour une glissière, la longueur de l'élément varie selon sa sortie. Pour une même charge, les déformations varient également selon cette sortie.

Voir les relations successives pour passer de la base (B_i) à (B_{i+1})

La déformation étant faible, la sécante peut remplacer la courbe**.**

$O_{i+1(th)}$: position théorique avant déformation

$O_{i+1(r)}$: position réelle après la déformation

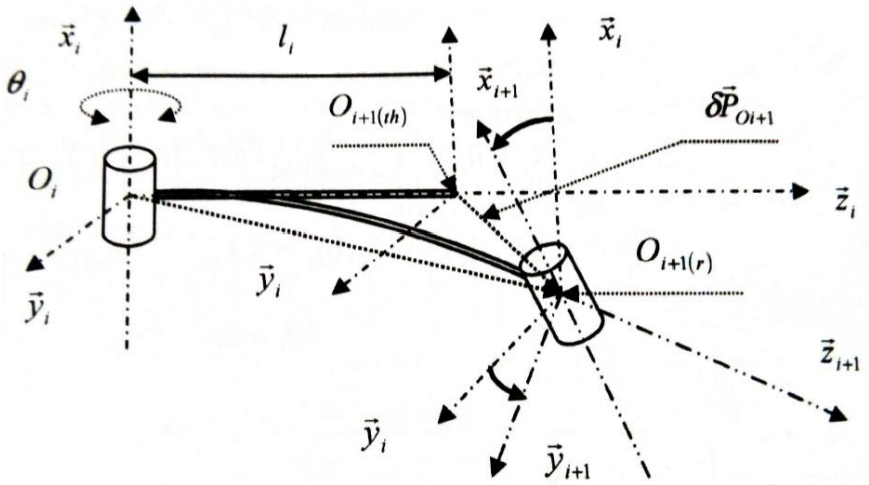


Figure II.4 : Représentation et paramétrage de la déformation d'un segment

b. Mise en équations d'un segment par le vecteur déplacement et orientation

Déplacement $\overrightarrow{O_i O_{i+1(r)}} = \overrightarrow{O_i O_{i+1(th)}} + \overrightarrow{\delta P_{Oi+1}}$ (II.13)

Orientation $\vec{\phi}_{Oi+1(r)} = \vec{\phi}_{Oi+1(th)} + \partial\vec{\phi}_{Oi+1} = \left[\vec{\phi}_{Oi} \rightarrow \vec{\phi}_{Oi+1}\right] + \vec{\phi}_{Oi+1}$ (II.14)

Ces erreurs sont calculées à partir des torseurs appliqués à chaque segment.

-Les déformations pour arriver en $O_{i+1(r)}$ sont dues à la flexion, à l'allongement ou au raccourcissement, à la variation de la température .Le flambement n'intervient que pour les grandes longueurs, type grue.

-Les variations des orientations proviennent de la torsion autour de l'axe $\vec{z}_{i+1}$ des flexions dans deux plans autours des axes $\vec{x}_i$ et $\vec{y}_i$, des tolérances de fabrication, de la perpendicularité de la liaison avec le segment, des déformations dans les paliers. [88]

$$\vec{P}_{i(r)} = \vec{P}_{i(th)} + \delta\vec{P}_{i(Bi)} \quad \text{(II.15)}$$

$$\vec{\phi}_{i+1(Bi)(r)} = \vec{\phi}_{i(Bi)(th)} + \delta\vec{\phi}_{i(Bi)} \quad \text{(II.16)}$$

(th)=Position et orientation théoriques

(r)=Position et orientation réelles

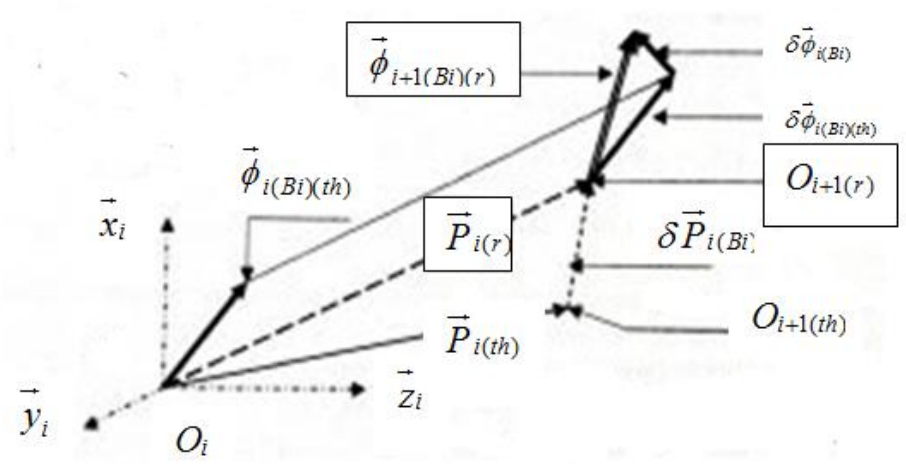


Figure II.5.Paramétrage des erreurs de déplacements et de rotations par les vecteurs

La position $\vec{p}_{i(th)}$ et l'orientation $\delta\vec{\phi}_{i(Bi)(th)}$ théorique sont calculées pour une structure parfaite. Les erreurs et les déformations sont données par les vecteurs $\delta\vec{p}_{i(Bi)}$ et $\delta\vec{\phi}_{i(Bi)}$ .Les vecteurs réels sont obtenus en faisant les sommes vectorielles, **Figure II.5**.

b.1 Vecteur position des erreurs linéaires :

$$\delta\vec{p}_{i(Bi)} = \delta x_i.\vec{x}_i + \delta y_i.\vec{y}_i + \delta z_i.\vec{z}_i \qquad \text{(II.17)}$$

Les δx_i, δy_i, δz_i intègrent toutes les erreurs de déformations linéaires.

b.2 Vecteur rotation des erreurs

$$\delta\vec{\phi}_{i(Bi)} = A^{(Bi)}_{(Bi)}.\Delta\alpha_i.(\vec{x}_i = \vec{u}_i) + A^{(Bi)}_{\vec{x}i,(\vec{v}i=\vec{q}i),\vec{w}i)}.\Delta\beta_i.(\vec{v}_i = \vec{q}_i) + A^{(Bi)}_{(\vec{p}i,\vec{q}i,(\vec{r}i=\vec{z}i+1))}.\Delta\gamma_i.(\vec{r}_i = \vec{z}_{i+1}) \qquad \text{(II.18)}$$

On utilise des bases intermédiaires :

$$(\vec{x}_i, \vec{y}_i, \vec{z}_i) \rightarrow ((\vec{u} = \vec{x}_i), \vec{v}_i, \vec{w}_i) \rightarrow (\vec{p}_i, (\vec{q}_i = \vec{v}_i), \vec{r}_i) \rightarrow (\vec{x}_{i+1}, \vec{y}_{i+1}, (\vec{r} = \vec{z}_{i+1})) \qquad \text{(II.19)}$$

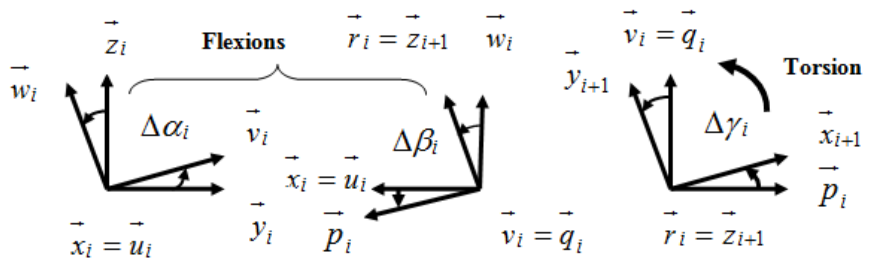


Figure II.6: Paramétrage des erreurs angulaires

On a de la flexion déviée.

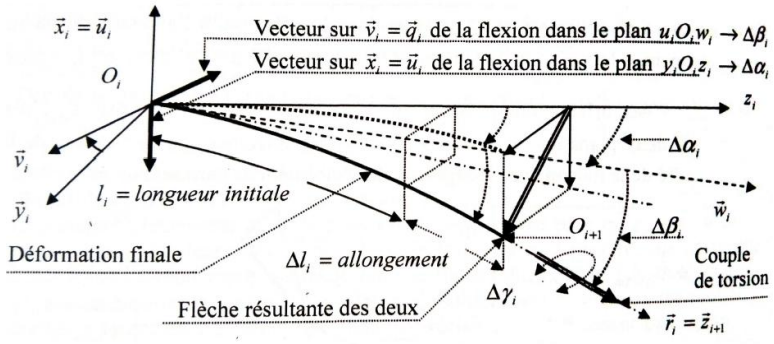


Figure II.7: Déformations d'un segment

Ces déformations sont calculées à partir des torseurs des actions extérieures et cinétiques qui agissent en amont.

c.Vecteur des erreurs de position et matrice d'orientation [3 x 3]

c.1 Vecteur position des erreurs linéaires : $\delta\vec{p}_{i(Bi)} = \delta x_i.\vec{x}_i + \delta y_i.\vec{y}_i + \delta z_i.\vec{z}_i$ **, allongement**

c.2 Matrice [3 x 3] des erreurs angulaires : $\left[\delta\vec{\phi}^{\,i}_{i+1}\right] = A^{(Bi)}_{(\vec{u}i,\vec{v}i,\vec{w}i)}.A^{(\vec{u}i,\vec{v}i,\vec{w}i)}_{(\vec{p}i,\vec{q}i,\vec{r}i)}.A^{(\vec{p}i,\vec{q}i,\vec{r}i)}_{(Bi+1)}$

$$\left[\delta\phi^{i}_{i+1}\right] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C\Delta\alpha_i & -S\Delta\alpha_i \\ 0 & S\Delta\alpha_i & C\Delta\alpha_i \end{bmatrix}.\begin{bmatrix} C\Delta\beta_i & 0 & S\Delta\beta_i \\ 0 & 1 & 0 \\ -S\Delta\beta_i & 0 & C\Delta\beta_i \end{bmatrix}.\begin{bmatrix} C\Delta\gamma_i & -S\Delta\gamma_i & 0 \\ S\Delta\gamma_i & C\Delta\gamma_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.20)}$$

[-----2 Flexions ou flexion déviée ----------] [----Torsion----------]

Pour un segment homogène, isotrope et de grande rigidité .la déformation est faible, elle peut être approximée par sa sécante .Cela n'est pas acceptable pour des structures élastiques et souples. On considère, ici, les angles faibles, les matrices se simplifient alors :

$C\Delta\alpha_i \cong 1$, $S\Delta\alpha_i \cong \Delta\alpha_i$ .Les expressions sont plus simples et linéaires.

$$\Rightarrow\left[\delta\phi^{i}_{i+1}\right] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -\Delta\alpha_i \\ 0 & \Delta\alpha_i & 1 \end{bmatrix}.\begin{bmatrix} 1 & 0 & \Delta\beta_i \\ 0 & 1 & 0 \\ -\Delta\beta_i & 0 & 1 \end{bmatrix}.\begin{bmatrix} 1 & -\Delta\gamma_i & 0 \\ \Delta\gamma_i & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.21)}$$

d.La matrice homogène [4x4] des erreurs d'un segment

$$\left[\Delta M_{i+1}^{i}\right]=\begin{bmatrix} & \left[\delta\phi_{i+1}^{i}\right] & & \begin{matrix}\delta x_i\\ \delta y_i\\ \delta z_i\end{matrix}\\ 0 & 0 & 0 & 1\end{bmatrix}.\begin{bmatrix} & \left[\delta\phi_{i+1}^{i}\right] & \delta\vec{P}_{i(Bi)}\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad \text{(II.22)}$$

Ecriture symbolique

$$\begin{cases} \textbf{[M de déform.du segment (i)]= [Rot ; } \vec{x}_i=\vec{u}_i\textbf{ ,} \Delta\alpha_i \textbf{]}^{\mathbf{T}} \textbf{.[Rot ; } \vec{v}_i=\vec{q}_i, \Delta\beta_i \textbf{]}^{\mathbf{T}} \\ \\ \textbf{[Tr ; } \vec{r}_i, \Delta l_i \textbf{].[Rot ; } \vec{r}_i=\vec{z}_{i+1}, \Delta\gamma_i \textbf{]}^{\mathbf{T}} \end{cases} \qquad \text{(II.23)}$$

$\Delta\gamma_i$ = Déformation du segment l_i , due à la torsion

Les deux premières matrices décrivent la déformation des flexions, la troisième l'allongement, la quatrième la déformation de torsion.

$$\left[\begin{matrix}M.de.déform.d'un\\ segment(i),souple\end{matrix}\right]=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & -\cos\Delta\alpha_i & -\sin\Delta\alpha_i & 0\\ 0 & \sin\Delta\alpha_i & \cos\Delta\alpha_i & 0\\ 0 & 0 & 0 & 1\end{bmatrix}.\begin{bmatrix}\cos\Delta\beta_i & 0 & -\sin\Delta\beta_i & 0\\ 0 & 1 & 0 & 0\\ -\sin\Delta\beta_i & 0 & \cos\Delta\beta_i & 0\\ 0 & 0 & 0_i & 1\end{bmatrix}.\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & \Delta l_i\\ 0 & 0 & 0 & 1\end{bmatrix}.$$

$$\begin{bmatrix}\cos\Delta\gamma_i & -\sin\Delta\gamma_i & 0 & 0\\ \sin\Delta\gamma_i & \sin\Delta\gamma_i & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad \text{(II.24)}$$

Pour des petits angles : $\cos\Delta\alpha_i\cong 1, \sin\Delta\alpha_i\cong\Delta\alpha_i$, idem pour $\Delta\beta_i$ et $\Delta\gamma_i$

$$\left[M.de.déform.du.seg(i).à.faible \quad déform\right]=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & -\Delta\alpha_i & 0\\ 0 & \Delta\alpha_i & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}.\begin{bmatrix}1 & 0 & \Delta\beta_i & 0\\ 0 & 1 & 0 & 0\\ -\Delta\beta_i & 0 & 1 & 0\\ 0 & 0 & 0_i & 1\end{bmatrix}.\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & \Delta l_i\\ 0 & 0 & 0 & 1\end{bmatrix}.\begin{bmatrix}1 & -\Delta\gamma_i & 0 & 0\\ \Delta\gamma_i & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad \text{(II.25)}$$

$$\begin{bmatrix} M.de.déform.du.segment \\ (i).à.faibles.déform. \end{bmatrix} = \begin{bmatrix} 1 & -\Delta\gamma_i & \Delta\beta_i & \Delta l_i.\Delta\beta_i \\ [\Delta\gamma_i + \Delta\alpha_i.\Delta\beta_i] & [1-\Delta\alpha_i.\Delta\beta_i.\Delta\gamma_i] & -\Delta\alpha_i & -\Delta l_i.\Delta\alpha_i \\ [-\Delta\beta_i + \Delta\alpha_i.\Delta\gamma_i] & [\Delta\alpha_i + \Delta\beta_i.\Delta\gamma_i] & 1 & \Delta l_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.26)}$$

Les infiniment petits d'ordre supérieur à 1 sont négligés, la matrice de déformation du segment est considérablement simplifiée.

$$\begin{bmatrix} M.de.déform.du.segment \\ (i).à.faibles.déform. \end{bmatrix} = \begin{bmatrix} 1 & -\Delta\gamma_i & \Delta\beta_i & 0 \\ \Delta\gamma_i & 1 & -\Delta\alpha_i & 0 \\ -\Delta\beta_i & \Delta\alpha_i & 1 & \Delta l_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.27)}$$

Ces déformations sont calculées ultérieurement.

e. Application à un segment en tenant compte de sa variable généralisée

Le modèle est écrit à partir des figures II.5- II.6- II.7, matrice [4 x4]. La position de o_{i+1} et la base associée à cette liaison sont données par l'expression :

$$\left[M_{i+1}^{i} \quad \overrightarrow{o_i o_{i+1(Bi)}}\right]^{\Delta} = \left[Rot;\vec{x}_i = u_i,(\theta_i + [\Delta\theta_i + \Delta\alpha_i])\right]^{T}.\left[Rot;\vec{v}_i = \vec{q}_i,\Delta\beta_i\right]^{T}. \quad \text{(II.28)}$$
$$\left[Tr;\vec{r}_i,(l_i + \Delta l_i)\right].\left[Rot;\vec{r}_i = \vec{z}_{i+1},\Delta\gamma_i\right]^{T}$$

Cette formule n'est valable que pour ce segment et ce paramétrage.

θ_i = Variable généralisée suivant l'axe $\vec{x}_i = \vec{u}_i$.

$\Delta\psi_i = [\Delta\theta_i + \Delta\alpha_i]$ = Somme des erreurs comptabilisées sur l'axe $\vec{x}_i = \vec{u}_i$.

$\Delta\theta_i$ = Erreur sur la motorisation, la transmission et les capteurs autours de $\vec{x}_i = \vec{u}_i$.

Voir la représentation figure **II.12**, ou l'axe $\vec{z}_i$ et non pas $\vec{x}_i = \vec{u}_i$.

$\Delta\alpha_i$ = Rotation de la base en o_{i+1} due à la flexion autour de l'axe $\vec{x}_i = \vec{u}_i$.

$\Delta\beta_i$ = Rotation de la base en o_{i+1} due à la flexion autour de l'axe $\vec{v}_i = \vec{q}_i$.

$\Delta\gamma_i$ = Rotation de la base en o_{i+1} due à la flexion autour de l'axe $\vec{r}_i = \vec{z}_{i+1}$.

l_i = Longueur du segment et Δl_i = Erreur sur la longueur de ce segment suivant $\vec{z}_{i+1}$.

$\overrightarrow{o_i o_{i+1}} = (l_i + \Delta l_i).\vec{z}_{i+1}$ = Vecteur position, $\vec{r}_i = \vec{z}_{i+1}$

$$\left[M_{i+1}^{i}\overrightarrow{o_i o_{i+1(Bi)}}\right]^{\Delta} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C(\theta_i + \Delta\psi_i) & -S(\theta_i + \Delta\psi_i) & 0 \\ 0 & S(\theta_i + \Delta\psi_i) & C(\theta_i + \Delta\psi_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} C\Delta\beta_i & 0 & S\Delta\beta_i & 0 \\ 0 & 1 & 0 & 0 \\ -S\Delta\beta_i & 0 & C\Delta\beta_i & 0 \\ 0 & 0 & 0_i & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & (l_i + \Delta l_i) \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot$$

$$\begin{bmatrix} C\Delta\gamma_i & -S\Delta\gamma_i & 0 & 0 \\ S\Delta\gamma_i & C\Delta\gamma_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.29)}$$

$\Delta\psi_i$ est petit : $C(\theta_i + \Delta\psi_i) \cong (C\theta_i - \Delta\psi_i S\theta_i)$ et $S(\theta_i + \Delta\psi_i) \cong (S\theta_i + \Delta\psi_i C\theta_i)$

$$\left[M_{i+1}^{i}\overrightarrow{o_i o_{i+1(Bi)}}\right]^{\Delta} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & (C\theta_i + \Delta\psi_i S\theta_i) & -(S\theta_i + \Delta\psi_i C\theta_i) & 0 \\ 0 & (S\theta_i + \Delta\psi_i C\theta_i) & (C\theta_i + \Delta\psi_i S\theta_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & \Delta\beta_i & 0 \\ 0 & 1 & 0 & 0 \\ -\Delta\beta_i & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & (l_i + \Delta l_i) \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot$$

$$\begin{bmatrix} 1 & -\Delta\gamma_i & 0 & 0 \\ \Delta\gamma_i & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.30)}$$

Après développement des calculs, on ne considère que les infiniment petits d'ordre 1.La modélisation des chaines ouvertes avec leurs erreurs est simple, en revanche, la détermination des modules des déformations exige beaucoup de calculs en temps réel.

II.3.1.3. Erreurs locales-Modélisations- Equations

Les erreurs sont amplifiées par la longueur et la succession des segments. Ce paragraphe détaille les erreurs les plus importantes.

a. Erreurs au niveau des paliers

a.1 Tolérances de fabrication –Erreurs géométriques

Les principales erreurs proviennent des tolérances de fabrication et des déformations locales dues aux charges .Ces dernières varient selon la géométrie instantanée de la structure, des masses manipulées et des effets dynamiques.

-Les tolérances radiales de fabrication, angulaires (a) et de montage.

-Positionnement longitudinal (p).

Ces erreurs sont connues par des mesures de « contrôle-qualité », comme pour les machines outils. C'est le travail du constructeur de la partie opérative.

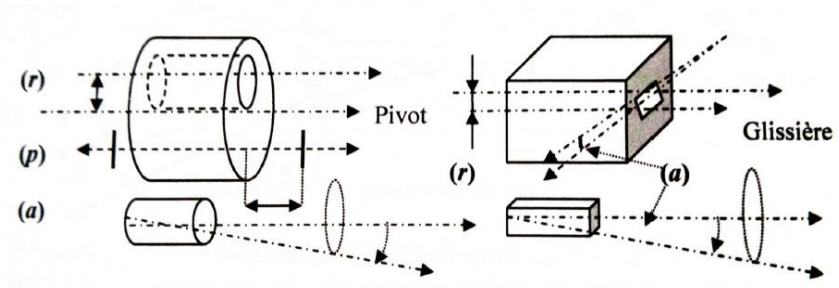


Figure II.8: Erreurs dues à la fabrication et au montage

a.2 Déformation élastiques

Déformations élastiques locales au niveau des contacts et des éléments roulants. Des composants en céramique répondent bien au problème mais craignent les chocs. Les éléments roulants cylindriques et coniques permettent une rigidité et des charges plus élevées que les billes. Les jeux dans les roulements sont réduits par des précontraintes. Pour les calculs, consulter les catalogues des fabricants.

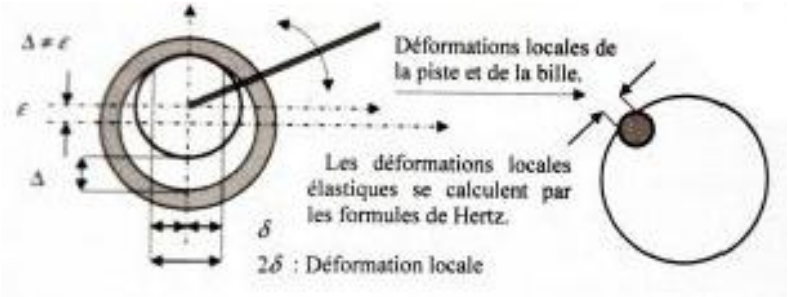


Figure II.9: Déformation locales élastiques dans les paliers

a.3 Influence des déformations locales, des tolérances, des jeux sur l'extrémité d'un bras

Lorsque plusieurs segments composent un bras, les erreurs s'additionnement.

si h=50 mm, l=1000 mm, ε la déformation locale =0.01 mm

$tg(\Delta\alpha) = \frac{0.01}{2.5} = 4.10^{-4}$ et $\frac{2\varepsilon}{h} = \frac{\delta}{l} \rightarrow \delta = \frac{2\varepsilon.l}{h} = \frac{0.02.1000}{50} = 0.4mm$ valeur importante

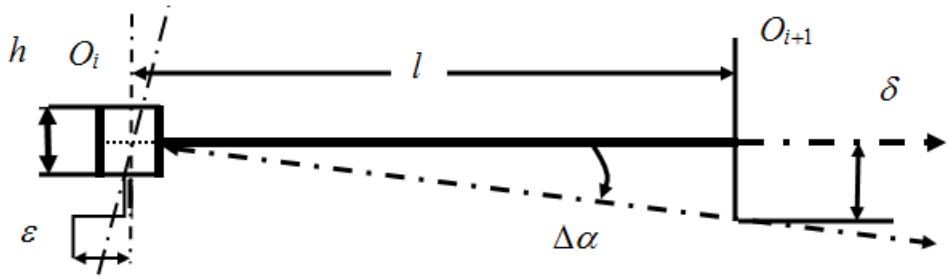


Figure II.10: Influence d'une déformation locale sur l'extrémité d'un bras

Plus le bras de levier est long, plus le débattement en B augmente. (Figure II.11)

Les erreurs proviennent :

- Du jeu dans le palier
- De l'erreur de perpendicularité

$\Delta\alpha_i, \Delta\beta_i, \Delta\gamma_i, \Delta\delta_i$ L'allongement

Elles sont définies par mesures.

Les déformations élastiques résultent des charges statiques et dynamiques

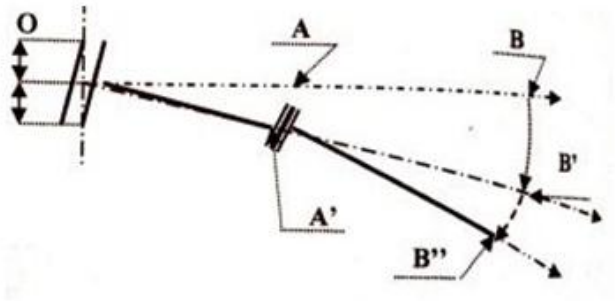


Figure II.11: Erreurs dues aux tolérances de fabrication

Modélisation –Indice p=palier

On retrouve des expressions similaires aux précédentes.

a.3.1-Vecteur position des erreurs linéaires : $\delta\vec{p}_{p,i(Bi)} = \delta x_{p,i}.\vec{x}_i + \delta y_{p,i}.\vec{y}_i + \delta z_{p,i}.\vec{z}_i$

-Matrices [3 x3] des erreurs angulaires : $\lfloor \delta\phi^i_{p,i+1} \rfloor$

a.3.2- La matrice homogène [4x4] des erreurs, on néglige les IP d'ordre supérieur à 1.

$$\left[\Delta M_{p,(i)}\right]_{(Bi+1)}^{(Bi)} = \begin{bmatrix} \left[\delta\phi_{i+1}^{i}\right] & \delta\vec{P}_{p,i(Bi)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \cong \begin{bmatrix} 1 & -\Delta\gamma_{p,i} & \Delta\beta_{p,i} & \delta x_{p,i} \\ \Delta\gamma_{p,i} & 1 & -\Delta\alpha_{p,i} & \delta y_{p,i} \\ -\Delta\beta_{p,i} & \Delta\alpha_{p,i} & 1 & \delta z_{p,i} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.31)}$$

a.3.3-Les vectrices positions et orientation

*Le vecteur position des erreurs linéaires : $\delta\vec{p}_{p,i(Bi)} = \delta x_{p,i}.\vec{x}_i + \delta y_{p,i}.\vec{y}_i + \delta z_{p,i}.\vec{z}_i$ idem

*L e vecteur orientation des erreurs angulaires :

$$\delta\vec{\phi}_{p,i(Bi)} = \left[M_{Bi}^{Bi}\right].\Delta\alpha_{p,i}.(\vec{x}_i = \vec{u}_i) + \left[M_{(\vec{u},\vec{v},\vec{w})}^{Bi}\right].\Delta\beta_{p,i}.(\vec{v} = \vec{q}_i) + \left[M_{(\vec{p},\vec{q},\vec{r})}^{Bi}\right].\Delta\gamma_{p,i}.(\vec{r} = \vec{z}_{i+1}) \quad \text{(II.32)}$$

b.Erreurs qui agissent directement sur les axes du robot

C'est la somme de l'ensemble des erreurs des actionneurs, freins, capteurs sur les axes et les déformations élastiques des transmissions causées par les chargements qui génèrent aussi des vibrations. La précision des composants est fournie par le fabricant. La mise en équation se fait par les trois descriptions du modèle géométrique.

b.1 Les erreurs sur un axe motorisé en rotation

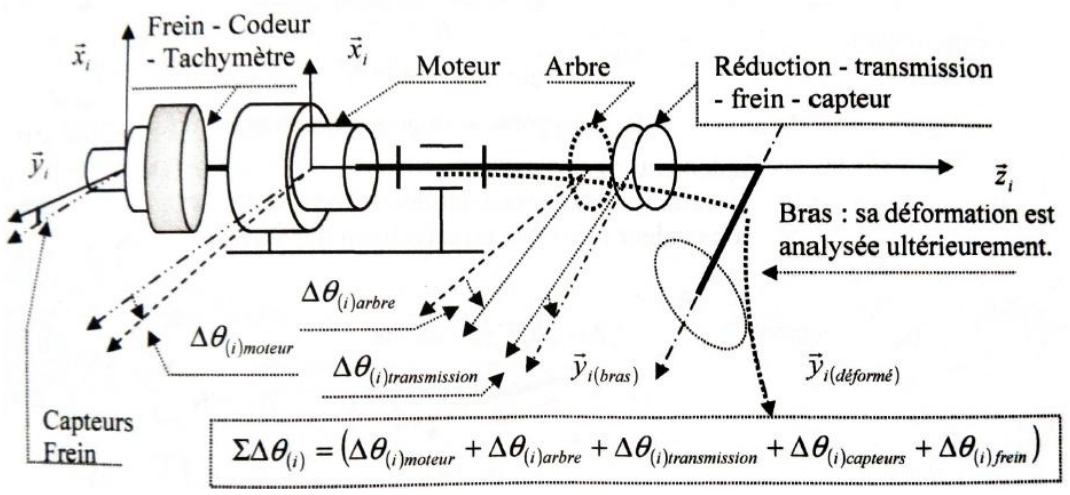


Figure II.12: Les erreurs sur un axe motorisé en rotation

$$\Sigma\Delta\theta_{(i)} = (\Delta\theta_{(i)moteur} + \Delta\theta_{(i)arbre} + \Delta\theta_{(i)transmission} + \Delta\theta_{(i)capteurs} + \Delta\theta_{(i)\,frein} \quad \text{(II.33)}$$

$\Delta\theta_{(i)moteur}$: Erreur angulaire du moteur, glissement, perte de pas, retard…

$\Delta\theta_{(i)capteurs}$: Erreur angulaire des capteurs, tachymètre, résolution, temps de réponse …

$\Delta\theta_{(i)\,frein}$: Temps de réponse au blocage, glissement …

$\Delta\theta_{(i)arbre}$: Erreur due à la déformation élastique de l'arbre.

$\Delta\theta_{(i)transmission}$: Erreur du réducteur et de la transmission, déformations sous charge :

Erreurs au niveau des accouplements, jeux, déformations locales aux efforts, contacts

Entre dentures, pression de Hertz, flexion des dents, déformations de la transmission

… On obtient les valeurs par essais.

La matrice homogène de l'axe $\vec{z}_i$:

$$\begin{bmatrix} C(\theta_i+\sum\Delta\theta_i) & -S(\theta_i+\sum\Delta\theta_i) & 0 & 0 \\ S(\theta_i+\sum\Delta\theta_i) & C(\theta_i+\sum\Delta\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{(Bi)} \quad \text{(II.34)}$$

b.2 Erreurs sur l'axe d'une motorisation linéaire

b.2.1 Les vérins usuels –Ici l'axe est $\vec{z}$

Matrice des erreurs :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \left[\sum\Delta_{(i)Vérin.et.transmission}\right] \\ 0 & 0 & 0 & 1 \end{bmatrix}_{(Bi)} \quad \text{(II.35)}$$

Cette matrice inclut toutes les erreurs de la distribution hydraulique et du vérin.

b.2.2 Vérin linéaire, vis à billes motorisée par un moteur à courant continu ou pas à pas.

Matrice des erreurs :

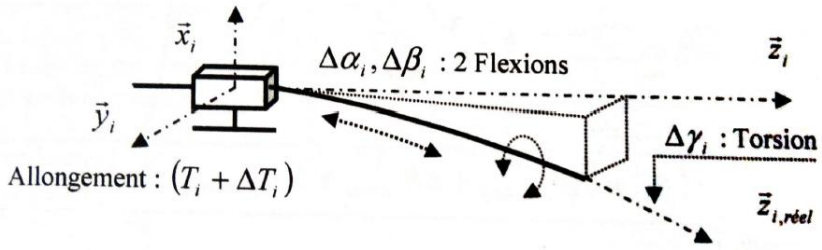
$$\begin{bmatrix} C(\theta_i+\sum\Delta\theta_{(i)m+arb}) & -S(\theta_i+\sum\Delta\theta_{(i)m+arb}) & 0 & 0 \\ S(\theta_i+\sum\Delta\theta_{(i)m+arb}) & C(\theta_i+\sum\Delta\theta_{(i)m+arb}) & 0 & 0 \\ 0 & 0 & 1 & \sum\Delta_{(i)réduction-Transmission} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.36)}$$

$\Delta l_{(i)actionneur}$: Somme de toutes les erreurs, jeux, pas, précision de l'actionneur linéaire.

$\Delta l_{(i)transmission}$: Somme de toutes les déformations, allongement, compression, flambage, jeu dans les accouplements…

Beaucoup d'erreurs sont variables et dépendent des efforts, elles sont difficilement quantifiables. On approche une valeur moyenne en effectuant des séries d'essais.

b.2.3 Glissière

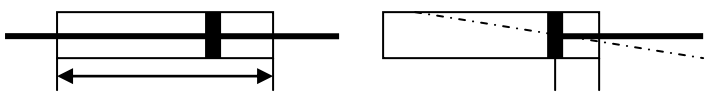


Allongement : $(T_i + \Delta T_i)$

Figure II.13: Déformation d'une tige de glissière

La longueur l_i est remplacée par la variable T_i et Δl_i par ΔT_i. Les flèches varient en fonction de la sortie de la tige longues doivent être vérifiées au flambage, soit par une des méthodes de vérifications d'Euler, de Rankine et de Dutheil.

Un vérin à deux tiges présente une meilleure précision radiale qu'un vérin à simple tige.



Déformation due à l'élasticité des joints

Déformation due à l'élasticité des joints.

b.3 Les capteurs circulaires et linéaires

On retrouve sensiblement les mêmes types d'erreurs que pour les moteurs rotatifs et linéaires, sauf les erreurs des déformations, puisque les efforts son nuls.

b.4 Les erreurs du calculateur proviennent essentiellement des algorithmes utilisés .Ils peuvent être améliorés par l'adaptation de la gestuelle à la tache imposée.

b.5 Les traitements des informations et les asservissements. Les défauts des matériels dépendant des qualités intrinsèques de chaque composant, informations fournies par le fabricant. Toute conception est faite à partir d'un cahier de charges et par comparaison avec des systèmes existants.

Des essais vérifient les caractéristiques globales. Les erreurs du calculateur, des traitements et des asservissements sont comptabilisées sur l'axe motorisé. On attribue un coefficient à chaque application.

b.6 Les transmissions longues, pour le report des moteurs en arrière, favorisent l'équilibrage et un gain sur l'encombrement, mais elles génèrent des erreurs torsionnelles et des mouvements vibratoires. Certaines erreurs sont calculées, l'ensemble est testé, une carte des erreurs est dressée, ce qui permet d'introduire des valeurs moyennes dans la matrice des erreurs .Les variations dépendent des charges, des efforts et des inerties.

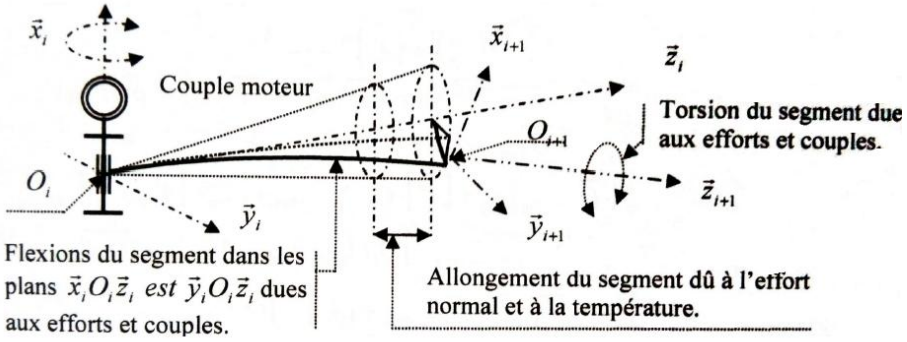
c.Les déformations thermiques

La variation de longueur : $\Delta l = l.\beta.\Delta t^0$, $\beta =$ Coefficient de dilatation du matériau.

II.3.1.4. Matrices des erreurs d'un segment et d'une chaine articulée

a. Un segment

Les déformations, par ordre d'importance, sont celles de la flexion composée, de torsion, d'allongement, celles dues aux cisaillements sont négligeables. Flexions et torsion entrainent des pivotements de la section droite et du repère en amont, en o_{i+1}



Flexions du segment dans les plans $\vec{x_i}o_i\vec{z_i}$ est $\vec{y_i}o_i\vec{z_i}$ dues aux efforts et couples.

Allongement du segment du à l'effort normal et à la température.

En $O_i \mapsto R_i = (O_i; \vec{x_i}, \vec{y_i}, \vec{z_i})$

En $O_{i+1} \mapsto R_{i+1} = (O_{i+1}; \vec{x_{i+1}}, \vec{y}_{i+1}, \vec{z}_{i+1})$

Figure II.14: Déformation d'un segment sous l'effet de sollicitations composées

Matrice complète avec erreurs et écriture simplifiées, Δ = réelle déformée,(th)=théorique

$$\left[M^{i}_{i+1}\overrightarrow{o_i o_{i+1(Bi)}}\right]^{\Delta} = \left[M_{(i)}\right]^{\Delta(\beta_i)}_{(\beta_{i+1})} = \begin{bmatrix} [A]^{\Delta(\beta_i)}_{(\beta_{i+1})} & \left[\overrightarrow{O_iO_{i+1(th)}} + \overrightarrow{\delta O_iO_{i+1}}\right]_{(Bi)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \qquad \text{(II.37)}$$

-La matrice [3 x 3] avec erreurs et le vecteur position comportent les mêmes éléments .On ne garde que les infiniment petits du premiers ordre.

-La matrice $[A]^{\Delta(\beta_i)}_{(\beta_{i+1})}$ ou $\left[A^{i}_{i+1}\right]^{\Delta}$ regroupe la variable généralisée, la somme de toutes les erreurs angulaires au niveau de l'axe, celles des rotations des sections planes dues à la flexion déviée et à la torsion en O_{i+1} On définit ainsi l'orientation du repère en O_{i+1} par rapport à celui en O_i.

-Segment réel : $\left[\overrightarrow{O_iO_{i+1(th)}} + \overrightarrow{\delta O_iO_{i+1}}\right]_{(Bi)}$ et $\left[\phi_{i+1(th)} + \delta\phi_{i+1}\right]_{(Bi)}$,la matrice (3x3) $[A]^{\Delta(\beta_i)}_{(\beta_{i+1})}$

-Segment parfait : les vecteurs : $\left[\overrightarrow{O_iO_{i+1(th)}}\right]_{(Bi)}$ et $\left[\phi_{i+1(th)}\right]_{(Bi)}$, la matrice (3x3) : $[A]^{\Delta(\beta_i)}_{(\beta_{i+1})}$

-Segment théorique : $\left[M_{i+1}^{i}\overrightarrow{o_i o_{i+1(Bi)}}\right]^{\Delta} = \left[M_{(i)}\right]_{(\beta_{i+1})}^{\Delta(\beta_i)} = \begin{bmatrix} [A]_{(\beta_{i+1})}^{\Delta(\beta_i)} & \left[\overrightarrow{O_i O_{i+1(th)}}\right]_{(Bi)} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix}$ (II.38)

- Matrices des erreurs du segment i : $\left[\Delta M_{(i)}\right]_{(\beta_{i+1})}^{(\beta_i)} = \left[M_{(i)}\right]_{(\beta_{i+1})}^{\Delta(\beta_i)} - \left[M_{(i)}\right]_{(\beta_{i+1})}^{(\beta_i)}$ Elle peut encore être obtenue par essais.

b.Une chaine articulée ouverte avec ses erreurs

-Matrice homogène : $\left[M(\overrightarrow{OP})_{(BO)}\right]^{\Delta} = \left[M_{(0)}\right]_{(BO)}^{\Delta(BO)} . \left[M_{(1)}\right]_{(B1)}^{\Delta(BO)} . \left[M_{(2)}\right]_{(B2)}^{\Delta(B1)} ..... \left[M_{(P)}\right]_{(Bp)}^{\Delta(Bp-1)}$ II.39)

La matrice $\left[M_{(0)}\right]_{(BO)}^{\Delta(BO)}$ considère les déformations de la partie fixe reliée au bâti.

-Matrices des erreurs de la structure : $\left[\Delta M(\overrightarrow{OP_{(BO)}})\right] = \left[M(\overrightarrow{OP_{(BO)}})\right]^{\Delta} - \left[M(\overrightarrow{OP_{(th)(BO)}})\right]$ (II.40)

- Matrices des erreurs de l'orientation : $\left[\Delta(A_P^0)\right] = \left[A_P^0\right]^{\Delta} - \left[A_P^0\right]$ (II.41)

-Erreur sur la position : $\Delta\overrightarrow{OP}_{(BO)} = (\overrightarrow{OP}_{(BO)}^{\Delta} - \overrightarrow{OP}_{(BO)}) = \delta\overrightarrow{OP}_{(BO)}$ (II.42)

-Théorique : $\overrightarrow{O_0P}_{(th)} = \overrightarrow{O_0O}_{1(th)} + \overrightarrow{O_1O}_{2(th)} + ... + \overrightarrow{O_nO}_{(th)}$ (II.43)

$$\overrightarrow{O_0P}_{(th)(BO)} = \left[A_0^0\right]\overrightarrow{O_0O}_{1(th)(B0)} + \left[A_1^0\right]\overrightarrow{O_1O}_{2(th)(B1)} + ... + \left[A_n^0\right]\overrightarrow{O_nP}_{(th)(Bn)} \quad \text{(II.44)}$$

-Réelle : $\overrightarrow{O_0P}_{(r)} = \overrightarrow{OP}^{\Delta} = \left[\overrightarrow{O_0O}_{1(th)} + \delta\vec{P}_{O1}\right] + \left[\overrightarrow{O_1O}_{2(th)} + \delta\vec{P}_{O2}\right] + ... + \left[\overrightarrow{O_nO}_{(th)} + \delta\vec{P}_P\right]$ (II.45)

$$\overrightarrow{OP}_{(B0)}^{\Delta} = \left[A_0^0\right]^{\Delta} . \left[\overrightarrow{O_0O}_{1(th)} + \delta\vec{P}_{O1}\right]_{(B0)} + \left[A_1^0\right]^{\Delta} . \left[\overrightarrow{O_1O}_{2(th)} + \delta\vec{P}_{O2}\right]_{(B1)} + ...$$
$$+ \left[A_n^0\right]^{\Delta} \left[\overrightarrow{O_nO}_{(th)} + \delta\vec{P}_P\right]_{(Bn)} = \delta\overrightarrow{OP}_{(B0)} \quad \text{(II.46)}$$

-Erreur sur l'orientation : $\Delta\vec{\phi}_{P(B0)} = (\vec{\phi}_{P(B0)}^{\ \Delta} - \vec{\phi}_{P(B0)})$ ou $\delta\vec{\phi}_{P(B0)}$. (II.47)

II.4.Systèmes hyperstatiques

Lorsque les articulations sont bloquées, les structures des chaines fermés et des robots parallèles deviennent hyperstatiques, alors qu'elles ne le sont pas en mouvement

II.4.1.Théorème de Ménabréa –Résolution de structures hyperstatiques

Le théorème de Castigliano dit que la projection du déplacement du point d'application d'une force sur la direction de cette force (ou la rotation d'un couple) est égale à la dérivée partielle de l'énergie de déformation par rapport à cette force (ou à ce couple).En considérant une action de contact ,le déplacement de celle-ci est nul si l'appui est fixe ou si l'action est perpendiculaire à l'appui ,appui sans frottement .Dans ces deux cas, la projection du déplacement sur la ligne d'action de la force de contact est nulle. Ainsi, dans des systèmes hyperstatiques, les inconnus superflues prennent des valeurs pour lesquelles l'énergie potentielle de déformation est minimale .Le théorème est aussi connu comme théorème de travail minimal des forces extérieures appliquées au système. On note des hyper statismes extérieurs et intérieurs.

II.4.2. Robots parallèles-Flambement des barres et des actionneurs linéaires

Pour ne pas risquer le flambage ou flambement, les barres et les actionneurs linéaires ne doivent pas avoir des tiges trop fines.

Les plateaux sont considérés indéformables, seuls les actionneurs linéaires subissent des déformations de raccourcissement, on suppose qu'il n'y ait pas de flambage, alors :

$$\sigma_{icompression} = \frac{F_i}{S_i} = E_i \frac{\Delta l_i}{l_i} \rightarrow \Delta l_i = \frac{F_i l_i}{E_i S_i}$$

(II.48)

Les F_i sont données par les équations d'équilibre du système. Ici, les articulations ne sont pas bloquées, ce sont des rotules.

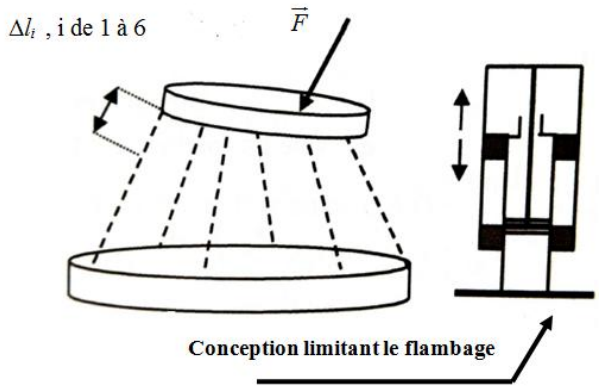


Figure II.15: Robot parallèle

Les vérifications au flambage des tiges de vérins sont présentées succinctement, elles sont suffisantes pour ces conceptions. [88]

Mode opératoire

-Longueur libre de flambage

L=longueur libre de flambage *-l* =longueur de la barre.

Pour définir la longueur libre de flambage, il faut bien analyser les liaisons des attaches

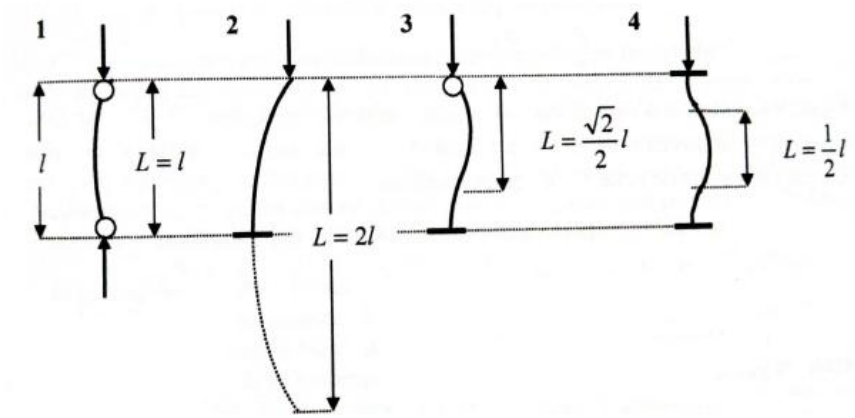


Figure II.16: Les longueurs libres de flambages

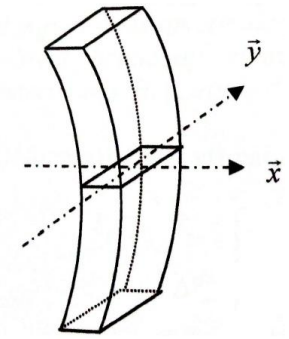
1 : Pivots ou rotules aux deux extrémités

2 : Encadrement à une extrémité et libre à l'autre

3 : Encadrement et pivot ou rotule

4 : Deux encastrements

Elancement d'une barre



Le flambage se fait selon le moment quadratique minimal et la déformation s'effectue toujours selon (+ *x*) ou (- *x*). C'est le manque d'homogénéité, d'isotropie ou un effort latéral, même minimal, qui conditionnent le sens de la déformation à droite ou à gauche. On a intérêt à utiliser des sections symétriques ayant même moment quadratique dans tous les sens.

Les rayons de giration : $r_{Gx} = \sqrt{\frac{I_{Gx}}{S}}$ et $r_{Gy} = \sqrt{\frac{I_{Gy}}{S}}$, S = surface

I_{Gx} et I_{Gy} = Moment quadratiques par rapport aux axes.

L'élancement d'une barre comprimée : $\gamma = \frac{L}{r_{G\min}}$

-méthodes pour les vérifications de flambage :

Trois méthodes sont présentées pour les vérifications

1-Méthode d'Euler-Charge critique de flambage d'Euler : $P_{CE} = \frac{\pi^2 E I_{G\min}}{L^2}$

(II.49)

Charge admise : $P \le \frac{P_{CE}}{S}$ S = Coefficient de sécurité, pour l'acier il est de 3 à 5

La formule est applicable quels que soient le moment quadratique et la longueur de la barre comprimée. Si $I_{Gaxe\min}$ est grand et L faible pour une charge critique P_{CE} élevée, la contrainte est $\sigma = \frac{P_{CE}}{S} \succ \sigma_e$ .Il faut alors faire un calcul de compression pure.

La formule d'Euler n'est valable que si $\frac{P_{CE}}{S} \prec \sigma_{e=élastique}$.

La barre serait détruite par compression pure avant apparition du flambage.

2-Méthode de Rankine :

$$\frac{P}{S} \le \frac{\sigma_w}{1+\beta\gamma^2}$$

(II.50)

Avec σ_w = Contrainte d'utilisation à la compression simple et β =0.0001 pour l'acier.

L'élancement γ intervient dans l'expression. Si l'élancement est faible, la contrainte d'utilisation est proche de celle de la compression pure.

3-Méthode Dutheil -Elle comporte 5 étapes : P= Charge et S = Section

- $\sigma = \frac{P}{S}$, $P_{CE} = \frac{\pi^2 E I_{G\min}}{L^2}$, P_{CE}= critique d'Euler $\rightarrow \sigma_{CE} = \frac{P_{CE}}{S}$

-Vérification $\sigma \prec \sigma_{CE}$ Si cela n'est pas le cas, il faut recommencer en changeant S et $I_{G\min}$

- Contrainte intermédiaire : $\sigma_I = \frac{1}{2}(\sigma_{CE} + 1{,}3\sigma_{e=\text{limite élastique}})$

-Contrainte d'affaissement : $\sigma_{aff} = \sigma_I - \sqrt{\sigma_I{}^2 - \sigma_{CE}\,\sigma_{e=\text{élas}}}$

-Vérification finale : $\sigma = \frac{P}{S} \leq \frac{2}{3}\sigma_{aff}$, avec un coefficient de sécurité de 2/3.

II.5.Protocoles de mesures

II.5.1.Précision statique d'une chaine articulée

La précision statique caractérise l'aptitude du robot à positionner et à orienter le trièdre de conformité avec la position et l'orientation programmée .Chaque erreur peut être déterminée indépendamment de l'action des autres erreurs. Pour la sommation on applique le principe de superposition. Les erreurs de la commande sont supposées aléatoires.

a.Les mesures

On suppose que le point programmé se situe à mi-course de chaque mouvement élémentaire. La précision peut dépendre du mode d'accostage. Pour s'effectuer $n \geq 10$ mesures de positions, définissant la situation du repère réel (Rr) avec des accostages différents.

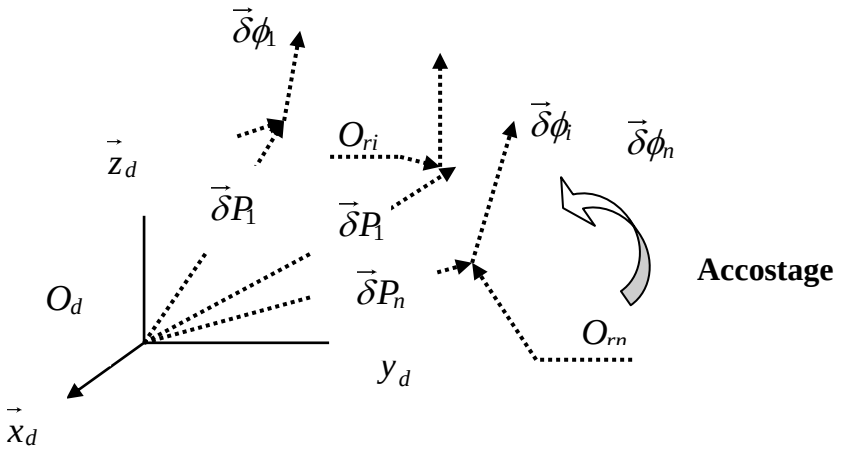


Figure II.17 : Précision statique -Famille de mesure

On utilise les deux vecteurs : orientation et position.

Indice : d=désiré, r=réel

La précision de la situation est mesurée par les moyennes des deux vecteurs :

$\left\|\delta\vec{\phi}\right\| = \frac{1}{n}.\sum_{i=1}^{n}\left\|\delta\overrightarrow{\phi_{(i)}}\right\|$ et $\left\|\delta\vec{P}\right\| = \frac{1}{n}.\sum_{i=1}^{n}\left\|\delta\overrightarrow{P_{(i)}}\right\|$ i =numéro de la mesure.

L'erreur mesurée est donnée par : $\{\varepsilon_{rot+dépl} = \begin{cases}\delta\vec{\phi}\\ \delta\vec{P}\end{cases}_{Base}$ Vecteurs définis précédemment

Ces valeurs permettent une optimisation du calibrage des robots et des convergences du système de commande pour atteindre la position et l'orientation désirées.

b. Les convergences –Les situations dans l'espace sont des vecteurs.

i = numéro de la mesure.

$\vec{\varepsilon}_i$ =erreur à la i^{éme} mesure, $\vec{\varepsilon}_{\min}$ = le minimum de l'erreur, $\vec{\varepsilon}_0$ =valeur donnée.

Pr(E) : Désigne la probabilité d'occurrence de l'événement E

-La convergence probabiliste : $\lim_{i\to\infty}\Pr\left(\left\|\vec{\varepsilon}_i - \vec{\varepsilon}_{\min}\right\| \geq \vec{\varepsilon}_0\right) = \vec{0}$

-La convergence au sens des moindres carrés : $\lim_{i\to\infty}\left(\left\|\vec{\varepsilon}_i - \vec{\varepsilon}_{\min}\right\|^2\right) = \vec{0}$

-La convergence presque sûre : $\lim_{i\to\infty}\Pr\left(\left\|\vec{\varepsilon}_i - \vec{\varepsilon}_{\min}\right\| = \vec{0}\right) = 1$

Si les composantes aléatoires des erreurs de situations ont un caractère stationnaire sur un intervalle de temps comparable à la durée du travail du robot, entre deux contrôles ou deux calibrages successifs, le processus itératif d'évaluation ou de compensation des erreurs s'arrête quand la séquence des mesures présente également un caractère stationnaire et que les variations successives de l'indice de performance sont petites .Ces méthodes conduisent à un indice de performance qui exprime globalement la qualité de positionnement et d'orientation des robots .Il est ainsi facile de comparer les performances des robots de même type, voire d'autres types. Un robot qui présente de moins bonnes performances globales peut être plus précis dans certaines situations.

II.5.2.Précision cinématique et dynamique -performances –Estimations-Répétitivité

-La précision cinématique caractérise l'aptitude du robot à faire suivre une trajectoire au trièdre de conformité selon la trajectoire programmée.

-La précision dynamique génère la trajectoire dans les mêmes conditions imposées, quels que soient les efforts extérieurs et cinétiques autorisés dans la plage définie.

On ne peut résoudre la précision dynamique que sous certaines hypothèses. La difficulté majeure provient de ce que l'on est en présence de plusieurs couples cinématiques. D'une part, il se produit des variations d'effort, des chocs dans les articulations dues aux tolérances et aux jeux, mêmes faibles. D'autre part, le mouvement n'est pas continu mais alterné, vibratoire, décrivant de petits mouvements dans ces jeux. La précontrainte des liaisons favorise la précision.

La mesure de l'indice des performances dynamiques peut être réalisée de manière analogue aux méthodes utilisées en statique, elle est complétée par des mesures de vitesses, de suivis de trajectoires sans et avec effort extérieurs et cinétiques. Les mouvements du repère programmé et du repère réel dépendent du temps : $R_{d=désiré}(t)$ et $R_{r=réel}(t)$. La nature des trajectoires et des lois cinématiques est liée à la tache. Selon la précision de la trajectoire, les mesures sont réalisées sur des segments, des intervalles de temps plus ou moins petits, au moyen des caractéristiques de la matrice Jacobienne de commande variationnelle, ou par mesures continues des composantes du torseur cinématique

Exemples

-Les robots de soudure par points, d'insertion demandent une précision statique.

-Les robots de dépose de cordon de soudure, de colle, de traçage de découpage au jet d'eau et laser nécessitent une précision cinématique.

-Les robots de manipulation précise, de montage avec efforts, d'assemblage, requièrent une précision statique avec adaptation et une déformation minimale de la structure.

-Les robots rapides exigent une précision dynamique associant la précision cinématique à celle de la maitrise des efforts extérieurs et cinétiques.

a. Estimation de la précision des robots

Les erreurs primaires, technologiques, jeux, de commandes sont toujours représentées par des variables aléatoires indépendantes avec des lois de répartition connues. On construit un spectre d'estimation de la précision qui caractérise ces erreurs en utilisant des modèles mathématiques, des moyennes quadratiques, des probabilités, des estimations d'erreurs…Il existe aussi des erreurs primaires d'origine méthodologique qui proviennent de l'insuffisance des capacités technologiques, de connaissances ou d'estimations.

b.La répétitivité ou précision de répétition

Elle est caractérisée par des écarts entre les positions et les orientations du trièdre lié au terminal par rapport au trièdre de référence au cours de nombreux cycles identiques .Elle est exprimée par des valeurs numériques par rapport aux écarts types des erreurs.

b.1.La répétitivité statique

$$\text{Répétitivité statique} = \sum_{i=1}^{n} (\vec{P}^{*} - \vec{P}_i) = \vec{0}$$

$$\vec{P}^{*} = \lim_{n\to\infty} \frac{1}{n} . \sum_{i=1}^{n} \vec{P}_i$$

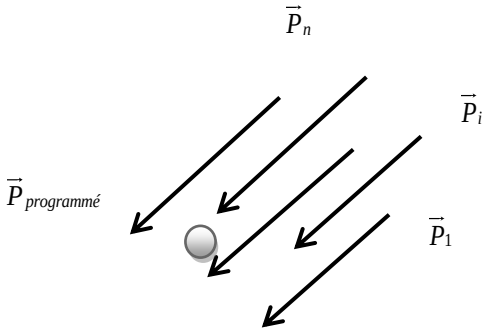


Figure II.18 : répétitivité statique

b.1.La répétitivité dynamique

$$\text{Répétitivité} = 3 \lim_{n\to\infty} \sqrt{\frac{\sum_{i=1}^{n} \left\| \vec{P}_i - \vec{P}^{*} \right\|}{n-1}}$$

Avec $\vec{P}^{*} = \lim_{n\to\infty} \frac{1}{n} . \sum_{i=1}^{n} \vec{P}_i$

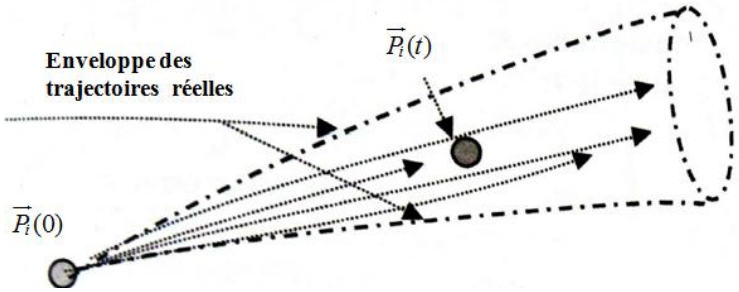


Figure II.19 : répétitivité dynamique

La répétitivité statique est plus précise que la dynamique. On recommande d'effectuer les mesures de répétitivité sur les trajectoires dont le suivi exige, pour chaque articulation du robot, un débattement maximal compatible avec les butées.

II.5.3.Les méthodes de mesures usuelles-Qualités des mesures

La mesure des écarts de situation du terminal d'un robot n'est qu'une des composantes de l'évaluation de ses performances, le robot étant générateur de mouvements et de forces /couples. Les variations de précision dues à l'infinité de situations possibles engendrées par un robot sont englobées dans un critère général de type probabiliste, des spectres de trajectoires visualisent les mouvements.

Les mesures doivent être réalisées :

-pour différentes charges allant du minimum à vide au maximum de charge.

-avec et sans connaissance des centres de masse et des caractéristiques cinétiques.

-à des vitesses variables avec contraintes d'accélération et décélération importantes.

-en régime stable, en enceinte isotherme pour se libérer des variations de température.

a-.Mesure absolues ou relatives

Les situations sont mesurées dans un repère de référence R_0, puis on mesure les situations du terminal ou les écarts de situations dans plusieurs repères fixes, dont on connait parfaitement les situations par rapport à R_0.

b-.Mesure absolues

Elles peuvent être faites par comparaison ou non. Un générateur de situation étalon, machine de mesure tridimensionnelle ou robot de mesure, est utilisé. Le terminal de cet étalon est déplacé en conformité avec celui du robot, de manière à pouvoir calculer ou comparer

numériquement les situations. Se référer aussi au contrôle des machines outils.

c-.Mesure avec ou sans contact entre le terminal du robot et l'environnement

Le choix des capteurs est fonction des mesures : capteurs de proximité, codeurs, micro rupteurs, systèmes industriels, accéléromètres, capteurs à ultra sons, infrarouges, à ondes électromagnétiques, par courant de Foucault, interféromètres, laser. Ces systèmes sont embarqués sur le robot et/ou implantés dans l'environnement.

d-Qualités exigées par les méthodes de mesures

La précision, la rapidité, la disponibilité, l'évaluation en ligne, la simplicité de mise en œuvre, la facilité d'exploitation des résultats par les asservissements et le calculateur, l'adaptabilité aux contraintes, elles doivent être autocorrectives, robustes, de maintenance facile, insensibles aux influences et bruits extérieurs, non dépendantes d'autres systèmes. Les mesures sont redondantes pour confirmer les résultats. Il faut définir avec précision les modes opératoires et les référentiels fixes ou mobiles utilisés, voir les normes.

II.6. Catégories d'étalonnage

Selon les sources d'imprécision à compenser, Elatta *et al.* (2004) présentent deux grandes familles d'étalonnage des robots, soit l'EG, destiné à améliorer le modèle cinématique du robot et l'étalonnage non géométrique qui vise à traiter les erreurs d'origine non géométrique. Quant aux erreurs articulaires, elles sont généralement incluses dans l'EG.

II.6.1. Etalonnage articulaire (niveau 1)

Selon Roth *et al.* (1987), aux deux catégories d'étalonnage citées précédemment, s'ajoute l'étalonnage dit de niveau 1. Celui-ci consiste à compenser les erreurs des articulations actives du robot, i.e. modéliser la relation entre les valeurs affichées par les capteurs (théoriquement les consignes) et les déplacements réels des articulations correspondantes par rapport à leurs positions zéro (*home position*). Cependant, dans la majorité des travaux de recherche, ces erreurs sont traitées dans l'EG (niveau 2). A titre d'exemple, Durango *et al.* (2010) proposent une méthode d'EG d'un robot parallèle cinq-barres qui compense aussi les erreurs articulaires. Celles-ci ont été, modélisées par l'introduction, dans le modèle cinématique, d'un gain et un offset à leurs valeurs théoriques.

Par conséquent, certains travaux de recherche procèdent uniquement à l'étalonnage de niveau 1 plusieurs d'entre eux n'identifient que les positions zéro des actionneurs (*home positions*), plus fréquemment si les actionneurs du robot sont de haute précision : i.e.

n'utilisent pas de réducteurs de vitesse et sont dotés d'encodeurs de très bonne résolution. Dans ce contexte, Ding *et al.* (2005) présentent l'étalonnage d'un robot planaire à deux ddl, utilisant des servomoteurs sans boîtes d'engrenages, en corrigeant uniquement les erreurs des positions zéro des articulations. Ces mêmes erreurs faisaient l'objet du travail d'étalonnage de niveau 1 de Zhang *et al.* (2007) mené sur un robot parallèle redondant à deux ddl. Plus récemment, l'identification de la position zéro des articulations a été effectuée par Chen *et al.*(2008) pour améliorer la précision d'un robot sérial à six ddl.[89]

II.6.2. Etalonnage géométrique (niveau 2)

Cette approche appelée aussi étalonnage cinématique considère que les liens du robot sont parfaitement rigides et que celui-ci n'est pas en mode dynamique, c.-à-d. sans considérer l'impact des mouvements du robot sur sa précision. Par conséquent, l'opération d'étalonnage consiste à introduire des corrections uniquement sur le modèle cinématique du robot, en identifiant les valeurs de ses paramètres.

Cette méthode apporte une meilleure amélioration de la précision pour les robots parallèles, car leurs erreurs non géométriques sont assez faibles, vu la rigidité élevée de leurs structures.

Ceci justifie le fait que la majorité des travaux de recherche destinés à l'étalonnage de cette catégorie de robots se base sur des approches géométriques, par exemple : Wu *et al.* (1988), Oliviers *et al.* (1995), Masory *et al.* (1997), Zhuang *et al.* (1998) et Gatla *et al.* (2007).[33]

II.6.3. Etalonnage non géométrique (niveau 3)

Ce type d'étalonnage, aussi appelé étalonnage non cinématique, est utilisé pour compenser les erreurs non géométriques. Il consiste à développer des modèles mathématiques pour compenser ces erreurs. Ces modèles sont souvent très complexes et selon Judd et Knasinski (1990) et Damak (1996), ils apportent peu d'amélioration à la précision, comparativement à l'EG. Ceci pourrait expliquer la rareté des travaux de recherche effectués dans ce domaine.[88]

Néanmoins, effectuer un étalonnage non géométrique après avoir effectué un étalonnage cinématique pourrait permettre d'atteindre des précisions élevées chez les robots sériels, notamment (ces derniers étant davantage affectés par les erreurs non géométriques).

Parmi les études disponibles, citons le modèle proposé par Everett (1993), qui se base sur la modélisation globale des erreurs non géométriques en utilisant les séries de Fourier.

D'autres travaux sont effectués pour modéliser des erreurs spécifiques, comme celles des systèmes d'engrenage (Judd et Knasinski, 1990) ou encore les erreurs de torsion des liens (Damak, 1996). Gong *et al.* (2000) présentent une compensation des effets de la température sur la précision d'un robot sériel à six ddl, en utilisant un modèle empirique d'erreurs thermiques.[90]

Dans d'autres travaux, Oiwa (2002, 2005) procède à la compensation des effets de déformation causés par la température et par des forces externes sur les liens et la base d'un robot parallèle. Mentionnons ici que plusieurs travaux de recherche traitent exclusivement de l'EG et ne couvre pas les erreurs non géométriques. Ainsi, pour simplifier le texte, le terme étalonnage fera dorénavant référence à l'étalonnage géométrique et le terme paramètres, aux paramètres géométriques.

II.7. Conclusion

Nous avons examiné dans ce chapitre les sources d'erreurs qui affectent les robots manipulateurs industriels par l'analyse mathématique qui aide à dégager les paramètres qui influent sur la précision et répétabilité du robot sur lequel on agit pour améliorer la précision globale de positionnement et utiliser les méthodes d'étalonnage adaptées pour compenser les sources d'imprécision.

Partant de la connaissance des résultats de l'analyse mathématique, nous abordons dans les chapitres suivants le calcul numérique de l'erreur de précision et de l'orientation par un programme interprété dans une fenêtre graphique qui doit calculer et afficher l'erreur ensuite on utilisera trois méthodes pour extraire l'effet des paramètres géométriques et comparer avec le calcul analytique mathématique.

Chapitre III

Le robot parallèle planaire 3RPR

III.1. Description

Le 3-RPR est un robot parallèle planaire à trois degrés de libertés (deux translations dans un plan et Une orientation autour de l'axe orthogonal au plan).Des applications potentielles de ce robot (pick –and –place) comprend l'usinage de surface plane, une base mobile pour des manipulateurs spatiaux et une plateforme mobile [67].

Son architecture parallèle est composée de trois articulations prismatiques motorisées (actives) notées P et six articulations rotoïdes passives notées R, placées respectivement aux points fixes Ai (i=1,3) et aux points mobiles, sommets Pi (i=1,3) de la plateforme triangulaire. Les articulations passives sont couplées au robot de façon à créer des contraintes de telle sorte que le robot reste dans son plan de travail rotoïdes.

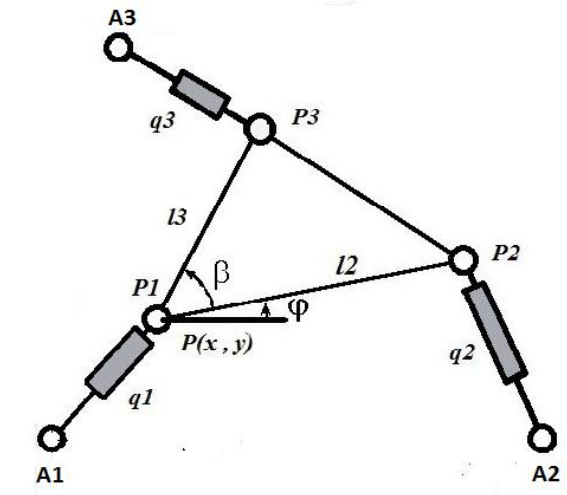


Figure III.1 : Présentation générale du robot parallèle planaire 3-RPR.

III. 2. Problème du géométrique inverse

Soit $X = \begin{bmatrix} x & y & \varphi \end{bmatrix}^T$, le vecteur des coordonnées opérationnelles (3 degrés de liberté DDL du robot) qui correspondent à la position $\begin{bmatrix} x & y \end{bmatrix}^T$ du centre géométrique P_1 du repère de la plateforme et à l'orientation de cette dernière par rapport à l'axe des x du repère de base situé en A_1, choisi pour des raisons de facilité de formulation mathématique, comme origine du repère de référence.

IL est cependant clair que dans le robot parallèle réel, l'effecteur est situé au centre de gravité **C** de la plateforme, dont la position peut être calculée aisément à partir de la connaissance des coordonnées des sommets de la plateforme. De la même manière, on pourra toujours définir l'orientation de la plateforme comme l'angle formé entre une direction fixe arbitraire (axes y du repère de base par exemple) et une direction $\mathbf{CP_i}$ (i=1,3) quelconque.

Soit $q = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^T$ le vecteur des variables articulaires correspondant aux élongations des articulations actives des jambes $i\,(i = 1,2,3)$. La modélisation géométrique du robot a pour but de déterminer les relations entre les coordonnées opérationnelles x et les variables articulaires q.

L'expression $X = F(q)$ exprime le modèle géométrique direct. Quant à l'expression

$q = H(X) = F^{-1}(q)$**,** elle correspond au modèle géométrique inverse.

A partir du schéma du robot 3-RPR (figure III.2), on peut déduire la solution au problème géométrique inverse.

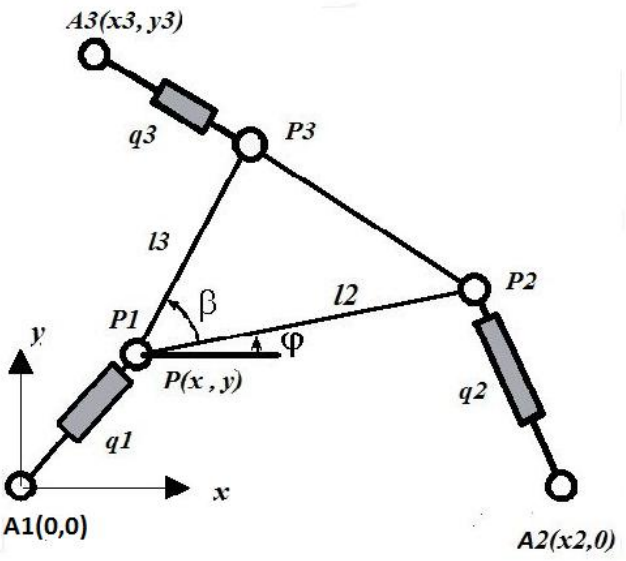


Figure III.2 : Représentation schématique du robot 3 RPR

Elle s'écrit facilement de la manière suivante :

$$q_1 = \sqrt{x^2 + y^2} \qquad \text{(III.1)}$$

$$q_2 = \sqrt{x + l_2 \cos\varphi - x_2)^2 + (y + l_2 \sin\varphi)^2} \quad \text{(III.2)}$$

$$q_3 = \sqrt{x + l_3 \cos(\varphi + \beta) - x_3)^2 + (y + l_3 \sin(\varphi + \beta) - y_3)^2} \quad \text{(III.3)}$$

III.3. Problème cinématique inverse

Soit $X = [x \ \ y \ \ \theta]^T$ le vecteur contenant les vitesses linéaires du centre C de la plateforme mobile et la vitesse angulaire autour de l'axe **z** de cette dernière. Soit $\dot{q} = [\dot{q}_1 \ \ \dot{q}_2 \ \ \dot{q}_3]^T$ le vecteur des vitesses des jambes du robot (articulations actives). La relation entre les vitesses articulaires et opérationnelles s'établit à l'aide de la matrice jacobienne notée J. L'expression des vitesses opérationnelles en fonction des vitesses articulaires $\dot{X}$, s'écrit $\dot{X} = J\dot{q}$ et elle correspond au modèle cinématique direct du robot. L'expression des vitesses articulaires $\dot{q}$ en fonction des vitesses opérationnelles $\dot{X}$ **,** correspondant au modèle cinématique inverse, qui s'écrit $\dot{q} = J^{-1}\dot{X}$ dans le cas où J^{-1} est inversible (non singulière).

Pour la plupart des robots parallèles, la résolution du problème cinématique inverse est aisée. **C**haque ligne de la matrice jacobienne inverse J^{-1}**,** est associée à une jambe du robot parallèle. La figure (**III.**3) montre une plate-forme mobile liée à une base fixe par l'intermédiaire d'une jambe $A_i P_i$

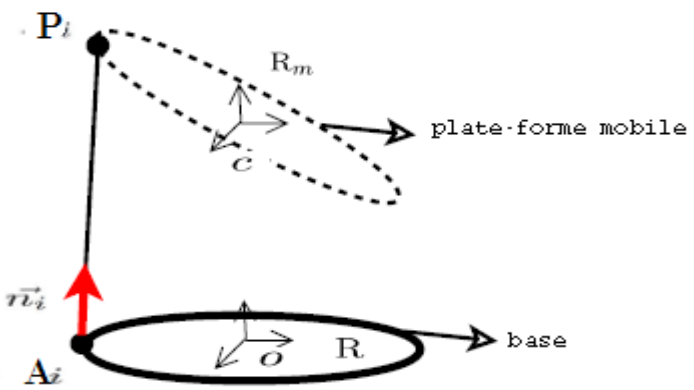


Figure III.3 : Schéma générique d'un robot parallèle.

On écrit l'équation vectorielle associée à la jambe i comme suit :

$$\overrightarrow{A_i P_i} = q_i \vec{n}_i \quad \text{i=(1,...k)} \quad \text{(III.4)}$$

Où k est le nombre d'articulations prismatiques du robot et $\vec{n_i}$ le vecteur unitaire de la jambe, donné par :

$$n_i = \frac{\overrightarrow{A_i P_i}}{\left\| A_i P_i \right\|} \qquad \text{(III.5)}$$

Considérons la plate-forme mobile en mouvement par rapport à un repère de base R. Le repère R_m est attaché à la plate-forme. Dans le cas du manipulateur 3-RPR, la vitesse du point P_i, est donnée par :

$$V_{P_i/R} = \left. \frac{d\overrightarrow{A_i P_i}}{dt} \right| R = q_i \vec{n_i} + \vec{\Omega}(J_i \setminus R) \wedge q_i \vec{n_i} \qquad \text{(III.6)}$$

Où $\vec{\Omega}(J_i \setminus R)$ est le vecteur de vitesse angulaire de la jambe i dans le repéré fixe R. Le point A_i est fixe, alors $V_{A_i/R} = 0$.

On peut faire le lien entre les vitesses des points P_i et C par la loi de transport suivante :

$$V_{P_i/R} = \vec{V}_{C/R} + \vec{\Omega}(R_m \setminus R) \wedge C_i \vec{P_i}$$

(III.7)

Où $\vec{\Omega}(R_m \setminus R)$ est le vecteur de vitesse angulaire de la plate-forme par rapport au repère fixe R de la base.

On a alors :

$$\dot{q} = \vec{V}_{C/R}.\vec{n_i} + (\vec{\Omega}(R_m \setminus R) \wedge C_i \vec{P_i}).\vec{n_i}$$

(III.8)

Le produit mixte (vectorielle/ scalaire) est commutatif, ainsi on peut écrire :

$$\dot{\vec{q}} = \vec{n_i}.\vec{V}_{C/R} + (\overrightarrow{CP_i} \wedge \vec{n_i}).\vec{\Omega}(R_m \setminus R) \qquad \text{(III.9)}$$

En appliquant l'équation générique (III.5) sur le robot planaire 3-RPR, on a alors :

$$\overrightarrow{n_1} = \begin{pmatrix} \dfrac{\overrightarrow{A_1P_1}}{\|A_1P_1\|}x \\ \dfrac{\overrightarrow{A_1P_1}}{\|A_1P_1\|}y \\ \dfrac{\overrightarrow{A_1P_1}}{\|A_1P_1\|}\varphi \end{pmatrix} = \begin{pmatrix} \dfrac{x}{q_1} \\ \dfrac{y}{q_1} \\ 0 \end{pmatrix} \qquad \text{(III.10)}$$

$$\overrightarrow{n_2} = \begin{pmatrix} \dfrac{x + l_2\cos\varphi - x_2}{q_2} \\ \dfrac{y + l_2\sin\varphi}{q_2} \\ 0 \end{pmatrix} \qquad \text{(III.11)}$$

$$\overrightarrow{n_3} = \begin{pmatrix} \dfrac{x + l_3\cos(\varphi+\beta) - x_3}{q_3} \\ \dfrac{y + l_3\sin(\varphi+\beta) - y_3}{q_3} \\ 0 \end{pmatrix} \qquad \text{(III.12)}$$

$$\overrightarrow{CP_1} = 0\,; \quad \overrightarrow{CP_2} = \begin{pmatrix} l_2\cos\varphi \\ l_2\sin\varphi \\ 0 \end{pmatrix}; \quad \overrightarrow{CP_3} = \begin{pmatrix} l_3\cos(\varphi+\beta) \\ l_3\sin(\varphi+\beta) \\ 0 \end{pmatrix} \qquad \text{(III.13)}$$

D'après (III.9), le modèle cinématique inverse du 3-RPR exprimé sous forme matricielle, s'écrit :

$$\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = \left(\begin{array}{c|c|c} \dfrac{x}{q_1} & \dfrac{y}{q_1} & 0 \\ \dfrac{x + l_2c\varphi - x_2}{q_2} & \dfrac{y + l_2s\varphi}{q_2} & \dfrac{l_2yc\varphi - l_2xs\varphi + l_2x_2s\varphi}{q_2} \\ \dfrac{x + l_3c(\varphi+\beta) - x_3}{q_3} & \dfrac{x + l_3s(\varphi+\beta) - y_3}{q_3} & \dfrac{(l_3y - l_3y_3)c(\varphi+\beta) + (l_3x_3 - l_3x)s(\varphi+\beta)}{q_3} \end{array}\right) \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\varphi} \end{pmatrix} \qquad \text{(III.14)}$$

Où :

$$J^{-1}=\begin{pmatrix} \frac{x}{q_1} & \frac{y}{q_1} & 0 \\ \frac{x+l_2c\varphi-x_2}{q_2} & \frac{y+l_2s\varphi}{q_2} & \frac{l_2yc\varphi-l_2xs\varphi+l_2x_3s\varphi}{q_2} \\ \frac{x+l_3c(\varphi+\beta)-x_3}{q_3} & \frac{x+l_3s(\varphi+\beta)-y_3}{q_3} & \frac{(l_3y-l_3y_3)c(\varphi+\beta)+(l_3x_3-l_3x)s(\varphi+\beta)}{q_3} \end{pmatrix} \quad \text{(III.15)}$$

III.4. Problème cinématique direct

La matrice J^{-1} étant carrée, elle est inversible lorsqu'elle est de plein rang. Cela nous permet par la suite de déduire l'expression de la matrice jacobienne directe qui intervient dans le modèle cinématique direct. On a:

$$\dot{q}=J^{-1}\dot{X} \quad \text{(III.16)}$$

Alors la matrice jacobienne intervenant dans le modèle cinématique direct est donnée par :

$$\dot{X}=J\dot{q} \quad \text{(III.17)}$$

J étant l'inverse de J^{-1}.

III. 5. Espace de travail :

L'espace de travail correspond à l'ensemble des poses (positions et orientations) que le point caractéristique de l'effecteur (organe terminal) peut atteindre. L'espace de travail d'un robot manipulateur est considéré comme une caractéristique primordiale pour les différentes applications en robotique, il est utilisé pour analyser les performances des robots manipulateurs et également pour la conception optimale. L'espace de travail est déterminé par les paramètres géométriques du robot manipulateur tels que les longueurs des éléments, ainsi que les limites articulaires des liaisons motorisées et passives. Du point de vue industriel, l'espace de travail d'un robot manipulateur est une donnée primordiale et fondamentale. En effet, quand une trajectoire est programmée, nous devons vérifier que toutes les poses se situent à l'intérieur de l'espace de travail du robot manipulateur. Dans d'autres cas plus complexes, tels que l'évitement des collisions, beaucoup de travaux de recherche traitent de l'analyse de l'espace de travail d'un robot manipulateur. Ces travaux présentent différentes méthodes et techniques pour déterminer, représenter, et caractériser les différents types d'espaces de travail.

Pour notre robot c'est la zone opératoire du robot où il est possible de fixer et la position $p(x, y)$ et l'orientation φ. Elle est obtenue en traçant les trois cercles centrés sur les points fixes A_i, et ayant pour rayons respectifs, les trois coordonnées articulaires q_1, q_2 et q_3.

Sur la fenêtre graphique, cette procédure a été automatisée, en sélectionnant le menu Plan de travail puis le sous-menu Surface de travail. Le logiciel affiche alors un éditeur qui permet d'introduire pour chaque articulation les intervalles de variation.

Le domaine délimité par les trois cercles définit ainsi l'espace de travail;

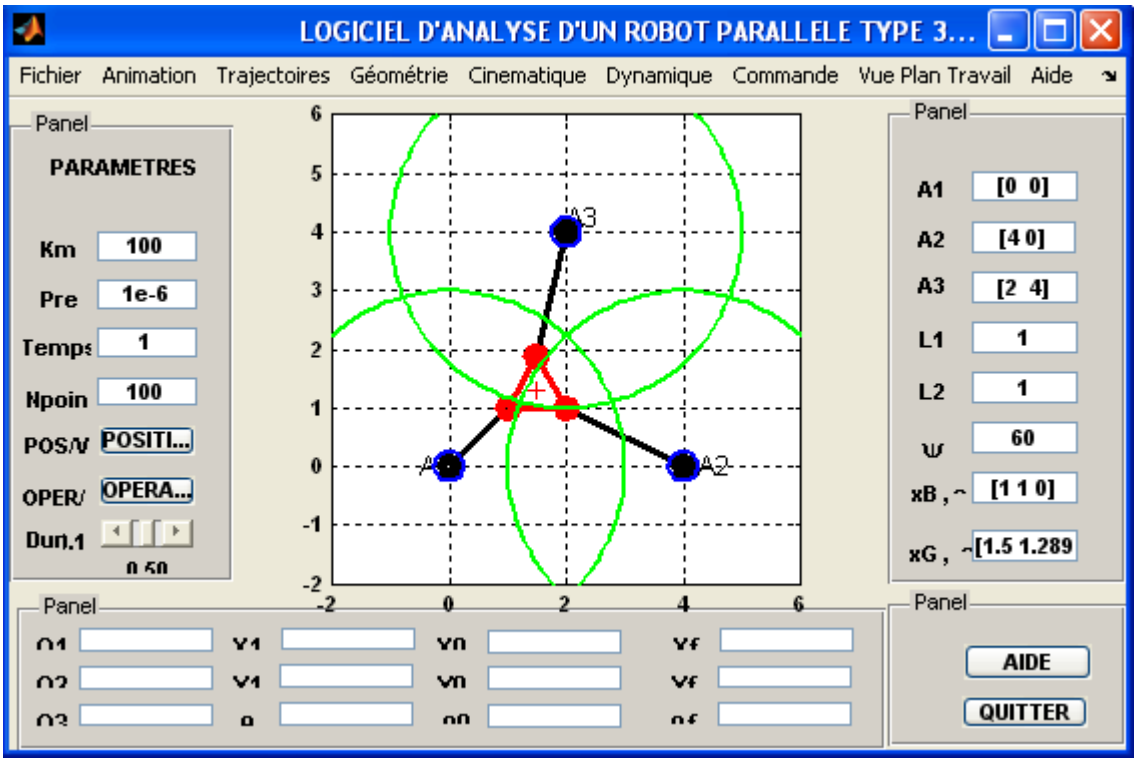


Figure III.4 : Vue de l'Espace de travail

III. 6.Modèle géométrique direct

III.6.1. Solution polynomiale

On peut résoudre le problème géométrique direct en utilisant la méthode analytique proposée par [GOS88]. On exprime les équations de fermeture de boucle par l'expression de l'équation

$$\|A_i\ P_i\ \| = q_i^2 \quad \text{i=(1,3)} \qquad \text{(III.18)}$$

A partir du système d'équations (2.1), les équations du problème géométrique inverse peuvent être écrites de la façon suivante :

$$q_1^2 = x^2 + y^2$$

(III.19)

$$q_2^2 = (x + l_2 \cos\varphi - x_2)^2 + (y + l_2 \sin\varphi)^2$$

(III.20)

$$q_3^2 = (x + l_3 \cos(\varphi + \beta) - x_3)^2 + (y + l_3 \sin(\varphi + \beta) - y_3)^2 \quad \text{(III.21)}$$

On peut écrire le système d'équations (**III.**19- **III.**20) sous les formes simplifiées :

$$q_1^2 = x^2 + y^2 \quad \text{(III.22)}$$

$$q_2^2 = x^2 + y^2 + Qx + Ry + C \quad \text{(III.23)}$$

$$q_3^2 - q_1^2 = Ux + Vy + F \quad \text{(III.24)}$$

Avec les variables intermédiaires :

$$Q = 2l_2 \cos\varphi - 2x_2 \quad \text{(III.25)}$$

$$R = 2l_2 \sin\varphi \quad \text{(III.26)}$$

$$C = x_2^2 + l_2^2 - 2l_2 x_2 \cos\varphi \quad \text{(III.27)}$$

$$U = 2l_3 \cos(\varphi + \beta) - 2x_3 \quad \text{(III.28)}$$

$$V = 2l_3 \sin(\varphi + \beta) - 2y_3 \quad \text{(III.29)}$$

$$F = x_3^2 + y_3^2 + l_3^2 - 2l_3 x_3 \cos(\varphi + \beta) - 2l_3 y_3 \sin(\varphi + \beta) \quad \text{(III.30)}$$

On simplifie encore le système d'équation (III**.**21- III**.**23) en substituant la première équation dans les deux autres équations :

$$q_1^2 = x^2 + y^2 \quad \text{(III.31)}$$

$$q_2^2 - q_1^2 = Qx + Ry + C$$

(III.32)

$$q_3^2 - q_1^2 = Ux + Vy + F \quad \text{(III.33)}$$

On résout les deux équations linéaires tirées des deux dernières équations du système (III**.**32-III**.**33).

$$Qx + Ry = S \quad \text{(III.34)}$$

$$Ux + Vy = W \quad \text{(III.35)}$$

Avec :

$$S = q_2^2 - q_1^2 - C = q_2^2 - q_1^2 - l_2^2 - x_2^2 + 2l_2x_2 \quad \text{(III.36)}$$

$$W = q_3^2 - q_1^2 - F = q_3^2 - q_1^2 - l_3^2 - x_3^2 - y_3^2 + 2l_3x_3\cos(\varphi+\beta) + 2l_3y_3\sin(\varphi+\beta) \quad \text{(III.37)}$$

On obtient la solution (x, y)

$$x = \frac{SV - RW}{QV - RU} = \frac{D_x}{D} \quad \text{(III.38)}$$

$$y = \frac{QW - SU}{QV - RU} = \frac{D_y}{D} \quad \text{(III.39)}$$

Avec :

$$D = (\sin(\varphi)(l_2x_3 - l_3x_2\cos(\beta) - l_3y_2\sin(\beta)) - \cos(\varphi)(l_2y_3 + l_3x_2\sin(\beta)) + x_2y_3 + l_2l_3\sin(\beta)) \quad \text{(III.40)}$$

$$\begin{aligned} D_x = {} & 2(q_2^2 - q_1^2)(-y_3 + l_3\sin(\varphi+\beta)) - l_2\sin(\varphi)(q_3^2 - q_1^2) + (x_2^2 + l_2^2) \\ & (y_3 - l_3\sin(\varphi+\beta)) + l_2\sin(\varphi)(x_3^2 + y_3^2 + l_2^2) - 2x_2l_2y_3\cos(\varphi) + \\ & \cos^2(\varphi)2l_2l_3x_2\sin(\beta) + \sin^2(\varphi)2l_2l_3(x_3\sin(\beta) - y_3\cos(\beta)) + \\ & l_2l_3\sin(2\varphi)(x_2\cos(\beta) - x_3\cos(\beta) - y_3\sin(\beta))) \end{aligned} \quad \text{(III.41)}$$

$$\begin{aligned} D_y = {} & 2(q_2^2 - q_2^2)(x_2 - l_2\cos(\varphi+\beta)) + (q_2^2 - q_2^2) + \\ & (-x_2 + l_2\cos(\varphi)) + (x_2^2 + l_2^2)(-x_3 + l_3\cos(\varphi+\beta)) + \\ & (x_3^2 + y_3^2 + l_3^2)(x_2 - l_2\cos(\varphi)) + 2x_2x_3l_2\cos(\varphi) - \\ & 2x_2x_3l_3\cos(\varphi+\beta) - 2x_2y_3l_3\sin(\varphi+\beta) + \cos(\varphi) - \\ & 2x_2x_3l_3\cos(\varphi+\beta) - 2x_2y_3l_3\sin(\varphi+\beta) + \\ & \cos^2(\varphi)2l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta)) + \\ & l_2l_3\sin(2\varphi)(y_3\cos(\beta) + x_2\sin(\beta) - x_3\sin(\beta))) \end{aligned} \quad \text{III.42)}$$

Par la suite, on applique la substitution classique suivante :

$$t = \tan(\frac{\varphi}{2}) \ ; \ \cos(\varphi) = \frac{1-t^2}{1+t^2} \ ; \quad \sin(\varphi) = \frac{2t}{1+t^2} \quad \text{(III.43)}$$

Soit après factorisation, on obtient le polynôme du second degré en t

$$D = 4(c_2 t^2 + c_1 t + c_0) \quad \text{(III.44)}$$

$$D = 4((l_2 + x_2)(y_3 + l_3 \sin(\beta))t^2 + 2(l_2 x_3 - x_3 l_3 \cos(\beta))t + (l_3 \sin(\beta) + y_3)(l_2 - x_2)) \quad \text{(III.45)}$$

$$c_2 = (l_2 + x_2)(y_3 + l_3 \sin(\beta)) \quad \text{(III.46)}$$

$$c_1 = 2(l_2 x_3 - l_3 x_2 \cos(\beta)) \quad \text{(III.47)}$$

$$c_0 = (l_3 \sin(\beta) + y_3)(l_2 - x_2) \quad \text{(III.48)}$$

De la même manière, on exprime D_x et D_y :

$$D_x = 2(a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0) \quad \text{(III.49)}$$

$$a_4 = (y_3 + l_3 \sin(\beta))(l_2^2 + 2l_2 x_2 + q_1^2 - q_2^2 + x_2^2) \quad \text{(III.50)}$$

$$a_3 = 2l_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) + 2l_3 \cos(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) + 3l_2 l_3(x_3 \cos(\beta) - x_2 \cos(\beta) + y_3 \sin(\beta)) \quad \text{(III.51)}$$

$$a_2 = 2y_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) + 8l_2 l_3(y_3 \cos(\beta) - x_3 \sin(\beta)) - 2l_2 l_3 x_2 \sin(\beta) \quad \text{(III.52)}$$

$$a_1 = 2l_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) + 2l_3 \cos(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) - 3l_2 l_3(x_3 \cos(\beta) - x_2 \cos(\beta) + y_3 \sin(\beta) \quad \text{(III.53)}$$

$$a_0 = (y_3 - l_3 \sin(\beta))(l_2^2 - 2l_2 x_2 + q_1^2 - q_2^2 + x_2^2) \quad \text{(III.54)}$$

$$D_y = 2(b_4 t^4 + b_3 t^3 + b_2 t^2 + b_1 t + b_0) \quad \text{(III.55)}$$

$$b4 = l2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) + x_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) - l_3 \cos(\beta)(l_2^2 + q_3^2 - q_2^2 + x_2^2) + 2l_2 l_3(x_3 \cos(\beta) - x_2 \cos(\beta) + y_3 \sin(\beta) - 2l_2 x_2 x_3 + 2l_3 x_2 x_3 \cos(\beta) + 2l_3 x_2 x_3 \sin(\beta) \quad \text{(III.56)}$$

$$b3 = 3l_3x_2x_3\cos(\beta) - 3l_2l_3(y_3\cos(\beta) + x_2\sin(\beta) - x_3\sin(\beta) - 3l_3x_2y_3\cos(\beta) - 2l_3\sin(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) \quad \text{(III.57)}$$

$$b_2 = 2x_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - 2x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) - 3l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta)) \quad \text{(III.58)}$$

$$b_1 = 3l_2l_3(y_3\cos(\beta) - x_2\sin(\beta) - x_3\sin(\beta)) - 2l_3\sin(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) - 3l_3x_2y_3\cos(\beta) + 3l_3x_2x_3\sin(\beta) \quad \text{(III.59)}$$

$$b_0 = x_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - l_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) - x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) + l_3\cos(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) + 2l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta)) - 2l_2x_2x_3 - 2l_3x_2x_3\cos(\beta) - 2l_3x_2y_3\sin(\beta) \quad \text{(III.60)}$$

Remplaçant cette solution dans la première équation du système (III.22), cela conduit à une équation du huitième degré qui dépend uniquement de φ. Ce polynôme peut être ramené à un polynôme du sixième degré car divisible par le facteur (t^2+1).Sur l'annexe A, on donne l'expression analytique littérale des coefficients de ce polynôme.

On obtient alors un polynôme en la variable t défini dans tout le domaine R, et dont chaque solution permet d'obtenir φ qui à son tour permet de déterminer x et y .La solution au problème géométrique direct n'est donc pas unique et nécessite l'utilisation de méthodes numériques (Méthode de Bairstow ou la méthode de Newton-Raphson) qui nécessite néanmoins la connaissance préalable du domaine de recherche pour déterminer l'estimé initial et une étude pour déterminer le nombre de solutions :

Si le polynôme s'écrit $p(t)=0$, la solution s'écrit :

$$t_{n+1} = t_n - \frac{P(t_n)}{\frac{dP}{dt}(t_n)}$$

Le programme qui résout la géométrie directe polynomiale ***geo_direct_poly*** et les sous-programmes auxquels il fait appel sont donnés au chapitre A

III.7. Méthode géométrique :

III.7.1. Principe :

Le point de départ est l'observation qu'il est possible de déterminer par construction géométrique la géométrie directe du robot en traçant trois cercles concentriques $C_i (i=1:3)$ dont les centres sont placés aux points fixes d'articulation A_i $(i=1:3)$ et dont les rayons sont égaux aux coordonnées articulaires correspondantes q_i $(i=1:3)$ Il suffira alors de placer la plateforme de telle manière à ce que les sommets du triangle soient positionnés sur les cercles correspondants P_i $(i=1:3)$ Cette procédure peut être automatisée de la manière suivante :

1- On déplacera le sommet **1** de telle manière qu'il balaye tout le cercle. Soit P_1 sa position initiale $P_1(q_1,0)$.

2- Pour chaque position P_1, on tracera un cercle de rayon l_2 qui coupera le cercle C_2 en un point ou deux points **(** P_2 **et** P_2' **)**.La condition d'intersection doit être vérifiée au préalable: distance entre les deux centres inférieure à la somme $q_1 + q_2 + l_2$

3- A partir de P_2, on tracera un cercle de rayon l_1 et un autre cercle de centre P_1 et ayant un rayon égal à l_3. Ces deux cercles se coupent en un point P_3

4- On vérifiera si le point P_3 appartient au cercle C_3, en calculant la distance A_3 P_3 qui doit être égale à q_3 à une précision fixée à l'avance.

5- Le point sera retenu dans cette éventualité, sinon on procédera de même avec le second point P_2'

6- On déplacera le point P_1 sur le cercle jusqu'à le balayer complètement. Il est évident que la condition d'intersection des cercles permet de ne balayer que certains points du cercle C_1 au détriment d'autres.

7- Le pas de balayage étant forcément limité (nous avons pris 1000 points soit un incrément de 0.36 °), une fois la solution trouvée, la procédure est reprise en déterminant les positions successives des solutions qui seront retenues, en affinant le pas pour trouver la solution optimale correspondant à l'erreur minimale (critère quadratique)

III.7.2. Calcul analytique :

Calcul analytique des coordonnées intersection de deux cercles connus par leurs centres et leurs rayons $C_1(x_1, y_1, r_1)$ et $C_2(x_2, y_2, r_2)$

- Cas où $x_1 = x_2$ (III.61)

$$y_c = \frac{r_1^2 - r_2^2 + y_2^2 - y_1^2}{2(y_2 - y_1)} \quad \text{(III.62)}$$

$$x_c = x_1 \pm \sqrt{r_1^2 - \left(\frac{r_1^2 - r_2^2 + (y_2 - y_1)^2}{2(y_2 - y_1)} \right)^2} \quad \text{(III.63)}$$

- Cas où $y_1 = y_2$

$$x_c = \frac{r_1^2 - r_2^2 + x_2^2 - x_1^2}{2(x_2 - x_1)} \quad \text{(III.64)}$$

$$y_c = y_1 \pm \sqrt{r_1^2 - \left(\frac{r_1^2 - r_2^2 + (x_2 - x_1)^2}{2(x_2 - x_1)} \right)^2} \quad \text{(III.65)}$$

- Cas général

$$Ax^2 - 2Bx + C = 0 \quad \text{(III.66)}$$

$$A = 1 + \left(\frac{x_2 - x_1}{y_2 - y_1} \right)^2 \quad \text{(III.67)}$$

$$B = x_1 + \frac{(x_2 - x_1)}{(y_2 - y_1)^2} (r_1^2 - r_2^2 + x_2^2 - x_1^2 + (y_2 - y_1)^2) \quad \text{(III.68)}$$

$$C = x_1^2 - r_1^2 + \left(\frac{r_1^2 - r_2^2 + x_2^2 - x_1^2 + (y_2 - y_1)^2}{2(y_2 - y_1)} \right)^2 \quad \text{(III.69)}$$

- Cas où $B^2 - AC = 0$

$$x_c = \frac{B}{A} \quad \text{(III.70)}$$

$$y_c = y_1 \pm \sqrt{r_1^2 - (x_c - x_1)^2} \quad \text{(III.71)}$$

- Cas où $B^2 - AC \succ 0$

$$x_c = \frac{B \pm \sqrt{B^2 - AC}}{A} \qquad \text{(III.72)}$$

$$y_c = \frac{r_1^2 - r_2^2 + x_2^2 - x_1^2 + y_2^2 - y_1^2 - 2x_c(x_2 - x_1)}{2(y_2 - y_1)} \qquad \text{(III.73)}$$

- $B^2 - AC \prec 0$: **Pas de Solutions**

III.7.3. Graphe de Construction :

Ce graphe a été réalisé pour les données suivantes :

$A_1(0,0)$, $A_2(3,0)$, $A_3(2,3)$, $T(1,1,60)$, Et $Q = [1.3132 \;\; 2.2361 \;\; 2.1918]$

Correspondant à la pose initiale $P(1,1,0)$.

La méthode polynomiale nous donne deux solutions :

$P_1 = [1.0000 \quad -1.0000]$

$P_2 = [2.0000 \quad -1.0000]$

$P_3 = [1.5000 \quad -0.1330]$

Et :

$P_1 = [1.0150 \quad 0.9838]$

$P_2 = [2.0131 \quad 1.0277]$

$P_3 = [1.3773 \quad 1.8715]$

Nous avons représenté le graphe correspondant à la première solution:

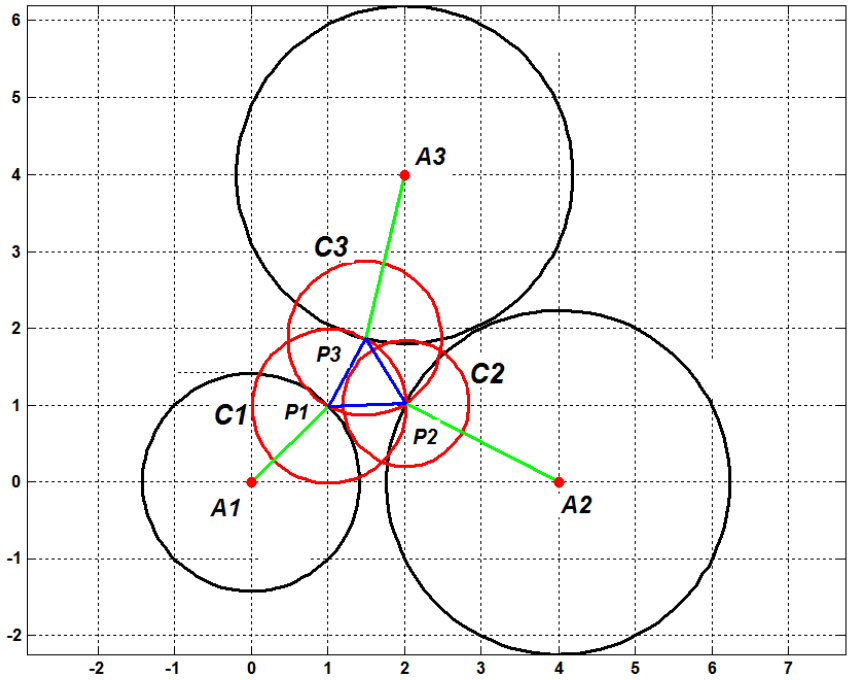


Figure III.5 : Construction Graphique de la solution

III.7.3. Remarques

Le balayage du cercle exige un temps de calcul non négligeable qui dépend de la puissance du processeur. Cette méthode est systématique et trouve toutes les solutions possibles même dans le cas de dégénérescence .Elle peut néanmoins générer plusieurs solutions en fonction du critère de précision fixé. Plus la précision est élevée, plus on s'assurera de ne retenir que les solutions exactes. Cependant, si on se suffit d'une précision acceptable, on trouve des solutions que le calcul analytique ne pourra pas donner.

Dans l'annexe B, on donne le programme ***geo_direct_geo*** qui calcule la géométrie directe par cette méthode graphique avec les sous programmes afférents.

Chapitre IV

Etude de la dégénérescence

IV.1-Introduction

Ce chapitre enquête sur deux situations dans laquelle la géométrie directe du robot parallèle 3-RPR est dégénérée. On définit la dégénérescence comme la propriété que présente un robot de présenter deux positions (x, y) pour un même angle φ.

Ces situations n'ont pas été étudiées auparavant. La première dégénérescence survient quand certaines conditions sont imposées sur les valeurs des coordonnées articulaires, qui produisent une dégénérescence pour la position particulière où l'orientation de la plateforme φ est nulle. Cette dégénérescence produit une racine double du polynôme caractéristique qui pourrait être interprétée faussement, mais qui en fait correspond à deux solutions distinctes des coordonnées cartésiennes du point P_1.

Une seconde situation peut se produire pour un angle quelconque, solution de l'équation caractéristique du système linéaire (CF chapitre III) mais sous réserve que les valeurs des coordonnées articulaires établissent entre elles des relations données.

Enfin, dans la dernière situation, nous montrons que les solutions dégénérées de la géométrie directe se produit dans tout l'espace articulaire et que le triangle de plateforme pour chaque solution est tourné d'un angle de 180°. Pour ces manipulateurs, la géométrie directe se réduit à la résolution d'un polynôme de $3^{ème}$ degré, chacune des racines étant de multiplicité deux.

IV.2-Formulation analytique

Nous avons déjà vu dans le précédent chapitre (chapitre 3) que les équations de la géométrie directe pouvaient se ramener à la résolution du système suivant :

$$Qx + Ry = S \qquad \text{(IV.1)}$$

Et :

$$Ux + Vy = W \qquad \text{(IV.2)}$$

Les expressions des coefficients du système ayant déjà été établies au chapitre III précédent.
Si le déterminant $D=QV-RU$ s'annule, c'est-à-dire :

$$D = QV + RU = 0 \qquad \text{(IV.3)}$$

Alors pour que le système admette des solutions, on doit avoir :

$$D_x = SV + RW = 0 \quad \text{(IV.4)}$$

Et :

$$D_y = QW - US = 0 \quad \text{(IV.5)}$$

Cette condition est vérifiée pour un ou deux angles t, donc deux angles $\varphi = 2arctg(t)$

IV.2.1 Dégénérescence du premier ordre

Dans ce cas, seule une position angulaire est dégénérée, et conduit à deux solutions, les quatre autres solutions de l'équation du sixième degré ne sont pas affectées.

Considérons le cas particulier d'un angle nul $\varphi = 0$

En reprenant l'équation du déterminant :

$$D = (\sin(\varphi)(l_2 x_3 - l_3 x_2 \cos(\beta) - l_3 y_2 \sin(\beta)) - \cos(\varphi)(l_2 y_3 + l_3 x_2 \sin(\beta)) + x_2 y_3 + l_2 l_3 \sin(\beta)) \ ; \quad \text{(IV.6)}$$

Qui peut s'écrire en effectuant le changement de variable : $t = \tan(\frac{\varphi}{2})$

$$D = 4(c_2 t^2 + c_1 t + c_0) = 4\left((l_2 + x_2)(y_3 + l_3 \sin(\beta))t^2 + 2(l_2 x_3 - l_3 x_2 \cos(\beta))t + (l_3 \sin(\beta) + y_3)(l_2 - x_2)\right) \quad \text{(IV.7)}$$

Avec :

$$c_2 = (l_2 + x_2)(y_3 + l_3 \sin(\beta)) \quad \text{(IV.8)}$$

$$c_1 = 2(l_2 x_3 - l_3 x_2 \cos(\beta)) \quad \text{(IV.9)}$$

$$c_0 = (l_3 \sin(\beta) + y_3)(l_2 - x_2) \quad \text{(IV.10)}$$

Si on prend $l_2 = x_2$, alors $t = 0$ est une racine de l'équation caractéristique D=0. Pour que le système admette une solution, on doit vérifier que D_x et D_y s'annulent tous deux pour cette valeur.

Rappelons les coefficients du système :

$$Q = 2l_2 \cos(\varphi) - 2x_2 \qquad U = 2l_3 \cos(\varphi + \beta) - 2x_3$$

$$R = 2l_2 \sin(\varphi) \qquad V = 2l_3 \sin(\varphi + \beta) - 2y_3$$

$$S = q_2^2 - q_1^2 - l_2^2 - x_2^2 + 2l_2 x_2 \qquad W = q_3^2 - q_1^2 - l_3^2 - x_3^2 - y_3^2 + 2l_3 x_3 \cos(\varphi + \beta) + 2l_3 y_3 \sin(\varphi + \beta)$$

Pour que D_x et D_y s'annulent, il faut en plus que S s'annule. Cela est obtenu pour $\mathbf{q_2 = q_1}$.

Pour tirer la position (x, y) de la solution dégénérée, on doit partir de la relation (IV.1) et tirer x (ou y) en fonction de y (ou x) et remplacer dans l'expression $x^2 + y^2 = q_1^2$.

Dans le cas général, on doit utiliser l'organigramme suivant :

*%**********

Résolution P(φ)= 0 (Polynôme ordre 6)

Calcul N : Nombre de solutions réelles

Pour I=1 :N FAIRE

Si D(φ)=0 (Equation Caractéristique système ordre 2)

Si Q=0

Si R=0

Si S=0

Si U=0

Si V=0

Si W=0

Dégénérescence dans tout l'espace

Sinon

Pas de Solutions

Sinon $(V \neq 0)$

$y = \frac{W}{V}$ si $q_1^2 \geq y^2, x = \pm\sqrt{q_1^2 - y^2}$ sinon pas de Solutions (IV.11)

Sinon $(U \neq 0)$

Si V=0

$x = \frac{W}{U}$ si $q_1^2 \geq x^2, y = \pm\sqrt{q_1^2 - x^2}$ sinon pas de Solutions (IV.12)

Sinon

$$\Delta = q_1^2\left(1+\left(\frac{U}{V}\right)^2\right)-\left(\frac{W}{V}\right)^2 \quad \text{(IV.13)}$$

Si $\Delta \geq 0$

$$x = \frac{\left(\left(\frac{UW}{V^2}\right) \pm \sqrt{\Delta}\right)}{\left(1+\left(\frac{U}{V}\right)^2\right)},\ y = \frac{W - Ux}{V} \quad \text{(IV.14)}$$

Sinon : Pas de Solutions

Sinon $(s \neq 0)$

Pas de Solutions $(Q = R = 0, s \neq 0)$

Sinon $(R \neq 0)$

$$x = \frac{S}{R} \text{ si } q_1^2 \geq y^2, x = \pm\sqrt{q_1^2 - y^2} \text{ sinon pas de Solutions} \quad \text{(IV.15)}$$

Sinon $(Q \neq 0)$

Si R=0

$$x = \frac{S}{Q} \text{ si } q_1^2 \geq x^2, y = \pm\sqrt{q_1^2 - x^2} \text{ sinon pas de Solutions} \quad \text{(IV.16)}$$

Sinon

$$\Delta = q_1^2 \left(1 + \left(\frac{Q}{R}\right)^2\right) - \left(\frac{S}{R}\right)^2 \quad \text{(IV.17)}$$

Si $\Delta \geq 0$

$$x = \frac{\left(\left(\frac{QS}{R^2}\right) \pm \sqrt{\Delta}\right)}{\left(1 + \left(\frac{Q}{R}\right)^2\right)}, \; y = \frac{S - Qx}{R} \quad \text{(IV.18)}$$

Sinon : Pas de Solutions **:** *Recherche de combinaisons [q1 q2 q3] Vérifiant les deux relations Dx=Dy=0*

Sinon $(D(\varphi) \neq 0)$

Pas de dégénérescence : Calculer la position (x, y)

$$x = \frac{D_x}{D}, \; y = \frac{D_y}{D} \quad \text{(IV.19)}$$

%***

Le calcul des quatre autres racines est donné par la relation D_x=0

$$\begin{aligned} D_x = {} & 2(q_2^2 - q_1^2)(-y_3 + l_3 \sin(\varphi + \beta)) - l_2 \sin(\varphi)(q_3^2 - q_1^2) + (x_2^2 + l_2^2) \\ & (y_3 - l_3 \sin(\varphi + \beta)) + l_2 \sin(\varphi)(x_3^2 + y_3^2 + l_3^2) - 2x_2 l_2 y_3 \cos(\varphi) + \\ & \cos^2(\varphi) 2 l_2 l_3 x_2 \sin(\beta) + \sin^2(\varphi) 2 l_2 l_3 (x_3 \sin(\beta) - y_3 \cos(\beta)) + \\ & l_2 l_3 \sin(2\varphi)(x_2 \cos(\beta) - x_3 \cos(\beta) - y_3 \sin(\beta))) \end{aligned} \quad \text{(IV.20)}$$

Sous forme polynomiale avec la variable intermédiaire t :

$$D_x = 2(a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0) \quad \text{(IV.21)}$$

$$a_4 = (y_3 + l_3 \sin(\beta))(l_2^2 + 2 l_2 x_2 + q_1^2 - q_2^2 + x_2^2) \quad \text{(IV.22)}$$

$$\begin{aligned} a_3 = {} & 2 l_2 (l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) + 2 l_3 \cos(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) + \\ & 4 l_2 l_3 (x_3 \cos(\beta) - x_2 \cos(\beta) + y_3 \sin(\beta)) \end{aligned} \quad \text{(IV.23)}$$

$$a_2 = 2y_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) + 8l_2l_3(y_3\cos(\beta) - x_3\sin(\beta)) - 4l_2l_3x_2\sin(\beta) \quad \text{(IV.24)}$$

$$a_1 = 2l_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) + 2l_3\cos(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) - 4l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta) \quad \text{(IV.25)}$$

$$a_0 = (y_3 - l_3\sin(\beta))(l_2^2 - 2l_2x_2 + q_1^2 - q_2^2 + x_2^2) \quad \text{(IV.26)}$$

Dans le cas où $l_2 = x_2$ et $q_2 = q_1$, on observe que $a_0 = 0$, ce qui était prévisible, car pour t=0, D=0, la condition :

$$x^2 + y^2 = q_1^2 \quad \text{(IV.27)}$$

S'écrit également :

$$D_x^2 + D_y^2 = q_1^2D^2 \quad \text{(IV.28)}$$

Comme t=0 est racine du polynôme D, alors D_x et D_y doivent également l'être. On obtient dans le cas général

On vérifie que D_y admet une racine double en t=0 c'est-à-dire $b_0 = b_1$, dans le cas où $q_1 = q_2$ et $x_2 = l_2$

$$\begin{aligned} D_y &= 2(q_2^2 - q_2^2)(x_2 - l_2\cos(\varphi + \beta)) + (q_2^2 - q_2^2) + \\ &(-x_2 + l_2\cos(\varphi)) + (x_2^2 + l_2^2)(-x_3 + l_3\cos(\varphi + \beta)) + \\ &(x_3^2 + y_3^2 + l_3^2)(x_2 - l_2\cos(\varphi)) + 2x_2x_3l_2\cos(\varphi) - \\ &2x_2x_3l_3\cos(\varphi + \beta) - 2x_2y_3l_3\sin(\varphi + \beta) + \cos(\varphi) - \\ &2x_2x_3l_3\cos(\varphi + \beta) - 2x_2y_3l_3\sin(\varphi + \beta) + \\ &\cos^2(\varphi)2l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta)) + \\ &l_2l_3\sin(2\varphi)(y_3\cos(\beta) + x_2\sin(\beta) - x_3\sin(\beta))) \end{aligned} \quad \text{(IV.29)}$$

$$D_y = 2(b_4t^4 + b_3t^3 + b_2t^2 + b_1t + b_0) \quad \text{(IV.30)}$$

Avec :

$$b4 = l2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) + x_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) - l_3\cos(\beta)(l_2^2 + q_3^2 - q_2^2 + x_2^2) + 2l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta)) - 2l_2x_2x_3 + 2l_3x_2x_3\cos(\beta) + 2l_3x_2x_3\sin(\beta) \quad \text{(IV.31)}$$

$$b3 = 3l_3x_2x_3\cos(\beta) - 3l_2l_3(y_3\cos(\beta) + x_2\sin(\beta) - x_3\sin(\beta)) - 3l_3x_2y_3\cos(\beta) - 2l_3\sin(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) \quad \text{(IV.32)}$$

$$b_2 = 2x_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - 2x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) - 3l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta)) \quad \text{(IV.33)}$$

$$b_1 = 3l_2l_3(y_3\cos(\beta) - x_2\sin(\beta) - x_3\sin(\beta)) - 2l_3\sin(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) - 3l_3x_2y_3\cos(\beta) + 3l_3x_2x_3\sin(\beta) \quad \text{(IV.34)}$$

$$b_0 = x_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - l_2(l_3^2 + q_1^2 - q_3^2 + x_3^2 + y_3^2) - x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) - x_3(l_2^2 + q_1^2 - q_2^2 + x_2^2) + l_3\cos(\beta)(l_2^2 + q_1^2 - q_2^2 + x_2^2) + 2l_2l_3(x_3\cos(\beta) - x_2\cos(\beta) + y_3\sin(\beta)) - 2l_2x_2x_3 - 2l_3x_2x_3\cos(\beta) - 2l_3x_2y_3\sin(\beta) \quad \text{(IV.35)}$$

L'équation résolvante dont est tirée la valeur de φ *(CF annexe)* est une équation du huitième degré qui peut toujours se ramener à une équation du sixième degré car elle admet comme solution l'imaginaire pur $\pm i$.

Le facteur t élevé au carré étant commun à D_x^2, D_y^2 et D^2, on aboutit alors à une équation d'ordre quatre qui donne les solutions non dégénérées. La solution dégénérée pour $t=\varphi=0$ s'obtient suivant le processus décrit ci-haut.

IV.2.2 Cas d'un angle quelconque

Dans le cas d'un angle de dégénérescence non nul, ce qui est le cas si $l_3\sin(\beta) \neq y_3$ et $x_2 \neq l_2$. La singularité est obtenue pour les deux angles solutions du déterminant du système linéaire. On doit alors calculer les valeurs que doivent vérifier les coordonnées articulaires pour que le système soit résoluble. Il suffit d'annuler D_x et D_y pour les deux valeurs de t trouvées

$$D = 4(c_2t^2 + c_1t + c_0) = 0$$

$$c_2 = (l_2 + x_2)(y_3 + l_3\sin(\beta))$$

$$c_1 = 2(l_2x_3 - l_3x_2\cos(\beta))$$

$$c_0 = (l_3\sin(\beta) + y_3)(l_2 - x_2)$$

$$\Delta = -l_2^2l_3^2\sin^2(\beta) + l_2^2x_3^2 + l_2^2y_3^2 - 2l_2l_3x_2x_3\cos(\beta) + l_3^2x_2^2 - x_2^2y_3^2 \quad \text{(IV.36)}$$

Qui peut se simplifier en :

$$\Delta = (x_3l_2 - x_2l_3\cos(\beta))^2 - (l_2^2 - x_2^2)(y_3^2 - l_3^2\sin^2(\beta)) \quad \text{(IV.37)}$$

Les deux racines dans le cas où Δ est positif sont :

$$t_{12} = \frac{(l_2x_3 - l_3x_2\cos(\beta)) \pm \sqrt{\Delta}}{(l_2 + x_2)(y_3 + l_3\sin(\beta))} \quad \text{(IV.38)}$$

$$t_{12} = \frac{(l_2 x_3 - l_3 x_2 \cos(\beta)) \pm \sqrt{(x_3 l_2 - x_2 l_3 \cos^2(\beta)) - (l_2^2 - x_2^2)(y_3^2 - l_3^2 \sin^2(\beta))}}{(l_2 + x_2)(y_3 + l_3 \sin(\beta))} \quad \text{(IV.39)}$$

Les valeurs des coordonnées articulaires doivent vérifier les conditions (i=1,2), avec le retour à la variable angulaire $\varphi_i = 2\arctan(t_i)$ i=1,2

$$\begin{aligned} &D_x(\varphi_i) = 0 = (q_2^2 - q_1^2)(-y_3 + l_3 \sin(\varphi_i + \beta)) - l_2 \sin(\varphi_i)(q_3^2 - q_1^2) \\ &+(x_2^2 + l_2^2)(y_3 - l_3 \sin(\varphi_i + \beta)) + l_2 \sin(\varphi_i)(x_3^2 + y_3^2 + l_3^2) \\ &-2x_2 l_2 y_3 \cos(\varphi_i) + 2l_2\ l_3 x_2 \cos^2(\varphi_i)\sin(\beta) + 2l_2 l_3 \sin^2(\varphi_i) \\ &(x_3 \sin(\beta) - y_3 \cos(\beta)) + l_2 l_3 \sin(2\varphi_i)(x_2 \cos(\beta) - x_3 \\ &\cos(\beta) - y_3 \sin(\beta))) \end{aligned} \quad \text{(IV.40)}$$

Et :

$$\begin{aligned} &D_y(\varphi_i) = 0 = (q_2^2 - q_1^2)(x_3 - l_3 \cos(\varphi_i + \beta)) + (q_3^2 - q_1^2)(-x_2 + l_2 \cos(\varphi_i)) \\ &+(x_2^2 + l_2^2)(-x_3 + l_3 \cos(\varphi_i + \beta)) + (x_3^2 + x_3^2 + l_3^2)(x_2 - l_2 \cos(\varphi_i)) \\ &+2x_2 x_3 l_3 \cos(\varphi_i + \beta) - 2x_2 y_3 l_3 \sin(\varphi_i + \beta) + 2l_2 l_3 \cos^2(\varphi)(x_3 \cos(\beta) \\ &-x_2 \cos(\beta) + y_3 \sin(\beta)) + l_2 l_3 \sin(2\varphi_i)(y_3 \cos(\beta) + x_2 \sin(\beta) - x_3 \sin(\beta)) \end{aligned} \quad \text{(IV.41)}$$

Nous aboutissons pour les deux solutions à un système de la forme :

$$\alpha_1(q_2^2 - q_1^2) + \beta_1(q_3^2 - q_1^2) = \gamma_1 \quad \text{(IV.42)}$$

$$\alpha_2(q_2^2 - q_1^2) + \beta_2(q_3^2 - q_1^2) = \gamma_2 \quad \text{(IV.43)}$$

$$\alpha_1 = (-y_3 + l_3 \sin(\varphi + \beta)) \quad \text{(IV.44)}$$

$$\beta_1 = -l_2 \sin(\varphi) \quad \text{(IV.45)}$$

$$\begin{aligned} &-\gamma_1 = (x_2^2 - l_2^2)(y_3 - l_3 \sin(\varphi_i + \beta)) + l_2 \sin(\varphi_i)(x_3^2 + y_3^2 + l_3^2) - 2x_2 l_2 y_3 \cos(\varphi_i) + 2l_3 l_2 x_2 \cos^2(\varphi_i)\sin(\beta) + \\ &2l_2 l_3 \sin^2(\varphi_i)(x_3 \sin(\beta) - y_3 \cos(\beta)) + l_2 l_3 \sin(2\varphi_i)(x_2 \cos(\beta) - x_3 \cos(\beta) - y_3 \sin(\beta)) \end{aligned} \quad \text{(IV.46)}$$

$$\alpha_2 = (x_3 - l_3 \cos(\varphi\ + \beta)) \quad \text{(IV.47)}$$

$$\beta_2 = (-x_2 + l_2 \cos(\varphi\ )) \quad \text{(IV.48)}$$

$$\begin{aligned} &\gamma_2 = (x_2^2 - l_2^2)(-x_3 + l_3 \cos(\varphi_i + \beta)) + (x_3^2 + y_3^2 + l_3^2)(x_2 - l_2 \cos(\varphi_i)) \\ &+2x_2 x_3 l_2 \cos(\varphi_i) - 2x_2 x_3 l_2 \cos(\varphi_i + \beta) - 2x_2 y_3 l_3 \sin(\varphi_i + \beta) + \\ &-2l_2 l_3 \cos^2(\varphi_i)(x_3 \cos(\beta) - x_2 \cos(\beta) - y_3 \sin(\beta)) \\ &+2l_2 l_3 \sin(2\varphi_i)(y_3 \cos(\beta) + x_2 \sin(\beta) - x_3 \sin(\beta)) \end{aligned} \quad \text{(IV.49)}$$

Qui peut être résolu, si est imposé de telle sorte à ne retenir que les valeurs positives de q_2^2 et q_3^2 . Cette combinaison des coordonnées articulaires donne lieu à des singularités au niveau de la géométrie inverse. On observe qu'il existe une infinité de combinaison de valeurs possibles $q=[q_1,q_2,q_2]$

IV.2.3 Dégénérescence dans tout l'espace articulaire

Ce cas est obtenu en cherchant les conditions qui annulent les coefficients du polynôme d'ordre deux de l'équation caractéristique. La condition de dégénérescence sera vérifiée indépendamment de la valeur des coordonnées articulaires :

$$c_2=(l_2+x_2)(y_3+l_3\sin(\beta))=0 \quad \text{(IV.50)}$$

$$c_1=2(l_2x_3-l_3x_2\cos(\beta))=0 \quad \text{(IV.51)}$$

$$c_0=(l_3\sin(\beta)+y_3)(l_2-x_2)=0 \quad \text{(IV.52)}$$

Il vient alors :

$$l_2=x_2 \quad \text{(IV.53)}$$

$$y_3=-l_3\sin(\beta) \quad \text{(IV.54)}$$

$$l_2x_3=l_3x_2\cos(\beta) \qquad \textbf{Soit :} \qquad x_3=l_3\cos(\beta) \quad \text{(IV.55)}$$

Ou encore : $l_3=\sqrt{x_3^2+y_3^2}$ (IV.56)

En exprimant les polynômes D_x et D_y comme précédemment, on trouve :

$$D_x=[a_3\ a_2\ a_1\ a_0]=2((q_1^2(x_2-x_3)+x_3q_2^2-x_3q_3^2+4x_2x_3(x_3-x_2))t^3+y_3(q_1^2-q_2^2+4x_2(x_2-x_3))t^2 +(q_1^2(x_2-x_3)+x_3q_2^2-x_2q_3^2+4x_2y_3^2)t+y_3(q_1^2-q_2^2)) \quad \text{(IV.57)}$$

$$a_4=0 \quad \text{(IV.58)}$$

$$a_3=2(q_1^2(x_2-x_3)+x_3q_2^2-x_2q_3^2+4x_2x_3(x_3-x_2)) \quad \text{(IV.59)}$$

$$a_2=2y_3(q_1^2-q_2^2+4x_2(x_2-2x_3)) \quad \text{(IV.60)}$$

$$a_1=2(q_1^2(x_2-x_3)+x_3q_2^2-x_2q_3^2+4x_2y_3^2) \quad \text{(IV.61)}$$

$$a_0=2y_3(q_1^2-q_2^2) \quad \text{(IV.62)}$$

Et :

$$D_y=[b_3\ b_2\ b_1\ b_0]=2((q_1^2(x_2-x_3)+x_3q_2^2-x_2q_3^2+4x_2x_3(x_3-x_2))t^4+y_3(q_1^2-q_2^2+4x_2(x_2-2x_3)))t^3 +(q_1^2(x_2-x_3)+x_3q_2^2-x_2q_3^2+4x_2y_3^2)t^2+y_3(q_1^2-q_2^2)t) \quad \text{(IV.63)}$$

$$b_4 = 2(q_1^2(x_2 - x_3) + x_3 q_2^2 - x_2 q_3^2 + 4x_2 x_3(x_3 - x_2)) \quad \text{(IV.64)}$$

$$b_3 = 2y_3(q_1^2 - q_2^2 + 4x_2(x_2 - 2x_3)) \quad \text{(IV.65)}$$

$$b_2 = 2(q_1^2(x_2 - x_3) + x_3 q_2^2 - x_2 q_3^2 + 4x_2 y_3^2) \quad \text{(IV.66)}$$

$$b_1 = 2y_3(q_1^2 - q_2^2) \quad \text{(IV.67)}$$

$$b_0 = 0 \quad \text{(IV.68)}$$

On remarquera que :

$$D_y = tD_x \quad \text{(IV.69)}$$

Donc $D_y{}^2 + D_x{}^2 = (t^2 + 1)D_x{}^2 = 0$ (IV.70)

Les solutions de l'équation caractéristique globale d'ordre 6, sont les mêmes que celles de D_x mais chaque racine est doublée, on retrouve également les deux racines.

Exemple :

$$x_2 = l_2 = 1; x_3 = 0; y_3 = 1; \beta = -\frac{\pi}{2}; q_1 = \frac{4}{5}; q_1 = \frac{3}{2}; q_1 = \frac{3}{2}$$

$$161t^3 - 239t^2 - 239t + 161 = 0 \quad \text{(IV.71)}$$

Il existe plusieurs méthodes de résolution d'équations polynomiales d'ordre 3 : Tschirnhaus, Ferrari, Tartaglia-Cardan, Descartes (ordre 4 qui se ramène à l'ordre 3). Il existe également des méthodes numériques comme celles de Newton ou de Bairstow, qui nécessitent la donnée d'un estimé initial. Dans ce qui suit, nous utiliserons la méthode de Cardan qui est donnée en annexe. On trouve :

En faisant le changement de variable :

$$X = t + \frac{a_2}{3a_3} = t - \frac{239}{483} \quad \text{(IV.72)}$$

On aboutit à la forme standard de l'équation d'ordre 3 : $X^3 + pX + q = 0$ (IV.73)

Avec $p = -2.2190$ et $q = 0.0231$

$$\Delta = -(4p^3 + 27q^2) = 43.6921 \succ 0 \quad \text{IV.74)}$$

Les solutions sont dans ce cas :

$$X(1) = 2\sqrt{-\frac{p}{3}} * \cos(\frac{1}{3} a\cos(-\frac{q}{2}\sqrt{-\frac{27}{p^3}}) = 1.4844 \quad \text{(IV.75)}$$

$$X(2) = 2\sqrt{-\frac{p}{3}} * \cos(\frac{1}{3} a\cos(-\frac{q}{2}\sqrt{-\frac{27}{p^3}} + \frac{2\pi}{3}) = -1.4948 \quad \text{(IV.76)}$$

$$X(3) = 2\sqrt{-\frac{p}{3}} * \cos(\frac{1}{3} a\cos(-\frac{q}{2}\sqrt{-\frac{27}{p^3}} + \frac{4\pi}{3}) = 0.0104 \quad \text{(IV.77)}$$

On en déduit les solutions en $t = \tan(\frac{\varphi}{2})$

$$t = X - \frac{a_2}{3a_3} = X + \frac{239}{483} = \begin{bmatrix} 1.9792 & -1.0000 & 0.5052 \end{bmatrix} \quad \text{(IV.78)}$$

Les trois angles correspondants à la dégénérescence seront alors :

$$\varphi = \begin{bmatrix} 126.3897^0 & -90.0000^0 & 53.6103^0 \end{bmatrix} \quad \text{(IV.79)}$$

A chaque angle, deux positions possibles (x, y) sont assignées. Le chapitre six (6) reprend cet exemple et donne les poses calculées par le logiciel.

Remplaçant ces valeurs, dans les expressions de Q, R, Q et U, V, W, on trouve

1-Pour $\varphi = -90^0$

$$Q = -2, R = -2 \text{ et } S = -\frac{39}{100} \text{ et } U = -2, V = -2 \text{ et } W = -\frac{39}{100} \quad \text{IV.80)}$$

Les conditions sont identiques, ce qui était prévisible, on utilisera donc une relation qui permet d'exprimer y en fonction de x et réciproquement : $x + y = \frac{39}{200}$, associée à l'équation (IV.1) :

$$x^2 + y^2 = q_1^2 = \frac{16}{25} \quad \text{(IV.81)}$$

On trouve les deux solutions :

$$x = \begin{bmatrix} 0.6547 & -0.4597 \end{bmatrix} \quad \text{(IV.82)}$$

Auxquelles sont associées les ordonnées :

$$y = \begin{bmatrix} 0.4597 & -0.6547 \end{bmatrix} \quad \text{(IV.83)}$$

Chapitre V

Principales méthodes mises en œuvre en intelligence artificielle

V.1 La logique floue définition et mise en œuvre

V.1.1. Introduction

Les systèmes de contrôle conventionnels nécessitent des modèles précis afin de créer les compensateurs appropriés pour contrôler un processus. Selon la méthode de contrôle, ces modèles peuvent être totalement déterministes (le modèle est supposé être parfait) ou stochastiques (lorsqu'une certaine incertitude est autorisée et est représentée par des signaux aléatoires). Mais le modèle n'est qu'une approximation du système réel; c'est une expression mathématique qui repose sur un certain nombre d'hypothèses. Par conséquent, lors d'une simulation du système, le contrôleur peut donner une performance satisfaisante, la réalisation du schéma de compensation sur le système actuel donne généralement des résultats assez médiocres et décevants. D'autre part, des ingénieurs expérimentés peuvent très bien maîtriser un processus sans avoir la moindre connaissance de son modèle mathématique. Ils peuvent contrôler le système en ne connaissant que ses caractéristiques physiques et en utilisant leur expérience. C'est exactement ce que les méthodes d'intelligence artificielle (IA) tentent d'imiter. Ils essaient de comprendre le raisonnement humain et de créer des contrôleurs qui n'auront pas besoin de modèle mais qui pourront néanmoins fonctionner de manière satisfaisante. Les principales méthodes constituant l'IA sont la logique floue (FL), les réseaux de neurones artificiels (ANN) et les algorithmes génétiques (GA).[91]

V.2. La nécessité d'une logique floue

L'idée de logique floue (FL) est née en juillet 1964 et a été publiée pour la première fois en 1965 par Lofti A. Zadeh (Université de Californie à Berkeley). Initialement, cette théorie a fait l'objet de nombreuses critiques et d'un scepticisme considérable de la part de la communauté scientifique. Les recherches de Zadeh, parrainées par le gouvernement des États-Unis, ont amené le Congrès à lui faire savoir qu'il s'agissait d'un gaspillage d'argent. Plus tard (à partir des années 90) et grâce aux efforts de nombreux chercheurs du monde entier, cette théorie est devenue une partie importante de l'ingénierie de contrôle. Cette nouvelle théorie a été déclenchée par le fait que les systèmes de contrôle modernes peuvent être extrêmement compliqués, ce qui rend les mathématiques pertinentes accessibles uniquement aux experts. De plus, le modèle mathématique est rarement précis et les non-linéarités influencent souvent le comportement du système. De plus, il peut ne pas être possible de modéliser les perturbations qui influencent également le système. Toutes ces caractéristiques font du contrôle précis une tâche extrêmement difficile et coûteuse. Ainsi, la nécessité d'une nouvelle

théorie, qui ne dépend pas d'un modèle du système, est devenue impérieuse. En FL, un modèle du système n'est pas nécessaire, mais le concepteur doit avoir une très bonne idée du système. Le contrôleur à concevoir va "penser" et "réagir" plutôt en tant qu'être humain dont l'expérience sur le problème spécifique est considérable. Les FLC sont utilisés dans de nombreux domaines de la théorie du contrôle. L'un de leurs principaux avantages est qu'ils sont très robustes et peuvent contrôler un système non linéaire. L'ironie est que la FL peut jouer dans des domaines où même le système conventionnel le plus compliqué a échoué. Par exemple, dans le domaine des technologies spatiales, la FL contribuait de manière significative et aidait à surmonter des problèmes qui semblaient insurmontables à l'origine. Comme nous le verrons plus tard, l'application de la FL présente un inconvénient majeur, car il n'existe pas de méthode claire pour concevoir un contrôleur. [91]

V.2.1. L'univers de discours :

L'univers de discours est l'ensemble de référence d'un mot du langage naturel .Il contient des termes qui évoquent à différents degrés le même concept que la variable floue et couvre l'ensemble des valeurs prises par cette variable.

V.2.2. Les termes et les variables linguistiques :

Pour décrire les procédés, les phénomènes ou les situations auxquelles nous sommes confrontés, nous avons recours à des expressions telles que : quelque, beaucoup, souvent, petit, lent, jeune …ect. De telles expressions portent le nom de *termes linguistiques*. Ils forment les valeurs des *variables linguistiques* de la logique floue. En général, une varaible linguistique est décomposée en un ensemble de termes qui couvrent tout son domaine de variation .Elle est caractérisée par le triplet (x, T(x), U) dans lequel x est le nom de la variable .T(x) est l'ensemble des termes linguistiques que peut prendre la variable x et U est l'univers de discours associes. Par exemple, la variable linguistique température peut avoir comme valeur les termes linguistiques qui appartiennent à l'ensemble T(x)= {très froid, froid, tiède, chaud, très chaud}.L'univers de discours est U=[0^0C , 40^0C]. (**Fig. V.1**)

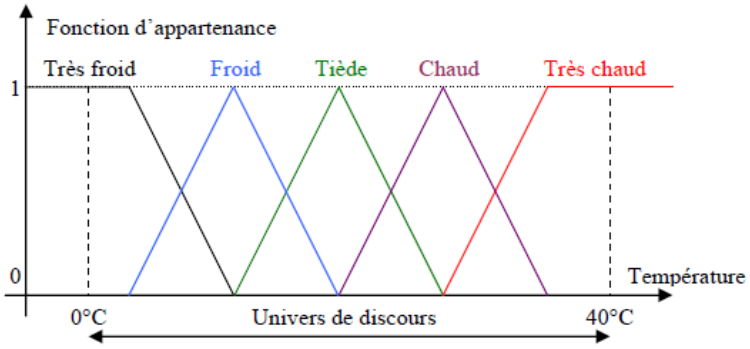


Figure. V.1 : Variable linguistique, termes linguistiques et univers de discours

V.*3.* Logique booléenne et floue

Supposons qu'il existe des objets de formes quelconques, Fig. V.2**,** et que la tâche souhaitée consiste à distinguer les formes comportant des angles et les formes comportant uniquement des courbes:

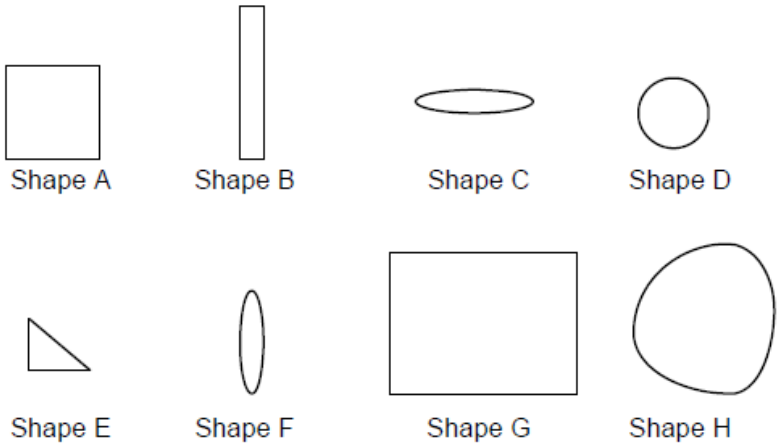


Figure. V.2 : Objets de formes différentes

Il est donc clair que les objets qui ont uniquement des courbes sont: la forme C, la forme D, la forme F et la forme H. Il est donc possible de créer un ensemble, A, contenant ces objets.

Mathématiquement, cela s'écrit : *A*= {Shape C, Shape D, Shape F, Shape H}. L'ensemble contenant tous les autres objets est : *B*= {Shape A, Shape B, Shape E, Shape G}.

L'ensemble contenant tous les objets l'univers du discours et est:

C= {Shape A, Shape B, Shape C, Shape D, Shape E, Shape F, Shape G, Shape H}.

Des sous-ensembles peuvent également être créés, par exemple un ensemble composé de tous les objets à 4 angles est : D = {Shape A, Shape B, Shape G} et est clairement un sous-ensemble de B : $D \subset B$.

L'ensemble union est un ensemble contenant tous les éléments de deux autres ensembles:

$D \cup B$ ={Shape C, Shape D, Shape F, Shape H, Shape A , Shape B , Shape G }
De toute évidence: $B \cup A = B \cup A = C$
L'ensemble intersection est un autre ensemble qui comporte les éléments communs en deux ensembles:

$B \cap E$ ={ Shape G }, où E={Shape G, Shape H }.Encore $B \cap E = E \cap B$
L'ensemble vide est l'ensemble qui ne contient aucun élément: $A \cap B = \varnothing$
Les autres propriétés sont:
1- $A \cup A = A$

2- $A \cap A = A$

3- $A \cup (B \cup C) = (A \cup B) \cup C = A \cup B \cup C$

4- $A \cap (B \cap C) = (A \cap B) \cap C = A \cap B \cap C$

5- $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

6- $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

7- $A \cup \varnothing = A$

8- $A \cap \varnothing = A$

Dans tout ce qui précède, un élément était membre à 100% de l'ensemble ou à 0%. Cette logique est appelée logique booléenne ou vraie et fausse. [91]

Supposons maintenant qu'il existe un ensemble de températures. Créez ensuite un sous-ensemble qui aura toutes les températures correspondant à un environnement chaud. Ensuite, en utilisant la logique précédente, on pourrait dire que les températures chaudes sont toutes les températures supérieures à 25 degrés: *HOT*= {*température* / *température* >25}, cette expression dit que l'ensemble Hot a toutes les températures de plus de 25 degrés. Mais qu'en est-il d'une température de 24,99 degrés, est-ce chaud? Évidemment, il en est ainsi, mais moins que 25 degrés. Cela introduit la nécessité de créer des ensembles (ensembles flous) avec des éléments qui ne suivent pas la logique booléenne, mais qui auront une fonction

d'appartenance qui déterminera leur appartenance à un ensemble. Par exemple, la température 25 est chaude à 100%, puis 24 à 90%, 20 à 50% et ainsi de suite:

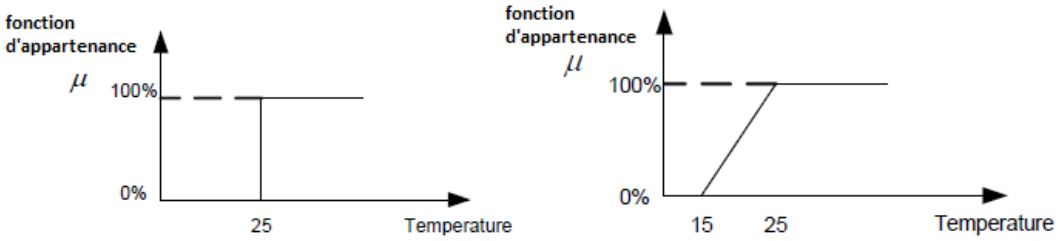


Figure. V.3 & V.4 : Ensembles normaux et flous

La fonction d'appartenance qui est montrée dans les Fig. V.3 & V.4 n'est rien de plus qu'une fonction qui montre à quel point un élément est membre ou non d'un ensemble flou. C'est à dire. si "t" est la température, sa fonction d'appartenance à un ensemble spécifique est ''μ (t)''. Dans le cas précédent pour t = 25, μ (t) = 1, pour t = 24, μ (t) = 0,9 et pour t = 15, μ (t) = 0. Par conséquent, pour définir complètement un ensemble flou, mentionner à quel élément appartient ou n'appartient pas; n'est pas suffisant. Une paire est nécessaire pour définir l'élément et son appartenance ou non, c'est-à-dire sa fonction d'appartenance: $A = \{x, \mu(x) | x \in X\}$, cette expression dit que l'ensemble A se compose de tous les éléments x munis de la fonction d'appartenance μ (x) lorsque x appartient à l'univers du discours X.

La première question à laquelle on doit répondre lorsqu'on utilise des ensembles flous est la forme de la fonction d'appartenance, c'est-à-dire le mappage de la valeur x sur une valeur comprise entre 0 et 1. Les principales fonctions d'appartenance dont dispose Matlab sont les suivantes:

1. forme triangulaire
2. forme trapézoïdale
3. formes gaussiennes
4. formes sigmoïdales
5. formes polynomiales

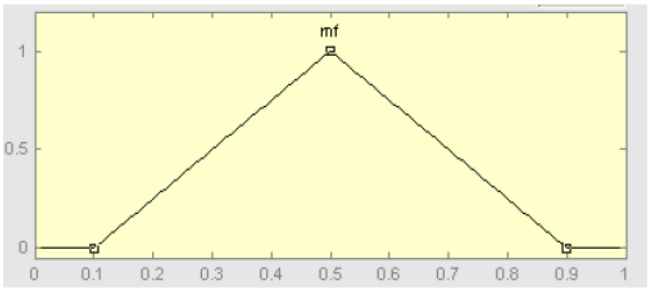


Figure. V. 5 : Forme triangulaire

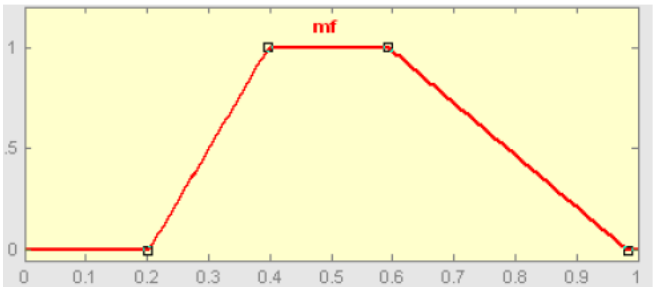


Figure. V.6 : Forme trapézoïdale

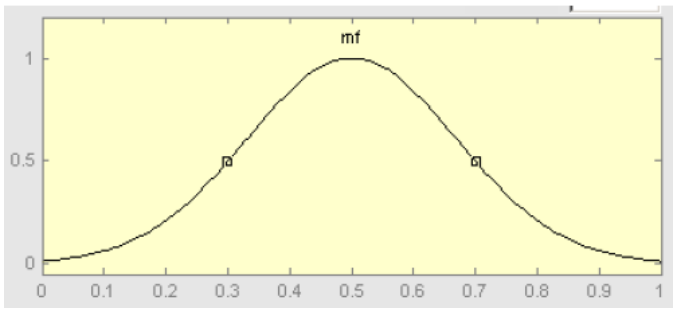


Figure. V. 7 : Forme gaussienne

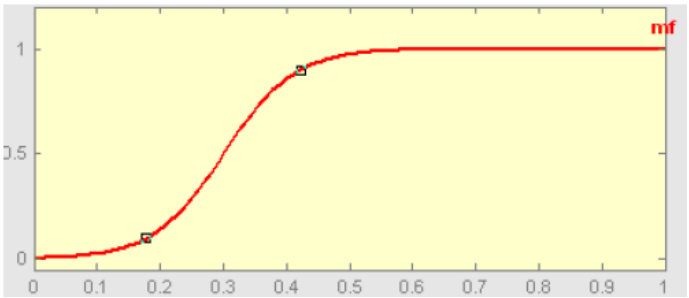


Figure. V.8 : Forme sigmoïdale

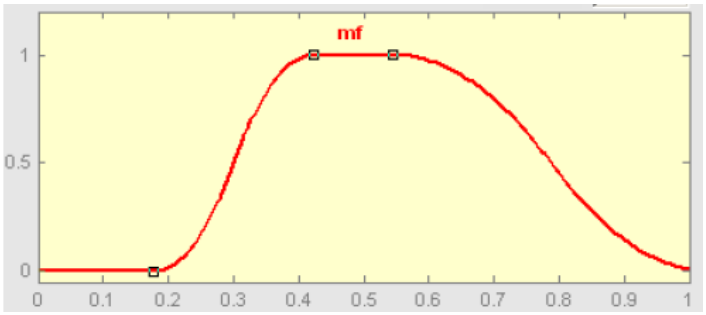


Figure. V.9 : Formes polynomiales

Le choix de la forme de la fonction d'appartenance n'est pas simple et seule l'expérience peut aider le concepteur. Les choix les plus courants sont le triangle et le trapèze, en raison de leur linéarité (linéaires par morceaux)

V.4. Opérateurs logiques (booléens et flous)

Dans la théorie des ensembles classique, des opérateurs logiques régissent les ensembles. Ce sont les opérateurs AND, OR et NOT.

Si l'objectif est de trouver l'intersection de deux ensembles, l'opérateur AND est utilisé. Par exemple assumer les ensembles A= {1, 3, 5,10} et B= {5, 6, 7,8}. L'intersection est l'ensemble constitué des éléments communs aux deux. D'où des éléments qui appartiennent à un ensemble A ET à un ensemble B.

Si l'élément «1» est membre de l'ensemble A et B, il appartient alors à l'intersection. Le reste des éléments suivra la même logique. Il existe une table associée vérité ou fausse (1 ou 0) associée à cet opérateur qui résume la logique AND. L'opérateur OR est associé à l'union de deux ensembles. Puisque l'union est un ensemble avec des éléments de A OU B:

A	B	AND	OR
1	0	0	1
0	1	0	1
1	1	1	1
0	0	0	0

Et NON (NOT) :

A	NOT
1	0
0	1

Comment ces opérateurs peuvent-ils être étendus aux ensembles Flous? Les ensembles booléens peuvent être considérés comme un cas particulier des ensembles Flous où les fonctions d'appartenance sont des unités et des zéros. L'opérateur AND est remplacé maintenant par le minimum des deux fonctions d'appartenance et l'opérateur OR par le maximum:

A	B	Min(A,B)	Max(A,B)
1	0	0	1
0	1	0	1
1	1	1	1
0	0	0	0

Par exemple, quelle est l'intersection des ensembles:

Chaud = {*température* / *température* >25} & *Très Chaud* = {*température* / *température*>30}

Il est clair que l'intersection est:

Inter = {*température* / *température* >25 & *température* <*30*}

Ou l'union: *Union* = {*température* / *température* >25 } puisque *Très Chaud* est un sous-ensemble de *Chaud*

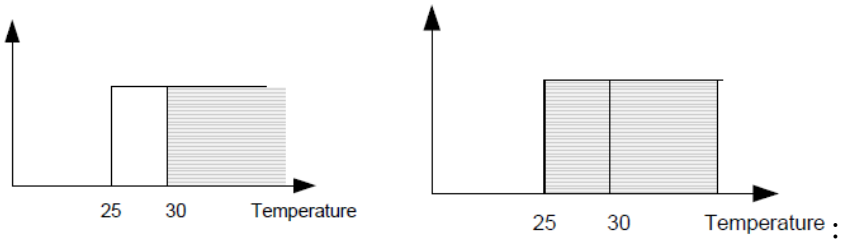


Figure. V.10 & V.11 : Intersection et union booléenne

Les mêmes opérateurs avec la logique floue sont:

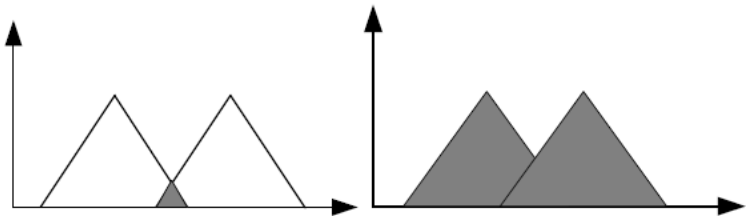


Figure. V.12 et V.13 : Intersection floue et union

V.5. Logique floue avancée, Notation - Définitions

Comme il a été dit, un ensemble précis est un ensemble qui ne comporte que des valeurs distinctes. Si le jeu est booléen et discret, il est appelé jeu classique simple ou simple jeu. Si le jeu est flou et discret, il est appelé ensemble flou fragile. De plus, dans la logique booléenne classique, la fonction d'appartenance n'a que deux valeurs, 0 et 1. On peut même dire que la fonction d'appartenance est un autre ensemble précis avec seulement deux valeurs: $\mu(x_k)=\{0,1\}, \forall k \in N$ La définition de l'ensemble booléen est donc la suivante: $A=\{(x_k, \mu(x_k)) : \mu(x_k)=1\}, \forall k \in N$. Comme il n'y a qu'une seule valeur pour la fonction d'appartenance, elle peut être omise: $A=\{x_k\}$.

D'autre part, un ensemble flou contient des éléments avec des fonctions d'appartenance de 0 à 1: $\mu(x_k)=\{[0,1]\}, \forall k \in N$. Et par conséquent, un ensemble flou est défini comme:
$A=\{(\mu(x_k) : \mu(x_k)=\{[0,1]\}\}$ ou simplement $A=\{x_k, \mu(x_k)\}, \forall k \in N$. Une autre notation populaire que l'on trouve dans de nombreux manuels est $A=\{x_k / \mu(x_k)\}, \forall k \in N$.
On peut donc voir qu'un ensemble booléen est effectivement un sous-ensemble d'un ensemble flou. S'il existe un ensemble avec un seul élément, avec une fonction d'appartenance différente de 0 et 1, cet ensemble est appelé ensemble flou approprié.

En outre, on peut dire que tous les éléments d'un ensemble quelconque (Flou ou Booléen) peuvent être considérés comme appartenant à un ensemble complet. Par conséquent, on peut dire que les éléments d'un ensemble flou appartiennent à sous ensemble d'un univers de discours et que, par conséquent, cet ensemble n'est rien de plus qu'un sous-ensemble l'univers original du discours.

Un sous-ensemble flou A de X est appelé normal s'il existe au moins un élément $x \in X : \mu(x) = 1.$
La hauteur d'un sous-ensemble flou A de X est la valeur de la fonction d'appartenance maximale: $Height(A) = \max \mu(x_k) \forall k \in N$ .On peut voir que la hauteur d'un ensemble normal est 1.

La prise en charge d'un sous-ensemble flou A de X est un ensemble précis comprenant tous les éléments de A ayant une fonction d'appartenance autre que zéro:

$$Supp(A) = \{x_k : \mu(x_k) \succ 0\}, \forall k \in N$$

Le noyau d'un sous-ensemble flou A de X est un autre ensemble précis qui contient tous les éléments du sous-ensemble A ayant la fonction d'appartenance
1: $Core(A) = \{x_k : \mu(x_k) = 0\}, \forall k \in N$.
Le sous-ensemble d'un ensemble flou est défini comme suit:

$$A \subset B, if\ \mu_A(x_k) \le \mu_B | (x_k), \forall k \in N$$

Et enfin, la puissance d'un flou fixe la somme: $Power(A) = \sum_{k=1}^{n} \mu_A(x_k), k = \{1, 2, ..., n\}$

Exemples:
1. L'ensemble *A*= {1/1 ,3/0 ,4.5 j/1} n'est pas un ensemble flou approprié.
2. L'ensemble *B*= {1/1, 3/0, 4.5 j/0.1} est un ensemble flou approprié, puisque l'élément x=4.5j a une fonction d'appartenance de 0.1.
3. L'ensemble *C*={1/0.01,3/0,4.5 j/0.1 } n'est pas un ensemble flou normal.
4. L'ensemble *D*={1/0.01,3/1,4.5 j/0.1} est un ensemble flou normal, puisque l'élément 3 a une fonction d'appartenance de 1.

5. La hauteur de l'ensemble A est max (1, 0,1)=1. La hauteur de l'ensemble B est max(1,0,0.1)=1. La hauteur de l'ensemble C est max (0.01,0, 0.1) = 0.1.

6. Le support de l'ensemble flou *E*={1/0,3/1,4.5 /0.1} est l'ensemble *F* = {3,4.5}

7. Le noyau de l'ensemble flou E est l'ensemble $G = \{3\}$

8. Enfin le pouvoir de l'ensemble flou E est 0 + 1 + 0,1 = 1,1

V.6. Opérations sur les ensembles flous

Dans cette section, certains concepts communs de la logique booléenne sont développés de manière à inclure Ensembles flous (FS). À ce stade, il convient de noter que si les ensembles sont réduits à des valeurs booléennes normales, la signification de ces propriétés doit également être réduite à leur signification d'origine.

Supposons les deux sous-ensembles flous A et B de X. Leur union est un sous-ensemble flou C de X, noté $C = A \cup B$ tel que $\forall x \in X \quad \mu_c|(x) = \max(\mu_A(x), \mu_B(x))$.

Leur intersection est un sous-ensemble flou D de X, noté $D = A \cap B$ tel que $\forall x \in X$

$\mu D_c(x) = \min(\mu_A(x), \mu_B(x))$.
Certaines propriétés importantes sont

- $D \subset C$
- $A \subset C$ et $B \subset C$
- $D \subset A$ et $D \subset B$

Un autre moyen courant de calculer la valeur minimale et maximale de deux nombres consiste à utiliser les expressions suivantes:

- $\max(a,b) = \dfrac{a+b+|a-b|}{2}$
- $\min(a,b) = \dfrac{a+b-|a-b|}{2}$

Supposons que A et B sont deux sous-ensembles flous de X. Le complément relatif de B par rapport à A est un autre ensemble E = A-B, également un sous-ensemble de X défini comme suit: $\mu_E(x) = \max(0, \mu_A(x) - \mu_B(x))$
Supposons que A est un sous-ensemble de deux flous. Le complément ou la négation de A est un autre ensemble $\overline{A} = X - A$, également de X défini comme suit:
$\mu_{\overline{A}}(x) = 1 - \mu_A(x)$

Certaines propriétés de la négation sont:

- $\overline{A} = A$
- $\left.\begin{array}{l}(\overline{A \cup B} = \overline{A} \cap \overline{B} \\ (\overline{A \cap B}) = \overline{A} \cup \overline{B}\end{array}\right\}$ *De* Morgan's Law
- $\overline{\varnothing} = X$
  $\overline{X} = \varnothing$
- Si $A \subset B$ alors $\overline{A} \supset \overline{B}$ et $A - b = \varnothing$

Supposons A, un sous-ensemble flou de X et a > 0. Un autre sous-ensemble flou B = A^a de X peut être défini comme suit: $\mu_B(x) = (\mu_B(x))^a$

Il peut être prouvé que si a> 1 alors $A^a \subset A$ et si a $\prec$1 $A^a \supset A$.Si a> 1, l'opération ci-dessus est appelée opération de concentration et est appelée dilatation si a <1.

Une opération étroitement liée à ce qui précède est celle de l'intensification du contraste:

$$B = \{x, \mu_B(x) : \left\{\begin{array}{l}\mu_B(x) = (\mu_A(x))^2, \mu_A(x) \in [0, 0.5] \\ \mu_B(x) = (\mu_A(x))^{1/2}, \mu_A(x) \in [0.5, 1]\end{array}\right\}$$

La somme bornée de deux sous-ensembles flous A et B de X est un autre sous-ensemble $D = A \oplus B$ de X tel que $\mu_D(x) = \min(1, \mu_A(x) + \mu_B(x))$.

Supposons un sous-ensemble A de X et un nombre [1,0] ∈a, puis un nouvel ensemble peut être défini comme F=aA comme $\mu_F(x) = a\mu_A(x)$.

Supposons un sous-ensemble A de X; l'ensemble de niveau a de A, noté A^a, est le sous-ensemble net de X tel que $\mu_A(x) \geq a$ or : $A_a = \{x : \mu_A \geq a, \forall x \in X\}$

Exemples:

1. Si A={1/0.5,3/0.6,4/0.1} et B={2/0.7,3/0.2,4/0.8} leur union est
   $A \cup B = \{1/0.5, 2/0.7, 3/\max(0.6, 0.2), 4/\max(0.1, 0.8)\} \Leftrightarrow$
   $A \cup B = \{1/0.5, 2/0.7, 3/0.6, 4/0.8\}$
2. Si A={1/0.5,3/0.6,4/0.1} et B={2/0.7,3/0.2,4/0.8} leur l'intersection est
   {3/min(0.6,0.2),4/min(0.1,0.8)}={3/0.2,4/0.1}
3. Si $D = A \cap B$ et $C = A \cup B$ puis C={1/0.5,2/0.7,3/0.6,4/0.8} et D= {3/0.2,4/0.1}.

L'ensemble D peut également être écrit comme D= {1/0,2/0,3/0.2,4/0.1}.

On peut voir que pour tous les éléments de D et C $\mu_D(x) \prec \mu_C(x)$ Cela implique que $D \subset C$

4. également A={1/0.5,2/0,3/0.6,4/0.1}, B={1/0,2/0.7,3/0.2,4/0.8} et

C={1/0.5,2/0.7,3/0.6,4/0.8}. On peut voir que pour tous les éléments de A, B et C

$\mu_A(x) \prec \mu_C(x)$ et $\mu_B(x) \prec \mu_C(x)$ Cela implique que $A \subset C$ et $B \subset C$.

5. Si D={1/0 ,2/0,3/0.2,4/0.1} Par conséquent $\forall x \in X \quad \mu_A(x) \succ \mu_D(x)$ et $\mu_B(x) \succ \mu_D(x)$, donc $D \subset A$ et $D \subset B$.
6. Calculez les min et max de (0.1,0.5), (0.5,0.5) en utilisant

$$\max(a,b) = \frac{a+b+|a-b|}{2} \text{ et } \min(a,b) = \frac{a+b-|a-b|}{2}$$

a. $\max(0.1,0.5) = \frac{0.1+0.5+|0.1-0.5|}{2} = \frac{1}{2} = 0.5$

b. $\min(0.1,0.5) = \frac{0.1+0.5-|0.1-0.5|}{2} = \frac{0.6-0.4}{2} = 0.1$

c. $\max(0.5,0.5) = \frac{0.5+0.5+|0.5-0.5|}{2} = 0.5$

d. $\min(0.5,0.5) = \frac{0.5+0.5-|0.5-0.5|}{2} = 0.5$

7. Trouvez le complément pertinent de B par rapport à A. Les ensembles A et B sont A={1/0.5 ,3/0.6,4/0.1} , B={2/0.7 ,3/0.2,4/0.8 } $\mu_E(x) = \max(0, \mu_A(x) - \mu_B(x))$, Par conséquent E={1/0.5 ,2/0 ,3/0.4,4/0} .
8. Trouvez le complément de A: $\overline{A}$={1/0.5,3/0.4,4/0.9}

V.7. Relations floues

Supposons qu'il existe un ensemble X net avec les éléments x_1, x_2, .. x_n et un autre ensemble Y net avec les éléments y_1, y_2, .. y_m. Le produit cartésien de ces deux ensembles est un autre ensemble de toutes les paires: $X \times Y = \{(x_k, y_j), \forall k, j \in N\}$. Par conséquent, cet ensemble aura m x n éléments. Graphiquement:

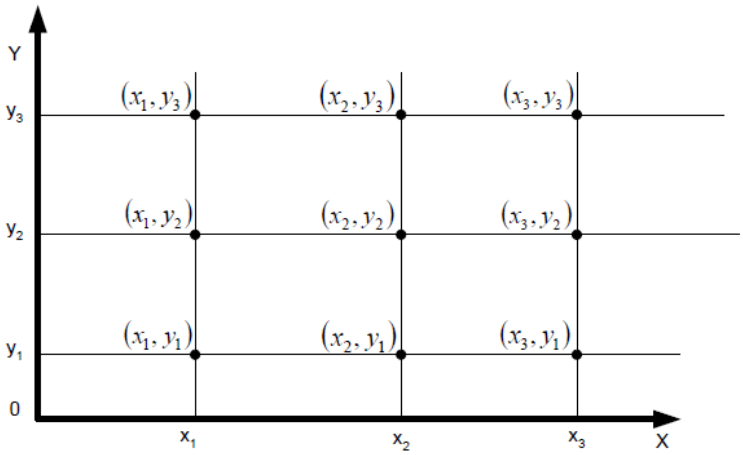


Figure. V.14 : Produit cartésien

Cet ensemble (X x Y) peut être considéré comme un univers de discours pour un ensemble flou A: $A=\{(x_k,y_j)/\mu_A(x_k,y_j),\forall k,j\in N\}$. L'ensemble flou A peut être aussi appelé une relation floue sur la paire X et Y.

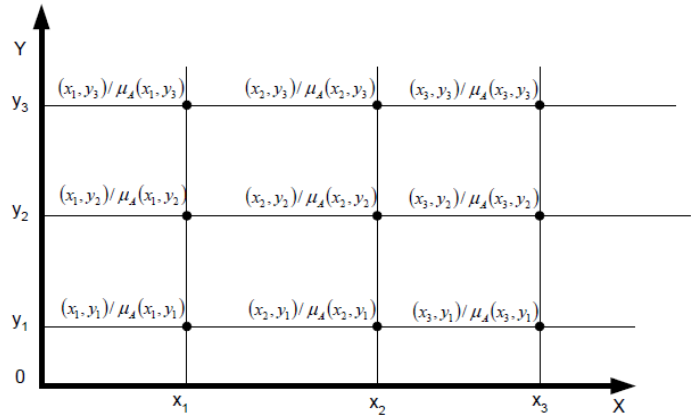


Figure. V.15 : Produit cartésien flou

La projection du sous-ensemble flou A sur X est un sous-ensemble flou $A^0=Proj_XA$ défini comme:

$$\mu_{A^0}(x_k)=\max(\mu_A(x_k,y_i))\forall k,j\in N$$

Ou :

$$\mu_{A^0}(x_k)=\max(\mu_A(x_k,y_1),\mu_A(x_k,y_2),\mu_A(x_k,y_3)....,\mu_A(x_k,y_m))$$

Supposons maintenant deux sous-ensembles flous A et B de X et Y respectivement, leur produit cartésien (A x B) est une relation floue sur l'ensemble X x Y, notée T = A x B où

$$\mu_T(x_k,y_i)=\min(\mu_A(x_k),\mu_B(y_i))\forall k,j\in N$$

Ou :

$$T=\{(x_k,y_j),\mu_T(x_k,y_j):\mu_T(x_k,y_j)=\min(\mu_A(x_k),\mu_B(y_j))\forall k,j\in N\}$$

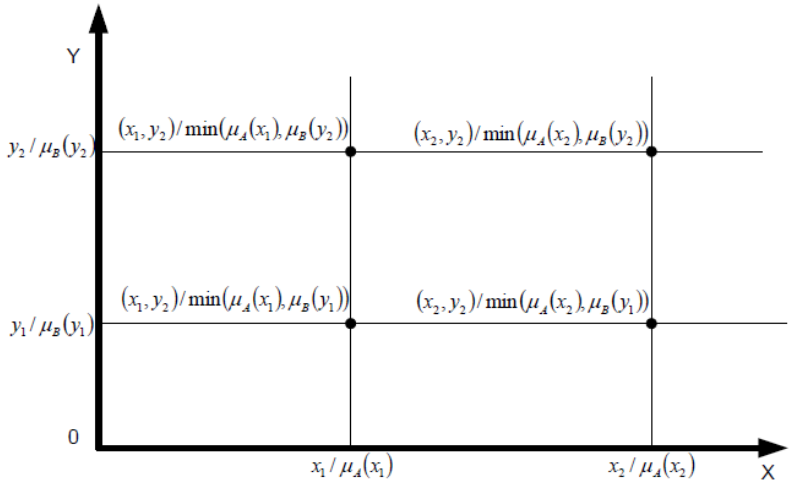


Figure. V.16 : Projection du produit cartésien flou

V.8. Principe d'extension

Une fonction est une relation qui associe de manière unique les membres d'un ensemble à ceux d'un autre. Plus formellement, une fonction de X à Y est un objet f telle que ∀x∈ X soit uniquement associé à un objet y ∈Y, f: X → Y. L'ensemble X est appelé domaine et l'ensemble Y est appelé plage ou image de X sous f.

La même chose peut maintenant être appliquée aux ensembles flous : Supposons un mappage f, de X à Y (X et Y sont des ensembles précis) tel que x∈ X ∃ y∈Y. Supposons également un sous-ensemble flou A de X: $A = \{x / \mu_A(x)\}$.

La fonction f, va mapper chaque élément de A dans Y comme f (x) et avec une fonction d'appartenance de $\mu_A(x)$: $f(A) = \cup_X \{ f(x) / \mu_A(x)\}$. Si B = f (A) alors :

$$\mu_B(y_j) = \max_{\forall x_k : f(x_k) = y_j} (\mu_A(x_k))$$

Le principe d'extension peut être utilisé pour des opérations arithmétiques avec des ensembles flous. Supposons trois ensembles précis: $X = \{x_1, x_2, ..., x_n\}, Y = \{y_1, y_2, ..., y_m\}$ et $Z = \{z_1, z_2, ..., z_k\}$ .L'ajout d'éléments de X et Y peut être envisagé être un mappage du produit cartésien X x Y en Z: $f : X\ X\ Y \rightarrow Z$ quand $f(x_i, y_j) = x_i + y_j = z_i, \forall i, j, l \in N$. Par conséquent :

$z = \{z_l : x_i + y_j = z_l\}, \forall i, j, l \in N$ Supposons maintenant trois sous-ensembles flous A, B, C de X, Y et Z respectivement, puis en utilisant le principe d'extension C = f (A + B) et

$$\mu_C(z_l) = \max_{\forall x_k : f(x_i, y_j) = x_i + y_j = z_l} (\min(\mu_A(x_i), \mu_B(y_j))) .$$

La même chose peut être appliquée à toute opération arithmétique.

V.9. Mesures du flou

Il a été dit qu'un ensemble flou contient des éléments dont la fonction d'appartenance est comprise entre 0 et 1, tandis que les sous ensembles booléens ont des fonctions d'appartenance égales à 0 ou 1. Par conséquent, plus la répartition des fonctions d'appartenance d'un ensemble flou est proche de 0 ou 1 est, moins flou est cet ensemble flou. Moins flou est un ensemble flou et plus grande est la différence entre l'ensemble flou et son complément. Par conséquent, on peut dire que la différence entre un ensemble flou et son complément peut décrire le flou de cet ensemble flou.

Assumer un univers de discours $X = \{x_1, x_2, ..., x_n\}$ ou $n \prec \infty$.Cet ensemble peut être écrit sous la forme d'un vecteur colonne tel que : $X = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$

De plus, les valeurs des fonctions d'appartenance d'un sous-ensemble flou A de X peuvent être écrit sous la forme d'un vecteur: $\mu_A(x) = \begin{bmatrix} \mu_A(x_1) \\ \mu_A(x_2) \\ . \\ . \\ \mu_A(x_n) \end{bmatrix}$.

La fonction d'appartenance du complément de A est donc:

: $\mu_{\bar{A}}(x) = \begin{bmatrix} 1-\mu_A(x_1) \\ 1-\mu_A(x_2) \\ . \\ . \\ 1-\mu_A(x_n) \end{bmatrix}$.

La différence maintenant entre ces deux vecteurs est une indication du flou d'un ensemble flou:

$D(A,\overline{A}) = \mu_A(x) - \mu_{\overline{A}}(x) = \begin{bmatrix} \mu_A(x_1) \\ \mu_A(x_2) \\ . \\ . \\ \mu_A(x_n) \end{bmatrix} - \begin{bmatrix} 1-\mu_A(x_1) \\ 1-\mu_A(x_2) \\ . \\ . \\ 1-\mu_A(x_n) \end{bmatrix} = \begin{bmatrix} 2\mu_A(x_1)-1 \\ 2\mu_A(x_2)-1 \\ . \\ . \\ 2\mu_A(x_n)-1 \end{bmatrix}$. Ainsi, la longueur de ce vecteur peut décrire le flou. La longueur d'un vecteur peut être trouvée par sa norme :

$\left|D(A,\overline{A})\right|_P = \left[\sum_{i=1}^{n} \left|2\mu_A(x_i)-1\right|^P\right]^{1/P}$ pour P=1, 2,3,…

Notez que si A n'est pas flou alors $\left|D(A,\overline{A})\right|_P = \left[\sum_{i=1}^{n} \left|2-1\right|^P\right]^{1/P} = \left[\sum_{i=1}^{n} 1\right]^{1/P} = n^{1/P}$

et c'est la distance maximale entre les deux ensembles.
Le flou d'un ensemble flou peut maintenant être décrit comme suit :

$FUZ_p = \dfrac{n^{1/P} - \left|D(A,\overline{A})\right|_P}{n^{1/P}}$, ce qui donnera 0 si le jeu est booléen.

Avoir le maximum FUZ_p la $\left|D(A,\overline{A})\right|_P$ doit être zéro. Par conséquent

$\mu_A(x_i) = 0.5, i = 1,2...n$

V.10. Variables linguistiques et fuzzification

Le raisonnement humain n'utilise pas de valeurs numériques, c'est-à-dire qu'il ne dit pas que la température ici est de 15,5 degrés et permet donc d'allumer l'appareil de chauffage. Le raisonnement humain dit qu'il fait froid ici, allumez l'appareil de chauffage. Il s'agit d'une variable linguistique au lieu d'une variable numérique. Pour cette raison, la FL utilise des variables linguistiques pour nommer les ensembles flous. Le premier processus du contrôleur de logique floue consiste à mapper toutes les entrées possibles à des ensembles flous et à attribuer des valeurs correspondant à leur importance. Ce processus s'appelle fuzzification. Par exemple, supposons que le prochain ensemble flou:

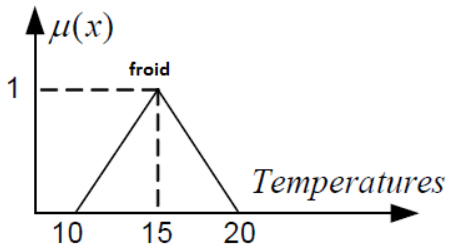


Figure. V.17 : Ensemble flou «froid»

Ensuite, une température de 11° C va être assignée à la valeur de 0.2 pour le froid réglé.

L'univers global doit être entièrement couvert par des ensembles flous. Ces ensembles flous peuvent se chevaucher ou non et avoir différentes formes:

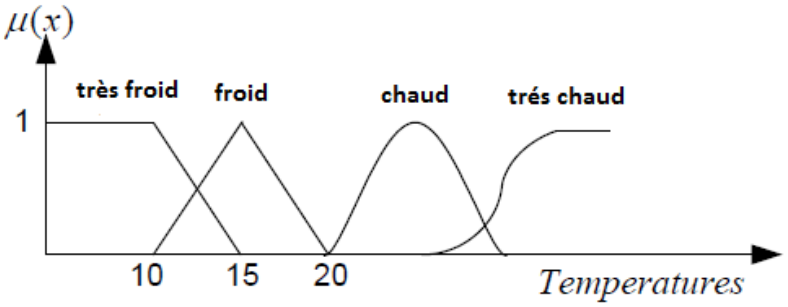


Figure. V.18 : Ensemble flou «froid»

On peut donc dire maintenant que la température 11 appartient au groupe flou «Froid» à 20% et au groupe flou «Très Froid» à 80%.

Ainsi, chaque entrée réelle (dans ce cas la température) est mappée sur un ensemble flou spécifique et a une valeur spécifique pour décrire à quel point elle appartient ou non à cet ensemble. Cette valeur dépendra directement de la forme de la fonction d'appartenance.

V.11. Inférence Floue

Le système d'inférence flou (FIS) est la partie du contrôleur flou (FLC) qui connecte les entrées fuzzifiées pour produire des ensembles flous. C'est le point essentiel de la FL, qui consiste à imiter le raisonnement humain. Ce raisonnement peut être décrit par les règles SI et ALORS. Par exemple, «SI la pièce est froide ALORS allumer le chauffage au maximum» ou «SI la vitesse est élevée ALORS appuyer sur le frein au maximum». La partie SI est appelée "prémisse" et la partie ALORS, "conclusion". C'est exactement la structure d'un contrôleur FL (FLC). La partie SI sera l'entrée du contrôleur et la partie ALORS la sortie. Les entrées et les sorties seront décrites par des ensembles flous.

Par exemple, supposons que le contrôleur doit déterminer le niveau de freinage en fonction de la vitesse d'une voiture. Ensuite, l'entrée sera la vitesse et le niveau de freinage la sortie. Supposons également que les univers du discours pour l'entrée et la sortie ne contiennent qu'un seul ensemble flou. [91]

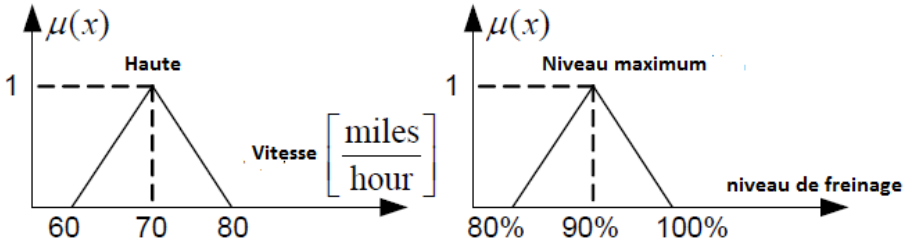


Figure. V.19 & V.20 : Entrée et sortie pour un contrôleur flou

Supposons également que la vitesse est de 65 miles / heure. Cette vitesse fait partie du jeu flou «Élevé» à 50% ou à 0,5. Cette valeur est également appelée degré de prise en charge de cette règle. Supposons également que la règle IF-THEN utilisée est la suivante : Si la vitesse est élevée, le niveau de freinage est maximal. Cela signifie que la vitesse de 65 miles par heure est partiellement vraie pour l'ensemble “High”. Mais le jeu “High” est connecté à la sortie définie “Maximum” par la règle précédente. Puisque maintenant l'entrée était partiellement vraie, l'ensemble de sortie est également partiellement vrai avec le même degré que l'entrée, c'est-à-dire 50%:

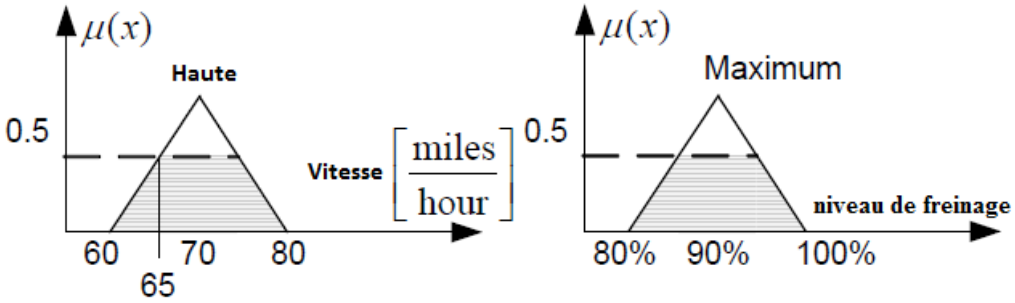


Figure. **V.20 et V.21** : Entrée et sortie pour un contrôleur flou utilisant la méthode min

Puisque l'ensemble flou de la sortie a été tronqué jusqu'à la valeur de 0,5, cette méthode est appelée méthode «min». Une autre approche populaire est la méthode dite prod ou squash où le jeu flou en sortie est mis à l'échelle:

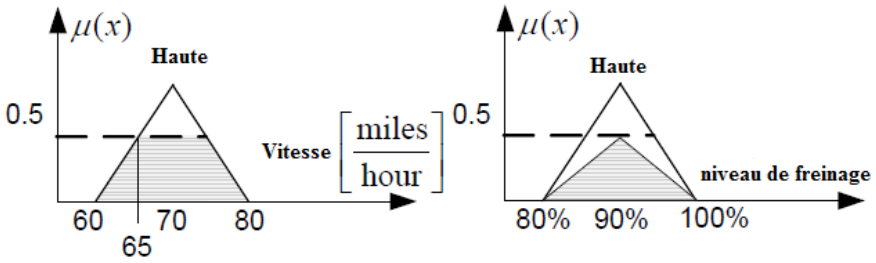


Figure. V.23 & V.24 : Entrée et sortie pour un contrôleur flou utilisant la méthode prod

La valeur maintenant de la sortie sera déterminée par la méthode de défuzzification qui sera utilisée. Dans certaines applications, une entrée ne peut pas décrire complètement une situation. Par exemple, pour que l'homme puisse décider d'appuyer sur le frein, les informations relatives à la vitesse ne suffisent pas. La distance de la voiture suivante ou la limite de vitesse maximale seraient également nécessaires. Par conséquent, pour décrire complètement une situation avec des règles IF-THEN, des opérateurs logiques doivent être utilisés. Le problème ici est de trouver le degré de soutien pour cette règle. Si la phrase

comporte l'opérateur «ou», le maximum entre les deux nombres sera utilisé. Sinon, si la règle comporte l'opérateur «et», le minimum est pris: Si la vitesse est élevée ou si la voiture suivante est très proche, appuyez sur le frein au maximum.

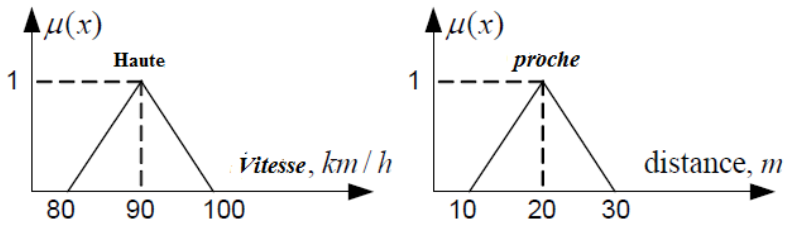


Figure. V.25 et V.26 : Entrée et sortie pour un contrôleur flou utilisant la méthode prod

Supposons que la vitesse appartient à la valeur floue «Élevée» pour 0,5 et à la distance jusqu'au paramètre flou «Fermer» à 0,6. Puisque la règle est «ou», l'opération max sera utilisée, c'est-à-dire que le degré de support est max (0.5, 0.6) = 0.6, d'où:

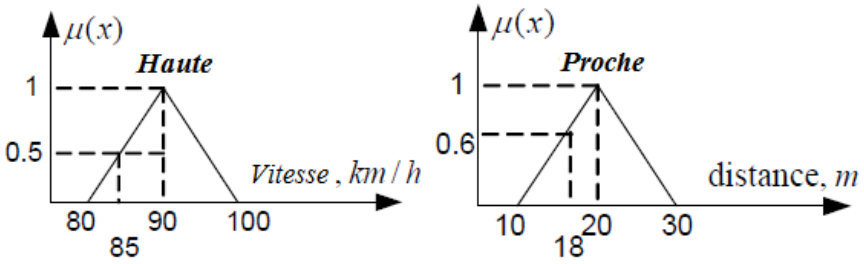


Figure. V.27 & V.28 : 2 Entrées floue

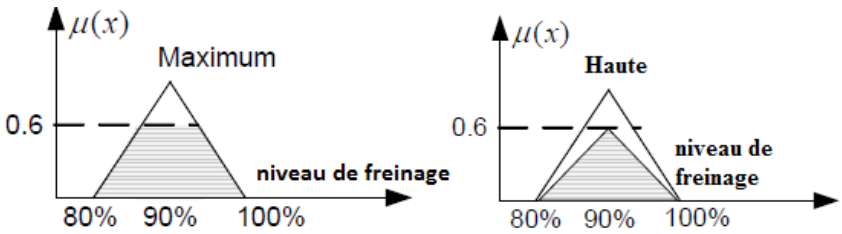


Figure. V.29 & V.30 : Sortie floue des méthodes min et prod

Afin de produire un schéma de contrôle satisfaisant, une règle floue n'est pas suffisante. Dans l'exemple du freinage en vitesse, le compensateur ne fonctionnerait pas si la vitesse était très basse ou très élevée. D'une manière générale plus l'ensemble flou recouvre les entrées et sorties des univers de discours meilleure sera le contrôle. Par exemple, le contrôleur précédent ne ferait rien si la vitesse était de 79 km / h. Si plus d'ensembles sont utilisés, alors :

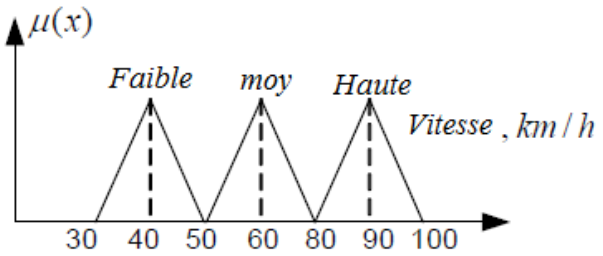


Figure. V.31 : Fuzzification d'entrée en utilisant 3 ensembles

Et la sortie:

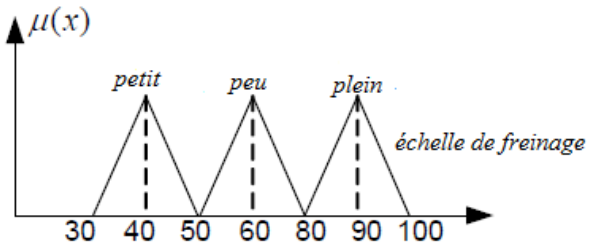


Figure. V.32 : Univers global de sortie avec 3 ensembles

Règles typiques Si - Alors:

1. Si la vitesse est basse, le frein est petit (faiblement actionné)

2. Si la vitesse est moyenne alors le frein est un peu (moyennement actionné)

3. Si la vitesse est élevée, le frein est plein (fortement actionné)

Par conséquent, si la vitesse est de 35 km / h, l'entrée est :

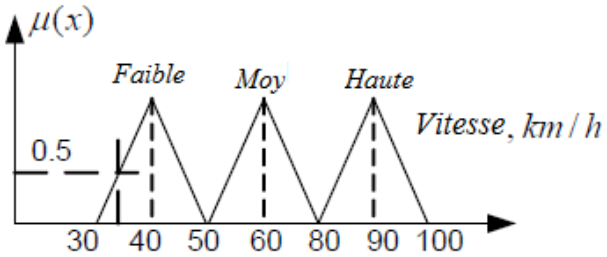


Figure. V.33 : Univers d'entrée du discours, lorsque l'entrée réelle est de 35 km / h

Et donc le résultat en utilisant la méthode min est:

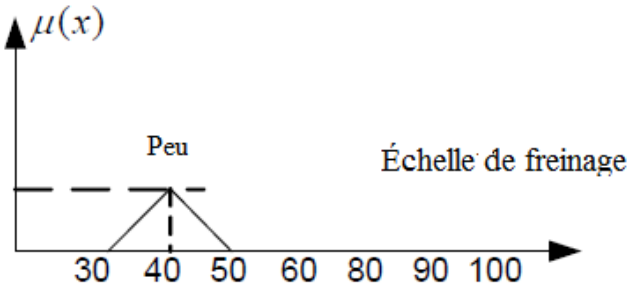


Figure. V.34 : Univers de discours pondéré en sortie, lorsque l'entrée réelle est de 35 km / h

Une meilleure couverture de l'univers du discours inclurait le chevauchement des ensembles flous:

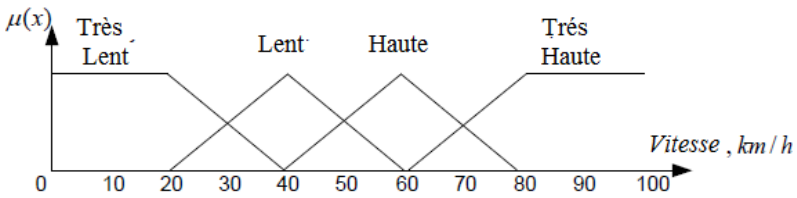


Figure. V.35 : Fuzzification d'entrée à l'aide de 4 ensembles superposés

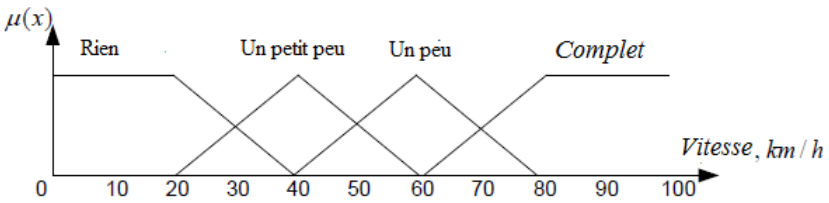


Figure. V.36 : Fuzzification de la sortie en utilisant 4 ensembles superposés

Et les règles IF - THEN:

1. Si la vitesse == très basse puis le frein == rien

2. Si la vitesse == faible puis le frein == un petit peu

3. Si la vitesse == élevée puis freine == un peu

4. Si la vitesse == très élevée puis le frein == complet

Supposons maintenant que la vitesse est de 25 km / h. Cette vitesse correspond à la valeur très basse 0.75 et à la valeur basse 0.25. Cela implique que le degré de soutien pour la règle 1 est

0.75 et le degré de soutien pour la règle 2 est 0.25. Par conséquent, les deux ensembles flous de sortie sont les suivants:

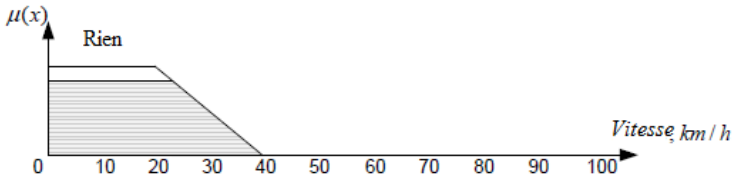


Figure. V.37 : ensemble flou de sortie pour la règle 1

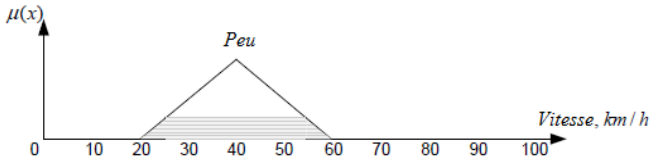


Figure. V.38 : Fuzzification de la sortie en utilisant 4 ensembles superposés

Maintenant, le problème est de trouver un moyen d'extraire l'ensemble de sorties floues globales. Il existe de nombreuses façons d'agréger toutes les sorties pour produire un ensemble flou de sortie globale. Les trois plus communs sont:

• Max (maximum)

• Prodor (probabiliste ou)

• Somme

En utilisant maintenant l'opérateur maximum, l'ensemble flou global est le suivant:

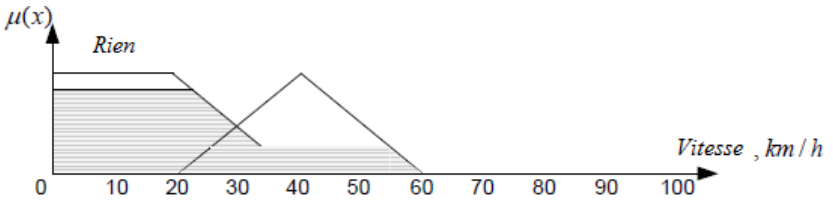


Figure. V.39 : Sortie floue définie à l'aide de l'opérateur max.

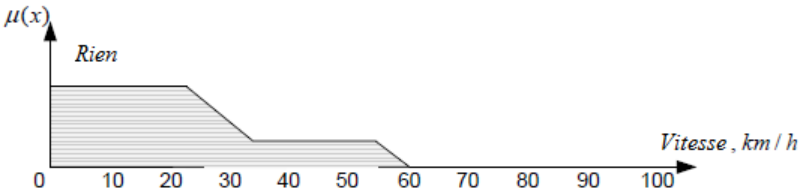


Figure. V.40 : Sortie floue définie à l'aide de l'opérateur max

V.12. Défuzzification

La dernière étape de la conception d'un FLC consiste à défuzzifier le flou de sortie réglé sur une valeur réelle, le signal de contrôle. Il n'existe pas de méthode claire pour défuzzifier le résultat et il n'existe certainement pas de base théorique ou de méthodologie pour laquelle on est meilleur. Tout dépend de l'application spécifique et de l'expérience du concepteur.

V.12.1 maximum

Dans cette méthode, la sortie réelle est la valeur qui correspond à la valeur maximale de l'ensemble flou de sortie.

$$out = y : \mu(y) = \max$$

Cette méthode est très simple, mais elle présente des inconvénients importants lorsque les ensembles flous en sortie ont plusieurs maxima. Par exemple, il est très difficile de l'utiliser dans l'ensemble flou de la Fig. V.37

V.12.2 Moyenne des maxima (MOM)

Dans cette méthode, la valeur moyenne de tous les maxima est calculée. Par conséquent, le problème précédent est surmonté. Évidemment, s'il n'y a qu'un maximum, cette méthode donnera les mêmes résultats qu'auparavant:

$$out = \frac{1}{m}\sum_{j=1}^{m} y_j : \mu(y_j) = \max$$

V.12.3 Centre de la surface (COA)

Dans cette méthode, la sortie est calculée par le centre du volume global de l'ensemble flou de sortie:

$$out = \frac{\sum_{i=1}^{m} y_j \mu(y_j)}{\sum_{i=1}^{m} \mu(y_j)}$$

V.12.4 Centre de gravité (COG) et autres

Dans cette méthode, la sortie est calculée par le centre de gravité du jeu flou de sortie.
Les autres méthodes (incluses dans Matlab) sont:
• Plus grand du maximum
• Plus petit des maximum

V.13 Avantages et inconvénients de réglage par logique floue

a- Avantages

Les avantages principaux des régulateurs flous sont les suivants :

• L'incorporation directe des informations floues et linguistiques, provenant d'un expert humain, dans le système flou.

• Il n'y a pas nécessaire de faire un modèle mathématique du système à régler.

• Le système flou est un approximateur universel, c'est-à-dire, il est suffisamment général pour générer n'importe quelle action.

• La logique floue est facile à comprendre par ceux qui ne sont pas des spécialistes, car elle imite la stratégie du raisonnement humain.

• On peut maîtriser les systèmes non linéaires et difficiles à modéliser.

b- Inconvénients

• Manque de directives précises pour la conception d'un régulateur,

• Précision de réglage en général peu élevée.

Théories des réseaux de neurone et Neuro-Floue

V.14. Les réseaux de neurones :

V.14.1. Introduction :

Les réseaux de neurones sont des ensembles d'éléments de base appelés neurones. La philosophie derrière ces réseaux de neurones est d'imiter le cerveau humain, mais l'écart entre les réseaux de neurones et le cerveau est toujours grand, dû à la complexité de ce dernier. Cette complexité et une connaissance toujours améliorée du cerveau ont amené une multitude de solutions pour la conception des réseaux de neurones. L'absence de normalisation ajoute aussi à la difficulté de présenter clairement une théorie sur les réseaux de neurones. Nous allons présenter certains concepts qui sont normalement respectés. Les réseaux de neurones se modifient pour tenir compte de leur environnement. Cette capacité d'apprendre à partir d'exemples est d'un grand intérêt et elle s'apparente à celle du cerveau. Les études des réseaux de neurones artificiels (RNA) datent depuis les années 1940.Grâce aux développements des recherches sur le cerveau et la disponibilité des outils de simulation, les chercheurs étudièrent des ensembles de neurones formels interconnectés[92]. Ces réseaux, déjà développés à l'époque, permettaient d'effectuer quelques opérations logiques simples. Jusqu'aux années 1980, la recherche était freinée par la limitation théorique du perceptron. Peu après cette époque, Hopfield lança de nouveau en 1982 la recherche dans ce domaine après avoir montré l'analogie entre les RNA et les systèmes physiques. Après les années 1990, quelques travaux scientifiques ont vu le jour dans le domaine de la robotique, parmi ces

applications, on trouve la commande des systèmes d'entraînement et des systèmes de positionnement de haute performance.

V.15 Le neurone biologique

Les réseaux neurones artificiels sont inspirés des neurones biologiques. Le cerveau se compose d'un grand nombre de neurones (approximativement 10^{11}) fortement connectés. Les neurones ont trois composants principal : Le corps cellulaire, les dendrites, l'axone. [93] La jonction entre deux neurones est appelée la synapse (Figure V.41).

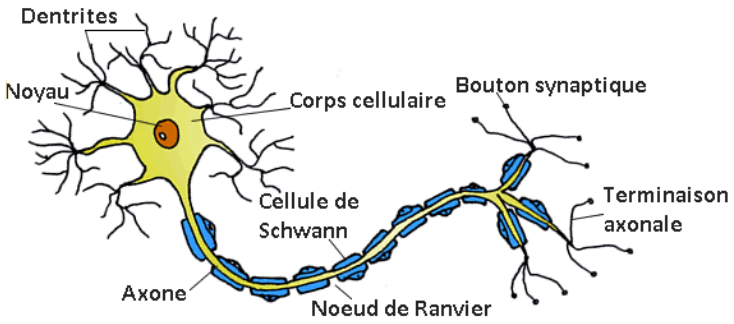


Figure. V.41 : Schéma d'un neurone biologique.

1-Le corps cellulaire :

Il contient le noyau du neurone ainsi que la machine biochimique nécessaire à la synthèse des enzymes. Ce corps cellulaire de forme sphérique ou pyramidale contient aussi les autres molécules essentielles à la vie de la cellule. Sa taille est de quelques microns de diamètre. [94]

2-Les dendrites :

Ce sont de fines extensions tubulaires qui se ramifient autour du neurone et forment une sorte de vaste arborescence. Les signaux envoyés au neurone sont captés par les dendrites. Leur taille est de quelques dizaines de microns de longueur. [94]

3-L'axone :

C'est le long de l'axone que les signaux partent du neurone. Contrairement aux dendrites qui se ramifient autour du neurone, l'axone est plus long et se ramifie à son extrémité ou il se connecte aux dendrites des autres neurones. Sa taille peut varier entre quelques millimètres à plusieurs mètres.[*94*]

4- Synapse

Une synapse est une jonction entre deux neurones, et généralement entre l'axone d'un neurone et une dendrite d'un autre neurone (mais il existe aussi des synapses axo-axonales par exemple).

V.16 Fonctionnement

Au point de vu fonctionnel, il faut considérer le neurone comme une entité polarisée, c'est-à-dire que l'information ne se transmet que dans un seul sens : des dendrites vers l'axone.

Le neurone va donc recevoir des informations, venant d'autres neurones, grâce à ses dendrites. Il va ensuite y avoir sommation, au niveau du corps cellulaire, de toutes ces informations et via un potentiel d'action (un signal électrique) le résultat de l'analyse va transiter le long de l'axone jusqu'aux terminaisons synaptiques. A cet endroit, lors de l'arrivée du signal, des vésicules synaptiques vont venir fusionner avec la membrane cellulaire, ce qui va permettre la libération des neurotransmetteurs (médiateurs chimiques) dans la fente synaptique. Le signal électrique ne pouvant pas passer la synapse (dans le cas d'une synapse chimique), les neurotransmetteurs permettent donc le passage des informations, d'un neurone à un autre.

Les neurotransmetteurs excitent (neurotransmetteurs excitateurs) ou inhibent (neurotransmetteurs inhibiteurs) le neurone suivant et peuvent ainsi générer ou interdire la propagation d'un nouvel influx nerveux. En effet, au niveau post-synaptique, sur la membrane dendritique, se trouvent des récepteurs pour les neurotransmetteurs. Suivant le type de neurotransmetteur et le type des récepteurs, l'excitabilité du neurone suivant va augmenter ou diminuer, ce qui fera se propager ou non information. Les synapses possèdent une sorte de «mémoire» qui leur permet d'ajuster leur fonctionnement. En fonction de leur «histoire», c'est-à-dire de leur activation répétée ou non entre deux neurones, les connexions synaptiques vont donc se modifier. Ainsi, la synapse va faciliter ou non le passage des influx nerveux. Cette plasticité est à l'origine des mécanismes d'apprentissage.

V.17 Neurone formel

Le neurone formel est un modèle mathématique simplifié du neurone biologique. Il présente un certain nombre d'entrées, les dendrites, un corps traitant les entrées suivant la méthode du tout ou rien, et un axone véhiculant la réponse du neurone. La première modélisation d'un neurone découle des résultats des travaux significatifs de Mac Culloch et Pitts (1943) [95].

La (Figure V.42) représente un modèle de base d'un neurone formel.

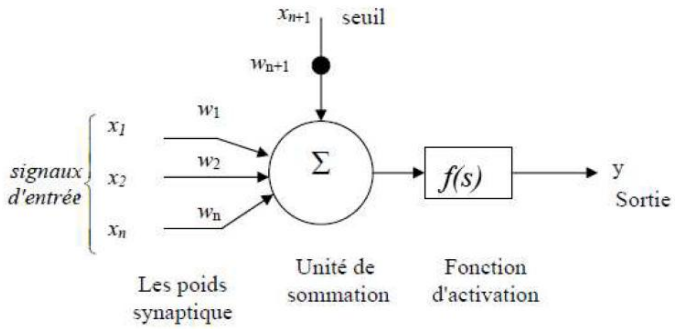


Figure. V.42 : Le modèle d'un neurone formel.

Un neurone formel est une fonction algébrique non linéaire et bornée, dont la valeur dépend des paramètres appelés poids synaptiques ou poids des connexions.D'une façon plus générale, un neurone formel est un élément de traitement (opérateur mathématique) possédant *n* entées (qui sont les neurones externes ou les sorties des autres neurones), et une seule sortie. Ce modèle est décrit mathématiquement par les équations suivantes:

$$s = \sum_{i=1}^{n+1} w_i x_i$$

$$y = f(s) \qquad \text{(V.11)}$$

Où w_i , x_i , f , y sont respectivement, les poids synaptiques (paramètres), les entrées, la fonction d'activation et la sortie du neurone.[96]

V.18. Architecture des réseaux

a- Réseau monocouche

La structure d'un réseau monocouche est telle que des neurones organisés en entrée soient entièrement connectés à d'autres neurones organisés en sortie par une couche modifiable de poids (figure. V.43).

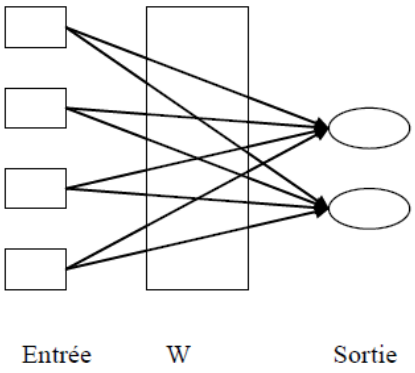


Figure. V.43 : Réseau Monocouche.

b- Réseau multicouche

Couche d'entrée, l'ensemble des neurones d'entrée, couche de sortie, l'ensemble des neurones de sortie. Les couches intermédiaires n'ayant aucun contact avec l'extérieur sont appelées couches cachées (figure. V.44).

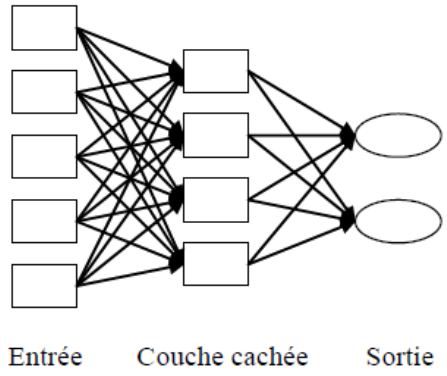


Figure. V.44 : Réseau multicouche.

c- Réseau à connexion complète

C'est la structure d'interconnexion la plus générale. Chaque neurone est connecté à tous les neurones du réseau (et à lui-même) (figure. V.45).

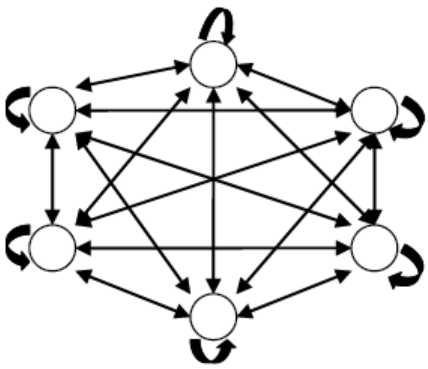


Figure. V.45 : Réseau à connexion complète.

d- Réseau à connexions locales

Il s'agit d'une structure multicouche, mais Chaque neurone entretient des relations avec un nombre réduit et localisé de neurones de la couche avale. Les connexions sont donc moins nombreuses que dans le cas d'un réseau multicouche classique (figure. V.46).

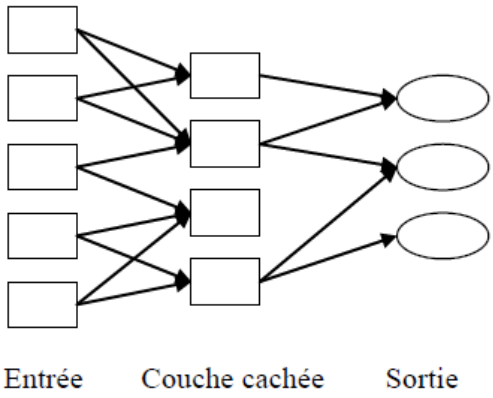


Figure. V.46 : Réseau à connexion locales.

V.19 La fonction d'activation:

Les fonctions d'activations représentent généralement certaines formes de non linéarité. L'une des formes de non linéarité la plus simple, et qui est appropriée aux réseaux discret, est la fonction signe (Figure V.47.a). Une autre variante de ce type des non linéarités est la fonction de Heaviside (Figure V.47.b).Pour la majorité des algorithmes d'apprentissage il est nécessaire d'utiliser des fonctions sigmoïdes différentiables, telles que la fonction sigmoïde unipolaire, Figure V.47.c et la fonction sigmoïde bipolaire (Figure V.47.d). La classe, la plus utilisée des fonctions d'activation, dans le domaine de la modélisation et de la commande des systèmes non linéaires est la fonction sigmoïde bipolaire.[96]

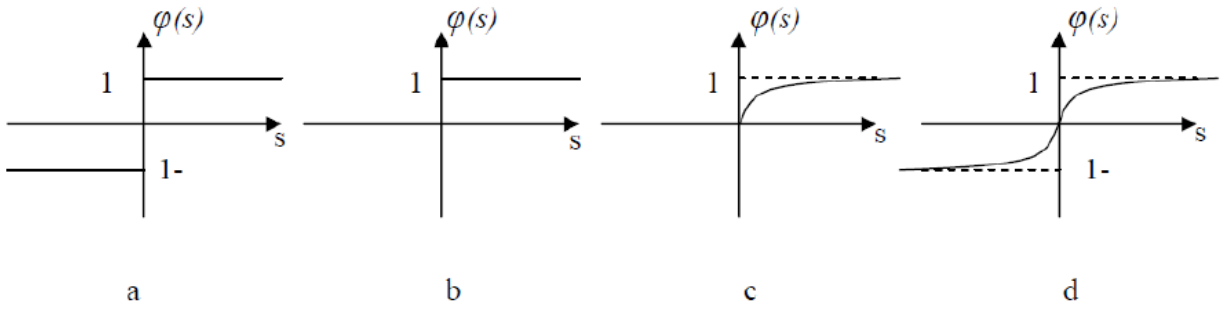


Figure V.47 : Différents types de fonctions d'activation pour le neurone formel,

V.20 Les réseaux multicouches

Nous présentons ici une des architectures de réseaux les plus utilisées.

Elle correspond à une organisation des neurones en *n* couches successives (*n*_3). La première couche, dont les neurones voient leur activation forcée à la valeur des données d'entrée, est appelée couche d'entrée. La dernière est appelée couche de sortie. Les seules connexions

présentes dans ce type de réseau relient chaque neurone avec l'ensemble de ceux de la couche suivante (voir la figure suivante). La propagation de l'information se déroule ainsi en sens unique depuis la couche d'entrée vers la couche de sortie. La fonction d'activation utilisée pour les neurones peut être n'importe quelle fonction croissante et dérivable. On utilise souvent une fonction sigmoïde telle que :

$$f(x) = \frac{1}{1+\exp(-x)} \qquad \text{(V.12)}$$

Elle prend pour paramètre la somme pondérée des entrées du neurone :

$$s_i = \sum_j w_{ij} x_j + b_i \qquad \text{(V.13)}$$

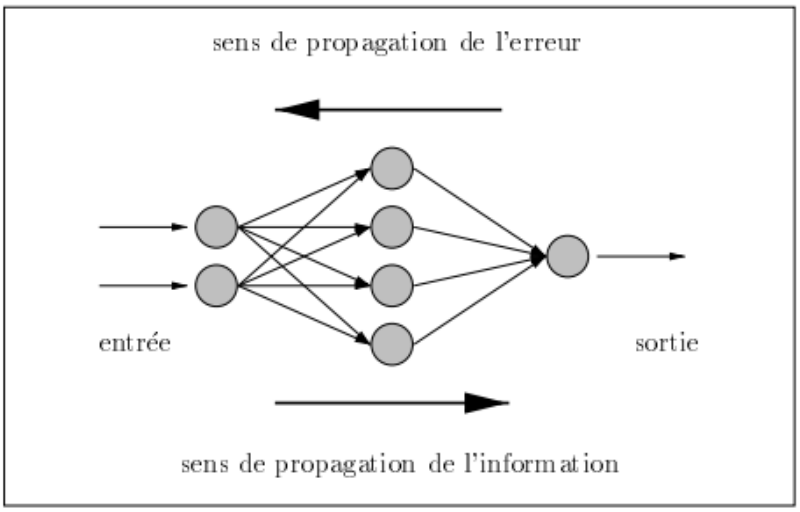


Figure V.48 : Un réseau multi-couches comportant 2 neurones d'entrée, 4 neurones cachés et un neurone de sortie.

Où j parcourt l'ensemble des neurones envoyant une connexion vers le neurone i, w_{ij} est le poids de la connexion entre le neurone j et le neurone i et, b_i est un paramètre optionnel appelé biais.[97]

V.21 Apprentissage des réseaux de neurones

Le besoin d'apprentissage se manifeste lorsque l'information à priori est incomplète, et son type dépend du degré de plénitude de cette information, Comme l'information que peut acquérir un réseau de neurones est représentée dans les poids des connexions entre les neurones, l'apprentissage consiste donc à ajuster ces poids de telle façon que le réseau présente certains comportements désirés. Mathématiquement l'apprentissage est défini par :

$$\frac{\partial w}{dt} \neq 0 \qquad \text{(V.14)}$$

Où *w* est la matrice des poids.

On peut distinguer trois types d'apprentissage, un apprentissage supervisé, apprentissage non supervisé et apprentissage par renforcement. [97]

V.21.1 L'apprentissage supervisé

Ce type d'apprentissage nécessite que la réponse désirée du système à entrainer soit connue à priori (présence d'un maître qui fournit la réponse désiré), et il est effectue de la façon suivante : en présente au réseau les valeurs d'entrée et on calcule sa sortie actuelle correspondante ensuite les poids sont ajustés de façon à réduire l'écart entre la réponse désirée et celle du réseau en utilisant l'erreur de sortie (la différence entre 1a réponse du réseau et celle désirée). Cette procédure est répétée jusqu'a ce qu'un critère de performance soit satisfait. Une fois 1a procédure d'apprentissage est achevée, les coefficients synaptiques prennent des valeurs optimales au regard des configurations mémorisées et le réseau peut être opérationnel. [98]

V.21.2 Apprentissage non supervisé

Dans ce cas, la connaissance à priori de la sortie désirée n'est pas nécessaire, et la procédure d'apprentissage est basée uniquement sur les valeurs d'entrées. Le réseau s'auto organisé de façon à optimisé une certaine fonction de coût, sans qu'on lui fournit 1a réponse désirée. Cette propriété est appelée auto-organisation.[98]

V.21.3 Apprentissage par renforcement

L'idée de base de l'apprentissage par renforcement est inspirée des mécanismes d'apprentissage chez les animaux. Dans ce type d'apprentissage on suppose qu'il n'existe pas de maître (superviseur) qui peut fournir la réponse correcte, mais le système à entrainer est informé, d'une manière indirecte, sur l'effet de son action choisie. Cette action est renforcée si elle conduit à une amélioration des performances du système entrainé, et les éléments qui contribuent dans La génération de cette action sont soit récompensés ou punis [98].

V.22. La rétro propagation du gradient de l'erreur

L'algorithme de rétro propagation du gradient de l'erreur à été créé en généralisant les règles d'apprentissage de Widrow Hoff aux réseaux multicouches à fonction de transfert non

linéaire. C'est un algorithme utilisé avec des réseaux de types feedforward pour l'apprentissage de fonction, la reconnaissance de forme et la classification [99].

1. Principe

La rétro propagation du gradient de l'erreur est utilisée pour ajuster les poids et les biais du réseau afin de minimiser l'erreur quadratique entre la sortie du réseau et la sortie réelle. A chaque couple entrée/sortie, une erreur est calculée, le gradient, ou pente de l'erreur est déterminé. Ensuite les poids et les biais sont modifiés en ligne sur le réseau. On réitère ces calculs jusqu'à l'obtention du critère d'arrêt.

2. Algorithme

L'algorithme de la rétro propagation du gradient de l'erreur se résume aux étapes suivantes :

1.Initialisation des poids $w^{[q]}$ à des petites valeurs aléatoires.

2. Présentation d'une entrée x_k et de la sortie désirée d_k .

3. Calcul de la sortie actuelle par propagation à travers les couches :

$$y_j^{[q]} = F(\sum_i w_{ji}^{[q]}.y_i^{[q-1]}) \qquad \textbf{(V.15)}$$

Où F est la fonction de transfert du neurone et $[q]$ la $q^{ième}$ couche du réseau.

4. Accumulation des erreurs en sortie :

$$\varepsilon = \sum (d_k - y_k^{[s]})^2 \qquad \text{(V.16)}$$

Où d_k est la sortie désirée associée au vecteur d'entrée x_k .

$y_k^{[s]}$ est la sortie obtenue sur la dernière couche au temps *t*.

ε est l'erreur cumulée pour k présentations de couples (x_k, d_k)

5. Rétro propagation du gradient de l'erreur (δ) depuis la dernière couche vers la première couche :

- Pour chaque cellule de sortie :

$$\delta_i^{[s]} = -(d_i - y_i^{[s]}).F(p_i^{[s]}) \qquad \text{(V.17)}$$

- Pour chaque cellule cachée :

$$\delta_i^{[q]} = -\sum_k \delta_k^{[q+1]}.w_{ki}.F(p_i^{[s]}) \qquad \text{(V.18)}$$

6.Mise à jour des poids selon la règle :

$$\Delta w_{ij}^{[q]} = \alpha.(\delta_i^{[q]}.x_i^{[q]}) \qquad \text{(V.19)}$$

Ou α est le coefficient d'apprentissage compris dans l'intervalle [0,1].

7. Retour à 2 tant qu'il ya des couples à présenter.

3. Choix du critère à minimiser

Dans le cas de la retro propagation de l'erreur à minimiser est un erreur quadratique .L'application de l'algorithme du gradient nécessite la dérivabilité de la fonction de transfert [ben brahim et kurosawa, 1993] .Le critère de minimisation d'erreur est le suivant :

$$\varepsilon = \sum (d_k - y_k^{[s]})^2 \qquad \text{(V.20)}$$

L'algorithme présenté ici est de type « on-line » c'est-à-dire que l'on met a jour les poids pour chaque échantillon d'apprentissage présenté dans le réseau de neurones .Une autre méthode est dite en « batch »,c'est-à-dire que l'on calcule d'abord les erreurs pour tous les échantillons sans mettre à jour les poids (on additionne les erreurs) et lorsque l'ensemble des données est

passé une fois dans le réseau ,on applique la retro propagation en utilisant l'erreur totale .Cette façon de faire est préférée pour des raisons de rapidité et de convergence .

V.23. Les applications des réseaux de neurones

Les réseaux de neurones servent aujourd'hui à toutes sortes d'applications dans divers domaines (informatique, électronique, science cognitive, neurobiologie et même philosophie). L'étude des réseaux de neurones est une voie prometteuse de l'intelligence Artificielle, qui a des applications dans de nombreux domaines [100] :

- Industrie : Contrôle qualité, diagnostic de panne, corrélations entre les données fournies par différents capteurs, analyse de signature ou l'écriture manuscrite…
- Finance : prévision et modélisation du marché (cours de monnaie …), sélection d'investissements, attribution de Crédits….
- Télécommunications et informatique : analyse du signal, élimination du bruit, reconnaissance de formes (bruits, images, paroles), compression de données…
- Environnement : évaluation des risques

V.24 L'algorithme de rétropropagation du gradient

L'apprentissage supervisé pour les réseaux de neurones, consiste à adapter les poids synaptiques de telle manière que l'erreur entre la sortie du réseau et la sortie désirée soit aussi petite que possible. La plupart des algorithmes d'apprentissage des réseaux de neurones est basée sur les méthodes du gradient: ils cherchent à minimiser un critère de la forme suivante:

$$J = \frac{1}{2}\sum_{q=1}^{m}(y_{rq}(X_p) - y_q^d(X_p))^2 \qquad (V.17)$$

Où y_{rq} et y_q^d sont la sortie du réseau et la sortie désirée pour le vecteur d'entrée *Xp*. Cette optimisation se fait d'une manière itérative, en modifiant les poids en fonction du gradient de la fonction de coût selon la loi d'adaptation suivante:

$$w_{ij}^l(k+1) = w_{ij}^l - \mu\frac{\partial J}{\partial w_{ij}^l} \qquad (V.18)$$

Où

w_{ij}^l est le poids de la connexion entre le $i^{éme}$ neurone de la couche *l* et le $j^{éme}$ neurone de la couche *l*-1

j= 1,…., n+ *1* la j^{éme} composante du vecteur d'entrée,

i= 1,…. ,*m+1* la i^{éme} composante du vecteur de sortie,

l=1,…. ,*L* l'ordre d'une couche dans le réseau de neurone,

μ est une constante positive appelée taux d'apprentissage.

Le gradient est calculé par une méthode spécifique aux réseaux de neurones, dites méthode de rétro propagation. Dans cette méthode il est possible de calculer le gradient de la fonction coût en retropropageant l'erreur commise en sortie vers les couches cachées.[96]

V.25. *Les Réseaux Hybrides Neuro-flous*

V.25.1. Introduction :

Les réseaux de neurones et la logique floue présentent séparément des avantages et des inconvénients. Le point de départ des travaux sur la fusion de ces deux approches, le neuro flou fut la volonté de réduire leurs inconvénients en alliant les caractéristiques sémantiques du flou et les capacités d'apprentissage des systèmes neuronaux. Les deux approches contribuent à l'apprentissage et à la généralisation pour surmonter les inconvénients de l'un et de l'autre, des travaux de recherches ont opté pour la combinaison des deux systèmes, pour donner des systèmes neuro-flous, ce qui fait accroître la performance de ces systèmes.

V.25.2. Objectif

- Les réseaux neuro-flous sont nés de l'association des réseaux de neurones avec la logique floue, de manière à tirer profit des avantages de chacune de ces deux techniques.

- La principale propriété des réseaux neuro-flous est leur capacité à traiter dans un même outil des connaissances numériques et symboliques d'un système.

- Ils permettent donc d'exploiter les capacités d'apprentissage des réseaux de neurones d'une part et les capacités de raisonnement de la logique floue d'autre part.

V.25.3. Définition des Réseaux neuro-flous

Définition 1

Le système Neuro-flou est un système flou formé par un algorithme d'apprentissage inspiré de la théorie des réseaux de neurones. La technique d'apprentissage opère en fonction de l'information locale et produit uniquement des changements locaux dans le système flou d'origine [96].

Définition 2

Un système Neuro-Flou est un réseau de neurones qui est typologiquement équivalent à la structure d'un système flou. Les entrées/sorties du réseau ainsi que les poids sont des nombres réels, mais les noeuds implémentent des opérations spécifiques aux systèmes flous : fuzzification, opérateurs flous (conjonction, disjonction), défuzzification [101].

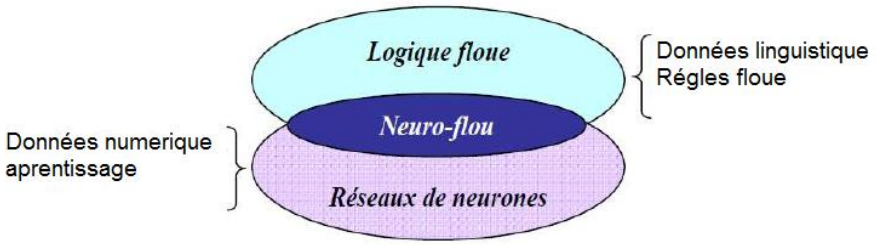


Figure V.49 : Principe du Neuro-flou

La figure. V.49 résume le principe du système neuro-flou qui représente l'intersection entre la logique floue et les réseaux de neurones. Afin de résumer 1'apport du neuro-flou, le tableau I.2 regroupe les avantages et les inconvénients de la logique floue et des réseaux de neurones [102].

V.25.4. Avantages et inconvénients de la logique floue et des réseaux de neurones :

L'utilisation simultanée des réseaux de neurones et de la logique floue, permet de tirer les avantages des deux méthodes : les capacités d'apprentissage de la première et la lisibilité et la souplesse de la seconde.

Afin de résumer l'apport du neuro-flou, le Tableau (V. 1) regroupe les avantages et les inconvénients de la logique floue et des réseaux de neurones, [103].

Les systèmes neuro-flous sont créés afin de synthétiser les avantages et de surmonter les inconvénients des réseaux neuronaux et des systèmes flous Les algorithmes d'apprentissage peuvent être employés pour déterminer les paramètres des systèmes flous. Ceci revient à créer ou améliorer un système flou de manière automatique, au moyen des méthodes spécifiques aux réseaux neuronaux. Un aspect important est que le système reste toujours interprétable en terme de règles floues, vu qu'il est basé sur un système flou.

Réseaux de Neurones	**Logique Floue**
Avantages	
• Aucune connaissance antérieure sur les règles. • Le modèle mathématique non requis. • Plusieurs algorithmes d'apprentissage sont disponibles.	• La connaissance antérieure sur les règles peut être utilisée. • Le modèle mathématique non requis. • Une interprétation et implémentation simple.
Inconvénients	
• Boite noire (manque de traçabilité) • L'adaptation aux environnements différents est difficile et le réapprentissage est souvent Obligatoire (sauf pour le RBF). • La connaissance antérieure ne peut pas être employée (apprentissage à partir de zéro) (sauf pour le RBF). • Aucune garantie sur la convergence de l'apprentissage.	• Ne peut pas apprendre. • Les règles doivent être disponibles. • Adaptation difficile au changement de l'environnement. • Aucunes méthodes formelle pour l'ajustement ;

Tableau. V. 1 : Comparaison entre la logique floue et les réseaux de neurones.

Les règles floues codées dans un système neuro-flou représentent les échantillons imprécis et peuvent être vues en tant que prototypes imprécis des données d'apprentissage. Par contre un système neuro-flou ne devrait pas être vu comme un système expert (flou), et il n'a rien à voir avec la logique floue dans le sens strict du terme. On peut aussi noter que les systèmes neuro-flous peuvent être utilisés comme des approximateurs universels [102].

V.26. Principe de fonctionnement

Les réseaux de neuro-flous hybrides apprennent des rapports et des modèles en utilisant un algorithme d'apprentissage supervisé qui examine les données dans un ensemble de la formation qui consiste en exemples d'entrées et leurs sorties associées. Pendant la phase d'apprentissage, un réseau neuro-flou hybride modifie sa structure interne pour refléter le rapport entre les entrées et les sorties dans l'ensemble de base de connaissance. L'exactitude d'un réseau neuro-flou est vérifiée après que le cycle d'apprentissage soit terminé en utilisant un ensemble séparé d'entrées et de sorties appelé l'ensemble de la validation (figure. V.50).

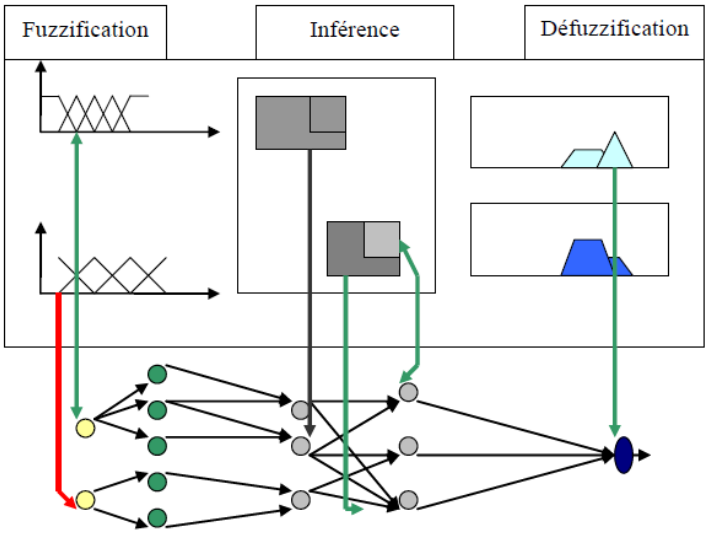


Figure. V.50 : Principe de fonctionnement du système neuro-flou.

V.27. Architecture des neuro-flous

Il existe quatre grandes catégories de combinaisons des réseaux de neurones avec la logique floue : réseau flou neuronal, système neuronal/flou simultanément, modèles neuro-flous coopératifs et modèles neuro-flous hybrides [96].

Réseau flou neuronal

Des techniques floues sont utilisées pour augmenter les possibilités d'apprentissage ou l'exécution d'un réseau neuronal. Ce genre d'approche ne doit pas être confondu avec les approches mixtes.

Système neuronal/flou simultanément

Le réseau neuronal et le système flou fonctionnent ensemble sur la même tâche, mais sans s'influencer. Habituellement le réseau neuronal traite les entrées, ou post-traite les sorties du système flou (voir figure. V.51).

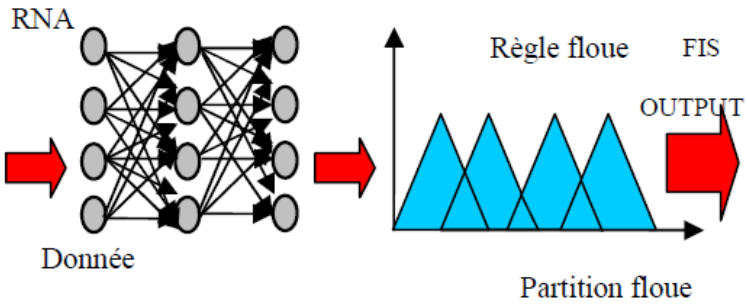


Figure. V.51 : Exemple d'association en série d'un réseau de neurone et d'un système.

Modèles neuro-flous coopératifs

Le réseau neuronal est employé pour déterminer les paramètres (les règles et les ensembles flous) d'un système flou. Après la phase d'apprentissage, le système flou fonctionne sans le réseau neuronal. C'est une forme simple des systèmes neuro-flous.

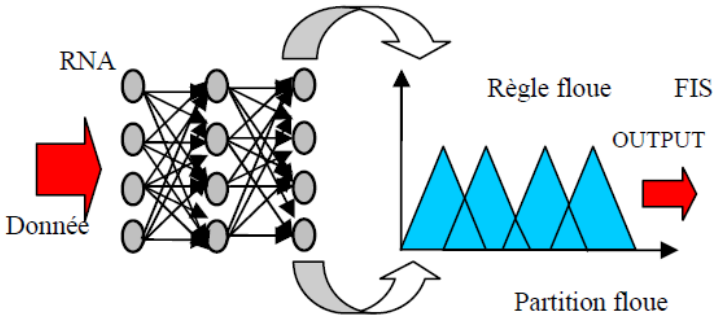


Figure. V.52 : Exemple d'association en parallèle d'un réseau de neurone et d'un système flou.

V.28. Méthodes neuro-floues :

Plusieurs méthodes ont été développées depuis 1988 et sont le plus souvent orientées vers la commande de systèmes complexes et les problèmes de classification. Il existe ainsi trois méthodes neuro-floues.

V.28.1. Première méthode neuro-floue :

Cette méthode neuro-floue est basée sur le codage du système d'inférence floue sous la forme d'un réseau de neurones multicouches dans lequel les poids correspondent aux paramètres du système. L'architecture du réseau dépend du type de règles et des méthodes d'inférence, d'agrégation et de défuzzification choisies, figure (V.53).

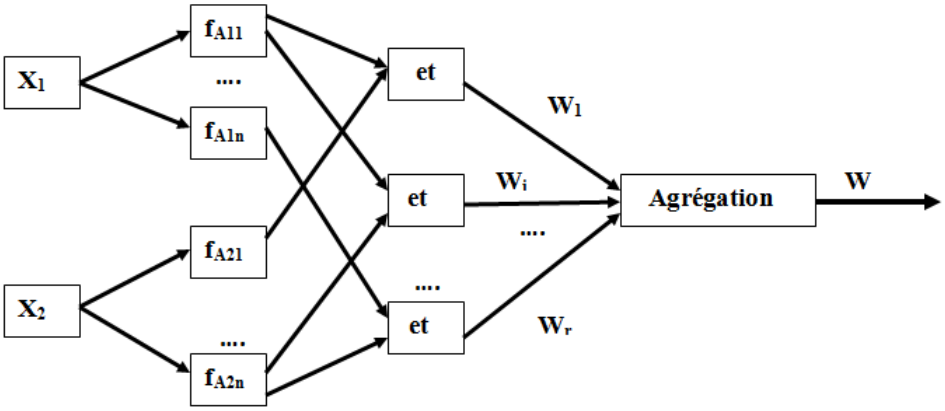


Figure V.53 : Premières architecture des réseaux neuro-floues

V.28.2. Deuxième méthode neuro-floue :

Elle consiste à utiliser les réseaux de neurones pour remplacer chacune des composantes d'un système de commande floue. Ces réseaux sont destinés à l'apprentissage des fonctions d'appartenance, au calcul de l'inférence et à la réalisation de la phase d'agrégation et de défuzzification. Ils peuvent réaliser l'extraction des règles floues en analysant la corrélation qui existe entre les entrées et les sorties du réseau de neurones.

Ces approches permettent de résoudre deux problèmes importants de la logique floue : la détermination des fonctions d'appartenance et l'adaptation à l'environnement du système.

V.28.3. Troisième méthode neuro-floue :

Cette troisième méthode utilise des réseaux de neurones et des systèmes flous associés en série ou en parallèle. Plusieurs variantes d'utilisation sont ainsi possibles :

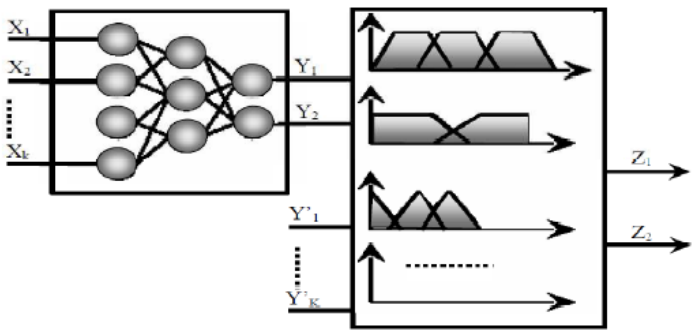


Figure V.54 : Troisième architecture des réseaux Neuro-Flou Réalisation en série

Le réseau de neurones fonctionne en amont du système flou. Les variantes d'entrées du système flou sont déterminées à partir des sorties du réseau de neurones (dans le cas où elles ne sont pas mesurables directement) ou encore un réseau de neurones effectue une tâche de classification ou de reconnaissance de formes, suivie d'un système flou d'aide à la décision.

Un réseau de neurones qui fonctionne en aval du système flou, dans le but d'ajuster les sorties d'un système de commande floue à de nouvelles connaissances obtenues, les variables de sorties étant les erreurs sur les variables de sortie du système flou,….

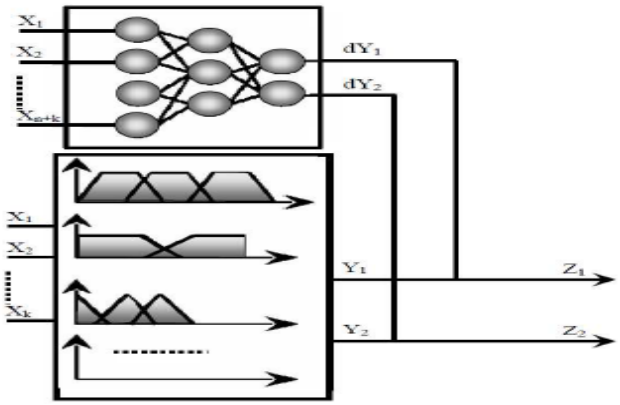


Figure V.55 : Troisième architecture des réseaux Neuro-Flou Réalisation en parallèle

V.29. Le Modèle ANFIS

L'ANFIS c'est un système d'inférence adaptatif neuro-flou qui consiste à utiliser un réseau neurone de type MLP à 5 couches pour lequel chaque couche correspond à la réalisation d'une étape d'un système d'inférence floue de type **Takagi-Sugeno**. Pour la simplicité, nous supposons que le système d'inférence flou à deux entrées x et y, et une sortie f. Supposons que la base de règle contient deux règles floues de type **Takagi-Sugeno**.

Règle 1 : *SI x est* A_1 *et y est* B_1 *ALORS* $f1 = p_1x + q_1y + C_1$ (V.19)

Règle 2 : *SI x est* A_2 *et y est* B_2 *ALORS* $f2 = p_2x + q_2y + C_2$ (V.20)

L'ANFIS a une architecture formée de cinq couches comme représenté sur la figure V.56

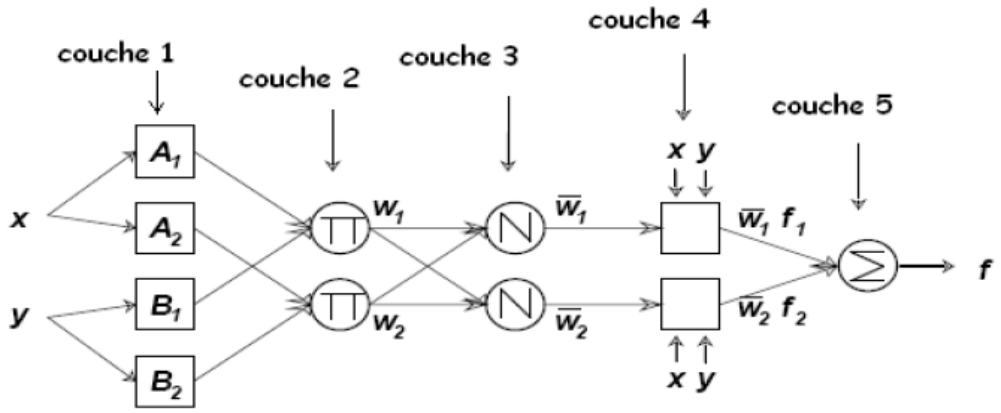


Figure V.56: L'Architecture générale de l'ANFIS.

Une architecture classique peut être décrite de la manière suivante ;

La première couche : comporte autant de neurones qu'il y'a de sous-ensembles flous dans le système d'inférence représenté. Chaque neurone d'un sous ensemble flou particulier calcule le degré de vérité par sa fonction de transfert. La seule restriction sur le choix de cette fonction concerne sa dérivabilité. En retrouve dans la littérature, l'utilisation, des fonctions gaussiennes ou les paramètres modifiables sont le centre et la pente de la gaussienne (variance).

La fonction d'activation des neurones i de la première couche :

$$O_i^1 = \mu_{Ai}(x) \qquad \text{(V.21)}$$

Avec x est l'entrée sur le neurone i, et A_i est un sous ensemble floue correspondant à la variable x. En d'autres façon, O_i^1 est la fonction d'appartenance du A_i qu'indique le degré auquel un x donné satisfait. Nous choisissons $\mu_{Ai}(x)$ pour être sous forme de (Gaussien, triangle, trapézoïdal) avec le maximum égal à 1 et le minimum égal à 0, tels que les fonctions généralisées de ces formes soient :

Triangle:

$$\mu(x) = \max(\min\left(\frac{x-a}{b-a}, \quad \frac{c-x}{c-b}, \quad 0\right)) \qquad \text{(V.22)}$$

Trapézoïdale :

$$\mu(x) = \max(\min\left(\frac{x-a}{b-a}, 1, \quad \frac{c-x}{c-b}, \quad 0\right)) \qquad \text{(V.23)}$$

Gaussienne :

$$\mu(x) = \exp(-\frac{(x-m)^2}{2\sigma}) \quad \text{(V.24)}$$

Où (a,b,c,m,σ) est l'ensemble des paramètres. Pendant que les valeurs de ces paramètres changent, les fonctions précédentes changent en conséquence, er de ce fait présentent des fonctions d'appartenance de diverses formes sur la variable linguistique A_i . Les paramètres dans cette couche sont désignés par le nom des paramètres de fonction d'appartenance.

La deuxième couche: qu'on dit première couche cachée, sert à calculer le degré d'activation des prémisses. Chacun des neurones de cette couche représente la prémisse d'une règle.

Ils reçoivent le degré de vérité des différents sous-ensembles flous composant cette prémisse et ont en charge le calcul de leur propre degré de vérité. Les fonctions d'activation utilisées pour ces neurones dépendent des opérateurs présents dans les règles **OU**.

La fonction d'activation des neurones i de la première couche :

$$O_k^2 = w_k = \mu_{Ai}(x).\mu_{Bi}(y) \quad \text{(V.25)}$$

Ou k : représente le nombre de règles,*i* : représente le nombre de partition de *x* et *j* le nombre de partition de *y*

La troisième couche: deuxième couche cachée, normalise les degrés d'activations des règles.

Chaque neurone dans cette couche est alors, un neurone de normalisation. L'i^{ème} neurone calcule le rapport entre l''i^{èm} poids de règles et la somme de tous les poids des règles.

$$O_k^3 = \overline{w_k} = \frac{w_k}{\sum w_i} \quad \text{(V.26)}$$

La quatrième couche: troisième couche cachée, sert à déterminer la partie conséquence des règles par l'intermédiaire des paramètres **(p,q,C)** La fonction de chaque neurone dans cette couche peut se formuler comme suit :

$$O_k^3 = \overline{w_k}.f_k = \overline{w_k}.(p_k x + q_k y + C_k) \quad \text{(V.27)}$$

Où $\overline{w_k}$ est la sortie de couche précédente, et $(\mathbf{p}_i,\mathbf{q}_i,\mathbf{C}_i)$ est l'ensemble des paramètres.

Ces paramètres sont désignés sous le nom de "paramètres conséquents".

La couche de sortie : cette couche contient un seul neurone, dont leur fonction est de calculer la sortie globale (résultante), comme addition de tous les signaux entrants, c'est-à-dire :

$$O_k^5 = \sum_{k=1}^{n} \overline{w_k} \cdot f_k \qquad (V.28)$$

La figure (V.57) représente un système ANFIS, à 2 entrées chaque entrée repartie en trois (3) sous ensembles flous et neuf (9) règles.

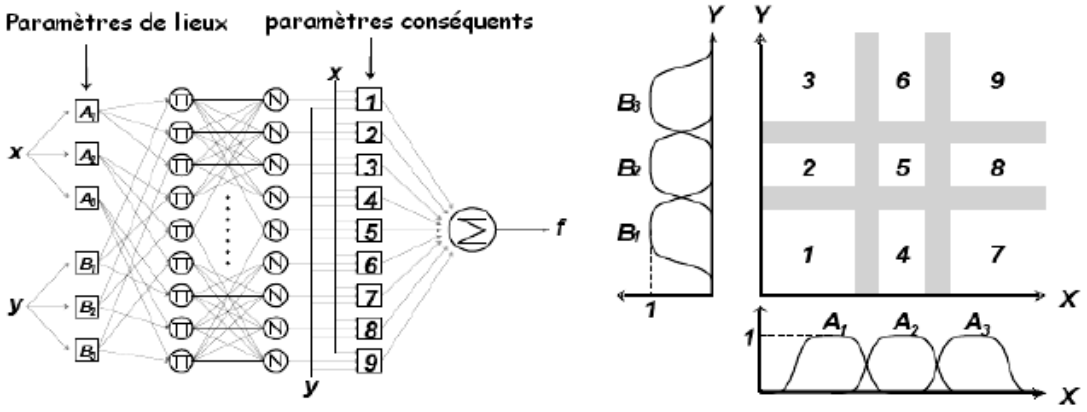


Figure V.57 : Architecture d'un ANFIS à 2 entées avec 9 règles.

Couches	Types des couches	Le nombre de neurone dans la couche
Couche 0	Couche d'entrée	**n**
Couche 1	Fuzzification	**(p,n)**
Couche 2	Inférence	p^n
Couche 3	La normalisation	p^n
Couche 4	Linéarisation des fonctions	p^n
Couche 5	Sortie	1

Tableau. V.2 : les différentes couches d'un système ANFIS

Tel que :

p : Le nombre des sous-ensembles flous d'entrée (partition flou).

n : Le nombre d'entrées.

Noter que les neurones dans ANFIS rencontrent généralement:

- fonction d'appartenance définie par différentes formes.
- Règles (habituellement produit).
- Normalisation (division de somme et d'arithmétique).

- Fonctions (régressions linéaires et multiplication avec $\overline{w}$, tel que $\overline{w}$ est la normalisation du poids w).
- la sortie (Somme Algébrique).

V.29. Conclusion

Dans ce chapitre, nous avons rappelé les éléments de base relatifs aux réseaux de neurones, la théorie de la logique floue et les systèmes neuro-flous. En effet, des définitions de base sur les réseaux de neurones, les différents architectures et types d'apprentissage, ainsi que la rétro propagation de gradient ont été présentés. Ensuite, nous avons élaboré des notions de base sur les variables linguistiques, les sous-ensembles flous, le système d'inférence flou et les types de contrôleurs et à la fin, nous citons des applications de la logique floue. La dernière section de ce chapitre a été consacrée aux définitions de base des ensembles neuro-flous et les différentes combinaisons de la logique floue et réseau de neurone, ensuite, nous avons présenté les différentes structures hybrides neuro-floues.

Chapitre VI

VI.RESULTATS

Dans ce qui suit, nous présenterons les résultats obtenus par la simulation avec les trois méthodes : la méthode utilisant la géométrique analytique, celle utilisant la méthode polynomiale, enfin celle utilisant la logique floue qui est l'objet principal de ce travail. Pour les deux premières méthodes, quatre exemples ont été étudiés : le premier correspondant à deux solutions non dégénérées, le second présentant quatre solutions dont trois non dégénérées, le troisième, six solutions dont quatre non dégénérées, enfin le dernier exemple illustrant le cas de la dégénérescence touchant l'ensemble des solutions (six). Pour la dernière méthode, en raison de la complexité de sa mise en œuvre, nous somme limités au premier exemple uniquement.

VI.1. Méthode Géométrique

VI.1.1. Cas de non dégénérescence

Pour la combinaison suivante :

$$x_2 = 4, \quad x_3 = 2, \quad y_3 = 4, \quad l_2 = 1, \quad l_3 = 1, \quad \beta = 60$$

$$q_1 = 1.4142, \quad q_2 = 3.1196, \quad q_3 = 2.6084$$

L'appel de la fonction Matlab *geo_direct_geo*, qui fait appel elle-même à la fonction *geo_direct_optim* qui permet de trouver la solution optimale, présentant l'erreur quadratique minimale, et qui affiche les graphes des poses du robot correspondants ; donne les résultats suivants :

P_1	#1	#2
$\varphi(\deg rés)$	–53.9651	60.0
x	0.3434	1
y	1.3719	1

Tableau VI.1: Solutions pour un manipulateur non dégénéré

Auxquelles correspondent les deux positions P_i (i=1 :3) des sommets de la plateforme du robot

P_i	#1		#2	
P_1	0.3434	1.3719	1	1
P_2	0.9316	0.5632	1.500	1.866
P_3	1.3378	1.4770	0.500	1.866

Tableau VI.2 : Positions Pi pour un manipulateur non dégénéré

P_i	#1	#2
q_1	1.414213562373095	1.414213562373095
q_2	1.414213562373095	3.119623504137780
q_3	2.608419547897759	2.608419794289794
Erreur (10^{-6})	0.95087	1.19726

Tableau VI.3 : Positions Pi, le vecteur articulaire et l'erreur résultante globale pour un manipulateur non dégénéré

De la géométrie directe c'est une optimisation qui permet de trouver la solution optimale, en présentant le minimum erreur quadratique, et les poses du robot correspondant. Un programme Matlab a été développé pour calculer le minimum erreur quadratique et le robot pose (Tableaux VI.1, VI.2, VI.3). A quoi correspondent les deux positions Pi (i 1: 3) des sommets de la plate-forme du robot. Les deux poses de la plate-forme qui correspond à ces deux solutions dont les données ci-dessous, les segments de flèche pointer vers les numéros des solutions du tableau précédent(Fig.VI.1).

Les deux poses de la plateforme qui correspondent à ces deux solutions dont données ci-dessous, les segments fléchés pointent sur les numéros des solutions du tableau précédent.

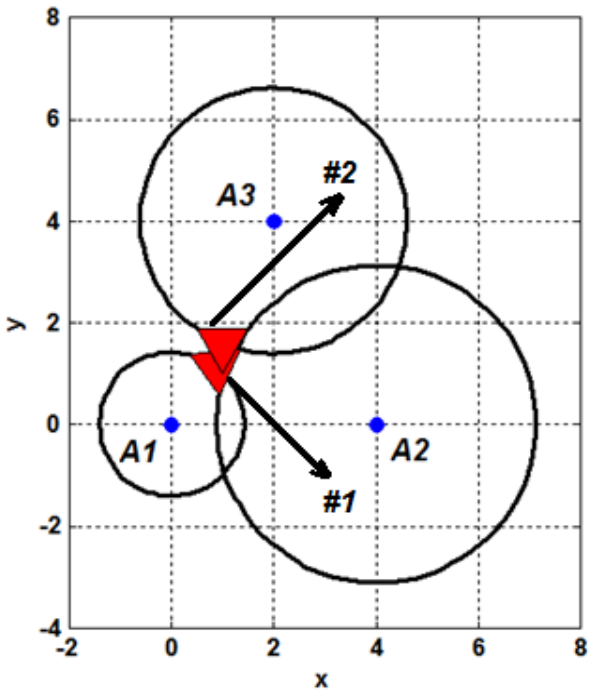


Figure VI.1 : Le mode d'assemblage correspondant au tableau 1 pour la solution géométrique non dégénérée

VI.1.2. Cas dégénérescence premier ordre :

Pour la combinaison suivante :

$$x_2 = 2, \quad x_3 = 0, \quad y_3 = 1, \quad l_2 = 2, \quad l_3 = 1.5, \quad \beta = 60, \quad q_1 = q_2 = q_3 = 1$$

Quatre solutions sont trouvées :

P_1	#1	#2	#3	#4 °
$\varphi(\text{deg } rés)$	–38.6320	0	0	29.5331
x	–0.4885	-0.7138	-0.0036	-0.9618
y	0.8723	0.7003	-0.9993	-0.2736

Tableau VI.4 : Les quatre solutions pour un manipulateur dégénéré de premier ordre

Avec les positions P_i correspondantes :

P_i	#1		#2		#3		#4 °	
P_1	-0.4885	0.8723	-0.7138	0.7003	-0.0036	-0.9993	-0.9618	-0.2736
P_2	1.0734	-0.3722	1.2861	0.7003	1.9638	-0.9993	2.7019	0.7122
P_3	0.9080	1.4188	0.0361	1.9993	0.7138	0.2996	0.9740	1.2263

Tableau VI.5 : Positions Pi pour un manipulateur dégénéré de premier ordre

P_i	#1	#2	#3	#4 °
q_1	1.000000000000000	0.9999999999999999	1.0000000000000	1.0000000000000
q_2	0.9999999999999992	0.9999999999999971	0..9999999999999955	0.9999999999999768
q_3	1.000004023116677	1.000001598632977	9.999948946771762	0.9999959643548794
Erreur (10^{-6})	4.0231	1.5986	5.1053	4.03568

Tableau VI.6 : Le vecteur articulaire et l'erreur globale résultante pour un manipulateur dégénéré du premier ordre

Les graphes des quatre poses sont représentés ci-dessous :

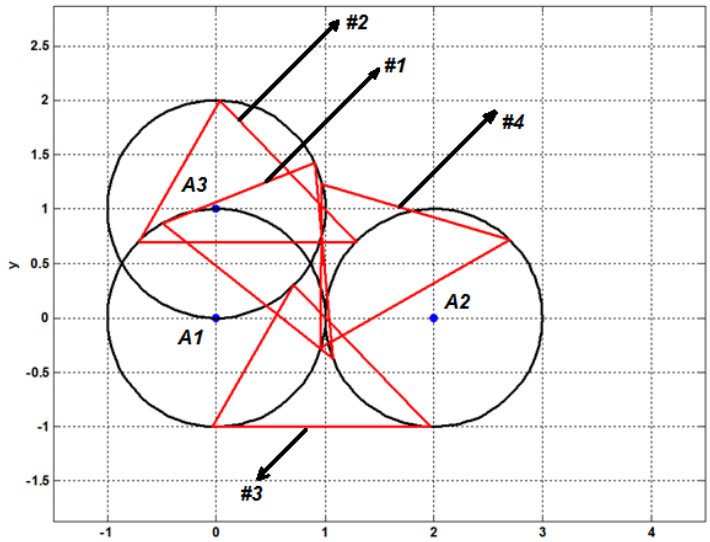


Figure VI.2 : Les quatre modes d'assemblage correspondant au tableau 3 pour deux solutions non dégénérées, l'un dégénéré

VI.1.3.Cas de dégénérescence du premier ordre: quatre racines non dégénérées, une dégénérée

Pour la combinaison suivante :

$x_2 = 2, \quad x_3 = 0.5, \quad y_3 = 1, \quad l_2 = 2, \quad l_3 = 1.5, \quad \beta = 60, \quad q_1 = q_2 = q_3 = 0.7$

On trouve les quatre solutions, dont une $(\varphi = 0)$ dégénérée.

P_i	#1	#2	#3	#4 °	#5 °	#6
φ(deg *rés*)	–43.8047	–6.6275	0	0	23.6387	58.4875
x	–0.3395	-0.9849	-09487	-0.1393	0.6631	0.9768
y	0.9405	0.1728	0.3126	-0.9902	-0.7484	-0.5141

Tableau VI.7 : Les six solutions pour un manipulateur dans le cas de quatre racines non dégénérées, l'une dégénérée

Les poses P_i correspondantes sont :

P_i	#1		#2		#3		#4 °		#5 °		#6	
P_1	-0.3395	0.9406	-0.9850	0.1728	-0.9499	-0.3126	-0.1394	-0.9902	0.9768	-0.2141	0.6632	-0.7485
P_2	1.1039	-0.4838	1.0017	-0.0580	1.0501	-0.3126	1.8606	-0.9902	2.8090	0.5878	-1.7085	0.9566
P_3	1.1009	1.3590	-0.0900	1.3766	-0.1999	0.9864	0.6106	0.3088	1.1430	1.2766	-0.0523	0.5699

Tableau VI.8 : Positions Pi pour un manipulateur dans le cas de quatre racines non dégénérées, une dégénérée

P_i	#1	#2	#3	#4 °	#5 °	#6
q_1	1.0000000000000	1.0000000000000	1.0000000000000	1.0000000000000	1.0000000000000	1.0000000000000
q_2	0.999999999999998	0.999999999999997	1.000000000000034	1.0000000000001	0.999999999999974	0.699991607557378
q_3	0.699993182215874	0.699992614668121	0.700005720554232	0.699993461795613	0.999999999999997	0.699997896382708
Erreur (10^{-5})	0.6817	0.7385	0.5720	0.6538	0.8392	0.2103

Tableau VI.9 : vecteur articulaire et erreur résultante globale pour un manipulateur dans le cas de quatre racines non dégénérées, une dégénérée

Les quatre poses sont représentées sur la figure suivante :

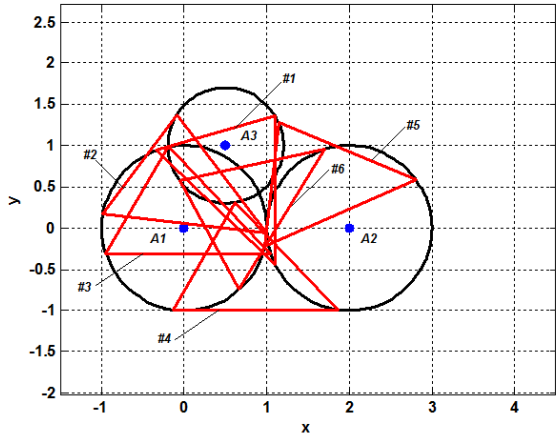


Figure VI.3 : Les six modes d'assemblage correspondant au tableau 5 pour quatre solutions non dégénérées, l'un dégénéré

VI.1.4. Cas de la dégénérescence d'ordre trois.

Pour la combinaison suivante :

$$x_2 = 2, \quad x_3 = 1, \quad y_3 = 1, \quad l_2 = 1, \quad l_3 = 1, \quad \beta = -90, \quad q_1 = 0.8, \; q_2 = q_3 = 1.5$$

On trouve :

P_i	#1	#2	#3	#4 °	#5 °	#6
φ(deg *rés*)	–155.6267	–90	–90	–48.6331	–48.6331	53.6099
x	0.4335	-0.4597	0.6547	0.2826	-0.7484	0.3963
y	0.6724	0.6547	-0.4597	-0.7484	0.2826	0.6950

P_i	#7	#8	#9	#10
φ(deg *rés*)	53.6099	126.3898	126.3898	155.6268
x	–0.7945	0.0933	0.6950	0.6724
y	0.0933	-0.7945	0.3963	0.4335

Tableau VI.10 : Les six solutions pour un manipulateur en cas de dégénérescence d'ordre 3

Les positions P_i correspondantes :

P_i	#1		#2		#3		#4 °		#5 °		#6	
P_1	0.4335	0.6724	-0.4597	0.6547	0.6547	-0.4597	0.2826	-0.7484	-0.7484	0.2826	0.3963	0.6950
P_2	-0.4774	0.2597	-0.4597	-0.3453	0.6547	-1.4597	0.9434	-1.4989	-0.0876	1.0331	0.9895	1.500
P_3	0.8462	-0.2385	-1.4597	0.6547	-0.3453	-0.4597	1.0331	-0.0876	-1.4989	0.9434	1.2013	0.1017

P_i	#7		#8		#9		#10 °	
P_1	-0.7945	0.0933	0.0933	-0.7945	0.6950	0.3963	0.6724	0.4335
P_2	-0.2013	0.8983	-0.5000	0.0105	0.1017	1.2013	-0.2385	0.8462
P_3	0.0105	-0.500	0.8983	-0.2013	1.5000	0.9896	0.2597	-0.4774

Tableau VI.11 : Positions Pi, pour un manipulateur pour le cas de la dégénérescence de l'ordre trois

Nous avons pour cet exemple déterminé le vecteur articulaire et l'erreur résultante globale :

P_i	#1	#2	#3	#4 °	#5 °	#6
q_1	0.800000000000000	0.8000000000000002	0.80000000000000000e	8.000000000000000	8.0000000000000000e	8.000000000000002
q_2	1.500000000000000	1.500000000000000	1.500000000000000	1.500000000000000	1.5000000000000000e	1.500000000000000
q_3	1.500001310578265	1.499998227294674	1.499998618717735e	1.499997131024234	1.499994545565069e	1.499992739634409
Erreur (10^{-6})	1.31.5	1.7727	1.38183	2.8689	5.4544	7.2603

P_i	#7	#8	#9	#10 °
q_1	0.8000000000000000	0.8000000000000000	0.80000000000008	8.000000000000000
q_2	1.499999999999998	1.500000000000000	1.500000000000000	1.5000000000000000e
q_3	1.500001037586281	1.499998508919643	1.500000440110932	1.500001649812265
Erreur (10^{-5})	1.0375	1.4910	9.9541	1.6498

Tableau VI.12 : Vecteur articulaire et erreur résultante globale pour un manipulateur dans le cas de dégénérescence d'ordre 3

Les différentes poses sont décrites ci-dessous :

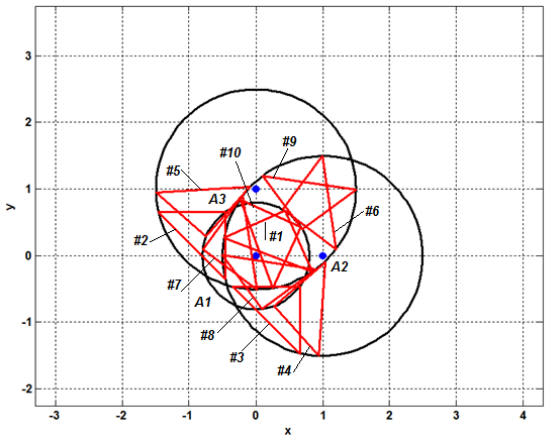


Figure VI.4. Les dix modes d'assemblage correspondant au tableau 7 pour la dégénérescence de l'ordre trois

VI .1.5.Conclusion

La méthode géométrique est une méthode systématique qui permet de trouver toutes les solutions générées par la méthode polynomiale, (CF ci-dessous), avec la même précision. Elle permet également de générer d'autres solutions qui remplissent les conditions de test de précision de la méthode (dix solutions au lieu de six dans le dernier exemple). La précision de la solution peut-être améliorée en réduisant le pas de balayage du cercle centré sur le point A1 et de rayon q1, comme décrit dans le chapitre III

VI.2. Méthode Polynomiale

Nous avons repris les quatre exemples précédemment étudiés par la méthode graphique.

VI .2.1.Cas de non dégénérescence : deux solutions

On a déjà établi qu'au maximum pour une combinaison donnée des variables articulaires, il peut avoir six solutions possibles. Les solutions sont toujours paires : deux ou quatre ou six. Dans le cas où la solution existe, elle est toujours appariée. Dans le cas de non existence, toutes les solutions sont complexes.

Pour la combinaison suivante :

$$x_2 = 4, \quad x_3 = 2, \quad y_3 = 4, \quad l_2 = 1, \quad l_3 = 1, \quad \beta = 60$$

$$q_1 = 1.4142, \quad q_2 = 3.1196, \quad q_3 = 2.6084$$

Cette combinaison a été obtenue par le calcul de la géométrie inverse. (programme geo_inv_par qui à partir de la donnée de la position des trois points fixes A, dont les coordonnées sont représentés par la matrice A, la configuration du triangle représentant la plateforme (matrice T [l_2 l_3 angle ouverture ß (en degrés)], la donnée des coordonnées du point M=P_1(x, y, ϕ) calcule le vecteur articulaire $Q = [q_1, q_2, q_3]$ et les autres points P restants (P_2 et P_3).

La syntaxe de la commande est :

[Q,P]= *geo_inv_par (A, T, M).*

L'appel de la fonction Matlab *geo_dir_poly* implantant la méthode polynomiale et affichant les poses de la plateforme avec l'indication de l'angle et des coordonnées du sommet P_1 , donne les valeurs des autres points Pi dans la variable structurée P et les angles correspondants dans le vecteur PHI. Les arguments d'entrée sont la matrice A déjà définie, le vecteur T et le vecteur Q.

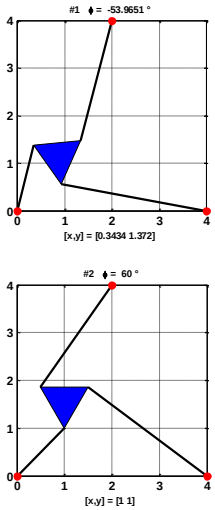


Figure VI.5 : Deux modes d'assemblage correspondant au tableau 13 correspondent à non dégénérescence avec deux solutions

Les deux solutions possibles figurent dans le tableau suivant :

P_1	#1	#2
$\varphi(\deg rés)$	–53.9651	60.0
x	0.3434	1
y	1.3719	1

Tableau VI.13 : Les deux ensembles de solutions pour un manipulateur correspondant ne dégénèrent pas en deux solutions

P_i	#1		#2	
P_1	0.3434	1.3719	1	1
P_2	0.9316	0.5632	1.500	1.866
P_3	1.3378	1.4770	0.500	1.866

Tableau VI.14 : Positions Pi pour un manipulateur correspondant à aucune dégénérescence avec deux solutions

L'erreur de positionnement pour chaque pose est calculée par la fonction Matlab calcul_erreur_globale, qui à partir des positions Pi(i=1 :3), de la matrice A, et du vecteur T, adresse la fonction Matlab *calcul_mod* qui renvoie les coordonnées articulaires réelles et le compare à la position articulaire voulue en utilisant un critère quadratique. Ces deux fonctions sont données en Annexe.

On trouve :

Erreur = 10^{-14} [0.0888 0.1110]

VI .2.2.Dégénérescence d'ordre 1 : quatre solutions

Pour la combinaison suivante :

$$x_2 = 2, \quad x_3 = 0, \quad y_3 = 1, \quad l_2 = 2, \quad l_3 = 1.5, \quad \beta = 60, \quad q_1 = q_2 = q_3 = 1$$

Nous sommes dans le cas où $x_2=l_2$ et $q_1=q_2$. On sera donc en présence d'une dégénérescence du premier ordre. L'angle phi égal à zéro, admet deux positions

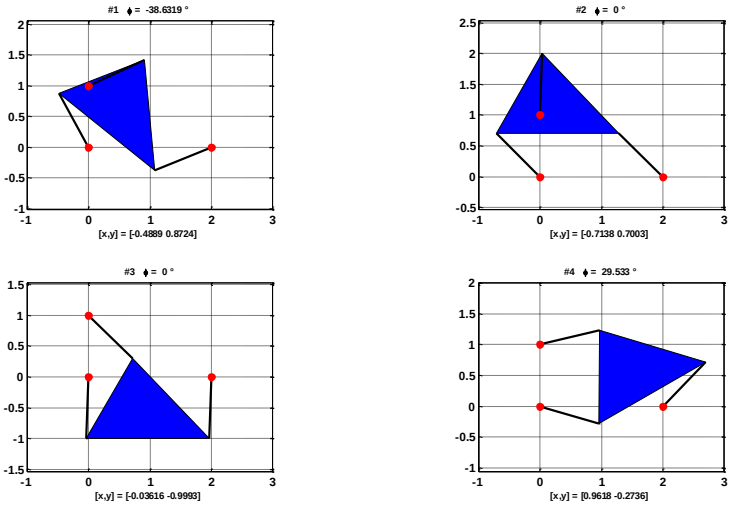


Figure VI.6 : Les quatre modes d'assemblage correspondant au tableau 15 correspondant à la dégénérescence de l'ordre 1 avec quatre solutions

P_1	#1	#2	#3	#4 °
φ(deg *rés*)	0	0	–38.6319	29.5330
x	0.7138	-0.0362	-0.4889	0.9618
y	0.7003	-0.9993	0.8724	-0.2736

Tableau VI.15 : Les quatre ensembles de solutions pour un manipulateur correspondent à la dégénérescence de l'ordre 1 a quatre solutions

P_i	#1		#2		#3		#4 °	
P_1	-0.7138	0.7003	-0.0362	-0.9993	-0.4889	0.8724	0.9618	-0.2736
P_2	1.2862	0.7003	1.9638	-0.9993	1.0735	-0.3763	2.7020	0.7122
P_3	0.0362	1.9993	0.7138	0.2997	0.9080	1.4189	0.9741	1.2263

Tableau VI.16 : Positions Pi pour un manipulateur les deux dernières correspondent à une dégénération de l'ordre 1 a quatre solutions

Erreur =10^{-13} [0.00890.0011 0.0022 0.1132] ;

VI .2.3. Dégénérescence d'ordre 1 : six solutions

Pour la combinaison suivante :

$$x_2 = 2, \quad x_3 = 0.5, \quad y_3 = 1, \quad l_2 = 2, \quad l_3 = 1.5, \quad \beta = 60, \quad q_1 = q_2 = 1, q_3 = 0.7$$

Comme précédemment, $x_2=l_2$ et $q_1=q_2$.. L'angle phi égal à zéro, admet deux positions, les quatre autres solutions seront non dégénérées.

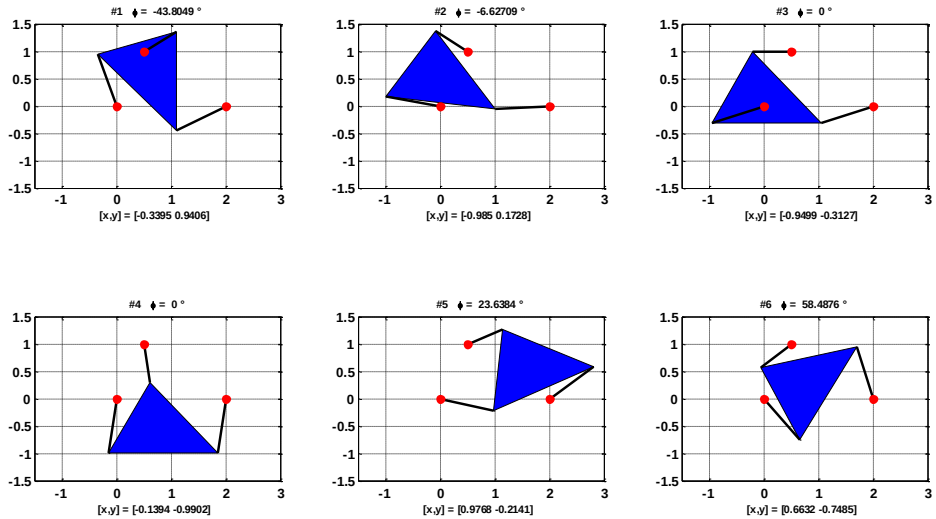


Figure VI.7 : Les six modes d'assemblage correspondant au tableau 14 avec les deux derniers correspondant à la racine dégénérée

P_i	#1	#2	#3	#4 °	#5 °	#6
φ(deg *rés*)	-43.8049	-6.6271	0	0	23.6384	58.4876
x	-0.3395	-0.9850	-0.9499	-0.1394	0.9768	0.6632
y	0.9406	0.1728	-0.3127	-0.9902-	-0.2141	-0.7485

Tableau VI.17 : Les six ensembles de solutions pour un manipulateur, les deux derniers correspondant à la racine dégénérée

P_i	#1		#2		#3		#4 °		#5 °		#6	
P_1	-0.3395	0.9406	-0.9850	0.1728	-0.9499	-0.3127	-0.1394	-0.9902	0.9768	-0.2141	0.6632	-0.7485
P_2	1.1039	0.4438	1.0017	0.0580	1.0501	-0.3127	1.8606	-0.9902	2.8090	0.5878	1.7085	0.9566
P_3	1.1010	1.3590	-0.0900	1.3768	-0.1999	0.9864	0.6106	0.3088	1.1430	1.2766	-0.0523	0.5699

Tableau VI.18 : Positions Pi, pour un manipulateur dont les deux derniers correspondent à la racine dégénérée

Erreur=10^{-13} [0.0067 0.5473 0.0033 0.0100 0.0100 0.0344]

VI .2.4 Dégénérescence sur tout l'espace

Pour la combinaison suivante :

$$x_2 = 2, \quad x_3 = 1, \quad y_3 = 1, \quad l_2 = 1, \quad l_3 = 1, \quad \beta = -90, \quad q_1 = 0.8, \; q_2 = q_3 = 1.5$$

Les conditions de dégénérescence pour les trois solutions sont vérifiées pour ces données.

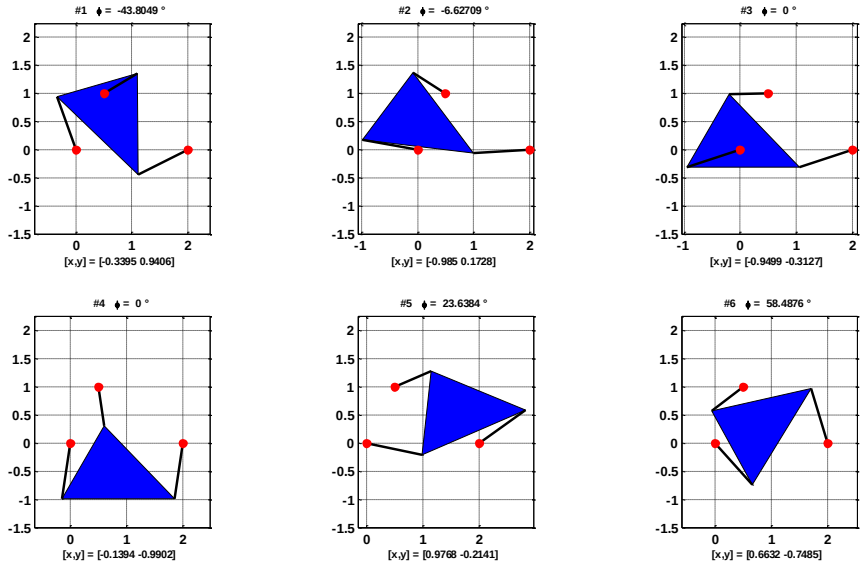


Figure VI.8 : Les six modes de montage (Dégénérescence sur tout l'espace) correspondant au tableau VI.19

P_i	#1	#2	#3	#4 °	#5 °	#6
φ(deg *rés*)	-90	-90	53.6102	53.6102	126.389	126.389
x	-0.4597	0.6547	-0.7945	0.3963	0.6950	0.0933
y	0.654745	-0.4597	0.0933	0.6950	0.3963	-0.7945

Tableau VI.19 : Les six ensembles de solutions pour un manipulateur dégénéré

P_i	#1		#2		#3		#4 °		#5 °		#6	
P_1	-0.4597	0.6547	0.6547	-0.4597	-0.7945	-0.0933	0.3963	0.6950	0.6950	0.3963	0.0933	-0.7945
P_2	-0.4597	-0.3453	0.6547	-1.4597	-0.2013	0.8983	0.9895	1.500	0.1017	1.2013	-0.500	0.0105
P_3	-1.4597	0.6547	-0.3453	-0.4597	0.0105	-0.5000	1.2013	0.1017	1.500	0.9895	0.8983	-0.2013

Tableau VI.20: Positions Pi pour un manipulateur dégénéré

Erreur = [2.2210e-15 1.7763e-15 4.4408e-16 2.2204e-16 1.1738e-08 1.1738e-08]

VI .2.5.Conclusion

La méthode polynomiale est systématique. Elle donne toujours des solutions paires dans le cas où le polynôme d'ordre six admet des solutions. Le cas problématique est celui où la solution est dégénérée qui correspond à une singularité, c'est-à-dire que pour les mêmes coordonnées articulaires, correspond deux solutions distinctes qui doivent être calculées selon la procédure explicitée dans le chapitre IV. Ce type de singularité est distinct de celui rencontré lors du calcul de la cinématique directe, où le calcul des vitesses opérationnelles peut s'avérer impossible pour certaines configurations du robot.

VI.3. Méthode utilisant la logique floue

VI .3.1.Principe

Nous avons illustré cette méthode sur le premier exemple analysé par les deux précédentes méthodes, et qui admet deux solutions. La fonction Matlab *geo_directe_fuzzy* qui implémente cette méthode est donnée en annexe . Elle appelle les remarques suivantes :

1- Elle fait appel à l'outil Matlab **anfis** qui permet de générer automatiquement les règles d'inférences utilisant les réseaux de neurones dee trois structures à trois entrées Q1, Q2 et Q3 et une sortie, X, Y et PHI, respectivement anfis1, anfis2 et anfis3.

2- Le nombres de voies pouvant être empruntée par la sortie est fonction du nombre de fonctions d'appartenance adoptées pour chaque entrée. Si on appelle n1, n2 et n3 le nombre de ces fonctions d'appartenance (MF : Membership Function), alors le nombre de voies de la structure neuronale sera égal à n1xn2xn3

3- Dans un premier temps, on prendra deux fonctions candidates par entrée de type fonction en cloche (gbellMF), qui génèrera 8 voies à la sortie, puis par la suite on prendra trois fonctions par entrée qui génèrera 27 voies. Les structures floues seront déommées fismat1, fismat2 et fismat3.

4- Les valeurs des intervalles de varaiation pour chaque valeur articulaire, celui des paramètres des fonctions d'appartenance, ainsi que les coefficients linéaires et non linéaires des sorties sont reproduits en annexe C

5- La matrice des données d'entrée pour chaque structure aura une taille de n^3 si n est le pas de quadrillage adopté pour chacune des trois dimensions (meshgrid). On constate que cette taille devient rapidement volumineuse et peut conduire à un dépassement de la mémoire signalé par Matlab. Les durées d'exécution pour les deux exemples étudiés sont en moyenne repectivement de deux (02) et neuf (09) minutes pour chaque structure

des deux exemples. Les durées de compilation deviennent rapidement rédhibitoires dès quand augmente la densité du quadriillage (mesh).

6- Pour pouvoir appliquer cette méthode, nous avons été conduits à réduire le domaine de recherche de la solution et à scinder la taille du quadrillage par deux en affectant la moitié des données à l'étape d'initiation (les indices impairs), et la moitié restante (indices pairs) à l'apprentissage qui va générer la structure floue finale. Les fichiers de définition sont donnés en annexe () pour les deux exemples.

VI .3.2 Premier cas : cas où deux fonctions d'appartenance par entrée sont utilisées

Le domaine de recherche d'analyse qui permet de générer le quadrillage à trois dimensions est comme suit :

$0.8 \le X \le 1.2$, $0.8 \le Y \le 1.2$, $45^0 \le PHI \le 90^0$

L'appel de la fonction Matlab *geo_directe_fuzzy* avec ces paramètres utilise la fonction anfis qui choisit par défaut deux fonctions d'appartenance en cloche pour l'initiation de la structure floue et la génération automatique des règles d'inférence.

Le schéma de la structure obtenu par les commandes Matlab *ruleedit* ou *ruleview* suivies de la sélection des menus « Edit » puis « Structure » est donné ci-dessous (seul l'opérateur ET est utilisé):

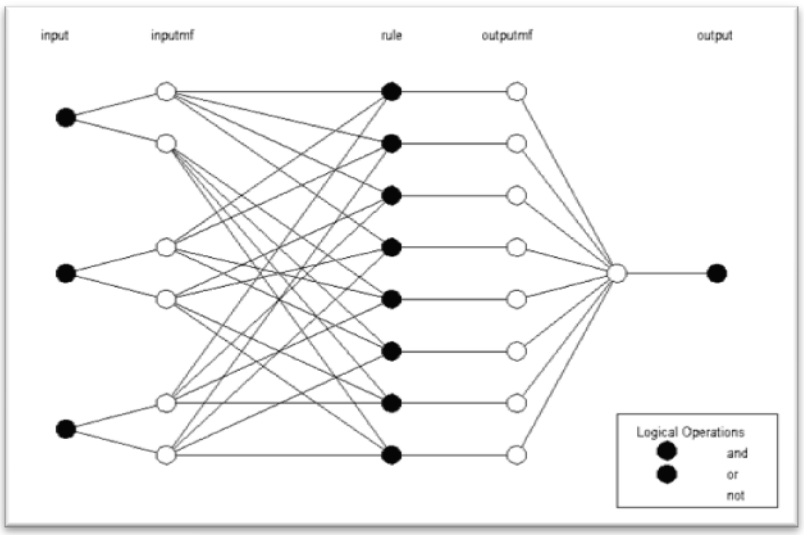


Figure VI.9 : Structure de la sortie

Les 2^3=8 règles d'inférence sont comme suit : (commande ruleEdit)

1- (Si Q1 est inmf1) ET (Q2 est inmf1) ET (Q3 est inmf1) ALORS (X est outmf1)

2- (Si Q1 est inmf1) ET (Q2 est inmf1) ET (Q3 est inmf2) ALORS (X est outmf2)
3- (Si Q1 est inmf1) ET (Q2 est inmf2) ET (Q3 est inmf1) ALORS (X est outmf3)
4- (Si Q1 est inmf1) ET (Q2 est inmf2) ET (Q3 est inmf2) ALORS (X est outmf4)
5- (Si Q1 est inmf2) ET (Q2 est inmf1) ET (Q3 est inmf1) ALORS (X est outmf5)
6- (Si Q1 est inmf2) ET (Q2 est inmf1) ET (Q3 est inmf2) ALORS (X est outmf6)
7- (Si Q1 est inmf2) ET (Q2 est inmf2) ET (Q3 est inmf1) ALORS (X est outmf7)
8- (Si Q1 est inmf2) ET (Q2 est inmf2) ET (Q3 est inmf2) ALORS (X est outmf8)

Les fonctions d'appartenance en cloche pour les trois entrées peuvent être visualisées par la commande Matlab *mfedit* ou bien calculées directement à partir des données de leurs paramètres a,b,c . On peut également adresser la fonction Matlab gbellmf

$$f(x) = \frac{1}{1 + \left| \frac{x-c}{a} \right|^{2b}} \quad \text{(VI.1)}$$

Les valeurs de ces coefficients pour les deux fonctions pour les trois entrées sont reproduites à partir des valeurs données en annexe (). Seul le tracé pour la sortie X est donné ci-dessous

	a	b	c
Q_1	*0.2926122351942617*	*1.999575554910495*	*1.137526254458919*
	0.2921104456450778	*1.999508511833954*	*1.691752052184750*
Q_2	*0.3309800878348759*	*1.998893941500382*	*2.879697690925664*
	0.3315245202873079	*1.998958418473497*	*3.359159191615123*
Q_3	*0.6411531208672442*	*1.999966155038961*	*2.117796859719111*
	0.6412853049205336	*1.999975382640182*	*3.399587220647379*

Tableau VI.21 : des coefficients a, b, c des Fonctions en cloche

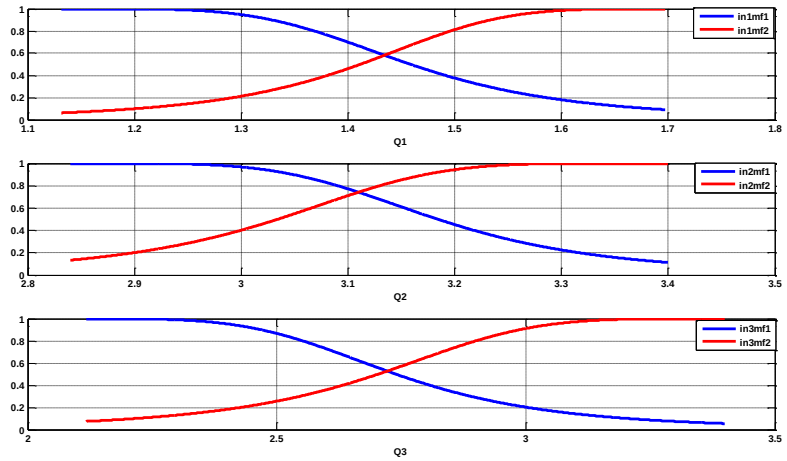


Figure VI.10: Tracé des Fonctions d'appartenance

Les courbes des trois sorties en fonction des trois entrées auront l'allure suivante :

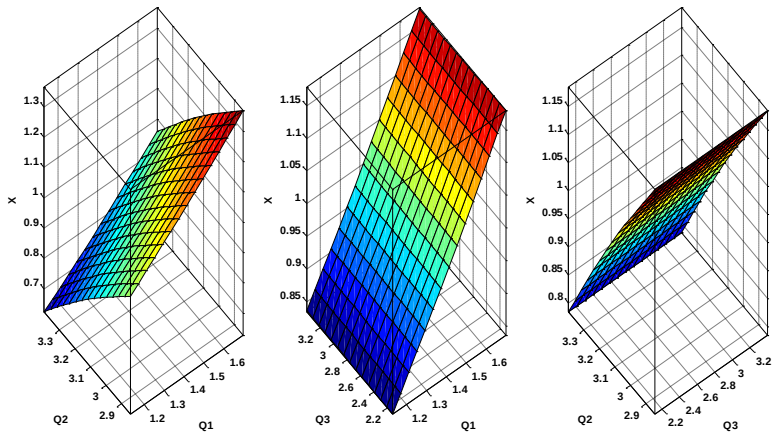


Figure VI.11: Courbes de la sortie X=f(Q1,Q2,Q3)

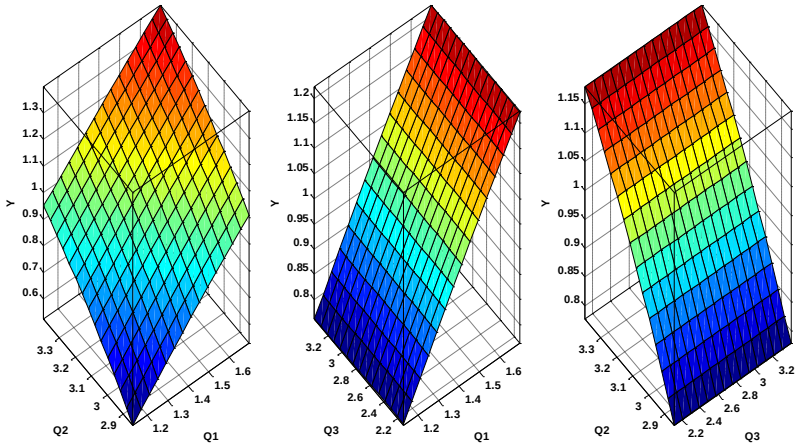


Figure VI.12 : Courbes de la sortie Y=g(Q1,Q2,Q3)

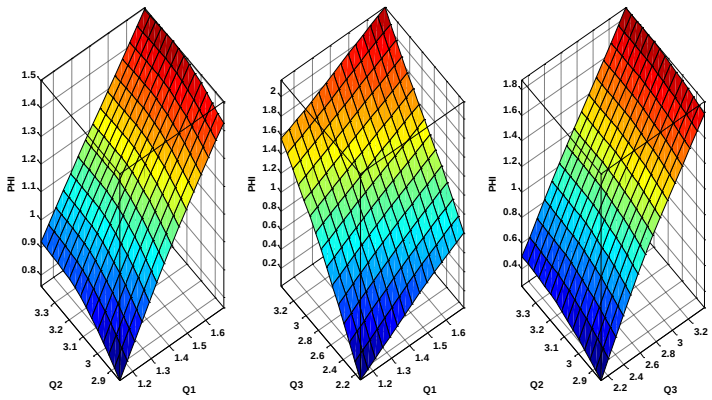


Figure VI.13 : Courbes de la sortie PHI=h (Q1, Q2, Q3)

Nous pourrons alors évaluer les sorties prédites pour les trois structures qui correspondent à la solution du problème de la géométrie directe, par la fonction Matlab *evalfis*

Vecteur articulaire d'entrée :

Q= [1.414213562373095 3.119623504137779 2.608418597022603]

Sortie X =evalfis (Q, anfis1) = 1.000027167658954
Sortie Y =evalfis (Q, anfis2) = 1.000670581045958
Sortie PHI=evalfis (Q, anfis3) = 1.049580733914649

Valeur théorique [1, 1, pi/4= 1.047197551196598]

Soit une erreur quadratique absolue égale à : 2.47 10^{-3}
Pour améliorer cette précision, on peut soit diminuer la valeur du pas du meshgrid, soit augmenter le nombre de fonction d'appartenance donc, le nombre de règles d'inférences.
Nous avons adopté cette approche pour le deuxième exemple

VI .3.3.Second cas : Utilisation de trois fonctions d'appartenance MF par entrée

Les trois fonctions induisent vingt sept 27 règles obtenues par combinaison factorielle de 3x3x3 valeurs. La structure neuronale obtenue (fismat1, fismat2 et fismat3) pour les trois sorties sera comme suit, seul le connecteur ET est employé.

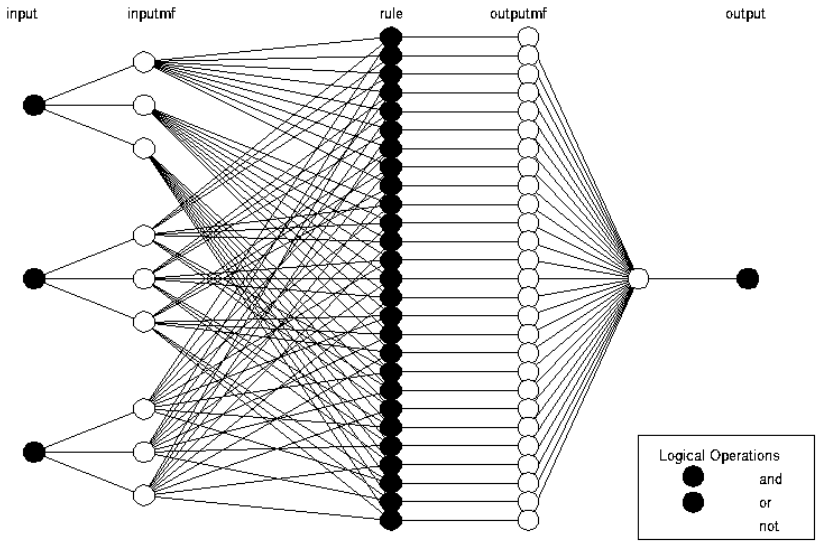


Figure VI.14 : Structure de la sortie

Les coefficients a, b, c pour les trois entrées sont représentés dans le tableau suivant :

	a	b	c
Q_1	*0.1869765621511652*	*1.998865172765118*	*1.162004288131832*
	0.1957437245602229	*2.000463685625106*	*1.412000420106174*
	0.1853454687358375	*1.999372490381056*	*1.658321693403733*
Q_2	*0.1964474545565682*	*1.997127645803953*	*1.154088654049545*
	0.1881717541684498	*2.000496920186938*	*1.412078678112379*
	1.842899913945532	*1.999333777721351*	*1.660393593238623*
Q_3	*0.2028818362109726*	*1.995342301460311*	*1.144793336155356*
	0.1828246535062499	*2.001584855884983*	*1.424752650303841*
	0.1598563243030001	*2.000327527063421*	*1.675333465729023*

Tableau VI.22 : Les coefficients a, b, c des Fonctions candidates en cloche

Comme précédemment, seul le tracé des fonctions d'appartenance pour la sortie X est donné. L'allure de variation de ces fonctions est donnée par :

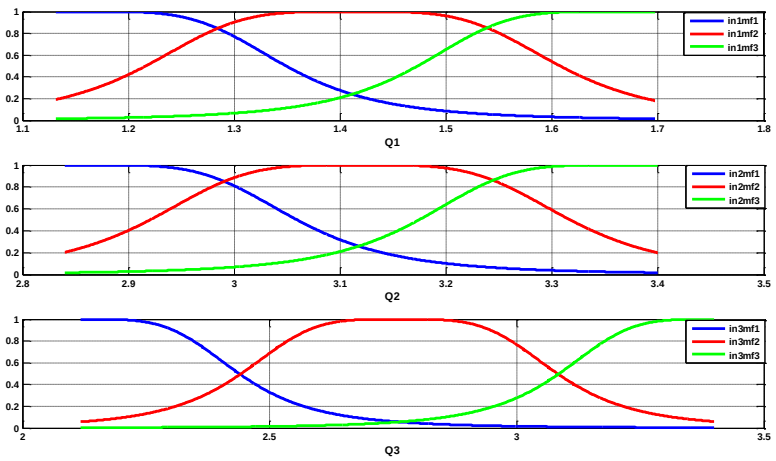


Figure VI.15 : Tracé des Fonctions d'appartenance

Les graphes des sorties en fonction des entrées sont représentés par les courbes à deux dimensions

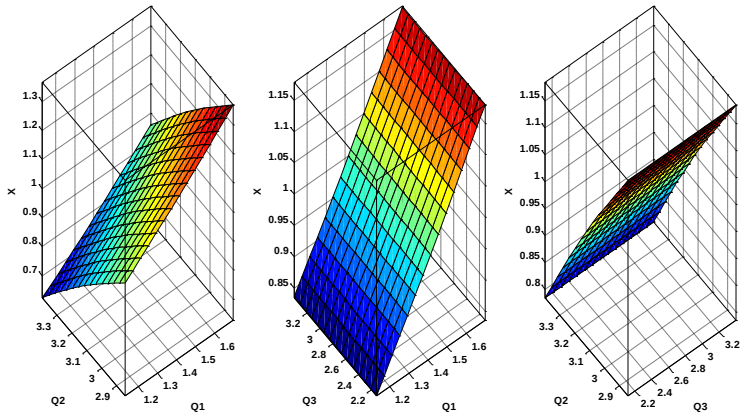


Figure VI.16 : Tracé des Fonctions d'appartenance sortie X

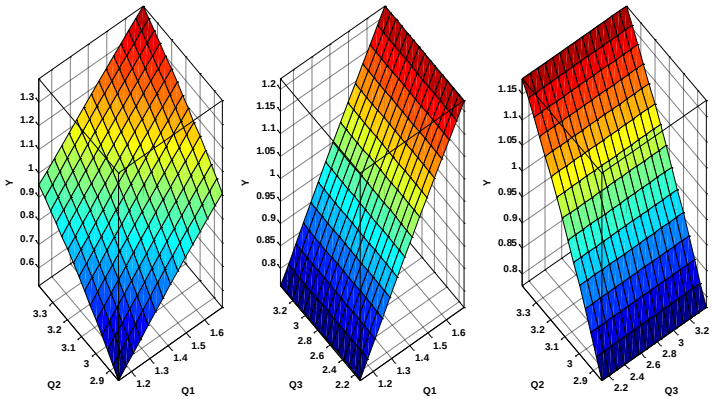


Figure VI.17 : Tracé des Fonctions d'appartenance sortie Y

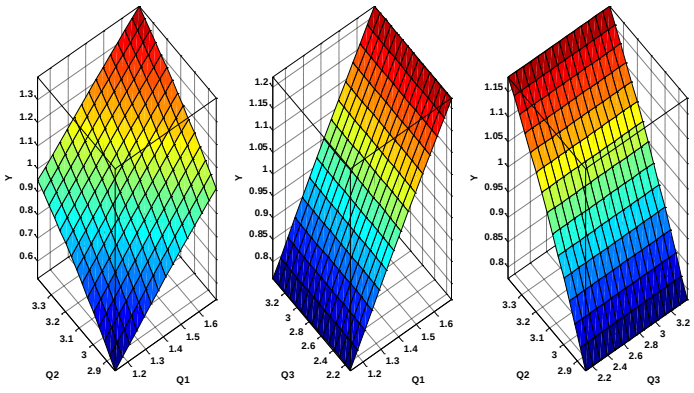


Figure VI.18 : Tracé des Fonctions d'appartenance sortie PHI

La solution trouvée en suivant la même démarche, sera :

X= 1.000000956587682

Y= 0.9999828841351883

Soit une erreur quadratique absolue égale à : $1.57\ 10^{-4}$.

Une réduction du pas de 25% (n=75), nécessitera une durée d'exécution moyenne de vingt minutes (20 min) par structure et donne les résultats suivants

$X = 1.000000956587682$

$Y = 1.000000956587682$

$\mathrm{PHI} = 1.049506257922610$

Le modèle du PC utilisé est Intel type I3, 2.4 GHz, et 4Go de RAM.

Soit une erreur quadratique absolue égale à : $1.56\ 10^{-4}$. Du même ordre que celle trouvée ci-haut mais avec trois entrées.

Une augmentation du nombre d'entrées, le pas du quadrillage étant maintenu (n=5), améliore la précision absolue sur l'abscisse X) et la porte à $0.9\ 10^{-5}$, mais au prix d'une durée d'exécution pour une seule structure (sortie X) de deux heures cinquante minutes (2h 50 min).

VI .3.4.Conclusion

Le calcul de la géométrie directe par l'approche réseaux de neurones- logique floue requiert des durées d'exécutions assez importantes pour la détermination des structures qui établissent la correspondance entrées-sorties. Mais cette étape nécessaire pour déterminer la structure floue, une fois réalisée, va pouvoir déterminer la solution du problème pour n'importe quel vecteur articulaire d'entrée, pour autant qu'il appartienne aux intervalles de variations.

La précision est nettement améliorée si on réduit la valeur du pas du quadrillage du domaine de variation. Elle l'est également si on augmente le nombre de fonctions d'appartenance par entrée. Cette méthode ne peut cependant déterminer toutes les solutions correspondant à un vecteur articulaire donné, ni a fortiori les solutions dégénérées par ailleurs, le quadrillage de l'ensemble du domaine, pour une précision satisfaisante conduit à des tailles de matrices prohibitives, et conduira inéluctablement à des dépassements de mémoire.

La solution est soit de travailler sur une station de travail performante disposant de puissance de calcul et d'une mémoire vive assez conséquente, soit de partitionner le domaine en sous-domaines, sur lesquels autant de structures seront définies, et chercher la solution optimale qui

vérifie les conditions d'appartenance au domaine et présentant l'erreur minimale si plusieurs solutions sont trouvées.

N'ayant pu pour des raisons de contraintes de temps et de moyens informatiques performants, prospecter ces deux approches, notre souhait est que cette voie soit explorée dans un travail futur.

.

CONCLUSION GENERALE

CONCLUSION GENERALE

L'objectif général de cette thèse est la formalisation générale d'un problème de la géométrie directe pour le robot parallèle **3 RPR.**

Nous avons commencé par aborder en détail la description de notre robot ainsi que la géométrie inverse qui admet pour ce cas précis une solution analytique unique. Il en est de même de la cinématique inverse. Par contre l'approche usuelle du calcul de la géométrique directe est une résolution numérique utilisant l'algorithme de Newton basé sur le calcul de la matrice jacobienne et son inversion. Enfin, nous avons donné une méthode de construction de l'espace de travail.

Apres on est passé à l'objectif principal de notre thèse, qui consiste à mener une investigation du modèle géométrique direct que nous essayons de résoudre avec plusieurs méthodes en maintenant une bonne précision, avec la méthode polynomiale qui cherche l'existence des angles φ puis calcule les coordonnées de la plate-forme $P_1(x,y)$. La deuxième méthode est une formulation analytique de la méthode géométrique graphique qui détermine toutes les positions possibles de la plateforme à partir d'intersection de cercles. L'implémentation de cette méthode permet de retrouver toutes les solutions déduites de la première méthode.

Nous avons également analysé les cas des singularités (type et solution) générées par la méthode polynomiale.

La troisième méthode basée sur la logique floue a nécessité de présenter un bref rappel sur la logique floue, ainsi que sur les réseaux de neurones pour comprendre la fonction ANFIS de Matlab qui génère automatiquement les règles d'inférence , puis la géométrie directe de notre robot, en faisant apparaitre les limites et les difficultés rencontrées lors de sa mise en œuvre.

Nous avons réalisé pour chaque méthode, un programme en code Matlab, pour la détermination des solutions et l'affichage des graphes pertinents sur interface graphique (**GUI).**

Les résultats obtenus lors de notre travail encouragent la poursuite des recherches dans cet axe. Nous pouvons également envisager les points suivants :

- Optimiser les programmes existants pour minimiser les durées d'exécution
- Faire tourner les programmes réalisés sur des machines performantes et déterminer les structures optimales ANFIS permettant d'avoir la précision maximale pour une taille de données raisonnables.

Bibliographique

Bibliographie

[1] Shiakolas, P. S., Conrad, K. L., & Yih, T. C. (2002). On the accuracy, repeatability, and degree of influence of kinematics parameters for industrial robots. *International journal of modelling and simulation*, *22*(4), 245-254.

[2] Greenway, B. (1999). On the money: The importance of robot accuracy is increasing as applications become more sophisticated. *Robotics World*, *17*(5), 44-44.

[3] Hidalgo, F., & Brunn, P. (1998). Robot metrology and calibration systems-a market review. *Industrial Robot: An International Journal*, *25*(1), 42-47.

[4] Pan, Z., Zhang, H., Zhu, Z., & Wang, J. (2006). Chatter analysis of robotic machining process. *Journal of materials processing technology*, *173*(3), 301-309.

[5] Elsheikh, A. H., Showaib, E. A., & Asar, A. E. (2013). Artificial neural network based forward kinematics solution for planar parallel manipulators passing through singular configuration. *Adv Robot Autom*, *2*(106), 2.

[6] Duka, A. V. (2014). Neural network based inverse kinematics solution for trajectory tracking of a robotic arm. *Procedia Technology*, *12*, 20-27.

[7] Duka, A.V.: ANFIS based Solution to inverse kinematics of a 3 DOF planar manipulator. In: The 8 International conference Interdisciplinarity in Engineering, INTER-ENG 2014, 9–10 October 2014,Tirgu-Mures, Romania (2014)

[8] Ali, T.H., Ismail, N., Hamouda, A.M.S., Aris, I., Marhaban, M.H., et al.: Artificial neural network-based kinematics Jacobian solution for serial manipulator passing through singular configurations.Adv. Eng. Softw. **41**, 359–367 (2010)

[9] Pozinor, B.P., Dieulot, J. Dubois, L.: « Introduction à la commande floue »Edition Technique 1998 Paris 6. Gosselin, C.M., Jean,M.: Determination of the workspace of planar parallel manipulators with joint limits. Robot. Auton. Syst. **17**,129–138 (1996)

[10] Cheng, H., Liu, G.F., Yiu, Y.K., Xiong, Z.H., Li, Z.X.: Advantages and dynamics of parallel manipulators with redundant actuation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems Proceedings, HI (2001)

[11] Gosselin, C.: Kinematic analyse optimization and programming of parallel robotic manipulators. PhD thesis, McGill University, Montréal, 15 June 1988

[12] Gosselin, C.M., Merlet, J.P.: The direct kinematics of planar parallel manipulators: special architectures and number of solutions.Mech. Mach. Theory **29**(8), 1083–1097 (1994).

[13] Merlet J-P., “Parallel robots”, 2ème Edition, Heidelberg, Springer, - 394 p., 2005.

[14] Gwinnett J.E., “Amusement device”, Brevet américain No. 1 789 680, déposé le 1er octobre 1928, émis le 20 janvier 1931.

[15] Bonev I.A., “Les robots parallèles : de la recherche vers les applications”, Journées Nationales de la Recherche en Robotique, Obernai, France, 9–12 Octobre, 2007.

[16] Pollard W.L.V., “Position controlling apparatus”, Brevet américain No. 2 286 571, depose le 22 avril 1938, émis le 16 juin 1942.

[17] Stewart D., “A platform with six degrees of freedom”, Proceedings of the IMechE, Vol. 180, Pt. 1, No. 15, pp. 371–385, 1965.

[18] Gough V.E., Whitehall S.G., “Universal tyre test machine”,Proceedings of the FISITA Ninth International Technical Congress,pp. 117–137, 1962.

[19] Bonev I.A., “Delta parallel robot—The story of success”, article en ligne : http://www.parallemic.org/Reviews/Review002.html, 2001.

[20] Bonev I.A., “Les robots parallèles : de la recherche vers les applications”, Journées Nationales de la Recherche en Robotique,Obernai, France, 9–12 Octobre, 2007.

[21] Angeles J., “Fundamentals of Robotic Mechanical Systems”, Theory, Methods and Algorithms, Springer, -549p, 2007.

[22] Chen S.-t. “Dynamic model of a hybrid robot manipulator based on Stewart platform”, DE-Vol. 72, Robotics: Kinematics, Dynamics and Controls, ASME, pp. 249-253, 1994.

[23] Merlet J.-P., Gosselin C., "Nouvelle architecture pour un manipulateur parallèle à 6 degrés de liberté", J. Mechanism and Machine Theory 26(1), 1991, p.77-90.

[24] Pasqui V., Bidaud P., Ben ouezdou F., "Analyse des performances Cinématiques des Macro/Mini-Manipulateurs", Acte du 12ième congrès Français de mécanique, Strasbourg, 4-8 septembre, p. 401-404, 1995.

[25] Waldron K. J., Raghavan M., Roth B., "Kinematiks of a Hybrid Serial-Parallel Manipulation System", Trans. of the ASME, J. of. Mechanisms, Trans and Auto. In design, Vol 111, 1989, p.211-221.

[26] Zhang C. de, Song s. m., "Geometry and position analysis of a novel class of hybrid manipulators", DE-Vol. 72, Robotics: Kinematics, Dynamics and Controls, ASME, 1994, p. 1-9.

[27] Tanio K. Tanev, "Kinematics of a hybrid (parallel-serial) robot manipulator", J. Mechanism and Machine Theory 35, 2000, p.1183-1196.

[28] Charentus S., Renaud M., "Modeling and controle of a modular redundant robot manipulator". First International Symposium on Experimental Robotics, Monreal, Canada, juin 1989.

[29] Pierrot F., “Parallel mechanisms and redudancy”, In 1st Int.Colloquium, Collaborative Research Centre 562, pp. 261-277,Braunschweig, 29-30 Mai 2002.

[30] Tsai L.W., “Robot Analysis: the Mechanics of Serial and Parallel Manipulators”, New York, John Wiley and Sons, -505p., 1999.

[31] D. McCoy, « Some comparisons of serial-driven and parallel-driven manipulators », Robotica, volume 8, pp. 355-362, 1990.

[32] M.J.D. Hayes, M.L. Husty, « On the kinematic constraint surfaces of general three-legged planar robot platforms », Mechanism and Machine Theory 38, pp. 379–394, 2003.

[33] M.J.D. Hayes, P.J. Zsombor-Murray, C. Chen, « Unified Kinematic Analysis of General Planar Parallel Manipulators » ASME Journal of Mechanical Design, vol 126, no 5, pp. 866-874. septembre 2004.

[34] I.A. Bonev, D. Zlatanov, C.M. Gosselin Singularity analysis of 3-DOF planar parallel mechanisms via screw theory Journal of Mechanical Design., 125 (2003), pp. 573-581

[35] Chablat D., Wenger P., “Design of a spherical wrist with parallel architecture: application to vertebrae of an eel robot”, Proceedings of IEEE Conference on Robotics and Automation, pp.3347-3352,Barcelone, 2005.

[36] Hunt K. H., “Structural kinematics of in-parallel-actuated robot arms”,ASMR Transaction, Journal Mechanics Transmissions Automatation Design 105, pp.705-7012, 1983.

[37] Tlusty J., Ziegert J.C., Ridgeway S., “Fundamental comparison of the use of serial and parallel kinematics for machine tools”, Annals of the CIRP, Vol. 48, No. 1, pp. 351–356, 1999.

[38] Wenger Ph., Gosselin C. et Maille B., “A comparative study of serial and parallel mechanism topologies for machine tools”, Proceedings of PKM’99, pp. 23–32, Milan, Italy, 1999.

[39] Weck M., et Staimer M., “Parallel Kinematic Machine Tools –Current State and Future Potentials”, Annals of the CIRP, Vol. 51, No.2, pp. 671-683, 2002.

[40] Chablat D., Wenger Ph., “Architecture Optimization of a 3-DOF Parallel Mechanism for Machining Applications, the Orthoglide”,IEEE Transactions on Robotics and Automation, Vol. 19/3, pp. 403-410, Juin 2003.

[41] Pashkevich A., Wenger Ph., Chablat D., “Design Strategies for the Geometric Synthesis of Orthoglide-type Mechanisms”, Journal of Mechanism and Machine Theory, Vol. 40, Issue 8, pp. 907-930,August 2005.

[42] Hervé J. M., Sparacino F., “Structural synthesis of parallel robots generating spatial translation”, Proceedings 5th Int. Conf. Advanced Robotics, Vol. 1, pp. 808–813, 1991.

[43] X. Kong et C. M. Gosselin, “Type synthesis of linear translational parallel manipulators”, Advances in Robot Kinematic, J. Lenarcic and F. Thomas, Eds. Norwell, MA: Kluwer, pp. 453–462, 2002.

[44] K. E. Neumann, Robot. U. S. Patent 4 732 525, 22 Mars 1988.

[45] Toyama T. et al., "Machine tool having parallel structure", U. S.Patent 5 715 729, February. 10, 1998.

[46] Clavel R., "DELTA, A Fast Robot with Parallel Geometry", Proceedings of 18th international symposium on industrial robots, Lausanne, pp. 91-100, 1988.

[47] Company O., Pierrot F., Launay F., Fioroni C., "Modeling and preliminary design issues of a 3-axis parallel machine tool», Proceedings International Conference PKM 2000, pp. 14–23, Ann Arbor, MI, 2000.

[48] Chablat D., Wenger Ph., "Device for the movement and orientation of an object in space and use thereof in rapid machining", Brevet Européen EP1597017, 23/11/2005; Brevet Canadien CA2515024, (26/08/2004). Demande PCT: WO2004071705, 26/08/2004. Déposant : Centre National de la Recherche Scientifique CNRS/Ecole Centrale de Nantes. Mandataire : Cabinet LAVOIX.

[49] Majou F., "Analyse Cinétostatique des Machines Parallèles à Translations", Thèse de doctorat de l'Ecole Centrale de Nantes, 2004.

[50] Caro S., Wenger P., Bennis F. et Chablat D., "Sensitivity Analysis of the Orthoglide, A 3-DOF Translational Parallel Kinematic Machine",ASME Journal of Mechanical Design, Vol.128, pp. 392–402, Mars 2006.

[51] Kim, H.S., et Tsai, L.W., "Evaluation of a Cartesian manipulator», Advances in Robot Kinematic, J. Lenarcic and F. Thomas, Eds.Norwell, MA: Kluwer, pp. 21-38, 2002.

[52] X. Kong et C. M. Gosselin, "Type synthesis of linear translational parallel manipulators", Advances in Robot Kinematic, J. Lenarcic and F. Thomas, Eds. Norwell, MA: Kluwer, pp. 453–462, 2002.

[53] Gogu G., "Fully-isotropic T1R2-type parallel robots with three degrees of freedom", Proceedings International design Engineering Technical Conferences & Computers and information in Engineering Conference, Long Beach, 2005.

[54] Gogu G., "Fully-isotropic three-degree-of-freedom parallel wrists», Proceedings IEEE International Conference on Robotics and Automation, Rome, pp.895-900, 2007.
[55] Company O., Pierrot F., "Modelling and Design Issues of a 3-axis Parallel Machine-Tool", Mechanism and Machine Theory, Vol. 37, pp 1325-1345, 2002.

[56] Merlet J.-P. "Determination of the orientation workspace of parallel manipulators", Journal of Intelligent and Robotic Systems Vol 13, No 1, pp.143-160, 1995.

[57] Waldron K. J., Raghavan M., et Roth B., "Kinematics of a hybrid series-parallel manipulation system", Journal of Dynamic System,Measuring, Control., Vol.111, No. 2, pp. 211–221, 1989.

[58] Liu K., Fitzgerald J. M., Lewis F. L., "Kinematic analysis of a Stewart platform manipulator", IEEE Transaction Ind. Electron., Vol. 40, pp.282–293, Février 1993.

[59] Innocenti C., Parenti-Castelli V., "Direct kinematics of the 6-4 fully parallel manipulator with position and orientation decoupled", Robotic Systems, S. G. Tzafestas, Ed. Norwell, MA: Kluwer, pp. 3–10, 1992.

[60] Ji P., Wu H., "A closed-form forward kinematics solution for the 6 – 6 Stewart platform", IEEE Transaction Robotic Automatation, Vol. 17,pp. 522–526, Aout 2001.

[61] Dieudonne J. E., Parrish R. V., Bardusch R. E., "An actuator extension transformation for a motion simulator and an inverse transformation applying Newton–Raphson's method", NASA Langley Research Center, Hampton, VA, Tech. Rep. NASA TND-7067, 1972.

[62] Merlet J. P., "Direct kinematics of parallel manipulator", IEEE Transaction Robotic Automation, Vol. 9, pp. 842–846, Décembre 1993.

[63] Han K., Chung W., Youm Y., "Local structurization for the forward kinematics of parallel manipulators using extra sensor data", Proceedings IEEE International Conference Robotics, Automation, pp. 514–520, 1995.

[64] Tancredi L., Teillaud M., Merlet J-P, "Extra sensors data for solving the forward kinematics problem of parallel manipulators", 9th IFToMM World Congress on the Theory of Machines and Mechanisms, pp. 2122-2126, Milan, Septembre, 1995.

[65] Parikh P.J., Lam S.S.Y., "A hybrid strategy to solve the forward kinematics problem in parallel manipulators", IEEE Transaction on Robotics and Automation, Vol. 21, No 1, pp.18-25, Février 2005.

[66] Gosselin C.M., Angeles J., "Singularity Analysis of Closed-Loop Kinematic Chains", IEEE Transactions on Robotics and Automation,Vol.6, pp.281-290, Juin 1990.

[67] Sefrioui J., Gosselin C.M., "Singularity Analysis and Representation of Planar Parallel Manipulators", Robotic and Autonomous Systems, pp.209-224, 1993.

[68] Gosselin C.M., Wang J., "Singular Loci of Planar Parallel Manipulators", 9th World Congress on the Theory of Machines and Mechanisms, Vol.3, Milan, Italie, Aôut/Septembre 1995.

[69] Ma O., Angeles J., "Architecture Singularities of Platform Manipulators", Proceedings of the IEEE International Conference on Robotics and Automation, pp.1542-1547, Sacramento, Californie, Avril 1991.

[70] Khalil W., Murareci D., "Kinematic Analysis and Singular Configurations of a Class of Parallel Robots", Mathematics and Computers in Simulation, pp.377-390, 1996.

[71] Dheeman B., Ashitava G., "Singularity Analysis of Platform-Type Multi-Loop Spatial Mechanisms", Mechanism and Machine Theory,Vol.33, pp.375-389, 1997.

[72] Zlatanov D., Fenton R.G., Benhabib B., "Singularity analysis of mechanism and Robots via a Velocity Equation Model of the Instantaneous Kinematics", Proceedings of the IEEE International Conference on Robotics and Automation, pp. 986-991, San Diego,1994.

[73] Zlatanov D., Bonev I.A., Gosselin C.M., "Constraint singularities",Web review, www.parallemic.orgReviews/Review005.html,Laboratoire de robotique de l'Université Laval, Canada, 2001.

[74] Krut S., "Contributions à l'étude des robots parallèles légers, 3T-1R et 3T-2R, à forts débattements angulaires", Thèse de Doctorat, LIRMM, Université de Montpellier II, Octobre 2003.

[75] Dandurand A., "The rigidity of compound spatial grid", Structural Topology 10, pp. 43-55, 1984.

[76] Merlet J. P., "Singular Configurations of Parallel Manipulators and Grassmann Geometry", The International Journal of Robotics Research, Vol. 8, No. 5, 1989.

[77] Hao F., et McCarthy J.M., "Conditions for Line-Based Singularities in Spatial Platform Manipulators", Journal of Robotic Systems, Vol. 15, No.1, pp. 43-55, 1998.

[78] Jo D. Y., Haug E. J., « Workspace analysis of closed-loop mechanisms with unilateral constraints », ASME Design Automation Conference, March 1989.

[79] Haugh R. J., Adkins F. A.; Luh C. M., « Domain of Operation and Interference for Bodies in Mechanisms and Manipulators », In J-P. Merlet, B. Ravani editor, Computational Kinematics, pp. 193-202,

[80] Chablat D., "Domaines d'Unicité et Parcourabilité pour les Manipulateurs Pleinement Parallèles", Thèse de Doctorat, École Centrale de Nantes, Novembre 1998.

[81] Merlet J.P., "Parallel Robots, Solid Dynamics and its Applications", Kluwer Academic Publishers, 2000.

[82] Bonev I.A., "Geometric Analysis of Parallel Mechanisms", Thèse de doctorat, Université Laval, Novembre 2002.

[83] Merlet J.P., "An Improved Design Algorithm Based on Interval Analysis for Parallel Manipulator with Specified Workspace", Proceedings of the IEEE International Conference on Robotics and Automation, Seoul, 23-25 Mai 2001.

[84] Adel OLABI, Amélioration de la précision des robots industriels pour des applications d'usinage à grande vitesse. L'École Nationale Supérieure d'Arts et Métiers, 23 Novembre 2011

[85] Contribution à l'amélioration de la précision absolue des robots parallèles

[86] ISO: International Organization for Standardization. 1998. *Manipulating industrial robots –Performance criteria and related test methods*, NF EN ISO9283.

[87] Greenway, B. 2000. « Robot accuracy ». *Industrial Robot: An International Journal*, vol. 27, n° 4, p. 257-265.

[88] Traité de robotique 3 La partie commande, Gestuelle, commandes, precision, vibration, sécurité Charle BOP

[89] Elatta, A.Y., L. P. Gen, F.L. Zhi, Y. Daoyuan et L. Fei. 2004. « An overview of Robotic calibration ». *Information Technology Journal*, vol. 3, n° 1, p. 74-78.

[90] Schroer, K. 1994. « Robocal-the IPK robot calibration package ». *Industrial Robot: An International Journal*, vol. 21, n° 6. P. 35-39.

[91] Giaouris, D. EEE 8005–Student Directed Learning (SDL) Industrial Automation–Artificial Neural networks Written by: Shady Gadoue, *Threshold*, *1*, 1.

[92] Al-Refaie, A., and R. K. Arya (2012), of International Journal of Fuzzy System Applications, *Contemporary Theory and Pragmatic Approaches in Fuzzy Computing Utilization*, 293.

[93] Hagan, M., H. Demuth, and M. Beale (2003), Neural Network Design, 7th International Student Edition, edited, Vikas Publishing House.

[94] GHENNAM, A. (2012), Commande Compliante Intelligente d'un Bras Manipulateur Rigide pour des Applications de Chirurgie Médicale, Université de Batna 2.

[95] Moutarde, F. (2008). Introduction aux réseaux de neurones

[96] GUITANI Issam « Commande Adaptative Neuronale par Retour de Sortie des Systèmes Non Linéaires » thèse de magister à l'université de Constantine, 2007

[97] Eric GAUTHIER, « Utilisation des Réseaux de Neurones Artificiels pour la Commande d'un Véhicule Autonome », thèse de docteur à l'l'institut national polytechnique de GRENOBLE 1999.
[98] E. Davalo et P. Naim, ″ Des Réseaux de Neurones″, Editions Eyrolles, deuxième édition, 1993.

[99] Sofiane, CHEKROUN. Commande neuro floue sans capteur de vitesse d'une machine asynchrone triphasé, mémoire de Magister .Ecole Normale Supérieur d'Enseignement Technique d'Oran, Algérie ,2009.

[100] M. Parizeau ‘’Réseaux Neurones ‘’, GIF -21140 et GIF -64326 ,2004.

[101] Nauck, D. (2000), Data Analysis with Neuro Fuzzy Methods Habilitation thesis, *Otto-von-Guericke University of Magdeburg, Faculty of Computer Science, Magdeburg, Germany*.

[102] Vasile, O. E. (2008), Contribution au pronostic de défaillances par réseau neuro-flou: maîtrise de l'erreur de prédiction, *Thèse de doctorat*.

[103] Nauck. D, et R. Kruse, "What are Neuro-Fuzzy Classifiers?" *Seventh International Fuzzy Systems Association World Congress IFSA'97*, Vol. IV, pp. 228-233, Academie de Prague, 1997.

[104] Nauck, D., Klawonn, F., and Kruse, R., Foundation of Neuro-Fuzzy Systems, John Wiley & Sons Co., New York, 1997.

Annexes

ANNEXE A

ANNEXE A : METHODE POLYNOMIALE

ANNEXE A1 POLYNOME ORDRE SIX (SOLUTION ANGLE PHI)

$$p_6 t^6 + p_5 t^5 + p_4 t^4 + p_3 t^3 + p_2 t^2 + p_1 t + p_0 = 0$$

$$t = tan\left(\frac{\varphi}{2}\right)$$

*p6=l2^2*x3^4 - 2*l2^3*x3^3 + l2^4*x3^2 + l3^2*x2^4 + l3^4*x2^2 + l2^2*y3^4 + l2^4*y3^2 + q1^4*x2^2 + q1^4*x3^2 + q2^4*x3^2 + q3^4*x2^2 + q1^4*y3^2 + q2^4*y3^2 + x2^2*x3^4 - 2*x2^3*x3^3 + x2^4*x3^2 + x2^2*y3^4 + x2^4*y3^2 + l2^2*l3^4 + l2^4*l3^2 + l2^2*q1^4 + l3^2*q1^4 + l2^2*q3^4 + l3^2*q2^4 + 2*l2*l3^4*x2 + 2*l2*q1^4*x2 - 2*l2*q1^4*x3 + 2*l2*q3^4*x2 + 2*l2*x2*x3^4 + 2*l2*x2*y3^4 - 2*q1^4*x2*x3 + 4*l2*l3^2*x2^3 + 4*l2^3*l3^2*x2 - 4*l2^3*l3^2*x3 - 2*l2*q1^2*x3^3 - 2*l2^3*q1^2*x3 + 2*l2*q2^2*x3^3 + 2*l2^3*q3^2*x3 - 6*l2*x2^2*x3^3 + 4*l2*x2^3*x3^2 - 6*l2^2*x2*x3^3 + 4*l2^3*x2*x3^2 - 4*l3^2*x2^3*x3 + 4*l2*x2^3*y3^2 + 4*l2^3*x2*y3^2 - 2*l2^3*x3*y3^2 - 2*q1^2*x2*x3^3 - 2*q1^2*x2^3*x3 + 2*q2^2*x2*x3^3 + 2*q3^2*x2^3*x3 - 2*x2^3*x3*y3^2 - 2*l2^2*l3^2*q2^2 - 2*l2^2*l3^2*q3^2 - 2*l2^3*l3^3*cos(beta) - 2*l2^2*q1^2*q3^2 - 2*l3^2*q1^2*q2^2 + 6*l2^2*l3^2*x2^2 + 4*l2^2*l3^2*x3^2 + 4*l2^2*l3^2*y3^2 + 4*l2^2*q1^2*x3^2 - 2*l2^2*q2^2*x3^2 - 2*l3^2*q2^2*x2^2 - 2*l2^2*q3^2*x3^2 - 2*l3^2*q3^2*x2^2 - 2*l2^2*q2^2*y3^2 - 2*l2^2*q3^2*y3^2 - 2*l3^3*x2^3*cos(beta) - 2*q1^2*q2^2*x3^2 - 2*q1^2*q3^2*x2^2 - 2*q1^2*q2^2*y3^2 + 6*l2^2*x2^2*x3^2 + 4*l3^2*x2^2*x3^2 + 6*l2^2*x2^2*y3^2 + 2*l2^2*x3^2*y3^2 + 4*l3^2*x2^2*y3^2 + 4*q1^2*x2^2*x3^2 - 2*q2^2*x2^2*x3^2 - 2*q3^2*x2^2*x3^2 - 2*q2^2*x2^2*y3^2 - 2*q3^2*x2^2*y3^2 + 2*x2^2*x3^2*y3^2 + 2*l3*x2^4*y3*sin(beta) + 2*l2^2*l3^2*x3^2*cos(2*beta) - 2*l2^2*l3^2*y3^2*cos(2*beta) + 4*l3^2*q1^2*x2^2*cos(beta)^2 + 2*l3^2*x2^2*x3^2*cos(2*beta) - 2*l3^2*x2^2*y3^2*cos(2*beta) - 2*l2*l3^3*q1^2*cos(beta) - 2*l2^3*l3*q1^2*cos(beta) + 2*l2*l3^3*q2^2*cos(beta) + 2*l2^3*l3*q3^2*cos(beta) - 6*l2*l3^3*x2^2*cos(beta) - 6*l2^2*l3^3*x2*cos(beta) + 4*l2^2*l3^3*x3*cos(beta) + 4*l2^2*l3^3*x3*cos(beta) - 6*l2^3*l3^3*x2*cos(beta) - 2*l2^3*l3*y3^2*cos(beta) - 2*l3*q1^2*x2^3*cos(beta) - 2*l3^3*q1^2*x2*cos(beta) + 2*l3^3*q2^2*x2*cos(beta) + 2*l3*q3^2*x2^3*cos(beta) + 4*l3*x2^2*x3^3*cos(beta) - 6*l3*x2^3*x3^2*cos(beta) + 4*l3^3*x2^2*x3*cos(beta) - 2*l3*x2^3*y3^2*cos(beta) + 4*l2^2*l3*y3^3*sin(beta) + 4*l2^2*l3^3*y3*sin(beta) + 4*l3*x2^2*y3^3*sin(beta) + 4*l3^3*x2^2*y3*sin(beta) - 2*l2^3*l3^2*x3*cos(2*beta) - 2*l2^3*l3*x2^3*x3*cos(2*beta) - 2*l2*l3^2*y3*sin(2*beta) - 2*l3^2*x2^3*y3*sin(2*beta) - 2*l2*l3*q1*cos(beta) + 2*l2*l3^3*q1^2*cos(beta) + 2*l2*l3^3*q2^2*cos(beta) + 2*l2^3*l3*q3^2*cos(beta) - 6*l2*l3^3*x2^2*cos(beta) - 6*l2^2*l3^3*x2*cos(beta) + 4*l2^2*l3^3*x3*cos(beta) + 4*l2^2*l3^3*x3*cos(beta) - 6*l2^3*l3^3*x2*cos(beta) - 2*l2^3*l3*y3^2*cos(beta) - 2*l3*q1^2*x2^3*cos(beta) - 2*l3^3*q1^2*x2*cos(beta) + 2*l3^3*q2^2*x2*cos(beta) + 2*l3*q3^2*x2^3*cos(beta) + 4*l3*x2^2*x3^3*cos(beta) - 6*l3*x2^3*x3^2*cos(beta) + 4*l3^3*x2^2*x3*cos(beta) - 2*l3*x2^3*y3^2*cos(beta) + 4*l2^2*l3*y3^3*sin(beta) + 4*l2^2*l3^3*y3*sin(beta) + 4*l3*x2^2*y3^3*sin(beta) + 4*l3^3*x2^2*y3*sin(beta) - 2*l2^3*l3^2*x3^2*cos(2*beta) - 2*l2*l3^3*x2^3*cos(2*beta) - 2*l2^3*l3^2*y3^2*sin(2*beta) - 2*l2*l3^3*x2^3*sin(2*beta) - 2*l2*l3^3*q1*cos(beta) - 4*l2*l3^3*q1^2*x3 - 4*l2*l3^3*q2^2*x2 + 4*l2*l3^3*q2^2*x3 - 4*l2*l3^3*q3^2*x2 + 2*l2^4*l3*x3*cos(beta) + 2*l2*q1^2*q2^2*x3 - 4*l2*q1^2*q3^2*x2 + 2*l2*q1^2*q3^2*x3 - 2*l2*q2^2*q3^2*x3 + 8*l2*l3^2*x2*x3^2 - 12*l2*l3^2*x2^2*x3 - 12*l2^2*l3^2*x2*x3 + 8*l2*l3^2*x2*y3^2 - 2*l3*q1^4*x2*cos(beta) + 2*l3*q1^4*x3*cos(beta) + 2*l3*q2^4*x3*cos(beta) + 8*l2*q1^2*x2*x3^2 - 6*l2*q1^2*x2^2*x3 - 6*l2^2*q1^2*x2*x3 - 4*l2*q2^2*x2*x3^2 - 4*l3^2*q1^2*x2*x3 - 4*l2*q3^2*x2*x3^2 + 6*l2*q3^2*x2^2*x3 + 6*l2^2*q3^2*x2*x3 + 4*l3^2*q2^2*x2*x3 - 2*l2*q1^2*x3*y3^2 - 4*l2*q2^2*x2*y3^2 + 2*l2*q2^2*x3*y3^2 - 4*l2*q3^2*x2*y3^2 + 2*l3*x2^4*x3*cos(beta) + 2*q1^2*q2^2*x2*x3 + 2*q1^2*q3^2*x2*x3 - 2*q2^2*q3^2*x2*x3 + 4*l2*x2*x3^2*y3^2 - 6*l2*x2^2*x3*y3^2 - 6*l2^2*x2*x3*y3^2 + 2*l2^4*l3*y3*sin(beta) - 2*q1^2*x2*x3*y3^2 + 2*q2^2*x2*x3*y3^2 + 2*l3*q1^4*y3*sin(beta) + 2*l3*q2^4*y3*sin(beta) + 4*l2^2*l3^2*q1^2*cos(beta)^2 - 2*l2*l3^2*q1^2*x3*cos(2*beta) + 2*l2*l3^2*q2^2*x3*cos(2*beta) + 8*l2*l3^2*q1^2*x2*cos(beta)^2 + 4*l2*l3^2*x2*x3^2*cos(2*beta) - 6*l2*l3^2*x2^2*x3*cos(2*beta) - 6*l2^2*l3^2*x2*x3*cos(2*beta) - 4*l2*l3^2*x2*y3^2*cos(2*beta) - 2*l3^2*q1^2*x2*x3*cos(2*beta) + 2*l3^2*q2^2*x2*x3*cos(2*beta) - 2*l2*l3^2*q1^2*y3*sin(2*beta) + 2*l2*l3^2*q2^2*y3*sin(2*beta) - 6*l2*l3^2*x2^2*y3*sin(2*beta) - 6*l2^2*l3^2*x2*y3*sin(2*beta) + 4*l2^2*l3^2*x3*y3*sin(2*beta) - 2*l3^2*q1^2*x2*y3*sin(2*beta) + 2*l3^2*q2^2*x2*y3*sin(2*beta) + 4*l3^2*x2^2*x3*y3*sin(2*beta) + 8*l2*l3^3*x2*x3*cos(beta) + 8*l2*l3^3*x2^2*y3*sin(beta) + 8*l2*l3^3*x2^3*y3*sin(beta) + 8*l2*l3^3*x2*y3*sin(beta) + 8*l2^3*l3*x2*y3*sin(beta) - 4*l2^3*l3*x3*y3*sin(beta) - 4*l3*x2^3*x3*y3*sin(beta) + 2*l2*l3*q1^2*q2^2*cos(beta) + 2*l2*l3*q1^2*q3^2*cos(beta) - 2*l2*l3*q2^2*q3^2*cos(beta) - 6*l2*l3*q1^2*x2^2*cos(beta) - 6*l2^2*l3*q1^2*x2*cos(beta) - 6*l2*l3*q1^2*x3^2*cos(beta) + 8*l2^2*l3*q1^2*x3*cos(beta) + 6*l2*l3*q2^2*x3^2*cos(beta) + 6*l2^2*l3*q3^2*x2*cos(beta) - 4*l2^2*l3*q2^2*x3*cos(beta) - 2*l2*l3*q1^2*y3^2*cos(beta) + 2*l2*l3*q2^2*y3^2*cos(beta) - 2*l3*q1^2*x2^2*x3*cos(beta) + 2*l3*q1^2*x3^2*cos(beta) + 2*l3*q2^2*x2^2*x3*cos(beta) - 2*l3*q2^2*x3^2*cos(beta) - 18*l2*l3*x2^2*x3^2*cos(beta) - 18*l2^2*l3*x2*x3^2*cos(beta) + 12*l2^2*l3*x2^2*x3*cos(beta) - 6*l2*l3*x2^2*y3^2*cos(beta) - 6*l2^2*l3*x2*y3^2*cos(beta) + 4*l2^2*l3*x3*y3^2*cos(beta) - 6*l3*q1^2*x2*x3^2*cos(beta) + 8*l3*q1^2*x2^2*x3*cos(beta) + 6*l3*q2^2*x2*x3^2*cos(beta) - 4*l3*q2^2*x3^2*cos(beta) - 4*l3*q3^2*x2^2*x3*cos(beta) - 2*l3*q1^2*x2*y3^2*cos(beta) + 4*l3*x2^2*x3*y3^2*cos(beta) - 4*l3*q1^2*q2^2*y3*sin(beta) - 4*l2^2*l3*q3^2*y3*sin(beta) + 4*l3*x2^2*x3*y3^2*cos(beta) - 4*l3*q1^2*q2^2*y3*sin(beta) + 12*l2^2*l3*x2^2*y3*sin(beta) + 4*l2^2*l3*x3^2*y3*sin(beta) - 4*l3*q2^2*x2^2*y3*sin(beta) - 4*l3*q3^2*x2^2*y3*sin(beta) + 4*l3*x2^2*x3^2*y3*sin(beta) + 8*l2*l3^2*x2*x3*y3*sin(2*beta) + 16*l2*l3*q1^2*x2*x3*cos(beta) - 8*l2*l3*q2^2*x2*x3*cos(beta) - 8*l2*l3*q3^2*x2*x3*cos(beta) + 8*l2*l3*x2*x3*y3^2*cos(beta) - 4*l2*l3*q1^2*x3*y3*sin(beta) - 8*l2*l3*q2^2*x2*y3*sin(beta) + 4*l2*l3*q2^2*x3*y3*sin(beta) - 8*l2*l3*q3^2*x2*y3*sin(beta) + 8*l2*l3*x2*x3^2*y3*sin(beta) - 12*l2*l3*x2^2*x3*y3*sin(beta) - 12*l2^2*l3*x2*x3*y3*sin(beta) - 4*l3*q1^2*x2*x3*y3*sin(beta) + 4*l3*q2^2*x2*x3*y3*sin(beta)*

*p5=4*l2^3*y3^3 + 4*l2*q1^4*y3 + 8*l2^3*l3^2*y3 + 4*l2*q1^2*y3^3 + 4*l2^3*q1^2*y3 - 4*l2*q2^2*y3^3 - 4*l2^3*q3^2*y3 + 4*l2*x2^2*y3^3 + 8*l2^2*x2*y3^3 + 4*l2^3*x3^2*y3 - 4*l3^3*x2^3*sin(beta) + 8*l2^2*l3^2*x3^2*sin(2*beta) - 8*l2^2*l3^2*y3^2*sin(2*beta) + 8*l3^2*q1^2*x2^2*sin(2*beta) + 8*l3^2*x2^2*x3^2*sin(2*beta) - 8*l3^2*x2^2*y3^2*sin(2*beta) - 8*l2^2*l3*y3^3*cos(beta) - 8*l2^2*l3^3*y3*cos(beta) - 8*l3*x2^2*y3^3*cos(beta) - 8*l3^3*x2^2*y3*cos(beta) - 8*l2*l3^3*x2^2*sin(beta) - 4*l2^2*l3^3*x2*sin(beta) + 8*l2^2*l3*x3^3*sin(beta) + 8*l2^2*l3^3*x3*sin(beta) - 8*l2^3*l3^3*x2*sin(beta) + 8*l2^3*l3*y3^2*sin(beta) - 4*l3*q1^2*x2^3*sin(beta) - 4*l3^3*q1^2*x2*sin(beta) + 4*l3^3*q2^2*x2*sin(beta) + 4*l3*q3^2*x2^3*sin(beta) + 8*l3*x2^2*x3^3*sin(beta) - 12*l3*x2^3*x3^2*sin(beta) + 8*l3^3*x2^2*x3*sin(beta) - 4*l3*x2^3*y3^2*sin(beta) + 4*l2^3*l3^2*y3*cos(2*beta) + 8*l3^2*x2^3*y3*cos(2*beta) - 4*l2^3*l3^2*x3*sin(2*beta) -*

*8*l3^2*x2^3*x3*sin(2*beta) + 8*l2*l3^2*q1^2*y3 - 8*l2*l3^2*q2^2*y3 - 4*l2^4*l3*y3*cos(beta) - 4*l2*q1^2*q2^2*y3 - 4*l2*q1^2*q3^2*y3 + 4*l2*q2^2*q3^2*y3 + 8*l2*l3^2*x2^2*y3 + 16*l2^2*l3^2*x2*y3 - 4*l3*q1^4*y3*cos(beta) - 4*l3*q2^4*y3*cos(beta) + 4*l2*q1^2*x2^2*y3 + 8*l2^2*q1^2*x2*y3 + 4*l2*q1^2*x3^2*y3 - 16*l2^2*q1^2*x3*y3 - 4*l2*q2^2*x3^2*y3 - 4*l2*q3^2*x2^2*y3 - 8*l2^2*q3^2*x2*y3 - 4*l3*x2^4*y3*cos(beta) + 4*l2^4*l3*x3*sin(beta) + 4*l2*x2^2*x3^2*y3 + 8*l2^2*x2*x3^2*y3 - 4*l3*q1^4*x2*sin(beta) + 4*l3*q1^4*x3*sin(beta) + 4*l3*q2^4*x3*sin(beta) + 4*l3*x2^4*x3*sin(beta) - 16*l2*q1^2*x2*x3*y3 + 4*l2*l3^2*q1^2*y3*cos(2*beta) - 4*l2*l3^2*q2^2*y3*cos(2*beta) + 20*l2*l3^2*x2^2*y3*cos(2*beta) + 16*l2^2*l3^2*x2*y3*cos(2*beta) - 16*l2^2*l3^2*x3*y3*cos(2*beta) + 8*l3^2*q1^2*x2*y3*cos(2*beta) - 8*l3^2*q2^2*x2*y3*cos(2*beta) + 8*l2*l3^2*q1^2*x2*sin(2*beta) - 4*l2*l3^2*q1^2*x3*sin(2*beta) + 4*l2*l3^2*q2^2*x3*sin(2*beta) - 16*l3^2*x2^2*x3*y3*cos(2*beta) + 16*l2*l3^2*x2*x3^2*sin(2*beta) - 20*l2*l3^2*x2^2*x3*sin(2*beta) - 16*l2^2*l3^2*x2*x3*sin(2*beta) - 16*l2*l3^2*x2*y3^2*sin(2*beta) - 8*l3^2*q1^2*x2*x3*sin(2*beta) + 8*l3^2*q2^2*x2*x3*sin(2*beta) - 16*l2*l3*x2^3*y3*cos(beta) - 16*l2*l3^3*x2*y3*cos(beta) - 16*l2^3*l3*x2*y3*cos(beta) + 16*l2^3*l3*x3*y3*cos(beta) + 8*l3*x2^3*x3*y3*cos(beta) + 16*l2*l3*x2*x3^3*sin(beta) + 16*l2*l3*x2^3*x3*sin(beta) + 16*l2^3*l3*x2*x3*sin(beta) - 16*l2^3*l3*q1^2*y3*cos(beta) + 8*l2^2*l3*q2^2*y3*cos(beta) + 8*l2^2*l3*q3^2*y3*cos(beta) + 8*l3*q1^2*q2^2*y3*cos(beta) - 24*l2^2*l3*x2^2*y3*cos(beta) - 8*l2^2*l3*x3^2*y3*cos(beta) + 8*l3*q2^2*x2^2*y3*cos(beta) + 8*l3*q3^2*x2^2*y3*cos(beta) - 8*l2*l3*q1^2*x2^2*sin(beta) - 4*l2^2*l3*q1^2*x2*sin(beta) - 8*l2*l3*q1^2*x3^2*sin(beta) + 8*l2*l3*q2^2*x3^2*sin(beta) + 8*l2*l3*q3^2*x2^2*sin(beta) - 8*l2^2*l3*q2^2*x3*sin(beta) + 4*l2^2*l3*q3^2*x2*sin(beta) - 8*l2^2*l3*q3^2*x3*sin(beta) + 8*l2*l3*q1^2*y3^2*sin(beta) - 8*l2*l3*q2^2*y3^2*sin(beta) - 8*l3*x2^2*x3^2*y3*cos(beta) + 4*l3*q1^2*q2^2*x2*sin(beta) - 8*l3*q1^2*q2^2*x3*sin(beta) + 4*l3*q1^2*q3^2*x2*sin(beta) - 4*l3*q2^2*q3^2*x2*sin(beta) - 32*l2*l3*x2^2*x3^2*sin(beta) - 28*l2^2*l3*x2*x3^2*sin(beta) + 24*l2^2*l3*x2^2*x3*sin(beta) + 12*l2^2*l3*x2*y3^2*sin(beta) + 8*l2^2*l3*x3*y3^2*sin(beta) - 12*l3*q1^2*x2*x3^2*sin(beta) + 16*l3*q1^2*x2^2*x3*sin(beta) + 12*l3*q2^2*x2*x3^2*sin(beta) - 8*l3*q2^2*x2^2*x3*sin(beta) - 8*l3*q3^2*x2^2*x3*sin(beta) - 4*l3*q1^2*x2*y3^2*sin(beta) + 4*l3*q2^2*x2*y3^2*sin(beta) + 8*l3*x2^2*x3*y3^2*sin(beta) - 32*l2*l3^2*x2*x3*y3*cos(2*beta) - 16*l2*l3*q1^2*x2*y3*cos(beta) + 16*l2*l3*q1^2*x3*y3*cos(beta) + 16*l2*l3*q2^2*x2*y3*cos(beta) - 16*l2*l3*q2^2*x3*y3*cos(beta) + 16*l2*l3*q3^2*x2*y3*cos(beta) - 16*l2*l3*x2*x3^2*y3*cos(beta) + 32*l2*l3*x2^2*x3*y3*cos(beta) + 40*l2^2*l3*x2*x3*y3*cos(beta) + 8*l3*q1^2*x2*x3*y3*cos(beta) - 8*l3*q2^2*x2*x3*y3*cos(beta) + 16*l2*l3*q1^2*x2*x3*sin(beta) - 16*l2*l3*q2^2*x2*x3*sin(beta) - 16*l2*l3*q3^2*x2*x3*sin(beta) + 16*l2*l3*x2*x3*y3^2*sin(beta)*

*p4=3*l2^2*x3^4 - 2*l2^3*x3^3 + 3*l2^4*x3^2 + 3*l3^2*x2^4 + 3*l3^4*x2^2 + 3*l2^2*y3^4 + 3*l2^4*y3^2 + 3*q1^4*x2^2 + 3*q1^4*x3^2 + 3*q2^4*x3^2 + 3*q3^4*x2^2 + 3*q1^4*y3^2 + 3*q2^4*y3^2 + 3*x2^2*x3^4 - 6*x2^3*x3^3 + 3*x2^4*x3^2 + 3*x2^2*y3^4 + 3*x2^4*y3^2 + 3*l2^2*l3^4 + 3*l2^4*l3^2 + 3*l2^2*q1^4 + 3*l3^2*q1^4 + 3*l2^2*q3^4 + 3*l3^2*q2^4 + 2*l2*l3^4*x2 + 2*l2*q1^4*x2 - 2*l2*q1^4*x3 + 2*l2*q3^4*x2 + 2*l2*x2*x3^4 + 2*l2*x2*y3^4 - 6*q1^4*x2*x3 + 4*l2*l3^2*x2^3 + 4*l2^3*l3^2*x2 - 4*l2^3*l3^2*x3 - 2*l2*q1^2*x3^3 - 2*l2^3*q1^2*x3 + 2*l2*q2^2*x3^3 + 2*l2^3*q3^2*x3 - 6*l2*x2^2*x3^3 + 4*l2*x2^3*x3^2 - 2*l2^2*x2*x3^3 + 4*l2^3*x2*x3^2 - 12*l3^2*x2^3*x3 + 4*l2*x2^3*y3^2 + 4*l2^3*x2*y3^2 - 2*l2^3*x3*y3^2 - 6*q1^2*x2*x3^3 - 6*q1^2*x2^3*x3 + 6*q2^2*x2*x3^3 + 6*q3^2*x2^3*x3 - 6*x2^3*x3*y3^2 + 6*l2^2*l3^2*q1^2 - 6*l2^2*l3^2*q2^2 - 6*l2^2*l3^2*q3^2 - 6*l2^3*l3^3*cos(beta) - 6*l2^2*q1^2*q3^2 - 6*l3^2*q1^2*q2^2 + 2*l2^2*l3^2*x2^2 + 12*l2^2*l3^2*x3^2 + 12*l2^2*l3^2*y3^2 - 4*l2^2*q1^2*x3^2 + 6*l3^2*q1^2*x2^2 - 6*l2^2*q2^2*x3^2 - 6*l3^2*q2^2*x2^2 - 6*l2^2*q3^2*x3^2 - 6*l3^2*q3^2*x2^2 + 16*l2^2*q1^2*y3^2 - 6*l2^2*q2^2*y3^2 - 6*l2^2*q3^2*y3^2 - 2*l3^3*x2^3*cos(beta) - 6*q1^2*q2^2*x3^2 - 6*q1^2*q3^2*x2^2 - 6*q1^2*q2^2*y3^2 + 2*l2^2*x2^2*x3^2 + 12*l3^2*x2^2*x3^2 + 2*l2^2*x2^2*y3^2 + 6*l2^2*x3^2*y3^2 + 12*l3^2*x2^2*y3^2 + 12*q1^2*x2^2*x3^2 - 6*q2^2*x2^2*x3^2 - 6*q3^2*x2^2*x3^2 - 6*q2^2*x2^2*y3^2 - 6*q3^2*x2^2*y3^2 + 6*x2^2*x3^2*y3^2 + 2*l3*x2^4*y3*sin(beta) - 10*l2^2*l3^2*x3^2*cos(2*beta) + 10*l2^2*l3^2*y3^2*cos(2*beta) - 10*l3^2*q1^2*x2^2*cos(2*beta) - 10*l3^2*x2^2*x3^2*cos(2*beta) + 10*l3^2*x2^2*y3^2*cos(2*beta) - 6*l2*l3^3*q1^2*cos(beta) - 6*l2^3*l3*q1^2*cos(beta) + 6*l2*l3^3*q2^2*cos(beta) + 6*l2^3*l3*q3^2*cos(beta) - 2*l2*l3^3*x2^2*cos(beta) - 6*l2^2*l3^3*x2*cos(beta) + 4*l2^2*l3*x3^3*cos(beta) + 4*l2^2*l3^3*x3*cos(beta) - 2*l2^3*l3*x3^2*cos(beta) - 22*l2^3*l3*y3^2*cos(beta) - 2*l3*q1^2*x2^3*cos(beta) - 2*l3^3*q1^2*x2*cos(beta) + 2*l3^3*q2^2*x2*cos(beta) + 2*l3*q3^2*x2^3*cos(beta) + 4*l3^3*x2^2*x3*cos(beta) - 6*l3*x2^3*x3^2*cos(beta) + 4*l3^3*x2^2*x3*cos(beta) - 2*l3*x2^3*y3^2*cos(beta) + 4*l2^2*l3*y3^3*sin(beta) + 4*l2^2*l3^3*y3*sin(beta) + 4*l3*x2^2*y3^3*sin(beta) + 4*l3^3*x2^2*y3*sin(beta) - 2*l2^3*l3^2*x3*cos(2*beta) + 10*l3^2*x2^3*x3*cos(2*beta) - 2*l2^3*l3^2*y3*sin(2*beta) + 10*l3^2*x2^3*y3*sin(2*beta) - 6*l2*l3*q1^4*cos(beta) + 4*l2*l3^2*q1^2*x2 - 4*l2*l3^2*q1^2*x3 - 4*l2*l3^2*q2^2*x2 + 4*l2*l3^2*q2^2*x3 - 4*l2*l3^2*q3^2*x2 + 2*l2^4*l3*x3*cos(beta) + 2*l2*q1^2*q2^2*x3 - 4*l2*q1^2*q3^2*x2 + 2*l2*q1^2*q3^2*x3 - 2*l2*q2^2*q3^2*x3 + 8*l2*l3^2*x2*x3^2 - 12*l2*l3^2*x2^2*x3 - 4*l2^2*l3^2*x2*x3 + 8*l2*l3^2*x2*y3^2 - 2*l3*q1^4*x2*cos(beta) + 2*l3*q1^4*x3*cos(beta) + 2*l3*q2^4*x3*cos(beta) + 8*l2*q1^2*x2*x3^2 - 6*l2*q1^2*x2^2*x3 - 2*l2^2*q1^2*x2*x3 - 4*l2*q2^2*x2*x3^2 - 12*l3^2*q1^2*x2*x3 - 4*l2*q3^2*x2*x3^2 + 6*l2*q3^2*x2^2*x3 + 2*l2^2*q3^2*x2*x3 + 12*l3^2*q2^2*x2*x3 - 2*l2*q1^2*x3*y3^2 - 4*l2*q2^2*x2*y3^2 + 2*l2*q3^2*x2*y3^2 - 4*l2*q3^2*x2^2*y3^2 + 2*l3*x2^4*x3*cos(beta) + 6*q1^2*q2^2*x2*x3 + 6*q1^2*q3^2*x2*x3 - 6*q2^2*q3^2*x2*x3 + 4*l2*x2*x3^2*y3^2 - 6*l2*x2^2*x3^2*y3^2 + 6*q2^2*x2*x3^2*y3^2 + 2*l3*q1^4*y3*sin(beta) + 2*l3*q2^4*y3*sin(beta) + 6*l2^2*l3^2*q1^2*cos(2*beta) + 4*l2*l3^2*q1^2*x2*cos(2*beta) - 2*l2*l3^2*q1^2*x3*cos(2*beta) + 2*l2*l3^2*q2^2*x3*cos(2*beta) - 28*l2*l3^2*x2*x3^2*cos(2*beta) + 26*l2*l3^2*x2^2*x3*cos(2*beta) + 14*l2^2*l3^2*x2*x3*cos(2*beta) + 28*l2*l3^2*x2*y3^2*cos(2*beta) + 10*l3^2*q1^2*x2*x3*cos(2*beta) - 10*l3^2*q2^2*x2*x3*cos(2*beta) - 2*l2*l3^2*q1^2*y3*sin(2*beta) + 2*l2*l3^2*q2^2*y3*sin(2*beta) + 26*l2*l3^2*x2^2*y3*sin(2*beta) + 14*l2^2*l3^2*x2*y3*sin(2*beta) - 20*l2^2*l3^2*x3*y3*sin(2*beta) + 10*l3^2*q1^2*x2*y3*sin(2*beta) - 10*l3^2*q2^2*x2*y3*sin(2*beta) - 20*l3^2*x2^2*x3*y3*sin(2*beta) - 8*l2*l3*x2*x3^3*cos(beta) - 8*l2*l3*x2^3*x3*cos(beta) - 8*l2*l3^3*x2*x3*cos(beta) - 8*l2^3*l3*x2*x3*cos(beta) - 8*l2*l3*x2*y3^3*sin(beta) - 8*l2*l3^3*x2*y3*sin(beta) + 20*l2^3*l3*x3*y3*sin(beta) - 4*l3*x2^3*x3*y3*sin(beta) + 6*l2*l3*q1^2*q2^2*cos(beta) + 6*l2*l3*q1^2*q3^2*cos(beta) - 6*l2*l3*q2^2*q3^2*cos(beta) - 2*l2*l3*q1^2*x2^2*cos(beta) - 6*l2^2*l3*q1^2*x2*cos(beta) - 2*l2*l3*q1^2*x3^2*cos(beta) + 8*l2^2*l3*q1^2*x3*cos(beta) + 2*l2*l3*q2^2*x3^2*cos(beta) - 4*l2^2*l3*q2^2*x3*cos(beta) + 6*l2^2*l3*q3^2*x2*cos(beta) - 4*l2^2*l3*q3^2*x3*cos(beta) - 22*l2*l3*q1^2*y3^2*cos(beta) + 22*l2*l3*q2^2*y3^2*cos(beta) + 2*l3*q1^2*q2^2*x2*cos(beta) - 4*l3*q1^2*q2^2*x3*cos(beta) + 2*l3*q1^2*q3^2*x2*cos(beta) - 2*l3*q2^2*q3^2*x2*cos(beta) + 10*l2*l3*x2^2*x3^2*cos(beta) + 14*l2^2*l3*x2*x3^2*cos(beta) - 20*l2^2*l3*x2^2*x3*cos(beta) - 18*l2*l3*x2^2*y3^2*cos(beta) - 38*l2^2*l3*x2*y3^2*cos(beta) + 4*l2^2*l3*x3*y3^2*cos(beta) - 6*l3*q1^2*x2*x3^2*cos(beta) + 8*l3*q1^2*x2^2*x3*cos(beta) + 6*l3*q2^2*x2*x3^2*cos(beta) - 4*l3*q2^2*x2^2*x3*cos(beta) - 4*l3*q3^2*x2^2*x3*cos(beta) - 2*l3*q1^2*x2*y3^2*cos(beta) - 4*l2^2*l3*x2^2*y3*sin(beta) + 4*l3*q1^2*q2^2*y3*sin(beta) - 20*l2^2*l3*x2^2*y3*sin(beta) + 4*l2^2*l3*x3^2*y3*sin(beta) - 4*l3*q2^2*x2^2*y3*sin(beta) - 4*l3*q3^2*x2^2*y3*sin(beta) + 4*l3*x2^2*x3*y3^2*cos(beta) - 56*l2*l3^2*x2*x3*y3*sin(2*beta) + 16*l2*l3*q1^2*x2*x3*cos(beta) + 8*l2*l3*q2^2*x2*x3*cos(beta) + 8*l2*l3*q3^2*x2*x3*cos(beta) - 8*l2*l3*x2*x3*y3^2*cos(beta) - 32*l2*l3*q1^2*x2*y3*sin(beta) + 20*l2*l3*q1^2*x3*y3*sin(beta) + 8*l2*l3*q2^2*x2*y3*sin(beta) - 20*l2*l3*q2^2*x3*y3*sin(beta) + 8*l2*l3*q3^2*x2*y3*sin(beta) - 8*l2*l3*x2*x3^2*y3*sin(beta) + 28*l2*l3*x2^2*x3*y3*sin(beta) + 52*l2^2*l3*x2*x3*y3*sin(beta) - 4*l3*q1^2*x2*x3*y3*sin(beta) + 4*l3*q2^2*x2*x3*y3*sin(beta)*

*p3=8*l2^3*y3^3 + 8*l2*q1^4*y3 + 16*l2^3*l3^2*y3 + 8*l2*q1^2*y3^3 + 8*l2^3*q1^2*y3 - 8*l2*q2^2*y3^3 - 8*l2^3*q3^2*y3 + 8*l2*x2^2*y3^3 + 8*l2^3*x3^2*y3 - 8*l3^3*x2^3*sin(beta) - 16*l2^2*l3*y3^3*cos(beta) - 16*l2^2*l3^3*y3*cos(beta) - 16*l3*x2^2*y3^3*cos(beta) - 16*l3^3*x2^2*y3*cos(beta) - 8*l2^2*l3^3*x2*sin(beta) + 16*l2^2*l3*x3^3*sin(beta) + 16*l2^2*l3^3*x3*sin(beta) - 8*l3*q1^2*x2^3*sin(beta) - 8*l3^3*q1^2*x2*sin(beta) + 8*l3^3*q2^2*x2*sin(beta) + 8*l3*q3^2*x2^3*sin(beta) + 16*l3*x2^2*x3^3*sin(beta) - 24*l3*x2^3*x3^2*sin(beta) + 16*l3^3*x2^2*x3*sin(beta) - 8*l3*x2^3*y3^2*sin(beta) + 8*l2^3*l3^2*y3*cos(2*beta) - 8*l2^3*l3^2*x3*sin(2*beta) + 16*l2*l3^2*q1^2*y3 - 16*l2*l3^2*q2^2*y3 - 8*l2^4*l3*y3*cos(beta) - 8*l2*q1^2*q2^2*y3 - 8*l2*q1^2*q3^2*y3 + 8*l2*q2^2*q3^2*y3 + 16*l2*l3^2*x2^2*y3 - 8*l3*q1^4*y3*cos(beta) - 8*l3*q2^4*y3*cos(beta) + 8*l2*q1^2*x2^2*y3 + 8*l2*q1^2*x3^2*y3 - 8*l2*q2^2*x3^2*y3 - 8*l2*q3^2*x2^2*y3 - 8*l3*x2^4*y3*cos(beta) + 8*l2^4*l3*x3*sin(beta) + 8*l2*x2^2*x3^2*y3 - 8*l3*q1^4*x2*sin(beta) + 8*l3*q1^4*x3*sin(beta) + 8*l3*q2^4*x3*sin(beta) + 8*l3*x2^4*x3*sin(beta) - 32*l2*q1^2*x2*x3*y3 + 8*l2*l3^2*q1^2*y3*cos(2*beta) - 8*l2*l3^2*q2^2*y3*cos(2*beta) - 24*l2*l3^2*x2^2*y3*cos(2*beta) + 16*l2*l3^2*q1^2*x2*sin(2*beta) - 8*l2*l3^2*q1^2*x3*sin(2*beta) + 8*l2*l3^2*q2^2*x3*sin(2*beta) - 32*l2*l3^2*x2*x3^2*sin(2*beta) + 24*l2*l3^2*x2^2*x3*sin(2*beta) + 32*l2*l3^2*x2*y3^2*sin(2*beta) + 16*l3*x2^3*x3*y3*cos(beta) - 32*l2^2*l3*q1^2*y3*cos(beta) + 16*l2^2*l3*q2^2*y3*cos(beta) + 16*l2^2*l3*q3^2*y3*cos(beta) + 16*l3*q1^2*q2^2*y3*cos(beta) + 16*l2^2*l3*x2^2*y3*cos(beta) - 16*l2^2*l3*x3^2*y3*cos(beta) + 16*l3*q2^2*x2^2*y3*cos(beta) + 16*l3*q3^2*x2^2*y3*cos(beta) - 8*l2^2*l3*q1^2*x2*sin(beta) - 16*l2^2*l3*q2^2*x3*sin(beta) + 8*l2^2*l3*q3^2*x2*sin(beta) - 16*l2^2*l3*q3^2*x3*sin(beta) - 16*l3*x2^2*x3^2*y3*cos(beta) + 8*l3*q1^2*q2^2*x2*sin(beta) - 16*l3*q1^2*q2^2*x3*sin(beta) + 8*l3*q1^2*q3^2*x2*sin(beta) - 8*l3*q2^2*q3^2*x2*sin(beta) + 8*l2^2*l3*x2*x3^2*sin(beta) - 16*l2^2*l3*x2^2*x3*sin(beta) - 40*l2^2*l3*x2*y3^2*sin(beta) + 16*l2^2*l3*x3*y3^2*sin(beta) - 24*l3*q1^2*x2*x3^2*sin(beta) + 32*l3*q1^2*x2^2*x3*sin(beta) + 24*l3*q2^2*x2*x3^2*sin(beta) - 16*l3*q2^2*x2^2*x3*sin(beta) - 16*l3*q3^2*x2^2*x3*sin(beta) - 8*l3*q1^2*x2*y3^2*sin(beta) + 8*l3*q2^2*x2*y3^2*sin(beta) + 16*l3*x2^2*x3*y3^2*sin(beta) + 64*l2*l3^2*x2*x3*y3*cos(2*beta) - 48*l2^2*l3*x2*x3*y3*cos(beta) + 16*l3*q1^2*x2*x3*y3*cos(beta) - 16*l3*q2^2*x2*x3*y3*cos(beta)*

*p1=3*l2^2*x3^4 + 2*l2^3*x3^3 + 3*l2^4*x3^2 + 3*l3^2*x2^4 + 3*l3^4*x2^2 + 3*l2^2*y3^4 + 3*l2^4*y3^2 + 3*q1^4*x2^2 + 3*q1^4*x3^2 + 3*q2^4*x3^2 + 3*q3^4*x2^2 + 3*q1^4*y3^2 + 3*q2^4*y3^2 + 3*x2^2*x3^4 - 6*x2^3*x3^3 + 3*x2^4*x3^2 + 3*x2^2*y3^4 + 3*x2^4*y3^2 + 3*l2^2*l3^4 + 3*l2^4*l3^2 + 3*l2^2*q1^4 + 3*l3^2*q1^4 + 3*l2^2*q3^4 + 3*l3^2*q2^4 - 2*l2*l3^4*x2 - 2*l2*q1^4*x2 + 2*l2*q1^4*x3 - 2*l2*q3^4*x2 - 2*l2*x2*x3^4 - 2*l2*x2*y3^4 - 6*q1^4*x2*x3 - 4*l2^3*l3^2*x2^3 - 4*l2^3*l3^2*x2 + 4*l2^3*l3^2*x3 + 2*l2*q1^2*x3^3 + 2*l2^3*q1^2*x3 - 2*l2*q2^2*x3^3 - 2*l2^3*q3^2*x3 + 6*l2*x2^2*x3^3 - 4*l2*x2^3*x3^2 - 2*l2^2*x2*x3^3 - 4*l2^3*x2*x3^2 - 12*l3^2*x2^3*x3 - 4*l2*x2^3*y3^2 - 4*l2^3*x2*y3^2 + 2*l2^3*x3*y3^2 - 6*q1^2*x2*x3^3 - 6*q1^2*x2^3*x3 + 6*q2^2*x2*x3^3 + 6*q3^2*x2^3*x3 - 6*x2^3*x3*y3^2 + 6*l2^2*l3^2*q1^2 - 6*l2^2*l3^2*q2^2 - 6*l2^2*l3^2*q3^2 - 6*l2^3*l3^3*cos(beta) - 6*l2^2*q1^2*q3^2 - 6*l3^2*q1^2*q2^2 + 2*l2^2*l3^2*x2^2 + 12*l2^2*l3^2*x3^2 + 12*l2^2*l3^2*y3^2 - 4*l2^2*q1^2*x3^2 + 6*l3^2*q1^2*x2^2 - 6*l2^2*q2^2*x3^2 - 6*l3^2*q2^2*x2^2 - 6*l2^2*q3^2*x3^2 - 6*l3^2*q3^2*x2^2 + 16*l2^2*q1^2*y3^2 - 6*l2^2*q2^2*y3^2 - 6*l2^2*q3^2*y3^2 + 2*l3^3*x2^3*cos(beta) - 6*q1^2*q2^2*x3^2 - 6*q1^2*q3^2*x2^2 - 6*q1^2*q2^2*y3^2 + 2*l2^2*x2^2*x3^2 + 12*l3^2*x2^2*x3^2 + 2*l2^2*x2^2*y3^2 + 6*l2^2*x3^2*y3^2 + 12*l3^2*x2^2*y3^2 + 12*q1^2*x2^2*x3^2 - 6*q2^2*x2^2*x3^2 - 6*q3^2*x2^2*x3^2 - 6*q2^2*x2^2*y3^2 - 6*q3^2*x2^2*y3^2 + 6*x2^2*x3^2*y3^2 - 2*l3*x2^4*y3*sin(beta) - 10*l2^2*l3^2*x3^2*cos(2*beta) + 10*l2^2*l3^2*y3^2*cos(2*beta) - 10*l3^2*q1^2*x2^2*cos(2*beta) - 10*l3^2*x2^2*x3^2*cos(2*beta) + 10*l3^2*x2^2*y3^2*cos(2*beta) - 6*l2*l3^3*q1^2*cos(beta) - 6*l2^3*l3*q1^2*cos(beta) + 6*l2*l3^3*q2^2*cos(beta) + 6*l2^3*l3*x2^2*cos(beta) - 6*l2^2*l3^3*x2*cos(beta) + 4*l2^2*l3*x3^3*cos(beta) - 4*l2^2*l3^3*x3*cos(beta) - 2*l2^3*l3*x3^2*cos(beta) - 22*l2^3*l3*y3^2*cos(beta) + 2*l3*q1^2*x2^3*cos(beta) + 2*l3^3*q1^2*x2*cos(beta) - 2*l3^3*q2^2*x2*cos(beta) - 2*l3*q3^2*x2^3*cos(beta) - 4*l3*x2^2*x3^3*cos(beta) + 6*l3*x2^3*x3^2*cos(beta) - 4*l3^3*x2^2*x3*cos(beta) + 2*l3*x2^3*y3^2*cos(beta) - 4*l2^2*l3*y3^3*sin(beta) - 4*l2^2*l3^3*y3*sin(beta) - 4*l3*x2^2*y3^3*sin(beta) - 4*l3^3*x2^2*y3*sin(beta) + 2*l2^3*l3^2*x3*cos(2*beta) + 10*l3^2*x2^3*x3*cos(2*beta) + 2*l2^3*l3^2*y3*sin(2*beta) + 10*l3^2*x2^3*y3*sin(2*beta) - 6*l2*l3*q1^4*cos(beta) - 4*l2*l3^2*q1^2*x2 + 4*l2*l3^2*q1^2*x3 + 4*l2*l3^2*q2^2*x2 - 4*l2*l3^2*q2^2*x3 + 4*l2*l3^2*q3^2*x2 - 2*l2^4*l3*x3*cos(beta) - 2*l2*q1^2*q2^2*x3 + 4*l2*q1^2*q3^2*x2 - 2*l2*q1^2*q3^2*x3 + 2*l2*q2^2*q3^2*x3 - 8*l2*l3^2*x2*x3^2 + 12*l2*l3^2*x2^2*x3 - 4*l2^2*l3^2*x2*x3 - 8*l2*l3^2*x2*y3^2 + 2*l3*q1^4*x2*cos(beta) - 2*l3*q2^4*x3*cos(beta) - 8*l2*q1^2*x2*x3^2 + 6*l2*q1^2*x2^2*x3 - 2*l2^2*q1^2*x2*x3 + 4*l2*q2^2*x2*x3^2 - 12*l3^2*q1^2*x2*x3 + 4*l2*q3^2*x2*x3^2 - 6*l2*q3^2*x2^2*x3 + 2*l2^2*q3^2*x2*x3 + 12*l3^2*q2^2*x2*x3 + 2*l2*q1^2*x3*y3^2 + 4*l2*q2^2*x2*y3^2 - 2*l2*q2^2*x3*y3^2 + 4*l2*q3^2*x2*y3^2 - 2*l3*x2^4*x3*cos(beta) + 6*q1^2*q2^2*x2*x3 + 6*q1^2*q3^2*x2*x3 - 6*q2^2*q3^2*x2*x3 - 4*l2*x2*x3^2*y3^2 + 6*l2*x2^2*x3^2*y3^2 - 2*l2*l3^2*q1^2*x2*y3*sin(2*beta) - 2*l2*l3^2*q2^2*y3*sin(2*beta) - 26*l2*l3^2*x2^2*y3*sin(2*beta) + 14*l2^2*l3^2*x2*y3*sin(2*beta) - 20*l2^2*l3^2*x3*y3*sin(2*beta) + 10*l3^2*q1^2*x2*y3*sin(2*beta) - 10*l3^2*q2^2*x2*y3*sin(2*beta) - 20*l2^2*l3^2*x2^2*y3*sin(2*beta) - 8*l2*l3^2*x2^3*x3*cos(beta) - 8*l2*l3^3*x2^2*x3*cos(beta) - 8*l2^3*l3^2*x2*x3*cos(beta) - 8*l2*l3^3*x2*y3^3*sin(beta) - 8*l2*l3^3*x2^3*y3*sin(beta) - 8*l2^3*l3^2*x2*y3*sin(beta) + 20*l2^3*l3^2*x3*y3*sin(beta) + 4*l3*x2^3*x3*y3*sin(beta) + 6*l2*l3^2*q1^2*q2^2*cos(beta) + 6*l2*l3^2*q1^2*q3^2*cos(beta) - 6*l2*l3^2*q2^2*q3^2*cos(beta) - 2*l2*l3^2*x2^2*cos(beta) + 6*l2^2*l3^2*q1^2*x2*cos(beta) - 2*l2*l3^2*q1^2*x3^2*cos(beta) - 8*l2^2*l3^2*q1^2*x3*cos(beta) + 2*l2*l3^2*q2^2*x3^2*cos(beta) - 8*l2^2*l3^2*x2^2*x3*cos(beta) + 2*l2*l3^2*q3^2*x2^2*cos(beta) + 4*l2^2*l3^2*q2^2*x3*cos(beta) - 6*l2^2*l3^2*q3^2*x2*cos(beta) + 4*l2^2*l3^2*q3^2*x3*cos(beta) - 22*l2*l3^2*q1^2*y3^2*cos(beta) + 22*l2*l3^2*q2^2*y3^2*cos(beta) - 2*l3*q1^2*q2^2*x2*cos(beta) + 4*l3*q1^2*q2^2*x3*cos(beta) - 2*l3*q1^2*q3^2*x2*cos(beta) + 2*l3*q2^2*q3^2*x2*cos(beta) + 10*l2*l3*x2^2*x3^2*cos(beta) - 14*l2^2*l3*x2^2*x3^2*cos(beta) + 20*l2^2*l3*x2^2*x3*cos(beta) - 18*l2*l3*x2^2*y3^2*cos(beta) + 38*l2^2*l3*x2*y3^2*cos(beta) - 4*l2^2*l3*x3*y3^2*cos(beta) + 6*l3*q1^2*x2*x3^2*cos(beta) - 8*l3*q1^2*x2^2*x3*cos(beta) - 6*l3*q2^2*x2*x3^2*cos(beta) + 4*l3*q2^2*x2^2*x3*cos(beta) + 4*l3*q3^2*x2^2*x3*cos(beta) + 2*l3*q1^2*x2*y3^2*cos(beta) - 2*l3*q2^2*x2*y3^2*cos(beta) + 4*l2^2*l3*q2^2*y3*sin(beta) + 4*l2^2*l3*q3^2*y3*sin(beta) - 4*l3*x2^2*x3*y3^2*cos(beta) + 4*l3*q1^2*q2^2*y3*sin(beta) + 20*l2^2*l3*x2^2*y3*sin(beta) - 4*l2^2*l3*x3^2*y3*sin(beta) + 4*l3*q2^2*x2^2*y3*sin(beta) + 4*l3*q3^2*x2^2*y3*sin(beta) - 4*l3*x2^2*x3^2*y3*sin(beta) + 56*l2*l3^2*x2*x3*y3*sin(2*beta) + 16*l2*l3*q1^2*x2*x3*cos(beta) + 8*l2*l3*q2^2*x2*x3*cos(beta) + 8*l2*l3*q3^2*x2*x3*cos(beta) - 8*l2*l3*x2*x3*y3^2*cos(beta) - 32*l2*l3*q1^2*x2*y3*sin(beta) + 20*l2*l3*q1^2*x3*y3*sin(beta) + 8*l2*l3*q2^2*x2*y3*sin(beta) - 20*l2*l3*q2^2*x3*y3*sin(beta) + 8*l2*l3*q3^2*x2*y3*sin(beta) + 28*l2*l3*x2^2*x3*y3*sin(beta) - 52*l2^2*l3*x2*x3*y3*sin(beta) + 4*l3*q1^2*x2*x3*y3*sin(beta) - 4*l3*q2^2*x2*x3*y3*sin(beta)*

*b(6)=4*l2^3*y3^3 + 4*l2*q1^4*y3 + 8*l2^3*l3^2*y3 + 4*l2*q1^2*y3^3 + 4*l2^3*q1^2*y3 - 4*l2*q2^2*y3^3 - 4*l2^3*q3^2*y3 + 4*l2*x2^2*y3^3 - 8*l2^2*x2*y3^3 + 4*l2^3*x3^2*y3 - 4*l3^3*x2^3*sin(beta) - 8*l2^2*l3^2*x3^2*sin(2*beta) + 8*l2^2*l3^2*y3^2*sin(2*beta) - 8*l3^2*q1^2*x2^2*sin(2*beta) - 8*l3^2*x2^2*x3^2*sin(2*beta) + 8*l3^2*x2^2*y3^2*sin(2*beta) - 8*l2^2*l3^3*y3*cos(beta) - 8*l2^2*l3^3*y3^3*cos(beta) - 8*l3^3*x2^2*y3^3*cos(beta) - 8*l3^3*x2^2*y3*cos(beta) + 8*l2^2*l3^3*x3*sin(beta) + 8*l2*l3^3*x2^2*x3*sin(beta) - 8*l3^3*q1^2*x2*x3*sin(beta) + 8*l3^3*q2^2*x2*x3*sin(beta) + 8*l3*q3^2*x2^3*x3*sin(beta) - 16*l3*x2^2*x3^3*sin(beta) + 8*l3^3*x2^2*x3*sin(beta) - 8*l3*x2^3*y3^2*sin(beta) + 8*l2^3*l3^2*y3*cos(2*beta) - 8*l2^3*l3^2*x3*sin(2*beta) + 16*l2*l3^2*q1^2*y3 - 16*l2*l3^2*q2^2*y3 - 8*l2^4*l3*y3*cos(beta) - 8*l2*q1^2*q2^2*y3 - 8*l2*q1^2*q3^2*y3 + 8*l2*q2^2*q3^2*y3 + 16*l2*l3^2*x2^2*y3 - 8*l3*q1^4*y3*cos(beta) - 8*l3*q2^4*y3*cos(beta) + 8*l2*q1^2*x2^2*y3 + 8*l2*q1^2*x3^2*y3 - 8*l2*q2^2*x3^2*y3 - 8*l2*q3^2*x2^2*y3 - 8*l3*x2^4*y3*cos(beta) + 8*l2^4*l3*x3*sin(beta) + 8*l2*x2^2*x3^2*y3 - 8*l3*q1^4*x2*sin(beta) + 8*l3*q1^4*x3*sin(beta) + 8*l3*q2^4*x3*sin(beta) + 8*l3*x2^4*x3*sin(beta) - 32*l2*q1^2*x2*x3*y3 + 8*l2*l3^2*q1^2*y3*cos(2*beta) - 8*l2*l3^2*q2^2*y3*cos(2*beta) - 24*l2*l3^2*x2^2*y3*cos(2*beta) + 16*l2*l3^2*q1^2*x2*sin(2*beta) - 8*l2*l3^2*q1^2*x3*sin(2*beta) + 8*l2*l3^2*q2^2*x3*sin(2*beta) - 32*l2*l3^2*x2*x3^2*sin(2*beta) + 24*l2*l3^2*x2^2*x3*sin(2*beta) + 32*l2*l3^2*x2*y3^2*sin(2*beta) + 16*l3*x2^3*x3*y3*cos(beta) - 32*l2^2*l3*q1^2*y3*cos(beta) + 16*l2^2*l3*q2^2*y3*cos(beta) + 16*l2^2*l3*q3^2*y3*cos(beta) + 16*l3*q1^2*q2^2*y3*cos(beta) + 16*l2^2*l3*x2^2*y3*cos(beta) - 16*l2^2*l3*x3^2*y3*cos(beta) + 16*l3*q2^2*x2^2*y3*cos(beta) + 16*l3*q3^2*x2^2*y3*cos(beta) - 8*l2^2*l3*q1^2*x2*sin(beta) - 16*l2^2*l3*q2^2*x3*sin(beta) + 8*l2^2*l3*q3^2*x2*sin(beta) - 16*l2^2*l3*q3^2*x3*sin(beta) - 16*l3*x2^2*x3^2*y3*cos(beta) + 8*l3*q1^2*q2^2*x2*sin(beta) - 16*l3*q1^2*q2^2*x3*sin(beta) + 8*l3*q1^2*q3^2*x2*sin(beta) - 8*l3*q2^2*q3^2*x2*sin(beta) + 8*l2^2*l3*x2*x3^2*sin(beta) - 16*l2^2*l3*x2^2*x3*sin(beta) - 40*l2^2*l3*x2*y3^2*sin(beta) + 16*l2^2*l3*x3*y3^2*sin(beta) - 24*l3*q1^2*x2*x3^2*sin(beta) + 32*l3*q1^2*x2^2*x3*sin(beta) + 24*l3*q2^2*x2*x3^2*sin(beta) - 16*l3*q2^2*x2^2*x3*sin(beta) - 16*l3*q3^2*x2^2*x3*sin(beta) - 8*l3*q1^2*x2*y3^2*sin(beta) + 8*l3*q2^2*x2*y3^2*sin(beta) + 16*l3*x2^2*x3*y3^2*sin(beta) + 64*l2*l3^2*x2*x3*y3*cos(2*beta) - 48*l2^2*l3*x2*x3*y3*cos(beta) + 16*l3*q1^2*x2*x3*y3*cos(beta) - 16*l3*q2^2*x2*x3*y3*cos(beta)*

*4*l3*q3^2*x2^3*sin(beta) + 8*l3*x2^2*x3^3*sin(beta) - 12*l3*x2^3*x3^2*sin(beta) + 8*l3^3*x2^2*x3*sin(beta) - 4*l3*x2^3*y3^2*sin(beta) + 4*l2^3*l3^2*y3*cos(2*beta) - 8*l3^2*x2^3*y3*cos(2*beta) - 4*l2^3*l3^2*x3*sin(2*beta) + 8*l3^2*x2^3*x3*sin(2*beta) + 8*l2*l3^2*q1^2*y3 - 8*l2*l3^2*q2^2*y3 - 4*l2^4*l3*y3*cos(beta) - 4*l2*q1^2*q2^2*y3 - 4*l2*q1^2*q3^2*y3 + 4*l2*q2^2*q3^2*y3 + 8*l2*l3^2*x2^2*y3 - 16*l2^2*l3^2*x2*y3 - 4*l3*q1^4*y3*cos(beta) - 4*l3*q2^4*y3*cos(beta) + 4*l2*q1^2*x2^2*y3 - 8*l2^2*q1^2*x2*y3 + 4*l2*q1^2*x3^2*y3 + 16*l2^2*q1^2*x3*y3 - 4*l2*q2^2*x3^2*y3 - 4*l2*q3^2*x2^2*y3 + 8*l2^2*q3^2*x2*y3 - 4*l3*x2^4*y3*cos(beta) + 4*l2^4*l3*x3*sin(beta) + 4*l2*x2^2*x3^2*y3 - 8*l2^2*x2*x3^2*y3 - 4*l3*q1^4*x2*sin(beta) + 4*l3*q1^4*x3*sin(beta) + 4*l3*q2^4*x3*sin(beta) + 4*l3*x2^4*x3*sin(beta) - 16*l2*q1^2*x2*x3*y3 + 4*l2*l3^2*q1^2*y3*cos(2*beta) - 4*l2*l3^2*q2^2*y3*cos(2*beta) + 20*l2*l3^2*x2^2*y3*cos(2*beta) - 16*l2^2*l3^2*x2*y3*cos(2*beta) + 16*l2^2*l3^2*x3*y3*cos(2*beta) - 8*l3^2*q1^2*x2*y3*cos(2*beta) + 8*l3^2*q2^2*x2*y3*cos(2*beta) + 8*l2*l3^2*q1^2*x2*sin(2*beta) - 4*l2*l3^2*q1^2*x3*sin(2*beta) + 4*l2*l3^2*q2^2*x3*sin(2*beta) + 16*l3^2*x2^2*x3*y3*cos(2*beta) + 16*l2*l3^2*x2*x3^2*sin(2*beta) - 20*l2*l3^2*x2^2*x3*sin(2*beta) + 16*l2^2*l3^2*x2*x3*sin(2*beta) - 16*l2*l3^2*x2*y3^2*sin(2*beta) + 8*l3^2*q1^2*x2*x3*sin(2*beta) - 8*l3^2*q2^2*x2*x3*sin(2*beta) + 16*l2*l3*x2*y3^3*cos(beta) + 16*l2*l3*x2^3*y3*cos(beta) + 16*l2*l3^3*x2*y3*cos(beta) + 16*l2^3*l3*x2*y3*cos(beta) - 16*l2^3*l3*x2*x3*sin(beta) - 16*l2*l3*x2*x3^3*sin(beta) - 16*l2*l3^3*x2*x3*sin(beta) - 16*l2^3*l3*x2*x3*sin(beta) - 16*l2^2*l3*q1^2*y3*cos(beta) + 8*l2^2*l3*q2^2*y3*cos(beta) + 8*l2^2*l3*q3^2*y3*cos(beta) + 8*l3*q1^2*q2^2*y3*cos(beta) - 24*l2^2*l3*x2^2*y3*cos(beta) - 8*l2^2*l3*x3^2*y3*cos(beta) + 8*l3*q2^2*x2^2*y3*cos(beta) + 8*l3*q3^2*x2^2*y3*cos(beta) + 8*l2*l3*q1^2*x2^2*sin(beta) - 4*l2^2*l3*q1^2*x2*sin(beta) + 8*l2*l3*q1^2*x3^2*sin(beta) - 8*l2*l3*q2^2*x3^2*sin(beta) - 8*l2*l3*q3^2*x2^2*sin(beta) - 8*l2^2*l3*q2^2*x3*sin(beta) + 4*l2^2*l3*q3^2*x2*sin(beta) - 8*l2^2*l3*q3^2*x3*sin(beta) - 8*l2*l3*q1^2*y3^2*sin(beta) + 8*l2*l3*q2^2*y3^2*sin(beta) - 8*l3*x2^2*x3^2*y3*cos(beta) + 4*l3*q1^2*q2^2*x2*sin(beta) - 8*l3*q1^2*q2^2*x3*sin(beta) + 4*l3*q1^2*q3^2*x2*sin(beta) - 4*l3*q2^2*q3^2*x2*sin(beta) + 32*l2*l3*x2^2*x3^2*sin(beta) - 28*l2^2*l3*x2*x3^2*sin(beta) + 24*l2^2*l3*x2^2*x3*sin(beta) + 12*l2^2*l3*x2*y3^2*sin(beta) + 8*l2^2*l3*x3*y3^2*sin(beta) - 12*l3*q1^2*x2*x3^2*sin(beta) + 16*l3*q1^2*x2^2*x3*sin(beta) + 12*l3*q2^2*x2*x3^2*sin(beta) - 8*l3*q2^2*x2^2*x3*sin(beta) - 8*l3*q3^2*x2^2*x3*sin(beta) - 4*l3*q1^2*x2*y3^2*sin(beta) + 4*l3*q2^2*x2*y3^2*sin(beta) + 8*l3*x2^2*x3*y3^2*sin(beta) - 32*l2*l3^2*x2*x3*y3*cos(2*beta) + 16*l2*l3*q1^2*x2*y3*cos(beta) - 16*l2*l3*q1^2*x3*y3*cos(beta) - 16*l2*l3*q2^2*x2*y3*cos(beta) + 16*l2*l3*q2^2*x3*y3*cos(beta) - 16*l2*l3*q3^2*x2*y3*cos(beta) + 16*l2*l3*x2*x3^2*y3*cos(beta) - 32*l2*l3*x2^2*x3*y3*cos(beta) + 40*l2^2*l3*x2*x3*y3*cos(beta) + 8*l3*q1^2*x2*x3*y3*cos(beta) - 8*l3*q2^2*x2*x3*y3*cos(beta) - 16*l2*l3*q1^2*x2*x3*sin(beta) + 16*l2*l3*q2^2*x2*x3*sin(beta) + 16*l2*l3*q3^2*x2*x3*sin(beta) - 16*l2*l3*x2*x3*y3^2*sin(beta)*

*p0=l2^2*x3^4 + 2*l2^3*x3^3 + l2^4*x3^2 + l3^2*x2^4 + l3^4*x2^2 + l2^2*y3^4 + l2^4*y3^2 + q1^4*x2^2 + q1^4*x3^2 + q2^4*x3^2 + q3^4*x2^2 + q1^4*y3^2 + q2^4*y3^2 + x2^2*x3^4 - 2*x2^3*x3^3 + x2^4*x3^2 + x2^2*y3^4 + x2^4*y3^2 + l2^2*l3^4 + l2^4*l3^2 + l2^2*q1^4 + l3^2*q1^4 + l2^2*q3^4 + l3^2*q2^4 - 2*l2*l3^4*x2 - 2*l2*q1^4*x2 + 2*l2*q1^4*x3 - 2*l2*q3^4*x2 - 2*l2*x2*x3^4 - 2*l2*x2*y3^4 - 2*q1^4*x2*x3 - 4*l2*l3^2*x2^3 - 4*l2^3*l3^2*x2 + 4*l2^3*l3^2*x3 + 2*l2*q1^2*x3^3 + 2*l2^3*q1^2*x3 - 2*l2*q2^2*x3^3 - 2*l2^3*q3^2*x3 + 6*l2*x2^2*x3^3 - 4*l2*x2^3*x3^2 - 6*l2^2*x2*x3^3 - 4*l2^3*x2*x3^2 - 4*l3^2*x2^3*x3 - 4*l2*x2^3*y3^2 - 4*l2^3*x2*y3^2 + 2*l2^3*x3*y3^2 - 2*q1^2*x2*x3^3 - 2*q1^2*x2^3*x3 + 2*q2^2*x2*x3^3 + 2*q3^2*x2^3*x3 - 2*x2^3*x3*y3^2 - 2*l2^2*l3^2*q2^2 - 2*l2^2*l3^2*q3^2 - 2*l2^3*l3^3*cos(beta) - 2*l2^2*q1^2*q3^2 - 2*l3^2*q1^2*q2^2 + 6*l2^2*l3^2*x2^2 + 4*l2^2*l3^2*x3^2 + 4*l2^2*l3^2*y3^2 + 4*l2^2*q1^2*x3^2 - 2*l2^2*q2^2*x3^2 - 2*l3^2*q2^2*x2^2 - 2*l2^2*q3^2*x3^2 - 2*l3^2*q3^2*x2^2 - 2*l2^2*q2^2*y3^2 - 2*l2^2*q3^2*y3^2 + 2*l3^3*x2^3*cos(beta) - 2*q1^2*q2^2*x3^2 - 2*q1^2*q3^2*x2^2 - 2*q1^2*q2^2*y3^2 + 6*l2^2*x2^2*x3^2 + 4*l3^2*x2^2*x3^2 + 6*l2^2*x2^2*y3^2 + 2*l2^2*x3^2*y3^2 + 4*l3^2*x2^2*y3^2 + 4*q1^2*x2^2*x3^2 - 2*q2^2*x2^2*x3^2 - 2*q3^2*x2^2*x3^2 - 2*q2^2*x2^2*y3^2 - 2*q3^2*x2^2*y3^2 + 2*x2^2*x3^2*y3^2 - 2*l3*x2^4*y3*sin(beta) + 2*l2^2*l3^2*x3^2*cos(2*beta) - 2*l2^2*l3^2*y3^2*cos(2*beta) + 4*l3^2*q1^2*x2^2*cos(beta)^2 + 2*l3^2*x2^2*x3^2*cos(2*beta) - 2*l3^2*x2^2*y3^2*cos(2*beta) - 2*l2*l3^3*q1^2*cos(beta) - 2*l2^3*l3*q1^2*cos(beta) + 2*l2*l3^3*q2^2*cos(beta) + 2*l2^3*l3*q3^2*cos(beta) - 6*l2*l3^3*x2^2*cos(beta) + 6*l2^2*l3^3*x2*cos(beta) - 4*l2^2*l3*x3^3*cos(beta) - 4*l2^2*l3^3*x3*cos(beta) - 6*l2^3*l3*x3^2*cos(beta) - 2*l2^3*l3*y3^2*cos(beta) + 2*l3*q1^2*x2^3*cos(beta) + 2*l3^3*q1^2*x2*cos(beta) - 2*l3^3*q2^2*x2*cos(beta) - 2*l3*q3^2*x2^3*cos(beta) - 4*l3*x2^2*x3^3*cos(beta) + 6*l3*x2^3*x3^2*cos(beta) - 4*l3^3*x2^2*x3*cos(beta) + 2*l3^3*q3^2*x2*cos(beta) - 4*l2*l3^3*x3^3*sin(beta) - 4*l2^2*l3^3*y3^2*sin(beta) - 4*l3*x2^2*y3^3*sin(beta) - 4*l3^3*x2^2*y3*sin(beta) + 2*l2^3*l3^2*x3*cos(2*beta) - 2*l3^2*x2^3*x3*cos(2*beta) + 2*l2^3*l3^2*y3*sin(2*beta) - 2*l3^2*x2^3*y3*sin(2*beta) - 2*l2*l3*q1^4*cos(beta) + 4*l2*l3^2*q1^2*x3 + 4*l2*l3^2*q2^2*x2 - 4*l2*l3^2*q2^2*x3 + 4*l2*l3^2*q3^2*x2 - 2*l2^4*l3*x3*cos(beta) - 2*l2*q1^2*q2^2*x3 + 4*l2*q1^2*q3^2*x2 - 2*l2*q1^2*q3^2*x3 + 2*l2*q2^2*q3^2*x3 - 8*l2*l3^2*x2*x3^2 + 12*l2*l3^2*x2^2*x3 - 12*l2^2*l3^2*x2*x3 - 8*l2*l3^2*x2*y3^2 + 2*l3*q1^4*x2*cos(beta) - 2*l3*q1^4*x3*cos(beta) - 2*l3*q2^4*x3*cos(beta) - 8*l2*q1^2*x2*x3^2 + 6*l2*q1^2*x2^2*x3 - 6*l2^2*q1^2*x2*x3 + 4*l2*q2^2*x2*x3^2 - 4*l3^2*q1^2*x2*x3 + 4*l2*q3^2*x2*x3^2 - 6*l2*q3^2*x2^2*x3 + 6*l2^2*q3^2*x2*x3 + 4*l3^2*q2^2*x2*x3 + 2*l2*q1^2*x3*y3^2 + 4*l2*q2^2*x2*y3^2 - 2*l2*q2^2*x3*y3^2 + 4*l2*q3^2*x2*y3^2 - 2*l3*x2^4*x3*cos(beta) + 2*q1^2*q2^2*x2*x3 + 2*q1^2*q3^2*x2*x3 - 2*q2^2*q3^2*x2*x3 - 4*l2*x2*x3^2*y3^2 + 6*l2*x2^2*x3*y3^2 - 2*l2^4*l3*y3*sin(beta) - 2*l3*q2^4*y3*sin(beta) + 4*l2^2*l3^2*q1^2*cos(beta)^2 + 2*l2*l3^2*q1^2*x3*cos(2*beta) - 2*l2*l3^2*q2^2*x3*cos(2*beta) - 8*l2*l3^2*q1^2*x2*cos(beta)^2 - 4*l2*l3^2*x2*x3^2*cos(2*beta) + 6*l2*l3^2*x2^2*x3*cos(2*beta) - 6*l2^2*l3^2*x2*x3*cos(2*beta) + 4*l2*l3^2*x2*y3^2*cos(2*beta) - 2*l3^2*q1^2*x2*x3*cos(2*beta) + 2*l3^2*q2^2*x2*x3*cos(2*beta) + 2*l2*l3^2*q1^2*y3*sin(2*beta) - 2*l2*l3^2*q2^2*y3*sin(2*beta) + 6*l2*l3^2*x2^2*y3*sin(2*beta) - 6*l2^2*l3^2*x2*y3*sin(2*beta) + 4*l2^2*l3^2*x3*y3*sin(2*beta) - 2*l3^2*q1^2*x2*y3*sin(2*beta) + 2*l3^2*q2^2*x2*y3*sin(2*beta) + 4*l3^2*x2^2*x3*y3*sin(2*beta) + 8*l2*l3*x2*x3^3*cos(beta) + 8*l2*l3^3*x2*x3*cos(beta) + 8*l2^3*l3*x2*x3*cos(beta) + 8*l2*l3*x2*y3^3*sin(beta) + 8*l2*l3^3*x2*y3*sin(beta) + 8*l2^3*l3*x2*y3*sin(beta) - 4*l2^3*l3*x3*y3*sin(beta) + 4*l3*x2^3*x3*y3*sin(beta) + 2*l2*l3*q1^2*x2*q2^2*cos(beta) + 2*l2*l3*q1^2*q3^2*cos(beta) - 2*l2*l3*q2^2*q3^2*cos(beta) - 6*l2*l3*q1^2*x2^2*cos(beta) + 6*l2^2*l3*q1^2*x2*cos(beta) - 6*l2*l3*q1^2*x3^2*cos(beta) - 8*l2^2*l3*q1^2*x3*cos(beta) + 6*l2*l3*q2^2*x3^2*cos(beta) + 6*l2*l3*q3^2*x2^2*cos(beta) + 4*l2^2*l3*q2^2*x3*cos(beta) - 6*l2^2*l3*q3^2*x2*cos(beta) + 4*l2^2*l3*q3^2*x3*cos(beta) - 2*l2*l3*q1^2*y3^2*cos(beta) + 2*l2^2*l3*q2^2*y3^2*cos(beta) - 2*l3*q1^2*q2^2*x2*cos(beta) + 4*l3*q1^2*q2^2*x3*cos(beta) - 2*l3*q1^2*q3^2*x2*cos(beta) + 2*l3*q2^2*q3^2*x2*cos(beta) - 18*l2*l3*x2^2*x3^2*cos(beta) + 18*l2^2*l3*x2*x3^2*cos(beta) - 12*l2^2*l3*x2^2*x3*cos(beta) - 6*l2*l3*x2^2*y3^2*cos(beta) + 6*l2^2*l3*x2*y3^2*cos(beta) - 4*l2^2*l3*x3*y3^2*cos(beta) + 6*l3*q1^2*x2*x3^2*cos(beta) - 8*l3*q1^2*x2^2*x3*cos(beta) - 6*l3*q2^2*x2*x3^2*cos(beta) + 4*l3*q2^2*x2^2*x3*cos(beta) + 4*l3*q3^2*x2^2*x3*cos(beta) + 2*l3*q1^2*x2*y3^2*cos(beta) - 2*l3*q2^2*x2*y3^2*cos(beta) + 4*l2^2*l3*q2^2*y3*sin(beta) - 4*l3*x2^2*x3*y3^2*sin(beta) + 4*l3*q1^2*q2^2*y3*sin(beta) - 12*l2^2*l3*x2^2*x3*y3*sin(beta) + 4*l3*q2^2*x2^2*y3*sin(beta) + 4*l3*q3^2*x2^2*y3*sin(beta) - 4*l3*x2^2*x3^2*y3*sin(beta) - 8*l2*l3^2*x2*x3*y3*sin(2*beta) + 16*l2*l3*q1^2*x2*x3*cos(beta) - 8*l2*l3*q2^2*x2*x3*cos(beta) - 8*l2*l3*q3^2*x2*x3*cos(beta) + 8*l2*l3*x2^2*x3*y3^2*cos(beta) - 4*l2*l3*q1^2*q2^2*y3*sin(beta) - 8*l2*l3*q2^2*x2^2*y3*sin(beta) + 4*l2*l3*q2^2*x3*y3*sin(beta) - 8*l2*l3*q3^2*x2^2*y3*sin(beta) + 8*l2*l3*x2*x3^2*y3*sin(beta) - 12*l2*l3*x2^2*x3*y3*sin(beta) + 12*l2^2*l3*x2*x3*y3*sin(beta) + 4*l3*q1^2*x2*x3*y3*sin(beta) - 4*l3*q2^2*x2*x3*y3*sin(beta)*

%**********
ANNEXE A2 CALCUL GEOMETREI INVERSE

```
function [Q,P]=geo_inv_par(A,T,M)
% A : POSITIONS POINTS ANCRAGE  A=[x1 y1 ;x2 y2;x3 y3]
% T : DIMENSIONS TRIANGLES : T = [l2 L3 phi(deg)]
% M : POSITION [x,y,phi(deg)]
% Q : VECTEUR POSITIONS ARTICULAIRES Q=[q1 q2 q3]
% P : POSTIONS SOMMET TRIANGLE PLATEFORME

x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
l2=T(1);
l3=T(2);
BETA=T(3)*pi/180;
CB=cos(BETA);
SB=sin(BETA);
if (BETA==pi/2)||(BETA==-pi/2)
   CB=0;
end
if (BETA==pi)||(BETA==-pi)
   SB=0;
end

l1=l2^2+l3^2-2*l3*l2*cos(BETA);
x=M(1);
y=M(2);
PHI=M(3)/180*pi;
CP=cos(PHI);
SP=sin(PHI);
if (PHI==pi/2)||(PHI==-pi/2)
   CP=0;
end
if (PHI==pi)||(PHI==-pi)
   SP=0;
end

q1=sqrt(x^2+y^2);
q2=sqrt((x2-x-l2*CP)^2+(y+l2*SP)^2);
q3=sqrt((x3-x-l3*(CP*CB-SP*SB))^2+(y3-y-l3*(SP*CB+CP*SB))^2);
Q=[q1 q2 q3];
P(1,:)=[M(1) M(2)];
P(2,:)=[M(1)+l2*CP  M(2)+l2*SP];
P(3,:)=[M(1)+l3*(CB*CP-SB*SP)  M(2)+l3*(SB*CP+CB*SP)];
 %*************************************************************************************************************
```

ANNEXE A3 : CALCUL ERREUR GLOBALE POSITIONNEMENT ET ANGLE PHI) A PARTIR DONNE MATRICE P

```
 function [Err,PHI]=calcul_erreur_global(A,T,P,Q)
% A : POSITIONS Ai
% T : DIMENSIONS PLATEFORME
% Q COORDONNES ARTICLAIRES
% P : POINTS Pi SOLUTION
% Ereur globale
Err=[];
PHI=[];
q1=Q(1);
q2=Q(2);
q3=Q(3);
x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
%q1=Q(1);q2=Q(2);q3=Q(3);
l2=T(1);
l3=T(2);
BETA=T(3)*pi/180;
CB=cos(BETA);
SB=sin(BETA);
if (BETA==pi/2)||(BETA==-pi/2)
   CB=0;
end

l1=sqrt(l2^2+l3^2-2*l3*l2*CB);
if size(P,3)==1
```

```
n=size(P,2);
   for ii=1:n/2
    erreur= abs(norm([P(1,2*(ii-1)+1)-x1  P(1,2*ii)-y1])-q1);
    erreur=erreur+ abs(norm([P(2,2*(ii-1)+1)-x2  P(2,2*ii)-y2])-q2);
    erreur=erreur+ abs(norm([P(3,2*(ii-1)+1)-x3  P(3,2*ii)-y3])-q3);
    Err=[Err,erreur];
    PHI=atan2(y2-P(2,1),x2-P(1,1));
   end
else
  m=size(P,3);
  for ii=1:m
   erreur= abs(norm([P(1,1,ii)-x1  P(1,2,ii)-y1])-q1);
   erreur= erreur+abs(norm([P(2,1,ii)-x2  P(2,2,ii)-y2])-q2);
   erreur= erreur+abs(norm([P(3,1,ii)-x3  P(3,2,ii)-y3])-q3);
   Err=[Err,erreur];
   PHI=[PHI,atan2(y2-P(2,1,ii),x2-P(1,1,ii))];
   end
end
%*****************************************************************************************************
```

ANNEXE A4 DETERMINATION VECTEUR ARTICULAIRE A PARTIR DONNEE MATRICE P

```
function q=calcul_mod(A,T,P)
% A : POSITIONS POINTS ANCRAGE  A=[x1 y1 ;x2 y2;x3 y3]
% T : DIMENSIONS TRIANGLES : T = [l2 L3 PHI(deg)]
% Q : VECTEUR POSITIONS ARTICULAIRES Q=[q1 q2 q3]
close all
x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
%q1=Q(1);q2=Q(2);q3=Q(3);
l2=T(1);
l3=T(2);
BETA=T(3)*pi/180;
if size(P,3)==1
l1=sqrt(l2^2+l3^2-2*l3*l2*cos(BETA));
q(1)=norm([P(1,1)-x1 P(1,2)-y1]);
q(2)=norm([P(2,1)-x2 P(2,2)-y2]);
q(3)=norm([P(3,1)-x3 P(3,2)-y3]);
else
   for ii=1:size(P,3)
      q(ii,:)=[norm([P(1,1,ii)-x1 P(1,2,ii)-y1]); norm([P(2,1,ii)-x2 P(2,2,ii)-y2]);norm([P(3,1,ii)-x3 P(3,2,ii)-y3])];
   end
end
%******************************************************** ********************************************
```

ANNEXE A5 : METHODE POLYNOMIALE ET DEGENERESCENCE

```
 function [P,PHI_st]=geo_direct_poly(A,T,Q)
% A : POSITIONS POINTS ANCRAGE  A=[x1 y1 ;x2 y2;x3 y3]
% T : DIMENSIONS TRIANGLES : T = [l2 L3 PHI(deg)]
% Q : VECTEUR POSITIONS ARTICULAIRES Q=[q1 q2 q3]
close all
flag=1;
x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
q1=Q(1);q2=Q(2);q3=Q(3);
l2=T(1);
l3=T(2);
BETA=T(3)*pi/180;
CB=cos(BETA);
SB=sin(BETA);
if (BETA==pi/2)||(BETA==-pi/2)
   CB=0;
elseif  (BETA==pi)||(BETA==-pi)
   SB=0;
else
end
l1=sqrt(l2^2+l3^2-2*l3*l2*CB);
%axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
a0=-(q2^2-q1^2-x2^2-l2^2)*y3;
a1=(q2^2-q1^2-x2^2-l2^2)*l3*SB-2*x2*l2*y3;
```

```
a2=(q2^2-q1^2-x2^2-l2^2)*l3*CB-(q3^2-q1^2-x3^2-y3^2-l3^2)*l2;
a3=2*l2*l3*x2*SB;
a4=2*l2*l3*(x3*SB-y3*CB);
a5=2*l2*l3*(x2*CB-x3*CB-y3*SB);
b0=(q2^2-q1^2-x2^2-l2^2)*x3-(q3^2-q1^2-x3^2-y3^2-l3^2)*x2;
b1=(q3^2-q1^2-x3^2-y3^2-l3^2)*l2-(q2^2-q1^2-x2^2-l2^2)*l3*CB+2*x2*x3*l2-2*x2*x3*l3*CB-2*x2*l3*y3*SB;
b2=(q2^2-q1^2-x2^2-l2^2)*l3*SB+2*x2*x3*l3*SB-2*x2*l3*y3*CB;
b3=2*l2*l3*(x3*CB-x2*CB+y3*SB);
b4=0;
b5=2*l2*l3*(y3*CB+x2*SB-x3*SB);
c0=x2*y3+l2*l3*SB;
c1=-(l2*y3+x2*l3*SB);
c2=x3*l2-x2*l3*CB;
CC=[c0-c1 2*c2 c0+c1]
roots(CC)
DDX=[a0-a1+a3  2*(a2-a5)  2*(a0-a3+2*a4)  2*(a2+a5)  a0+a1+a3];
DDY=[b0-b1+b3  2*(b2-b5)  2*(b0-b3+2*b4)  2*(b2+b5)  b0+b1+b3];
% CALCUL POLYNOME ORDRE 8
roots(conv(DDX,DDX)+conv(DDY,DDY))
roots(conv(DDX,DDX))
roots(conv(DDY,DDY))
POLY=conv(DDX,DDX)+conv(DDY,DDY)-4*q1^2*conv([1 0 2 0 1],conv(CC,CC))
% CALCUL RACINES t=tan((PHI/2))
r=roots(POLY)
index=find(abs(imag(r))<=1e-6);
P=zeros(3,2,2);
PHI_s=zeros(1,1);
for ii=1:length(index)
%METHODE DIRECTE
PHI=2*atan(real(r(index(ii))));
if abs(PHI)<=1e-14
   PHI=0;
end
CF=cos(PHI);
SF=sin(PHI);
%if (PHI==pi/2)||(PHI==-pi/2)
 %   CF=0;
%elseif  (PHI==pi)||(PHI==-pi)
 %   SF=0;
%else
%end
Dx=2*(-(q2^2-q1^2-x2^2-l2^2)*y3+CF*((q2^2-q1^2-x2^2-l2^2)*l3*SB-2*x2*l2*y3)+SF*((q2^2-q1^2-x2^2-l2^2)*l3*CB-(q3^2-q1^2-x3^2-
y3^2-l3^2)*l2)+CF^2*2*l2*l3*x2*SB+SF^2*2*l2*l3*(x3*SB-y3*CB)+2*l2*l3*CF*SF*(x2*CB-x3*CB-y3*SB));
Dy=2*((q2^2-q1^2-x2^2-l2^2)*x3-(q3^2-q1^2-x3^2-y3^2-l3^2)*x2+CF*((q3^2-q1^2-x3^2-y3^2-l3^2)*l2-(q2^2-q1^2-x2^2 -l2^2)*l3*CB
+2*x2*x3*l2-2*x2*x3*l3*CB-2*x2*l3*y3*SB) +SF*((q2^2-q1^2-x2^2    -l2^2)*l3*SB+2*x2*x3*l3*SB-
2*x2*l3*y3*CB)+CF^2*2*l2*l3*(x3*CB-x2*CB+y3*SB)+2*l2*l3*CF*SF*(y3*CB+x2*SB-x3*SB));
D=4*(l2*l3*SB+x2*y3-CF*(l2*y3+x2*l3*SB)+SF*(x3*l2-x2*l3*CB));
Dx
Dy
D

% METHODE POLYNOMIALE
%Dx=2*polyval(DDX,r(index(ii)))/polyval([1 0 2 0 1],r(index(ii)));
%Dy=2*polyval(DDY,r(index(ii)))/polyval([1 0 2 0 1],r(index(ii)));
%D=4*polyval(CC,r(index(ii)))/polyval([1 0 1],r(index(ii)));
% POSITIONS P
if abs(Dx)<=1e-14
   Dx=0;
end
if abs(Dy)<=1e-14
   Dy=0;
end
if abs(D)<=1e-14
   D=0;
end

PHI_s(ii)=PHI ;
QQ=2*(l2*CF-x2)
RR=2*l2*SF
SS=q2^2-q1^2-x2^2-l2^2+2*l2*x2*CF
UU=2*(l3*(CF*CB-SF*SB)-x3)
VV=2*(l3*(SF*CB+CF*SB)-y3)
WW=q3^2-q1^2-x3^2-y3^2-l3^2+2*x3*l3*(CF*CB-SF*SB)+2*y3*l3*(SF*CB+CF*SB)


if ~any([Dx  Dy D])||(D==0)%||(PHI ==0)
      if flag==1
         disp('b1')
```

```
[x,y]=solve_2nd(Q,QQ,RR,SS,UU,VV,WW,flag);
            if ~isempty(x)
               P(:,:,ii)=[x                    y
                         x+l2*CF        y+l2*SF
                         x+l3*(CF*CB-SF*SB)   y+l3*(SF*CB+CF*SB)];
                         flag=0;
            end
      else
         disp('b2')
       [x,y]=solve_2nd(Q,QQ,RR,SS,UU,VV,WW,flag);
             if ~isempty(x)
                          P(:,:,ii)=[x                    y
                          x+l2*CF        y+l2*SF
                          x+l3*(CF*CB-SF*SB)   y+l3*(SF*CB+CF*SB)];
                         flag=1;
             end
      end
else
   disp('aa'),ii
x=Dx/D;
y=Dy/D;
P(:,:,ii)=[x                    y
x+l2*CF        y+l2*SF
x+l3*(CF*CB-SF*SB)   y+l3*(SF*CB+CF*SB)];
PHI_s(ii)=PHI;
end
end
P
nn=size(index,1);
[PHI_st,ixx]=sort(PHI_s);
for ii=1:nn
   if nn==1
   elseif nn==2
  subplot(2,1,ii);
  V_axis(ii,:)=plot_graphes_par(A,P,ixx(ii));
  mn=[2,1];
   elseif nn==3
      subplot(1,3,ii)
      mn=[3,1];
   elseif nn==4
     subplot(2,2,ii)
     V_axis(ii,:)=plot_graphes_par(A,P,ixx(ii));
     mn=[2,2];
   else
     subplot(2,nn/2,ii)
     V_axis(ii,:)=plot_graphes_par(A,P,ixx(ii));
     mn=[2,3];
   end
%axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
%mx=max(max([P(1,1,:) P(2,1,:) P(3,1,:)]));
%Mx=max(max([P(1,2,:) P(2,2,:) P(3,2,:)]));
%axis([-0.5 mx -0.5 Mx]);
%h=plot(x1,y1,'.r');set(h,'MarkerSize',20)
%hold on
%h=plot(x2,y2,'.r');set(h,'MarkerSize',20)
%h=plot(x3,y3,'.r');set(h,'MarkerSize',20)
%axis equal
%set(gcf,'color','w')
%set(gca,'fontsize',8,'fontweigh','bold')
%hx=xlabel('x');set(hx,'fontsize',8,'fontweigh','bold');
%grid on
end
hh=findobj('type','axes');
for ii=1:length(hh);
   V_axis(ii,:)=axis(hh(ii));
end

for ii=1:length(hh)
   subplot(mn(1),mn(2),ii)
   axis([min(V_axis(:,1)), max(V_axis(:,2))  min(V_axis(:,3)), max(V_axis(:,4)) ]);
  %axis([-0.5 2.5 -1.5 2.25]);
   h=title(['#',int2str(ii),'   ','\phi =  ',num2str(PHI_s(ixx(ii))*180/pi,6), ' °']);
   set(h,'fontsize',8,'fontweigh','bold');
   h=xlabel(['[x,y] = ',mat2str([P(1,1,ixx(ii)) P(1,2,ixx(ii))],4)]);
  set(h,'fontsize',8,'fontweigh','bold');
  axis equal
end
set(gcf,'color','w')
```

```
PP=P(:,:,ixx);
P=PP;

%*************************************************************************************************************
function  [x,y]=solve_2nd(Q,QQ,RR,SS,UU,VV,WW,flag);
q1=Q(1);q2=Q(2);q3=Q(3);x=[];y=[];
if abs(UU)<=1e-10
   UU=0;
end
if abs(VV)<=1e-10
   VV=0;
end
if abs(WW)<=1e-10
   WW=0;
end
if abs(QQ)<=1e-10
   QQ=0;
end
if abs(RR)<=1e-10
   RR=0;
end
if abs(SS)<=1e-10
   SS=0;
end
if QQ==0
   if RR==0
      if  SS==0
%*************************************************************************************************************
        % CAS OU   :  QQ=RR=SS=0
            if UU==0
                if VV==0
                    if  WW==0
                    else %WW~=0
                    % PAS DE SOLUTION
                    end
                else  %VV~=0
                    y=WW/VV;
                    if flag==1
                    x=-sqrt(q1^2-y^2);
                    else
                    x= sqrt(q1^2-y^2);
                    end
                end
            else  %UU~=0
                if VV==0
                  x=WW/UU;
                    if flag==1
                    y=-sqrt(q1^2-x^2);
                    else
                    y=sqrt(q1^2-x^2);
                    end
                else  %VV~=0
                   delta=q1^2*(1+UU^2/VV^2) -WW^2/VV^2;
                   if delta>=0
                        if flag==1
                         x=(WW*UU/VV^2-sqrt(delta))/(1+UU^2/VV^2);
                           y=(WW-UU*x)/VV;
                        else
                            x=(WW*UU/VV^2+sqrt(delta))/(1+UU^2/VV^2);
                            y=(WW-UU*x)/VV;
                        end
                   end
                end
            end

%*************************************************************************************************************
      else  % QQ=RR=0, SS~=0
         % PAS DE SOLUTION
      end
   else      % QQ=0  RR~=0
      y=SS/RR;
      if flag==1
         x=-sqrt(q1^2-y^2);
      else
          x=sqrt(q1^2-y^2);
      end
   end
else %QQ~=0
```

```
if RR==0
    x=SS/QQ;
    if flag==1
    y=-sqrt(q1^2-x^2);
    else
     y=sqrt(q1^2-x^2);
    end
  else %RR~=0
    delta=q1^2*(1+QQ^2/RR^2) -SS^2/RR^2
    if delta>=0
        if flag==1
            disp('c1')
             x=(SS*QQ/RR^2-sqrt(delta))/(1+QQ^2/RR^2)
             y=(SS-QQ*x)/RR
        else
             disp('c2')
              x=(SS*QQ/RR^2+sqrt(delta))/(1+QQ^2/RR^2)
              y=(SS-QQ*x)/RR
        end
    end
  end
end
%*************************************************************************************************
function  V_axis=plot_graphes_par(A,P,ii);
x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
h=patch([P(1,1,ii) P(2,1,ii) P(3,1,ii) P(1,1,ii) ],[P(1,2,ii) P(2,2,ii) P(3,2,ii) P(1,2,ii)],[0 0 1]);
set(h,'linewidth',1);
hold on
grid on
h=plot([x1 P(1,1,ii)],[y1 P(1,2,ii)]);set(h,'linewidth',2,'color','k');
h=plot([x2 P(2,1,ii)],[ y2 P(2,2,ii)]);set(h,'linewidth',2,'color','k');
h=plot([x3 P(3,1,ii)],[y3  P(3,2,ii)]);set(h,'linewidth',2,'color','k');
h=plot(x1,y1,'.r');set(h,'MarkerSize',20)
h=plot(x2,y2,'.r');set(h,'MarkerSize',20)
h=plot(x3,y3,'.r');set(h,'MarkerSize',20)
set(gca,'box','on','fontweigh','bold')
V_axis=axis;

%*************************************************************************************************
```

ANNEXE A6 : RESOLUTION METHODE TROISIEME DEGRE PAR METHODE DE CARDAN

```
function X=methode_cardan(A)
% COEFFICIENTS POLYNOME
% X RACINES
% CALCUL RACINE POLYNOME a3X^3+a2X^2+a1X+a0 = 0 PAR METHODE DE CARDAN
a3=A(1);
a2=A(2);
a1=A(3);
a0=A(4);
% RAMENE LE POLYNOME A LA FORME CLASSIQUE X^3+pX+q=0
p=(a1/a3-a2^2/3/a3^2);
q=-a2/3/a3*(a1/a3-a2^2/3/a3^2)+a0/a3-a2^3/27/a3^3;
delta=-(4*p^3+27*q^2);
if abs(delta)<=1e-10
X(1)=  2*(0.5*(-q+sqrt(-delta/27)))^(1/3)-a2/3/a3;
X(2)=2*cos(2*pi/3)*(0.5*(-q+sqrt(-delta/27)))^(1/3)-a2/3/a3;
X(3)=2*cos(4*pi/3)*(0.5*(-q+sqrt(-delta/27)))^(1/3)-a2/3/a3;
elseif delta >0
for kk=0:2
   X(kk+1)=2*sqrt(-p/3)*cos(1/3*acos(-q/2*sqrt(-27/p^3))+2*kk*pi/3);
   X(kk+1)=X(kk+1)-a2/3/a3;
end
else
   for kk=0:2
   z=exp(2*pi*1j/3);
   X(kk+1)=z^kk*(0.5*(-q+sqrt(-delta/27)))^(1/3)+z^(-kk)*(0.5*(-q-sqrt(-delta/27)))^(1/3);
   X(kk+1)=X(kk+1)-a2/3/a3;
end
end
%*************************************************************************************************
```

ANNEXE B : METHODE GEOMETRIQUE

ANNEXE B1 FONCTION INTERSECTION

```
function M=calcul_inter(C1,C2,type)
%close all
% C1 : Coordonnes centre et rayon (x,y,r) cercle 1
% C2 : Coordonnes centre et rayon (x,y,r) cercle 2
% type =1 : Tracé cercle
% M : Coordonnées point intersection
xx1=C1(1);yy1=C1(2);r1=C1(3);
xx2=C2(1);yy2=C2(2);r2=C2(3);
d=sqrt((xx1-xx2)^2+(yy1-yy2)^2);
if d>r1+r2
  M=[];
else
  if (xx1==xx2)&&(yy1~=yy2)
    M(1,2)=(r1^2-r2^2+yy2^2-yy1^2)/2/(yy2-yy1);
    M(1,1)=xx1+sqrt(r1^2-(1/2/(yy2-yy1)*(r1^2-r2^2+(yy2-yy1)^2))^2);
    M(2,2)=M(1,2);
    M(2,1)=xx1-sqrt(r1^2-(1/2/(yy2-yy1)*(r1^2-r2^2+(yy2-yy1)^2))^2);

  elseif (yy1==yy2)&&(xx1~=xx2)
    M(1,1)=(r1^2-r2^2+xx2^2-xx1^2)/2/(xx2-xx1);
    M(1,2)=yy1+sqrt(r1^2-(1/2/(xx2-xx1)*(r1^2-r2^2+(xx2-xx1)^2))^2);
    M(2,1)=M(1,1);
    M(2,2)=yy1-sqrt(r1^2-(1/2/(xx2-xx1)*(r1^2-r2^2+(xx2-xx1)^2))^2);

  elseif (yy1==yy2)&&(xx1==xx2)
M=[];

   else
     A=1+(xx2-xx1)^2/(yy2-yy1)^2;
    B=xx1+0.5*(xx2-xx1)/(yy2-yy1)^2*(r1^2-r2^2+xx2^2-xx1^2+(yy2-yy1)^2);
    C=xx1^2+(1/2/(yy2-yy1)*(r1^2-r2^2+xx2^2-xx1^2+(yy2-yy1)^2))^2-r1^2;
    if  B^2-A*C>0
    M(1,1)=(B+sqrt(B^2-A*C))/A;
    M(2,1)=(B-sqrt(B^2-A*C))/A;
    M(1,2)=(r1^2-r2^2+xx2^2-xx1^2+yy2^2-yy1^2-2*M(1,1)*(xx2-xx1))/2/(yy2-yy1);
    M(2,2)=(r1^2-r2^2+xx2^2-xx1^2+yy2^2-yy1^2-2*M(2,1)*(xx2-xx1))/2/(yy2-yy1);
    elseif  B^2-A*C==0
    M(1,1)=B/A;
    M(2,1)=M(1,1);
    M(1,2)=yy1+sqrt(r1^2-(M(1,1)-xx1)^2);
    M(2,2)=yy1-sqrt(r1^2-(M(1,1)-xx1)^2);
    else
    M=[];
    end

  end
end

err11=abs(norm([M(1,1)-xx1 M(1,2)-yy1])-r1);
err12=abs(norm([M(1,1)-xx2 M(1,2)-yy2])-r2);
err21=abs(norm([M(2,1)-xx1 M(2,2)-yy1])-r1);
err22=abs(norm([M(2,1)-xx2 M(2,2)-yy2])-r2);
if (er11>1e-10)&&(err12>1e10)
   if (err21>1e-10)&&(err22>1e10)
      M(2,:)=[];
   end
else
    M(1,:)=[];
end


if type==1
figure(1);
phi=linspace(0,2*pi,100);
x=xx1+r1*cos(phi);
y=yy1+r1*sin(phi);
hh=plot(x,y,'k');set(hh,'linewidth',2);
grid on
hold on
x=xx2+r2*cos(phi);
y=yy2+r2*sin(phi);
```

```
hh=plot(x,y,'k');set(hh,'linewidth',2);
if ~isempty(M)&~all(imag(M))
hh=plot(M(1,1),M(1,2),'.r');set(hh,'MarkerSize',20);
hh=plot(M(2,1),M(2,2),'.r');set(hh,'MarkerSize',20);
axis equal
set(gcf,'color','w')
set(gca,'fontsize',8,'fontweigh','bold')
hx=xlabel('x');set(hx,'fontsize',8,'fontweigh','bold');
hy=ylabel('y');set(hy,'fontsize',8,'fontweigh','bold');
plot(xx1,yy1,'+r')
plot(xx2,yy2,'+r')
end
end

%*************************************************************************************************************
```

ANNEXE B2: CONSTRUCTION GRAPHIQUE

```
function methode_geo(A,T,Q,P)
% A : POSITIONS Ai
% T : DIMENSIONS PLATEFORME
% Q : COORDONNES ARTICLAIRES
% P : POINTS Pi SOLUTION
close all
figure(1);
l2=T(1);
l3=T(2);
beta=T(3)*pi/180;
l1=sqrt(l2^2+l3^2-2*l2*l3*cos(beta));
x=A(1,1);
y=A(1,2);
trace_cercle([x,y,Q(1)],'k')
hold on
x=A(2,1);
y=A(2,2);
trace_cercle([x,y,Q(2)],'k')
x=A(3,1);
y=A(3,2);
trace_cercle([x,y,Q(3)],'k')
P=P(:,:,1);
x=P(1,1);
y=P(1,2);
trace_cercle([x,y,l2],'r')
x=P(2,1);
y=P(2,2);
trace_cercle([x,y,l1],'r')

x=P(3,1);
y=P(3,2);
trace_cercle([x,y,l3],'r')
grid on
axis equal
shg


x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
h=plot([x1 P(1,1,1)],[y1 P(1,2,1)]);set(h,'linewidth',2,'color','g');
h=plot([x2 P(2,1,1)],[ y2 P(2,2,1)]);set(h,'linewidth',2,'color','g');
h=plot([x3 P(3,1,1)],[y3  P(3,2,1)]);set(h,'linewidth',2,'color','g');
h=plot([P(1,1,1) P(2,1,1) P(3,1,1) P(1,1,1)],[P(1,2,1) P(2,2,1) P(3,2,1) P(1,2,1)],'b');
set(h,'linewidth',2)
h=plot(x1,y1,'.r');set(h,'MarkerSize',20);
h=plot(x2,y2,'.r');set(h,'MarkerSize',20);
h=plot(x3,y3,'.r');set(h,'MarkerSize',20);
set(gcf,'color','w')
set(gca,'box','on','FontSize',10,'FontWeight','bold')

% APPEL SOUS-PROGRAMME TRACE
 %*******************************************
function trace_cercle(C,col)
phi=linspace(0,2*pi,100);
xx=C(1);
yy=C(2);
```

```
r=C(3);
x=xx+r*cos(phi);
y=yy+r*sin(phi);
hh=plot(x,y,col);set(hh,'linewidth',2);
```

****

ANNEXE B3 FONCTION METHODE GEOMETRIQUE

```
function [P,Err,PHI,q]=geo_direct_geo(A,T,Q)
% A : POSITIONS POINTS ANCRAGE  A=[x1 y1 ;x2 y2;x3 y3]
% T : DIMENSIONS TRIANGLES : T =  [l2 L3 phi(deg)]
% Q : VECTEUR POSITIONS ARTICULAIRES Q=[q1 q2 q3]
% P : SOLUTIONS
% Erreurs (Precision)
close all
theta=[];
flag=1;
P1=[];erreur1=[];indice1=[];
P2=[];erreur2=[];indice2=[];
x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
q1=Q(1);q2=Q(2);q3=Q(3);
l2=T(1);
l3=T(2);
phi=T(3)*pi/180;
l1=sqrt(l2^2+l3^2-2*l3*l2*cos(phi));
axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
hold on
h=plot(x1,y1,'.b');set(h,'MarkerSize',20)
h=plot(x2,y2,'.b');set(h,'MarkerSize',20)
h=plot(x3,y3,'.b');set(h,'MarkerSize',20)
grid on
phi=linspace(0,2*pi,100);
x=x1+q1*cos(phi);
y=y1+q1*sin(phi);
hh=plot(x,y,'k');set(hh,'linewidth',2)
x=x2+q2*cos(phi);
y=y2+q2*sin(phi);
hh=plot(x,y,'k');set(hh,'linewidth',2)
x=x3+q3*cos(phi);
y=y3+q3*sin(phi);
hh=plot(x,y,'k');set(hh,'linewidth',2)
set(gcf,'color','w')
set(gca,'fontsize',10,'fontweigh','bold','box','on')
axis([-1.5*q1 1.5*(x2+q2) -1.5*max(q1,q2) 1.5*(q3+y3)]);
axis equal
hx=xlabel('x');set(hx,'fontsize',10,'fontweight','bold')
hy=ylabel('y');set(hy,'fontsize',10,'fontweight','bold')
phi=linspace(0,2*pi,1000);
for ii=1:length(phi)-1
   x(ii)=q1*cos(phi(ii));
   y(ii)=q1*sin(phi(ii));
    if (norm([x1-x3 y1-y3])-q1-q3<=l3)&&(norm([x1-x2 y1-y2])-q1-q2<=l2)&&(norm([x2-x3 y2-y3])-q2-q3<=l1)
      C1=[x(ii) y(ii) l2];
      C2=[x2 y2 q2];
      %disp('aa')
      M=calcul_inter_n(C1,C2,0);
      %disp('bb')
      %pause
       pause(0.1)
      if ~isempty(M)%&(imag(M)==0)
        C1=[x(ii) y(ii) l3];
        C2=[M(1,1) M(1,2) l1];
       % disp('00')
        MM=calcul_inter_n(C1,C2,0);
       %disp('11')
      % pause
      if ~isempty(MM)
        if flag==1
       %h=patch([P(1,1,ii) P(2,1,ii) P(3,1,ii) P(1,1,ii) ],[P(1,2,ii) P(2,2,ii) P(3,2,ii) P(1,2,ii)],[0 0 1]);
      % hhh=plot([x(ii) M(1,1) MM(1,1) x(ii)],[y(ii) M(1,2) MM(1,2) y(ii)],'g');
       hhh=patch([x(ii) M(1,1) MM(1,1) x(ii)],[y(ii) M(1,2) MM(1,2) y(ii)],[0 0 1]);
       %set(hhh,'FaceColor ',[0  0 1])
       %set(hhh,'linewidth',2),
       % disp('a')
```

```
 pause(0.1)
flag=0;
end
%set(hhh,'XData',[x(ii) M(1,1) MM(2,1) x(ii)],'YData',[y(ii)
%M(1,2) MM(2,2) y(ii)]);
%M
%MM
%[x(ii) M(1,1) MM(2,1) x(ii)]
% [y(ii) M(1,2) MM(2,2) y(ii)]
set(hhh,'XData',[x(ii) M(1,1) MM(2,1) x(ii)],'YData',[y(ii) M(1,2) MM(2,2) y(ii)],'FaceColor',[0 0 1]);
%set(hhh,'linewidth',2,'color','g'),
%disp('b')
pause(0.1)
%MM,q3,x3,y3
%abs(norm([MM(1,1)-x3 MM(1,2)-y3])-q3),ii
%abs(norm([MM(2,1)-x3 MM(2,2)-y3])-q3),ii
if abs(norm([MM(1,1)-x3 MM(1,2)-y3])-q3)<=1e-2
   P1=[P1,[x(ii) y(ii);M(1,1) M(1,2);MM(1,1) MM(1,2)]];
   %set(hhh,'XData',[x(ii) M(1,1) MM(1,1) x(ii)],'YData',[y(ii) M(1,2) MM(1,2) y(ii)]);
   axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
   set(hhh,'XData',[x(ii) M(1,1) MM(1,1) x(ii)],'YData',[y(ii) M(1,2) MM(1,2) y(ii)],'FaceColor',[1 0 0 ]);
  % set(hhh,'color','r'),
  %disp('c')
   pause(0.1)
   %set(hhh,'Xdata',[x(ii) M(1,1) MM(1,1) x(ii)],'YData',[y(ii) M(1,2) MM(1,2) y(ii)]);
   set(hhh,'FaceColor','b');
   erreur1=[erreur1,abs(norm([MM(1,1)-x3 MM(1,2)-y3])-q3)];
   indice1=[indice1,ii];
   %break
elseif abs(norm([MM(2,1)-x3 MM(2,2)-y3])-q3)<=1e-2
   P2=[P2,[x(ii) y(ii);M(1,1) M(1,2);MM(2,1) MM(2,2)]];
   %set(hhh,'XData',[x(ii) M(1,1) MM(2,1) x(ii)],'YData',[y(ii) M(1,2) MM(2,2) y(ii)]);
   axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
   set(hhh,'XData',[x(ii) M(1,1) MM(2,1) x(ii)],'YData',[y(ii) M(1,2) MM(2,2) y(ii)],'FaceColor',[1 0 0]);
   %set(hhh,'color','r'),
   %disp('d')
   pause(0.1)
   %plot([x(ii) M(1,1) MM(2,1) x(ii)],[y(ii) M(1,2) MM(2,2) y(ii)],'g');
    set(hhh,'FaceColor','b');
   erreur2=[erreur2,abs(norm([MM(2,1)-x3 MM(2,2)-y3])-q3)];
   indice2=[indice2,ii];
 % break
else
end
end

   C1=[x(ii) y(ii) l3];
   C2=[M(2,1) M(2,2) l1];
% disp('22')
   MM=calcul_inter_n(C1,C2,0);
   %disp('33')
   %pause
   if ~isempty(MM)
   %set(hhh,'XData',[x(ii) M(2,1) MM(1,1) x(ii)],'YData',[y(ii) M(2,2) MM(1,2) y(ii)])
   axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
   set(hhh,'XData',[x(ii) M(2,1) MM(1,1) x(ii)],'YData',[y(ii) M(2,2) MM(1,2) y(ii)],'FaceColor',[0 0 1]);
   %set(hhh,'linewidth',2,'color','g'),
   %disp('e')
   pause(.1)
   %set(hhh,'XData',[x(ii) M(2,1) MM(2,1) x(ii)],'YData',[y(ii) M(2,2) MM(2,2) y(ii)]);
   axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
   set(hhh,'XData',[x(ii) M(2,1) MM(2,1) x(ii)],'YData',[y(ii) M(2,2) MM(2,2) y(ii)],'FaceColor',[0 0 1]);
   %set(hhh,'linewidth',2,'color','g'),
   %disp('f')
   pause(.1)
   %abs(norm([MM(1,1)-x3 MM(1,2)-y3])-q3),ii
   %abs(norm([MM(2,1)-x3 MM(2,2)-y3])-q3),ii
   if abs(norm([MM(1,1)-x3 MM(1,2)-y3])-q3)<=1e-2
      P1=[P1,[x(ii) y(ii);M(2,1) M(2,2);MM(1,1) MM(1,2)]];
     % set(hhh,'XData',[x(ii) M(2,1) MM(1,1) x(ii)],'YData',[y(ii) M(2,2) MM(1,2) y(ii)]);
     axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
      set(hhh,'XData',[x(ii) M(2,1) MM(1,1) x(ii)],'YData',[y(ii) M(2,2) MM(1,2) y(ii)],'FaceColor',[1 0 0]);
      %set(hhh,'color','r'),
      %disp('g')
      pause(0.5)
      %plot([x(ii) M(2,1) MM(1,1) x(ii)],[y(ii) M(2,2) MM(1,2) y(ii)],'g');
      set(hhh,'FaceColor','b');
      erreur1=[erreur1,abs(norm([MM(1,1)-x3 MM(1,2)-y3])-q3)];
```

```
          indice1=[indice1,ii];
          %plot([x(ii) M(2,1) MM(1,1) x(ii)],[y(ii) M(2,2) MM(1,2) y(ii)],'g');
          %break
        elseif abs(norm([MM(2,1)-x3 MM(2,2)-y3])-q3)<=1e-2
          %set(hhh,'color','r'),pause(0.5)
          P2=[P2,[x(ii) y(ii);M(2,1) M(2,2);MM(2,1) MM(2,2)]];
          %set(hhh,'XData',[x(ii) M(2,1) MM(2,1) x(ii)],'YData',[y(ii) M(2,2) MM(2,2) y(ii)]);
          axis([-1.2*q1 1.2*(q2+max(A(:,1))) -1.2*max(q1,q2) 1.2*(y3+q3)]);
          set(hhh,'XData',[x(ii) M(2,1) MM(2,1) x(ii)],'YData',[y(ii) M(2,2) MM(2,2) y(ii)],'FaceColor',[1 0 0]);
          %set(hhh,'color','r'),
          pause(0.5)
          %plot([x(ii) M(2,1) MM(2,1) x(ii)],[y(ii) M(2,2) MM(2,2) y(ii)],'g');
          set(hhh,'FaceColor','b');
          erreur2=[erreur2,abs(norm([MM(2,1)-x3 MM(2,2)-y3])-q3)];
          indice2=[indice2,ii];
          %break

          %plot([x(ii) M(2,1) MM(2,1) x(ii)],[y(ii) M(2,2) MM(2,2) y(ii)],'g');

        else
        end
        end
      end
    end
end

[P,Err,PHI,q]=geo_direct_optim(A,T,Q,indice1,indice2,phi);
nn=size(P,3);
[PHI,ixx]=sort(PHI);
P=P(:,:,ixx);
q=q(ixx,:);
Err=Err(ixx);
```

ANNEXE C : METHODE NEURONE-FLOUE

ANNEXE C1 :PROGRAMME METHODE NERONE-FLOUE

```
function M=geo_direct_fuzzy(A,T,dx,dy,dphi,n,m,Q)
% A : POSITIONS POINTS ANCRAGE  A=[x1 y1 ;x2 y2;x3 y3]
% T : DIMENSIONS TRIANGLES : T = [l2 L3 phi(deg)]
% Q : VECTEUR POSITIONS ARTICULAIRES Q=[q1 q2 q3]
% n : taille mesh
% m : Nombre entrées MF
% dX : Intervalle de recherche suivante ,
% dY : intervalle suivant dy et
% dphi  : intervalle angulaire
% M : POSITION [x,y,phi(deg)]
x1=A(1,1);
y1=A(1,2);
x2=A(2,1);
y2=A(2,2);
x3=A(3,1);
y3=A(3,2);
l2=T(1);
l3=T(2);
BETA=T(3)*pi/180;
l1=l2^2+l3^2-2*l3*l2*cos(BETA);
xx=linspace(dx(1),dx(2),n);
yy=linspace(dy(1),dy(2),n);
phi=linspace(dphi(1),dphi(2),n);
[XX,YY,PHI]=meshgrid(xx,yy,phi);
X2=x2*ones(n^3,1);X3=x3*ones(n^3,1);
Y2=y2*ones(n^3,1);Y3=y3*ones(n^3,1);
BET=BETA*ones(n^3,1);
Q1=sqrt(XX(:).^2+YY(:).^2);
Q2=sqrt((X2-XX(:)-l2*cos(BET)).^2+(YY(:)+l2*sin(BET)).^2);
Q3=sqrt((X3-XX(:)-l3*cos(BET+PHI(:))).^2+(Y3-YY(:)-l3*sin(BET+PHI(:))).^2);

%data1 = [Q1 Q2 Q3 XX(:)]; % création  Q1-Q2-Q3-XX  dataset
%data2 = [Q1 Q2 Q3 YY(:)]; % création  Q1-Q2-Q3-YY  dataset
%data3 = [Q1 Q2 Q3 PHI(:)]; % création  Q1-Q2-Q3-PHI dataset

 data11 = [Q1(1:2:end,:) Q2(1:2:end,:) Q3(1:2:end,:) (XX(1:2:end))']; % création  Q1-Q2-Q3-XX   dataset   training
data12 = [Q1(2:2:end,:) Q2(2:2:end,:) Q3(2:2:end,:) (XX(2:2:end))']; % création  Q1-Q2-Q3-XX   dataset   checking
data21 = [Q1(1:2:end,:) Q2(1:2:end,:) Q3(1:2:end,:) (YY(1:2:end))']; % création  Q1-Q2-Q3-YY   dataset   training
data22 = [Q1(2:2:end,:) Q2(2:2:end,:) Q3(2:2:end,:) (YY(2:2:end))']; % création  Q1-Q2-Q3-YY   dataset   checking
data31 = [Q1(1:2:end,:) Q2(1:2:end,:) Q3(1:2:end,:) (PHI(1:2:end))']; % création  Q1-Q2-Q3-PHI   dataset  training
data32 = [Q1(2:2:end,:) Q2(2:2:end,:) Q3(2:2:end,:) (PHI(2:2:end))']; % création  Q1-Q2-Q3-PHI  dataset   checking
initfit1= genfis1(data11,m,'gbellmf');
initfit2= genfis1(data21,m,'gbellmf');
initfit3= genfis1(data31,m,'gbellmf');
epochs_n = 30;
[fismat1,trnerr1,ss1,fismat1c,chkerr1]=anfis(data11,initfit1,epochs_n,[0 0 0 0],data12);
[fismat2,trnerr2,ss2,fismat2c,chkerr2]=anfis(data21,initfit2,epochs_n,[0 0 0 0],data22);
tic
[fismat3,trnerr3,ss3,fismat3c,chkerr3]=anfis(data31,initfit3,epochs_n,[0 0 0 0],data32);
toc
for ii=1:3
fismat1.input(ii).name=['Q',num2str(ii)];
fismat2.input(ii).name=['Q',num2str(ii)];
fismat3.input(ii).name=['Q',num2str(ii)];
end
fismat1.output.name='X';
fismat2.output.name='Y';
fismat3.output.name='PHI';
vect=[1 2;1 3;3 2;1 2;1 3;3 2;1 2;1 3;3 2];
nn=[1,2,3,1,2,3,1,2,3];
mm=[1,1,1,2,2,2,3,3,3];

for ii=1:3
   figure(ii)
   for jj=1:3
subplot(1,3,jj);
gensurf(eval(['fismat',num2str(ii)]),vect(jj,:),1);
set(ii,'color','w');
set(gca,'box','on','fontSize',10,'Fontweight','bold');
set(get(gca,'Xlabel'),'Fontweight','bold');
set(get(gca,'Ylabel'),'Fontweight','bold');
set(get(gca,'Zlabel'),'Fontweight','bold');
```

```
 end
end

%*************************************************************************************************************

%anfis1 = anfis(data1); % train first ANFIS network
%anfis2 = anfis(data2); % train second ANFIS network
%tic
%anfis3 = anfis(data3); % train second ANFIS network
%toc
%for ii=1:3
%anfis1.input(ii).name=['Q',num2str(ii)];
%anfis2.input(ii).name=['Q',num2str(ii)];
%anfis3.input(ii).name=['Q',num2str(ii)];
%end

%anfis1.output.name='X';
%anfis2.output.name='Y';
%anfis3.output.name='PHI';
 Nmf=m;
%vect=[1 2;1 3;3 2;1 2;1 3;3 2;1 2;1 3;3 2];
%nn=[1,2,3,1,2,3,1,2,3];
%mm=[1,1,1,2,2,2,3,3,3];
%TRACE FONCTIONS MF
for ii=1:3
      figure(ii+3)
            for jj=1:3
                  subplot(3,1,jj);
                        range=eval(['AFIS',num2str(ii),'.input(',num2str(jj),').range']);
                        xx=linspace(range(1),range(2),1000);
                        for kk=1:nMF
                              if kk==1
                             par=eval(['AFIS',num2str(ii),'.input(',num2str(jj),').mf(',num2str(kk),').params']);
                              a=par(1);
                              b=par(2);
                              c =par(3);
                              yy=1./(1.+(abs(xx-c)/a).^(2*b));
                              h=plot(xx,yy,'b');
                              set(h,'linewidth',3)
                              set(gca,'fontSize',8,'Fontweight','bold','box','on')
                              hold on
                              grid on
                              hx=xlabel('Q1')
                              else
                              par=eval(['AFIS',num2str(ii),'.input(',num2str(jj),').mf(',num2str(kk),').params']);
                              a=par(1);
                              b=par(2);
                              c =par(3);
                              yy=1./(1.+(abs(xx-c)/a).^(2*b));
                               subplot(3,1,jj);
                              h=plot(xx,yy,'r');
                              set(h,'linewidth',3)
                              set(gca,'fontSize',8,'Fontweight','bold','box','on')
                              grid on
                               end
                        end
hx=xlabel(['Q',num2str(jj)]);
set(hx,'fontweight','bold');
legend(['in',num2str(jj),'mf1'],['in',num2str(jj),'mf2']);
set(hx,'Fontweight','bold');
end
end
set(gcf,'color','w');
% CALCUL SOLUTION POUR ENTREE Q
M(1)=evalfis(Q,fismat1);
M(2)=evalfis(Q,fismat2);
M(3)=evalfis(Q,fismat3);
```

ANNEXE C2 : PREMIERE STRUCTURE ANFIS1 SORTIE X : : 2MF

[System]
Name='anfis1'
Type='sugeno'
Version=2.0
NumInputs=3
NumOutputs=1
NumRules=8
AndMethod='prod'
OrMethod='max'
ImpMethod='prod'
AggMethod='max'
DefuzzMethod='wtaver'

[Input1]
Name='Q1'
Range=[1.13137084989848 1.69705627484771]
NumMFs=2
MF1='in1mf1':'gbellmf',[0.292612235194262 1.9995755549105 1.13752625445892]
MF2='in1mf2':'gbellmf',[0.292110445645078 1.99950851183395 1.69175205218475]

[Input2]
Name='Q2'
Range=[2.8400071559866 3.39977366439042]
NumMFs=2
MF1='in2mf1':'gbellmf',[0.330980087834876 1.99889394150038 2.87969769092566]
MF2='in2mf2':'gbellmf',[0.331524520287308 1.9989584184735 3.35915919161512]

[Input3]
Name='Q3'
Range=[2.11776435066446 3.39977366439042]
NumMFs=2
MF1='in3mf1':'gbellmf',[0.641153120867244 1.99996615503896 2.11779685971911]
MF2='in3mf2':'gbellmf',[0.641285304920534 1.99997538264018 3.39958722064738]

[Output1]
Name='X'
Range=[0.8 1.2]
NumMFs=8
MF1='out1mf1':'linear',[0.635774434944807 -0.617934672772152 -0.00208082800835871 2.04604909683858]
MF2='out1mf2':'linear',[0.595716185161632 -0.638176644964861 -0.00117402434237554 2.15630570802109]
MF3='out1mf3':'linear',[0.547364391098737 -0.782933381253647 -0.00100703881143561 2.66439974319152]
MF4='out1mf4':'linear',[0.528686085487732 -0.773315616877531 -0.00121126253785542 2.65796428193019]
MF5='out1mf5':'linear',[0.67388688981788 -0.663959232664152 0.000935325731505734 2.10371121751462]
MF6='out1mf6':'linear',[0.653312378057267 -0.634666104254941 0.000867949029006901 2.05169626464253]
MF7='out1mf7':'linear',[0.621754901549701 -0.792611640907116 0.00120847735912372 2.59634849199036]
MF8='out1mf8':'linear',[0.58351370991761 -0.794911255805113 0.00171078643620215 2.6576746038131]

[Rules]
1 1 1, 1 (1) : 1
1 1 2, 2 (1) : 1
1 2 1, 3 (1) : 1
1 2 2, 4 (1) : 1
2 1 1, 5 (1) : 1
2 1 2, 6 (1) : 1
2 2 1, 7 (1) : 1
2 2 2, 8 (1) : 1

ANNEXE C3 : SECONDEE STRUCTURE ANFIS2 SORTIE Y : 2MF

[System]
Name='anfis2'
Type='sugeno'
Version=2.0
NumInputs=3
NumOutputs=1
NumRules=8
AndMethod='prod'
OrMethod='max'
ImpMethod='prod'
AggMethod='max'
DefuzzMethod='wtaver'

[Input1]
Name='Q1'
Range=[1.13137084989848 1.69705627484771]
NumMFs=2
MF1='in1mf1':'gbellmf',[0.299141279460712 1.99954235830746 1.14338406164624]
MF2='in1mf2':'gbellmf',[0.292328333774036 1.99924215754455 1.69358051089566]

[Input2]
Name='Q2'
Range=[2.8400071559866 3.39977366439042]
NumMFs=2
MF1='in2mf1':'gbellmf',[0.331117225991325 1.99877835857616 2.87883585635876]
MF2='in2mf2':'gbellmf',[0.330126855549728 1.99913128621097 3.35965191973663]

[Input3]
Name='Q3'
Range=[2.11776435066446 3.39977366439042]
NumMFs=2
MF1='in3mf1':'gbellmf',[0.641243102546115 1.99996070383854 2.1178615846668]
MF2='in3mf2':'gbellmf',[0.641337768816867 1.99996143399596 3.39957820193566]

[Output1]
Name='Y'
Range=[0.8 1.2]
NumMFs=8
MF1='out1mf1':'linear',[0.8702729353648 0.885547941916096 0.0052337450561803 -2.99779317162771]
MF2='out1mf2':'linear',[0.908074927929557 0.870183755325938 0.000890499807789886 -2.98723762380546]
MF3='out1mf3':'linear',[0.793444920049502 0.63227254619426 -0.000463364516373136 -2.10523319514498]
MF4='out1mf4':'linear',[0.818032224470001 0.639700133517917 0.0024323671880743 -2.16538661467184]
MF5='out1mf5':'linear',[0.774382421094582 0.762286851181414 -0.00291999434762494 -2.46904929378847]
MF6='out1mf6':'linear',[0.810466889696643 0.802629369082266 0.000551996421030527 -2.65335452842892]
MF7='out1mf7':'linear',[0.750971584405537 0.607230787097182 -0.000756650347913061 -1.94514787401024]
MF8='out1mf8':'linear',[0.783266842730383 0.625109581081353 -0.00467508670101739 -2.04905408906011]

[Rules]
1 1 1, 1 (1) : 1
1 1 2, 2 (1) : 1
1 2 1, 3 (1) : 1
1 2 2, 4 (1) : 1
2 1 1, 5 (1) : 1
2 1 2, 6 (1) : 1
2 2 1, 7 (1) : 1
2 2 2, 8 (1) : 1

ANNEXE C4 : TROISIEME STRUCTURE ANFIS3 SORTIE PHI : : 2MF

[System]
Name='anfis3'
Type='sugeno'
Version=2.0
NumInputs=3
NumOutputs=1
NumRules=8
AndMethod='prod'
OrMethod='max'
ImpMethod='prod'
AggMethod='max'
DefuzzMethod='wtaver'

[Input1]
Name='Q1'
Range=[1.13137084989848 1.69705627484771]
NumMFs=2
MF1='in1mf1':'gbellmf',[0.307027665589648 1.99966652697967 1.15172908699893]
MF2='in1mf2':'gbellmf',[0.28825725346597 1.99896666854045 1.69970514003119]

[Input2]
Name='Q2'
Range=[2.8400071559866 3.39977366439042]
NumMFs=2
MF1='in2mf1':'gbellmf',[0.319728617084349 1.99819314268019 2.86457327191487]
MF2='in2mf2':'gbellmf',[0.326170463720999 1.99959199729879 3.36016885240972]

[Input3]
Name='Q3'
Range=[2.11776435066446 3.39977366439042]
NumMFs=2
MF1='in3mf1':'gbellmf',[0.637655234766923 1.9956685413686 2.10181051698125]
MF2='in3mf2':'gbellmf',[0.669562016896446 2.00049376797142 3.3726715663907]

[Output1]
Name='PHI'
Range=[0.785398163397448 1.5707963267949]
NumMFs=8
MF1='out1mf1':'linear',[1.73698896340592 0.775382399895264 1.58907260379332 -7.87295168221017]
MF2='out1mf2':'linear',[0.961345405434135 0.306580604215183 1.030390346428 -4.0078354380659]
MF3='out1mf3':'linear',[1.21084272413243 0.0917389815191804 1.20725250665163 -4.12334854307864]
MF4='out1mf4':'linear',[0.921097121372049 -0.00202381018336596 0.993206675880491 -2.85016581939552]
MF5='out1mf5':'linear',[1.25192270970879 0.377703615136204 1.24817438014747 -5.15541073893423]
MF6='out1mf6':'linear',[0.94869061628918 0.221068323488853 1.11971301260128 -3.97899648195236]
MF7='out1mf7':'linear',[1.06112155052438 0.0105607692146305 1.13184581753682 -3.40387406811268]
MF8='out1mf8':'linear',[1.01510984879869 -0.0792142718530147 1.21160088963345 -3.37066370331802]

[Rules]
1 1 1, 1 (1) : 1
1 1 2, 2 (1) : 1
1 2 1, 3 (1) : 1
1 2 2, 4 (1) : 1
2 1 1, 5 (1) : 1
2 1 2, 6 (1) : 1
2 2 1, 7 (1) : 1
2 2 2, 8 (1) : 1

ANNEXE C5 :PREMIERE STRUCTURE ANFIS1 SORTIE X : :32MF

[System]
Name='fismat1'
Type='sugeno'
Version=2.0
NumInputs=3
NumOutputs=1
NumRules=27
AndMethod='prod'
OrMethod='max'
ImpMethod='prod'
AggMethod='max'
DefuzzMethod='wtaver'

[Input1]
Name='Q1'
Range=[1.13137084989848 1.69705627484771]
NumMFs=3
MF1='in1mf1':'gbellmf',[0.186976562151165 1.99886517276512 1.16200428813183]
MF2='in1mf2':'gbellmf',[0.195743724560223 2.00046368562511 1.41200042010617]
MF3='in1mf3':'gbellmf',[0.185345468735838 1.99937249038106 1.65832169340373]

[Input2]
Name='Q2'
Range=[2.8400071559866 3.39977366439042]
NumMFs=3
MF1='in2mf1':'gbellmf',[0.194193085703225 1.99715293331261 2.86502532140668]
MF2='in2mf2':'gbellmf',[0.198338033489649 1.99913295192428 3.11877560403719]
MF3='in2mf3':'gbellmf',[0.189897433605816 1.99814006419994 3.36428462029541]

[Input3]
Name='Q3'
Range=[2.11776435066446 3.39977366439042]
NumMFs=3
MF1='in3mf1':'gbellmf',[0.315703588753658 2.00017287795872 2.12271820139806]
MF2='in3mf2':'gbellmf',[0.320100650937797 1.99999431305839 2.76238845304297]
MF3='in3mf3':'gbellmf',[0.31611321455864 2.00015720487923 3.40159687415189]

[Output1]
Name='X'
Range=[0.8 1.2]
NumMFs=27
MF1='out1mf1':'linear',[0.669841615342268 -0.344280812156547 0.099728236063055 1.05479195218452]
MF2='out1mf2':'linear',[0.61374634793602 -0.550843352623667 0.00421583191100562 1.87329095075893]
MF3='out1mf3':'linear',[0.88099754871791 -0.360432464907213 0.00558735250549338 1.07699594570637]
MF4='out1mf4':'linear',[0.47730832836782 -0.103475775491586 -0.0393400215345871 0.711095572624436]
MF5='out1mf5':'linear',[0.523264825365292 -0.657562309884702 -0.00123481075467052 2.29423772998155]
MF6='out1mf6':'linear',[0.370447504533672 -0.292051991460374 -0.00317090720605022 1.31665071898132]
MF7='out1mf7':'linear',[0.703705115305846 -0.524135341740326 0.0461065379368025 1.46396775466058]
MF8='out1mf8':'linear',[0.579656308057527 -0.803713070638916 0.00116781998984474 2.69342760386061]
MF9='out1mf9':'linear',[0.81501793017261 -0.682874130296881 0.00460489975565211 1.99548815236859]
MF10='out1mf10':'linear',[0.882087998090871 -0.673143643249547 -0.0122246090877736 1.86679090298946]
MF11='out1mf11':'linear',[0.66507546487466 -0.626279060491136 -6.83395108171561e-006 2.01614554023163]
MF12='out1mf12':'linear',[1.09048829825821 -0.726754497821782 -0.00248299933105583 1.69176191126254]
MF13='out1mf13':'linear',[0.467347934088891 -0.88863017080748 0.00494478191443154 3.09380121754422]
MF14='out1mf14':'linear',[0.558343447014696 -0.748033030225354 -0.000191104050728682 2.54197367405859]
MF15='out1mf15':'linear',[0.323506791495486 -0.917613927519138 0.00363387147748432 3.3964700086634]
MF16='out1mf16':'linear',[0.871654563044804 -0.896926508037543 -0.00718608500669454 2.6306280051406]
MF17='out1mf17':'linear',[0.659608472072915 -0.83400485785865 0.00043331762490161 2.6852071991579]
MF18='out1mf18':'linear',[1.02843042474336 -0.87987781379666 -0.00665852734212626 2.35381169671434]
MF19='out1mf19':'linear',[0.838353631146505 -0.511970151671547 0.0053148191296121 1.38864300399458]
MF20='out1mf20':'linear',[0.677536815271381 -0.602146420265932 0.000257954821556303 1.92299370837664]
MF21='out1mf21':'linear',[0.851665318661971 -0.286456934335862 0.0145708515068411 0.689394382027974]
MF22='out1mf22':'linear',[0.576144337547922 -0.456573378709201 -0.00309603077972637 1.63963295170364]
MF23='out1mf23':'linear',[0.636462565696805 -0.678037988296858 -0.000528906976091689 2.21650578859684]
MF24='out1mf24':'linear',[0.531753697225722 -0.175747808051429 -0.0216535630921103 0.899879063641966]
MF25='out1mf25':'linear',[0.802958517810701 -0.705311681581462 0.0059365975893554 1.94568264131331]
MF26='out1mf26':'linear',[0.677134185667875 -0.797493599577687 0.00107343040716776 2.51379153307385]
MF27='out1mf27':'linear',[0.872706225015486 -0.629262049988767 0.0460965449325135 1.41159803509907]

[Rules]
1 1 1, 1 (1) : 1
1 1 2, 2 (1) : 1
1 1 3, 3 (1) : 1
1 2 1, 4 (1) : 1

1 2 2, 5 (1) : 1
1 2 3, 6 (1) : 1
1 3 1, 7 (1) : 1
1 3 2, 8 (1) : 1
1 3 3, 9 (1) : 1
2 1 1, 10 (1) : 1
2 1 2, 11 (1) : 1
2 1 3, 12 (1) : 1
2 2 1, 13 (1) : 1
2 2 2, 14 (1) : 1
2 2 3, 15 (1) : 1
2 3 1, 16 (1) : 1
2 3 2, 17 (1) : 1
2 3 3, 18 (1) : 1
3 1 1, 19 (1) : 1
3 1 2, 20 (1) : 1
3 1 3, 21 (1) : 1
3 2 1, 22 (1) : 1
3 2 2, 23 (1) : 1
3 2 3, 24 (1) : 1
3 3 1, 25 (1) : 1
3 3 2, 26 (1) : 1
3 3 3, 27 (1) : 1