REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT GENIE ELECTRIQUE

PROJET DE FIN D'ETUDE EN VUE DE L'OBTENTION DU DIPLOME D'INGENIEUR D'ETAT

REALISATION ET COMMANDE D'UN BRAS MANIPULATEUR PAR L'INTERFACE D'AXE TURBO UMAC

Proposé et dirigé par:

Mr STIHI Omar
Mr RIACH

REALISE PAR :

Salih ABDELAZIZ
Salim ISSAOUNI

Année universitaire : 2004/2005

REMERCIEMENTS

Le travail présenté dans le cadre de cette thèse a été réalisé au laboratoire de commande des processus de l'Ecole Nationale Polytechnique.

Nous exprimons notre profonde gratitude à monsieurs O.STIHI et L.RIACH professeur à l'ENP pour avoir assumé la responsabilité de nous encadrer, nous orienter et de nous conseiller tout au long de la réalisation de ce travail ainsi pour la confiance qu'ils nous ont accordée.

Nous remercions très chaleureusement les membres du jury pour l'honneur qu'ils nous ont fait en acceptant d'être rapporteurs de notre mémoire.

Nos vifs remerciements à tous nos enseignants du primaire à l'université.

Nous tenons à remercier toutes personnes qui nous ont aidé dans notre cursus et spécialement les agents du génie mécanique.

Je dédie ce modeste travail :

À Celle à qui mon coeur depuis sa naissance n'a pas pu éprouver qu'amour et reconnaissance, à celle qui a donné un sens à mon existence en m'offrant une éducation digne de confiance

À ma chère Mère.

À mon père, pour son amour et son soutien moral depuis mon enfance.

Père merci.

À la mémoire de mes chères grande mère.

A tout les membres de ma famille ; ma soeurs, mon frères, mes deux nièces,

Mon beau frère, ma belle sœur, ma fiancée.

À la famille ABDELAZIZ.

À toutes ma famille.

À mes meilleurs amis :

Toufik, Belhadj, Bachir, Yles, Nadir, farès, Hadj omar, Yacine, A.madjid.

À mes amis de la cité universitaire BOURAOUI Amar,

et à tous qui m'ont connu et aidé de prés et de loin dans la réalisation de ce travail.

Salih

Je dédie ce modeste travail :

À Celle à qui mon coeur depuis sa naissance n'a pas pu éprouver qu'amour et reconnaissance, à celle qui a donné un sens à mon existence en m'offrant une éducation digne de confiance

À ma chère Mère.

À mon père, pour son amour et son soutien moral depuis mon enfance.

Père merci.

À la mémoire de mes chères grande mère.

A tout les membres de ma famille ; mes soeurs, mes frères

À la famille ISSAOUNI

À la famille MAZOUZ sourtout mon grand père 'TAHAR'

À la famille 'GACHTALLE '

À tout mes amis de Bouandas, surtout les membre de groupe SMA

À mes amis de la cité universitaire BOURAOUI Amar,

À tout les membres de l'association « EL-MAARIFA »

À Mme. 'DIAB Djaouida' l'infirmerie de l 'ENP qui m'aide dans toute la période de ma maladie et tout le groupe de centre médical.

À Mr. 'Mustapha Meziani' .

À MON AMI 'TAFARGANIT Fayçel '

Et à tous qui m'ont connu et aidé de prés et de loin dans la réalisation de ce travail.

Salim

Sommaire

CHAPITRE IV. ***DESCRIPTION DU LANGUAGE DE PROGRAMMATION DE L'UMAC***

CHAPITRE V. ***ASSERVISSEMENT DU MOTEUR A COURANT CONTINUE***

Introduction

Etymologiquement, le mot robot a connu plusieurs définitions selon différentes langues, le mot « robot » tire sa racine du Bulgare « robu » qui signifie « serviteur », donnant naissance au mot russe « robota » qui signifie « travail » et au mot tchèque « robota » qui se traduit par « travail forcé ».

Vers les années vingt (1920), l'écrivain tchèque Karel CAPEK a popularisé le terme «robot » à travers une pièce théâtrale intitulée « Rossum's Universal Robots », elle met en scène des petits êtres artificiels qui répondent aux ordres de leurs maîtres comme des esclaves, de cela le mot « robot » a pris sa véritable signification.

Les premières applications de la robotique étaient la saisie et le transfert d'objets, le moulage par injection et l'emboutissage où le robot est simplement équipé d'une presse pour décharger et transférer ou empiler la partie finie. Ces premiers robots étaient capables d'être programmés pour exécuter une séquence de mouvement, comme aller à un emplacement donné, fermer une pince, etc. mais n'avaient pas de capteurs externes. Certaines applications plus compliquées, comme la soudure, le meulage, l'ébavurage, et l'assemblage exigent non seulement un mouvement plus complexe mais également un certain nombre de capteurs externes comme la vision et les capteurs de force vu la forte interaction du robot avec son environnement. Aujourd'hui les robots ont pu remplacé l'homme dans son travail directement en ayant la possibilité d'effectuer la quasi-totalité des travaux industriels.

Le robot industriel est un instrument sophistiqué, dont le rôle est principalement la manipulation de pièces ou d'outils. Sa conception fait appel à des compétences dans divers domaines tels que la mécanique, l'électrotechnique ou l'hydraulique, l'informatique, l'électronique numérique, etc…

Les progrès réalisées dans un seul de ces domaines permettant d'améliorer les performances de l'ensemble: de ce fait, les besoins de l'industrie qui étaient faibles il y a quelques années ne cessent d'augmenter avec simultanément un élargissement des domaines d'utilisation.

Notre mémoire s'articule autour de cinq chapitres :

CHAPITRE I. ***TERMINOLOGIE ET DEFINTIOINS GENERALES.***

CHAPITRE II. ***MODELISATION DU ROBOT.***

CHAPITRE III. ***DESCRIPTION MATERIELLE DE L'UMAC.***

CHAPITRE IV. ***DESCRIPTION DU LANGUAGE DE PROGRAMMATION DE L'UMAC.***

CHAPITRE V. ***ASSERVISSEMENT DU MOTEUR A COURANT CONTINUE***

I.1.INTRODUCTION

Selon l'A.F.R.I., Association Française de Robotique Industrielle, il convient de distinguer les manipulateurs et les robots industriels selon la classification suivante:

- Manipulateur manuel : il s'agit d'un engin de manipulation motorisé, commandé par l'homme, ayant au moins quatre degrés de liberté. Cela peut être un manipulateur d'assistance musculaire, comme une girafe de déplacement de charge, ou un télémanipulateur, piloté à distance pour la manutention en ambiance dangereuse, nucléaire ou subaquatique par exemple.
- Manipulateur automatique : c'est un engin manipulateur de deux axes ou plus, non asservis, à cycle automatique. Il peut être à séquence fixe ou variable. On peut citer comme exemple les manipulateurs pneumatique de chargement/déchargement de machine-outil.
- Robot programmable : c'est un manipulateur automatique de trois axes ou plus, dont au moins deux sont programmables par apprentissage et/ou par langage symbolique.
- Robot dit intelligent : il s'agit alors d'un manipulateur automatique programmable par capable d'analyser les modifications de son environnement et de réagir en conséquence, à l'exclusion des modifications triviales par des capteurs tout ou rien.

Seules les deux dernières classes entrent dans la catégorie des robots industriels. Quand à la norme AFNOR, elle donne la définition suivante du robot industriel :

"Manipulateur automatique, asservi en position, reprogrammable, polyvalent, capable de positionner et d'orienter des matériaux, pièces, outils ou dispositifs spécialisés au cours de mouvements variables et programmés pour l'exécution de tâches variées.

Il se présente souvent sous la forme d'un ou plusieurs bras (articulations) se terminant par un poignet. Son unité de commande utilise, notamment, un dispositif de mémoire et éventuellement de perception de l'environnement et des circonstances ainsi que d'adaptation en résultant.

Ces machines polyvalentes sont généralement étudiées pour effectuer la même fonction de façon cyclique et peuvent être adaptées à d'autres fonctions sans modification permanente du matériel. "

Les robots manipulateurs peuvent être classés en deux catégories : ceux qui ont une structure de chaîne cinématique ouverte et ceux qui ont une structure de chaîne cinématique fermée.

Dans ce chapitre, nous rappelons, les classes des mécanismes des robots manipulateurs. Après cela, nous procédons à la définition de quelques notions de base de la robotique.

I.2.CONSTITUANTS MECANIQUES DES ROBOTS [2]

Un robot manipulateur est constitué par deux sous-ensembles distincts, un (ou plusieurs) organes terminaux et une structure mécanique articulée :

- sous le terme organe terminal, on regroupe tout dispositif destiné à manipuler des objets (dispositifs de serrage, dispositifs magnétiques, à dépression ...) ou à les transformer (outils, torche de soudage, pistolet de peinture...). Il s'agit donc d'une interface permettant au robot d'interagir avec son environnement.
- Le rôle de la structure mécanique articulée est d'amener l'organe terminal dans une situation (position et orientation) donnée, selon des caractéristiques de vitesse et d'accélération données. Son architecture est une chaîne cinématique de corps généralement rigides, assemblés par des liaisons appelées articulations. Les chaînes peuvent être soit ouvertes simples (figure 1.1), soit arborescentes (figure 1.2), soit fermées (figure 1.3).

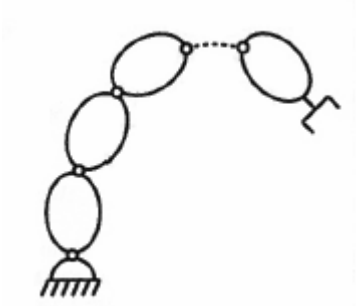


Figure I.1 *:* Structure ouverte simple**.**

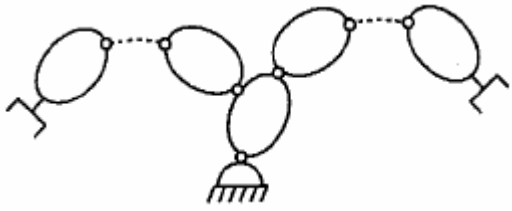


Figure I.2 : Structure arborescente.

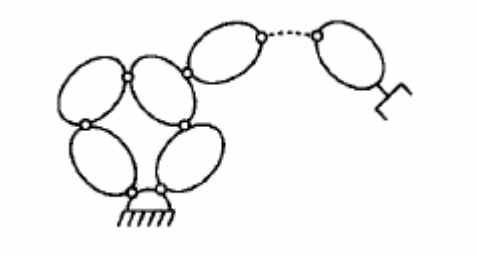


Figure I.3 : Structure fermée.

La figure 1.4 montre une génération différente des robots – les robots parallèles- dans laquelle l'organe terminal est relié à la base du mécanisme par plusieurs chaînes parallèles.

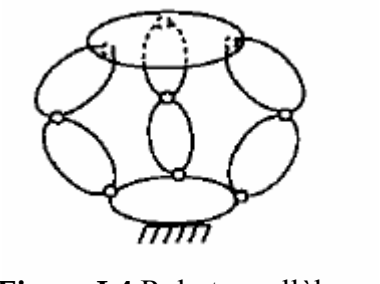


Figure I.4 Robot parallèle.

I.3.LES DIFFERENTES MODELESATIONS D'UN ROBOT MANIPULATEUR [1][3]

Un robot manipulateur, quelle que soit la fonction qui lui est attribuée, doit positionner et orienter son organe terminal dans l'espace opérationnel, car le concepteur perçoit de façon naturelle, par la vision, la tâche à accomplir dans cet espace. La difficulté de la commande provient du fait que les actionneurs, dont est doté le robot manipulateur, n'agissent pas directement sur la situation de l'organe terminal, mais agissent sur la configuration du robot manipulateur. D'où la nécessité de définir les passages entre les espaces opérationnel et généralisé (articulaire).

Ainsi nous pouvons distinguer les modèles suivants :

- Le modèle géométrique direct, qui consiste à calculer les coordonnées opérationnelles X , en fonction des coordonnées généralisées q ; ce modèle est noté : $X = f(q)$.
- Le modèle géométrique inverse, qui consiste à calculer les coordonnées généralisées q , en fonction des coordonnées généralisées X ; ce modèle est noté : $q = g(X)$.

- Le modèle cinématique direct, qui consiste à calculer les vitesses des coordonnées opérationnelles $\dot{X}$, en fonction de vitesses des coordonnées généralisées $\dot{q}$; ce modèle est noté : $\dot{X} = J\dot{q}$ (J est la matrice jacobienne du robot manipulateur).
- Le modèle cinématique inverse, qui consiste à calculer les vitesses des coordonnées généralisées $\dot{q}$, en fonction de vitesses des coordonnées opérationnelles $\dot{X}$; ce modèle est noté : $\dot{q} = J^{-1}\dot{X}$ (J^{-1} est la matrice jacobienne inverse du robot manipulateur).
- Les modèles dynamiques direct et inverse, qui relient les coordonnées, vitesses et accélérations généralisées aux efforts généralisés (ces efforts généralisés sont les forces appliquées au niveau des liaisons prismatiques ou les couples au niveau des liaisons rotoïdes). Ces modèles tiennent compte des efforts extérieurs appliqués sur l'organe terminal.

I.4.DEFINITIONS GENERALES [2]

I.4.1.ARTICULATIONS[2]

Une articulation lie deux corps successifs en limitant le nombre de degrés de liberté de l'un par rapport à l'autre. Soit m le nombre de degrés de liberté résultant, encore appelé mobilité de l'articulation. La mobilité est telle que $0 \leq m \leq 6$.

Lorsque m=1, ce qui est le cas le plus fréquent en robotique, l'articulation est soit rotoïde, soit prismatique.

I.4.1.1.Articulation rotoïde [2]

Il s'agit d'une articulation de type pivot réduisant le mouvement entre deux corps à une rotation autour d'un axe qui leur est commun. La situation relative entre les deux corps est donnée par l'angle autour de cet axe. L'articulation rotoïde est représentée ci-joint dans la figure (figure 1.5). Le symbole R caractérise cette articulation.

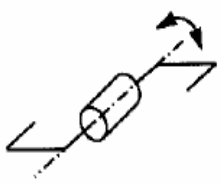


Figure I.5 : Symbole de l'articulation rotoïde **.**

I.4.1.2.Articulation prismatique [2]

Il s'agit d'une articulation de type glissière réduisant le mouvement entre deux corps à une translation le long d'un axe commun. La situation relative entre les deux corps est mesurée par la distance le long de cet axe. La figure (figure1.6) ci-contre donne sa représentation symbolique notée P.

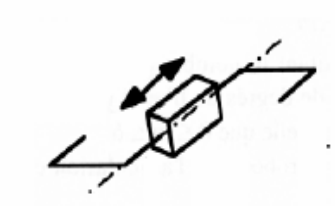


Figure I.6 : Symbole de la liaison prismatique**.**

I.4.2. ESPACE ARTICULAIRE [2]

L'espace articulaire d'un robot est celui dans lequel est représentée la situation de tous ses corps. La solution la plus simple consiste à utiliser les variables ou coordonnées articulaires. L'espace de ces variables, noté $\mathbf{IR^q}$, est appelé aussi espace des configurations. Sa dimension N est égale au nombre de variables articulaires indépendantes et correspond au nombre de degrés de liberté de la structure mécanique. Dans une structure ouverte (simple ou arborescente), les variables articulaires sont généralement indépendantes, tandis qu'une structure fermée impose nécessairement des relations entre ces variables.

Sauf mention particulière, on considère dans la suite de cet ouvrage qu'un robot à N degrés de liberté dispose de N articulations motorisées

I.4.3.ESPACE OPERATIONNEL [2]

L'espace opérationnel est celui dans lequel est représentée la situation de l'organe terminal (on considère donc autant d'espaces opérationnels qu'il y a d'organes terminaux). La solution la plus simple consiste à utiliser les coordonnées cartésiennes dans $\mathbf{IR^3}$ pour la position et le groupe **SO(3)** des rotations propres de $\mathbf{IR^3}$ pour l'orientation. On note $\mathbf{IR^X}$ l'ensemble de ces deux espaces, égal à $\mathbf{IR^3}$ x **SO(3)**. On remarquera que l'espace opérationnel n'est pas un espace vectoriel. La dimension M de $\mathbf{IR^X}$ constitue le nombre de degrés de liberté maximum que peut

avoir l'organe terminal et est au nombre de paramètres nécessaires pour décrire la situation de l'organe terminale dans l'espace. Dans l'espace tridimensionnel ce nombre est de six (trois pour placer un point du corps en un point quelconque de cet espace et trois pour orienter ce corps de façon quelconque) . On peut donc en conclure que $M \leq 6$ et $M \leq N$.

I.5. LES NOTATIONS ET LES OUTILS MATHEMATIQUES MIS EN ŒUVRE POUR LA MODELISATION DES ROBOTS [2]

En robotique, on associe à tout élément du poste de travail un ou plusieurs repères. Ces repère sont généralement définis de telle sorte que leurs axes et leurs origines correspondent respectivement à des directions et à des points privilégies ayant un rôle fonctionnel lors de l'exécution de la tache.

Ils permettent de situer dans l'espace les objets fixes de l'environnement (distributeurs, dispositifs de bridage,…) ainsi que les corps mobiles constitutifs de robot ou transportés par lui.

La notion de transformation de repère est donc fondamentale. Elle permet :

- d'exprimer les situations des différents corps du robot les uns par rapport aux autres ;
- de spécifier les situations que doit prendre le repère associe à l'origine terminale du robot pour réaliser une tache donnée ainsi que les vitesses correspondantes ;
- de décrire et de contrôler les efforts mis en jeu lorsque le robot interagit avec son environnement ;
- d'intégrer à la commande les informations sensorielles issues de capteur ayant chacun son système de référence propre.

Nous présentons dans ce chapitre une notation qui permet de décrire de façon homogène les différents systèmes de coordonnées. Cette notation, les transformations homogènes, est la plus répandue en robotique.

On montre comment représenter de cette façon un point, un vecteur libre et les transformations entre repères.

I.5.1.COORDONNEES HOMOGENES [2]

I.5.1.1.Représentation d'un point [2]

Soit P un point de coordonnées cartésiennes p_x, p_y, p_z. (figure1.7). On appelle coordonnées homogènes du point P les termes $w.p_x, w.p_y, w.p_z$, et w où w est un facteur d'échelle, égal à 1 en robotique. On représente alors les coordonnées homogènes d'un point par le vecteur :

$$P = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \qquad \textbf{(1.1)}$$

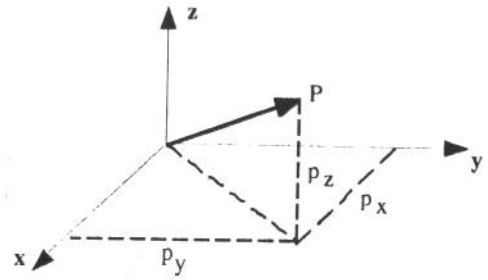


Figure I.7 : Représentation d'un point

I.5.1.2.Représentation d'une direction [2]

La représentation d'une direction (vecteur libre) se fait aussi par quatre composantes, mais la quatrième est nulle, indiquant un point à l'infini. Si l'on note u_x, u_y, u_Z les coordonnées cartésiennes d'un vecteur unitaire **u**, en coordonnées homogènes on écrit :

$$u = \begin{bmatrix} u_x \\ u_y \\ u_z \\ 0 \end{bmatrix} \qquad \textbf{(1.2)}$$

I.5.1.3.Représentation d'un plan [2]

Le plan $\alpha x + \beta y + \gamma z + \delta = 0$ est représenté par un vecteur ligne **Q** :

$$Q = \begin{bmatrix} \alpha & \beta & \gamma & \delta \end{bmatrix} \qquad \textbf{(1.3)}$$

Pour tout point **p** appartenant au plan **Q,** le produit matriciel **Qp** est nul :

$$QP = \begin{bmatrix} \alpha & \beta & \gamma & \delta \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = 0 \qquad \textbf{(1.4)}$$

I.5.2.TRANSFORMATIONS HOMOGENES [2] [4]

I.5.2.1.Transformation des repères [2][4]

Faisons subir une transformation quelconque, de translation et/ou de rotation. Au repère $\mathbf{R_i}$, transformation qui l'amène sur le repère $\mathbf{R_j}$ (figure 1.8). Cette transformation est définie par la matrice ${}^{i}\mathbf{T_j}$, appelée matrice de transformation homogène. de dimension (4x4), telle que :

$${}^{i}T_j = \begin{bmatrix} {}^{i}s_j & {}^{i}n_j & {}^{i}a_j & {}^{i}p_j \end{bmatrix} = \begin{bmatrix} s_x & n_x & a_x & p_x \\ s_y & n_y & a_y & p_y \\ s_z & n_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(1.5)}$$

Où ${}^{i}s_j$; ${}^{i}n_j$; ${}^{i}a_j$ désignent respectivement les vecteurs unitaires suivant les axes $\mathbf{x}_j$. $\mathbf{y}_j$ et $\mathbf{z}_j$ du repère $\mathbf{R_j}$ exprimés dans le repère $\mathbf{R_j}$; et où ${}^{i}p_j$ est le vecteur exprimant l'origine de repère $\mathbf{R_j}$ dans le repère le repère $\mathbf{R_i}$.

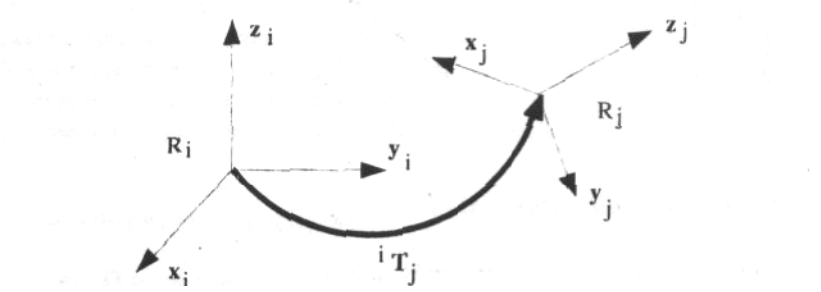


Figure I.8. Transformation des repères

On dit également que la matrice ${}^{i}T_{j}$ définit le repère $\mathbf{R_j}$ dans le repère $\mathbf{R_i}$. Par la suite, on notera souvent la matrice de transformation (1.5) sous forme d'une matrice partitionnée :

$$ {}^{i}T_{j} = \begin{bmatrix} {}^{i}A_{j} & {}^{i}P_{j} \\ 0\ \ 0\ \ 0\ 1 \end{bmatrix} = \begin{bmatrix} {}^{i}S_{j} & {}^{i}n_{j} & {}^{i}a_{j} & {}^{i}p_{j} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(1.6)} $$

Il s'agit d'un abus de notation, puisque les vecteurs n'ont alors que trois composantes, mais cette description, en isolant la matrice d'orientation ${}^{i}A_{j}$, est parfois plus pratique. De toute manière, la distinction entre représentations à trois ou à quatre composantes sera toujours claire dans le contexte.

I.5.2.2.Transformation des vecteurs [2][4]

Soit un vecteur ${}^{j}p_{1}$ définissant le point P_1 dans le repère $\mathbf{R_j}$ (figure 1.9). Compte tenu de la définition des coordonnées homogènes, on calcule les coordonnées du point P_1 dans le repère $\mathbf{R_i}$; grâce à l'équation suivante :

$$ {}^{i}p_{1} = {}^{i}(o_i p_1) = {}^{i}s_{j}\ {}^{j}p_{1_x} + {}^{i}n_{j}\ {}^{j}p_{1_y} + {}^{i}a_{j}\ {}^{j}p_{1_z} + {}^{i}p_{j} = {}^{i}T_{j}\ {}^{j}p_{1} \qquad \textbf{(1.7)} $$

La matrice ${}^{i}T_{j}$ permet donc d'exprimer dans le repère $\mathbf{R_i}$ les coordonnées d'un point donné dans le repère $\mathbf{R_j}$.

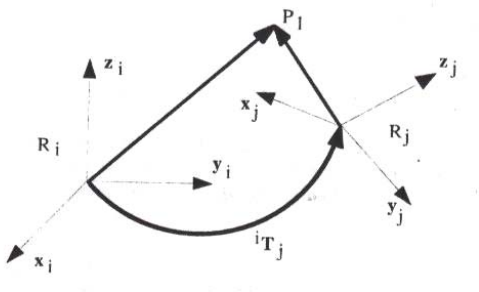


Figure I.9: Transformation d'un vecteur

I.5.2.3.Transformation des plans[2][4]

La position relative d'un point par rapport à un plan est indépendante des transformation appliquée à l'ensemble (point, plan).Ainsi :

$$ {}^{j}\mathrm{Q}\ {}^{j}P = {}^{i}\mathrm{Q}\ {}^{i}P = {}^{i}\mathrm{Q}\,{}^{i}T_{j}\ {}^{j}P \qquad \textbf{(1.8)} $$

Il s'ensuit que :

$$ {}^{j}\mathrm{Q} = {}^{j}\mathrm{Q}\,{}^{i}T_{j} \qquad \textbf{(1.9)} $$

I.5.2.4 Matrice de transformation de translation pure [2][4]

Soit **Trans**(a, b, c) cette transformation, où a, b, et c désignent les composantes de la translation le long des axes x, y et z respectivement. L'orientation étant conservée dans cette transformation, **Trans**(a, b, c) a pour expression (Figure1.10) :

$$ {}^{i}T_{j} = Trans\ (a,b,c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(1.10)} $$

Par la suite, on utilisera aussi la notation **Trans**(u, d) pour désigner une translation d'une valeur d le long d'un axe u. Ainsi, la matrice **Trans**(a,b, c) peut être décomposée en un produit de trois matrices **Trans**(x, a) **Trans**(y, b) **Trans**(z, c), l'ordre des multiplications étant quelconque.

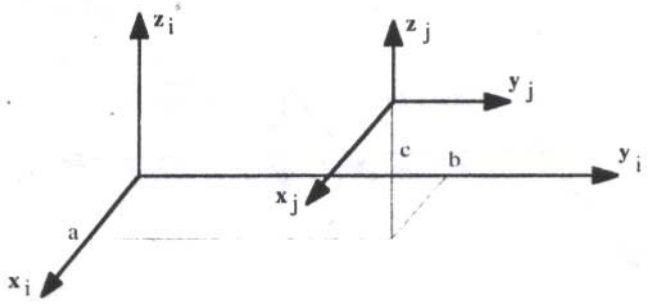


Figure I.10 : Transformation de translation

I.5.2.5.Matrice de transformation de rotation autour des axes principaux[2][4]

I.5.2.5.1. Matrice de transformation correspondant à une rotation θ autour l'axe x

Soit Rot(x, 0) cette rotation. On déduit de la figure 1.11 les composantes des vecteurs unitaires ${}^{i}s_j, {}^{i}n_j, {}^{i}a_j$ portés respectivement par les axes x_j,y_j et z_j du repère $\mathbf{R_j}$ et exprimés dans $\mathbf{R_i}$. Si l'on note $S\theta$ et $C\theta$ les sinus et cosinus de θ respectivement, elles s'écrivent :

$$\begin{cases} {}^{i}s_j = [1 \quad 0 \quad 0 \quad 0]^T \\ {}^{i}n_j = [0 \quad C\theta \quad S\theta \quad 0]^T \\ {}^{i}a_j = [0 \quad -S\theta \quad C\theta \quad 0]^T \end{cases} \tag{1.11}$$

L'exposant T désignant la transposition. On obtient alors :

$${}^{i}T_j = Rot(x,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\theta & -S\theta & 0 \\ 0 & S\theta & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & rot(x,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.12}$$

rot(x, θ) désignant la matrice d'orientation de dimension (3x3).

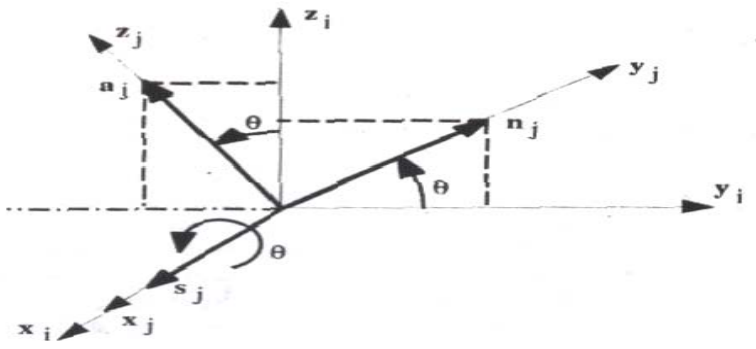


Figure I.11 : Transformation de rotation pure autour de l'axe X

I.5.2.5.2.Matrice de transformation correspondant à une rotation θ autour de l'axe y

Avec un résonnement analogue , on obtient :

$$ {}^{i}T_{j} = Rot(y,\theta) = \begin{bmatrix} C\theta & 0 & S\theta & 0 \\ 0 & 1 & 0 & 0 \\ -S\theta & 0 & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & rot(y,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(1.13)} $$

I.5.2.5.3.Matrice de transformation correspondant à une rotation θ autour de l'axe z

On vérifier que :

$$ {}^{i}T_{j} = Rot(z,\theta) = \begin{bmatrix} C\theta & -S\theta & 0 & 0 \\ S\theta & C\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & rot(z,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(1.14)} $$

I.5.2.6.Proprieté des matrices de transformation homogène [2]

- Une matrice de transformation peut se mettre, d'après la relation, sous la forme :

$$ T = \begin{bmatrix} s_x & n_x & a_x & P_x \\ s_y & n_y & a_y & P_y \\ s_z & n_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & A & & P \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(1.15)} $$

La matrice A représente la rotation alors que la matrice colonne **P** représente la translation. Pour une transformation de translation pure, $\mathbf{A = I_3}$ ($\mathbf{I_3}$ représente la matrice unité d'ordre 3), tandis que pour une transformation de rotation pure, **P = 0**. Les éléments de la matrice **A** représentent les cosinus directeurs. Elle ne contient que trois paramètres indépendants (l'un des vecteurs s, n ou a se déduit du produit vectoriel des deux autres ; par exemple, $s = n \times a$; en outre, le produit scalaire $n.a$ est nul et les normes de **n** et de **a** sont égales à 1).

- La matrice **A** est orthogonale, c'est-à-dire que ta matrice inverse est égale à la matrice transposée : $A^{-1} = A^T$ **(1.16)**
- L'inverse de la matrice ${}^{i}T_{j}$ définit la matrice ${}^{j}T_{j}$.

L'expression **(1.7)** donne les composantes du vecteur ${}^{j}p_1$ dans le repère **R_i**;. Inversement, pour exprimer les composantes du vecteur ${}^{i}p_1$ dans le repère Ri, on écrit :

$${}^{j}P_1 = {}^{j}T_i \, {}^{i}P_1 \quad \textbf{(1.17)}$$

On peut aussi multiplier à gauche les termes de l'expression **(1.7)** par ${}^{i}T_j^{-1}$ l'inverse de ${}^{i}T_j$:

$${}^{i}T_j^{-1} \, {}^{i}P_1 = {}^{j}P_1 \quad \textbf{(1.18)}$$

On déduit des relations **(1.17)** et **(1.18)** :

$${}^{j}T_{j-1} = {}^{j}T_i \quad \textbf{(1.19)}$$

- On vérifie aisément que :

$$Rot^{-1}(u,\theta) = Rot(u,-\theta) = Rot(-u,\theta) \quad \textbf{(1.20)}$$

$$Trans^{-1}(u,d) = Trans(-u,d) = Trans(u,-d) \quad \textbf{(1.21)}$$

- L'inverse d'une matrice de transformation représentée par la relation **(1.15)** peut être calculée par :

$$T^{-1} = \begin{bmatrix} & & & -s^T P \\ & A^T & & -n^T P \\ & & & -a^T P \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & A^T & & -A^T P \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(1.22)}$$

- **Composition de deux matrices**

La multiplication de deux matrices de transformation donne une matrice de transformation :

$$T_1T_2 = \begin{bmatrix} & A_1 & & P_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} & A_2 & & P_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & A_1A_2 & & A_1P_2 + P_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(1.23)}$$

II importe de se rappeler à cc propos que le produit de deux matrices de transformation n'est pas commutatif ($T_1T_2 \neq T_2T_1$) .

- Si un repère $\mathbf{R_o}$ a subi **k** transformations consécutives **(figure I.12)** et si chaque transformation **i**, (**i=1, ..., k**), est définie par rapport au repère courant $\mathbf{R_{i-1}}$; alors la transformation ${}^0T_k$ peut être déduite de la composition des multiplications à droite de ces transformations :

$${}^0T_k = {}^0T_1\,{}^1T_2\,{}^2T_3 \ldots\ldots {}^{k-1}T_k \quad \textbf{(1.24)}$$

- Si un repère $\mathbf{R_j}$ défini dans le repère $\mathbf{R_i}$ par la transformation ${}^iT_j$ subit une transformation **T** exprimée dans le repère $\mathbf{R_i}$, le repère $\mathbf{R_j}$ se transforme en $\mathbf{R_{j'}}$, **(figureI.13)** avec :

$${}^iT_j = T\,{}^iT_j \quad \textbf{(1.25)}$$

A partir des deux derniers propriétés, on déduit que :

- une multiplication à droite de la transformation ${}^iT_j$ signifie que la transformation est faite par rapport au repère courant $\mathbf{R_j}$;
- une multiplication à gauche signifie que la transformation est faite par rapport au repère $\mathbf{R_i}$.

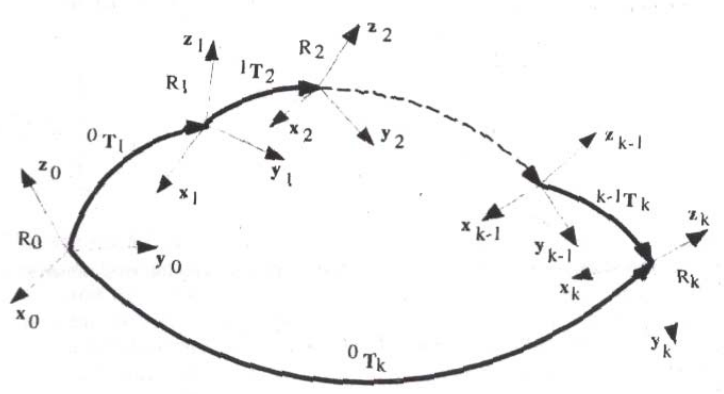


Figure I.12 : Composition des transformations :multiplication à droite

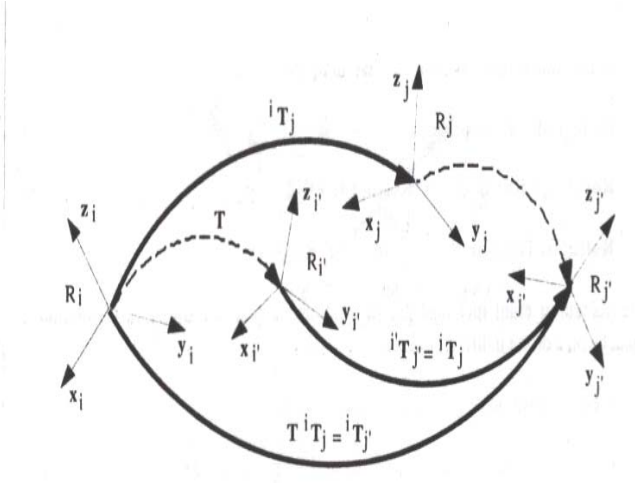


Figure I.13 : Composition des transformations : multiplication à gauche

I.6.CONCLUSION

❖ Dans ce chapitre, nous avons rappelé les différentes notions nécessaires pour l'étude des robots manipulateurs et présenté les différentes morphologies des robots-manipulateurs et de leurs constituants.

❖ Nous avons précisé la définition de certains termes qui sont nécessaires, pour l'étude des robots manipulateurs et présentons les notions et outils mathématiques mis en oeuvre pour la modélisation des robots.

❖ Pour compléter ces généralités,nous abordons dans les chapitres suivants le problème de la modélisation des robots manipulateurs.

II.1.INTRODUCTION [1][3][5][6]

Dans l'automatique, modéliser un système consiste à établir un ensemble de relations mathématiques qui permettent de décrire avec précision suffisante les interactions entre ce système et son environnement extérieur. Lorsque les relations sous-citées sont issues de la physique, le modèle obtenu est dit modèle de connaissance, ces relations découlent des observations disponibles sur le système. On aboutit aussi au modèle de représentation, en passant par l'identification.

La conception et la commande des robots nécessitent le calcul de certains modèles mathématiques. Tels que :

- les modèles de transformation entre l'espace opérationnel (dans lequel est définie la situation de l'organe terminal) et l'espace articulaire (dans lequel est définie la configuration du robot). On distingue :
    - Les modèles géométriques direct et inverse qui expriment la situation de l'organe terminal en fonction des variables articulaires du mécanisme et inversement ;
    - Les modèles cinématiques direct et inverse 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.

La modélisation des robots de façon systématique et automatique exige une méthode adéquate pour la description de leur morphologie. Plusieurs méthodes et notations ont été proposées. La plus rependue est celle de **Denavit-Hartenberg** (**D-H**) qui facilite la description géométrique du manipulateur, cette dernière nous permet d'aboutir au modèle cinématique et géométrique direct et inverse du robot. La même transformation offre une souplesse dans le calcul du modèle dynamique direct en utilisant le formalisme d'Euler-Lagrange.

Dans ce chapitre, nous allons introduire la notion de paramétrisation de **Denavit-Hartenberg** , afin de les utiliser pour le calcul des différents modèles de représentation du

notre robot ; nous commençons par la partie mécanique qui concerne la description de notre robot et sa réalisation , les moteurs à courant continue utilisés et ensuite les différents modèles directs et inverses.

Après l'obtention du modèle direct, nous présentons une méthode analytique pour le calcul des modèles inverses du robot et par la suite l'application de cette approche sur notre robot.

II.2.DESCRIPTION DU ROBOT

Le bras manipulateur qu'on a réalisé possède trois degrés de liberté caractérisées par des mouvements de rotations ; Les trois premières articulations de ce manipulateur (***rotoïde***) caractérisent pour la première une rotation autour d'un axe vertical, la seconde et la troisième suivant deux axes horizontaux dont les mouvements sont identifiés par les variables q_1, q_2 et q_3 .

Donc il appartient à la classe des robots à structure ouverte simple.

II.3. LA REALISATION DU ROBOT

Les pièces présentées ci-dessous sont fabriquées au sein du département Génie mécanique. L'atelier d'usinage est composé de trois pôles. Chaque pôle contient ces propres machines. Le premier dispose des tours conventionnels et le second est le champ du soudage. Le troisième pôle est équipé de fraiseuses conventionnelles.

Nous commençons d'abord par une démonstration des machines utilisées lors de notre réalisation :

- **Le Tour :** Cette machine sert principalement à usiner des pièces de révolution. La pièce est fixée dans le mandrin. Celui-ci est mis en rotation par le moteur de broche. L'outil suit une trajectoire qui interfère avec la pièce. L'outil est muni d'une arête coupante, il en résulte un enlèvement de matière.
- **La fraiseuse :** Cette machine sert principalement à usiner des pièces prismatiques. La pièce est fixée dans l'étau. L'outil est mis en rotation par le moteur de broche, il suit une trajectoire qui interfère avec la pièce. L'outil est muni d'une arête coupante, il en résulte un enlèvement de matière.
- **Perceuse :** Ceci permettre de faire des trous de différents diamètres en associant des outils de perçage : les forets.
- **Plieuse :** qui sert à plier les plaques métalliques à un angle donné.

- **Guillotine :** est une machine à couper mais à une épaisseur maximal de 3mm.

En suite nous présentons les outils manuels primordiaux à la fabrication de ce prototype :

✓ **Taraudage :** On peut obtenir un taraudage en utilisant des tarauds au diamètre souhaité.

✓ **Alésage :** Cette opération définit l'obtention d'un trou de qualité dans une pièce. On y associe des outils de perçage bien précis : les alésoirs ou les outils à aléser et dresser.

Entre autre nous décortiquons brièvement chaque pièce fabriquée (la conception est faite à base du logiciel mécanique ″**Solid Concept**″ qui nous a permis de dimensionner correctement le bras et de visualiser son assemblage)

❖ ***La Base :*** À l'aide d'une table traçante et un trusquin, on peut avoir une forme rectangulaire de la pièce. Ainsi le tour à quatre mors permet un évidement circulaire **(figure II.1).**

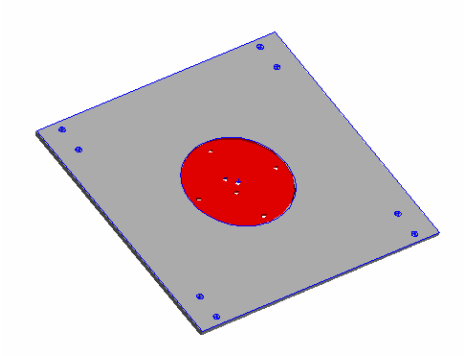


Figure II.1 *:* la base

❖ ***Les Pieds :*** la plieuse nous a permis d'avoir la structure présentée ci contre **(figureII.2).**

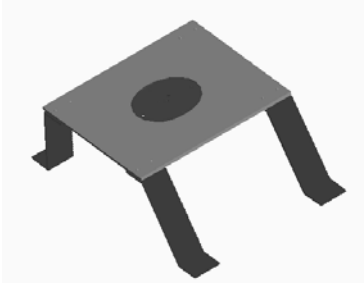


Figure II.2 *:* les pieds en assemblage avec la base

❖ ***La rampe :*** Ceci a une forme cylindrique donc elle est montée dans un mandrin « élément du tour » pour pouvoir effectuer deux évidements intérieurs, un pour avoir l'habilité de coincer la tourelle et le second pour le guidage (**figure II.3**).

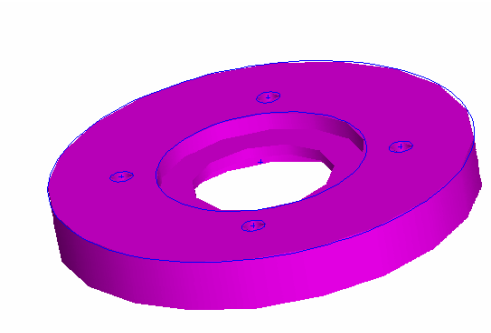


Figure II.3 : La rampe

❖ ***La tourelle :*** Obtenue par un tour (trois épaulements) et bien évidemment une gorge est prévu pour l'emplacement d'un circlipse afin d'assurer l'attachement des deux pièce. Ainsi une fraiseuse est utilisée pour faire une rainure dont laquelle le bras peut pivoter (**figure II.4**).

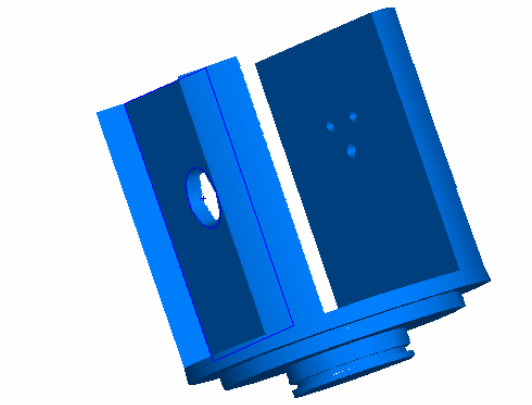


Figure II.4 : La tourelle

❖ ***Le bras :*** en perçant des profiles en aluminium, pour avoir les trous du fixation des moteurs **(figure II.5).**

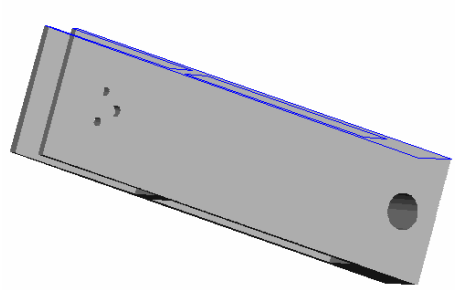


Figure II.5 : Le bras

Après l'assemblage de toutes ces pièces le bras est donné par la figure suivante (**figure II.6**) :

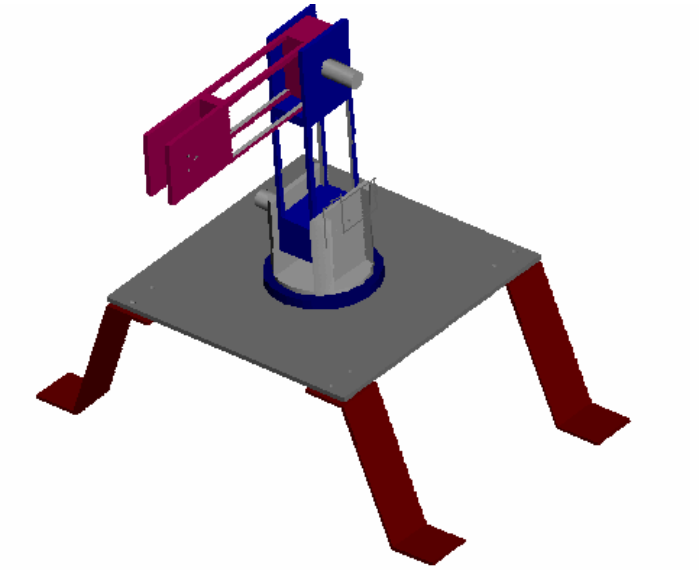


Figure II.6 : Le robot après assemblage

Figure II.7: Le robot réalisé

II.4.CARACTERISTIQUES DU MOTEUR (PITMAN GM9236) [7]

Les moteurs utilisés sont à courant continue qui ont les caractéristiques suivantes (**tableau 2.1**) :

	Symbole	Unités	Valeur
Tension de référence	E	V	24
Vitesse sans charge	V_{sc}	rad/s	7.4
Couple continu	C_c	Nm	3.4
Couple maximal	$C_{\max}$	Nm	18
Poids	m	g	671
Résistance	R	Ω	2.49
Inductance	L	mH	2.63
Courant sans charge	I_{sc}	A	0.16
Couple résistif	C_r	Nm	$5.6E-03$
Inertie du moteur	J	$Kg.m^2$	$7.1E-06$
Température maximum d'enroulement	$T_{\max}$	$^{\circ}C$	155
Rapport de réduction			65.5
Résolution de l'encodeur		CPR	500
Constante du couple	K_T	$N-m/A$	$4.58E-02$
Constante électrique	K_e	$V/rad/s$	$4.58E-02$
Constante de temps électrique	τ_E	ms	1.06
Constante de temps mécanique	τ_M	ms	8.5

Tableau 2.1 : Les caractéristiques du moteur

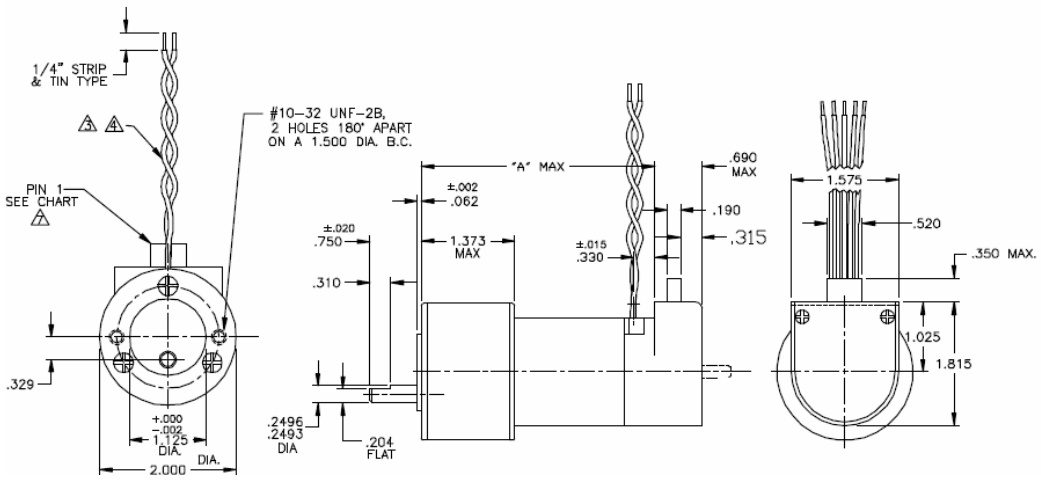


Figure II.8: La représentation des différentes parties du moteur

II.5.DESCRI¨PTION DE LA GEOMETRIE DES ROBOTS A STRUCTURE OUVERT SIMPLE [2]

Ce paragraphe présente la méthodologie à suivre pour décrire les robots à structure ouverte simple.

Une structure ouverte simple est composée de n+1 corps notés $C_0,...,C_n$ et de n articulations. Le corps C_0 désigne la base du robot et le corps C_n le corps qui porte l'organe terminal. L'articulation **j** connecte le corps C_j au corps C_{j-1} (**figure II.9**).

La méthode de description est fondée sur les règles et conventions suivantes:

- les corps sont supposés parfaitement rigides. Ils sont connectés par des articulations considérées comme idéales (pas de jeu mécanique, pas d'élasticité), soit rotoïdes, soit prismatiques ;
- le repère $\mathbf{R_i}$ est lié au corps $\mathbf{C_j}$;
- la variable de l'articulation j est notée $\mathbf{q_j}$.

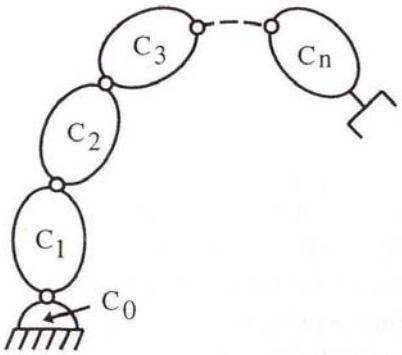


Figure II.9 : Robot à structure ouvert simple

Le repère $\mathbf{R_j}$ **,** fixé au corps $\mathbf{C_j}$ est défini de sort que :

- l'axe $\mathbf{z_j}$ est porté par l'axe de l'articulation j ;
- l'axe $\mathbf{x_j}$ est porté par la perpendiculaire commune aux axes $\mathbf{z_j}$ et $\mathbf{z_{j+1}}$. Si les axes $\mathbf{z_j}$ et $\mathbf{z_{j+1}}$ sont parallèles ou colinéaires, le choix de xi n'est pas unique : des considérations de symétrie ou de simplicité permettent alors un choix rationnel.

Le passage du repère $\mathbf{R_{j-1}}$ au repère $\mathbf{R_j}$ s'exprime en fonction des quatre paramètres géométriques suivants (**figure II.10**) :

- α_j : angle entre axes $\mathbf{z_{j-1}}$ et $\mathbf{z_j}$ correspondant à une rotation autour de $\mathbf{x_{j-1}}$;
- d_j : distance entre $\mathbf{z_{j-1}}$ et $\mathbf{z_j}$ le long de $\mathbf{x_{j-1}}$;

- θ_j : angle entre les axes $\mathbf{x_{j-1}}$ et $\mathbf{x_j}$ correspondant à une rotation autour de $\mathbf{z_j}$;
- r_j : distance entre $\mathbf{x_{j-1}}$ et $\mathbf{x_j}$ long de $\mathbf{z_j}$.

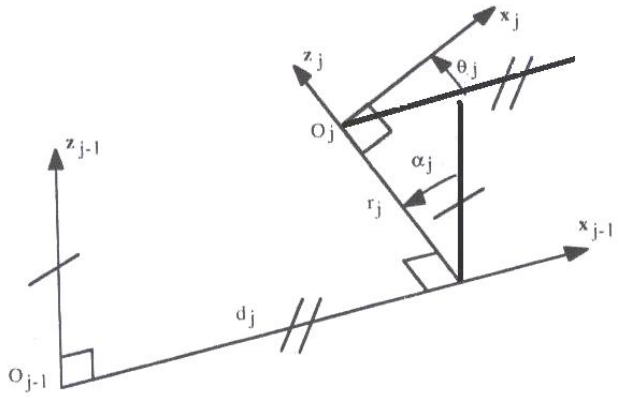


Figure II.10 : Paramètres géométriques dans le cas d'une structure ouverte simple

La variable articulaire $\mathbf{q_j}$ associe à la j^iéme^ articulation est soit θ_j, soit r_j ,selon que cette articulation est de type rotoïdes ou prismatique, ce qui se traduit par la relation :

$$q_j = \bar{\sigma}_j \theta_j + \sigma_j r_j \qquad \textbf{(2.1)}$$

Avec :

- $\sigma_j = 0$ si l'articulation j est rotoïdes ;
- $\sigma_j = 0$ si l'articulation j est prismatique ;
- $\bar{\sigma}_j = 1 - \sigma_j$.

Par analogie, on définit le paramètre $\bar{q}_j$ qui sera utilisé par la suite, tel que :

$$\bar{q}_j = \sigma_j \theta_j + \bar{\sigma}_j r_j \qquad \textbf{(2.2)}$$

La matrice de transformation définissant le repére R_j dans le repére R_{j-1} est donnée par (**figure II.10**)

$$^{j-1}T_j = Rot(x,\alpha_j)Trans(x,d_j)Rot(z,\theta_j)Trans(z,r_j)$$

$$= \begin{bmatrix} C\theta_j & -S\theta_j & 0 & d_j \\ C\alpha_j S\theta_j & C\alpha_j C\theta_j & -S\alpha_j & -r_j S\alpha_j \\ S\alpha_j S\theta_j & S\alpha_j C\theta_j & C\alpha_j & r_j C\alpha_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(2.3)}$$

Avec :

$$C\theta_j = \cos\theta_j ; S\theta_j = \sin\theta_j \ ; C\alpha_j = \cos\alpha_j ; S\alpha_j = \sin\alpha_j$$

La matrice ${}^{j-1}T_j$ est dite : **La matrice de transformation Denavit-Hartenberg .**

II.5.1.PARAMETRES DE DENAVIT-HARTENBERG [2][5][8]

Denavit et Hartenberg ont proposé une méthode qui repose sur l'assignation d'un repère unique pour chaque lien. Cette convention est une méthode systématique. Elle permet le passage entre articulations adjacentes d'un système robotique. Elle concerne les chaînes cinématiques ouvertes où l'articulation possède uniquement un degré de liberté, les surfaces adjacentes restent en contact.. Le choix adéquat des repères dans les liaisons facilite le calcul des matrices homogènes de **Denavit-Hartenberg** et permet d'arriver à exprimer rapidement des informations de l'élément terminal dans la base et vice versa.

II.6.MODELISATON GEOMETRIQUE

Dans la modélisation géométrique on s'intéresse au mouvement sans tenir compte des forces qui le provoque. Elle s'intéresse à l'études de la géométrie du robot en vu de décrire ses paramètres géométrique : position et orientation.

II.6.1.MODELE GEOMETRIQUE DIRECT [2][5][9]

Une tache fréquente d'un robot est d'aller prendre une pièce et de l'amener à un autre endroit, lorsque nous le faisons manuellement, c'est très simple car on voit la pièce à prendre et on amène notre main devant.

Pour un robot c'est moins simple, on doit trouver une façon de définir la position de l'objet de manière à ce qu'il sache où elle se trouve par rapport à un système de référence connu. La façon de faire est de fixer un système de coordonnées sur l'objet ainsi que sur l'outil et d'essayer de trouver les transformations nécessaires que doit subir le système de coordonnée de l'outil pour coïncider avec celui de l'objet.

L'étude de la géométrie directe des robots manipulateurs comprend plusieurs étapes pour arriver aux objectifs visés par cette modélisation. Dans un premier temps on fixe les repères aux différentes parties du mécanisme et on décrit les relations entre ces repères, et aussi on localise ces repères lorsque le manipulateur s'articule.

Le modèle géométrique direct (MGD) est l'ensemble des relations qui permettent d'exprimer la situation de l'organe terminale , c'est-à-dire les coordonnées opérationnelles

du robot, en fonction de ses coordonnées articulaire dans le cas d'une chaîne ouvert simple , il peut être représenté par la matrice de passage ${}^0T_n$.

$$ {}^0T_n = {}^0T_1(q_1)\,{}^1T_2(q_2)\ldots{}^{n-1}T_n(q_n) \qquad \textbf{(2.4)} $$

Le modèle géométrique direct du robot peut aussi être représenté par la relation :

$$ X = f(q) \qquad \textbf{(2.5)} $$

Avec : $X \in R^3$ les coordonnée cartésiennes, et $q \in R^3$ les coordonnées articulaires.

On va admettre quelques hypothèses dans le but de simplifier la modélisation des robots. Ces hypothèses sont les suivants :

- Les liaisons du manipulateur sont rigides ;
- Les jeux dans les articulations sont négligeables ;

Les capteurs ont un gain unitaire et de dynamique négligeable

II.6.2.MODELE GEOMETRIQUE INVERSE [3][5][6]

Le modèle géométrique direct d'un robot permet de calculer les coordonnées opérationnelles donnant la situation de l'organe terminal en fonction des coordonnées articulaires. Le problème inverse consiste à calculer les coordonnées articulaires correspondant à une situation donnée de l'organe terminal.

Le problème posé est le suivant : étant donné la position et l'orientation de l'outil par rapport à la station de travail, comment calculer l'ensemble des angles articulaires qui accomplissent cet objectif **?** La réponse à cette question constitue le modèle géométrique inverse du robot manipulateur.

La solution du problème, concernant la recherche des angles articulaires nécessaires pour positionner le repère de l'outil, par rapport au repère de la station de travail, est décomposé en deux parties. En premier lieu, sont déterminées les transformations nécessaires pour trouver le repère du poignet, par rapport au repère de la base et après, le modèle géométrique inverse est utilisé pour trouver les angles des articulations.

Une approche analytique a été utilisée pour le robots de type série ,elle consiste à éliminer à chaque étape une des coordonnées généralisées (articulaires) par la multiplication de la matrice de transfert finale ${}^0T_E$ par les matrices de transformation intermédiaires.

Pour le cas d'un robot à trois degrés de liberté la matrice de transformation s'écrit sous la forme :

$$^{0}T_{3} = {}^{0}T_{1}(\theta_1)\,{}^{1}T_{2}(\theta_2)\,{}^{2}T_{3}(\theta_3) \ ; \qquad \textbf{(2.6)}$$

Pour l'organe terminale (**E**) la matrice de transformation par rapport au repère trois est donnée par

$$^{3}T_{E} = \begin{bmatrix} 1 & 0 & 0 & l_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \textbf{(2.7)}$$

$l_3 =$; La langueur de deuxième bras.

Donc la matrice de transformation homogène est :

$$^{0}T_{E} = {}^{0}T_{3}\,{}^{3}T_{E} \qquad \textbf{(2.8)}$$

Par exemple la première étape sera comme suit :

On multiplie de part et d'autre par la matrice $^{0}T_{1}(\theta_1)$, le résultat sera :

$$\left[{}^{0}T_{1}\right]^{-1}\,{}^{0}T_{3} = {}^{1}T_{3} \qquad \textbf{(2.9)}$$

On s'intéresse toujours à la dernière colonne de la matrice, qui contient à chaque étape les équations découplées qui permettent de résoudre le problème du modèle géométrique inverse.

Toutefois le modèle géométrique comporte aussi des inconvénients:

1) la non unicité du modèle géométrique inverse implique qu'il existe plusieurs "chemins" pour se rendre d'un point à un autre.
2) le traitement par incrément peut amener à des imprécisions.
3) des singularités, mécaniques et/ou mathématiques apparaissent.

Ces inconvénients sont évités ou contournés de la manière suivante:

1) Le problème des multiples solutions du modèle inverse n'intervient pas. En effet, le robot possède une configuration initiale connue, et se rend à une autre position à partir de celle-ci. Il ne sert à rien de calculer des configurations du robot qui seraient impossibles à atteindre depuis la position courante.

2) Une haute précision des solutions obtenues n'est pas nécessaire puisqu'il suffit de fournir à l'utilisateur une vision globale, le calcul des accroissements est à chaque fois effectué à partir d'une nouvelle configuration exacte du robot.
3) Quant au problème des singularités, il existe plusieurs méthodes mathématiques pour les traiter ou les éviter.

II.7.MODELISATION CINEMATIQUE [5]

Le modèle cinématique est, littéralement, un modèle des vitesses. Il exprime les relations entre les vitesses articulaires de chaque joint et les vitesses cartésiennes d'un point de la chaîne cinématique, généralement l'organe terminal. Ce modèle est donc un modèle par accroissements élémentaires: chaque variation élémentaire de la valeur d'une articulation implique une variation de position de l'organe terminal, et inversement. Lorsque ces variations infinitésimales sont exprimées par rapport au temps, on peut les considérer comme des vitesses.

Le modèle cinématique permet donc non seulement de compléter éventuellement le modèle géométrique en tenant compte des vitesses, mais aussi de remplacer le modèle géométrique: en agissant par accroissements successifs, on peut se déplacer d'un point donné à un autre.

Le modèle cinématique possède une propriété essentielle: il est une différentiation du modèle géométrique. Il est donc une linéarisation du système d'équations non linéaires représentant le modèle géométrique. Par conséquent on peut toujours facilement obtenir les transformations inverses puisqu'elles proviennent de l'inversion d'un problème linéaire.

II.7.1.MODELE CINEMATIQUE DIRECT [1][5]

Le modèle cinématique direct d'un robot manipulateur décrit les vitesses des coordonnées opérationnelles en fonction des vitesses articulaires. Il est noté :

$$\dot{X} = J(q)\dot{q} \qquad \textbf{(2.10)}$$

Où $J(q)$ désigne la matrice jacobienne de dimension (m x n) du mécanisme, égale à $\frac{\partial X}{\partial q}$ et fonction de la configuration articulaire q. La même matrice jacobienne intervient dans le calcul du modèle différentiel direct qui donne les variations élémentaires dx des coordonnées opérationnelles en fonction des variations dq, soit :

$$dx = J(q)dq \qquad \textbf{(2.11)}$$

II.7.1.1.La matrice jacobienne

L'outil principalement utilisé pour traiter le problème de la cinématique des robots est la matrice jacobienne. Elle représente un opérateur permettant de lier les vitesses des corps d'un robot exprimées dans différents espaces vectoriels.

II.7.1.2.L'intérêt de la matrice jacobienne est multiple

- elle est à la base du modèle cinématique inverse, permettant de calculer une solution locale des variations articulaires dq connaissant les variations opérationnelles dx ;
- en statique, on utilise le jacobien pour établir la relation liant les efforts exercés par l'organe terminal sur l'environnement aux forces et couples des actionneurs ;
- elle facilite le calcul des singularités et de la dimension de l'espace opérationnel accessible du robot.

II.7.1.3.Calcul de la matrice jacobienne par dérivation du MGD

Le calcul de la matrice jacobienne peut se faire en dérivant le MGD, $X = f(q)$ à partir de la relation :

$$J_{ij} \frac{\partial f_i(q)}{\partial q_j} \qquad i = 1,...,m \; ; \; j = 1,...,n$$

Où J_{ij} est l'élément (i,j) de la matrice jacobienne J .

Cette méthode est facile à mettre en oeuvre pour les robots à deux ou trois degrés de liberté, mais elle devient de plus en plus compliquée dans le cas du manipulateur à six degré de liberté.

II.7.2.MODELE CINEMATIQUE INVERSE [5]

L'objectif du modèle cinématique inverse est de calculer, à partir d'une configuration q donnée, les vitesses articulaires $\dot{q}\cdot$ qui assurent au repère terminal une vitesse opérationnelle $\dot{X}\cdot$ imposée. On peut déterminer la différentielle articulaire dq correspondante à une différentielle des coordonnées opérationnelles spécifiée dX . Pour obtenir le modèle cinématique inverse, on inverse le modèle cinématique direct en résolvant un système d'équations linéaires. La mise en oeuvre peut être faite de façon analytique ou numérique :

- la solution analytique a pour avantage de diminuer considérablement le nombre d'opérations, mais on doit traiter séparément tous les cas singuliers.

- les méthodes numériques sont plus générales, la plus répandue étant fondée sur la notion de pseudo inverse : les algorithmes traitent de façon unifiée les cas réguliers, singuliers et redondants. Elles nécessitent un temps de calcul relativement important.

On peut écrire le modèle cinématique inverse sous la forme :

$$\dot{q} = J^{-1}\dot{X} \qquad \textbf{(2.12)}$$

II.8.MODELISATION DU ROBOT RÉALISÉ [2][8]

II.8.1.MODELE GEOMETRIQUE DIRECT [8]

Le bras manipulateur qu'on a réalisé possède trois degrés de liberté caractérisées par des mouvements de rotations ; Les trois premières articulations de ce manipulateur (***rotoïde***) caractérisent pour la première une rotation autour d'un axe vertical, la seconde et la troisième suivant deux axes horizontaux dont les mouvements sont identifiés par les variables q_1, q_2 et q_3 . Donc il appartient à la classe des robots à structure ouverte simple.

La représentation du bras de robot suivant la convention de **Denavit- Hartenberg** est représentée dans la figure (**figure II.11**).

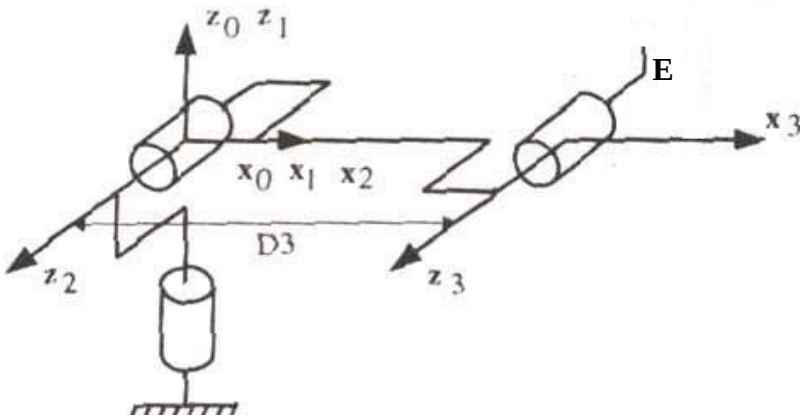


Figure II.11 : Placement des repères

En respectant la position d'origine, la définition des repères des liaisons présentées par la **figure II.10**, les paramètres du bras de robot suivant la convention de **Denavit-Hartenberg** sont représentés dans le tableau **Tableau 2.2 :**

j **Numéro de la liaisons**	σ_j	α_j degrés	d_j métres	θ_j variable	r_j métres
1	0	0	0	θ_1	0
2	0	-90	0	θ_2	0
3	0	0	l_2	θ_3	0

Tableau 2.2 : Paramètres de **Denavit-Hartenberg** du bras de robot

Dans notre application, La langueur de deuxième bras est $l_2 = 470$ mm .

Les transformations appropriées en utilisant le formalisme de **Denavit-Hartenberg** pour les trois premières articulations sont :

$$ {}^0T_1 = \begin{bmatrix} C_1 & -S_1 & 0 & 0 \\ S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} ; {}^1T_2 = \begin{bmatrix} C_2 & -S_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -S_2 & -C_2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} ; {}^2T_3 = \begin{bmatrix} C_3 & -S_3 & 0 & l_2 \\ S_3 & C_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.13)} $$

Avec : $C_i = \cos\theta_i ; S_i = \sin\theta_i ; C_{ij} = \cos(\theta_i + \theta_j) ; S_{ij} = \sin(\theta_i + \theta_j)$

La matrice de transformation homogène de l'organe terminale par rapport au dernier repère est donnée par :

$$ {}^3T_E = \begin{bmatrix} 1 & 0 & 0 & l_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.14)} $$

On peut calculer la matrice de transformation totale ${}^0T_E$ pour pouvoir exprimer la position d'un objet dans l'espace dans un référentiel lié à la base, en utilisant la relation suivante :

$$ {}^{0}T_{E} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = {}^{0}T_{3}\,{}^{3}T_{E} = {}^{0}T_{1}(\theta_1)\,{}^{1}T_{2}(\theta_2)\,{}^{2}T_{3}(\theta_3) \begin{bmatrix} 1 & 0 & 0 & l_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.15)} $$

Avec $l_3 = 200$; la langueur de deuxième bras.

Donc on a :

$$ {}^{0}T_{E} = \begin{bmatrix} C_1C_{23} & -C_1S_{23} & -S_1 & (l_2C_2 + l_3C_{23})C_1 \\ S_1C_{23} & -S_1S_{23} & C_1 & (l_2C_2 + l_3C_{23})S_1 \\ -S_{23} & -C_{23} & 0 & -l_2S_2 - l_3S_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.16)} $$

Avec : $C_i = \cos\theta_i ; S_i = \sin\theta_i ; C_{ij} = \cos(\theta_i + \theta_j) ; S_{ij} = \sin(\theta_i + \theta_j)$

Donc : $r_{11} = C_1C_{23}$ **(2.17)**

$r_{12} = -C_1S_{23}$ **(2.18)**

$r_{13} = -S_1$ (**2.19**)

$r_{21} = S_1C_{23}$ (**2.20**)

$r_{22} = -S_1S_{23}$ **(2.21)**

$r_{23} = C_1$ **(2.22)**

$r_{31} = -S_{23}$ **(2.23)**

$r_{32} = -C_{23}$ **(2.24)**

$r_{33} = 0$ **(2.25)**

$P_x = (l_2C_2 + l_3C_{23})C_1$ **(2.26)**

$P_y = (l_2C_2 + l_3C_{23})S_1$ **(2.27)**

$P_z = -l_2S_2 - l_3S_{23}$ **(2.28)**

II.8.2.MODELE GEOMETRIQUE INVERSE [2][3][5]

On peut citer quelques solutions qui existent :

- Méthode analytique [Craig 89],
- Transformation inverse, par Paul et Al 1981.
- Approche géométrique, par Lee et Ziegler 1984.

- Méthode des matrices duelles, par Denavit 1956.
- Quaternions duaux, par Yang et Arenden Stein 1964.

Nous nous intéressons à la méthode analytique.

On a :

$$ {}^{0}T_{E} = \begin{bmatrix} C_1C_{23} & -C_1S_{23} & -S_1 & (l_2C_2 + l_3C_{23})C_1 \\ S_1C_{23} & -S_1S_{23} & C_1 & (l_2C_2 + l_3C_{23})S_1 \\ -S_{23} & -C_{23} & 0 & -l_2S_2 - l_3S_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.29)} $$

On pose :

$$ {}^{0}T_{E}^{*} = \begin{bmatrix} & & & P_x \\ & R_{ij} & & P_y \\ & & & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = U_0 \quad \textbf{(2.30)} $$

<u>1^{iére} Etape :</u>

On a : $ {}^{0}T_{1}\,{}^{1}T_{2}\,{}^{2}T_{3}\,{}^{3}T_{E} = {}^{0}T_{E} = {}^{0}T_{E}^{*} = U_0 $ **(2.31)**

$$ \Rightarrow \quad {}^{1}T_{2}\,{}^{2}T_{3}\,{}^{3}T_{E} = {}^{1}T_{0}\,{}^{0}T_{E}^{*} = U_1 = {}^{1}T_{E}^{*} \quad \textbf{(2.32)} $$

La dernière colonne de la matrice $ {}^{1}T_{E}^{*} $ est :

$$ \begin{cases} U_1(1) = C_1P_x + S_1P_y. \\ U_1(2) = -S_1P_X + C_1P_y. \\ U_1(3) = P_z \end{cases} \quad \textbf{(2.33)} $$

Les éléments de droite sont données par la quatrième colonne de $ {}^{0}T_{E} $ calculée précédemment :

$$ \begin{cases} V_1(1) = l_2C_2 + l_3C_{23} \\ V_1(2) = 0 \\ V_1(3) = -l_3S_{23} - l_2S_2 \end{cases} \quad \textbf{(2.34)} $$

En identifiant $ V_1(2) $ *et* $ U_1(2) $:

$$ -S_1P_x + C_1P_y = 0 \quad \textbf{(2.35)} $$

D'après l'équation (**2.35**) , on a : $ \dfrac{S_1}{C_1} = \dfrac{P_y}{P_x} $ **(2.36)**

A partir de l'équation **(2.36)** , q_1 est donnée par :

$$q_1 = \theta_1 = ATAN2(P_{y,} P_x) \qquad \textbf{(2.37)}$$

2iére Etape :

On a : $${}^1T_2\ {}^2T_3\ {}^3T_E = {}^1T_0\ {}^0T_E^* = U_1 = {}^1T_E^* \qquad \textbf{(2.38)}$$

$$\Rightarrow \quad {}^2T_3\ {}^3T_E = {}^2T_1\ {}^1T_E^* = U_2 = {}^2T_E^* \qquad \textbf{(2.39)}$$

La dernière colonne de la matrice ${}^2T_E^*$ est :

$$\begin{cases} U_2(1) = C_2(C_1P_x + S_1P_y) - P_zS_2. \\ U_2(2) = -S_2(C_1P_X + S_1P_y) - C_2P_2. \\ U_2(3) = C_1P_y - S_1P_x \end{cases} \qquad \textbf{(2.40)}$$

Les éléments de droite sont données par la quatrième colonne de ${}^1T_E$ calculée précédemment :

$$\begin{cases} V_2(1) = l_2 + l_3C_3 \\ V_2(2) = l_3S_3 \\ V_2(3) = 0 \end{cases} \qquad \textbf{(2.41)}$$

On peut calculer q_2 et q_3 en considérons les deux premières équations et en résolvant un système de type 6.

On obtient d'abord une équation en q_2 telle que :

$$\begin{cases} U_2(1) = C_2(C_1P_x + S_1P_y) - P_2S_2 \qquad (1) \\ U_2(2) = -S_2(C_1P_x + S_1P_y) - C_2P_2 \qquad (2) \\ V_2(1) = C_3l_3 + l_2 \qquad (3) \\ V_2(2) = l_3S_3 \qquad (4) \end{cases} \qquad \textbf{(2.42)}$$

On va multiplier l'équation (1) dans le système d'équations (2.42) par $2l_2$

On pose :

$$B_1 = C_1P_x + S_1P_y \qquad \textbf{(2.43)}$$

$$\textbf{(1)} \Rightarrow 2l_2C_2B_1 - 2l_2P_2S_2 = 2C_3L_2l_3 + 2l_2^2 = \alpha \qquad \textbf{(2.44)}$$

$$\Rightarrow \alpha = l_2^2 + (l_2 + C_3l_3)^2 - (C_3l_3)^2 + l_3^2 - l_3^2 \qquad \textbf{(2.45)}$$

On pose $\gamma = (C_3l_3)^2$ **(2.46)**

$\Rightarrow l_3^2 - \gamma = V_2^2(2)$ **(2.47)**

Donc : $\alpha = l_2^2 + V_2^2(1) + V_2^2(2) - l_3^2$ **(2.48)**

Comme : $\begin{cases} V_2(1) = U_2(1) \\ V_2(2) = U_2(2) \end{cases}$ **(2.49)**

Donc $\alpha = l_2^2 - l_3^2 + U_2^2(1) + U_2^2(2)$ **(2.50)**

Par la suit on a : $U_2^2(1) + U_2^2(2) = B_1^2 + P_z^2$ **(2.51)**

$\Rightarrow \alpha = l_2^2 - l_3^2 + B_1^2 + P_z^2$ **(2.52)**

Enfin on obtient une équation en $q_2 = \theta_2$ telle que :

$$X.S_2 + Y.C_2 = Z \quad \textbf{(2.53)}$$

Avec :

$$\begin{cases} X = -2P_z l_2 \\ Y = -2B_1 l_2 \\ Z = l_3^2 - l_2^2 - B_1^2 - P_z^2 \end{cases} \quad \textbf{(2.54)}$$

La solution est donnée par :

$$\begin{cases} C_2 = \dfrac{YZ - \varepsilon X\sqrt{X^2 + Y^2 - \alpha^2}}{X^2 + Y^2} \\ S_2 = \dfrac{YZ + \varepsilon X\sqrt{X^2 + Y^2 - \alpha^2}}{X^2 + Y^2} \end{cases} \quad avec \ : \varepsilon = \pm 1 \quad \textbf{(2.55)}$$

Donc :

$$q_2 = \theta_2 = ATAN2(S_2, C_2) \quad \textbf{(2.56)}$$

Connaissant q_2, on peut calculer q_3 à partir de :

$$\begin{cases} C_3 = \dfrac{P_z C_2 - \beta_1 S_2}{L_3} \\ S_3 = \dfrac{-P_z S_2 - \beta_1 C_2 + L_2}{L_3} \end{cases} \quad \textbf{(2.57)}$$

Alors :

$$q_3 = \theta_3 = ATAN2(S_3, C_3) \quad \textbf{(2.58)}$$

II.8.3.MODELE CINEMATIQUE DIRECT [1][5]

On a la matrice de transformation homogène :

$$ {}^{0}T_{E} = \begin{bmatrix} C_1C_{23} & -C_1S_{23} & -S_1 & (l_2C_2 + l_3C_{23})C_1 \\ S_1C_{23} & -S_1S_{23} & C_1 & (l_2C_2 + l_3C_{23})S_1 \\ -S_{23} & -C_{23} & 0 & -l_2S_2 - l_3S_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.59)} $$

On s'intéresse au quatrième colonne de la matrice de transformation homogène qui nous donne le modèle géométrique directe ; on trouve que :

$$ P_x = (l_2C_2 + l_3C_{23})C_1 \quad \textbf{(2.60)} $$

$$ P_y = (l_2C_2 + l_3C_{23})S_1 \quad \textbf{(2.61)} $$

$$ P_z = -l_2S_2 - l_3S_{23} \quad \textbf{(2.62)} $$

Comme le modèle cinématique inverse est définie par : $\dot{X} = J(q)\dot{q}$ en dérivent le modèle geometrique direct .

On cherche à trouver :

$$ \begin{cases} \dot{X}_1 = f(\dot{q}_1, \dot{q}_2, \dot{q}_3) = \dfrac{d(P_x)}{dt} \\ \dot{X}_2 = g(\dot{q}_1, \dot{q}_2, \dot{q}_3) = \dfrac{d(P_y)}{dt} \\ \dot{X}_3 = h(\dot{q}_1, \dot{q}_2, \dot{q}_3) = \dfrac{d(P_z)}{dt} \end{cases} \quad \textbf{(2.63)} $$

On procède comme suit :

$$ \frac{dP_x}{dt} = \frac{dP_x}{dq_1}\frac{dq_1}{dt} = -S_1(l_2C_2 + l_3C_{23})\dot{q}_1 \quad \textbf{(2.64)} $$

$$ \frac{dP_x}{dt} = \frac{dP_x}{dq_2}\frac{dq_2}{dt} = -C_1(-l_2S_2 - l_3S_{23})\dot{q}_2 \quad \textbf{(2.65)} $$

$$ \frac{dP_x}{dt} = \frac{dP_x}{dq_3}\frac{dq_3}{dt} = -l_3C_1S_{23}\dot{q}_3 \quad \textbf{(2.66)} $$

De même manière pour P_y et P_z on trouve :

$$\dot{X} = \begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \\ \dot{X}_3 \end{bmatrix} = J(q)\dot{q} \qquad \textbf{(2.67)}$$

$$\Rightarrow \dot{X} = \begin{bmatrix} -S_1(l_2C_2 + l_3C_{23}) & C_1(-l_2S_2 - l_3S_{23}) & -l_3C_1S_{23} \\ C_1(l_2C_2 + l_3C_{23}) & S_1(-l_2S_2 - l_3S_{23}) & -l_3S_1S_{23} \\ 0 & -l_2C_2 - l_3C_{23} & -l_3C_{23} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} \qquad \textbf{(2.68)}$$

Avec : $C_i = \cos\theta_i ; S_i = \sin\theta_i ; C_{ij} = \cos(\theta_i + \theta_j) ; S_{ij} = \sin(\theta_i + \theta_j)$

Donc :

$$J(q) = \begin{bmatrix} -S_1(l_2C_2 + l_3C_{23}) & C_1(-l_2S_2 - l_3S_{23}) & -l_3C_1S_{23} \\ C_1(l_2C_2 + l_3C_{23}) & S_1(-l_2S_2 - l_3S_{23}) & -l_3S_1S_{23} \\ 0 & -l_2C_2 - l_3C_{23} & -l_3C_{23} \end{bmatrix} \qquad \textbf{(2.69)}$$

II.8.2.MODELE CINEMATIQUE INVERSE [5]

Le modèle cinématique inverse est définit par :

$$\dot{q} = J^{-1}\dot{X} \qquad \textbf{(2.70)}$$

Avec :

$$J^{-1} = \begin{bmatrix} \dfrac{-S_1}{12C_2 + 13C_{23}} & \dfrac{C_1}{12C_2 + 13C_{23}} & 0 \\ \dfrac{-C_1C_{23}}{12(S_2C_{23} - C_2S_{23})} & \dfrac{-S_1C_{23}}{12(S_2C_{23} - C_2S_{23})} & \dfrac{S_{23}}{12(S_2C_{23} - C_2S_{23})} \\ \dfrac{(12C_2 + 13C_{23})C_1}{12.13.(S_2C_{23} - C_2S_{23})} & \dfrac{(12C_2 + 13C_{23})S_1}{12.13.(S_2C_{23} - C_2S_{23})} & \dfrac{-(12S_2 + 13S_{23})}{12.13.(S_2C_{23} - C_2S_{23})} \end{bmatrix} \qquad (\textbf{2.71})$$

Avec :

$$C_i = \cos\theta_i ; S_i = \sin\theta_i ; C_{ij} = \cos(\theta_i + \theta_j) ; S_{ij} = \sin(\theta_i + \theta_j)$$

II.9.CONCLUSION

- La modélisation des robots manipulateurs est basée sur la modélisation de la structure mécanique, les actionneurs et les capteurs. Elle permet la détermination des relations directes et inverses entre les coordonnées généralisées q et les coordonnées cartésiennes X et leurs dérivées respectives($\dot{q}, \dot{X}$).

- Nous avons montré dans ce chapitre comment calculer le modèle géométrique direct des robots à structure ouverte simple. La méthode de description utilise des règles qui ont une logique intrinsèque facilitant leur application. Le modèle géométrique direct est unique et est donné sous forme d'équations explicites.
- La convention de Denavit-Hartenberg fournit une souplesse dans le calcul pour la localisation de la position de l'élément terminal. La position initiale du robot influe sur toute la modélisation. La définition des axes selon Denavit-Hartenberg influe sur le modèle dynamique parce que les moments d'inertie des différents axes changent.
- Le passage de l'espace des coordonnées cartésiennes de l'élément terminal à celui des coordonnées généralisées ou articulaires présente des singularités physiques (point en dehors de l'espace de travail), ces singularités apparaissent sous forme de contraintes lors du calcul du modèle géométrique inverse.
- L'étude des espaces de travail des robots industriels et de leur configuration de singularité irréversible constituant un grand problème dans l'industrie, nous a mener a la détermination du MCD basée sur la matrice jacobienne de base, du MCI qui est obtenu directement par inversion matricielle simple du MCD dans le cas régulier ou par la pseudo-inverse dans le cas singulier.

INTRODUCTION GENERALE SUR L'UMAC

La commande de mouvement est d'une très grande importance dans le domaine industriel. Elle a fait l'objet de nombreux travaux de recherche, basés essentiellement sur les stratégies de commande et de leur implémentation.

L'évolution des technologies informatiques en général et numériques en particulier a permis l'amélioration de la commande des moteurs grâce à l'exploitation de leur grande vitesse de traitement de l'information et la possibilité d'appliquer les différentes techniques qu'on ne pouvait pas utiliser avec les anciens circuits de commande. Cela à dynamiser l'industrie de l'électronique, de nouveaux matériels spécialisés dans la commande sont alors apparus, plusieurs firmes entre autre **DELTA TAU** proposent une gamme variée de cartes de commande parmi elles : l'interface Turbo UMAC objet de notre étude.

Cette interface est autonome et programmable à partir d'un PC par une liaison série, elle contrôle jusqu'à huit moteurs simultanément et peut être étendue jusqu'à 32 moteurs, elle fournie une grande flexibilité et un large champ d'utilisation technologique.

En vue d'exploiter les facultés de l'interface UMAC, notre travail est porté sur «Commande du bras manipulateur par l'interface d'axe UMAC ».

III.1 PRESENTATION GENERALE

L'UMAC « *Universal Motion Automation Controller* » est un contrôleur universel de mouvement et d'automatisation, constitué d'un système modulaire en format 3U, combinant la puissance et la flexibilité de la famille des contrôleurs de mouvement multi-axes programmables PMAC « *Programmable Multi-Axis Controller* » produit par la firme **DELTA TAU**. l'unité centrale de traitement du système UMAC est la carte Turbo PMAC2-3U CPU, reliée aux cartes d'entrée/sortie DSPGATE et IOGATE, d'amplification AMP-2 et d'alimentation électrique par la surface arrière UBUS (UMAC Bus) implantée dans une armoire munie de cartes : ces cartes sont dans un ordre optimal du bas en haut afin que le danger et le bruit électromagnétique soient éloignés aux maximum, la Figure III.1.a donne l'image l'UMAC et la Figure III.I.b présente la configuration de la carte en format 3U de ce dernier.

Figure III.1.a :
Turbo UMAC

Figure III.1.b : Configuration de la carte en format 3U Sur un UBUS

Les différent cartes composant l'UMAC sont :

III.1.1 Turbo PMAC2-3U CPU[10]: C'est l'unité centrale du traitement du système, elle joue le rôle d'un gestionnaire du système et d'un contrôleur pour les moteurs, Elle contient un DSP Motorola 56303, des mémoires où sont implémentés les micro-programmes et un connecteur série RS 232/422.

III.1.2 DSPGATE[11]: Nous disposons de deux DSPGATE qui jouent le rôle d'une première interface entre le CPU (Turbo PMAC2-3U CPU) et les signaux d'entrée/sortie pour

les moteurs, chacune d'elles peut contrôler quatre axes et chaque axe envoie les signaux de sortie de commande pour les amplificateurs.

III.1.3 IOGATE[12] : C'est un composant d'adaptation des signaux d'entrée-sortie digital avec isolation optique, elle est faite pour compléter les fonctions d'automatisation de l'UMAC.

III.1.4 AMPLIFICATEUR DE PUISSANCE[13] : Il y'a huit amplificateurs répartis sur deux cartes, chaque amplificateur a pour rôle de rendre les signaux de commande délivrée par les DSPGATE amplifié et exploitable.

III.1.5 CIRCUIT D'ALIMENTATION ELECTRIQUE[14] : Assurant une alimentation de 5V, 15V et -15V continue nécessaire pour l'alimentation des circuits analogiques et numériques du système.

III.2 FONCTIONNEMENT GENERAL

Le programme de mouvement une fois élaboré sur un PC sera chargé dans une zone mémoire située dans le Turbo PMAC2-3U CPU grâce à un logiciel associé à l'interface et par l'intermédiaire d'une liaison série RS232/422 (Figure III.2).

A l'exécution des instructions de commande de mouvement qui sont dans le programme, le CPU communique avec les registres de la DSPGATE par l'intermédiaire de l'UBUS en envoyant et en lissent les signaux de commande et de mesure de l'encodeur.

A ce moment la DSPGATE envoi les signaux de commande analogique en courant (après la conversion digitale/analogique) à l'amplificateur et compte les impulsions venant de l'encodeur tout en mettent l'état des d'indicateurs (par exemple l'état des fins de course) dans des registres bien spécifiques.

L'amplificateur pour sa part génère un courant proportionnel à la tension de commande livrée par la DSPGATE en appliquent une tension avec modulation de largeur des impulsions (PWM) sur les bornes du moteur.

Si des signaux entrées ou sortie digitaux sont utilises pour une application donnée, alors ces signaux vont être achemines par l'IOGATE qui va les conditionner et les sauvegarder dans de registre pour que le CPU puisse lire ou écrire les données.

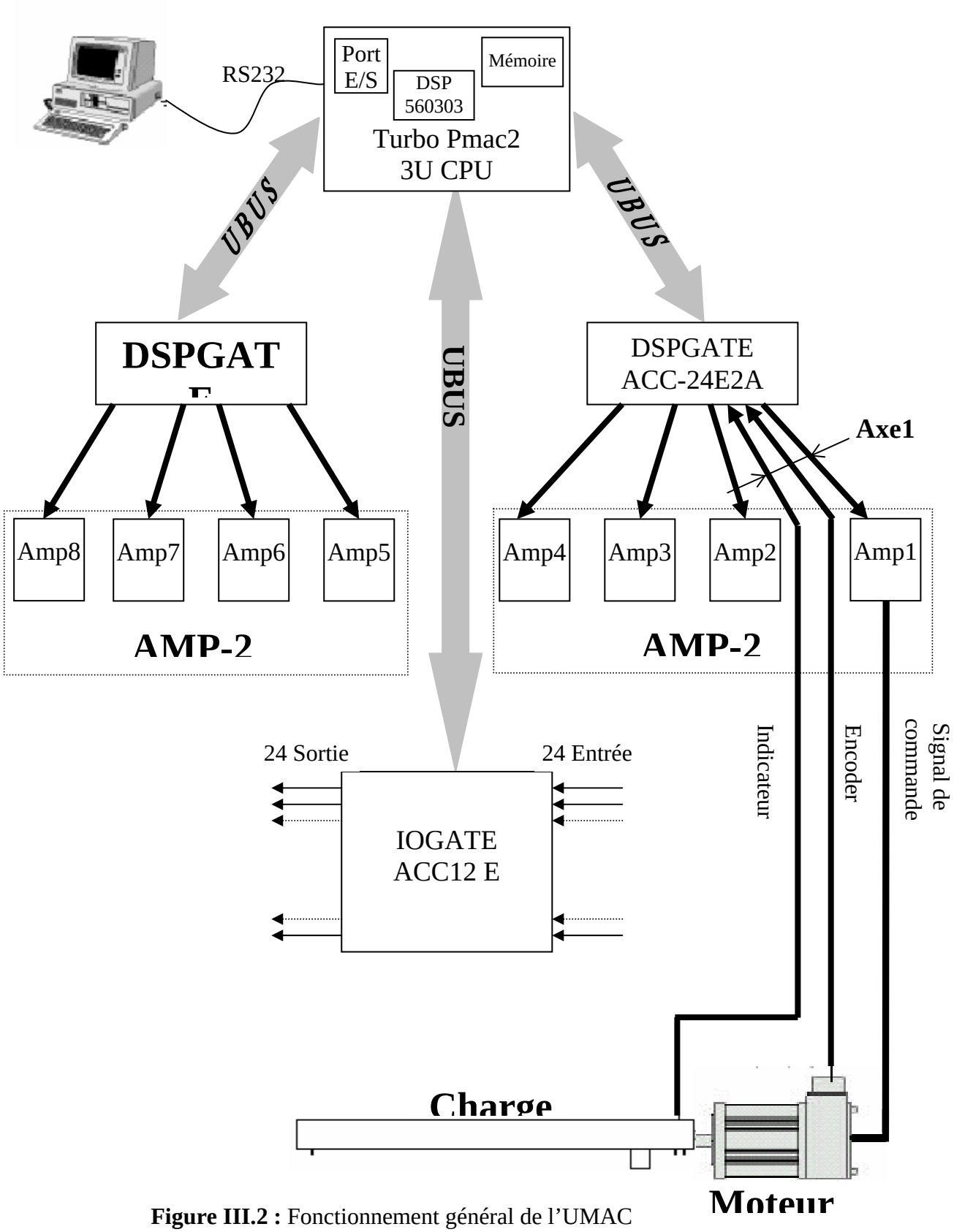


Figure III.2 : Fonctionnement général de l'UMAC

III.3 DESCRIPTION DU TURBO PMAC2 3U CPU[10][15][16][17]

PMAC, prononcé " *Pe'-MAC* ", représente un contrôleur multi-axes programmable, c'est une famille des contrôleurs a rendement élevé, capables de commander jusqu'à huit moteurs (huit axes) simultanément et extensible jusqu'à 32 axes, pour cette famille de contrôleur, le Turbo Pmac2 3U CPU est considéré parmi les plus récents avec un niveau très sophistiqué, l'image du Turbo Pmac2 3U CPU est présenté dans la Figure III.3.

Figure III.3 : Turbo Pmac2 3U CPU

Le processeur de traitement signal numérique **DSP56303** de **Motorola** est l'élément principal du Turbo Pmac2 3U CPU, car c'est l'unité centrale de cette carte ou tous les algorithmes de commande seront calculés pour les huit axes.

Cette carte a sa propre mémoire et microprocesseur, par conséquent elle fonctionne comme un contrôleur autonome ou un ordinateur qu'on peut commander par l'intermédiaire d'un port série. Le port série est connecté directement à cette carte, le diagramme en block du Turbo Pmac2 3U CPU est présenté par la Figure suivant :

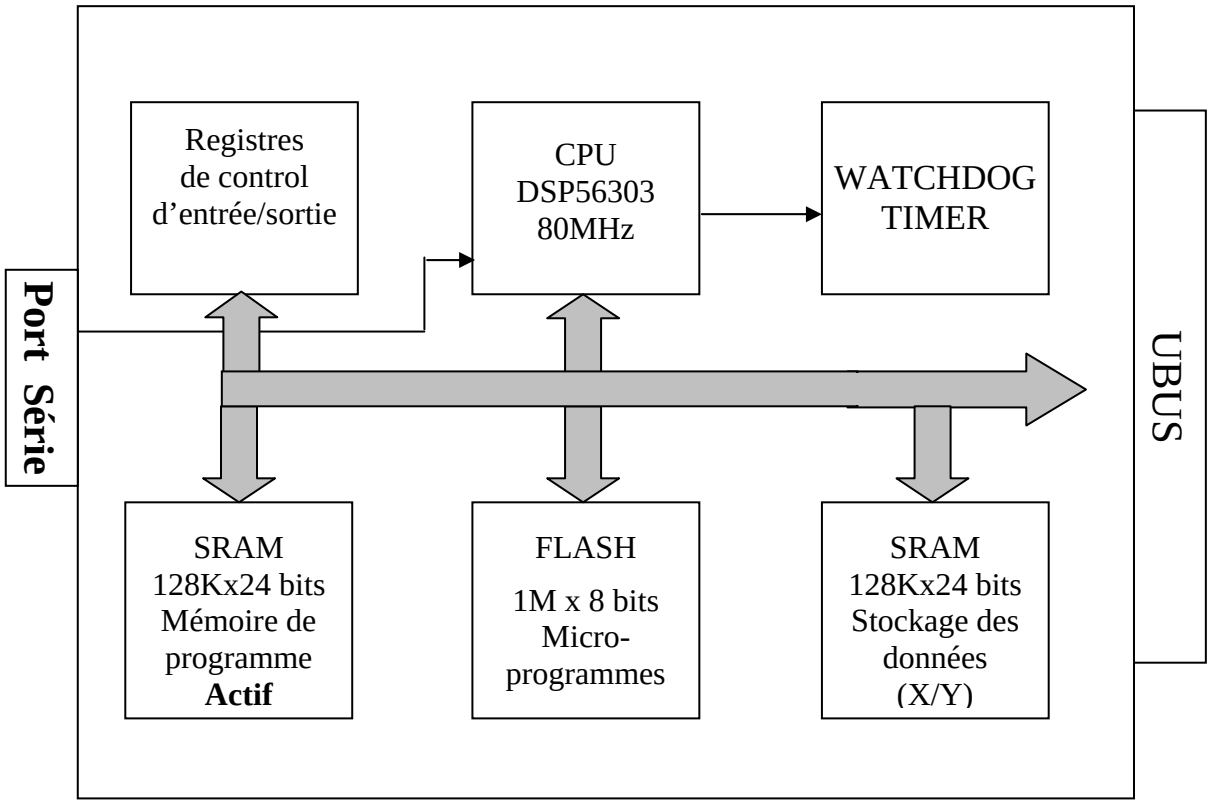


Figure III.4 : Diagramme en block du Turbo Pmac2 3U CPU

D'après ce schéma on distingue les principaux éléments suivants :

- Une unité centrale de traitement Motorola DSP56303 80 MHz 24 bits.
- Mémoire flash (1M x 8 bits) pour la sauvegarde d'utilisateur et les micro-programmes avec version 1.936.
- Mémoires SRAM (128k X 24 bits) actif pour les programmes, la compilation et l'assemblage.
- Mémoires SRAM (128k X 24 bits) de données d'utilisateur.
- Connecteur avec interface série RS-232/422.
- Whatchdoge timer.
- Registres de control.

III.3.1 DESCRIPTION DU DSP56303[18][19][20][21][22][23]

Le DSP56303 est un processeur de la famille DSP56300 spécialisé en traitement numérique du signal, commercialisé en 1996. Cette famille utilise un rendement élevé, avec un cycle d'exécution d'instruction augmenté doublement par rapport à la famille des DSP56000 tout en ayant le même code instruction. Les principales caractéristiques du DSP56002 sont les suivantes :

- 80 millions d'instructions par seconde (MIPS) à une fréquence d'horloge de 80 MHz.
- Architecture Harvard parallèle permettant l'exécution d'une instruction en parallèle avec l'accès mémoire.
- Technologie CMOS très faible consommation.
- Code instruction compatible avec le noyau DSP56000.
- Unité arithmétique et logique de données (ALU) avec un multiplieur-accumulateur parallèle de 24 x 24 bits fonctionnant en un cycle machine, et support arithmétique de 24-bits ou 16 bits sous la commande du logiciel.
- Unité de commande de programme (PCU), mode d'adressage optimalisé pour les applications de DSP, contrôleur d'instruction cache intégré.

III.3.2 TYPE DE MEMOIRE UTILISE[10]

Pour profité du parallélisme d'adressage du DSP56303, trois types de mémoire sont utilisés :

- Mémoire Flash
- Mémoire actif du programme
- Mémoire de données ("X / Y")

III.3.2.1 Mémoire Flash

Un circuit de mémoire Flash (instantané) est installé sur la carte qui forme la mémoire non-volatile pour les micro-programmes, les variables d'installation (variables I), les programmes utilisateurs, les tables de conversion, et les algorithmes de commande (par exemple algorithme de PID). Vue par le DSP56303, la mémoire flash est situé dans le champ

de mémoire " P ", ce circuit de taille 1M x 8 bits est le **28F008S3** de **Intel,** il est situé à U10 dans la carte.

III.3.2.2 Mémoire actif du programme

Un arrangement de trois circuits de mémoire SRAM (*Static RAM,* qui n'a pas besoin de rafraîchissement) sont installés sur la carte, ces circuits forment la mémoire de compilation/assemblage active (résultats et données intermédiaires) pour les micro-programmes, les PLCs compilé et les algorithmes de commande écrits par l'utilisateur. Vue par le DSP56303 les mémoires SRAM de compilation/assemblage sont situé dans le champ de mémoire " P ", ces trois circuits de taille 128K x 8 bits chacune, sont des **GVT73128A8J-10** de **GALVANECH,** ils sont situées dans U14, U15, et U16 dans la carte.

III.3.2.3 Mémoire de données ("X / Y")

Un arrangement de trois circuits de mémoire SRAM est installé sur la carte, ces circuits forment la mémoire active pour la sauvegarde des programmes de mouvement d'utilisateur, les programmes PLC non compilés et les données. Vue par le DSP56303 les mémoires SRAM de données sont situés dans les champs de mémoire " X " et " Y ", ces trois circuits de taille 128k x 8 bits chacune, sont des **TC55V8128B** de **TOSHIBA**, ils sont situés dans U11, U12, et U13 dans la carte.

III.3.3 CONNECTEUR SERIE RS-232/422[10]

Un connecteur série RS-232/422 DB25 est mis en place pour la communication séquentielle entre l'ordinateur principale et le Turbo PMAC2-3U CPU, cette communication se fait avec un protocole asynchrone en mode half-duplex (transmission en deux sens mais pas simultanément) et un format de donnée en code ASCII, la disposition des broches de ce connecteur est détaillé dans l'Annexe 1.

Cette fois si, les lignes de l'interface SCI (*Serial Communications Interface*) du DSP56303 sont utilisées, car les lignes RXD, TXD et SCLK du port série sont connecté à cette interface, après une adaptation des signaux (CMOS/TTL) faite par le circuit **MAX 8215** situé à U47.

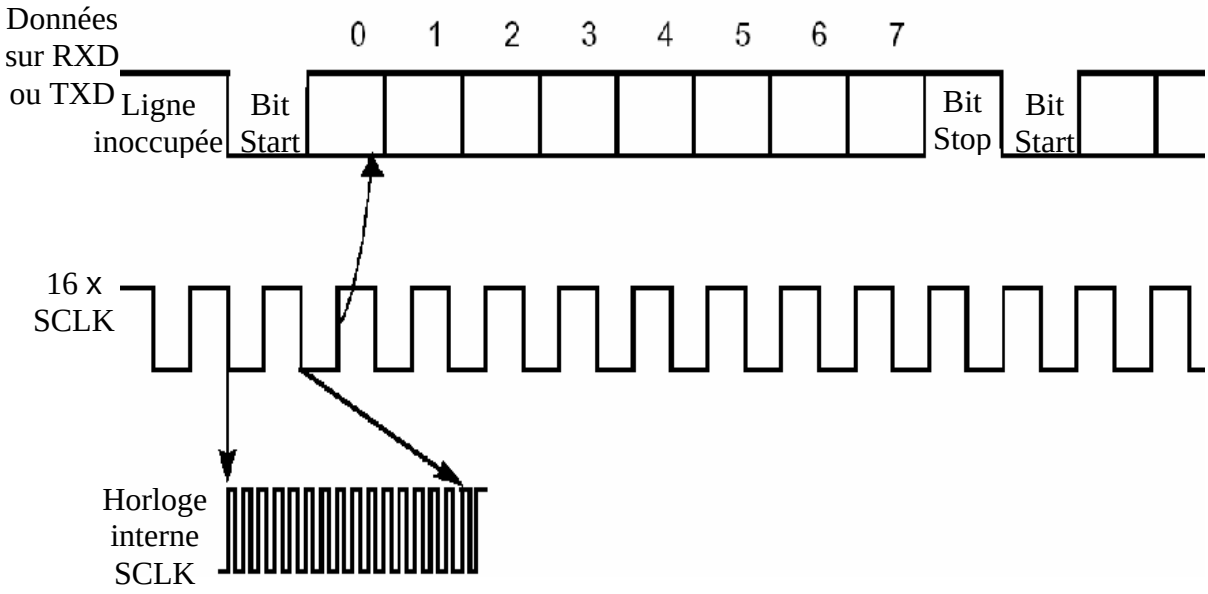


Figure III.5 ***: Transmission série des données***

Comme le montre la Figure III.5, l'émission et la réception utilisent une horloge interne qui est 16 fois la fréquence du débit des données au SCI. Le format de données exige que chaque octet de données à un bit de départ (Start) et un bit d'arrêt (Stop) sans bits de parité et une vitesse de transmission 38200 bits/sec (baud rate).

Par défaut, la connexion RS232 est utilisée, elle se caractérise par des signaux de transmission bipolaire ±12 V, ceci dit par configuration la connexion RS422 peut être utilisée pour des signaux de transmission différentielle 5 V.

III.3.4 WATCHDOG TIMER (CHIEN DE GARDE)[10][16]

C'est un système dont son travail est de détecter un certain nombre de conditions, comme une sous tension ou une faible fréquence d'exécution des programmes, qui pourraient avoir des conséquences dangereuses sur le fonctionnement de la carte ; Si c'est le cas il se déclenchera, et arrêtera la carte.

III.3.5 REGISTRES DE CONTROL

La représentation des registres de contrôle dans un block dans la Figure III.4 n'est que symbolique, car ces derniers sont répartis dans des zones mémoire bien séparé du CPU et

dans toutes les autres cartes (DSPGATE et IOGATE), ces registres fixe l'état des entrées/sorties ainsi que le fonctionnement du système global. Pour ce fait, des jumpers (Annexe 1) sont installés sur la carte ainsi des variables I accessibles par logiciel pour configurer l'état des registres de contrôle, en conséquence personnalise le fonctionnement.

III.4 DESCRIPTION DE LA DSPGATE[11]

L'unité centrale de traitement (Turbo Pmac2 3U CPU) communique avec les capteurs et les amplificateurs par un assemblement de circuit fait sur commande particulière de **DELTA TAU**, désigné sous le nom de DSPGATES (portes de la DSP). Nous disposons de deux DSPGATES nommées ACC-24E 4A, qui sont l'équivalent de deux ACC24E2, la Figure III.6 montre l'image d'une DSPGATE.

Figure III. 6 : DSPGATE ACC-24E 4A

Chaque DSPGATE contient quatre canaux (pour commander 4 moteurs), de ce fait, il y'a huit canaux qui peuvent être utilisés pour commander jusqu'à huit moteurs. Connectés à la surface arrière UBUS, l'ACC-24E 4A prend un total de deux slots (deux connections).

Une DSPGATE ne contient aucun processeur, il y'a quatre canaux contenant les circuits de conversion numérique/analogique DAC, compteurs, temporisateurs et filtres digitaux pour la commande en associant autour d'eux des buffers et des connecteurs pour les accès entrée/sortie.

Cette DSPGATE permet les types de commande suivants :

- Commandes en modulation de largeur des impulsions de ±10V.
- Commandes analogiques de couple de ±10V.
- Commandes d'impulsion -et- direction pour les moteurs pas à pas.

Chaque canal de ces DSPGATE est constitué par :

- 3 sorties pour les signaux de commande des amplificateurs configurables pour les types de commande suivant :
  - 3 sorties de commande en tension avec modulation de largeur des impulsions PWM.
  - 2 sorties de Convertisseur Digital Analogique DAC, commande en courant.
  - 1 sortie de signaux impulsion -et- direction pour les moteurs pas à pas.
- 3 entrées pour lignes de l'encodeur incrémentale.
- 8 indicateurs d'entrés, 2 indicateurs de sorties.

III.4.1 ENTREES ENCODEUR[24]

Il y'à trois entrées d'encodeur incrémental CHA, CHB et CHC, où comme le montre la Figure III.7 l'encodeur envoie des imputions quant il y'a un mouvement du moteur, pour la mesure de position, ces impulsions ont un rôle important, ceci est décrit par la suite :

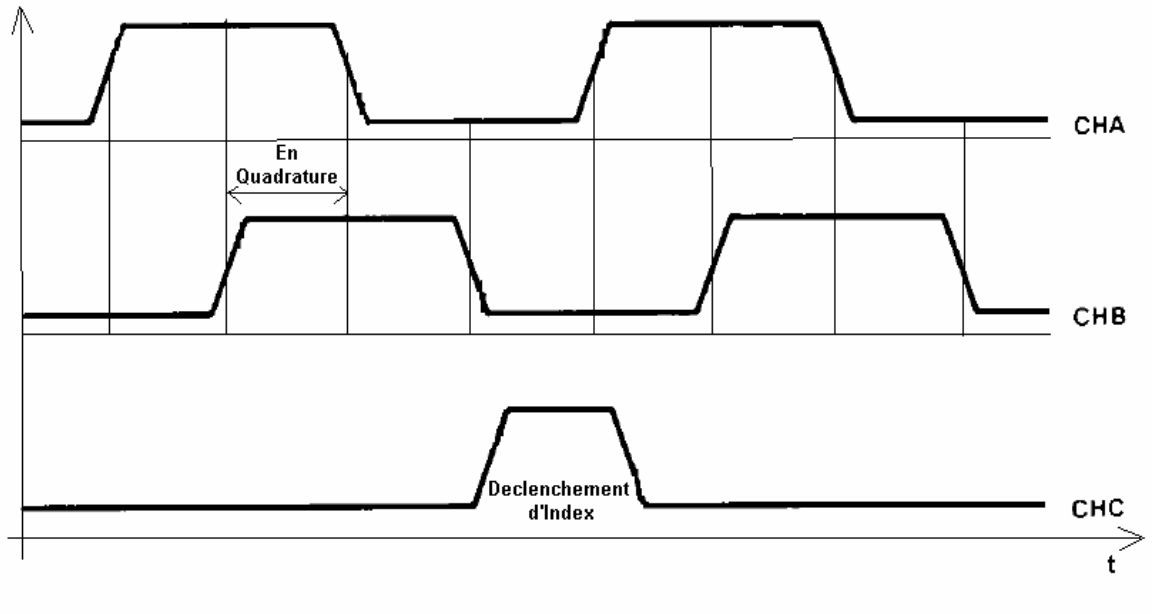


Figure III.7 : Signaux délivrés par l'encodeur lors d'un mouvement

- **CHA :** Quant le moteur tourne, le disque de l'encodeur tourne et la DPSGATE reçoit sur CHA des impulsions a chaque incrément, ces impulsions vont être acheminées vers un compteur et deux timers pour déterminer la position du disque et le temps correspondent respectivement.
- **CHB :** De même la DPSGATE reçoit sur CHB des impulsions à chaque incrément, sauf que cette fois ci ces impulsions sont en quadrature par rapport aux impulsions qui sont reçues sur CHA, ces impulsions sont importantes pour déterminer le sens de la rotation du disque de l'encodeur, car dans le cas ou ces impulsions seront en retard par rapport aux impulsions qui sont reçues sur CHA, le disque tourne dans le sens positif et le compteur qui est utilisé pour la mesure de la position s'incrémente par quatre (pour plus de précision), et dans le cas contraire alors le disque tourne dans le sens négatif et le compteur se décrémente par quatre. la détermination du sens de la rotation est fait matériellement par le décodeur installé dans la DSPGATE qui est formé par deux bascules D montés en parallèle comme il est montré par la Figure III.8.

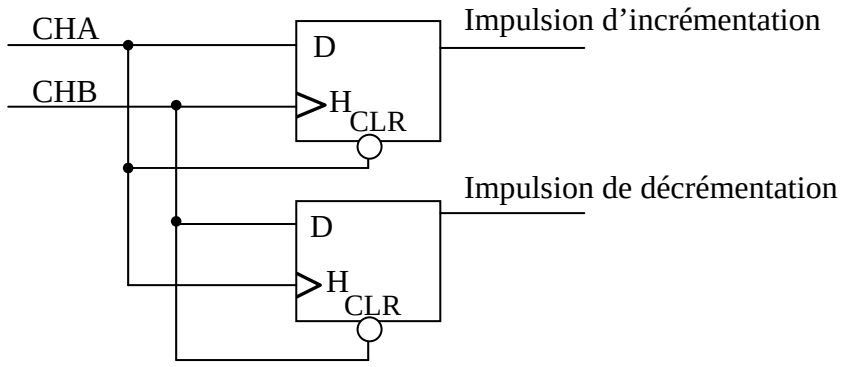


Figure III.8 : Décodeur du sens de mouvement implémenté dans la DSPGATE

- **CHC** : cette entrée représente l'index de l'encodeur qui reçoit une impulsion à chaque tour du disque de l'encodeur, cette dernière est importante pour la détermination du nombre exact d'incrément par tour (ou compte par révolution), comme c'est le cas pour les moteurs dont nous disposons qui ont un encodeur avec 500 CPR.

III.4.2 FILTRE DIGITAL[25]

Ce filtre est composé d'un ensemble de quatre bascules D montés en cascade sur chaque ligne d'entrée digitale, par exemple la Figure III.8 illustre le filtrage des lignes CHA et CHB des entrées d'encodeur.

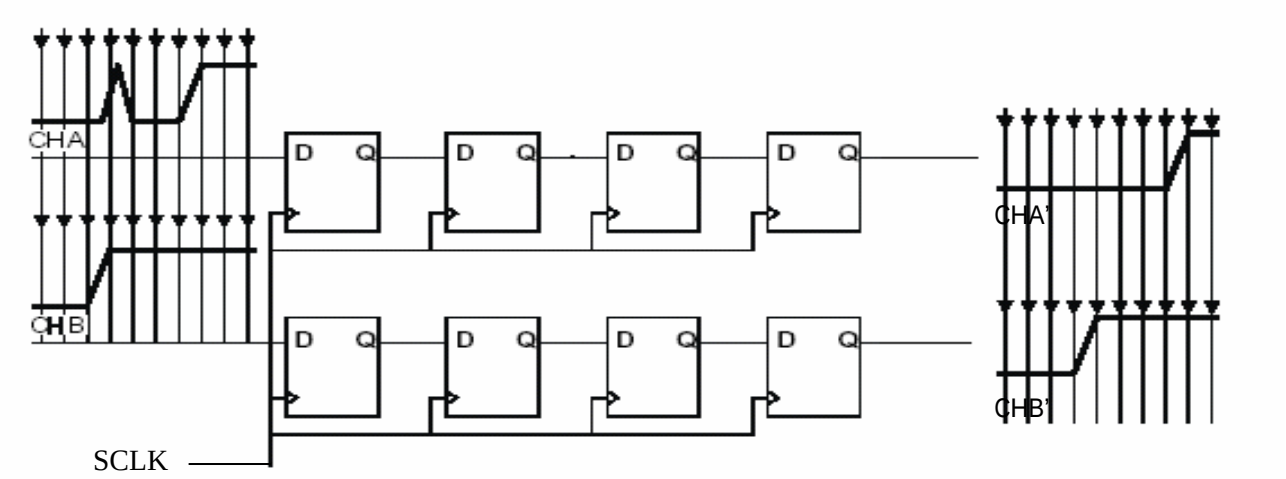


Figure III.8 : Filtre digital implémenté dans la DSPGATE

Les bascules D du filtre digital sont synchronisées par le signal d'horloge SCLK, de ce fait le rendement de ce filtre peut être amélioré par augmentation ou par diminution la fréquence de signal d'horloge SCLK.

III.5 DESCRIPTION DE L' IOGATE[12]

Nous disposons d'une carte IOGATE (*Input/Output GATE*) classée comme accessoire ACC12E de DELTA TAU, c'est une carte d'entrée/sortie d'usage universel au système UMAC, en format 3U conçus pour la transmission et la réception des données de l'extérieur du système au CPU, par l'intermédiaire de l'UBUS, elle fournit 24 lignes d'entrées optiquement isolées et 24 lignes de sorties à tension élevée qui sont aussi optiquement isolés, la lecture et l'écriture des données d'entrées/sorties est effectuée en utilisant les variables M, qui correspondent à des pointeurs de zone mémoire, la Figure III.18 montre l'image de cette carte.

Figure III.9 : IOGATE

III.5.1 CIRCUIT D'ENTREE

Les entrées de la carte IOGATE sont activées dans un intervalle de 12V à 24V, et elles peuvent être soit au mode sinking ou soit en mode sourcing (décrit auparavant), selon la tension de référence connectée à la carte, Le circuit d'entrée opto-isolation utilisé est le **PS2705-4NEC-ND,** il est composé de quadripôle comprenant un phototransistor à la sortie (Figure III.10), de ce fait, le circuit permet au courant de circuler librement dans les deux sens.

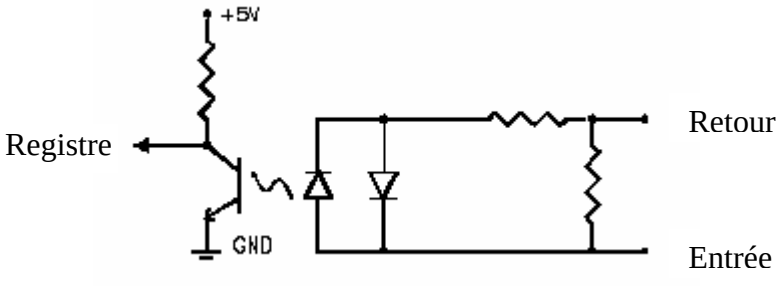


Figure III.10 : Circuit d'entrée de IOGATE

III.5.2 CIRCUIT DE SORTIES

Le relais semi-conducteur utilisé pour les sorties à courant continue est le **CRYDOM DMO063**, et pour les sorties à courant alternatif c'est le **CRYDOM ASO242**, pour chaque circuit le courant de sortie maximale est évalué à 1A.

Remarque : Les registres de DSPGATE et IOGATE sont vu par le CPU et plus exactement par le DSP comme étant des zones mémoire qu'on peut lire ou écrire.

III.6 Description de l'Amplificateur (AMP-2) [13]

Nous disposons de deux cartes d'amplificateur AMP-2 (quatre axes analogiques, amplification PWM, 48 VDC 3/5A) en format 3U prenant en globalité 4 slot (quatre connections), destinés pour être connectés à la DSPGATE ACC24E2A et pour contrôler des moteurs à courant continue ou pas à pas, chaque carte possède quatre axes d'amplifications avec une boucle de régulation analogique de courant (pour la régulation du couple du moteur à courant continue), et avec des signaux de sortie du type modulation en larguer des impulsions PWM. La tension maximum d'alimentation pour ces amplificateurs est de 40VDC, et l'estimation du courant maximum pour chaque amplificateur est de 3A continue pour une puissance maximale de 120W, l'image de cette carte est présentée par la Figure III.21.

Figure III.11 Amplificateur AMP-2

III.6.1 LIGNES AENA/FAULT

Comme pour les DSPGATEs, les ligne AENA/FAULT (*Amplifier Enable/Amplifier Fault*) sont utilisées pour la sûreté du fonctionnement. Alors la ligne FAULT est utilisée comme sortie à la DSPGATE pour indiquer s'il y à un défaut d'amplification, et la ligne AENA est utilisée comme une entrée pour arrêter instantanément l'amplificateur.

III.6.2 BOUCLE DE COURANT

D'après l'équation de base des moteurs, le couple est proportionnel au courant qui circule dans le moteur, c'est pour cette raison que ces amplificateurs sont destinés pour être commandé en courant (en couple) par le CPU, ils produisent un courant (respectivement couple) proportionnel à la tension d'entée de commande qui est produite par la DSPGATE par l'intermédiaire de la ligne DAC, tout en appliquant sur les bornes du moteur une tension avec modulation de largeur des impulsions PWM, de ce fait une boucle de courant avec un régulateur PI est implémenté.

L'élément principal de cette boucle est le circuit **LMD18200 3A, 55V H-bridge** de **Nationnal Semiconductor**.

III.6.2.1 Description du circuit LMD18200 [26]

Le LMD18200 est un pont en format H, conçu pour les applications de commande de mouvement, le dispositif est établi en utilisant un processus multi-technologique qui combine les circuits bipolaires et les circuits de commande CMOS avec des dispositifs de puissance DMOS sur la même structure monolithique, idéal pour piloter les moteurs a courant continue et les moteurs pas à pas, la Figure III.12 pressent le schéma interne du LMD 18200.

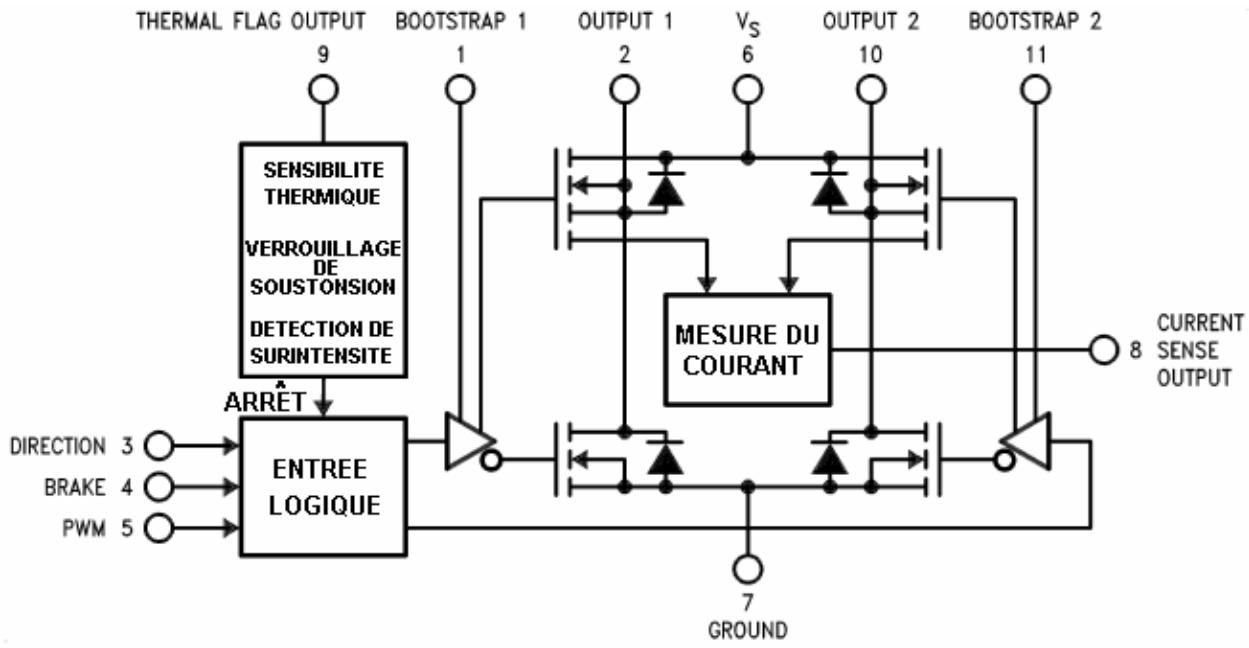


Figure III.12 : Schéma interne du LMD 18200

Comme le montre la Figure III.12, le LMD18200 possède 11 broches, les fonctions de ces broches sont :

- **Pin 1 et 11, BOOTSTRAP : S**ont des entrées où des condensateurs devront être branchés, ce qui est nécessaire pour le fonctionnement du circuit.
- **Pin 2 et 10, OUTPUT1 et OUTPUT2 :** ces sorties forment toutes les deux le signal de commande que doit recevoir le moteur en modulation de largeur des impulsions PWM.

• **Pin 3 et 5, DIRECTION et PWM :** Ces entrées forment toutes deux le signal de commande à amplifier.

• **Pin 4, BRAKE :** Cette entrée digitale est utilisée pour l'arrêt instantané de l'amplification en cas de nécessité, réellement cette entrée est branchée directement avec la sortie AENA de la DSPGATE pour que l'ordre d'arrêt soit donné par le CPU.

• **Pin 6 et 7, Alimentation électrique :** ces entrées sont utilisées pour l'alimentation électrique de puissance, car si la tension d'alimentation de ce circuit serait Vs, alors le circuit délivre un signal en PWM entre Vs et -Vs.

• **Pin 8, CURRENT SENSE :** cette sortie est nécessaire pour la fermeture de boucle du courant, car elle fournit la mesure de courant dans une tension proportionnelle à ce courant.

• **Pin 9, THERMAL FLAG :** cette sortie qui s'active à la température 145°C, est un indicateur de dépassement de la température acceptable pour l'environnement du circuit.

III.6.2.2 Fonctionnement de la boucle

Etant présent le courant de référence sur la sortie DAC de la DSPGATE, elle va se comparer avec le courant qui circule dans le moteur (Figure III.13), la différence passe par un filtre PI (Proportionnelle Intégrale).

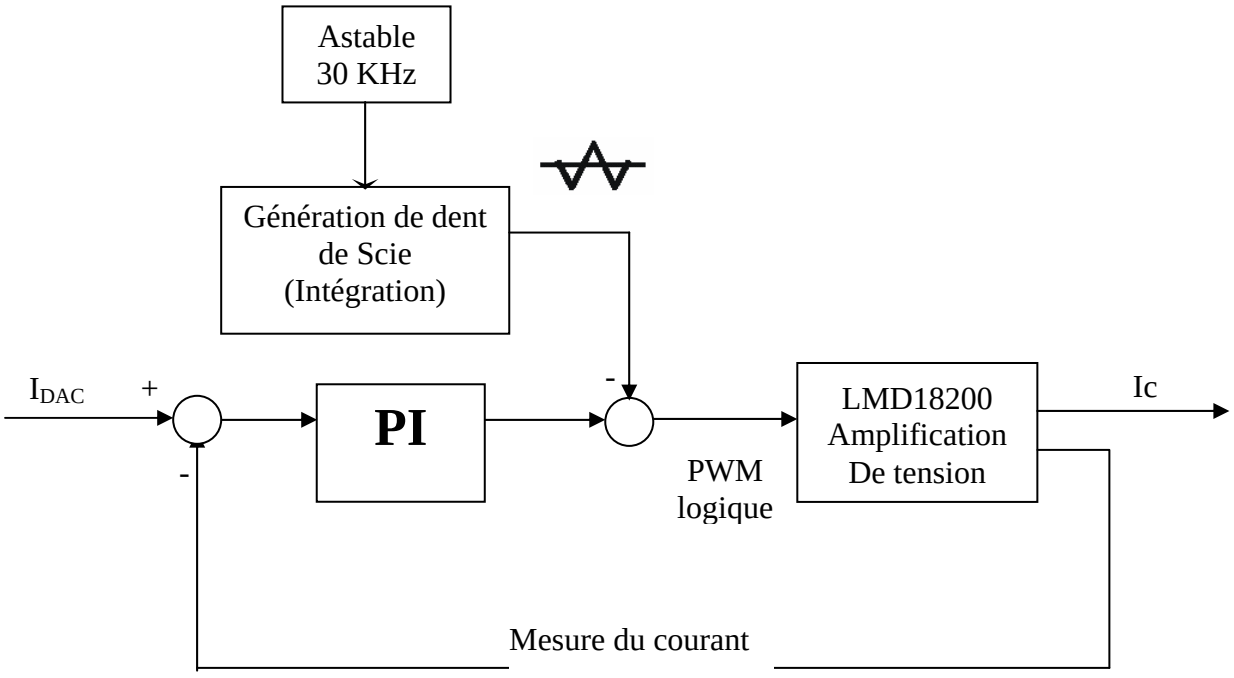


Figure III.13 : Schéma fonctionnel de la boucle de courant

La sortie du filtre va se comparer avec un signal en dent de scie de 30 KHz qui est générée séparément de la boucle et dépendamment de la tension d'alimentation, pour donner un signal de comparaisons logique, la Figure III.14 illustre cette opération.

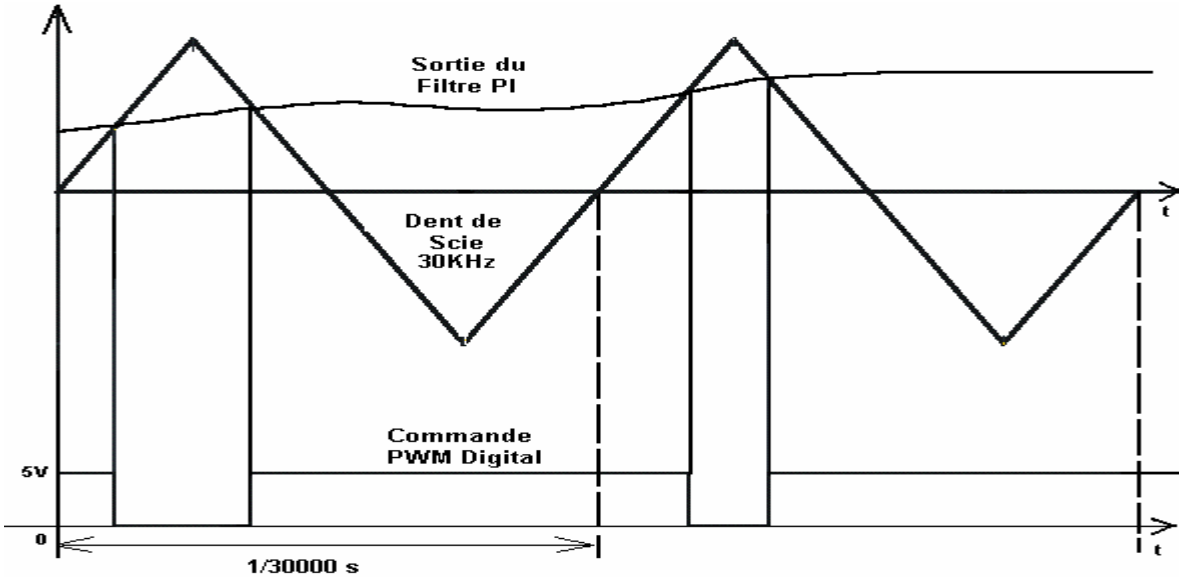


Figure III.14 : Présentation du principe de génération de la commande PWM

Le signal résultant de comparaison va être utilisé comme une entrée DIRECTION du circuit LMD18200 et l'entrée PWM sera mise en état haut, ce qui va donner sur la sortie OUTPUT1 le même signal amplifie et sur output2 le signal inversement amplifie, il en résulte la tension Vo1-Vo2 sur les bornes du moteur à commander, ceci est illustre par la Figure III.15.

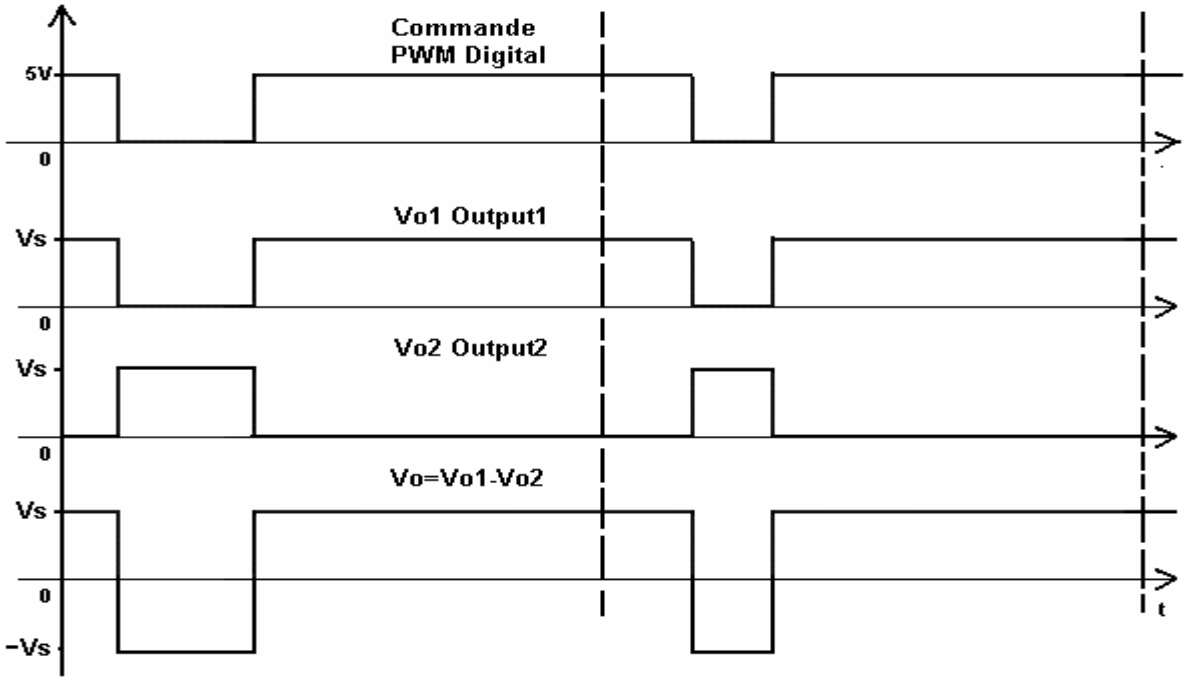


Figure III.15 : Amplification de la PWM digitale

III.6.2.3 Gain courant/tension

La transconductance (ou le gain courant / tension) de ces amplificateurs est par défaut 0.5A/V pour une tension qui ne dépasse pas ± 10V ce qui veut dire que le courant peut atteindre une valeur de 5A.

III.6.3 ALIMENTATION ET CONDITION DE FONCTIONNEMENT

L'alimentation électrique aux circuits numérique est de 15V fournie par l'intermédiaire de l'UBUS, et l'alimentation électrique nécessaire pour l'amplification est comprise entre 15V et 40V fournie par l'intermédiaire des lignes d'entrée BUS+ et BUS, le courant ne peut pas excéder 12A (3A par axe) en moyenne et 20A (5 par axe) en pique, une installation des fusibles de 5A a action lente est mise en place pour protéger les shunts (faible résistance), et des fusibles de 15A a action rapide pour une protection des lignes d'alimentation.

La température opérationnelle de ces amplificateurs est de 0^{o}C à 55^{o}C, et la température de stockage et de -12^{o}C à 82^{o}C pour un pourcentage d'humidité de 0% à 95%.

III.7 BUS DE CONNECTION (UBUS) [27]

Le désir d'avoir une meilleure flexibilité lors de la conception d'une application de contrôle de mouvement complète a causé le développement de l'UBUS (UMAC BUS), c'est un protocole de bus parallèle d'architecture ouverte, et qui est conçu comme interface simple et robuste pour câbler les différents dispositifs (module) en l'utilisant comme une surface arrière commune et familière (standard pour les produits DELTA TAU), cette interface crée un mécanisme uniforme par lequel un processeur peut contrôler toute les cartes qui sont dans le même système.

Les connecteurs des cartes sont de modèle **DIN41612** à 96 lignes, le nombre de slot désigne le nombre de connecteurs de ce type (indépendamment de l'alimentation électrique) qui sont présents sur l'UBUS, pour notre cas ont dispose de deux UBUS un pour connecter les amplificateurs avec quatre slots et un pour connecter le CPU, les DSPGATEs et IOGATE avec six slots comme le montre la Figure III.16.

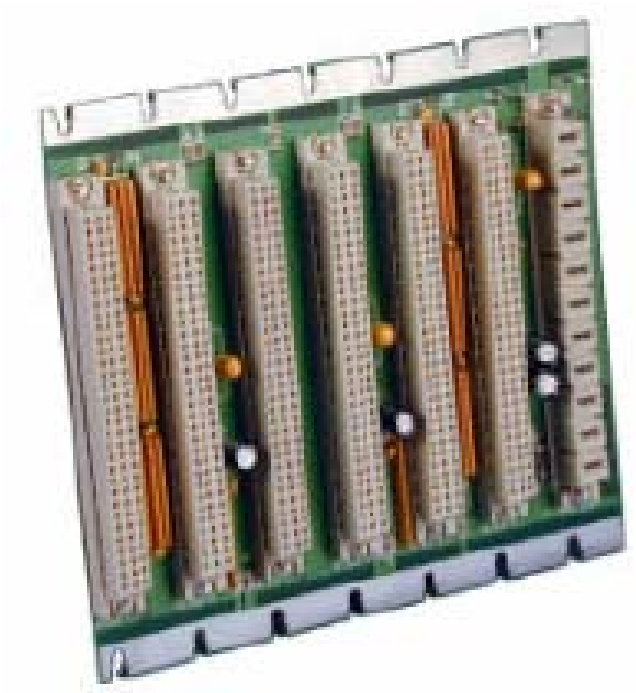


Figure III.16 : Image de l'UBUS

III.8 ALIMENTATION ELECTRIQUE [14]

C'est une carte qui fourni une alimentation à découpage électrique stabilisée de ±15V pour les circuits analogiques avec un courant maximal de 1.5A et 14A pour les circuits numériques. Elle fournit l'alimentation nécessaire pour toutes les cartes, de plus elle fournit deux masses l'une pour les circuits analogiques AGND (*Analogique Ground*) et l'autre pour les circuits numérique GND, ces deux lignes peuvent être rassembler pour une masse commune. La Figure III.17 montre l'image de la carte d'alimentation électrique de l'UMAC.

Figure III.17 : Carte d'alimentation électrique de l'UMAC

III.9 CONCLUSION

De ce qui précède, on voit bien que la disposition matérielle de l'interface UMAC est idéale pour concevoir des applications de commande de mouvement, sa conception montre la puissance qu'elle peut atteindre au niveau de l'exécution simultanée de nombreuses taches et la flexibilité de se placer dans différentes situations. Reste à l'exploiter convenablement, et pour cela une bonne connaissance du logiciel est requise.

IV.1 INTRODUCTION

Pour la gestion de n'importe quelle interface on à besoin d'un pilote approprié fournissant un environnement permettant la configuration ainsi que la programmation de cette dernière ; L'interface que nous disposons possède un logiciel de gestion combiné de plusieurs applications dont l'utilisation est classée par ordre. On peut accéder à toutes ces applications à partir de l'exécutable principale PEWIN32Pro développer par DELTA TAU.

PEWIN32 nous permet de configurer et de commander l'UMAC d'une manière très flexible en accédant à toutes ses variables internes. Il est conçu comme un outil de développement interactif pour créer des applications de mouvement avec l'UMAC, il contient un éditeur de programme de mouvement et de programme de PLC. De plus, PEWIN32 contient une suite d'outils pour configurer et travailler avec UMAC, ces outils contiennent des interfaces pour les différents types de moteurs et des fenêtres pour visionner : les diverses variables internes de l'interface, les registres d'états des moteurs en temps réel (telle que la position, la vitesse et l'erreur associée).

Il est recommandé que l'utilisateur fait appel aux applications accompagnant PEWIN32Pro par ordre, lors de l'installation initiale, afin de bien s'organiser lors de la configuration de l'application désirée :

- En premier lieu on cite l'application Turbo Setup Pro, c'est l'application où on peut définir l'état du fonctionnement de l'interface associé aux accessoires disponibles, ainsi que la configuration des paramètres d'une manière interactive toutes en visualisant les différentes valeurs affectées à ces derniers.
- En second lieu PMAC Tunig Pro, il est utilisé pour la configuration et l'amélioration des différents paramètres des boucles d'asservissement, avec laquelle on peut optimiser les gains numériques du régulateur PID de la boucle de position/vitesse implémentée dans l'interface, toute en visualisant les réponses des moteurs, et possibilité d'introduire les paramètres des filtres utilisés ainsi que le calibrage de l'offset.

- Pour avoir les profils de la position, la vitesse, l'accélération et l'erreur associée a la position commandée, de n'importe quelle trajectoire programmée, l'application PmacPlot Pro soit utilisé, elle permet l'acquisition des données et la présentation des profils sous forme de graphe.

IV.2 VARIABLES DE TRAVAIL [16][17]

Pour permettre à l'utilisateur de travailler dans un environnement adéquat et sûr, DELTA TAU a crée ses logiciels en mettant en point des variables qui permet de programmer et de configurer des applications sans avoir a s'encombrer avec les spécifications des adresses, ces variables ont chacune leur fonction et se devisent en quatre classes :

- **Variables I :** Permet d'accéder au registre de contrôle indirectement et de configurer l'interface pour les applications désirées.
- **Variables P :** se sont les variables utilisées pour le traitement des données (variables globales).
- **Variables Q :** se sont des variables utilisées comme des arguments pour les fonctions (variables locales).
- **Variables M :** Permet de pointer des adresses mémoires.

La disponibilité de ces variables permet aussi la séparation entre données et adresses, et entre registre de contrôle et registre de données, ce qui éloigne toute confusion.

IV.2.1 VARIABLES I

Les Variables I (variables d'installation ou d'initialisation) déterminent la personnalité de la carte mère de l'UMAC (c à d la carte Turbo PMAC2) et ses accessoires. De ce faite, pour une application donnée, il est nécessaire de les configurer correctement. Ils sont dans des endroits fixes dans la mémoire Flash et ils sont des significations prédéfinies. Notons que l'interface est configurée déjà par défaut pour être satisfaisante dans les types d'applications les plus communes. La plupart sont des valeurs de nombre entier (leur gamme change selon la particularité de la variable), comme ils peuvent être le résultat d'une expression.

Les variables I sont classées par catégorie pour un totale de 8191, dont on peut citer les gammes les plus importantes du point de vue utilisation :

- I0 - I99 : variable d'installation globale de l'interface.
- Ixx00 – Ixx99 : configuration du moteur xx pour les applications désirées.
- I7mn0 – I7mn9 : configuration du canal n de la DSPGATE numéros m.

(Pour plus de détail sur la spécification de chaque variable consulté le document [7]).

On peut accéder à ces variables directement par le biais de la fenêtre Terminale disponible dans le PEWIN32, par exemple si on veut avoir la valeur actuelle d'une variable quelconque on écrit simplement la ligne de commande I {constante} <RC> dans cette fenêtre, et l'interface retournera la valeur qui correspond à cette variable. Par contre si nous voulons la modifiée on écrit I {constante}= {constante} <RC>, ou si on veut affecter sa valeur par défaut on écrit I {constante}=* (pour tout un rang de variables on fait I {constante}.. {constante}=*), après avoir affecter une nouvelle valeur a une variable, il faut la sauvegarder par la commande SAVE dans le terminale.

Pour les applications particulières, il n'est pas nécessaire de chercher toutes ces variables car le logiciel communique avec l'interface « Turbo Setup Pro » et il nous fournie les étapes de configuration préliminaire adéquate pour chaque applications, toute en visualisant les valeurs affectées aux variables a chaque étape, (préliminaire car une configuration supplémentaire faite par l'utilisateur est nécessaire).

IV.2.2 VARIABLES P

Les variables P, comme les variables Q, sont des variables d'usage universel pour l'utilisateur, ils sont des variables réels stockées dans des cases mémoires dans l'interface sur 48-bit, son une utilisation est prédéfinie, pour un totale de 1024 variables.

Toutes les variables P sont accessibles par les systèmes en coordination (contrairement aux variables Q qui sont couplés avec), ceci tient compte de l'information utile passant entre les différents systèmes en coordination.

Si une constante est envoyée à l'interface comme une commande, elle sera affecter à la variable P0 ; par exemple, si nous envoyons la commande 342<CR>, l'interface l'interprétera

en tant que P0=342<CR>. Par conséquent, il n'est recommandé d'employer P0 pour d'autres buts que ce dernier, parce qu'il est facile de la changer accidentellement. La gamme pratique des variables Q à employer sans risque dans les programmes de mouvement est donc Q1 à Q99.

IV.2.3 VARIABLES Q

Les variables Q, sont des variables d'usage universel de 48-bit sans fonction prédéfinie. Cependant, la signification d'une variable Q donnée dépend du quel système en coordination (décrit si dessous) est utilisée.

Ceci permet à plusieurs systèmes en coordination d'employer le même programme (par exemple, contenant la ligne X(Q1+25) Y(Q2)) sans pour autant s'inquiéter du mélange de leurs valeurs. Plusieurs variables Q ont une utilisation particulière, comme par exemple :

- La fonction ATAN2 utilise automatiquement Q0 comme deuxième argument.
- La commande READ place les valeurs qu'elle lit après les lettres A à Z dans Q101 à Q126, respectivement.
- La commande S dans le programme de mouvement place sa valeur dans Q127.

IV.2.4 VARIABLES M

Les variables M sont fournies pour permettre un accès facile de l'utilisateur à la mémoire interne de l'interface et aux registres d'entrées/sorties. Elles sont affectées à des locations mémoires dont l'utilisateur peut définir la taille (en bits), le format (virgule fixe ou flottante), et la valeur attribuée à cette location. D'une façon générale, une définition d'une variable M doit être faite une fois seulement par une commande en ligne (la commande SAVE doit être utilisée pour la sauvegarde de la définition).

Comme les autres variables, elles peuvent être attribuées par une constante ou une expression : M576 ou M(P1+20). La définition d'une variable M est faite en utilisant le symbole '' -> '', elles peuvent prendre un des types de préfixe suivants :

- X : 1 a 24 bits virgule fixe dans la mémoire X

- Y : 1 a 24 bits virgule fixe dans la mémoire y
- D (double) : 48 bits virgule fixe à travers les deux mémoires X et Y.
- L (long) : 48 bits virgule flottante a travers les deux mémoires X et Y.
- DP : 32 bits virgule fixe (bit de poids faible de X)
- *: Pas de définition d'adresse ; utilisation comme variable générale.

La définition d'une variable M est faite par une définition du bit de début, le nombre des bits et le format. On cite si dessous un exemple de définition typique des variables M :

M12->X:$C003,0,24,S
M01->D:$002B,U
M89->DP :$Dff2

Où $: adresse spécifié en hexadécimale.
S : signifie que le nombre est signé.
U : spécifie que le nombre est non- signé.

Par la suite, les définitions des variables M sont stockées dans des cases mémoires situées dans le champ Y à l'intervalle Y:$BC00 (pour M0), Y:$BFFF (pour M1023), les 16 bits de poids faible contiennent l'adresse du registre pointé par la variable M, et les 8 bits du poids fort indiquent quelle partie de l'adresse est employée et comment l'interpréter.

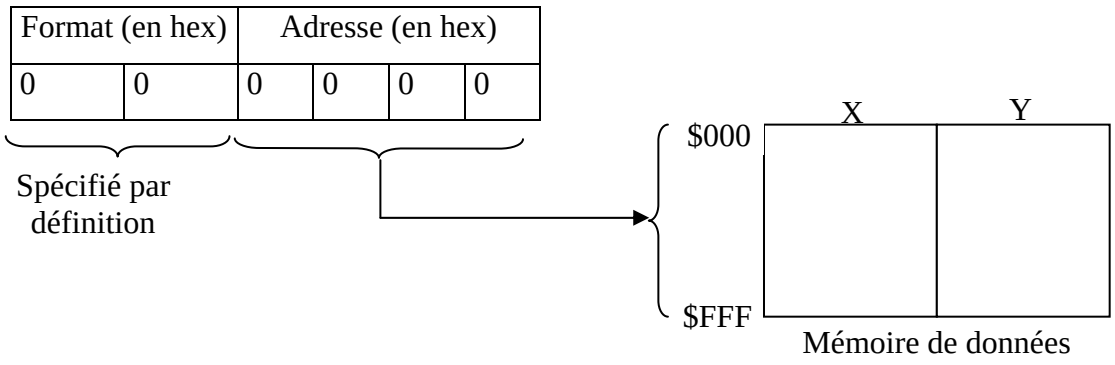


Figure IV.1 : Format de la définition d'une variable M

Une fois définie, une variable M peut être employée dans les programmes justes comme n'importe quelle autre variable – par des expressions. Quand l'expression est évaluée, le

processeur lit l'endroit de mémoire défini, calcule la valeur basée sur la taille et le format définis, et l'utilise dans l'expression.

La prudence dans utilisation des variables M est exigée, car celle ci peut être changée sur l'évaluation d'une expression au cours de l'exécution d'un micro-programme.

Par exemple, si dans l'expression « (M16-M17)*(M16+M17) » les variables M sont lus instantanément de la mémoire, il n'est pas sûr que M16 ou M17 auront la même valeur pour les deux termes dans l'expression précédente, ce problème peut être surmonté en affectant P1=M16 et P2=M17.

IV.3 LES SYSTEMES EN COORDINATION[16][17]

Un système en coordination est un groupement de moteurs élaboré afin d'avoir la même synchronisation du mouvement. L'exécution d'un programme de mouvement nécessite la définition des systèmes en coordination (même avec un seul moteur). Avec l'UMAC on peut avoir jusqu'à huit systèmes en coordination (extensible jusqu'à 32), adressés par &1 jusqu'à &8, avec un mode très flexible (exemples : 8 systèmes en coordination contenant un moteur chacun, un système en coordination contenant 8 moteurs, 4 systèmes en coordination contenant 2 moteurs chacun, … etc).

En générale si on veut que certains moteurs bougent en mode coordonnée, on les met dans le même système en coordination, et si on veut les faire bouger indépendamment les un des autres, on les met dans des systèmes en coordination séparés.

Les systèmes en coordination peuvent exécuter diffèrent programmes de mouvement indépendamment, mais aussi d'exécuter le même programme à des moments différent ou simultanément. Un système en coordination doit être d'abord établit en affectant des axes aux moteurs dans un rapport de définition d'axe (décrit ci dessous), cependant, il doit y avoir au moins un moteur attaché à un axe, sinon il ne peut pas exécuter un programme de mouvement.

V.4 PROGRAMME DE MOUVEMENT[16][17]

C'est un programme qui contient des informations et des lignes de commande où le processeur de l'UMAC les utilise pour planifier et déclencher le mouvement, l'interface peut supporter jusqu'à 256 programmes de mouvement en même temps (avec extension de 7 autres cartes Turbo PMAC2- 3U). N'importe quel système en coordination peut exécuter n'importe lequel de ces programmes à tout moment, même si un autre système en coordination exécute déjà ce même programme.

Un programme de mouvement peut appeler n'importe quel autre programme de mouvement comme sous-programme, avec ou sans arguments. Le langage de programme de mouvement de UMAC peut-être mieux décrit comme couplage entre le langage évolué et de langage machine.

IV.4.1 DEFINITION D'AXE

Un axe de point de vue logiciel est un élément important des systèmes en coordination, il se réfère à un moteur par une lettre alphabétique sélectionnée parmi X, Y, Z, A, B, C, U, V, et W de telle façon à ce qu'une affectation d'une valeur à cette lettre corresponde à un déplacement du moteur.

Un axe est défini en l'affectant à un moteur avec un facteur de graduation et un offset (notons que X, Y, et Z peuvent être définis en tant que combinaisons linéaires de trois moteurs, de même que pour U, V, et W).

La plus simple définition est : #1->X (attaché le moteur #1 à l'axe X pour le système en coordination courent), maintenant si l'axe X exécute un mouvement, le moteur #1 fera ce mouvement. Pour affiner une application (de point de vue réglage et précision) il faut définir les caractéristiques d'axe suivantes :

- Graduation d'axe.
- Offset.
- Type d'axe.

IV.4.1.1 Graduation d'axe

Dans cette partie, on définit une nouvelle unité d'utilisation (en se basant sur un certains nombres d'incrémentation de l'encodeur) dont elle sera la nouvelle résolution de déplacement du moteur.

Exemple : Nous disposons d'un encodeur de 500 CPR (compte par révolution) avec un moteur dont le rapport de réduction est $\eta = 65.5$. Etant donné que la résolution est augmentée de 4 fois par l'interface (suivant la configuration de I7mn0), ce qui signifie qu'on a 2000 compte par tour d'encodeur, alors une rotation complète de l'arbre du moteur corresponde à : $\eta * 2000 = 131000$ comptes.

Ce qui fait que l'affectation « #1->131000X » permet de travailler avec une unité de déplacement du moteur #1 égale à un tour.

IV.4.1.2 Offset

Par exemple le rapport #1->10000X+20000 place un offset (de 2 unités d'utilisateur) de la position zéro du moteur #1, notons que cet offset est rarement utilisé.

IV.4.1.3 Type d'axe

Pour la définition de ce type d'axe, il peut y avoir deux attributs géométriques où chacune correspond à un mode de mouvement dans l'espace, le troisième type est une propriété :

- Les axes cartésiens.
- Les axes de rotation.
- Les axes fantôme.

Notons que pour la plupart des applications, le choix du type d'axe n'est pas nécessaire. Cependant, pour certaines applications comme l'implémentation du modèle géométrique d'un robot, on ne peut utiliser que les noms particuliers X, Y, Z, U, V et W qui correspondent aux axes cartésiens.

A) Axes cartésiens

Ce type d'axe peut être groupé suivant une combinaison linéaire de deux ou trois axes pour causer le mouvement de deux ou de trois moteurs : X, Y, Z forment la première base, U, V, W forment la seconde.

Exemple : #1->8660.25X-5000Y
#2->5000X+8660.25Y ; avec : cos (30°) = 0.866 et sin (30°) = 0.5.

Dans ce cas le mouvement d'un seul axe X ou Y cause le mouvement des deux moteurs avec un écart final de 30° entre les deux axes des moteurs.

Remarque : On ne peut avoir d'autres combinaisons entre les axes. C'est à dire non- linéaire, cependant, les relations non- linéaires qui peuvent exister entre les axes se feront dans des programmes spéciaux.

B) Axes de rotation (A, B, et C)

Quand on affecte un moteur à ces axes, la fonction *rollover* du moteur est activée, et quand le programme fonctionne en mode absolue (ABS), cette fonction permet au moteur de prendre la plus courte distance du point de départ au point destiné au tour d'un rang spécifié par Ixx27.

Exemple : I127 = 131000 compte par tour
#1->363.889A ; A est dans les unités des degrés

Dans ce cas, si on applique la commande de mouvement 'A270' dans un programme en mode ABS, le moteur fait le mouvement de 0 à -9000 (-90°), au lieu de faire le mouvement de 0 à 27000 (ce qui l'aurai fait avec I127 = 0).

Remarque : Contrairement aux axes linéaires, ces axes ne peuvent pas être dans une combinaison linéaire.

C) Axes fantôme

Un axe peut n'avoir aucun moteur attaché ''un axe fantôme'', bien que les mouvements programmés pour cet axe ne causent aucun mouvement, le fait qu'un mouvement a été programmé pour cet axe, il puisse affecter les mouvements d'autres moteurs. Par exemple, si on désire d'avoir des profils sinusoïdaux sur un axe simple, la manière la plus facile de le faire, est d'avoir en second lieu un axe fantômes et a programmé des mouvements d'interpolation circulaire (voire si dessous).

IV.4.2 ELABORATION D'UN PROGRAMME DE MOUVEMENT

Les commandes usuelles pour l'élaboration d'un programme sont :

1) OPEN PROG {constante} : Ouvre un programme dans un buffer sur l'UMAC, où la constante représente le numéro du programme, et qui varie entre 0 à 32767 (de 0 à 2^{15}-1).
Le programme 0 assure le chargement des lignes de programmes. Pour les longs programmes le chargement se fait au fur et à mesure de l'exécution de ce dernier.

2) CLEAR : Efface le contenu du programme ouvert.

3) LINEAR, RAPID, CIRCLE, PVT, et SPLINE : Ces commandes servent à définir les trajectoires (profil de vitesse).

4) INC et ABS : indiquent le mode de mouvement, la commande INC (mode incrémentale) définie un déplacement relatif à la position actuelle, la commande ABS (mode absolu) définie un déplacement absolu.

5) Commande de mouvement : Un mouvement est défie par la spécification d'un axe suivi d'un déplacement.

Tous les mouvements spécifiés dans une même ligne de commande seront exécutés simultanément, tandis que les lignes consécutives s'exécuteront séquentiellement.

6) Paramètres temporels : Si les temps de mouvement TA (temps d'accélération), TS (temps de courbure pour le mode S-Curve), TM (temps de mouvement), et/ou la vitesse F (Feedrate) ne sont pas déclaré, les valeurs par défaut sont fournies par les paramètres Ixx87, Ixx88, et Ixx89.

Cependant, il est recommandé de ne pas compter sur ces paramètres et à déclarer les temps de mouvement dans le programme. Ceci permettra une meilleure maintenance du programme.

7) Instructions usuelles : Dans un programme de mouvement, on peut faire appel aux instructions d'itérations condition tel que la boucle WHILE, IF, ELSE ainsi que les commande de branchement tel que GOTO, et la commande GOSUB : qui signifie branchement au sous- broutine, on peut aussi faire appel à un autre programme avec la commande CALL à partir de n'importe quelle ligne du programme, exemple :

CALL 20.150 : entrer dans le programme 20 à la ligne 150.

8) CLOSE : ferme le buffer ouvert. Il doit être utilisée immédiatement après l'entrée d'un programme de mouvement, sinon toutes commandes en ligne (écrites dans le Terminale Windows) risquent d'entrer dans le programme. Pour éviter ceci il est recommander d'inclure CLOSE au début et à la fin de chaque programme à télécharger à UMAC.

Exemple :

```
CLOSE                  ; ferme tous les buffers ouverts.
DELETE GATHER          ; Effacer toutes les données intermédiaires.
UNDEFINE ALL           ; Effacer les définitions lié a tous les système en coordination.
#1-> 363.889A          ; lier le moteur #1 a l'axe A avec une graduation d'axe en degré
                       ;  65.5*2000 / 360 = 363,889.
OPEN PROG 1 CLEAR      ;  Ouvrir le buffer pour écrire le programme.
LINEAR                 ; interpolation Linéaire (le profile de vitesse est linéaire).
INC                    ; mode Incrémentale.
F50                    ; le Feedrate est 50 unités par I5x90 msec.
```

X180 ; 180 unités de distance : 180°.

CLOSE ; ferme le buffer, programme 1.

I5x90 : doit être initialisé à 1000 pour avoir une échelle de temps en msec pour le système en coordination A.

IV.5 ELABORATION DES PROGRAMMES GEOMETRIQUES DIRECTES (GD)

```
&1 OPEN FORWARD
CLEAR
    {Les instructions & les fonctions du modèle }
CLOSE
```

Les fonctions du modèle décrivent les équations qui existent entre les coordonnées cartésiennes et articulaires.

Avant n'importe quelle exécution du programme géométrique direct, UMAC placera les positions actuelles pour chaque moteur xx, dans le système en coordination, dans les variables globale Pxx, le programme peut alors employer ces variables en tant qu'entrée dans les calculs.

Après n'importe quelle exécution du programme géométrique direct, UMAC prendra les valeurs dans Q1– Q9 pour le système en coordination et copie ces derniers dans les 9 registres d'axes contenus dans ce système en coordination.

Variable Q	Axe affecté	Variable Q	Axe affecté	Variable Q	Axe affecté
Q1	A	Q4	U	Q7	X
Q2	B	Q5	V	Q8	Y
Q3	C	Q6	W	Q9	Z

Tableau IV.1 : Correspondance des variables Q avec les axes

Le but de base du programme géométrique directe, est de prendre les valeurs des coordonnées articulaires trouvées dans P1 – P32 pour les moteurs utilisés dans le système en

coordination, et calculent les valeurs cartésiennes correspondantes, et les placent dans les variables Q1- Q9.

IV.6 ELABORATION DES PROGRAMMES GEOMETRIQUES INVERSES (GI)

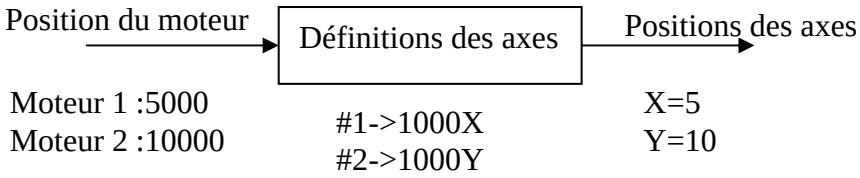
&1 OPEN INVERSE
CLEAR
{Les instructions & les fonctions du modèle inverse}
CLOSE

Avant n'importe quelle exécution du programme géométrique inverse, UMAC placera les positions actuelles pour chaque axe dans le système en coordination dans les variables Q1– Q9, le programme peut alors employer ces variables en tant que " entrées " dans les calculs.

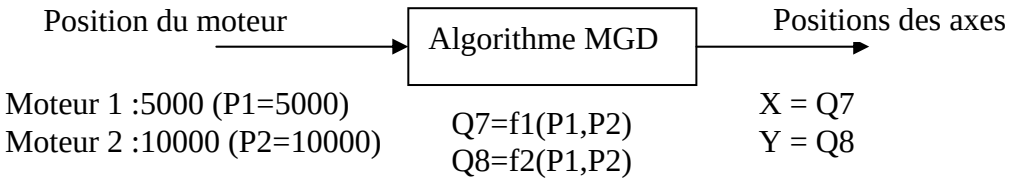
Après n'importe quelle exécution du programme géométrique inverse, UMAC lira les valeurs dans les variables Pxx (P1– P32) correspondent aux moteurs xx, dont la commande de définition d'axe est # xx->I.

IV.7 RECAPITULATIVE DE L'EXECUTION DE LA CONVERSION

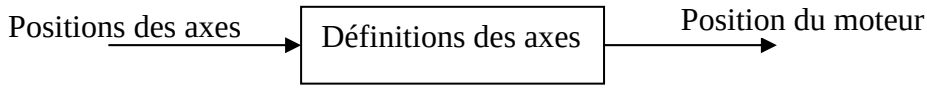
- Conversion du moteur a l'axis sans le programme de GD:



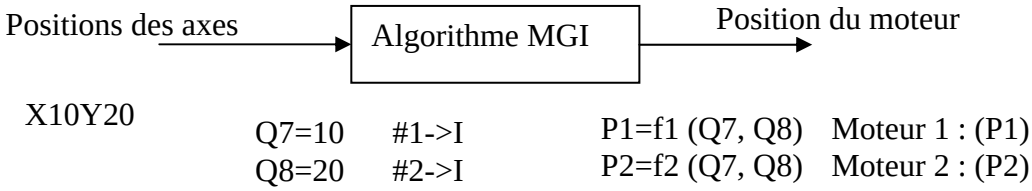
- Conversion du moteur a l'axis avec le programme de GD :



- Conversion de l'axe au moteur sans le programme de GI :



- Cc X10 Y10 le l'axe au m #1->1000X e progra Moteur 1 :10000
  #2->1000Y Moteur 2 :20000



IV.8 CONCLUSION

La description du langage de programmation de l'interface nous a permet de voir les grandes capacités de l'UMAC dans le domaine de programmation, elle peut supporter jusqu'à 32K programmes de mouvement et exécuter jusqu'à 256 programmes de mouvement simultanément (grâce à une extension des cartes Turbo PMAC2-3U CPU) avec une très grande précision. La disposition des variables et leurs classifications engendre un mode très flexible pour la configuration et la programmation de l'interface UMAC grâce au logiciel PEWIN32PRO développé par la firme **DELTA TAU** qui fournie un environnement interactif pour l'utilisateur permettant l'accès à toutes les variables internes de l'interface.

V.1 INTRODUCTION

Dans ce chapitre on traite initialement l'asservissement en position du moteur à courant continu en se basant sur le modèle du moteur *Pittman GM9236S027* .Cela est réalisé en premier lieu avec des régulateurs P et PD continus et par la suite par des régulateurs P et PD discrets qui vont nous permettre d'approximer les régulateurs à implémenter sur l'UMAC.

V.2 ASSERVISSEMENT EN POSITION DU MOTEUR À COURANT CONTINUE

On a la fonction de transfert du moteur à courant continu entre la position et la tension :

$$\theta(s) = \frac{K_T s}{K_T K_E + (R_a + L_a s)(K_f + Js)} U_a - \frac{s(R_a + L_a s)}{K_T K_E + (R_a + L_a s)(K_f + Js)} C_r \qquad (5.1)$$

V.2.1 ASSERVISSEMENT DU COURANT

Pour asservir le courant, on utilise un simple régulateur proportionnel dont sa fonction de transfert est :

$$G_P(s) = K_{pc} \qquad (5.2)$$

V.2.2 ASSERVISSEMENT EN POSITION

Pour cela, on introduit un régulateur PD continu qui permet d'éliminer les oscillations. Entre autre sa fonction de transfert est donnée par la relation suivante :

$$G_{PD}(s) = K_{pp} + K_{dp} s \qquad (5.3)$$

V.3 SIMULATIONS

On a effectué des simulations pour des faibles déplacements et grands déplacements, nous obtenons ce qui suit **(figure V.1)** :

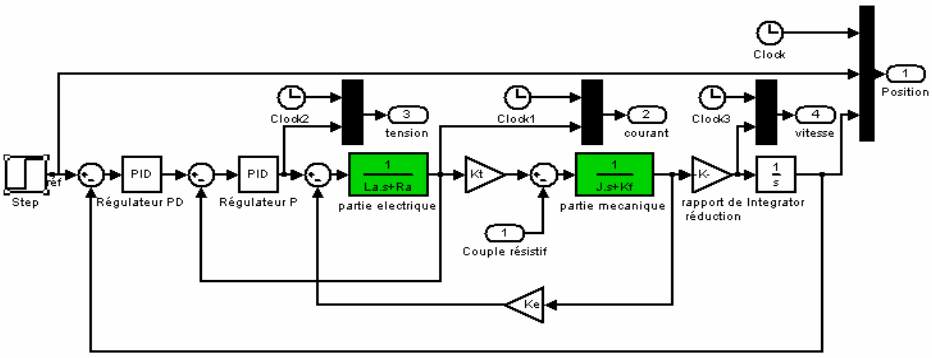


Figure V.1 : schéma bloc d'asservissement du moteur

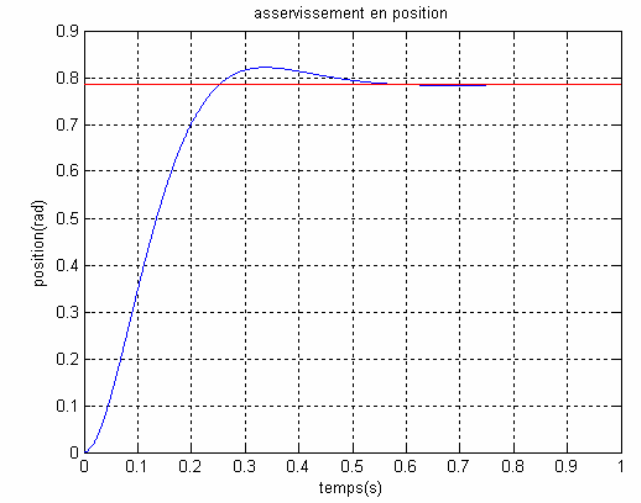
Les schémas suivants représentent l'asservissement du moteur par les deux régulateurs P et PD

- Pour une référence

$$\theta_d = \frac{\pi}{4}$$ **(Figure V.2)**

Ainsi les paramètres de régulation

$$K_{pp} = 2; K_{dp} = 0; K_{pc} = 15.$$



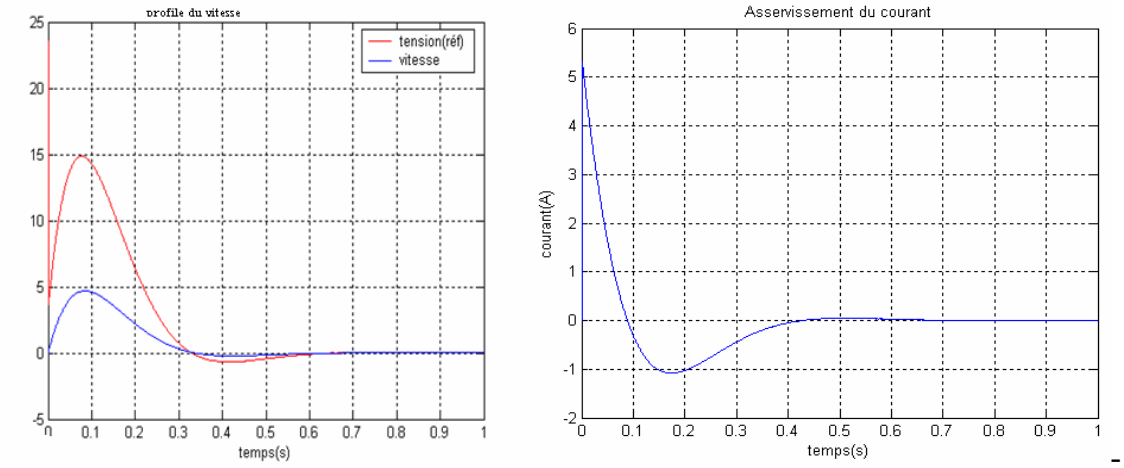


Figure V.2 : asservissement en position, courant ;et profile du vitesse

$$\text{Pour } \theta_d = \frac{\pi}{4}, K_{pp} = 2; K_{dp} = 0; K_{pc} = 15.$$

Pour la même consigne, mais avec un coefficient de dérivation égal à :

$K_{dp} = 0.125$, **(Figure V.3)**

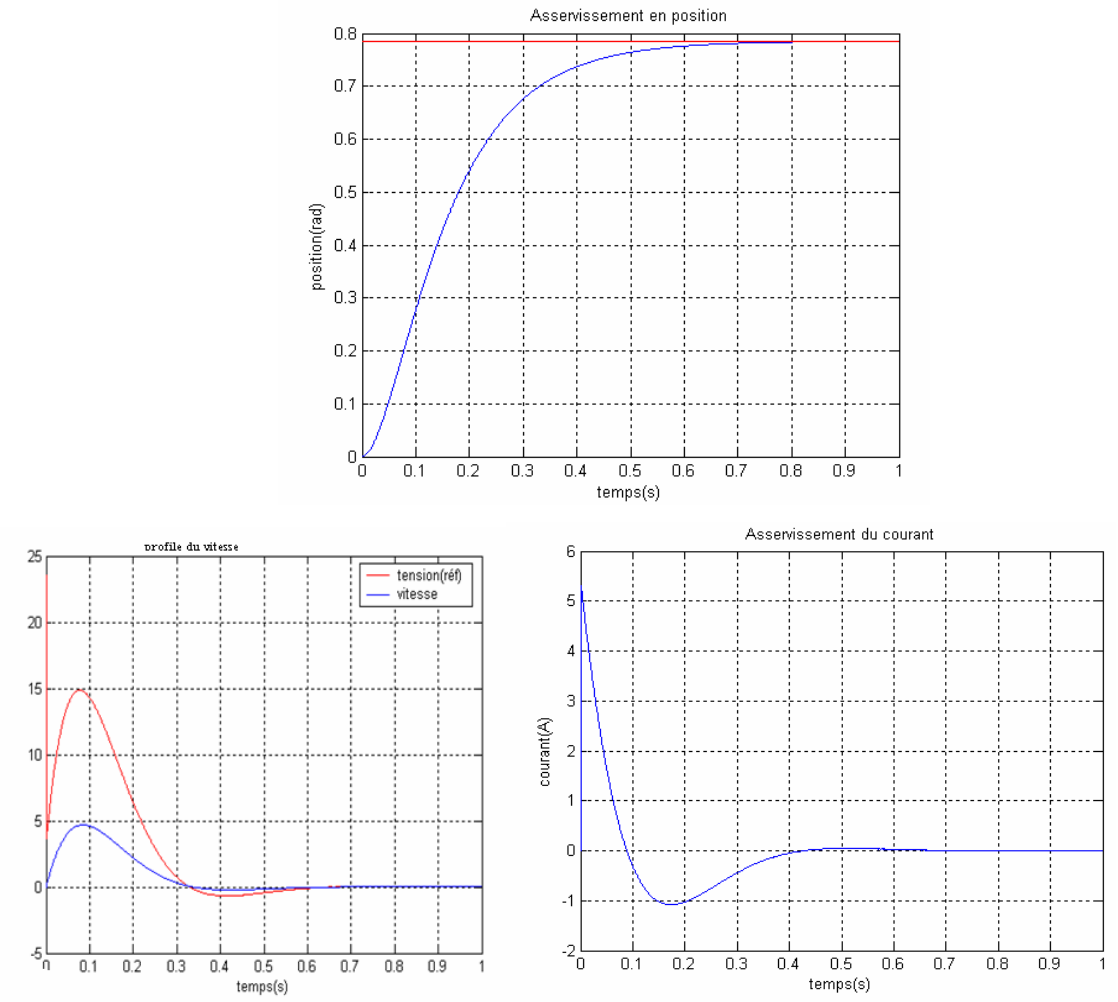


Figure V.3 : asservissement en position, courant ;et profile du vitesse

$$\text{Pour } \theta_d = \frac{\pi}{4}, K_{pp} = 2; K_{dp} = 0.125; K_{pc} = 15. \oplus$$

On remarque que pour ces paramètres de régulation, le courant atteint la valeur 5.2A or que le courant maximale de ce moteur est de 9.64A. Entre autre la tension à l'instant t=0 atteint la valeur 24V, cependant la tension maximale est de 24V.

On déduit que $K_{pc} = 15$ est le gain maximal à ne pas dépasser pour $\theta_d = \frac{\pi}{4}$.

On constate aussi que le coefficient K_{dp} permet de diminuer les oscillations, cependant si on augmente sa valeur le système devient de plus en plus long.

- Pour une consigne de:

$$\theta_d = \pi$$

Ainsi les paramètres de régulation :

$K_{pc} = 3; K_{dp} = 0.5; K_{pp} = 2$, **(Figure V.4)**

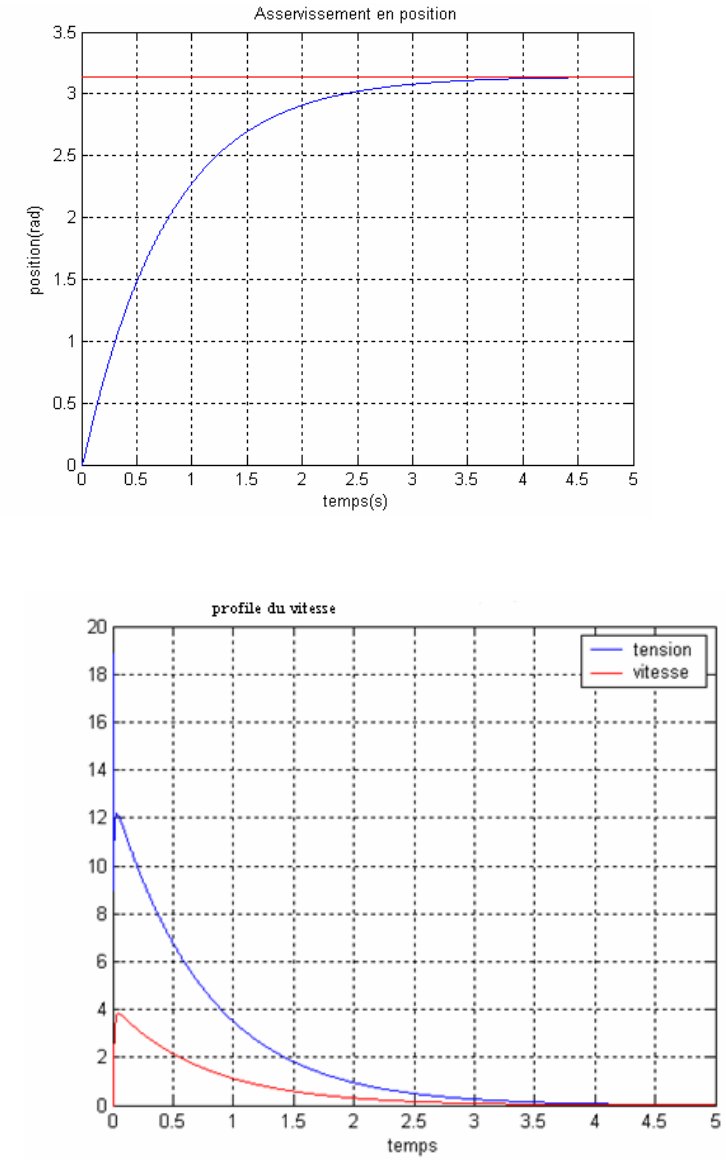


Figure V.4 : asservissement en position ;et profile du vitesse
Pour $\theta_d = \pi$, $K_{pp} = 2; K_{dp} = 0.5; K_{pc} = 3.$

- Pour une référence de :

$\theta_d = 2*\pi$, **(Figure V.5)** ; avec les même paramètre de régulation.

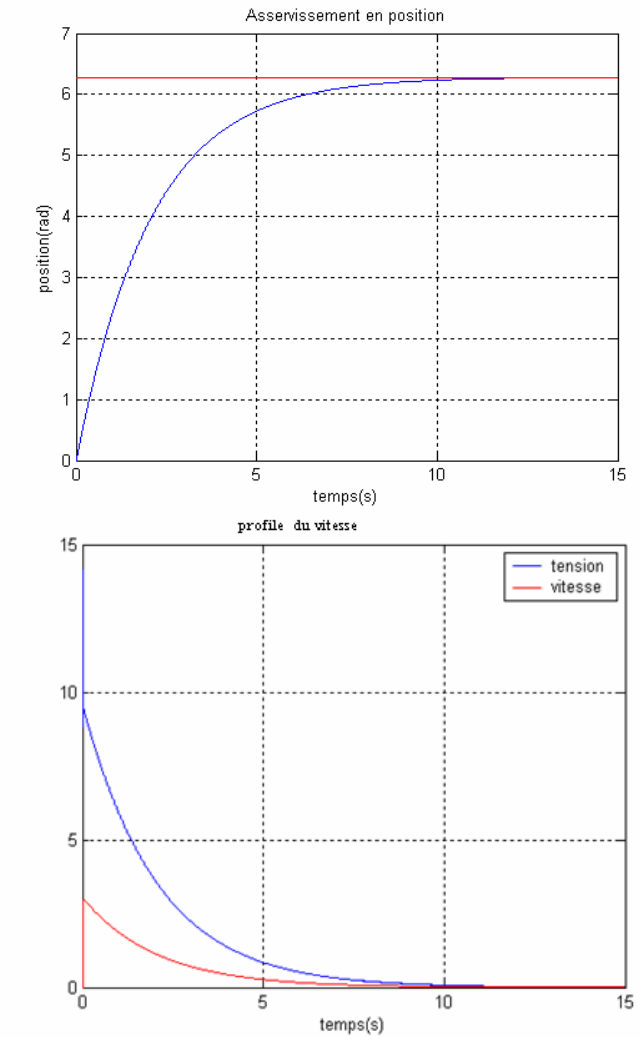


Figure V.5 : asservissement en position ;et profile du vitesse

Pour $\theta_d = 2\pi$, $K_{pp} = 2; K_{dp} = 0.5; K_{pc} = 3.$

V.4.COMMENTAIRE ET CONCLUSION :

Pour les trois consignes établies précédemment on constate ce qui suit :

- ✓ Le gain K_{dp} permet de diminuer les oscillations. Toute fois si on augment excessivement cette constante, le système devient de plus en plus lourd.
- ✓ Le coefficient K_{pp} permet de contrôler le courant à l'instant t=0. Par conséquence le couple de démarrage doit être important pour pouvoir tourner le moteur. Cependant le courant ne doit pas dépasser sa valeur maximale.
- ✓ Si on augmente la constante K_{pc}, le système tend rapidement à la consigne. Néanmoins, ceci peut provoquer la surtension, c à d la tension est supérieure à la tension d'alimentation maximale.

Conclusion générale

Du temps que la conception du bras manipulateur à trois degrés de liberté était fait à base du logiciel « solid concept » d'une manière rigoureuse, ceci nous à permit de réaliser un prototype modeste qui ouvre par la suit des chemins pour les futurs ingénieurs.
On a opté pour une structure simple, vu les moyens qu'on dispose. Entre autre, on a prévu un système de freinage électromagnétique constitué d'un relais et d'une bobine. Le relais travail en alternance entre la commande des actionneurs et le frein. Cependant, l'aluminium a un coefficient de frottement faible, ce qui résulte un mauvais freinage. Pour remédier à ce problème, il est conseillé d'utiliser un ferrodeau.

Une fois la réalisation a était faite, on a effectué une modélisation géométrique en se basant sur la notation de **denavit-Hartenberg** afin de pouvoir par la suite, positionner l'organe terminal.

L'interface d'axe UMAC présente une flexibilité et une disposition matérielle parfaite pour concevoir des applications de commande de mouvement. Sa conception montre la puissance qu'elle peut atteindre. Ainsi, par son biais de communication, o, arrive à exploiter essaiment ses ressources.

En ce qui concerne la commande, on peut appliqué une simple régulation qui permet la correction des erreurs. Un régulateur PD pour corriger la position .Néanmoins, le dérivateur provoque l'amplification du bruit. Pour cela, il faut prévoir un filtre passe-bas. Ainsi, un régulateur P pour le courant.

REFERENCES BIBLIOGRAPHIQUES

[1] **MARK W.SPONG AND M. VIDYASAGAR** « Robot Dynamics and Control », Quinn-Woodbine, USA, 1981.

[2] **WISAMA KHALIL ET ETIENNE DOMBRE** « Modélisation Identification et Commande des robots », 2e édition, Hermes Science Publications, Paris, 1999.

[3] **WEIPING LI AND WILLIAM A.WOLOVICH** « Robotics :Basic and Design »,

[4] **Paul R.C.P** « robot manipulators: mathematics, programming and control », MIT Press, Canbridge, 1981.

[5] **JOHN J. CRAIG** « Introduction to Robotics: Mechanics and Control »,2nd ed, Addison-Wesley Puplishning Company, Canada, 1989.

[6] **SCHILING, R.J** « Fundamentals of Rbotics : analysis an Control », Prentice Hall, 1990.

[7] www.pennmotion.com.

[8] **BRAIN ARMSTRONG, OUSSAMA KHATIB AND JOEL BURDICK** «The Explixit Dynamique Model and Inertial Parameters of the Puma 560 Arm », In Proc. 1986 IEEE Int. Conf. On Robotics and Automation, S an Francisco, CA.

[9] **BUCK LEY, J.J** « Fundamantal of Robotics : Analyse and Control » J.Wiley. 1989.

[10] **DELTA TAU DATA SYSTEMS, INC. UMAC Turbo (3U Turbo PMAC2) Hardware Reference Manual, Disponible en format pdf de** ftp://ftp.deltatau.com/UMAC/umactbhw.pdf, **Août 2001.**

[11] **DELTA TAU DATA SYSTEMS, INC. UMAC ACC-24E2A, the analog +/- 10 Volts axes board, Disponible en format pdf de** ftp://ftp.deltatau.com/UMAC/acc24e2a.pdf, **Octobre 2001.**

[12] **DELTA TAU DATA SYSTEMS, INC. UMAC I/O Accessory, HIGH POWER OPTO 24 INPUT/24 OUTPUT BOARD, Disponible en format pdf de** ftp://ftp.deltatau.com/UMAC/acc12e.pdf, **Décembre 2000.**

[13] **DELTA TAU DATA SYSTEMS, INC. AMP-2, four axes analog PWM amplifier, 48 VDC 3/5 A, Disponible en format pdf de** ftp://ftp.deltatau.com/UMAC/amp-2.pdf, **Avril 2001.**

[14] **DELTA TAU DATA SYSTEMS, INC. UMAC Power supply: 20 A @ 5 V, 1 A @ +/- 15 V , Disponible en format pdf de** ftp://ftp.deltatau.com/UMAC/acce2.pdf, **Février 2002.**

[15] **DELTA TAU DATA SYSTEMS, INC. UMAC Products Guide version 2.1**, **Disponible en format pdf de** ftp://ftp.deltatau.com/NewIdeasInMotion/umacpg.pdf, **Janvier 2001**.

[16] **DELTA TAU DATA SYSTEMS, INC. PMAC(1) Family Quick Reference Manual version 2.1 Disponible en format pdf de** ftp://ftp.deltatau.com/NewIdeasInMotion/pmacqref.pdf, **Juin 2000.**

[17] **DELTA TAU DATA SYSTEMS, INC. PMAC1 USER'S MANUAL, Disponible en format pdf de** ftp://ftp.deltatau.com/NewIdeasInMotion/pmac1usr.pdf, **Janvier 2000.**

[18] **MOTOROLA, SEMICONDUCTOR PRODUCT INFORMATION. DSP56303 24-Bit General-Purpose Digital Signal Processor Data Sheet, Disponible en format pdf de** http://e-www.motorola.com/brdata/PDFDB/docs/DSP56303DS.pdf, **Janvier 2002.**

[19] **MOTOROLA, SEMICONDUCTOR PRODUCT INFORMATION. DSP56303 24-Bit Digital Signal Processor User's Manual, Disponible en format pdf de** http://e-www.motorola.com/brdata/PDFDB/docs/DSP56303UM.pdf, **Janvier 2001.**

[20] **MOTOROLA, SEMICONDUCTOR PRODUCT INFORMATION. Advance Information. 4-BIT DIGITAL SIGNAL PROCESSOR, Disponible en format pdf de** http://e-www.motorola.com/brdata/PDFDB/docs/DSP56303P.pdf, **Juillet 2001.**

[21] **T.Redheendran. Booting DSP563xx Devices Through the Serial Communication Interface (SCI), Disponible en format pdf de** http://e-www.motorola.com/brdata/PDFDB/docs/AN1781.pdf**, Novembre 98.**

[22] **Barbara Johnson.Software Differences Between the DSP56002 and the DSP56303, Disponible en format pdf de** http://e-www.motorola.com/brdata/PDFDB/docs/AN1829.pdf, **Octobre 2001.**

[23] **Barbara Johnson.DSP56303 Hardware Differences Between the DSP56002 and the DSP56303, Disponible en format pdf de** http://e-www.motorola.com/brdata/PDFDB/docs/AN1830.pdf, **Octobre 2001.**

[24] **DELTA TAU DATA SYSTEMS, INC. PMAC2 USER'S MANUAL, Disponible en format pdf de** ftp://ftp.deltatau.com/NewIdeasInMotion/pmac2usr.pdf, **Janvier2000.**

[25] **DELTA TAU DATA SYSTEMS, INC. Turbo PMAC software manual (version 1.936) , Disponible en format pdf de** ftp://ftp.deltatau.com/PMACManual/t1t2sw.exe, **Septembre 2001.**

[26] **National Semiconductor. LMD 18200 H-Bridge Data Sheet, Disponible en format pdf de** http://www.nsc.com/ LMD18200.pdf, **Décembre 1999.**

[27] **DELTA TAU DATA SYSTEMS, INC. UMAC UBUS Preliminary Specification, Disponible en format pdf de** ftp://ftp.deltatau.com/UMAC/ubus.pdf, **Avril 2000**.

ملخص:

العمل المقدم من خلال هذه المذكرة يتمثل في دراسة ثم انجاز ذراع آلية والتي يتم التحكم فيها بواسطة "UMAC".

ومن أجل هذا بطرح نموذج أولي ، ثم قمنا بدراسة النموذج عن طريق برنامج يسمى "Solid Concept" الذي مكننا من وضع الأبعاد والقياسات بدقة لهذه الذراع الآلية. غير أن اختيار هذه البنية كان تحت تاثير بعض العوامل كالفعالية و الإمكانيات المتوفرة في دائرة الهندسة الميكانيكية. ثم قمنا باستخراج مختلف النماذج .

نظرا لكون المحركات ذات التيار المستمر سهلة التحكم ، قمنا باختيار محركات PITMAN GM9236 و التي تتميز بمعامل تحول كبير مما يعني عزم مزدوجة كبير .

في الجزء الثاني ، تطرقنا الى التحكم في المحركات بواسطة "UMAC " والتي تتميز بالمرونة و إمكانية التحكم إلى غاية 8 محاور.

كلمات مفتاحية: **ذراع آلية، محركات، آلي.**

Résumé : Le travail présenté dans ce mémoire est une conception puis une réalisation d'un bras manipulateur commandé par l'interface d'axe UMAC « Universal Motion Automation Controller ». Pour cela, Nous avons proposé un prototype et puis faire la conception de ce bras manipulateur à base du logiciel « Solid Concept » qui nous a permis de dimensionner rigoureusement ce robot. Entre autre, le choix de cette structure est fait sous contraintes de la faisabilité et les moyens existant au sein du département Génie mécanique. Puis les différentes modélisations ont été élaborées. Du moment que la commande des moteurs à courant continue est simple ainsi la disponibilité, on a opté pour les moteurs PITMAN GM9236 qui ont un rapport de réduction important donc un fort couple. En seconde partie, on s'y intéressé à la commande des actionneurs par l'interface d'axe UMAC qui présente une flexibilité et un pouvoir de contrôler jusqu'au 8 axes.

Mots clés : bras manipulateur, actionneurs**,** robot.

Abstract : The work presented in this memory is a design then a realization of robot arm controlled by the interface of axis UMAC "Universal Motion Automation Controller". For that, We proposed a prototype and then to make the design of this arm, we've used the software "Solid Concept" which enabled us to dimension this robot rigorously. Amongst other things, the choice of this structure is made under stress of feasibility and the means existing within the mechanical department Engineering. Then various modelings were elaborate. Since that the control of a direct current motors is simple thus availability, we have chose the motors PITMAN GM9236 which thus have a significant ratio, so a strong couple. In the second part, we've interested in the control of the actuators by the interface of axis UMAC which presents a flexibility and a capacity to control up to 8 axes.

Key words :arm manipulator, actuators, robot.