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 Génie Electriques

Spécialité : Automatique Des Systèmes

THEME

Asservissement visuel d'un robot mobile

Proposé et dirigé par,

DR.R.KARA

Préparé et présenté par,

M. BENAISSA Elmahdi

Melle. BENYAIHIA Dalal

Dédicaces

Je dédie ce travail :

A mes chers parents pour leur amour, leurs encouragements, leur patience et leurs précieux conseils,

À mes chers frères et ma chère sœur Batoula,

À toute ma famille,

À tous mes amis.

Dalal

Dédicace

Je dédie se modeste travail à mes chers parents qui m'ont beaucoup

encouragé et soutenu pendant tout leur vie pour voir le jour-là et pour

ces précieux conseils ainsi pour ces recommandation.

A mes très chers frères et ma sœur.

A toute ma famille, grande et petite sans exception.

A tous mes amis

Que se mémoire soit l'expression de mon profond respect et mon grand

amour.

Mahdi

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 Mr : ***KARA Reda*** *de ses précieux conseils et son aide 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 :

يهـدف هـذا المشـروع إلـى تحقيـق المراقبـة أو التقييـد البصـري للروبـوت المتحـرك، مـن أجـل ذلـك قمنـا بتطـوير البنيـة الهيكليـة للأجهـزة بإضـافة لوحـة آردوينـو و راسـبيري بـاي، مـن جهـة أخـرى اسـتطعنا تحقيـق الـتحكم في الروبوت المتحرك عن طريق البيانات والمعلومات البصرية المستخرجة.

الكلمات المفتاحية :تقييد بصري، الروبوتات المتحركة، راسبيري باي، آردوينو

Résumé

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

Mots clés : asservissement visuel ; robotiques mobiles ; Raspberry Pi ; Arduino ;

Abstract

This project is to provide a visual servoing of a mobile robot. To do this, we developed an embedded hardware architecture combining a Raspberry card to an Arduino board. In addition, we have implemented a controller capable of enslaving mobile robot movements from visual data only.

Keywords: visual servoing; mobile robotics; Raspberry Pi; Arduino

Sommaire

Listes des figures

Listes des tableaux

Notation et acronymes

Introduction générale 1

Chapitre I : Introduction à la robotique mobile

I.1. Introduction 4

I.2. Aperçu historique 4

I.3. Exemples d'application 7

I.4. Architecture des robots mobiles 9

I.4.1. La structure mécanique et la motricité 9

I.4.1.1. Les mobiles à roues 9

I.4.1.2. Les mobiles à chenilles 9

I.4.1.3. Les mobiles marcheurs 10

I.4.1.4. Les robots rampants 10

I.4.1.5. La motricité et l'énergie 10

I.4.2. Les organes de sécurité 10

I.4.3. Traitement des informations et gestion des taches 12

I.5. Les robots mobiles à roues 12

I.5.1. L'unicycle 13

I.5.2. Le tricycle 14

I.5.3. Les véhicules (structure d'ackerman) 14

I.5.4. Robot à traction synchrone 15

I.6. Capteurs en robotique mobile 16

I.7. Mesure de la rotation des roues 16
I.8. Mesure d'orientation et de position 17
I.8.1. Mesure de la position : le GPS 17
I.8.2.Mesure de l'orientation : Gyromètres 18
I.8.3. Autres capteurs (moins fréquents) 19
I.8.4. Mesure de proximité et de distance 20
I.10. Conclusion 22

Chapitre II : Etude des systèmes embarqués utilisées

II.1. Introduction 24
II.2. Arduino 24
II.2.1. Définition 24
II.2.2. Applications 25
II.2.3. Pourquoi avons nous choisi le système carte Arduino 25
II.2.4. Les outils Arduino 26
II.2.4.1. Le matériel 26
II.2.4.2. Constitution de la carte Arduino 27
II.2.5. La connectique 28
II.2.6. Le Logiciel de gestion de la carte Arduino 28
II.2.6.1 Les boutons 29
II.2.6.2. Moniteur sériel 30
II.2.7. La carte Arduino mega AT2560 30
II.2.7.1. Présentation de la carte arduino Mega AT2560 : 30
II.2.7.2. Synthèse des caractéristiques 32
II.2.7.3. Alimentation 32
II.2.7.4. Mémoire 34
II.2.7.5. Entrées et sorties 34
II.2.7.6. Communication 36

II.2.7.7. Programmation 37
II.2.7.8. Réinitialisation automatique 37
II.2.7.9. Protection du port USB contre la surcharge en intensité 38
II.2.8.Synthèse 38
II.3. La carte Raspberry Pi 39
II.3.1. Historique 39
II.3.1.1. Conception 39
II.3.1.2. Prototype 40
II.3.2. Configurations 40
II.3.3. Les composants standards de Raspberry 42
II.3.4. Les GPIO 44
II.3.5. Mise en marche d'un Raspberry Pi : 45
II.3.5.1. Une carte mémoire SD 45
II.3.5.2. Alimentation micro USB externe 45
II.3.5.3. Un clavier et une souris USB 45
II.3.5.4. Un câble HDMI ou RCA vidéo 46
II.3.5.5. Un câble Ethernet RJ45 46
II.3.5.6. L'écran 46
II.3.5.7. La sortie vidéo composite via la prise Cinch ou RCA 47
II.3.6. Le réseau 47
II.3.7. Préparation de la carte SD 47
II.3.8. Système d'exploitation 48
II.3.9. A la découverte du système d'exploitation Raspbian 48
II.3.9.1. LXDE, le bureau de Raspbian 49
II.3.10. Programmation 49
II.3.11. Synthèse 49
II.4. Communication par USB entre Raspberry Pi et Arduino 50

II.4.1. Installation de la bibliothèque pyserial 51
II.5 Conclusion 52

Chapitre III : Introduction à l'asservissement visuel

III.1. Introduction 54
III.2. Introduction à l'asservissement visuel 54
III.3 Architectures des systèmes asservis par la vision : vers une taxonomie 56
III.4. Classification des systèmes d'asservissements visuels 59
III.5. Modélisation 63
III.6. Conclusion 66

Chapitre IV : Description de la plate-forme robot-vision

IV.1. Introduction 68
IV.2. Robot mobile Arduino 69
IV.2.1. Description 69
IV.2.2 Modélisation 70
IV.3. Le schéma d'asservissement visuel du robot mobile Arduino 71
IV.4. L'architecture matérielle de gestion et de commande de la plate-forme robot-vision 73
IV.4.1. La carte Raspberry Pi : le « cerveau » 73
IV.4.2. La bibliothèque graphique OpenCV 75
IV.4.2.1. Les bases d'OpenCV 76
IV.4.2.2 Les principaux traitements sous OpenCV 76
IV.4.3. L'organigramme 88
IV.4.4. Description du programme d'asservissement visuel 89
IV.4.5. Conclusion 92
Conclusion générale 93

LISTE DES FIGURES

Chapitre I

Figure I.1 : La tortue de Grey Walter (nommée “machina speculatrix” et surnommée Elsi……..……5

Figure I.2 : Robot "Beast" de l’université John Hopkins dans les années 1960…….…..6

Figure I.3 : Le robot Shakey de Stanford en 1969 a été une plate-forme de démonstration des recherches en intelligence artificielle……..….6

Figure I.4 : Genghis, développé par Rodney Brooks au MIT au début des années 1990…7

Figure I.5 : Architecture d’un robot mobile……...……9

Figure I.6 : Synoptique de la sécurité…….……11

Figure I.7. Robot mobile à commande différentielle…….……...……...13

Figure I.8. Robot mobile de type unicycle……..13

Figure I.9. Robot mobile de type tricycle ……..14

Figure I.10. Robot mobile de type voiture…….….15

Figure I.11. Robot mobile à traction synchrone…….….15

Figure I.12. Gyroscope mécanique de Foucault et gyromètre Futaba à structure vibrante.19

Chapitre II :

Figure II.1 : logiciel arduino……...29

Figure II.2 : Présentation des boutons……...30

Figure II.3 : Raspberry Pi Modèle B (image Raspberry Pi Foundation)……40

Figure II.4 : les composants standards d’un raspberry……...42

Figure II.5: communication Arduino et Raspberry……50

Chapitre III

Figure III.1 : Asservissement visuel avec approche en position de type « dynamic look and move»..... 56

Figure III.2 : Asservissement visuel avec approche fondée sur l'image de type « dynamic look and move ». ……..……57

Figure III.3 : Asservissement visuel avec approche en position.……..……...57

Figure III.4 : Asservissement visuel avec approche fondée sur l'image……..……58

Figure III.5 : Principe de l'asservissement visuel 3D…….……59

Figure III.6 : Principe de l'asservissement visuel 2D…….60

Figure III.7 : Asservissement visuel 2D1/2…. ……. 61

Figure III.8 : Asservissement visuel cinématique……...…….. 62

Figure III.9 : Asservissement visuel dynamique……...……..62

Figure III.10 : modèle générique de caméra……..……...…64

Chapitre IV :

Figure IV.1 : schéma bloc du projet……...…… 68

Figure IV.2 : Robot mobile de type tricycle……70

Figure IV.3 : Un robot mobile de type tricycle et son CIR……...……..70

Figure IV.4:structure d'asservissement visuel utilisé…….……72

Figure IV.5 : Raspberry Pi utilisé ……...73

Figure IV.6 : logo de la bibliothèque graphique OpenCV ……..75

Figure IV.7 : schéma bloc des principaux traitements sous OpenCV ……... 76

Figure IV.8 : Chaîne fonctionnel du traitement d'image ……... 77

Figure IV .9 : graphe d'histogramme…….. 78

Figure IV.10 : les défauts d'une image (deux photos à gauche) /séparation des objets du fond (la photo à droite) ……78

Figure IV.11:Graphe Histogramme en utilise LUT…… 79

Figure IV.12:Caractéristique d'image ……83

Figure IV.13 : image intégrale ……84

Figure IV.14:calcule des caractéristiques……... 84

Figure IV.15:Elaboration du classificateur…… 86

Figure IV.16 : masque binaire de la balle verte ……..89

Figure IV.17 : test de détermination de rayon R et X, cas 1 ……..90

Figure IV.18 : test de détermination de rayon R et X, cas 2 ……..……...90

Figure IV.19 : test de détermination de rayon R et X, cas 3 ……..……91

Figure IV.20 : test de détermination de rayon R et X, cas 4……...……91

LISTE DES TABLEAUX

Chapitre I:

Tableau I.1 : Applications des robots mobiles……8

Chapitre II :

Tableau II.1 : caractéristiques de la carte arduinomega 2560……...32

Tableau II.2 : configuration d'un raspberry……...41

Chapitre III :

Tableau III.1 : Description des caméras à point central unique : a_p, a_h, b_h, a_e, b_e dépendent uniquement des paramètres intrinsèques du miroir d et p……..…63

Chapitre IV :

Tableau IV.1 : caractéristique raspberry pi (modèle B) ……74

Notations

Asservissement visuel

$\boldsymbol{s}$: 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).

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

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

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

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

$\boldsymbol{e}$: Fonction de tâche.

$\boldsymbol{L_s}$: 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.

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

$\boldsymbol{R_o}$: 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.

Liste des ACRONYMES

Chapitre I :

CIR: Centre Instantané de Rotation.

AGV: Automatic Guided Vehicle. Véhicule à guidage automatique, ou véhicule autoguidé.

PPT: Points Par Tour. (Unité de mesure)

GPS: Global Positioning System.

SMM: Silicon Micro Machines.

Chapitre 2

Mini-PC: mini personal computer

PIC: Peripheral Interface Controller (contrôleur d'interface périphérique)

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)

FTDI : Future Technology Devices International

TTL: Transistor-Transistor Logic

SPI : Interface Série Périphérique

MISO : multi-input single-output

MOSI : multi-output single-input

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

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

U2C : inter-integrated circuit

FPGA: field-programmable gate array

IHM: Interface (interaction) homme machine

HDMI: High Definition Multimedia Interface

WIFI: Wireless

LXDE: Lightweight X11 Desktop Environment

CPU: Central processing unit

RAM: Random Access Memory (mémoire vive)

LCD: liquid crystal display

GPIO: General Purpose Input/Output

DVI: Digital visual interface

VGA: Video Graphics Array

MDI: **M**edium dependent interface

LX: réception data

TX: émission data

HTML: Hyper Text Markup Language

Chapitre 3

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.

Chapitre 4

CMOS : Complementary Metal Oxide Semiconductor,

CCD : Un capteur photographique

CIR : Centre Instantané de Rotation

VAO : La vision artificielle ou vision assistée par ordinateur

HSV : Hue-Saturation-Valeur

Introduction générale

L'homme a toujours souhaité se libérer du travail dans ce qu'il y a d'astreignant, de fatiguant, d'inintéressant. Les progrès qu'ont connu les technologies depuis un demis siècle ont répondu en bonne partie à son désir par l'intermédiaire de deux outils générique: l'ordinateur pour les travaux mentaux et le robot pour les travaux physiques.

Malgré son aspect récent, la robotique tire ses origines des civilisations les plus antiques. Il est peut être même vrai que toutes les autres sciences ont existé juste pour permettre à la robotique de se développer afin d'aider l'Homme à créer cet« esclave » qui appliquera ses ordres et qui le libérera à jamais du travail.

De l'Homme préhistorique qui s'empressa d'inventer des extensions technologiques de ses membres afin de faciliter son travail, comme la massue par exemple, jusqu'au robot mobile autonome le plus performant, la robotique mobile n'a cessé de connaître des progrès spectaculaires. En effet, les générations de robots mobiles se sont vite succédées offrant à chaque fois plus de performances en matière de perception, d'autonomie et de pouvoir décisionnel. Ces progrès étaient un résultat inévitable des développements qu'ont connu la mécanique, la microélectronique et l'informatique. C'est pour cela qu'en très peu de temps, les générations de robots se succèdent pour voir naître une branche de la robotique visant de plus amples horizons : c'est l'ère de la robotique mobile.

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 [1]. 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.

L'objectif principal de notre travail est de réaliser un asservissement visuel d'un robot mobile en utilisant une carte électronique Arduino, un Raspberry et un simple module caméra. La tache principale de notre robot est de détecter une cible de couleur spécifique et de la suivre en temps réel

Le présent travail est composé et organisé en quatre chapitres comme suit :

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 la carte raspberry et la carte Arduino.

Le troisième chapitre est une introduction aux techniques d'asservissement visuel. Nous décrirons les déférentes architectures qui existent en asservissement visuel et nous présenterons une classification de ces systèmes de commande.

Le quatrième chapitre concernera le détaille de notre réalisation. Nous présenterons l'agencement et le fonctionnement de l'architecture matérielle que nous avons mise en œuvre. Par ailleurs, nous validerons, à partir d'une application réelle de suivi de cible, notre algorithme de commande basé sur l'exploitation de données visuelles.

Enfin, la conclusion nous permettra de résumer l'ensemble des travaux réalisés. Nous terminerons ce mémoire par quelques perspectives envisageables dans un futur proche ou plus lointain.

Chapitre I :

Introduction à la robotique mobile

I.1. Introduction

La robotique a atteint son plus grand succès jusqu'à ce jour dans le monde de la fabrication industrielle.

Boulonné à son épaule à une position spécifique dans la ligne d'assemblage, le bras de robot peut se déplacer avec une grande rapidité et précision pour effectuer des tâches répétitives telles que le soudage par points et la peinture. Dans l'industrie de l'électronique, les manipulateurs place des composants montés en surface avec une précision surhumaine, ce qui rend le téléphone portable et ordinateur portable possible.

Cependant, pour tous leurs succès, ces robots commerciaux souffrent d'un inconvénient majeur: le manque de mobilité. Un manipulateur fixe dispose d'une gamme limitée de mouvement qui dépend de l'endroit où elle est boulonnée. En revanche, un robot mobile serait en mesure de voyager à travers l'usine de fabrication, en appliquant avec souplesse ses talents partout où il est le plus efficace.

De manière générale, on regroupe sous l'appellation robots mobiles l'ensemble des robots à base mobile, par opposition notamment aux robots manipulateurs. L'usage veut néanmoins que l'on désigne le plus souvent par ce terme les robots mobiles à roues. Les autres robots mobiles sont en effet le plus souvent désignés par leur type de locomotion, qu'ils soient marcheurs, sous-marins ou aériens.

Le but de ce chapitre est de situé le robot mobile objet de notre étude. A partir des descriptions des différents éléments constituants les robots mobiles ainsi que les différentes structures mécaniques qui existent, nous justifierons les solutions que nous avons mises en œuvre pour réaliser l'asservissement visuel du robot mobile Arduino.

I.2. Aperçu historique

C'est 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 Leob.

Les machines de Russell (1913) et de Stephens (1929), les tortues cybernétiques de Grey Walter (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. [2]



Figure I.1 *La tortue de Grey Walter (nommée "machina speculatrix" et surnommée Elsie)*

Concernent la Tortue construite par Grey Walter dans les années 1952 (Figure I.1), Grey Walter n'utilise que quelques composants analogiques, dont des tubes à vide, mais son robot est capable de se diriger vers une lumière qui marque un but, de s'arrêter face à des obstacles et de recharger ses batteries lorsqu'il arrive dans sa niche.

Dans les années 60, les recherches en électronique vont conduire, avec l'apparition du transistor, à des robots plus complexes mais qui vont réaliser des tâches similaires. Ainsi le robot "*Beast*" (Figure I.2) de l'université John Hopkins est capable de se déplacer au centre des couloirs en utilisant des capteurs ultrason, de chercher des prises électriques (noires sur des murs blancs) en utilisant des photodiodes et de s'y recharger.

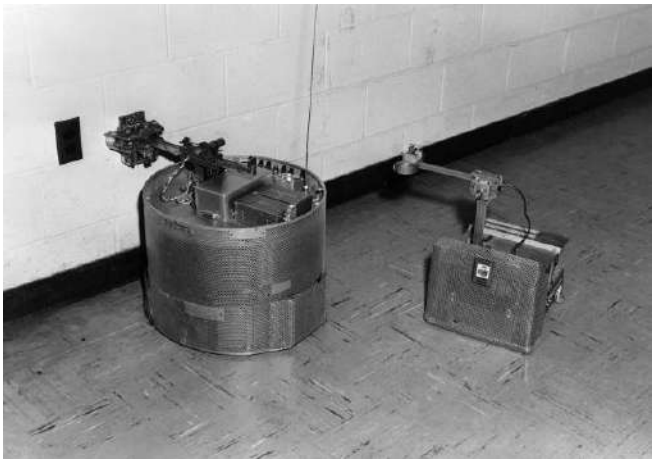


Figure I.2 *Robot "Beast" de l'université John Hopkins dans les années 1960.*

Les premiers liens entre la recherche en intelligence artificielle et la robotique apparaissent à Stanford en 1969 avec « *Shakey* » (Figure I.3). Ce robot utilise des télémètres à ultrason et une caméra et sert de plate-forme pour la recherche en intelligence artificielle, qui à l'époque travaille essentiellement sur des approches symboliques de la planification. La perception de l'environnement, qui à l'époque est considérée comme un problème séparé, voire secondaire, se révèle particulièrement complexe et conduit là aussi à de fortes contraintes sur l'environnement. Ces développements de poursuivent avec le « *StanfordCart* » dans la fin des années 1970, avec notamment les premières utilisations de la stéréo-vision pour la détection d'obstacles et la modélisation de l'environnement.

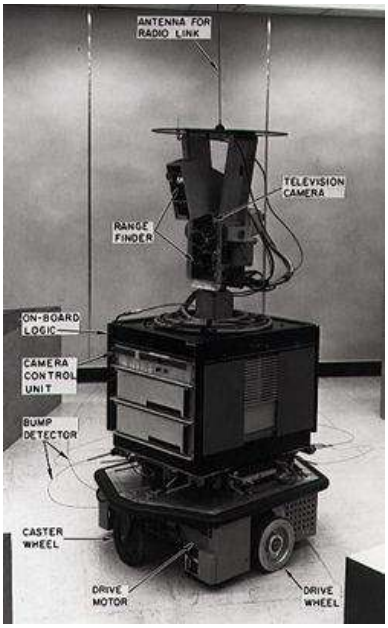


Figure I.3 *Le robot Shakey de Stanford en 1969 a été une plate-forme de démonstration des recherches en intelligence artificielle.*



Figure I.4 *Genghis, développé par Rodney Brooks au MIT au début des années 1990.*

Une étape importante est à signaler au début des années 1990 avec la mise en avant de la robotique réactive, représentée notamment par Rodney Brooks. Cette nouvelle approche de la robotique, qui met la perception au centre de la problématique, a permis de passer de gros robots très lents à de petits robots (Figure I.4), beaucoup plus réactifs et adaptés à leur environnement. Ces robots n'utilisent pas ou peu de modélisation du monde, problématique qui s'est avérée être extrêmement complexe.

Ces développements ont continué et l'arrivée sur le marché depuis les années 1990 de plates-formes intégrées telles que le « *pioneer* » de la société Mobile Robots a permis à de très nombreux laboratoires de travailler sur la robotique mobile et à conduit à une explosion de la diversité des thèmes de recherche. Ainsi, même si les problèmes de déplacement dans l'espace et de modélisation de l'environnement restent difficiles et cruciaux, des laboratoires ont pu par exemple travailler sur des approches multi-robot, la problématique de l'apprentissage ou sur les problèmes d'interactions entre les hommes et les robots.

I.3. Exemples d'application

Aujourd'hui, le marché commercial de la robotique mobile est toujours relativement restreint en dehors des robots aspirateurs vendus à plusieurs millions d'exemplaires. Cependant, il existe de nombreuses perspectives de développement qui en feront probablement un domaine important dans le futur. Les applications des robots peuvent se trouver dans de nombreuses activités, le tableau ci-après résume de manière non exhaustive les diverses applications des robots mobiles.

Tableau I.1 *Applications des robots mobiles*

Industrie nucléaire	-surveillance de sites - manipulation de matériaux radio-actifs - démantèlement de centrales
Sécurité civile	- neutralisation d'activité terroriste - déminage - pose d'explosif - surveillance de munitions
Militaire	- surveillance, patrouille - pose d'explosifs - manipulation de munitions
Chimique	- surveillance de site - manipulation de matériaux toxiques
Médecine	- assistance d'urgence - aide aux handicapés physiques, aux aveugles
Lutte contre l'incendie	- localisation d'une source d'incendie - détection de fumée - suppression de flammes
Sous-marine	-pose de câbles - recherche de nodules - recherche de navires immergés - inspection des fonds marins
Agricole	-cueillette de fruits - traite, moisson, traitement des vignes…
Construction BTP	- projection mortier - lissage du béton
Nettoyage	- coque de navire - nettoyage industriel
Espace	- exploration
Industriel	- convoyage - surveillance

I.4. 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.

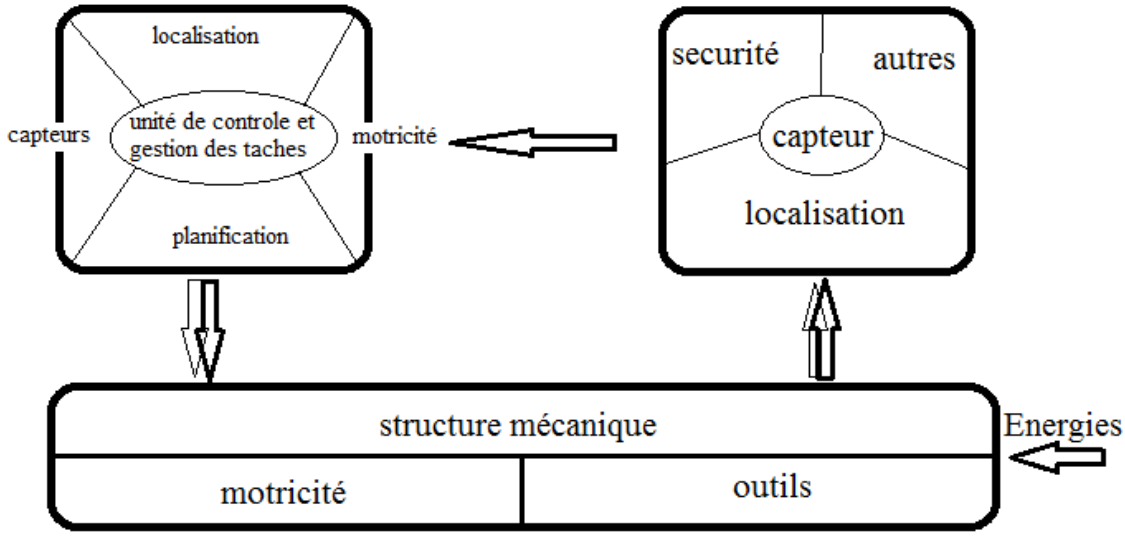


Figure I.5 *Architecture d'un robot mobile.*

I.4.1. La structure mécanique et la motricité

On peut dénombrer quatre types de structures mécaniques assurant la motricité.

I.4.1.1. Les mobiles à roues

La mobilité par roues est la structure mécanique la plus communément appliquée. Cette technique assure selon l'agencement et les dimensions des roues un déplacement dans toutes les directions avec une accélération et une vitesse importante.

I.4.1.2. Les mobiles à chenilles

L'utilisation des chenilles présente l'avantage d'une bonne adhérence au sol et d'une faculté de franchissement d'obstacles. L'utilisation est orientée vers l'emploi sur sol accidenté ou de mauvaise qualité au niveau de l'adhérence (présence de boue, herbe…).

I.4.1.3. Les mobiles marcheurs

Les robots mobiles marcheurs sont destinés à réaliser des tâches variées dont l'accès au site est difficile, dangereux ou impossible à l'homme. Leur anatomie à nombreux degrés de liberté permet un rapprochement avec les robots manipulateurs. La locomotion est commandée en termes de coordonnées articulaires. Les méthodes de commande des articulations définissent le concept d'allure qui assure le déplacement stable de l'ensemble. Les différentes techniques étudiées se rapprochent de la marche des animaux et notamment de celle des insectes.

I.4.1.4. Les robots rampants

La reptation est une solution de locomotion pour un environnement de type tunnel qui conduit à réaliser des structures filiformes.

Le système est composé d'un ensemble de modules ayant chacun plusieurs mobilités. Les techniques utilisées découlent des méthodes de locomotion des animaux.

I.4.1.5. La motricité et l'énergie

Les déplacements des robots sont réalisés par des moteurs de types électrique, thermique ou hydraulique.

L'énergie électrique la plus fréquemment employée offre l'avantage d'une commande aisée. Par contre le transport et la génération présentent des difficultés. Plusieurs méthodes sont employées :

- Par batteries qui sont soit rechargées périodiquement de manière automatique ou manuelle, soit par un échange avec d'autres lorsqu'elles sont déchargées.
- Par groupe électrogène embarqué dont l'inconvénient constitue la masse élevée. L'énergie de base est alors thermique.
- Par cordon ombilical qui réduit l'autonomie du robot.

I.4.2. Les organes de sécurité

Un robot, selon la tâche qui lui est confiée, peut être amené à travailler au voisinage du personnel. A ce titre, il est obligatoire qu'il soit doté d'organes garantissant la sécurité. Des capteurs sont disponibles tout autour du robot mobile afin de détecter un obstacle sur un domaine le plus étendu possible. Deux types de capteurs sont employés : les capteurs proximétriques assurant la détection avant collision (ultra-son, hyper fréquence, infrarouge…) et les capteurs de contact

détectant une collision ou un choc avec l'environnement (contact électrique sur pare-chocs, résistance variable, fibre optique…). L'organisation de la sécurité est représentée sur le schéma de la figure I.6.

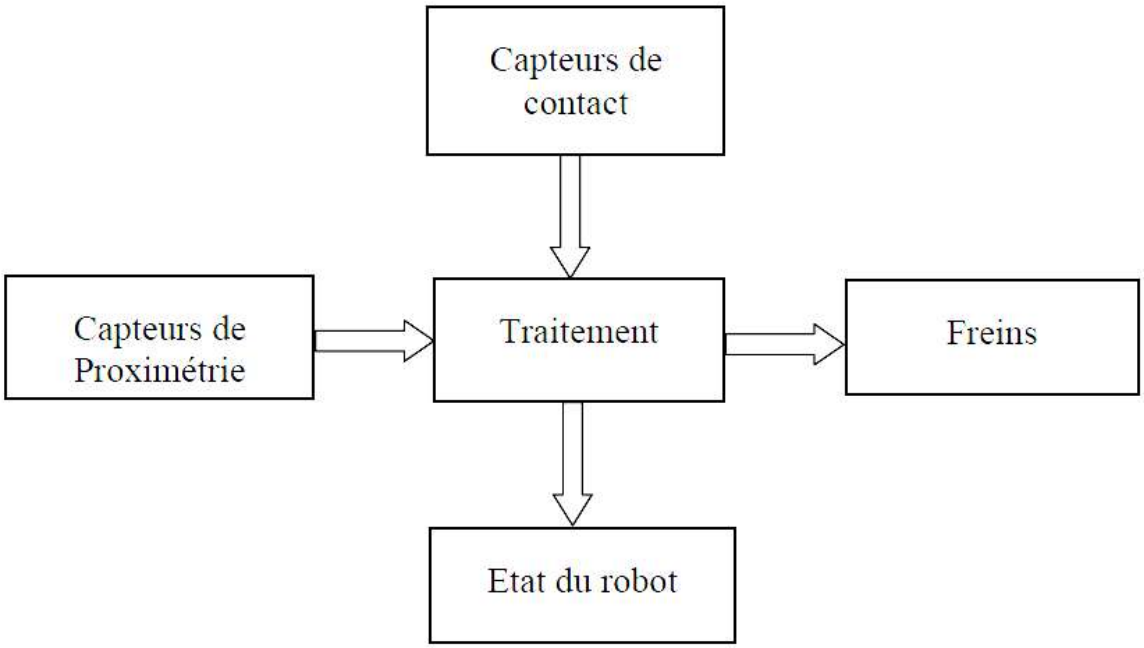


Figure I.6 *Synoptique de la sécurité.*

Le traitement de la détection s'effectue selon plusieurs cas. Si le capteur à contact est sollicité, le robot s'immobilise soit définitivement soit tant que le contact persiste, ou il effectue un mouvement opposé au contact. Par contre si un proximètre détecte une présence, la stratégie consiste soit à immobiliser le robot en attendant que la personne s'éloigne, soit à ralentir le mouvement si la personne n'est pas trop proche, soit à choisir un autre chemin qui l'éloigne de la personne.

I.4.3. Traitement des informations et gestion des taches

L'ensemble de traitement des informations et gestion des tâches constitue le module information central qui établit les commandes permettant au mobile de réaliser un déplacement et d'activer les divers organes en accord avec l'objectif. D'ailleurs, c'est cette partie du robot mobile qui fait l'objet de notre travail. Nous allons élaborer et mettre en œuvre un système de gestion et de traitement de l'information basé autour de deux cartes électroniques : la carte « ***Rapberry*** » et la carte « ***Arduino*** ». Cet ensemble embarqué fournira à notre robot mobile une puissance de calcule élevée et une autonomie importante dans la réalisation de tâches complexes.

I.5. Les robots mobiles à roues

Les roues sont le moyen de locomotion le plus répandu en matière de robotique mobile [3]. En fait, les robots mobiles à roues sont faciles à réaliser et présentent de grandes possibilités de déplacement et de manœuvrabilité avec une vitesse et une accélération importantes. Bien évidemment, pour un ensemble donné de roues, toute disposition ne conduit pas à une solution viable. Un mauvais choix peut limiter la mobilité du robot ou occasionner d'éventuels blocages. Par exemple, un robot équipé de deux roues fixes non parallèles ne pourrait pas aller en ligne droite ! Pour qu'une disposition de roues soit viable et n'entraîne pas de glissement des roues sur le sol, il faut qu'il existe pour toutes ces roues un unique point de vitesse nulle autour duquel tourne le robot de façon instantanée. Ce point, lorsqu'il existe, est appelé centre instantané de rotation (CIR). Les points de vitesse nulle liés aux roues se trouvent sur leurs axes de rotation, il est donc nécessaire que le point d'intersection des axes de rotation des différentes roues soit unique. Pour cette raison, il existe en pratique trois principales catégories de robots mobiles à roues, que l'on va présenter par la suite. En général, les robots mobiles à roues ont des structures à locomotion différentielle. La figure I.7 montre une structure type de robot mobile à commande différentielle où le déplacement est assuré par le biais de deux moteurs d'entraînement indépendants, chacun d'eux est lié à une roue motrice.

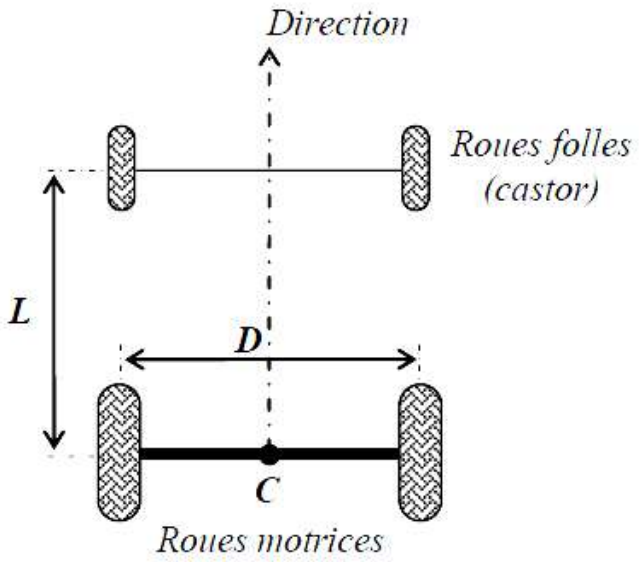


Figure I.7 *Robot mobile à commande différentielle*

En ce qui concerne la commande des moteurs, cette indépendance permet au robot mobile de changer de direction en jouant uniquement sur les vitesses des deux roues motrices. Les roues folles sont libres, leur rôle essentiel est d'assurer la stabilité de la plate-forme du robot.

I.5.1. L'unicycle

On désigne par unicycle un robot actionné par deux roues indépendantes et possédant éventuellement un certain nombre de roues folles assurant sa stabilité. Le schéma d'un robot de type unicycle est donné à la figure 1.8 On y a omis les roues folles, qui n'interviennent pas dans la cinématique, dans la mesure où elles ont été judicieusement placées. Ce type de robot est très répandu en raison de sa simplicité de construction et de ses propriétés cinématiques intéressantes.

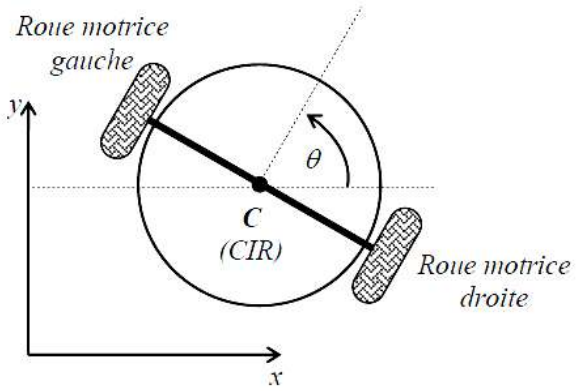


Figure I.8 *Robot mobile de type unicycle*

I.5.2. Le tricycle

L'architecture d'un robot mobile de type tricycle est donnée sur la figure I.9, il utilise une roue motrice avant et deus roues passives arrières (ou vice versa). Le mouvement est conféré au robot par deux actions : la vitesse longitudinale et l'orientation de la roue orientable. Ce type de robots est très connu dans les applications des AGVs pour leur simplicité inhérente. Pour la commande et la localisation ce type de robot : un capteur d'orientation est associé à la roue orientable, et deux encodeurs sont associés aux roues motrices.

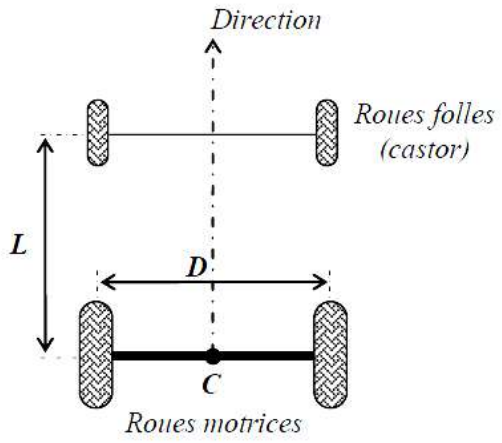


Figure I.9 *Robot mobile de type tricycle*

I.5.3. Les véhicules (structure d'ackerman)

Le cas du robot de type voiture est très similaire à celui du tricycle. La différence se situe au niveau du train avant, qui comporte deux roues au lieu d'une. Cela va de soit, on rencontre beaucoup plus souvent ce type de systèmes. On parle de robot dès lors que la voiture considérée est autonome [4], donc sans chauffeur ni télépilotage. Il s'agit là d'un des grands défis issus de la robotique mobile. La figure I.10 représente ce type d'architecture.

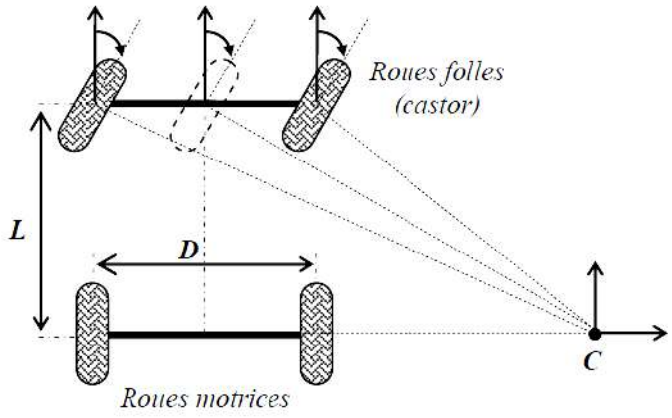


Figure I.10 *Robot mobile de type voiture*

I.5.4. Robot à traction synchrone

La traction synchrone est une technique utilisée pour minimiser l'effet de glissement et augmenter la force de traction.

On rencontre ce type de robot dans l'industrie automobile, et dans les robots tous terrains. La configuration du robot à traction synchrone est similaire à un robot à trois roues couplées de façon qu'elles soient actionnées en même temps. La figure I.11 montre un robot à quatre roues couplées avec des chaînes.

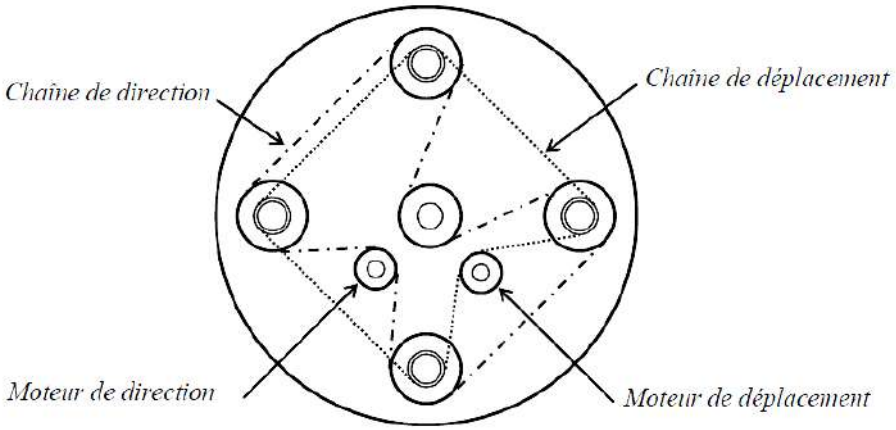


Figure I.11 *Robot mobile à traction synchrone.*

I.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, à l'image de la perception chez les êtres vivants, on parle de 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 appelés capteurs extéroceptifs. Il s'agit de capteurs donnant la distance du robot à l'environnement, la température, signalant la mise en contact du robot avec l'environnement, etc.

L'étude détaillée des capteurs, qui relève à 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 présentés. On tachera simplement de garder à l'esprit que les défauts inhérents aux différents systèmes de mesure utilisés (bruit, erreurs ou échecs de mesures, difficulté de modélisation) influent fortement sur la perception que le robot a de l'environnement.

I.7. Mesure de la rotation des roues

Chaque roue motrice d'un robot mobile est généralement associée à un servomoteur. Celui-ci est équipé d'un dispositif de mesure de rotation à l'aide d'un capteur disposé sur l'axe lui-même. Il s'agit d'un capteur proprioceptif, car il renseigne sur la position, c'est-à-dire sur l'état interne du système.

Les capteurs utilisés en robotique mobile ne se différencient pas de ceux rencontrés en commande d'axe de manière plus générale. On notera cependant qu'il n'est pas utile de disposer de codeur absolu les roues d'un robot mobile, la position initiale du robot ne pouvant être donnée simplement par une valeur codeur : il suffit pour s'en persuader de soulever le robot et de le poser ailleurs.

On notera ensuite que dans le cas des robots mobiles la mesure utile est généralement la vitesse de rotation et non la position angulaire. Cette mesure de vitesse mesure peut être obtenue par une mesure directe, à l'aide d'une génératrice tachymétrique notamment. Cependant, pour des raisons évidentes (moindre cout, moindre encombrement, moindre entretien), elle est le plus souvent réalisée de manière indirecte. La technologie la plus classique consiste à utiliser des codeurs

optiques, délivrant des impulsions carrées lorsque l'arbre tourne. On peut estimer que le comptage de ces impulsions suffit à déterminer la vitesse de rotation des roues, qui découle de la fréquence des impulsions. Cette mesure est nécessairement entachée d'un bruit de quantification, qui peut être important, notamment aux basses vitesses. Pour réduire ce bruit, on choisit des codeurs de meilleure résolution, possédant un nombre élevé de points par tour (PPT), c'est-à-dire mesurant des incréments angulaires très petits (typiquement 2000 PPT). Une alternative est d'utiliser des *synchrorésolveurs*, qui utilisent des signaux sinusoïdaux et dont la précision est bien meilleure. La technologie des codeurs incrémentaux et des *synchrorésolveurs* a été évoquée dans le cours de robotique générale [5]. Une plus vaste gamme de capteurs (potentiomètres, capteurs inductifs, capacitifs, magnétiques) est par ailleurs à la disposition du roboticien aventureux.

I.8. Mesure d'orientation et de position

I.8.1. Mesure de la position : le GPS

Il existe très peu de systèmes donnant la position absolue d'un point dans un repère fixe donné. Le GPS (Global Positioning System), initialement développé pour les applications militaires américaines [6] est actuellement à la disposition du grand public. On peut cependant considérer que son utilisation dans ce cadre n'est pas garantie. La mise en place du projet civil européen Galileo [7] devrait offrir une alternative au GPS.

Le GPS fonctionne avec un ensemble de satellites, qui effectuent des émissions synchronisées dans le temps. Par recoupement des instants d'arrivée des signaux et de la position des satellites émetteurs, les récepteurs peuvent calculer leur position. Le principe de calcul de la position est basé sur une triangulation, à l'aide de quatre signaux reçus simultanément (le quatrième signal assure la robustesse de la mesure).

En termes de précision, la localisation ainsi obtenue est entachée d'une erreur de l'ordre de la quinzaine de mètres, ce qui n'est bien évidemment pas suffisant pour permettre à un robot de naviguer de manière robuste. Ainsi, on a systématiquement recours à une méthode différentielle pour obtenir des résultats plus satisfaisants.

La localisation se fait à l'aide de deux récepteurs, dont l'un est statique et positionné avec précision dans l'environnement. On peut alors obtenir une précision

de l'ordre du centimètre. Pour capter les signaux mis par des satellites, ce type de dispositif est donc plutôt destiné au positionnement en extérieur. Par ailleurs sa faible précision et son prix élevé le réservent plutôt à des systèmes multi robots, associant par exemple des robots mobiles à des engins volants et sous-marins, comme cela est à l'étude dans les applications militaires les plus récentes. Enfin, ce système de mesure, autorisant des rafraîchissements à des fréquences de l'ordre de seulement 5 Hz, n'est pas utilisable pour une commande en temps-réel. Il est donc principalement utile pour des recalages ponctuels.

I.8.2.Mesure de l'orientation : Gyromètres

Les gyromètres sont des capteurs proprioceptifs qui permettent de mesurer l'orientation du corps sur lequel ils sont placés, ceci par rapport à un référentiel fixe et selon un ou deux axes. Montés sur un robot mobile plan, un gyromètre à un axe permet donc de mesurer son orientation. Il existe plusieurs sortes de gyromètres : mécaniques et optiques pour les plus connus, mais aussi à structures vibrantes, capacitifs, etc.

Le gyroscope mécanique, imaginé par L. Foucault pour mettre en évidence le mouvement de rotation de la Terre (voir figure I.12), repose sur le principe suivant [8] : lorsqu'un corps, qui n'est soumis à aucune force extérieure, est animé d'un mouvement de rotation autour d'un de ses axes principaux d'inertie, cet axe doit rester parallèle à lui-même si l'on vient à déplacer le corps d'une manière quelconque, tout en laissant l'axe libre de prendre toutes les directions. L'image classique d'un gyroscope mécanique est un rotor suspendu par un cardan sans friction isolant le centre du rotor des couples extérieurs, comme le montre la figure I.12. Le gyroscope, selon le principe de conservation du moment angulaire, montre une grande stabilité vis-à-vis des déplacements de la structure extérieure. Les gyromètres mécaniques sont des capteurs exploitant l'effet gyroscopique.

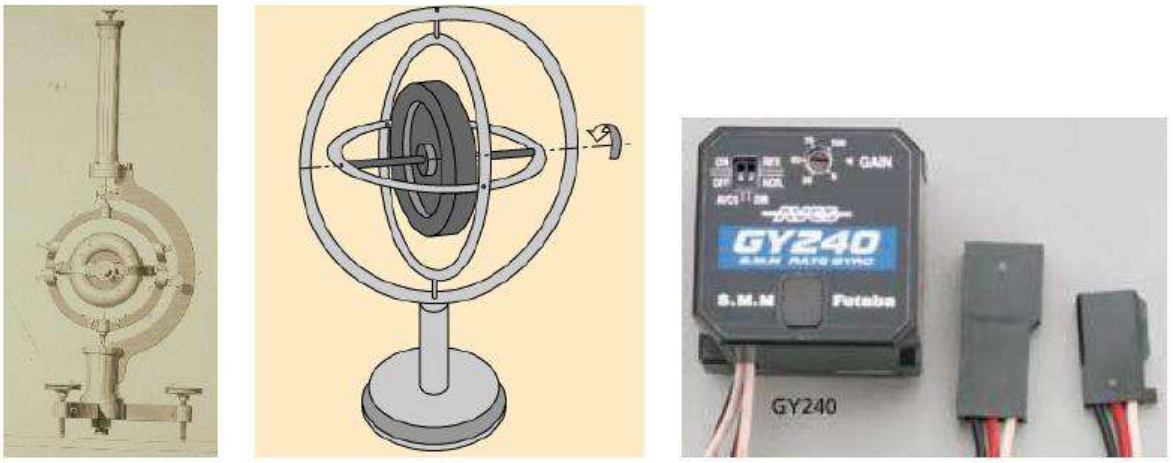


Figure I.12 *Gyroscope mécanique de Foucault et gyromètre Futaba à structure vibrante*

Les gyromètres optiques exploitent le fait que la vitesse de la lumière reste inchangée dans tout référentiel. Deux faisceaux lasers sont émis depuis une même source, pour parcourir des chemins identiques, l'un dans le sens des aiguilles d'une montre, l'autre en sens opposé. Lors de la mise en rotation du gyromètre il existe une différence de marche des deux rayons et des interférences apparaissent. On peut alors déduire la vitesse de rotation du système de cette mesure. L'utilisation de fibres optiques a conféré à cette technologie des performances qui dépassent de très loin la qualité de mesure requise en robotique mobile. Leur coût élevé les destine néanmoins presque exclusivement aux applications aéronautiques et spatiales.

Les gyromètres apparus récemment dans une gamme plus grand public sont à structure vibrante, en technologie SMM (silicon micro machines). Leur apparition coïncide notamment avec la création du système de stabilisation d'assiette EPS dans le secteur automobile, qui a participé à la démocratisation de ce type de capteur. Un gyromètre très bon marché, comme le Futaba GY240, gyromètre à un axe pour le modélisme, il modifie la largeur des impulsions qu'on lui applique en entrée pour donner une information de vitesse de rotation autour de son axe.

I.8.3. Autres capteurs (moins fréquents)

Inclinomètres

Les inclinomètres sont des capteurs mesurant des inclinaisons par rapport à la gravité terrestre. Il existe plusieurs technologies qui utilisent, soit des masses inertielles, dont la position est mesurée par un détecteur optique, soit des

technologies capacitives permettant d'obtenir des capteurs de faible encombrement et d'un excellent rapport performance/coût.

Accéléromètres

Il existe une grande diversité de méthodes pour obtenir l'accélération d'un système en mouvement. Les principales techniques utilisées sont mécaniques (jauges de contraintes), électromécaniques (effets piézoélectrique), électriques (effet capacitif ou réluctance variable) ou encore optique. Les applications de robotique mobile exploitent souvent peu la pleine dynamique des véhicules. Ainsi, ces capteurs haut de gamme demeurent peu répandus en robotique mobile et restent plutôt destinés à la navigation précise de systèmes rapides, essentiellement dans le domaine aéronautique et spatial.

I.8.4. Mesure de proximité et de distance

On appelle télémétrie toute technique de mesure de distance par des procédés acoustiques, optiques ou radio électriques. L'appareil permettant de mesurer les distances est lui appelé télémètre. De même qu'il existe différentes techniques de mesure de distance (mesure du temps de vol d'une onde, triangulation), il existe différentes technologies pour réaliser des télémètres. Nous présentons ici le plus répandus en robotique mobile, à savoir le capteur infrarouge.

Les capteurs infrarouges sont constitués d'un ensemble émetteur/récepteur fonctionnant avec des radiations non visibles, dont la longueur d'onde est juste inférieure à celle du rouge visible. La mesure des radiations infrarouges étant limitée et, en tout état de cause, la qualité très dégradée au delà d'un mètre, ces dispositifs ne servent que rarement de télémètres. On les rencontrera le plus souvent comme détecteurs de proximité, ou dans un mode encore plus dégradé de présence. Il faut noter que ce type de détection est sensible aux conditions extérieures, notamment à la lumière ambiante, à la spécularité des surfaces sur lesquelles se réfléchissent les infrarouges, à la température et même à la pression ambiante. Ces capteurs ne sont pas complètement directionnels et leur caractéristique (à l'image des capteurs ultrasons présentés par la suite) présente une zone de détection conique à l'origine d'incertitudes. Enfin, l'alternance de phases d'émission et de réception impose une distance de détection minimale.

I.9. La navigation

Les sections précédentes ont permis de mettre en place les outils nécessaires pour faire naviguer un robot mobile dans un environnement d'intérieur : la compréhension du mode de locomotion et de localisation de ce robot dans son environnement. Il s'agit maintenant d'utiliser au mieux la motricité du robot et sa localisation pour accomplir la tâche de navigation de manière autonome.

Un mouvement est une application définie en fonction du temps t, reliant un point initial à l'instant t_0 à un point final à l'instant t_f Une trajectoire est le support d'un mouvement. Il s'agit donc d'une courbe paramétrée par une variable s quelconque, par exemple l'abscisse curviligne normalisée ($s \in [0,1]$) de la courbe sur laquelle se déplace le robot. L'évolution du paramètre s en fonction du temps t est appelée mouvement sur la trajectoire.

Le problème de navigation d'un robot mobile consiste de la manière la plus générale à trouver un mouvement dans l'espace des configurations sans collisions, traditionnellement noté C_{free}. Ce mouvement amène le robot d'une configuration initiale $q_0 = q\ (t_0)$ à une configuration finale $q_f = q\ (t_f)$. On peut néanmoins donner des définitions différentes de la tâche de navigation à accomplir, selon le but recherché par exemple on peut souhaiter seulement placer le robot dans une zone donnée et relâcher la contrainte d'orientation, etc.

La tâche de navigation ainsi définie est donc limitée à un seul mouvement. Il existe néanmoins une très grande variété de travaux et de méthodes permettant d'aborder ce problème difficile. Pour différencier les techniques de navigation, on peut de manière générale distinguer deux approches :

- la première consiste à planifier le mouvement dans l'espace des configurations et à l'exécuter par asservissement du robot sur le mouvement de consigne (schéma planification exécution) ;
- la seconde consiste à offrir un ensemble de primitives plus réactives. Elles correspondent alors à des sous tâches (suivre un mur, éviter un obstacle) dont on estime que l'enchaînement est du ressort d'un planificateur de tâches ayant décomposé la tâche globale.

I.10. 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 nous avons choisie d'utiliser pour gérer et commander le robot mobile Arduino.

Chapitre II :

Etude des systèmes embarqués utilisées

II.1. Introduction

Dans ce chapitre nous allons introduire et présenter les deux cartes électroniques embarquées 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 développé s'articule essentiellement autour de la carte ***Arduino*** et la carte ***Raspberry Pi***. Le Raspberry Pi sera 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 les cartes Arduino de manière générale, puis le robot mobile Arduino et enfin l'Arduino Mega AT2560 que nous avons utilisé.

Dans un second temps, nous décrirons l'architecture et le fonctionnement du mini-PC Raspberry Pi.

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

II.2. Arduino

II.2.1. Définition

Arduino est un projet créé par une équipe de développeurs. Cette équipe a créé le "système Arduino". C'est un outil qui va permettre aux débutants, amateurs ou professionnels de créer des systèmes électroniques plus ou moins complexes.

C'est une plate-forme de prototypage d'objets interactifs à usage créatif constituée d'une carte électronique et d'un environnement de programmation.

Sans tout connaître ni tout comprendre de l'électronique, cet environnement matériel et logiciel permet à l'utilisateur de formuler ses projets par l'expérimentation directe avec l'aide de nombreuses ressources disponibles en ligne.

Pont tendu entre le monde réel et le monde numérique, Arduino permet d'étendre les capacités de relations humain/machine ou environnement/machine.

Ce dernier est un projet en source ouverte : la communauté importante d'utilisateurs et de concepteurs permet à chacun de trouver les réponses à ses questions.

II.2.2. Applications

Le système Arduino nous permet de réaliser un grand nombre de choses, qui ont une application dans tous les domaines, l'étendue de l'utilisation de l'Arduino est gigantesque. Pour vous donner quelques exemples, vous pouvez :

- contrôler les appareils domestiques,
- fabriquer votre propre robot,
- faire un jeu de lumières,
- communiquer avec l'ordinateur,
- télécommander un appareil mobile (modélisme),
- etc…

Avec Arduino, nous allons faire des systèmes électroniques tels qu'une bougie électronique, une calculatrice simplifiée, un synthétiseur, etc. Tous ces systèmes seront conçus avec pour base une carte Arduino et un panel assez large de composants électroniques.

II.2.3. Pourquoi avons nous choisi le système carte 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 voir pourquoi choisir l'Arduino.

- Multiplateforme : le logiciel Arduino, écrit en Java, tourne sous les systèmes d'exploitation Windows, Macintosh et Linux. La plupart des systèmes à microcontrôleurs sont limités à Windows.
- Un environnement de programmation clair et simple : l'environnement de programmation Arduino (= le logiciel Arduino) est facile à utiliser pour les débutants, tout en étant assez flexible pour que les utilisateurs avancés puisse en tirer profit également.

- Logiciel Open Source et extensible : le logiciel Arduino et le langage Arduino sont publiés sous licence open source, disponible pour être complété par des programmateurs expérimentés.
- Matériel Open source et extensible : les cartes Arduino sont basées sur les microcontrôleurs Atmel ATMEGA8, ATMEGA168, ATMEGA 328, ATMEGA 2560 etc... Les schémas des modules sont publiés sous une licence Creative Commons, et les concepteurs de circuits expérimentés peuvent réaliser leur propre version des cartes Arduino, en les complétant et en les améliorant .Même les utilisateurs relativement inexpérimentés peuvent fabriquer la version sur plaque d'essai de la carte Arduino, dans le but de comprendre comment elle fonctionne et pour économiser de l'argent.
- La communauté: La communauté Arduino est impressionnante et le nombre de ressources à son sujet est en constante évolution sur internet. De plus, on trouve les références du langage Arduino ainsi qu'une page complète de tutoriels sur le site arduino.cc (en anglais) et arduino.cc (en français).

II.2.4. Les outils Arduino

À présent, rapprochons-nous de « l'utilisation » du système Arduino et voyons comment il se présente. Il est composé de deux choses principales, qui sont : le matériel et le logiciel. Ces deux outils réunis, il nous sera possible de faire n'importe quelle réalisation.

II.2.4.1. Le matériel

La carte Arduino repose sur un circuit intégré (un mini ordinateur appelé également microcontrôleur) associée à des entrées et sorties qui permettent à l'utilisateur de brancher différents types d'éléments externes :

- Côté entrées, des *capteurs* qui collectent des informations sur leur environnement comme la variation de température via une sonde thermique, le mouvement via un détecteur de présence ou un accéléromètre, le contact via un bouton-poussoir, etc.
- Côté sorties, des *actionneurs* qui agissent sur le monde physique telle une petite lampe qui produit de la lumière, un moteur qui actionne un bras articulé, etc.

Comme le logiciel Arduino, le circuit électronique de cette plaquette est libre et ses plans sont disponibles sur internet. On peut donc les étudier et créer des dérivés. Plusieurs constructeurs proposent ainsi différents modèles de circuits électroniques programmables et utilisables avec le logiciel Arduino.

II.2.4.2. Constitution de la carte Arduino

Dans ce qui va suivre, nous détaillerons les points importants de ce système et surtout leurs utilités.

Le microcontrôleur

Le microcontrôleur le cerveau de notre carte. C'est lui qui va recevoir le programme que vous aurez créé et qui va le stocker dans sa mémoire puis l'exécuter. Grâce à ce programme, il va savoir faire des choses, qui peuvent être : faire clignoter une LED, afficher des caractères sur un écran, envoyer des données à un ordinateur, connexion entre deux périphériques a l'aide d'un bus U2C,…

L'alimentation

Afin de faire la mise en marche de la carte, elle a besoin d'une alimentation. Le microcontrôleur fonctionne sous 5V, la carte peut être alimentée en 5V par le port USB ou bien par une alimentation externe qui est comprise entre 7V et 12V. Cette tension doit être continue et peut par exemple être fournie par une pile 9V. Un régulateur se charge ensuite de réduire la tension à 5V pour le bon fonctionnement de la carte. Donc pas de danger de tout griller. Veuillez seulement à respecter l'intervalle de 7V à 15V (même si le régulateur peut supporter plus, pas la peine de le retrancher dans ses limites)

La visualisation

Les trois LED qu'on trouve sur la carte sont de la taille de l'ordre du millimètre. Ces LED servent 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.

II.2.5. La connectique

La carte Arduino ne possédant pas de composants qui peuvent être utilisés pour un programme, mis a par la LED connectée à la broche 13 du microcontrôleur, il est nécessaire de les rajouter. Mais pour ce faire, il faut les connecter à la carte. C'est là qu'intervient la connectique de la carte. Par exemple, on veut connecter une LED sur une sortie du microcontrôleur. Il suffit juste le la connecter, avec une résistance en série, à la carte, sur les fiches de connections de la carte. Cette connectique est importante et a un brochage qu'il faudra respecter. Nous le verrons quand nous apprendrons à faire notre premier programme. C'est avec cette connectique que la carte est « extensible », car l'on peut y brancher tous types de montages et modules.

II.2.6. Le Logiciel de gestion de la carte Arduino

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

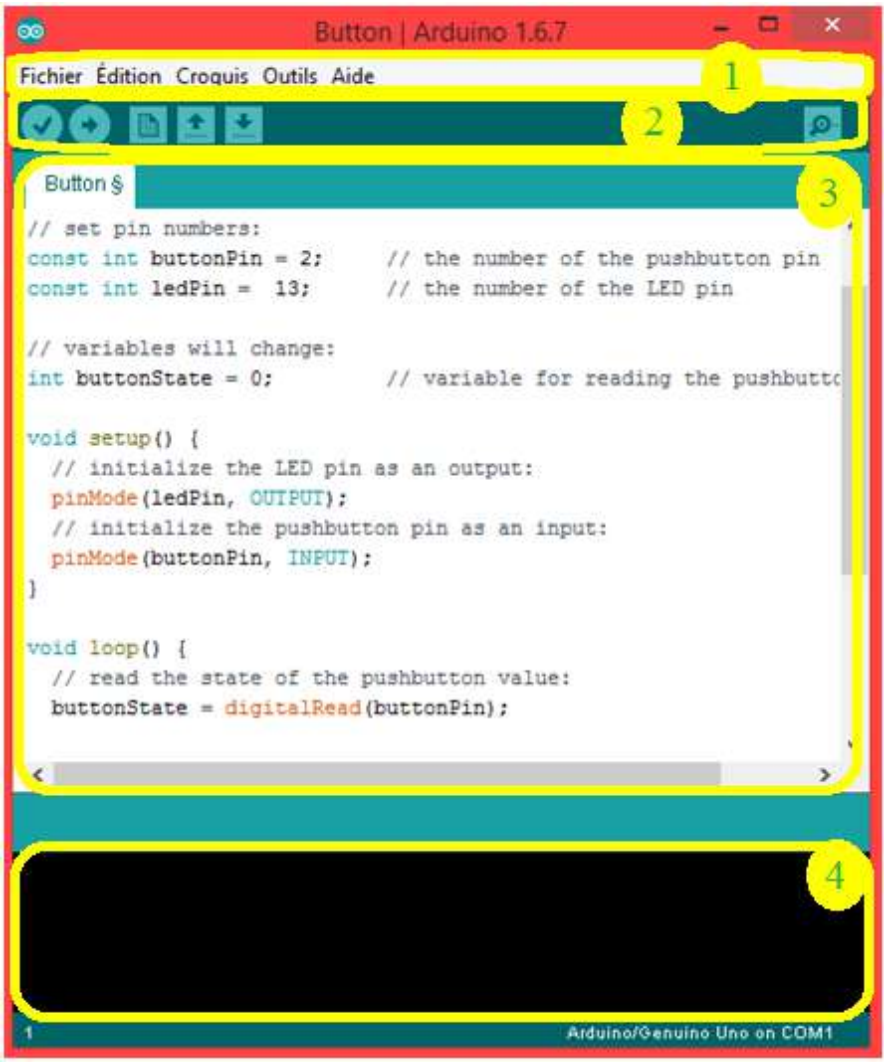


Figure II.1 *logiciel arduino*

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

II.2.6.1 Les boutons

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



Figure II.2 *Présentation des boutons*

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

II.2.6.2. Moniteur sériel

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 *baudrate* 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 ».

II.2.7. La carte Arduino mega AT2560

Dans la partie suivante nous allons présenter la carte arduino MEGA 2560 qui nous allons utiliser dans notre projet ainsi que la partie qui contient les roues el les moteur de l'arduino robot.

II.2.7.1. Présentation de la carte arduino Mega AT2560 :

La carte Arduino MegaAT2560 est une carte à microcontrôleur basée sur un ATmega2560.

Cette carte dispose :

- De 54 broches numériques d'entrées/sorties (dont 14 peuvent être utilisées en sorties PWM (largeur d'impulsion modulée)),
- De 16 entrées analogiques (qui peuvent également être utilisées en broches entrées/sorties numériques),
- De 4 UART (port série matériel),
- D'un quartz 16Mhz,
- D'une connexion USB,
- D'un connecteur d'alimentation jack,
- D'un connecteur ICSP (programmation "in-circuit"),
- et d'un bouton de réinitialisation (reset).

Elle contient tout ce qui est nécessaire pour le fonctionnement du microcontrôleur; Pour pouvoir l'utiliser et se lancer, il suffit simplement de la connecter à un ordinateur à l'aide d'un câble USB (ou de l'alimenter avec un adaptateur secteur ou une pile, mais ceci n'est pas indispensable, l'alimentation étant fournie par le port USB).

La carte Arduino Mega 2560 est compatible avec les circuits imprimés prévus pour les cartes Arduino Uno, Duemilanove ou Diecimila.

II.2.7.2. Synthèse des caractéristiques

Tableau II.1 *caractéristiques de la carte arduino mega 2560*

Microcontrôleur	ATmega2560
Tension de fonctionnement	5V
Tension d'alimentation (recommandée)	7-12V
Tension d'alimentation (limites)	6-20V
Broches E/S numériques	54 (dont 14 disposent d'une sortie PWM)
Broches d'entrées analogiques	16 (utilisables en broches E/S numériques)
Intensité maxi disponible par broche E/S (5V)	40 mA (ATTENTION : 200mA cumulé pour l'ensemble des broches E/S)
Intensité maxi disponible pour la sortie 3.3V	50 mA
Intensité maxi disponible pour la sortie 5V	Fonction de l'alimentation utilisée -500 mA max si port USB utilisé seul
Mémoire Programme Flash	256 KB dont 8 KB sont utilisés par le boot loader
Mémoire SRAM (mémoire volatile)	8 KB
Mémoire EEPROM (mémoire non volatile)	4 KB
Vitesse d'horloge	16 MHz

II.2.7.3. Alimentation

La carte Arduino Mega 2560 peut-être alimentée soit via la connexion USB (qui fournit 5V jusqu'à 500mA) ou à l'aide d'une alimentation externe. La source d'alimentation est sélectionnée automatiquement par la carte.

L'alimentation externe (non-USB) peut être soit un adaptateur secteur (pouvant fournir typiquement de 3V à 12V sous 500mA) ou des piles (ou des accus). L'adaptateur secteur peut être connecté en branchant une prise 2.1mm positif au centre dans le connecteur jack de la carte. Les fils en provenance d'un bloc de piles ou d'accus peuvent être insérés dans les connecteurs des broches de la carte appelées GND (masse ou 0V) et Vin (Tension positive en entrée) du connecteur d'alimentation.

La carte peut fonctionner avec une alimentation externe de 6 à 20 volts. Cependant, si la carte est alimentée avec moins de 7V, la broche 5V pourrait fournir moins de 5V et la carte pourrait être instable. Si on utilise plus de 12V, le régulateur de tension de la carte pourrait chauffer et endommager la carte.

La carte Arduino Mega2560 diffère de toutes les autres cartes car elle n'utilise pas le circuit intégré FTDI usb-vers-série. A la place, elle utilise un Atmega8U2 programmé en convertisseur USB-vers-série.

Les broches d'alimentation sont les suivantes :

- Vin La tension d'entrée positive : lorsque la carte Arduino est utilisée avec une source de tension externe (à distinguer du 5V de la connexion USB ou autre source 5V régulée). Nous pouvons alimenter la carte à l'aide de cette broche, ou, si l'alimentation est fournie par le jack d'alimentation, accéder à la tension d'alimentation sur cette broche.
- 5V : La tension régulée utilisée pour faire fonctionner le microcontrôleur et les autres composants de la carte (pour info : les circuits électroniques numériques nécessitent une tension d'alimentation parfaitement stable dite "tension régulée" obtenue à l'aide d'un composant appelé un régulateur et qui est intégré à la carte Arduino). Le 5V régulé fourni par cette broche peut donc provenir soit de la tension d'alimentation Vin via le régulateur de la carte, ou bien de la connexion USB (qui fournit du 5V régulé) ou de tout autre source d'alimentation régulée.
- 3V3 : Une alimentation de 3.3V fournie par le circuit intégré FTDI (circuit intégré faisant l'adaptation du signal entre le port USB de

votre ordinateur et le port série de l'ATmega) de la carte est disponible : ceci est intéressant pour certains circuits externes nécessitant cette tension au lieu du 5V). L'intensité maximale disponible sur cette broche est de 50mA

- GND : Broche de masse (ou 0V).

II.2.7.4. Mémoire

L'ATmega 2560 a 256Ko de mémoire FLASH pour stocker le programme (dont 8Ko également utilisés par le boot loader). L'ATmega 2560 a également 8 ko de mémoire SRAM (volatile) et 4Ko d'EEPROM

II.2.7.5. Entrées et sorties

II.2.7.5.1. Broches numériques

Chacune des 54 broches numériques de la carte Mega peut être utilisée soit comme une entrée numérique, soit comme une sortie numérique, en utilisant les instructions pinMode(), digitalWrite()et digitalRead() du langage Arduino. Ces broches fonctionnent en 5V. Chaque broche peut fournir ou recevoir un maximum de 40mA d'intensité et dispose d'une résistance interne de "rappel au plus" (pull-up) (déconnectée par défaut) de 20-50 KOhms. Cette résistance interne s'active sur une broche en entrée à l'aide de l'instruction. Digital Write (broche, HIGH)

De plus, certaines broches ont des fonctions spécialisées :

- **Communication Série :**
  PorteSérie Serial : 0 (RX) et 1 (TX);
  Porte Série Serial 1: 19 (RX) et 18 (TX);
  Porte Série Serial 2: 17 (RX) et 16 (TX);
  PorteSérie Serial 3: 15 (RX) et 14 (TX).
  Utilisées pour recevoir (RX) et transmettre (TX) les données séries de niveau TTL. Les broches 0 (RX) et 1 (TX) sont connectées aux broches correspondantes du circuit intégré ATmega8U2 programmé en convertisseur USB-vers-série de la carte, composant qui assure l'interface entre les niveaux TTL et le port USB de l'ordinateur.
- **Interruptions Externes :** Broches 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), et 21 (interrupt 2). Ces broches peuvent être configurées pour déclencher une interruption sur

une valeur basse, sur un front montant ou descendant, ou sur un changement de valeur.

- **Impulsion PWM (largeur d'impulsion modulée) :** Broches 0 à 13. Fournissent une impulsion PWM 8-bits à l'aide de l'instruction analogWrite()
- **SPI (Interface Série Périphérique) :** Broches 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). Ces broches supportent la communication SPI (Interface Série Périphérique) disponible avec la librairie pour communication SPI. Les broches SPI sont également connectées sur le connecteur ICSP qui est mécaniquement compatible avec les cartes Uno, Duemilanove et Diecimila.
- **I2C**: Broches 20 (SDA) et 21 (SCL). Supportent les communications de protocole I2C (ou interface TWI (TwoWire Interface - Interface "2 fils"). Noter que ces broches n'ont pas le même emplacement que sur les cartes Uno, Duemilanove ou Diecimila.
- **LED**: Broche 13. Il y a une LED incluse dans la carte connectée à la broche 13. Lorsque la broche est au niveau HAUT, la LED est allumée, lorsque la broche est au niveau BAS, la LED est éteinte.

II.2.7.5.2. Broches analogiques

La carte Mega2560 dispose de 16 entrées analogiques, chacune pouvant fournir une mesure d'une résolution de 10 bits (c.-à-d. sur 1024 niveaux soit de 0 à 1023) à l'aide de la très utile fonction analogRead() du langage Arduino. Par défaut, ces broches mesurent entre le 0V (valeur 0) et le 5V (valeur 1023), mais il est possible de modifier la référence supérieure de la plage de mesure en utilisant la broche AREF et l'instruction analogReference() du langage Arduino.

Note : les broches analogiques peuvent être utilisées en tant que broches numériques.

II.2.7.5.3. Autres broches

Il y a deux autres broches disponibles sur la carte :

- **AREF** : Tension de référence pour les entrées analogiques (si différent du 5V). Utilisée avec l'instruction analogReference().

- **Reset** : Mettre cette broche au niveau BAS entraîne la réinitialisation (= le redémarrage) du microcontrôleur. Typiquement, cette broche est utilisée pour ajouter un bouton de réinitialisation sur le circuit qui bloque celui présent sur la carte.

II.2.7.6. Communication

La carte Arduino Mega2560 dispose de toute une série de facilités pour communiquer avec un ordinateur, une autre carte Arduino, ou avec d'autres microcontrôleurs.

L'ATmega2560 dispose de quatre UARTs (Universal Asynchronous Receiver Transmitter ou émetteur-récepteur asynchrone universel en français) pour communication série de niveau TTL (5V) et qui est disponible sur les broches 0 (RX) et 1 (TX). Un circuit intégré ATmega8U2 sur la carte assure la connexion entre cette communication série de l'un des ports série de l'ATmega 2560 vers le port USB de l'ordinateur qui apparaît comme un port COM virtuel pour les logiciels de l'ordinateur. Le code utilisé pour programmer l'ATmega8U2 utilise le driver standard USB COM, et aucun autre driver externe n'est nécessaire.

Le logiciel Arduino inclut une fenêtre terminal série (ou moniteur série) sur l'ordinateur et qui permet d'envoyer des textes simples depuis et vers la carte Arduino. Les LEDs RX et TX sur la carte clignote lorsque les données sont transmises via le circuit intégré ATmega8U2 utilisé en convertisseur USB-vers-série et la connexion USB vers l'ordinateur (mais pas pour les communications série sur les broches 0 et 1).

Une librairie Série Logicielle permet également la communication série (limitée cependant) sur n'importe quelle broche numérique de la carte UNO.

L'ATmega2560 supporte également la communication par protocole I2C (ou interface TWI (TwoWire Interface - Interface "2 fils") et SPI :

- Le logiciel Arduino inclut la librairie *Wire* qui simplifie l'utilisation du bus I2C.
- Pour utiliser la communication SPI (Interface Série Périphérique), la librairie pour communication SPI est disponible.

II.2.7.7. Programmation

La carte Mega2560 peut être programmée avec le logiciel Arduino. Il suffit de sélectionner "Arduino Mega" dans le menu ***Tools >Board.***

Le microcontrôleur ATmega2560 présent sur la carte Arduino Mega2560 est livré avec un boot loader (petit programme de démarrage) préprogrammé qui vous permet de transférer le nouveau programme dans le microcontrôleur sans avoir à utiliser un matériel de programmation externe. Ce boot loader communique avec le microcontrôleur en utilisant le protocole original STK500

Vous pouvez bien sûr passer outre le boot loader et programmer le microcontrôleur via le connecteur ICSP (In-Circuit Serial Programming - "Programmation Série Dans le circuit" en français);

La source du code pour le circuit intégré ATmega8U2 est disponible. L'ATmega8U2 est chargé avec un boot loader DFU qui peut être activé en connectant le cavalier au dos de la carte (près de la carte de l'Italie) et en réinitialisant le 8U2. En utilisant le logiciel FLIP de chez Atmel (Windows) ou le programmeur DFU (Mac OS X et Linux) pour charger le nouveau code. Ou bien en utilisant le connecteur ICSP avec un programmateur externe (pour réécrire le boot loader DFU).

II.2.7.8. Réinitialisation automatique

Plutôt que de nécessiter un appui sur le bouton poussoir de réinitialisation avant un transfert de programme, la carte Arduino Mega2560 a été conçue de telle façon qu'elle puisse être réinitialisée par un logiciel tournant sur l'ordinateur. Une des broches matérielles de contrôle du flux (DTR) du circuit intégré ATmega8U2 est connecté à la ligne de réinitialisation de l'ATmega2560 via un condensateur de 100 nanofarads. Lorsque cette broche est mise au niveau BAS, la broche de réinitialisation s'abaisse suffisamment longtemps pour réinitialiser le microcontrôleur. Le logiciel Arduino utilise cette possibilité pour vous permettre de transférer votre programme dans la carte par un simple clic sur le bouton de transfert de la barre de boutons de l'environnement Arduino. Cela signifie que le boot loader peut avoir un temps mort plus court, la mise au niveau bas de la broche DTR étant bien coordonnée avec le début du transfert du programme.

Ce fonctionnement a d'autres implications. Quand la carte Mega2560 est connectée à un ordinateur fonctionnant soit sous Mac Os X ou Linux, la carte se

réinitialise à chaque fois qu'une connexion se fait entre elle et le logiciel (via le port USB). Pendant la demi-seconde suivante, le boot loader s'exécute sur la carte Mega2560. Puisqu'il est programmé pour ignorer les données mal formatées (c'est à dire toute donnée en plus du transfert d'un nouveau programme), il interceptera les quelques premiers octets de donnée envoyé à la carte après qu'une connexion soit établie. Si un programme s'exécutant sur la carte reçoit une configuration initiale ou d'autres données quand il démarre, il faut s'assurer que le logiciel avec lequel il communique attend une seconde après l'ouverture de la connexion avant d'envoyer ces données.

La carte Mega2560 comporte une piste (du circuit imprimé) que vous pouvez couper pour désactiver la réinitialisation automatique. Les bords de chaque côté de la piste peuvent solidarisés ensemble pour la réactiver. Vous pouvez également désactiver la réinitialisation automatique en connectant une résistance de 110 Ohms entre le +5V et la ligne de réinitialisation (reset).

II.2.7.9. Protection du port USB contre la surcharge en intensité

La carte Arduino Mega2560 intègre un polyfusible ré-initialisable qui protège le port USB de votre ordinateur contre les surcharges en intensité (le port USB est généralement limité à 500mA en intensité). Bien que la plupart des ordinateurs aient leur propre protection interne, le fusible de la carte fournit une couche supplémentaire de protection. Si plus de 500mA sont appliqués au port USB, le fusible de la carte coupera automatiquement la connexion jusqu'à ce que le court-circuit ou la surcharge soit stoppé.

II.2.8.Synthèse

Au travers de ce descriptif détaillé de la carte Arduino, nous avons mis en exergue les spécificités ainsi que les principales caractéristiques qui rendent son utilisation aisée et rapide à mettre en œuvre. Cette partie se veut une synthèse pour que les lecteurs intéressés par cette carte puissent avoir une prise en main rapide et développer rapidement leurs applications.

Quant à nous, nous avons utilisé cette carte embarquée pour interfacer l'ensemble capteur / actionneur au Mini-PC Raspberry. La carte Arduino associée à d'autres extensions (Shield) garantira d'une part la sécurité de fonctionnement des moteurs du robot mobile et d'autre part de contrôler les sens de rotation et les vitesses des moteurs du robot. L'utilisation de l'Arduino rendra notre architecture

matérielle évolutive dans les sens ou peut rajouter ou retirer aisément un ou plusieurs capteurs sans que cela n'implique un développement couteux en temps et en énergie. C'est l'atout majeur de ce type de système.

L'autre Atout de la carte Arduino est qu'elle peut être relier de diverses méthodes (usb, I2C, …) à un PC ou à la carte Raspberry Pi. C'est cette dernière carte que nous avons choisi d'intégrer à notre architecture matérielle. Dans la deuxième partie de ce chapitre, nous vous décrirons en détaille la constitution et l'utilisation (programmation) de ce mini-PC.

II.3. La carte Raspberry Pi

Nous présenterons dans la suite, le Raspberry Pi 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 est un petit ordinateur (Mini-PC) qui fonctionne sous le système d'exploitation Linux sur carte SD destiné à des applications d'informatique embarquée. Le cœur de l'ordinateur est un FPGA (Broadcom 2835) intégrant un processeur ARM11 cadencé à 700MHz et de nombreux périphériques.[9]

Le Raspberry Pi peut être directement connecté à une IHM classique, souris/clavier/écran HDMI ou vidéo composite, cependant comme tout ordinateur Linux, Raspberry Pi peut intégrer ses propres outils de développement et une interface homme-machine reposant sur SSH contrôlable depuis un autre ordinateur par Ethernet ou WIFI.

Le connecteur d'extension supporte les entrées/sorties parallèles ainsi que la plupart des bus de communication. C'est un support particulièrement économique et puissant qui peut être facilement mis en œuvre dans de petits systèmes nécessitant un accès au monde physique par des capteurs/actionneurs disposants d'interfaces numériques.

II.3.1. Historique

II.3.1.1. Conception

En 2006, les premiers prototypes du Raspberry Pi sont développés sur des microcontrôleurs Atmel ATmega 644. Le schéma et le plan du circuit

imprimé sont rendus publics. Cet ordinateur s'inspire du BBC Micro d'Acorn Computer (1981) et est destiné à encourager la jeunesse à la programmation. Le premier prototype ARM est intégré dans un boitier de la même taille qu'une clé USB avec un port USB d'un côté et un port HDMI de l'autre. [10]

II.3.1.2. Prototype

En août 2011, 50 cartes version Alpha sont construites, ces cartes étant identiques du point de vue fonctionnel au modèle B prévu mais elles sont plus grandes pour faciliter le débogage (placement des sondes). Une démonstration montre la carte exécutant une distribution Debain avec un bureau LXDE, *Quake*3 en 1080p et une vidéo en Full HD MPEG-4 par HDMI.

En octobre 2011, une version de RISC OS 5 tournant sur la carte est présentée. Après une année de développement la version grand public sera terminée en novembre 2012. En décembre 2011, 25 cartes modèle B ont été construites et testées. Le design des cartes version Beta est le même que les cartes grand public. Une seule erreur a été découverte dans le design, certaines broches du CPU ne fonctionnaient pas correctement ; l'erreur a été corrigée avant la première production. Les cartes version Beta sont présentées démarrant Linux, jouant une bande annonce en 1080p ou exécutant le benchmark rightware Samurai OpenGL ES.

II.3.2. Configurations

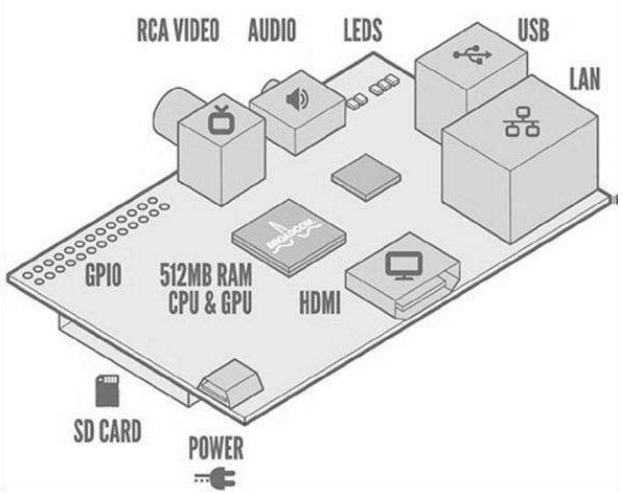


Figure II.3 *Raspberry Pi Modèle B (image Raspberry Pi Foundation)*

Il existe deux modèles de Pi : modèle A et modèle B, les deux sont semblables dés que le modèle B a de plus caractéristique que le A, la table suivante (tableau 2.1) représente la déférence entre ces deux modèles.

Dans notre projet, nous avons choisi le modèle B grâce à ces caractéristiques, la présence d'un seul port USB impose de recourir à un hub USB externe dès que l'on veut utiliser plus d'un périphérique, nous allons utiliser un module caméra pour la visualisation de notre robot, plus le port que nous allons l'utiliser pour reliant l'arduino. La RAM et aussi importante, la petite taille celle du modèle A réduit notablement les performances et de risque d'interdit même l'exécution de certains logiciels.

Dans cette partie, nous allons voir que pour faire fonctionner notre Raspberry Pi nous avons besoin d'un écran et un clavier, donc au lieu d'utiliser chaque périphérique seul nous allons utiliser un PC, et pour éviter d'utiliser un hub USB parce que ce dernier peut influence un mal performance à nos équipements, nous allons se connecter le PC et notre Raspberry Pi à l'aide d'un porte réseau, et ce dernier ne trouve pas dans le modèle A. [11]

Tableau II.2 *Configuration d'un raspberry*

Modèle A	Modèle B
Broadcom BCM2835	Broadcom BCM2835
RAM : 256 Mo	RAM : 521 Mo
Stockage : SD/ MMC / SDIO card slot	Stockage : SD/MMC/SDIO card slot
Audio : 3.5 mm jack, HDMI	Audio : 3.5 mm jack, HDMI
Pas de porte réseau	Un porte réseau 10/100 Mbits/s
Ports USB 2.0 : 1	Ports USB 2.0 : 2
Sortie vidéo : HDMI, RCA, Composite	Sortie vidéo : HDMI, RCA, Composite
Alimentation : 5V via MicroUSB ou GPIO	Alimentation : 5V via MicroUSB ou GPIO

II.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 suivante représente tout à la fois son organisation générale et les différents connecteurs disponible d'un Raspberry standard :

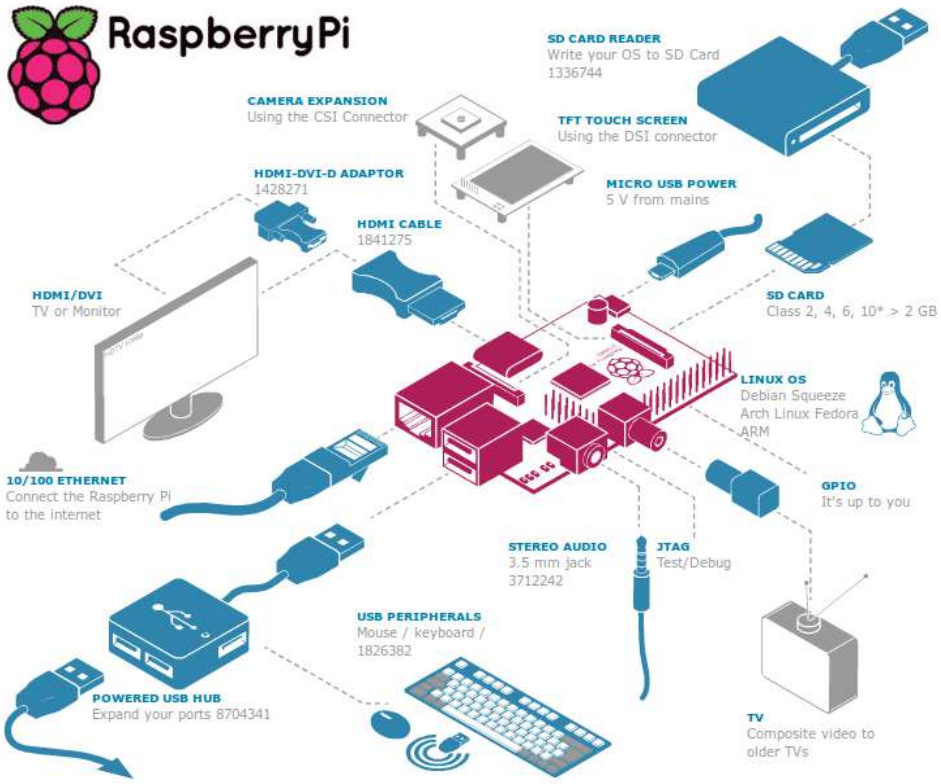


Figure II.4 *Les composants standards d'un raspberry*

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.

L'autre gros circuit intégré visible sur cette face du circuit imprimé n'est autre que le contrôleur réseau, absent bien évidemment sur le modèle A.

La connectique qui occupe la majorité de la surface de la carte et non les composants actifs, et ce d'autant que des choix pour le moins curieux ont été faits au niveau des connecteurs, dont certains sont véritablement archaïques sur un produit aussi récent.

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

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

Dans l'angle situé à coté de ces connecteurs on trouve cinq LED d'état du Raspberry Pi :

- Une LED verte d'activité du processeur ;
- Une LED rouge de mise sous tension ;
- Une LED verte signalant le mode full duplex de la connexion réseau ;
- Une autre LED verte signalant l'activité réseau ;
- Une LED jaune indiquant que le réseau fonctionne à 100 Mbits par seconde.

La carte comporte :

- **Un processeur ARM** : les architectures ARM sont des architectures de processeur, à faible consommation, introduit à partir de 1983 par « Acorn computers » et développées depuis 1990 par « ARM Ltd ».
- **Mémoire vive RAM** : c'est la mémoire dans laquelle le raspberry place les données lors de son traitement ;
- **Une connectique variée** :
  - **HDMI** : « Hight Definition Multimedia Interface » permet de relier le raspberry Pi à un dispositif compatible (écran LCD ou un vidéoprojecteur), capable de surcroît de fournir de la vidéo au format full HD de 1080p, et véhicule également le son.
  - **Port USB 2.0** : le port « Universal Serial Bus » est un port série qui sert à connecter le raspberry aux autres périphériques.
  - **Port Ethernet** : c'est un port qui correspond au protocole international ETHERNET de réseau local à communication de paquets.
  - **Port RCA** : « Radio Corporation of America », véhicule quant à lui le signal vidéo généré par le Raspberry Pi au format vidéo composite PAL ou NTSC et ne permet donc qu'un affichage en basse résolution, sur un vieux moniteur à tube cathodique de surcroit.

- **Un slot les cartes SD** : le raspberry a besoin d'une mémoire externe supplémentaire pour fonctionner. Ce slot permet de connecter la mémoire externe.
- **Une prise jack** : c'est une connectique femelle stéréo de 3.5 mm de diamètre destiné à la connexion d'un casque, d'un pair de haut-parleurs amplifiés comme ceux que l'on utilise avec les PC, ou bien encore d'un amplificateur externe.
- **GPIO** : « General Purpose Input/Output » sont des ports d'Entres/Sorties c'est-à-dire de toutes les interfaces externes propoées par le microprocesseur qui équipe le Raspberry

II.3.4. Les GPIO

GPIO simples : entrées /sorties numériques : interrupteurs ON/OFF numériques (manette de jeu, bouton-poussoir, allumer-éteindre lumière ou moteur, etc.)

PWM : « Modulation de la largeur des pics » signaux carres largeur variable : commande moteur pas à pas, variateur lumière LED, variateur vitesse moteur continu,...

Il y a un seul port GPIO configurable en PWM hardware sur Raspberry Pi, ce dernier est nécessaire pour piloter les servomoteurs (temps de réponse courts pour encodeur en plus régularité précise).

Pour contourner ce problème : la bibliothèque WiringPI permet d'ajouter autant de PWM que souhaite par soft (valable pour périphériques autres que servomoteurs).

UART : échange informations série asynchrone (pas d'horloge). Dialogue avec PC Windows (Putty).

I2C : échange informations série synchrone (horloge). Tous les récepteurs au même niveau. → Automobile, matériel hifi.

SPI : échange informations série synchrone (horloge). Récepteur maitre/esclave, maitre tournant possible. → Pilotage écrans LCD.

Toutes ces choses nous allons les exploiter dans les parties suivantes pour relier notre Raspberry au monde extérieur et lui permettre ainsi de commander et de s'interfacer avec la carte arduino.

Autrement, notre carte dispose de connecteur micro USB qui ne véhicule pas d'interface USB comme on aurait pu le penser, mais ce connecteur sert uniquement à alimenter le Raspberry Pi.

Par ailleurs, le Raspberry Pi ne comporte pas de convertisseur analogique/numérique ADC/DAC. Il est impossible de lire directement les signaux des capteurs analogiques. C'est pour cela que l'utilisation de la carte Arduino vient palier cette limitation.

II.3.5. Mise en marche d'un Raspberry Pi :

Pour faire fonctionner notre Raspberry Pi, nous avons besoin des accessoires suivants [R2] :

II.3.5.1. 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.

II.3.5.2. 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, tel que nous l'avons vu en figure II.4.

Théoriquement, toute alimentation capable de fournir cette tension et de délivrer au moins les 750 mA requis peut convenir, mais cela s'avère très souvent insuffisant. En effet, notre Raspberry Pi dispose également de deux ports USB sur lesquels nous allons être amenés à connecter divers périphérique tels que les classiques claviers et souris par exemple.

II.3.5.3. 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.

II.3.5.4. Un câble HDMI ou RCA vidéo

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 vue

En fonction du type d'écran que nous allonsutiliser, 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. Si nous souhaitons utiliser un câble RCA et que nous voulons avoir du son, il faudra nous procurer des enceintes ou un câble reliant la sortie audio du Raspberry Pi à l'entrée de notre écran.

En plus des quatre éléments ci-dessus, nous pouvons éventuellement nous procurer le suivant :

II.3.5.5. Un câble Ethernet RJ45

Le modèle B du Raspberry Pi que nous allons utiliser dans notre projet est muni d'un port RJ45 qu'il puisse être relié sur un réseau, il faut donc prévoir un câble de ce type pour pouvoir utiliser ce port. Le modèle A du Raspberry Pi ne dispose pas de port RJ45.

II.3.5.6. L'écran

Comme nous avons vu précédemment, il existe deux solutions forts différentes : l'utilisation de la sortie HDMI, vivement conseillée en raison de ses possibilités, et l'utilisation de la prise Cinch ou RCA.

Voyons tout d'abord l'utilisation de la prise HDMI et de ses variantes que sont le DVI et, dans une moindre mesure, le VGA.

II.3.5.7. La sortie vidéo composite via la prise Cinch ou RCA

Cette prise Cinch délivre, comme nous l'avons indiqué ci-dessus, un signal vidéo composite de qualité bien inférieure à celle permise par la sortie HDMI. Le recours à cette sortie ne sera donc qu'un pis-aller et permettra pas de dépasser une résolution de 640 par 480p, très en deçà de ce dont nous avons l'habitude aujourd'hui.

II.3.6. Le réseau

Même s'il n'est pas absolument indispensable de connecter le Raspberry Pi à un réseau, puisque, par exemple, le modèle A est dépourvu d'une telle interface ; cela s'avère cependant très vite nécessaire du fait de l'accès à internet que cela procure.

En effet, tous les logiciels susceptibles d'être utilisés avec le Raspberry Pi sont téléchargeables sur internet ou doivent être mis à jour par ce biais, ce qui rend bien vite cet accès indispensable.

Fort heureusement, si vous disposez d'un réseau filaire, sa connexion au Raspberry Pi est immédiate puisqu'il suffit de faire appel à n'importe quel câble RJ45 standard.

L'interface réseau du Raspberry Pi est évidemment de type auto MDI et accepte donc indifféremment les câbles réseau droits ou croisés.

Le Raspberry Pi n'intègre pas en standard un module WIFI donc il faudra faire l'acquisition d'une clé USB-WIFI si nous voulons l'utiliser.

II.3.7. Préparation de la carte SD

Le Raspberry Pi ne dispose en standard d'aucune mémoire de masse comme nous avons déjà indiquée. Ce rôle est donc dévolu à une carte mémoire de type SD qu'il faut enficher dans le connecteur prévu dans la figure 2.2. Cette carte mémoire, de type SD ou SD HC, peut avoir toute taille comprise entre 2 Go et 32 Go. Comme elle supporte le système d'exploitation ainsi que les éventuels logiciels supplémentaires que nous allons utiliser avec le Raspberry Pi comme la librairie graphique OpenCV

En effet, pour des raisons inexplicables, même par son concepteur, et alors que l'interface des cartes SD et SD HC est parfaitement normalisée, toutes ne fonctionnent pas nécessairement avec le Raspberry Pi.

Tout comme le disque dur d'un PC doit être préparé avec le système d'exploitation que l'on souhaite utiliser, la carte mémoire destinée au Raspberry Pi doit également être initialisée avec le système d'exploitation choisi.

Le Raspberry Pi n'utilisant pas un processeur disposant d'un jeu d'instructions compatibles de la famille x86 d'Intel, puisqu'il repose sur un processeur de type ARM 11, il n'est pas possible d'utiliser Windows, pas plus d'ailleurs que les différentes versions de Linux destinées à un PC. Fort heureusement, Linux appartenant au monde du logiciel libre, plusieurs versions (on dit habituellement « distribution ») ont été recompilées pour le Raspberry Pi et c'est donc avec ce système d'exploitation que vous allez devoir l'utiliser.

II.3.8. 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 [6].

Tous langages qui peuvent être compilés sur ARMv6 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.

II.3.9. A la découverte du système d'exploitation Raspbian

Raspbian, et plus précisément Raspbian Wheezy, est le système d'exploitation du Raspberry Pi par lequel il est conseillé de commencer.

C'est d'ailleurs celui que nous avons fait installer sur la carte mémoire à la fin de la partie précédente. [12]

Ne croyez pas cependant qu'il s'agisse d'un système d'exploitation « au rabais » et qu'il va vous falloir oublier les interfaces à fenêtre, les icones et le travail à la souris auquel Windows vous a peut-être habitué. Certes, un certain nombre d'action, essentiellement techniques et/ou de configuration, vont devoir être réalisées en mode ligne de commande mais Raspbian dispose aussi d'une interface

graphique de très belle facture, propre à réconcilier les aficionados de Windows avec le monde Linux.

II.3.9.1. LXDE, le bureau de Raspbian

A partir de l'invite de commande précédente, en tapant *startx*ce qui a pour effet de lancer un programme appelé LXDE pour *Lightweight X11 Desktop Environment* que l'on peut traduire en abrégé par environnement de bureau X11 allégé. L'écran, beaucoup plus sympathique, visible figure 2.3, s'affiche alors après quelques secondes d'attente.

Ce bureau fonctionne bien évidemment comme n'importe quel bureau Windows et s'utilise de façon très classique avec la souris. Nous n'allons donc pas lui consacrer de longs développements mais seulement passer en revue les différents programmes proposés dont les noms, pour qui n'est pas habitué à Linux, sont parfois un peu déroulants.

Les programmes sont accessibles soit au moyen de leurs raccourcis, dont certains sont déjà présents sur le bureau, soit au moyen du menu démarrer de Windows.

II.3.10. 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 concis que le langage Java par exemple, ce qui augmente la productivité du développeur et réduit mécaniquement le nombre de bugs.

II.3.11. Synthèse

La faible performance de l'arduino nous impose l'utilisation du Raspberry, le processeur de ce dernier est puissant et rapide ainsi que mémoire d'une capacité

relativement importante. De ce fait, nous avons repartie les charges et les opérations nécessaires à l'asservissement du robot mobile. Le Raspeberry va se charger de la partie acquisition, traitement d'images et calcul des lois de commandes qu'il va transmettre à la carte Arduino. Cette dernière se chargera de garantir la bonne exécution de ces consignes au niveau des moteurs du robot.

Les deux cartes vont communiquer via le port USB. Dans ce qui va suivre, nous allons exposer la méthode que nous avons mis en œuvre pour gérer les échanges de données entre la carte Raspberry et Arduino.

II.4. Communication par USB entre Raspberry Pi et Arduino

Le Raspberry est un outil formidable pour ses ports GPIOs, mais ceux-ci sont parfois trop limités pour des interactions électroniques beaucoup plus poussées. Il y a plusieurs façons de connecter le Raspberry Pi et Arduino, comme l'utilisation du GPIO et des broches série et en utilisant I2C. Mais cela pourrait être l'un des plus simple pour les faire parler, parce que le matériel qui est minime : tout ce que nous aurons besoin est un câble micro USB fourni avec l'Arduino.

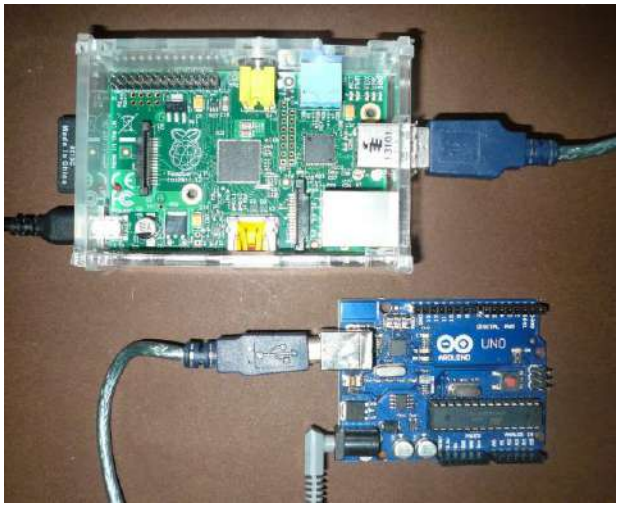


Figure II.5 *Communication Arduino et Raspberry*

On a établir une communication entre un Raspberry Pi et un Arduino pour quelques raisons possible :

- Le Raspberry Pi n'est pas idéal pour accomplir des actions avec un timing précis, car Linux pourrait décider de se consacrer temporairement à une autre

tâche. Par exemple, si nous contrôlons un moteur pas à pas avec le Raspberry Pi, le moteur pourrait s'arrêter momentanément, de temps à autres. Arduino est plus fiable dans ce genre de situation.

- Nos capteurs peuvent se trouver sur un shield conçu pour être inséré sur un Arduino. Évidemment, la disposition des pins GPIO du Raspberry Pi n'est pas la même.

La bonne nouvelle, c'est que c'est tout à fait possible, et même relativement facile, d'établir une communication entre un Raspberry Pi et un Arduino. Et ce ne sont pas les options qui manquent : communication série (UART), USB, I2C, Ethernet, communication sans fil...

II.4.1. Installation de la bibliothèque pyserial

Installons d'abord la bibliothèque *pyserial* dans le Raspberry Pi : elle nous facilitera grandement la tâche lors de la conception de nos scripts en Python. Pour ce faire, on utilise cette commande pendant que le Raspberry Pi est branché à l'internet :

// suduapt-getinstallpyserial //

Il suffit donc de brancher le connecteur USB sur le Raspberry Pi au connecteur USB de l'Arduino. C'est tout ce que nous devons faire. Il y a une légère curve ball en ce que le Raspberry Pi peut potentiellement affecter l'Arduino un des deux ports, de sorte que lors de l'ouverture du port série à l'Arduino, utilisez le code suivant Snippe

//import serial
Ser = serial. Serial ('/dev/ttyACM0', 9600)
//

Cela suppose que nous ayons configuré le code dans l'Arduino pour utiliser le port série à la même vitesse de transmission 115200 avec une commande commencé
Serial. Begin(9600)

Nous pouvons utiliser toute vitesse de transmission que les deux systèmes peuvent utiliser, mais ils doivent correspondre. Du côté Raspberry Pi, utilisez « *ser.write ()* » pour envoyer tout ce qui est dans les crochets à l'Arduino et le dos = *ser.read ()* pour obtenir un octet de retour. S'il n'y a rien reçu après le délai d'attente

été défini lors de l'ouverture du port, les retours d'appels. Nous pouvons obtenir autant d'octets de retour de cet appel que nous avons mis entre parenthèses.

II.5 Conclusion

Ce chapitre a été pour nous l'occasion de présenter les deux cartes Raspberry PI et Arduino 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 calcule 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 vous présenterons dans le dernier chapitre répondra en tous points au cahier des charges que nous nous somme imposé. Mais avant de rentrer dans ces détails de conception, nous allons, dans le prochain chapitre, vous présenter les aspects théoriques de l'asservissement visuel de robot mobile.

Chapitre III :

Introduction à l'asservissement visuel

III.1. Introduction

Comme nous l'avons cité précédemment, le Chercheur ***Grey Walter*** a conçu un robot *tortu* capable de se diriger vers une lumière qui désigne une cible. A partir du même concept et avec des cartes électroniques plus petites et plus puissantes, nous allons concevoir un robot mobile qui ressemble à une tortue disposant d'un système électronique embarqué, capable de se suivre une cible de couleur spécifique grâce à un système de vision. Ce type de commande est appelé asservissement visuel.

Dans ce chapitre, nous allons introduire les techniques d'asservissement visuel. Après un bref historique, nous décrirons en détail les différentes architectures existantes des systèmes asservis par vision. Ce descriptif sera suivi d'une présentation des différentes classifications des systèmes d'asservissement visuel selon des déférents critères tel que la position de capteur visuel, la nature des grandeurs asservies et le type de commande. Finalement, nous allons parler d'une modélisation spécifique à l'asservissement visuel à savoir la modélisation à point central unique.

Ce chapitre ainsi conçu nous permettra de situer et de justifier la technique de commande que nous avons utilisé pour asservir notre robot mobile.

III.2. Introduction à l'asservissement visuel

Historiquement, les premières expérimentations combinaient les informations visuelles aux mouvements d'un robot dans une approche en boucle ouverte présentant successivement les opérations« regarde » (looking), puis «bouge» (moving). La précision du système ainsi réalisé est liée à la précision du capteur visuel, à celle du robot, ainsi qu'à celle de sa commande. Pour améliorer la précision de l'ensemble, il est alors nécessaire d'affiner autant que possible la précision des trois briques qui le composent. Une alternative à cette escalade dans la recherche qualitative des sous-éléments du système consiste à créer une boucle de commande par retour, traditionnelle en automatique, asservie à partir d'informations issues du capteur de vision (visual-feedback).

Le terme d'asservissement visuel, « visual servoing » en anglais, fut introduit pour la première fois par Hill et Park en 1979 [13] pour distinguer leur approche d'expérimentations antérieures dites « blocks world » dans lesquelles les systèmes alternaient les étapes « prises d'images » puis « bouge ».

L'asservissement visuel est le résultat de la fusion de différents domaines spécialisés tels le traitement d'images, la cinématique, la dynamique, la théorie du contrôle, ou encore l'informatique temps réel.

C'est un domaine qui a beaucoup en commun avec la vision active et la *« structure from motion »*(reconstruction 3D à partir du mouvement caméra).

Les premiers travaux rapportés sur l'asservissement visuel remontent à 1980. Les progrès dans l'art de commander un robot par la vision ont été laborieux, cependant, depuis quelques années, le domaine a connu un regain d'intérêt, comme le prouve l'abondante littérature qui y est consacrée.

Le positionnement du capteur visuel

- La configuration du capteur embarqué : Comme l'expression l'indique, le capteur visuel est monté sur l'organe terminal du robot. Cette configuration est dénommée "eye in hand" dans la littérature anglo-saxonne. L'objectif de l'asservissement est généralement de déplacer le capteur visuel pour atteindre ou suivre par exemple un objet en mouvement.
- La configuration du capteur déporté : Pour cette configuration "eye to hand", le capteur visuel n'est pas solidaire du robot asservi. L'objectif est de faire évoluer le robot par rapport à son environnement de travail, en percevant généralement à l'aide du capteur visuel l'image du robot et des objets avec les quels il interagit.

Quelle que soit la configuration du capteur, il s'agit de sélectionner au mieux un ensemble de mesures, permettant de contrôler les degrés de liberté souhaités, et d'élaborer une loi de commande afin que ces mesures atteignent une valeur désirée ou suivent une trajectoire spécifiée. Le principe de la commande se ramène ensuite à réguler (c'est-à-dire amener et maintenir à zéro) le vecteur d'erreur entre mesure et consigne.

Avec un capteur de vision, fournissant à la base des informations 2D, la nature des mesures potentielles est extrêmement riche, puisque l'on peut considérer

en asservissement visuel aussi bien des mesures 2D, telles que les coordonnées de points caractéristiques dans l'image par exemple, que des mesures 3D, fournies par un module de localisation exploitant les informations 2D extraites. De cette richesse provient la difficulté majeure de l'asservissement visuel, à savoir, parmi l'ensemble des informations potentielles, comment sélectionner, construire et combiner celles qui fourniront un comportement satisfaisant au système.

III.3 Architectures des systèmes asservis par la vision : vers une taxonomie

En 1980, Sanderson et Weiss [14] ont introduit une taxonomie de l'asservissement visuel qui permet de catégoriser les différents systèmes rencontrés.

La taxonomie qui en résulte comporte quatre catégories que nous allons décrire ci-après.

Si l'architecture du système de contrôle est hiérarchique et utilise le système de vision pour fournir une commande aux moteurs, en utilisant un retour articulaire pour stabiliser le robot, elle est référencée comme *« dynamic look and move »*.

Par opposition, les systèmes *«visual servo»* éliminent le contrôleur robotique au profit d'un contrôle visuel qui calcul le rayon et la position d'objet directement à partir du système de vision afin de stabiliser le mécanisme.

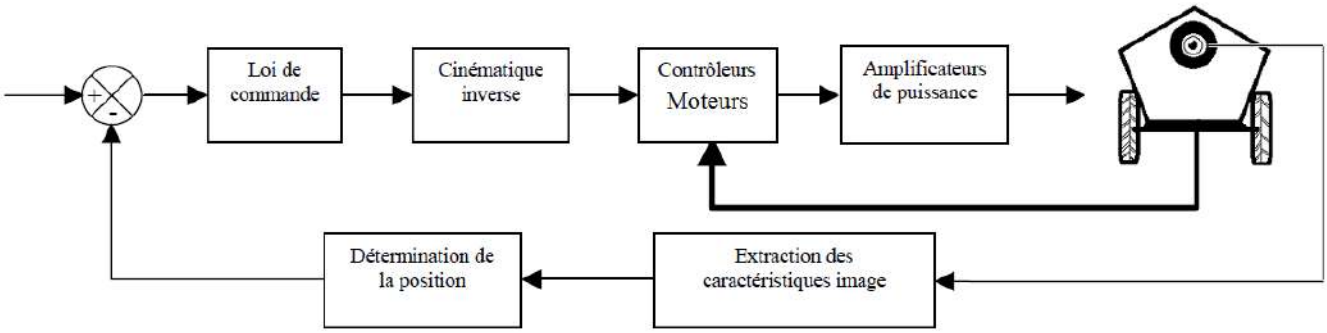


Figure III.1 *Asservissement visuel avec approche en position de type « dynamic look and move»*

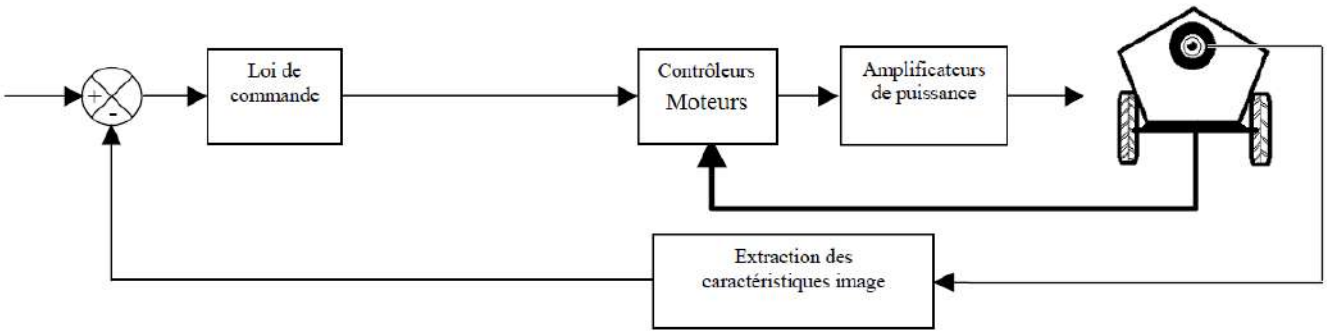


Figure III.2 *Asservissement visuel avec approche fondée sur l'image de type « dynamic look and move ».*

Pour différentes raisons, la plupart des systèmes actuels implémentent le « dynamic look and move » :

- premièrement, la faiblesse du taux d'échantillonnage du système de vision rend le contrôle direct de l'effecteur du robot ayant une dynamique complexe et non linéaire très difficile. L'emploi d'un retour interne ayant une fréquence d'échantillonnage élevée implique de supposer que le contrôleur visuel possède une dynamique de ses axes idéalisée [14],
- deuxièmement, la plupart des robots ont une interface qui gère les commandes en position et vitesse dans l'espace cartésien, ce qui simplifie la construction du système de contrôle visuel,
- troisièmement, l'approche « look and move » isole les singularités cinématiques inhérentes au mécanisme articulaire du contrôleur visuel, ce qui permet de considérer le robot comme un appareil au mouvement purement cartésien.

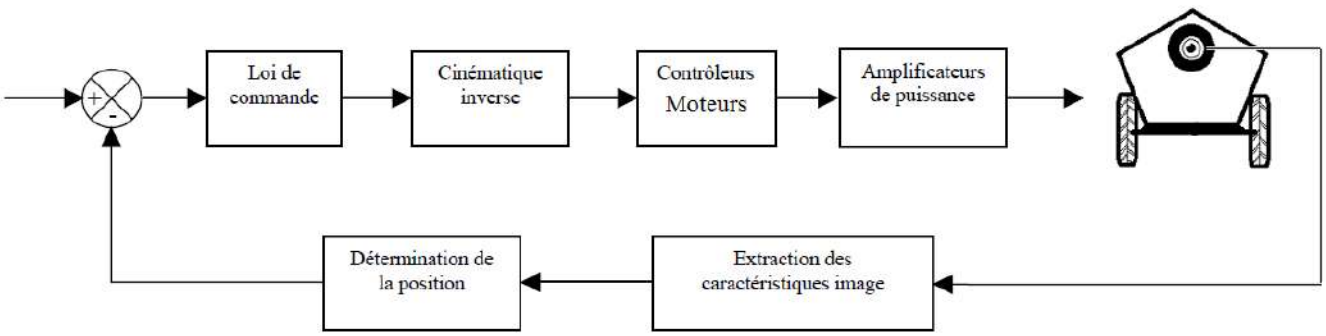


Figure III.3 *Asservissement visuel avec approche en position.*

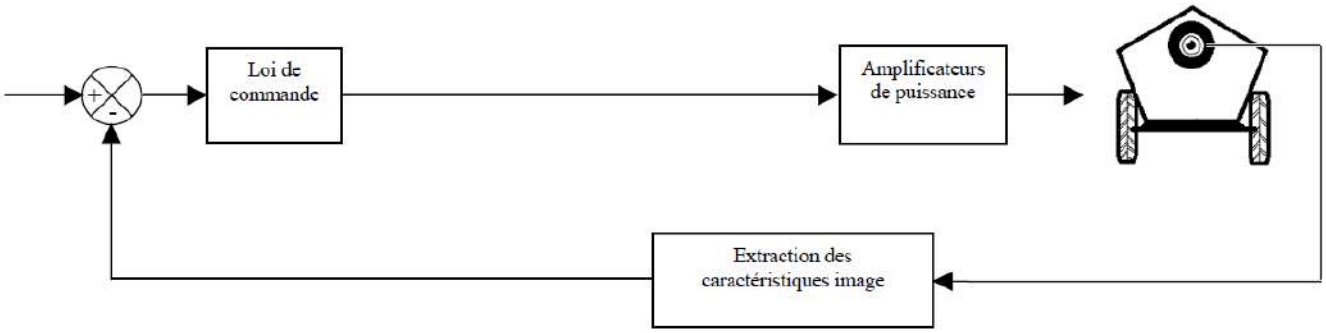


Figure III.4 : *Asservissement visuel avec approche fondée sur l'image.*

La deuxième classification des systèmes est liée à la distinction du système de contrôle en position, ou à partir des images :

- pour les systèmes asservis en position, les caractéristiques intéressantes (point effecteur, point cible, …) sont extraites des images et utilisées en conjonction avec les modèles géométriques de la cible et de la caméra afin de déterminer la position de la cible relativement à la caméra.

  Le retour de la commande est obtenu en réduisant une erreur de position dans l'espace, estimée des positions des éléments de la scène robotique,
- pour les systèmes asservis dans l'image, les valeurs des commandes sont calculées directement à partir des caractéristiques images, supprimant ainsi l'opération de modélisation requise par l'approche en position.

Les structures de commande référencées « dynamic look and move » font appel à un retour articulaire (Figure III.1 et Figure III.2), tandis que les structures d'asservissement visuel avec approche fondée sur la position et l'image (Figure III.3 et Figure III.4) s'en affranchissent.

Cette classification réalisée en 1980 par Sanderson et Weiss reste d'actualité dans ces grandes lignes, bien que la discipline ait beaucoup évolué ces dernières années.

Pour ce faire, nous allons effectuer une classification en prenant en compte deux critères :

- la configuration caméra-robot (c'est-à-dire le positionnement de la caméra par rapport au robot),

- le type des grandeurs asservies (la dimension de l'espace de référence, de mesure et de commande).

III.4. Classification des systèmes d'asservissements visuels

a) Selon la position de capteur visuel

Dans la robotique en générale on distingue grossièrement deux types de positionnement selon l'application souhaitée :

- Capteur visuel attaché à l'organe terminal du robot. Ce type de positionnement permet des applications nécessitant un grand volume de travail (surtout si le robot est mobile comme notre cas) et se prête bien à des applications de suivis de cibles.

- Capteur visuel fixe et observant la scène. Ce type de positionnement permet des applications nécessitant un faible volume de travail et se prête bien à la robotique chirurgicale où il est faut observer à la fois l'outil et l'objet.

b) Selon lanature des grandeurs asservies

- **Asservissement visuel 3D**

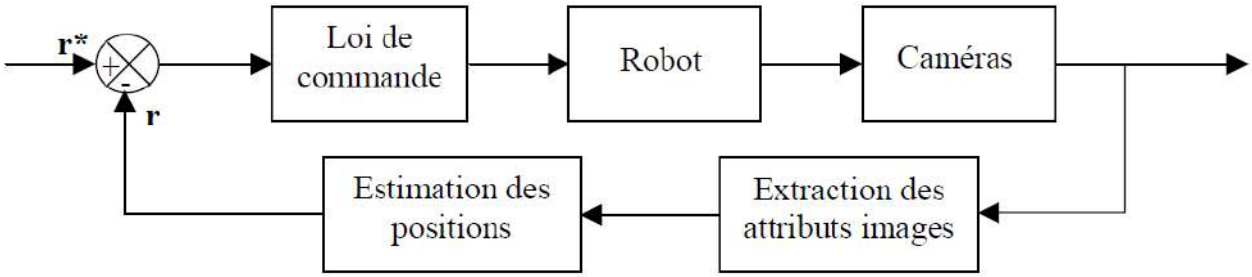


Figure III.5 *Principe de l'asservissement visuel 3D.*

Dans un asservissement visuel 3D, l'objectif est de contrôler le déplacement du robot dans l'espace cartésien. La grandeur à asservir correspond directement à l'attitude d'un repère lié rigidement à l'effecteur du robot par rapport à un repère attachéà l'objet d'intérêt. L'inconvénient majeur de ce type d'asservissement est la nécessité d'une étape de reconstruction 3D permettant de fournir une mesure

d'attitude. Cette dernière peut par exemple être obtenue par triangulation en utilisant un système de vision stéréoscopique calibré, ou par des techniques de reconstruction de pose dans le cas d'un capteur monoculaire. Les techniques de reconstruction de pose utilisent des primitives extraites de l'image et sont généralement basées sur la connaissance d'un modèle de la scène observée. Malheureusement quelle que soit la méthode de reconstruction de pose utilisée, l'estimation de l'attitude est très sensible aux bruits de mesure et aux erreurs de calibrage du capteur visuel. Il en résulte, après convergence de l'asservissement lorsque le régime permanent est atteint, un biais entre la position du robot et l'objet d'intérêt à atteindre. Cet inconvénient constitue le problème majeur de l'asservissement visuel 3D.

- **Asservissement visuel 2D**

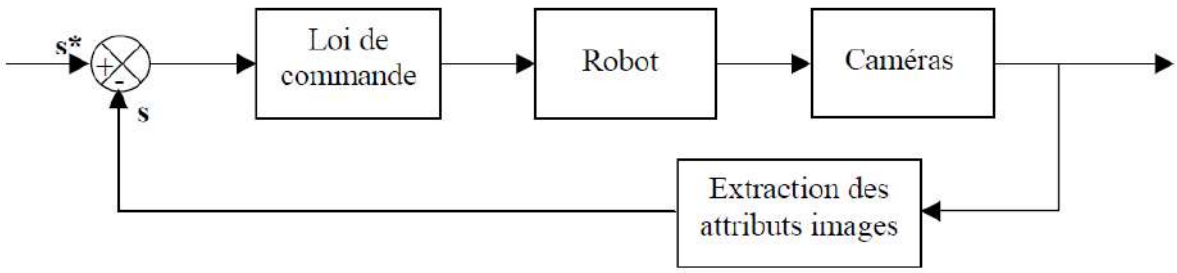


Figure III.6 *Principe de l'asservissement visuel 2D.*

Dans un asservissement visuel 2D, la grandeur à asservir est exprimée sous la forme de primitives visuelles dans l'image. Pour ce type d'asservissement, on parle également "d'asservissement référencé image". Contrairement à un asservissement visuel 3D, le contrôle du robot n'est pas effectué dans l'espace cartésien, mais directement dans l'image. Les primitives extraites de l'image sont généralement des formes géométriques élémentaires qui proviennent de l'objet d'intérêt. Différents types de primitives peuvent être utilisés. Des coordonnées de points dans l'image sont les plus couramment utilisées. Ces primitives peuvent également être des droites, des ellipses, des cylindres, des invariants projectifs, etc... Pour des scènes complexes qui ne contiennent aucune primitive géométrique simple, l'information du mouvement dans l'image ou directement l'information photométrique d'une région d'intérêt de l'image peuvent être utilisée comme primitives visuelles.

- **Asservissement visuel dynamique d2D/dt**

Pour des scènes complexes qui ne contiennent aucune primitive géométrique simple à extraire, l'information du mouvement dans l'image peut être choisie comme primitive visuelle [15]. Dans ce cas, une mesure du flot optique représentant le mouvement 2D apparent dans l'image est régulée à un champ de vitesses désiré.

- **Asservissement visuel hybride 2D ½**

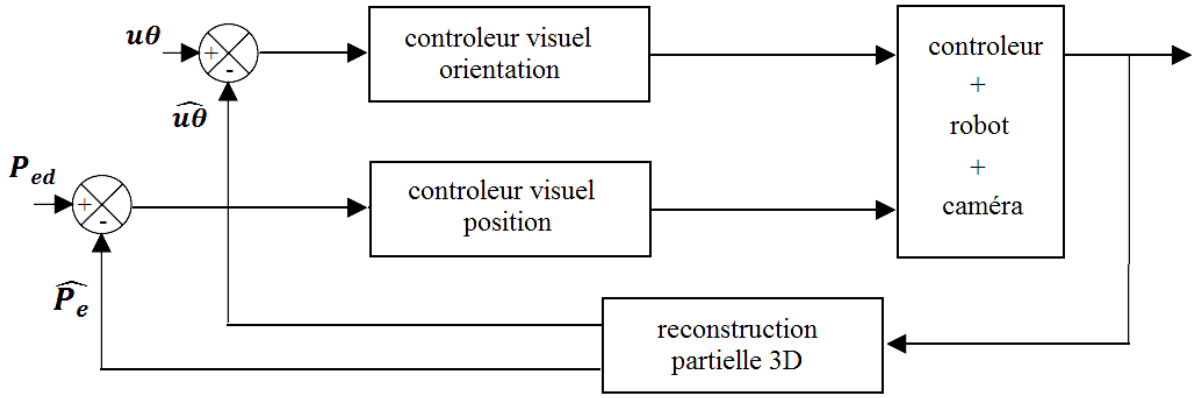


Figure III.7 *Asservissement visuel 2D1/2*

L'asservissement visuel 2D 1/2 tire partie des avantages des approches 3D et 2D sans en prendre leurs inconvénients. Un vecteur s de dimension 6 est établi à la fois à partir de grandeurs exprimées dans le plan image et dans le repère de la caméra. L'approche la plus connue [15] permet d'obtenir un découplage partiel entre les mouvements de translation et de rotation de la caméra. Elle génère une trajectoire correcte dans l'espace tout en garantissant le contrôle de la projection du centre de l'objet dans l'image afin de maintenir sa visibilité. De plus, cette approche ne requiert pas de calcul de pose et la condition de stabilité asymptotique globale du système a été démontrée [15].

c) Selon le type de commande

- **Asservissement visuel séquentiel**

La commande séquentielle qui effectue une approche de type «Static Look & Move» ou encore « Look then Move » est un asservissement en boucle ouverte (donc pas vraiment un asservissement) qui consiste à acquérir et analyser un ensemble d'images issues du capteur puis en générer une trajectoire à atteindre par le robot en boucle ouverte selon le but recherché.

Ce type de commande est utilisé pour des capteurs permettant de faibles fréquences d'acquisition **[**16**]** et peut être utilisé pour des missions de re-planification de tâches nécessitant des déplacements lents en utilisant la vision. Cependant, l'absence de retour visuel rend la convergence de l'asservissement très sensible aux incertitudes de modélisation du robot et du capteur ainsi qu'au aux méthodes de reconstruction de pose.

- **Asservissement visuel cinématique**

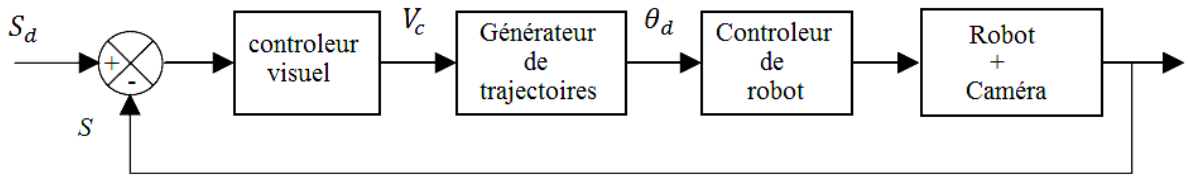


Figure III.8 *Asservissement visuel cinématique*

La commande cinématique qui effectue une approche de type « Dynamic Look & Move » ou encore « Look and Move » est un asservissement en boucle fermée, où le contrôleur visuel intervient dans une boucle de haut niveau et fourni un état de consigne en vitesse assurant une convergence asymptotique qui est transformé par le Générateur de Trajectoires en consignes angulaires transmises au contrôleur interne bas niveau du robot qui les réalise.

Le robot est considéré comme un simple système de positionnement dont la dynamique est approximée par un gain et les performances du système dépendent essentiellement de la synthèse du contrôleur visuel et du retard introduit par celui-ci dans la boucle de commande. Ce type de commande dit « indirecte » est utilisé pour des asservissements visuels lents dont le but recherché est la robustesse **[**16**]**.

- **Asservissement visuel dynamique**

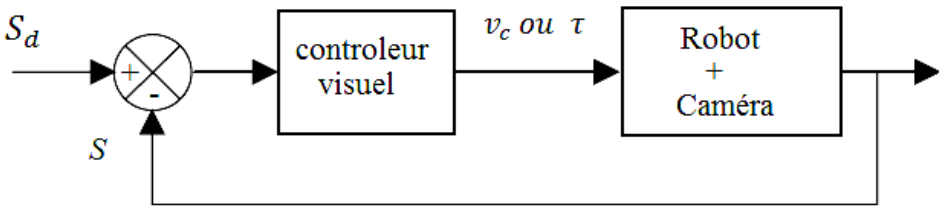


Figure III.9 *Asservissement visuel dynamique*

La commande dynamique est aussi une alternative de l'approche « Dynamic Look & Move » ou encore « Look and Move » d'asservissement en boucle fermée, où cette fois-ci le contrôleur visuel se substitue au contrôleur interne du robot en envoyant directement les consignes en vitesses ou en couples aux variateurs des moteurs.

La conception du contrôleur visuel tient en compte d'un comportement plus réaliste de la dynamique du robot et parfois même de celui du capteur visuel et/ou des actionneurs pour améliorer les performances dynamiques du système. Ce type de commande dit « directe » est utilisé pour des asservissements visuels rapides [17].

III.5. Modélisation

Dans notre application, nous avons développé un programme écris en langage Python et utilisant les fonctions prédéfinie dan une bibliothèque Open source de traitement d'images. Le but de ce logiciel est l'extraction d'informations visuelles comme le rayon et la position du centre de la cible. Ces informations permettront par la suite le calcul de la loi de commande pour que le robot puisse suivre la cible.

Un centre de projection unique est une propriété intéressante pour un capteur de vision. Cela, implique que les droites liant un point 3D de l'espace et sa projection sur le miroir passent par un point 3D unique appelé point central de projection. Les caméras perspectives conventionnelles sont des capteurs à centre unique. Baker *et al.* montrent dans [18] qu'un système catadioptrique peut être obtenu en combinant un miroir hyperbolique, elliptique ou plan avec une caméra perspective et un miroir parabolique avec une caméra orthographique.

Tableau III.1 *Description des caméras à point central unique : a_p, a_h, b_h, a_e, b_e dépendent uniquement des paramètres intrinsèques du miroir d et p*

Caméra	Surface du miroir	ξ	φ
parabolique	$z = \frac{x^2+y^2}{2a_p} - \frac{a_p}{2}$	1	1+ 2p
hyperbolique	$\frac{\left(z+\frac{d}{2}\right)^2}{a_h^2} - \frac{x^2+y^2}{b_h^2} = 1$	$\frac{d}{\sqrt{d^2+4p^2}}$	$\frac{d+2p}{\sqrt{d^2+4p^2}}$
elliptique	$\frac{\left(z+\frac{d}{2}\right)^2}{a_h^2} + \frac{x^2+y^2}{b_e^2} = 1$	$\frac{d}{\sqrt{d^2+4p^2}}$	$\frac{d-2p}{\sqrt{d^2+4p^2}}$
Plan	$z = \frac{d}{2}$	0	1
conventionnelle	aucun	0	1

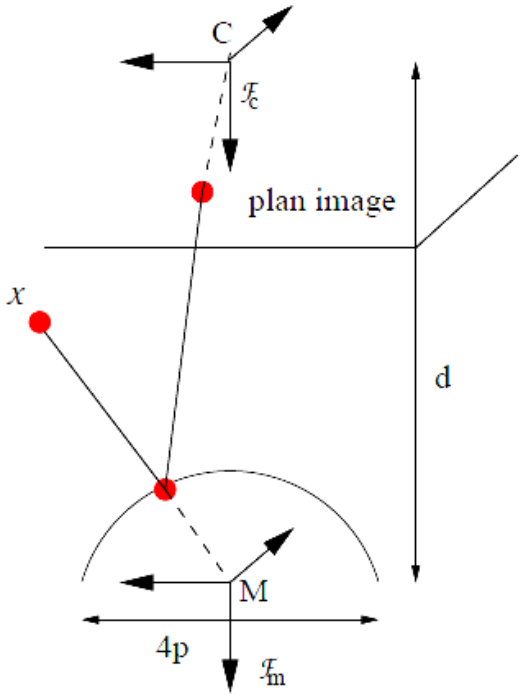


Figure III.10 *Modèle générique de caméra*

Afin de simplifier les notations, les caméras perspectives classiques seront incluses dans l'ensemble des caméras catadioptriques à point central unique. Geyer *et al.* Dans [19] proposent un modèle unifié de projection pour les caméras panoramiques à point central unique. D'après ce modèle, toutes les caméras à point

central unique peuvent être modélisées par une première projection centrale sur une sphère suivie d'une seconde projection perspective sur le plan image (voir Figure 1).

Ce modèle générique peut être paramètrisé par un couple de réels (ξ,φ) (voir Tab.1 et se référer à [19]). Soit $F_c$et F_m des repères attachés à la caméra conventionnelle et au miroir respectivement.

Dans la suite, on supposera que $F_c$et F_m sont liés par une translation le long de l'axe Z. Les centres C et M de F_c et F_m seront appelés centre optique et point principal respectivement. Soit X un point 3D de coordonnées $\mathbf{X} = (X\ Y\ Z)$ dans F_m.

D'après le modèle générique de projection proposé dans [19], X est projeté dans le plan image en un point $\mathbf{x} = (x\ y\ 1)$ avec : x= KMF(X)

Où K est une matrice triangulaire supérieure contenant les paramètres de calibrage interne de la caméra conventionnelle, et :

$$M = \begin{pmatrix} \varphi - \xi & 0 & 0 \\ 0 & \varphi - \xi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad F(X) = \begin{pmatrix} \frac{X}{z+\xi\sqrt{X^2+Y^2+Z^2}} \\ \frac{Y}{z+\xi\sqrt{X^2+Y^2+Z^2}} \\ 1 \end{pmatrix}$$

Nous supposerons sans perte de généralité, que les matrices K et M sont les matrices identités (ce qui implique qu'une calibration du capteur, même grossière, a été réalisée). La fonction de projection décrivant une projection par une caméra à point centrale unique est alors donnée par x = f (X).

III.6. Conclusion

Dans ce chapitre, nous avons décrit les déférentes architectures qui existent en asservissement visuel ainsi qu'une classification des ces systèmes de commande. Dans notre travail, nous avons utilisé une de ces architectures de commande référencées « ***Dynamic look and move*** » qui est la structure avec approche fondée l'image.

A partir des images fournies par une simple webcam, nous utilisons des algorithmes de traitement d'images pour extraire les caractéristiques pertinentes de la scène. Dans notre application, ces caractéristiques visuelles sont le contour de la cible ainsi que la position de son centre de gravité dans l'espace image. A partir de ces informations visuelles, une loi de commande est directement calculée est transmise à l'étage de puissance composé de l'ensemble Arduino + Shield.

Dans le prochain chapitre, nous détaillerons l'architecture de la plate-forme que nous avons conçue. Nos choix quant à l'utilisation des cartes électroniques embarquées et des algorithmes de traitement d'image et d'asservissement visuel seront testés in situ au travers d'applications de suivi de cible.

IV. Description de la plate-forme robot-vision

IV.1. Introduction

Comme nous avons motionné avant, nous avons constaté que pour réaliser la commande de la robotique 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 CCD.

Notre projet vise d'une part à réaliser une architecture matérielle composée d'un Raspberry et d'un ensemble Arduino /shield et d'autre part la programmation d'un algorithme de contrôle d'un robot mobile capable de réaliser un suivi de cible.

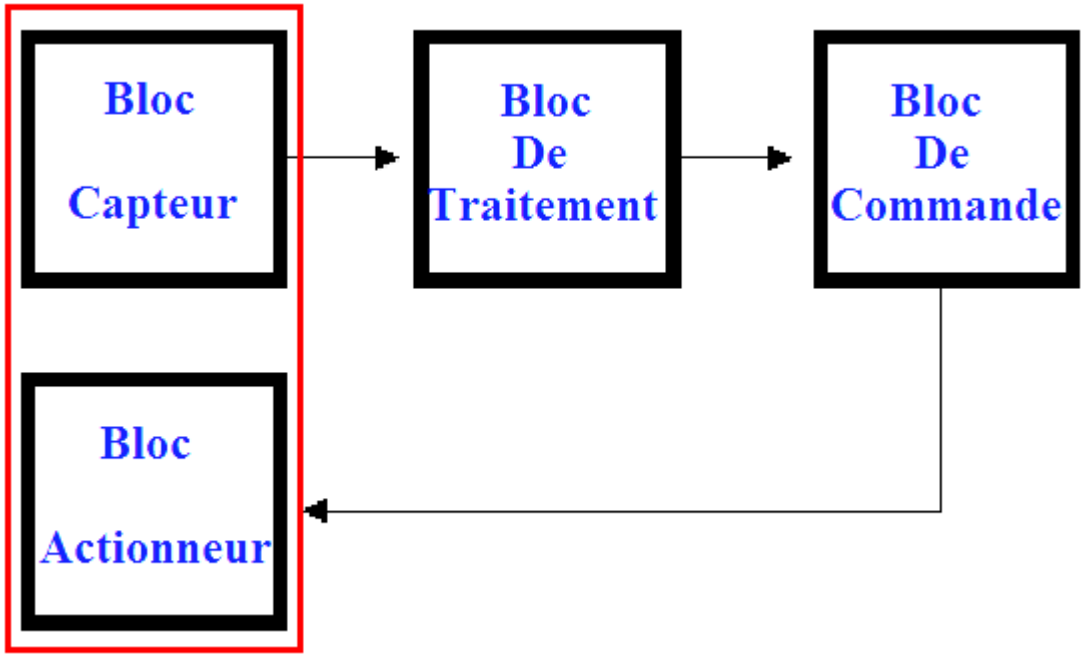


Figure IV.1 *Schéma bloc du projet*

La figure **IV**.1 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 d'une part à gérer les capteurs proprioceptif et la commande bas niveau des deux moteurs du robot. Tandis que les deux derniers

blocs ils sont regroupés au sein de la carte raspberry qui a pour rôle l'acquisition, le traitements des images issus de la webcam et de calculer la loi de commande nécessaire à la poursuite de la cible.

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 plate-forme simple, prête-à-utilisé, libre et facile à utiliser, mais nous avons utilisé seulement la plate-forme 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.

IV.2. Robot mobile Arduino

IV.2.1. Description

Notre robot mobile Arduino est de type tricycle, représenté à la figure de la figure 4.2. 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.

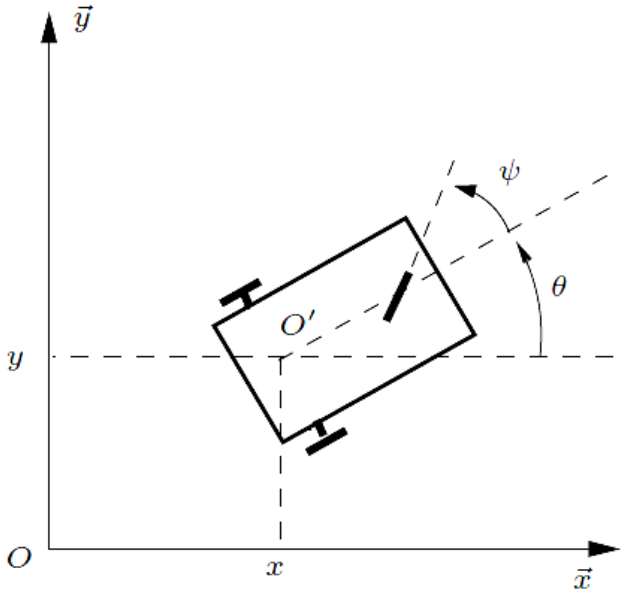


Figure IV.2 *Robot mobile de type tricycle*

IV.2.2 Modélisation

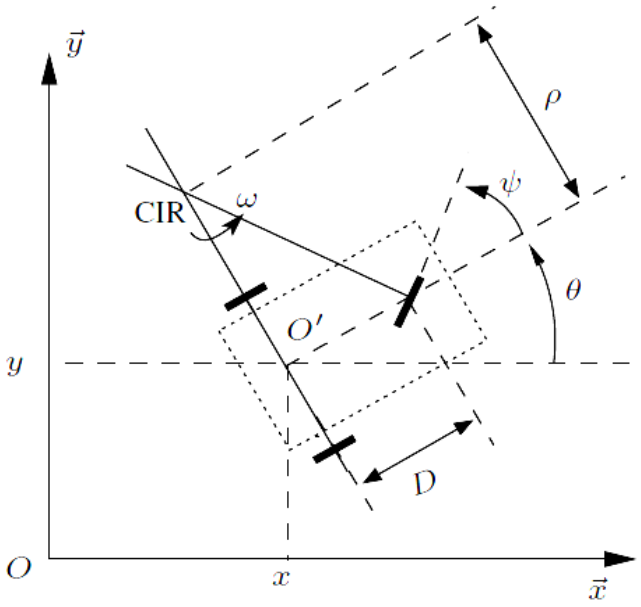


Figure IV.3 *Un robot mobile de type tricycle et son CIR*

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 4.3. 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 v du véhicule (vitesse en O') et de ρ :

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

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

Ce type de robot peut se diriger en ligne droite pour $\psi = 0$ 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 :

$$\dot{x} = v\cos\theta$$

$$\dot{y} = v\sin\theta$$

$$\dot{\theta} = \frac{v}{D}\tan\psi$$

$\dot{\psi} = \eta$,

Où $u = (v\ \eta)^T$est le vecteur de commande cinématique, η représentant la vitesse d'orientation imposée à la roue orientable. Ces équations sont celles du modèle cinématique en configuration simplifié, la configuration associée $q = (xy\theta\psi)^T$ ne décrivant pas les rotations propres des différentes roues.

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

IV.3. Le schéma d'asservissement visuel du robot mobile Arduino

L'eleboration de la loi de commande dans l'espace des images est la partie la plus importante de notre projet. Dans notre application, nous avons placé une caméra numérique sur le robot pour repérer la cibleque nous avons designé au préalable.

Le schema de commande que nous avons adopté est un asservissement visuel avec approche fondée sur l'image tel décrit par la figure 4.5.

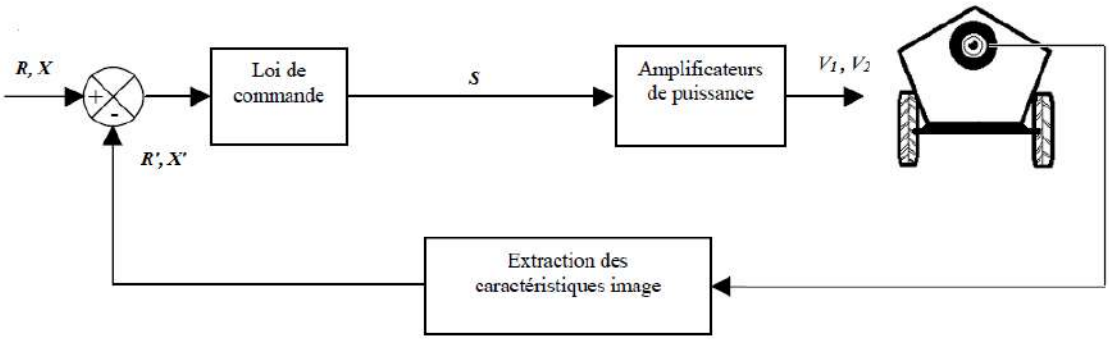


Figure IV.4 *Structure d'asservissement visuel utilisé*

Cette structure consiste à calculer directement les valeurs des commandes à partir des caractéristiques 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 webcam. Lorsque la cible désignée apparaît dans le champs de la caméra, le traitement d'images va repérer cette cible en traçant un contour autour d'elle 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 le rayon du contour de la cible. Si la valeur de ce rayon grandie, la cible est proche de la caméra et donc du robot et inversement si, la valeur du rayon diminue, la cible s'éloigne du robot. 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 R 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.

IV.4. L'architecture matérielle de gestion et de commande de la plate-forme robot-vision

Pour contrôler les actions du robot, la partie électronique programmée doit gérer : les consignes à fournir aux moteurs, la réception des informations fournies par les différents capteurs (proprioceptif et extéroceptifs), le traitement de ces données pour la prise de décision. Pour réaliser toutes ces tâches, nous avons concu un ensemble composé d'une carte Raspberry Pi doté d'une mémoire et d'une puissance de calcul importante et dune carte Arduino avec son extension (Shield) capable de gérer et de contrôler les moteurs électriques du robot mobile.

IV.4.1. La carte Raspberry Pi : le « cerveau »

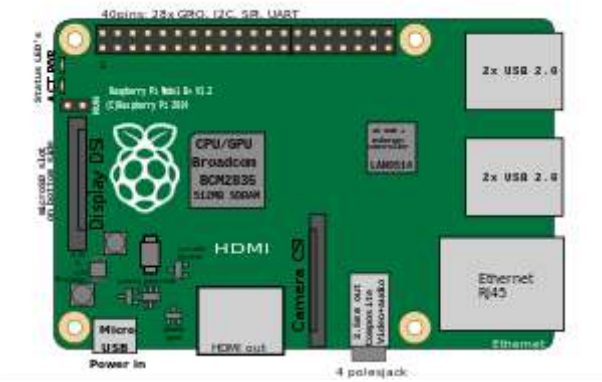


Figure IV.5 *Raspberry Pi utilisé*

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

Le modèle de Raspberry Pi utilisé le modèle B, nous avons choisi ce modèle grâce à ces caractéristiques, le tableau suivant les présente :

Tableau IV.1 : *caractéristique raspberry pi (modèle B)*

Processeur	BCM2835 de Broadcom basé sur un ARM11 à 700 MHz GPU intégrée (videocore 4 de broadcom)
Mémoire	512 Mo Sdram
Mémoire flash	Sur carte SD
Tension d'entrée	5V
Consommation	700 mA (3.5 W)
Ethernet	10/100 Ethernet
USB	4 USB 2.0
Sortie Vidéo	Composite et HDMI
Développement	Langage Python, Scratch, Squeak

Grace au système Linux, nous avons pu programmé l'ensemble de nos algorithmes en langage Python et exploité les fonctions pre-compilées de la bibliothèque graphique OpenCV (**Open** Source **C**omputer **V**ision).

Dans la suite de ce chapitre, nous décrirons brièvement les techniques que nous avons mis en œuvre pour réaliser notre asservissement visuel à savoir : le traitement d'image basé sur l'exploitation des ressources fournie par l'OpenCV, 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.

IV.4.2. La bibliothèque graphique OpenCV



Figure IV.6 *Logo de la bibliothèque graphique OpenCV*

Nous avons utilisé la bibliothèque graphique OpenCV (Open Source Computer Vision) pour faire le traitement d'image nécessaire à l'acquisition d'images et à l'extraction des données visuelles. Nous avons opter pour cette technique car elle contient toutes les fonctions "usuelles" du traitement d'image.Elle est mise à disposition par la société INTEL, spécialisée dans le traitement d'images en temps réel et elle est librement téléchargeable.

Toutes les librairies d'OpenCV ont été programmées en langage C. elle est livrée avec une interface de programmation en C, C++, Python et Android.

La bibliothèque OpenCV met à disposition de nombreuses fonctionnalités très diversifiées. Elle propose presque toutes les opérations conventionnelles de traitement d'images:

- lecture, écriture et affichage d'image
- calcul de l'histogramme des niveaux de gris ou d'histogramme de couleur
- lissage, filtrage.

Cette bibliothèque s'est imposée comme un standard dans le domaine de la recherche en robotique car elle permet une utilisation facile et rapide à mettre en œuvre de nombreux outils et méthodes de traitement vidéo:

- lecture, écriture et affichage d'une vidéo depuis un fichier ou une caméra
- détection de visage par la méthode de Viola et Jones, détection de mouvements, poursuite d'objets

IV.4.2.1. Les bases d'OpenCV

Toutes les classes et les méthodes d'OpenCV en C++ ont un espace de travail (namespace) nommée « **cv** ». Une image peut être mémorisée à l'intérieur d'une structure en C du type « Cvmat » ou « Ipimage ». La structure « Ipimage » est un vieux format d'image original compatible Intel IPP. Depuis la version 2.1 d'OpenCV l'accent a été mis sur les matrices et les opérations sur celles-ci.

En effet, la structure de base est la matrice. Une image peut être considérée comme une matrice de pixel. Ainsi, toutes les opérations de bases des matrices sont disponibles La classe Mat est un nouveau format qui est né depuis la version 2.x de OpenCV. La classe « Cv » : mat est la version C++ de « Cvmat », son avantage est de ne pas associer des pointeurs et les problèmes inhérents à ces pointeurs.

IV.4.2.2 Les principaux traitements sous OpenCV

IV.4.2.2.1 Introduction à la vision assistée par ordinateur

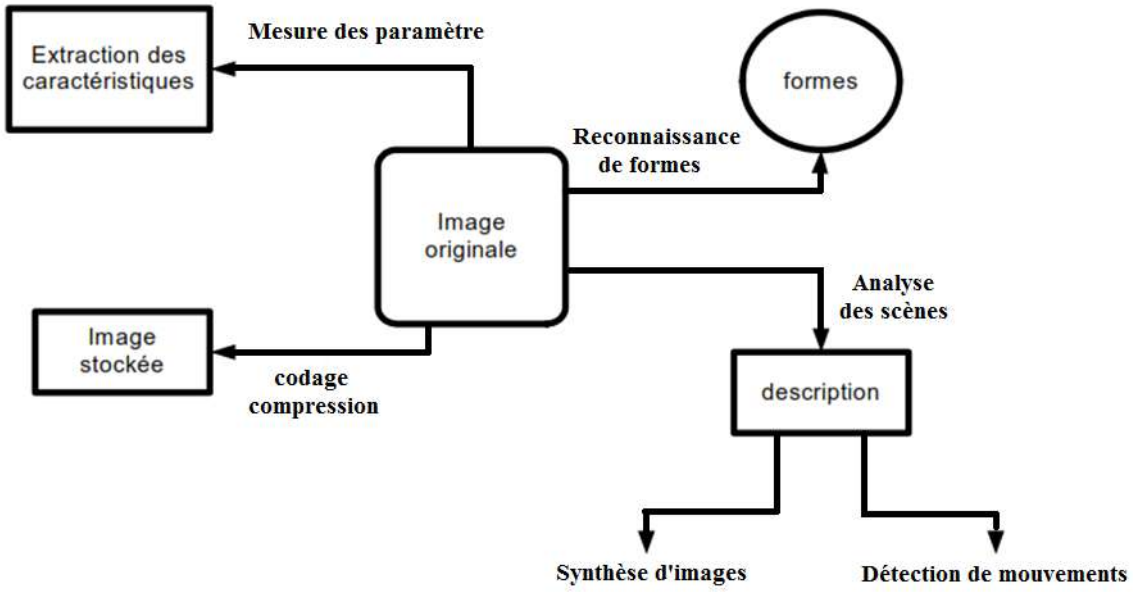


Figure IV.7 *Schéma bloc des principaux traitements sous OpenCV*

a) Définitions

V.A.O: Chaîne visuelle partant de la perception d'une image, interprétant son contenu pour agir sur l'environnement.

Traitement de l'image : Modification de l'image en vue d'en interpréter son contenu. Le traitement de l'image s'intègre comme un des outils de la chaîne de VAO.

b) Chaîne fonctionnel du traitement d'image pour la reconnaissance de formes

Une image en reconnaissance de forme est une somme d'objets (caractérisés par des contours) et des fonds que l'on appelle texture. La figure ci-dessous résumé les différentes étapes nécessaires à la reconnaissance de forme en traitement d'images.

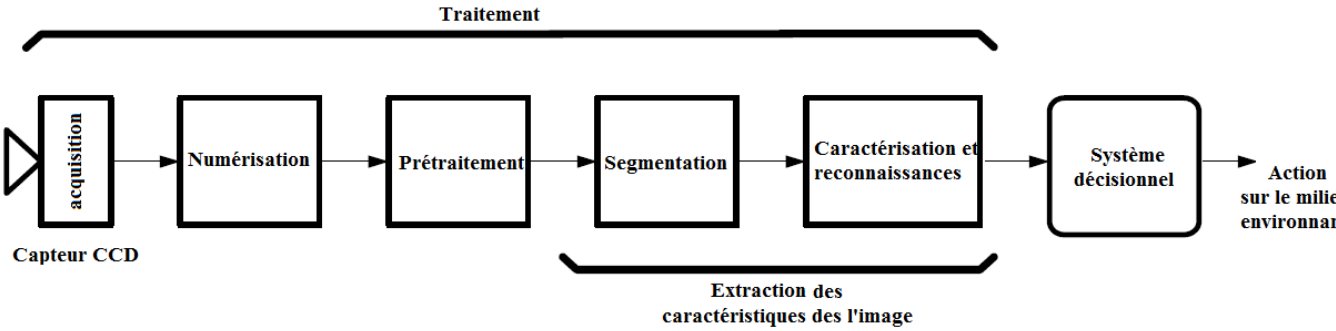


Figure IV.8 *Chaîne fonctionnel du traitement d'image*

c) Le prétraitement des images

a) Introduction

Le prétraitement d'une image a pour but d'améliorer la qualité des images (Look Up Table ou LUT, Filtrage). Le but de cette partie est de mettre en œuvre des outils permettant d'effectuer des traitements et prétraitements sur l'image affichée précédemment avant d'effectuer la phase de décision.

b) Intérêt du prétraitement

Il s'effectue sur une image brute, et permet:

- La restauration de l'image en supprimant un certain nombre de défauts tels que le bruit (pixels isolés de valeur erronée) ou le flou de mise au point.
- l'amélioration de l'image, par amélioration de l'aspect visuel ou meilleure adéquation aux besoins par exemple en assombrissant une image trop claire.

c) Les outils du prétraitement

i. L'histogramme

C'est un graphique permettant de connaitre la population de chaque niveau de gris de l'image.

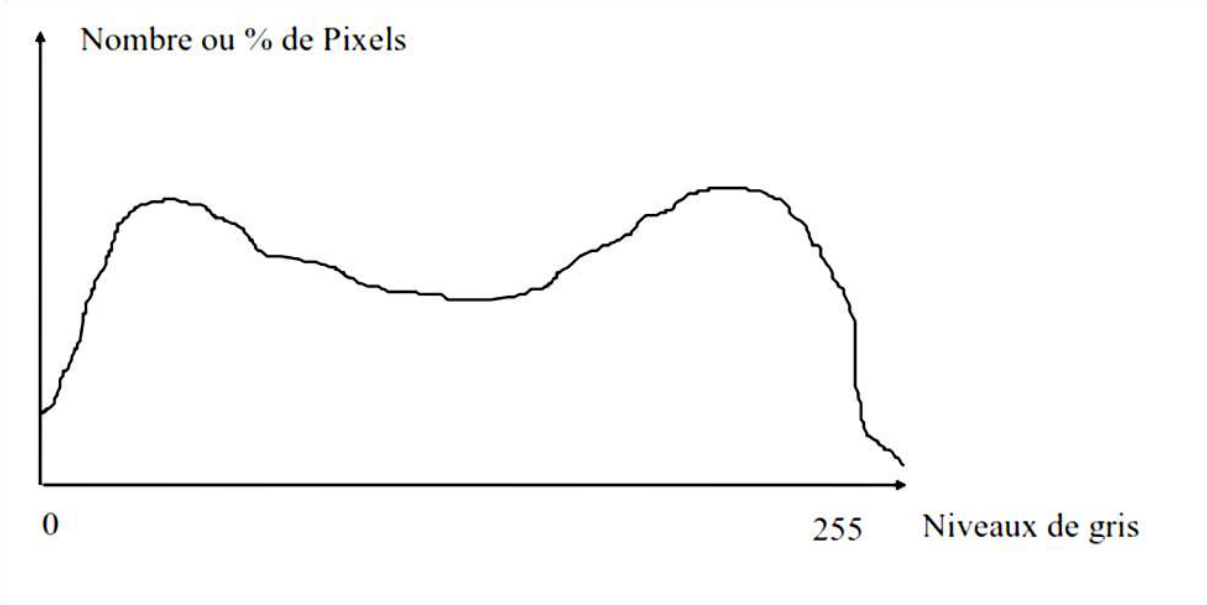


Figure IV.9 *Graphe d'histogramme*

L'observation des histogrammes permet de déceler les défauts d'une image (trop claire, trop foncée, manque de contraste) ou de déduire certaines propriétés (séparation des objets du fond).

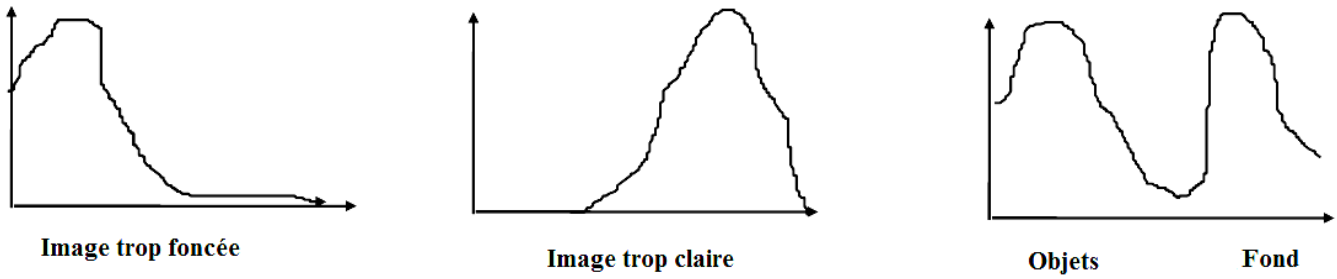


Figure IV.10 *Les défauts d'une image (deux photos à gauche) /séparation des objets du fond (la photo à droite)*

ii. Modification des histogrammes avec une LUT

Utilisation des LUT (Look Up Table)

C'est une table de conversion qui réalise une fonction de transfert entre les données numériques d'entrée et sa sortie numérique. A un niveau de gris en entrée elle fait correspondre un niveau de gris en sortie.

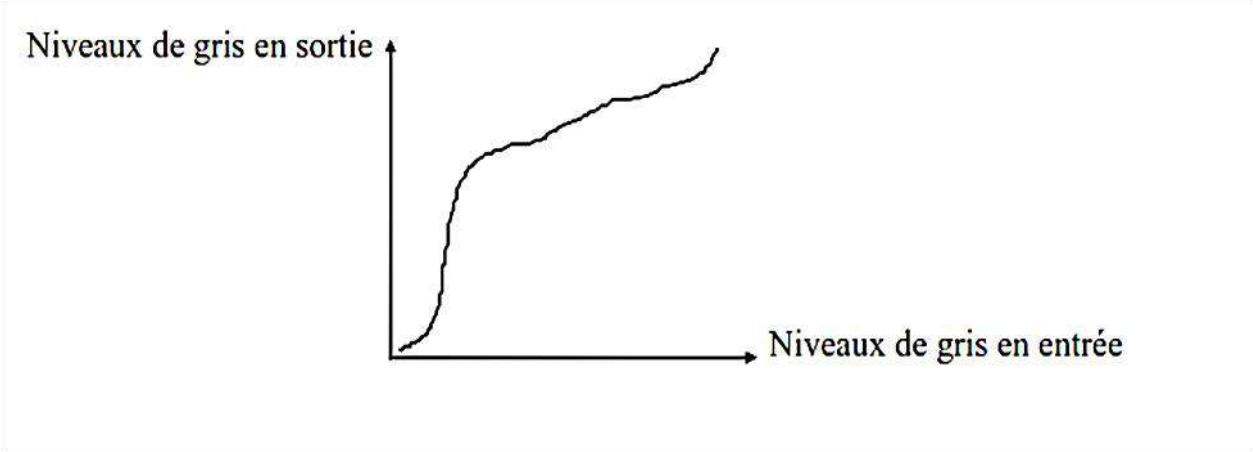


Figure IV.11 *Graphe Histogramme en utilise LUT*

En informatique, une LUT peut donc être modélisée par un tableau dont l'index représente les niveaux de gris en entrée et le contenu du tableau les niveaux de gris en sortie.

iii. Le Filtrage

Les opérations précédentes étaient ponctuelles, chaque pixel étant transformé indépendamment de la valeur de ses voisins. Dans le filtrage, au contraire chaque pixel est traité en fonction de sa propre valeur et de celle de ses voisins. On utilise souvent un masque de filtrage (matrice de convolution) qui prend en compte les 8 voisins immédiats du point traité.

$$\mathrm{H} = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} \mathrm{I}(x,y)$$

$$= [a * I(x-1, y-1) + b * I(x-1, y) + c * I(x-1, y+1) + \cdots]/K$$

iv. La segmentation

Définition

La segmentation regroupe toutes les techniques permettant de faire apparaître sur l'image les objets ou les entités d'intérêts. Ces techniques sont la détection de contour et le seuillage.

Dans une image un contour est un ensemble de points qui correspond à un changement rapide (vis à vis du voisinage des points du contour) du niveau.

Une approche classique repose sur le traitement en 2 étapes :

- Une accentuation des contours
- Un seuillage qui permet d'obtenir une image binaire.

Objectifs de la segmentation

La segmentation d'image sert à :

- fournir des régions homogènes (selon un critère donné)
- réduire le bruit.
- localiser de manière précise les contours des régions.

v. Binarisation d'une image

La binarisation est une opération de traitement simple qui consiste à ne conserver dans une image que 2 niveaux de gris (0=noir 1=blanc);

- soit en convertissant en noir tous les niveaux de gris inférieurs à un seuil et en blanc tous ceux qui dépassent le seuil
- soit en convertissant en blanc toutes les valeurs comprises entre deux seuils, et en noir tout le reste.

Le choix d'un seuil de binarisation à partir d'un histogramme permet par exemple de différencier des objets (noir) d'un fond (blanc).

IV.4.2.2.2 La détection de formes sur une image

A. Détection de contours

a. Le seuillage

Une image numérique, pour pouvoir être exploitée a généralement besoin d'être simplifiée, c'est le but du seuillage. Ce dernier consiste à transformer l'image codée sur 6, 8 ou 16 bits, en une image binaire ou les pixels à 1 correspondent aux objets et les pixels à 0 au fond de l'image.

A la différence des différents traitements de l'image numérique, le seuillage est un passage obligatoire pour toute analyse morphologique ultérieure. Le seuillage permet de sélectionner les parties de l'image qui intéressent l'opérateur.

Après seuillage, les parties de l'image sélectionnées seront traduites en noir et blanc. L'image, digitalisée par l'ordinateur (0 et 1), est appelée image binaire. Cette dernière, tout comme l'image numérique contient des informations superfétatoires, qu'il convient d'éviter, ou masquées qu'il faut révéler. Les traitements suivants permettent de modifier l'image binaire à ces fins.

b. Trouver les contours des objets (find Contours)

Cette fonction permet de trouver les contours en remplissant un vecteur de point qui correspond aux points qui sont définies sur le contour extérieur de l'objet. En fait, plusieurs constantes sont définies dans OpenCV et permettent de trouver les points extérieurs de l'objet mais aussi ceux qui font parties des « trous » internes de l'objet. On peut aussi vouloir créer une hiérarchie d'objet en passant au travers une liste des contours extérieurs jusqu'aux contours intérieurs. Dans le module précédent, nous avons appliqué un filtre de Canny pour ressortir les pixels faisant partie du contour d'un objet. On peut ensuite, avec la fonction « find Contours » trouver les points sur ce contour. Cette séquence de point est essentielle dans la reconnaissance de forme. Nous allons donc appliquer cette fonction aux objets de façon à déterminer leur forme. Cette fonction trouve tous les contours dans une image binaire.

c. Isoler une couleur sous OpenCV

L'étape que nous allons faire ici pour traiter l'image en HSV est exactement la même à la différence que maintenant nous binarisons les pixels qui se trouvent dans un certain intervalle de couleur. Malheureusement, à moins d'avoir une étendue de pixel dont la couleur est uniforme, ce qui est rarement le cas surtout si l'image provient d'un dispositif de capture comme une caméra, la binarisation ne donnera pas un bon résultat si on utilise la technique du seuil. Rappelant que dans cette technique, le seuil est une valeur qui nous permet de départager les pixels selon la relation suivante :

Pixel < Valeur du Seuil alors Pixel = 0

Pixel >= Valeur du Seuil alors Pixel = 255

Pour la couleur, il faut plutôt se donner un intervalle dans lequel le seuil se situera.

Ainsi la relation est plutôt la suivante :

$$(\mathrm{H} - \mathrm{Tolerance} \leq \mathrm{Hpixel} < H + Tolerance)\&\&(\mathrm{S} - \mathrm{Tolerance} \leq \mathrm{Spixel} < S + Tolerance)$$

Où H : valeur de la teinte (Hue)

S : Valeur de la saturation (Saturation)

IV.4.2.2.3. La détection d'objets sous OpenCV

A. La méthode de Haar : détection d'objet

Les chercheurs Paul Viola et Michael Jones en 2001 ont proposés une méthode de détection d'objet dans une image numérique. Elle fait partie des toutes premières méthodes capables de détecter efficacement et en temps réel des objets dans une image. Inventée à l'origine pour détecter des visages, elle peut également être utilisée pour détecter d'autres types d'objets comme des voitures ou des avions. La méthode de Viola et Jones est l'une des méthodes les plus connues et les plus utilisées, en particulier pour la détection de visages et la détection de personnes. Une étape préliminaire et très importante est l'apprentissage du classificateur. Il s'agit d'entraîner le classificateur afin de le sensibiliser à ce que l'on veut détecter. Pour cela, il est mis dans deux situations. La première où une énorme quantité d'images positives (avec l'objet à détecter) lui sont présentés et la deuxième où, à l'inverse, une énorme quantité de d'images négatives lui sont présentés. Concrètement, une banque d'images contenant des objets à détecter est passée en revue afin d'entraîner le classificateur. Ensuite, une banque d'images ne contenant pas d'objets est passée.

B. L'apprentissage du classificateur

a) Caractéristiques d'une image

Viola et Jones proposent d'utiliser des caractéristiques, c'est à dire une représentation synthétique et informative, calculée à partir des valeurs des pixels. Viola et Jones définissent des caractéristiques très simples, les caractéristiques pseudo-Haar, qui sont calculées par la différence des sommes de pixels de deux ou plusieurs zones rectangulaires adjacentes. La figure ci-dessus donne des exemples des caractéristiques proposées par Viola et Jones à 2, 3 ou 4 rectangles, dans lesquelles la somme des luminances de pixels sombres est soustraite de la somme des luminances des pixels blancs. Ces caractéristiques ont été choisies de façon à détecter des motifs. Par exemple, la reconnaissance des visages est rendue possible par:

- La variation de l'intensité de la lumière entre les yeux et le nez (caractéristique n°2)

- la variation de l'intensité de la lumière entre les yeux et les pommettes (caractéristique n°3)

Les caractéristiques sont calculées en soustrayant la somme des pixels noirs à la somme des pixels blancs.

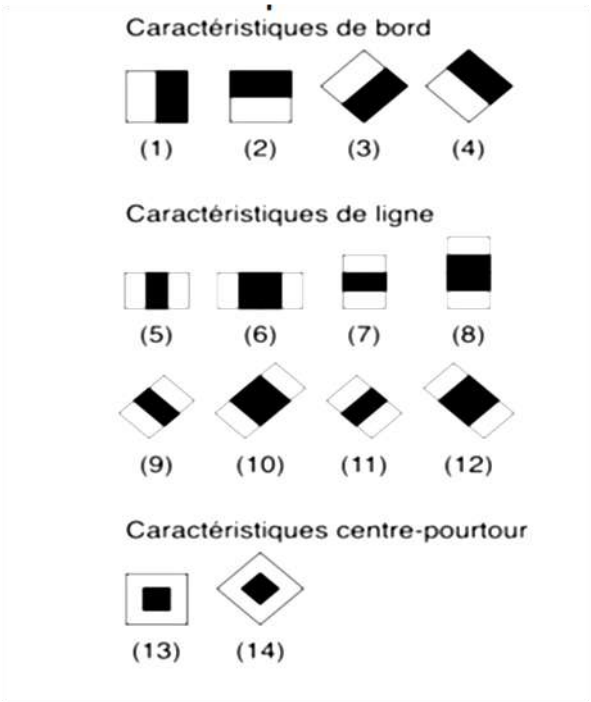


Figure IV.12 *Caractéristique d'image*

b) image intégrale

Pour calculer rapidement et efficacement ces caractéristiques sur une image, les auteurs proposent également une nouvelle méthode, qu'ils appellent « image intégrale ». C'est une représentation sous la forme d'une image, de même taille que l'image d'origine, qui en chacun de ses points contient la somme des luminances des pixels situés au-dessus de lui et à sa gauche. Plus formellement, l'image intégrale ii au point (x,y) est définie à partir de l'image i par :

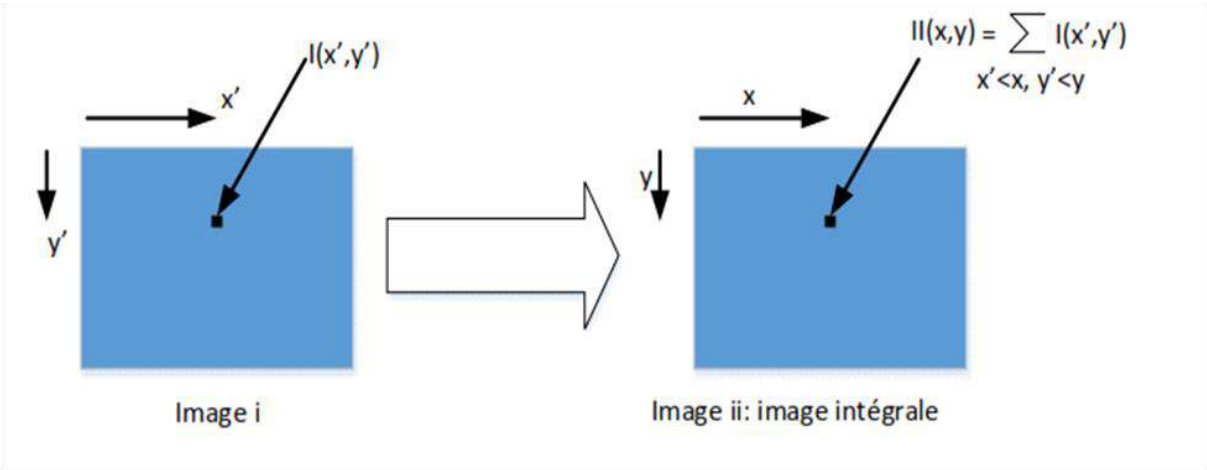


Figure IV.13 *Image intégrale*

c) Calcul

Les caractéristiques sont calculées à toutes les positions et à toutes les échelles dans une fenêtre de détection de petite taille, typiquement de 24 × 24 pixels ou de 20 × 15 pixels. Un très grand nombre de caractéristiques par fenêtre est ainsi généré, Viola et Jones donnant l'exemple d'une fenêtre de taille 24 × 24 qui génère environ 160 000 caractéristiques pour une image.

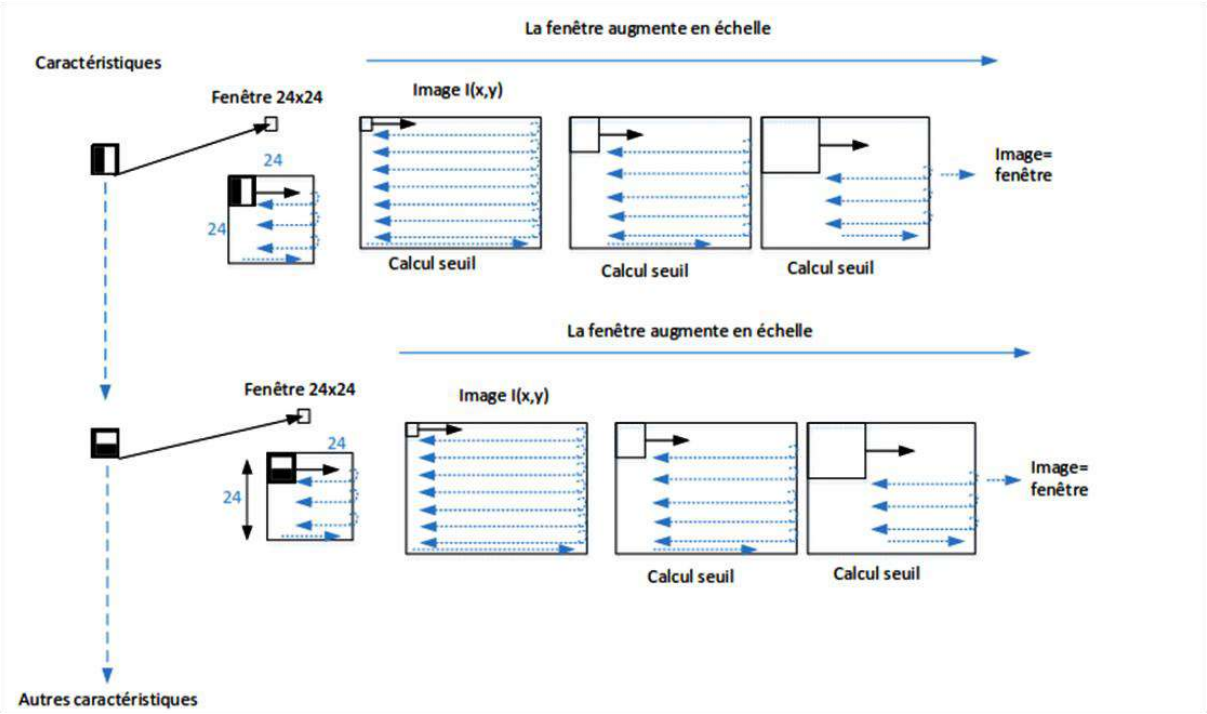


Figure IV.14 *Calcule des caractéristiques*

La taille de la caractéristique de pseudo-Haar est variable et permet d'augmenter le nombre de caractéristiques évaluées En phase de détection, l'ensemble de l'image est parcouru en déplaçant la fenêtre de détection (24x24) d'un certain pas dans le sens horizontal et vertical (ce pas valant 1 pixel dans l'algorithme original). Les changements d'échelles se font en modifiant successivement la taille de la fenêtre de détection. Viola et Jones utilisent un facteur multiplicatif de 1,25, jusqu'à ce que la fenêtre couvre la totalité de l'image. Finalement, et afin d'être plus robuste aux variations d'illumination, les fenêtres sont normalisées par la variance des pixels de la fenêtre.

C. Elaboration du classificateur

Sélection de caractéristiques par boosting Le deuxième élément clé de la méthode de Viola et Jones est l'utilisation d'une méthode de boosting afin de sélectionner les meilleures caractéristiques (sur les 14 disponibles). Le boosting est un principe qui consiste à construire un classificateur « fort » à partir d'une combinaison pondérée de classificateur « faibles », c'est-à-dire donnant en moyenne une réponse meilleure qu'un tirage aléatoire. Viola et Jones adaptent ce principe en assimilant une caractéristique à un classificateur faible, en construisant un classificateur faible qui n'utilise qu'une seule caractéristique. Après avoir calculé toutes les caractéristiques provenant de chacune des formes de la fonction de Haar, c'est le temps de définir un seuil selon lequel on décide si l'image en traitement est un visage ou non. Chaque paramètre passe sur toutes les images intégrales calculées à partir des images originales de la base de données adoptée. Ainsi, ces seuils seront enregistrés dans un fichier qui contiendra les poids de tous les classificateurs de Haar.

Une fois le calcul de la fonction Haar a été complété pour les images positives, la moyenne, l'écart-type, le min et le max sont calculés pour toutes les valeurs données par chaque image. Ces paramètres sont calculés afin de catégoriser un classificateur comme bon ou mauvais par une distribution gaussienne : on commence à partir de la valeur moyenne et on élargit selon l'écart type, plus l'écart-type est grand, plus la reconnaissance sera pour les visages. On doit faire le même calcul pour les images sans objets. Pour tester les classificateur obtenus et décider quels sont les forts classificateur qu'on

va utiliser ensuite dans la chaîne en cascade, chaque classificateur, ayant une distribution gaussienne, doit vérifier 3 conditions :

- Obtenir une reconnaissance d'objets de près de 100% (presque toutes les images d'objets à détecter sont détectées).
- Obtenir une reconnaissance sans objets très faible (la plupart des images sans objets ne sont pas détectées).
- Erreur totale pour le classificateur inférieure à 50%.

Si et seulement si ces trois conditions sont réunies, on continue avec le classificateur en traitement.

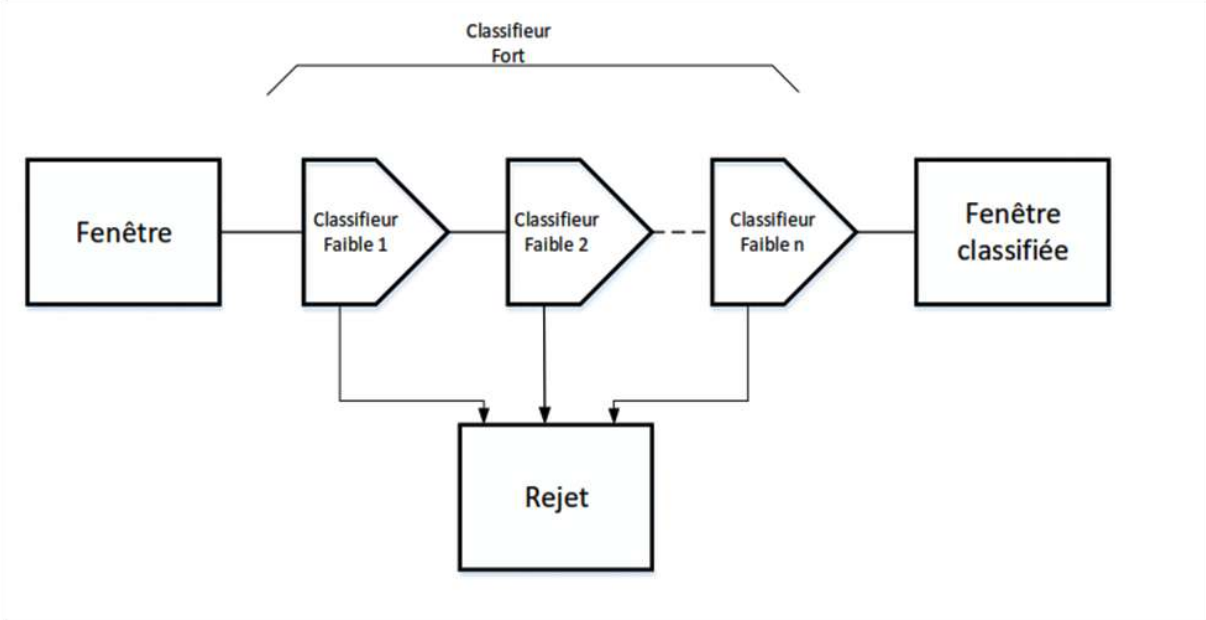


Figure IV.15 *Elaboration du classificateur*

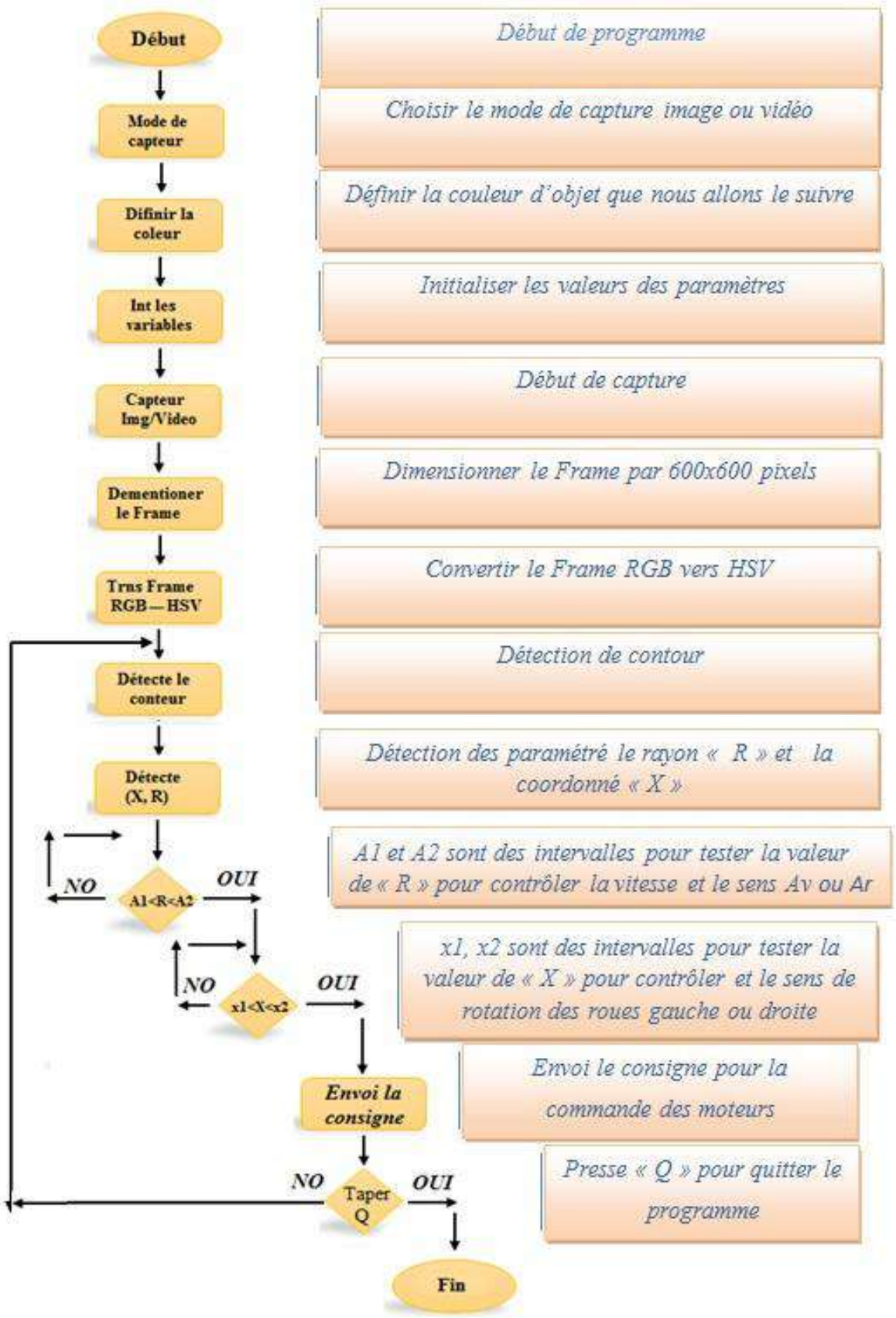
L'apprentissage du classificateur faible consiste alors à trouver la valeur du seuil de la caractéristique qui permet de mieux séparer les exemples positifs (image dans l'objet est présenté) des exemples négatifs (image dont l'objet est absent). Le classificateur se réduit alors à un couple (caractéristique, seuil). Au final, un classificateur est une association entre une caractéristiques pseudo-Haar et un seuil. C'est ce qu'on appelle un classificateur faible.

IV.4.2.2.4. La Détection

La détection s'applique sur une image de test, dans laquelle on souhaite déceler la présence et la localisation d'un objet. En voici les étapes :

- parcours de l'ensemble de l'image à toutes les positions, avec une fenêtre de taille 24 × 24 pixels et variable, application de la cascade à chaque sous-fenêtre, en commençant par le premier étage :
- calcul des caractéristiques pseudo-Haar de l'image utilisée par le classificateur de l'étage courant.
- calcul de la réponse du classificateur.
- passage ensuite à l'étage supérieur si la réponse est positive, à la sous-fenêtre suivante sinon, et enfin l'exemple est déclaré positif si tous les étages répondent positivement.
- fusion des détections multiples : l'objet peut en effet générer plusieurs détections, à différentes positions et échelles ; cette dernière étape fusionne les détections qui se chevauchent pour ne retourner qu'un seul résultat.

IV.4.3. L'organigramme



IV.4.4. Description du programme d'asservissement visuel

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 définissons le mode de capture : soit le mode image ou le mode vidéo. Par la suite, on va définir les limites inférieures et supérieures de la couleur de notre cible dans l'espace couleur HSV, ces limites vont nous permettre de détecter la couleur choisie dans la vidéo. Une fois cette partie d'initialisation des paramètres effectuée, nous commençons par exécuter la partie proprement dite du traitement d'images.

Cette étape cruciale commence par un pré-traitement en redimensionnant la taille du cadre (frame en anglais)qui est dans notre cas fixée à 600x600 pixels. A partir de la, une routine de la librairie openCV est appelée pour convertir les images successives du standard RGB vers le standard HSV. Ces fonctions sont basées sur les techniques de traitement d'images citées précédemment dans ce chapitre. Cette opération est le prélude pour la calcule du contour et du centre de gravité de la cible précédemment désignée par sa couleur.

Masque binaire résultant de l'opération précédente

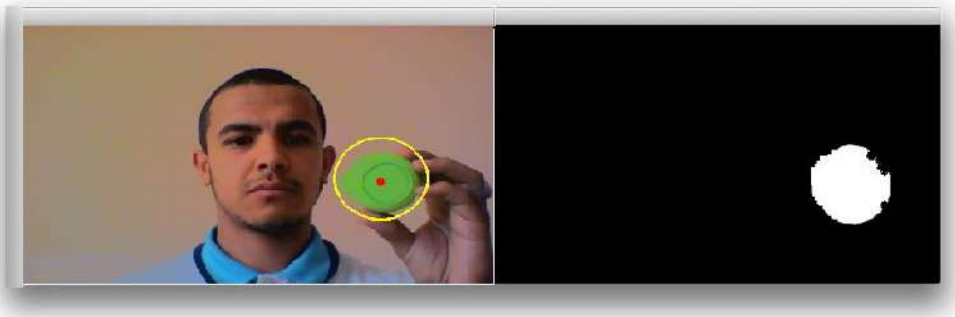


Figure IV.16 *Masque binaire de la balle verte*

A ce stade du déroulement de notre programme, nous disposons des caractéristiques visuelles de notre cible, à savoir le rayon R du contour de celle-ci et de sa position X dans l'image.

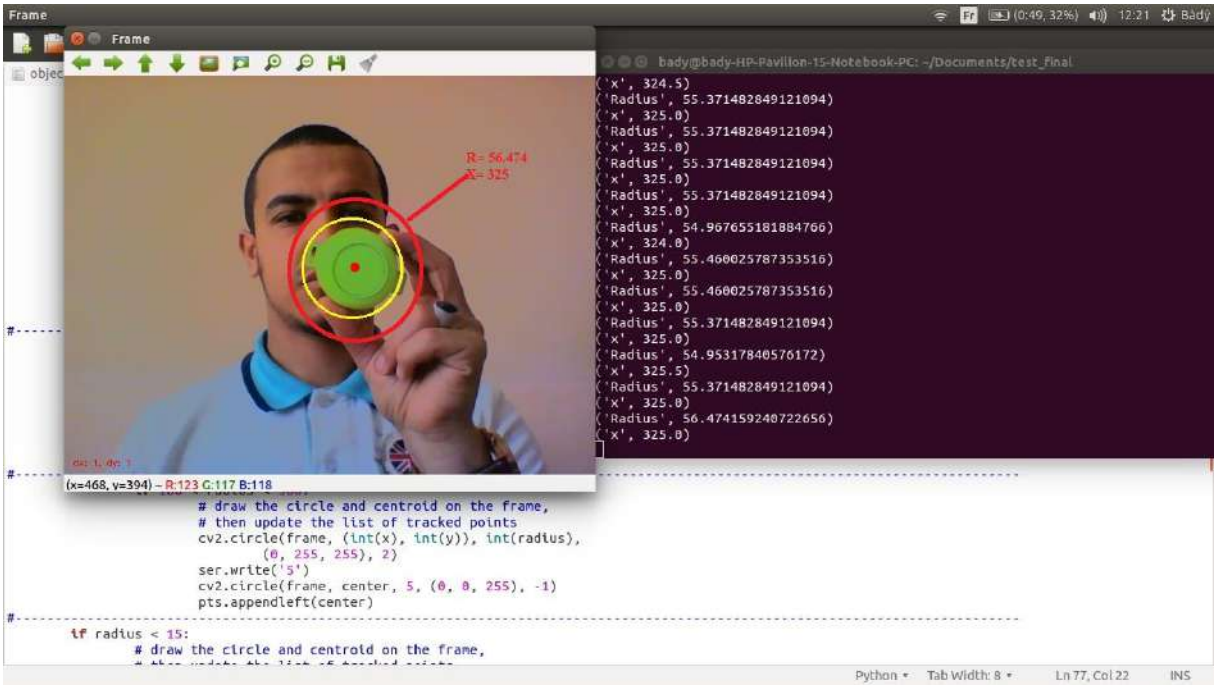


Figure IV.17 *Teste de détermination de rayon R et de X, cas 1*

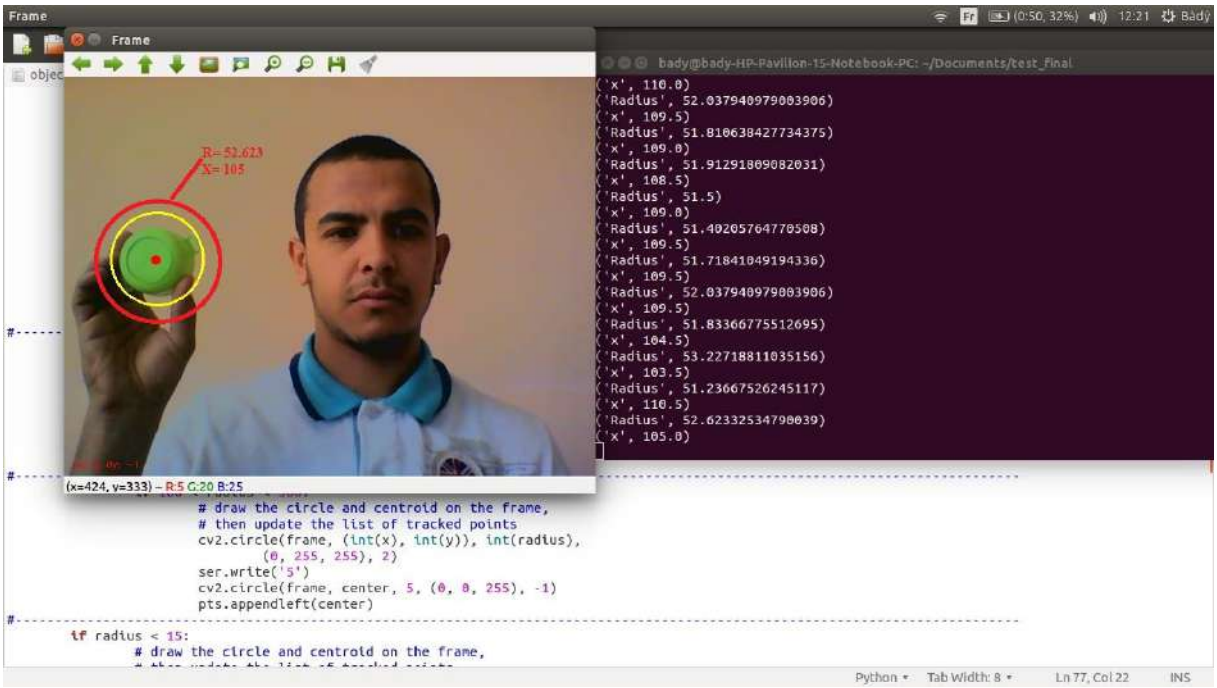


Figure IV.18 *Teste de détermination de rayon R et X, cas 2*

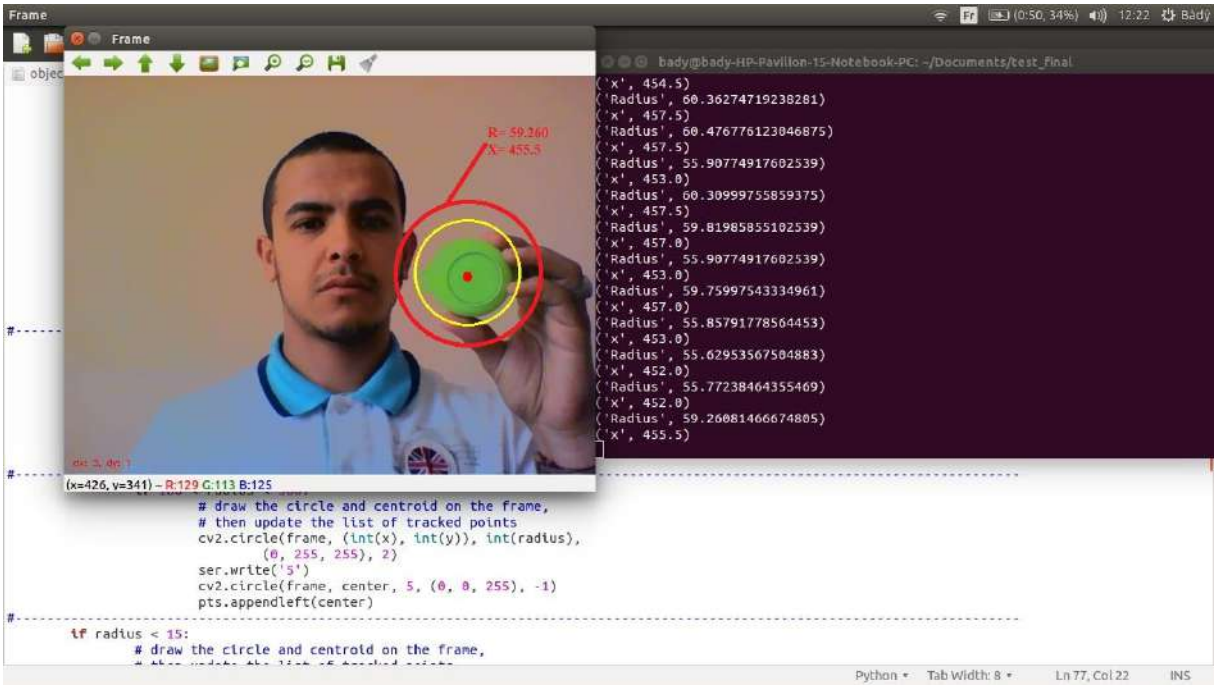


Figure IV.19 *Teste de détermination de rayon R et X, cas 3*

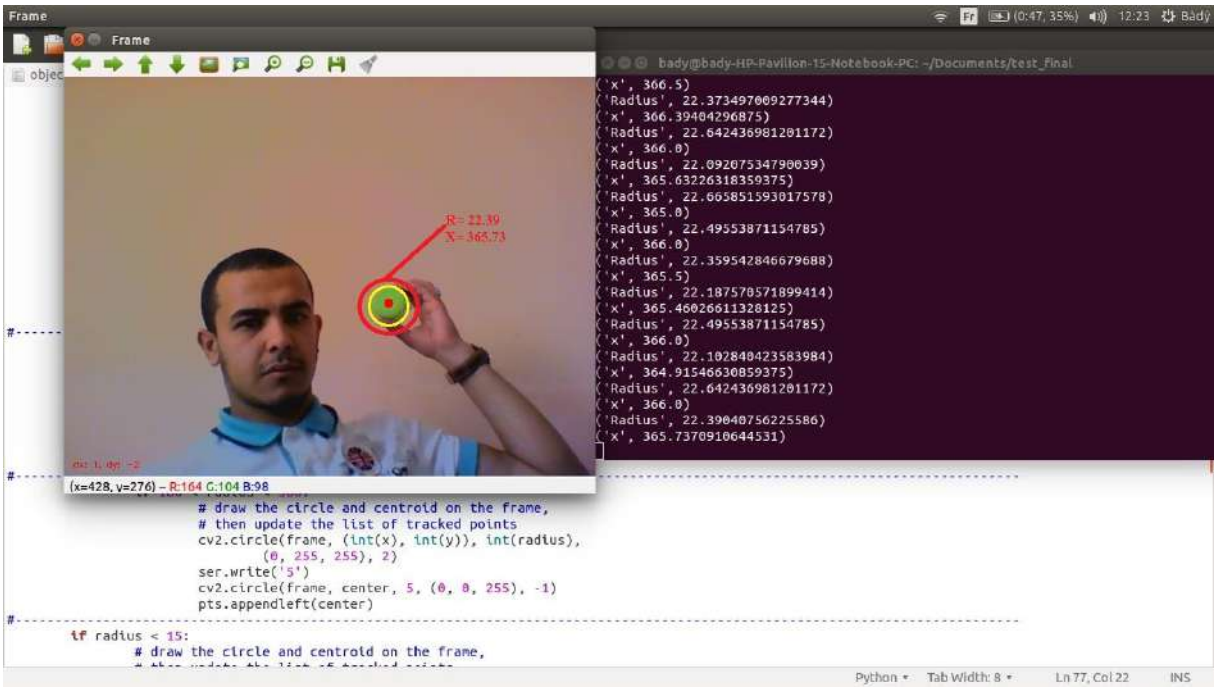


Figure IV.20 *Teste de détermination de rayon R et de X, cas 4*

A partir de ces deux données visuelles, nous pouvons commander les mouvements du robot pour maintenir la cible d'une part dans le centre de l'image (grâce à l'utilisation de la donnée X) et d'autre part maintenir la cible à une distance du robot prédéfinie (grâce à l'exploitation de la donnée R). En effet, plus la cible est

loin du robot plus le rayon du contour de cette cible est faible, donc le contrôleur fournira aux deux moteurs une vitesse et un sens de rotation pour augmenter, dans l'espace image, la valeur de R. Une fois la valeur de référence de R atteinte, le contrôleur génèrera des consignes d'arrêt aux moteurs du robot. A la cadence vidéo, ces consignes sont automatiquement calculées est transmises via le port USB à la carte Arduino. Celle-ci est munie d'une carte extension (Shield). Cet ensemble aura pour tache de traduire ces consignes en tensions nécessaires aux mouvements des moteurs du robot mobile.

Cet algorithme a été testé avec succès sur notre robot mobile. Plusieurs cibles, de couleurs différentes, ont été utilisées. Le robot a pu les suivre à chaque fois.

IV.4.5. Conclusion

Dans ce chapitre, nous avons présenté le fonctionnement détaillé de notre solution pour réaliser le suivi d'une cible à partir de données visuelles. Dans un premier temps, nous avons présenté l'agencement et l'organisation de notre système embraqué composé de la carte Raspberry Pi et de la carte Arduino. Celles-ci grâce à leur interconnexion, via le port USB, communiquent et s'échangent les données nécessaires à la commande du robot mobile.

Dans une seconde partie, nous avons détaillé le rôle de chaque étage. Le plus important est l'acquisition, le traitement des données visuelles et le calcule des lois de commande. Ces trois opérations ont été regroupées dans un seul programme écrit en Python et exécute sur la carte Raspberry. Les performances de cette carte ont permis la réalisation d'un contrôleurvisuel performant.

Enfin, l'application expérimentale que nous avons mis en œuvre à permis de valider d'une part, nos choix quant à l'utilisation de ces cartes embarquées et d'autre part notre algorithme de commande basée sur un schéma d'asservissement visuel fondé sur l'approche image. Bien entendu, d'autres testes sont à prévoir pour étudier de plus prêt 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 plate-forme évolutive, puissante et embraquée pour asservir un robot mobile à partir de données visuelles.

Conclusion générale

Au terme de ce travail, nous avons conçu et réalisé une architecture matérielle évolutive, puissante que nous avons pu embarquée sur la structure mécanique du robot mobile. Ces caractéristiques ont pu être obtenu grâce à notre utilisation des cartes électronique Raspberry Pi et Arduino. En effet, la combinaison de ces deux cartes nous a permis de développer une plate-forme capable de commander tout type de robot tricycle à condition d'avoir accès aux entrées de ces moteurs qu'il soient à courant continu, pas à pas ou de type servomoteur. De plus, cette plate-forme pourra s'adapter voir être modifiée facilement en fonction de la tache robotique à réalisée en rajoutant ou en retirant des capteurs à la carte Arduino. En effet, l'atout majeur de cette carte est la simplicité avec laquelle elle peut gérer les données fournies par divers capteurs (ultrason, infrarouge, luminosité,…).

Par ailleurs, notre utilisation de la carte Raspeberry dans l'asservissement visuel de notre robot mobile à permis de démontrer ses performances en terme de puissance de calcule, de mémoire et de connectivité (USB, I2C, WIFI,…).

D'autre part, l'application expérimentale que nous avons mis en œuvre nous a permis de valider le fonctionnement de cette plate-forme mais aussi la validation de l'algorithme de commande que nous avons élaboré. En effet, nous avons mis en œuvre un contrôleur capable d'asservir les mouvements d'un robot mobile uniquement à partir d'informations visuelles. L'utilisation d'une seule caméra ne permet pas directement d'estimer la position de la cible en profondeur. Nous avons levé cette limitation, en proposant une solution qui consiste à utiliser la largeur du contour de la cible pour estimer cette donnée.

Eu égard des résultats obtenus lors de nos divers testes, nous avons pu valider le fonctionnement de l'ensemble, algorithme et plate-forme matérielle.

Enfin, ce travail nous a permis d'aborder les différentes problématiques inhérentes à la robotique mobile. Mais surtout, de nous initier à l'utilisation de cartes électroniques embarquées, à la programmation en langage Python et aux techniques de traitements d'images à travers l'utilisation de la bibliothèque graphique OpenCV.

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

- L'utilisation de capteurs ultrason pour détecter les obstacles. Mais ceci nécessitera l'élaboration de lois de commande plus complexes.
- La conception d'une tète robotique motorisée capable de supporter un système de vision plus performant (caméra 3D, Vison binoculaire).
- Remplacer la carte Raspberry par une autre carte plus performante encore (carte FPGA par exemple).

Références bibliographiques

[1] H. Hadj Abdelkader1, Y. Mezouar1, P. Martinet1, F. Chaumette2, Asservissement visuel en vision omnidirectionnelle à partir de droites* Visual Servoing from straight lines, janvier 2005 page 470

[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

[3] A. Louchene, *système de navigation pour robot mobile,* thèse de doctorat, de l'université de Batna 2004. Mazen Al Homsi

[4] T. Kanade, C. Thorpe et W. *Whittaker. Autonomous Land Vehicle Project at CMU*. In ACM Annual Computer Science Conference, pages 71–80. ACM Press, 1986.

[5] J. Gangloff. Cours de robotique, 2004.
http ://eavr.u-strasbg.fr.

[6] Peter H. Dana. Global Positioning System Overview, 2001.
http ://www.colorado.edu/geography/gcraft/notes/gps/gps f.html.

[7] ESA. Galileo, Système européen de navigation par satellite, 2004.
http ://europa.eu.int/comm/dgs/energy transport/galileo/index fr.htm.

[8] INRP. Inventaire des Instruments Scientifiques Anciens dans les Etablissements Publics : Gyroscopes, 2004.
http ://www.inrp.fr/she/instruments/instr mec gyroscope.htm.

[9] Christian Dupaty, raspberry pi, installation-configuration interfaces de communications, BTS Systèmes Numérique, lycée Fourcade- Gardanne.

[10] https://fr.wikipedia.org/wiki/Raspberry_Pi

[11] Christian Tavernier, raspberry pi : prise en main et premières réalisations, DUNOD, à la découverte du raspberry pi

[12] Christian Tavernier, raspberry pi : prise en main et premières réalisations, DUNOD, à la découverte de Raspbian

[13] J. Hill and W. T. Park, "Real time control of a robot with a mobile camera," presented at proceedings of the 9th ISIR, Washington D.C., USA, 1979.

[14] William JACQUET, PARADIGME CONNEXIONNISTE POUR L'APPRENTISSAGE DE FONCTIONS ROBOTIQUES : APPLICATION AUX PROBLEMES D'ASSERVISSEMENT VISUEL DE BRAS MANIPULATEURS, Mars 1992, page 21, 22

[15] Alexandre Krupa, Contributions à l'asservissement visuel échographique, L'Université de Rennes 1 Spécialité : Traitement du signal, décembre 2012, page 37

[16] Ryad CHELLAL, ASSERVISSEMENT VISUEL SUR HOAP-3 Université de Montpellier 2, Septembre 2010, page 7

[17] Ryad CHELLAL, ASSERVISSEMENT VISUEL SUR HOAP-3 Université de Montpellier 2, Septembre 2010, page 8

[18] S. Baker et S. K. Nayar. A theory of single-viewpoint catadioptric image formation. *Int. Journal of Computer Vision*, 35(2) :1–22, Novembre 1999.

[19] H. Hadj Abdelkader1, Y. Mezouar1, P. Martinet1, F. Chaumette2, Asservissement visuel en vision omnidirectionnelle à partir de droites* Visual Servoing from straight lines, janvier 2005 page 471, 472

Raspberry Pi



Raspberry Pi 2, Model B

Product Name	**Raspberry Pi 2, Model B**
Product Description	The Raspberry Pi 2 delivers 6 times the processing capacity of previous models. This second generation Raspberry Pi has an upgraded Broadcom BCM2836 processor, which is a powerful ARM Cortex-A7 based quad-core processor that runs at 900MHz. The board also features an increase in memory capacity to 1Gbyte.
RS Part Number	**832-6274**

Specifications

Chip	Broadcom BCM2836 SoC
Core architecture	Quad-core ARM Cortex-A7
CPU	900 MHz
GPU	Dual Core VideoCore IV® Multimedia Co-Processor Provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30 H.264 high-profile decode Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure
Memory	1GB LPDDR2
Operating System	Boots from Micro SD card, running a version of the Linux operating system
Dimensions	85 x 56 x 17mm
Power	Micro USB socket 5V, 2A

Connectors:

Ethernet	10/100 BaseT Ethernet socket
Video Output	HDMI (rev 1.3 & 1.4) Composite RCA (PAL and NTSC)
Audio Output	3.5mm jack, HDMI
USB	4 x USB 2.0 Connector
GPIO Connector	40-pin 2.54 mm (100 mil) expansion header: 2x20 strip Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines
Camera Connector	15-pin MIPI Camera Serial Interface (CSI-2)
JTAG	Not populated
Display Connector	Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane
Memory Card Slot	Micro SDIO

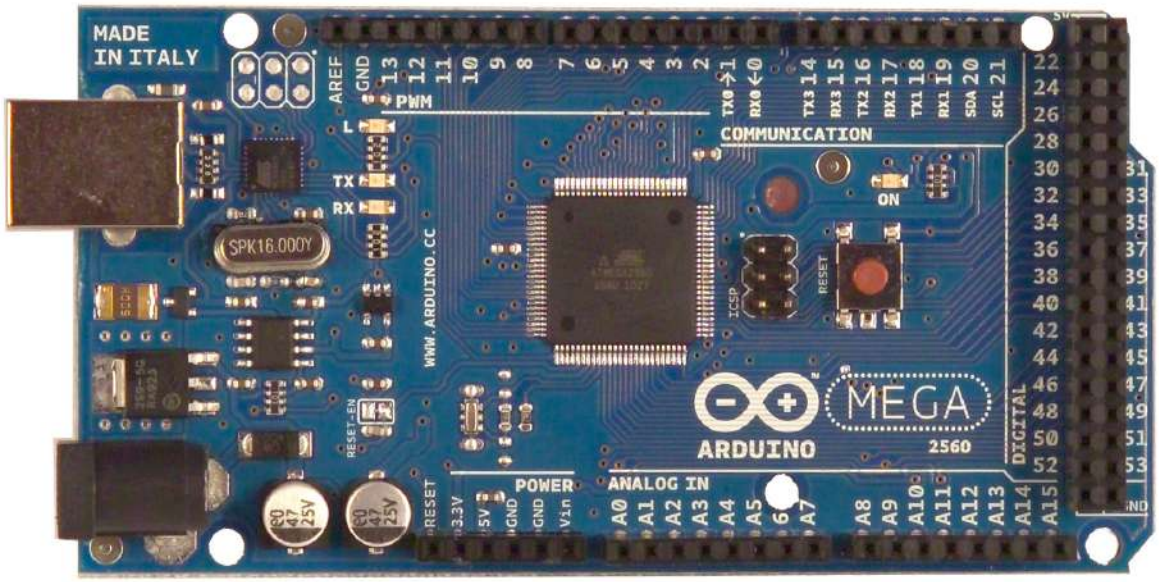


www.robotshop.com

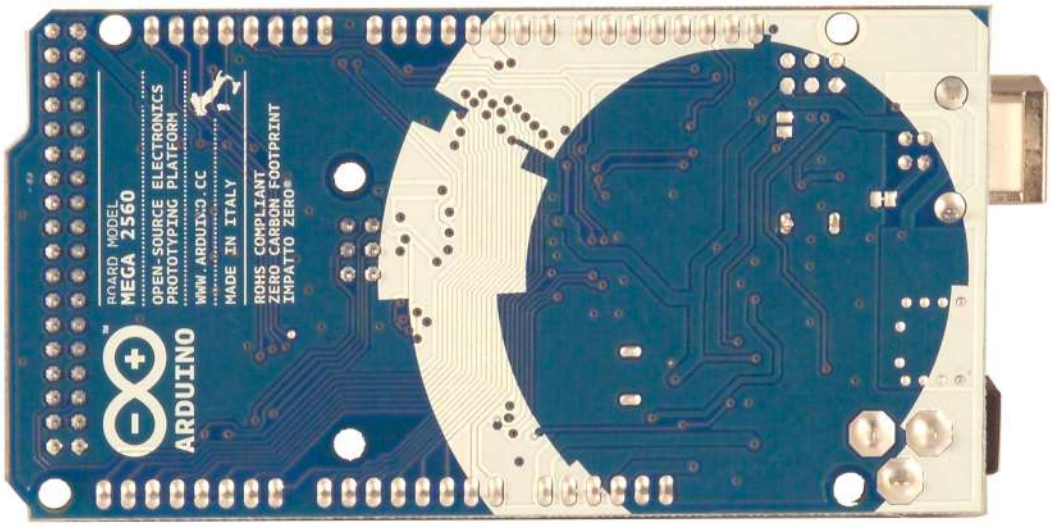
La robotique à votre service! - Robotics at your service!

ARDUINO

Arduino Mega 2560 Datasheet



RobotShop
www.robotshop.com
La robotique à votre service! - Robotics at your service!



Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Schematic & Reference Design

EAGLE files: arduino-mega2560-reference-design.zip



Schematic: [arduino-mega2560-schematic.pdf]

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode() , digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH



value, the LED is on, when the pin is LOW, it's off.
- **I2C: 20 (SDA) and 21 (SCL).** Support I2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I2C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).
A SoftwareSerial library allows for serial communication on any of the Mega2560's digital pins.
The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. For SPI communication, use the SPI library.

Programming

The Arduino Mega can be programmed with the Arduino software (download). For details, see the reference and tutorials.
The ATmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It

communicates using the original STK500 protocol (reference, C header files).
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.
This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.
The Mega2560 contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility



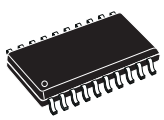
The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

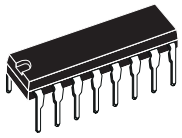
The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. *Please note that I^2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).*

L293D
L293DD

PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES

- 600mA OUTPUT CURRENT CAPABILITY PER CHANNEL
- 1.2A PEAK OUTPUT CURRENT (non repetitive) PER CHANNEL
- ENABLE FACILITY
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)
- INTERNAL CLAMP DIODES





SO(12+4+4) **Powerdip (12+2+2)**

ORDERING NUMBERS:

L293DD L293D

DESCRIPTION

The Device is a monolithic integrated high voltage, high current four channel driver designed to accept standard DTL or TTL logic levels and drive inductive loads (such as relays solenoides, DC and stepping motors) and switching power transistors.

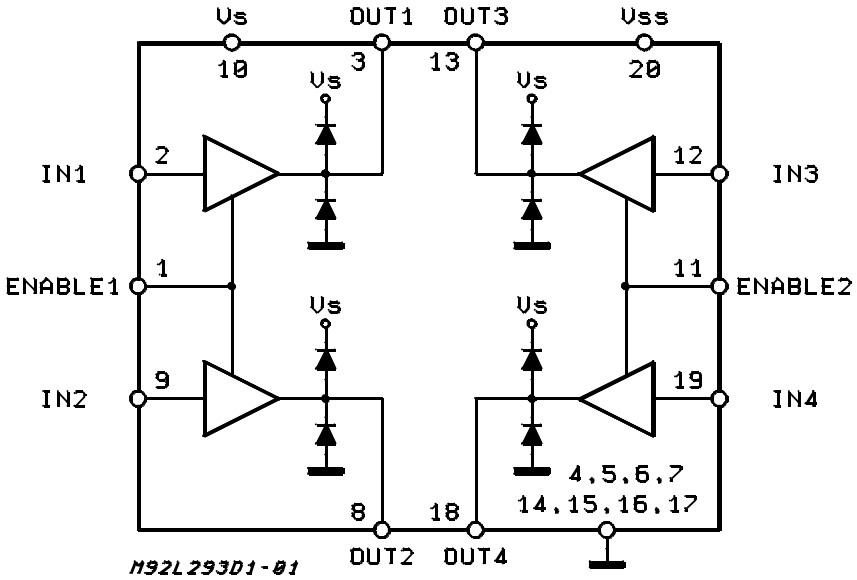
To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included.

This device is suitable for use in switching applications at frequencies up to 5 kHz.

The L293D is assembled in a 16 lead plastic packaage which has 4 center pins connected together and used for heatsinking

The L293DD is assembled in a 20 lead surface mount which has 8 center pins connected together and used for heatsinking.

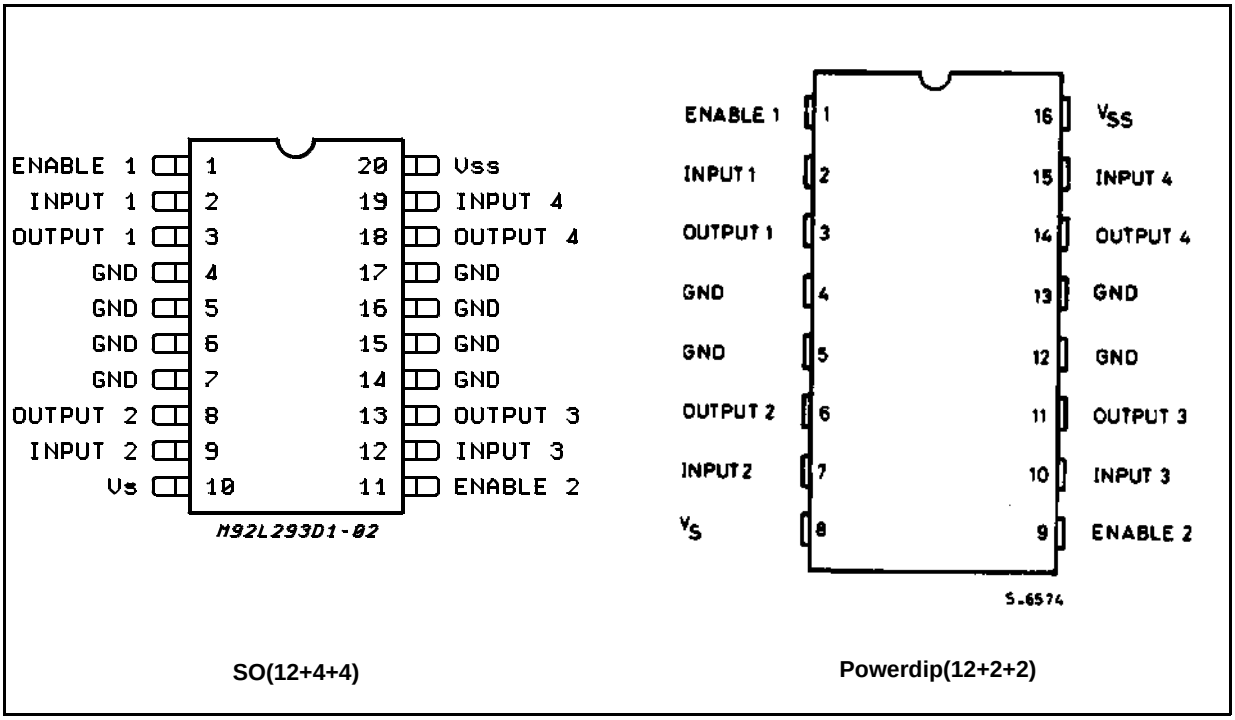
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Supply Voltage	36	V
V_{SS}	Logic Supply Voltage	36	V
V_i	Input Voltage	7	V
V_{en}	Enable Voltage	7	V
I_o	Peak Output Current (100 µs non repetitive)	1.2	A
P_{tot}	Total Power Dissipation at T_{pins} = 90 °C	4	W
T_{stg}, T_j	Storage and Junction Temperature	– 40 to 150	°C

PIN CONNECTIONS (Top view)



SO(12+4+4) **Powerdip(12+2+2)**

THERMAL DATA

Symbol	Decription		DIP	SO	Unit
$R_{th\ j\text{-}pins}$	Thermal Resistance Junction-pins	max.	–	14	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance junction-ambient	max.	80	50 (*)	°C/W
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	max.	14	–	

(*) With 6sq. cm on board heatsink.

ELECTRICAL CHARACTERISTICS (for each channel, V_S = 24 V, V_{SS} = 5 V, T_{amb} = 25 °C, unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 10)		V_{SS}		36	V
V_{SS}	Logic Supply Voltage (pin 20)		4.5		36	V
I_S	Total Quiescent Supply Current (pin 10)	V_i = L ; I_O = 0 ; V_{en} = H		2	6	mA
		V_i = H ; I_O = 0 ; V_{en} = H		16	24	mA
		V_{en} = L			4	mA
I_{SS}	Total Quiescent Logic Supply Current (pin 20)	V_i = L ; I_O = 0 ; V_{en} = H		44	60	mA
		V_i = H ; I_O = 0 ; V_{en} = H		16	22	mA
		V_{en} = L		16	24	mA
V_{IL}	Input Low Voltage (pin 2, 9, 12, 19)		– 0.3		1.5	V
V_{IH}	Input High Voltage (pin 2, 9, 12, 19)	$V_{SS} \leq 7$ V	2.3		V_{SS}	V
		$V_{SS} > 7$ V	2.3		7	V
I_{IL}	Low Voltage Input Current (pin 2, 9, 12, 19)	V_{IL} = 1.5 V			– 10	µA
I_{IH}	High Voltage Input Current (pin 2, 9, 12, 19)	2.3 V ≤ V_{IH} ≤ V_{SS} – 0.6 V		30	100	µA
$V_{en\ L}$	Enable Low Voltage (pin 1, 11)		– 0.3		1.5	V
$V_{en\ H}$	Enable High Voltage (pin 1, 11)	$V_{SS} \leq 7$ V	2.3		V_{SS}	V
		$V_{SS} > 7$ V	2.3		7	V
$I_{en\ L}$	Low Voltage Enable Current (pin 1, 11)	$V_{en\ L}$ = 1.5 V		– 30	– 100	µA
$I_{en\ H}$	High Voltage Enable Current (pin 1, 11)	2.3 V ≤ $V_{en\ H}$ ≤ V_{SS} – 0.6 V			± 10	µA
$V_{CE(sat)H}$	Source Output Saturation Voltage (pins 3, 8, 13, 18)	I_O = – 0.6 A		1.4	1.8	V
$V_{CE(sat)L}$	Sink Output Saturation Voltage (pins 3, 8, 13, 18)	I_O = + 0.6 A		1.2	1.8	V
V_F	Clamp Diode Forward Voltage	I_O = 600nA		1.3		V
t_r	Rise Time (*)	0.1 to 0.9 V_O		250		ns
t_f	Fall Time (*)	0.9 to 0.1 V_O		250		ns
t_{on}	Turn-on Delay (*)	0.5 V_i to 0.5 V_O		750		ns
t_{off}	Turn-off Delay (*)	0.5 V_i to 0.5 V_O		200		ns

(*) See fig. 1.

TRUTH TABLE (one channel)

Input	Enable (*)	Output
H	H	H
L	H	L
H	L	Z
L	L	Z

Z = High output impedance
(*) Relative to the considered channel

Figure 1: Switching Times

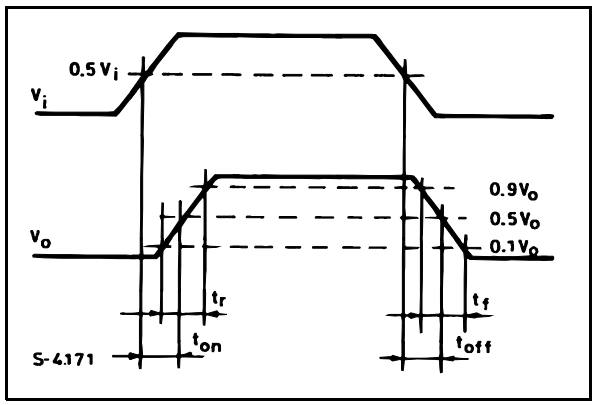
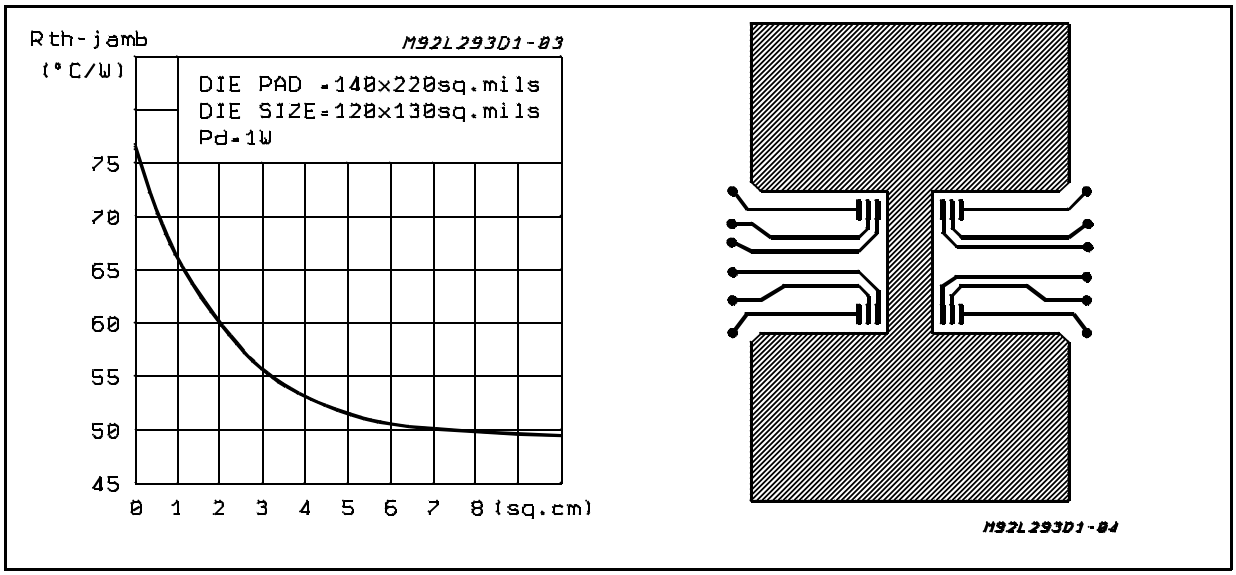
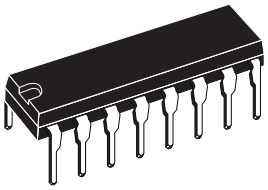


Figure 2: Junction to ambient thermal resistance vs. area on board heatsink (SO12+4+4 package)

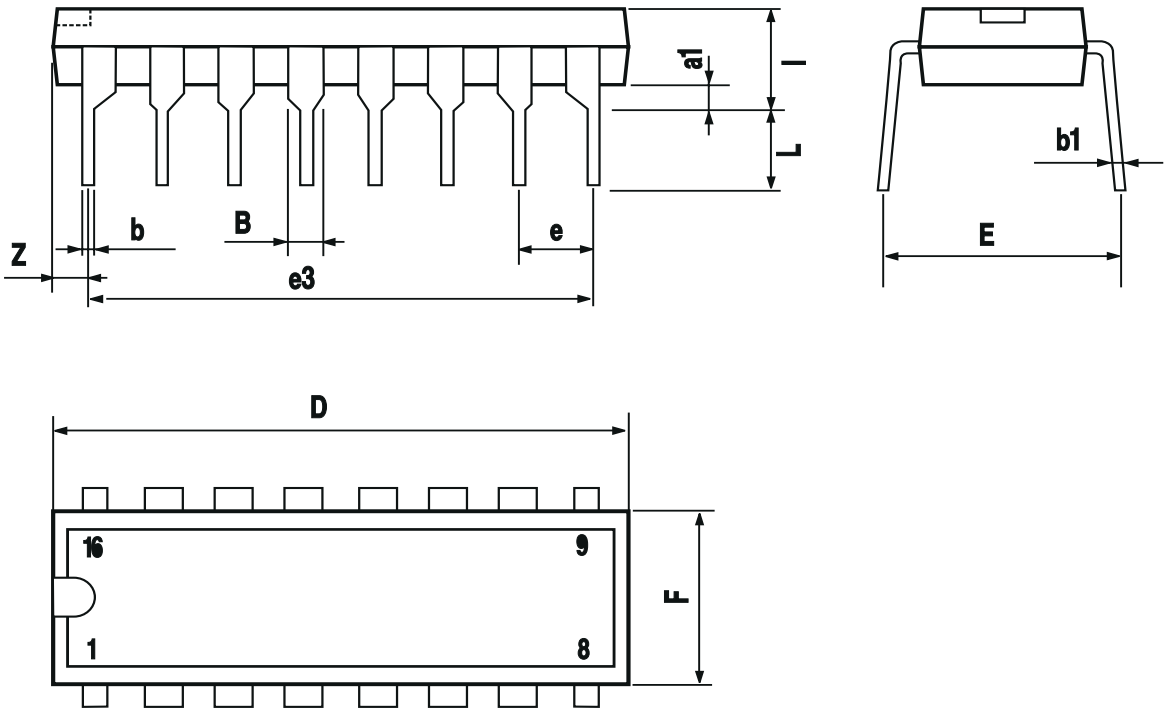


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
a1	0.51			0.020		
B	0.85		1.40	0.033		0.055
b		0.50			0.020	
b1	0.38		0.50	0.015		0.020
D			20.0			0.787
E		8.80			0.346	
e		2.54			0.100	
e3		17.78			0.700	
F			7.10			0.280
I			5.10			0.201
L		3.30			0.130	
Z			1.27			0.050

OUTLINE AND MECHANICAL DATA

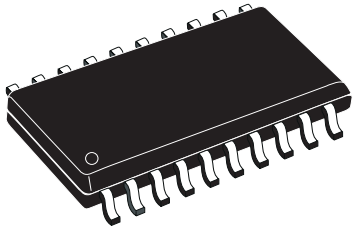


Powerdip 16

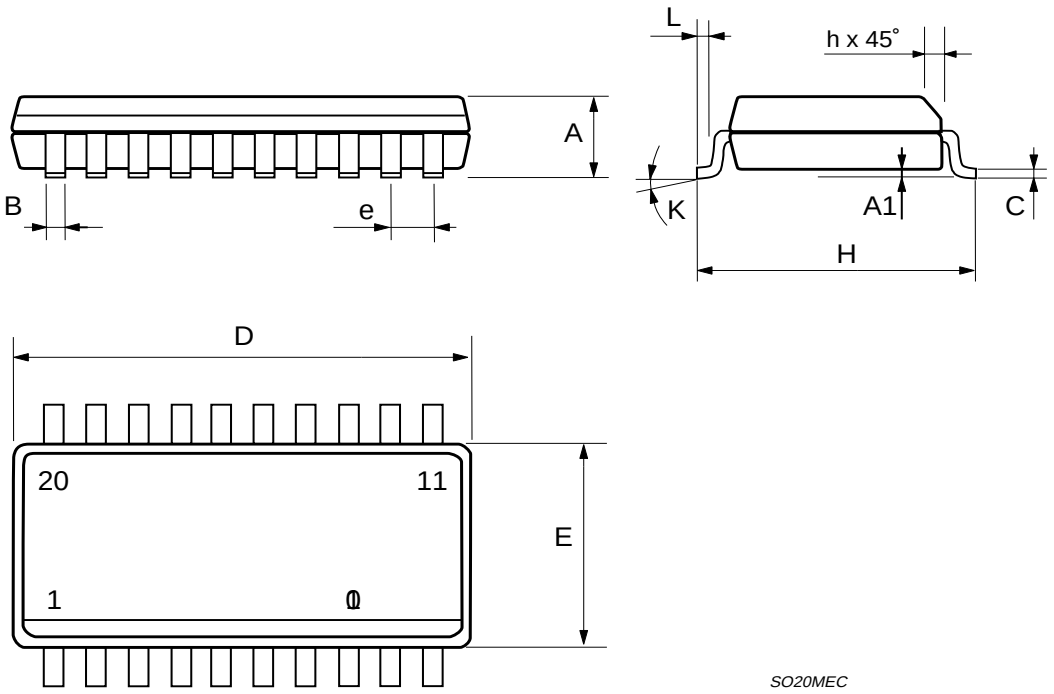


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A	2.35		2.65	0.093		0.104
A1	0.1		0.3	0.004		0.012
B	0.33		0.51	0.013		0.020
C	0.23		0.32	0.009		0.013
D	12.6		13	0.496		0.512
E	7.4		7.6	0.291		0.299
e		1.27			0.050	
H	10		10.65	0.394		0.419
h	0.25		0.75	0.010		0.030
L	0.4		1.27	0.016		0.050
K	0˚ (min.)8˚ (max.)					

OUTLINE AND MECHANICAL DATA



SO20



SO20MEC

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specification mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics

© 2003 STMicroelectronics – Printed in Italy – All Rights Reserved

STMicroelectronics GROUP OF COMPANIES

Australia - Brazil - Canada - China - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States.

http://www.st.com