الجمهورية الجزائرية الديمقراطية الشعبية

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DR. YAHIA FARES DE MEDEA

FACULTE DES SCIENCES ET DE LA TECHNOLOGIE

Département du Génie Électrique et de l'Informatique



Projet de fin d'études

Présenté pour l'obtention du
Diplôme De Master en Automatique

Spécialité : Automatique et Systèmes

THEME

Implémentation d'une commande neuronale pour l'asservissement visuel d'un robot mobile

Proposé et dirigé par ;

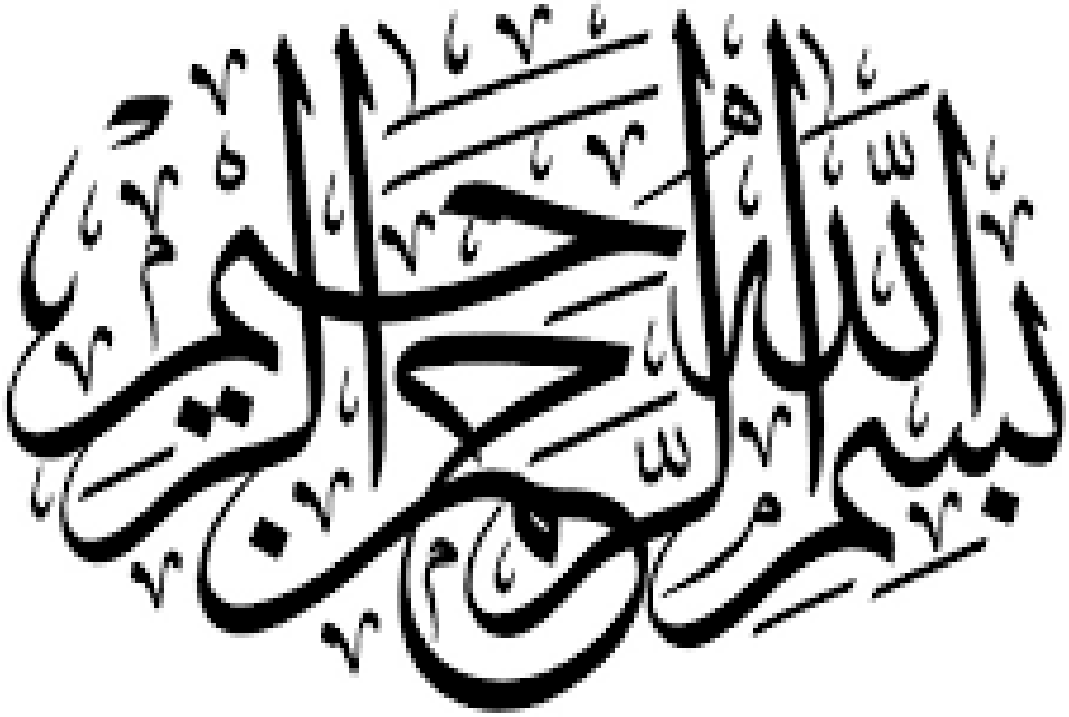
Dr R. KARA

Préparé et présenté par,

Mr FOUDI Mohammed El Fatih

Mr TOUMACHE M'hammed

Année Universitaire 2017-2018


بسم الله الرحمن الرحيم

Dédicace

Je dédie ce modeste travail à mes chers parents qui m'ont beaucoup encouragé et soutenu pendant tout leur vie pour voir ce jour-là et pour leurs précieux conseils ainsi que leurs recommandations.

A mes très chers frères et mes chères sœurs.

A toute ma famille, grande et petite sans exception.

A tous mes amis

FOUDI

Dédicace

Pour mes parents

Pour leur soutien moral.

A toute ma famille

Pour mes collègues de promotion et mes amis.

En fin, je dédie ce travail à toute personne

Qui me connais de près ou de loin.

TOUMACHE

Remerciement

Nous tenons tout d'abord à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la Patience d'accomplir ce modeste travail.

En second lieu, nous tenons à remercier notre encadreur

Dr. KARA Reda pour leurs précieux conseils

et leurs aides durant toute la période du travail.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail Et de l'enrichir par leurs propositions.

Enfin, nous tenons également à remercier toutes les personnes qui ont participé de près ou de loin à la réalisation de ce travail.

ملخص:

يهدف هذا المشروع الى إجراء المراقبة المرئية للروبوت المتحرك. للقيام بذلك ، قمنا بتطوير بنية الأجهزة المدمجة من خلال الجمع بين بطاقة الراسبيري مع بطاقة الأردوينو ، وعلاوة على ذلك ، قمنا بتنفيذ تحكم الذكاء الاصطناعي الذي يمكنه التحكم في حركات الروبوت المحمول من البيانات البصرية

الكلمات المفتاحية: تقييد بصري؛ الروبوتات المتحركة؛ راسبيري باي؛ آردوينو؛الشبكة العصبية

Résumé

Ce projet consiste à réaliser un asservissement visuel d'un robot mobile. Pour ce faire, n o u s avons développé une architecture matérielle embarquée en combinant une carte Raspberry à une carte Arduino Par ailleurs, nous avons mis en œuvre d'une contrôle d'intelligence artificiel capable d'asservir les mouvements d'un robot mobile à partir de données visuelles

Mots clés : asservissement visuel ; robotiques mobiles ; Raspberry Pi ; Arduino ; réseau de neurone

Summary :

This project involves performing a visual servoing of a mobile robot. To do this, we have developed an embedded hardware architecture by combining a Raspberry card with an Arduino card. Moreover, we have implemented an artificial intelligence control that can enslave the movements of a mobile robot from data. visual

Key words: visual servoing; mobile robotics; Raspberry Pi; Arduino; neural network

Table des matières

Introduction générale 1

Chapitre I : Généralités sur les robots mobiles

1. Introduction 4

2. Historique 5

3. Les bases mobiles 6

    3.1 Holonomie 6

    3.2 Les plates-formes différentielles 6

    3.3 Les plates-formes omnidirectionnelles 7

    3.4 Les plates-formes non holonomes 8

4. Types des robots 9

    4.1 Robots manipulateurs 9

        4.1.1 La structure mécanique et la motricité 10

        4.1.2. Deux générations de robots parallèles 12

        4.1.3 Domaines d'application 13

    4.2 Robots mobiles 14

        4.2.1 Classification 15

        4.2.2 Architecture des robots mobiles 16

5. Les robots à roues 16

    5.1. Robot unicycle 16

    5.2. Robot tricycle 17

    5.3. Robot voiture 17

    5.4. Robot omnidirectionnel 18

6. Capteurs en robotique mobile 19

7. Conclusion 22

Chapitre II : Les systèmes embarqués utilisées

1. Introduction 24

2. La carte Arduino 24

2.1. Définition 24

2.2. Les Applications 25

2.3. Les atouts majeurs d'un système à carte Arduino 25

2.4. Arduino Uno 27

2.5. Présentation 27

2.6. Structure de la carte Arduino Uno 28

2.7. Spécifications techniques 30

2.8. Choix de l'Arduino 31

2.9. Le Logiciel de gestion de la carte Arduino 32

3. La carte Raspberry pi 35

3.1 La carte mère Raspberry Pi 3 Type B 36

3.2 La Consommation électrique de Raspberry pi 3 B 36

3.3 Les composants standards de Raspberry 37

3.4 Spécifications physiques 40

3.5. Mise en marche d'un Raspberry Pi 3 40

3.6. Programmation 42

3.7. Système d'exploitation 43

3.8. Synthèse 44

4. L'extension (Shield) de commande des moteurs 44

5. La caméra Pixy 46

5.1. Aperçu 46

5.2. Bref historique 48

5.3. Simplicité de mise en œuvre du système de vision 48
5.4. Support du contrôleur 49
5.5. La détection d'objets 49
5.6. Définition simple de l'objet à suivre 50
5.7. L'application PixyMon 50
5.8. Spécifications techniques 51
6. Conclusion 52

Chapitre III : Les réseaux de neurones

1. Introduction 54
2. Définitions 55
3. Historique 55
3.1. W. James(1890) 55
3.2. Le neurone formel de McCulloch et Pitts(1943) 55
3.3. D. Hebb(1949) 56
3.4. Les premiers succès 56
3.4.1. F. Rosenblatt (1957) 56
3.4.2.B. Widrow(1960) 57
3.5. L'ombre : 58
3.6. Le renouveau 58
3.6.1. J. J. Hopfield(1982) 58
3.7. La levée des limitations 58
3.7.1. Les machines de Boltzmann (1983) 58
4. Utilité 59
5. Avantages 61
6. Inconvénients 61
7. Les neurones biologiques 61

7.1. Structure des neurones 62
7.1.1. Le corps cellulaire 62
7.1.2. Les dendrites 62
7.1.3. L'axone 62
8. Neurone formel 63
9. Apprentissage : 65
9.1. L'apprentissage supervisé 66
9.2. L'apprentissage nonsupervisé 66
9.3. L'apprentissage hybride 66
10. Quelques topologies des réseaux de neurones 67
10.1 Le réseau de Hopfield 67
10.2. Le réseau de neurone de Hamming 68
10.3. Le classificateur de Carpenter et Crossberg 69
10.4. Perceptron à une seule couche 70
10.5. Le perceptron multicouches 70
10.6. L'ADALINE 72
10.7. Carte auto-organisatrice de Kohonen 73
11. Résultat de nos différents testes 74
Pourquoi L'ADALINE ? 74
TESTE N°1 74
TESTE N°2 78
12. Conclusion 80

Chapitre IV : Validation expérimentale de notre contrôleur neuronal

1. Introduction 82

2. Robot mobile Arduino 83

2.1. Description 83

2.2. Modélisation 84

3. Le système de vision Pixy 85

3.1. La détection d'objet objet avec Pixy 85

3.2. Objets à signatures multiples 88

3.3. Balance des blancs 89

4. L'utilisation de l'application PixyMon 89

4.1. Signature tuning 91

5. La carte Raspberry Pi: le cerveau 94

6. Schéma bloc du projet 95

7. Le schéma d'asservissement visuel du robot mobile Arduino 99

8. Description du programme de la commande neuronale 98

9. Obtenir des données du Pixy 100

10. Les tests en situation de notre contrôleur neuronal dans l'asservissement visuel du robot mobile 101

Test N°1 : trajectoire rectiligne 101

Test N°2 : trajectoire courbée 109

11. Conclusion 113

Conclusions et Perspectives 115

Listes des figures

•

Figure I.1 : *Schéma des interactions d'un robot avec son environnement. Selon les approches, un modèle interne de l'environnement peut être utilisé ou non* 4

Figure I.2 *: La tortue de Grey Walter (nommée "machina speculatrix" et surnommée Elsie) et une illustration de sa trajectoire pour rejoindre sa niche.* 5

Figure I.3 : *Exemple de plate-forme différentielle. Pioneer 2 DX de la société Mobile Robots. Urban Robot de la société iRobot.* 7

Figure I.4 : *Exemple de plate-forme omnidirectionnelle à roues suédoises.* 8

Figure I.5 : *Exemple de plate-forme non holonome de type Ackerman.* 9

Figure I.6 : *structure générale d'un robot industriel* 10

Figure I.7: *le robot IRB 7600-150*. 11

FigureI.8 : *Le robot Hexamove-System* 12

Figure I.9 : *Le robot FlexPicker (ABB)*. 13

Figure I.10 : *Architecture d'un robot mobile.* 16

Figure I.11 : *Robot de type unicycle.* 17

Figure I.12 : *Robot de type tricycle*. 18

Figure I.13 : *Robot de type voiture.* 18

Figure I.14 : *Robot de type omnidirectionnel Comparaison des différents types.* 19

Figure II.1 : *Carte arduino UNO* 27

Figure II.2 : *Les compositions de la carte arduino UNO* 28

Figure II.3 : *Structure de la carte Arduino Uno.* 30

Figure II.4 : *logiciel Arduino*. 33

Figure II.5 : *Présentation des boutons.* 34

Figure II.6 : *Raspberry pi 3 B* 35

Figure II.7 : *Alimentation électrique de Raspberry pi 3 B* 37

Figure II.8 : *Les composants standards d'un Raspberry.* 38

Figure II.9 : *Spécifications physiques d'un Raspberry pi*..... 40

Figure II.10 : *Branchements de carte Raspberry pi* 42

Figure II.11: *Adafruit Motor Shield.*..... 44

Figure II.12 : *fonctionnement de L293D.* 45

Figure II.13: *montage L293D avec Shield.*..... 46

Figure II.14 : *Pixy CMUcam5* 47

Figure III.1. *Modèle de l'activité perspective* 57

Figure III.2. *Neurone biologique*..... 62

Figure III.3. *Le modèle d'un neurone biologique*..... 63

Figure III.4. *Le modèle de McCulloch-Pitts*..... 64

Figure III.5. *Quelques types de fonctions d'activations.* 65

Figure III.6.*Quelques types de réseaux de neurones*..... 67

Figure III.7. *Réseau de Hopfield* 68

Figure III.8. *Réseau de Hamming*..... 69

Figure III.9. *Perceptron à une seule couche*..... 70

Figure III.11.*regle d'apprentissage d'ADALINE* 73

Figure III.12. *Teste de la fonction sinus avec ADALINE*..... 75

Figure III.13. *Teste de fonction sinus avec ADALINE*..... 76

Figure III.14. *Teste de fonction sinus avec ADALINE*..... 77

Figure III.15. *Teste de fonction sinus cardinal avec ADALINE* 78

Figure IV.1. *Robot mobile de type tricycle.* 83

Figure IV.2. *Un robot mobile de type tricycle et son CIR.* 84

Figure.IV.3. *des objets avec des teintes bonnes et distinctes.* 86

Figure IV.4. *Des objets avec teintes non distinctes.* 86

Figure IV.5. *La majeure partie de l'objet (bon) couvre par une grille* 88

Figure.IV.5. *maintenez la signature de l'objet que vous souhaitez enseigner.* 90

Figure.IV.6. *sélectionner la région que vous souhaitez que Pixy utilise pour apprendre l'objet.* 90

Figure.IV.7. *Pixy "apprendra" l'objet* 91

Figure.IV.8. *Pixy Paramètres.* 92

Figure.IV.9. *ajuster l'inclusivité de la signature 1.* 92

Figure.IV.10. *la détection intermittente ou clairsemé.* 93

Figure.IV.11. *valeur de curseur qui procure une bonne détection.* 93

Figure.IV.12. *la carte Raspbery Pi utilisé.* 94

Figure.IV.13. *Schéma bloc du projet* 95

Figure.IV.14. *structure d'asservissement visuel utilisé* 96

Figure.IV.15. *structure d'asservissement visuel utilisé* 99

Figure.IV.16. *Voir ce que voit pixy.* 102

Figure.IV.17.*test de détermination de largeur L, hauteur H et de X*..... 100

Figure.IV.18. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur* 102

Figure.IV.19. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur* 102

Figure.IV.20. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur* 103

Figure.IV.21. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur* 103

Figure.IV.22. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur* 104

Figure.IV.23. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur* 104

Figure.IV.24 *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur* 105

Figure.IV.25. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur* 105

Figure.IV.26. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur* 106

Figure.IV.27. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur* 107

Figure.IV.28. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur* 108

Figure.IV.29. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur* 108

Figure.IV.30. *la trajectoire courbée utilisé* 109

Figure.IV.31. *graphe des vitesses v1, v1_est et erreur entre eux*..... 110

Figure.IV.32. *graphe des vitesses v2, v2_est et erreur entre eux*..... 110

Figure.IV.33. *graphe des vitesses v1, v1_est et erreur entre eux*..... 111

Figure.IV.34. *graphe des vitesses v2, v2_est et erreur entre eux*..... 111

Figure.IV.35. *graphe des vitesses v1, v1_est et erreur entre eux*..... 112

Figure.IV.36. *graphe des vitesses v2, v2_est et erreur entre eux*..... 112

Notation

$\boldsymbol{s}$: Grandeur ou Primitive visuelle asservie par la boucle de vision.

$\boldsymbol{P}$: Primitive visuelle 3D (dans notre cas Pose de la caméra).

$\boldsymbol{p}$: Primitive visuelle 2D (dans notre cas un point sur l'image).

$\boldsymbol{vc}$: Vitesse linéaire instantanée de la caméra.

$\boldsymbol{wc}$: Vitesse angulaire instantanée de la caméra.

$\boldsymbol{Vc}$: Torseur cinématique de la caméra.

$\boldsymbol{\tau}$: Vecteur des forces généralisées appliqués aux articulations du robot.

$\boldsymbol{e}$: Fonction de tâche.

$\boldsymbol{Ls}$: Matrice d'interaction relative à la primitive visuelle.

$\boldsymbol{L\theta u}$: Matrice d'interaction relative à la rotation.

$\boldsymbol{\lambda}$: Matrice de gain de contrôle de la boucle de commande visuelle.

$\boldsymbol{t}$: Vecteur de translation.

$\boldsymbol{\theta}$: Angle de rotation.

$\boldsymbol{u}$: Vecteur de rotation.

$\boldsymbol{Rc}$: Repère actuel de la caméra.

$\boldsymbol{Ro}$: Repère de l'objet.

$\boldsymbol{T}$: Matrice de transformation homogène.

$\boldsymbol{Z}$: Profondeur des points attachés à l'objet dans le repère de la caméra.

C : vitesse de son.

T : température (° C).

H : humidité (g/m³).

$\boldsymbol{w}1$: Vitesse angulaire de roue 1.

$\boldsymbol{w}2$: Vitesse angulaire de roue 2.

r : rayent de roue.

v : Vitesse de véhicule.

$\boldsymbol{w}$: Vitesse angulaire de véhicule.

l : distance entre les roues.

d : distance.

S : la surface d'objet.

L : largeur.

H : hauteur.

X : valeur de référence.

Ɛ : l'erreur.

Kp : gain proportionnelle.

V : vitesse de référence.

W : poids

α : Taux d'apprentissage

Liste des acronymes

CCD: Charge Coupled Device.

GPS: Global Positioning System.

Mini-PC : mini personal computer.

LED: light-emitting diode.

USB: Universal Serial Bus.

IDE : Integrated Development Environment.

PWM : Pulse Width modulation (Modulation de la largeur des pics).

ICSP : In-Circuit Serial Programming ou (Programmation Série Dans le circuit).

SRAM : Static Random Access Memory (mémoire vive statique).

GND : Ground (la masse).

TTL : Transistor-Transistor Logic.

SPI : Interface Série Périphérique.

TW : Two Wire Interface ou (Interface "2 fils").

UART : Universal Asynchronous Receiver Transmitter ou (émetteur-récepteur asynchrone universel).

I2C: inter-integrated circuit.

FPGA: field-programmable gate array.

HDMI: High Definition Multimedia Interface.

WIFI: Wireless Internet for Frequent Interface.

CPU: Central processing unit.

RAM: Random Access Memory (mémoire vive).

LCD: liquid crystal display.

GPIO: General Purpose Input/ Output.

MDI: Medium dependent interface.

LX: Réception data.

TX: émission data.

HTML: Hyper Text Markup Language.

VIN: Voltage Input.

CSI: Camera Serial Interface.

DSI: Desktop Serial Interface.

FFC: Flexible Flat Cable.

LAN: Local Area Network.

JTAG: Joint Test Action Group.

ADC: Analog to Digital Converter.

DAC: Digital to Analog Converter.

DMA: Direct Access Memory.

ROM: Read-Only Memory.

SDA: Serial Data Line.

SCL: Serial Clock Line.

MCU: Multipoint Control Unit.

PCB: Printed Circuit Board.

HD: High Definition.

SD: Secure Digital.

SDHC: Secure Digital High-Capacity.

RGB: Red/Green/Blue.

USB OTG: Universal Serial Bus On-The-Go.

AREF : Analogique Référence.

TWI : Two Wire Interface.

AC: Alternating Current.

DC: Direct Current.

SOC: System On a Chip.

3D : signifie trois dimensions. Ces dimensions permettent de définir un objet possédant trois axes x, y et z, associés respectivement à la longueur, la hauteur et la profondeur.

2D : signifie deux dimensions. Ces dimensions permettent de définir un objet possédant deux axes x, y, associés respectivement à la longueur, la hauteur

Introduction générale

Les robots mobiles jouent un rôle très important dans le domaine industriel et scientifique et notre vie moderne et ils ont bouleversés tous les domaines et ils ont devenus indispensables dans les usines, les boulangeries et les chantiers et on ne peut pas imaginer notre future sans les robots.

La robotique est un domaine de recherche important qui fait appel aux connaissances croisées de plusieurs disciplines. L'objectif est l'automatisation des systèmes mécaniques, en les dotant par des capacités de perception, de décision et d'action permettant au robot d'interagir rationnellement avec son environnement sans intervention humaine . La robotique mobile est un domaine dans lequel l'expérience pratique est primordiale.

La volonté des chercheurs à reproduire les capacités humaines de perception et d'action dans les systèmes robotisés a conduit à l'intégration des données issues de capteurs extéroceptifs, et plus particulièrement de celles issues d'une caméra. L'objectif affiché est que les capteurs visuels fournissent une information suffisamment riche pour permettre aux robots de réaliser, de manière autonome, des tâches dans des environnements partiellement connus, ou complètement inconnus. D'un point de vue méthodologique, l'asservissement visuel consiste à intégrer directement dans la boucle de commande des robots, des informations extraites des images fournies par des caméras afin de réaliser l'action souhaitée

En robotique, la commande revêt usuellement la forme de programme informatique séquentiel et répétitif à structure de données fixes. Il en résulte consécutivement deux limitations majeures :

- la nécessité d'une connaissance et d'une maîtrise parfaites du comportement du système robot-vision (modélisation mathématique) de manière à assurer une bonne précision d'exécution des tâches,
- une incapacité de prise en considération les variations du fonctionnement ou de la géométrie du système robot-vision.

A ces inconvénients, les techniques adaptatives en général, et les réseaux de neurones artificiels apportent des solutions. C'est dans ce contexte que s'inscrit notre projet.

Ce mémoire s'articule autour de deux objectifs majeurs. Le premier étant la maitrise de l'architecture matérielle qui constitue notre plate-forme robot-vision. En effet, cette structure lui offre la possibilité au robot mobile de réaliser un suivi d'objets uniquement a partir d'informations visuel fournies par un dispositifs de vision appelée Cam Pixy. Le deuxième but consiste implémenter un contrôleur basé sur un réseau de neurones. L'avantage des de ce type de contrôleur est qu'ils ne nécessitent aucune connaissance analytique (modélisation mathématique du problème). Le problème est modélisé par apprentissage à partir d'une base d'exemples. L'adaptation constitue une qualité intrinsèque essentielle des réseaux de neurones. Leur fonction d'apprentissage rend ce type de contrôleur en effet naturellement évolutifs.[1]

Ce document est organisé en quatre chapitres qui portent sur les aspects suivants :

Le premier chapitre présente une introduction à la robotique mobile, des descriptions des différents éléments constituants les robots mobiles ainsi que les différentes structures mécaniques qui existent.

Le deuxième chapitre est consacré aux systèmes embarqués utilisés dans notre projet, tel que le système de vision Pixy, la carte raspberry Pi3, la carte shield L293D et la carte Arduino.

Le troisième chapitre est consacré à la présentation des réseaux de neurones, et à l'utilisation des réseaux de neurones pour la commande robotique. Nous vous illustrons leurs fonctionnements avec des fonctions simples

Le quatrième chapitre est une présentation du fonctionnement détaillé et des modifications majeures apportées à la plate-forme robot mobile pour réaliser le suivi d'une cible à partir de données visuelles. Cette mise en situation nous donnera l'occasion de vous décrire notre conception du neuro-contrôleur ainsi que l'évaluation de ses performances au travers d'une série de tests pratiques réalisée sur cette plate-forme robot-vision.

Chapitre I :

Généralités sur les robots mobiles

1. Introduction

La robotique est un très bon exemple de domaine pluri-disciplinaire qui implique de nombreuses thématiques telles que la mécanique, la mécatronique, l'électronique, l'automatique, l'informatique ou l'intelligence artificielle. En fonction du domaine d'origine des auteurs, il existe donc diverses définitions du terme robot, mais elles tournent en général autour de celle-ci :

Un robot est une machine équipée de capacités de perception, de décision et d'action qui lui permettent d'agir de manière autonome dans son environnement en fonction de la perception qu'il en a. [1]

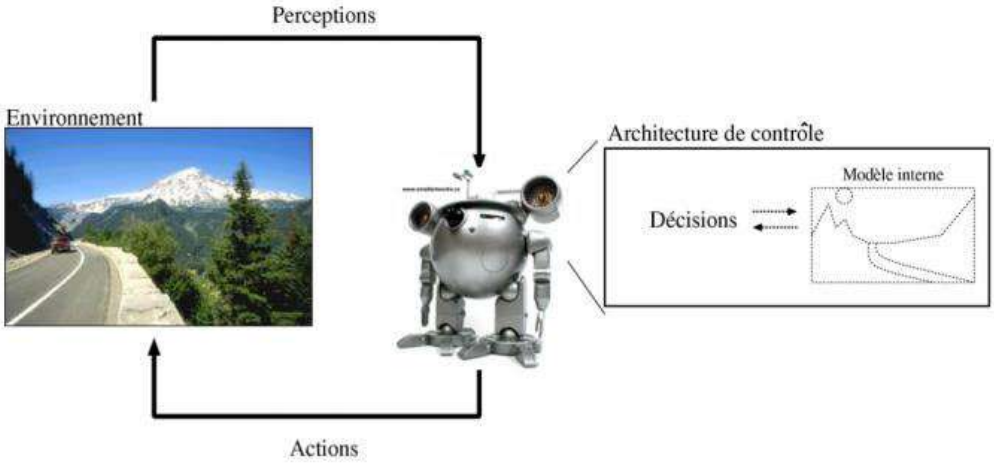


Figure I.1 : *Schéma des interactions d'un robot avec son environnement. Selon les approches, un modèle interne de l'environnement peut être utilisé ou non*

Cette définition s'illustre par un schéma classique des interactions d'un robot avec son environnement (Figure I.1). Les différentes notions que nous présenterons dans ce chapitre sont essentiellement issues de cette vision de la robotique, très orientée sur l'Intelligence

Artificielle, qui place au centre des préoccupations l'enchaînement de ce cycle Perception/Décision/Action.

La manière dont un robot gère ces différents éléments est définie par son architecture de contrôle, qui la plupart du temps va faire appel à un modèle interne de l'environnement qui lui permettra de planifier ses actions à long terme. [1]

2. Historique

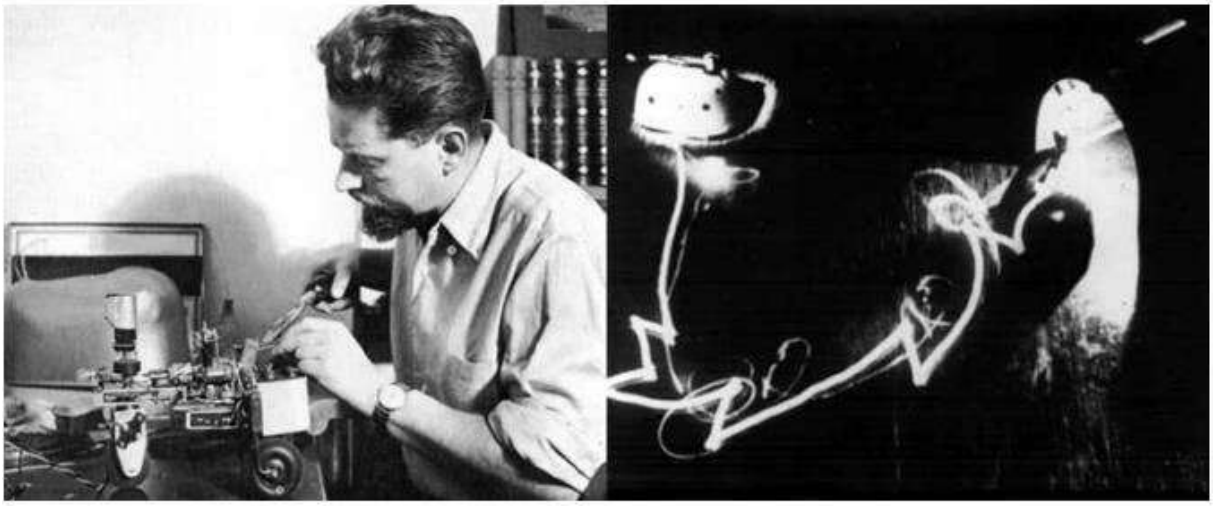


Figure I.2 : *La tortue de Grey Walter (nommée "machina speculatrix" et surnommée Elsie) et une illustration de sa trajectoire pour rejoindre sa niche.[2]*

Ce n'est qu'au tout début du XX^{e} siècle que les robots firent leur apparition, suite aux travaux d'ingénieurs qui voulaient tester des hypothèses émises par des biologistes et des psychologues. Le chien électrique conçu par Hammond et Miessner en 1915 était attiré par une lumière, selon le phototropisme animal mis en évidence par Loeb en 1918. Les machines de Russell (1913) et de Stephens (1929), les tortues cybernétiques de Grey Walter (Figure I.2) (1950), le renard électronique de Ducrocq (1953) ou l'homéostat d'Ashby (1952) étaient, elles, dotées de capacités d'apprentissage directement issues des travaux des psychologues Thorndike (1911), Hull (1943) et du physiologiste Pavlov (1903) sur l'Homme et l'animal.

Ces réalisations sont des robots, car elles ne se comportent plus comme de simples automates dont les organes moteurs - leurs mécanismes - obéissent à un programme préétabli. À la différence des automates, ils ont des organes sensoriels - les capteurs - qui recueillent des informations de l'environnement qui vont, elles, influencer l'activité de leurs organes moteurs - les actionneurs. [2]

3. Les bases mobiles

Nous présentons rapidement les différents types de bases mobiles utilisées en robotique, en nous focalisant sur les plateformes mobiles terrestres pour le milieu intérieur.

3.1 Holonomie

En robotique, une plateforme est dite holonome lorsque que le nombre de degrés de libertés contrôlables est égal au nombre total de degrés de liberté.

Pour un robot se déplaçant sur un plan, il y a 3 degrés de liberté (deux translations et une rotation).

A partir d'une position donnée, une plateforme holonome devra donc pouvoir se déplacer en avant, sur le côté et tourner sur elle-même. Cette capacité permet de contrôler très simplement le robot car tous les déplacements imaginables sont réalisables, ce qui simplifie le problème de planification de trajectoire. Nous allons cependant voir quelques mécanismes permettant d'obtenir des plateformes holonomes, ou s'en approchant. [3]

3.2 Les plates-formes différentielles

Une des configurations les plus utilisées pour les robots mobiles d'intérieur est la configuration différentielle qui comporte deux roues commandées indépendamment. Une ou plusieurs roues folles sont ajoutées à l'avant ou à l'arrière du robot pour assurer sa stabilité (FigureI.3). Cette plate-forme est très simple à commander, puisqu'il suffit de spécifier les vitesses des deux roues, et permet de plus au robot de tourner sur place. Cette possibilité permet de traiter dans certains cas le robot comme un robot holonome, ce qui va simplifier la planification de déplacement et la commande du robot. [3]

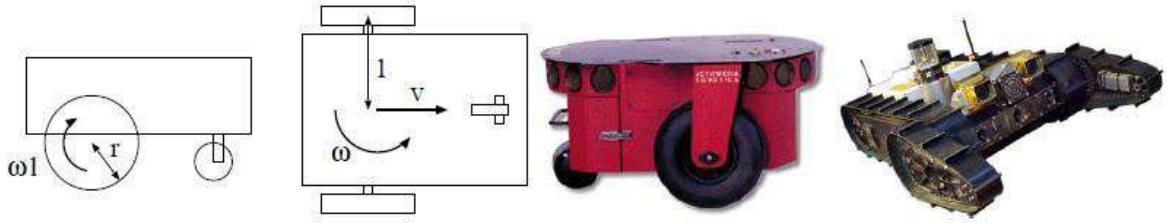


Figure I.3 : *Exemple de plate-forme différentielle. Pioneer 2 DX de la société Mobile Robots. Urban Robot de la société iRobot.[3]*

L'estimation du déplacement par odométrie est également très simple à partir de la mesure des vitesses de rotation des deux roues w1 et w2. Les vitesses de translation v et de rotation w sont en effet données par :

$$v = \frac{\omega_1 r + \omega_2 r}{2}$$
$$\omega = \frac{\omega_1 r - \omega_2 r}{2l}$$

3.3 Les plates-formes omnidirectionnelles

Les plates-formes omnidirectionnelles permettent de découpler de manière plus nette le contrôle de la rotation et de la translation d'un robot et sont donc quasiment holonomes.

Il existe différents types de plateformes omnidirectionnelles. Le premier utilise trois ou quatre roues qui tournent à la même vitesse pour fournir une translation et un mécanisme qui permet d'orienter simultanément ces roues dans la direction du déplacement souhaitée

Le corps du robot lui-même n'effectue pas de rotation mais uniquement des translations. Ce système permet un contrôle très simple et relativement rapide car les changements de direction ne concernent que les roues et peuvent donc se faire très vite. Par contre ces plates-formes sont relativement limitées en capacité de franchissement et requièrent un sol très plan.

Une deuxième catégorie de plateformes utilise des roues dites "suédoises", qui n'offrent pas de résistance au déplacement latéral (Figure I.4). La plateforme comporte trois roues dont les axes sont fixes. Les déplacements dans toutes les directions et en rotation sont obtenus en faisant varier individuellement les vitesses des roues. La plateforme tourne sur place lorsque les trois

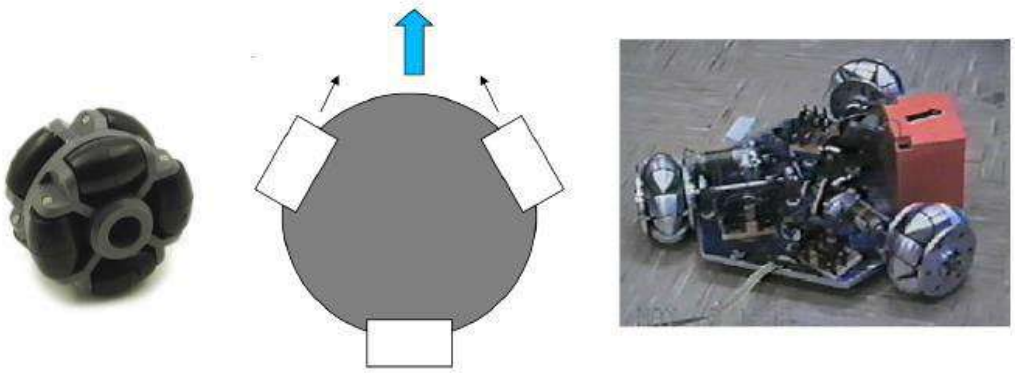


Figure I.4 : *Exemple de plate-forme omnidirectionnelle à roues suédoises.*

Roues tournent dans le même sens, à la même vitesse. Lorsqu'une roue est fixe, et que les deux autres tournent en sens opposé, la plateforme avance en direction de la roue fixe. Différentes combinaisons de vitesses permettent d'obtenir des déplacements quelconques.

3.4 Les plates-formes non holonomes

Des plates-formes non holonomes, telles que les voitures, sont également utilisées en robotique mobile (Figure I.5). C'est plus particulièrement le cas dans le domaine des véhicules intelligents. Ces plates-formes sont toutefois plus difficiles à commander car elles ne peuvent pas tourner sur place et doivent manœuvrer, ce qui peut être difficile dans des environnements encombrés.

La commande de ces plates-formes pour réaliser un déplacement particulier est un problème à part entière que nous n'aborderons pas dans ce cours. Par contre, il est possible de prendre en compte ces contraintes de manière relativement simple dans la planification. [3]

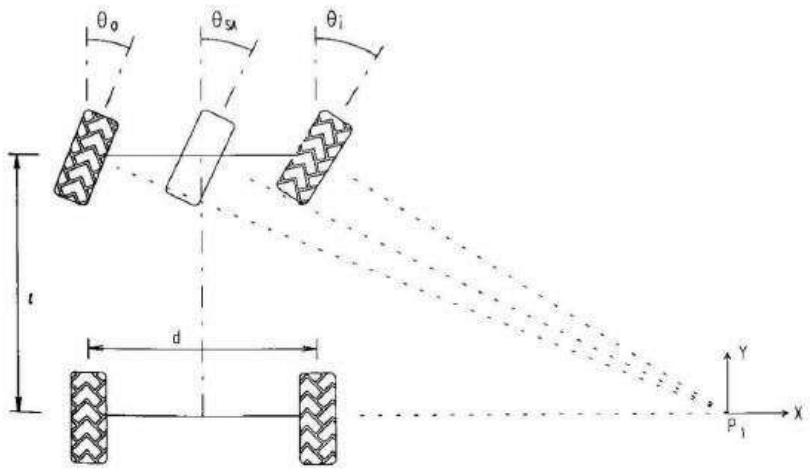


Figure I.5 : *Exemple de plate-forme non holonome de type Ackerman.*

4. Types des robots

Au cours de l'histoire, 2 types différents de robots sont apparus attestant chacun de nouvelles évolutions : les robots manipulateurs et les robots mobiles.

4.1 Robots manipulateurs

Un robot manipulateur, quelle que soit la fonction qui lui est attribuée (transfert d'objet, soudage, assemblage), est un mécanisme capable de déplacer et de situer un objet appelé « organe terminal» ou «point outil» dans une partie de l'espace appeler « volume de travail ».

Un robot industriel est un manipulateur à plusieurs degrés de liberté contrôlé automatiquement, reprogrammable et multitâche qui peut être fixe ou mobile pour une application en automatisation industrielle.

Le robot industriel se compose d'une structure mécanique animée par des actionneurs, à partir d'ordres élaborés par un calculateur. Ces ordres dépendent des informations délivrées par les capteurs. L'utilisation de capteurs externes, capteurs'' extéroceptifs'', pour évaluer et mesurer l'interaction du robot avec l'environnement directement depuis son organe terminal devient une pratique de plus en plus courante dans les applications robotiques de haute précision.[4]

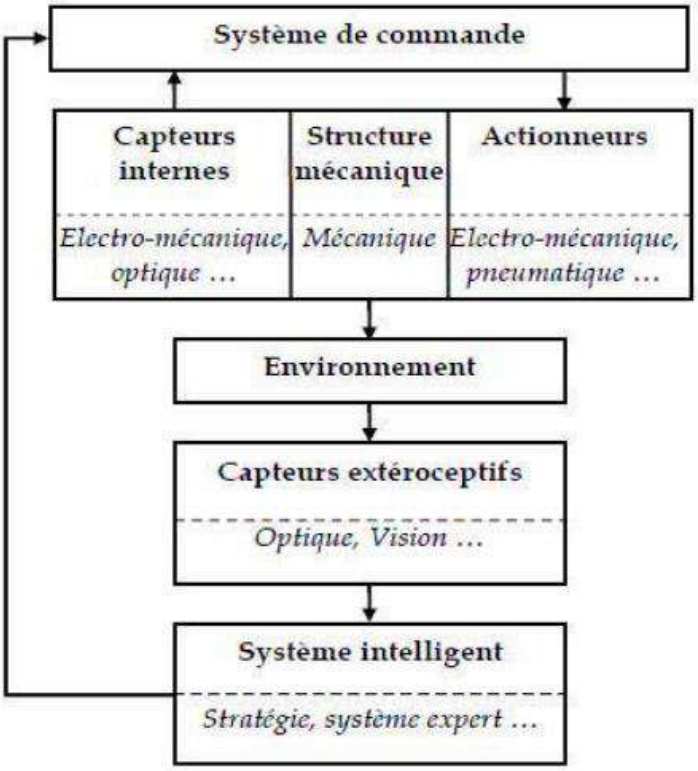


Figure I.6 : *structure générale d'un robot industriel. [4]*

4.1.1 La structure mécanique et la motricité

On peut dénombrer deux types de structures mécaniques assurant la motricité :

a) Les robots séries

Définition : La plupart des robots industriels construits à ce jour est de type sériel, c'est -à-dire que leur structure mobile est une chaîne ouverte formée d'une succession de segments reliés entre eux par des liaisons à un degré de liberté. Chaque articulation est commandée par un actionneur situé à l'endroit de l'articulation ou sur un des segments précédents



Figure I.7: *le robot IRB 7600-150.*

Les robots séries possèdent l'avantage de disposer d'un grand volume de travail et d'être relativement simples sur le plan des calculs liés à leur commande. Leurs principaux inconvénients sont les suivants :

- Inertie élevée à cause de la répartition des masses sur toute la chaîne cinématique (actionneurs, organes de transmission).
- Manque de rigidité par la mise en série d'éléments élastiques, fatigue et usure des liaisons de puissance assurant l'alimentation des actionneurs (câbles, tuyaux flexibles).
- Fatigue et usure des liaisons assurant la circulation des informations entre les capteurs et la commande, ce point est très important lorsqu'il s'agit de sûreté de fonctionnement, puisqu'une erreur de transmission peut avoir des conséquences néfastes sur les mouvements du robot.

b) Les robots parallèles

Pour certaines applications industrielles telles que l'usinage grand vitesse (UGV) ou la manutention est rapide, les manipulateurs à cinématique sérielle ne semblent plus être les mieux adaptés. En effet, ce type d'architecture implique que chaque axe motorisé supporte le suivant. Les masses en mouvement sont donc élevées, ce qui pénalise les performances dynamiques.

C'est pourquoi, les manipulateurs à cinématique parallèle sont aujourd'hui de plus en plus utilisés. Leurs performances dynamiques élevées ainsi que leurs capacités

de charges importantes sont avantageusement mises à profit dans le monde industriel, ce qui fait des robots parallèles des robots industriels par excellence.

Définition : Un manipulateur parallèle est un mécanisme en chaîne cinématique fermé dont l'organe terminal est relié à la base par plusieurs chaînes cinématiques indépendante par exemple la figure I.8



FigureI.8 : *Le robot Hexamove-System [4].*

Les robots parallèles seront présentés comme étant une solution aux limitations des robots sériels.

4.1.2. Deux générations de robots parallèles

Il existe de nombreux historiques concernant les robots parallèles tels ceux de Bonevou Merlet qui précisent leurs véritables origines. Nous souhaitons seulement insister sur l'existence de deux générations essentielles de robot parallèle :

- La première génération incarnée par les plates-formes de Goughet Stewart.

On parle aujourd'hui de robots hexapodes : 6 vérins relient le bâti à une plate-forme mobile, rendant possible l'exécution de mouvements complexes par la mise en parallèle des chaînes cinématiques,

- La deuxième génération incarnée par la structure Delta de Clavel: les robots de cette famille sont capables de performances exceptionnelles (vitesses jusqu'à10 m/s et accélération jusqu'à 20 G). On parle dès lors de robots parallèles légers avec pour principales caractéristiques :
- des actionneurs fixes sur le bâti,
- des composants mobiles légers (Figure I.9).



Figure I.9 : *Le robot FlexPicker (ABB).*

4.1.3 Domaines d'application

La robotique est un domaine en plein essor depuis quelques années. Les évolutions technologiques, dépassant sans cesse nos espérances, permettent maintenant de réaliser des solutions technologiques s'adaptant au moindre problème.

Par conséquent, la robotique est utilisée dans des domaines extrêmement rigoureux et exigeants. Nous allons explorer ces différents domaines. [4]

- ✓ **L'industrie.** Le but premier des robots est de remplacer l'homme dans des activités fastidieuses ou onéreuses pour l'employeur. Les robots ont donc commencé à être utilisés dans les chaînes d'assemblage industrielles. Dans ces chaînes d'assemblage, on retrouve des robots soudeurs, manipulateurs, peintres ...
- ✓ **Le domaine militaire.** Les robots sont de plus en plus utilisés dans le domaine militaire. En effet, la miniaturisation permet aujourd'hui de créer des robots discrets mais dotés de nombreux capteurs, ce qui est idéal pour des missions d'espionnage ou d'éclairement.
- ✓ **La santé.** Les robots commencent à être de plus en plus utilises dans le domaine médical, qu'il s'agisse de « simples » échographies ou d'opérations chirurgicales plus délicates. En fait ces robots ne sont pas complètement autonomes mais ils assistent les médecins ou chirurgiens, jusqu'à permettre des opérations médicales à distance (télémédecine). On parle de surgétique (mot né de l'anglais « surgery » : chirurgie) c'est -à-dire tout ce qui consiste à introduire les derniers outils des technologies informatiques et robotiques dans la pratique médico-chirurgicale. Cette pratique de « chirurgie assistée » est émergente donc bien que peu répandue, elle est en phase de devenir la chirurgie du futur.

4.2 Robots mobiles

Les robots mobiles ont une place particulière en robotique. Leur intérêt réside dans leur mobilité qui ouvre des applications dans de nombreux domaines. Comme les robots manipulateurs, ils sont destinés à assister l'homme dans les tâches pénibles (transport de charges lourdes), monotones ou en ambiance hostile (nucléaire, marine, spatiale, lutte contre l'incendie, surveillance…).

L'aspect particulier de la mobilité impose une complexité technologique et méthodologique qui s'ajoute en général aux problèmes rencontrés par les robots manipulateurs. La résolution de ces problèmes passe par l'emploi de toutes les ressources disponibles tant au niveau technologique (capteurs, motricité, énergie) qu'à celui du traitement des informations par l'utilisation des techniques de l'intelligence artificielle ou de processeurs particuliers (vectoriel, cellulaires).

L'autonomie du robot mobile est une faculté qui lui permet de s'adapter ou de prendre une décision dans le but de réaliser une tâche malgré un manque d'informations préliminaires ou éventuellement erronées. Dans d'autres cas d'utilisation, comme celui des véhicules d'exploration de planètes, l'autonomie est un point fondamental puisque la télécommande est alors impossible par le fait de la durée du temps de transmission des informations.[4]

4.2.1 Classification

Une classification est proposée dans la littérature qui définit le degré d'autonomie du robot mobile.

- Véhicule télécommandé par un opérateur qui lui impose chaque tâche élémentaire à réaliser.
- Véhicule télécommandé au sens de la tâche à réaliser. Le véhicule contrôle automatiquement ses actions.
- Véhicule semi-autonome réalisant sans l'aide de l'opérateur des tâches prédéfinies.
- Véhicule autonome qui réalise des tâches semi-définies. Ce type de véhicule

pose des problèmes d'un niveau de complexité élevé de représentation des connaissances, de capacité décisionnelle et de génération de plans qui sont résolus à bord dans la mesure du possible.

L'ensemble des problèmes particuliers liés à la conception de tels robots sont :

- La conception mécanique liée à la mobilité
- La détermination de la position et de la latitude (orientation)
- La détermination du chemin optimal pour atteindre le lieu de la tâche.

4.2.2 Architecture des robots mobiles

L'architecture des robots mobiles se structure en quatre éléments :

• La structure mécanique et la motricité

• Les organes de sécurité

• Le système de traitement des informations et gestion des tâches.

• Le système de localisation

L'architecture des robots mobiles est représentée sur la figure 1.10.

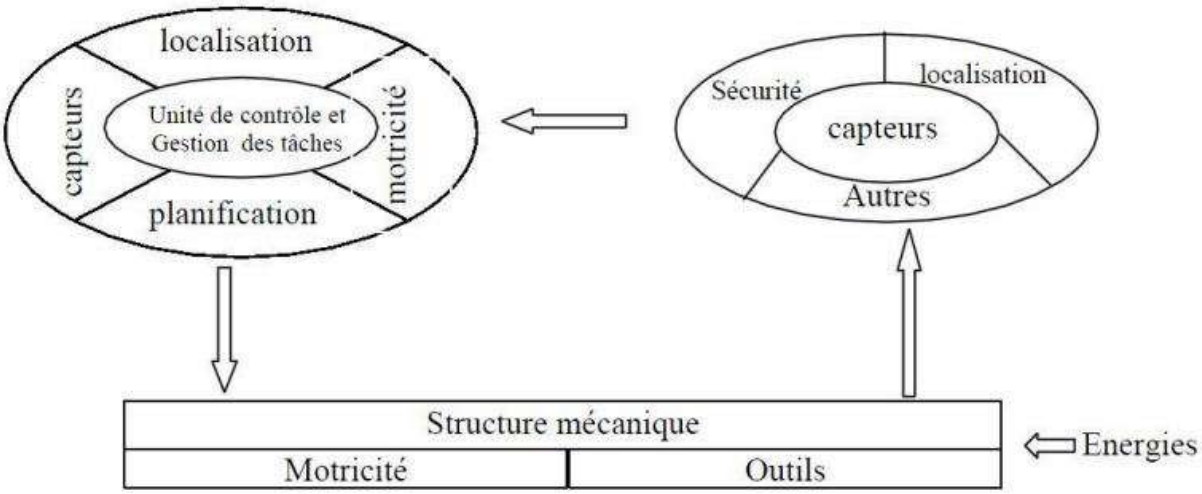


Figure I.10 : *Architecture d'un robot mobile.*

5. Les robots à roues

Il existe plusieurs classes de robots à roues déterminées, principalement, par la position et le nombre de roues utilisées.

Nous citerons ici les quatre classes principales de robots à roues. [5]

5.1. Robot unicycle

Un robot de type unicycle est actionné par deux roues indépendantes, il possède éventuellement des roues folles pour assurer sa stabilité. Son centre de rotation est situé sur l'axe reliant les deux roues motrices.

C'est un robot non-holonome, en effet il est impossible de le déplacer dans une direction perpendiculaire aux roues de locomotion.

Sa commande peut être très simple, il est en effet assez facile de le déplacer d'un point A un autre par une suite de rotations simples et de lignes droites.

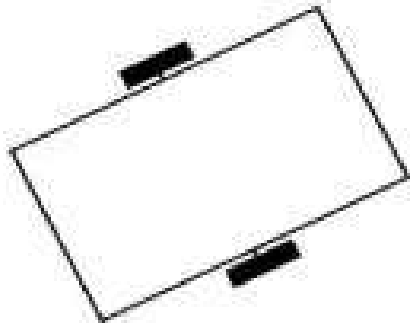


Figure I.11 : *Robot de type unicycle.*

5.2. Robot tricycle

Un robot de type tricycle est constitué de deux roues fixes placées sur un même axe et d'une roue centrée orientable placée sur l'axe longitudinal. Le mouvement du robot est donné par la vitesse des deux roues fixes et par l'orientation de la roue orientable. Son centre de rotation est situé à l'intersection de l'axe contenant les roues fixes et de l'axe de la roue orientable.

C'est un robot non-holonome. En effet, il est impossible de le déplacer dans une direction perpendiculaire aux roues fixes. Sa commande est plus compliquée. Il est en général impossible d'effectuer des rotations simples à cause d'un rayon de braquage limité de la roue orientable.

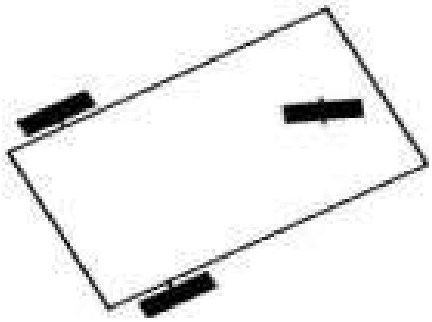


Figure I.12 : *Robot de type tricycle.*

5.3. Robot voiture

Un robot de type voiture est semblable au tricycle, il est constitué de deux roues fixes placées sur un même axe et de deux roues centrées orientables placées elles aussi sur un même axe.

Le robot de type voiture est cependant plus stable puisqu'il possède un point d'appui supplémentaire.

Toutes les autres propriétés du robot voiture sont identiques au robot tricycle, le deuxième pouvant être ramené au premier en remplaçant les deux roues avant par une seule placée au centre de l'axe, et ceci de manière à laisser le centre de rotation inchangé.

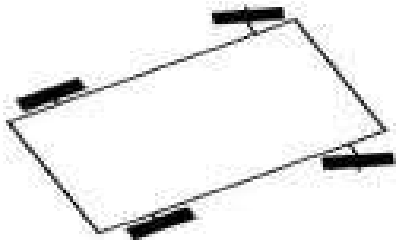


Figure I.13 : *Robot de type voiture.*

5.4. Robot omnidirectionnel

Un robot omnidirectionnel est un robot qui peut se déplacer librement dans toutes les directions. Il est en général constitué de trois roues décentrées orientables placées en triangle équilatéral.

L'énorme avantage du robot omnidirectionnel est qu'il est holonome puisqu'il peut se déplacer dans toutes les directions. Mais ceci se fait au dépend d'une complexité mécanique bien plus grande. [5]



Figure I.14 : *Robot de type omnidirectionnel Comparaison des différents types.*

Tableau I.2 : *Les avantages et des inconvénients des différents types de robots à roues.*

Type de robot	Avantage	Inconvénient
Unicycle	• Stable • Rotation sur soi-même • Complexité mécanique faible	• Non-holonome
Tricycle	• Complexité mécanique modérée	• Non-holonome • Peu stable • Pas de rotation sur soi-même
Voiture	• Stable • Complexité mécanique modérée	• Non-holonome • Pas de rotation sur soi-même
Omnidirectionnel	• Holonome • Stable • Rotation sur soi-même	• Complexité mécanique importante

6. Capteurs en robotique mobile

En robotique mobile, on classe traditionnellement les capteurs en deux catégories selon qu'ils mesurent l'état du robot lui-même ou l'état de son environnement. Dans le premier cas, a l'image de la perception chez les ` êtres vivants, on parle de ˆ proprioception et donc de capteurs proprioceptifs. On trouve par exemple dans cette catégorie les ´ capteurs de position ou de vitesse des roues et les capteurs de charge de la batterie. Les capteurs renseignant sur l'état de l'environnement, donc de ce qui est extérieur au ´ robot lui-même, sont eux appel ˆ es´ capteurs extéroceptifs ´ . Il s'agit de capteurs donnant la distance du robot a l'environnement, la température, signalant la mise en contact du ´ robot avec l'environnement, etc.

L'étude d´entaillée des capteurs, qui relève ` a la fois de la physique, de l'électronique ´ et du traitement du signal, ne sera pas vue ici. Nous nous contenterons d'expliquer les principes de fonctionnement des capteurs présent ´ es. On tachera simplement de garder ´ a` l'esprit que les défauts inhérents aux différents systèmes de mesure utilisé ` es (bruit, erreurs ´ ou échecs de mesures, difficulté ´ e de modélisation) influent fortement sur la perception ´ que le robot a de l'environnement.

Les principaux capteurs utilisés en robotique mobile sont : les capteurs télémétriques (les ultrasons, les lasers et les infrarouges), le GPS et les caméras. [6]

Les capteurs télémétriques

On appelle télémétrie toute technique de mesure de distance par des procédés acoustiques, optiques ou radioélectriques. L'appareil permettant de mesurer les distances est appelé télémètre. Il existe différentes technologies pour réaliser un télémètre. Nous présentons dans ce qui suit les télémètres les plus utilisés dans les systèmes de robotique mobile, en donnant une idée de leur gamme de mesure et d'application.

Les GPS

Un des systèmes les plus connus de ces dernières années est le Système de Positionnement Global (GPS) du département de la défense des Etats-Unis qui exploite une constellation de vingt-quatre satellites en orbite à haute altitude.

Chaque satellite détecte les récepteurs terrestres fixes ou mobiles qui se trouvent dans sa zone de visibilité et transmet à chacun, à période fixe, un signal contenant des informations

telles que son identification, l'heure d'émission du signal et les paramètres définissant son orbite donc sa position. Théoriquement, un récepteur doit être visible d'au moins trois satellites, pour pouvoir calculer sa position dans un repère centré sur la terre, en calculant l'intersection de trois sphères (trois cercles pour un système mobile au sol) ; de fait, pour tenir compte des décalages temporels entre les mesures fournies par les satellites, quatre mesures de distance entre le satellite et le récepteur sont requises pour disposer d'une précision satisfaisante [6].

Ce système est cependant loin de résoudre tous les problèmes de localisation des systèmes mobiles. Il fonctionne en effet difficilement dans des environnements urbains, et n'est pas utilisable à l'intérieur des bâtiments car les ondes employées sont fortement atténuées en traversant les murs. Sa précision est de plus souvent trop faible pour qu'un système mobile terrestre puisse utiliser ces informations seules. En pratique, il est souvent couplé à un système inertiel qui permet de pallier aux pertes du signal GPS et il ne remplace de toute façon pas les capteurs du système mobile qui lui permettent de percevoir son environnement immédiat.

Mis à part son utilisation pour la localisation, ce système est aussi utilisé pour l'aide à la navigation d'avions et de bateaux lors de l'approche d'aéroports ou de ports, ainsi que dans la surveillance de camionnettes de livraison et de véhicules d'urgence, en vue de choisir au mieux leurs itinéraires.

Les caméras

Depuis les années 70, les caméras CCD (Charge Coupled Device) ont été déterminantes dans l'évolution de la vision : la rapidité d'acquisition, la robustesse et la miniaturisation sont autant d'avantages qui ont facilité leur intégration.

Elles sont très performantes en termes de portée, précision et quantité d'informations exploitables.

Elles sont de plus les seules capables de restituer une image sensorielle de l'environnement la plus proche de celle perçue par l'être humain. C'est sans doute le capteur fournissant la plus grande variété d'informations à partir d'une seule acquisition. En effet, en appliquant divers algorithmes de traitement d'images, il est possible d'extraire des éléments caractéristiques de l'environnement comme des segments de droite, des plans, mais aussi des balises et des couleurs.

En revanche, l'inconvénient majeur de tels capteurs se situe d'abord au niveau de la gestion du flux important de données exploitables (traiter une image demeure une opération délicate et surtout coûteuse en temps de calcul), ensuite à leur sensibilité aux conditions d'éclairage.

Pour augmenter le champ de vision de ces caméras, on peut les associer soit avec d'autres caméras ou soit à des systèmes de réflexion de type miroir : il s'agit des techniques de vision omnidirectionnelle. Cette association permet d'obtenir une vue sur 360 degrés de l'environnement [6].

7. Conclusion

La présentation faite dans cette première partie nous a permis de décrire les éléments constituants d'un robot mobile. A partir de ce descriptif, nous avons constaté que pour réaliser la commande de ce type de robot, nous devions répondre à trois impératifs à savoir la perception, la décision et l'action.

La combinaison d'informations issues de capteur proprioceptifs et de capteurs extéroceptifs permet d'envisager la réalisation de tâches robotiques complexes. Dans le cas de la robotique mobile, le traitement de toutes ces informations nécessite une puissance de calcul importante.

Dans le prochain chapitre nous vous présenterons donc, les cartes électroniques que utilisées pour gérer et commander le robot mobile.

Chapitre II :

Les systèmes embarqués utilisées

1. Introduction

Dans ce chapitre nous allons introduire et présenter les deux cartes électroniques embarquées ainsi que du système de vision que nous utiliserons dans l'élaboration de notre système d'acquisition de données, leur traitement et le calcul des lois de commandes nécessaires à l'asservissement de notre robot mobile. L'architecture matérielle que nous avons développée s'articule essentiellement autour de la carte Arduino de la carte Raspberry Pi et du système de vision Pixy. La caméra embarquée Pixy est considérée comme l'œil humain dans le sens ou grâce à son électronique elle nous fournira directement les caractéristiques visuelles nécessaires à la poursuite d'un objet cible. Le Raspberry Pi sera quant à lui assimilé au cerveau de notre robot tandis que le Arduino aura pour rôle la gestion de l'ensemble Capteurs / Actionneurs du robot.

Dans un premier temps, nous décrirons la carte Arduino de manière générale, et la carte Arduino UNO que nous avons utilisé.

Dans un second temps, nous décrirons l'architecture et le fonctionnement du mini-PC Raspberry Pi. Et on présenter le moteur shield L293D, Enfin nous détaillerons les spécifications et le fonctionnement du système de vision Pixy.

Finalement nous vous décrirons la méthode que nous avons adopté pour faire communiqué l'ensemble des cartes et de la caméra dans le but de réaliser un ensemble cohérent capable de gérer toute les opérations nécessaires à la commande visuelle du robot mobile.

2. La carte Arduino

2.1. Définition

Arduino est une plate-forme électronique open-source basée sur un matériel et un logiciel faciles à utiliser. Arduino est né à l'Ivrea Interaction Design Institute comme un outil facile pour le prototypage rapide, destiné aux étudiants sans formation en électronique et en programmation. Au fil des années, Arduino a été le cerveau de milliers de projets, des objets

de tous les jours aux instruments scientifiques complexes. Une communauté mondiale de décideurs - étudiants, amateurs, artistes, programmeurs et professionnels - s'est réunie autour de cette plate-forme open source, leurs contributions ont contribué à une connaissance incroyable de connaissances accessibles qui peuvent être d'une grande aide pour les novices et les experts. [7]

2.2. Les Applications

Les cartes Arduino peuvent lire des entrées - lumière sur un capteur, un doigt sur un bouton ou un message Twitter - et le transformer en sortie - activer un moteur, allumer une LED, publier quelque chose en ligne.

L'étendue de l'utilisation de l'Arduino est gigantesque. Pour vous donner quelques exemples, vous pouvez :

- Contrôler des systèmes intelligents (smart home)
- Fabriques des robots et des machines (bras robots, cnc, imprimante 3D)
- Communiquer avec des plateformes différentes (web, serveur)
- Systèmes de détection (gaz, mouvement, visage)

2.3. Les atouts majeurs d'un système à carte Arduino

Grâce à son expérience utilisateur simple et accessible, Arduino a été utilisé dans des milliers de projets et d'applications différents. Le logiciel Arduino est facile à utiliser pour les débutants, mais assez flexible pour les utilisateurs avancés. Il fonctionne sur Mac Windows et Linux, Les enseignants et les étudiants l'utilisent pour construire des instruments scientifiques à faible coût, pour prouver les principes de chimie et de physique, ou pour commencer avec la programmation et la robotique.

Les concepteurs et les architectes créent des prototypes interactifs, des musiciens et des artistes l'utilisent pour les installations et expérimentent de nouveaux instruments de musique. Les fabricants, bien sûr, l'utilisent pour construire plusieurs des projets exposés chez Maker Faire, par exemple. Arduino est un outil clé pour apprendre de nouvelles choses. Toute

personne - les enfants, les amateurs, les artistes, les programmeurs - peut commencer à bricoler juste après les instructions étape par étape d'un kit, ou en partageant des idées en ligne avec d'autres membres de la communauté Arduino. [7]

- ***Les cartes Arduino peu coûteuses*** : sont relativement peu coûteuses par rapport aux autres plates-formes de microcontrôleurs. La version la moins chère du module Arduino peut être assemblée à la main, et même les modules Arduino préassemblés coûtent moins de 50 $.

- ***Cross-Platform :*** *Le* logiciel Arduino (IDE) s'exécute sur les systèmes d'exploitation Windows, Macintosh OSX et Linux. La plupart des systèmes de microcontrôleurs sont limités à Windows.

- ***Environnement de programmation simple et clair*** : Le logiciel Arduino (IDE) est facile à utiliser pour les débutants, mais suffisamment flexible pour que les utilisateurs avancés puissent en profiter. Pour les enseignants, il est commodément basé sur l'environnement de programmation de traitement, afin que les élèves qui apprennent à programmer dans cet environnement connaîtront la façon dont l'IDE Arduino fonctionne.

- ***Logiciel open source et extensible*** : Le logiciel Arduino est publié sous la forme d'outils open source, disponibles pour l'extension par des programmeurs expérimentés. La langue peut être étendue à travers les bibliothèques C ++, et les personnes souhaitant comprendre les détails techniques peuvent faire passer le saut d'Arduino au langage de programmation AVR C sur lequel elles se basent. De même, vous pouvez ajouter le code AVR-C directement dans vos programmes Arduino si vous le souhaitez.

- ***Matériel open source et extensible*** : Les plans des cartes Arduino sont publiés sous une licence Créative Commons, de sorte que les concepteurs de circuits expérimentés peuvent créer leur propre version du module, l'étendre et l'améliorer. Même les utilisateurs relativement inexpérimentés peuvent créer la version boardboard du module pour comprendre comment cela fonctionne et économiser de l’argent. [7]

2.4. Arduino Uno

Arduino Uno est un panneau de microcontrôleur basé sur l'ATmega328P. Il dispose de 14 broches d'entrée / sortie numériques (dont 6 peuvent être utilisées comme sorties PWM), 6 entrées analogiques, un cristal à quartz 16 MHz, une connexion USB, une prise de courant, un en-tête ICSP et un bouton de réinitialisation. Il contient tout le nécessaire pour supporter le microcontrôleur ; Il suffit de le connecter à un ordinateur avec un câble USB ou de l'alimenter avec un adaptateur AC-DC ou une batterie pour commencer. Vous pouvez bricoler avec votre UNO sans trop déranger pour faire quelque chose de mal, le pire scénario que vous pouvez remplacer Puce pour quelques dollars et recommencer. [7]

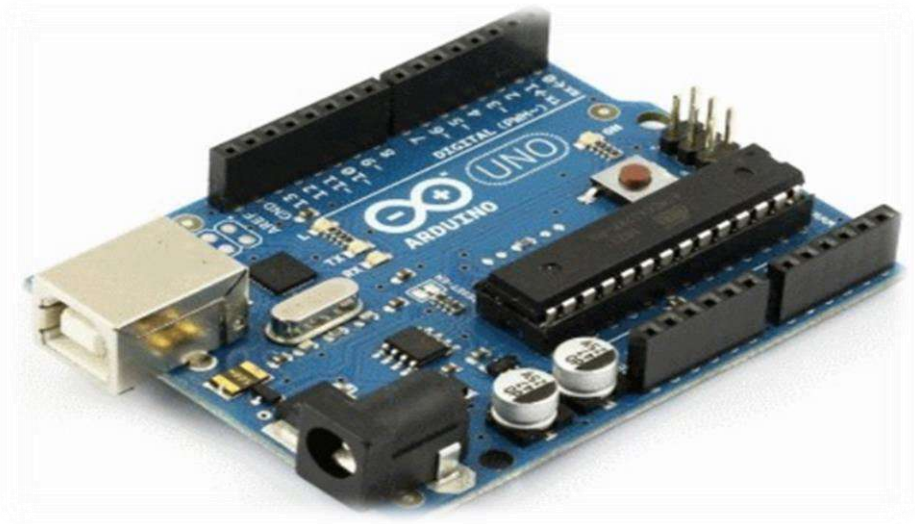


Figure II.1 : *Carte arduino UNO [7]*

2.5. Présentation

Il s'agit d'une carte électronique basée autour d'un microcontrôleur Atmega du fabricant Atmel, dont le prix est relativement bas pour l'étendue possible des applications. Voilà à quoi ressemble la carte que nous allons utiliser dans la figure II.2 :

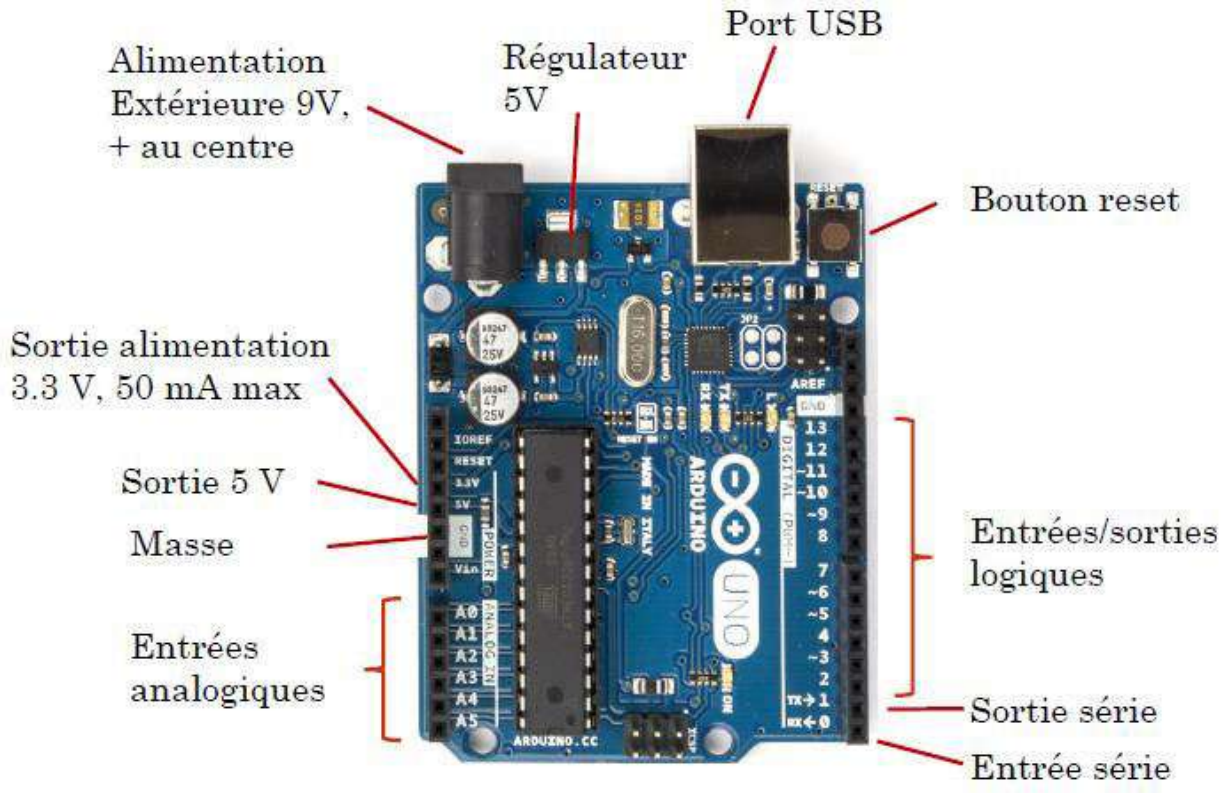


Figure II.2 : *Les compositions de la carte arduino UNO [7]*

2.6. Structure de la carte Arduino Uno

Dans ce qui va suivre, nous détaillerons la structure de ce système et leurs intérêts.

- ***Le microcontrôleur***

Le microcontrôleur correspond le cerveau de notre carte, c'est lui qui va traiter provenant des capteurs dans l'entrée et qui va donner la réponse à la sortie à travers le programme que vous avez créé et qui va le stocker dans sa mémoire.

- ***L'alimentation***

Il faut tout d'abord connaître les spécifications de la carte Arduino que l'on possède. Elles utilisent tout d'abord toutes du courant continu.

Si l'on regarde les différentes broches possibles, on va généralement trouver une broche étiquetée 5V, une broche étiquetée 3,3V et une broche étiquetée VIN ou RAW.

- VIN ou RAW correspondent à l'entrée du régulateur de tension présent sur la carte. Pour alimenter l'Arduino via cette broche, il faut une alimentation dont la tension est comprise entre 7 et 12V.
- 5V n'a pas le même rôle selon que le microcontrôleur qui équipe l'Arduino est alimenté en 5V ou en 3,3V. Dans le premier cas, c'est directement la tension d'alimentation du microcontrôleur. Dans le second cas, le microcontrôleur est alimenté via un régulateur 3,3V dont l'entrée est connectée sur le 5V.
- De la même manière, le 3,3V va être la tension d'alimentation du microcontrôleur pour ceux alimentés en 3,3V et une tension de service pour connecter des dispositifs externes nécessitant du 3,3V si le microcontrôleur nécessite du 5V.

En peut alimenter la carte Arduino par :

- Alimentation par USB
- Alimentation par connecteur Jack
- Alimentation par 2 broches de l'Arduino : VIN et GND [7]

- ***La visualisation***

On a trois LED qu'on trouve sur la carte donner à deux choses :

- ***La première (L) :*** elle est connectée à une broche du microcontrôleur et va servir pour tester le matériel.

  Nota : Quand on branche la carte au PC, elle clignote quelques secondes.

- ***Les deux LED :*** servent à visualiser l'activité sur la voie série (une pour l'émission (TX) et l'autre pour la réception (LX)).

Le téléchargement du programme dans le microcontrôleur se faisant par cette voie, on peut les voir clignoter lors du chargement.

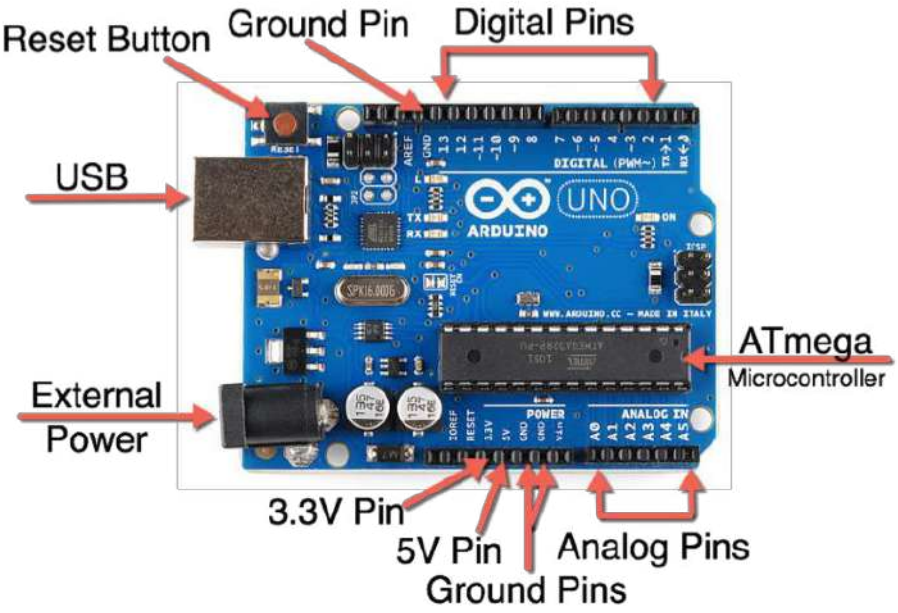


Figure II.3 : *Structure de la carte Arduino Uno.*

2.7. Spécifications techniques

Tableau 2.1 : *caractéristiques de la carte Arduino Uno. [7]*

Microcontrôleur	ATmega328P
Tension de fonctionnement	5V
Tension d'entrée (recommandée)	7-12V
Tension d'entrée (limite)	6-20V
Puces d'E / S numériques	14 (dont 6 fournissent une sortie PWM)
PWM Digital I / O Pins	6
Broches d'entrée analogique	6

Courant continu par broche E / S	20 mA
Courant pour broche de 3.3V	50 mA
Mémoire Flash	32Ko(ATmega328P) Dont 0,5 Ko utilisé par boot loader
SRAM	2 Ko (ATmega328P)
EEPROM	1 KB (ATmega328P)
Vitesse de l'horloge	16 MHz
LED_BUILTIN	13
Longueur	68,6 mm
Largeur	53,4 mm
Poids	25 g

2.8. Choix de l'Arduino

Il existe pourtant dans le commerce, une multitude de plateformes qui permettent de faire la même chose. Notamment les microcontrôleurs « PIC » du fabricant Micro chip. Nous allons choisir l'Arduino pour ces raisons

- **Le prix** : En vue des performances qu'elles offrent, les cartes Arduino sont relativement peu couteuses, ce qui est un critère majeur pour le débutant.
- **La liberté:** C'est un bien grand mot, mais elle définit de façon assez concise l'esprit de l'Arduino. Elle constitue en elle-même deux choses :

- **Le logiciel *:*** gratuit et open source, développé en Java, dont la simplicité d'utilisation relève du savoir cliquer sur la souris.
- **Le matériel :** cartes électroniques dont les schémas sont en libre circulation sur internet.

Cette liberté a une condition : le nom « Arduino » ne doit être employé que pour les cartes officielles ».

- ***Les cartes non officielles :*** on peut les trouver et les acheter sur Internet et sont pour la quasi-totalité compatibles avec les cartes officielles Arduino.

La compatibilité: Le logiciel, tout comme la carte, est compatible sous les plateformes les plus courantes (Windows, Linux et Mac), contrairement aux autres outils de programmation du commerce qui ne sont, en général, compatibles qu'avec Windows**.**

La communauté: La communauté Arduino est impressionnante et le nombre de ressources à son sujet est en constante évolution sur internet. Deplus, on trouve les références du langage Arduino ainsi qu'une page complète de tutoriels sur le site arduino.cc(en anglais) etarduino.cc(en français) [7].

2.9. Le Logiciel de gestion de la carte Arduino

Arduino avoir un environnement spécial (IDE en anglais) est une application écrite en java inspiré du langage Processing, L'IDE permet d'écrire, de modifier un programme et de convertir en une série d'instructions compréhensibles pour la carte.

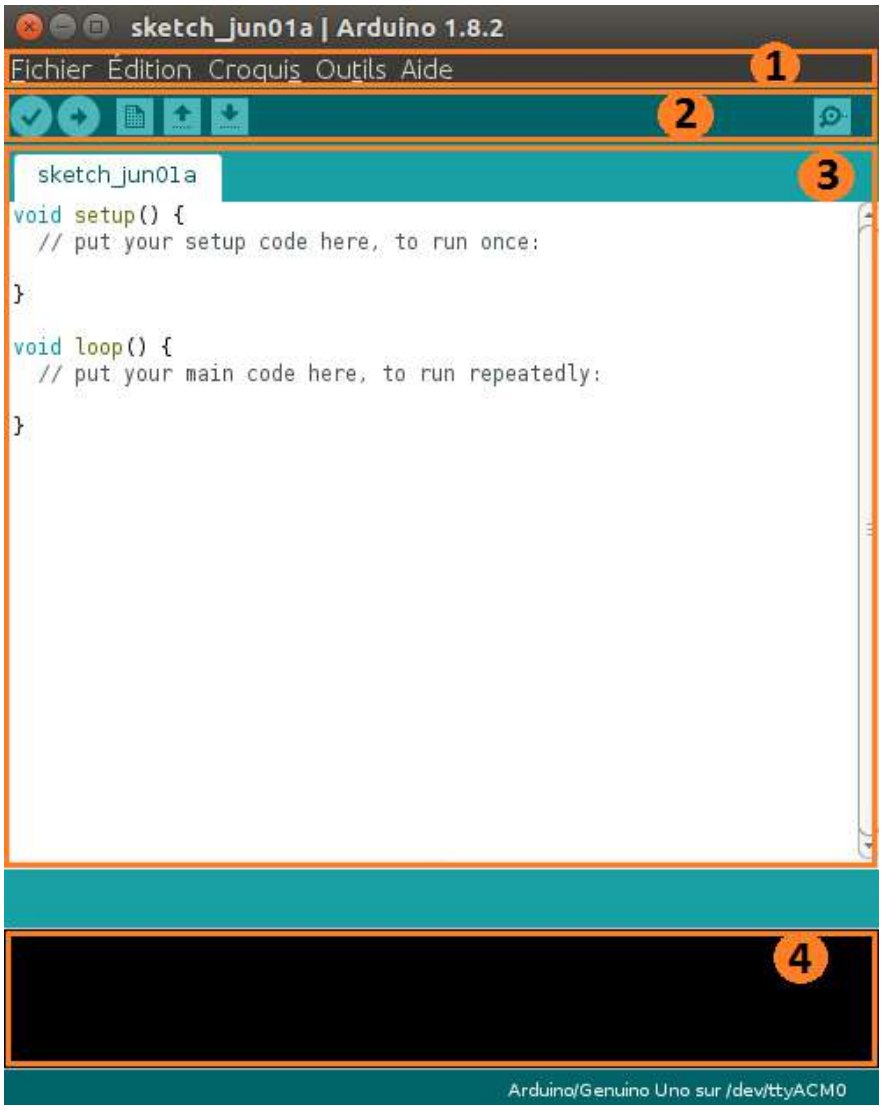


Figure II.4 : logiciel Arduino.

- Le cadre numéro 1 : options de configuration du logiciel.
- Le cadre numéro 2 : il contient les boutons qui vont nous servir lorsque l'on va programmer nos cartes.
- Le cadre numéro 3 : ce bloc va contenir le programme que nous allons créer.
- Le cadre numéro 4 : celui-ci est important, car il va nous aider à corriger les

  Fautes dans notre programme. C'est le débogueur.

- ***Les boutons***

Voyons à présent à quoi servent les boutons, encadrés en orange et numérotés par le chiffre 2.



Figure II.5 : *Présentation des boutons.*

1. Vérifiée le programme.
2. Compile le code et exporte le programme dans le microcontrôleur.
3. Création d'un nouveau fichier.
4. Ouverture d'un fichier.
5. Sauvegarde le fichier.
6. Ouvre le moniteur série.

- ***Moniteur série***

Le moniteur sériel est utilisé pour afficher l'information qui est envoyée par la carte Arduino vers l'application (habituellement par le câble USB). Il permet aussi d'envoyer de l'information à la carte Arduino. Pour ce faire, il suffit d'écrire du texte dans le champ situé en haut de la fenêtre et d'appuyer sur le bouton « Send ». Bien évidemment, avant de pouvoir faire ceci il vous faut coder un programme approprié, prêt à recevoir cette information. Il est également possible de régler le baud rate du moniteur sériel, qui précise à quelle vitesse le transfert des données s'effectuera. Il est également possible d'établir une communication sérielle entre la carte Arduino et d'autres périphériques ou logiciels. Nous verrons plus en détails cette fonction dans les projets « Texte brillant » et « Oscilloscope ». [7]

3. La carte Raspberry pi

Le Raspberry Pi est un micro-ordinateur possédant un processeur embarqué ARM, Les avantages sont un prix attractif (entre 25 et 45 €), et une taille minimaliste

Nous présentons dans la suite, le Raspberry pi 3 Model B que l'on a utilisé dans notre projet.

Ce dernier est considéré comme le cerveau de notre système parce qu'il va gérer les informations issues de la caméra, les traiter et calculer les lois de commandes nécessaires aux mouvements du robot mobile.

Le Raspberry Pi 3 Model B est un petit ordinateur (Mini-PC) qui fonctionne sous le système d'exploitation Linux sur carte SD destiné à des applications d'informatique embarquée. [8]



Figure II.6 : *Raspberry pi 3 B*

Pourquoi Pi ?

• C'est économique.

• Très facile à utiliser

• C'est un code libre basé sur Linux avec un accès aux logiciels et téléchargements Gratuits pour vous aider avec votre projet.

• Il existe une grande communauté Pi.

• Excellente connectivité

• Flexible pour l'utilisateur. Vous voulez jouer à des jeux ? Diffuser de la musique

Et/ou des vidéos ? Travailler ? Créer une station météorologique ? Créer une caméra De sécurité domestique ? Aucune limite.

Pourquoi Pi 3 B ?

- Démarrage plus rapide
- Une meilleure compatibilité avec les derniers programmes et logiciels éducatifs
- Processeur intégré Quad-core ARM Cortex-A53 1.2 GHz (Broadcom BCM2837)
- RAM : 1024 Mo et Lecteur de cartes Micro SD, Wi-Fi b/g/n et Bluetooth 4.1
- distributions GNU/LINUX ARM, telles que Snappy Ubuntu, et Microsoft Windows 10.

3.1 La carte mère Raspberry Pi 3 Type B

Ce dernier Model est le Raspberry Pi 3. Il a remplacé l'original Raspberry Pi 2 Model B en février 2015 qui a été utilisé par nos collègues de l'année dernière, [8] Par rapport au Raspberry Pi 2, il a :

- Processeur intégré Quad-core ARM Cortex-A53 1.2 GHz (Broadcom BCM2837)
- RAM : 1024 Mo
- GPU Dual Core Video Core IV Multimedia Co-Processor
- Lecteur de cartes Micro SD
- Ports : HDMI, 4x USB, RJ45, jack 3.5 mm, connecteurs pour APN et écran tactile
- Wi-Fi b/g/n et Bluetooth 4.1
- Support des distributions dédiées basées sur Linux et Windows 10. [8]

3.2 La Consommation électrique de Raspberry pi 3 B

- Alimentation type chargeur Smartphone
- 700 mA (3.5 W)
- Estimation coût +/- 4 € pour 1 an
- Pas de bouton ON/OFF



Figure II.7 : *Alimentation électrique de Raspberry pi 3 B*

3.3 Les composants standards de Raspberry

Afin de bien visualiser ce qu'il est possible d'adjoindre à la carte Raspberry Pi nue, la figure III.8 suivante représente tout à la fois son organisation générale et les différents connecteurs disponible d'un Raspberry standard :

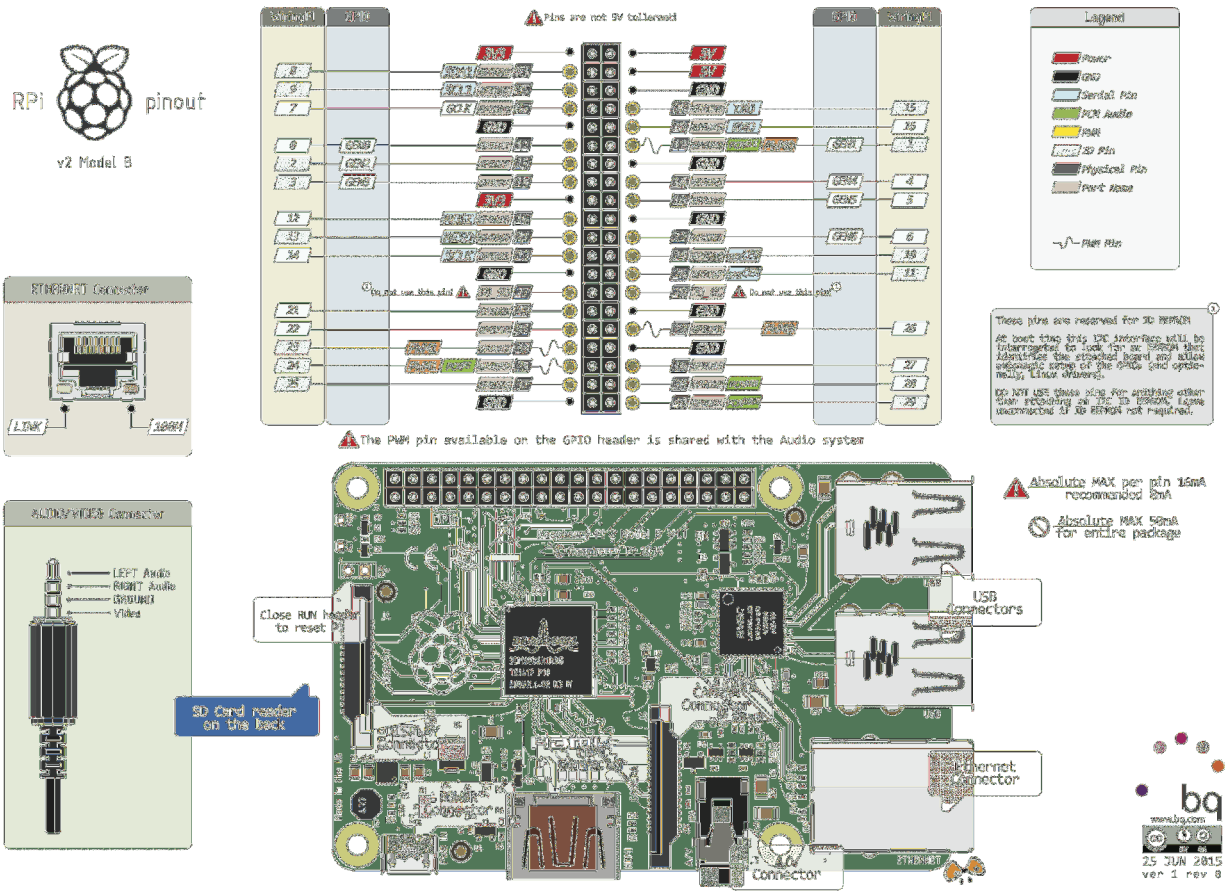


Figure II.8 : *Les composants standards d'un Raspberry.*

Le centre de la carte est occupé par le processeur ou, plus exactement, par un empilement du processeur et de sa mémoire qui sont littéralement montés l'un sur l'autre. [8]

La connectique qui occupe la majorité de la surface de la carte et non les composants actifs, Dans ce qui va suivre nous détaillerons les différents de la carte Raspberry pi 3 Model B :

- Connecteurs à câble plat flexible (FFC) à la Raspberry Pi 3B :

  - ***Connecteur d'affichage :*** près du connecteur micro-USB, est l'interface série d'affichage (DSI). Il permet une interface de bas niveau avec les écrans LCD et autres écrans avec Raspberry Pi. Il s'agit d'un connecteur plat flexible monté en surface à 15 broches, offrant deux voies de données, une voie d'horloge, 3.3V et GND.

- ***Connecteur de caméra :*** Situé entre le connecteur LAN et HDMI, est le connecteur d'interface MIPI Camera Serial Interface 2 (CSI-2) pour les modules de caméra. Il s'agit d'un connecteur flexible plat à 15 broches monté en surface, fournissant deux lignes de données, une voie d'horloge, une interface de commande bidirectionnelle compatible avec I2C, 3.3V et GND.

- La prise réseau accepte une prise normalisée RJ45 et tout réseau Ethernet Classique à 10 Mbits ou 100 Mbits par seconde.

- Quatre connecteur USB voisin et quant à lui conforme à la norme USB 2.0, rétro-compatible USB 2.0 comme il se doit.

- ***HDMI :*** (Hight Définition MultiMedia Interface) permet de relier le Raspberry Pi a un dispositif (écran LCD ou un vidéo projecteur), capable de surcroît de fournir de la vidéo au format full HD de 1080p, et véhicule également le son.

- ***Un slot les cartes SD :*** le Raspberry a besoin d'une mémoire externe supplémentaire pour fonctionner. Ce slot permet de connecter la mémoire externe.

- ***Prise jack audio/vidéo :*** Prise audio combinée de 3,5 mm et vidéo composite. [8]

3.4 Spécifications physiques

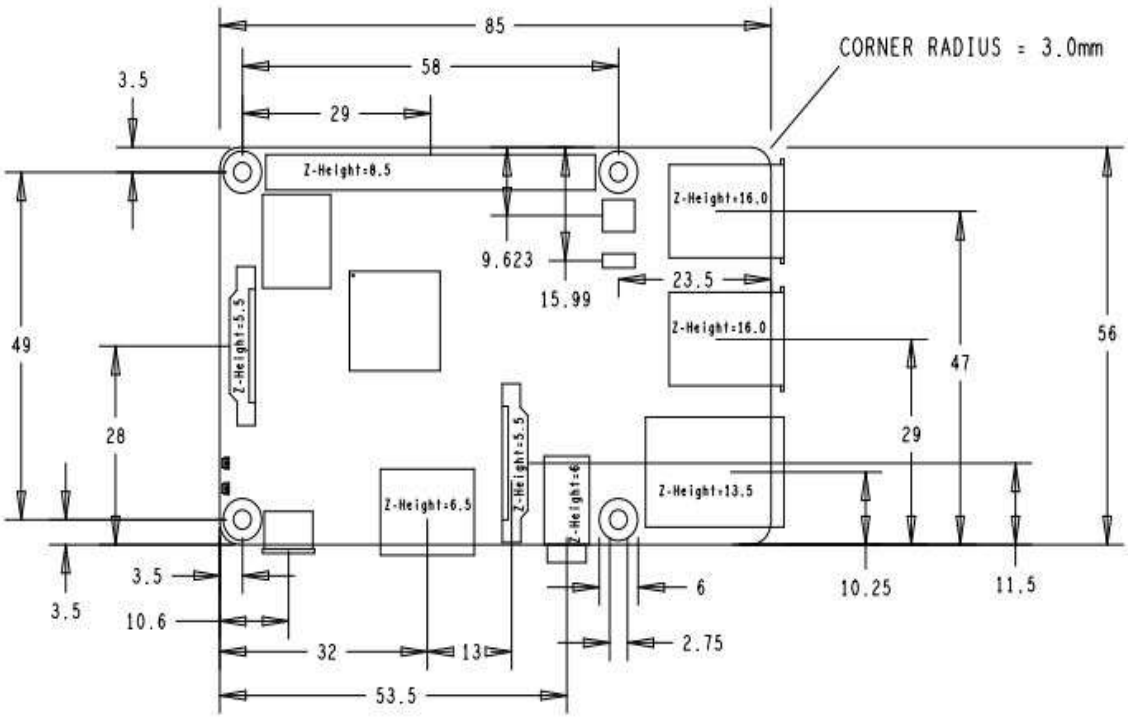


Figure II.9 : *Spécifications physiques d'un Raspberry pi*

3.5. Mise en marche d'un Raspberry Pi 3

Pour faire fonctionner notre Raspberry Pi, nous avons besoin des accessoires suivants :

- ***Une carte mémoire SD :*** Cette carte va contenir le système d'exploitation du Raspberry Pi, il ne pourra donc pas démarrer sans cette dernière. La fondation recommande une carte d'au moins 4 Go et de classe 4 minimum. [8]

- ***Alimentation micro USB externe :***L'alimentation peut être la cause de nombreux problèmes sur lesquels nous aurons l'occasion de revenir à plusieurs reprises. Le Raspberry Pi ne peut pas être alimenté par le port USB d'un ordinateur ou d'une télévision, il faut par conséquent utiliser une alimentation USB dédiée, Ce dernier s'alimente sous une tension unique de 5 volts ; tension sur laquelle il peut consommer jusqu'à 750 mA selon les taches qu'il exécute. Cette alimentation doit être normalement fournie via le connecteur micro USB placé dans un angle de la carte. [8]

- ***Un clavier et une souris USB :*** En cas général, pour faire fonctionner notre Raspberry Pi nous avons besoin d'un clavier et une souris, rien de bien particulier n'est à signaler en ce qui les concerne. N'importe quel clavier USB et n'importe quelle souris USB conviennent au Raspberry Pi et se connectent directement dans les deux ports USB prévus à cet effet sur le modèle que nous allons utiliser le modèle B.Mais attention car certains claviers ou certaines souris peuvent être de gros consommateurs d'énergie, il sera donc nécessaire dans ce cas de les brancher sur un HUB USB alimenté par une alimentation externe. C'est souvent le cas par exemple des claviers et des souris sans fils. La souris n'est pas indispensable mais elle devient très utile si on passe du mode texte au mode graphique. [8]
- ***Un câble HDMI :*** Par défaut, le Raspberry Pi délivre des signaux à la norme HDMI 1.4 comportant vidéo et son, Il peut donc être raccordé directement à tout moniteur informatique ou à tout téléviseur à écran plat disposant d'une telle entrée, Si l'écran choisi est équipé de haut-parleurs, ces derniers reproduiront le son fourni par le Raspberry Pi puisque celui-ci est véhiculé par la liaison HDMI, C'est bien sur les cas des téléviseurs à écran plat mais pas nécessairement celui des moniteurs informatiques, Dans une telle situation, nous pourrons alors récupérer le son depuis la prise jack stéréo de 3,5 mm que nous avons vu, En fonction du type d'écran que nous allons utiliser, nous devons nous procurer le câble

correspondant, L'usage de la sortie HDMI est bien entendu recommandé, l'image sera de meilleure qualité et le son sera intégré. [8]

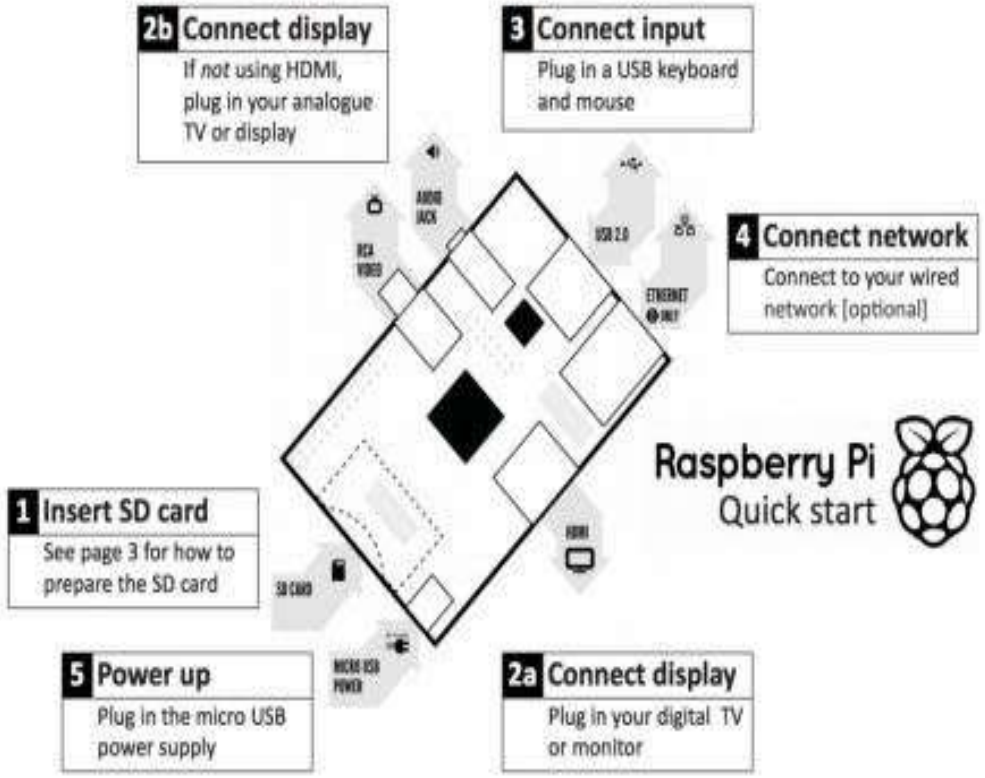


Figure II.10 : *Branchements de carte Raspberry pi*

3.6. Programmation

Il existe plusieurs langages de programmation qui peut être compilé sur le Raspberry Pi, telle que le langage C, C++, HTML5, le Java, le JavaScript...etc. Le langage utilisant dans notre projet et le Python.

Le Python est un langage de programmation inventé par Guido van Rossum, la première version de python est sortie en 1991.

C'est un langage de programmation interprété, c'est à dire qu'il n'est pas nécessaire de le compiler avant de l'exécuter.

Python étant un langage très riche, surtout si l'on prend en compte les nombreuses Bibliothèques complémentaires disponibles, Ce dernier est à la fois simple et puissant, c'est un langage facile à apprendre et son code est plus lisible, il est donc plus facile à maintenir. Il est parfois jusqu'à 5 fois plus concises que le langage Java par exemple, ce qui augmente la productivité du développeur et réduit mécaniquement le nombre de bugs. [8]

3.7. Système d'exploitation

La plupart des systèmes qui fonctionnent sur Raspberry Pi sont des versions du système d'exploitation Linux. Parce que Linux est open source, les développeurs peuvent l'adopter pour des buts spécifiques. Dans le cas du Raspberry Pi, le matériel léger nécessite un système réduit et privé d'applications inutiles.

La distribution est recommandée Raspbian. C'est un système d'exploitation libre basé sur Debian optimisé pour le matériel Raspberry Pi. Cependant Raspbian fournit plus d'un simple système d'exploitation : il est livré avec plus de 35.000 paquets, des logiciels précompilés qui nous facilitent le développement soft. [8]

Tous langages qui peuvent être compilés sur ARMv7 sont installés par défaut sur le Raspberry Pi : Python, C, C++, Java, Scratch et Ruby. La fondation Raspberry Pi recommande Python comme langage pour les étudiants.

3.8. Synthèse

La haute performance de l'Arduino uno et de Raspberry Pi 3B Insuffisantes pour atteindre, le bon performance à notre Robot, Le Raspberry va se charger de la partie acquisition, traitement d'images et calcul des lois de commandes qu'il va transmettre à la carte Arduino.

Tous ces processus font fonctionner lentement et cela affecte de manière significative à la vitesse de traitement, Pour résoudre ce problème et réduire la pression sur le Raspberry Ce qui est causée par le traitement d'images, Parmi les solutions l'utilisation du Pixy Cam cette caméra possède un processeur graphique qui sera fait le traitement d'image au lieu de faire ça dans le Raspberry et on a gardé comme cerveau pour calcul des lois des commandes.

Dans la partie suivante nous allons présenter la Cam Pixy qui nous allons utiliser dans notre projet. [8]

4. L'extension (Shield) de commande des moteurs

Cette extension (Adafruit Motor Shield) est basée sur le circuit CI L293D, qui est un double pilote à pont complet conçu pour gérer des charges inductives telles que les relais, les solénoïdes, les moteurs DC et les servomoteurs. Il vous permet de commander 4 moteurs DC avec votre carte Arduino, en contrôlant la vitesse et la direction de chacun d'eux indépendamment Adafruit Motor Shield ne doit être alimenté que par une alimentation externe, Parce que le L293D monté sur le blindage comporte deux connexions d'alimentation séparées, l'une pour la logique et l'autre pour l'alimentation des charges inductives. Le courant requis du moteur dépasse souvent la note maximale du courant USB.

L'alimentation externe (non-USB) peut provenir d'un adaptateur AC-DC ou d'une batterie.

L'adaptateur peut être branché en branchant une prise centrale positive de 2,1 mm dans la prise d'alimentation de la carte Arduino sur laquelle le blindage du moteur est monté ou en connectant les fils qui alimentent l'alimentation des bornes à vis (la pin Vin et GND). [9]

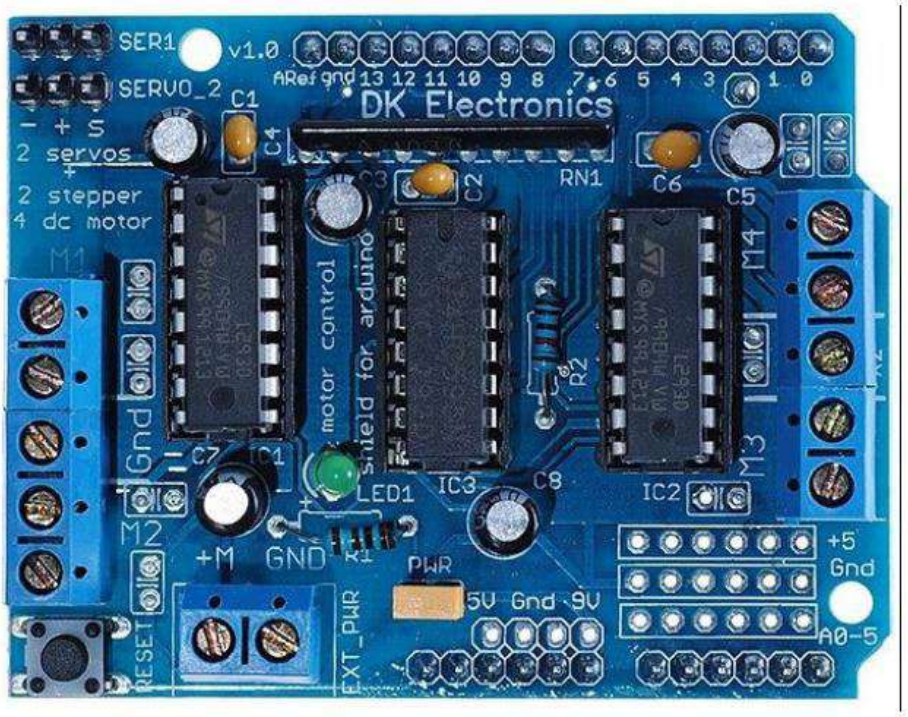


Figure II.11: *Adafruit Motor Shield.*

- ***Principales caractéristiques :***

- Tension d'alimentation du circuit logique : 5V
- Tension d'alimentation du circuit puissance : 0V→ 8V
- Tension minimale de sensibilité MLI : 2.3V
- Courant maximal du circuit puissance : 3A

- ***Les moteurs et le registre à décalage :***

Le blindage du moteur peut contrôler jusqu'à quatre moteurs à courant continu à l'aide de deux circuits intégrés de moteur à moteur à double pont L293D. Vous pouvez lire la fiche technique pour plus de détails, Comme le montre la Figure III.12, chaque CI possède quatre entrées numériques qui commandent bi directionnellement deux moteurs à courant continu. Les entrées sont par paires, et chaque paire contrôle un moteur. Par exemple, si la broche 2 est réglée en haut et que la broche 7 est bas, le moteur 1 est allumé, Si les entrées sont inversées, le moteur est à nouveau activé, mais son sens de rotation est inversé, et si les deux sont bas, le moteur est éteint. Il existe d'autres configurations possibles, mais celle-ci est supportée par le bouclier du moteur. [9]

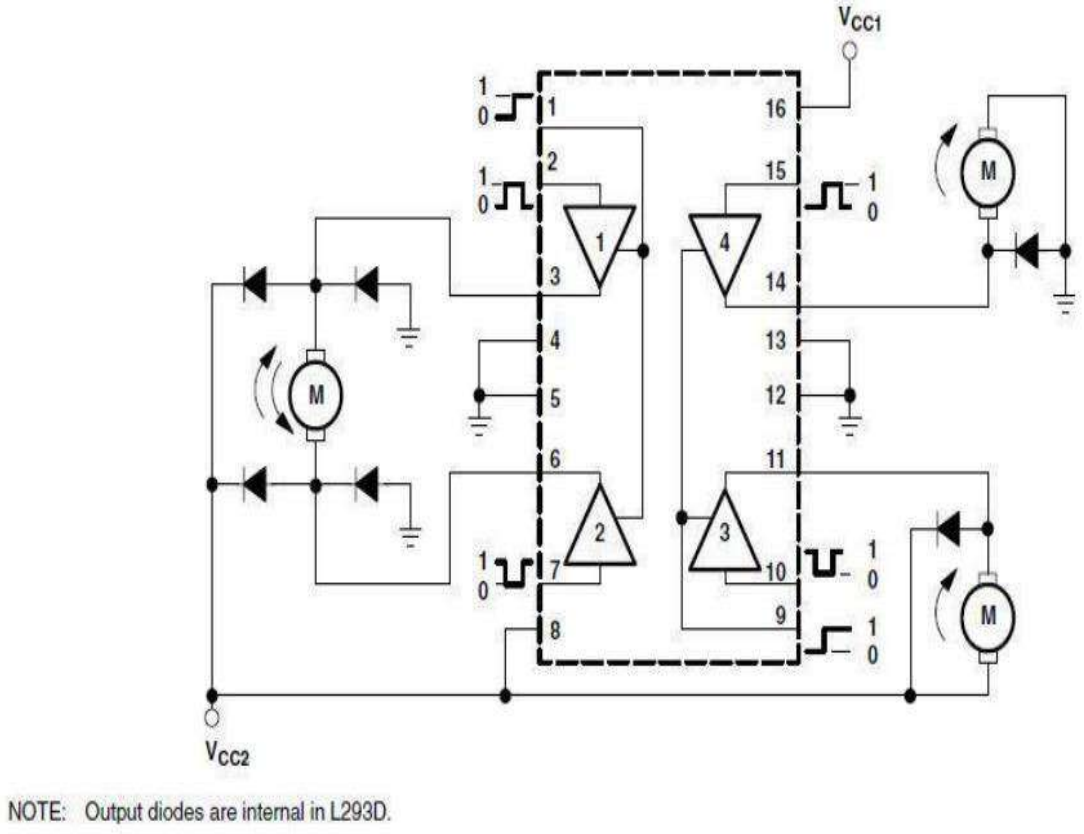


Figure II.12 : *fonctionnement de L293D.*

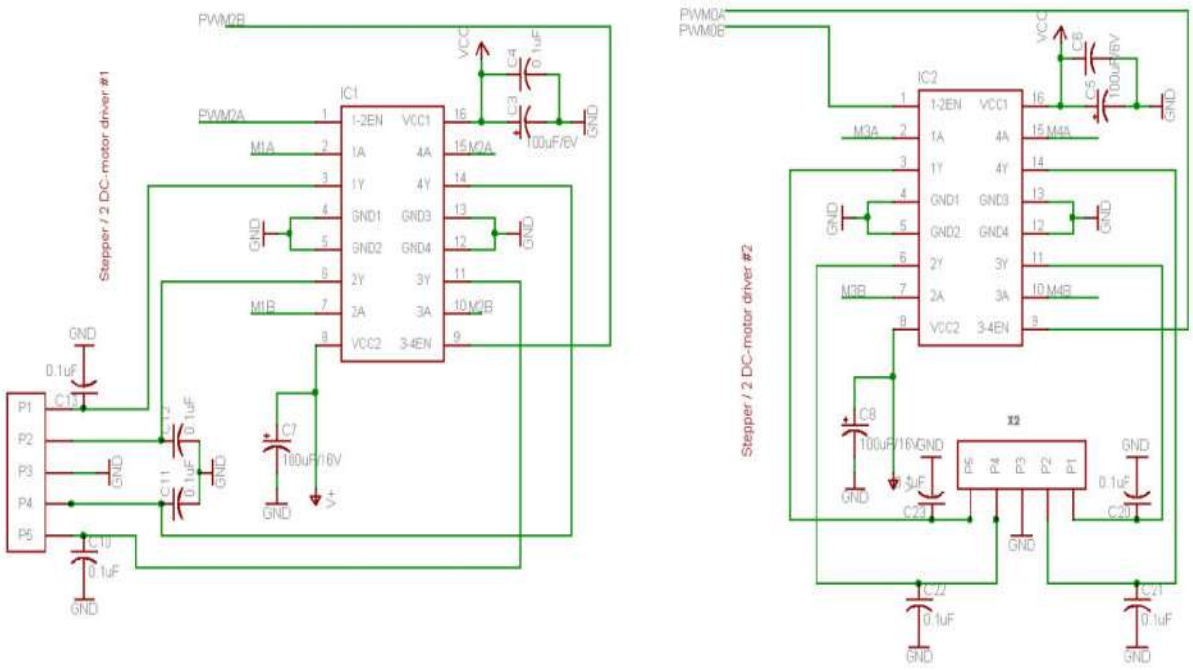


Figure II.13: *montage L293D avec Shield.*

Les entrées des moteurs sont effectivement câblées sur les sorties d'un registre à décalage 74HCT595N à 8 bits Figure 4.8 Cet IC vous permet d'alimenter en bits en série, mais il les présente ensuite comme des sorties parallèles. Il est contrôlé par quatre entrées numériques, de sorte qu'il vous permet de contrôler huit entrées aux moteurs (donc quatre moteurs) en utilisant quatre broches de l'Arduino. [9]

5. La caméra Pixy

5.1. Aperçu

- Système de vision petit, rapide, facile à utiliser, peu coûteux et facilement accessible.
- Apprends à détecter des objets que vous l'enseignez.
- Indique ce qu'il détecte 50 fois par seconde.

- Se connecte à Arduino avec un câble inclus. Fonctionne également avec Raspberry Pi, BeagleBoneet des contrôleurs similaires.

- Toutes les bibliothèques pour Arduino, Raspberry Pi, etc. sont fournies.
- C / C ++ et Python sont pris en charge.
- Communique via l'une des nombreuses interfaces : SPI, I2C, UART, USB ou sortie analogique / numérique.
- L'utilitaire de configuration s'exécute sous Windows, MacOs et Linux.
- Tout logiciel / firmware est open-source GNU-licence.
- Toute la documentation sur le matériel, y compris les schémas, la liste de matériaux, la mise en page des PCB, etc. [10]



Figure II.14 : *Pixy CMUcam5*

5.2. Bref historique

Pixy (CMUcam5) est un partenariat entre Carnegie Mellon Robotiques Institute et Charmed Labs, Pixy vient d'une longue ligne de CMUcams, mais Pixy a vraiment démarré comme une campagne de Kick Starter. Il a d'abord commencé à être livré en mars 2014, mais c'est déjà devenu le système de vision le plus populaire de l'histoire. [10]

5.3. Simplicité de mise en œuvre du système de vision

Si vous souhaitez que votre robot effectue une tâche, comme la collecte d'un objet, la poursuite d'une balle, la localisation d'une station de chargement etc, et que vous souhaitez qu'un seul capteur aide à accomplir toutes ces tâches, la vision est votre capteur. Les capteurs Vision (image) sont utiles car ils sont si flexibles Avec le bon algorithme, un capteur d'image peut détecter ou détecter pratiquement n'importe quoi. Mais il existe deux inconvénients avec les capteurs d'image :

- ils sortent beaucoup de données, des dizaines de mégaoctets par seconde.
- le traitement de cette quantité de données peut submerger de nombreux processeurs, Et si le processeur peut suivre les données, une grande partie de sa puissance de Traitement ne sera pas disponible pour d'autres tâches.

Pixy résout ces problèmes en associant un puissant processeur dédié au capteur d'image. Pixy traite les images du capteur d'image et envoie uniquement les informations utiles (par exemple, le dinosaure violet détecté à x = 54, y = 103) à votre microcontrôleur, Et cela se fait à cadence (50 Hz). L'information est disponible via l'une des nombreuses interfaces : UART série, SPI, I2C, USB ou sortie numérique / analogique. Ainsi, votre Arduino ou un autre microcontrôleur peut facilement parler avec Pixy et dispose de nombreuses CPU disponibles Pour d'autres tâches.

Il est possible de brancher plusieurs Pixys sur votre microcontrôleur - par exemple, un robot avec 4 Pixys et 360 degrés de détection, Ou utiliser Pixy sans microcontrôleur et utiliser les sorties numériques ou analogiques pour déclencher des événements, des commutateurs, des servons, etc. [10]

5.4. Support du contrôleur

Pixy peut facilement se connecter à de nombreux contrôleurs différents car il prend en charge plusieurs options d'interface (UART série, SPI, I2C, USB ou sortie numérique / analogique), mais Pixy a commencé sa vie à parler à Arduino. Au cours des derniers mois, nous avons ajouté un support pour Arduino Due, Raspberry Pi et BeagleBone Black. Des bibliothèques de logiciels sont fournies pour toutes ces plates-formes afin que vous puissiez vous lancer rapidement. En outre, nous avons ajouté une API Python si vous utilisez un contrôleur basé sur Linux (par exemple, Raspberry Pi, BeagleBone). [10]

5.5. La détection d'objets

Pixy utilise un algorithme de filtrage basé sur les couleurs pour détecter les objets, Les méthodes de filtrage basées sur les couleurs sont populaires car elles sont rapides, efficaces et relativement robustes. La plupart d'entre nous connaissent le RGB (rouge, vert et bleu) pour représenter les couleurs. Pixy calcule la couleur (teinte) et la saturation de chaque pixel RGB du capteur d'image et les utilise comme paramètres de filtrage primaires. La teinte d'un objet reste en grande partie inchangée avec les changements d'éclairage et d'exposition. Les changements d'éclairage et d'exposition peuvent avoir un effet frustrant sur les algorithmes de filtrage des couleurs, ce qui leur permet de se casser. L'algorithme de filtrage de Pixy est robuste en matière d'éclairage et de changements d'exposition.

Pixy peut trouver littéralement des centaines d'objets à la fois. Il utilise un algorithme de composants connectés pour déterminer où un objet commence et un autre se termine. Pixy compile les tailles et les emplacements de chaque objet et les signale à travers l'une de ses interfaces (par exemple SPI).

Dans la documentation technique de ce système de vision, nous retrouvons souvent le terme "50 images par seconde" ou "50 frames par seconde". En bref, cela signifie que Pixy est rapide. Pixy traite toute une image 640x400 toutes les 1 / 50ème de seconde (20 millisecondes). Cela signifie que vous obtenez une mise à jour complète de toutes les positions des objets détectés toutes les 20 ms. À ce rythme, il est possible de suivre le

parcours de la balle tombante / rebondissant. (Une balle qui se déplace à 30 mph se déplace moins d'un pied en 20 ms.) [10]

5.6. Définition simple de l'objet à suivre

Pixy est unique du fait que l'on peut rapidement et simplement définir l'objet à suivre. Par exemple si cet objet est un dinosaure violet placez le dinosaure devant l'objectif de la caméra Pixy et appuyez sur le seul bouton, Boule orange ? Placez le ballon devant Pixy et appuyez sur le bouton. C'est simple, et c'est rapide.

Plus précisément, vous enseignez à Pixy en tenant l'objet devant son objectif tout en maintenant le bouton situé en haut. En faisant cela, la LED RGB sous l'objectif fournit des commentaires concernant l'objet qu'il regarde directement. Par exemple, la LED devient orange lorsqu'une boule orange est placée directement devant Pixy. Relâchez le bouton et Pixy génère un modèle statistique des couleurs contenues dans l'objet et les stocke en flash. Il utilisera ensuite ce modèle statistique pour trouver des objets avec des signatures de couleurs similaires dans leur cadre à partir de là.

Pixy peut apprendre sept signatures couleur, numérotées 1-7. La signature couleur 1 est la signature par défaut. Pour enseigner à Pixy, les autres signatures (2-7) nécessitent un simple bouton qui appuie sur la séquence. [10]

5.7. L'application PixyMon

PixyMon est une application qui fonctionne sous Windows, MacOs et Linux. Il vous permet de voir ce que Pixy voit, en tant que vidéo brute ou transformée. Il vous permet également de configurer votre Pixy, de définir le port de sortie et de gérer les signatures des couleurs. PixyMon communique avec Pixy sur un câble mini USB standard.

PixyMon est idéal pour le débogage de votre application, Vous pouvez brancher un câble USB à l'arrière de Pixy et exécuter PixyMon, puis voir ce que Pixy voit lorsqu'il est accroché à votre Arduino ou à un autre microcontrôleur - pas besoin de débrancher quoi que ce soit. PixyMon est open source, comme tout le reste. [10]

5.8. Spécifications techniques

La puissance du système de vision Pixy est l'intégration sur une seule carte à la fois du système optique de vision est des composants électroniques (processeur, RAM, …) nécessaires à l'acquisition et aux traitements des images. [10]

- ***Processeur**:* NXP LPC4330, 204 MHz, dual core
- ***Capteur d'image**:* Omnivision OV9715, 1/4 ", 1280x800
- ***Lens field-of-view:*** 75 degrés horizontal, 47 degrés vertical
- ***Type d'objectif:*** standard M12 (plusieurs types différents disponibles)
- ***Consommation électrique***: 140 mA typique
- ***Entrée d'alimentation:*** entrée USB (5V) ou entrée non régulée (6V à 10V)
- ***RAM:*** 264K octets
- ***Flash:*** 1M octets
- ***Sorties de données disponibles:*** UART série, SPI, I2C, USB, numérique, analogique
- ***Dimensions:*** 2.1 "x 2.0" x 1.4
- ***Poids:*** 27 grammes

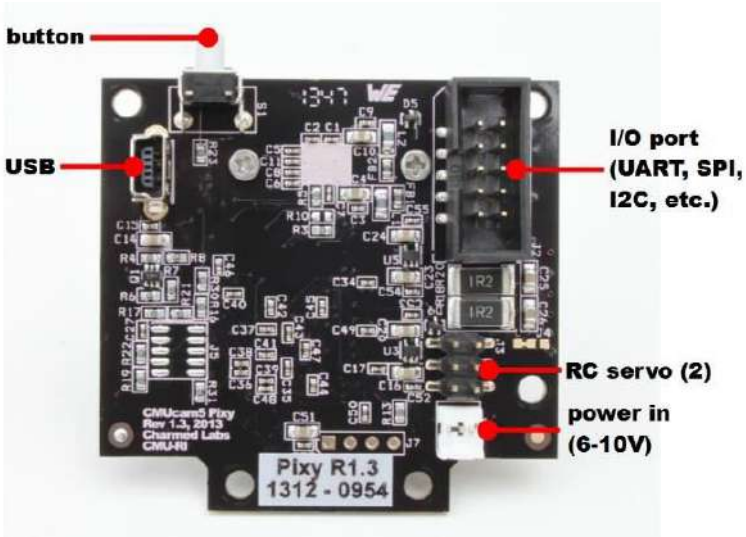


Figure II.15 : *structure de la Cam Pixy CMUcam5.*

6. Conclusion

Ce chapitre a été pour nous l'occasion de présenter les cartes Raspberry PI 3 B et Arduino ainsi que le système de vision Pixy et la carte shield l293D que nous avons utilisé pour concevoir l'architecture matérielle nécessaire à la commande du robot mobile.

L'asservissement de ce robot basé sur l'utilisation des données visuelles, nécessite d'une part une puissance de calcul importante et d'autre part des calculateurs peu volumineux et léger pouvant être embarqués sur la structure mécanique du robot mobile.

La solution que nous avons adoptée et qui sera présentée dans le dernier chapitre a considérablement amélioré les performances et le traitement d'image et d'asservissement visuel. De plus, cette configuration nous permettra d'implémenter notre contrôleur neuronal. Ce dernier basé sur le réseau de neurones ADALINE nécessite l'utilisation de ressources informatiques importantes (mémoire, puissance de calcul, …) que notre architecture est capable de fournir. Mais avant de détailler le fonctionnement de ce contrôleur, nous avons choisi, dans le chapitre qui suit, de vous décrire et d'évaluer les performances du réseau de neurone ADALINE sur des fonctions « *simple* ».

Chapitre III :

Les réseaux de neurones

1. Introduction

Pendant des décennies, les scientifiques ont été hantés du rêve de développer des machines intelligentes avec un grand nombre d'éléments simples en s'inspirant des cerveaux humains. Ce vieux rêve a vu le jour pendant les années 1940 quand les chercheurs ont développé des modèles simples des neurones biologiques. McCulloch et Pitts ont posé la première pierre en publiant leur étude systématique du réseau neurologique artificiel. Et ce n'était que le début bien que les réseaux de neurones aient connu des hauts et des bas dans les années 50 et 60.Cependant, les chercheurs, tels que Minsky et Papert, ont fortement défié ces travaux.

Le paradigme de réseaux de neurones a connu son essor dans les années quatre-vingts avec les extensions de travaux d'éminents chercheurs tels que ceux de Rosenblatt, de Widrow et de Hoff qui traitent l'apprentissage dans un réseau complexe multicouche.

L'intérêt pour les réseaux neurologiques vient de la capacité des réseaux d'imiter le cerveau humain aussi bien que sa capacité d'apprendre et répondre. En conséquence, des réseaux neurologiques ont été employés dans un grand nombre d'applications, et se sont avérés efficaces en exécutant des fonctions complexes dans une variété de champs. Ceux-ci incluent l'identification de modèle, la classification, la commande des systèmes, et la prévision [11].

Ce chapitre est consacré à la présentation des réseaux de neurones, et à l'utilisation des réseaux de neurones pour la commande robotique. Mais avant de développer leur application ce processus complexe, nous vous illustrons leurs fonctionnements avec des fonctions simples.

2. Définitions

Les réseaux de neurones artificiels ou réseaux connexionnistes sont fondés sur des modèles qui tentent de mimer les cellules du cerveau humain et leurs interconnexions. Le but d'un point de vue global, et d'exécuter des calculs complexes et de trouver, par apprentissage une relation non linéaire entre des données numériques et des paramètres.

3. Historique

3.1. W. James(1890)

L'origine de l'inspiration des réseaux de neurones artificiels remonte à 1890 où W.James, célèbre psychologue américain, introduit le concept de mémoire associative. Il propose ce qui deviendra une loi de fonctionnement pour l'apprentissage des réseaux de neurones, connue plus tard sous le nom de loi de Hebb. Le champ des réseaux neuronaux va démarrer par la présentation en 1943 par W. McCulloch et W. Pitts du neurone formel [12].

3.2. Le neurone formel de McCulloch et Pitts(1943)

Le premier modèle mathématique et informatique du neurone biologique est proposé par Warren McCulloch et Walter Pitts en 1943. En s'appuyant sur les propriétés des neurones biologiques connues à cette époque, issues d'observations neurophysiologiques et anatomiques, McCulloch et Pitts proposent un modèle simple de neurone formel. Il s'agit d'un neurone binaire, c'est-à-dire dont la sortie vaut 0 ou 1. Pour calculer cette sortie, le neurone effectue une somme pondérée de ses entrées (qui, en tant que sorties d'autres neurones formels, valent aussi 0 ou 1) puis applique une fonction d'activation à seuil : si la somme pondérée dépasse une certaine valeur, la sortie du neurone est 1, sinon elle vaut 0.

McCulloch et Pitts étudiaient en fait l'analogie entre le cerveau humain et les machines informatiques universelles. Ils montrèrent en particulier qu'un réseau (bouclé) constitué des neurones formels de leur invention a la même puissance de calcul qu'une machine de Turing. Malgré la simplicité de cette modélisation, ou peut-être grâce à elle, le neurone formel dit de McCulloch et Pitts reste aujourd'hui un élément de base des réseaux de neurones artificiels. De nombreuses variantes ont été proposées, plus ou moins biologiquement plausibles, mais s'appuyant généralement sur les concepts inventés par les deux auteurs. On sait néanmoins

aujourd'hui que ce modèle n'est qu'une approximation des fonctions remplies par le neurone réel et, qu'en aucune façon, il ne peut servir pour une compréhension profonde du système nerveux.

3.3. D. Hebb(1949)

Physiologiste américain explique le conditionnement chez l'animal par les propriétés des neurones eux-mêmes. Ainsi, un conditionnement de type pavlovien tel que, nourrir tous les jours à la même heure un chien, entraîne chez cet animal la sécrétion de salive à cette heure précise même en l'absence de nourriture. La loi de modification des propriétés des connexions entre neurones qu'il propose explique en partie ce type de résultats expérimentaux [13].

3.4. Les premiers succès

3.4.1. F. Rosenblatt (1957)

Développe le modèle du Perceptron. Il construit le premier neuro ordinateur basé sur ce modèle et l'applique au domaine de la reconnaissance de formes. notons qu'à cet époque les moyens à sa disposition sont limités et c'est une prouesse technologique que de réussir à faire fonctionner correctement cette machine plus de quelques minutes,

Le perceptron se voulait un modèle de l'activité perspective. Il se compose d'une rétine et d'une couche qui donne la réponse correspondant à la simulation donnée en entrée.

3.4.2.B. Widrow(1960)

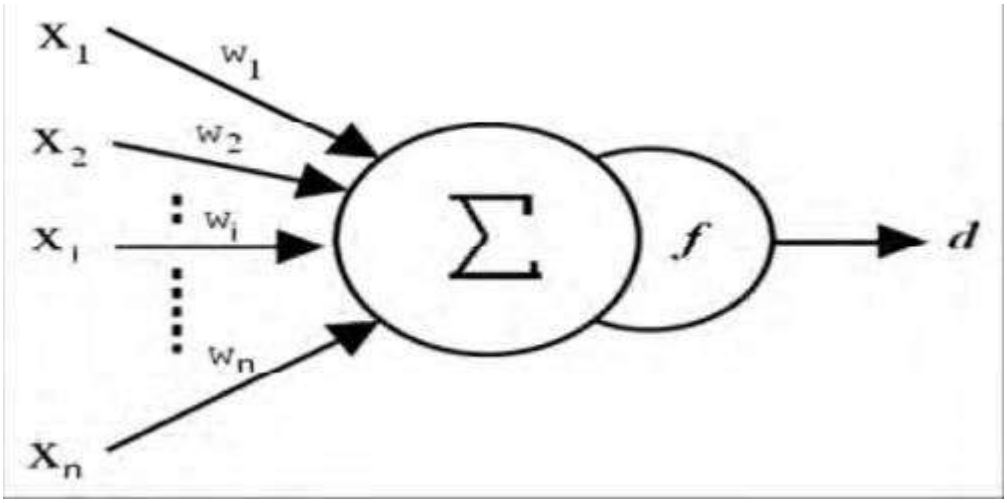


Figure III.1. *Modèle de l'activité perspective*

Un automaticien, développe le modèle ADALINE (Adaptive Linear Neuron ou plus tard Adaptive Linear Element) est un réseau de neurones artificiels simple-couche. Adaline s'appuie sur le neurone formel de McCulloch et Pitts. Il se compose d'un poids synaptique, d'un biais (une constante qu'on ajoute à l'entrée) et d'une fonction de sommation. La différence entre Adaline et le modèle de McCulloch-Pitts réside dans la phase d'apprentissage, où les poids synaptiques sont ajustés en fonction de la somme pondérée des données en entrées. Dans le perceptron standard (McCulloch-Pitts), le réseau passe par la fonction d'activation (ou fonction de transfert) et la fonction de sortie est utilisée pour ajuster les poids synaptiques, Il existe aussi une extension appelée Madaline.

En 1969, dans Perceptrons, Marvin Minsky coécrit avec Seymour Papert pour critiquer Frank Rosenblatt, il a montré les limites des réseaux de neurones de type perceptron, notamment l'impossibilité de traiter des problèmes non linéaires ou de connexité. Ceci a eu pour conséquence de drainer l'essentiel des crédits de recherche vers l'intelligence artificiell

Marvin Minsky défend l'idée que l'intelligence artificielle doit utiliser des approches multiples, notamment pour la représentation des connaissances, au lieu de se limiter à une seule approche qui serait censée être la meilleure. Les systèmes doivent disposer de "gestionnaires", capable de sélectionner les meilleures solutions à un problème donné. Pour représenter les connaissances.

3.5. L'ombre :

3.5.1. (1967-1982)

Toutes les recherches ne sont, bien sûr, pas interrompues. Elles se poursuivent, mais déguisées, sous le couvert de divers domaines comme : le traitement adaptatif du signal, la reconnaissance de formes, la modélisation en neurobiologie, etc. De grands noms travaillent durant cette période telle : S. Grossberg, T. Kohonen, ... dont nous reparlerons

3.6. Le renouveau

3.6.1. J. J. Hopfield(1982)

Est un physicien reconnu à qui l'on doit le renouveau d'intérêt pour les réseaux de neurones artificiels. A cela plusieurs raisons : Au travers d'un article court, clair et bien écrit, il présente une théorie du fonctionnement et des possibilités des réseaux de neurones; J. J. Hopfield fixe préalablement le comportement à atteindre pour son modèle et construit à partir de là, la structure et la loi d'apprentissage correspondant au résultat escompté. Ce modèle est aujourd'hui encore très utilisé pour des problèmes d'optimisation. D'autre part, entre les mains de ce physicien distingué, la théorie des réseaux de neurones devient respectable. Elle n'est plus l'apanage d'un certain nombre de psychologues et neurobiologistes hors du coup.

Notons qu'à cette date, l'IA est l'objet d'une certaine désillusion, elle n'a pas répondue à toutes les attentes et s'est même heurtée à de sérieuses limitations. Aussi, bien que les limitations du Perceptron mise en avant par M. Minsky ne soient pas levées par le modèle d'Hopfield, les recherches sont relancées.

3.7. La levée des limitations

3.7.1. Les machines de Boltzmann (1983)

La Machine de Boltzmann est le premier modèle connu apte à traiter de manière satisfaisante les limitations recensées dans le cas du perceptron.

Mais l'utilisation pratique s'avère difficile, la convergence de l'algorithme étant extrêmement longue (les temps de calcul sont considérables).

Les machines de Boltzmann sont un réseau de neurones récurrent et multicouches.

Ce fut l'un des premiers réseaux de neurone capable d'apprendre des représentations internes et de résoudre des problèmes combinatoires difficiles.

Les connexions des machines de Boltzmann sont généralement restreintes pour pouvoir résoudre efficacement des problèmes pratiques.

4. Utilité

Les réseaux de neurones, en tant que système capable d'apprendre, mettent en œuvre le principe de l'induction, c'est-à-dire l'apprentissage par l'expérience. Par confrontation avec des situations ponctuelles, ils infèrent un système de décision intégré dont le caractère générique est fonction du nombre de cas d'apprentissages rencontrés et de leur complexité par rapport à la complexité du problème à résoudre.

Par opposition, les systèmes symboliques capables d'apprentissage, s'ils implémentent également l'induction, le font sur base de la logique algorithmique, par complexification d'un ensemble de règles déductives (ex : prolog).

Grâce à leur capacité de classification et de généralisation, les réseaux de neurones sont généralement utilisés dans des problèmes de nature statistique, tels que la classification automatique de codes postaux ou la prise de décision concernant un achat boursier en fonction de l'évolution des cours. Autre exemple, une banque peut générer un jeu de données sur les clients qui ont effectué un emprunt constituées :de leur revenu, de leur âge, du nombre d'enfants à charge… et s'il s'agit d'un bon client. Si ce jeu de données est suffisamment grand, il peut être utilisé pour l'entraînement d'un réseau de neurones. La banque pourra alors présenter les caractéristiques d'un potentiel nouveau client, et le réseau répondra s'il sera bon client ou non, en généralisant à partir des cas qu'il connaît.

Si le réseau de neurones fonctionne avec des nombres réels, la réponse traduit une probabilité de certitude (par exemple: 1 pour « sûr qu'il sera un bon client », -1 pour « sûr qu'il sera mauvais client », 0 pour « aucune idée », 0,9 pour « presque sûr qu'il sera bon client»).

Le réseau de neurones ne fournit pas toujours de règle exploitable par un humain. Le réseau reste souvent une boîte noire qui fournit une réponse quand on lui présente une donnée, mais le réseau ne fournit pas de justification facile à interpréter.

Les réseaux de neurones sont réellement utilisés, par exemple:

- Pour la classification ; par exemple pour la classification d'espèces animales par espèce étant donnée une analyse ADN.

- Reconnaissance de motif ; par exemple pour la Reconnaissance optique de caractères (OCR), et notamment par les banques pour vérifier le montant des chèques, par La Poste pour trier le courrier en fonction du code postal, etc. ; ou bien encore pour le déplacement automatisé de robots mobiles autonomes.

- Approximation d'une fonction inconnue.

- Modélisation accélérée d'une fonction connue mais très complexe à calculer avec exactitude ; par exemple certaines fonctions d'inversions utilisées pour décoder les signaux de télédétection émis par les satellites et les transformer en données sur la surface de la mer.
- Estimations boursières :

  - apprentissage de la valeur d'une entreprise en fonction des indices disponibles : bénéfices, endettements à long et court terme, chiffre d'affaires, carnet de commandes, indications techniques de conjoncture. Ce type d'application ne pose pas en général de problème
  - tentatives de prédiction sur la périodicité des cours boursiers : Ce type de prédiction est très contesté pour deux raisons, l'une étant qu'il n'est pas évident que le cours d'une action ait de façon tout à fait convaincante un caractère périodique (le marché anticipe en effet largement les hausses comme les baisses prévisibles, ce qui applique à toute périodicité éventuelle une variation de période tendant à la rendre difficilement fiable), et l'autre que l'avenir prévisible d'une entreprise détermine au moins aussi fortement le cours de son action, si ce n'est plus encore que peut le faire son passé; les cas de Pan Am, Manufrance ou IBM permettent de s'en convaincre.

- Modélisation de l'apprentissage et amélioration des techniques de l'enseignement.

5. Avantages

Les principales qualités des réseaux de neurones sont leur capacité d'adaptabilité et d'auto-organisation et la possibilité de résoudre des problèmes non-linéaires avec une bonne approximation [14]. Ils ont une bonne immunité aux bruits et se prêtent bien à une implantation parallèle. La rapidité d'exécution est une qualité importante et elle justifie souvent à elle seule le choix d'implanter un réseau de neurones. Ces qualités ont permis de réaliser avec succès, plusieurs applications : classification, filtrage, compression de données, contrôleur, etc...

6. Inconvénients

La difficulté d'interpréter le comportement d'un réseau de neurones est un inconvénient pour la mise au point d'une application. Il est souvent impossible d'utiliser les résultats obtenus pour améliorer ce comportement. Il est également hasardeux de généraliser à partir d'expériences antérieures et de conclure ou de créer des règles sur le fonctionnement et le comportement des réseaux de neurones.

Plusieurs paramètres doivent être ajustés et aucune méthode ne permet de choisir des valeurs optimales. Beaucoup d'heuristiques sont utilisées, mais elles se contredisent parfois et elles ne permettent pas toujours de trouver des valeurs optimales.

7. Les neurones biologiques

Les cellules nerveuses, appelées neurones, sont les éléments de base du système nerveux central. Celui-ci en posséderait environ cent milliards. Les neurones possèdent de nombreux points communs dans leur organisation générale et leur système biochimique avec les autres cellules ils présentent cependant des caractéristiques qui leur sont propres et se retrouvent au niveau des cinq fonctions spécialisées qu'ils assurent :

- Recevoir des signaux en provenance de neurones voisins.
- Intégrer ces signaux.
- Engendrer un influx nerveux.
- Le conduire.
- Le transmettre à un autre neurone capable de le recevoir.

7.1. Structure des neurones

Un neurone est constitué de trois parties :

- Le corps cellulaire
- Les dendrites.
- L'axone.

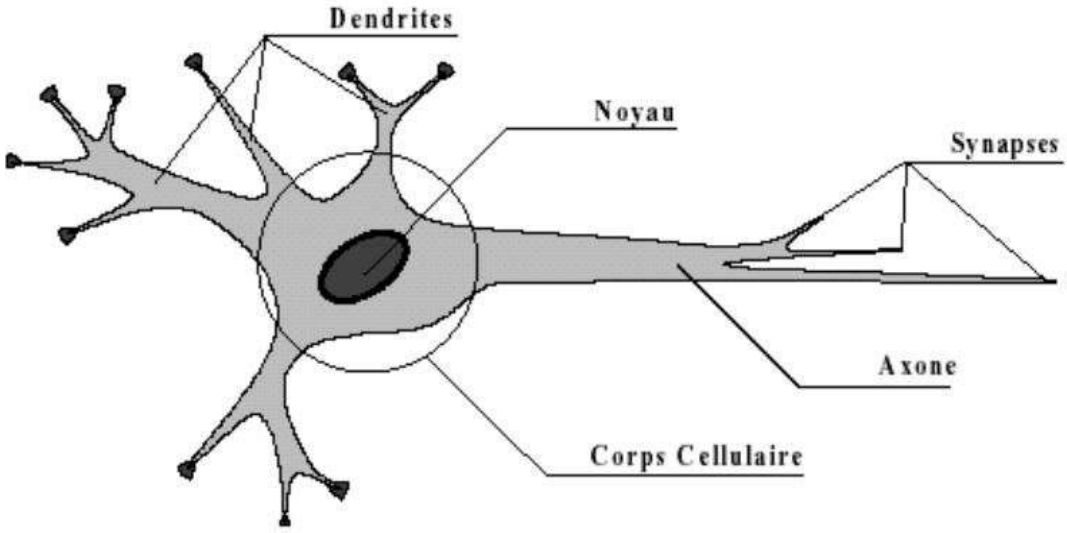


Figure III.2. *Neurone biologique*

7.1.1. Le corps cellulaire

Il contient le noyau du neurone et effectue les transformations biochimiques nécessaire à la synthèse des enzymes et des autres molécules qui assurent la vie du neurone, il a une forme pyramidales ou sphérique dans la plus part des cas.

7.1.2. Les dendrites

Chaque neurones possède une « chevelure » de dendrites Celles-ci sont de fines extensions tubulaires, de quelques dixième de microns. De diamètre et d'une longueur de quelques dizaine de microns. Elles sont les récepteurs principaux du neurone pour capter les signaux qui lui parviennent

7.1.3. L'axone

L'axone, qui est à proprement parler la fibre nerveuse, sert de moyen de transport pour les signaux émis par le neurone Pour former le système nerveux, les neurones sont

connectés les uns aux autres suivants des répartitions spatiales complexes, les connexions entre deux neurones se font en des endroits appelés synapses où ils sont séparés par un petit espace synaptique de l'ordre d'un centième de micron. D'une façon simple, on peut dire que le soma du neurone traite les courants électriques qui lui proviennent de ses dendrites, et qu'il transmet le courant électrique (sous forme d'impulsions chaque une de durée d'environ 1ms et une amplitude d'environ 100mv) résultant de ce traitement aux neurones auxquels il est connecté par l'intermédiaire de son axone.

Le schéma classique présenté par les biologistes est celui d'un soma effectuant une sommation des influx nerveux transmis par ses dendrites Si la sommation dépasse un certain seuil, le neurone répond par un influx nerveux au potentiel d'action qui se propage le long de son axone. Si la sommation est inférieure à ce seuil, le neurone reste inactif [14].

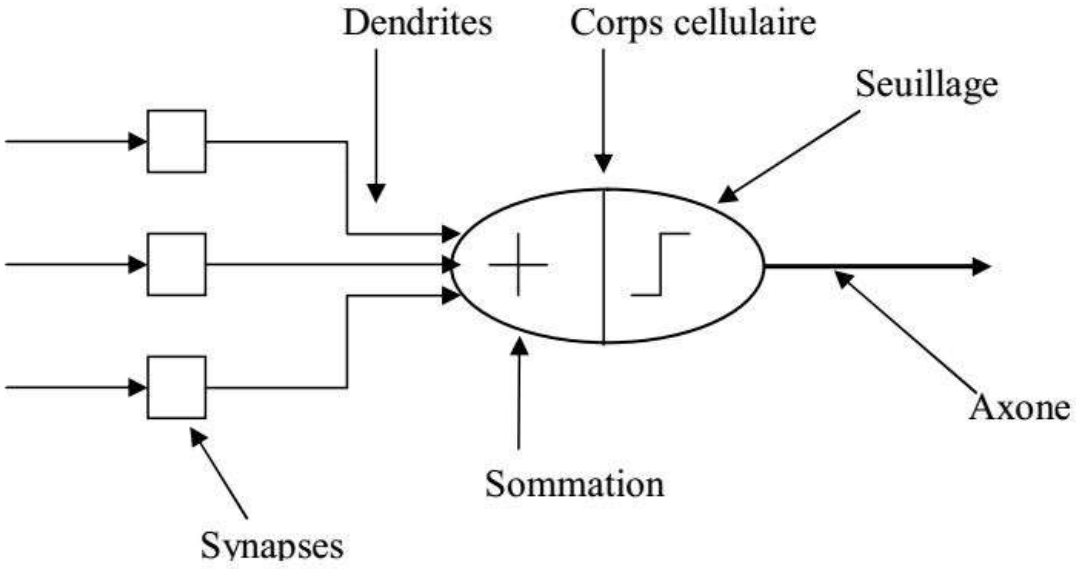


Figure III.3. *Le modèle d'un neurone biologique*

8. Neurone formel

McCulloch et pitts ont proposés une unité a seuil binaire comme un modèle de neuronal diagramme de McCulloch-Pitts est montre dans le Figure III-4:

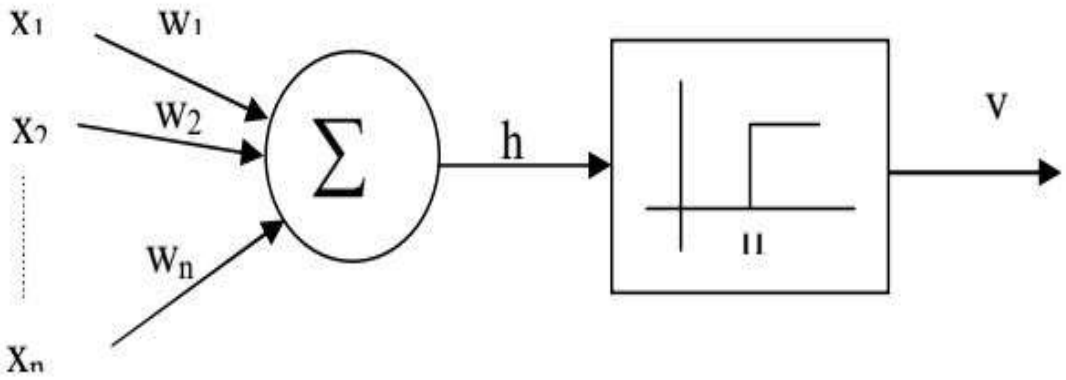


Figure III.4. *Le modèle de McCulloch-Pitts*

Ce modèle mathématique calcule la somme pondérée de ses n entrées, xj, j=1,2,…,n, Le nœud est caractérisé par un seuil [12], il génère une sortie de « 1 » si cette somme et supérieure au seuil μ , et autrement une sortie de « 0 » .Mathématiquement :

$$v = \begin{cases} 1 & si \quad \sum_{i=1}^{n} w_i x_i \geq \mu \\ 0 & si \quad \sum_{i=1}^{n} w_i x_i \prec \mu \end{cases}$$

Figure III.4. *Le modèle mathématique de McCulloch-Pitts*

où $\Phi(.)$ est la fonction d'activation, et w_j est le poids synaptiques associé a la j_{eme} sortie, parfois, on considère le seuil μ comme un autre poids $w_0 = -\mu$ et qui est connecté a un neurone avec une entrée constante, $x_0=1$. Les poids positifs correspondent à des synapses excitateurs, tandis que les poids négatifs correspondent aux synapses inhibitrices. Il y a une analogie évidente (Tableau II.1) avec les neurones biologiques :

Tableau III.1. L'analogie entre les neurones biologiques et les neurones artificiels

Neurones biologiques	Neurones artificiels
Synapses	Connections pondérées
Axons	Sorties
Dendrites	Entrées
Sommateur	Fonction d'activation

Le modèle de McCulloch et Pitts a été généralisé par de nombreuses façon, une généralisation évidente et celle d'utiliser une autre fonction d'activation que la fonction à seuil, ex., la fonction logistique, tangente hyperbolique, ou bien Gaussienne, qui sont montrées dans la Figure II.5.La fonction sigmoïde est de loin la plus utilisée dans les RNA, C'est une fonction qui est caractérisée par sa propriété asymptotique. La fonction sigmoïde standard est la fonction logistique et elle est définie par : $G(x)=1/(1+\exp(-\beta x))$

Où β est le paramètre de pente.

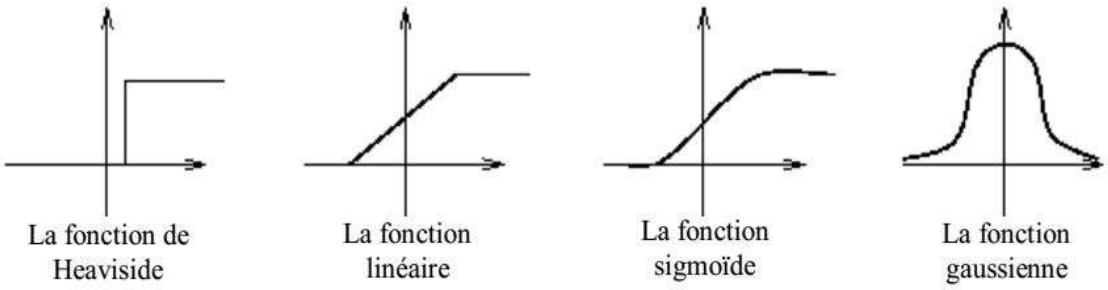


Figure III.5. *Quelques types de fonctions d'activations.*

9. Apprentissage :

La capacité à apprendre un trait fondamental de l'intelligence, d'un autre côté ce que veut dire l'apprentissage est parfois difficile à décrire, le processus d'apprentissage, dans le contexte des réseaux de neurones peut être vu comme un problème ajustement et d'adaptation de l'architecture et des poids des réseaux de neurone de façon que le RNA

exécute efficacement une tache spécifique. Au lieu de spécifier un ensemble de règles, les réseaux de neurones apprennent automatiquement à partir de la collection d'exemples, et c'est ce qui les rend très attractives.

Il y a trois types de paradigmes d'apprentissage :

- Supervisé
- Non supervisé
- Hybride

9.1. L'apprentissage supervisé

C'est actuellement le mode d'apprentissage le plus couramment utilisé. Son principe est élémentaire : on soumet au réseau à un grand nombre d'exemples pour lesquels l'entrée et la sortie associée sont connues et les poids sont modifiés de façon à corriger l'erreur commise par le réseau. Le plus répandu des algorithmes d'apprentissage supervisé est l'algorithme de rétro-propagation du gradient d'erreur qui, appliqué aux réseaux Multi Couches.

9.2. L'apprentissage nonsupervisé

Contrairement aux modes supervisés, seule une base d'entrées est fournie au réseau. Celui-ci doit déterminer lui-même ses sorties en fonction des similarités détectées entre les différentes entrées, c'est-à-dire en fonction d'une règle d'auto-organisation. Il explore la structure des données, ou encore la corrélation entre les modèles d'entrée et les organise à partir de cette corrélation en des catégories.

9.3. L'apprentissage hybride

Il combine l'apprentissage supervisé et non supervisé. Dans de nombreux problèmes on ne dispose pas de l'information nécessaire à la construction d'une base d'apprentissage complète. On ne dispose souvent que d'une information qualitative permettant l'évaluation de la réponse calculée, sans pour autant connaître la réponse la plus adaptée.

10. Quelques topologies des réseaux de neurones

Nous allons maintenant voir brièvement quelques types de réseaux de neurones, sur la Figure III.6 a une vue générale sur quelques topologies d' RNA

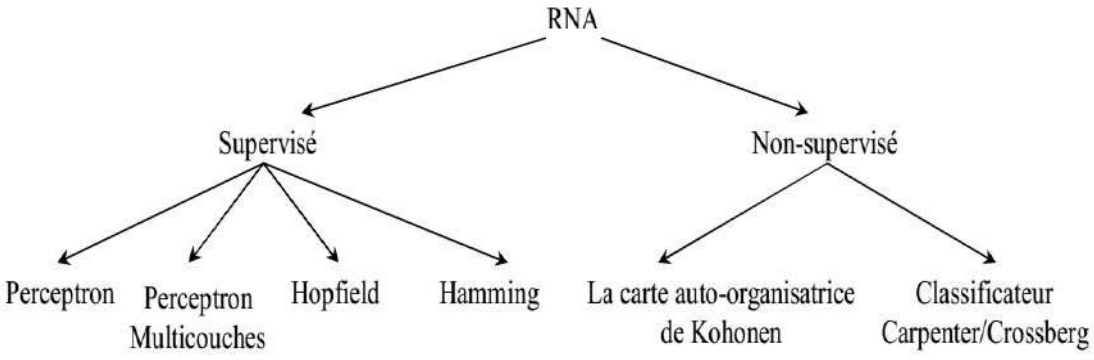


Figure III.6.*Quelques types de réseaux de neurones*

10.1 Le réseau de Hopfield

Le réseau de Hopfield ainsi que les deux autres réseaux montrés dans la Figure II.6 sont généralement employés avec des entrées binaires. Ces réseaux sont les plus appropriés lorsque des représentations binaires exactes sont possibles (ex. des images en noir et blanc où les éléments d'entrée sont des valeurs des pixels, ou avec un texte en code ASCII où chaque caractère peut être représenté par des bits). Sur la Figure II.7 on voit un réseau de Hopfield contenant N nœuds avec des fonctions seuil, des entrées binaires et des sorties pouvant prendre les valeurs +1 et −1. La sortie de chaque neurone est connectée à tous les autres nœuds par l'intermédiaire de facteurs de pondération (poids).

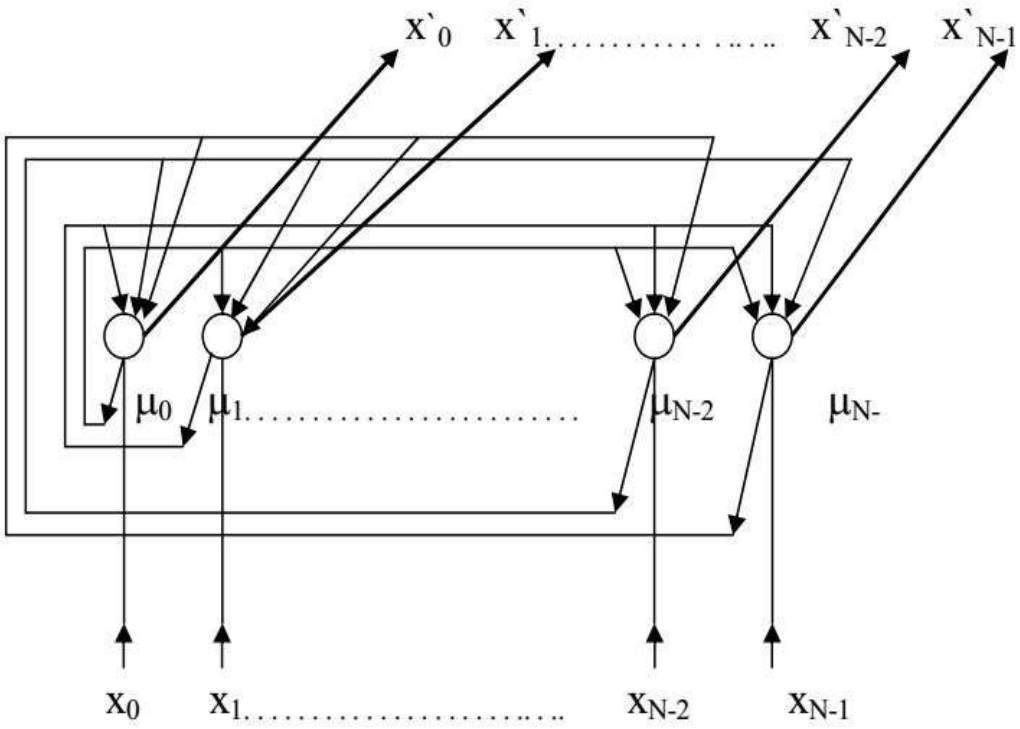


Figure III.7. *Réseau de Hopfield*

10.2. Le réseau de neurone de Hamming

Le réseau de Hopfield est souvent testé sur des problèmes où les entrées sont produites en choisissant un exemplaire en inversant les bits d'une manière aléatoire et indépendante avec une probabilité donnée. C'est un problème classique dans la théorie de communications qui se produit quand des signaux binaires à longueur constante sont envoyés par un canal binaire symétrique sans mémoire. Le classificateur optimal d'erreur minimale dans ce cas calcule la distance de Hamming de l'exemplaire pour chaque classe et choisit cette classe en fonction de la distance minimum de Hamming.

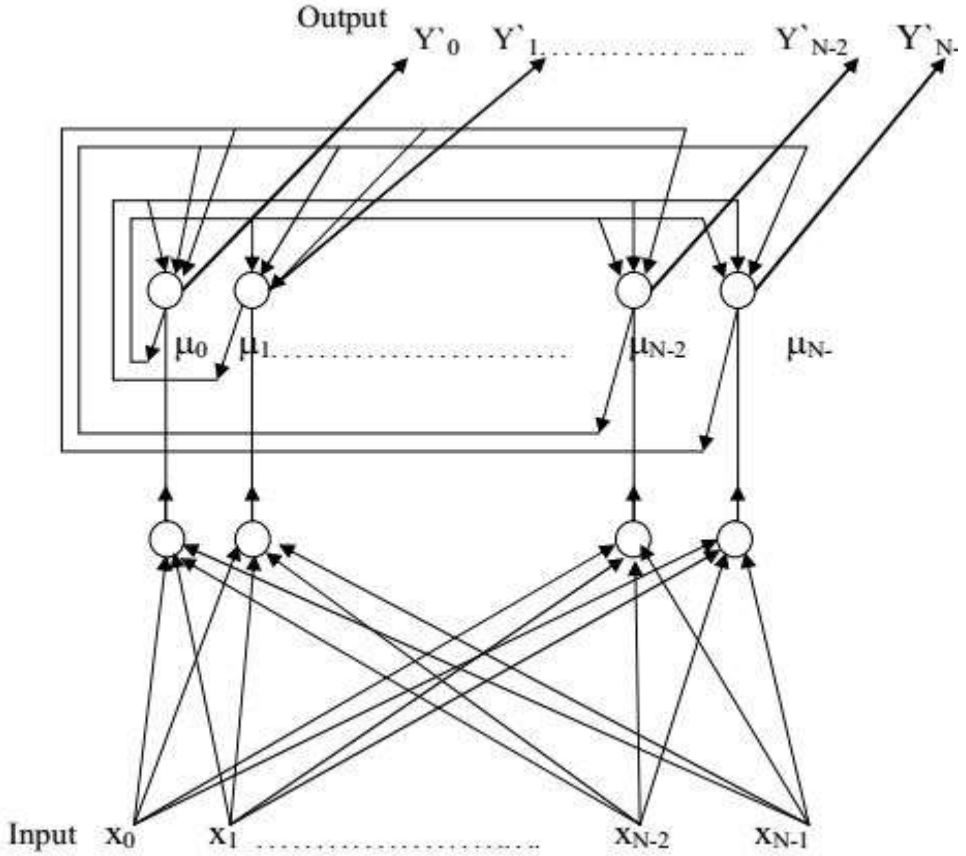


Figure III.8. *Réseau de Hamming*

10.3. Le classificateur de Carpenter et Crossberg

Carpenter et Grossberg, dans le développement de leur théorie de résonance adaptative ont conçu un réseau de neurones de classification ave un apprentissage nonsupervisé. Ce réseau met en application un algorithme de catégorisation qui est très semblable à l'algorithme séquentiel simple de chef. L'algorithme de chef choisit la première entrée comme exemplaire pour le premier faisceau. La prochaine entrée est comparée au premier exemplaire de faisceau. Elle suit "le chef" et est groupée avec le premier si la distance au premier est moins qu'un seuil. Autrement, c'est l'exemplaire pour un nouveau faisceau. Ce processus est répété pour toutes les entrées suivantes.

10.4. Perceptron à une seule couche

La perception à une seule couche est le premier de trois réseaux d'après la taxonomie de la Figure III.9 qui peuvent être employés avec des entrées binaires ou continues. Ce réseau simple a suscité beaucoup d'intérêt, quand initialement, a été développé en raison de sa capacité d'apprendre à identifier des modèles simples.

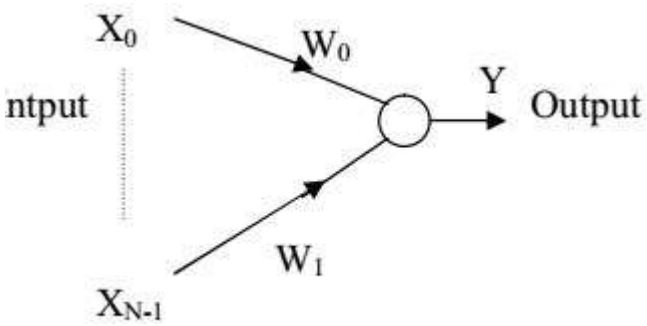


Figure III.9. *Perceptron à une seule couche*

Les poids de connexion ainsi que le seuil dans un perceptron à une seule couche peuvent être fixés ou adaptés en utilisant un certain nombre d'algorithmes. La procédure originale de convergence du perceptron à une seule couche pour l'ajustement des poids a été développée par Rosenblatt.

10.5. Le perceptron multicouches

Les perceptrons multicouches sont des réseaux non-récurrents avec une ou plusieurs couches de neurones entre les couches d'entrée et de sortie. Ces couches additionnelles contiennent des unités cachées ou des neurones qui ne sont pas directement reliés aux neurones des couches d'entrée et de sortie. Un perceptron avec trois couches dont deux couches cachées est montré dans la Figure III.10. Ils sont de plus en plus employés avec le développement de nouveaux algorithmes d'apprentissage.

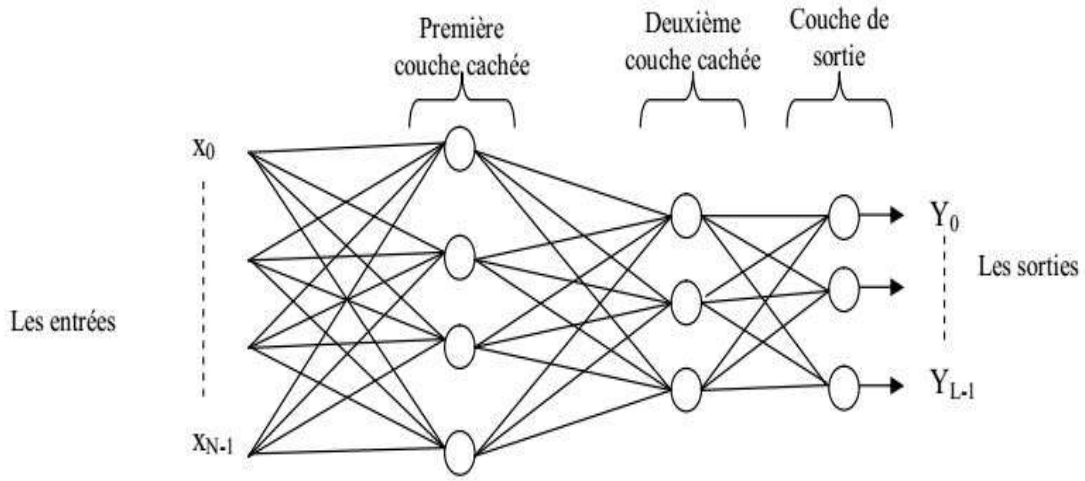
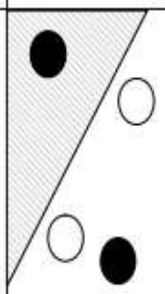
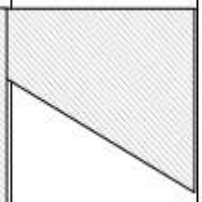
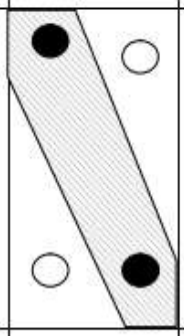
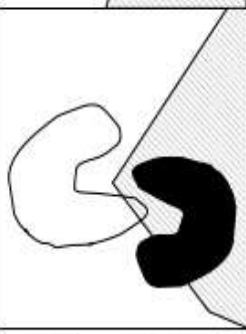
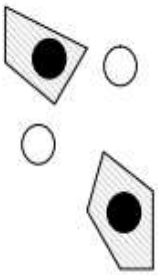
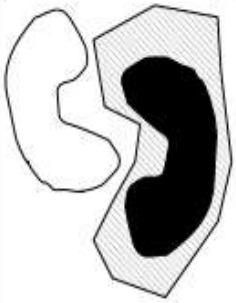
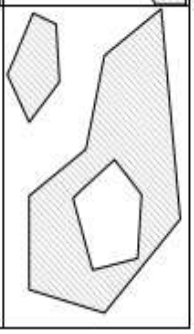


Figure III.10. *Perceptron Multi-Couche*

Les capacités des perceptrons multicouches proviennent des non-linéarités utilisées dans les nœuds. Les capacités des perceptrons avec une, deux, et trois couches qui utilisent des non-linéarités illustrées dans le Tableau III.2. La deuxième colonne dans cette Figure indique les types de régions de décision qui peuvent être formées par différents réseaux. Les deux colonnes suivantes montrent des exemples des régions de décision qui peuvent être formées pour le problème du OU exclusif et un problème avec des régions engrenées. La colonne en extrême droite donne des exemples des régions de décision les plus générales qui peuvent être formées.

Cette analyse fournit un aperçu sur le problème de choit du nombre de nœuds à employer dans un perceptron à deux couches.

Tableau III.2. *Interprétation géométrique du rôle des couches cachées*

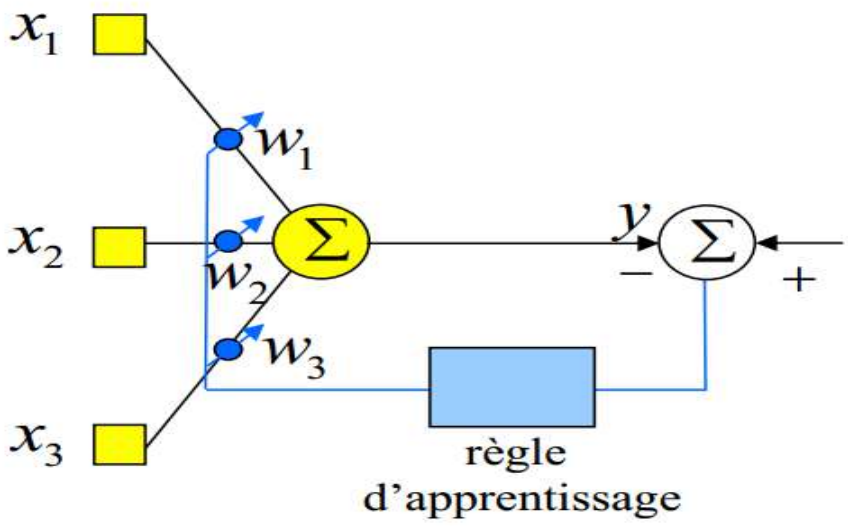
Architecture	Le problème de l' XOR	Type de la région de décision	Classes avec des régions chevauchées	Les formes de régions les plus générales
Simple couche		Un demi plan borné par un hyperplan		
Deux couche		Des régions convexes fermées ou ouvertes		
Trois couches		Arbitraires		

10.6. L'ADALINE

ADALINE est un réseau neuronal simple-couche, avec plusieurs nœuds, chacun acceptant plusieurs entrées et générant une sortiePour traiter plusieurs sorties, on utilise plusieurs Adaline en parallèle, on parle alors de MAdaline (Muti-Adaline).

L'apprentissage d'un Adaline s'effectue en ligne et est basé sur la minimisation d'une erreur, l'erreur quadratique moyenne.

Lorsque le nombre d'entrées (exemples) tend vers l'infini, l'Adaline tend vers la solution de Gauss ou le filtre de Wiener. Les poids de l'Adaline ont alors convergé vers les coefficients optimaux (au sens des moindres carrés) et sont équivalents aux coefficients du filtre de Wiener [14].



$$\mathbf{w}_{k+1} = \mathbf{w}_k + \Delta\mathbf{w}_k$$

$$\varepsilon_k = d_k - y_k = d_k - \mathbf{w}_k^T \mathbf{x}_k$$

On peut utiliser 2 règles :

μ LMS

$$\Delta\mathbf{w}_k = \mu\varepsilon_k \mathbf{x}_k \alpha$$

α LMS

$$\Delta\mathbf{w}_k = \alpha \frac{\varepsilon_k \mathbf{x}_k}{\lambda + \|\mathbf{x}_k\|^2}$$

Figure III.11.*regle d'apprentissage d'ADALINE*

10.7. Carte auto-organisatrice de Kohonen

Un important principe d'organisation des voies sensorielles dans le cerveau est que le placement des neurones soit ordonné et qu'il reflète souvent certaines caractéristiques physiques du stimulus externe ressenti. Par exemple, à chaque niveau de la voie auditive, des cellules nerveuses et des fibres sont arrangées anatomiquement en fonction de la

fréquence qui obtient la plus grande réponse dans chaque neurone. Cette organisation tonotopique dans la voie auditive s'étend jusqu'au cortex auditif. Bien qu'une grande partie de l'organisation de bas niveau soit génétiquement prédéterminée, il est probable qu'une partie de l'organisation à des niveaux plus élevés soit créée pendant l'apprentissage par des algorithmes qui favorisent l'auto-organisation. Kohonen présente un algorithme semblable, qui produit ce qu'il appelle les cartes à organisation automatique semblables à celles qui se produisent dans le cerveau.

11. Résultat de nos différents tests des simulation

Pourquoi L'ADALINE ?

- Capacité de représenter n'importe quelle fonction linéaire simple ou peu complexe
- Faculté d'apprentissage à partir d'exemples représentatifs
- L'apprentissage (ou construction du modèle) peu s'effectuer en ligne (in situ)
- Résistance au bruit ou au manque de fiabilité des données
- Simple à manier, beaucoup moins de travail personnel à fournir que les autres méthodes
- Comportement moins mauvais en cas de faible quantité de données
- Le choix des valeurs initiales des poids du réseau et le réglage du pas d'apprentissage, qui jouent un rôle important dans erreur et la sortie

Pour illustrer le fonctionnement de l'Adaline et montrer l'influence des différents paramètres intrinsèques sur ses performances, nous avons décidé d'utiliser les fonctions « simples ».

TESTE N°1

Dans un premier teste nous allons montrer l'influence du coefficient d'apprentissage et de l'initialisation des poids W du réseau Adaline.

La fonction utilisée est décrite comme suit :

$$y = \sin(x) \quad \text{quant} \quad x \in [-6, 6].$$

Les paramètres utilisés sont :

- Coefficient d'apprentissage alpha (α=1) ;
- Initialisation des poids w = 0 ;

Pour cela, nous obtenons les graphiques suivants :

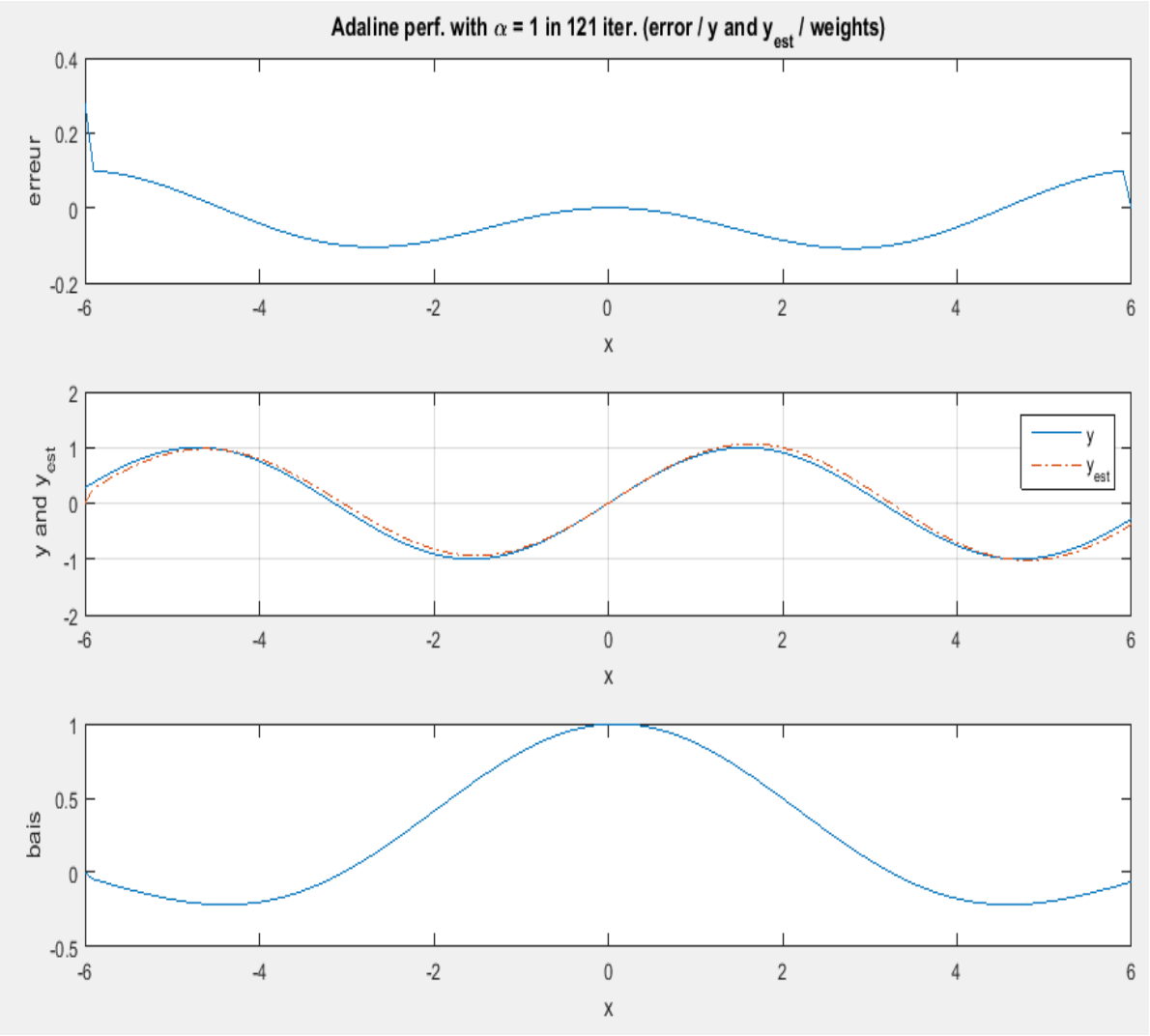


Figure III.12. *Teste de la fonction sinus avec ADALINE*

En modifiant les paramètres du réseau Adaline comme suit :

- Coefficient d'apprentissage alpha α=0.1

- Initialisation des poids w = 0 ,

nous obtenons les courbes suivantes :

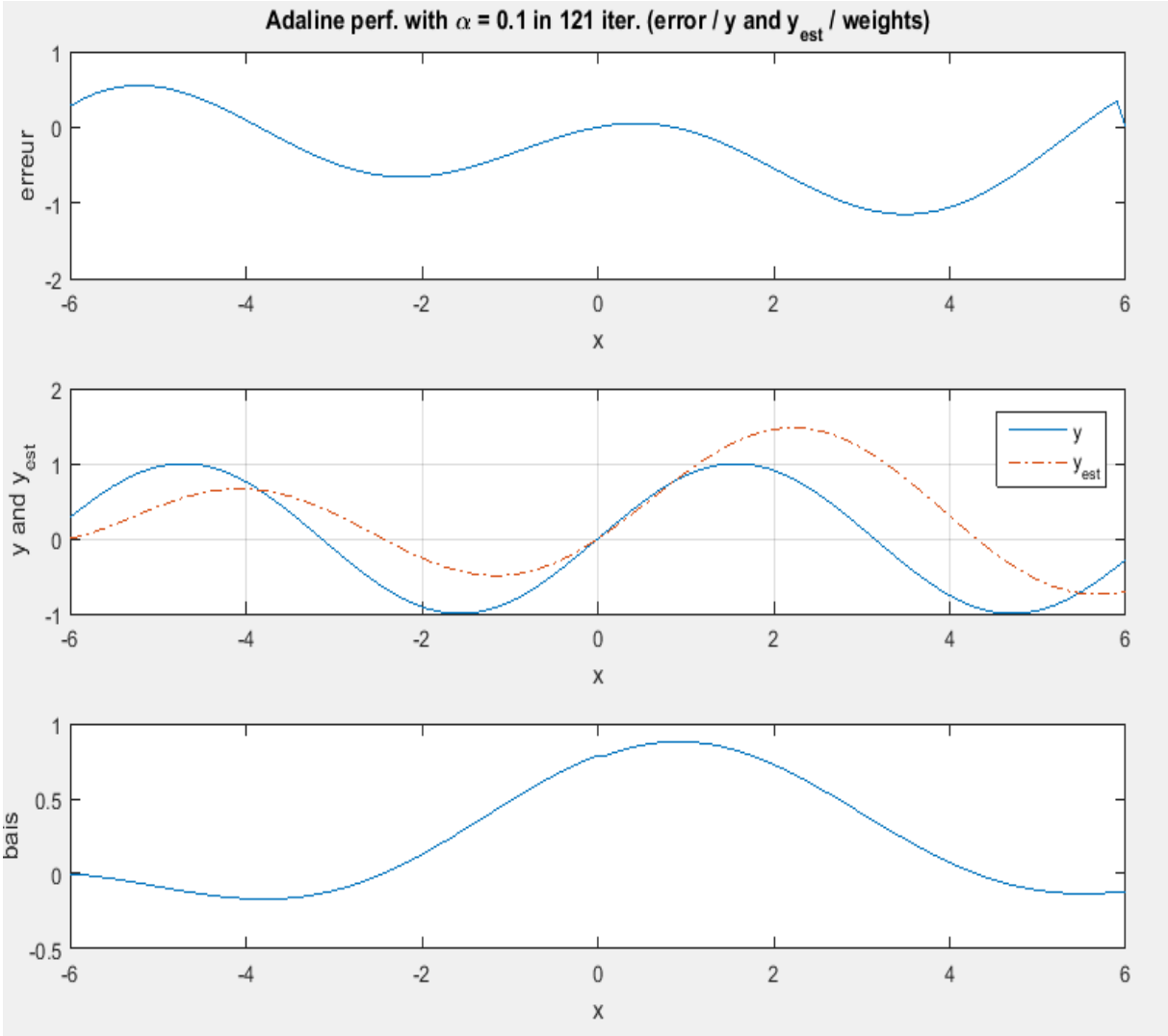


Figure III.13. *Teste de fonction sinus avec ADALINE*

Avec ce teste, nous constatons que la sortie du réseau (y_{est}) n'arrive pas à suivre la sortie réelle de la fonction y. En effet, même si le choix du coefficient d'apprentissage est un peu exagéré (de $\alpha = 1$ à $\alpha = 0.1$), mais nous pouvons dire que ce paramètre influe sur la modification des poids qui rentre dans le calcul de la sortie du réseau à chaque itération. Plus ce coefficient est faible, plus on s'approche de la zone dites de « sous apprentissage ». Dans ce cas-ci, la sortie du réseau y est toujours en retard par rapport à la sortie réelle y de la fonction.

L'autre paramètre qui peut influencer sur l'apprentissage de l'Adaline est l'initialisation de ses poids W. Pour illustrer leur influence, nous avons adopté les paramètres suivants :

- Coefficient d'apprentissage alpha α=1
- Initialisation des poids w = -1,

grâce auxquels nous avons obtenu ces courbes :

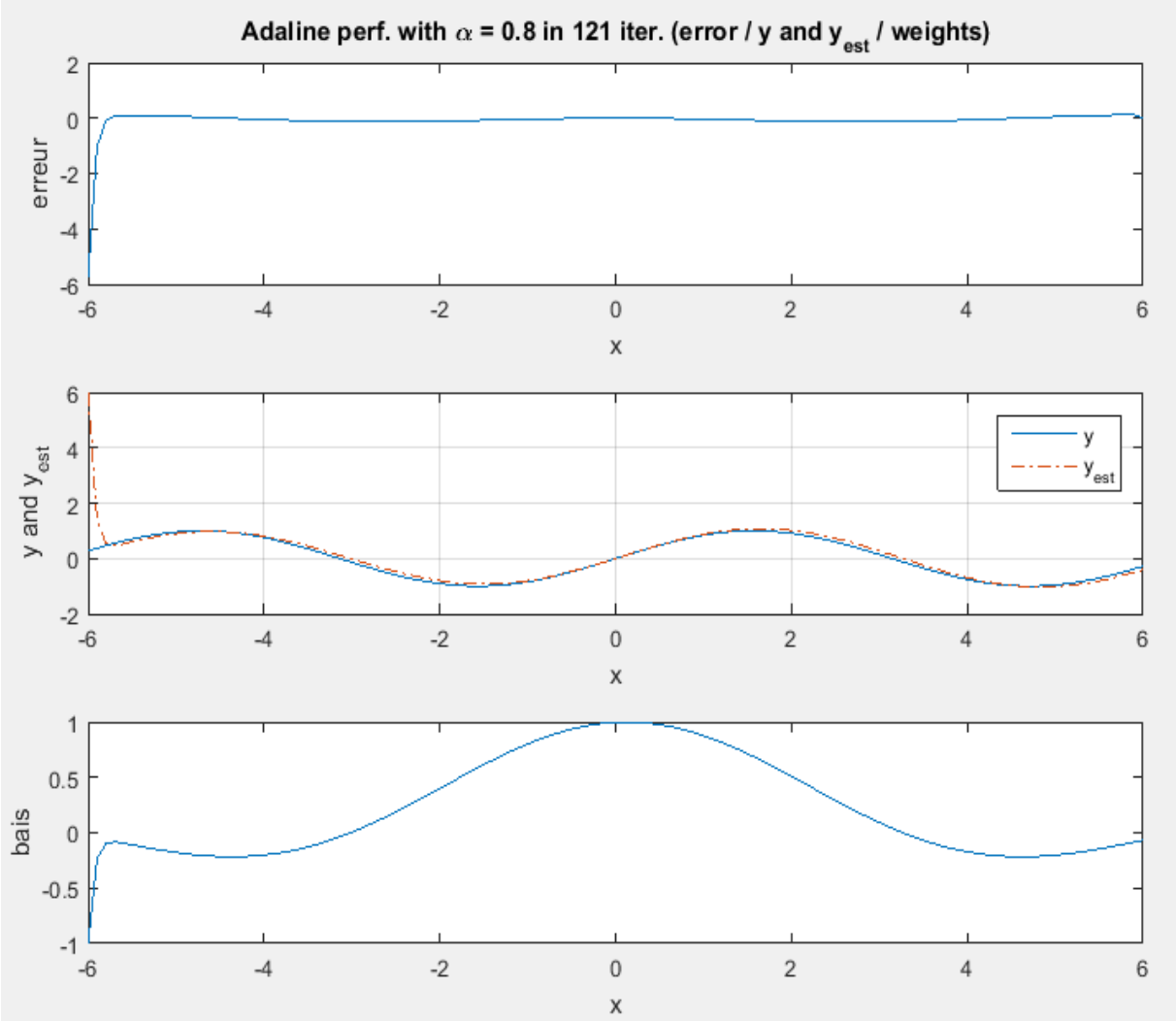


Figure III.14. *Teste de fonction sinus avec ADALINE*

On constate que l'erreur dans cette figure est au bout de quelques itérations moins importante que celles obtenues précédemment. Cela est dû à l'application d'une valeur adéquate lors de l'initialisation des poids W de l'Adaline. Ces valeurs dépendent des processus commandés et sont souvent très difficile à connaître à l'avance. C'est pour cela

qu'il faut réaliser plusieurs tests pour adopter définitivement les paramètres d'un réseau de neurones.

TESTE N°2

Dans ce deuxième teste, nous allons montrer l'influence du changement de signe brutale de la fonction dans l'évolution de l'erreur d'apprentissage du réseau Adaline.

La fonction utilisée (un sinus cardinal) est décrite comme suit :

$$y = \text{sinc}(x) \quad \text{quant} \quad x \in [-5, 5].$$

Les paramètres utilisés sont :

- Coefficient d'apprentissage alpha (α=1) ;
- Initialisation des poids w = 0 ;

et nous obtenons les figures suivantes :

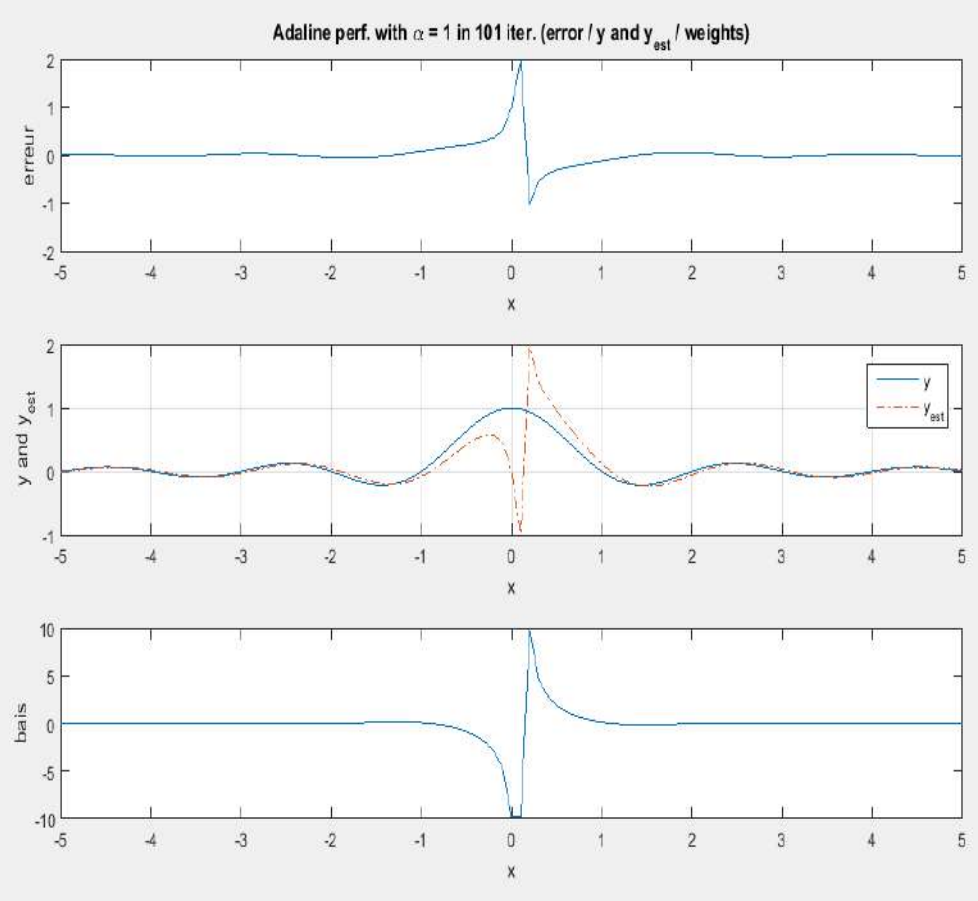


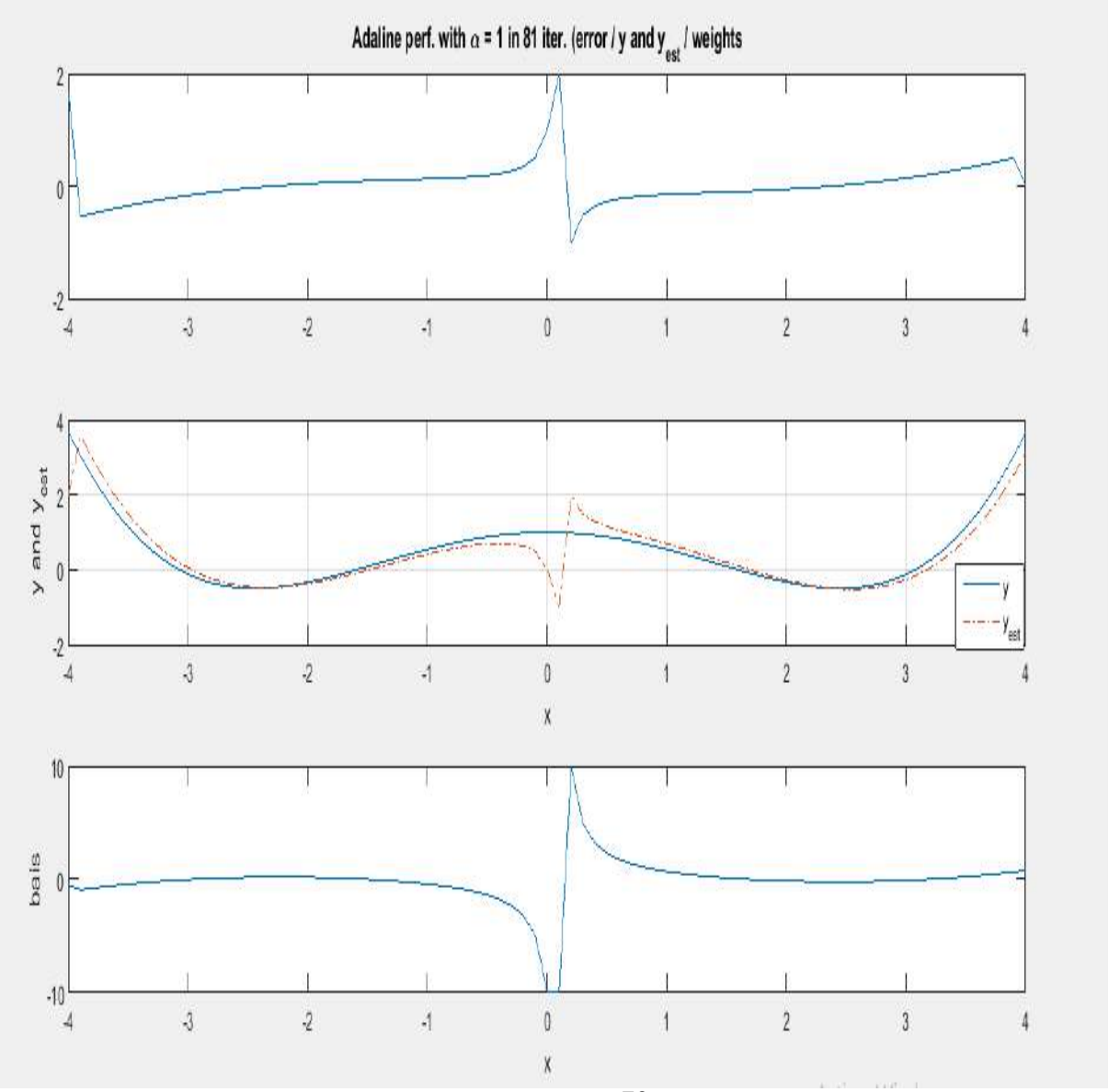
Figure III.15. *Teste de fonction sinus cardinal avec ADALINE*

De ces courbes, nous constatons un pique d'erreur important au moment où les entrées X du réseau Adaline changent de signe. Cela s'explique par le fait que le réseau Adaline apprend localement la pente de la fonction sinus cardinal et au moment où celle-ci change brutalement de signe, le réseau ne suit pas et il lui faudra même plusieurs itérations avant de reprendre et de fournir des sorties cohérentes. Ce phénomène est inhérent à la conception du réseau Adaline et non à la fonction apprise. D'ailleurs, nous vous présentons le comportement du réseau Adaline (avec les mêmes paramètres que précédemment) lors de l'apprentissage d'une autre fonction.

Cette fonction est décrite comme suit :

$$y = x^4 + 2.x^2 + 1 \text{ quant } \quad x \in [-4, 4].$$

et les résultats obtenus sont illustrer par les figures suivantes :



12. Conclusion

Ce chapitre a été pour nous l'occasion d'aborder les différents types de réseaux de neurones. Cette étude nous a permis de différencier ces réseaux de par leurs structures et de leurs algorithmes d'apprentissage.

Enfin, nous avons programmé et évaluer les performances du réseau de neurones ADALINE par le biais de fonctions déterministes. Ce prélude a sont implémentation sur notre plate-forme robotique nous a permis de comprendre l'influences des paramètres de l'ADALINE sur la qualité de l'apprentissage. Cette maitrise nous sera très utile lors de son utilisation dans la commande de notre robot mobile.

Chapitre IV :

Validation expérimentale de notre contrôleur neuronal

1. Introduction

Pour mettre en œuvre une commande neuronale de notre robot mobile, nous avons besoin d'une stratégie de commande. Pour cela, nous allons utiliser un asservissement visuel afin de contrôler la position d'un robot mobile. Cet objectif doit être atteint en utilisant uniquement les données visuelles issues d'une caméra Pixy.

Notre projet vise a programmé un algorithme de contrôle d'un robot mobile capable de réaliser un suivi de cible à base d'un contrôle neuronal. Ce dernier est conçu autour du réseau de neurones Adaline décrit dans le chapitre précèdent. Ce contrôleur est une évolution majeure de notre plate-forme robot mobile qui sera dotée d'une intelligence artificielle (Machine Learning).

2. Robot mobile Arduino

2.1. Description

Notre robot mobile Arduino est de type tricycle, représenté à la figure de la figure IV.1. Il est constitué de deux roues fixes de même axe et d'une roue centrée orientable placée sur l'axe longitudinal du robot. Le mouvement est conféré au robot par deux actions : la vitesse longitudinale et l'orientation de la roue orientable. De ce point de vue, il est donc très proche d'une voiture. [12]

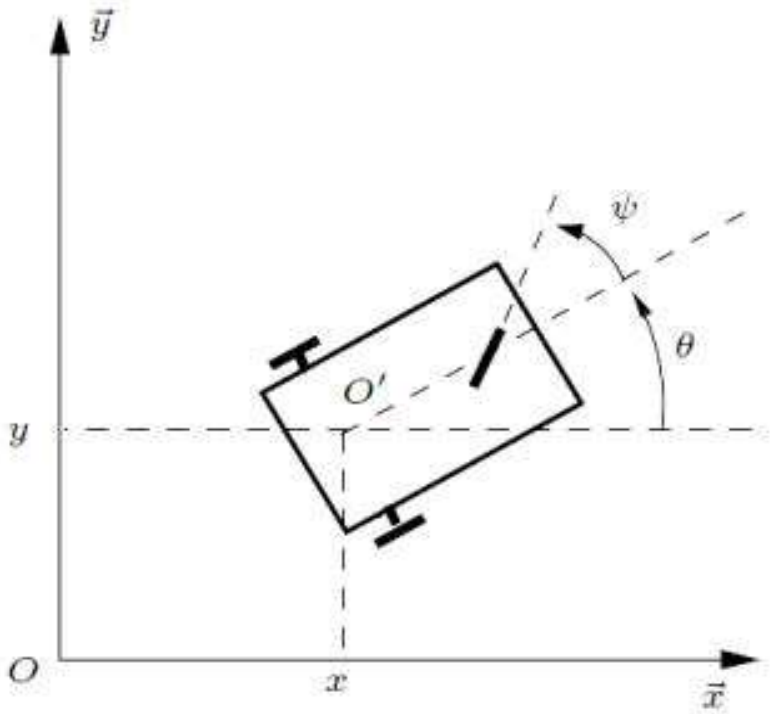


Figure IV.1. *Robot mobile de type tricycle.*

2.2. Modélisation

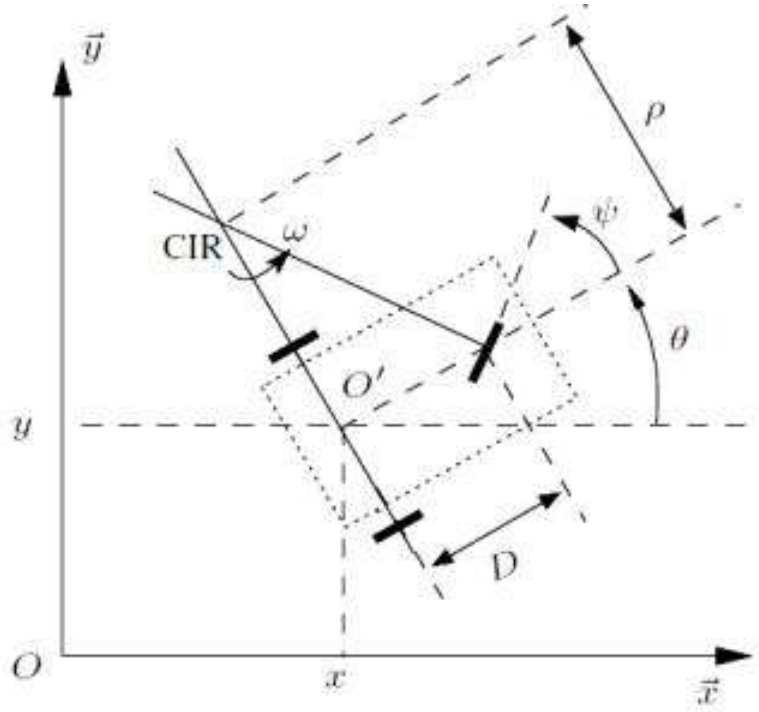


Figure IV.2. *Un robot mobile de type tricycle et son CIR***.**

Le CIR du robot mobile de type tricycle se situe à la rencontre des axes des roues fixes et de la roue orientable, comme cela est représenté à la figure IV.2. On peut déterminer ρ de manière géométrique à partir de l'angle d'orientation de la roue avant et ω à partir de la vitesse linéaire ϑ du véhicule (vitesse en O') et de ρ :

$$\rho = \frac{D}{\tan \psi}$$

$$\omega = \frac{\vartheta}{D} \tan \psi$$

Ce type de robot peut se diriger en ligne droite pour ψ=0 et théoriquement tourner autour du point O' (on pourrait dire sur place) pour $\psi = \frac{\pi}{2}$. Néanmoins, le rayon de braquage de la roue orientable, généralement limité, impose le plus souvent des valeurs de ψ telles que $-\frac{\pi}{2} < \psi < \frac{\pi}{2}$, interdisant cette rotation du robot sur lui-même.

L'écriture des contraintes sur chacune des roues et un raisonnement similaire à celui suivi dans le cas de l'unicycle permettent de déterminer les modèles cinématiques des robots de type tricycle. Toutefois, par un simple raisonnement géométrique, on établit les équations :

$$\dot{x} = v\cos\theta$$
$$\dot{y} = v\sin\theta$$
$$\dot{\theta} = \frac{v}{D}\tan\psi$$
$$\dot{\psi} = \eta,$$

Comme on l'a vu précédemment, l'existence d'un CIR unique n'impose que les axes des roues du robot soient concourants. [12]

3. Le système de vision Pixy

Comme nous l'avons mentionné dans le chapitre II sur les caractéristiques de la Cam Pixy, Nous présentons maintenant les étapes nécessaires pour détecter un objet à Pixy.

3.1. La détection d'objet objet avec Pixy

Enseigner à Pixy un objet est trés facile, mais abordons d'abord les objets qui fonctionneront bien avec Pixy. Pixy utilise un algorithme de filtrage des couleurs basé sur la teinte pour détecter les objets. Puisque Pixy utilise la teinte (couleur), l'objet doit avoir une teinte distincte. Voici quelques objets qui sont bons parce qu'ils ont des teintes bonnes et distinctes.



Figure.IV.3. *des objets avec des teintes bonnes et distinctes.*

Voici de mauvais objets car il n'y a pas de teinte (noir, blanc ou gris) ou la teinte n'est pas distincte.



Figure IV.4. *Des objets avec teintes non distinctes.*

Gardant ces remarques à l'esprit, choisissez un objet pour que Pixy le détecte. Lorsque vous allumez Pixy, il passera par une série de flashs LED. Attendez que la LED s'éteigne avant de présenter à Pixy un objet. Maintenez le bouton en haut de Pixy enfoncé. Après environ 1 seconde, la LED s'allume - premièrement blanche, puis rouge, puis d'autres couleurs – mais quand elle devient rouge, relâchez le bouton.

Lorsque vous relâchez le bouton, Pixy entrera dans ce que l'on appelle le mode "tube léger", où la couleur LED est la couleur de l'objet sur lequel Pixy a été "verrouillé". Pixy va se verrouiller sur les objets au centre de son cadre vidéo, alors maintenez l'objet directement devant Pixy, entre 6 et 20 cm de l'objectif. Pixy utilise un algorithme de croissance de région pour essayer de déterminer quels pixels font partie de votre objet et quels pixels font partie de l'arrière-plan. À l'aide de ces pixels, Pixy tentera de créer un modèle statistique de votre objet afin qu'il puisse le détecter de manière fiable dans différentes conditions d'éclairage. Utilisez la couleur LED comme retour pour déterminer si Pixy a un bon verrou sur l'objet, et utilisez les instructions suivantes pour juger :

1. Lorsque Pixy a verrouillé sur votre objet, la couleur LED doit correspondre à la couleur de votre objet.

2. Plus la LED est lumineuse, meilleure est la serrure. Déplacez l'objet un peu et voyez si vous pouvez optimiser la luminosité de la LED.

3. Regardez la grille de la région dans la fenêtre PixyMon et voyez comment la taille de la grille correspond à la couleur et à la luminosité de la LED. Une grille qui couvre plus de votre objet est supérieure à une grille qui ne couvre qu'une partie de votre objet et aboutit à une LED plus lumineuse. L'image ci-dessous montre une grille qui couvre la majeure partie de l'objet (bon). [19]

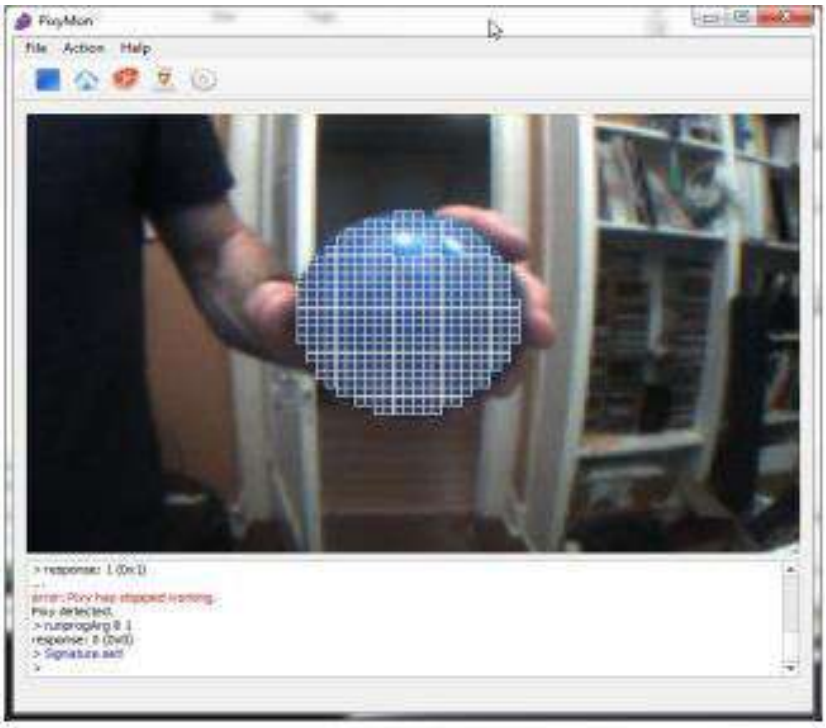


Figure IV.5. *La majeure partie de l'objet (bon) couvre par une grille.*

Lorsque vous êtes satisfait, appuyez et relâchez le bouton sur Pixy, comme si vous aviez cliqué sur votre souris. La LED clignotera quelques fois pour indiquer que Pixy a maintenant appris votre objet. Il va maintenant commencer à suivre cet objet.

3.2. Objets à signatures multiples

Pixy peut apprendre jusqu'à sept signatures de couleurs. Nous pouvons apprendre à Pixy la 1ère signature couleur en relâchant le bouton lorsque la LED devient rouge. Si nous continuons d'appuyer sur le bouton, la LED devient orange, jaune, etc., indiquant les signatures de couleur restantes. Voici les signatures dans l'ordre :

1. rouge 2. Orange 3. Jaune 4. Vert 5. Cyan (bleu clair) 6. Bleu 7. Violet

Le numéro de signature couleur est déterminé lorsque vous relâchez le bouton. Relâchez le bouton lorsque la LED est jaune et que vous êtes en train d'enseigner la signature 3. Relâchez le bouton lorsque la LED est bleue et que vous enseignez la signature 6. Ces couleurs ne sont pas liées à la teinte de l'objet. Les couleurs sont utilisées uniquement pour indiquer le numéro de signature. Ainsi, par exemple, la signature 1 peut être un objet jaune, même si la signature 1 est indiquée par une LED rouge, et la signature 2 pourrait être un objet rose même si la signature 2 est indiquée par une LED orange.

Après avoir enseigné la signature de Pixy, elle enregistre la signature en flash. Par conséquent, lorsque vous mettez le cycle de votre Pixy, il se souviendra des signatures que vous lui avez enseignées et continuerez à suivre les objets qui correspondent à ces signatures [19].

3.3. Balance des blancs

Certains types d'éclairage (comme les incandescents) ont une teinte rougeâtre et d'autres (comme les fluorescents) ont une teinte bleutée. L'éclairage peut affecter vos signatures de couleur. Par exemple, si vous apprenez un objet sous un éclairage incandescent et que vous vous déplacez dans une pièce avec un éclairage fluorescent, les signatures des couleurs ne fonctionneront probablement plus. Vous pouvez soit ré-enseigner toutes les signatures, soit ajuster la balance des blancs. [19]

4. L'utilisation de l'application PixyMon

Vous pouvez également enseigner à Pixy un objet à travers PixyMon. Cela peut être utile si l'objet que vous souhaitez enseigner est petit ou si vous souhaitez plus de contrôle sur les pixels utilisés pour l'enseignement. Commencez par brancher le câble USB entre Pixy et votre ordinateur et exécutez PixyMon.

Maintenez maintenant l'objet que vous souhaitez enseigner devant Pixy et sélectionnez **Action➜Set signature 1** dans le menu déroulant.

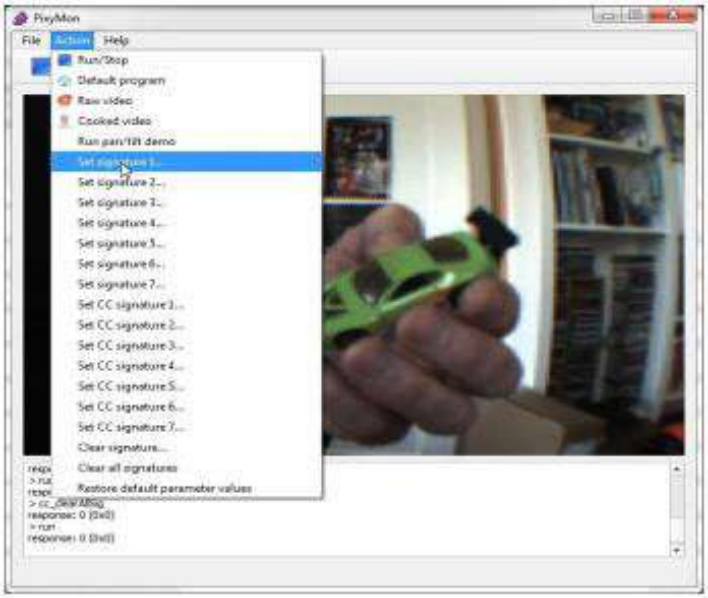


Figure.IV.5. *maintenez la signature de l'objet que vous souhaitez enseigner.*

À l'aide de la souris, cliquez et faites glisser pour sélectionner la région que vous souhaitez que Pixy utilise pour apprendre l'objet.

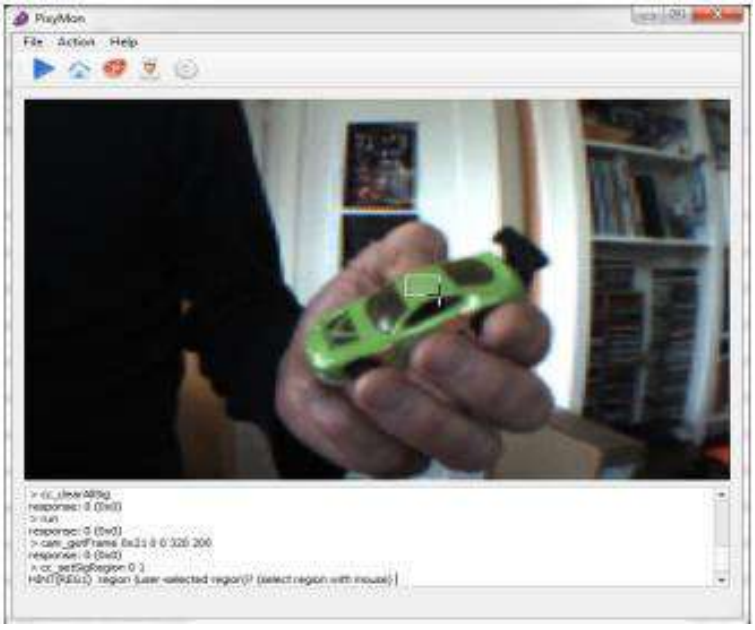


Figure.IV.6. *sélectionner la région que vous souhaitez que Pixy utilise pour apprendre l'objet.*

C'est tout ! Après avoir sélectionné la région, Pixy "apprendra" l'objet et entrera automatiquement dans le mode vidéo "cuit" afin de vérifier à quel point votre signature couleur fonctionne.

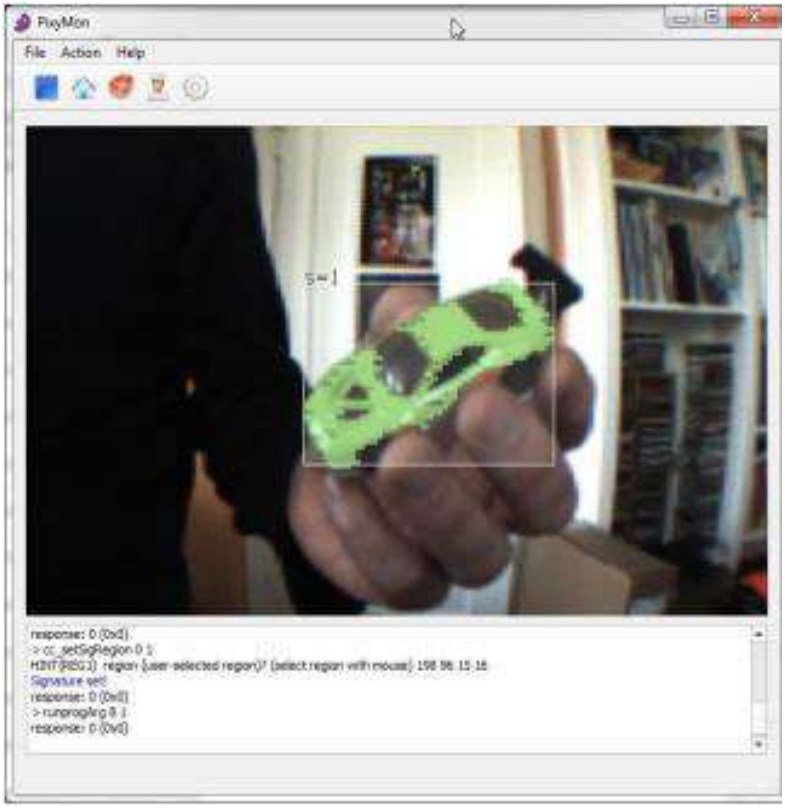


Figure.IV.7. *Pixy "apprendra" l'objet*

4.1. Signature tuning

Parfois, les signatures de couleurs que vous enseignez à Pixy doivent être "tordues" - c'est-à-dire que vous obtenez des faux positifs (Pixy détecte des objets qui ne sont pas les objets que vous vouliez) ou des faux négatifs (Pixy ne détecte pas l'objet Vous l'avez appris, ou il détecte l'objet par intermittence). Ceci peut etre resolu grâce à la modification d'un certains nombres de paramètres. Il suffit de faire apparaître la boîte de dialogue, sélectionnez File➜Configure), puis sélectionnez le **Signature Tuning** sous **Pixy Parameters**. [19]

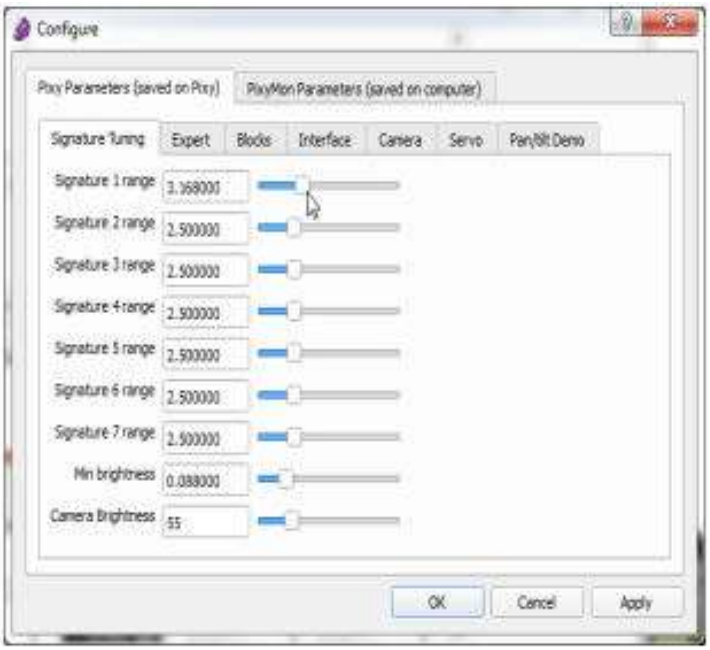


Figure.IV.8. *Pixy Paramètres.*

Utilisez le curseur pour **Signature 1 range** pour ajuster l'inclusivité de la signature 1 (en supposant que c'est la signature 1 que vous souhaitez ajuster). Faites-le glisser vers la gauche si vous voulez être moins inclusif (c'est-à-dire si vous voyez des faux positifs, comme l'image ci-dessous):

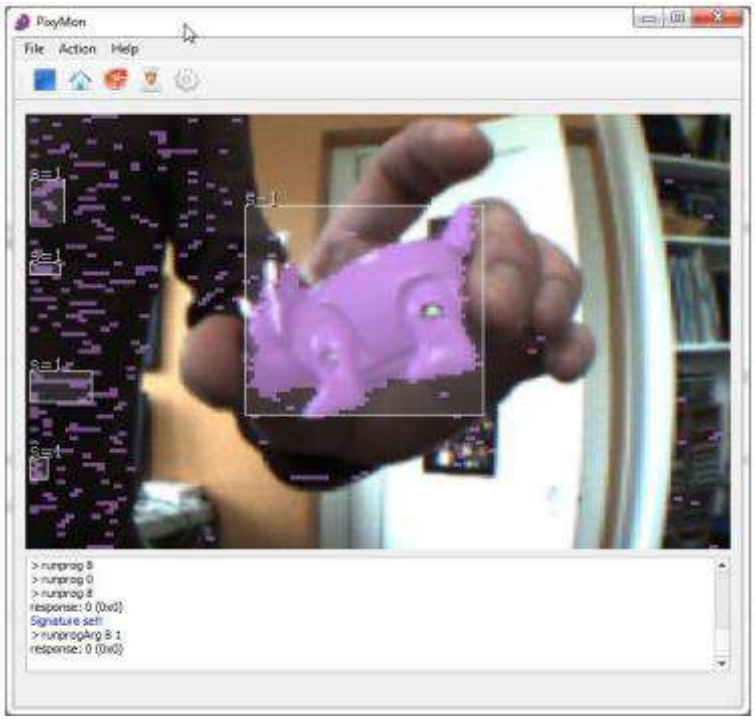


Figure.IV.9. *ajuster l'inclusivité de la signature 1.*

Ou glissez-le vers la droite si vous voulez être plus inclus (c'est-à-dire que vous voyez des faux négatifs) ou que la détection est intermittente ou clairsemée (comme dans l'image ci-dessous)

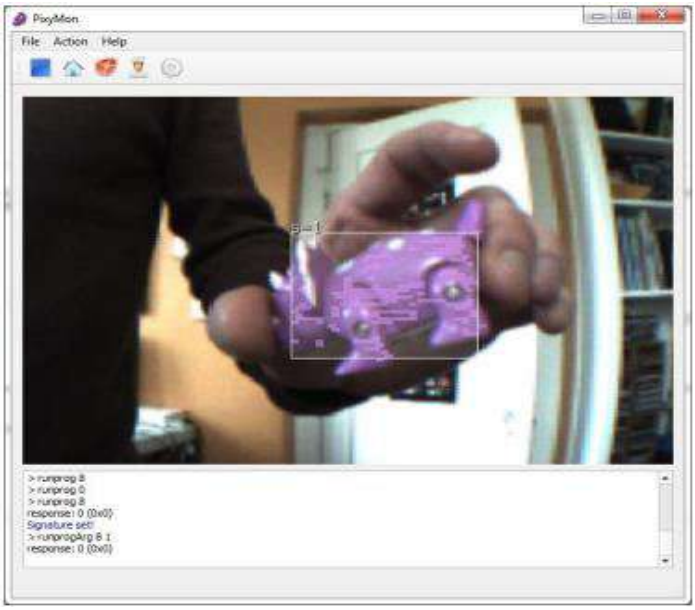


Figure.IV.10. *la détection intermittente ou clairsemé.*

Choisissez une valeur de curseur qui procure une bonne détection forte, comme l'image ci-dessous :

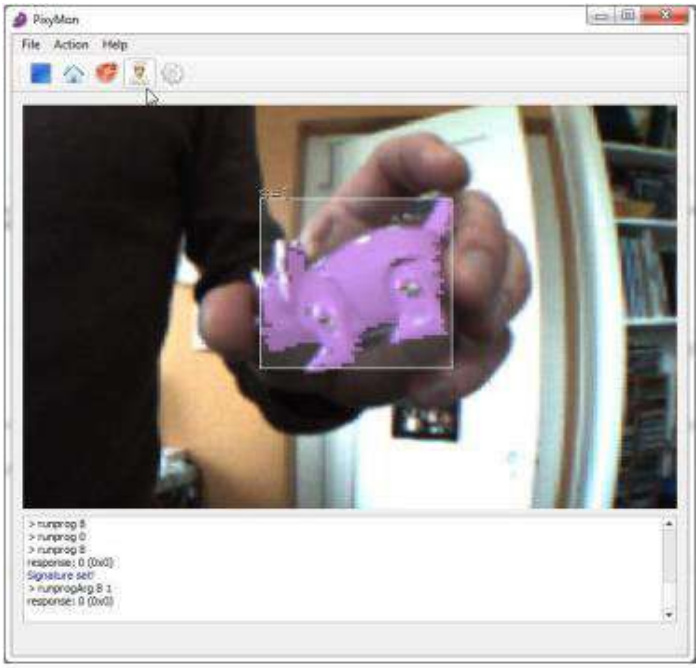


Figure.IV.11. *valeur de curseur qui procure une bonne détection.*

Vous pouvez ajuster toutes les signatures de sept couleurs de cette façon pour maximiser la précision de détection pour toutes les signatures. Assurez-vous d'appuyer sur **Apply** ou sur **OK** pour enregistrer les plages de curseurs ! Les valeurs ajustées ne seront pas sauvegardées si vous appuyez sur **Cancel** ou désistez la boîte de dialogue. [19]

5. La carte Raspberry Pi: le cerveau

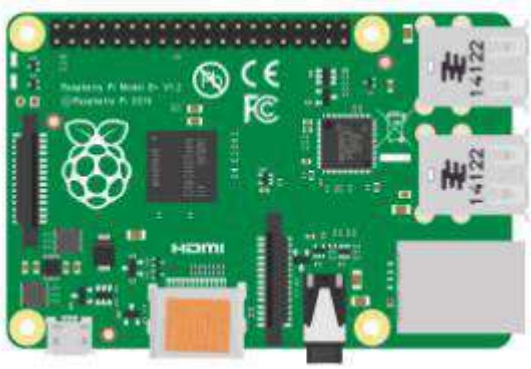


Figure.IV.12. *la carte Raspbery Pi utilisé.*

Cette carte, de taille d'une carte de crédit, est considérée comme étant le « cerveau » de notre robot parce qu'elle est en charge de l'acquisition, du calcul des lois de commande. Ce mini PC, doté d'une mémoire importante et d'un Processeur puissant fonctionne sous le système d'exploitation LINUX.

Le modèle de Raspberry Pi 3, nous avons choisi ce modèle grâce à ces caractéristiques, (voir Chapitre II (La carte Raspberry Pi)) Grace au système Linux, nous avons pu programmé l'ensemble de nos algorithmes en langage Python et exploité les fonctions précompilées de la Cam Pixy. Dans la suite de ce chapitre, nous décrirons brièvement les techniques que nous avons mises en œuvre pour réaliser notre asservissement visuel à savoir : le traitement d'image basé sur l'exploitation des ressources fournie par Pixy, les organigrammes écrits en langage Python. Ces programmes contiennent aussi une procédure pour transmettre, via le port USB, les consignes au module Arduino / Shield en charge de les convertir en taisions nécessaires aux mouvements des moteurs du robot mobile.

6. Schéma bloc du projet

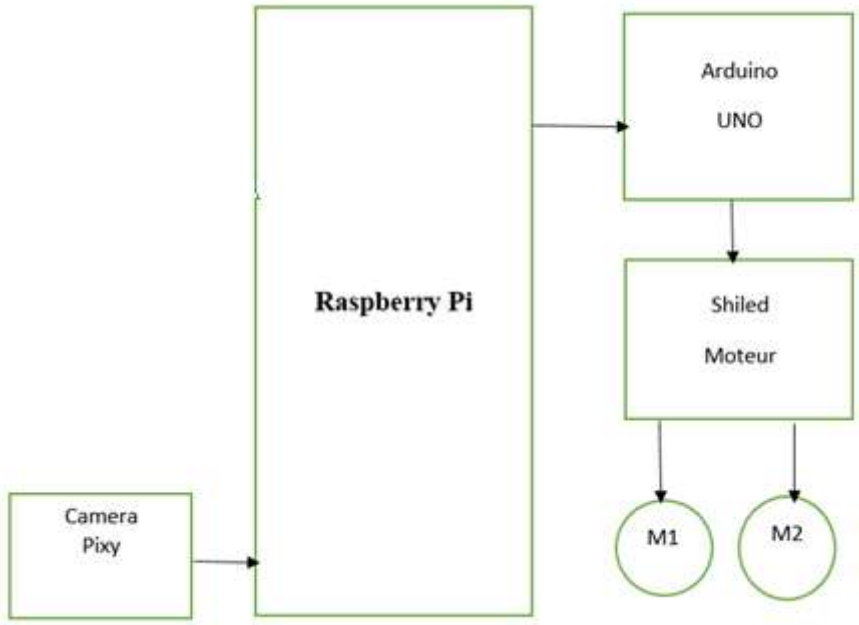


Figure.IV.13. *Schéma bloc du projet*

La figure IV.13 représente le schéma synoptique de l'ensemble robot mobile et le système de commande embarquée que nous avons réalisé au cours de notre projet. Ce dernier s'articule autour de 4 blocs principaux. Les deux premiers blocs représentent l'association du robot mobile (avec ces moteurs électriques à courant continu) et de la carte Arduino qui sert à gérer la commande bas niveau des deux moteurs du robot. Tandis que les deux derniers blocs regroupés au sein de la carte raspberry et qui ont pour rôle le calculer des lois de commandes nécessaire à la poursuite de la cible à partir des données fournies par le système de vision Pixy.

Dans le premier chapitre nous avons parlé sur la robotique mobile en général, nous avons abordé les grandes classes du robot mobile et leur modèle, Il existe plusieurs classes de robots à roues déterminées, principalement, par la position et le nombre de roues utilisées.

Le robot mobile que nous avons utilisé dans notre projet, est un robot mobile de type tricycle, ce robot est techniquement un Arduino robot, nous avons le choisi grâce à sa plateforme simple, prête-à-utilisé, libre et facile à utiliser, mais nous avons utilisé seulement la plateforme qui contient les moteurs et les roues. Car au fur et à mesure que nous progressions dans notre travail, nous nous sommes orientés vers une architecture matérielle

universelle capable de commander n'importe quel robot à condition d'avoir accès à l'alimentation de ces moteurs électriques qu'ils soient à CC, pas à pas ou de type servomoteurs. Il suffit juste d'apporter quelques adaptations mineures au programme de la carte Arduino.

7. Le schéma d'asservissement visuel du robot mobile Arduino

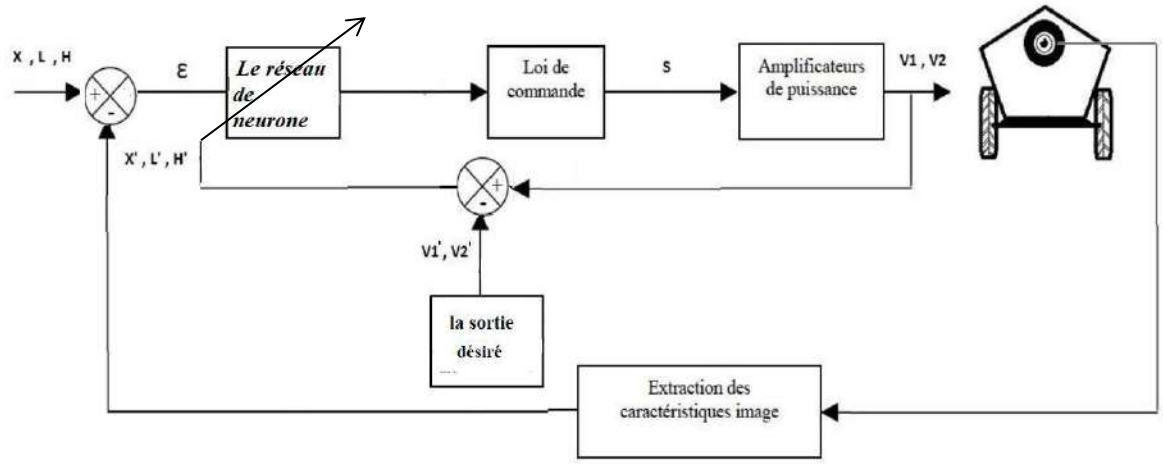


Figure.IV.14. *structure d'asservissement visuel utilisé*

Cette structure consiste à calculer directement les valeurs des commandes à partir des caractéristiques de la cible dans les images. L'erreur de positionnement est calculer directement dans l'espace images et c'est cette variable que le contrôleur devra réduire. Ces caractéristiques sont extraites à partir des informations issues de la Cam Pixy. Lorsque la cible désignée apparaît dans le champ de la caméra, le traitement d'images va repérer cette cible est va nous fournir un certains nombres de données sur cette objet en mouvement.

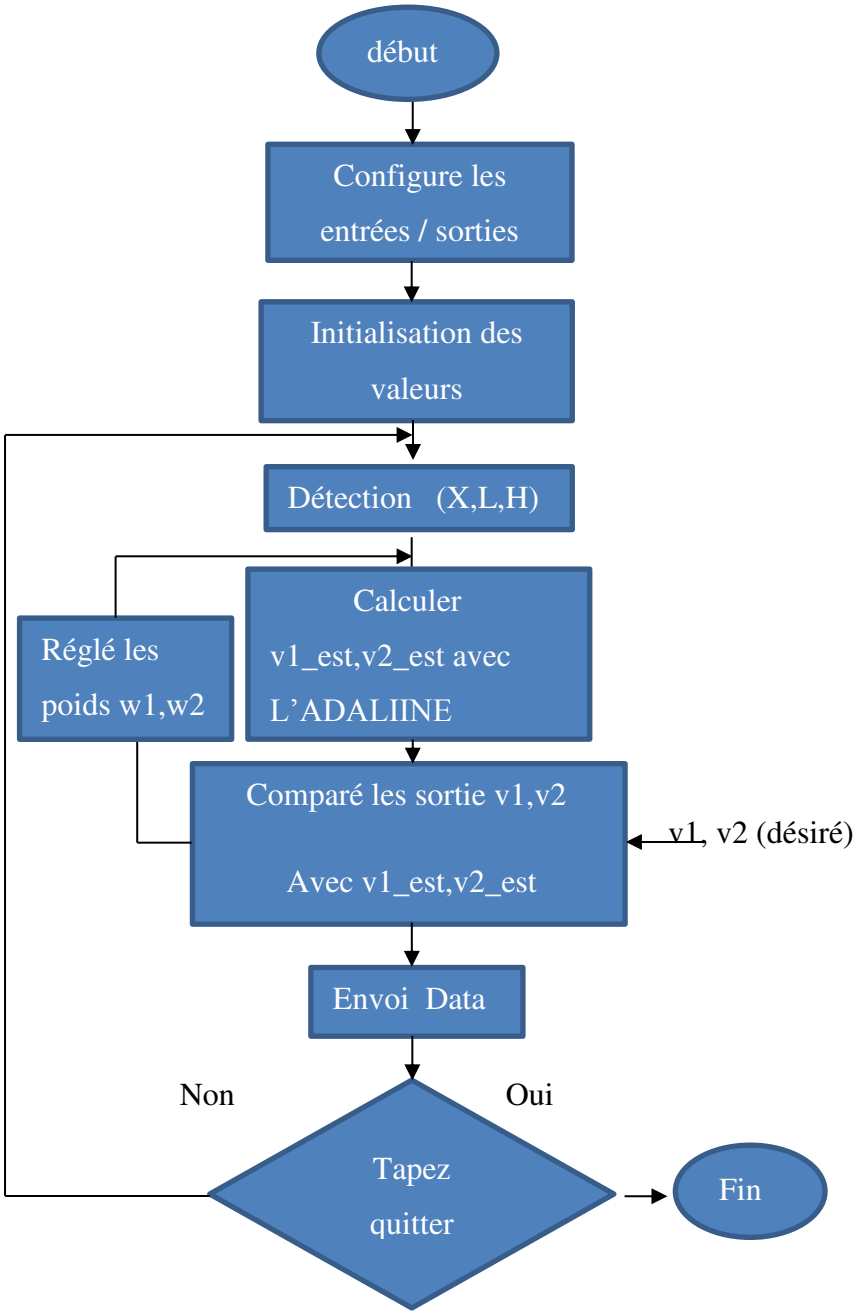
L'utilisation de la vision monoculaire, ne permet pas d'estimer la profondeur de la cible. Pour contourner cette limitation (en évitant d'introduire une deuxième caméra), nous avons calculé la surface de la cible. Si la valeur de surface grandie, la cible est proche de la caméra et donc du robot et inversement si, la valeur de la surface diminue, la cible s'éloigne du robot. Pour atteindre cette règle, il est conseillé d'utiliser une cible de ballon. Par conséquence, cette approche nous permettra d'asservir et de suivre la cible quand elle se déplace en perpendiculaire du plan image.

Quant aux déplacements de la cible sur le plan parallèle au plan image, nous utiliserons les coordonnées (estimées en pixels) X, Y du centre de gravité de la cible.

En réalité, nous utilisons uniquement la coordonnée X pour maintenir le centre de gravité de la cible au centre de l'image.

Par conséquent, grâce à l'estimation de la position de la cible dans l'espace cartésien (faite à partir des données visuelles L, H et X), nous avons contraint le robot à maintenir la cible à une distance donnée et surtout au centre de l'image. C'est le principe même de l'asservissement visuel avec une approche basée sur l'image.

8. Description du programme de la commande neuronale



L'organigramme précèdent décrit le fonctionnement du programme que nous avons mis en œuvre pour réaliser l'asservissement visuel du robot mobile. Dans un premier temps, nous initialisons les valeurs des paramètres. Par la suite, on va définir la surface et la valeur X de notre cible. Ensuite, nous calculons les deux vitesses estimées fourni par le réseau de neurones et les envoyées le port série Arduino/Raspberry. L'adaptation des poids du contrôleur neuronal est effectuée en comparant les vitesses fournies par le réseau avec celles calculer par le contrôleur traditionnel déjà implémenté et dont le fonctionnement a été déjà

validé. Nous avons utilisé cette méthode car, le robot n'est pas encore doté de codeurs optiques capables de nous fournir les vitesses réelles de déplacement des roues.

Une fois la cible détectée, le contrôleur visuel fournira les consignes nécessaires pour amener le robot vers celle-ci.

Note : la partie traitement d'image et la couleur de l'objet cible est réaliser indépendamment par la caméra Pixy.

Dans la suite, nous vous présentons la structure neuronale utilisée dans l'implémentation du nouveau contrôleur.

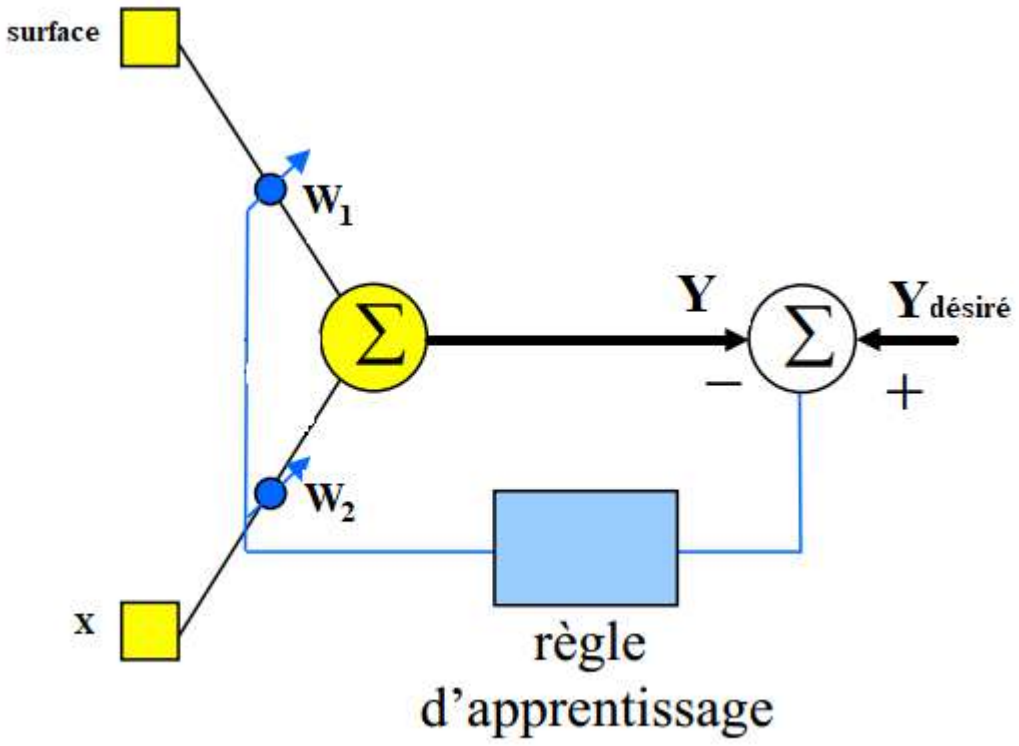
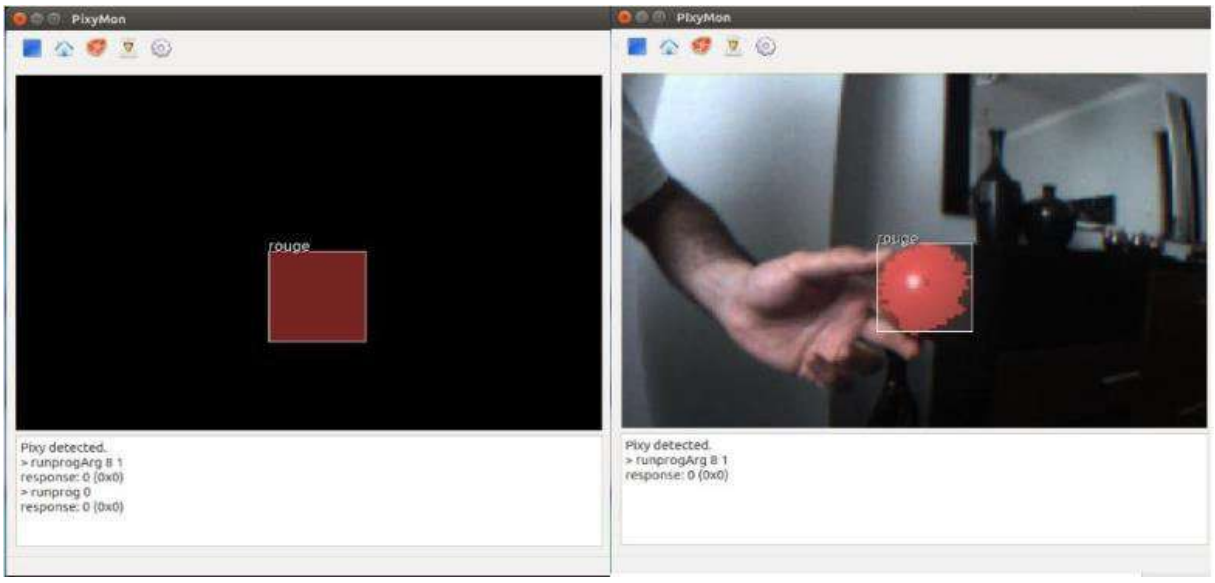


Figure.IV.15. *structure d'asservissement visuel utilisé*

Dans notre cas, nous avons deux entrées, la surface de la cible et la coordonné X du centre de gravité de la cible par rapport au centre de l'image. Les sorties de notre contrôleur neuronal sont les deux vitesses de rotation des roues du robot mobile.

9. Obtenir des données du Pixy



(b) (a)

Figure.IV.16. *Voir ce que voit pixy.*

La figure (a) à droit montre ce que Pixy voit lorsqu'il détecte l'objet cible. La figure (b) à gauche montre ce que Pixy voit lorsqu'il détecte l'objet cible et envoi ses caractéristiques visuelles, à savoir la largeur L et la hauteur H et de sa position X dans l'image

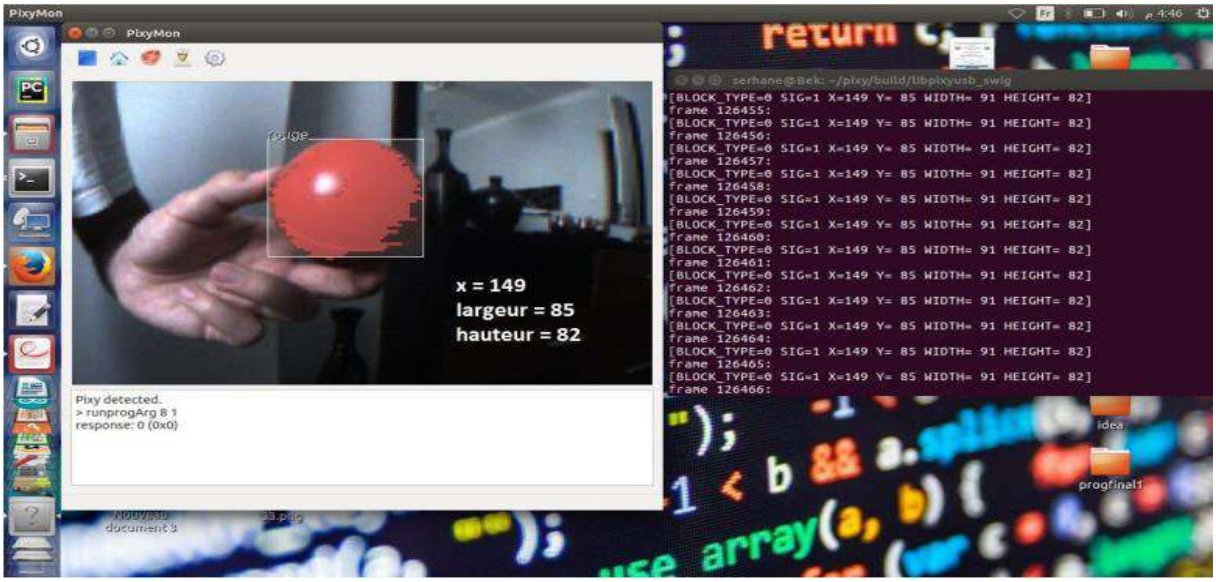


Figure.IV.17.*test de détermination de largeur L, hauteur H et de X*

10. Les tests en situation de notre contrôleur neuronal dans l'asservissement visuel du robot mobile

Après avoir évaluer et maitriser l'influence des paramètres intrinsèques du réseau Adaline, nous nous apprêtons à tester son fonctionnement et à évaluer ses performances dans la commande de notre robot mobile. Pour cela nous avons réalisé une série de tests pratiques pour, d'une part trouver les valeurs optimales des paramètres intrinsèques de notre contrôleur et d'autre part, valider et comparer son fonctionnement à celui de l'ancien contrôleur conventionnel.

Dans ce qui va suivre, nous vous présentons les résultats de nos expérimentations, d'abord sur une trajectoire rectiligne simple puis sur une trajectoire un peu plus complexe qui possède plusieurs changements de direction.

Test N°1 : trajectoire rectiligne

Cette première partie des tests nous permet de trouver les meilleurs taux d'apprentissage et les conditions initiales optimales des poids du réseau Adaline.

- **Choix du coefficient d'apprentissage α** :

➢ **Exemple 01 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=1.5)
- Initialisation des poids W = [0,0]

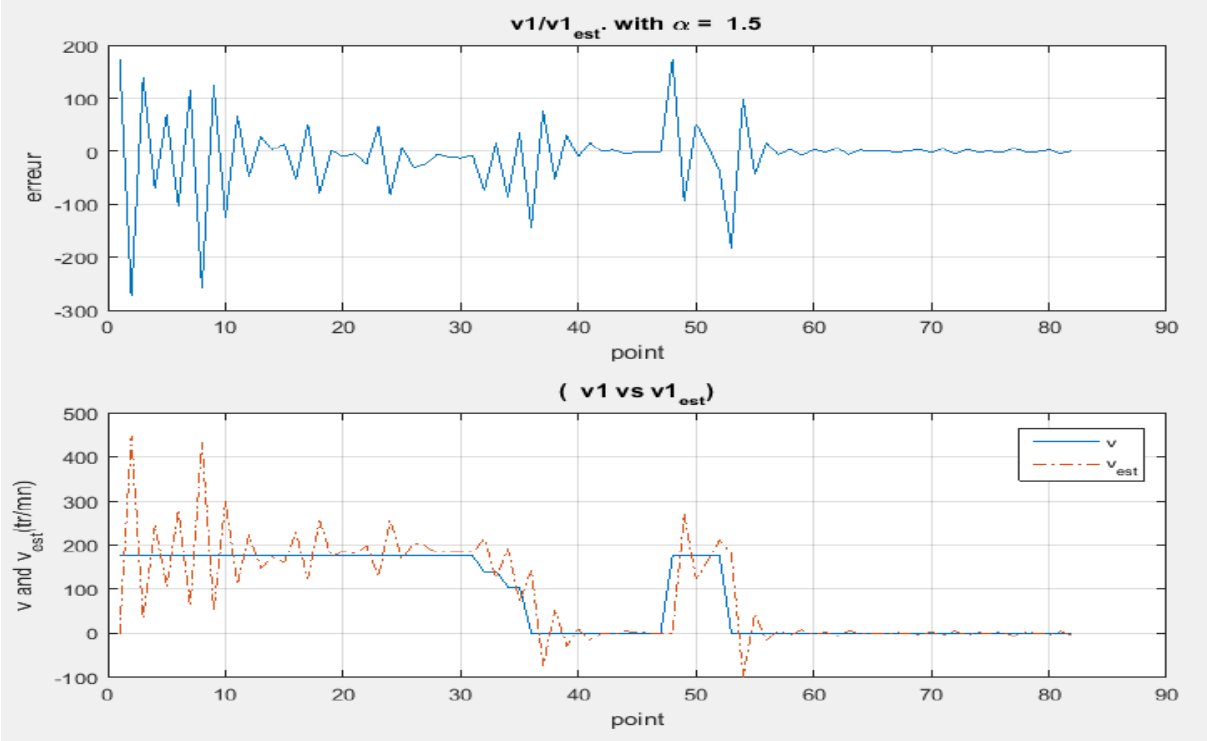


Figure.IV.18. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur*

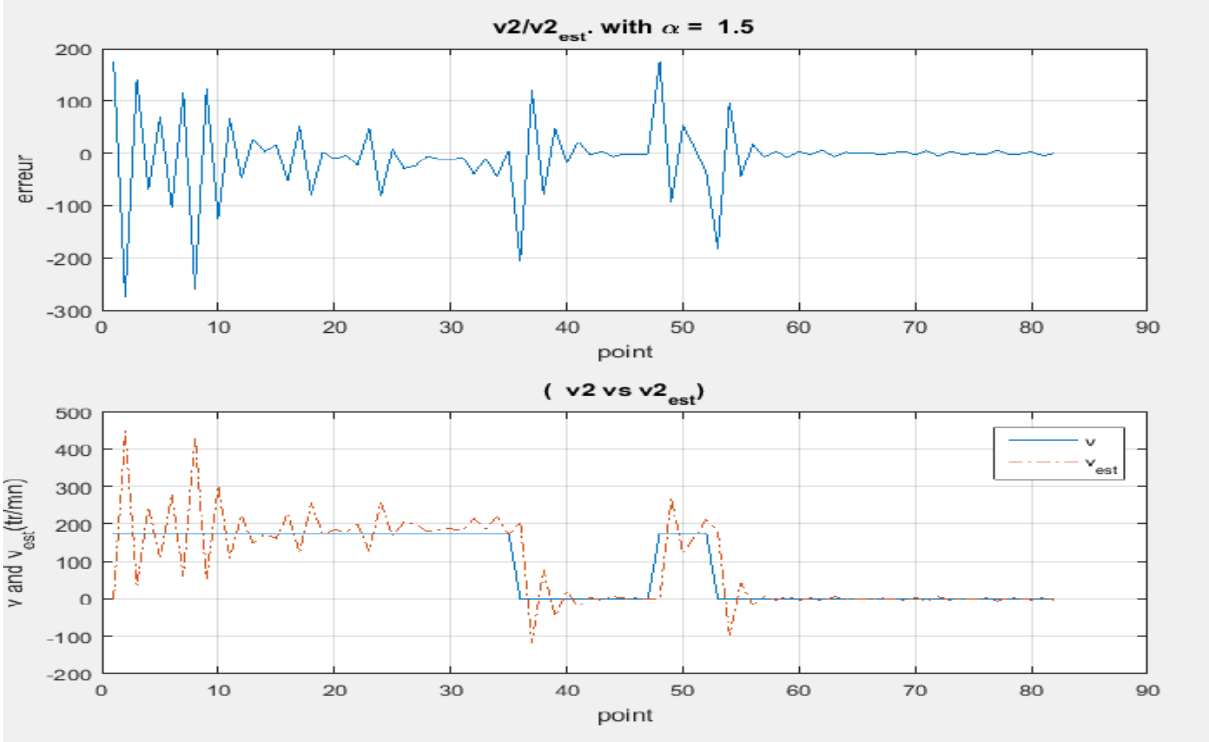


Figure.IV.19. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur*

➢ **Exemple 02 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=0.5)
- Initialisation des poids W = [0,0]

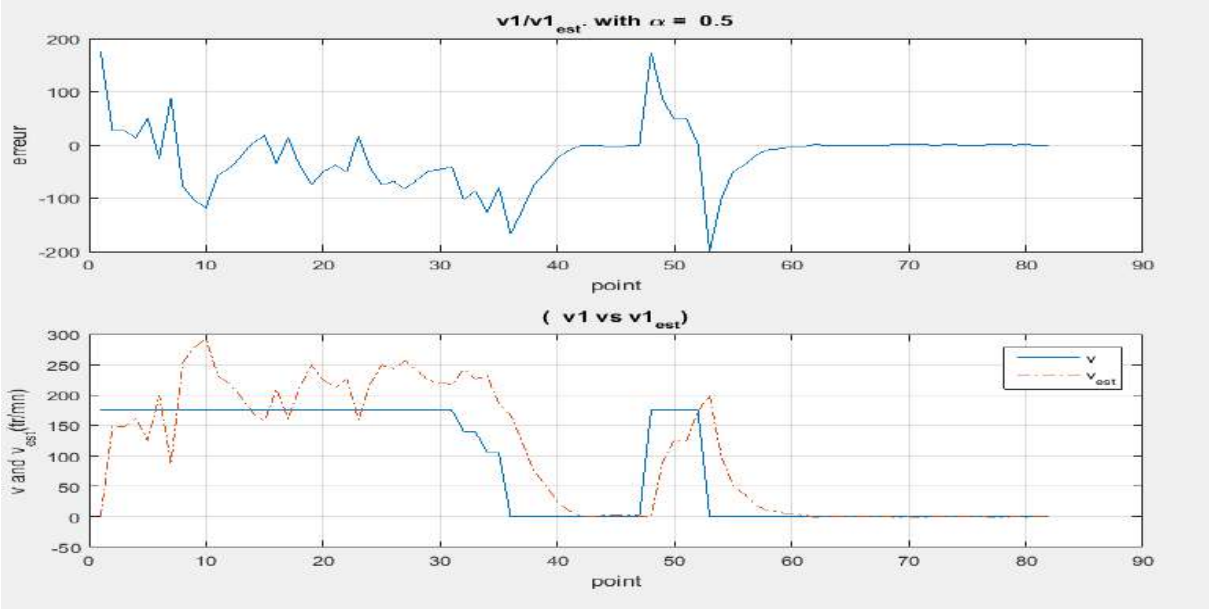


Figure.IV.20. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur*

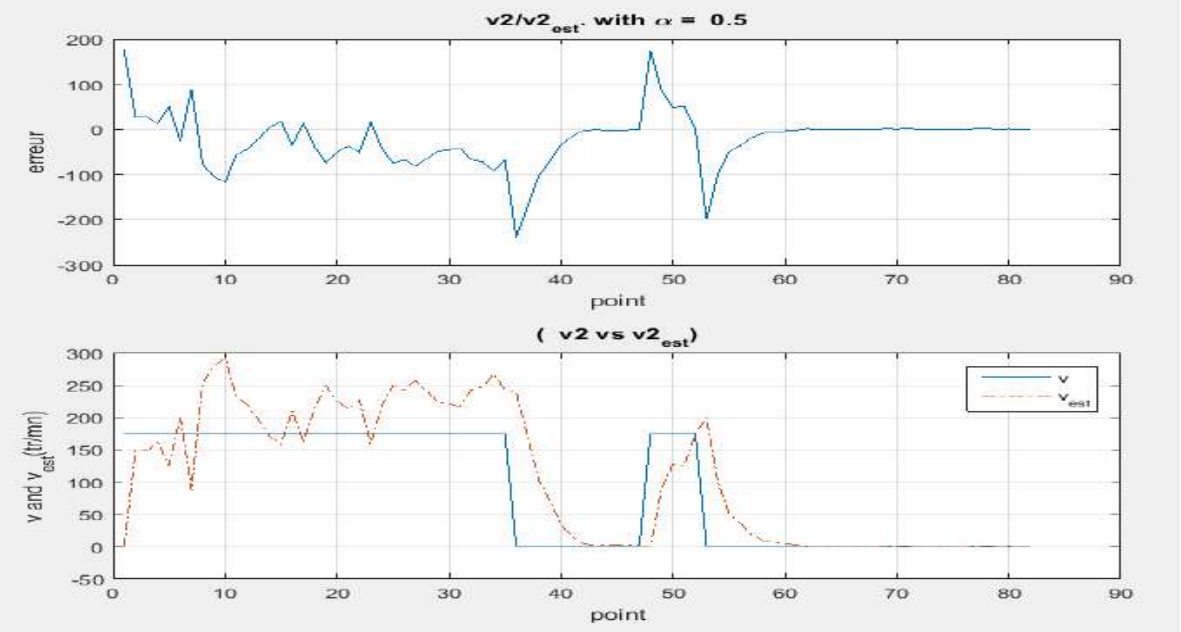


Figure.IV.21. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur*

➢ **Exemple 03 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha dégressif passant de α = 1.5 à α = 0.5
- Initialisation des poids W = [0,0]

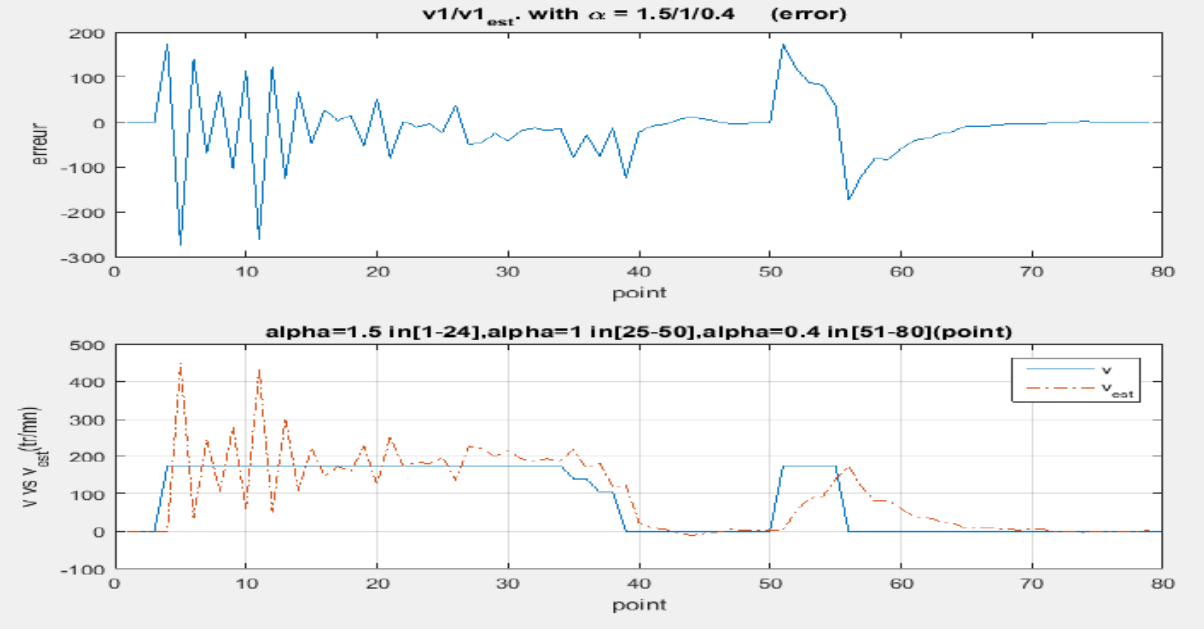


Figure.IV.22. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur*

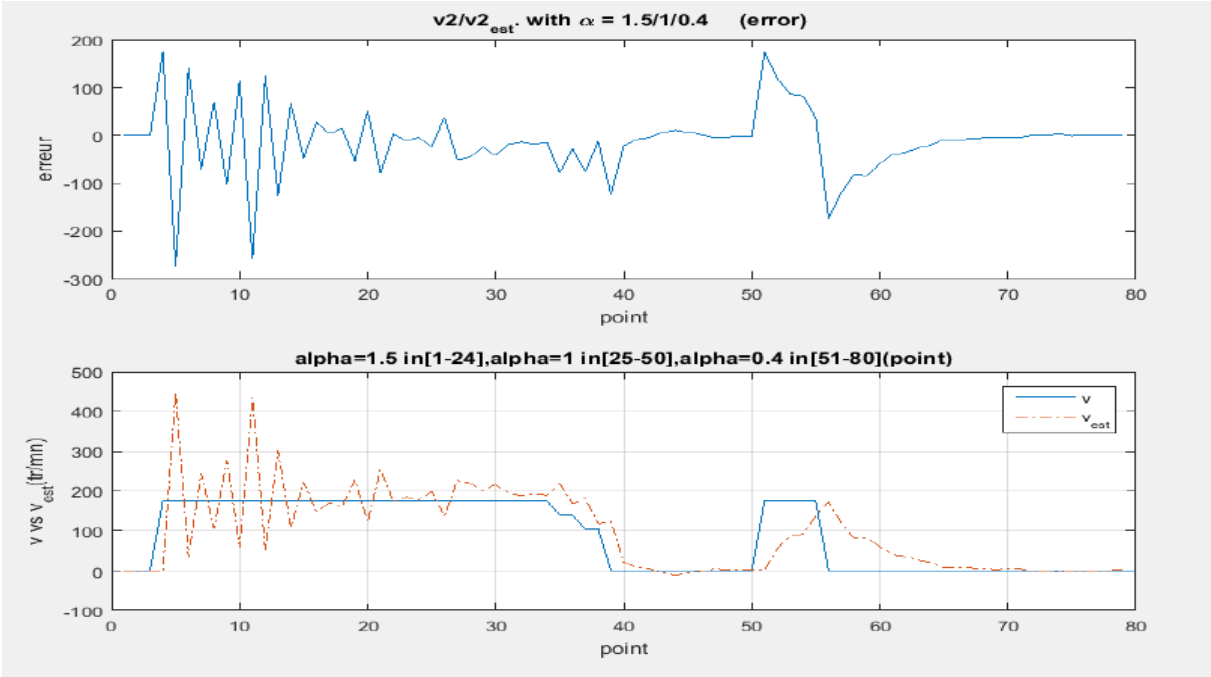


Figure.IV.23. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur*

➢ **Exemple 04 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=1)
- Initialisation des poids W = [0,0]

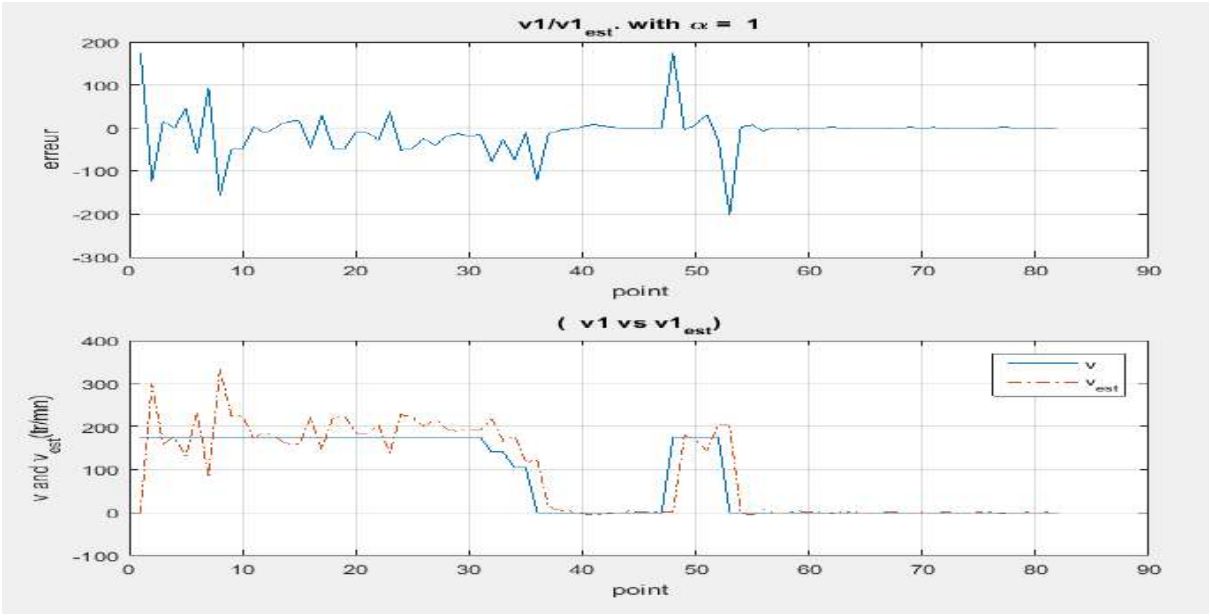


Figure.IV.24 *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur*

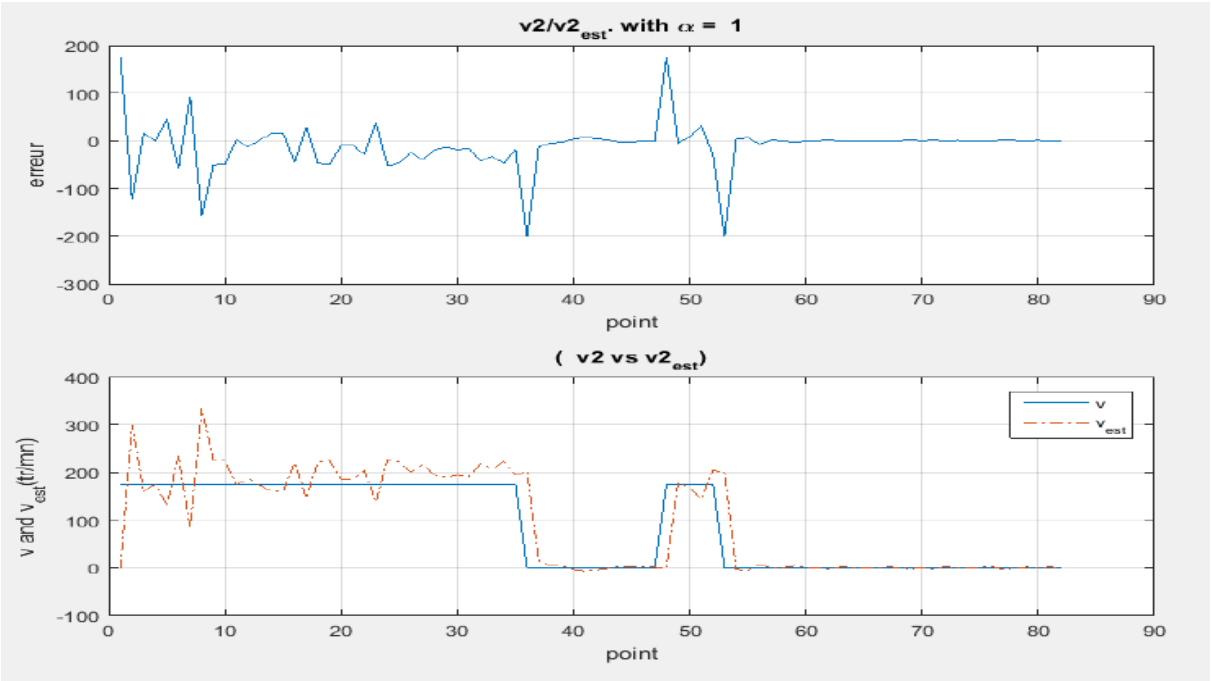


Figure.IV.25. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur*

Analyse :

Après l'analyse de ces graphes, nous constatons clairement que la valeur optimale du taux d'apprentissage du réseau Adaline est alpha =1.

- **Choix des valeurs initiales des poids du réseau Adaline**

  - **Exemple 01 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=1)
- Initialisation des poids W = [1 1]

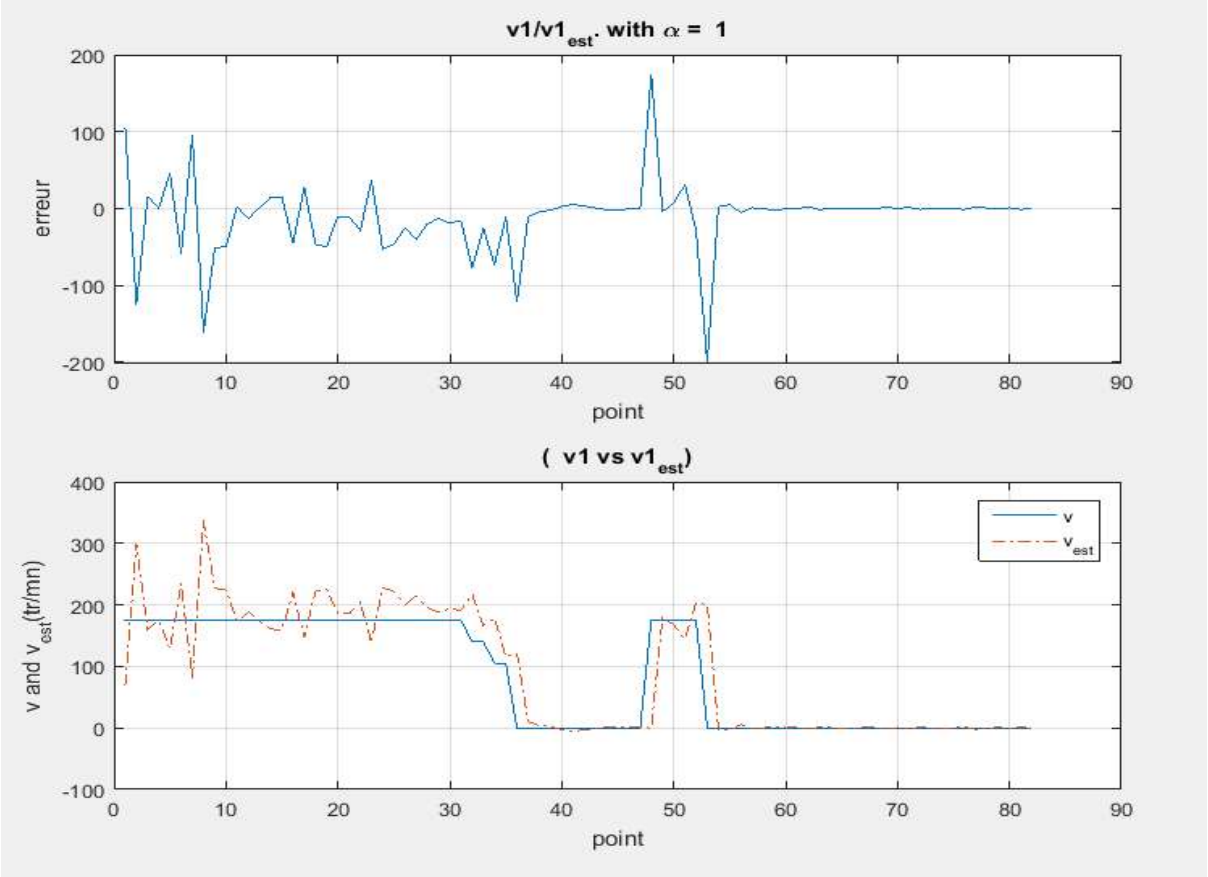


Figure.IV.26. *Graphe illustrant l'évolution des vitesses v1, v1_est et de l'erreur*

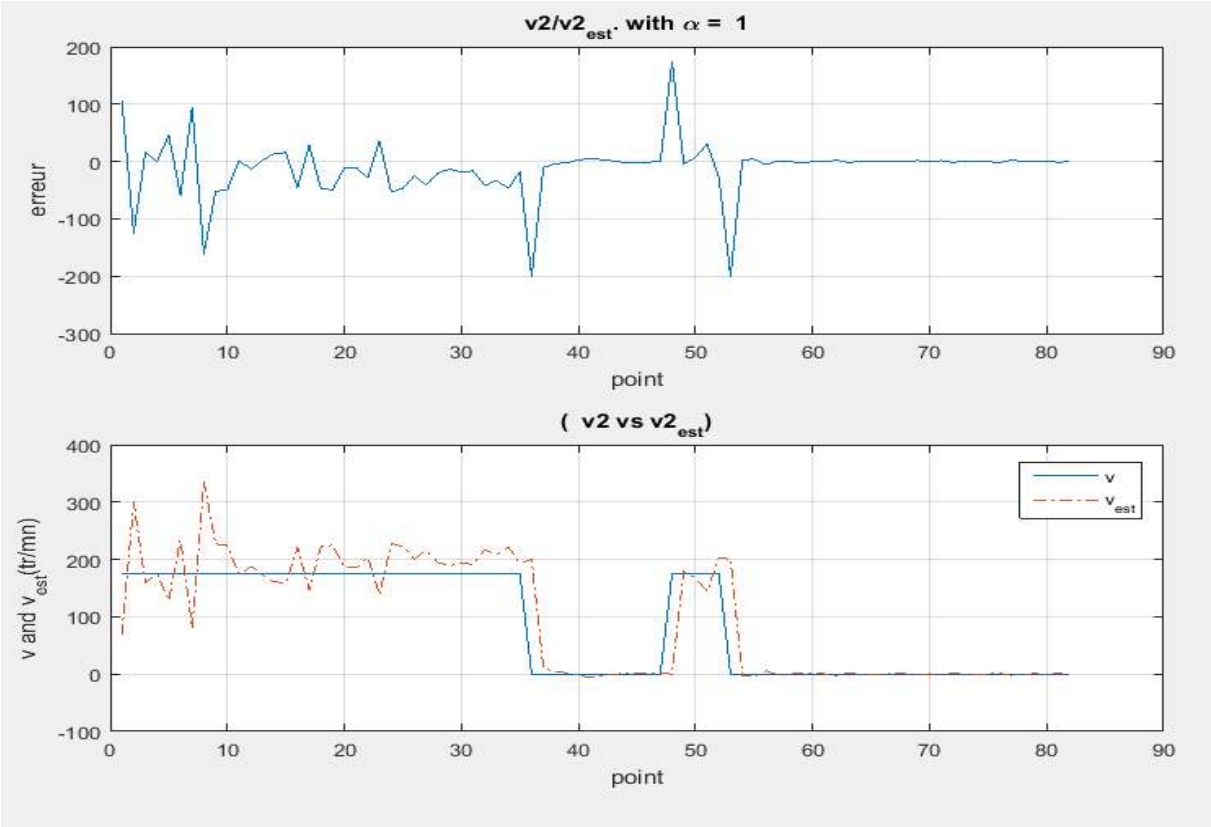


Figure.IV.27. *Graphe illustrant l'évolution des vitesses v2, v2_est et de l'erreur*

- **Exemple 02 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=1)
- Initialisation des poids W = [3 3]

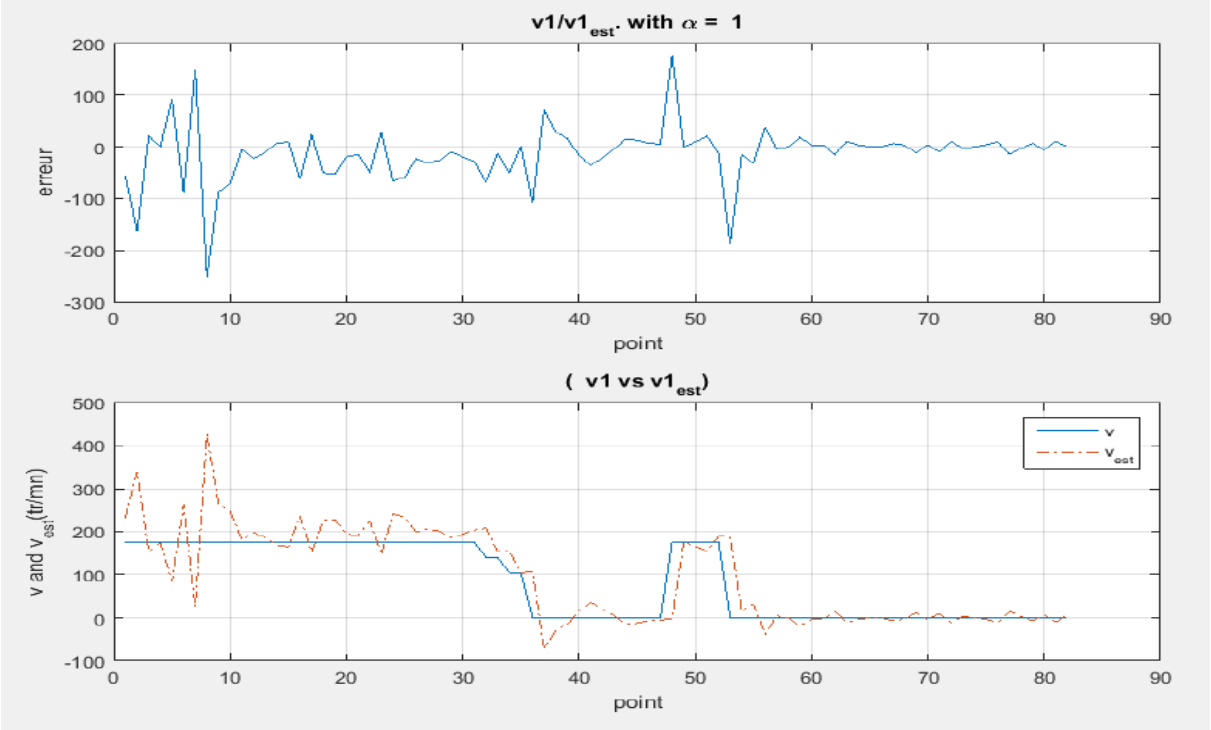


Figure.IV.28. *graphe des vitesses v1, v1_est et erreur entre eux*

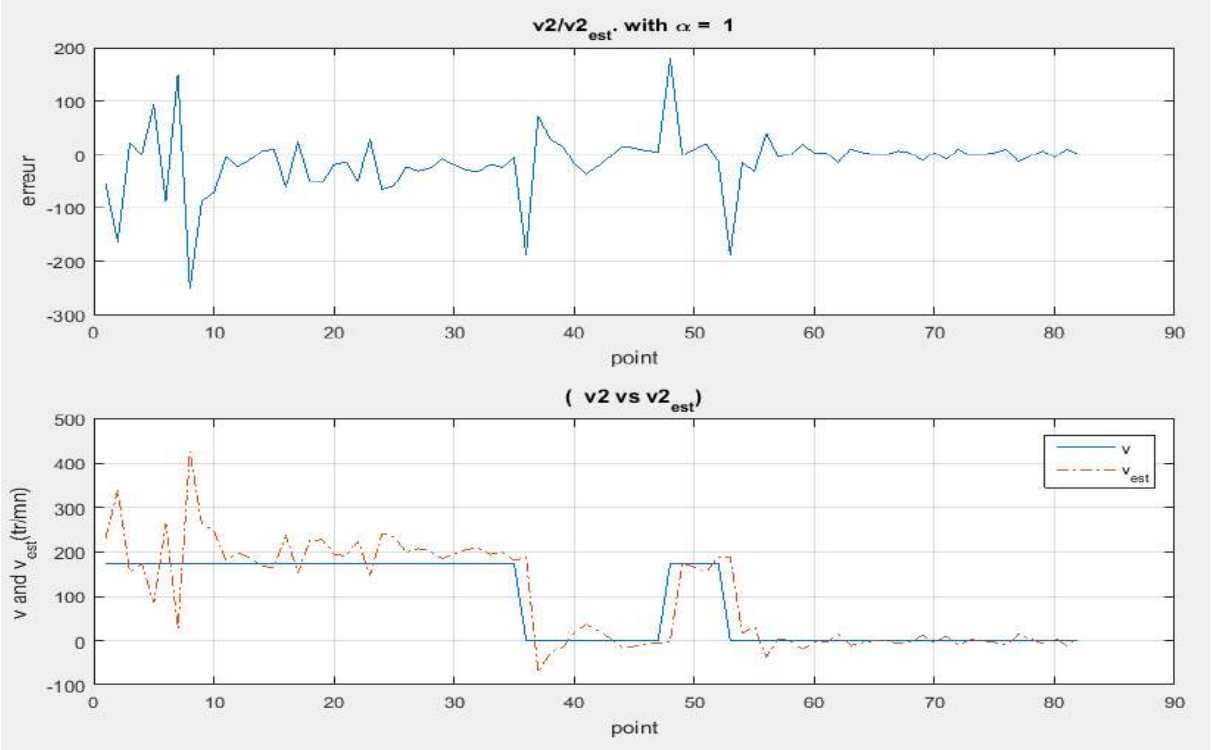


Figure.IV.29. *graphe des vitesses v2, v2_est et erreur entre eux*

Analyse :

Après l'analyse de ces graphes, on voit qu'il n'y avait pas de différence apparente dans ces graphes malgré le changement dans les poids initial.

Test N°2 : trajectoire courbée

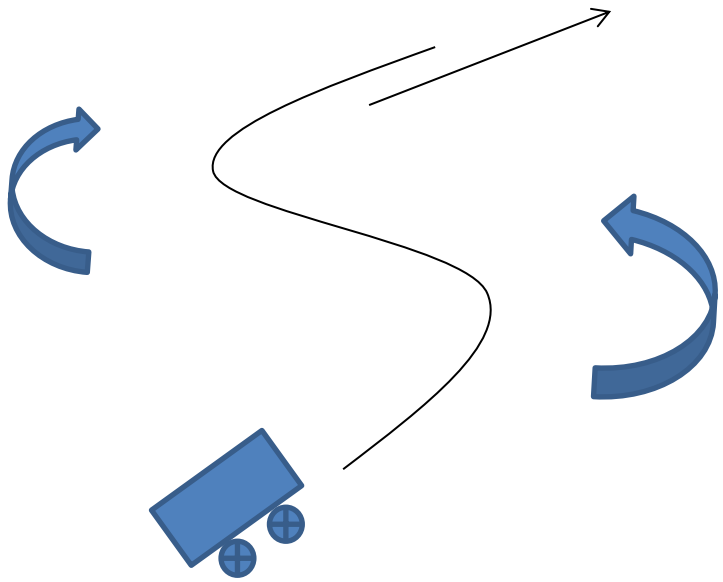


Figure.IV.30. *la trajectoire courbée utilisé*

Cette trajectoire est souvent une partie des mouvements que subi notre robot mobile et donc représentative de son fonctionnement normal. Elle permet de tester est mettre en exergue les performances du contrôleur neuronal lors des changement de direction (changement de signe des entrées).

Lors de ce test expérimental, nous avons logiquement commencer par retenir la configuration optimale du contrôleur neuronal obtenu précédemment du taux d'apprentissage du réseau Adaline est alpha =1

- **Choix des valeurs initiales des poids du réseau Adaline**
  - **Exemple 01 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=1)
- Initialisation des poids W = [0,0]

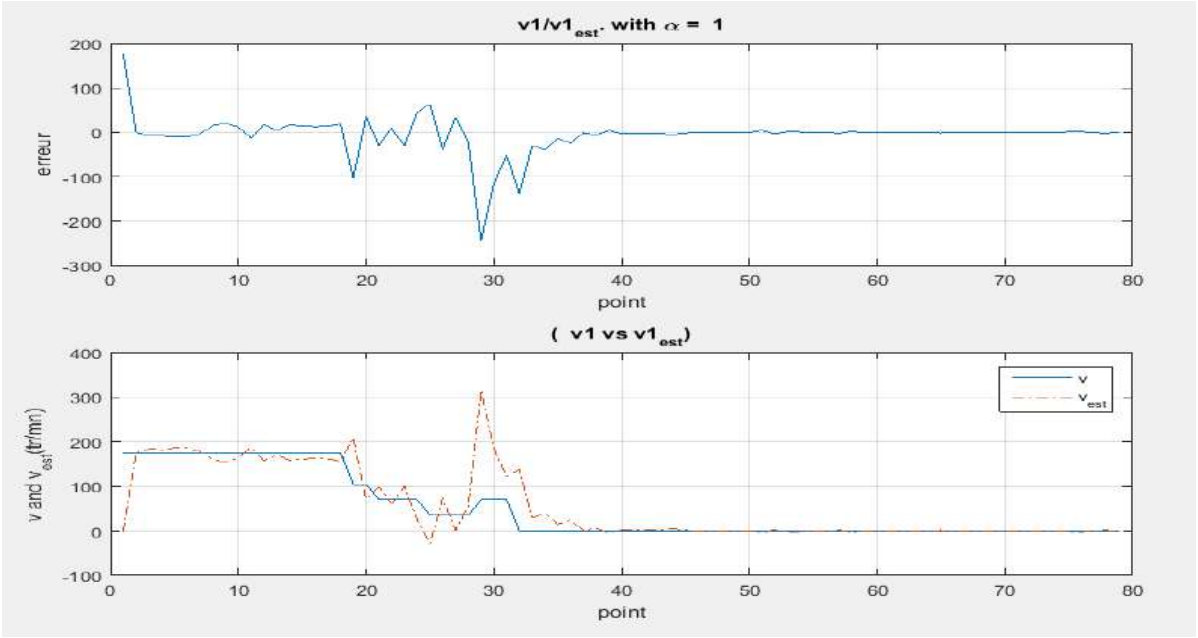


Figure.IV.31. *graphe des vitesses v1, v1_est et erreur entre eux*

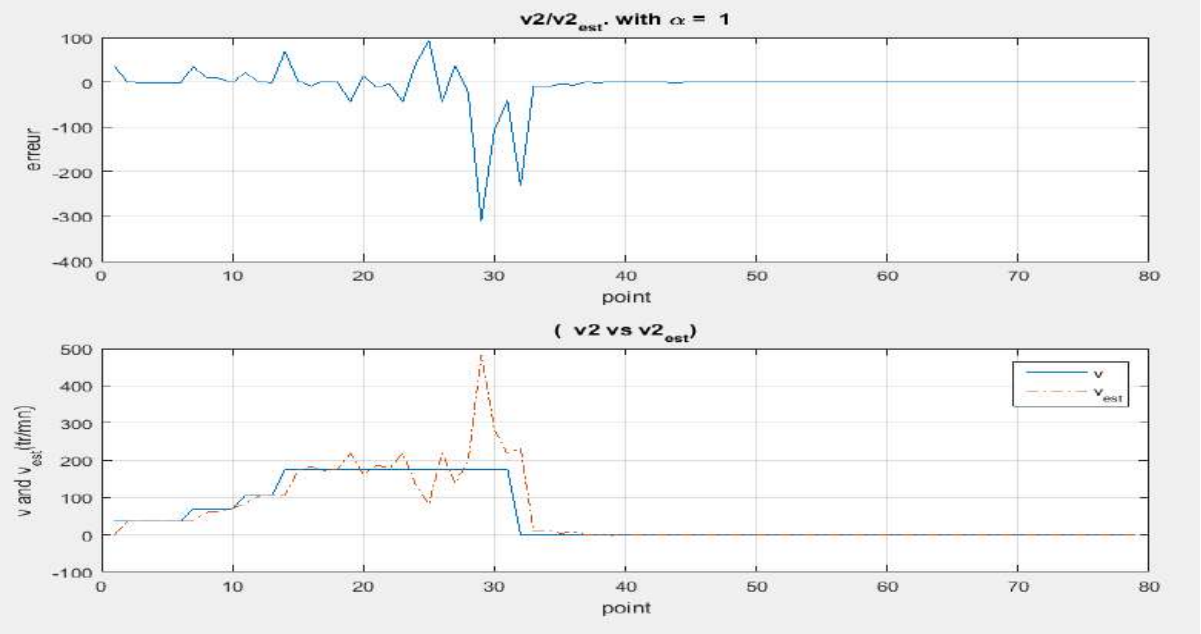


Figure.IV.32. *graphe des vitesses v2, v2_est et erreur entre eux*

- **Exemple 02 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=1)
- Initialisation des poids W = [1 1]

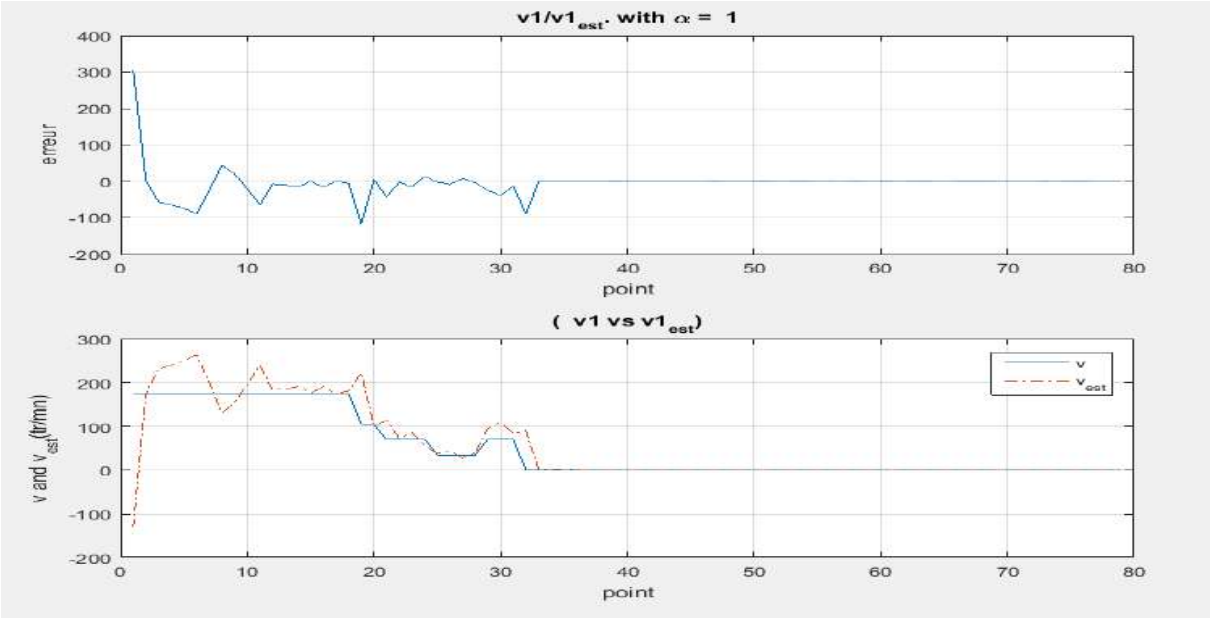


Figure.IV.33. *graphe des vitesses v1, v1_est et erreur entre eux*

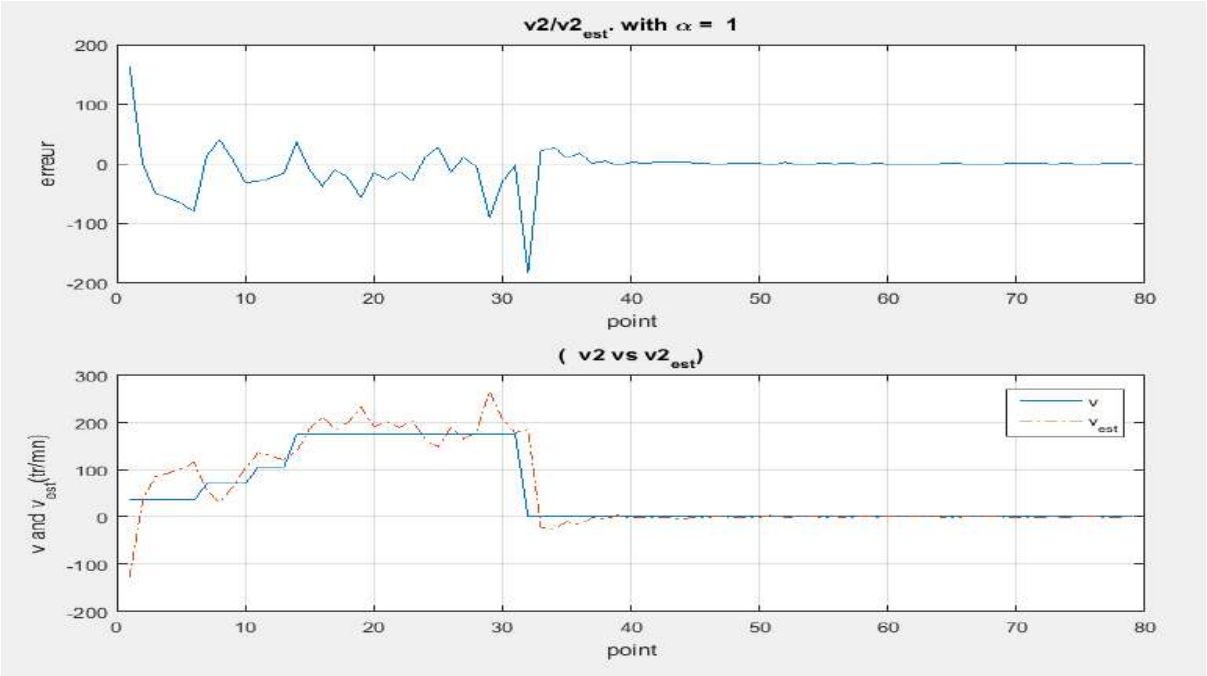


Figure.IV.34. *graphe des vitesses v2, v2_est et erreur entre eux*

- **Exemple 02 :**

Les paramètres utilisés :

- Coefficient d'apprentissage alpha (α=1)
- Initialisation des poids W = [3 3]

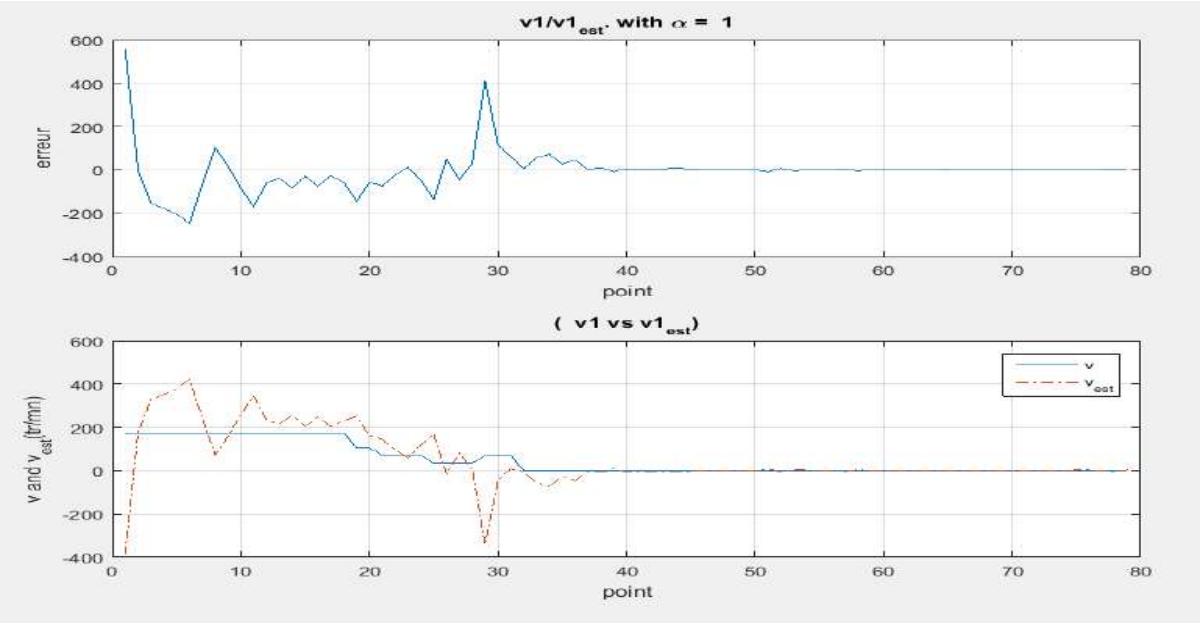


Figure.IV.35. *graphe des vitesses v1, v1_est et erreur entre eux*

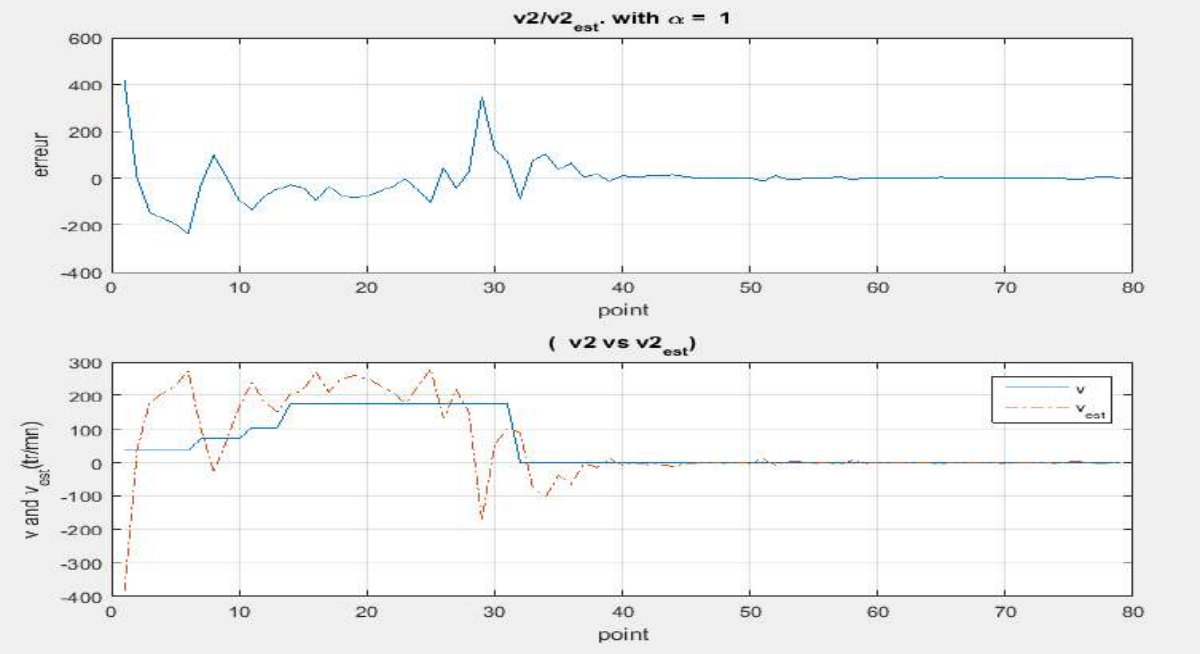


Figure.IV.36. *graphe des vitesses v2, v2_est et erreur entre eux*

Analyse des résultats obtenus :

Il en résulte de tous les tests expérimentaux que nous avons menés, que le meilleur taux d'apprentissage doit être proche de un (1) et que les meilleures initialisations des poids de ce réseau de neurones est (1,1), dans ce cas précis.

Les essais menés en laboratoire ont validé notre programmation du contrôleur neuronal qui arrivé à faire suivre, sous certaines conditions, au robot la cible mouvante uniquement à partir d'informations visuelles.

11. Conclusion

Dans ce chapitre, nous avons présenté le fonctionnement détaillé et les modifications majeures apportées à la plate-forme robot mobile pour réaliser le suivi d'une cible à partir de données visuelles à base d'une méthode de l'intelligence artificiel c'est la commande neuronale

Dans un premier temps, nous avons étudié et décrit l'agencement et l'organisation de notre système embraqué composé du système de vision Pixy de la carte Raspberry Pi3 et de la carte Arduino / Shield. Celles-ci grâce à leur interconnexion, via le port USB, communiquent et s'échangent les données nécessaires à la commande du robot mobile.

Dans un second temps, nous avons détaillé le rôle de chaque étage. Le plus important est l'acquisition, le traitement des données visuelles avec le système Pixy. Ces opérations ainsi que la loi de commande ont été regroupées dans un seul programme écrit en Python et exécute sur la carte Raspberry. Par ailleurs, les performances de cette carte ont permis l'implémentation de notre contrôleur neuronale qui permet d'asservir les mouvements du robot uniquement à partir d'informations visuelles.

Enfin, l'application expérimentale que nous avons mise en œuvre a permis de mettre en exergue d'une part, nos choix quant à l'utilisation de ces cartes embarquées et d'autre part l'utilisation notre algorithme neuronale. Au vu des résultats obtenus lors des différentes expérimentions ont montré de bon fonctionnement de notre contrôleur neuronal. En effet, ces performances ne sont pas phénoménales, mais toute fois correcte au vu de la simplicité de la structure du réseau de neurones Adaline. Mais notre travail a permis de mettre en exergue

qu'il est possible d'implémenter d'autres réseaux de neurones voir d'autres techniques intelligentes sur notre plate-forme robot-vision. Cela est dû qu'elle dispose de ressources matérielles suffisamment puissante pour accueillir ces contrôleurs complexes.

Bien entendu, d'autres tests sont à prévoir pour étudier de plus près les performances de notre contrôleur. Mais en tout état de cause, et au vu des résultats obtenus, nous avons rempli les conditions qui nous ont été imposées, à savoir la conception et la réalisation d'une plateforme évolutive, intelligente et embraquée pour asservir un robot mobile à partir de données visuelles.

Conclusions et Perspectives

Au terme de ce travail, nous avons apporté des modifications majeures à la plate-forme robot mobile pour réaliser le suivi d'une cible à partir des données visuelles. Cet apport se matérialise par l'implémentation d'un contrôleur neuronal.

La solution que nous avons adoptée dans notre projet a considérablement amélioré les performances et le traitement d'image et d'asservissement visuel. . De plus, cette configuration nous a permis d'implémenter notre contrôleur neuronal. Ce dernier basé sur le réseau de neurones ADALINE nécessite l'utilisation de ressources informatiques importantes (mémoire, puissance de calcul, …) que notre architecture est capable de fournir.

Lors de la réalisation de la boucle d'asservissement, le plus important etait l'acquisition, le traitement des données visuelles avec le système Pixy. Ces opérations ainsi que la loi de commande ont été regroupées dans un seul programme écrit en Python et exécute sur la carte Raspberry.

Par ailleurs, les différents tests expérimentaux que nous avons entrepris, nous ont permis de valider le fonctionnement de cette nouvelle version de la plate-forme mais aussi de mettre en évidence les performances de l'algorithme de commande que nous avons élaboré. En effet au vu des résultats obtenus lors des différentes expérimentions, nous pouvons conclure que notre contrôleur neuronal a rempli son rôle. Certes, ces performances ne sont pas phénoménales, mais toute fois correcte au vu de la simplicité de la structure du réseau de neurones Adaline.

Enfin, ce travail nous a permis d'aborder les différentes problématiques inhérentes à la robotique mobile, notre travail a permis de mettre en exergue qu'il est possible d'implémenter d'autres réseaux de neurones voir d'autres techniques intelligentes sur notre plate-forme robot-vision. Cela est dû qu'elle dispose de ressources matérielles suffisamment puissante pour accueillir ces contrôleurs complexes.

Quant aux perspectives futures que nous envisageons pour compléter ce travail :

-
- Faire bouger le système de vision Pixy pour mieux voir l'interface et rechercher la cible, que ce soit au démarrage du système ou après la perdre de la cible

- Implémenter des réseaux neurones plus complexes pour améliorer les performances du notre plate-forme robot-vision.
- Ajouter une source de lumière à l'avant de la Cam Pixy pour augmenter la clarté des couleurs.

Bibliographie:

[1]: Damien LHOMME-DESAGES, "*Commande d'un robot mobile rapide à roues non directionnelles sur sol naturel* ", Thèse de doctorat, l'Université Paris 6, 12 Avril 2013

[3] P. Gaussier, C. Joulain, J. P. Banquet, S. Lepretre et A. Revel : "The vi- sual homing problem : an example of robotics/biology cross-fertilisation". Robotics and autonomous systems, vol. 30(1–2), pp. 155–180, 2000.

[4] : Généralités sur la robotique mobile par ISTIA, Université Angers

[5] : Damien LHOMME-DESAGES, "Commande d'un robot mobile rapide à roues non Directionnelles sur sol naturel ", Thèse de doctorat, l'Université Paris 6, 12 Apr 2013

[11] Jean-François Jodouin. Les réseaux neuromimétiques, modèles et applications. © Hermès,
Paris, 1994

[12] Anil K, Jain. Jianchang Mao. K, Mohiuddin. Artificial Neural Network. IEEE computer Special Issue On Neural Computing, 1996.

[13] Vaseghi, Saeed V .Advanced Digital Signal Processing and Noise Reduction, .Published By: John Wiley & Sons, Ltd.Edition

[14] D. Robert Iskander, Abdelhak M. Zoubir. Estimation of the Parameters of the K-Distribution

[15] BENAISSA el mahdi, BENYAIHIA Dalal, " *Asservissement Visuel d'un robot mobile*", Projet de fin d'étude, Université Yahia Fares de Médéa, année 2016

Webographie :

[2] Mazen Al Homsi, “ main humaine, main robotique, qu’elle est la différence ? ”,

UTBM, 2016

http://http://mrmhutbm.e-monsite.com/pages/apercu-historique.html

[6] https://www.bananarobotics.com

[7] Arduino, “Arduino Board Uno”,

https://www.arduino.cc/en/main/arduinoBoardUno

[8] Raspberry pi, “Raspberry pi 3 model B, raspberry pi foundation UK registred charity

https://www.raspberrypi.org/products/raspberry-pi-2-model-b/

[9] ATMEL, “8-bit AVR Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash”

http://www.atmel.com/dyn/resources/prod_documents/doc8161.pdfasd Chapter 21 “2-wire Serial Interface”

[10] Anthony Rowe, Rich LeGrand, Scott Robinson, “Aperçu sur CMUcam5 Pixy”,

http://cmucam.org/projects/cmucam5

[16] Edward Getz, “*Teach Pixy an Object*”, 08/10/2015

http://cmucam.org/projects/cmucam5/wiki/Teach_Pixy_an_Object_