République algérienne démocratique et populaire

Ministre de l'enseignement et de la recherche scientifique

Université de Batna

Faculté des Sciences de l'ingénieur

Département d'Electronique

Mémoire

En vue de l'obtention du diplôme de :

Magistère en électronique

Option : Robotique

Présenté par :

Fourar Redha

Ingénieur d'état en électronique

Thème

Détection De Scènes Et Reconnaissance D'objets Dans Une Opération D'assemblage Robotisé

Soutenu le : ……/…../2009

Membres de jury

Dr. Said AOUGHLENT	*M.C*	*Univ.Batna*	*Président*
Dr. Djamel MELAAB	*M.C*	*Univ.Batna*	*Rapporteur*
Dr. Noureddine GOLEA	*Prof*	*Univ.Oum EL Bouaghi*	*Examinateur*
Dr. Noureddine ATHAMENA	*M.C*	*Univ.Batna*	*Examinateur*
Dr. Yassine ABDESSEMED	*M.C*	*Univ.Batna*	*Examinateur*

Détection De Scènes Et Reconnaissance D'objets Dans Une Opération D'assemblage Robotisé

SOMMAIRE

Introduction générale 1

Chapitre 1

Reconnaissance des formes 2

1.1. Introduction 3

1.2. Description du système de reconnaissance 3

1.3. Prétraitement 5

1.3.1. Champ de vision 5

1.3.2. Agrandissement du champ de vision 6

1.3.3. Seuillage 7

1.3.4. Détection de la présence d'objets 8

1.3.5. Filtrage 9

1.3.5.1. Filtrage Linéaire 9

a.Filtrage dans le domaine fréquentiel 9

b.Filtrage par convolution 12

1.3.5.2. Filtre non linéaire 16

1.3.6. Détection de contours 17

a.Opérateurs du premier ordre 18

b.Opérateurs de second ordre 22

c.Filtrage dans le domaine fréquentiel 23

1.4. Séparation des pièces 24

1.5. Conclusion 27

Chapitre 2

Méthodes corrélatives pour la reconnaissance 28

2.1. Introduction 28

2.2. Correspondance de modèle (template matching) 28

2.2.1. Position et orientation des objets 31

2.2.2. Modification géométrique d'image 38

a. Translation 38

b. Rotation 39

c. Dilatation 40

2.2.3. Détermination du seuil d'acquisition 41

2.3. Rectangle de contenance 42

2.4. Conclusion 43

Chapitre 3

Méthodes vectorielles pour la reconnaissance 44
3.1. Introduction 44
3.2. Moments de Fourier-Mellin 44
3.2.1. Invariants issus de la PATFM 45
3.2.2. Application aux images numériques 46
3.2.3. Représentation par pyramide 49
3.2.4. Effet de la rotation 50
3.3. Moments de Zernike 52
3.3.1. Effet de la rotation 53
3.3.2. Normalisation à la translation 54
3.3.3. Normalisation à l'échelle 54
3.4. Conclusion 56

Chapitre 4

Les réseaux de neurones artificiels (RNA) 57
4.1. Introduction 57
4.2. Le neurone biologique: 57
4.3. Le neurone formel 58
4.4. Réseau monocouche 58
4.5. Réseau multicouche 60
4.6. L'apprentissage des réseaux de neurones 61
4.6.1. L'apprentissage supervisé 61
4.6.2. L'apprentissage non supervisé 62
4.7. Normalisation des entrées 62
4.8. Algorithmes d'apprentissage 62
4.8.1. Rétro-propagation du gradient 62
4.8.2. Algorithme de Levenberg-Marquardt 66
4.9. Répartition de la base de d'exemples 69
4.10. Conclusion 69

Chapitre 5

Résultats expérimentaux 70
5.1. Introduction 70
5.2. Méthodes corrélatives 70

5.2.1. Correspondance de modèle 70
5.2.2. Rectangle de contenance 71
5.3. Méthodes vectorielles 72
5.3.1. Détermination de la structure du Réseaux 71
5.3.2. Résultats et évaluation 77
5.4. Conclusion 80

Chapitre 6

Le Robot Manipulateur 81
6.1. Introduction 81
6.2. Choix de robot 82
6.3. Systèmes de coordonnées 83
6.4. Modélisation 87
a.Modèle Géométrique Direct (MGD) 87
b.Modèle géométrique inverse (MGI) 91
c.Modèle cinématique 95
d.Modèle dynamique 97
6.5. Résultats de la modélisation obtenue 100
a.Géométrie 100
b.Cinématique 104
6.6. Commande du robot 110
a.Commande par la dynamique inverse 111
b.Création des trajectoires 112
6.7. Conclusion 119

Conclusion générale

Conclusion générale 120

Annexes

Annexes

Bibliographie

Bibliographie

Introduction Générale

Introduction Générale

En détection automatique de défauts d'aspects, l'information essentielle attendue d'un système de vision artificielle se réduit à une décision « binaire » du type absence ou présence de défauts dans la scène d'analyse.

On recherche de façon analogue en robotique, la présence d'objets et d'obstacles dans le champ de travail d'un manipulateur. Le problème de détection automatique suppose donc une reconnaissance de formes particulières (défauts, objets, obstacles) définies selon l'application en cours.

Si la décision finale s'énonce assez simplement, les étapes nécessaires à son obtention sont souvent complexes; complexité liée à la morphologie des formes recherchées et à la nature de l'environnement propre au champ d'analyse.

En l'absence d'une méthode globale, les nombreux travaux menés en reconnaissance de formes reposent sur trois types d'approches sensiblement différentes, à savoir:

- ***Les méthodes corrélatives***

La forme inconnue est comparée à un ensemble de formes types afin d'effectuer une opération de mise en correspondance (soustraction point à point, distance segment à segment, corrélation, . . .). L'identification issue de la meilleure correspondance est basée sur un critère d'optimalité.

L'inconvénient majeur de ces méthodes, bien que leur aptitude à résoudre des cas complexes ait été démontrée, réside dans la nécessité de devoir prétraiter préalablement les images, opérations coûteuses en temps de calcul et souvent prohibitives en analyse de scènes industrielles.

- ***Les méthodes syntaxiques***

Chaque forme est définie comme une succession de primitives dont les règles d'enchaînement constituent une grammaire. La phase d'apprentissage consiste en une représentation arborescente des enchaînements types, l'identification est effectuée par comparaison du chemin liant les primitives de la forme à identifier aux chemins types constituant l'arbre de recherche.

- ***Les méthodes vectorielles***

Une forme inconnue F est représentée dans l'espace d'observation par un vecteur X dont les composantes sont des caractéristiques particulières mesurées. La phase

d'apprentissage est un partitionnement de l'ensemble des vecteurs en classes dont les représentants $X_1, X_2, \ldots, X_n$ définissent des formes types $F_1, F_2, \ldots, F_n$. La reconnaissance est obtenue par un test d'appartenance de X à l'une des classes précédemment définies.

Parmi ces trois groupes de méthodes, nous avons utilisé dans ce travail pour la reconnaissance de formes les méthodes corrélatives et les méthodes vectorielles.

Objectif du travail

Il s'agit d'un système robotisé de séparation d'objets. Un ensemble (fini) d'objets arrive sur un tapis roulant. L'orientation de ces objets est quelconque. Ces objets sont destinés à être placées dans des positions différentes, selon leurs classes, pour une utilisation ultérieure. Le système est équipé de :

1. Une caméra de surveillance disposée au dessus du tapis
2. Un ordinateur (PC)
3. Un bras manipulateur pour porter les objets vers leurs positions appropriées

L'ensemble camera ordinateur doivent fournir les informations suivantes afin que le bras manipulateur peut s'orienter vers les objets.

1. La position, l'orientation et l'axe principal de ces objets
2. La largeur de ces pièces
3. Le type de chaque objet

Organisation de la thèse

Le mémoire résumant le travail réalisé est organisé autour de six chapitres.

Chapitre 1 : reconnaissance des formes.

Chapitre 2 : méthodes corrélatives pour la reconnaissance.

Chapitre 3: méthodes vectorielles pour la reconnaissance.

Chapitre 4 : les réseaux de neurones artificiels.

Chapitre 5 : résultats expérimentaux.

Chapitre 6 : le robot manipulateur.

CHAPITRE 1

Reconnaissance Des Formes

Reconnaissance Des Formes

1.1. Introduction

Le schéma classique d'un processus de reconnaissance de formes, présenté dans la figure 1.1, permet de décrire les principaux traitements que l'on peut être amené à effectuer et leurs objectifs. On peut distinguer deux étapes pour la reconnaissance d'objets, la première consiste à apprendre la description d'objets à partir d'une base d'exemples (apprentissage), la seconde va reconnaitre un objet à partir de sa description extraite de l'image (classification).

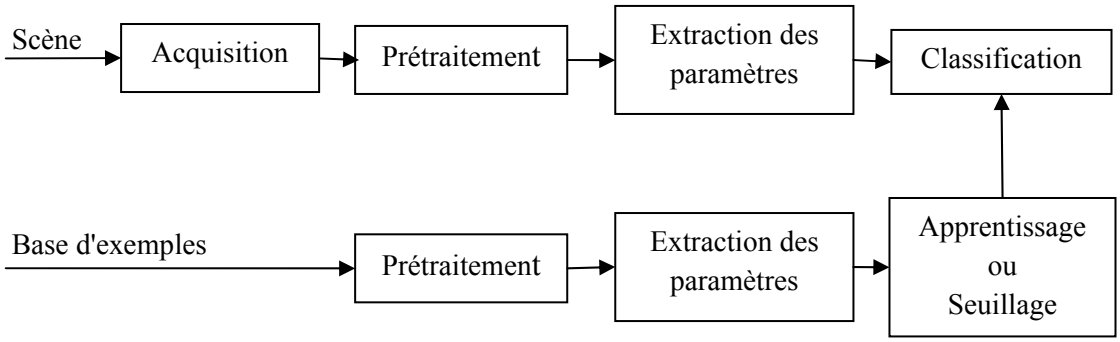


Figure 1.1 : Système de vision artificielle.

1.2. Description du système de reconnaissance

- ***Acquisition***

La première étape d'une chaîne de traitement et d'analyse des images numériques est celle de l'acquisition de l'image. On doit s'assurer d'être entouré de meilleures conditions pour de bonnes prises. Notre système d'acquisition est composé des éléments suivant, (Figure 1.2):

- Une caméra de type Webcam CCD (Charge Coupled Device)
- Un support de caméra constitué d'une tige d'une longueur de 102 cm, soudée sur une base circulaire de 15 cm de diamètre. L'assemblage de la caméra et de la tige est fait de sorte qu'il assure un déplacement facile de celle ci
- Un ordinateur PC

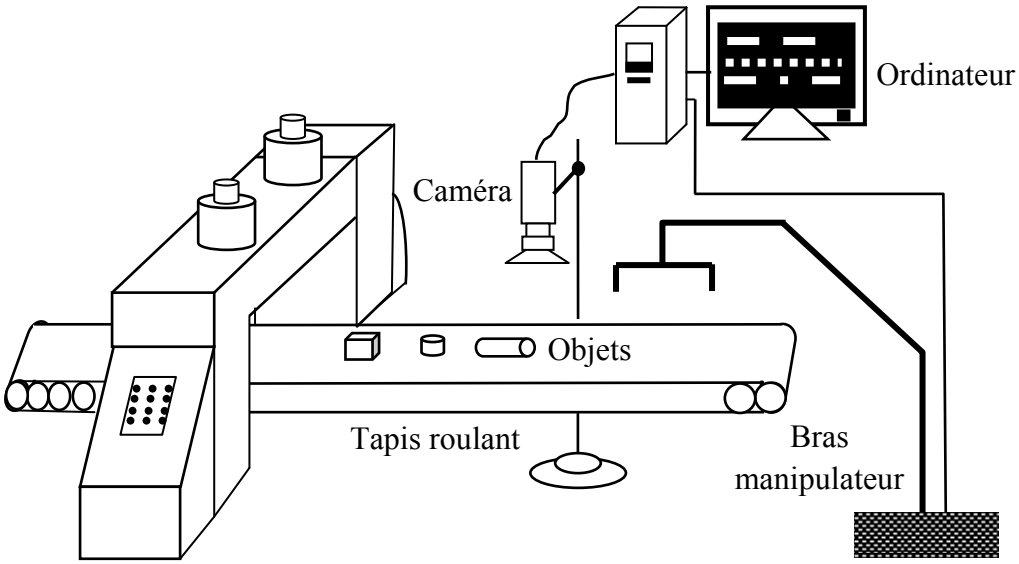


Figure 1.2 : Système de classement d'objets artificielle.

La caméra est composée d'une matrice de photodiode. Chaque photodiode traduit la luminance par l'émission d'un signal électrique positif compris entre 0,3v et 1v (0,3v : noir, 1v : blanc et entre les deux se situent les différents niveaux de gris). Le dispositif électronique d'acquisition se charge de récolter les signaux, émis par les photodiodes, ligne par ligne de la matrice CCD espacée d'un bref temps de synchronisation.

Vient ensuite la conversion des signaux analogiques en données numériques positives comprises entre 0 et 255 (0 : noir, 255 : blanc et entre les deux se situent les différents niveaux de gris).

L'image issue de la caméra CCD (caméra à transfert de charge) est obtenue sous forme de Bitmap dont la résolution est de 480x640 pixels, chaque pixel est codé sur 24bit donc il peut avoir une valeur comprise entre 0 et 255 soit 255^3 possibilités de couleurs.

L'acquisition se fait dans des conditions naturelles d'éclairage. Dans les situations de manque d'éclairage, un réglage de la caméra permet une compensation de ce manque. Le détail de ce réglage sera discuté dans les prochains chapitres.

- ***Prétraitement***

L'image brut résultat de l'acquisition n'est pas encore prête pour subir l'opération de classification. En effet le bruit qui accompagne l'opération d'acquisition, l'orientation et le positionnement quelconque des objets font que certains prétraitements sont nécessaires pour se doter de plus de chance de réussir cette

opération de classification. On parle alors de normalisation. Nous discuterons en détail comment est opérée une telle normalisation.

- ***Extraction des caractéristiques***

Le prétraitement a fini par nous donné une image dotée de meilleurs chance pour un classement correct. L'étape suivante est d'extraire de cette image un ensemble de paramètres (caractéristiques) qui serviront à l'identifier.

- ***Apprentissage et seuillage***

Cette étape dépend de la méthode utilisée pour identifier l'objet, apprentissage dans les cas des méthodes vectorielles et seuillage dans les cas des méthodes corrélatives.

- ***Classification des formes***

C'est la dernière étape de toute la chaîne. L'objet pièce a fini par être classé dans une telle ou telle classe selon le résultat des étapes précédentes.

1.3. Prétraitement

Le prétraitement, nous l'avons déjà dit, est l'opération qui prépare l'image pour la suite des opérations en lui donnant une sorte de normalisation. Cette normalisation concerne l'orientation, le positionnement et les dimensions des objets. Nous discuterons ici les méthodes utilisées pour ce prétraitement et leur influence sur la décision du système.

Les objets arrivent sur un tapis roulant d'une manière imprévisible : On ne connaît pas, à priori, le temps d'arrivé de ces objets. Le système est configuré de manière à assurer une détection automatique des ces objets. Un ***champ de vision*** est alors fixé. C'est un espace de contrôle et tout objet (pièce) qui y pénètre est automatiquement détecté. Cette détection déclenche alors une suite d'opération qui assure la détermination de la position et l'orientation de chaque objet.

1.3.1. Champ de vision

Le champ de vision est l'espace de surveillance. Il a été fixé prés du centre de l'image, avec une résolution de 240x320 pixels, pour diminuer les déformations dans les coins de celle ci. Les premiers essais nous ont montrés, en fait, que ces déformations peuvent influencer le calcul de la position et l'orientation.

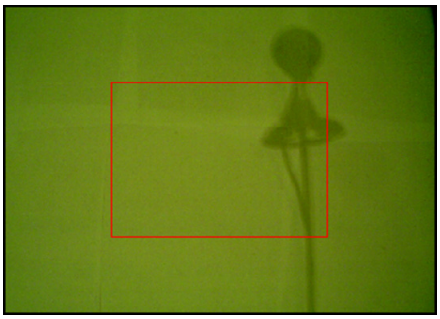


Figure 1.3 : Sélection de champ de vision.

1.3.2. Agrandissement du champ de vision

Il a été utilisé pour bien montrer l'influence des différentes opérations, détection de contour, filtrage, dilatation…etc. Cet algorithme repose sur ce qu'on appelle l'interpolation au plus proche voisin c'est à dire les pixels nouvellement crées prennent la valeur de leur plus proche voisin (Figure 1.4) [6]. Dans le cas de notre application, et après plusieurs essais, un facteur d'agrandissement égal à 2 a été retenu.

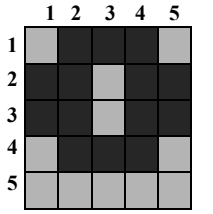


Image 5x5

a. Image originale.

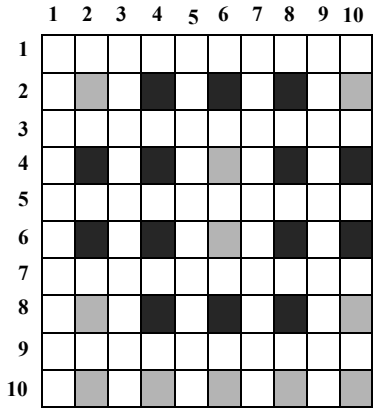


Image 10x10

b.Positions des pixels dans la nouvelle image.

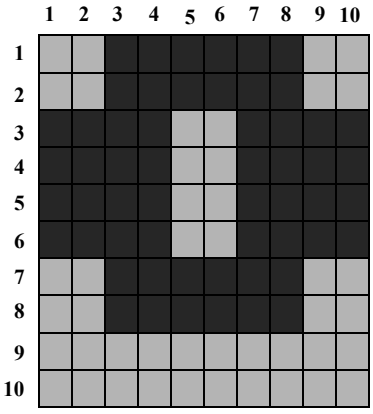
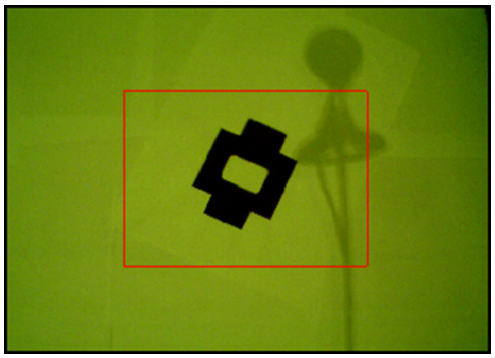
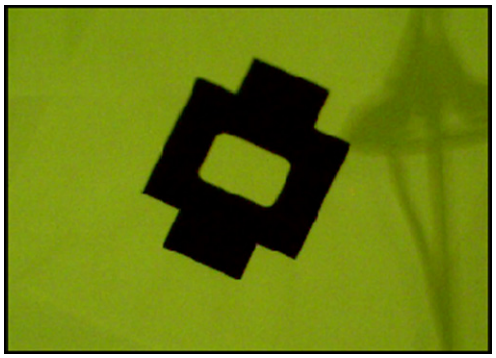


Image 10x10

c. Image agrandi.



d. Image originale de 240x320 pixels.



e.Image agrandi de 480x640 pixels.

Figure 1.4 : Agrandissement d'une image.

L'algorithme est le suivant:

Début

Calcul du facteur d'agrandissement horizontal et vertical.

Calcul des nouvelles positions des pixels de l'image original dans l'image agrandi.

Pour chaque pixel (i, j) de l'image agrandi **Faire**

Si le facteur d'agrandissement est un entier **ou** la partie fractionnaire de facteur d'agrandissement < 0.5.

On prend on considération seulement la partie entière de facteur d'agrandissement (arrondissement au plus petit).

Entourer chaque pixel de la nouvelle image par d'autres pixels portant la même valeur. Le nombre de tours dépend des facteurs d'agrandissement horizontal et vertical.

Sinon

Si la partie fractionnaire de facteur d'agrandissement >= 0.5.

Facteur d'agrandissement+1.

Entourer chaque pixel de la nouvelle image par d'autres pixels, portant la même valeur. Le nombre de tours dépend des facteurs d'agrandissement horizontal et vertical.

Fin si

Fin si

Fin pour

1.3.3. Seuillage

Une image binaire prend moins d'espace mémoire pour son stockage et elle est plus rapide à traiter qu'une image couleur ou a niveau de gris. Pour tirer profit de ces avantages on cherche, autant que possible, à ramener tout autre type d'image à une image binaire. C'est le cas pour certaines opérations en robotique telle la détection d'obstacles ou la reconnaissance d'objet. Cette conversion de type peut s'opérer facilement par un seuillage selon l'algorithme suivant:

1. Fixer un seuil
2. Donner la valeur «0» à tout pixel qui a un niveau supérieur à ce seuil
3. Donner la valeur «1» à tout pixel qui a un niveau inférieur à ce seuil

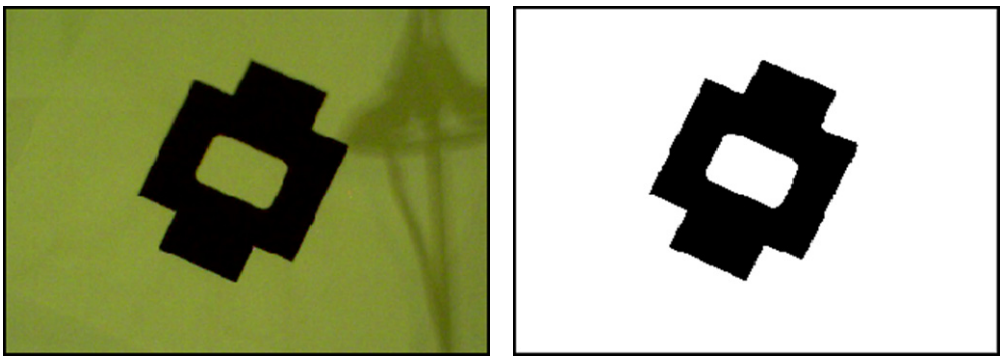
Ce seuillage donne donc à tout pixel de l'objet une couleur blanche et à tout pixel de la scène une couleur noir: c'est un objet blanc sur fond noir. L'opération inverse peut évidement être réalisée.

Le seuillage est un passage obligatoire pour toute analyse morphologique. Il permet de sélectionner les parties intéressantes de l'image, par exemple, pour notre cas où le fond a la couleur blanche on peut attribuer à tous les pixels de l'image numérique qui ont un niveau de gris compris entre deux valeurs i1 et i2 (valeurs de variation de la couleur blanche choisies empiriquement) la valeur 1 et à tous les autres pixels la valeur 0.

- ***Méthode de seuillage***

Soit $f\ (i,j)$ une image a K niveau de gris : $0, 1, 2, \dots, K-1$.Soit « T » un seuil, pouvant prendre ces valeurs dans la gamme de variation de K. Le processus du seuillage est un processus de comparaison simple, chaque valeur de pixel dans $f\ (i,j)$ est comparée au seuil T [2]. Baser sur cette comparaison, la valeur du pixel dans la nouvelle l'image g est affectée soit de « 1 » soit de « 0 ».

$$g(i,j) = \begin{cases} 1\ si\ f(i,j) \geq T \\ 0\ si\ f(i,j) < T \end{cases} \tag{1.1}$$



a. Image original. b. Image binarisée.

Figure 1.5 : Binarisation d'une image.

1.3.4. Détection de la présence d'objets

La détection de la présence d'objet est opérée selon un algorithme simple mais efficace décrit par les étapes suivantes :

- Lorsque les objets, portés par le tapis roulant, traversent le champ de vision rien ne se passe jusqu'à ce que ces objets atteignent une ligne destinée pour détecter leur présence;
- Cette ligne porte la couleur de fond, donc la somme des valeurs de ses pixels est égale à zéro ;
- La présence d'objets dans cette ligne implique un changement de la valeur de cette somme ;
- Pour bien s'assurer qu'il s'agit d'un objet et non pas d'un bruit cette somme est comparée à un seuil ;

1.3.5. Filtrage

Le bruit est l'ensemble de pixels qui ne font pas partie des pixels de la scène à traiter. C'est un intrus injecté par plusieurs sources incluant le capteur, la nature de la scène et le contexte d'acquisition. Il est préférable de nettoyer, autant que possible, l'image de cet intrus. On a recours alors au filtrage.

Différentes méthodes de filtrage ont été développées suivant le type et l'intensité du bruit, ou les applications auxquelles est destinée l'image. Les premières, et les plus simples, de ces méthodes sont basées sur le filtrage linéaire stationnaire (invariant par translations). Les limitations de ces techniques, en particulier leur mauvaise conservation des transitions, a conduit cependant au développement des filtres non-linéaire.

1.3.5.1. Filtrage Linéaire

Le bruit supplémentaire dans une image a généralement un spectre de haute fréquence que les composants normaux de l'image. Par conséquent, le filtre passe bas peut être efficace pour atténuer ce bruit, cette atténuation peut être effectuée à l'aide des deux méthodes de filtrages soit par filtrage dans le domaine fréquentiel ou filtrage dans le domaine spatial (par convolution) [2][5].

a. Filtrage dans le domaine fréquentiel

La transformée de Fourier $F(u, v)$ d'une image $f(x, y)$ est une transformation mathématique qui décompose l'image en une somme infinie de fonctions sinusoïdale. L'image obtenue n'est plus représentée par la valeur de ses pixels en fonction de leur

position dans l'espace, mais par l'amplitude et la phase de sinusoïdes en fonction de leur fréquence [3].

Soit un signal continu $f\ (x)$, vérifiant la condition suivante :

$$\int_{-\infty}^{+\infty} |f(x)|^2 dx \quad fini \tag{1.2}$$

C'est-à-dire $f\ (x)$ est à énergie finie.

Sa transformée de Fourier $F\ (u)$ est définie par :

$$F(u) = \int_{-\infty}^{+\infty} f(x) e^{-i2\pi ux} dx \tag{1.3}$$

La transformée de Fourier inverse est donnée par :

$$f(x) = \int_{-\infty}^{+\infty} F(u) e^{i2\pi ux} du \tag{1.4}$$

Une image est un signal bidimensionnel, de support fini et de nature discrète. Les intégrales des formules précédentes seront remplacées par des sommes partielles. On obtient alors le couple de transformées discrètes suivant:

$$F(u,v) = \frac{1}{N} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} f(j,k) e^{\frac{-i2\pi}{N}(uj+vk)} \tag{1.5}$$

$$F(j,k) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} f(u,v) e^{\frac{i2\pi}{N}(uj+vk)} \tag{1.6}$$

Le module $|F\ (u,\ v)|$ a l'avantage d'être une fonction réelle, désignée sous le nom de spectre d'amplitude. En notant $F_{re}\ (u,\ v)$ et $F_{im}\ (u,\ v)$ les parties réelle et imaginaire de $F\ (u,\ v)$, nous avons :

$$|F(u,v)| = \sqrt{F_{re}(u,v)^2 + F_{im}(u,v)^2} \tag{1.7}$$

La fonction $|F\ (u,\ v)|^2$ est connue sous le nom de spectre de puissance.

Le spectre de phase associé aux fréquences (argument de $F\ (u,\ v)$) est défini de la manière suivante :

$$\phi(u,v) = tan^{-1}\left(\frac{F_{im}(u,v)}{F_{re}(u,v)}\right) \quad (1.8)$$

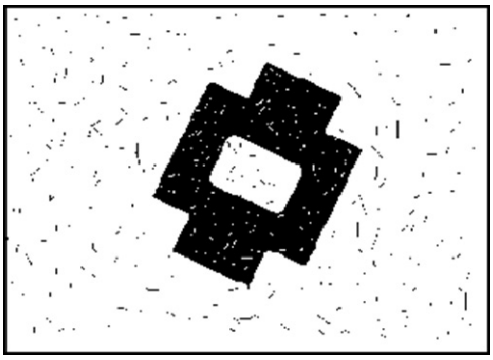
L'information pertinente sur l'image se trouve essentiellement dans l'amplitude des fréquences.

Le spectre d'amplitude de la figure 1.6.b, présenté selon une échelle logarithmique, est symétrique par rapport à l'origine. Le point situé à l'origine représente la composante continue $F\,(0,\,0)$ (la fréquence fondamentale). Tout autour de l'origine figurent les composantes de basse fréquence. Elle représente des variations de grande période. Ces composantes portent l'information de la structure de l'image. En périphérie figurent les termes de hautes fréquences. Ils représentent des variations très rapides d'un pixel à l'autre. C'est dans ce domaine que l'on rencontre les composantes liées au bruit de l'image [3][6].

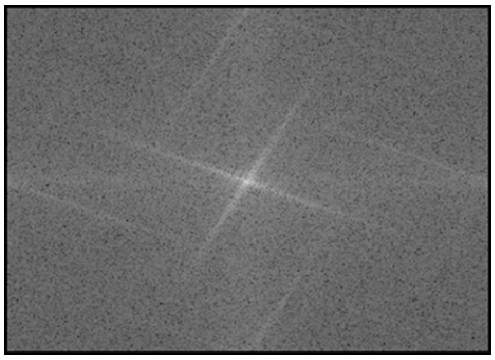
Le filtre passe-bas est défini comme suit:

$$\begin{cases} F_{pb}(u,v) = F(u,v) & si\ u \le u_m\ \ et\ v \le v_m \\ F_{pb}(u,v) = \quad 0 & sinon \end{cases} \quad (1.9)$$

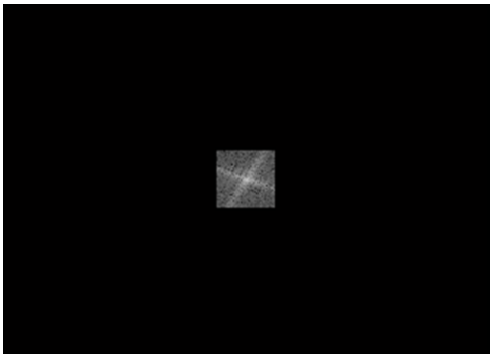
La Figure 1.6 illustre l'application de ce filtre sur une image.



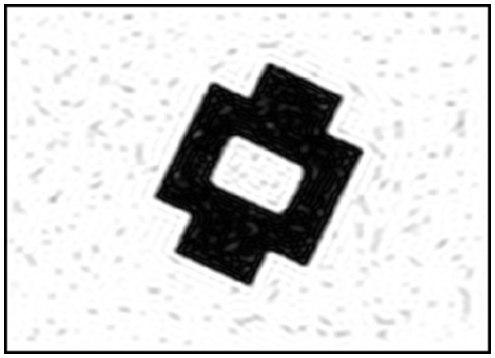
a. Image originale.



b.Spectre d'amplitude.



c. Application de filtre passe-bas.



d.Image filtrée.

Figure 1.6 : Filtrage dans le domaine fréquentiel.

La complexité de l'implantation par multiplication dans le domaine fréquentiel est celle de deux calculs de TF (TF directe et TF inverse).

b. Filtrage par convolution

Le filtrage peut être directement effectué dans l'espace image (domaine spacial) en utilisant l'opérateur de convolution, bien connu en traitement du signal. La formulation de la convolution est donnée ci-dessous, où f est l'image de départ, g l'image filtrée et h la réponse impulsionnelle du filtre [1].

$$g(i,j) = f(i,j) * h(i,j) = \sum_{n1=-\infty}^{+\infty} \sum_{n2=-\infty}^{+\infty} f(n_1, n_2) \, . \, h(i - n_1, j - n_2) \quad (1.10)$$

Où h ne dépend que de $i - n_1$ et $j - n_2$ on parle de système invariant ou stationnaire.

Pour une réponse impulsionnelle h (i, j) de dimension 3x3 le produit de convolution peut être exprimé comme suit:

$$\begin{aligned} g(i,j) = {} & h(3,3).f(i-1,j-1) + h(3,2).f(i-1,j) + h(3,1).f(i-1,j+1) \\ & + h(2,3).f(i,j-1) + h(2,2).f(i,j) + h(2,1).f(i,j+1) \\ & + h(1,3).f(i+1,j-1) + h(1,2).f(i+1,j) + h(1,1).f(i+1,j+1) \end{aligned} \quad (1.11)$$

Voir l'indice du filtre ci-dessus.

La multiplication dans le domaine fréquentiel correspond à une convolution dans le domaine spatial. Un grand nombre de filtres peut être obtenu à partir de noyaux de convolution symétriques et normalisés (de somme égale à 1). Voici trois familles de filtres parmi les plus utilisés :

- ***Filtre moyenneur***

Les filtres moyenneurs sont un type de filtres passe-bas dont le principe est de faire la moyenne des valeurs des pixels avoisinants. Le résultat de ce filtre est une image plus floue [2].

☛ *Réponse impulsionnelle:*

$$h(i,j) = \begin{cases} \frac{1}{\lambda^2} & si\ |i,j| \leq \frac{\lambda}{2} \\ 0 & Autrement \end{cases} \quad (1.12)$$

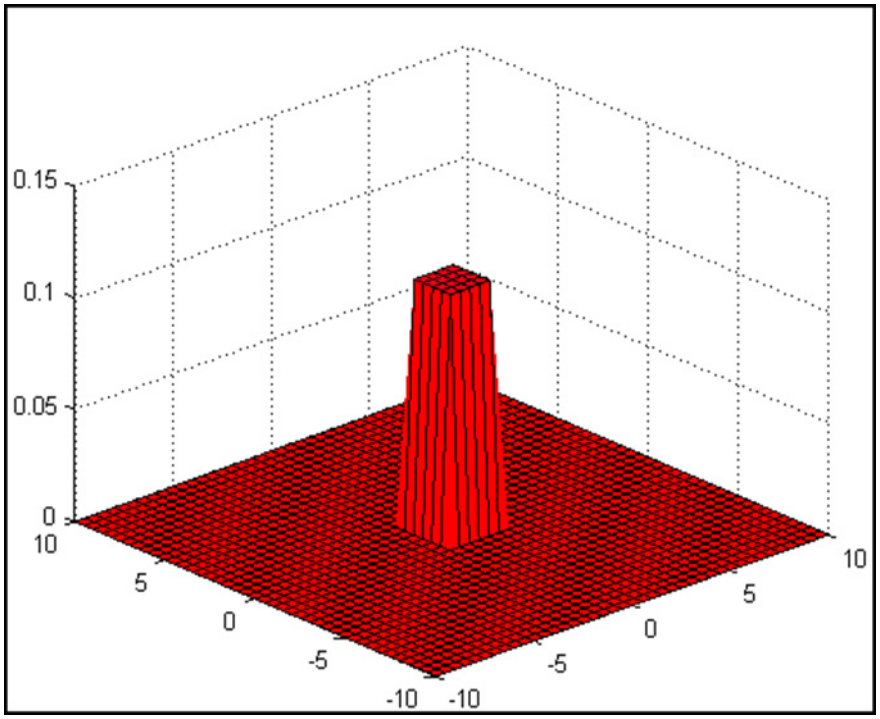
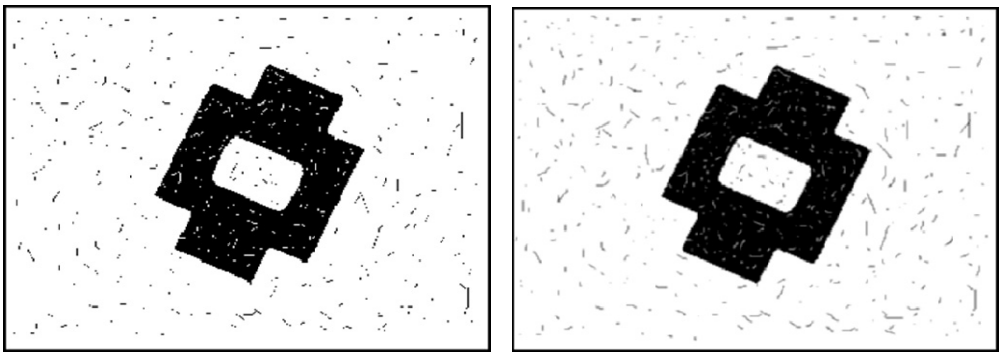


Figure 1.7 : Représentation graphique de filtre moyenneur ($\lambda = 3$)

☛ *Noyau de convolution discret pour* $\lambda = 3$:

$$h(i,j) = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{1.13}$$

La figure 1.8 illustre l'application du filtre moyen sur une image.



a.Image originale. b.Image filtrée.

Figure 1.8 : Application du filtre moyen (3x3).

- ***Filtre gaussien***

Le filtre gaussien utilise un paramètre "**σ**" appelé écart-type, ce filtre présente un meilleur compromis entre la préservation des structures locales et la réduction du bruit [2][3].

☛ *Réponse impulsionnelle:*

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{1.14}$$

Ce qui se traduit en discret par :

$$\mathrm{h(i,j)} = \frac{1}{2\pi\sigma^2} e^{\frac{-(i^2+j^2)}{2\sigma^2}} \tag{1.15}$$

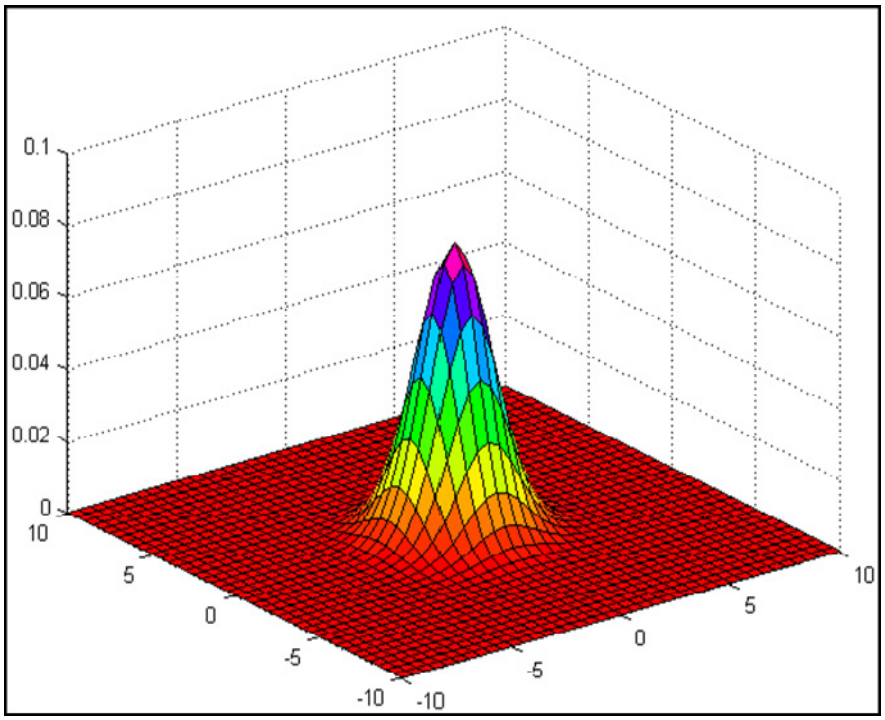


Figure 1.9 : Représentation graphique de filtre gaussien ($\sigma = 1.41$)

☛ *Noyau de convolution discret pour* $\sigma = 1.41$:

$$\mathrm{h}(i,j) = \begin{pmatrix} \ddots & & \vdots & \vdots & \vdots \\ & \ddots & \vdots & \vdots & \vdots \\ \dots & \dots & h(0,0) & h(0,1) & h(0,2) \\ \dots & \dots & h(1,0) & h(1,1) & h(1,2) \\ \dots & \dots & h(2,0) & h(2,1) & h(2,2) \end{pmatrix} \tag{1.16}$$

En replaçant $h\ (i,\ j)$ par sa valeur on trouve:

$$h(i,j) = \begin{pmatrix} \ddots & & \vdots & \vdots & \vdots \\ & \ddots & \vdots & \vdots & \vdots \\ \dots & \dots & 0.080 & 0.062 & 0.029 \\ \dots & \dots & 0.062 & 0.048 & 0.023 \\ \dots & \dots & 0.029 & 0.023 & 0.011 \end{pmatrix} \quad (1.17)$$

La multiplication par 1000 et la division par 864 donne un noyau de convolution symétrique et normalisé.

$$h(i,j) = \frac{1}{864}\begin{pmatrix} 11 & 23 & 29 & 23 & 11 \\ 23 & 48 & 62 & 48 & 23 \\ 29 & 62 & 80 & 62 & 29 \\ 23 & 48 & 62 & 48 & 23 \\ 11 & 23 & 29 & 23 & 11 \end{pmatrix} \quad (1.18)$$

- ***Filtre exponentiel***

Le filtre exponentiel utilise un paramètre $\gamma > 0$, qui détermine sa dispersion spatial (plus γ est grand, moins le filtre est dispersé spatialement). Par rapport aux autres filtres, le filtre exponentiel préserve mieux les transitions (contours), au prix d'une réduction moins visible du bruit [2].

☛ *Réponse impulsionnelle:*

$$h(x,y) = \frac{\gamma^2}{4}\, e^{(-\gamma(|x|+|y|))} \quad (1.19)$$

Ce qui donne au discret :

$$h(i,j) = \frac{\gamma^2}{4}\, e^{(-\gamma(|i|+|j|))} \quad (1.20)$$

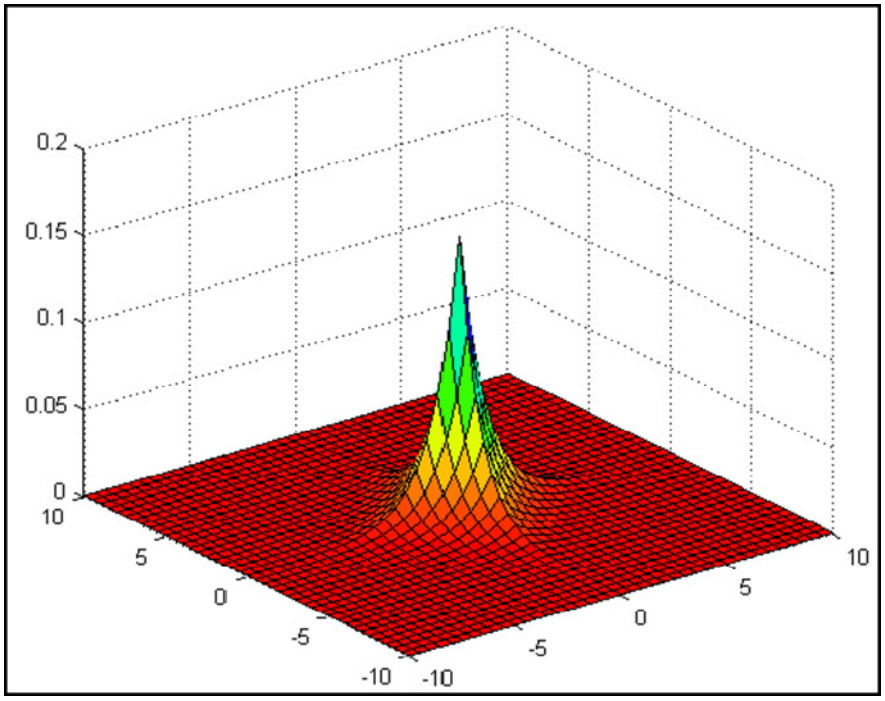


Figure 1.10 : Représentation graphique de filtre exponentiel ($\gamma = 0.8$)

☛ *Noyau de convolution discret pour* $\gamma = 0.8$:

$$h(i,j) = \frac{1}{80}\begin{pmatrix} 1 & 1 & 3 & 1 & 1 \\ 1 & 3 & 7 & 3 & 1 \\ 3 & 7 & 16 & 7 & 3 \\ 1 & 3 & 7 & 3 & 1 \\ 1 & 1 & 3 & 1 & 1 \end{pmatrix} \tag{1.21}$$

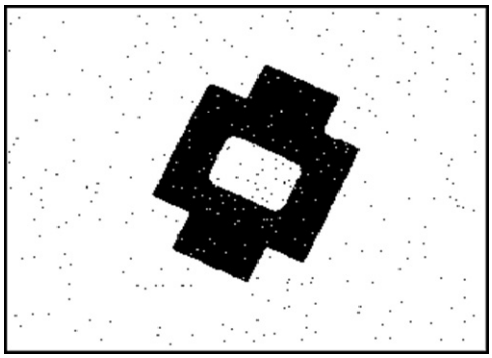
L'application du filtre linéaire rend les frontières des objets flous, ainsi il élimine mal le bruit impulsionnel figures (1.6 - 1.8).

1.3.5.2. Filtre non linéaire

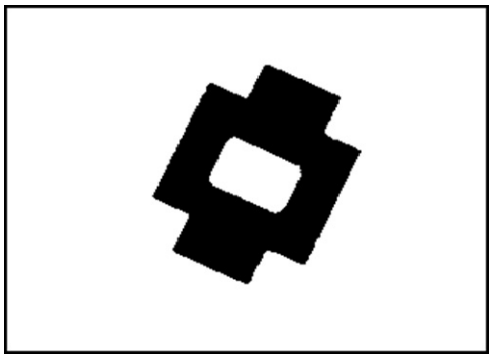
Ce type de filtrage a été développé pour pallier aux insuffisances des filtres linéaires, principalement la mauvaise conservation des contours, l'exemple le plus utilisé est le filtre médian. Pour ce filtre le niveau de gris du pixel central est remplacé par la valeur médiane de tous les pixels de la fenêtre d'analyse centrée sur le pixel [1]. La taille de la fenêtre dépend de la fréquence du bruit et de la taille des détails significatifs de l'image traitée [2]. Le calcul de la médiane est donné comme suit:

Considérons n valeurs numériques $x_1, \ldots, x_n$ (pas nécessairement distinctes), où n est impair. On les ordonne de la plus petite à la plus grande, ce qui donne la suite permutée $x_{i1}, \ldots, x_{in}$, où $\{i_1, ..., i_n\}$ est une permutation de $\{1, \ldots, n\}$. La médiane est alors la valeur placée au milieu de cette suite ordonnée, à savoir x_{im} pour m = (n+1)/2.

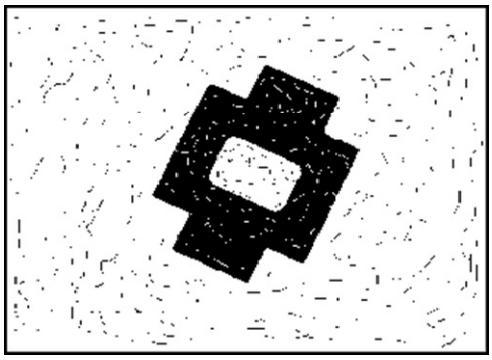
Le filtre médian est bien adapté au filtrage du bruit impulsif. Il s'appliquera aussi pour éliminer des griffes dans une image [1]. C'est ce que nous illustrons dans l'exemple ci-dessous.



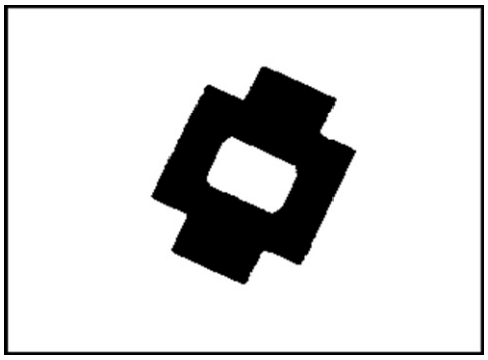
a.Image avec un bruit poivre et sel.



b.Filtrage médian 5x5 de l'image bruitée.



d.Image avec des griffes noires et blanches.



e.Filtrage médian 5x5 de l'image bruitée.

Figure 1.11 : Application du filtre médian.

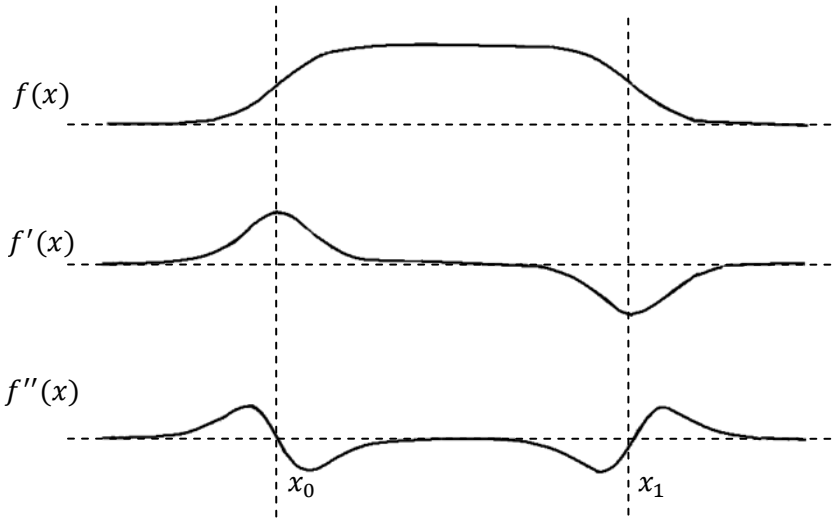
Les propriétés fondamentales du filtre médian, est qu'il ne crée pas de nouvelles valeurs de niveaux de gris dans l'image et peut s'appliquer aux images binaires (à deux valeurs), et le résultat restera une image binaire [1].

1.3.6. Détection de contours

L'idée de la détection de contours est de repérer les points d'une image numérique qui correspondent à un changement brutal de l'intensité lumineuse [2]. Elle réduit de manière significative la quantité de données et élimine les informations qu'on peut juger moins pertinentes, tout en préservant les propriétés structurelles importantes de l'image [4], la détection de contours dans notre cas permet de déterminer les limites d'objets pour la segmentation, dont le but est la séparation des objets.

Il existe un grand nombre de méthodes de détection de contours mais la plupart d'entre elles peuvent être regroupées en deux catégories. La première recherche les

extremums de la dérivée première, en général les maximums locaux de l'intensité du gradient. La seconde recherche les annulations de la dérivée seconde, en général les annulations du Laplacien (Figure 1.12) [1].



- x_0 et x_1 son les positions de contour.

Figure 1.12 : Variation de l'intensité lumineuse et ses dérivées.

a. Opérateurs du premier ordre

La première approche possible pour détecter les variations locales de la fonction $f\ (x,\ y)$ représentant l'intensité est d'utiliser une transformation du type gradient. Un contour d'orientation θ au point (x, y) est détecté par un maximum de la dérivée directionnelle, dans la direction ϕ du gradient $\overrightarrow{\nabla}$f (x, y), c'est à dire par le maximum de la fonction [2] :

$$G(x,y) = \overrightarrow{\nabla} f\ (x,y).\vec{n} \tag{1.22}$$

Où $\vec{n}$ représente le vecteur unitaire dans la direction du gradient :

$$\vec{n} = (cos\ \phi, sin\ \phi) \tag{1.23}$$

On a donc :

$$G(\mathrm{x},\mathrm{y}) = \frac{\partial f(x,y)}{\partial x}\ cos\phi\ + \frac{\partial f(x,y)}{\partial y}\ sin\phi \tag{1.24}$$

L'amplitude de gradient dans l'espace continu produit un détecteur isotrope, sensible à bords dans n'importe quelle direction (indépendant de ϕ) [2].

En plus de système de coordonnée original (x, y), on présente un nouveau système (x', y') viré d'un angle ϕ par rapport à l'ancien système, l'ensemble n'_x et n'_y sont les valeurs unitaires dans les directions x' et y' respectivement. Tant que l'amplitude de gradient est isotrope, on peut définir $G_{x'}$ et $G_{y'}$ les gradients suivants les deux axes x' et y' comme suit [2]:

$$G_{x'} = \vec{\nabla} f . \vec{n}_{x'} = \frac{\partial f(x,y)}{\partial x} \cos\phi + \frac{\partial f(x,y)}{\partial y} \sin\phi \quad (1.25)$$

$$G_{y'} = \vec{\nabla} f . \vec{n}_{y'} = -\frac{\partial f(x,y)}{\partial x} \sin\phi + \frac{\partial f(x,y)}{\partial y} \cos\phi \quad (1.26)$$

$$|G(x,y)| = \sqrt{{G_{x'}}^2 + {G_{y'}}^2} \quad (1.27)$$

$$|\mathrm{G(x,y)}| = \sqrt{\left(\frac{\partial \mathrm{f(x,y)}}{\partial \mathrm{x}} \cos\phi + \frac{\partial \mathrm{f(x,y)}}{\partial \mathrm{y}} \sin\phi\right)^2 + \left(-\frac{\partial \mathrm{f(x,y)}}{\partial \mathrm{x}} \sin\phi + \frac{\partial \mathrm{f(x,y)}}{\partial \mathrm{y}} \cos\phi\right)^2} \quad (1.28)$$

Donc l'amplitude de ce gradient est donnée par l'expression suivante:

$$|\mathrm{G(x,y)}| = \sqrt{\frac{\partial \mathrm{f(x,y)}^2}{\partial \mathrm{x}} + \frac{\partial \mathrm{f(x,y)}^2}{\partial \mathrm{y}}} \quad (1.29)$$

La figure 1.13 décrit la génération d'un gradient $G(i,j)$ de bord dans le domaine discret en termes de gradient de ligne $G_{lig}(i,j)$ et gradient de colonne $G_{col}(i,j)$. L'amplitude spatiale de gradient est donnée par :

$$|G(i,j)| = \sqrt{G_{lig}(i,j)^2 + G_{col}(i,j)^2} \quad (1.30)$$

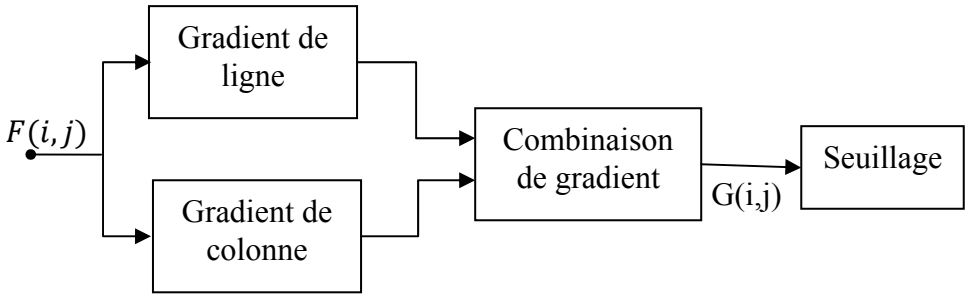


Figure 1.13 : Génération orthogonale de gradient.

L'amplitude de gradient est parfois rapprochée par la combinaison suivante [2] :

$$G(i,j) = \left|G_{lig}(i,j)\right| + \left|G_{col}(i,j)\right| \tag{1.31}$$

L'orientation du gradient est donnée par :

$$\phi(i,j) = arctan\left\{\frac{G_{col}(i,j)}{G_{lig}(i,j)}\right\} \tag{1.32}$$

- ***Génération de gradient***

La méthode la plus simple de la génération de gradient est de former la différence des Pixels le long des lignes et des colonnes de l'image [2]. Par exemple :

$$G_{lig}(i,j) = f(i,j) - f(i-1,j) \tag{1.33}$$

$$G_{col}(i,j) = f(i,j) - f(i,j-1) \tag{1.34}$$

Qui correspondent à des convolutions avec les noyaux $(-1\ \mathbf{1})$ pour G_{lig}, qui est l'approximation de $\frac{\partial f(x,y)}{\partial x}$, et $\begin{pmatrix}-1\\ \mathbf{1}\end{pmatrix}$ pour G_{col}, qui est l'approximation de $\frac{\partial f(x,y)}{\partial y}$ (le nombre en gras indique l'origine du noyau de convolution), on a fait ici un choix arbitraire quant au choix de l'origine dans l'opérateur de convolution, qui se traduit par un décalage vers la gauche dans l'estimation de la dérivée horizontale et un décalage vers le haut dans l'estimation de la dérivée verticale.

Pour cette raison, on a utilisé dans notre cas les opérateurs de convolution (-1 **0** 1) pour la dérivée horizontale, et $\begin{pmatrix}-1\\ 0\\ 1\end{pmatrix}$, pour la dérivée verticale, qui produiront des frontières bien centrées.

- ***Seuillage***

Le pixel (i,j) est déclaré comme faisant partie de la frontière si $G\,(i,\,j)$ est au dessus d'un certain seuil «t». L'ensemble des points de la frontière constituent la carte des frontières (edge map) $e(i,j)$ définie comme suit :

$$e(i,j) = \begin{cases} 1 & (i,j) \in I_g \\ 0 & ailleur \end{cases} \tag{1.35}$$

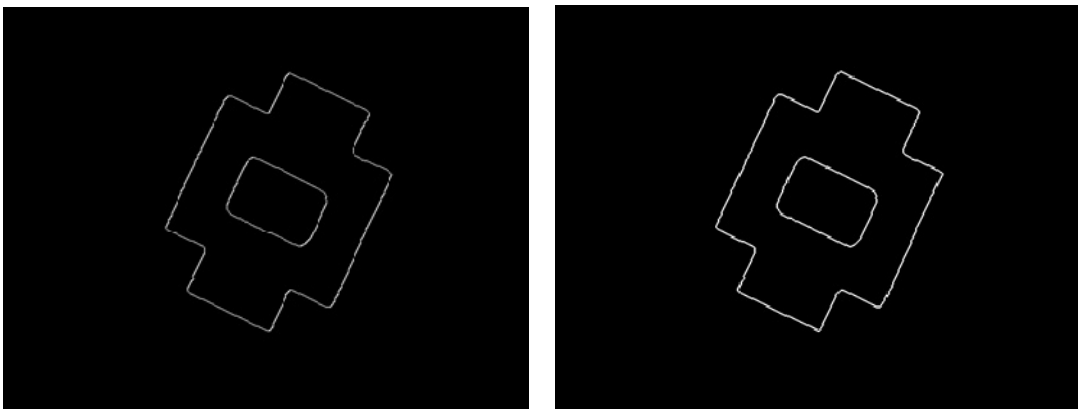
Où

$$I_g = \{(i,j); g(i,j) > t\} \tag{1.36}$$

Les masques de convolution sont donnés ci-dessous.

Opérateur	**Gradient horizontale**	**Gradient vertical**
Différence de pixel	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Différence de pixels séparés	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$

L'application de ces masques sur une image pour un seuil égale à 1 donne les résultats suivant:



a. Différence de pixel. b. Différence de pixels séparés.

Figure 1.14 : détection de contour.

b. Opérateurs de second ordre

Le Laplacien d'une image représente sa dérivée seconde. Cette dernière peut s'exprimer de plusieurs manières selon les différences finie utilisées pour approximer les dérivées partielles [1]. De ce fait, plusieurs masque de Laplacien très similaires on été développés.

Le Laplacien de la fonction Image est défini par [2] :

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{1.37}$$

Les approximations les plus simples des dérivées premières directionnelles s'obtiennent par simples différences finies [2]:

$$\frac{\partial f(x,y)}{\partial x} \rightarrow f_x(i,j) = f(i+1,j) - f(i,j) \tag{1.38}$$

La deuxième dérivée selon x peut être construite en appliquant la dérivée première à l'équation précédente. Cette deuxième application de la dérivée première peut être décalée pour (contrecarrer) annuler l'erreur présentée par la dérivée précédente [2].

$$\frac{\partial^2 f(x,y)}{\partial x^2} \rightarrow f_{xx}(i,j). = f_x(i,j). - f_x(i-1,j) \tag{1.39}$$

$$f_{xx}(i,j) = f(i+1,j) - 2f(i,j) + f(i-1,j) \tag{1.40}$$

$$f_{xx}(i,j) = [1 \quad -2 \quad 1] \tag{1.41}$$

De la même façon on trouve $f_{yy}(i,j)$:

$$f_{yy}(i,j) = f(i,j+1) - 2f(i,j) + f(i,j-1) \tag{1.42}$$

$$f_{yy}(i,j) = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \tag{1.43}$$

Donc:

$$\nabla^2 f(x,y) \rightarrow \widehat{\nabla}^2 f(x,y) = f(i+1,j) - 4f(i,j) + f(i-1,j) + f(i,j+1) + f(i,j-1) \tag{1.44}$$

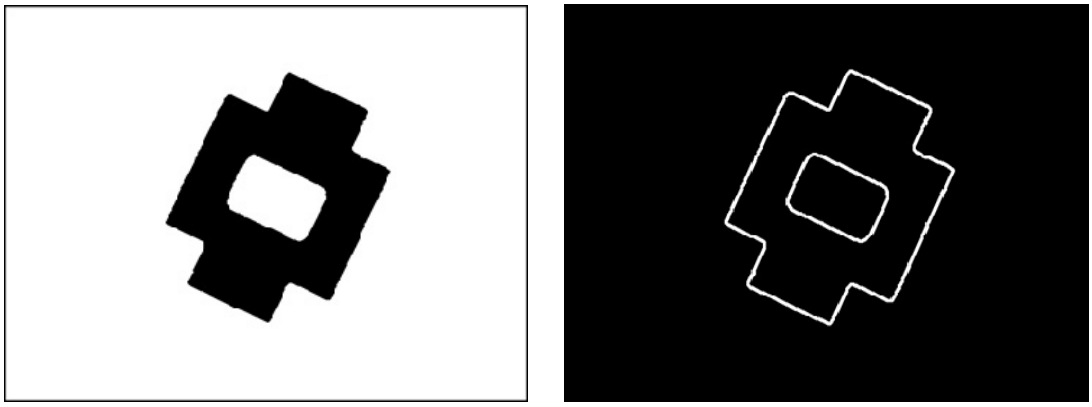
$$\nabla^2 f(x,y) = [1 \quad -2 \quad 1] + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{1.45}$$

D'autres filtres estimatifs peuvent être construits en employant la même méthode, le résultat final dépend de choix de l'approximation de la dérivée première.

Deux autres exemples sont donnés comme suit:

$$H_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad H_3 = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

Le Laplacien possède un inconvénient majeur qu'est sa grande sensibilité au bruit. En effet, cet opérateur réalisé une dérivée seconde de l'image et est donc très instable [2].



a.Image originale. b.Opérateur Laplacien.

Figure 1.15 : détection de contour.

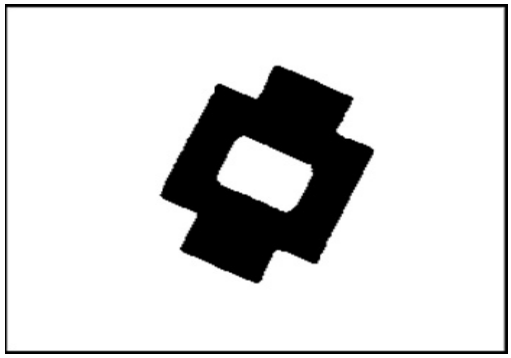
c.Filtrage dans le domaine fréquentiel

Les filtres passe-haut, à l'inverse des passe-bas, atténuent les composantes de basse fréquence de l'image et permettent notamment d'accentuer les détails et le contraste, c'est la raison pour laquelle le terme de "filtre d'accentuation" est parfois utilisé [3][5].

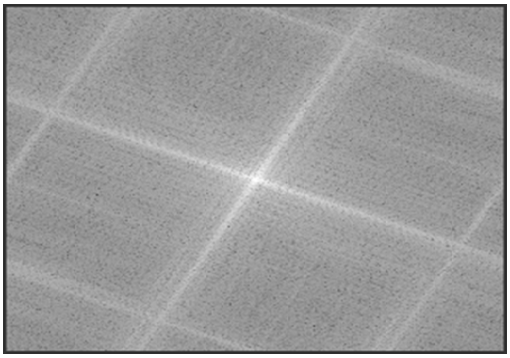
Le filtre passe haut est défini comme suit:

$$\begin{cases} F_{ph}(u,v) = F(u,v) & \text{si } u \geq u_m \text{ et } v \geq v_m \\ F_{ph}(u,v) = 0 & \text{sinon} \end{cases} \tag{1.46}$$

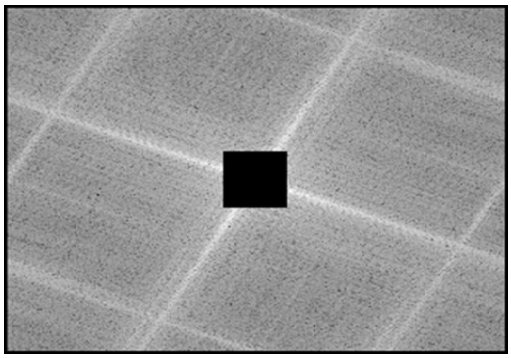
La figure 1.6 illustre l'application de ce filtre sur une image.



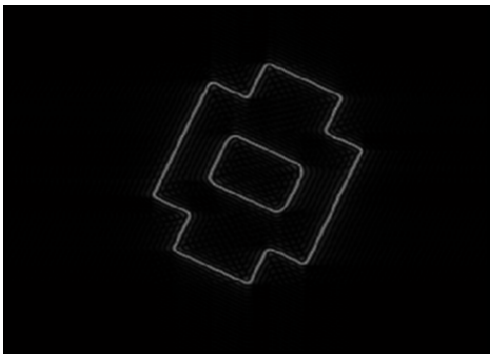
b.Image originale.



c.Spectre d'amplitude.



a.Application de filtre passe-haut.



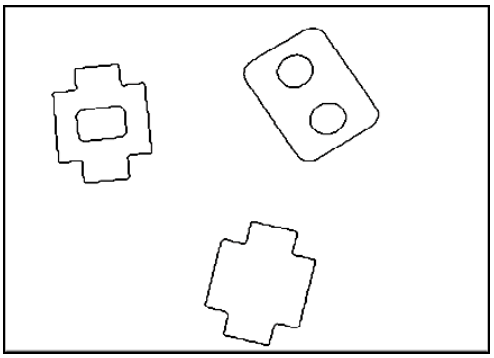
d.Image filtrée.

Figure 1.16 : Filtrage dans le domaine fréquentiel.

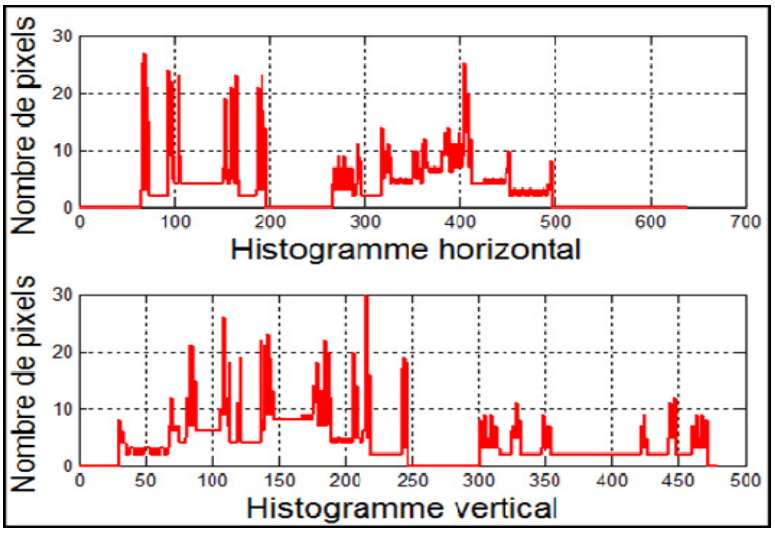
1.4. Séparation d'objets

Pour reconnaitre l'objet dans la scène, il faut pouvoir l'isoler ou l'extraire de la scène, car un classificateur n'est pas capable de reconnaître à la fois un ensemble d'objets dans l'image. La méthode utilisée dans notre cas est basé sur deux étapes, la première repose sur le calcul de l'histogramme horizontal et vertical de l'image (distribution de l'intensité lumineuse suivant les deux axes).

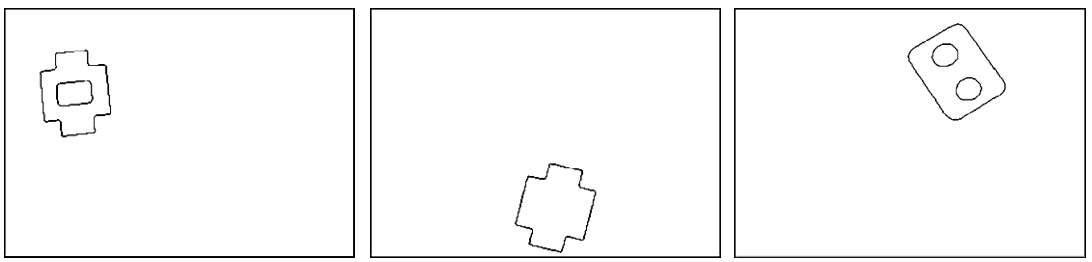
L'histogramme horizontal donne les localisations des objets suivant l'axe vertical et l'histogramme vertical donne les localisations des objets suivant l'axe horizontal (Figure 1.17).



a.Image originale.



b. Histogramme de la distribution de l'intensité lumineuse des objets.

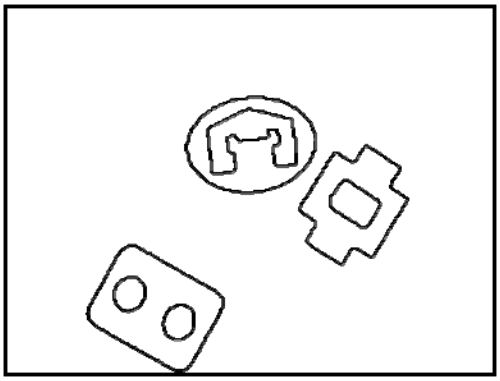


c.Objets séparés.

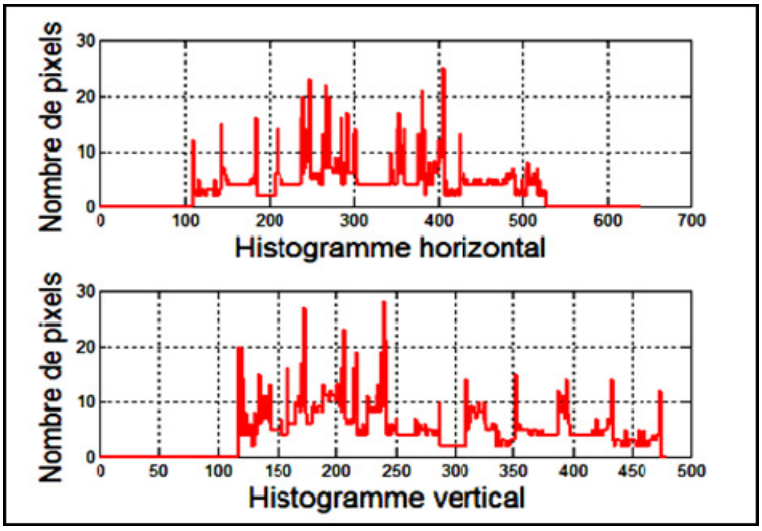
Figure 1.17 : Histogramme et séparation des pièces.

Comme on peut le constater les paquets des pixels non nuls dans l'histogramme indique qu'ils existent des objets dans la scène, le début et la fin de chaque paquet donnent les positions des extrémités d'objets suivant l'axe horizontal dans le cas de l'histogramme vertical et suivant l'axe vertical dans le cas de l'histogramme horizontal. Le nombre d'objets dans la scène est inférieur ou égal au nombre de paquets horizontaux multiplier par le nombre de paquets verticaux.

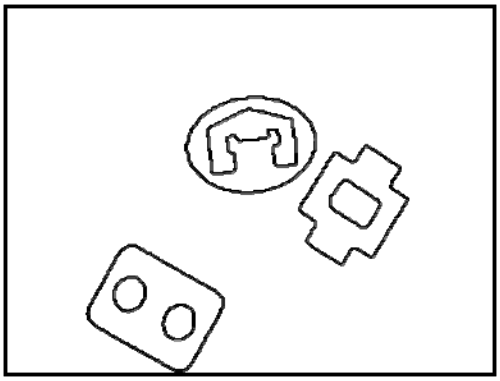
L'inconvénient de cette étape est qu'elle ne peut pas extraire les objets dans le cas ou il n'y a pas des séparations entre ces objets sur les deux axes (horizontal et verticale) (Figure 1.18).



a.Image originale.



b.Histogramme de l'image de la figure (a).

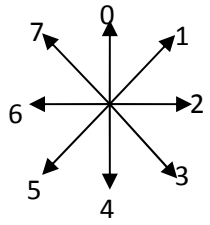


c.Objets non séparés.

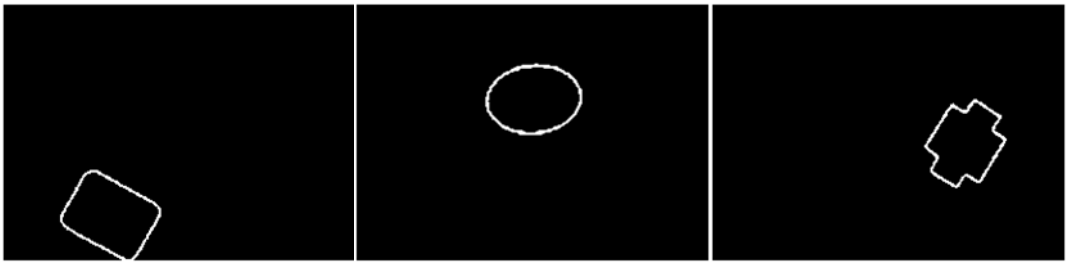
Figure 1.18 : histogramme et séparation des objets.

Une deuxième étape basée sur la segmentation est donc nécessaire pour contourner cet inconvénient. Elle consiste à faire un suivi de contour pour séparer les objets du fond. L'extraction dans ce cas se fait pixel par pixel, en appliquant les

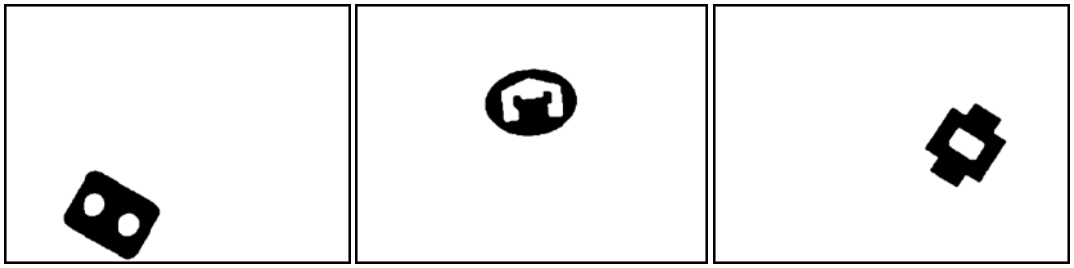
vecteurs des directions pour déterminer le pixel voisin dans la direction des aiguilles d'une montre.



a. Directions de la recherche du contour.



b. Extraction et suivi de contour.



c. Objets séparés.

Figure 1.19 : Suivi de contour et séparation des objets.

1.5. Conclusion

Nous avons décrit dans ce chapitre le rôle de chaque étape dans l'élaboration du processus de la reconnaissance ainsi les premières transformations nécessaires au traitement et à l'analyse des scènes, car les données issues du capteur sont les représentations initiales des données à partir desquelles des traitements permettent de construire celles qui seront utilisées pour la reconnaissance. Les données utiles sont bruitées, elles contiennent des informations parasites, et elles n'explicitent pas les informations utiles pour la reconnaissance, donc les prétraitements consistent à éliminer ces informations parasites et à conserver les informations pertinentes pour la reconnaissance.

CHAPITRE 2

Méthodes Corrélatives Pour La Reconnaissance

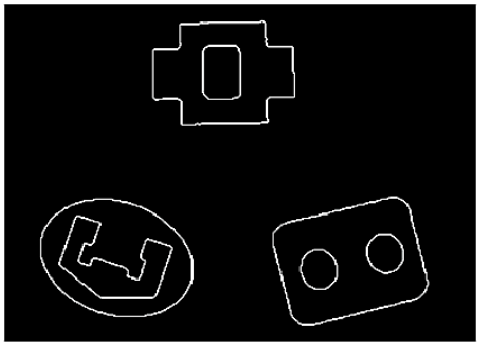
Méthodes Corrélatives Pour La Reconnaissance

2.1. Introduction

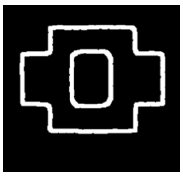
La reconnaissance des objets implique l'indentification correcte d'un objet indépendamment de la rotation, du facteur d'échelle et de la translation. Dans cette partie nous exposons deux méthodes corrélatives pour la reconnaissance la première est basée sur la technique de ***Correspondance de modèle*** (***template matching***), sensible à la rotation, facteur d'échelle et la translation, la deuxième basée sur le calcul du ***rectangle de contenance***, invariant à la rotation, facteur d'échelle et la translation.

2.2. Correspondance de modèle *(*template matching)

Un des moyens les plus fondamentaux de détection d'objet dans un champ d'image est par la correspondance de modèle (template matching), dans lequel une réplique d'un objet d'intérêt appelée aussi patron ou modèle (stockée sous la forme d'une matrice de points noirs et blancs) est comparée à tous les objets inconnue dans le champ de l'image (Figure 2.1). Si la correspondance entre un objet inconnu et le modèle (patron) est suffisamment étroite, l'objet inconnu est désigné comme le modèle (patron) [1].



a.Ensemble d'objets.



b.Modèle (Patron).

Figure 2.1 : Exemple, correspondance de modèle.

Le modèle (patron) parcoure séquentiellement l'image et la région commune entre le modèle et le champ d'image est comparé pour la similitude.

Les résultats obtenus à travers l'opération de prétraitement sont des objets séparés avec des orientations et des positions inconnues, et pour appliquer la méthode de la correspondance de modèle sans parcourir le modèle (patron) sur le champ d'image, les

objets séparés ont été déplacés vers le centre de l'image avec des orientations nules où le patron est comparé avec la forme à reconnaître.

La mesure de la différance entre le modèle et l'objet à reconnaître est l'erreur quadratique moyenne définie par [1]:

$$D(m,n) = \sum_j \sum_k [F(j,k) - T(j-m, k-n)]^2 \quad (2.1)$$

Où $F(j,\ k)$désigne l'image de la recherche et $T(j,k)$ le modèle (patron).

La recherche, bien sûr, est limitée à la région de chevauchement entre le modèle déplacé et le champ de l'image [1], on dit alors qu'une correspondance de modèle existe aux coordonnées (m,n) si :

$$D(m,n) < N(m,n) \quad (2.2)$$

$N(m,n)$: Est un niveau de seuil.

L'équation 2.1 peut être écrire sous la forme:

$$D(m,n) = D_1(m,n) - 2D_2(m,n) + D_3(m,n) \quad (2.3)$$

Où :

$$D_1(m,n) = \sum_j \sum_k [F(j,k)]^2 \quad (2.4)$$

$$D_2(m,n) = \sum_j \sum_k [F(j,k)T(j-m, k-n)] \quad (2.5)$$

$$D_3(m,n) = \sum_j \sum_k [T(j-m, k-n)]^2 \quad (2.6)$$

Le terme $D_3(m,n)$ représente une addition de l'énergie de modèle (patron). Il a une valeur constante et indépendante des coordonnées *(m, n)*. L'énergie d'image sur la région de fenêtre représentée par le premier terme $D_1(m,n)$ varie généralement plutôt lentement sur le champ d'image. Le deuxième terme est reconnu sous le nom de cross corrélation $R_{MI}(m,n)$ entre le champ d'image et le modèle. Sa valeur n'est pas toujours une mesure adéquate parce que l'énergie de l'image $D_1(m,n)$ est variante par rapport à la position [1]. Par exemple, la cross-corrélation peut devenir grande, même dans une

condition de disparité de modèle recherché. Cette difficulté peut être évitée par la comparaison normalisée de la cross-corrélation [1]:

$$\tilde{R}_{MI}(m,n) = \frac{D_2(m,n)}{D_1(m,n)} = \frac{\sum_j \sum_k [F(j,k)T(j-m,k-n)]}{\sum_j \sum_k [F(j,k)]^2} \quad (2.7)$$

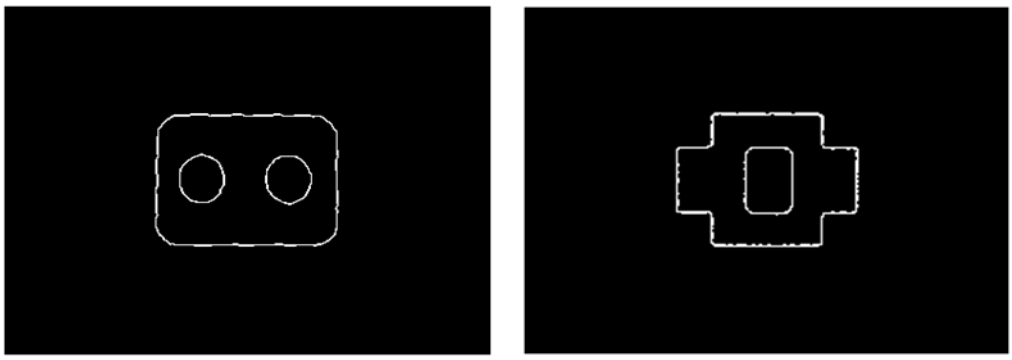
À un niveau de seuil $N_s(m,n)$. On dit qu'une correspondance de modèle existe si:

$$\tilde{R}_{MI}(m,n) > N_s(m,n) \quad (2.8)$$

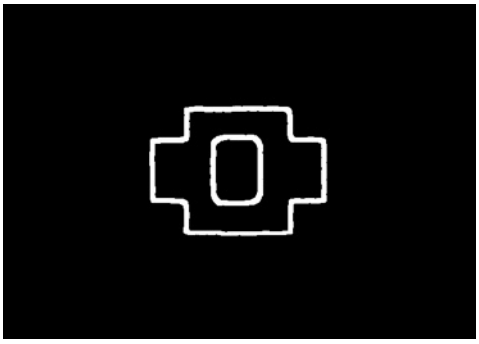
La cross-corrélation normalisée a une valeur maximale d'unité qui se produit si et seulement si la fonction d'image sous le modèle correspond exactement au modèle.

La correspondance de modèle est rarement toujours exacte à cause du bruit d'image et l'incertitude a priori quant à la forme et à la structure exacte d'un objet à détecter, pour cette raison une dilatation est réalisée au niveau du contour du modèle (patron) pour augmenter le taux de correspondance [1].

La figure 2.2 fournit un exemple de la correspondance de modèle pour des images binaires contenant des objets déplacés vers le centre de l'image et pivotés d'un angle θ pour la correspondance avec le modèle.



a.Objets séparés et normalisés à la translation et à la rotation.



b.modèle (patron) pour la comparaison.

Figure 2.2 : Application de la cross-corrélation normalisée.

2.2.1. Position et orientation des objets

Cette phase vise à déterminer les paramètres géométriques des objets en utilisant le concept de ***moments spatiaux***. On détermine pour chaque objet :

1. Les coordonnées $\bar{x}_j$, $\bar{y}_k$, dans le plan image, de son centre de gravité
2. L'angle θ de son axe principal : cet angle détermine l'orientation de l'objet par rapport à l'axe horizontal.

De la théorie de probabilité, le moment de la densité de probabilité commune $p(x, y)$ est défini comme [1]:

$$M(m,n) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^m y^n p(x,y) dxdy \tag{2.8}$$

Le moment central est donné par

$$U(m,n) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (x-\eta_x)^m (y-\eta_y)^n p(x,y) dxdy \tag{2.9}$$

Où η_x et η_y sont les moyens marginaux de $p(x, y)$.

La densité $p(x, y)$ de probabilité commune d'équations (2.8-2.9) est remplacée par la fonction continue d'image $f(x, y)$.

$$M(m,n) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^m y^n f(x,y) dxdy \tag{2.10}$$

$$U(m,n) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (x-\eta_x)^m (y-\eta_y)^n f(x,y) dxdy \tag{2.11}$$

Le concept de moment spatial peut être étendu aux images discrètes en formant des additions spatiales sur une fonction d'image discrète $f(j, k)$ Le moment géométrique spatial est défini comme [13]:

$$M(m,n) = \sum_{j=1}^{J} \sum_{k=1}^{K} (x_j)^m (y_k)^n f(j,k) \tag{2.12}$$

Le moment d'ordre zéro est donné par:

$$M(0,0) = \sum_{j=1}^{J} \sum_{k=1}^{K} f(j,k) \tag{2.13}$$

Les moments d'ordre un sont donnés par:

$$M(1,0) = \sum_{j=1}^{J} \sum_{k=1}^{K} x_j f(j,k) \tag{2.14}$$

$$M(0,1) = \sum_{j=1}^{J} \sum_{k=1}^{K} y_k f(j,k) \tag{2.15}$$

La position, représentée par le centre de gravité, s'exprime à partir des relations 2.13, 2.14 et 2.15 [1][13].

$$\bar{x}_j = \frac{M(1,0)}{M(0,0)} \tag{2.16}$$

$$\bar{y}_k = \frac{M(0,1)}{M(0,0)} \tag{2.17}$$

Les moments d'ordre inférieur définissent, donc le centre de la surface d'image appelé aussi le centre de gravité, c'est le point d'équilibre de la fonction d'image $f(j,\ k)$ tel que la masse de $f(j,\ k)$ au-dessus et dessous de $\bar{x}_j$ et à gauche et à adroite de $\bar{y}_k$ est égal [1].

- ***Surface d'objet***

La surface de l'objet est définie par tous les points situés à l'intérieur de contour (Figure 2.3).

Figure 2.3 : Image obtenue par l'étape de séparation des objets.

Les étapes suivis pour déterminer la surface d'objet sont les suivantes:

1. Tous les pixels situés entre deux points verticaux de contour prennent la même valeur que celui-ci, le résultat de cette opération est l'image suivante:

Figure 2.4 : Surface obtenue à partir des pixels verticaux.

2. Idem pour les pixels horizontaux situés entre deux points de contour, le résultat de cette opération est l'image suivante:

Figure 2.5 : Surface obtenue à partir des pixels horizontaux.

3. L'application d'un (***et***) logique entre les deux images précédentes donne la surface de l'objet.

Figure 2.6 : Surface de l'objet.

Dès équations 2.16 et 2.17 le centre de la surface d'objet est indiqué sur la figure 2.7:

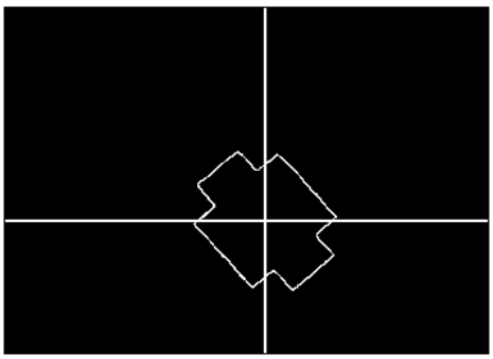


Figure 2.7 : Centre de l'objet.

Le moment central dans le cas discret est donné par :

$$U(m,n) = \sum_{j=1}^{J}\sum_{k=1}^{K}(x_j - \bar{x}_j)^m(y_k - \bar{y}_k)^n f(x,y) \tag{2.18}$$

Les trois moments centraux de second ordre sont :

- Le moment d'inertie de la ligne

$$U(2,0) = \sum_{j=1}^{J}\sum_{k=1}^{K}(x_j - \bar{x}_j)^2 f(x,y) \tag{2.19}$$

- Le moment d'inertie de la colonne

$$U(0,2) = \sum_{j=1}^{J}\sum_{k=1}^{K}(y_k - \bar{y}_k)^2 f(x,y) \tag{2.20}$$

- Le moment d'inertie ligne-colonne

$$U(1,1) = \sum_{j=1}^{J}\sum_{k=1}^{K}(x_j - \bar{x}_j)(y_k - \bar{y}_k) f(x,y) \tag{2.21}$$

Les trois moments d'inertie du deuxième ordre définis par les équations 2.19, 2.20 et 2.21 peuvent être utilisés pour créer la matrice de covariance d'inertie [1].

$$U = \begin{bmatrix} U(2,0) & U(1,1) \\ U(1,1) & U(0,2) \end{bmatrix} \tag{2.22}$$

L'exécution d'une décomposition en valeurs singulières de la matrice de covariance a comme conséquence une matrice diagonale [1]:

$$E^T U E = \Lambda \tag{2.23}$$

Où les colonnes de

$$E = \begin{bmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{bmatrix} \tag{2.24}$$

Sont les vecteurs propres de U et

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{2.25}$$

Contient les valeurs propres de U. Pour que les colonnes de E soient des vecteurs propres de la matrice U il faut que soient vérifiées les égalités suivantes :

$$UX = \lambda X \tag{2.26}$$

Ou

$$UX = \lambda IX \tag{2.27}$$

Autrement dit :

$$(U - \lambda I)X = 0 \tag{2.28}$$

Où λ est un nombre appelé valeur propre, le vecteur X est appelé vecteur propre de la matrice U.

Sous forme explicite l'égalité (2.26) s'écrit :

$$\begin{cases} U(2,0)x_1 + U(1,1)x_2 = \lambda x_1 \\ U(1,1)x_1 + U(0,2)x_2 = \lambda x_2 \end{cases} \tag{2.29}$$

Et l'égalité (2.28)

$$\begin{cases}(U(2,0)-\lambda)x_1+U(1,1)x_2=0\\ U(1,1)x_1+(U(0,2)-\lambda)x_2=0\end{cases} \quad (2.30)$$

Pour que le système (2.30) possède une solution non nulle, il faut et il suffit que le déterminant du système soit nul :

$$det\left(\begin{bmatrix}(U(2,0)-\lambda) & U(1,1)\\ U(1,1) & (U(0,2)-\lambda)\end{bmatrix}\right)=0 \quad (2.31)$$

Ou

$$det(U-\lambda I)=0 \quad (2.32)$$

On l'appelle équation caractéristique de la matrice U. Elle permet de calculer les valeurs propres λ.

$$\lambda^2-[U(2,0)+U(0,2)]\lambda+U(2,0)U(0,2)-U(1,1)^2=0 \quad (2.33)$$

Le discriminant est donné par :

$$\Delta=U(2,0)^2+U(0,2)^2-2\,U(2,0)U(0,2)+4U(1,1)^2 \quad (2.34)$$

C'est évident que :

$$[U(2,0)-U(0,2)]^2\geq 0 \quad (2.35)$$

Donc :

$$U(2,0)^2+U(0,2)^2-2\,U(2,0)U(0,2)\geq 0 \quad (2.36)$$

Aussi :

$$U(2,0)^2+U(0,2)^2-2\,U(2,0)U(0,2)+4U(1,1)^2\geq 0 \quad (2.37)$$

Donc le discriminant peut prendre deux cas :

1. $\Delta=0$ (2.38)

On trouve $\Delta=0$ dans le cas où le moment d'inertie de la ligne égale le moment d'inertie de la colonne et le moment d'inertie ligne-colonne égale à zéro (les deux axes x et y sont des axes de symétrie).

Donc lorsque $\Delta = 0$ l'orientation de l'objet égale à zéro.

2. $\Delta > 0$ (2.39)

On trouve :

$$\lambda_1 = \frac{1}{2}[U(2,0) + U(0,2)] + \frac{1}{2}[U(2,0)^2 + U(0,2)^2 - 2U(2,0)U(0,2) + 4U(1,1)^2]^{1/2} \quad (2.40)$$

$$\lambda_2 = \frac{1}{2}[U(2,0) + U(0,2)] - \frac{1}{2}[U(2,0)^2 + U(0,2)^2 - 2U(2,0)U(0,2) + 4U(1,1)^2]^{1/2} \quad (2.41)$$

Trouvons le vecteur propre correspondant à la valeur propre λ_1 à partir du système correspondant d'équations (2.30). Résolvant ce système nous trouvons :

$$\begin{cases} x_1 = m1 \\ x_2 = \dfrac{-U(1,1)m1}{[U(0,2) - \lambda_1]} \end{cases} \quad (2.42)$$

La même chose pour λ_2 on trouve :

$$\begin{cases} x_3 = m2 \\ x_4 = \dfrac{-U(1,1)m2}{[U(0,2) - \lambda_2]} \end{cases} \quad (2.43)$$

Où m1 et m2 sont des nombres arbitraires.

La matrice E sera donc :

$$E = \begin{bmatrix} m1 & m2 \\ \dfrac{-U(1,1)m1}{[U(0,2) - \lambda_1]} & \dfrac{-U(1,1)m2}{[U(0,2) - \lambda_2]} \end{bmatrix} \quad (2.44)$$

L'ensemble, $\lambda_M = \text{MAX}\{\lambda_1, \lambda_2\}$ et l'orientation θ sont définis comme :

$$\theta = \begin{cases} arctan\left\{\dfrac{e_{11}}{e_{21}}\right\}, & si\ \lambda_M = \lambda_1 \\ arctan\left\{\dfrac{e_{21}}{e_{22}}\right\}, & si\ \lambda_M = \lambda_2 \end{cases} \quad (2.45)$$

L'angle θ peut être exprimé explicitement :

$$\theta = arctan\left\{\frac{\lambda_M - U(0,2)}{U(1,1)}\right\} \quad (2.46)$$

2.2.2. Modification géométrique d'image

Une des transformations d'image les plus connues est la modification géométrique dans laquelle l'image est spatialement déplacée, tournée, où affichée d'une perspective différente [1].

Les transformations géométriques d'image sont habituellement basées sur une représentation de système de coordonnées cartésiennes dans lequel les Pixels sont de dimension unité, et l'origine $(0,0)$ est au centre du Pixel supérieur du coin gauche (Figure 2.8).

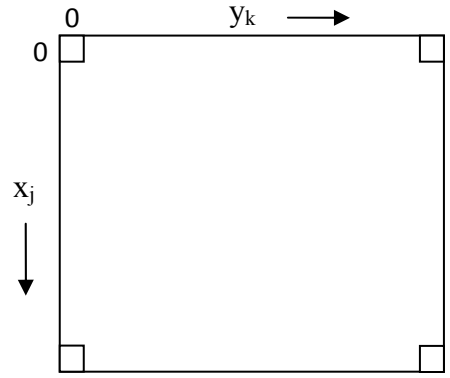


Figure 2.8 : Représentation d'une image discrète.

a. Translation

Le déplacement de l'objet $f(j,\ k)$ vers le centre de l'image pour produire l'objet $g(j,k)$ implique le calcul des positions de compensation relatives des deux images (Figure 2.9). Les relations des positions de déplacement sont:

$$x_j = \grave{x}_j + t_x \quad (2.47)$$

$$y_k = \grave{y}_k + t_y \quad (2.48)$$

Où t_x et t_y sont les valeurs des déplacements suivant les deux axes x et y respectivement.

$$t_x = X_{Centre(Im)} - X_{Centre(Ob)} \quad (2.49)$$

$$t_y = Y_{Centre(Im)} - Y_{Centre(Ob)} \quad (2.50)$$

Avec :

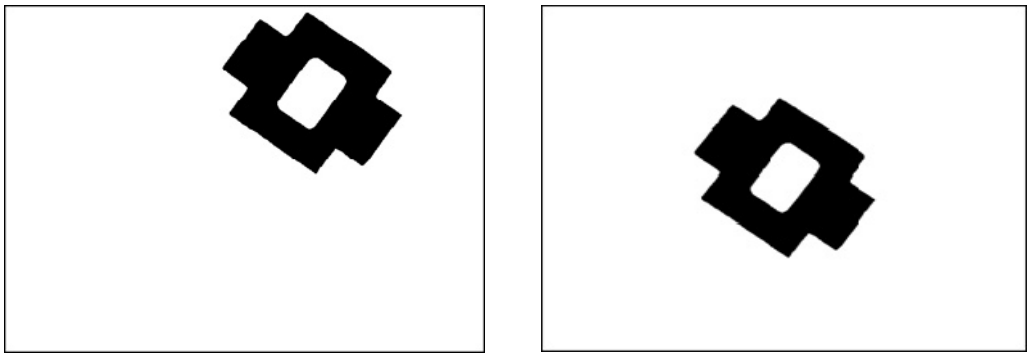
$X_{Centre(Im)}$: Le centre d'image suivant x, dans notre cas il vaut 240.

$Y_{Centre(Im)}$: Le centre d'image suivant y, dans notre cas il vaut 320.

$X_{Centre(Ob)}$: Le centre de gravité de l'objet suivant l'axe x.

$Y_{Centre(Ob)}$: Le centre de gravité de l'objet suivant l'axe y

$\dot{x}_j, \dot{y}_k$ sont les pixels source, représentent les positions des objets dans le champ de vision.



a.Image originale. b.objet déplacé vers le centre de l'image.

Figure 2.9 : Annulation de la translation.

b. Rotation

La rotation de l'objet est une translation suivant l'axe horizontal et l'axe vertical. Généralement ces translations sont des valeurs réelle c'est-à-dire un point de l'objet est déplacé vers un autre point situé entre deux pixels, comme ces translations n'existent qu'à des positions entières, il est nécessaire d'interpoler l'image (faire des approximations pour les nouvelles positions) [1][2].

Ces approximations introduit dans l'image des trous et pour les éliminer il suffit d'appliquer le filtre médian (Figure 2.10).

L'annulation de la rotation d'un objet peut être accomplir par le calcul des positions suivantes.

$$x_j = x_p\, cos(-\theta) + y_p\, sin(-\theta) + X_{Centre(Ob)} \quad (2.51)$$

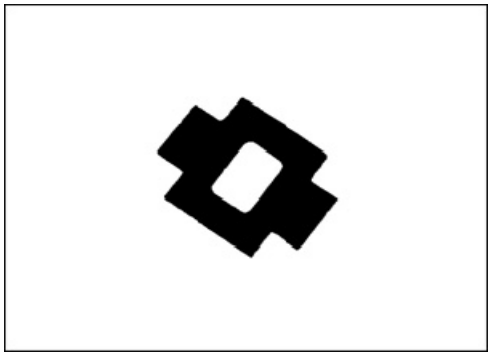
$$y_j = -x_p\, sin(-\theta) + y_p\, cos(-\theta) + Y_{Centre(Ob)} \quad (2.52)$$

Où θ est l'angle de rotation dans le sens des aiguilles d'une montre par rapport à l'axe horizontal de l'image source, x_p et y_p sont les cordonnés de l'objet par rapport au centre de gravité.

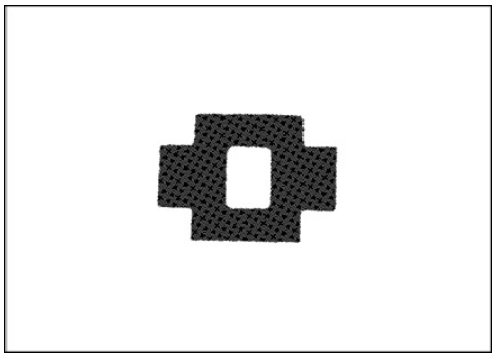
$$x_p = \dot{x_j} - X_{Centre(Ob)} \tag{2.53}$$

$$y_p = \dot{y_k} - Y_{Centre(Ob)} \tag{2.54}$$

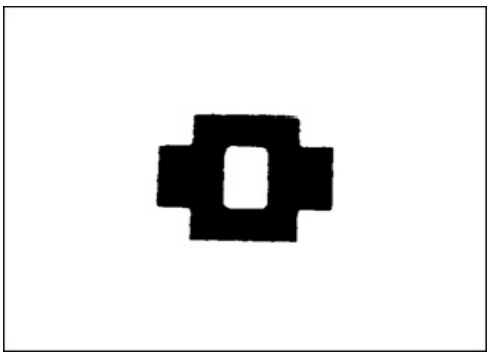
La rotation de l'objet par rapport à un point pivot (centre de gravité dans notre cas) peut être accomplir par la transformation de l'origine de l'image au point de pivot, effectuant la rotation, et puis retransformer l'origine à sa potion initiale équations (2.51-2.52).



a.Image originale.



b.Image avec une orientation nulle.



c.Image filtrée.

Figure 2.9 : Annulation de l'orientation.

c. Dilatation

Elle consiste à dilater le contour du modèle (patron) pour augmenter le taux de correspondance entre celui-ci et l'objet à identifier, de ce fait les points noirs isolé au milieu des parties blanches sont éliminées par la dilatation des parties blanches. Le concept est tout à fait simple: un petit masque impair de taille 3x3 est balayé au-dessus de l'image dont le but est de faire l'extraction des pixels de l'image. Le pixel central *X*

et ses huit voisins sont stockés dans une pile de pixels de voisinage, si ceux-ci remplissent certaine 'condition' prédéterminée le pixel central du masque changerait l'état de 0 à un 1 état logique (Figure 2.10) [1].

Le pixel central X prend l'état logique 1 si la condition suivante est remplie [1].

$$G(j,k) = X \cup (X_0 \cup X_1 \cup X_2 \cup X_3 \cup X_4 \cup X_5 \cup X_6 \cup X_7) \tag{2.55}$$

Les pixels extraient de l'image sont donnés comme suit:

X_3	X_2	X_1
X_4	X	X_0
X_5	X_6	X_7

Figure 2.10: Les pixels extraits de l'image.

2.2.3. Détermination du seuil d'acquisition

Dans notre application l'acquisition se fait dans des conditions naturelles d'éclairage et ça peut influencer sur le résultat final de la reconnaissance, car la taille de l'objet change en fonction de la lumière. Pour résoudre ce problème un objet appelé modèle est sauvegardé dans la mémoire du système après un prétraitement de normalisation dans le but de la comparaison, le réglage se fait ici, en utilise la méthode de cross-corrélation entre l'objet (modèle) stocké dans la mémoire et le même objet mais capté par la caméra, si la correspondance est inférieur à un certain taux, le seuil T de binarisation est augmenté d'une unité en commençant par une valeur minimale, jusqu'à ce que la correspondance atteint ou dépassé le taux de correspondance choisi (Figure 2.11).

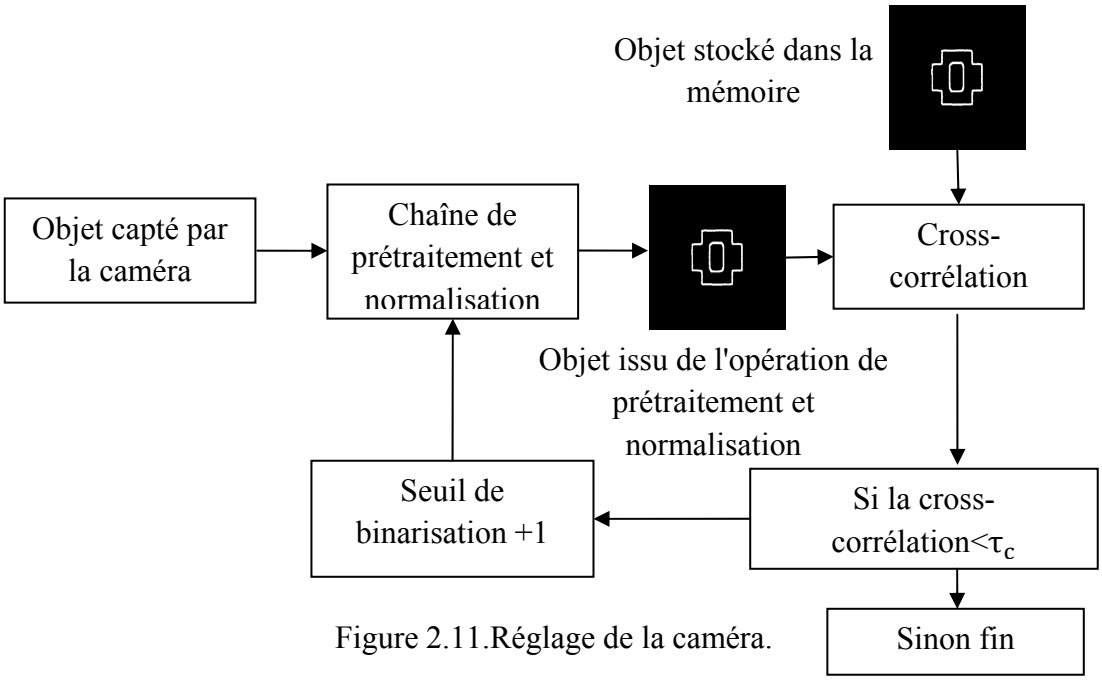


Figure 2.11.Réglage de la caméra.

2.3. Rectangle de contenance

Le rectangle de contenance est le plus petit rectangle contenant l'objet qui est aussi aligné avec son orientation (Figure 2.12) [1].

Pour évité le croisement entre le champ de vision et le rectangle de contenance l'objet a été déplacé vers le centre de l'image.

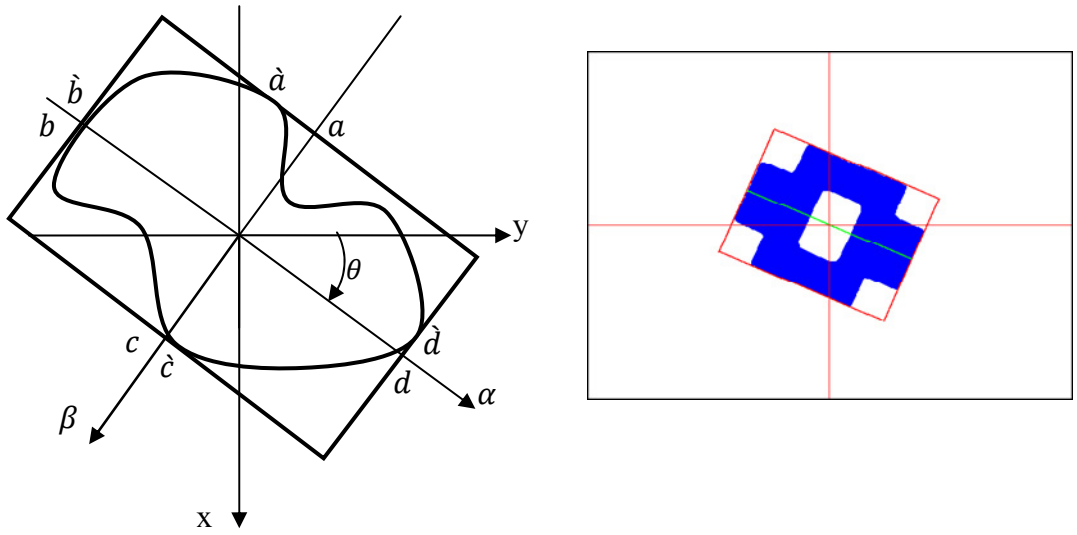


Figure 2.12.Rectangle de contenance

Pour déterminer ce rectangle on suit ces étapes:

1. D'abord il faut déterminer l'angle d'orientation θ de l'objet comme on l'a vu précédemment.

2. Une fois θ déterminé on utilise la transformation:

$$\alpha = x_p \ sin\,\theta + y_p \cos\theta \tag{2.56}$$

$$\beta = x_p \ \cos\theta - y_p \ \sin\theta \tag{2.57}$$

Pour les points de la frontière de l'objet, on cherche : α_{max}, α_{min}, β_{max} , β_{min}. Ces points correspondant respectivement aux points $\grave{d}, \grave{b}, \grave{c}$, et à a partir de ça, le rectangle de contenance est immédiatement connu avec sa :

Longueur

$$A = \alpha_{max} - \alpha_{min} \tag{2.57}$$

Largeur

$$B = \beta_{max} - \beta_{min} \tag{2.58}$$

Rapport

$$C = \frac{surface\ objet}{A.B} \tag{2.59}$$

Le rectangle de contenance nous permet d'identifier l'objet (le rapport surface de l'objet sur surface de rectangle) et de déterminer sa longueur et sa largeur.

2.4. Conclusion

Dans ce chapitre nous avons présenté deux méthodes corrélatives de reconnaissance : méthode de la cross-corrélation et méthode du rectangle de contenance. Pour améliorer le temps du calcul et le taux de reconnaissance, des normalisations sont opérées. Il s'agit de : normalisation de l'orientation, normalisation de la translation et la dilatation de modèle.

CHAPITRE 3

Méthodes Vectorielles Pour La Reconnaissance

Méthodes Vectorielles Pour La Reconnaissance

3.1. Introduction

Un des problèmes centraux en identification d'objets concerne le choix d'une représentation pertinente permettant d'accéder à des primitives significatives et fiables traduisant la classe de l'objet. La résolution de ce problème exige une méthode d'extraction de primitives qui peuvent générer des caractéristiques distinctes pour chaque objet, par conséquent, cette partie décrit l'analyse de deux méthodes d'extractions de primitives géométrique à savoir, les moments de Fourier-Mellin et les moments de Zernike en fonction de leur possibilité pour identifier des objets.

3.2. Moments de Fourier-Mellin

La transformée de Fourier-Mellin (TFM) est la combinaison de la représentation par coefficients de Fourier d'une fonction périodique (de la variable angulaire θ) et de la transformée de Mellin (de la variable radiale r).

$$\forall(v,k) \in \mathbb{R}X\mathbb{Z}, \qquad M_f(v,k) = \frac{1}{2\pi}\int_0^{+\infty}\int_0^{2\pi} f(r,\theta)e^{-ik\theta}r^{-iv}\,d\theta\,\frac{dr}{r} \tag{3.1}$$

La fonction $f(r,\theta)$ peut être reconstruite à partir de la TFM, en utilisant la relation suivante :

$$\forall(r,\theta) \in \mathbb{R}_+^* X\, \mathbb{S}^1, \qquad f(r,\theta) = \int_{-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} M_f(v,k)e^{ik\theta}r^{iv}dv \tag{3.2}$$

La transformée de Fourier-Mellin d'un objet existe si sa représentation f est intégrable sur $R^*_+ \times S^1$, ce qui se traduit par :

$$\int_0^{+\infty}\int_0^{2\pi}\left|f(r,\theta)d\theta\,\frac{dr}{r}\right| < \infty \tag{3.3}$$

La fonction $f(r,\theta)$ n'est pas intégrable au voisinage de zéro dans la majorité des cas. Dans la littérature, deux solutions à ce problème ont été présentées. Une première consiste à annuler un disque suffisamment petit au voisinage de zéro [7], ce qui conduit généralement à une perte considérable d'information. Une deuxième solution

introduite par **GHORBEL** [7] consiste à modifier la fonction f en introduisant le terme r^{σ} pour rendre la fonction intégrable au voisinage de zéro.

$$f_{\sigma}(r,\theta) = r^{\sigma} f(r,\theta) \tag{3.4}$$

Où σ est un nombre réel non nul et strictement positif

La transformée de cette nouvelle fonction f_{σ} est appelée prolongement analytique de Fourier-Mellin (PATFM) de f et elle s'écrit :

$$\forall (v,k) \in \mathbb{R}X\mathbb{Z}, \qquad M_{f\sigma}(v,k) = \frac{1}{2\pi}\int_0^{+\infty}\int_0^{2\pi} f(r,\theta) e^{-ik\theta} r^{\sigma - iv}\, d\theta \frac{dr}{r} \tag{3.5}$$

Avec : $k \in \mathbb{Z}, \theta \in \mathbb{R}, et\ \sigma \in \mathbb{R}_+^*$

La transformée inverse de la TFM de f_{σ} existe. Ainsi l'inverse du PATFM de f existe et s'écrit :

$$\forall (r,\theta) \in \mathbb{R}_+^* X\, \mathbb{S}^1, \qquad f(r,\theta) = r^{-\sigma}\int_{-\infty}^{+\infty}\sum_{k=-\infty}^{+\infty} M_{f\sigma}(v,k) e^{ik\theta} r^{iv} dv \tag{3.6}$$

3.2.1. Invariants issus de la PATFM

La construction de descripteurs de formes invariants à la rotation s'appuie sur l'application du théorème du retard, que l'on retrouve dans toute l'analyse de Fourier. Soit $g(r,\theta) = f(\alpha r, \theta + \beta)$ une forme modifiée en orientation et en échelle de f(r, θ), alors on a la relation suivante:

$$M_{g\sigma}(v,k) = \alpha^{-\sigma + iv} e^{ik\beta} M_{f\sigma}(v,k) \tag{3.7}$$

En prenant le module des termes de l'équation (3.7), il est possible de déduire des descripteurs qui sont invariants aux changements par rapport à la rotation de la forme mais pas aux changements d'échelle.

GHORBEL a proposé un ensemble de primitives invariantes aux changements d'échelle et d'orientation; celui-ci repose sur l'expression suivante:

$$I_{f\sigma}(v,k) = M_{f\sigma}(v,k)\left[M_{f\sigma}(0,1)\right]^{-k}\left|M_{f\sigma}(0,1)\right|^{k}\left[M_{f\sigma}(0,0)\right]^{-1+\frac{iv}{\sigma}} \tag{3.8}$$

Avec cette définition, on peut montrer que si $g(r,\theta) = f(\alpha r, \theta + \beta)$, alors $\mathrm{I}_{\mathrm{f}\sigma}(\mathrm{v},\mathrm{k}) = \mathrm{I}_{\mathrm{g}\sigma}(\mathrm{v},\mathrm{k})$.

3.2.2. Application aux images numériques

Plusieurs problèmes sont soulevés lors de la mise en œuvre de la PATFM sur des images numériques. Le premier provient de la nécessité d'échantillonner la quantité $\mathrm{M}_{\mathrm{f}\sigma}(\mathrm{v},\mathrm{k})$. Une solution simple consiste à échantillonner la variable v et à calculer $\mathrm{M}_{\mathrm{f}\sigma}(\mathrm{p},\mathrm{k})$ et $\mathrm{I}_{\mathrm{f}\sigma}(\mathrm{p},\mathrm{k})$ pour tout $\mathrm{v} = \mathrm{p} \in \mathbb{Z}$ [9].

Le second problème réside dans le choix nécessaire d'un point invariant et représentatif de la forme comme Centre de Développement (CdD) de la TFM. Un choix a priori raisonnable consiste à appliquer la TFM au centre de gravité de la forme. Le calcul des coordonnées du centre de gravité est facile et permet d'assurer que la forme est incluse dans un rayon donné, argument qui sera d'un grand intérêt dans la suite.

La troisième et principale difficulté est due à la structure bidimensionnelle, cartésienne et discrète de l'image numérique, sur laquelle on veut appliquer une transformation définie dans un repère polaire. Pour résoudre ce problème, deux options sont disponibles : l'interpolation de l'image en coordonnées polaires [8][9] ; ou bien la convolution de l'image en coordonnées cartésiennes, au CdD d'une forme, avec un banc de filtres appropriés [10].

Avec la première option, la TFM est obtenue grâce à la relation (3.5) appliquée au CdD de la forme considérée.

Cela suppose donc que l'image soit d'abord interpolée en coordonnées polaires pour couvrir la gamme entière de $\mathrm{r} \in [0, \mathrm{r}_{\max}]$ et $\theta \in [0, 2\pi]$ ($\mathrm{r}_{\max}$ étant le rayon maximal de la forme). Cette technique entraîne un coût de calcul qui peut être important, et peut aussi introduire de l'information redondante dans les données originales.

Avec la deuxième option, il n'est pas nécessaire d'interpoler la forme à partir de l'image. En effet, la version discrète de la TFM, calculée au CdD (m_0, n_0) peut être approchée de la façon suivante:

$$M_{f\sigma}(p,k). \approx \sum_{m}\sum_{n} h_{p,k}(m,n) f(m - m_0, n - n_0) \qquad (3.9)$$

$$1 \leq (m^2 + n^2) \leq \mathrm{r}^2{}_{\max}$$

Où:

$$h_{p,k}(m,n) = \frac{e^{-i\left(\frac{p}{2}\ln\left(m^2+n^2\right)+k\tan^{-1}\left(\frac{n}{m}\right)\right)}}{(m^2+n^2)^{1-\frac{\sigma}{2}}} \tag{3.10}$$

Cette formulation montre que la convolution, au centre de gravité, de la forme avec le filtre de réponse impulsionnelle $h_{p,k}(m,n)$ donne une approximation du coefficient d'ordre (p,k) de la PATFM. On peut remarquer que tous les filtres, sauf $h_{p,k}(0,0)$, sont à coefficients complexes. D'autre part, ces filtres sont orthogonaux par rapport à la variable circulaire k, mais pas par rapport à la variable radiale p équations (3.5, 3.10).

Une fois calculée la PATFM d'une forme donnée par le banc de filtre appliqué à son centre de gravité, on extrait des descripteurs invariants grâce à la relation (3.8)

Dans le cadre de notre application, du fait de sa simplicité, nous avons choisi la seconde option.

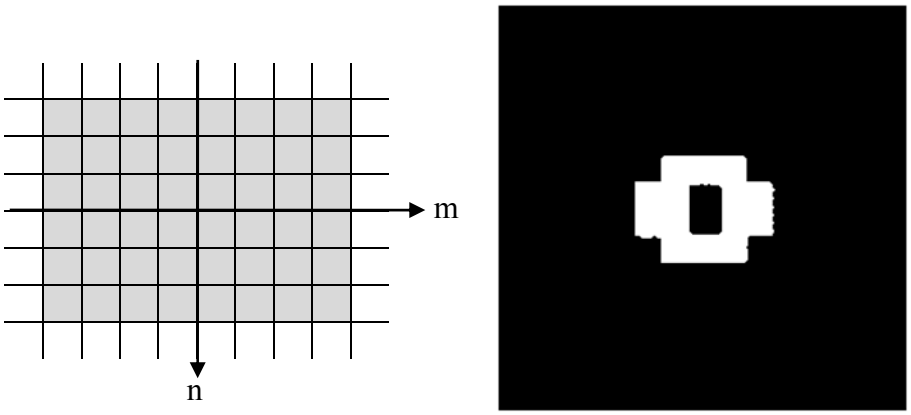


Figure 3.1 : Grille cartésienne et objet image utilisée pour illustrer le spectre du PATFM.

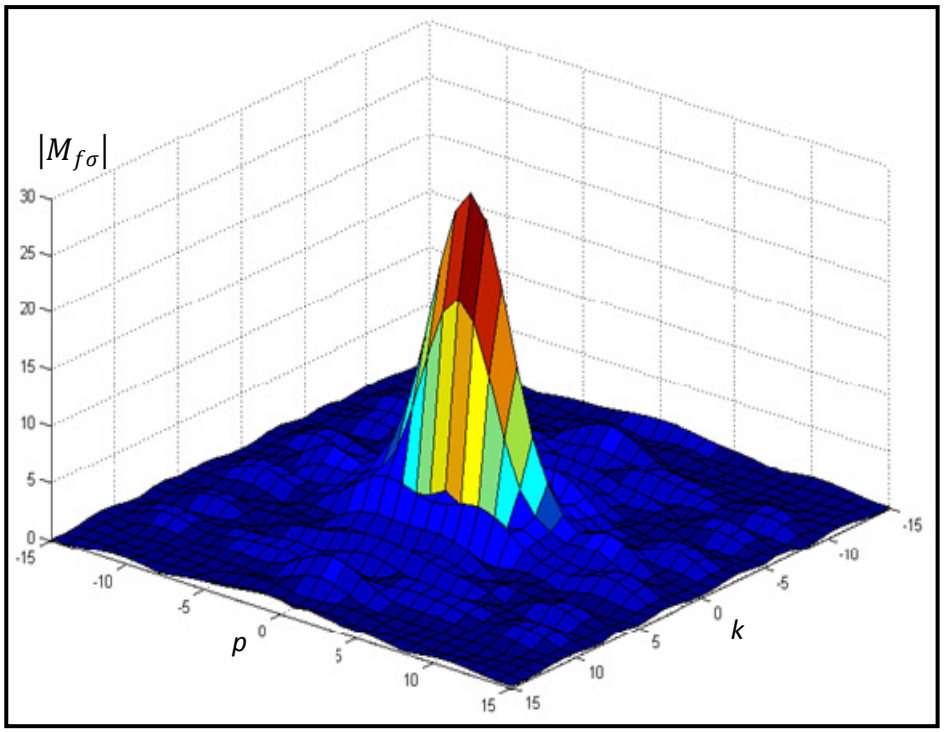


Figure 3.2 : Spectre du PATFM Cartésienne (p=k=15,$\sigma = 0.5, r = 95$).

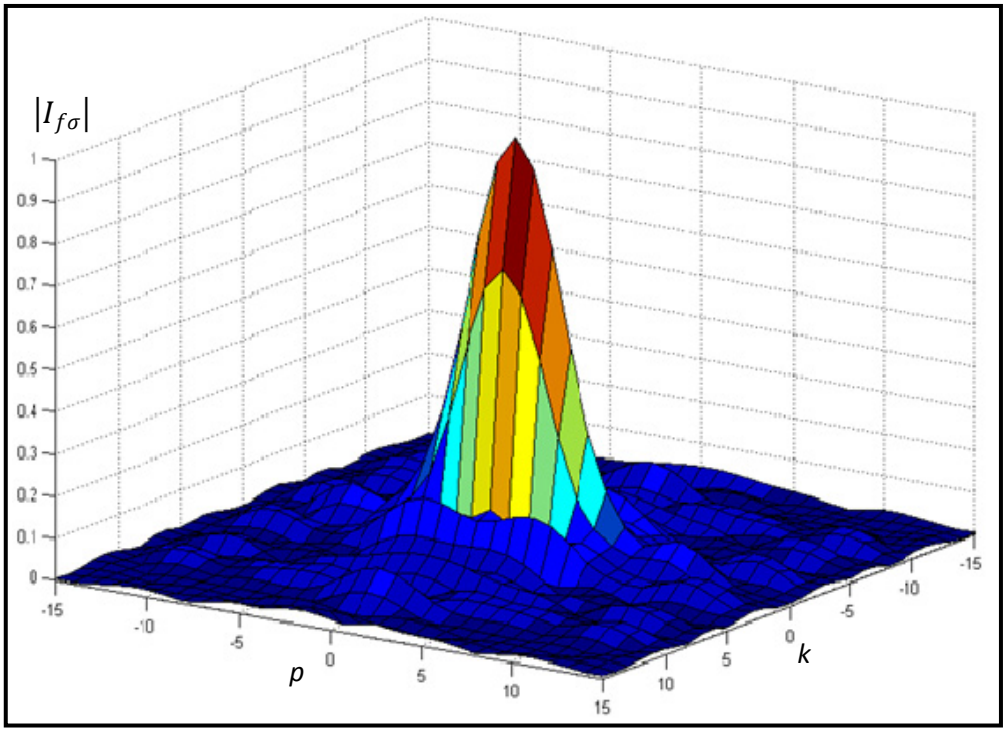


Figure 3.3 : Spectre du PATFM Cartésienne invariantes aux changements d'échelle (p=k=15,$\sigma = 0.5, r = 95$).

Pour améliorer le temps de calcul de ces primitives invariantes, il faut réduire la taille de l'image sans faire une normalisation d'échelle. La notion de pyramide peut être une solution pour ce problème.

3.2.3. Représentation par pyramide

Une pyramide P est une suite d'images I_h, $h \in [0, n]$. h est appelé niveau et $n+1$ est le nombre de niveaux ou hauteur de la pyramide. Dans le cas d'une structure arborescente quaternaire, la taille de l'image I_h et son niveau sont liés par la relation : $X_h = Y_h = [0, 2^{n-h} - 1]$. Un élément de la pyramide sera donc un triplet (j, k, h) où (j, k) est un élément de l'image I_h.

L'image I_n est appelée apex ou sommet de la pyramide. Cette image est constituée d'un seul élément. L'image I_o de dimensions 2^nx2^nest appelée base de la pyramide. Le plus souvent, cette image est l'image initiale avant traitement (Figure 3.4).

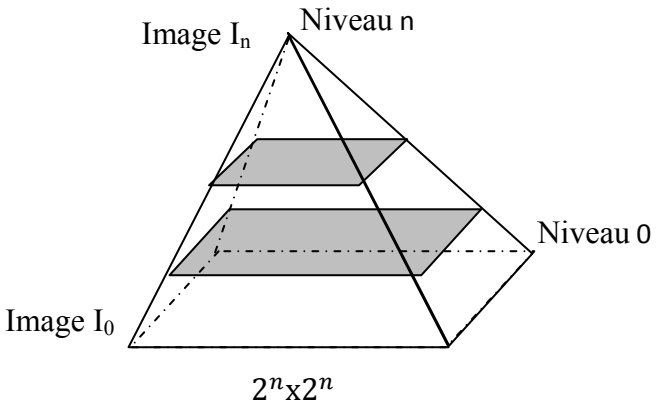


Figure 3.4 : Pyramide de l'image.

Les niveaux intermédiaires de l'image, sont crées à partir du bas vers le haut. Le pixel (j, k, h) au niveau h est crée à partir de ses quatre voisins du niveau $h-1$, en utilisant l'expression suivante [2]:

$$I_h(j,k) = g\begin{bmatrix} I_{h-1}(2j,2k) & I_{h-1}(2j,2k+1) \\ I_{h-1}(2j+1,2k) & I_{h-1}(2j+1,2k+1) \end{bmatrix} \quad (3.11)$$

Avec : $j,\ k = 0,1,\ldots,2^{n-h} - 1$.

Où $g[.]$ est une fonction de correspondance.

La pyramide peut être construite à partir d'image binaire. La fonction de correspondance peut être un OU logique dans le cas où l'information pertinente est représentée par un objet blanc, dans le cas inverse on applique un ET logique.

Pour réduire la taille de l'image, il faut suivre ces étapes:

1. Ajouter des pixels à l'image originale (480x640 pixels) pour construire la base de la pyramide, dans notre cas c'est une image de 1024x1024 pixels.

2. Appliquer la notion de la pyramide jusqu'à le niveau correspond à la résolution 256x256 pixels.
3. A partir de centre de l'image réduite on pend une fenêtre contenant l'objet de résolution 200x200 pixels.

Figure 3.5 : Image réduite 200x200.

3.2.4. Effet de la rotation

Afin d'étudier les effets de la rotation sur les invariants de $I_{f\sigma}(p,k)$, nous avons réalisé la rotation d'un objet par rapport à son centre de gravité par pas de 10°. Quelques exemples de ces images sont donnés à la figure 3.6. Nous avons calculé 6 invariants pour chaque image puis nous avons calculé la moyenne de l'erreur relative. L'erreur relative est calculée de la manière suivante :

$$\varepsilon = \frac{|x - x_0|}{x_0} \tag{3.12}$$

Où x est la valeur d'un invariant pour une image ayant subie une rotation et x_0 est la valeur calculée pour l'image originale.

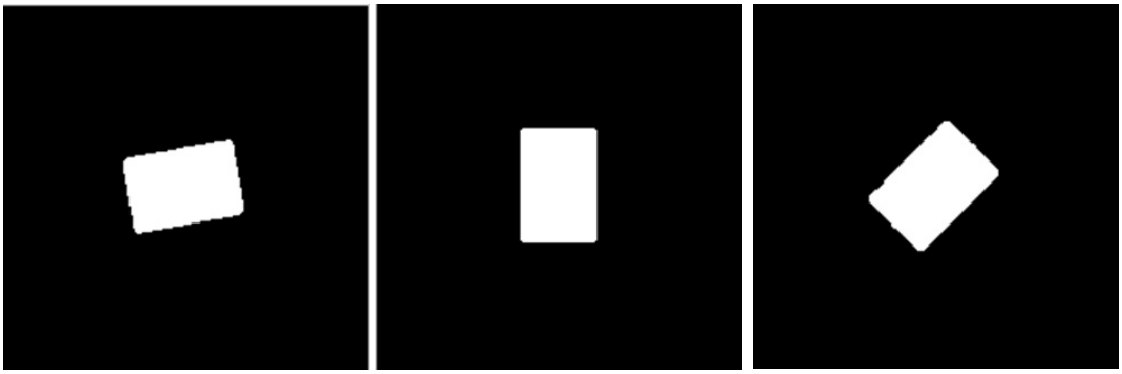


Figure 3.6 : Exemple d'images utilisées.

Les courbes des testes réalisés permettant de juger l'effet de la rotation pour les invariants de Fourier-Mellin, ce qui concerne les moments pour $k = 0$, ils sont nettement moins invariables puisque l'erreur maximale mesurée est inférieur à 0.53%, contre 50% pour les invariants possèdent $k \neq 0$.

Pour cette raison nous avons utilisé pour la reconnaissance dans la suite, seulement les invariants qui possède un k=0, dont le but d'augmenter le taux de reconnaissance.

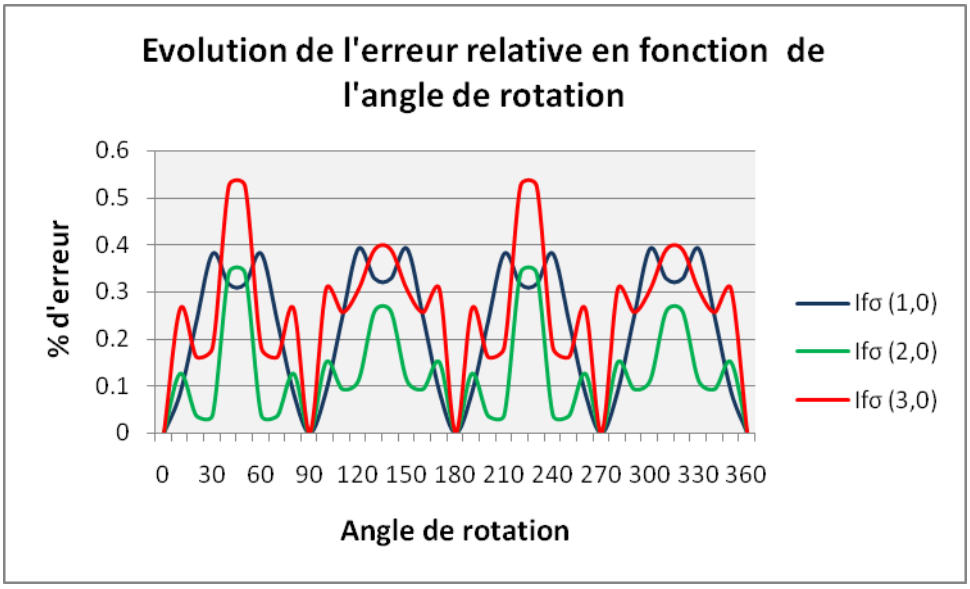


Figure 3.7 : Erreurs relatives pour k=0.

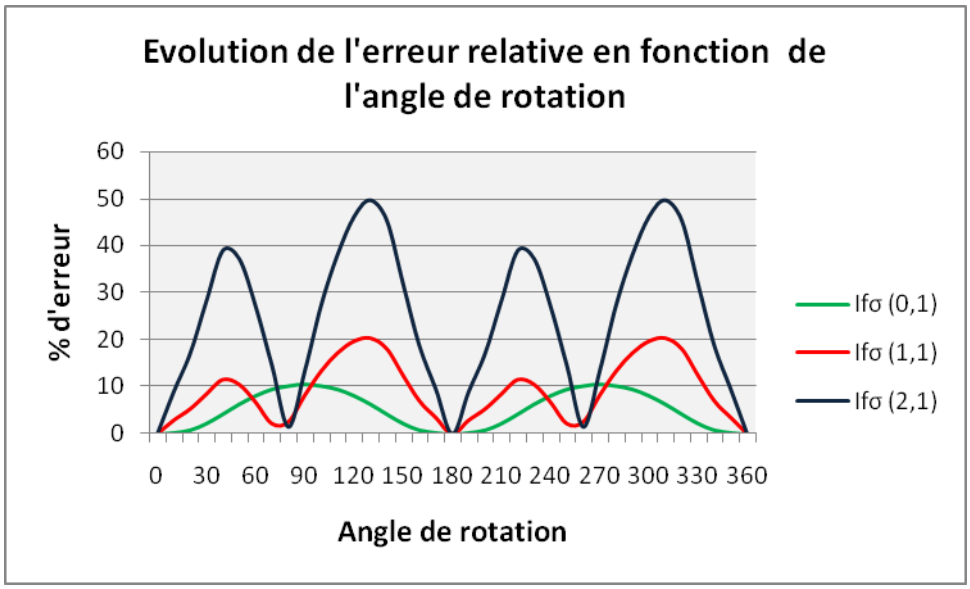


Figure 3.8 : Erreurs relatives pour k≠0.

3.3. Moments de Zernike

Les moments Zernike sont définis comme des polynômes complexes qui forment un ensemble orthogonal complet du disque unité $x^2 + y^2 \leq 1$, en coordonnées polaires (ρ, θ) [11].Présenté par Zernike (1934), cet ensemble polynomial est défini par l'équation (3.13) [14].

$$V(\rho, \theta) = R_{nm}(\rho) e^{(jm\theta)} \tag{3.13}$$

La base polynomiale de Zernike en coordonnées cartésiennes, d'ordre n et de répétition m, est donnée par :

$$V_{nm}(x, y) = R_{nm}(x, y) e^{\left(im \arctan\left(\frac{y}{x}\right)\right)} \tag{3.14}$$

$n \in N, m \in Z$, avec $(n - |m|)$ pair et $|m| \leq n$

Le rapport polynomial est défini comme suit :

$$R_{nm}(x, y) = \sum_{s=0}^{\frac{(n-|m|)}{2}} (-1)^s \frac{(n-s)!}{s!\left(\frac{n+|m|}{2} - s\right)!\left(\frac{n-|m|}{2} - s\right)!} (x^2 + y^2)^{\frac{(n-2s)}{2}} \tag{3.15}$$

Soit $\rho = \sqrt{x^2 + y^2}$ la longueur du vecteur à partir de l'origine du pixel (x,y) et $\theta = \arctan\left(\frac{y}{x}\right)$ l'angle entre l'axe x et ce vecteur. $R_{nm}(\rho)$, la représentation en coordonnées polaires ($x = \rho \cos\theta , y = \rho \sin\theta$) de $R_{nm}(x, y)$ est un polynôme de degré n.

$$R_{nm}(\rho) = \sum_{s=0}^{\frac{(n-|m|)}{2}} (-1)^s \frac{(n-s)!}{s!\left(\frac{n+|m|}{2} - s\right)!\left(\frac{n-|m|}{2} - s\right)!} \rho^{(n-2s)} \tag{3.16}$$

Les moments Zernike sont les projections de la fonction d'image $f(x, y)$ dans la base orthogonale de fonctions $V_{nm}(x, y)$ [12]. Le moment Zernike, d'ordre n et de répétition m, est un nombre complexe défini comme suit :

$$Z_{nm} = \frac{n+1}{\pi} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x, y)[V_{nm}(x, y)]^* \tag{3.17}$$

C'est évident à partir de l'équation (3.17) que les coordonnées de pixel d'image doivent être, d'abord, transformées sur le disque d'unité avant que le calcul de moment de Zernike soit effectué.

Les moments Zernike sont bien connus pour être invariants à la rotation. Une normalisation d'image est nécessaire pour rendre ces moments invariants à la translation et au facteur d'échelle.

3.3.1. Effet de la rotation

Nous avons utilisé une base d'image (Figure 3.11) pour tester l'invariance des moments Z_{nm}. Nous calculons comme exemple les 6 premiers invariants de Zernike. L'erreur relative pour les invariants possèdent m=0 est au maximum 14,03%, pour les autre invariant qui possède m≠0, nous pouvons voir ici que l'erreur devient vite très importante, dépassant 50%. Cette erreur peut influencer sur la décision de système et pour l'atténuer nous avons réalisé la rotation de l'objet de sorte que l'angle d'orientation de l'objet est toujours égal à zéro, Nous avons utilisé pour la reconnaissance seulement les invariant qui possède m=0, car ils sont moins sensible au bruit.

Figure 3.11 : Exemple d'images utilisées.

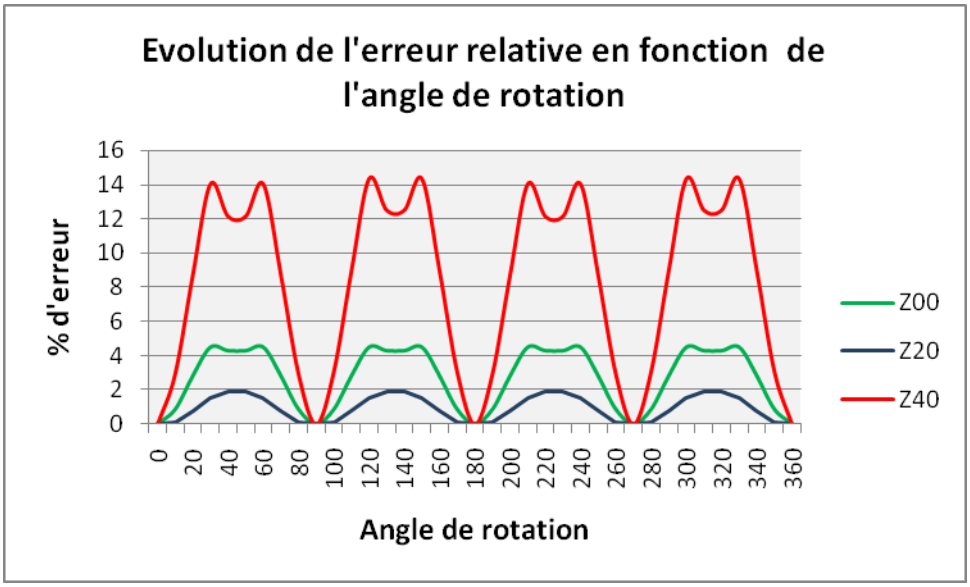


Figure 3.12 : Erreurs relatives pour m=0.

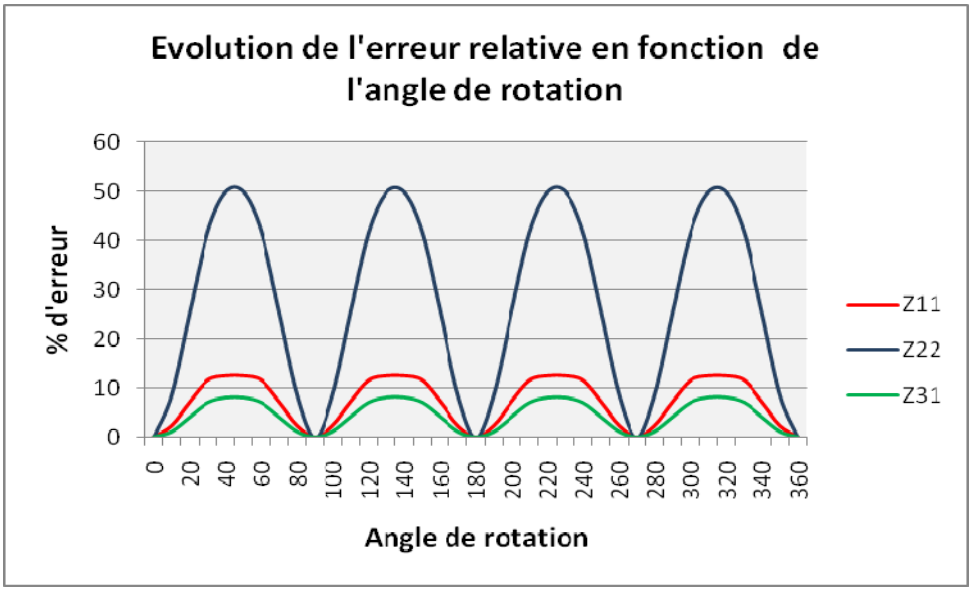


Figure 3.13 : Erreurs relatives pour m≠0.

3.3.2. Normalisation à la translation

Pour réaliser la normalisation à la translation, le centre de gravité de l'objet devrait être déplacé au centre de l'image original en utilisant les équations (2.47-2.48).

3.3.3. Normalisation à l'échelle

La taille de l'échelle spatiale d'une image peut être obtenue en modifiant les coordonnées cartésiennes de l'image source (Figure 2.8) selon les relations [1]:

$$x_j = s_x \grave{x}_j \tag{3.18}$$

$$y_k = s_y \grave{y}_k \tag{3.19}$$

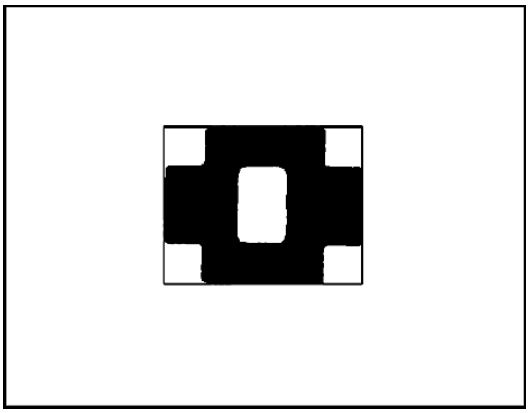
Où s_x et s_y sont des valeurs positives constantes, mais pas nécessairement des valeurs entières, appelées facteur d'échelle. Si s_x et s_y sont chacun plus grand que l'unité, le calcul des adresses a partir des équations (3.18-3.19) conduira à un grossissement. Réciproquement, si s_x et s_y sont chacun moins que l'unité, le résultat est une réduction.

Pour améliorer le temps de calcul de ces primitives invariantes, les objets ont été amenés à une taille égale 64x64 pixels.

$\grave{x}_j, \grave{y}_k$ sont les pixels source, représentant les positions des objets dans le champ de vision.

- ***Méthode de normalisation***

1 Extraction de l'objet de l'image originale en utilisant la méthode de rectangle de contenance.



a.Sélection de l'objet par la méthode de rectangle de contenance.

b.Objet sélectionné.

Figure 3.14 : Extraction de l'objet de l'image

2 Calculer les deux facteurs d'agrandissement et (ou) de réduction s_x et s_y pour changer la taille de l'objet à une taille égale 64x64 pixels

$$s_x = \frac{largeur\ de\ rectangle}{64} \tag{3.20}$$

$$s_y = \frac{\text{longueur}}{64} \tag{3.21}$$

3 Calculer les positions des pixels dans la nouvelle image en utilisant (3.18-3.19).

Figure 3.15 : objet avec une taille 64x64.

3.4. Conclusion

Dans ce chapitre, nous avons présenté deux méthodes permettant de représenter un objet quelconque au moyen d'un vecteur de caractéristiques. Ces deux méthodes sont d'une nécessité majeure pour la prochaine étape dont le but est de déterminer, parmi un ensemble fini de classe, à la quelle appartient un objet donné.

CHAPITRE 4

Les Réseaux De Neurones Artificiels (RNA)

Les Réseaux De Neurones Artificiels (RNA)

4.1. Introduction

Une première vague d'intérêt dans les réseaux de neurones (aussi connu sous le nom de modèles connexionnistes parallèle ou distribué) sont apparues après l'introduction de la modélisation des neurones par Mac Culloch et Pitts en 1943. Ces neurones ont été présentés comme les modèles de neurones biologiques et en tant que composants conceptuels pour les circuits qui pourraient effectuer des tâches de calcul [17].

4.2. Le neurone biologique:

Le neurone est une cellule composée d'un corps cellulaire et d'un noyau. Le corps cellulaire se ramifie pour former ce que l'on nomme les dendrites, C'est par Celles-ci que l'information est acheminée de l'extérieur vers le soma, corps du neurone. L'information traitée par le neurone chemine ensuite le long de l'axone pour être transmise aux autres neurones. La transmission entre deux neurones n'est pas directe. En réalité, il existe un espace intercellulaire de quelques dizaines d'Angstroms (10^{-9} m) entre l'axone d'un neurone (sortie) et les dendrites (entrée) d'un autre neurone. La jonction entre deux neurones est appelée la synapse [15].

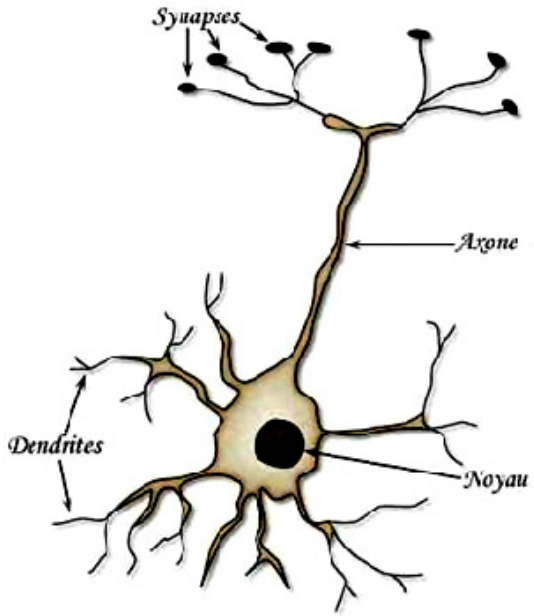


Figure 4.1 : Le neurone biologique.

4.3. Le neurone formel

La modélisation mathématique des neurones biologique par des neurones formels qui à été faite par le neuropsychiatre McCulloch et le logicien Pitts est illustré à la figure 4.2. Un neurone est essentiellement constitué d'un intégrateur qui effectue la somme pondérée de ses entrées. Le résultat n de cette somme est ensuite transformé par une fonction de transfert f qui produit la sortie y du neurone [17].

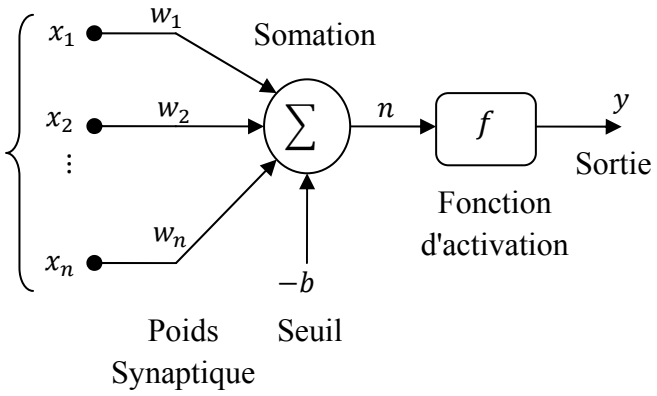


Figure 4.2.Un neurone artificiel

4.4. Réseau monocouche

Un réseau monocouche est composé de k neurones organisés en sortie et connectés à des entrées pondérées (Figure 4.3).

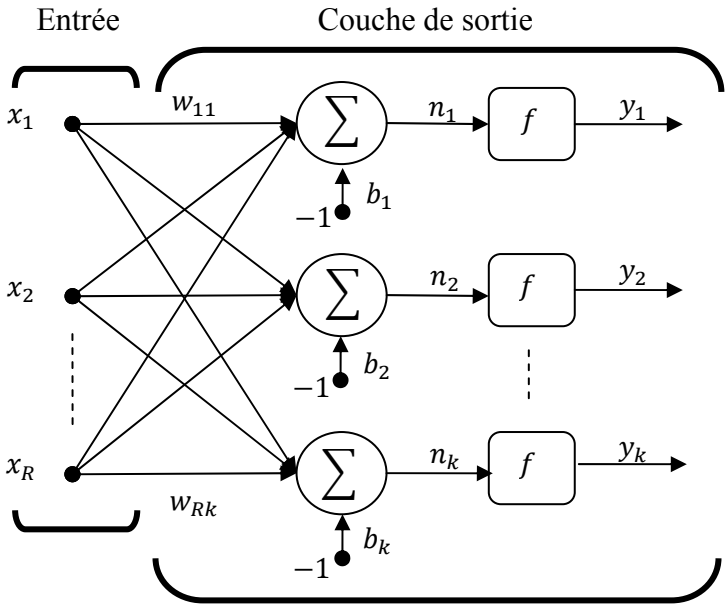


Figure 4.3 : Réseau monocouche.

Les R entrées du neurone k correspondent au vecteur $X = [x_1\ x_2\ \dots x_R]^T$, alors que $W = [w_{1k}\ w_{2k}\ \dots w_{Rk}]^T$, représente le vecteur des poids du neurone. La sortie n_k de l'intégrateur est donnée par l'équation suivante :

$$n_k = \sum_{i=1}^{R} w_{ik}x_i - b_K \tag{4.1}$$

Que l'on peut aussi écrire sous forme matricielle :

$$n = w^T X - b \tag{4.2}$$

En remplaçant w^T par une matrice $W = w^T$, l'équation (4.1) devient :

$$n = WX - b \tag{4.3}$$

La sortie n_k correspond à une somme pondérée des entrées moins ce qu'on nomme le biais b_K du neurone. Le résultat n_k de la somme pondérée s'appelle le niveau d'activation du neurone. Le biais b_K s'appelle aussi le seuil d'activation du neurone. La fonction d'activation f permet de définir l'état interne du neurone en fonction de son entrée totale, comme exemple on a la fonction Signe est définie par l'équation suivante :

$$f(n_k) = \begin{cases} 1 & si \quad n_k \geq 0 \\ -1 & sinon \end{cases} \tag{4.4}$$

Considérons le cas le plus simple, à savoir lorsque $R = 2$, c'est-à-dire lorsque les neurones ne sont reliés que de deux entrées. Dans ce cas, nous aurons $X = [x_1\ x_2]^T$, $W = [w_{1k}\ w_{2k}]^T$, $b = [b_K]$, où:

$$f(n_k) = \begin{cases} 1 & si \quad w_{1k}x_1 + w_{2k}x_2 - b_k \geq 0 \\ -1 & sinon \end{cases} \tag{4.5}$$

Cette dernière équation nous indique clairement que la sortie du réseau (neurone) peut prendre seulement deux valeurs distinctes selon le niveau d'activation du neurone : -1 lorsque ce dernier est strictement inférieur à 0; $+1$ dans le cas contraire. Il existe donc dans l'espace des entrées une frontière délimitant deux régions correspondantes. Cette frontière est définie par la condition $w_{1k}x_1 + w_{2k}x_2 - b_k = 0$ qui correspond à l'expression générale d'une droite.

$$x_2 = -\frac{w_{1k}}{w_{2k}} x_1 + \frac{b_k}{w_{2k}} \tag{4.6}$$

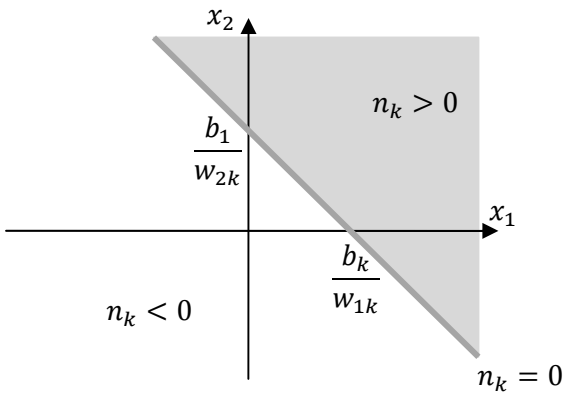


Figure 4.4 : Frontière de décision

On constate suivant la figure 4.4 que la droite définie par l'équation (4.6), peut seulement classifier les problèmes linéairement séparable par la modification des poids w_{1k}, w_{2k}et le seuil b_k.

4.5. Réseau multicouche

Nous avons vu que les réseaux monocouche ne pouvaient résoudre que des problèmes de classification linéairement séparables. Les réseaux multicouches permettent de lever cette limitation. Un réseau multicouche n'est rien d'autre qu'un assemblage de couches concaténées les unes aux autres, de la gauche vers la droite, en prenant les sorties d'une couche et en les injectant comme les entrées de la couche suivante [16].

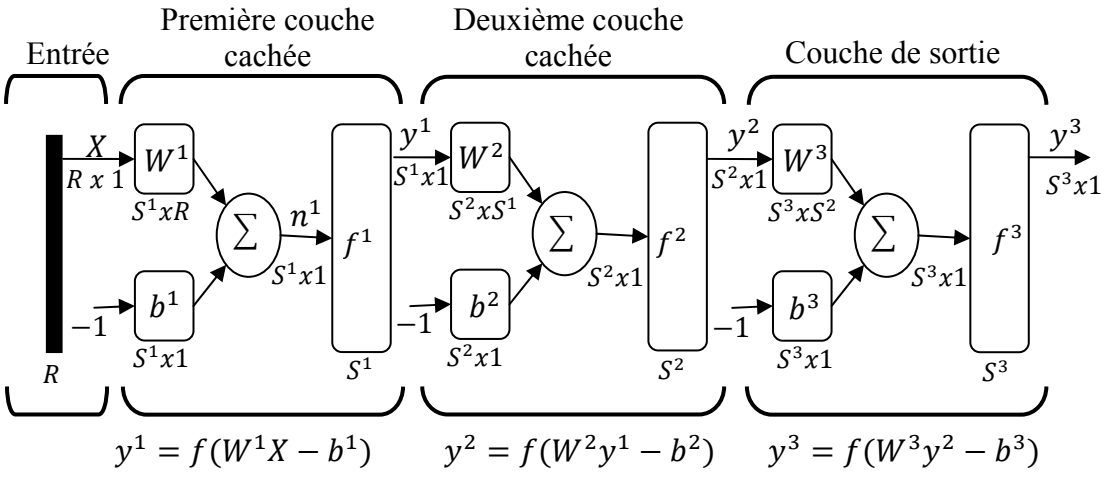


Figure 4.5 : Réseau multicouche.

L'équation qui décrit les sorties d'une couche l dans un réseau multicouche est donnée par :

$$y_k{}^{l+1} = f^{l+1}\left(\sum_{i=1}^{R_j} w_{ik}{}^{l+1} y_i{}^{l} - b_K{}^{l+1}\right) \tag{4.7}$$

Que l'on peut aussi écrire sous forme matricielle :

$$y^{l+1} = f^{l+1}(W^{l+1}y^{l} - b^{l+1}) \tag{4.8}$$

Pour $\mathrm{l} = 0, \ldots, \mathrm{M} - 1$.

Où M est le nombre total de couches, et $y^0 = X$ définit l'entrée de réseau. Les sorties du réseau correspondent alors à y^M.

Pour faire de la classification, on utilisera des réseaux soit à deux, soit à trois couches de neurones sigmoïdes. une seule couche cachée suffit à engendrer des frontières de décision convexes, ouvertes ou fermées, de complexité arbitraire, alors que deux couches cachées permettent de créer des frontières de décision concaves ou convexes, ouvertes ou fermées, de complexité arbitraire [16].

4.6. L'apprentissage des réseaux de neurones

L'apprentissage d'un réseau de neurone peut être considéré comme une action de la mise à jour de ses poids des connexions synaptiques, afin de résoudre le problème demandé. L'apprentissage est la caractéristique principale des réseaux de neurones et il peut se faire de différentes manières et selon différentes règles. On peut distinguer deux types d'apprentissage : l'apprentissage supervisé et l'apprentissage non-supervisé.

4.6.1. L'apprentissage supervisé

En général, l'apprentissage des réseaux de neurones est effectué de sorte que pour une entrée particulière présentée au réseau corresponde une cible spécifique. L'ajustement des poids se fait par comparaison entre la réponse du réseau (ou sortie) et la cible, jusqu'à ce que la sortie corresponde à la cible [16]. On utilise pour ce type d'apprentissage dit : supervisé un nombre conséquent de pair entrée/sortie.

4.6.2. L'apprentissage non supervisé

On utilise ces méthodes pour répartir les pixels entre un nombre fixé de classes sans aucune référence préalable à la nature de ces classes. Celles-ci sont déterminées par l'algorithme sans l'aide de l'utilisateur [16].

4.7. Normalisation des entrées

Avant tout apprentissage, il est indisponible de normaliser et de centrer toutes les variables d'entrées, en effet si ces entrées ont des grandeurs très différents, celle qui sont petites n'ont pas d'influence sur l'apprentissage.

Donc pour chaque vecteur d'entée x_i on effectue le changement de variable $\grave{x}_i$.

$$\grave{x_j} = \frac{x_j}{max(x_j) - min\ (x_j)} \tag{4.9}$$

Où $max(.)$ et $min(.)$ sont les valeurs min et max établies sur des données d'apprentissage.

4.8. Algorithmes d'apprentissage

4.8.1. Rétro-propagation du gradient

Le principe de l'algorithme est de minimiser l'erreur quadratique entre les sorties calculées et celles souhaitées équation (4.10). Il s'agit ensuite de calculer la contribution à cette erreur de chacun des poids synaptiques (l'erreur calculée en sortie est transmise en sens inverse vers l'entrée) [16].

L'erreur quadratique est définie par :

$$E = \frac{1}{2}\sum_{p=1}^{P}\sum_{k=1}^{K}(d_{p,k} - y_{p,k}^{M})^2 \tag{4.10}$$

Où : P est le nombre d'exemples et K le nombre de sorties.

Pour minimiser E, on calcule son gradient par rapport à chaque poids w_{ij}^{l} définie par :

$$\nabla E = \frac{\partial E}{\partial w_{ij}^{l}} \tag{4.11}$$

On cherche à minimiser l'erreur sur chaque présentation individuelle d'exemple. L'erreur pour un exemple est:

$$E = \frac{1}{2}\sum_{k=1}^{K}(d_k - y_k^M)^2 \tag{4.12}$$

On remarque que w_{ij}^l ne peut influencer la sortie du réseau qu'à travers le calcul de la quantité n_j^l (figure 4.5), ce qui nous autorise à écrire que :

$$\frac{\partial E}{\partial w_{ij}^l} = \frac{\partial E}{\partial n_j^l}\frac{\partial n_j^l}{\partial w_{ij}^l} \tag{4.13}$$

$$\frac{\partial E}{\partial w_{ij}^l} = \frac{\partial E}{\partial n_j^l} x_i^{l-1} \tag{4.14}$$

Il existe deux cas pour calculer la quantité : $\frac{\partial E}{\partial n_j^l}$

- ***La cellule j est une cellule de sortie (l=M)***

Dans ce cas, la quantité n_j^l ne peut influencer la sortie du réseau que par le calcul de y_j^l. Nous avons donc:

$$\frac{\partial E}{\partial n_j^l} = \frac{\partial E}{\partial y_j^l}\frac{\partial y_j^l}{\partial n_j^l} \tag{4.15}$$

$$\frac{\partial E}{\partial n_j^l} = \frac{\partial y_j^l}{\partial n_j^l}\frac{\partial}{\partial y_j^l}\left[\frac{1}{2}\sum_{k=1}^{K}\left(d_k - y_k^l\right)^2\right] \tag{4.16}$$

Seul le terme correspondant à $k = j$ a une dérivée non nulle, ce qui nous donne finalement :

$$\frac{\partial E}{\partial n_j^l} = -\frac{\partial y_j^l}{\partial n_j^l}\left(d_j - y_j^l\right) \tag{4.17}$$

$$\sigma_j^l = -\frac{\partial y_j^l}{\partial n_j^l}\left(d_j - y_j^l\right) \tag{4.18}$$

○ ***La cellule j est une cellule interne***

Dans ce cas, la quantité n_j^l va influencer sur les cellules qui prennent comme entrée la sortie y_j^l.

$$\frac{\partial E}{\partial n_j^l} = \sum_{t=1}^{T} \frac{\partial E}{\partial n_t^{l+1}} \frac{\partial n_t^{l+1}}{\partial n_j^l} \tag{4.19}$$

T : les cellules qui prennent comme entrée la sortie y_j^l.

$$\frac{\partial E}{\partial n_j^l} = \sum_{t=1}^{T} \frac{\partial E}{\partial n_t^{l+1}} \frac{\partial n_t^{l+1}}{\partial y_j^l} \frac{\partial y_j^l}{\partial n_j^l} \tag{4.20}$$

$$\frac{\partial E}{\partial n_j^l} = \sum_{t=1}^{T} \frac{\partial E}{\partial n_t^{l+1}} w_{jt}^{l+1} \frac{\partial y_j^l}{\partial n_j^l} \tag{4.21}$$

Soit encore:

$$\frac{\partial E}{\partial n_j^l} = \frac{\partial y_j^l}{\partial n_j^l} \sum_{t=1}^{T} \frac{\partial E}{\partial n_t^{l+1}} w_{jt}^{l+1} \tag{4.22}$$

$$\sigma_j^l = \frac{\partial y_j^l}{\partial n_j^l} \sum_{t=1}^{T} \sigma_t^{l+1} w_{jt}^{l+1} \tag{4.23}$$

Par l'étude de ces deux cas, nous avons obtenu deux équations (4.18-4.23) qui nous permettent de calculer les dérivées partielles $\frac{\partial E}{\partial n_j^l}$ pour toute cellule j. Le calcul devra être fait pour les cellules de sortie puis des cellules de l'avant-dernière couche jusqu'aux cellules de la première couche. C'est pour cette raison que l'on parle de « rétropropagation ». Grace à l'équation (4.14), nous pouvons calculer toutes les dérivées partielles $\frac{\partial E}{\partial w_{ij}^l}$.

L'application de la méthode du gradient nous invite donc à modifier le poids w_{ij}^l d'une quantité Δw_{ij}^l. Le seuil est actualisé de la même façon.

$$w_{ij}^l(t+1) = w_{ij}^l(t) + \Delta w_{ij}^l \tag{4.24}$$

Avec :

$$\Delta w_{ij}^{l} = -\varepsilon \frac{\partial E}{\partial w_{ij}^{l}} \tag{4.25}$$

ε : Pas d'apprentissage $0 < \varepsilon < 1$

Les inconvénients bien connus de cette méthode sont :

1. le choix de ε est empirique.
2. Si ε est trop petit, le nombre d'itérations peut être très élevé,
3. Si ε est trop grand, les valeurs de la suite risquent d'osciller autour du minimum sans converger,
4. Rien ne garantit que le minimum trouvé est un minimum global.

- ***Résumé de l'algorithme de rétro-propagation***

1. Initialisation aléatoire des poids w_{ij}^{l} dans [-0.5, 0.5] et appliquer un vecteur d'entrée X
2. Calculer les termes d'erreur de signal de la couche de sortie et les couches cachées

$$\sigma_j^M = -\frac{\partial y_j^M}{\partial n_j^M}\left(d_j - y_j^M\right) \tag{4.26}$$

$$\sigma_j^l = \frac{\partial y_j^l}{\partial n_j^l}\sum_{t=1}^{T} \sigma_t^{l+1} w_{jt}^{l+1} \tag{4.27}$$

3. Mise à jour les poids de la couche de sortie et les couches cachées

$$w_{ij}^{l}(t+1) = w_{ij}^{l}(t) - \varepsilon\, \sigma_j^l x_i^{l-1} \tag{4.28}$$

4. Répéter ce processus jusqu'à ce que l'erreur E_P devienne acceptable

La convergence du réseau par rétro-propagation est un problème crucial car il requiert de nombreuses itérations. Pour pallier à ce problème, il existe des techniques plus ou moins rapides, performantes et ne nécessite pas un grand espace mémoire. Il apparaît que la technique de Levenberg-Marquardt est un algorithme très rapide.

4.8.2. Algorithme de Levenberg-Marquardt

Parmi les algorithmes de la famille quasi-Newton, la méthode de LEVENBERG-MARQUARDT (LM) est un standard pour l'optimisation de l'erreur quadratique due à ses propriétés de convergence rapide et de robustesse [16].

La mise à jour de l'algorithme newton pour réduire au minimum une fonction V(x) en ce qui concerne le vecteur x est donnée par:

$$\Delta x = -|\nabla^2 V(x)|^{-1} \nabla V(x) \tag{4.29}$$

Où ∇^2V(x) est la matrice Hessienne et ∇V(x) est le gradient vecteur.

V (x) est la somme d'erreurs carrées donnée par :

$$V(x) = \sum_{h=1}^{N} e_h^2(x) \tag{4.30}$$

Ainsi,

$$\nabla V(x) = 2J^T(x)e(x) \tag{4.31}$$

$$\nabla^2 V(x) = 2J^T(x)J(x) + 2S(x) \tag{4.32}$$

Où e(x)est le vecteur d'erreur, J(x) est la matrice Jacobian donnée par:

$$J(x) = \begin{bmatrix} \frac{\partial e_1(x)}{\partial x_1} & \frac{\partial e_1(x)}{\partial x_2} & \cdots & \frac{\partial e_1(x)}{\partial x_n} \\ \frac{\partial e_2(x)}{\partial x_1} & \frac{\partial e_2(x)}{\partial x_2} & \cdots & \frac{\partial e_2(x)}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_N(x)}{\partial x_1} & \frac{\partial e_N(x)}{\partial x_2} & \cdots & \frac{\partial e_N(x)}{\partial x_n} \end{bmatrix} \tag{4.33}$$

Où S(x)est donné par:

$$S(x) = \sum_{h=1}^{N} e_h(x)\, \nabla^2 e_h(x) \tag{4.34}$$

Négligeant les dérivés de second ordre du vecteur d'erreur donc:

$$S(x) \approx 0 \tag{4.35}$$

$$\nabla^2 V(x) = 2J^T(x)J(x) \tag{4.36}$$

En remplaçant les équations (4.31-4.36) dans l'équation. (4.29) nous obtenons la mise à jour Gauss-Newton, donnée par:

$$\Delta x = -|J^T(x)J(x)|^{-1}J^T(x)e(x) \tag{4.37}$$

L'avantage de cette méthode est qu'il n'exige pas le calcul des dérivés de second ordre. Néanmoins, la matrice $J^T(x)J(x)$ peut être irréversible. Ceci est surmonté avec l'algorithme de Levenberg-Marquardt, donnée par:

$$\Delta x = -|J^T(x)J(x) + \mu\, I|^{-1}J^T(x)e(x) \tag{4.38}$$

Cette méthode tend vers la méthode de Newton pour une valeur de μ petite équation (4.37) mais est équivalente à la méthode du gradient pour une valeur de μ grande. Le Hessien est toujours défini positif ce qui assure la convergence vers un minimum de la solution.

- ***Pour μ grande***

De l'équation (4.38) on a :

$$\Delta w = -|\mu\, I|^{-1}J^T(w)e(w) \tag{4.39}$$

La matrice Jacobian est donnée par :

$$J(w) = \begin{bmatrix} \frac{\partial e_1(w)}{\partial w_{11}^l} & \frac{\partial e_1(w)}{\partial w_{21}^l} & \cdots & \frac{\partial e_1(w)}{\partial w_{i1}^l} \\ \frac{\partial e_2(w)}{\partial w_{12}^l} & \frac{\partial e_2(w)}{\partial w_{22}^l} & \cdots & \frac{\partial e_2(w)}{\partial w_{i2}^l} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_j(w)}{\partial w_{1j}^l} & \frac{\partial e_j(w)}{\partial w_{2j}^l} & \cdots & \frac{\partial e_j(w)}{\partial w_{ij}^l} \end{bmatrix} \tag{4.40}$$

La transposée de la matrice Jacobian est donnée par :

$$J^T(w) = \begin{bmatrix} \frac{\partial e_1(w)}{\partial w_{11}^l} & \frac{\partial e_2(w)}{\partial w_{12}^l} & \cdots & \frac{\partial e_j(w)}{\partial w_{1j}^l} \\ \frac{\partial e_1(w)}{\partial w_{21}^l} & \frac{\partial e_2(w)}{\partial w_{22}^l} & \cdots & \frac{\partial e_j(w)}{\partial w_{2j}^l} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e_1(w)}{\partial w_{i1}^l} & \frac{\partial e_2(w)}{\partial w_{i2}^l} & \cdots & \frac{\partial e_j(w)}{\partial w_{ij}^l} \end{bmatrix} \tag{4.41}$$

L'erreur entre la sortie désirée et la sortie de réseau est donné par :

$$e(w) = \begin{bmatrix} (d_1 - y_1) & 0 & \dots & 0 \\ 0 & (d_2 - y_2) & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & (d_j - y_j) \end{bmatrix} \tag{4.42}$$

L'adaptation finalement réalisée correspond à :

$$\Delta w_{ij} = -\begin{bmatrix} \varepsilon(d_1 - y_1)\frac{\partial e_1(w)}{\partial w_{11}^l} & \varepsilon(d_2 - y_2)\frac{\partial e_2(w)}{\partial w_{12}^l} & \cdots & \varepsilon(d_j - y_j)\frac{\partial e_j(w)}{\partial w_{1j}^l} \\ \varepsilon(d_1 - y_1)\frac{\partial e_1(w)}{\partial w_{21}^l} & \varepsilon(d_2 - y_2)\frac{\partial e_2(w)}{\partial w_{22}^l} & \cdots & \varepsilon(d_j - y_j)\frac{\partial e_j(w)}{\partial w_{2j}^l} \\ \vdots & \vdots & \ddots & \vdots \\ \varepsilon(d_1 - y_1)\frac{\partial e_1(w)}{\partial w_{i1}^l} & \varepsilon(d_2 - y_2)\frac{\partial e_2(w)}{\partial w_{i2}^l} & \cdots & \varepsilon(d_j - y_j)\frac{\partial e_j(w)}{\partial w_{ij}^l} \end{bmatrix} \tag{4.43}$$

$$\Delta w_{ij}^l = -\varepsilon(d_j - y_j)\frac{\partial e_j(w)}{\partial w_{ij}^l} \tag{4.44}$$

$$\Delta w_{ij}^l = -\varepsilon(d_j - y_j)\frac{\partial (d_j - y_j)}{\partial w_{ij}^l} \tag{4.45}$$

$$\Delta w_{ij}^l = -\varepsilon\frac{\partial E}{\partial w_{ij}^l} \tag{4.46}$$

Donc on constate que la méthode de Levenberg-Marquardt est équivalente à la méthode du gradient pour une valeur de μ grande

- ***Résumé de l'algorithme***

1. Présentez toutes les entrées au réseau et calculez les sorties et les erreurs correspondantes de réseau
2. Calculez la matrice de Jacobian J^T
3. Mise à jour de poids

$$w_{ij}^l(t+1) = w_{ij}^l(t) + \Delta w_{ij}^l \tag{4.47}$$

4. Recalculer l'erreur après la mise à jour des poids. Si cette erreur est inférieur à celle calculée à l'étape 1 le paramètre d'apprentissage μ est réduit par μ^-, sinon le paramètre d'apprentissage μ est augmenté par μ^+. Les paramètres μ^-, et μ^+ sont prédéfinis par l'utilisateur, généralement μ^+ est fixé à 10 et μ^- est fixé à 0.1.

5. L'algorithme converge quand la norme du gradient est inférieure à une certaine valeur prédéterminée, ou quand l'erreur a été réduite à un certain but d'erreur.

4.9. Répartition de la base de d'exemples

Avant d'établir un modèle, la base de données est subdivisée souvent en apprentissage, validation et ensemble de test.

Ensemble d'apprentissage : l'ensemble d'apprentissage est employé pour former ou établir un modèle.

Ensemble de validation : l'ensemble de la validation est souvent utilisé pour arrêter l'apprentissage lorsque l'indice de performance (une erreur) calculé sur les données de validation cesse de s'améliorer pendant plusieurs périodes d'entraînement.

Ensemble de test : quant un modèle est finalement choisi, son exactitude sur l'ensemble de validation est toujours une estimation optimiste de façon dont elle exécuterait avec des données inconnues. Donc la précision du modèle sur les données de test donne une estimation réaliste de la performance du modèle sur les données complètement inconnues.

4.10. Conclusion

Ce chapitre donne une revue brève sur le principe de fonctionnement des réseaux de neurones Nous avons jugé utile cette présentation puisque ces réseaux vont être utilisés par la suite dans le processus de reconnaissance en se servant des vecteurs de caractéristiques déjà calculés dans les chapitres précédents.

CHAPITRE 5

Résultats Expérimentaux

Résultats Expérimentaux

5.1.Introduction

Dans cette section, nous présentons les résultats expérimentaux obtenus par l'application de différentes méthodes de reconnaissance que nous avons vu précédemment ainsi que les avantages et les inconvénients de chaque méthode.

Comme nous l'avons dit dans les chapitres précédents, lorsque les pièces transportées par le tapis roulant traversent le champ de vision de la caméra, ce tapis s'arrête et le traitement de reconnaissance commence afin d'identifier les différents objets dans le champ de vision. Les algorithmes qui nous permettent d'identifier les objets ont été testés sur une collection comprend 14 objets bidimensionnel réparti en 14 classe. Le tableau 5.2 montre les objets utilisés ainsi que la classe de chacun de ces objets.

5.2.Méthodes corrélatives

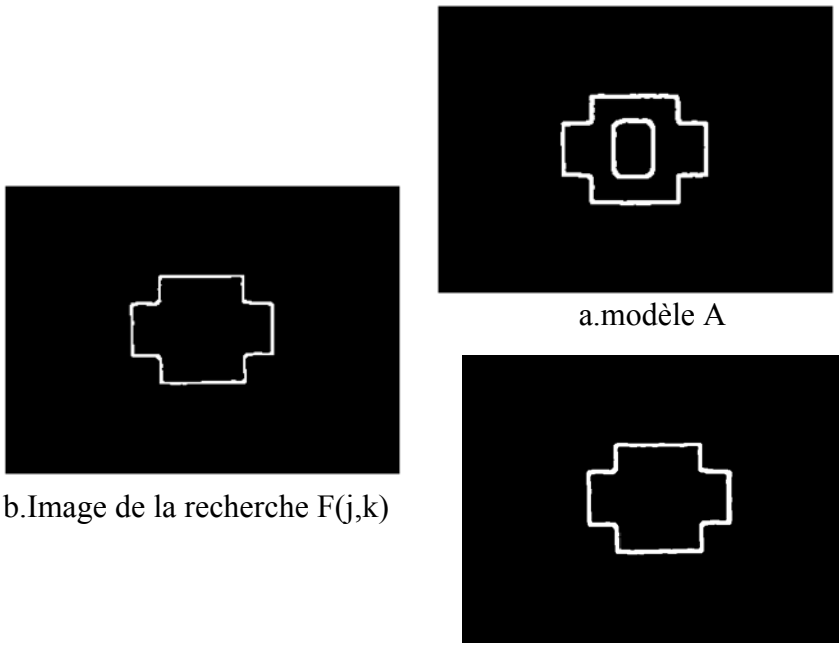
5.2.1.Correspondance de modèle

Cette méthode nous a donné un taux de reconnaissance élevé ≈100%. Certaines remarques sont cependant à soulever:

- Un grand espace mémoire utilisé pour stocker les modèles de comparaison (les 14 objets).
- La cross-corrélation normalisée entre le modèle et l'image de la recherche peut avoir comme résultat la valeur 1 malgré que les deux objets ne sont pas les mêmes. On a comme exemple la figure 5.1, suivant cette figure on peut voir facilement que la cross-corrélation entre le modèle A est l'image de la recherche est égale à 1, et pour éviter ce problème il faut que la comparaison s'effectue d'abord entre le modèle B et l'image de la recherche sans passer à la deuxième comparaison car l'objet dans l'image de la recherche est déjà identifié.

La cross-corrélation entre le modèle A est l'image de la recherche.

$$\tilde{R}_{MI} = \frac{\sum_j \sum_k [F(j,k) T_A(j,k)]}{\sum_j \sum_k [F(j,k)]^2} = 1 \qquad (5.1)$$



a.modèle A

b.Image de la recherche F(j,k)

c.modèle B

Figure 5.1 : Problème lier au calcule de la cross-corrélation.

5.2.2.Rectangle de contenance

Cette méthode basée essentiellement sur le calcul du facteur de reconnaissance : surface de l'objet sur surface de rectangle. Comme ce facteur n'a pas une valeur fixe à cause de bruit et des différentes opérations de prétraitement (filtrage, binarisation), l'identification de chaque objet a été liée à un intervalle traduisant la variation de ce facteur.

L'avantage de cette méthode est qu'elle n'est pas sensible à la variation de l'échelle, de la translation et de la rotation, mais l'utilisation de plusieurs objets provoque des chevauchements entre les intervalles qui traduisent les classes de ces objets. Pour cette raison on a utilisé pour la classification seulement 5 objets, le taux de reconnaissance dans ce cas est égale à 100% malgré le déplacement de la caméra d'une distance variée de 43cm jusqu'à 102 cm par rapport au plan de vision.

Tableau 5.1 Intervalles de variation des facteurs de reconnaissances

Objets	**Intervalles de variation de facteur de reconnaissance**
4	0.89-0.99
5	0.25-0.42
8	0.43-0.53
9	0.54-0.72
10	0.73-0.88

5.3.Méthodes vectorielles

Les moments calculés précédemment (moments de Fourier-Mellin ou moments de Zernike) et la classe de chaque objet forment un ensemble de données $\{(x_i, y_i)\}$,$i = 1 \dots M$ où chaque $x_i \in \mathbb{R}^d$ et $y_i \in \{1, \dots, N\}$ dans le cas où nous cherchons à reconnaître N classe différentes. Notre objectif est de construire une fonction qui estime les dépendances entre les exemples x_i et les classe y_i et qui minimise le risque d'erreur de classification pour un échantillon x_i n'appartenant pas à la base de données.

La base de données comporte au totale 112 échantillons pour chaque méthode de reconnaissance (méthode de Fourier-Mellin et méthode de Zernike). 56 de ces échantillons ont servi à constituer la base d'apprentissage et 28 échantillons la base de validation, les 28 autres étant réservés au test.

- ***La base de données dans le cas de la méthode de Fourier-Mellin***

La base de données comprend les éléments suivants :

Le vecteur d'entrées $X = [x_1 \dots x_{12}]$ qui représente le vecteur moment de Fourier-Mellin, et le vecteur de sortie $Y = [y_1 \dots y_4]$ qui définie la classe de chaque objet.

Les courbes de la figure 5.3 représentent les moments de Fourier-Mellin pour des images de dimension 200x200 pixels (Figure 5.2) avec $k = 0; p = 12, \sigma = 0.5$ et $r_{max} = 95$ pixels.

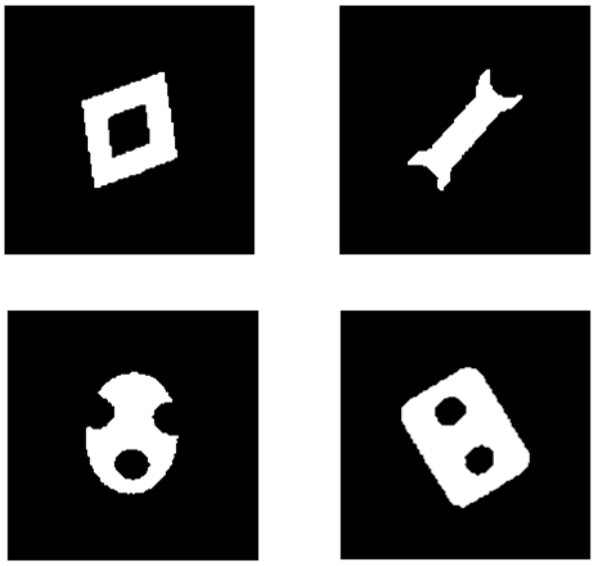


Figure 5.2 : Quelques objets de la base d'images

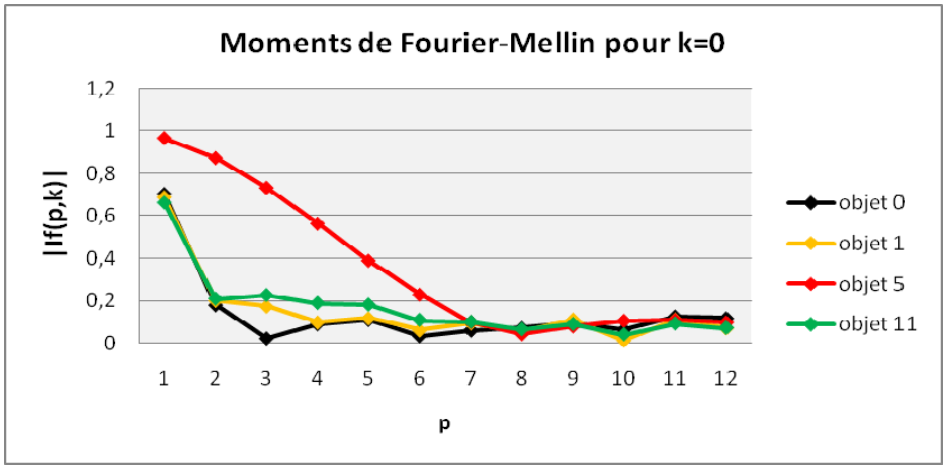


Figure 5.3 : Moments de Fourier-Mellin.

○ ***La base de données dans le cas de la méthode de Zernike***

La base de données comprend les éléments suivants :

Le vecteur d'entrées $X = [x_1 \dots x_8]$ qui représente le vecteur moment de Zernike, et le vecteur de sortie $Y = [y_1 \dots y_4]$ qui définie la classe de chaque objet.

Les courbes de la figure 5.5 représentent les moments de Zernike pour des images de dimension 64x64 pixels (Figure 5.4), et comme les coordonnées de l'objet doivent être définies dans un disque de rayon égal à l'unité $x^2 + y^2 \leq 1$ on a divisé toutes les coordonnées de l'objet par $r = 47$ qui correspond à l'unité.

Le r est choisi tel que $r \geq \sqrt{\left(\frac{64}{2}\right)^2 + \left(\frac{64}{2}\right)^2}$

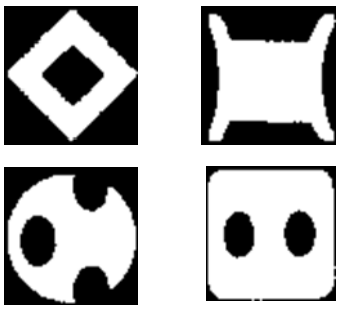


Figure 5.4 : Quelques objets de la base d'images.

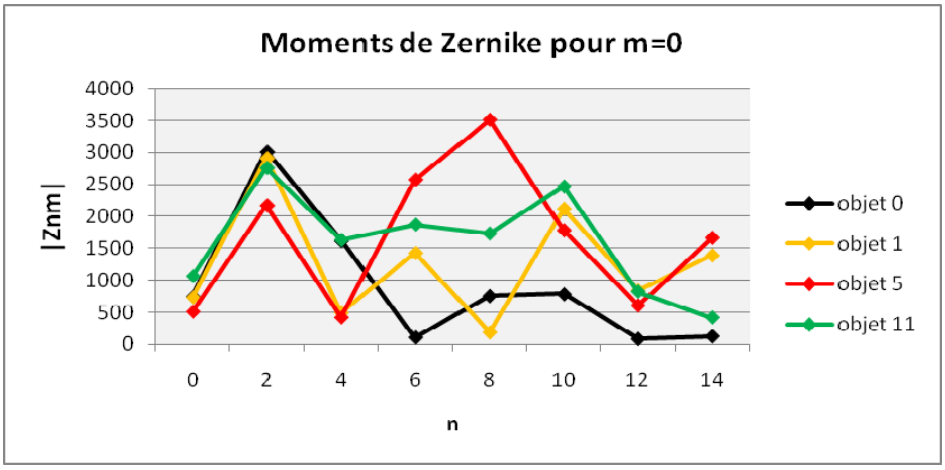


Figure 5.5 : Moments de Zernike.

5.3.1.Détermination de la structure du Réseaux

Le réseau utilisé est un perceptron multicouche (MLP). Le nombre de ces couches est variable. Ci-dessous nous décrirons la technique que nous avons adoptée pour fixer ce nombre. Nous montrons également la manière dont est fixé le nombre de neurones de chaque couche interne. Le nombre de cellules de la couche d'entrée est fixé par la dimension du vecteur des caractéristiques utilisées. Ce nombre est de 12 dans le cas de la méthode de Fourier-Mellin et 8 dans le cas de la méthode de Zernike. Pour la couche de sorti, on a adopté un codage binaire. De ce fait le nombre de neurone de cette couche est fixé à 4, ce qui donne un total de 16 combinaisons largement suffisant pour le codage de nos 14 sorties.

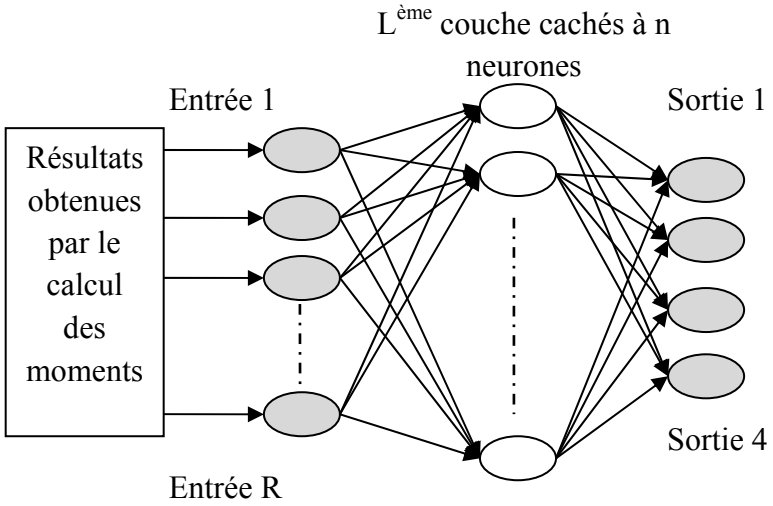


Figure 5.6 : Schéma général du réseau multicouche utilisé.

La sortie produit des nombres réels entre 0 et 1 indiquant la probabilité d'appartenance de l'échantillon à chacune des 14 classes.

Avant de commencer à présenter les décisions données par les réseaux utilisés, nous présentons la classe attribuée à chaque objet (Tableau 5.2).

Tableau 5.2 : Objets et leurs classes.

objets	Sorties du réseau				Classe
	Y_4	Y_3	Y_2	Y_1	
	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	2
	0	0	1	1	3
	0	1	0	0	4
	0	1	0	1	5
	0	1	1	0	6
	0	1	1	1	7
	1	0	0	0	8
	1	0	0	1	9
	1	0	1	0	10
	1	0	1	1	11
	1	1	0	0	12
	1	1	0	1	13

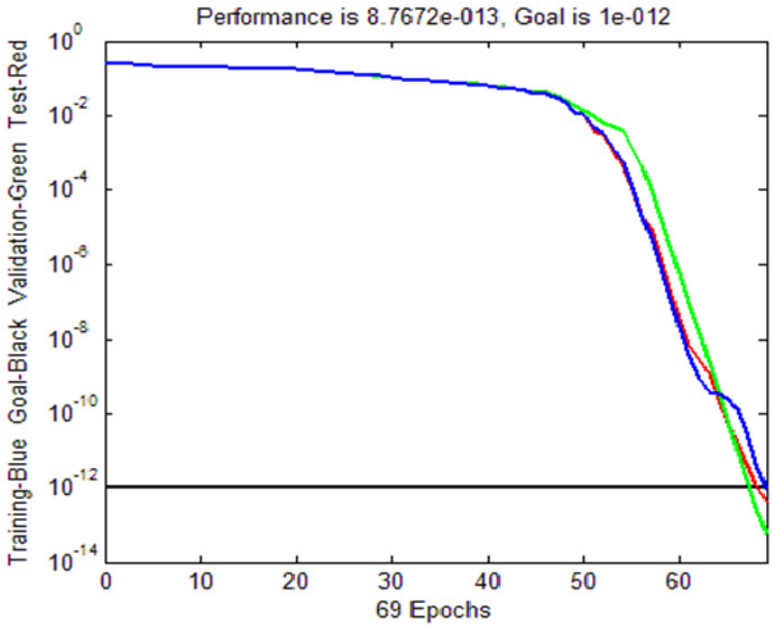
Pour les deux méthodes de reconnaissance (méthode de Fourier-Mellin et méthode de Zernike), c'est la méthode d'apprentissage LM qui est adoptée.

On a développé un programme qui permis de déterminer l'architecture du réseau : le nombre de couches internes et le nombre de neurones de chacune de ces couches. Cet algorithme est résumé ce dessous :

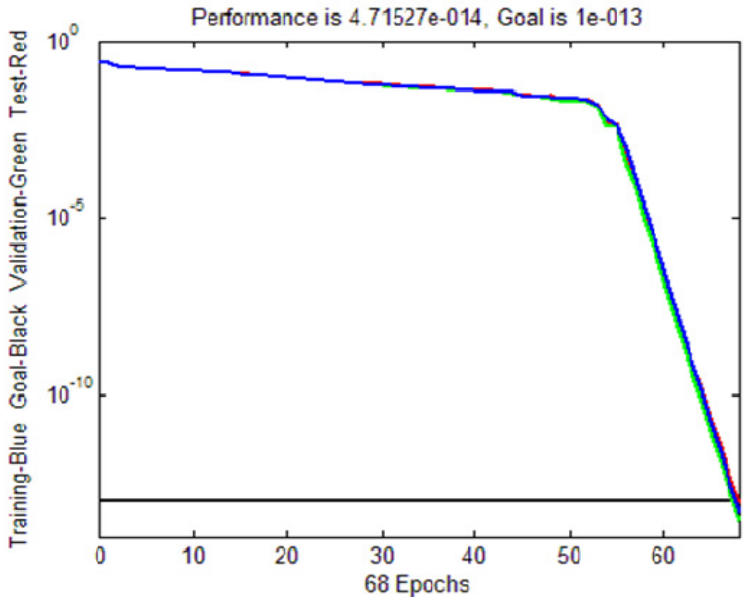
1. A partir des fichiers créés par l'utilisateur comprenant la base de données d'apprentissage, de validation et de test, le programme commence par une seule couche interne avec un seul neurone.
2. La phase d'apprentissage/validation/test est alors lancée. Ces trois opérations sont en fait opérées en parallèle. Ça permet d'avoir une image de la réponse du réseau une fois cette phase achevée.

3. Si l'objectif n'est pas achevé c'est-à-dire que l'erreur d'apprentissage n'est pas atteinte, le programme va augmenter le nombre de neurones jusqu'à une certaine valeur défini par l'utilisateur.
4. Si le nombre de neurones défini par l'utilisateur dans la première couche est achevé sans atteindre l'objectif, le programme va ajouter une deuxième couche en gardant les neurones de la couche précédente et on revient à l'étape 2.

Les courbes d'apprentissage, de validation et de test pour les deux méthodes sont données ci-dessous.



a.Apprentissage, Validation et test pour méthode de Fourier-Mellin



b.Apprentissage, Validation et test pour la méthode de Zernike

Figure 5.4 : Erreur d'apprentissage en utilisant la méthode LM

Pour la méthode de Fourier-Mellin on a trouvés un réseau de neurones de deux couches internes, la première couche contient 16 neurones et la deuxième couche contient 11 neurones. Pour la méthode de Zernike on a trouvé un réseau de neurones avec une seule couche cachée avec 7 neurones.

5.3.2.Résultats et évaluation

On a entrainé les réseaux de neurones pour les deux méthodes par une base de données créé à une distance fixe entre la caméra et le plan de vision. Ensuite on a testé les deux méthodes de reconnaissance pour des distances quelconques pour voir les performances de chaque algorithme. Les réseaux sont testés avec des échantillons n'appartenant pas à la base de données.

28 échantillons ont été testés. Les résultats obtenus sont résumées ci-dessous.

- ***Méthode de Fourier-Mellin***

Le taux de reconnaissance est égal à 100% pour des déplacements variés entre 65 et 71 cm autour de la position de création de la base de données.

Pour des déplacements supérieurs à 71 cm et inférieurs à 102 cm, le tableau suivant montre la réponse de réseau.

Tableau 5.3 : Diagnostic dans la phase de test utilisant LM par la méthode de Fourier-Mellin.

échantillon		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
évaluation		**Bon**	**Pas Bon**	**Pas Bon**	**Bon**	**Pas Bon**	**Pas Bon**	**Pas Bon**	**Bon**	**Bon**	**Pas Bon**
Sorties voulu	Y_{d4}	1	1	1	0	0	0	0	0	0	1
	Y_{d3}	0	0	1	1	1	0	0	0	1	1
	Y_{d2}	0	1	0	1	0	0	1	1	0	0
	Y_{d1}	1	1	0	1	0	0	0	1	1	1
Sorties réelle	Y_4	1	1	0	0	1	0	1	0	0	1
	Y_3	0	0	0	1	0	0	0	0	1	0
	Y_2	0	0	0	1	0	0	1	1	0	0
	Y_1	1	1	0.01	1	0	0.99	0	1	1	0

échantillon		**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
évaluation		**Bon**	**Pas Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Pas Bon**	**Pas Bon**
Sorties voulu	Y_{d4}	1	1	0	0	1	1	0	1	0	0
	Y_{d3}	0	0	0	1	0	0	1	1	1	0
	Y_{d2}	0	1	0	1	1	0	1	0	0	1
	Y_{d1}	0	0	1	0	1	1	1	0	0	0
Sorties réelle	Y_4	1	1	0	0	1	1	0	1	1	1
	Y_3	0	0	0	1	0	0	1	1	1	1
	Y_2	0	1	0	1	1	0	1	0	1	0
	Y_1	0	1	1	0	1	1	1	0	0	0

échantillon		**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
évaluation		**Pas Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Pas Bon**	**Bon**
Sorties voulu	Y_{d4}	0	0	0	1	1	1	0	
	Y_{d3}	0	0	1	0	1	0	0	
	Y_{d2}	1	0	0	0	0	1	0	
	Y_{d1}	1	0	1	0	1	0	1	
Sorties réelle	Y_4	0	0	0	1	1	1	0	
	Y_3	0.58	0	1	0	1	0	0	
	Y_2	1	0	0	0	0	1	0	
	Y_1	1	0	1	0	1	0	0	

- ***Méthode de Zernike***

Les résultats de cette méthode sont obtenus pour des distances différentes entre la caméra et le plan de vision (de 43cm jusqu'à 102cm)

Tableau 5.3 : Diagnostic dans la phase de test utilisant LM par la méthode de Zernike.

échantillon		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
évaluation		**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**	**Bon**
Sorties voulu	Y_{d4}	1	1	1	1	0	0	0	0	0	1
	Y_{d3}	0	0	1	1	1	0	0	0	1	1
	Y_{d2}	0	1	0	1	0	0	1	1	0	0
	Y_{d1}	1	1	0	0	0	0	0	1	1	1
Sorties réelle	Y_4	1	1	1	1	0	0	0	0	0	1
	Y_3	0	0	1	1	1	0	0	0	1	1
	Y_2	0	1	0	1	0	0	1	1	0	0
	Y_1	1	1	0	0	0	0	0	1	1	1

échantillon		11	12	13	14	15	16	17	18	19	20
évaluation		Bon	Bon	Bon	Bon	Bon	Bon	Bon	Bon	Bon	Bon
Sorties voulu	Y_{d4}	1	1	1	0	1	1	0	1	1	0
	Y_{d3}	0	0	0	1	0	0	1	1	1	0
	Y_{d2}	0	1	0	1	1	0	1	0	1	1
	Y_{d1}	0	0	0	0	1	1	1	0	0	0
Sorties réelle	Y_4	1	1	1	0	1	1	0	1	1	0
	Y_3	0	0	0	1	0	0	1	1	1	0
	Y_2	0	1	0	1	1	0	1	0	1	1
	Y_1	0	0	0	0	1	1	1	0	0	0

échantillon		21	22	23	24	25	26	27	28
évaluation		Bon	Bon	Bon	Bon	Bon	Bon	Bon	Bon
Sorties voulu	Y_{d4}	0	0	0	1	1	1	0	0
	Y_{d3}	0	0	1	0	1	0	0	1
	Y_{d2}	1	0	0	0	0	1	0	1
	Y_{d1}	1	0	1	0	1	0	1	0
Sorties réelle	Y_4	0	0	0	1	1	1	0	0
	Y_3	0	0	1	0	1	0	0	1
	Y_2	1	0	0	0	0	1	0	1
	Y_1	1	0	1	0	1	0	1	0

On peut conclure suivant les résultats obtenus que les moments de Zernike donnent un taux de reconnaissance élevé égale à 100%, quelque soit la variation de l'échelle, de la rotation et de la translation, par contre le taux obtenu par les moments de Fourier-Mellin est moyen, environ 60,71%. Ce taux moyen est dû aux emplacements des pixels qui n'existent qu'à des positions entières, c'est-à-dire que le changement de l'échelle ou de la rotation des objets vont influencer sur la position réelle de différents pixels de l'objet.

Tous les résultats obtenus sont consigné dans le tableau ci-dessous.

Tableau 5.4 : Taux de reconnaissance pour les différentes méthodes.

Méthode	Taux de reconnaissance
Correspondance de modèle	100%
Rectangle de contenance pour 5 objets	100%
Fourier-Mellin	60.71%
Zernike	100%

5.4.Conclusion

La combinaison qui a donnée les meilleurs résultats est celle correspondant à la structure (8,7,4) du réseau et l'utilisation des moments de Zernike. C'est donc ce couple (structure, moments de Zernike) qui est adopté.

CHAPITRE 6

Le Robot Manipulateur

Le Robot Manipulateur

6.1. Introduction

L'utilisation de manipulateurs dans l'industrie n'a cessé d'augmenter ces dernières années. Ils ont permis d'augmenter la production et la précision des mouvements effectués. La plus grande partie de ces manipulateurs est constituée des manipulateurs sériels dont la morphologie peut facilement être comparée à celle de l'être humain.

Un robot est un manipulateur reprogrammable à fonctions multiples. Il est conçu pour déplacer des matériaux, des pièces, des outils ou des instruments spécialisés suivant des trajectoires variables programmées, en vue d'accomplir des tâches très diverses.

On peut considérer trois grands domaines d'application des robots. Les problèmes à résoudre à l'intérieur de chacun de ces domaines ont une certaine ressemblance, mais ils peuvent être assez différents d'un domaine à l'autre.

- ***Le domaine de la production***

C'est dans ce domaine que l'effort est porté le plus particulièrement par les industriels, qui voient de nombreux avantages dans l'utilisation des robots, notamment un accroissement de la productivité et une diminution de la main d'œuvre. En fait, l'association de robots entre eux et avec d'autres machines amène deux avantages fondamentaux par rapport aux modes de production traditionnels :

1. L'automatisation quasi intégrale de la production qui peut s'accompagner :
    - d'une meilleure qualité du produit fini
    - d'une plus grande fiabilité dans le maintien de cette qualité
    - d'une meilleure adaptation de la quantité produite à la demande
2. La flexibilité, c'est-à-dire la rapidité de reconfiguration de l'unité de production quand on passe de la fabrication d'un produit à celle d'un produit voisin (exemple : fabriquer des modèles différents de voitures sur la même chaîne).

- ***Le domaine de l'exploration***

Il s'agit d'un problème différent. On veut exécuter des opérations dans un lieu :

1. d'accès difficile : exploitation forestière, construction, maintenance, réparations ou nettoyage de bâtiments, lignes électriques, ...etc.
2. Dans un milieu hostile ou dangereux pour l'homme :
    - le milieu sous-marin
    - le milieu spatial
    - le milieu irradié des centrales nucléaires
    - les sites de catastrophes ou d'accidents

- ***Le domaine de l'assistance individuelle***

Il y a un domaine où la robotique d'assistance individuelle tend à se développer, c'est celui de la robotique médicale permettant d'améliorer les conditions de vie des personnes handicapées, paralysées ou amputées. Le système que nous avons étudié dans les chapitres précédents entre dans le domaine de la production, dans lequel on va faire la séparation des objets.

Les objectifs de cette partie sont résumés par ces points :

1. choisir une structure adéquate d'un robot
2. modélisation géométrique, cinématique et dynamique du robot choisi.
3. générations des trajectoires
4. réaliser une commande qui permet de suivre ces trajectoires.

6.2. Choix de robot

Le robot est en effet conçu pour saisir et déplacer des objets. Ceci suppose donc une structure mécanique. La maîtrise d'un objet dans l'espace implique sa localisation et son orientation. Pour localiser un point de l'objet dans l'espace, il faut disposer de 3 degrés de liberté (DDL), qui peuvent être des translations ou des rotations. Un point de l'objet étant fixe, si l'on désire orienter l'objet de manière quelconque, il faut disposer en plus de 3 autres DDL qui sont nécessairement des rotations (si possible autour d'axes orthogonaux concourants).

Les 3 premiers DDL définissent ce que l'on appelle le porteur (bras articulé), alors que les suivants constituent un poignet à l'extrémité duquel se trouve l'organe terminal du robot.

Il faut donc, et il suffit de, 6 DDL pour qu'un système soit complet (cela ne signifie pas qu'il sera universel ; mais que l'on doit rechercher pour un robot, une multifonctionnalité maximale), le robot choisi pour accomplir la tache de déplacement des objets (pièces) vers des emplacements prédétermines est le Robot PUMA560, il possède 6 articulations rotoïdes (Figure 6.1).

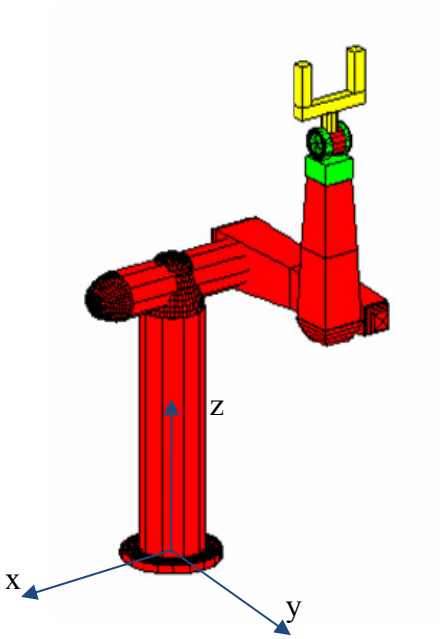


Figure 6.1: Robot Puma560

6.3. Systèmes de coordonnées

Pour qu'un robot puisse manipuler un objet, il doit pouvoir le localiser. La détermination cohérente des positions dans l'espace représente une tâche fondamentale pour tous les systèmes robotisés. C'est pourquoi des systèmes de coordonnées sont utilisés.

Les coordonnées du centre de gravité de l'objet sont toujours exprimées par rapport au repère associé à la base du robot. La figure 6.2 décrit les repères: base du robot, champ de vision, et positions de placements.

Les positions de placements des objets par rapport au repère du robot sont données par ce tableau:

Tableau 6.1 : Positions de placements des objets par rapport au repère du robot.

objet	POSITION X-Y (m)	objet	POSITION X-Y (m)
0	0.21-0.375	7	-0.15-525
1	0.03-0.375	8	-0.33-0.525
2	-0.15-0.375	9	-.51-0.525
3	-0.33-0.375	10	0.21-0.675
4	-.51-0.375	11	0.03-0.675
5	0.21-0.525	12	-0.15-0.675
6	0.03-0.525	13	-0.33-0.675

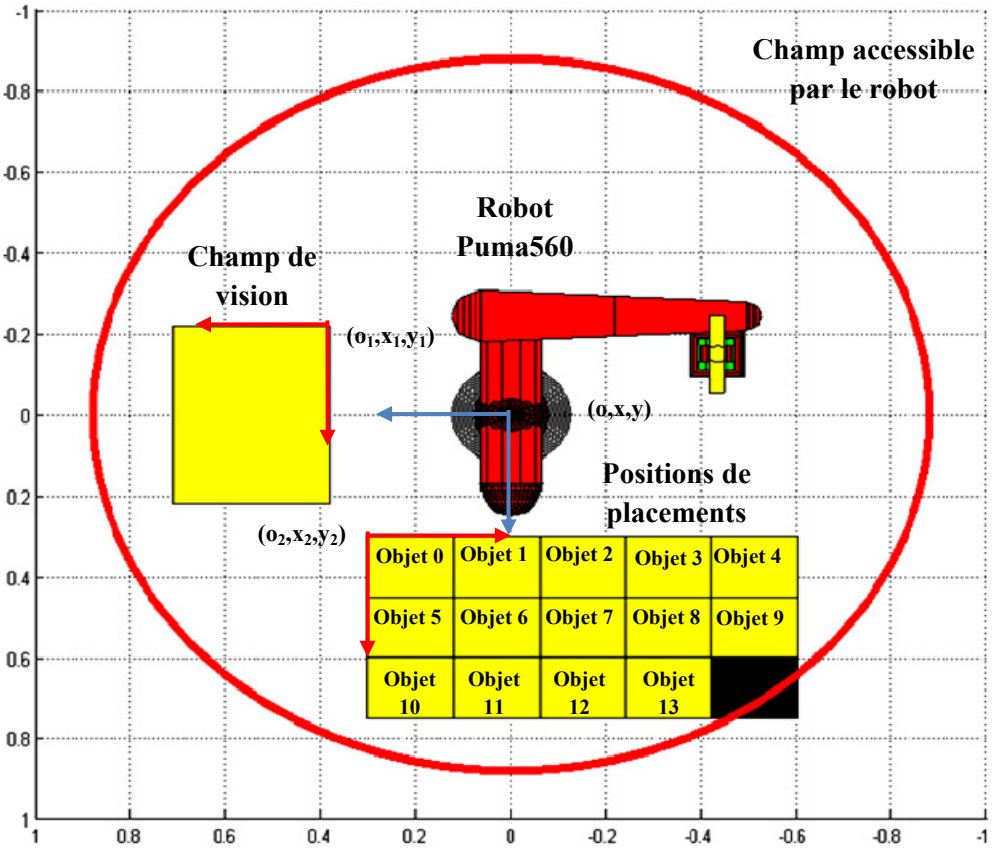


Figure 6.2: vue verticale du champ accessible par le robot.

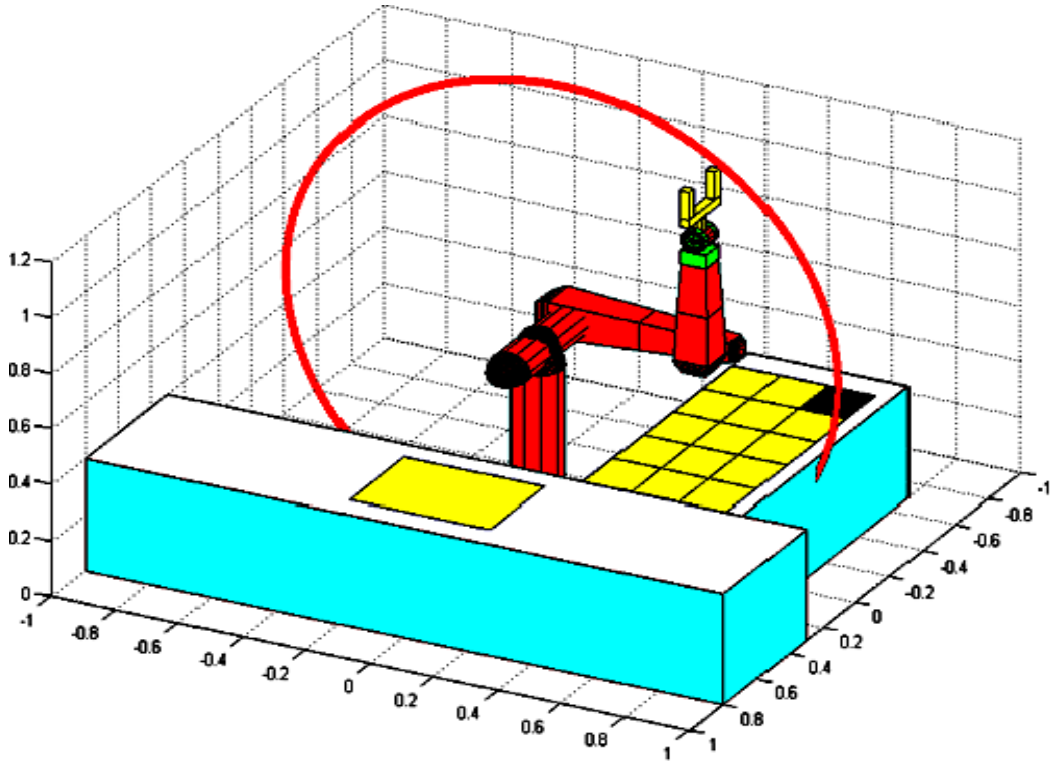


Figure 6.3: vue horizontal du champ accessible par le robot.

Le positionnement des objets est un gage de sécurité lors de l'élaboration de la trajectoire, la méthode utilisée dans notre cas consiste à utiliser la caméra comme appareil de mesure. Elle sert à définir des coordonnées caractéristiques des objets (orientation, position, largeur, longueur, axe principale).Pour passer des coordonnées de la caméra aux coordonnées du robot il faut faire un changement de base.

- ***Dimension de champ de vision par rapport à la base de robot***

Pour calculer les dimensions du champ de vision on a suivi ces étapes :

1. A l'aide d'une règle on calcul la largeur (ou la longueur) d'un objet quelconque de la base de données, on a choisi dans notre cas l'objet 11.
2. Pour le même objet choisi et à l'aide de la caméra on calcul sa largeur (ou sa longueur) en pixel.
3. Une simple application de la règle de trois donne:

$$Dimension\ en\ mètre = \frac{l_{M(objet11)}.dim_{pixel}}{l_{p(objet11)}} \qquad (6.1)$$

$l_{M(objet11)}$: Largeur (ou longueur) de l'objet 11en mètre

$l_{p(objet11)}$: Largeur (ou longueur) de l'objet 11en pixel.

Plusieurs essais ont étés effectués pour déterminer la longueur de l'objet 11, les résultats sont résumés dans le tableau suivant:

Tableau 6.2 : largeur de l'objet 11 en pixel.

Essai	résultat
1	113,846
2	114.1096
3	112.7627
4	115.9483
5	118.2582
6	119.2393
7	118.2582

La moyenne de ces résultats est donnée par:

$$l_{p(objet11)} = 116.06034\ pixel \tag{6.2}$$

La largeur de l'objet 11 en mètre est égale à

$$l_{M(objet11)} = 0.08\ m \tag{6.3}$$

On remplace dans l'équation (6.1) on trouve:

$$Dimension\ en\ mètre = 6.89.10^{-4}.dim_{pixel} \tag{6.4}$$

De l'équation (6.4) la dimension du champ de vision est égale 0.33x0.44 m
De même, la position de l'objet dans le repère du robot est donnée par:

$$X = 6.89.10^{-4}.\text{centre de gravité suivant X} + 0.38 \tag{6.5}$$

$$Y = 6.89.10^{-4}.centre\ de\ gravité\ suivant\ Y - 0.22 \tag{6.6}$$

(0.38,- 0.22) sont les cordonnées de repère (O_1, x_1, y_1) par rapport à la base du robot (O,x,y)

Tous les objets dans le champ de vision sont représentés par des parallélépipèdes dont les hauteurs sont fixées à 5cm (Figure 6.4). Pour les longueurs et les largeurs ils sont calculés à partir de l'équation(6.4).

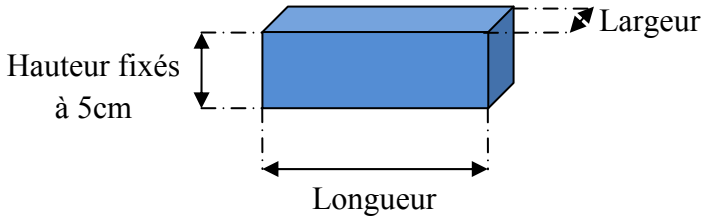


Figure 6.4: Représentation des objets par des parallélépipèdes

La hauteur du tapis roulant qui transporte les objets est égale à 0.4m, donc les coordonnées des centres de gravités de tous les objets suivant l'axe Z est fixées à 0.425m.

6.4. Modélisation

Un manipulateur est constitué de deux sous-ensembles distincts : un organe terminal et une structure mécanique articulée. L'organe terminal est le dispositif d'interaction, fixé à l'extrémité mobile de la structure mécanique. Il est désigné par organe terminal (OT) ou pince lorsqu'il s'agit d'une pince. 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.

Dans cette partie nous nous intéressons à établir les modèles géométrique, cinématique et dynamique du robot manipulateur Puma 560.

a.Modèle Géométrique Direct (MGD)

Le modèle géométrique direct (MGD) est un modèle mécanique utilisé en robotique pour les bras manipulateurs. Il permet de déterminer la configuration (position, orientation) de l'effecteur d'un robot en fonction de la configuration de ses liaisons équation(6.7) [18].

$$x = f(q) \tag{6.7}$$

- ***Convention de Denavit-Hartenberg***

Une convention couramment utilisée pour la sélection des formes de référence dans les applications robotiques est la convention Denavit-Hartenberg, ou convention de D-H [18]. Dans cette convention, chaque transformation homogène $Dh_{i-1,i}$ est représentée comme produit de quatre transformations de base:

$$Dh_{i-1,i} = Rot_{z,\theta_i} Trans_{z,d_i} Trans_{x,a_i} Rot_{x,\alpha_i} \tag{6.7}$$

$$Dh_{i-1,i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.8)$$

$$Dh_{i-1,i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & -\sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.9)$$

La matrice 3x3 formée des trois premières lignes et des trois premières colonnes représente l'orientation du repère R_i par rapport au repère R_{i-1}.Les trois premiers coefficients de la dernière colonne représentent la position de l'origine $O_i$du repère R_i par rapport à l'origine O_{i-1}du repère R_{i-1}.

- ***Paramètres de Denavit-Hartenberg***

Le paramètre d_i représente une translation le long de l'axe Z_{i-1} entre les axes x_{i-1} et x_i , le paramètre θ_i représente l'angle de rotation autour de l'axe Z_{i-1} entre les axes x_{i-1} et x_i, le paramètre a_i représente la distance minimale entre les axes Z_{i-1}et Z_i mesurée sur x_i, le paramètreα_i représente l'angle de rotation autour de l'axe $x_i$entre les axes Z_{i-1} et Z_i.

Les paramètres de Denavit-Hartenberg sont illustrés sur la figure suivante:

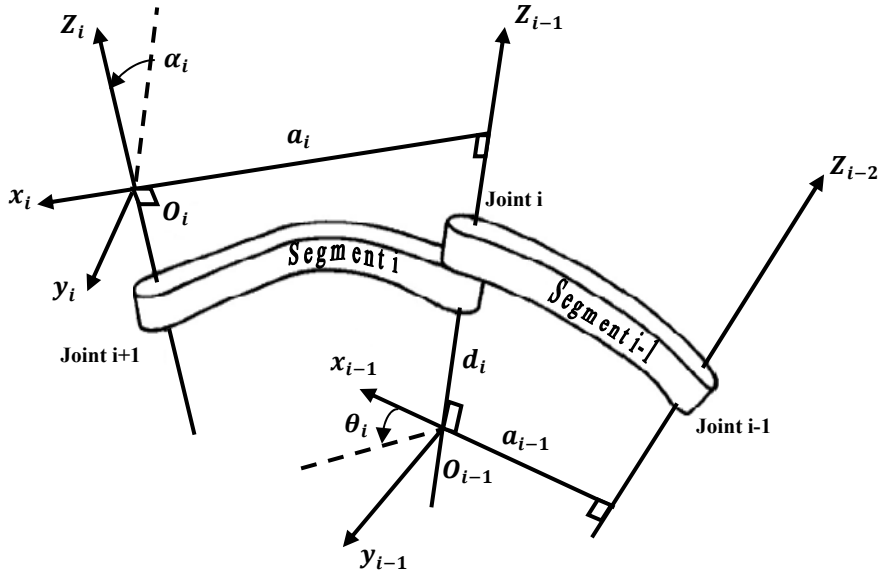


Figure 6.5: convention de Denavit-Hartenberg.

Le paramétrage du bras manipulateur Puma 560 est représenté à la figure 6.6.

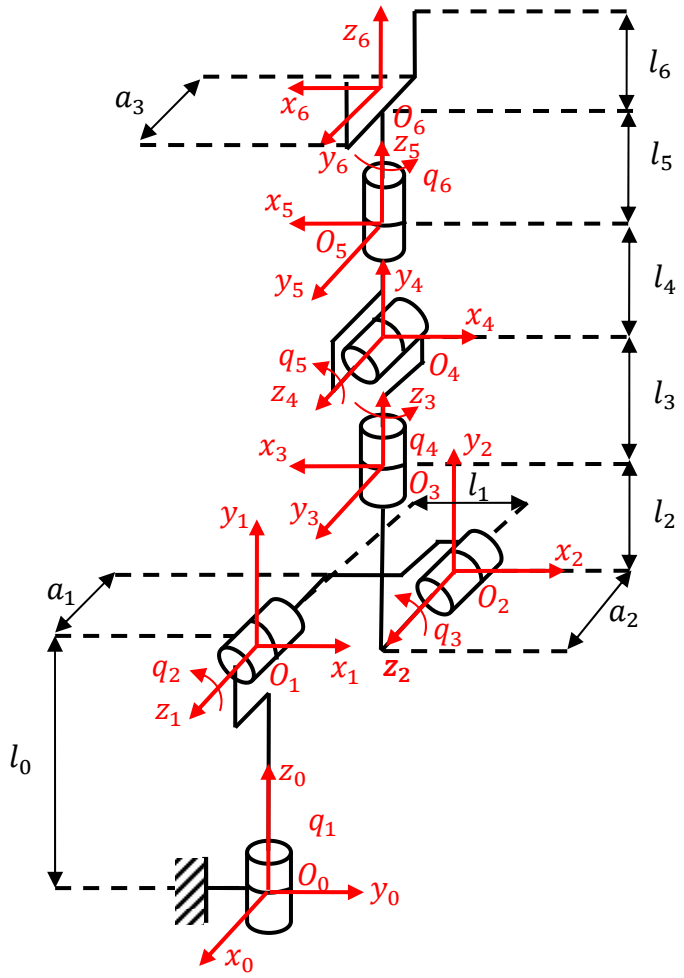


Figure 6.6: Repérage et paramétrage du bras manipulateur Puma560 selon la méthode des paramètres de Denavit-Hartenberg.

Le tableau 6.3 donne la liste des paramètres pour les 6 articulations du bras manipulateur. Le bras comporte 6 DDL de rotation ($q_1 = \theta_1, q_2 = \theta_2, q_3 = \theta_3, q_4 = \theta_4, q_5 = \theta_5, q_6 = \theta_6$).

Tableau 6.3 : Les paramètres du bras manipulateur

Corps	θ_i	a_i	α_i	d_i
1	$\frac{\pi}{2} + q_1$	0	$\frac{\pi}{2}$	l_0
2	q_2	l_1	0	$-a_1$
3	$\pi + q_3$	0	$\frac{\pi}{2}$	a_2
4	$\pi + q_4$	0	$\frac{\pi}{2}$	l_2+l_3
5	$\pi + q_5$	0	$\frac{\pi}{2}$	0
6	q_6	0	0	l_4+l_5

A partir du tableau 6.3 on peut calculer les matrices qui permettent le passage entre les articulations adjacentes :

$$Dh_{0,1} = \begin{bmatrix} -\sin q1 & 0 & \cos q1 & 0 \\ \cos q1 & 0 & \sin q1 & 0 \\ 0 & 1 & 0 & l_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.10}$$

$$Dh_{1,2} = \begin{bmatrix} \cos q2 & -\sin q2 & 0 & l_1 \cos q2 \\ \sin q2 & \cos q2 & 0 & l_1 \sin q2 \\ 0 & 0 & 1 & -a_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.11}$$

$$Dh_{2,3} = \begin{bmatrix} -\cos q3 & 0 & -\sin q3 & 0 \\ -\sin q3 & 0 & \cos q3 & 0 \\ 0 & 1 & 0 & a_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.12}$$

$$Dh_{3,4} = \begin{bmatrix} -\cos q4 & 0 & -\sin q4 & 0 \\ -\sin q4 & 0 & \cos q4 & 0 \\ 0 & 1 & 0 & l_2 + l_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.13}$$

$$Dh_{4,5} = \begin{bmatrix} -\cos q5 & 0 & -\sin q5 & 0 \\ -\sin q5 & 0 & \cos q5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.14}$$

$$Dh_{5,6} = \begin{bmatrix} \cos q6 & -\sin q6 & 0 & 0 \\ \sin q6 & \cos q6 & 0 & 0 \\ 0 & 1 & 1 & l_4 + l_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.15}$$

Comme il a été déjà motionné, pour localiser la position de l'objet présenté par le centre de gravité, il suffit d'utiliser les 3 premiers DDL définissant le porteur, les 3 derniers DDL représentant le poignet sont utilisés pour orienter l'organe terminal (la pince) du robot pour saisir l'objet.

Le calcul du MGD consiste donc à exprimer la position du point O_4 (Figure 6.6) par rapport au repère R_0, il faut pour cela multiplier les matrices de passage ($Dh_{i-1,i}$) successives reliant le repère R_0 au repère R_4 lié à O_4 :

$$Dh_{0,4} = Dh_{0,1} Dh_{1,2} Dh_{2,3} Dh_{3,4} \tag{6.16}$$

La matrice :

$$Dh_{0,4} = \begin{bmatrix} R_{0,4} & p_{0,4} \\ 0 & 1 \end{bmatrix} \tag{6.17}$$

Permet donc de connaître l'orientation du repère R_4 dans le repère de base et la position $p_{0,4} = (p_x, p_y, p_z)^T$ du point O_4.

Donc la matrice de rotation est donnée par :

$$R_{0,4} = \begin{bmatrix} -\sin q_1 \cos(q_2+q_3)\cos q_4 - \cos q_1 \sin q_4 & \sin q_1 \sin(q_2+q_3) & -\sin q_1 \cos(q_2+q_3)\sin q_4 + \cos q_1 \cos q_4 \\ \cos q_1 \cos(q_2+q_3)\cos q_4 - \sin q_1 \sin q_4 & -\cos q_1 \sin(q_2+q_3) & \cos q_1 \cos(q_2+q_3)\sin q_4 + \sin q_1 \cos q_4 \\ \sin(q_2+q_3)\cos q_4 & \cos(q_2+q_3) & \sin(q_2+q_3)\sin q_4 \end{bmatrix} \tag{6.18}$$

La position du point O_4 est donnée par les équations suivantes :

$$p_x = \sin q_1 \sin(q_2+q_3)(l_2+l_3) + \cos q_1 (a_2 - a_1) - l_1 \sin q_1 \cos q_2 \tag{6.19}$$

$$p_y = -\cos q_1 \sin(q_2+q_3)(l_2+l_3) + \sin q_1 (a_2 - a_1) + l_1 \cos q_1 \cos q_2 \tag{6.20}$$

$$p_z = \cos(q_2+q_3)(l_2+l_3) + l_1 \sin q_2 + l_0 \tag{6.21}$$

On peut constater que la position du point O_4 ne dépend pas de l'angle q_4.

b.Modèle géométrique inverse (MGI)

Le modèle géométrique direct présente l'inconvénient de déplacer l'organe terminal de façon aléatoire et imprévisible, c'est pour cela qu'on adopte le modèle géométrique inverse qui présente plus de souplesse et de contrôle. Lorsqu'on voudra commander un robot c'est les coordonnées de la position P, de l'objet, qui seront connues et il faudra déterminer les coordonnées articulaires q en conséquence [18].

A partir des coordonnées opérationnelles définies dans l'espace de la tâche, on utilise le MGI pour définir les coordonnées articulaires. Le modèle géométrique inverse est spécifié par la relation :

$$q = f^{-1}(x) \tag{6.22}$$

Après la détermination des coordonnées de centre de gravité de l'objet dans le repère de base on calcule les valeurs des variables généralisées $q_1, q_2, q_{3,}$qui amènent O_4 au point de centre de gravité. Il faut inverser les équations$(6.19 - 6.20 - 6.21)$.

Nous pouvons rencontrer trois types de difficulté:

- ***Problème géométrique***

Si le centre de gravité de l'objet ne se situe pas dans le volume atteignable du robot (Figures: 6.2-6.3), il est évident que le système d'équations $(6.19 - 6.20 - 6.21)$ est sans solution.

- ***Problème mécanique***

Pour des raisons liées à la construction mécanique du robot $q_1, q_2, q_{3,} q_4, q_5, q_6$ ne peuvent pas effectuer des rotations de 360°, donc le volume atteignable par le robot sera ainsi réduit (Figure 6.7).

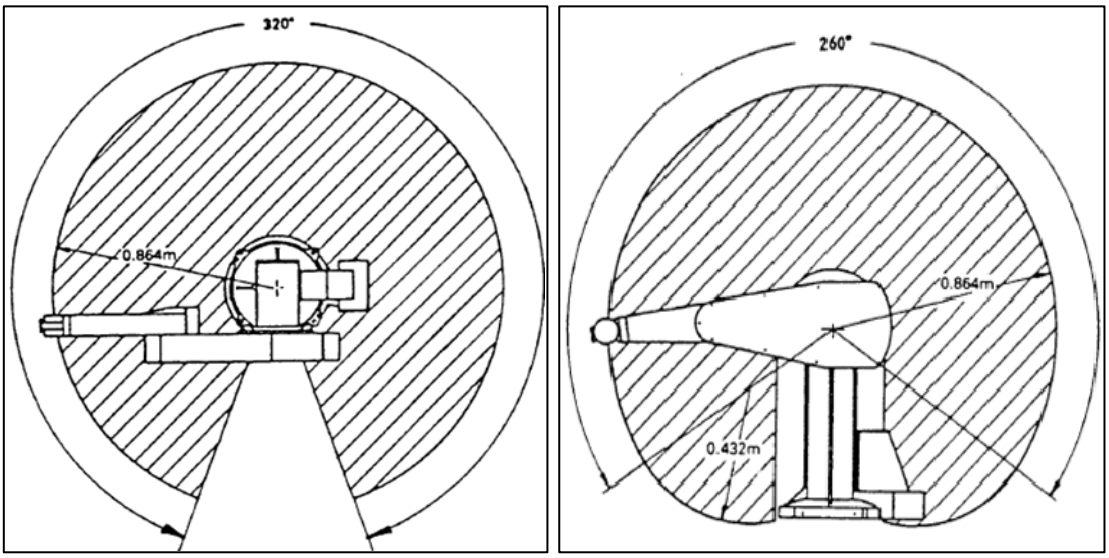


Figure 6.7: Volume atteignable du robot

Le tableau ci-dessous montre le degré de la variation de chaque articulation.

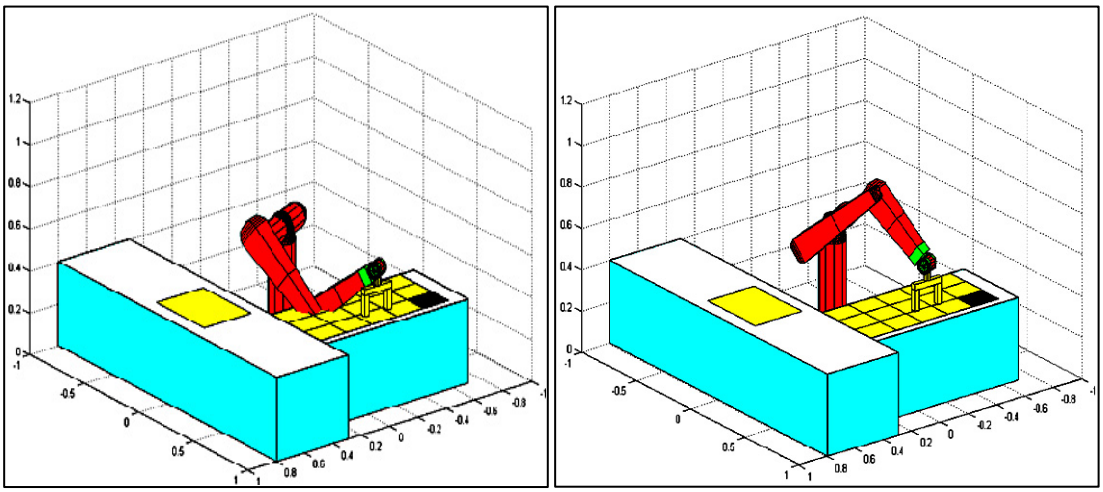
Tableau 6.3 : Le degré de variation des différentes articulations

Articulation	Angle d'orientation
1	320°
2	260°
3	284°
4	280°
5	200°
6	227°

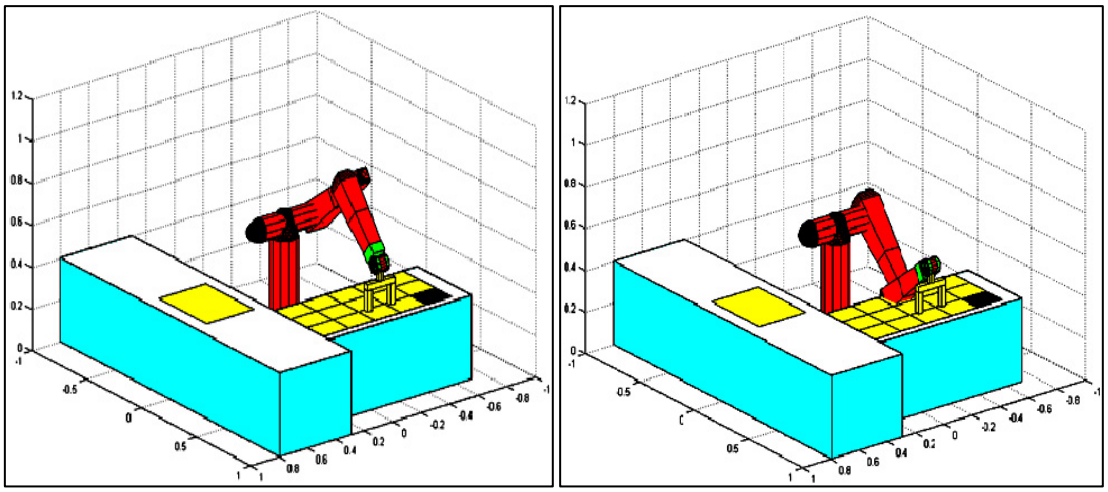
- ***Problème mathématique***

Pour une position de O_4 à l'intérieur de champ de vision ou dans le champ qui définit les positions de placement des objets, il existe plusieurs valeurs de q_1, q_2, q_3 qui remplissent les équations $(6.19 - 6.20 - 6.21)$. Seule une seule configuration est a retenir.

Si on prend comme exemple une position quelconque, que se soit dans le champ de vision ou dans le champ qui définit les positions de placement, on va trouver quatre configurations différentes permettant d'accéder à cette position.



a. Configuration -1- b. Configuration -2-



c. Configuration -3- d. Configuration -4-

Figure 6.8:Les configurations possible du robot

Parmi les quatre configurations on sélectionnera la plus convenable en s'appuyant sur deux considérations:

- Certaines positions ne sont pas mécaniquement atteignables à cause du degré de variation de $q_1, q_2, q_3, q_4, q_5, q_6$ et à cause des obstacles représentés par le tapis roulant et la table de placement des pièces.
- On prendra les solutions qui satisferont le mieux un critère. Par exemple on pourra utiliser comme critère la plus petite distance à parcourir dans l'espace des variables généralisées $q_1, q_2, q_3, q_4, q_5, q_6$ entre la position actuelle et la position désirée.

A titre d'illustration supposons que le point O_4 soit dans la position (-0.15, 0.675, 0.631) qui correspond à la figure 6.8. Les valeurs des variables généralisées $q_1, q_2, q_3, q_4, q_5, q_6$ pour chaque configuration sont données par le tableau suivant:

Tableau 6.4 : Les différentes configurations possible du robot

	Position de O_4			**Orientation de la pince**		
configuration	**q_1**	**q_2**	**q_3**	**q_4**	**q_5**	**q_6**
1	-154,77	-137.57	-166.81	0	-124.38	-154,77
2	-154.77	144.45	-12.61	0	-311.84	-154.77
3	-0.008	34.96	-166.81	0	-48.15	-0.008
4	-0.008	-41.84	-12.61	0	-125.55	-0.008

Selon les deux considérations précédentes, la solution de la configuration 3 est visiblement la plus intéressante.

Les valeurs de q_1, q_2, q_3, qui ont été choisies représentent un cas particulier d'une position, et pour d'autre cas que se soient dans le champ de vision ou dans le champ qui définit les positions de placement des objets le tableau suivant donne la variation de chaque articulation.

Tableau 6.5 : Les intervalles de variation des différentes articulations

		Position articulaire
Champ de vision		$q_1 < 0$
		$0 \leq q_2 \leq 90$
		$q_3 \leq 0$
Positions de placement	Objets 3-4-8-9-13	$0 \leq q_1$
		$0 \leq q_2 \leq 90$
		$q_3 \leq 0$
	Objets 0-1-2-5-6-7-10-11-12	$q_1 < 0$
		$0 \leq q_2 \leq 90$
		$q_3 \leq 0$

Le calcul des variables généralisées à partir d'une position P est effectué à l'aide du logiciel Maple, les étapes qu'on a suivies pour déterminer q_1, q_2, q_3 sont données comme suit:

- A partir d'un programme crée dans le logiciel Maple on fait un appel au logiciel Matlab qui contient le programme de l'identification des objets crée dans les chapitres précédentes.
- Le programme crée dans le logiciel Maple va récupérer les positions des objets qui sont à l'intérieur de champ de vision afin de déterminer les valeurs des variables généralisées q_1, q_2, q_3 qui nous donnent les positions des objets dans le champ de vision. Le calcul de q_1, q_2, q_3 se fait par l'inversion de MGD.

c.Modèle cinématique

Le modèle cinématique du bras manipulateur donne la relation entre les vitesses opérationnelles $\dot{X}$ et les vitesses généralisées $\dot{q}$ du bras manipulateur [18]:

$$\Delta X = J(q)\Delta q \quad ou \quad \dot{X} = J(q)\dot{q} \tag{6.23}$$

Où J(q) est la matrice Jacobienne de la fonction de déplacement f(q) définie dans le repère(X, Y, Z).

$$J = \begin{bmatrix} \frac{\partial f_X(q)}{\partial q_1} & \frac{\partial f_X(q)}{\partial q_2} & \cdots & \frac{\partial f_X(q)}{\partial q_n} \\ \frac{\partial f_y(q)}{\partial q_1} & \frac{\partial f_y(q)}{\partial q_2} & \cdots & \frac{\partial f_X(q)}{\partial q_n} \\ \frac{\partial f_z(q)}{\partial q_1} & \frac{\partial f_z(q)}{\partial q_2} & \cdots & \frac{\partial f_X(q)}{\partial q_n} \end{bmatrix} \quad (6.24)$$

Donc on peut obtenir le modèle cinématique par la dérivation de modèle géométrique.

La matrice Jacobienne peut être obtenue aussi en appliquant la loi de composition des vitesses :

Soit :

$$^0v_{0n}^{O_n} = {}^0v_{01}^{O_n} + {}^0v_{12}^{O_n} + \cdots + {}^0v_{n-1,n}^{O_n} = \sum_{i=1}^{n} {}^0v_{i-1,i}^{O_n} \quad (6.25)$$

Avec:

$^0v_{0n}^{O_n}$: vitesse de $O_n$dans le mouvement de R_n par rapport à R_0 exprimée dans R_0.

Si l'axe q_i est prismatique:

$$^0v_{i-1,i}^{O_n} = {}^0Z_{i-1}\dot{q}_i \quad (6.26)$$

Si l'axe q_i est rotoïde :

$$^0v_{i-1,i}^{O_n} = \dot{q}_i \, {}^0Z_{i-1} X \left(R_{0,i-1} {}^{i-1}O_n\right) \quad (6.27)$$

La matrice Jacobienne est définie par:

$$J = [J_1 \quad J_2 \quad \cdots \quad J_n] \quad (6.28)$$

Avec:

$J_i = \left[{}^0Z_{i-1}\right]$ si l'axe i est prismatique, et $J_i = \left[{}^0Z_{i-1} X \left(R_{0,i-1} {}^{i-1}O_n\right)\right]$ si l'axe i est rotoïde.

De même on peut calculer le Jacobien qui est relié entre les vitesses angulaires et les vitesses généralisées:

$$\omega = J(q)\dot{q} \quad (6.29)$$

Avec:

$J_i = [0]$ si l'axe i est prismatique, et $J_i = \left[{}^0Z_{i-1} \right]$ si l'axe i est rotoïde.

Le calcul de la solution cinématique inverse au niveau de la vitesse requiert le calcul du vecteur de la vitesse articulaire $\dot{q}$ pour une vitesse donnée $\dot{X}$ de l'organe effecteur à une configuration articulaire q.

d.Modèle dynamique

L'étude géométrique d'un robot réfère à son comportement statique et l'étude cinématique à un comportement en vitesse quasi-constante. Le modèle dynamique d'un robot désigne la fonction vectorielle D qui décrit le vecteur τ des forces et couples articulaires développés par les actionneurs, associé à la description d'une trajectoire de l'effecteur. Cette dernière est exprimée dans l'espace des coordonnées généralisées en termes de positions $q(t)$, vitesses $\dot{q}(t)$ et accélérations $\ddot{q}(t)$ articulaires [19].

La définition de cette fonction s'écrit [19]:

$$\boldsymbol{D}: \begin{cases} \mathbb{R}^{3xn} \longmapsto \mathbb{R}^n \\ [q(t)^T, \dot{q}(t)^T, \ddot{q}(t)^T]^T \longmapsto \tau = \boldsymbol{D}([q(t)^T, \dot{q}(t)^T, \ddot{q}(t)^T]^T) \end{cases} \quad (6.30)$$

Pour établir le modèle dynamique d'un robot manipulateur la fonction D est définie le plus souvent par les équations d'Euler-Lagrange.

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i \quad (6.31)$$

Avec:

- $L = K - V$ qui est appelé Lagrangien du système, K est l'énergie cinétique du système et V est l'énergie potentielle du système.
- q_i représente la ième coordonnée généralisée du système.
- τ_i est la force généralisée appliquée au ième élément du système. τ_i est un couple si l'articulation est rotoïde ou une force si l'articulation est prismatique.

L'énergie cinétique est donnée par:

$$K = \sum_{i=1}^{n} \frac{1}{2} m_i v_i^T v_i + \frac{1}{2} \omega_i^T I_i \omega_i \quad (6.32)$$

Avec:

I_i : Matrice d'inertie du corps i.

m_i : Masse du corps i.

v_i : vitesse linéaire du centre de gravité du corps i.

ω_i : vitesse angulaire du corps i.

Dès équations $(6.23 - 6.29)$ l'énergie cinétique du système peut s'écrire sous la forme:

$$K = \frac{1}{2}\dot{q}^T \sum_{i=1}^{n} [m_i J_{vi}^T(q) J_{vi}(q) + J_{\omega i}^T(q) R_{0i} I_i R_{0i}^T J_{\omega i}(q)]\, \dot{q} \qquad (6.33)$$

La vitesse de rotation est exprimée dans le même repère que celui qui a servi à la détermination de la matrice d'inertie I_i.

L'équation (6.33) peut s'écrire sous forme condensée:

$$K = \frac{1}{2}\dot{q}^T D(q)\dot{q} \qquad (6.34)$$

La matrice D(q) est symétrique définie positive de dimension nxn.

L'énergie potentielle est donnée par :

$$V = g^T \sum_{i=1}^{n} O_{gi} m_i \,(6.34) \qquad (6.35)$$

Avec :

g: Vecteur de gravité

O_{gi}: Coordonnées de centre de gravité du corps i dans le repère de base.

On peut déduire les équations d'Euler-Lagrange en définissant le lagrangien comme ce qui suit:

$$L = K - V = \frac{1}{2}\sum_{i,j} d_{ij}(q)\dot{q}_i\, \dot{q}_j - V(q) \qquad (6.36)$$

On a:

$$\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj}(q)\dot{q}_j \qquad (6.37)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj}(q)\ddot{q}_j + \sum_j \frac{d}{dt} d_{kj}(q)\, \dot{q}_j \qquad (6.38)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{kj}(q)\ddot{q}_j + \sum_{i,j}\frac{\partial d_{kj}}{\partial q_i}\dot{q}_i\dot{q}_j \tag{6.39}$$

De même :

$$\frac{\partial L}{\partial q_k} = \frac{1}{2}\sum_{i,j}\frac{\partial d_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j - \frac{\partial V}{\partial q_k} \tag{6.40}$$

Ainsi, les équations d'Euler-Lagrange deviennent :

$$\sum_j d_{kj}(q)\ddot{q}_j + \sum_{i,j}\left[\frac{\partial d_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial d_{ij}}{\partial q_k}\right]\dot{q}_i\dot{q}_j + \frac{\partial V}{\partial q_k} = \tau_k \tag{6.41}$$

En utilisant le fait que :

$$\sum_{i,j}\frac{\partial d_{kj}}{\partial q_i}\dot{q}_i\dot{q}_j = \frac{1}{2}\sum_{i,j}\left[\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j}\right]\dot{q}_i\dot{q}_j \tag{6.42}$$

On obtient :

$$\sum_{i,j}\left[\frac{\partial d_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial d_{ij}}{\partial q_k}\right]\dot{q}_i\dot{q}_j = \sum_{i,j}\frac{1}{2}\left[\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k}\right]\dot{q}_i\dot{q}_j \tag{6.43}$$

On note :

$$c_{ijk} = \frac{1}{2}\left[\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k}\right] \tag{6.44}$$

Et

$$\phi_k = \frac{\partial V}{\partial q_k} \tag{6.45}$$

Pour k fixe on a:

$$c_{ijk} = c_{jik} \tag{6.46}$$

Finalement les équations d'Euler-Lagrange deviennent :

$$\sum_J d_{kj}(q)\ddot{q}_j + \sum_{i,j} c_{ijk}(q)\dot{q}_i\dot{q}_j + \phi_k(q) = \tau_k \tag{6.47}$$

Ce qui peut s'écrire sous forme matricielle:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau \tag{6.48}$$

Où :

D : Matrice d'inertie du manipulateur.

$C(q,\dot{q})$: Vecteur de Coriolis et des couples centrifuges.

$g(q)$: Vecteur de gravité.

Dans $C(q,\dot{q})$ les termes impliquant un produit $\dot{q}_i^2$ sont appelés centrifuges. Et ceux impliquant un produit $\dot{q}_i\dot{q}_j$ avec $i \neq j$ sont les termes de Coriolis.

Les calculs de modèle dynamique du robot Puma 560 sont longs et avec des risques d'erreur, il faut beaucoup de temps pour établir les équations dynamique à la main. Dans notre cas et pour éviter ces problèmes on a développé un programme sous logiciel Maple qui fait la modélisation géométrique, cinématique et dynamique de n'importe quel robot série (voir annexe), en quelque secondes.

6.5. Résultats de la modélisation obtenue

a.Géométrie

Prenons le manipulateur montré dans la figure 6.6 avec le vecteur des coordonnées généralisées $q = [q_1\ q_2\ q_3\ q_4\ q_5\ q_6]$, soient $OC_1, OC_2, OC_3, OC_4, OC_5, OC_6$ les centres de gravités des segments par rapport à la base.

En appliquant la convention de Denavit-Hartenberg on obtient les positions des centres de gravités pour chaque segment du robot :

Segment 1 :

$$R_{0.1} = \begin{bmatrix} \cos q1 & -\sin q1 & 0 \\ \sin q1 & \cos q1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.49}$$

$$OC_1 = \begin{bmatrix} 0 \\ 0 \\ \frac{l_0}{2} \end{bmatrix} \tag{6.50}$$

Segment 2 :

$$R_{02} = \begin{bmatrix} -\sin q1 \cos q2 & \sin q1 \sin q2 & \cos q1 \\ \cos q1 \cos q2 & -\cos q1 \sin q2 & \sin q1 \\ \sin q2 & \cos q2 & 0 \end{bmatrix} \quad (6.51)$$

$$OC_2 = \begin{bmatrix} -\frac{l_1}{2} \sin q1 \cos q2 - a1 \cos q1 \\ \frac{l_1}{2} \cos q1 \cos q2 - a1 \sin q1 \\ \frac{l_1}{2} \sin q2 + l_0 \end{bmatrix} \quad (6.52)$$

Segment 3 :

$$R_{03} = \begin{bmatrix} \sin q1 \sin(q2+q3) & \sin q1 \cos(q2+q3) & \cos q1 \\ -\cos q1 \sin(q2+q3) & -\cos q1 \cos(q2+q3) & \sin q1 \\ \cos(q2+q3) & -\sin(q2+q3) & 0 \end{bmatrix} \quad (6.53)$$

$$OC_3 = \begin{bmatrix} \frac{l_2}{2} \sin q1 \sin(q2+q3) + (a2-a1)\cos q1 \\ -L1 \sin q1 \cos q2 \\ -\frac{l_2}{2} \cos q1 \sin(q2+q3) + (a2-a1)\sin q1 \\ -L1 \cos q1 \cos q2 \\ \frac{l_2}{2} \cos(q2+q3) + l_1 \sin q2 + l_0 \end{bmatrix} \quad (6.54)$$

Segment 4 :

$$R_{04} = \begin{bmatrix} \sin q1 \sin q4 \sin(q2+q3) + cos\ q1\ sin\ q4 & -\sin q1 \sin q4 \cos(q2+q3) \\ cos\ q1\ cos\ q4 & \sin q1 \sin(q2+q3) \\ -\cos q1 \cos q4 \cos(q2+q3) + sin\ q1\ sin\ q4 & \cos q1 \sin q4 \cos(q2+q3) \\ sin\ q1\ cos\ q4 & -\cos q1 \sin(q2+q3) \\ -\cos q4 \cos(q2+q3) & \sin q4 \sin(q2+q3) \\ & \cos(q2+q3) \end{bmatrix} \quad (6.55)$$

$$OC_4 = \begin{bmatrix} \frac{l_3}{2}\sin q1 \sin(q2+q3) + (a2-a1)\cos q1 \\ -l_1 \sin q1 \cos q2 + l_2 \sin q1 \sin(q2+q3) \\ -\frac{l_3}{2}\cos q1 \sin(q2+q3) + (a2-a1)\sin q1 \\ l_1 \cos q1 \cos q2 - l_2 \cos q1 \sin(q2+q3) \\ \frac{l_3}{2}\cos(q2+q3) + l_1 \sin q2 + l_0 \\ +l_2 \cos(q2+q3) \end{bmatrix} \quad (6.56)$$

Segment 5 :

R_{05}

$$= \begin{bmatrix} \sin q1 \sin q5 \cos q4 \cos(q2+q3) & \sin q1 \cos q4 \cos q5 \cos(q2+q3) & -\sin q1 \sin q4 \cos(q2+q3) \\ +cos\ q5\ sin\ q1 \sin(q2+q3) & -sin\ q5\ sin\ q1 \sin(q2+q3) & +\cos q1 \cos q4 \\ +\sin q4 \sin q5 \cos q1 & +\sin q4 \cos q1 \cos q5 & \\ -\cos q1 \sin q5 \cos q4 \cos(q2+q3) & -\cos q1 \cos q4 \cos q5 \cos(q2+q3) & \cos q1 \sin q4 \cos(q2+q3) \\ -cos\ q1\ cos\ q5 \sin(q2+q3) & +sin\ q5\ cos\ q1 \sin(q2+q3) & +\sin q1 \cos q4 \\ +\sin q4 \sin q5 \sin q1 & +\sin q4 \sin q1 \cos q5 & \\ -\cos q4 \sin q5 \sin(q2+q3) & -cos\ q4\ cos\ q5 \sin(q2+q3) & \sin q4 \sin(q2+q3) \\ +cos\ q5 \sin(q2+q3) & -sin\ q5 \cos(q2+q3) & \\ 0 & 0 & 0 \end{bmatrix} \quad (6.57)$$

$$OC_5 = \begin{bmatrix} \frac{l_4}{2}\sin q1 \sin q5 \cos q4 \cos(q2+q3) + (a2-a1)\cos q1 \\ -l_1 \sin q1 \cos q2 + l_2 \sin q1 \sin(q2+q3) + l_3 \sin q1 \sin(q2+q3) \\ \frac{l_4}{2}\cos q1 \sin q5 \sin q4 + \frac{l_4}{2}\sin q1 \cos q5 \sin(q2+q3) \\ -\frac{l_4}{2}\cos q1 \sin q5 \cos q4 \cos(q2+q3) + (a2-a1)\sin q1 \\ +l_1 \cos q1 \cos q2 - l_2 \cos q1 \sin(q2+q3) - l_3 \cos q1 \sin(q2+q3) \\ \frac{l_4}{2}\sin q1 \sin q5 \sin q4 - \frac{l_4}{2}\cos q1 \cos q5 \sin(q2+q3) \\ -\frac{l_4}{2}\cos q4 \sin q5 \sin(q2+q3) + \frac{l_4}{2}\cos q5 \cos(q2+q3) + l_0 \\ +(l_2+l_3)\cos(q2+q3) + l_1 \sin q2 \end{bmatrix} \quad (6.58)$$

Segment 6 :

R_{06}

$$= \begin{bmatrix} \sin q1 \cos q5 \cos q6 \\ * \cos q4 \cos(q2+q3) \\ -\cos q6 \sin q5 \sin q1 \sin(q2+q3) \\ -\sin q6 \sin q4 \sin q1 \cos(q2+q3) \\ +\cos q1 (\cos q6 \cos q5 \sin q4 \\ +\sin q6 \cos q4) \end{bmatrix} \begin{bmatrix} -\sin q6 \sin q1 \cos q4 \\ * \cos q5 \sin(q2+q3) \\ -\sin q6 \sin q5 \sin q1 \sin(q2+q3) \\ -\sin q4 \sin q1 \cos q6 \cos(q2+q3) \\ +\cos q1(-\sin q6 \cos q5 \sin q4 \\ +\cos q6 \cos q4) \end{bmatrix} \begin{bmatrix} \sin q5 \sin q1 \cos q4 \cos(q2+q3) \\ +\sin q1 \cos q5 \sin(q2+q3) \\ +\sin q5 \cos q1 \sin q4 \end{bmatrix}$$

$$\begin{bmatrix} -\cos q1 \cos q5 \cos q6 \\ * \cos q4 \cos(q2+q3) \\ +\cos q6 \sin q5 \cos q1 \sin(q2+q3) \\ +\sin q6 \sin q4 \cos q1 \cos(q2+q3) \\ +\sin q6 \cos q4 \sin q1 \end{bmatrix} \begin{bmatrix} \sin q6 \cos q1 \cos q4 \\ * \cos q5 \cos(q2+q3) \\ -\sin q6 \sin q5 \\ * \cos q1 \sin(q2+q3) \\ +\sin q4 \cos q1 \cos q6 \cos(q2+q3) \\ +\sin q1(-\sin q6 \cos q5 \sin q4 \\ +\cos q6 \cos q4) \end{bmatrix} \begin{bmatrix} -\sin q5 \cos q1 \cos q4 \cos(q2+q3) \\ -\cos q1 \cos q5 \sin(q2+q3) \\ +\sin q5 \sin q1 \sin q4 \end{bmatrix}$$

$$\begin{bmatrix} -\cos q6 \cos q4 \cos q5 \sin(q2+q3) \\ -\sin q5 \cos q6 \cos(q2+q3) \\ +\sin q4 \sin q6 \sin(q2+q3) \end{bmatrix} \begin{bmatrix} \sin q6 \cos q4 \cos q5 \\ * \sin(q2+q3) \\ +\sin q6 \sin q5 \cos(q2+q3) \\ \sin q4 \cos q6 \sin(q2+q3) \end{bmatrix} \begin{bmatrix} -\cos q4 \sin q5 \sin(q2+q3) \\ +\cos q5 \cos(q2+q3) \end{bmatrix}$$

$$\begin{bmatrix} 0 \end{bmatrix} \quad \begin{bmatrix} 0 \end{bmatrix} \quad \begin{bmatrix} 0 \end{bmatrix}$$

(6.59)

$$OC_6 = \begin{bmatrix} l_4 \sin q1 \sin q5 \cos q4 \cos(q2+q3) + (l_2 + l_3) \sin q1 \sin(q2+q3) \\ +\frac{l_5}{2} \cos q1 \sin q5 \sin q4 + l_4 \cos q1 \sin q5 \sin q4 \\ +\frac{l_5}{2} \sin q1 \sin q5 \cos q4 \cos(q2+q3) + (a2-a1)\cos q1 - l_1 \sin q1 \cos q2 \\ +\frac{l_5}{2} \sin q1 \cos q5 \sin(q2+q3) + l_4 \cos q5 \sin q1 \sin(q2+q3) \\ -l_4 \cos q1 \sin q5 \cos q4 \cos(q2+q3) - (l_2 + l_3) \cos q1 \sin(q2+q3) \\ +\frac{l_5}{2} \sin q1 \sin q5 \sin q4 + l_4 \sin q1 \sin q5 \sin q4 \\ -\frac{l_5}{2} \cos q1 \sin q5 \cos q4 \cos(q2+q3) + (a2-a1)\sin q1 + l_1 \cos q1 \cos q2 \\ -\frac{l_5}{2} \cos q1 \cos q5 \sin(q2+q3) - l_4 \cos q1 \cos q5 \sin(q2+q3) \\ -l_4 \sin q5 \cos q4 \sin(q2+q3) + l_1 \sin q1 + (l_2 + l_3) \cos(q2+q3) \\ -\frac{l_5}{2} \cos q4 \sin q5 \sin(q2+q3) + l_4 \cos q5 \cos(q2+q3) \\ +\frac{l_5}{2} \cos q5 \cos(q2+q3) \end{bmatrix}$$

(6.60)

b.Cinématique

- ***La vitesse linéaire***

Les vitesses linéaires des centres des gravités $OC_1, OC_2, OC_3, OC_4, OC_5, OC_6$ sont données par les matrices Jacobiennes suivantes:

Segment 1 :

$$J_{v1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.61}$$

Segment 2 :

$$J_{v2} = \begin{bmatrix} -\frac{l_1}{2}\, cos\ q1\ cos\ q2 + a1\ sin\ q1 & \frac{l_1}{2}\sin q1 \sin q2 & 0 & 0 & 0 & 0 \\ -\frac{l_1}{2}\, sin\ q1\ cos\ q2 + a1\ cos\ q1 & -\frac{l_1}{2}\cos q1 \sin q2 & 0 & 0 & 0 & 0 \\ 0 & \frac{l_1}{2}\cos q2 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.62}$$

Segment 3 :

$$J_{v3} = \begin{bmatrix} -\frac{l_2}{2}\cos q1 \sin(q2+q3) & \frac{1}{2}\sin q1\,[l_2 \cos(q2+q3) \\ +(a1-a2)\sin q1 - l_1\ \cos q1 \cos q2 & +2\ l_1 \sin q2] \\ \frac{l_2}{2}\sin q1 \sin(q2+q3) & -\frac{1}{2}\cos q1\,[l_2 \cos(q2+q3) \\ +(a2-a1)\cos q1 - l_1\ \sin q1 \cos q2 & +2\ l_1 \sin q2] \\ 0 & -l_2 \sin(q2+q3) + l_1 \cos q2 \end{bmatrix}$$

$$\begin{bmatrix} \frac{l_2}{2}\sin q1 \cos(q2+q3) & 0 & 0 & 0 \\ -\frac{l_2}{2}\cos q1 \cos(q2+q3) & 0 & 0 & 0 \\ -\frac{l_2}{2}\sin(q2+q3) & 0 & 0 & 0 \end{bmatrix} \tag{6.63}$$

Segment 4 :

$$J_{v4} = \begin{bmatrix} l_2 \cos q1 \sin(q2+q3) + \frac{l_3}{2}\cos q1 \sin(q2+q3) & \frac{1}{2}\sin q1\,[2\,(l_2+l_3) \\ +(a1-a2)\sin q1 - l_1\ \cos q1 \cos q2 & *\cos(q2+q3) + 2\ l_1\ \sin q2] \\ l_2 \sin q1 \sin(q2+q3) + \frac{l_3}{2}\sin q1 \sin(q2+q3) & -\frac{1}{2}\cos q1\,[2\,(l_2+l_3) \\ +(a2-a1)\cos q1 - l_1\ \sin q1 \cos q2 & *\cos(q2+q3) + 2\ l_1\ \sin q2] \\ 0 & -l_2 \sin(q2+q3) - \frac{l_3}{2}\sin(q2+q3) \\ & l_1\ \cos q2 \end{bmatrix}$$

$$\begin{bmatrix} -\frac{1}{2}\sin q1\,[-2\,(l_2+l_3)\cos(q2+q3)] & 0 & 0 & 0 \\ \frac{1}{2}\cos q1\,[-2\,(l_2+l_3)\cos(q2+q3)] & 0 & 0 & 0 \\ -l_2\sin(q2+q3)-\frac{l_3}{2}\sin(q2+q3) & 0 & 0 & 0 \end{bmatrix} \quad (6.64)$$

Segment 5 :

$$J_{v5}=\begin{bmatrix} \frac{l_4}{2}\sin q5\cos q1\cos q4\cos(q2+q3) & \\ +\frac{l_4}{2}\cos q1\cos q5\sin(q2+q3) & \frac{1}{2}\sin q1\,[-l_4\,\sin q5\cos q4\sin(q2+q3) \\ +(l_2+l_3)\,\cos q1\sin(q2+q3) & +l_4\cos q5\cos(q2+q3)+2\,(l_2+l_3)\cos(q2+q3) \\ +(a1-a2)\,\sin q1-l_1\,\cos q1\cos q2 & +2\,l_1\,\sin q2] \\ -\frac{l_4}{2}\sin q5\sin q1\sin q4 & \\ -\frac{l_4}{2}\sin q5\sin q1\cos q4\cos(q2+q3) & \\ +\frac{l_4}{2}\sin q1\cos q5\sin(q2+q3) & -\frac{1}{2}\cos q1\,[-l_4\,\sin q5\cos q4\sin(q2+q3) \\ +(l_2+l_3)\,\sin q1\sin(q2+q3) & +l_4\cos q5\cos(q2+q3)+2\,(l_2+l_3)\cos(q2+q3) \\ +(a2-a1)\,\cos q1-l_1\,\sin q1\cos q2 & +2\,l_1\,\sin q2] \\ +\frac{l_4}{2}\sin q5\sin q4\cos q1 & \\ 0 & -\frac{l_4}{2}\cos q4\,\sin q5\cos(q2+q3)-\frac{l_4}{2}\cos q5\sin(q2+q3) \\ & -(l_2+l_3)\sin(q2+q3)+\,l_1\,\cos q2 \end{bmatrix}$$

$$\begin{bmatrix} -\frac{1}{2}\sin q1\,[l_4\sin q5\cos q4\sin(q2+q3) & \frac{l_4}{2}\sin q5\,[-\sin q1\sin q4 \\ -l_4\cos q5\cos(q2+q3)-2\,(l_2+l_3)\cos(q2+q3)] & *\cos(q2+q3)+\cos q1\cos q4] \\ -\frac{1}{2}\cos q1\,[l_4\sin q5\cos q4\sin(q2+q3) & -\frac{l_4}{2}\sin q5\,[-\cos q1\sin q4 \\ -l_4\cos q5\cos(q2+q3)-2\,(l_2+l_3)\cos(q2+q3)] & *\cos(q2+q3)-\sin q1\cos q4] \\ -\frac{l_4}{2}\cos q4\sin q5\cos(q2+q3) & \\ -\frac{l_4}{2}\cos q5\sin(q2+q3)-\,(l_2+l_3)\cos(q2+q3) & \frac{l_4}{2}\sin q4\sin q5\sin(q2+q3) \end{bmatrix}$$

$$\begin{bmatrix} -\frac{l_4}{2}[-\cos q5\sin q1\cos q4\cos(q2+q3) & 0 \\ -\sin q1\sin q5\sin(q2+q3)-\cos q5\cos q1\sin q4] & \\ \frac{l_4}{2}[-\cos q5\cos q1\cos q4\cos(q2+q3) & 0 \\ +\cos q1\sin q5\sin(q2+q3)-\cos q5\sin q1\sin q4] & \\ \frac{l_4}{2}[-\cos q4\cos q5\sin(q2+q3)-\sin q5\cos(q2+q3)] & 0 \end{bmatrix} \quad (6.65)$$

Segment 6 :

$$J_{v6}=\begin{bmatrix}
\begin{matrix}(l_4+\frac{l_5}{2})\ \sin q5\cos q1\cos q4\cos(q2+q3)\\ +(l_4+\frac{l_5}{2})\cos q1\cos q5\sin(q2+q3)\\ +(l_2+l_3)\ \cos q1\sin(q2+q3)\\ +(a1-a2)\ \sin q1-l_1\ \cos q1\cos q2\\ -(l_4+\frac{l_5}{2})\ \sin q5\sin q1\sin q4\end{matrix} & \begin{matrix}\frac{1}{2}\sin q1\,[-(2l_4+l_5)\ \sin q5\\ *\cos q4\sin(q2+q3)\\ +(2l_4+l_5)\cos q5\cos(q2+q3)\\ +2\ (l_2+l_3)\cos(q2+q3)\\ +2\ l_1\ \sin q2]\end{matrix}\\
\begin{matrix}(l_4+\frac{l_5}{2})\ \sin q5\sin q1\cos q4\cos(q2+q3)\\ +(l_4+\frac{l_5}{2})\sin q1\cos q5\sin(q2+q3)\\ +(l_2+l_3)\ \sin q1\sin(q2+q3)\\ +(a2-a1)\ \cos q1-l_1\ \sin q1\cos q2\\ +(l_4+\frac{l_5}{2})\ \sin q5\sin q4\cos q1\end{matrix} & \begin{matrix}-\frac{1}{2}\cos q1\,[-(2l_4+l_5)\ \sin q5\\ *\cos q4\sin(q2+q3)\\ +(2l_4+l_5)\cos q5\cos(q2+q3)\\ +2\ (l_2+l_3)\cos(q2+q3)+2\ l_1\ \sin q2]\end{matrix}\\
0 & \begin{matrix}-(l_4+\frac{l_5}{2})\cos q4\ \sin q5\cos(q2+q3)\\ -(l_4+\frac{l_5}{2})\cos q5\sin(q2+q3)\\ -(l_2+l_3)\sin(q2+q3)+\ l_1\ \cos q2\end{matrix}
\end{bmatrix}$$

$$\begin{bmatrix}
\begin{matrix}-\frac{1}{2}sin\,q1\,[(2l_4+l_5)\,sin\,q5\,cos\,q4\,sin(q2+q3)\\ -(2l_4+l_5)\,cos\,q5\,cos(q2+q3)\\ -2\ (l_2+l_3)\,cos(q2+q3)]\end{matrix} & \begin{matrix}(l_4+\frac{l_5}{2})\,sin\,q5\,[-sin\,q1\,sin\,q4\\ *cos(q2+q3)+cos\,q1\,cos\,q4]\end{matrix}\\
\begin{matrix}-\frac{1}{2}cos\,q1\,[(2l_4+l_5)\,sin\,q5\,cos\,q4\,sin(q2+q3)\\ -(2l_4+l_5)\,cos\,q5\,cos(q2+q3)\\ -2\ (l_2+l_3)\,cos(q2+q3)]\end{matrix} & \begin{matrix}-(l_4+\frac{l_5}{2})\,sin\,q5\,[-cos\,q1\,sin\,q4\\ *cos(q2+q3)-sin\,q1\,cos\,q4]\end{matrix}\\
\begin{matrix}-(l_4+\frac{l_5}{2})\,cos\,q4\,sin\,q5\,cos(q2+q3)\\ -(l_4+\frac{l_5}{2})\,cos\,q5\,sin(q2+q3)\\ -(l_2+l_3)\,cos(q2+q3)\end{matrix} & (l_4+\frac{l_5}{2})\,sin\,q4\,sin\,q5\,sin(q2+q3)
\end{bmatrix}$$

$$\begin{bmatrix}
\begin{matrix}-(\mathrm{l}_4+\frac{\mathrm{l}_5}{2})[-\cos q5\sin q1\cos q4\cos(q2+q3)\\ -\sin q1\sin q5\sin(q2+q3)-\cos q5\cos q1\sin q4]\end{matrix} & 0\\
\begin{matrix}(\mathrm{l}_4+\frac{\mathrm{l}_5}{2})[-\cos q5\cos q1\cos q4\cos(q2+q3)\\ +\cos q1\sin q5\sin(q2+q3)-\cos q5\sin q1\sin q4]\end{matrix} & 0\\
(\mathrm{l}_4+\frac{\mathrm{l}_5}{2})[-\cos q4\cos q5\sin(q2+q3)-\sin q5\cos(q2+q3)] & 0
\end{bmatrix} \qquad (6.66)$$

- ***La vitesse angulaire***

La vitesse angulaire de chaque segment est donnée comme suit:

Segment 1 :

$$J_{\omega 1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.67}$$

Segment 2 :

$$J_{\omega 2} = \begin{bmatrix} 0 & \cos q1 & 0 & 0 & 0 & 0 \\ 0 & \sin q1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.68}$$

Segment 3 :

$$J_{\omega 3} = \begin{bmatrix} 0 & \cos q1 & \cos q1 & 0 & 0 & 0 \\ 0 & \sin q1 & \sin q1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.69}$$

Segment 4 :

$$J_{\omega 4} = \begin{bmatrix} 0 & \cos q1 & \cos q1 & \sin q1 \sin(q2+q3) & 0 & 0 \\ 0 & \sin q1 & \sin q1 & -\cos q1 \sin(q2+q3) & 0 & 0 \\ 1 & 0 & 0 & \cos(q2+q3) & 0 & 0 \end{bmatrix} \tag{6.70}$$

Segment 5 :

$$J_{\omega 5} = \begin{bmatrix} 0 & \cos q1 & \cos q1 & \sin q1 \sin(q2+q3) & -\sin q1 \sin q4 \cos(q2+q3) + \cos q1 \cos q4 & 0 \\ 0 & \sin q1 & \sin q1 & -\cos q1 \sin(q2+q3) & \cos q1 \sin q4 \cos(q2+q3) + \sin q1 \cos q4 & 0 \\ 1 & 0 & 0 & \cos(q2+q3) & \sin q4 \sin(q2+q3) & 0 \end{bmatrix} \tag{6.71}$$

Segment 6 :

$$J_{\omega 6} = \begin{bmatrix} 0 & \cos q1 & \cos q1 & \sin q1 \sin(q2+q3) & -\sin q1 \sin q4 \cos(q2+q3) + \cos q1 \cos q4 \\ 0 & \sin q1 & \sin q1 & -\cos q1 \sin(q2+q3) & \cos q1 \sin q4 \cos(q2+q3) + \sin q1 \cos q4 \\ 1 & 0 & 0 & \cos(q2+q3) & \sin q4 \sin(q2+q3) \end{bmatrix}$$

$$\begin{matrix} \sin q5 \sin q1 \cos q4 \cos(q2+q3) + \sin q1 \cos q5 \sin(q2+q3) + \sin q5 \cos q1 \sin q4 \\ -\sin q5 \cos q1 \cos q4 \cos(q2+q3) - \cos q1 \cos q5 \sin(q2+q3) + \sin q5 \sin q1 \sin q4 \\ -\cos q4 \sin q5 \sin(q2+q3) + \cos q5 \cos(q2+q3) \end{matrix} \Bigg]$$

(6.72)

- ***Energie cinétique***

Pour calculer l'énergie cinétique, il faut déterminer la matrice D à partir des vitesses linéaires et angulaires calculées précédemment :

$$D = \sum_{i=1}^{6} [m_i J_{vi}^T(q) J_{vi}(q) + J_{\omega i}^T(q) R_{0i} I_i R_{0i}^T J_{\omega i}(q)] \quad (6.73)$$

Avec:

m_1, m_2, m_3, m_4, m_5 et m_6 : Sont les masses de différents segments.

Les matrices d'inerties sont données comme suit:

$$I1 = \begin{bmatrix} I1_{11} & 0 & 0 \\ 0 & I1_{22} & 0 \\ 0 & 0 & I1_{33} \end{bmatrix} \quad (6.74)$$

$$I2 = \begin{bmatrix} I2_{11} & 0 & 0 \\ 0 & I2_{22} & 0 \\ 0 & 0 & I2_{33} \end{bmatrix} \quad (6.75)$$

$$I3 = \begin{bmatrix} I3_{11} & 0 & 0 \\ 0 & I3_{22} & 0 \\ 0 & 0 & I3_{33} \end{bmatrix} \quad (6.76)$$

$$I4 = \begin{bmatrix} I4_{11} & 0 & 0 \\ 0 & I4_{22} & 0 \\ 0 & 0 & I4_{33} \end{bmatrix} \quad (6.77)$$

$$I5 = \begin{bmatrix} I5_{11} & 0 & 0 \\ 0 & I5_{22} & 0 \\ 0 & 0 & I5_{33} \end{bmatrix} \quad (6.78)$$

$$I6 = \begin{bmatrix} I6_{11} & 0 & 0 \\ 0 & I6_{22} & 0 \\ 0 & 0 & I6_{33} \end{bmatrix} \quad (6.79)$$

Ainsi tous les paramètres qui interviennent dans l'expression (6.73) sont définis. Il reste uniquement à calculer la somme correspondante.

- ***Energie potentielle***

A partir de l'équation (6.35) l'énergie potentielle du robot est donnée comme suit:

$$V_1 = \frac{1}{2} m_1\, g\, L0 \tag{6.80}$$

$$V_2 = m_2\, g\, \left(\frac{l_1}{2}\, sin(q2) + l_0\right) \tag{6.81}$$

$$V_3 = m_3\, g\, \left(\frac{l_2}{2} cos(q2+q3) + l_1\, sin(q2) + l_0\right) \tag{6.82}$$

$$V_4 = m_4\, g\, \left(\left(l_2 + \frac{l_3}{2}\right) cos(q2+q3) + l_1\, sin(q2) + l_0\right) \tag{6.83}$$

$$V_5 = m_5\, g((l_2+l_3)cos(q2+q3) + l_1\, sin(q2) + l_0 - \frac{l_4}{2} cos(q4)\, sin(q5)\, sin(q2+q3)$$
$$+ \frac{l_4}{2} cos(q5) cos(q2+q3)) \tag{6.84}$$

$$V_6 = m_6\, g((l_2+l_3)cos(q2+q3) + l_1\, sin(q2) + l_0 - \frac{l_5}{2} cos(q4)\, sin(q5)\, sin(q2+q3)$$
$$- \frac{l_5}{2} cos(q5)\ cos(q2+q3) - l_4 cos(q4)\, sin(q4)\, sin(q2+q3) + l_4 cos(q5) cos(q2+q3)) \tag{6.85}$$

L'énergie potentielle totale :

$$V = V_1 + V_2 + V_3 + V_4 + V_5 + V_6 \tag{6.86}$$

- ***Equations d'Euler-Lagrange***

Finalement pour déterminer les équations d'Euler-Lagrange il suffit de calculer c_{ijk}et ϕ_k à partir des équations $(6.44 - 6.45)$.

On trouve pour ϕ_k:

$$\phi_1 = 0 \tag{6.87}$$

$$\phi_2 = \frac{1}{2} g(-2\,(l_2+l_3)\, m_6\, sin(q2+q1) - m_3 l_2\, sin(q2+q3) - m_5\, l_4\, cos\, q5\ sin(q2+q3)$$
$$-2\, m_6 l_4 \cos(q4) \sin q5\ \cos(q2+q3) - m_6 l_5 \cos q4 \sin q5 \cos(q2+q3)$$

$$-m_5 l_4 \cos q4 \sin q5 \cos(q2+q3) - m_6 l_5 \cos q5 \sin(q2+q3) - 2\, m_5\, l_2 \sin(q2+q3)$$
$$-2\, m_5 l_3 \sin(q2+q3) - 2\, m_6 l_4 \cos q5 \sin(q2+q3) - m_4 l_3 \sin(q2+q3)$$
$$-2\, m_4 l_2 \sin(q2+q3) + 2\, m_6 l_1 \cos q2 + 2\, m_5 l_1 \cos q2 + 2\, m_3\, l_1 \cos q2$$
$$+2\ m_4 l_1 \cos q2 + \ m_2 l_1 \cos q2) \qquad (6.88)$$

$$\phi_3 = \frac{1}{2} g(-2m_6(l_2+l_3)\sin(q2+q3) - m_6 l_2 \sin(q2+q3) - m_5 l_4 \cos q5 \sin(q2+q3)$$
$$-2\, m_6 l_4 \cos q4 \sin q5 \cos(q2+q3) - m_6 l_5 \cos q4 \sin q5 \cos(q2+q3)$$
$$-m_5 l_4 \cos q4 \sin q5 \cos(q2+q3) - m_6 l_5 \cos q5 \sin(q2+q3) - 2\, l_2 m_5 \sin(\mathrm{q2+q3})$$
$$-2 l_3 m_5 \sin(q2+q3) - 2\, m_6 l_4 \cos q5 \sin(q2+q3) - 2 m_4 l_2 \sin(q2+q3)$$
$$-m_4 l_2 \sin(q2+q3)) \qquad (6.89)$$

$$\phi_4 = \frac{1}{2} g(2m_6 l_4 \sin q4 \sin q5 \sin(\mathrm{q2+q3}) + m_6 l_5 \sin q4 \sin q5 \sin(\mathrm{q2+q3})$$
$$+m_5 l_4 \, sin\, q4 \, sin\, q5 \, sin(q2+q3) \qquad (6.90)$$

$$\phi_5 = \frac{1}{2} g(-m_5 l_4 \sin q5 \cos(q2+q3) - 2m_6 l_4 \cos q4 \cos q5 \sin(q2+q3)$$
$$-m_6 l_5 \cos q4 \cos q5 \sin(q2+q3) - m_5 l_4 \cos q4 \cos q5 \sin(q2+q3)$$
$$-m_6 l_5 \sin q5 \sin(q2+q3) - 2m_6 l_4 \sin q5 \cos(q2+q3) \qquad (6.91)$$

$$\phi_6 = 0 \qquad (6.92)$$

Comme pour la matrice D les résultats du calcul des termes c_{ijk} n'ont pas été expicités à cause des calculs qui sont très longs.

6.6. Commande du robot

Les équations d'un robot manipulateur sont fortement non linéaires et couplées. Physiquement, les termes d'accouplement représentent des couples gravitationnels qui dépendent de la configuration du robot, des couples de réaction dues à l'accélération des autres articulations, les couples de Coriolis et les couples centrifuges. La signification de ces couples d'interaction dépend des paramètres physiques du manipulateur et la charge qu'il porte [20].

La conception de système de commande est évidemment compliquée en considérant ces propriétés des équations dynamiques.

La commande des robots manipulateurs exige la détermination efficace des couples appliquées à chaque actionneur pour chaque point de consigne sur une trajectoire prévue en temps réel [20].

a.Commande par la dynamique inverse

L'idée de la dynamique inverse est de rechercher une loi de commande non linéaire telle que [18]:

$$\tau = f(q, \dot{q}, t) \tag{6.93}$$

Lorsqu'on remplace l'équation (6.93) dans l'équation dynamique (6.48), le résultat en boucle fermée est un système linéaire. En inspectant l'équation (6.48) on constate que si nous choisissons τ selon l'équation:

$$\tau = D(q)a_q + C(q, \dot{q})\dot{q} + g(q) \tag{6.94}$$

On trouve à partir des équations (6.48) et (6.94) que:

$$\ddot{q} = a_q \tag{6.95}$$

Le terme a_q représente une nouvelle entrée au système choisi.

La loi de commande non linéaire (équation 6.94) est appelée la commande par la dynamique inverse, elle réalise un résultat assez remarquable, à savoir le nouveau système (6.95), qui est un système linéaire découplé. Cela signifie que chaque entrée a_q peut être conçue de manière à contrôler un système linéaire scalaire. En outre, en supposant que a_{qk} est une fonction seulement de q_k et de ses dérivés, alors a_{qk} affectera q_k indépendamment du mouvement des autres liens [18].

Puisque a_{qk} peut maintenant être conçu pour contrôler un système linéaire de deuxième ordre, on peut le choisir comme suit [18] :

$$a_q = \ddot{q}^d + k_1(\dot{q}^d - \dot{q}) + k_0(q^d - q) \tag{6.96}$$

Où k_0 et $k_1$sont des matrices diagonales de gains constants et des dimensions nxn

En remplaçant a_q par sa valeur on trouve:

$$(\ddot{q}^d - \ddot{q}) + k_1(\dot{q}^d - \dot{q}) + k_0(q^d - q) = 0 \tag{6.97}$$

$$\ddot{e}(t) + k_1\dot{e}(t) + k_0 e(t) = 0 \tag{6.98}$$

Un choix simple pour les matrices de gain k_0 et k_1 est:

$$k_0 = diag\{\omega_1^2, \dots, \omega_n^2\} \quad (6.99)$$

$$k_1 = diag\{2\xi_1\omega_1, \dots, 2\xi_n\omega_n\} \quad (6.100)$$

Si on remplace dans l'équation (6.98) on trouve:

$$\ddot{e}(t) + 2\xi_n\omega_n\dot{e}(t) + \omega_n^2 e(t) = 0 \quad (6.101)$$

C'est un système de deuxième ordre en boucle fermé, qui est totalement découplées.ξ_n et ω_n sont des nombre réels positifs. ω_n est appelée pulsation propre du système; ξ_n est appelée coefficient ou facteur d'amortissement.

La figure 6.9 illustre la notion de boucle intérieure / boucle extérieure de commande. Le calcul de la commande non linéaire donnée par l'équation (6.94) est effectué dans une boucle interne. La boucle externe dans le système sert au calcul de l'entrée complémentaire a_q.

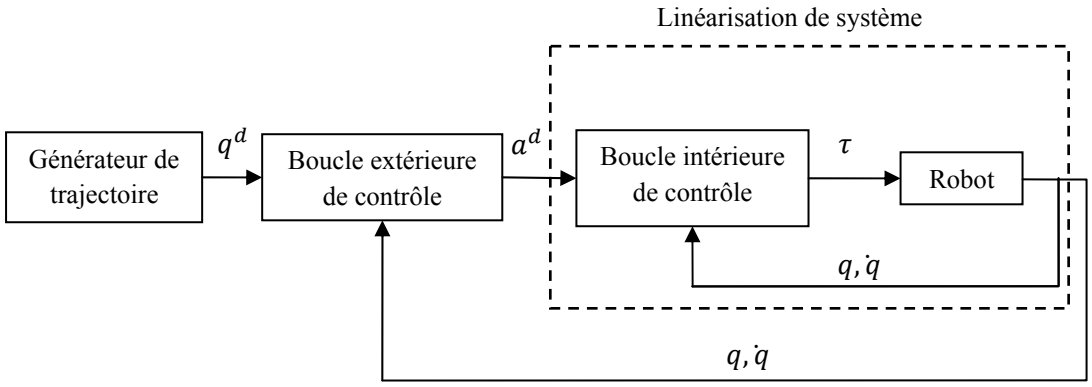


Figure 6.9 : Architecture de la boucle intérieure / boucle extérieure de commande.

b.Création des trajectoires

La tâche désirée dans notre application consiste à aller saisir les objets dans le champ de vision et les portées vers leurs positions appropriées. Le problème majeur à surmonter est la multitude de trajectoires permettant d'accomplir cette tâche. Pour résoudre ce problème il est très important de créer des points de passage. La figure 6.10 en donne un exemple.

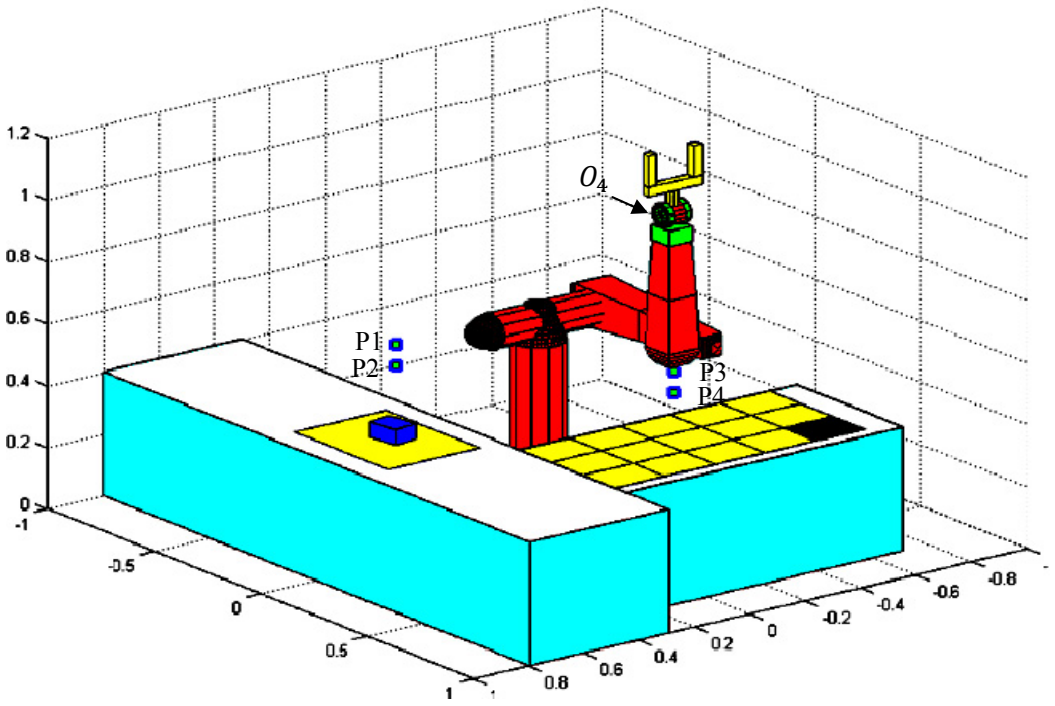


Figure 6.10 : Les points de passage.

Pour ramener les objets présents dans le champ de vision, un seul dan le cas de la figure, à leurs positions correspondantes on procède ainsi :

1. A l'aide de la commande choisi il faut déplacer le point O_4 du poignet vers la position P1 de coordonnée $(x1, y1, 0.7)$, où x1 et y1 sont les coordonnée du centre de gravité de l'objet.
2. Ensuite le point O_4 est déplacé vers la position P2 de coordonnée $(x1, y1, 0.631)$ pour saisir l'objet.
3. Ramener le point O_4 à la position P1, puis vers la position P3 de coordonnée $(x2, y2, 0.7)$. où x2 et y2 sont les coordonnée de centre de gravité de l'objet dans la zone qui contient la position de placement.
4. A la fin le point O_4 est déplacé vers la position P4 de coordonnée $(x2, y2, 0.631)$ pour déposer l'objet, ensuite on ramène le point O_4 à la position précédente.

Une fois les points de passage crées, une suite de vecteur X est générer (suite de positions du point O_4dans le repère de base R_0). Elles constituent une description de la trajectoire qu'on désire faire suivre au robot. Le programme écrit sous Maple devra traduire ces $X_{P1}, X_{P2}, X_{P1}, X_{P3}, X_{P4}, X_{P3}$ en une succession de q_1, q_2 et q_3 à l'aide du

transformateur de coordonnées inverse f^{-1}. Les articulations q_4, q_5 et q_6 peuvent être déterminées à partir de la caméra et les articulations q_1, q_2 et q_3 (figure (xxx)).

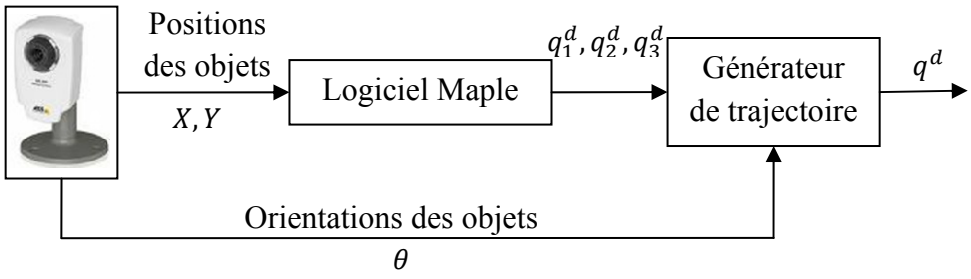


Figure 6.10: Méthode de détermination des coordonnées généralisées.

Les trois premières articulations q_1^d, q_2^d et q_3^d qui donnent la trajectoire du point O_4 sont données comme suit:

$$q_1^d(t) = q_1(t_i) + (q_1^d - q_1(t_i))(1 - e^{-T_0(t-t_i)}), \qquad t \in [t_i, +\infty[\tag{6.102}$$

$$q_2^d(t) = q_2(t_i) + (q_2^d - q_2(t_i))(1 - e^{-T_0(t-t_i)}), \qquad t \in [t_i, +\infty[\tag{6.103}$$

$$q_3^d(t) = q_3(t_i) + (q_3^d - q_3(t_i))(1 - e^{-T_0(t-t_i)}), \qquad t \in [t_i, +\infty[\tag{6.104}$$

Les trois dernières articulations qui définissent l'orientation de l'organe terminale sont données comme suit:

$$q_4^d(t) = 0, \qquad t \in [t_i, +\infty[\tag{6.105}$$

$$q_5^d(t) = q_5(t_i) + (-\pi - q_2(t) - q_3(t) - q_5(t_i))(1 - e^{-T_0(t-t_i)}), \qquad t \in [t_i, +\infty[\tag{6.106}$$

$$q_6^d(t) = q_6(t_i) + (\theta + q_1(t) - q_6(t_i))(1 - e^{-T_0(t-t_i)}), \qquad t \in [t_i, +\infty[\tag{6.107}$$

T_0 est fixé à 0.8 et $(t-t_0)$ à 6sec

c.Résultats de la simulation

La simulation, quant à elle, se présente comme une visualisation animée du comportement de robot. Le gain majeur de cette simulation réside dans la possibilité de prévoir et donc de corriger, avant la descente sur site, les défauts éventuels qui pourraient conduire à une mise en service longue et coûteuse des outils de production.

La simulation du robot Puma 560 à été crée à partir des équations géométriques qui nous donnent la possibilité d'animer le robot figure (6.11-6.12)

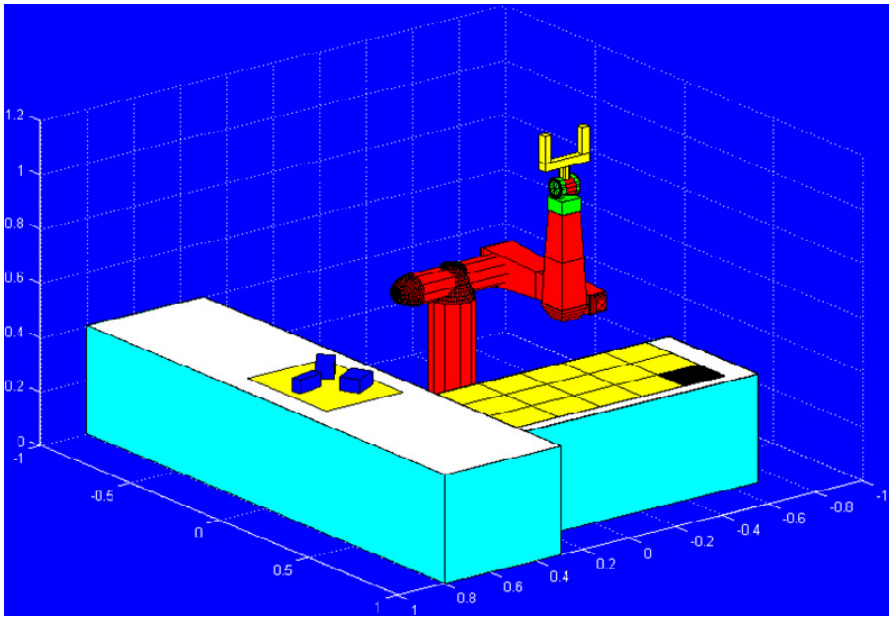


Figure 6.11: Image de la simulation avant le déplacement des objets.

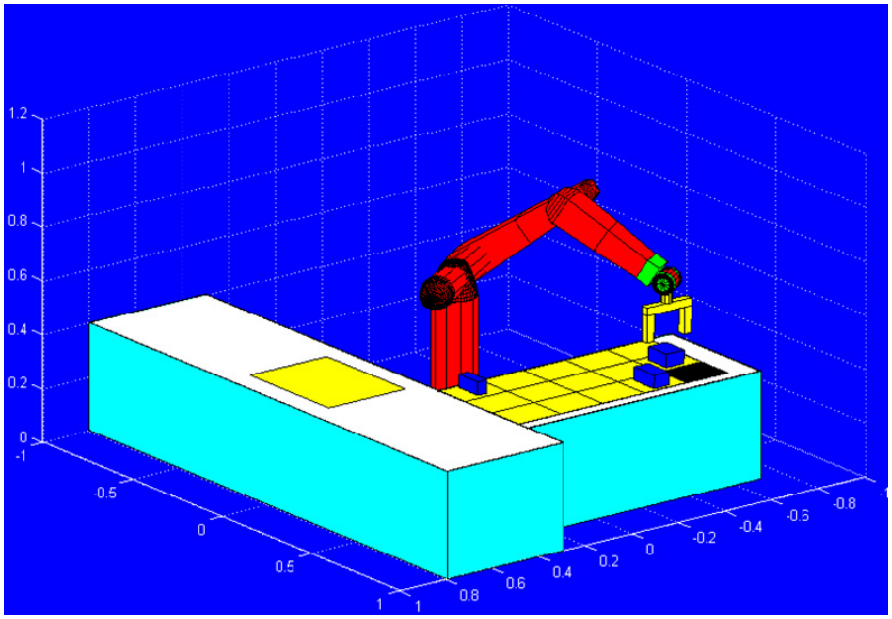


Figure 6.12: Image de la simulation après le déplacement des objets.

Le déplacement des objets est effectué par le suivi d'une trajectoire définie par les articulations $q_1^d, q_2^d, q_3^d, q_4^d, q_5^d, q_6^d$. Les figures suivantes montre le suivi des $q_1^d, q_2^d, q_3^d, q_4^d, q_5^d, q_6^d$ en utilisant la commande par la dynamique inverse.

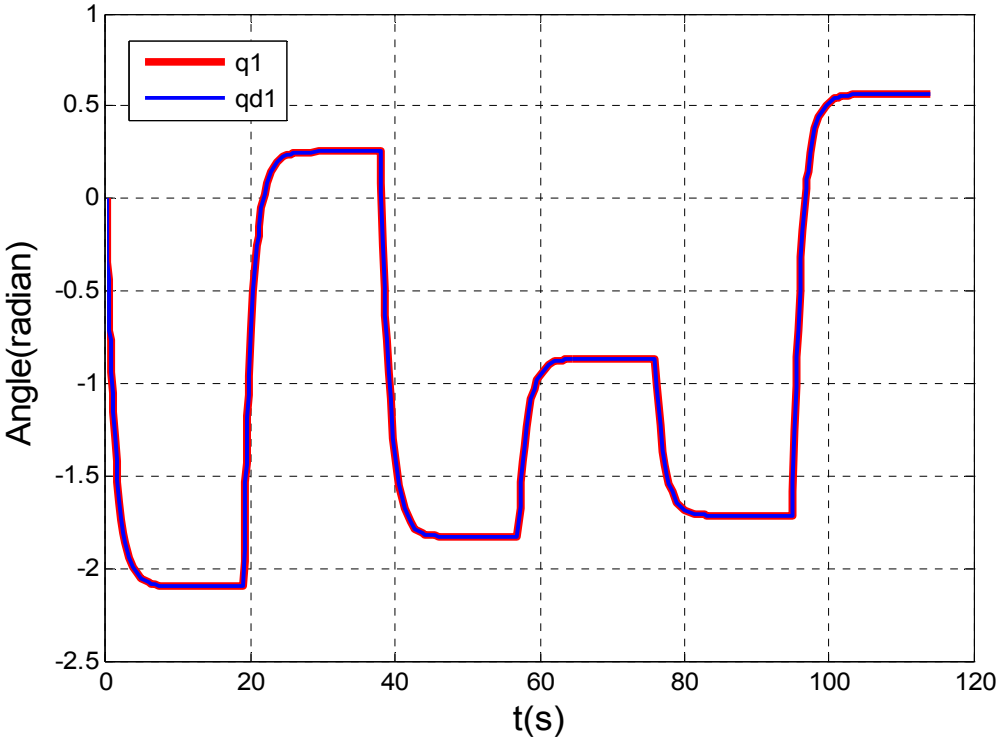


Figure 6.12: Position de l'articulation q_1

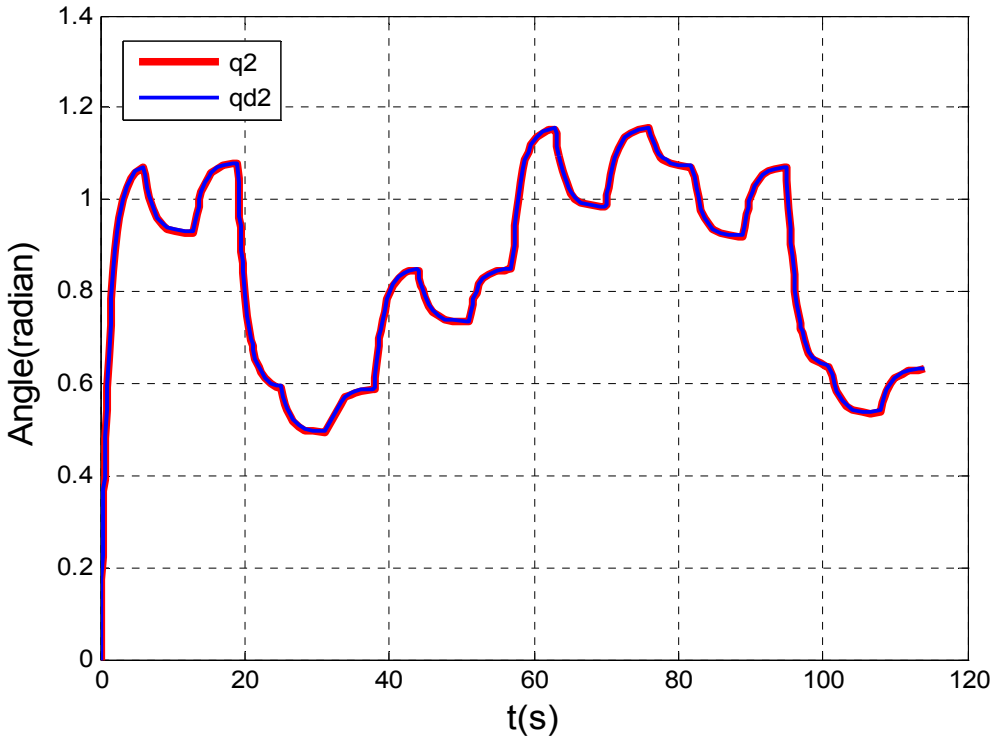


Figure 6.13: Position de l'articulation q_2

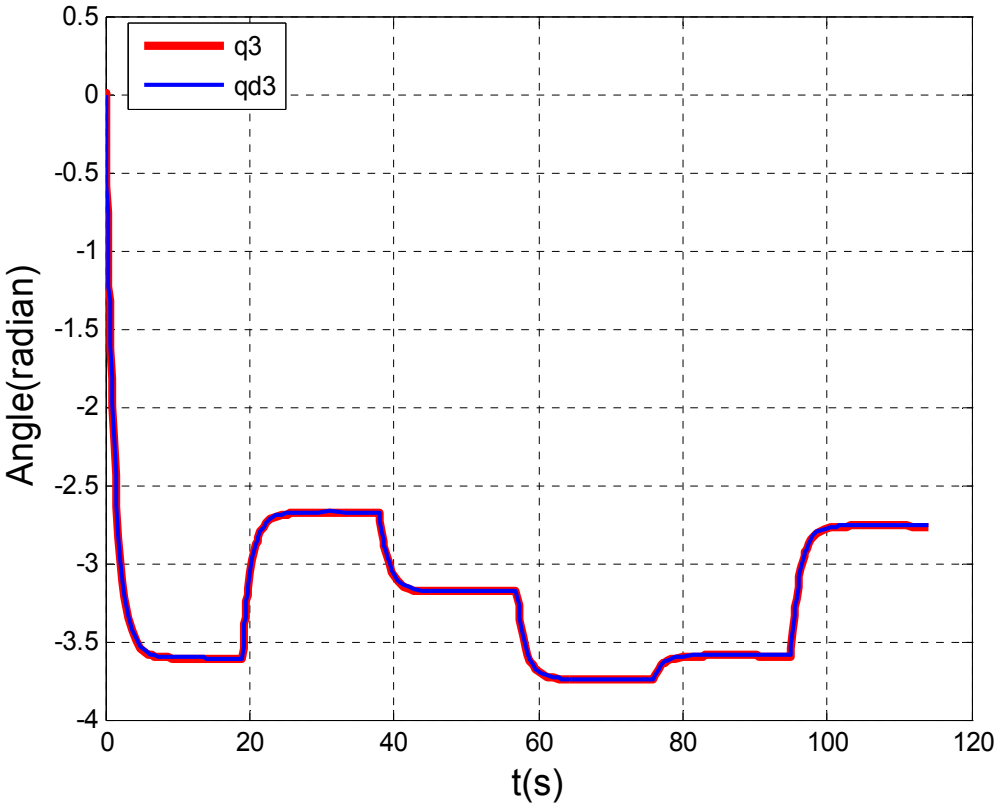


Figure 6.14: Position de l'articulation q_3

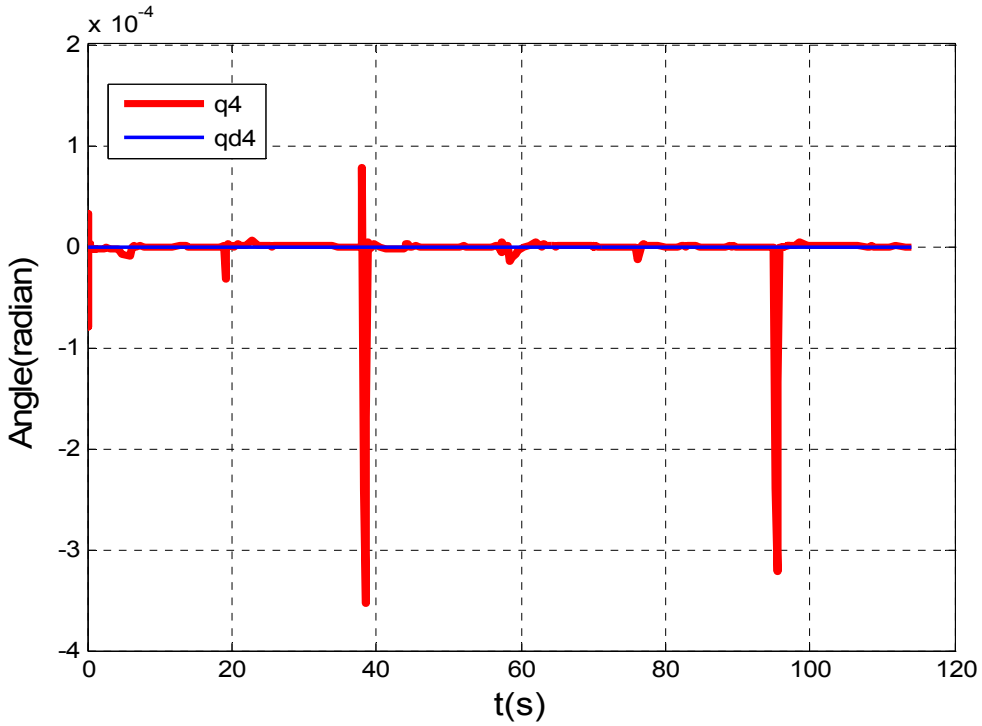


Figure 6.15: position de l'articulation q_4

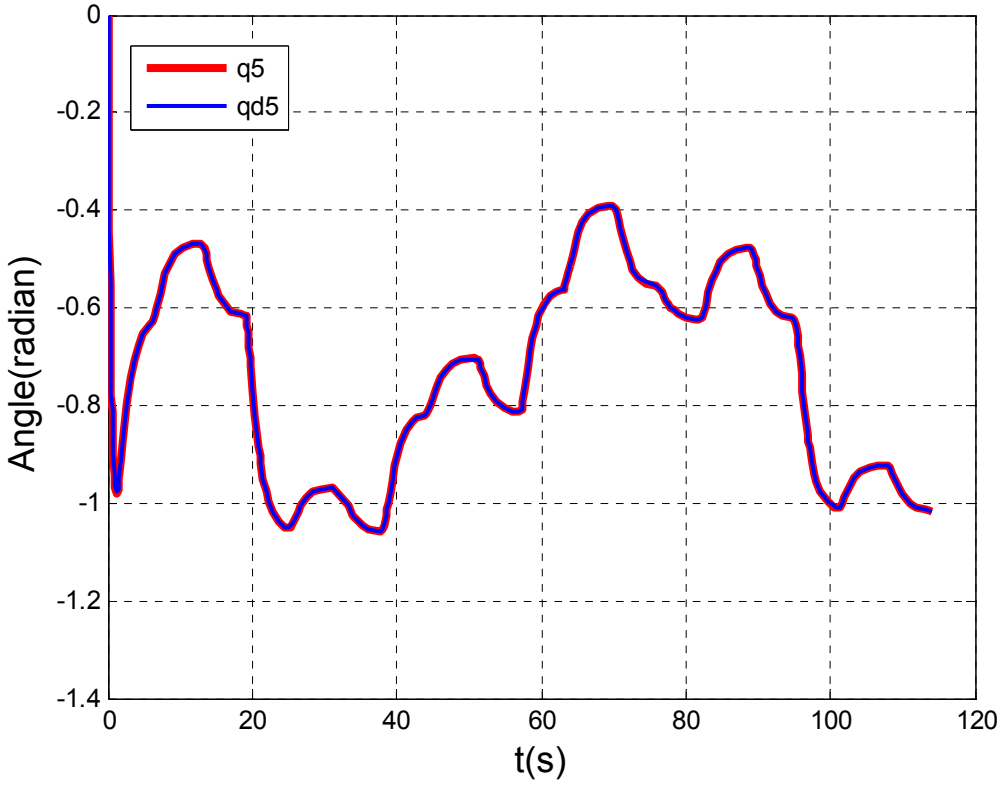


Figure 6.16: Position de l'articulation q_5

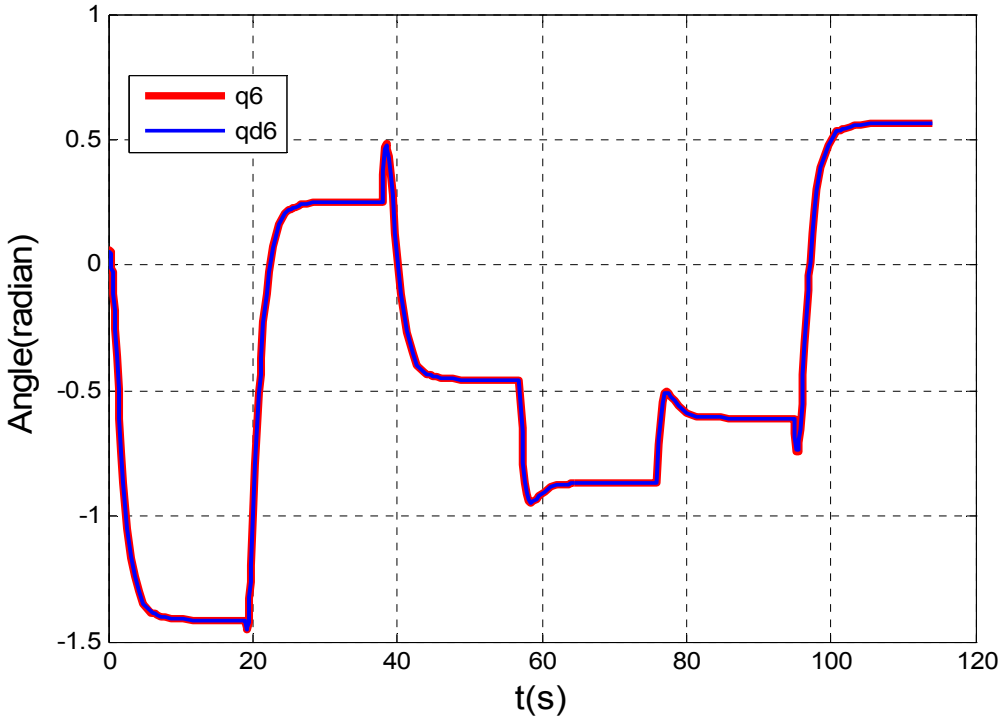


Figure 6.17: position de l'articulation q_6

6.7. Conclusion

Ce chapitre a présenté les modèles géométrique, différentiel (cinématique) et dynamique du bras manipulateur Puma 560. Ces modèles sont utilisés pour le développement de la commande par la dynamique inverse dont l'objectif principal est de placer le manipulateur dans la meilleure configuration possible pour effectuer la tâche finale de saisir et de déplacer les objets.

Conclusion Générale

Conclusion générale

L'objectif assigné à ce travail était d'élaborer un système à base d'un bras manipulateur qui peut déplacer des objets après un traitement de reconnaissance.

La première étape de travail à porté sur l'acquisition de l'image comportant les objets a travers une caméra de type Webcam CCD qui assure la transformation des objets physique en une image numérique sous forme du Bitmap. De ce fait, il a été nécessaire de mettre au point des programmes effectuant des opérations de prétraitement: seuillage pour réduire la quantité d'information traité, filtrage pour éliminer le bruit dans l'image et normalisation pour appliquer les différentes méthodes de reconnaissance.

Dans la seconde étape on a utilisé deux méthodes corrélatives pour identifier les objets dans l'image. Les inconvénients de ces méthodes sont la sensibilité au facteur d'échelle, au changement d'orientation et au changement de la position dans le cas de la méthode de Correspondance de modèle. Pour le cas de la méthode de rectangle de contenance le seul inconvénient rencontré et le nombre réduit des objets à identifier.

Par la suite, on a extrait les paramètres pertinents pour crées une base de donnée dont le but de l'utilisée pour identifier les formes des objets par l'application des réseaux de neurones. Dans ce cadre, deux méthodes d'extractions des paramètres pertinents sont utilisées: méthode des moments de Fourier-Mellin et méthode des moments de Zernike, ce qui fait que chaque objet est représenté par un ensemble de moments. Les moments de Zernike donne de bien meilleurs résultats au sens de la classification.

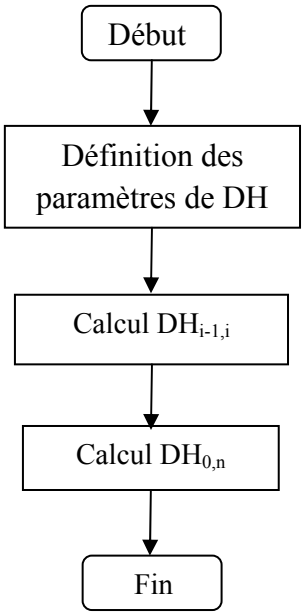
Dans la dernière partie concernant la modélisation et la commande du robot Puma 560, on a commencé en premier lieu par la détermination de modèle géométrique, cinématique et dynamique du robot en utilisant le logiciel Maple afin d'appliquer la commande qui va permettre au bras de se déplacer suivant une trajectoire prédéterminé. Le travail est achevé par une simulation qui nous permet de visualisé la réussite du déplacement des objets.

En termes de perspective, l'un des axes principaux d'amélioration du résultat de cette étude est d'utilisé des objets couleurs tridimensionnels où la face de l'objet opposé à la camera peut changer.

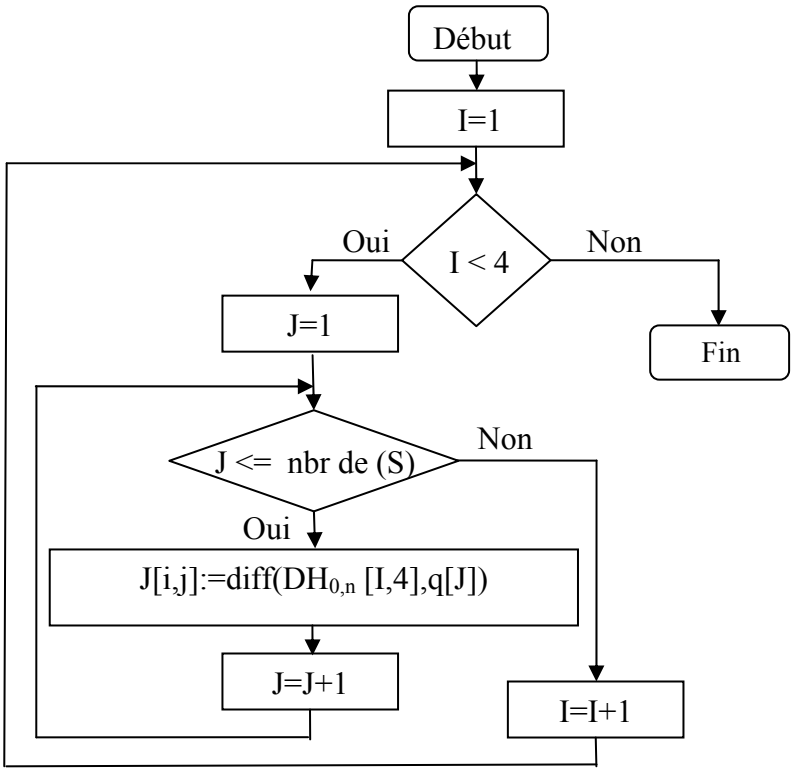
Annexes

Modélisation du robot
Organigramme

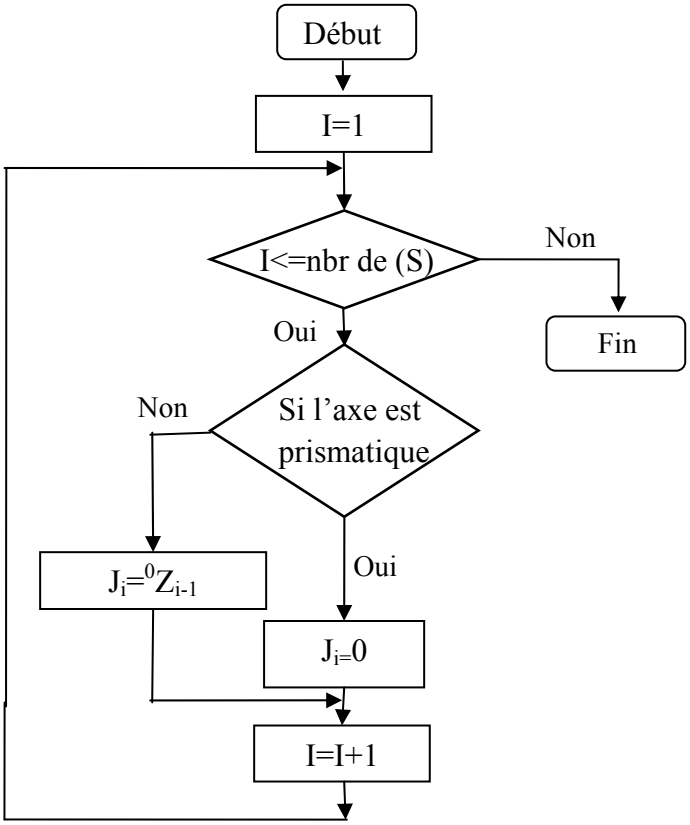
- L'organigramme qui nous permet de déduire la position et l'orientation de l'organe terminale est donné comme suit :



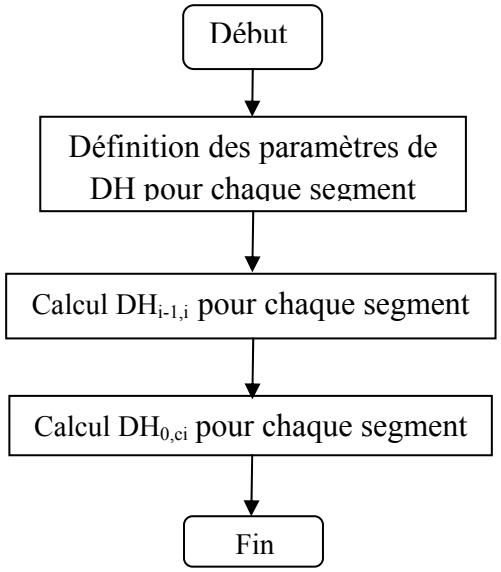
- L'organigramme qui nous permet de déterminer la vitesse linéaire est donné comme suit :



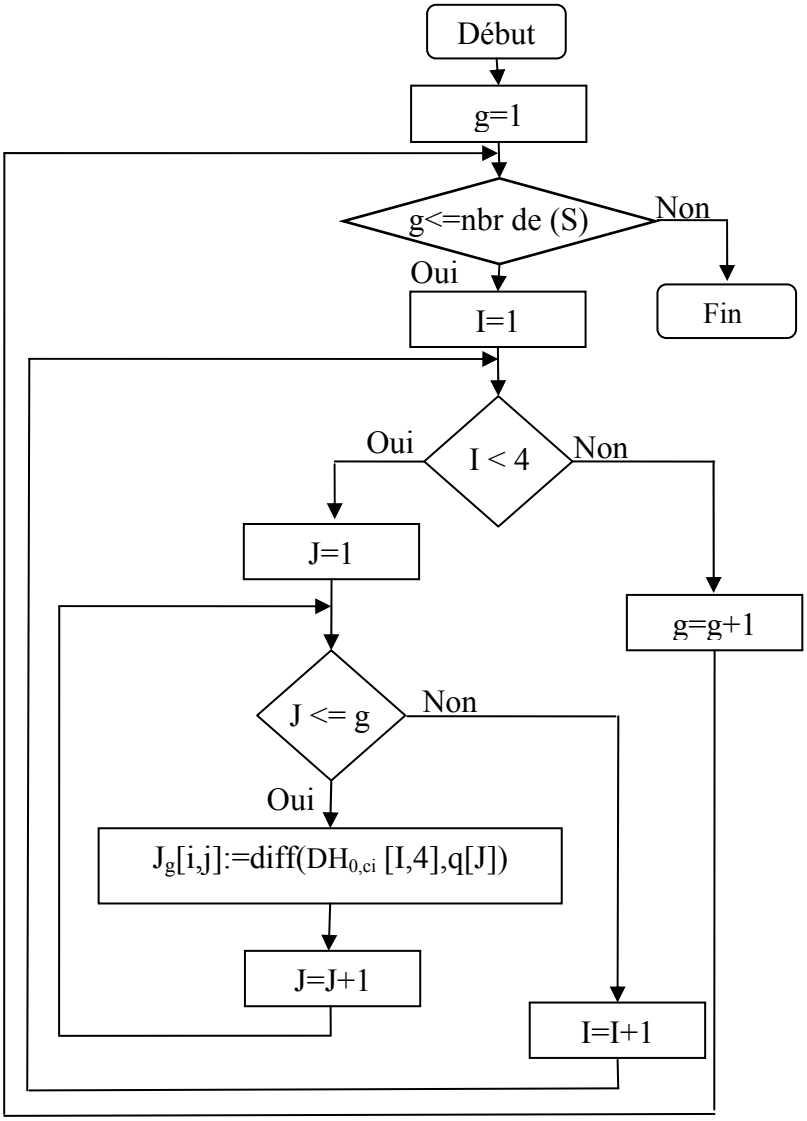
- L'organigramme qui nous permet de déterminer la vitesse angulaire est donné comme suit :



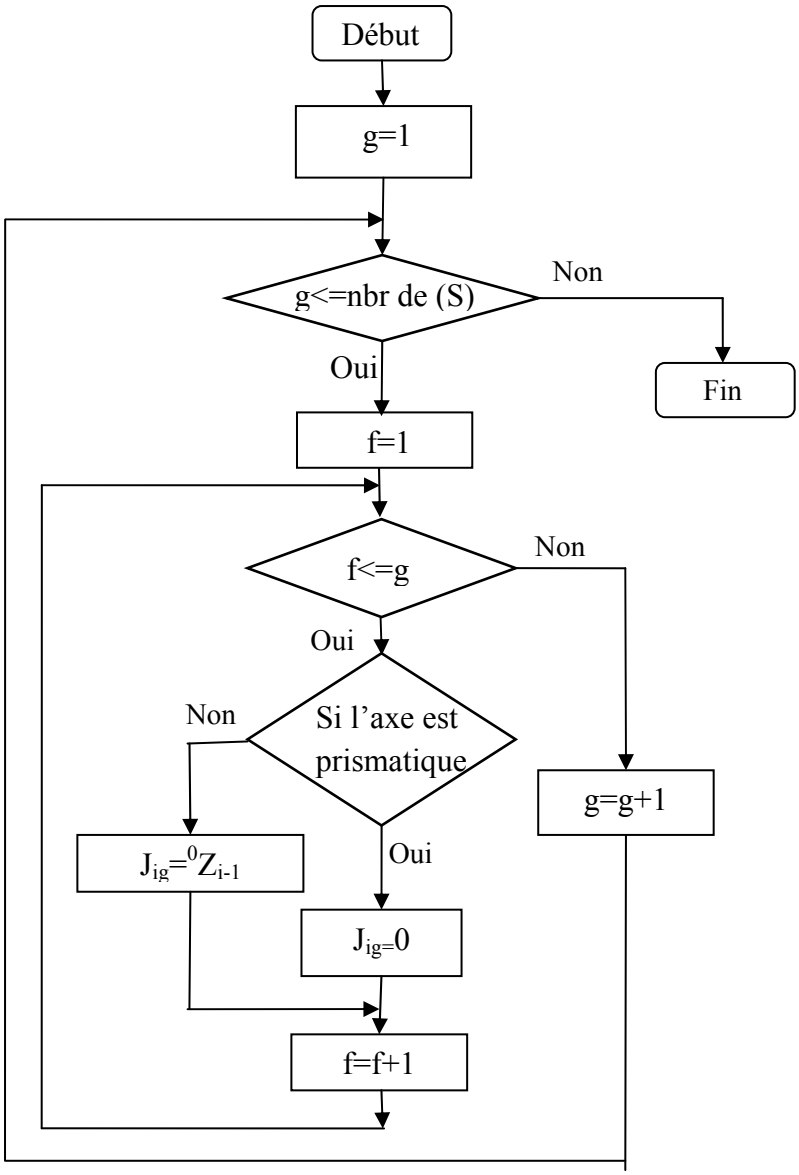
- L'organigramme qui nous permet de calculer la position de Centre de gravité de chaque solide par rapport à l'origine est donné comme suit :



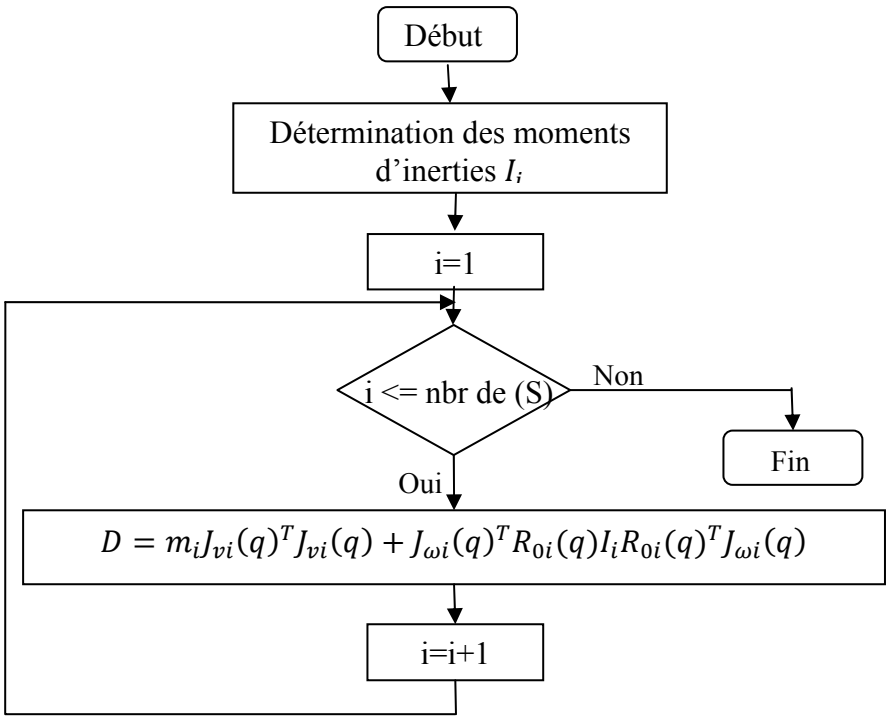
- L'organigramme qui nous permet de calculer La vitesse linéaire de centre de graviter de chaque solide est donné comme suit :



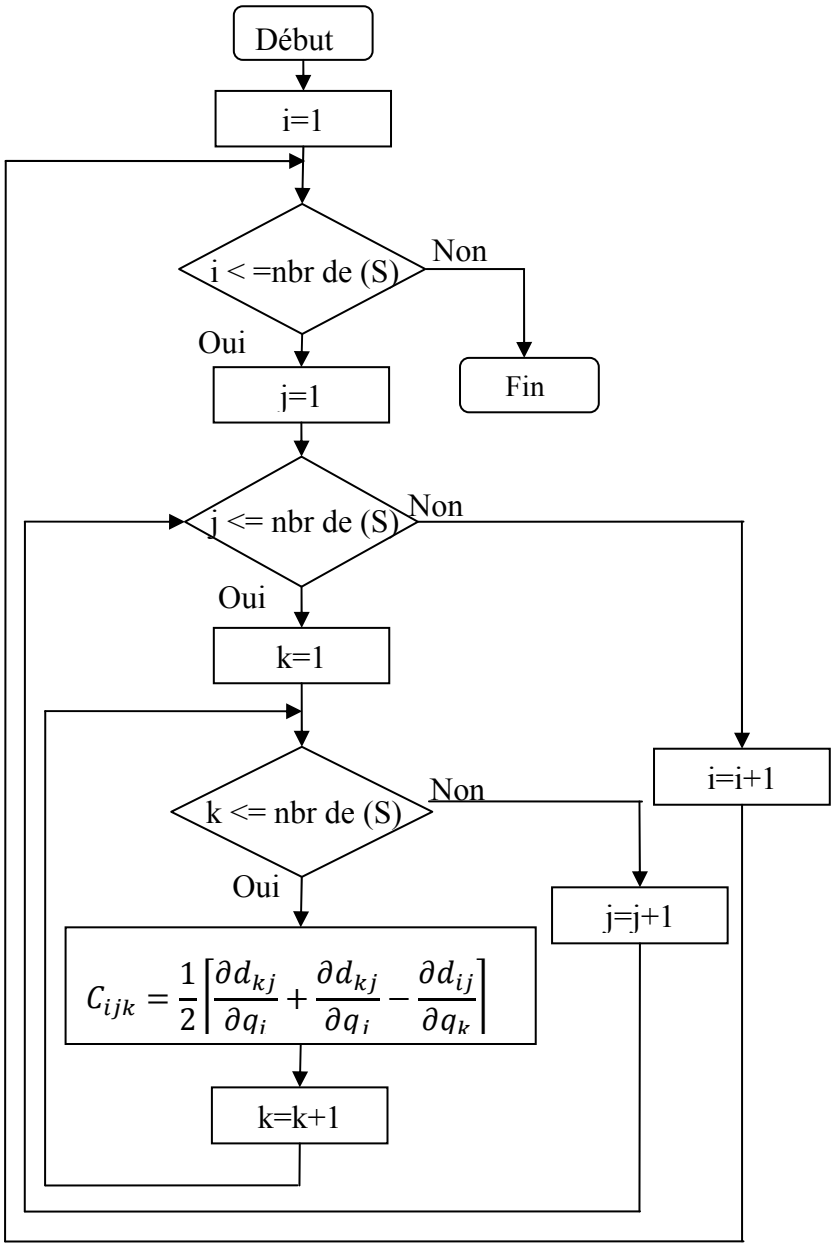
- L'organigramme qui nous permet de calculer La vitesse angulaire de centre de graviter de chaque solide est donné comme suit :



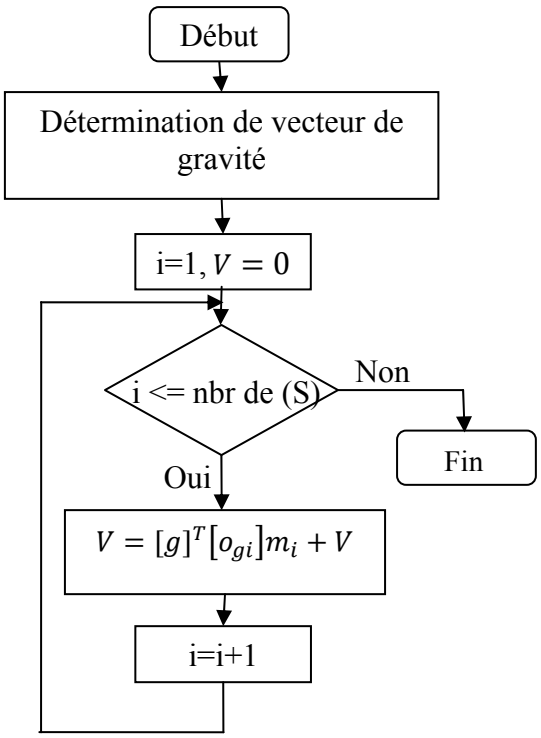
- L'organigramme qui nous permet de calculer la Matrice d'inertie est donné comme suit :



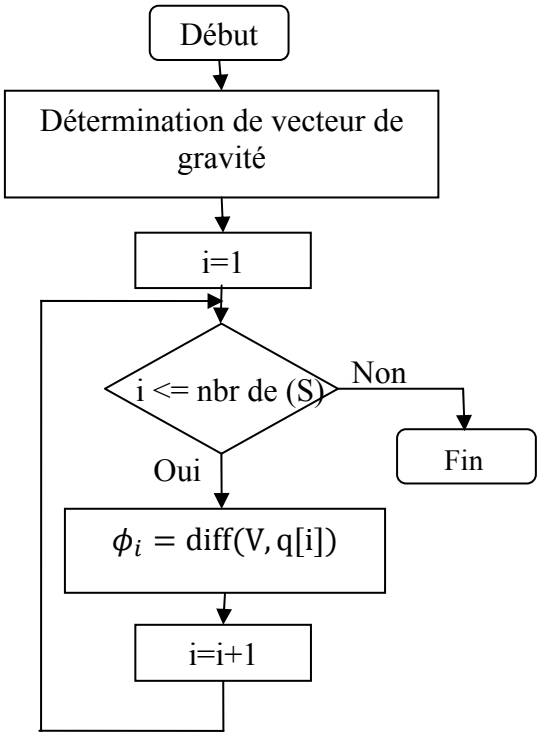
- L'organigramme qui nous permet de calculer la matrice des forces de couplage C est donné comme suit :



- L'organigramme qui nous permet de calculer l'énergie potentielle est donnée comme suit :



- L'organigramme qui nous permet de calculer $\phi_k$est donné comme suit :



Paramètre physique du robot

Les longueurs

L0=0.672 m.
L1=0.4318 m.
L2=0.3681 m.
L3=0.065 m.
L4=0.058 m.
L5=0.018 m.
L6=0.13 m.
a1=0.2435 m.
a2=0.0934 m.
a3=0.19 m.

La gravité

g=9.81N/kg;

Les masses

m1=17.08 kg.
m2=39.42 kg.
m3=18.51 kg.
m4=4.56 kg.
m5=1.21 kg.
m6=0.51 kg.

Les matrices d'inerties

$$I_1 = \begin{bmatrix} 0.66 & 0 & 0 \\ 0 & 0.66 & 0 \\ 0 & 0 & 0.098 \end{bmatrix} \text{kg.m}^2.$$

$$I_2 = \begin{bmatrix} 0.36 & 0 & 0 \\ 0 & 3.57 & 0 \\ 0 & 0 & 3.71 \end{bmatrix} \text{kg.m}^2.$$

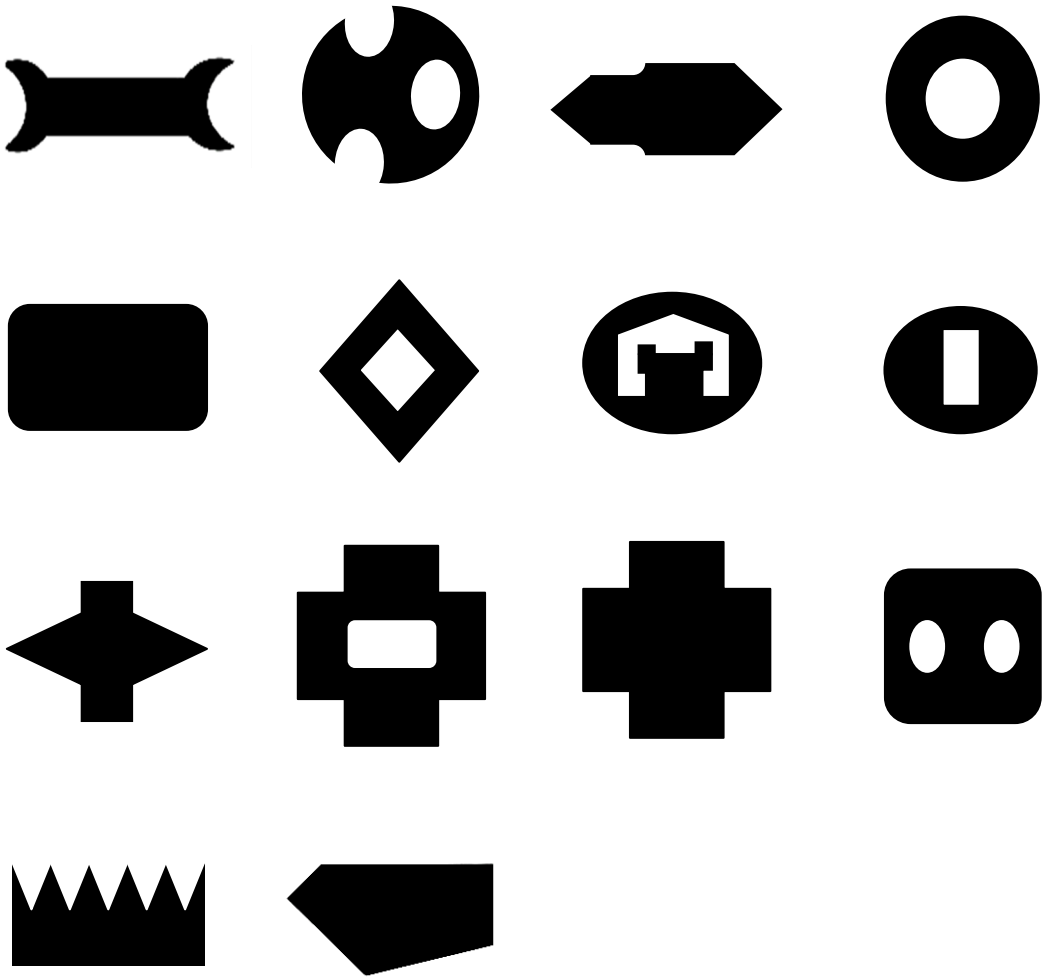
$$I_3 = \begin{bmatrix} 0.38 & 0 & 0 \\ 0 & 0.39 & 0 \\ 0 & 0 & 0.66 \end{bmatrix} \text{kg.m}^2.$$

$$I_4 = \begin{bmatrix} 0.012 & 0 & 0 \\ 0 & 0.009 & 0 \\ 0 & 0 & 0.012 \end{bmatrix} \text{kg.m}^2.$$

$$I_5 = \begin{bmatrix} 0.0007 & 0 & 0 \\ 0 & 0.0014 & 0 \\ 0 & 0 & 0.00073 \end{bmatrix} \text{kg.m}^2.$$

$$I_6 = \begin{bmatrix} 0.0017 & 0 & 0 \\ 0 & 0.0071 & 0 \\ 0 & 0 & 0.001 \end{bmatrix} \text{kg.m}^2.$$

Base de données



Bibliographie

[1] **Pratt, William K**, *«Digital Image Processing», 4th ed. USA (2007).*

[2] **Al Bovik, Alan C. Bovik, Jerry D. Gibson**, *«Handbook of Image and Video Processing». Canada (2000).*

[3] **Gerard Blanchet, Maurice Charbit**, *«Digital Signal And Image Processing Using Matlab», edition 2006.*

[4] **Bernd Jähne** , *« Digital Image Processing», edition 2002.*

[5] **Gonzales Raphael, Woods Richard, Eddins Steven**, *« Digital Image Processing Using Matlab», edition 2004.*

[6] **Sylvain Fassino**,*« Agrandissement d'images et de séquences vidéo», Thèse de Doctorat, Institut National Polytechnique De Grenoble,2004.*

[7] **PE. Zwicke, Z. Kiss**. *« A new implementation of the mellin transform and its application to radar classification ». IEEE Trans. on PAMI, 5(2) :191–199, March 1983.*

[8] **F. Ghorbel**, *« A complete invariant description for gray level images by the harmonic analysis approach », Pattern Recognition Letters 15, pp.1043-1051 (1994).*

[9] **S. Derrode**, *« Représentation de formes planes à niveaux de gris par différentes approximations de Fourier-Mellin analytique en vue d'indexation de bases d'images », Thèse de Doctorat, Université de Rennes 1, 1999.*

[10] **G. Ravichandran, M. Trivedi**, *« Circular-Mellin features for texture segmentation », IEEE Trans. Image Processing, 4, pp. 1629-1640 (1995).*

[11] **B. Potocnik**, *«Assessment of Region-Based Moment Invariants for Object Recognition», 48th IS, Multimedia Signal Processing and Communications. pp. 27-32 (2006).*

[12] **C.-H, teh and R.T. Chin**, *« On Image Analysis by the Methods of Moments», IEEE transactions on pattern analysis and machine intelligence, vol. 10. No. 4, july 1988.*

[13] **A.J. Nor'aini, P. Raveendran, N. Selvanathan**, *« A Comparative Analysis of Feature Extraction Methods for Face Recognition System », IEEE, 2005 Asian Conference on new Techniques in Pharmaceutical and Biomedical Research,5-7, September,2005.*

[14] **G.A. Papakostas, Y.S. Boutalis, D.A. Karras, B.G. Mertzios**, *« A new class of Zernike moments for computer vision applications », Science Directe, International Journal, Vol.177.pp.2802-2819 (2007).*

[15] **Vania ANDRONOVA**, *« Utilisation de données météo et des réseaux de neurones pour la prédiction de vitesses de vent», Projet de fin d'étude Master 2nd année,18 juillet 2006.*

[16] **Marc Parizeau**, *«Réseaux de neurones», édition 2004.*

[17] **Ben Krose, Patrick van der Smagt**, *«An introduction to Neural Networks», edition 1996.*

[18] **Mark W. Spong, Seth Hutchinson, and M. Vidyasagar**, *«Robot Dynamics and Control», Second Edition 2004.*

[19] **Stéphane BRETON**, *« Une approche neuronale du contrôle robotique utilisant la vision binoculaire par reconstruction tridimensionnelle », Thèse de Doctorat, Université de HAUTE-ALSACE, 1999.*

[20] **A. Katbab** *, «Nonlinear Adaptive Control of Robotic Manipulators - Hyperstability Approach », Robotics and Autonomous Systems Vol(4).pp. 265-273(1988).*