

INTRODUCTION

Dans les entreprises manufacturières, des tâches pénibles, répétitives réalisées par des opérateurs humains peuvent être avantageusement confiées à des systèmes mécaniques articulés (SMA) dont la dextérité est, sans égaler à celle de l'homme, suffisamment proche de celui-ci pour exécuter des mouvements assez complexes. L'emploi de ces dispositifs s'avère d'ailleurs nécessaire pour des tâches d'intervention inaccessibles ou fort risqués pour l'homme et ils sont de plus en plus présents aussi bien en milieu industriel qu'en éducation.

Dans ce cadre, l'Institut Supérieur des Etudes Technologiques de Nabeul nous a proposé un projet de fin d'études qui consiste à réaliser une interface de commande d'un robot à quatre articulations.

Dans la première partie de ce rapport nous procéderons à une description du robot et de son environnement (le robot T45, Le module de commande T45, le boîtier d'alimentation) et de la communication entre le robot et l'ordinateur.

La spécification et la conception de l'application ainsi que la modélisation feront l'objet de la deuxième partie.

La troisième et dernière partie sera consacrée à la réalisation, nous présenterons l'environnement de travail et l'implantation de l'application.

Un manuel d'utilisation sera livré le jour de la soutenance.

CAHIER DES CHARGES

Commande d'un robot à quatre articulations.

Le sujet consiste à concevoir un programme qui permet la commande d'un robot à quatre articulations. Ce programme doit permettre la simulation graphique des mouvements du robot, ensuite il doit passer à l'étape d'exécution réelle des mouvements déjà simulés.

Le matériel existant :

- Un robot T45
- Une interface de puissance qui communique avec l'ordinateur à travers l'interface série suivant un protocole spécifique. Cette interface de puissance transmet les ordres vers le robot suivant les commandes émises par le port série de l'ordinateur.

Un programme en *GW-Basic* est aussi disponible, il réalise tous les mouvements possibles des articulations du robot.

PREMIERE PARTIE

Etude de l'existant

I. Description générale du robot et de ses modules :

Le robot T45 est associé à d'autres modules qui forment ensemble un petit équipement robotique constitué par :

1. un robot de référence Tergane45,
2. un module de commande référence 45000,
3. un boîtier d'alimentation,
4. un micro-ordinateur.

1. Le robot T45 :

Le robot est capable d'assurer cinq mouvements :

Quatre mouvements asservis (base, épaule, coude et poignet), la commande de chaque mouvement est réalisée par un bloc moto-réducteur alimenté par une tension de +12V. La tension recopie (chaîne de retour) est formée par un potentiomètre rotatif linéaire fixé sur le bloc moto-réducteur. Ce potentiomètre est alimenté par une tension de + 10V.

Le mouvement en boucle ouvert, est celui de la pince. La commande est identique aux mouvements précédents, sauf qu'il n'a pas de potentiomètre de recopie.

Les principales caractéristiques de ce robot sont :

- le moteur de chaque articulation est à courant continu de puissance 2,5 Watt, alimenté par 12V,
- les potentiomètres de chaque articulation sont à couche plastique, leur tension de référence est + 10V,
- le déplacement maximal de la base est de 293°,
- le déplacement maximal de l'épaule est de 107°,
- le déplacement maximal du coude est de 284°,
- le déplacement maximal du poignet est de 360°,
- pour la pince :
 - ♦ arrêt automatique par microswitch lors de l'ouverture,
 - ♦ vitesse de fermeture réglable.

2. Module de commande 45000 :

Le module de commande permet par l'intermédiaire de n'importe quel ordinateur possédant une interface série RS232 ou un kit de développement de commander et de contrôler :

- quatre mouvements asservis,
- un mouvement en boucle ouverte,
- huit entrées,
- huit sorties.

a) Mouvements asservis :

Les caractéristiques primaires du robot sont modifiées. En effet des valeurs limites sont fixées électroniquement grâce au module 45000 qui exprime par exemple les limites des rotations angulaires :

- le déplacement de la base devient 261° ,
- le déplacement de l'épaule devient 85° ,
- le déplacement du coude devient 249° ,
- le déplacement du poignet devient 180° .

b) Mouvement en boucle ouverte :

La tension de commande émise par la carte permet de contrôler la vitesse du mouvement non asservi (pince)

c) Entrée/Sorties :

Huit entrées et huit sorties tout ou rien (logiques) sont disponibles et permettent la communication avec des éléments extérieurs au robot.

d) Les principales caractéristiques de la carte 45000 :

- Permet la commande de quatre mouvements asservis avec une tension de commande +12V et une tension de recopie de +10V.
- Permet la commande d'un mouvement non asservi avec une tension de commande +12V.
- Permet la réception de huit entrées avec 0V pour le niveau actif et 5V pour le niveau passif.
- Permet la commande de huit sorties logiques.
- La carte donne une alimentation de +15V et +5V.

3. Boîtier d'alimentation :

Il permet à partir du secteur 110V/50HZ ou 220V/50HZ d'alimenter le module 45000 avec :

- une tension +15V et un courant 1.5A,
- une tension -15V et un courant 1.5A,
- une tension +5V et un courant 3A,
- protection secteur par fusible.

II. Communication robot/ordinateur :

1. Paramètres de transmission :

Sept straps permettent de paramétrer la transmission :

- 1 à 4 : vitesse de 50 à 9600bauds,
- 5 : nombre de bits de stop (1 ou 2),
- 6 : parité paire ou impaire,
- 7 : parité (avec ou sans).

2. Principe de fonctionnement :

Cycliquement, le microprocesseur du module de commande vient lire les paramètres de transmission, si un strap est modifié un sous programme permet de mettre à jour les paramètres du Timer, si aucune modification n'est détectée le module n'est pas actif.

Le module de commande passe par quatre états :

Etat 1 : attente de cinq caractères formant une commande.

Etat 2 : renvoi des cinq caractères reçus + LF + RC.

Etat 3 : analyse synoptique et sémantique de la commande et renvoi d'un message.

Etat 4 : exécution de la commande si elle est correcte.

3. Messages émis par la carte 45000 :

A la mise sous tension, après une remise à zéro ou à chaque changement de paramètres, la carte envoie le message « PRET A LA SUITE D'UNE MISE SOUS TENSION OU D'UN CHANGEMENT DE PARAMETRE »

A la suite d'une commande issue du calculateur maître, le module de commande envoie :

- les mêmes cinq caractères de commande reçus,
- le message « MESSAGE CORRECT » ou « MESSAGE INCORRECT ».

4. Commande du robot :

Les commandes du module 45000 sont toujours codées sur cinq caractères ASCII selon les règles suivantes :

- Premier caractère : élément à commander
 - B = Base.
 - E = Epaule.
 - C = Coude.
 - D = Différentiel 1 (non actif dans la version actuelle du robot)
 - F = Différentiel 2 (actuelle rotation de poignet)
 - P = Pince.
 - S = Sorties.
 - I = Entrées.
 - X = Interdiction du mode écho
- Deuxième caractère : signe indiquant le sens de déplacement
 - + : signe positif
 - : signe négatif
- Les trois derniers caractères : Valeur ASCII du déplacement absolu :
 - Valeur minimale : 0
 - Valeur maximale : 511

DEUXIEME PARTIE

Etude théorique

I. Introduction :

Dans cette partie nous allons procéder à une étude théorique de notre application et de son environnement. Tout d'abord nous allons modéliser le robot T45, puis spécifier et concevoir l'application.

II. Modélisation du robot :

1. Introduction :

Lors de l'exécution d'une tâche, la principale fonction du SMA (Système Mécanique Articulé) d'un robot industriel est de positionner et orienter correctement dans l'espace l'organe terminal (O.T), d'ailleurs la modélisation géométrique du SMA consiste à établir la relation entre les coordonnées articulaires et les coordonnées cartésiennes. Dans notre cas, le robot T45 est composé de trois rotations dont une perpendiculaire et deux parallèles ($R \perp R/R$).

2. Notions de modélisation :

La modélisation est décrite dans un repère affine orthonormé (R.A.O.N) fixe qui est le référentiel lié à la base du SMA : R_0 .

De même on associe un R.A.O.N à l'organe terminal fixé sur le dernier solide du SMA. Il sera désigné par R_{i+1} .

Dans ce projet on utilise la modélisation de type dynamique qui prennent en compte en plus les caractéristiques géométriques du SMA, les caractéristiques inerties de ces solides.

Le système est composé de $(n + 1)$ corps $\Rightarrow n$ articulations.

Le corps S_0 désigne la base du robot et le solide S_n porte l'OT.

a) Méthode Denavit-Hartenberg :

- On commence par représenter le SMA dans une configuration particulière (la plus simple possible)

- On numérote les axes A_i ($i = 1 \dots n$) autour desquels s'effectuent les mouvements relatifs des solides.
- L'origine O_{i+1} du repère R_{i+1} est situé sur l'axe A_{i+1} et il est confondu avec le pied de la \perp commune aux axes A_i et A_{i+1} . Si ses axes sont confondus ou parallèles on choisit arbitrairement O_{i+1} sur A_i .
- Le vecteur \vec{X}_{i+1} de A_{i+1} a pour support la \perp commune aux axes A_i et A_{i+1} et il est orienté de A_i vers A_{i+1} . Si les deux axes sont confondus ou parallèles le support de \vec{X}_{i+1} est \perp l'axe A_{i+1} ou bien le point O_{i+1} .
- \vec{Z}_{i+1} de R_{i+1} a pour support l'axe A_{i+1} et on choisit arbitrairement son orientation.
- $\vec{Y}_{i+1} = \vec{Z}_{i+1} * \vec{X}_{i+1}$.

b) Modèle Géométrique Direct (MGD) :

○ Définition :

Le MGD est la fonction F qui permet d'exprimer la situation de l'organe terminal du robot en fonction de la configuration du robot lui-même.

On appelle MGD d'un robot industriel la relation suivante :

$X = F(q) = F(q_1, q_2, \dots, q_n)$ tel que q est le vecteur de configuration du SMA. le système contient n articulation de type R ou P.

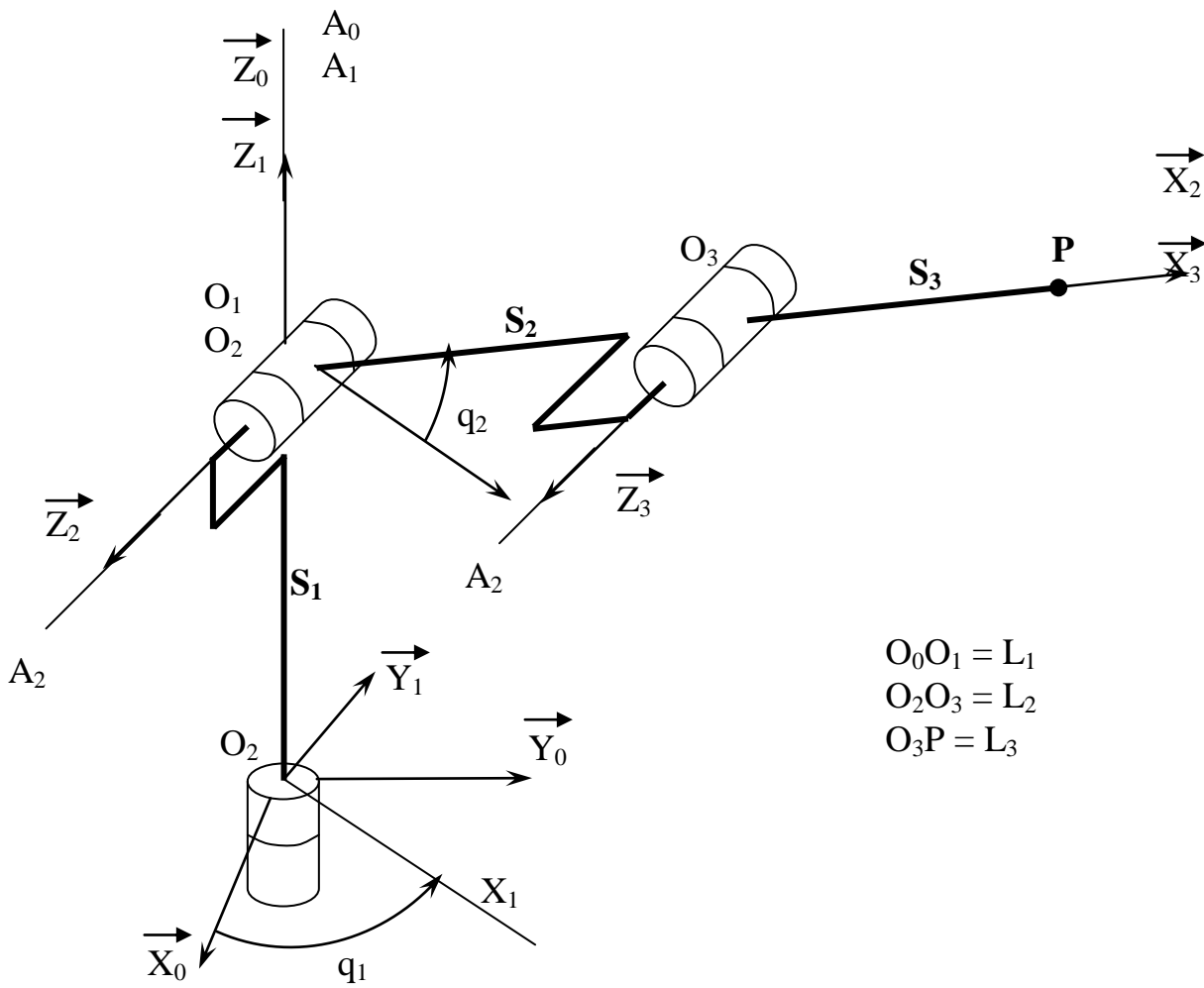
○ Méthode de calcul du MGD :

Elle se décompose selon les étapes suivantes :

- Description du SMA.
- Mise en place itérative des repères affines associés aux solides constituant le SMA.
- Etablissement des paramètres de tableau de D-H.
- Calcul des matrices du passage homogène (M.P.H) élémentaire T_0, T_1, \dots, T_n .

- Calcul de la M.P.H globale : $T_{0,n} = (T_{0,1} \cdot T_{1,2}) \cdot T_{2,3} \dots T_{n-1,n}$.
- Nous aurons comme résultat les identités suivantes :
- Position de O_n par rapport à R_0
- L'orientation de R_n

3. Modèle géométrique direct :



Paramétrage de Denavit-Hartenberg

a) Paramètres du tableau de D-H :

Solide S_i	1	2	3
Paramètres			
Type d'articulation	R	R	R
δ_i	0	0	0
a_i	0	0	L_2
α_i	0	$+\pi/2$	0
r_i	L_1	0	0
θ_i	θ_1	θ_2	θ_3
q_i	q_1	q_2	0

b) Matrices de passage :

Les abréviations suivantes seront utilisées pour tout calcul :

$$\begin{aligned}
 C_i &\rightarrow \cos \theta_i \\
 S_i &\rightarrow \sin \theta_i \\
 C_{\alpha_i} &\rightarrow \cos \alpha_i \\
 S_{\alpha_i} &\rightarrow \sin \alpha_i
 \end{aligned}$$

$$O_{i+1(R_i)} = \begin{pmatrix} a_i C_i \\ a_i S_i \\ r_i \end{pmatrix}$$

- Matrices de passage homogènes :

$$M_{i,i+1} = \begin{pmatrix} C_i & -C_{\alpha_i} S_i & S_{\alpha_i} S_i \\ S_i & C_{\alpha_i} C_i & -S_{\alpha_i} C_i \\ 0 & S_{\alpha_i} & C_{\alpha_i} \end{pmatrix}$$

Lorsque deux bases ont une orientation quelconque l'une par rapport les termes $M_{i,i+1}$ sont assez complexe, Ce calcul est effectuer de la manière précédente. Mais si la base R_i par une simple rotation d'angle θ autour de l'axe X ou Y ou Z, la matrice de passage a une forme simple.

Pour une rotation autour de l'axe Z et compte tenu de la définition de la matrice de passage, on a :

$$R(Z,\theta) = \begin{bmatrix} C\theta & -S\theta & 0 \\ S\theta & C\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_{0,1} = \begin{bmatrix} C1 & -S1 & 0 \\ S1 & C1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_{1,2} = \begin{bmatrix} C1 & -S1 & 0 \\ S1 & C1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_{2,3} = \begin{bmatrix} C1 & -S1 & 0 \\ S1 & C1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$O_1 = (0, 0, 0)_{R_0}$ (coordonnées de O_1 dans le repère R_0)

$$T_{i,i+1} = \left(\begin{array}{ccc|c} & & & O_1 \\ \hline & M_{0,1} & & \\ \hline 0 & 0 & 0 & 1 \end{array} \right)$$

$$T_{0,1} = \begin{bmatrix} C1 & -S1 & 0 & 0 \\ S1 & C1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$(O_3)_{R1} = (O_2)_{R1} + M_{1,2} (O_3)_{R2} = (L2C2,0,L1-L2S2)_{R1}$$

Au lieu de calculer $T_{1,2} * T_{2,3}$ on peut profiter du fait que les deux rotations successives d'axes Z_2 et Z_3 peuvent être considérées comme une seule d'angle $\theta_4 = \theta_2 + \theta_3$ pour trouver $T_{1,3}$

$$T_{1,3} = T_{1,2} * T_{2,3} = \begin{pmatrix} C4 & -S4 & 0 & L2C2 \\ S4 & C4 & 0 & 0 \\ 0 & 0 & 1 & L1 - L2C2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Matrice de passage globale :

$$T_{0,3} = T_{0,1} * T_{1,2} * T_{2,3}$$

$$T_{0,1} = \begin{pmatrix} C1 & -S1 & 0 & 0 \\ S1 & C1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_{1,3} = \begin{pmatrix} C4 & -S4 & 0 & L2C2 \\ S4 & C4 & 0 & 0 \\ 0 & 0 & 1 & L1 - L2S2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$D'où T_{0,3} = T_{0,1} * T_{1,3}$$

On pose $\theta_5 = \theta_1 + \theta_4$ pour simplifier l'écriture de l'expression.

$$T_{0,3} = \begin{pmatrix} C5 & -S5 & 0 & L2C1C2 \\ S5 & C5 & 0 & 0 \\ 0 & 0 & 1 & L1 - L2S2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Donc on peut obtenir les coordonnées de O_3 dans le référentiel R_0 est le suivant :

$$O_{3(R0)} = \begin{pmatrix} L2C1C2 \\ 0 \\ L1 - L2S2 \end{pmatrix}$$

Le cosinus directeur de X_3 dans le référentiel R_0 est le suivant :

$$\vec{X}_{3(R0)} = \begin{bmatrix} C5 \\ S5 \\ 0 \end{bmatrix}$$

Le cosinus directeur de Y_3 dans le référentiel R_0 est le suivant :

$$\vec{Y}_{3(R0)} = \begin{bmatrix} -S5 \\ -C5 \\ 0 \end{bmatrix}$$

Le cosinus directeur de Z_3 dans le référentiel R_0 est le suivant :

$$\vec{Z}_{3(R0)} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Les coordonnées de P dans R_3 sont :

$$P_{(R3)} = \begin{bmatrix} L3 \\ 0 \\ 0 \end{bmatrix}$$

D'où les coordonnées de P dans le référentiel R_0 sont :

$$P_{(R0)} = \begin{bmatrix} L2C1C2 + L3C5 \\ L3C5 \\ L1 - L2S2 \end{bmatrix}$$

4. Le modèle géométrique inverse (MGI) :

Le MGD d'un robot permet de calculer les coordonnées opérationnelles en fonctions des coordonnées articulaires : q_i . $X = f(q)$ (MGD)

$$q = f^{-1}(X) \quad X = (X_1, \dots, X_m), q = (q_1, \dots, q_n)$$

m : le nombre de coordonnées opérationnelles

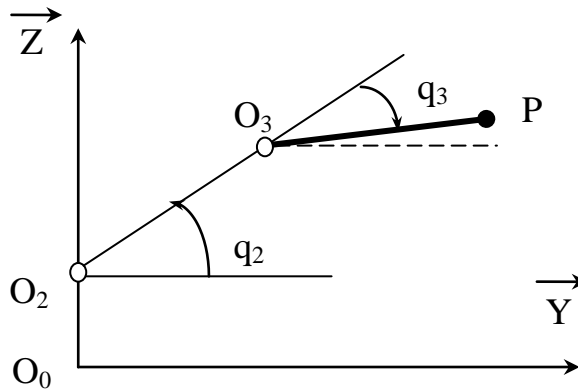
n : le nombre de coordonnées articulaires

Il s'agit de résoudre le problème par rapport aux q_i

$q = f^{-1}(X)$ est un système de m équations à n inconnus.

Dans notre cas nous allons travailler en deux parties, l'une Fig2 pour la base et l'autre Fig1 pour l'épaule et le coude.

Le robot T45 possède (R⊥R//R) donc $m = n = 3$.



$$\begin{aligned} O_0O_1 &= L_1 \\ O_2O_3 &= L_2 \\ O_3P &= L_3 \end{aligned}$$

Fig1

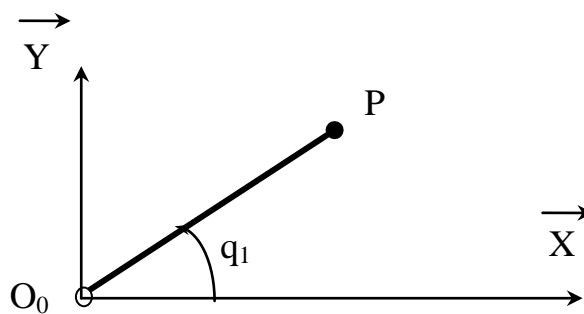


Fig2

Manipulateur 3R(spatial)

En se basant sur la Fig1 on obtient :

$$q = (\widehat{Y, O_3 P}) = (\widehat{Y, O_2 O_3}) + (O_2 \widehat{O_3, O_2 O_3}) = q_2 + q_3 = X_4$$

La projection de la relation vectorielle sur Y et Z donne les deux équations suivantes :

$$\left. \begin{aligned} X_1 &= L_2 * Cq_2 + L_3 * C(q_2 + q_3) \quad (a) \\ X_2 &= L_2 * Sq_2 + L_3 * S(q_2 + q_3) \quad (b) \\ \text{En plus on a : } X_4 &= q_2 + q_3 \quad (c) \end{aligned} \right\} \quad (1)$$

En reportant l'expression (c) dans (a) et (b) on se ramène à un système de deux équations à deux inconnus :

$$L_2 * Cq_2 = X_1 - L_3 * C X_4 = Y \quad (a')$$

$$L_2 * Sq_2 = X_2 - L_3 * S X_4 = Z \quad (b')$$

Y et Z sont les coordonnées de O₂.

$(a')^2 + (b')^2 \Rightarrow$ on élimine l'inconnu q₂

$$L_2^2 + L_3^2 + 2L_2 * L_3 * Cq_3 = Y^2 + Z^2$$

On pose : $r^2 = Y^2 + Z^2 = (O_2 O_3)^2$

$$Cq_3 = \frac{r^2 - (L_2^2 + L_3^2)}{2 * L_2 * L_3} ; \text{avec } q_3 = \pm \alpha$$

$$\Rightarrow q_3 = \text{ArcCos} \frac{r^2 - (L_2^2 + L_3^2)}{2 * L_2 * L_3} ; 0 < \alpha < \pi$$

Reprenons le système (1), en développant C(q₂ + q₃) et S(q₂ + q₃) :

$$\left. \begin{aligned} (L_2 + L_3 * Cq_3) * cq_2 - L_3 * Sq_3 * Sq_2 &= Y \\ (L_2 + L_3 * Cq_3) * Sq_2 + L_3 * Sq_3 * Cq_2 &= Z \end{aligned} \right\} (2) \quad \left\{ \begin{aligned} a_1 * X_1 + b_1 * X_2 &= c_1 \\ a_2 * X_1 + b_2 * X_2 &= c_2 \end{aligned} \right.$$

On résout le système (2) linéaire par rapport aux deux inconnues Cq₂ et Sq₂. Le déterminant vaut r², Cq₂ et Sq₂ sont donnés par :

$$Cq_2 = \frac{1}{r^2} \left[Y * (L_2 + L_3 * Cq_3) + Z * L_3 * Sq_3 \right]$$

$$Sq_2 = \frac{1}{r^2} \left[Z * (L_2 + L_3 * Cq_3) - Y * L_3 * Sq_3 \right]$$

$$X_1 = \frac{\begin{vmatrix} c_1 b_1 \\ c_2 b_2 \end{vmatrix}}{r^2} ; X_2 = \frac{\begin{vmatrix} a_1 c_1 \\ a_2 c_2 \end{vmatrix}}{r^2}$$

D'où :

$$q_2 = \text{ArcTg} \frac{Z * (L_2 + L_3 * Cq_3) - Y * L_3 S q_3}{Y * (L_2 + L_3 * Cq_3) + Z * L_3 S q_3}$$

En se basant sur la Fig2 on obtient :

$$q_1 = \widehat{(X, O_0P)}$$

La projection de la relation vectorielle sur X et Y donne les deux équations suivantes :

$$Y' = O_0P * Cq_1 = L_3 * C_5 * Cq_1 \quad (a'')$$

$$X' = O_0P * Sq_1 = (L_2 * C_1 * C_2 + L_3 * C_5) * Sq_1 \quad (b'')$$

La relation $\frac{(a'')}{(b'')}$ nous donne :

$$\text{Tg}(q_1) = \frac{L_3 * C_5}{L_2 * C_1 * C_2 + L_3 * C_5}$$

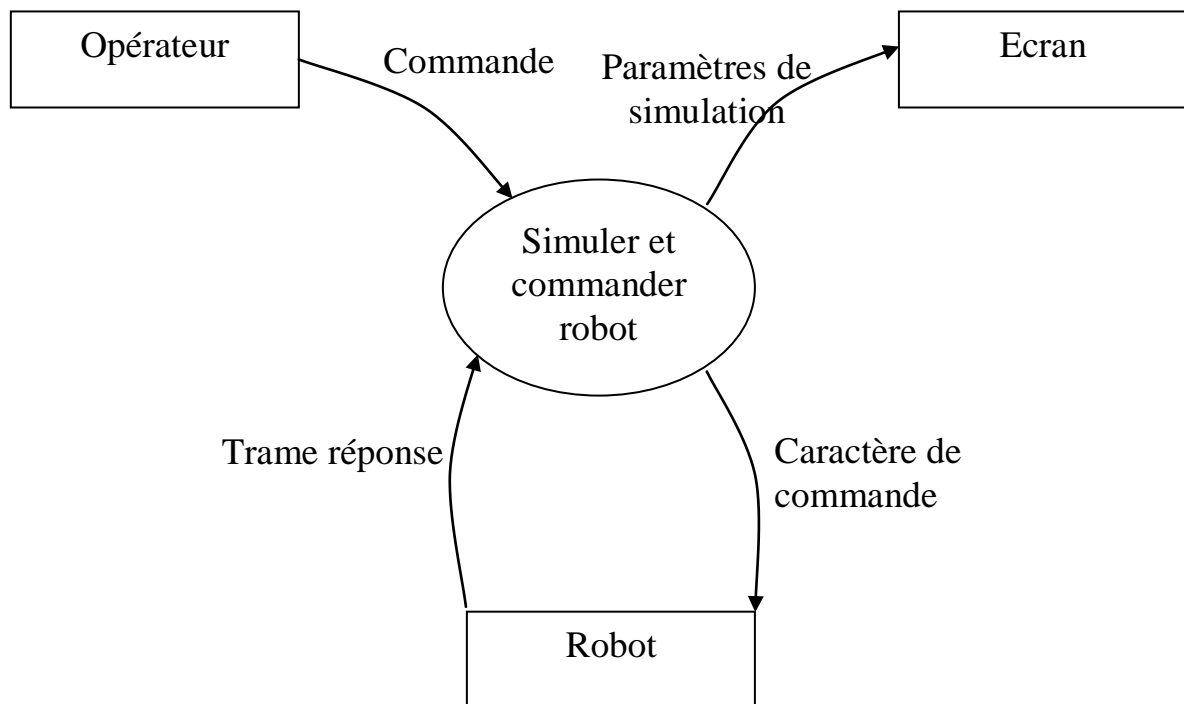
D'où :

$$q_1 = \text{ArcTg} \frac{L_3 * C_5}{L_2 * C_1 * C_2 + L_3 * C_5}$$

III. Spécification

1. Introduction :

Dans cette partie, nous allons spécifier notre application avec la méthode SART selon l'approche *Ward & Miller(WM)*. Il est à noter que nous n'allons réaliser que la spécification selon la méthode SART, pour la conception, nous utiliserons la méthode orientée objet pour faciliter la réalisation.

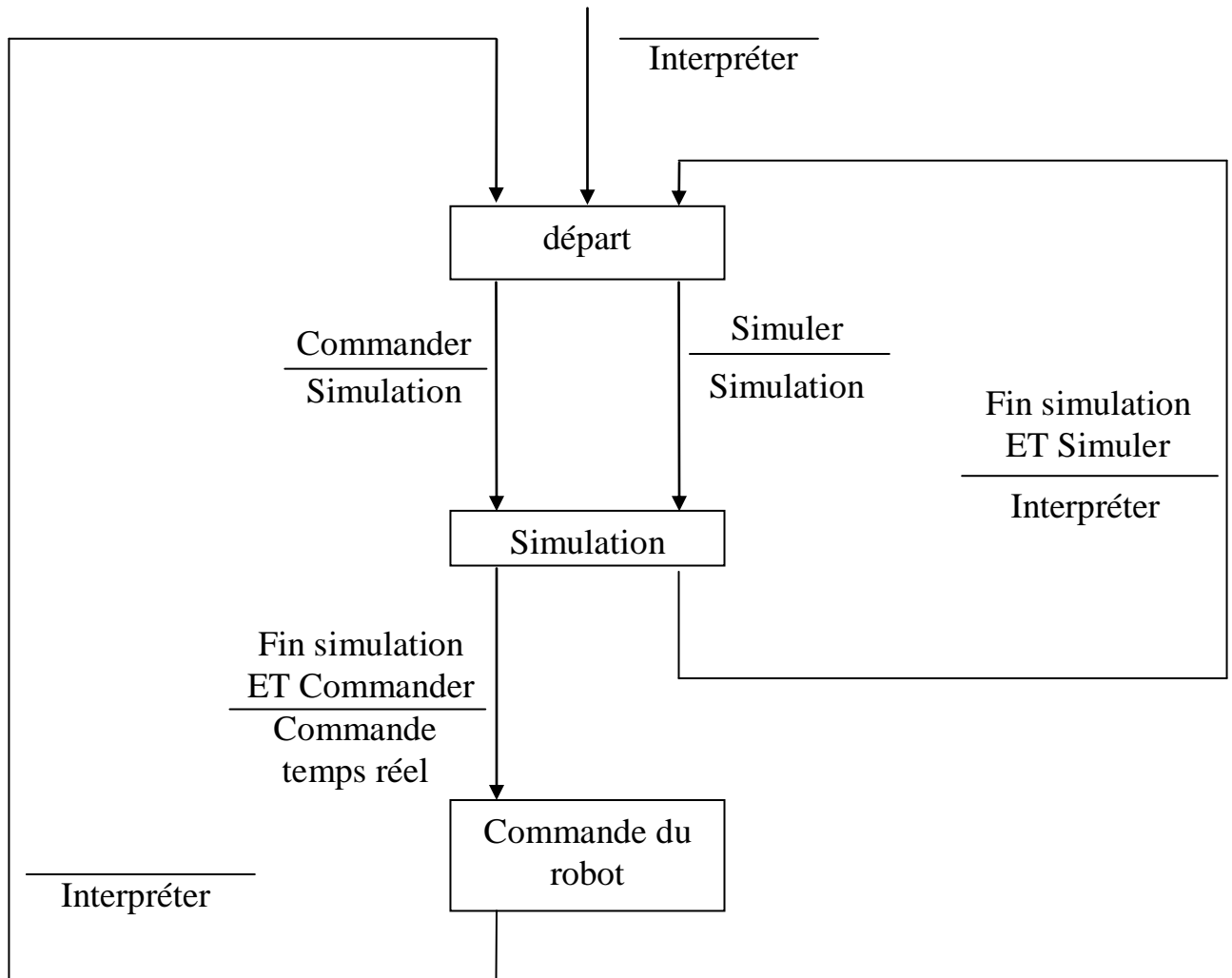
2. spécification, selon Ward & Miller :a) diagramme de contexte :**DIAGRAMME DE CONTEXTE : Simuler et commander robot**

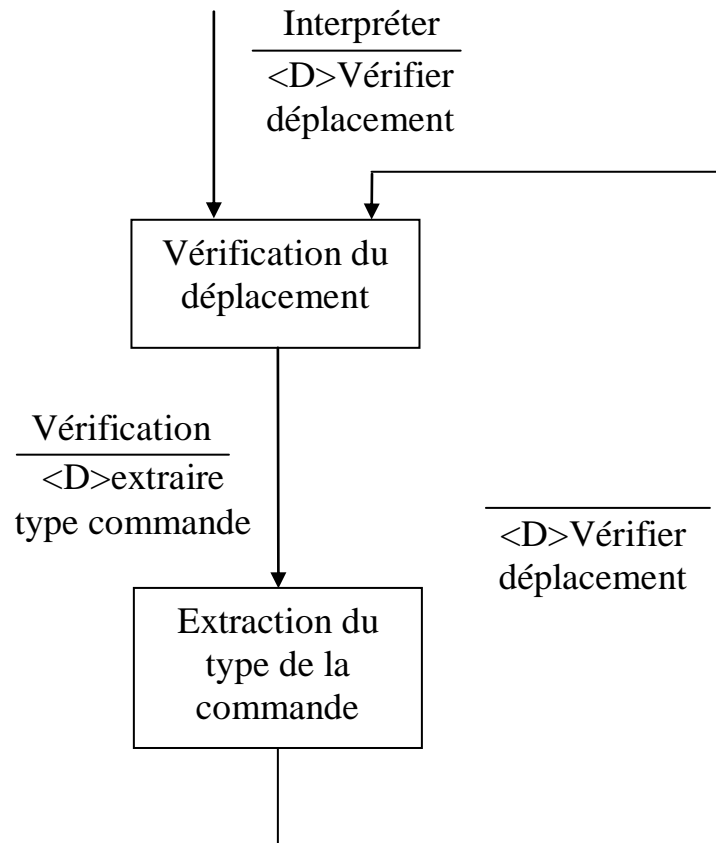
b) diagramme préliminaire :

c) Diagramme interpréter commande :

d) Diagramme commander :

e) Diagramme simuler :

f) Diagramme d'état transition: commande et simulation du robot**Diagramme d'état transition : commande et simulation du robot**

g) Diagramme d'état transition : interpréter commande**Diagramme d'état transition : interpréter commande**

h) Diagramme d'état transition: commande robot

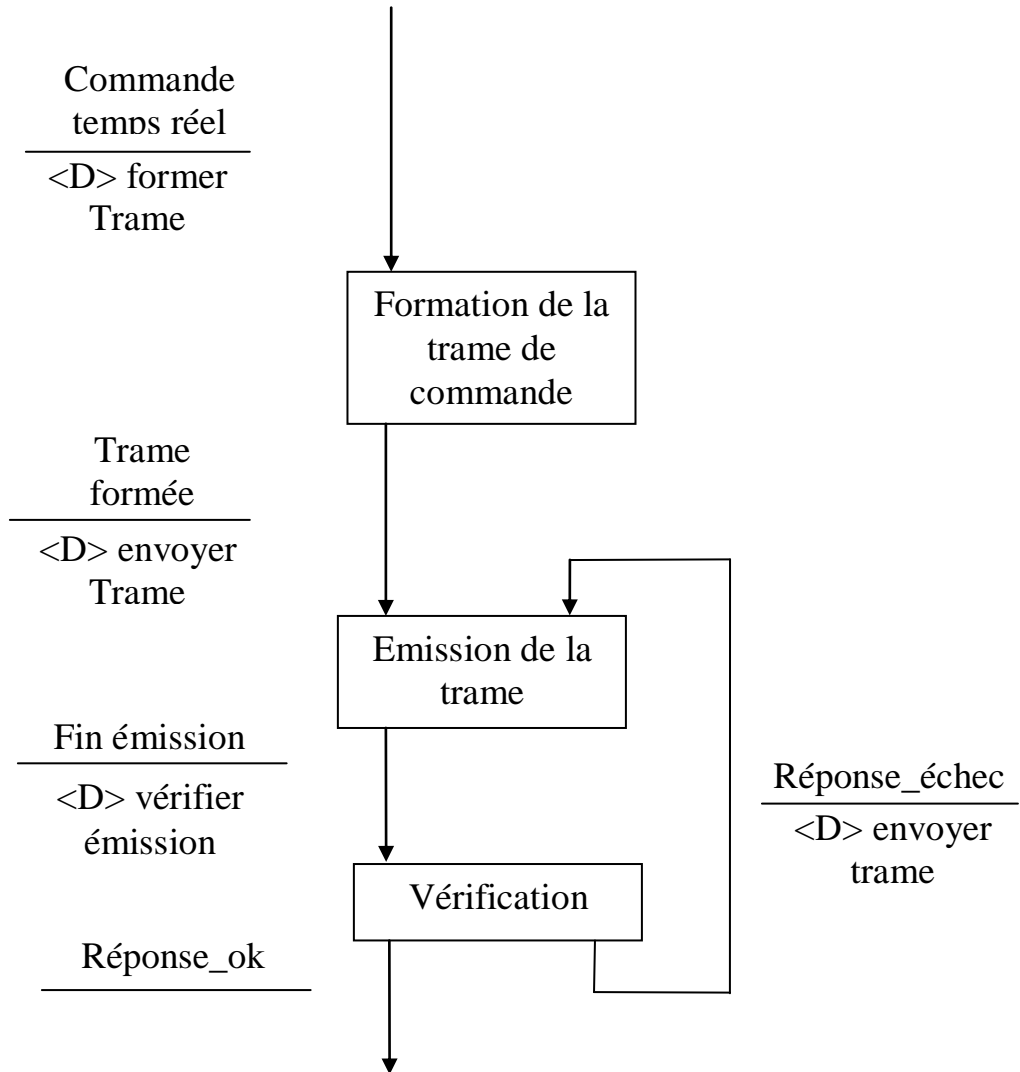
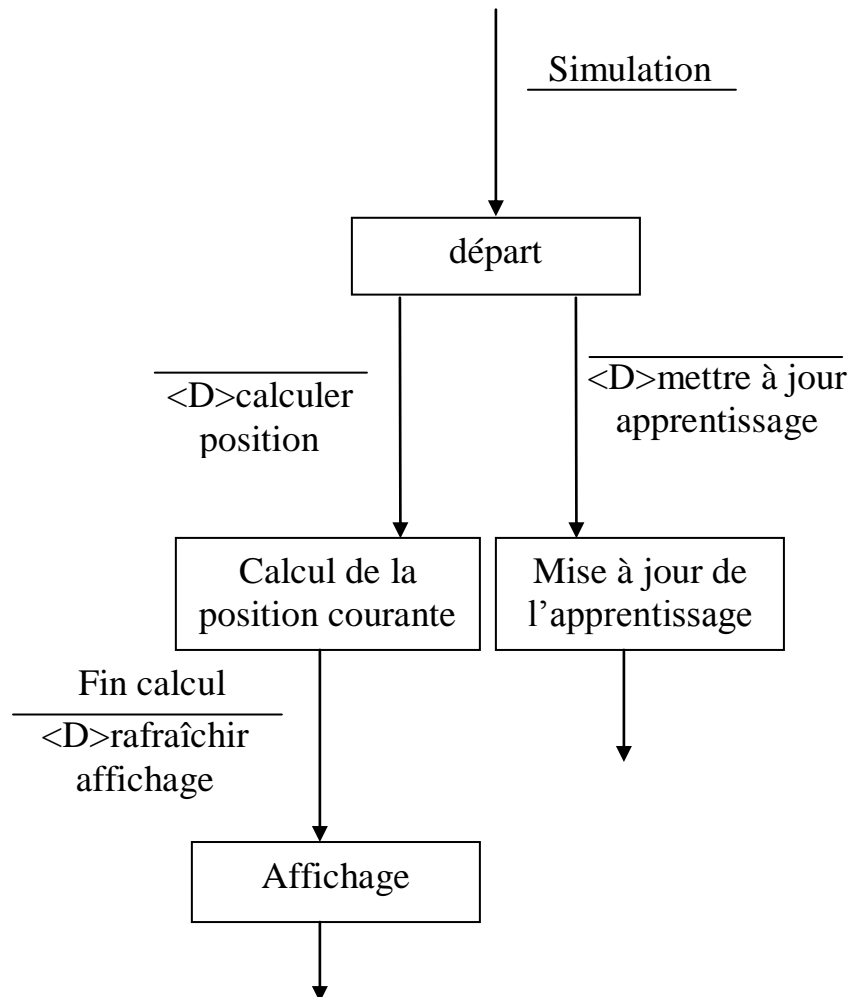


Diagramme d'état transition : commande robot

i) Diagramme état transition : simuler**Diagramme d'état transition : simuler**

j) dictionnaire de données :

Caractère de commande = [B | C | E | F | I | P | S | X | + | - | 0...9]

Commande = [Simulation / Commander] + Déplacement

Simuler = * bit de précision pour la simulation sur écran *

Commander = * bit de précision pour la commande du robot *

Interpréter = * demande d'interprétation de la commande reçue *

Déplacement * valeur numérique du déplacement *

Organe = [B | C | E | F | I | P | S]

Trame de commande = 5{ Caractère de commande }5

Commande temps réel = * demande de commande du robot *

Simulation = * demande de simulation sur écran *

Fin simulation = * fin de la simulation *

Paramètres simulation = * paramètres permettant de rafraîchir l'écran *

Trame réponse = [caractère de commande + [MESSAGE CORRECT / MESSAGE INCORRECT]]

Réponse_ok = * événement produit si l'émission est correcte *

Réponse_echec = * événement produit si erreur d'émission *

Fin émission = * événement indiquant la fin de l'émission *

Etat robot = * zone de stockage pour les différents états du robot *

Stockage trame = * zone de stockage pour communiquer entre le processus former trame et les processus envoyer trame et vérifier émission *

Apprentissage = * zone de stockage pour la liste d'apprentissage *

Anciennes position = * zone de stockage pour l'ancien état du robot *

IV. Conception :

1. Idée sur l'orienté objet

a) Introduction :

La phase de conception est l'une des phases les plus importantes dans le cycle de vie d'un logiciel, car c'est d'elle que dépendent toutes les étapes qui suivent.

Une bonne conception → un bon produit.

Dans ce chapitre nous présentons l'OMT, connue pour sa simplicité et sa maintenabilité. En plus, nous présentons les différents modèles de notre conception.

b) Méthodologie de conception :

La technique de modélisation par objet (Object Modeling Technique ou OMT) est une méthode de modélisation caractérisée par son aspect orienté objet et par sa simplicité. Une conception OMT se compose de :

- le modèle objet,
- le modèle dynamique,
- le modèle fonctionnel.
- Le modèle objet :

Le modèle objet présente l'aspect statique d'un système en montrant ses objets, les relations entre ces objets, ainsi que les attributs (variables) et les méthodes fonctions qui caractérisent classe d'objet. La construction d'un système se base sur les objets et non pas sur les fonctionnalités, car un modèle orienté objet correspond au modèle réel. Par conséquent les modifications sont plus souples à réalisées. Le modèle objet offre une représentation graphique intuitive d'un système, et permet de communiquer facilement avec les clients.

- Objets et classes :
 - ◆ Les objets :

Un objet est une abstraction ayant des limites très claires et un sens précis dans le contexte du problème étudié. Chaque objet a une identité et peut être distingué des autres.

- ◆ Les classes :

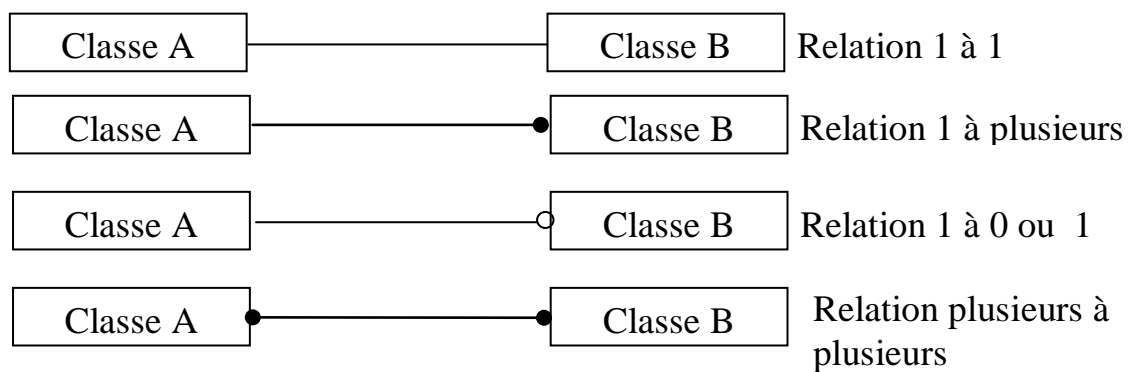
Les objets d'une classe ont le même type de comportement et les mêmes attributs. L'unicité de la plupart de ces objets provient souvent de différences dans leurs valeurs d'attributs et dans leurs relations avec les autres objets.

- Le diagramme d'objets :

Les diagrammes d'objets proposent une graphique formelle qui permet de modéliser les objets, les classes et leurs relations. Les diagrammes d'objets offrent l'avantage d'être facile à comprendre et de bien fonctionner dans la pratique.

- Liens et associations :

Les liens et les associations permettent d'établir des relations entre objets et classes. Un lien est une connexion physique ou conceptuelle entre des instances d'objets. Une association décrit un groupe de liens ayant une structure et une sémantique communes.



Types de liens entre les classes

- Le modèle dynamique :

Le modèle dynamique décrit les aspects du système en relation avec le temps et la séquence des opérations : les événements qui marquent les changements, les séquences d'événements, les états qui définissent le contexte des événements et l'organisation des états et des événements. Le modèle dynamique décrit les séquences d'opérations, sans attacher d'intérêt à ce qu'effectuent les opérations, à ce sur quoi elles opèrent ou à la façon dont elles sont implantées.

- Les événements :

Un événement est quelque chose qui produit à un moment donné dans le temps. Il n'a pas de durée (instantané) Un événement peut précéder ou suivre un autre. Chaque événement est une occurrence unique.

- Les états :

Un état est une abstraction des valeurs des attributs et des liens d'un objet. Des ensembles de valeurs sont groupés dans un état selon les propriétés qui affectent le comportement d'ensemble de l'objet. Un état correspond à l'intervalle entre deux événements reçus par l'objet. Les événements représentent un point dans le temps. Un état est généralement associé à une activité qui prend un temps d'exécution.

- Les diagrammes d'états :

Un diagramme d'état relie des événements à deux états. Quand un événement est reçu, l'état suivant dépend de l'état courant autant que de l'événement. Une modification d'état provoquée par un événement est appelée transition. Un diagramme d'état est un graphe dont les nœuds sont des états et les arcs sont des transitions désignées par les noms d'événements.

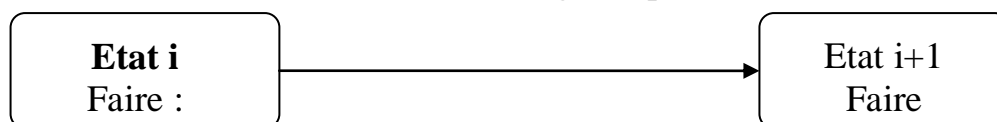


Diagramme d'état

- Le modèle fonctionnel :

Le modèle fonctionnel décrit les aspects relatifs aux transformations des valeurs. Il modélise ce que fait un système, sans s'occuper de la façon ou du moment où il le fait.

- Les traitements et les flots de données :

Le traitement transforme les valeurs de données. Un flot de donnée relie la sortie d'un objet ou d'un traitement à l'entrée d'un autre traitement. Il représente une valeur de donnée intermédiaire dans un calcul. La valeur n'est pas changée par le flot de donnée.

- Les acteurs et les réservoirs de données :

Un acteur est un objet actif qui dirige le graphe de flots de données en produisant ou en consommant des valeurs. Les acteurs sont attachés aux entrées et aux sorties d'un diagramme à flot de données.

Un réservoir de données est un objet passif à l'intérieur d'un diagramme à flot de données qui stocke des données pour un accès ultérieur. A la différence d'un acteur, un réservoir de données n'engendre pas d'opérations par lui-même mais répond simplement à des requêtes pour stocker les données et y accéder.

- Les diagrammes de flot de données :

Un diagramme de flot de données montre les relations fonctionnelles entre les valeurs calculées par un système, y compris les valeurs entrantes, les valeurs sortantes et les réservoirs de colonnes internes. C'est un graphe qui montre le flot des valeurs de données à partir de leur source dans l'objet, en passant par les traitements qui les transforment, vers leur destination dans d'autres objets.

Remarque : dans notre conception, nous nous intéressons seulement aux modèles objet et dynamique parce que l'aspect fonctionnel est déjà décrit par la spécification S.A.R.T.

2. Modèle objet :

a) Classe Robot :

Robot
Port : entier Base : bras Epaule : bras Coude : bras Poignet : bras Pince : Terminal Bras actif : type énuméré
TestCOM () Reset () Activer (bras) Déplacer () Simuler ()

○ Les attributs :

- Port : port COM sur lequel est connecté le robot
- Base : l'organe base du robot
- Epaule : l'organe épaule du robot
- Coude : l'organe coude du robot
- Poignet : poignet du robot
- Pince : l'organe terminal du robot
- BrasActif : précise l'organe actif

○ Les méthodes :

- TestCOM () : vérifie si la communication entre l'ordinateur et le robot
- Reset () : réinitialise le robot
- Activer (bras) : modifie l'organe actif
- Déplacer () : envoie une commande de déplacement au robot (fait appel au bras actif)
- Simuler () : rafraîchit l'état du robot sur l'écran

b) Classe Bras :

Bras
CodeBras : chaîne BrasPrécédent : bras Longueur : entier Largeur : entier Profondeur : entier AngleLiberté : entier Valeur : réel AncienneValeur : réel PositionX : réel PositionY : réel PositionZ : réel
CalculerPosition () Déplacer ()

○ Les attributs :

- CodeBras : Caractère identifiant le bras
- BrasPrécédent : l'organe auquel est lié le bras
- Longueur : longueur de l'axe à l'axe du bras
- Largeur : largeur de bord en bord du bras
- Profondeur : taille de haut en bas du bras
- AngleLiberté : angle maximal que peut parcourir le bras
- Valeur : comprise entre -511 et 511 et précise le déplacement du bras
- AncienneValeur : dernier déplacement effectué par le bras
- PositionX, PositionY, PositionZ : position absolue du bras

○ Les méthodes :

- CalculerPosition () : calcule les coordonnées absolues du bras
- Déplacer () : envoie une commande de déplacement du bras au robot

c) Classe Terminal :

Terminal
BrasPrécédent : bras Ouvert : booléen Vitesse : réel PositionX : réel PositionY : réel PositionZ : réel
Ouvrir () Fermer () CalculerPosition ()

- Les attributs :
 - BrasPrécédent : Bras auquel est lié l'organe terminal
 - Ouvert : précise l'état de l'organe terminal (ouvert / fermé)
 - Vitesse : valeur comprise entre -511 et 511 précisant la vitesse de fermeture ou d'ouverture de l'organe terminal
 - PositionX, PositionY, PositionZ : position absolue de l'organe terminal
- Les méthodes :
 - Ouvrir () : ouvre la pince
 - Fermer () : ferme la pince
 - CalculerPosition () : calcule la position absolue de l'organe terminal

d) Relation entre les classes :

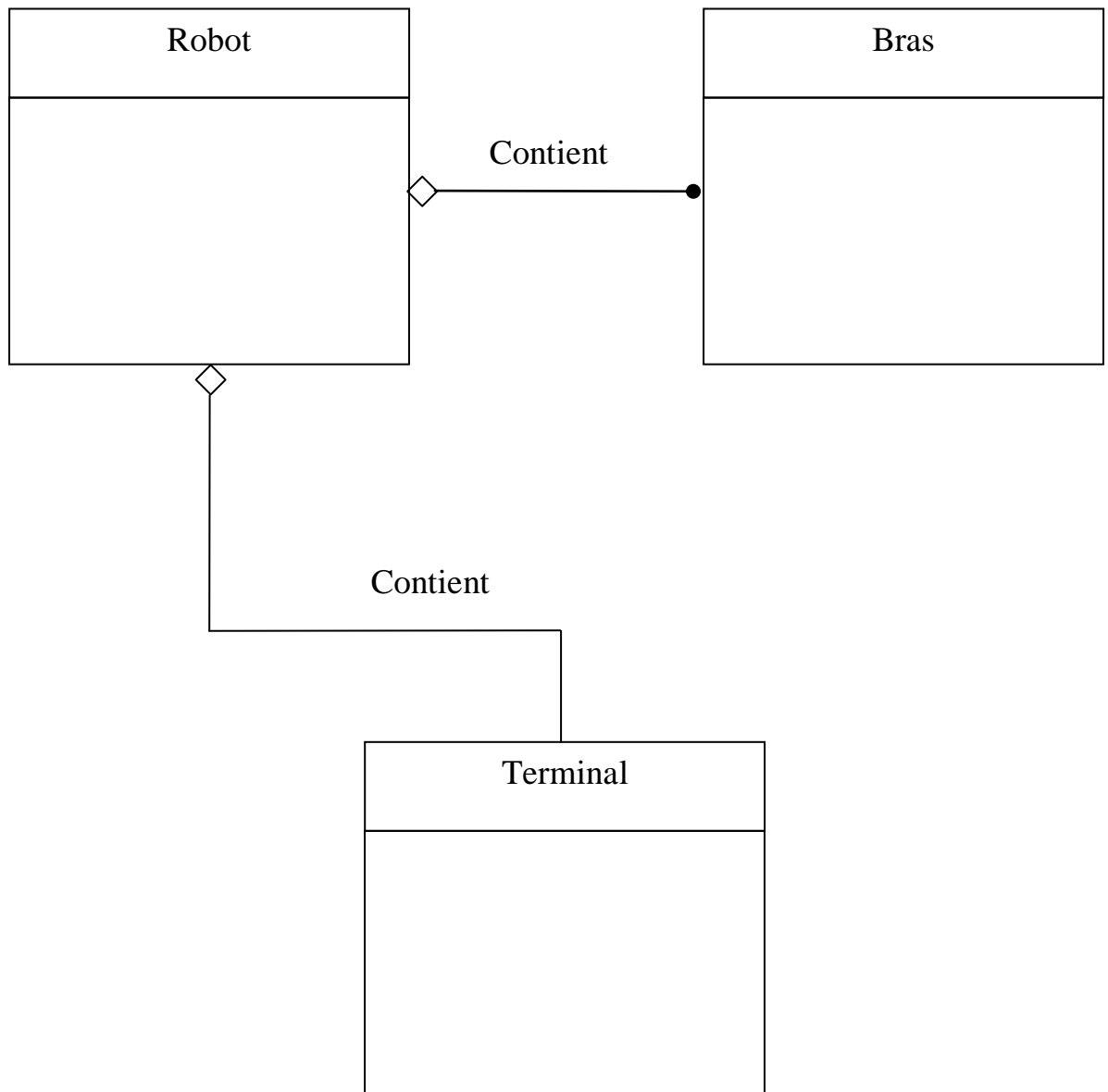


Diagramme des relations entre les classes

3. Modèle dynamique

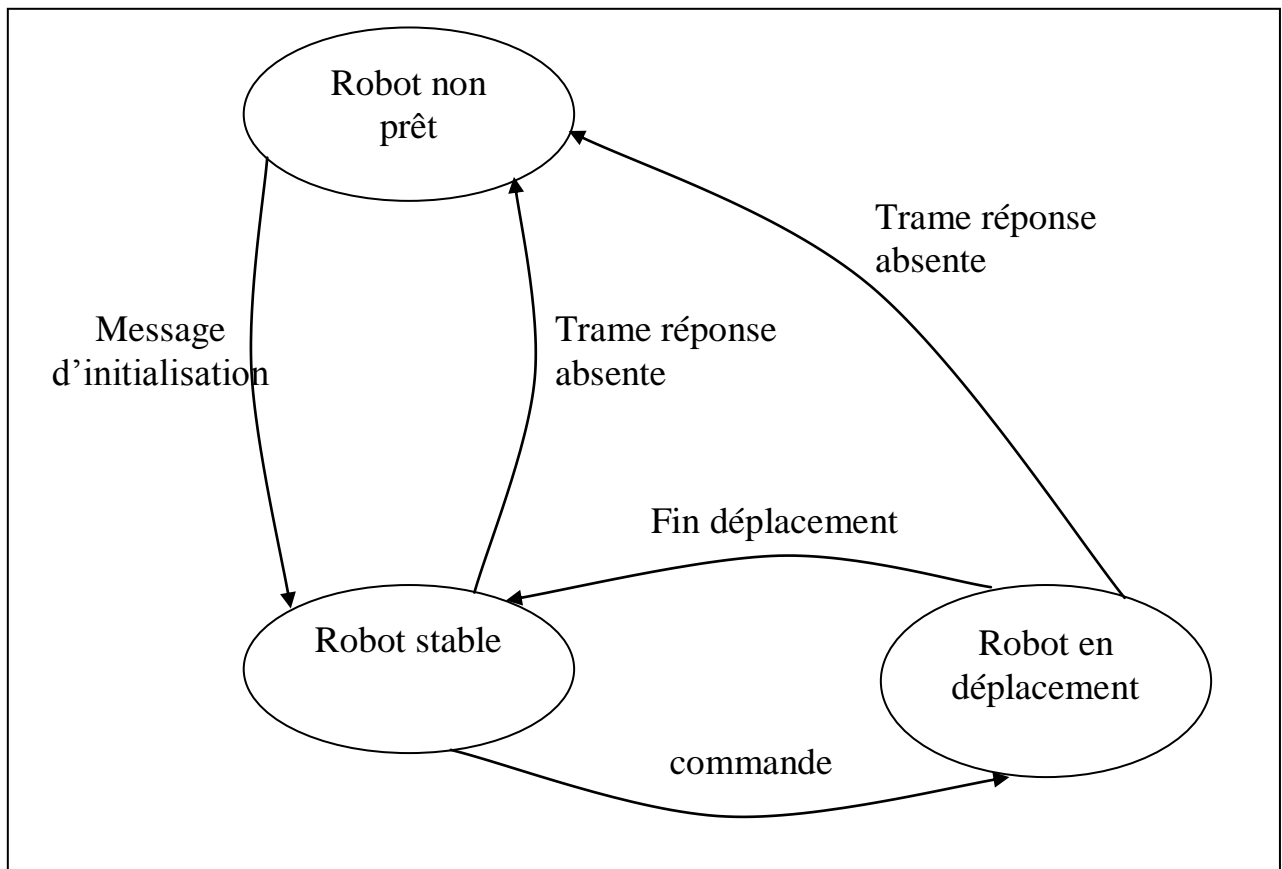


Diagramme d'état de la classe Robot

V. Aspect matériel

1. Introduction

Vu l'absence de la documentation du module de commande 45000 et sa brève description dans un ancien rapport de projet de fin d'études, nous étions face au problème de méconnaissance des différentes commandes de la carte, nous avons alors recours à rechercher une solution pour réaliser une commande du robot en boucle fermée. Cette solution est une carte à base de convertisseur analogique numérique qui aura pour rôle d'assurer le suivi de l'état du robot lors de son déplacement.

Cette carte n'a pas été réalisée faute de temps puisque nous avons vainement recherché la documentation du module de commande et l'idée de remédier à ce problème était tardive.

2. Problématique

Le module de commande 45000 offre huit entrées ainsi que huit sorties numériques, nous devons alors communiquer avec la carte avec ses entrées/sorties pour, à la fois, la commander et en recevoir les données.

Le robot dispose de quatre capteurs de position qui indiquent la position de la base, de l'épaule, du coude et du poignet ; les plages de valeurs sont les suivants :

- Base : de -7,3V à 7,3V
- Epaule : de -5V à 5V
- Coude : de -7V à 7V
- Poignet : de -2,34V à 2,34V

Pour avoir une précision optimale nous devons respecter les 1023 positions que le robot est capable d'assurer pour chaque bras.

3. Conception de la carte

Une première solution serait d'utiliser le circuit ADS7806, c'est un convertisseur analogique numérique 12 bits d'une plage d'entrées de $\pm 10V$, il comporte une interface pour microprocesseur et des sorties séries et parallèles ce qui le rend facilement interfaçable. Il possède huit sorties parallèles et une entrée de commande pour sélectionner l'octet à émettre. Mais vu l'absence de ce circuit ou l'un de ses équivalents dans le marché tunisien nous utiliserons un montage à base du convertisseur ADC0808.

Le circuit ADC0808 est un convertisseur analogique numérique 8 bits d'une plage d'entrée de 0-5V et comportant un multiplexeur analogique à huit canaux commandé par trois entrées d'adresses.

Pour son fonctionnement, le ADC0808 nécessite une horloge d'entrée qui sera assurée par un circuit NE555 (voir schéma) qui va générer une fréquence de 600 KHz.

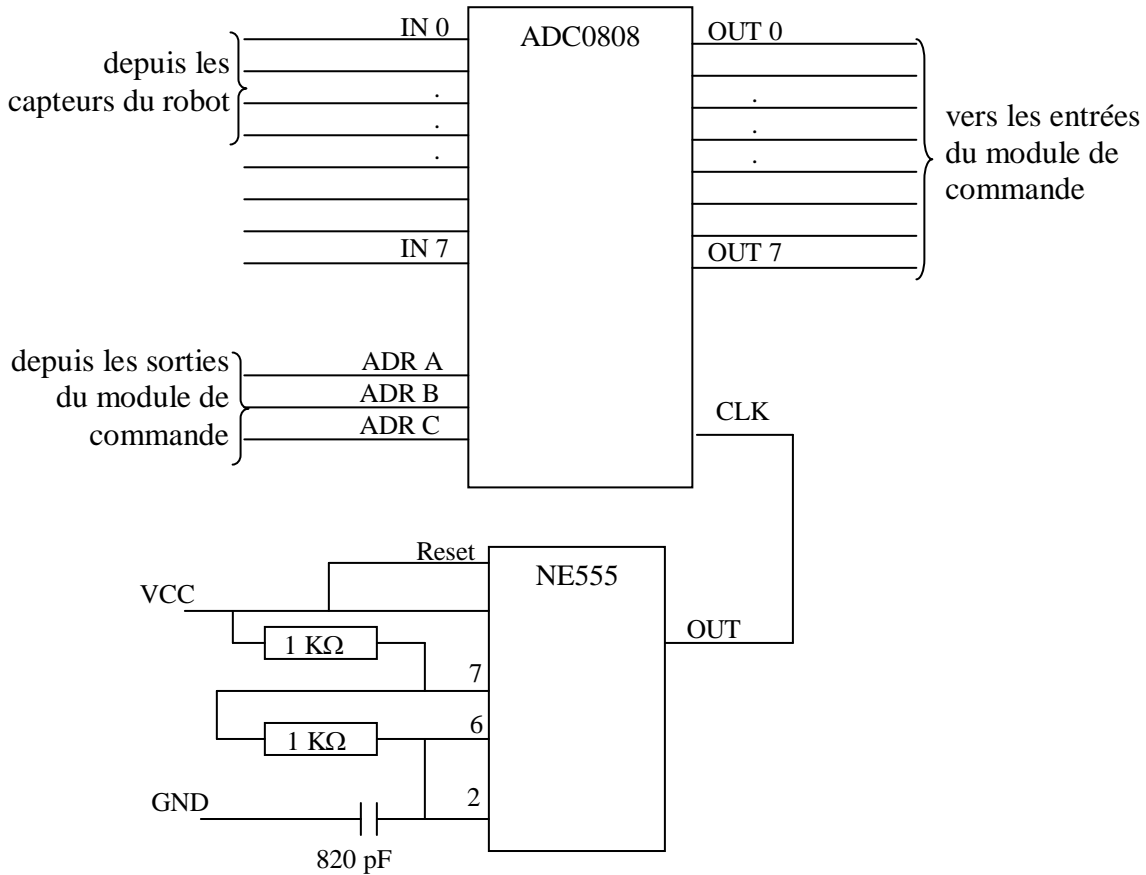
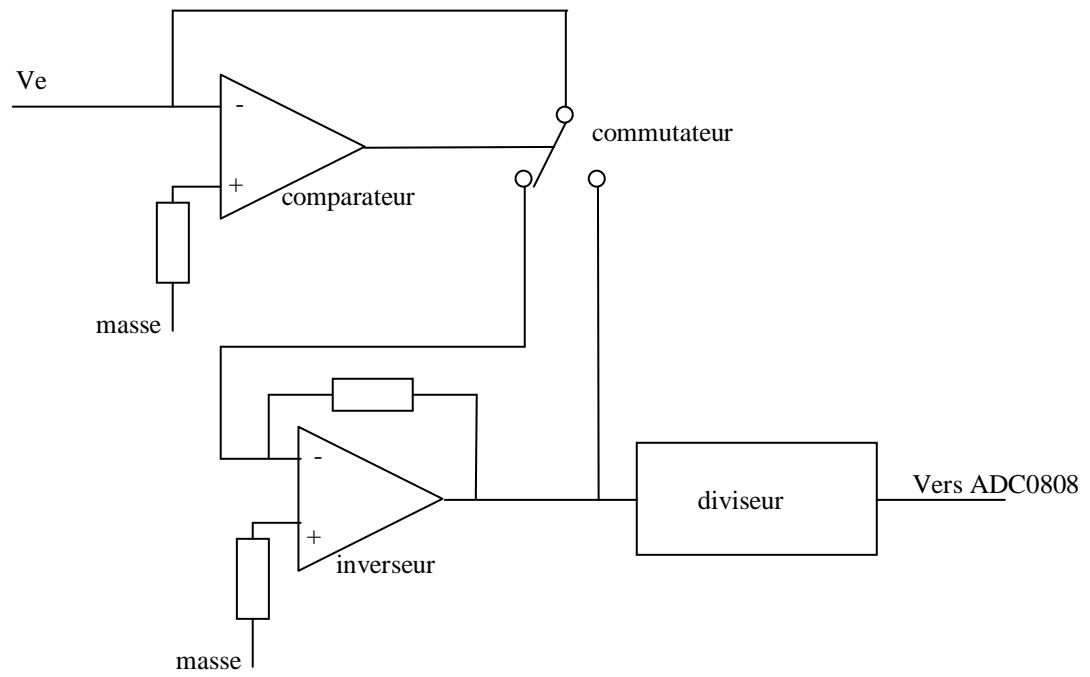


Schéma de principe

Puisque la plage des valeurs des capteurs du robot varie entre -7,3V et 7,3V et que les entrées du DAC0808 ont une plage d'entrée de 0-5V seulement, les valeurs des capteurs doivent être adaptées au convertisseur. Pour ce faire, les valeurs des capteurs passeront à travers un comparateur puis un inverseur si la tension est négative et enfin un diviseur ramènera les tensions à des tensions entre 0 et 5V selon le schéma suivant :



TROISIEME PARTIE

Implantation

I. Aspect logiciel :

1. Introduction

Dans cette section nous allons présenter la façon avec laquelle nous avons implanté notre application. Dans VB 6, chaque classe est déclarée dans un module de classe dans un fichier séparé qui contient la déclaration des propriétés, l'implantation des méthodes et des événements de la classe.

2. Environnement de travail

L'environnement de programmation à utiliser pour l'implantation de l'application est *Microsoft Visual Basic (6.0)* vue la simplicité et l'efficacité de ce langage en la conception d'interfaces graphiques sous *Microsoft Windows 9x* et le fait que c'est le seul outil de programmation sous *Microsoft Windows 9x* intégré à notre plan d'études.

3. Implantation des classes

a) Classe Robot :

- Propriétés :

```
Public Port As Byte
Public BBase As Bras
Public BEpaule As Bras
Public BCoude As Bras
Public BPoignet As Bras
Public TPince As Terminal
Public BrasActif As MbrActif
```

- Les méthodes :

- Reset ()

```
Public Sub Reset()
    BBase.Valeur = 0           'Réinitialise la base
    BrasActif = MbrBase
    Deplacer
    FrmMain.mTimer.Interval = 100

    BEpaule.Valeur = 0       'Réinitialise l'épaule
    BrasActif = MbrEpaule
```

Deplacer

BCoude.Valeur = 0 *'Réinitialise le coude*

BrasActif = MbrCoude

Deplacer

BPoignet.Valeur = 0 *'Réinitialise le poignet*

BrasActif = MbrPoignet

Deplacer

TPince.Valeur = 511 *'Ferme la pince*

TPince.Fermer

TPince.Valeur = 0

BrasActif = MbrBase

End Sub

- Activer (bras)

Public Sub Activer(BrasSelect As MbrActif)

BrasActif = BrasSelect

End Sub

- Déplacer ()

Selon la valeur de *BrasActif*, cette méthode fait appel à la méthode *Déplacer* du bras actif. Dans le cas de l'organe terminal, on fait appel soit à la méthode *Fermer* ou soit à la méthode *Ouvrir* de la pince.

Public Sub Deplacer()

Select Case BrasActif

Case MbrBase

BBase.Deplacer

Case MbrEpaule

BEpaule.Deplacer

Case MbrCoude

BCoude.Deplacer

Case MbrPoignet

BPoignet.Deplacer

Case MbrPince

If TPince.Ouvert Then

TPince.Fermer

Else

TPince.Ouvrir

End If

End Select

End Sub

b) Classe Bras :

- Propriétés :

Public CodeBras As String
Public BrasPrecedent As Bras
Public Longueur As Byte
Public Largeur As Byte
Public Profondeur As Byte
Public AngleLiberté As Integer
Public Valeur As Integer
Public AncienneValeur As Integer
Public PositionX As Integer
Public PositionY As Integer
Public PositionZ As Integer

- Les méthodes :

- Déplacer ()

```
Public Sub Deplacer()
  Dim StrCommande As String           'Chaîne de commande
  Dim i As Byte

  FrmMain.mTimer.Interval = 2       'Délai entre l'émission de deux
                                       'caractères successifs

  StrCommande = CodeBras
  If Valeur >= 0 Then
    StrCommande = StrCommande & "+"
  End If
  If Abs(Valeur) < 100 Then StrCommande = StrCommande & "0"
  If Abs(Valeur) < 10 Then StrCommande = StrCommande & "0"
  StrCommande = StrCommande & Valeur
  For i = 1 To 5
    FrmMain.MSComm.Output = Mid(StrCommande, i, 1)
    Delai = False
    FrmMain.mTimer.Enabled = True
    Do While Delai = False
      DoEvents
    Loop
  Next
```

End Sub

c) Classe Terminal :

- Les attributs :

Public BrasPrecedent As Bras
Public Ouvert As Boolean
Public Valeur As Integer
Public PositionX As Integer
Public PositionY As Integer
Public PositionZ As Integer

○ Les méthodes :

• Ouvrir () :

```

Public Sub Ouvrir()
    Dim StrCommande As String
    Dim i As Integer
    StrCommande = "P-"
    If Valeur < 100 Then StrCommande = StrCommande & "0"
    If Valeur < 10 Then StrCommande = StrCommande & "0"
    StrCommande = StrCommande & Valeur
    For i = 1 To 5
        FrmMain.MSComm.Output = Mid(StrCommande, i, 1)
        Delai = False
        FrmMain.mTimer.Enabled = True
        Do While Delai = False
            DoEvents
        Loop
    Next
    Ouvert = False
End Sub

```

• Fermer () :

```

Public Sub Fermer()
    Dim StrCommande As String
    Dim i As Integer
    StrCommande = "P+"
    If Valeur < 100 Then StrCommande = StrCommande & "0"
    If Valeur < 10 Then StrCommande = StrCommande & "0"
    StrCommande = StrCommande & Valeur
    For i = 1 To 5
        FrmMain.MSComm.Output = Mid(StrCommande, i, 1)
        FrmMain.mTimer.Enabled = True
        Do While Delai = False
            DoEvents
        Loop
    Next
    Ouvert = True

```

End Sub

II. Perspectives :

A cause de l'absence de la documentation de la carte de puissance (module de commande), nous proposons la réalisation d'une carte à base de microcontrôleur pour remplacer celle existante à fin de pouvoir mieux exploiter le robot. Cette carte pourra réaliser les tâches suivantes :

- commander le déplacement du robot,
- informer le PC sur l'état du robot et la position des différentes articulations,
- offrir des entrées/sorties logiques ou même analogiques.

Cette carte peut invoquer des éventuelles améliorations à l'application, on pourrait alors commander le robot en boucle fermée, c'est-à-dire avoir la possibilité de vérifier la position des articulations du robot après une commande de déplacement pour en assurer le bon fonctionnement.

Une autre amélioration peut être apportée à l'application, nous pourrions réaliser la simulation du robot en trois dimensions au lieu d'une simulation plane en présence d'une bibliothèque de l'environnement de travail facilitant des représentations graphiques rapides et fluides.

CONCLUSION

Au terme de ce projet de fin d'études, nous soulignons l'occasion qui nous a été offerte de s'approfondir dans le domaine de la robotique, domaine très en vogue dans l'industrie d'aujourd'hui, et de la réalisation d'applications sous *Windows*.

Dans le cadre de ce travail nous avons modélisé le robot T45, spécifié l'application avec la méthode SART et l'avons conçue en s'appuyant sur le modèle OMT.

Au niveau de la programmation, nous avons implanté notre application sous *Microsoft Visual Basic 6.0* assurant ainsi la commande et la simulation du robot T45.

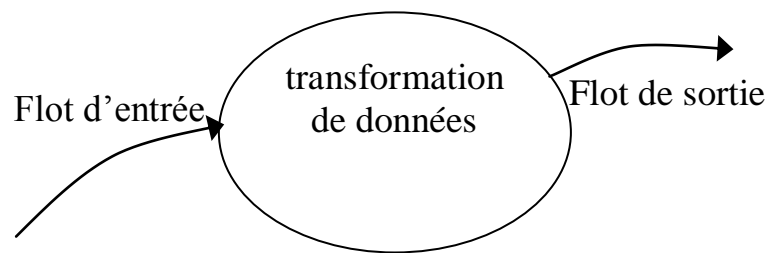
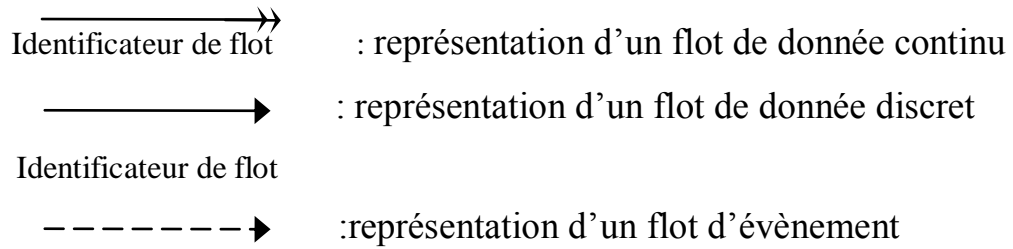
Nous pouvons suggérer, en terme de perspective de ce travail, d'intégrer le robot T45 avec d'autres applications automatisées didactiques à fin de former une application automatisée didactique proche de celles industrielles.

ANNEXES

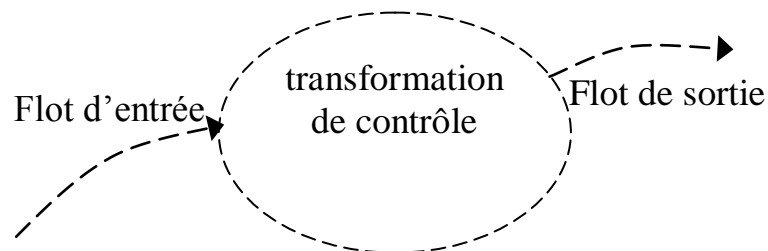
Règles de spécification

- Composants des schémas de transformations (ST) :

L'aspect fonctionnel du modèle de comportement est représenté à l'aide des outils de l'analyse structurée, avec l'extension des flots de données continus ou discrets dans le temps et dont les représentations sont les suivantes :



Représentation de transformation de données



Représentation de transformation de contrôle

- Règles de formation d'un ST :

La cohabitation des données et des évènements, des traitements et du contrôle, impose des règles spécifiques de formation des diagrammes.

Les relations entre les activeurs/désactiveurs sont les suivantes :

Acteur suivant \ Acteur courant	D	A	I
D	Sans effet	Interdit	interdit
A	interdit	Sans effet	A
I	Interdit	I	Sans effet
A	autorisation	D	déclenchement
I	Inhibition		

Les principales règles de formation qui permettent de séparer les traitements du contrôle, ainsi que les données des évènements, sont les suivantes :

- ♦ une transformation de contrôle n'accepte que des flots d'évènements en entrée et ne produit que des flots d'évènements en sorties,
- ♦ une transformation de contrôle peut avoir des activeurs/désactiveurs en entrée, c'est à dire être activée ou désactivée, par une transformation de contrôle,
- ♦ une transformation de contrôle peut avoir des activeurs/désactiveurs en sortie, c'est à dire être activée ou désactivée des transformations de contrôle ou de données,
- ♦ seuls des flots de données et des activeurs/désactiveurs sont acceptés en entrées des transformations de données primitives, mais elles peuvent produire, en sortie, des flots de données ou d'évènement,
- ♦ toute transformation primitive de flot de donnée continu doit être explicitement activée ou désactivée,

♦ une transformation de donnée primitive peut exécuter un contrôle sur l'environnement, par la biais d'un flot d'évènement en sortie.

Le tableau suivant résume l'ensemble des connexions possibles, toutes les transformations de données du tableau sont considérées comme primitives.

de \ vers	transf. de contrôle	transf. de contrôle	stock . de données	stock . évènement	bord
transf. de donnée	→ →→	--→	→	--→	---→ → →→
transf. de donnée	A/I/D →	A/I/D → --→	Interdit	--→	Interdit
stock . évènement	→	Interdit	Interdit	Interdit	Interdit
stock . évènement	Interdit	--→	Interdit	Interdit	Interdit
bord	→ →→	--→	Interdit	--→	Interdit
Connexion Autorisée :					
Par activ / desac : A/I/D →			Par flot de donnée discret : →		
Par flot d'évènement : --→			Par flot de donnée continu : →→		

- Spécification des données composées :

La spécification des données composées utilise une notation syntaxique stricte, dite dictionnaire de données :

SYMBOLE	SIGNIFICATION
=	Composé de
+	Regroupement, sans ordre
[] ou [/]	Ou exclusif, sélection
{ }	Itération non bornée
variantes :	
$n\{ \}_p$	Itération de n à p
$n\{ \}$	Itération d'au moins n
$\{ \}_p$	Itération d'au plus n
$n\{ \}_n$	Itération d'exactly n
()	Optionnel, équivalent à $\{ \}_1$
« »	Délimite l'expression littérale d'une donnée
* *	Délimite un commentaire

BIBLIOGRAPHIE

- Rapport de projet de fin d'études Pr 7/93
Commande d'un bras de robot didactique par micro-ordinateur
Réalisé par : AMARA Nabil
Encadré par : Mr CHAABEN Meher
- Robotique, aspects fondamentaux :
J.P. LALLEMAND
S. ZAGHLOUL
Edition Masson
- Modèles des robots manipulateurs :
B. GORLA
M. RENAUD
Edition Cepadues
- Programmation C++, orienté objet :
Edition micro application
- OMT : Modélisation et conception orientées objet :
J. RUMBAUGH
- Systèmes temps réel :
J.P. PEREZ
Edition Bordas
- Site du laboratoire de robotique mobile du GIT
<http://www.cc.gatech.edu/aimosaic/robot-lab/MRLHome.html>