الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

Mention Électronique
Spécialité : Systèmesde Vision et de Robotiques

présenté par

Ben Messaoud Mahrez

&

Sediki laid

Conception d'un robot mobile muni d'une tourelle asservis par vision artificielle

Proposé par : KAZED Boualem

Année Universitaire 2017-2018

1.1. Historique

Durant plusieurs millénaires, les automates ont reposé sur des ressorts, engrenages et autres mécanismes, ce qui ne les a pas empêchés d'atteindre une grâce qui force l'admiration. Et puis l'informatique est venue changer la donne en permettant de stocker de très larges quantités d'informations et de séquences d'actions dans une petite puce. Le robot est ainsi arrivé à une sophistication telle qu'il peut désormais tenter de trouver par lui-même la solution de certains problèmes.

L'histoire des robots a ainsi traversé les étapes suivantes :

a. Les masques et statues animés de l'antiquité

L'origine des masques et statues animés remonte à l'Egypte ancienne où l'on a recensé un masque à l'effigie de Thot (tête d'Ibis) ou d'Horus (tête de faucon) qui pareillement semblent doués de mouvement. Ce qui caractérise ces divers artefacts, c'est que l'automatisme y est caché, mis à profit par des castes religieuses pour assurer leur pouvoir sur le peuple comme sur les souverains.

b. L'horloge

C'est en 246 avant J.C. que nous trouvons la trace du premier inventeur d'envergure, un dénommé Ctésibios qui habite la ville d'Alexandrie. Ctésibios est parvenu à créer une horloge si précise que son cadran fait exactement un tour par année solaire ! Pour la première fois, il existe une parfaite concordance entre un instrument de mesure humain et un phénomène issu du monde physiqueextérieur.

Figure 1.1 L'horloge de Ctésibios

c. Des automates de l'orient aux jacquemarts européens

Les arabes sont les premiers à mettre en pratique à une grande échelle les techniques décrites par le mathématicien et mécanicien grec Héron d'Alexandrie (et aussi par Phylon de Byzance). Dès 809, Charlemagne reçoit de la part du sultan Haroun Al Rachid un automate mécanique

Puis, lors des huit expéditions en Orient menées à l'occasion des Croisades - de 1096 à 1291 - les européens découvrent de visu l'étonnant raffinement des horloges à eau réalisées par Al Jazari pour le compte de ce même Haroun Al Rachid.

Pour obtenir un écoulement constant de l'eau, Al Jazari a développé un système d'une rare ingéniosité, inspiré d'un système inventé par Archimède. La plus grande de ses horloges mesure 3,3 mètres de hauteur et 1,35 mètre de larger

d. Vers l'âge d'or des automates

Le 18ème siècle apparaît comme l'âge d'or des automates. L'un des grands inventeurs d'engins mécaniques de l'époque est le protégé du roi Louis XV, Jacques de Vaucanson (1709 - 1792). Il développe un "canard mécanique" qui force l'admiration. Celui ci "allonge le cou pour aller prendre le grain dans la main, l'avale, le digère " Après avoir transformé l'aliment en bouillie, il le rejette par les voies ordinaires, pleinement digéré. Les créations que réalise Vaucanson tel le joueur de flûte qui exécute onze airs différents et aussi celle de ses disciples séduisent l'Europe entière et s'exportent aux Etats-Unis.

e. L'ordinateur, potentielle intelligence du robot ?

Niels Bohr a décrit dans ses travaux publiés vers 1913 que l'électron peut déplacer d'un atome à l'autre une vitesse vertigineuse. D'où l'idée de créer des circuits exploitant cette incroyable mobilité. En 1937, Turing énonce les principes d'une machine qui calculerait à la vitesse de l'électronique, et serait donc capable de traiter d'énormes volumes d'informations codées sous la forme booléenne (0 et 1).L'arrivée des ordinateurs est appelée à jouer un rôle majeur dans l'élaboration des machines intelligentes que sont les robots.

Sous l'impulsion de Turing, un premier ordinateur apparaît en 1943. Sa puissance de calcul est mise à contribution dans la guerre et joue un rôle décisif en facilitant le décryptage du code Enigma mis au point par les nazis pour leurs échanges de messages.

1.2. Robot mobile

1.2.1. Definition

Écrivain tchèque, Karel Capek, dans son drame, introduit le mot robot au monde en 1921. Il est dérivé du mot tchèque robota qui signifie "travailleur forcé"

Un "robot" est une machine pouvant manipuler des objets en réalisant des mouvements variés dictés par un programme aisément modifiable.

Programmer un robot consiste, dans un premier temps, à lui spécifier la séquence des mouvements qu'il devra réaliser.

1.2.2. l'architecture globale d'un robot mobile

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

a. La structure mécanique et la motricité
b. Les organes de sécurité
c. Le système de traitement des informations et gestion des tâches

1.3. structure mécanique et la motricité

1.3.1. robots à roues

Il existe plusieurs classes de robots à roues déterminées, principalement, par la position et le nombre de roues utilisées.
Nous citerons ici les quatre classes principales de robots à roues.

1.3.2. Robot unicycle

Un robot de type unicycle est actionné par deux roues indépendantes, il possède éventuellement des roues folles pour assurer sa stabilité. Son centre de rotation est situé sur l'axe reliant les deux roues motrices.
C'est un robot non-holonome, en effét il est impossible de le déplace dans une direction perpendiculaire aux roues de locomotion.
Sa commande peut etre trés simple, il est en effét assez facile de le déplacer d'un point à un autre par une suite de rotations simples et de lignes droites.

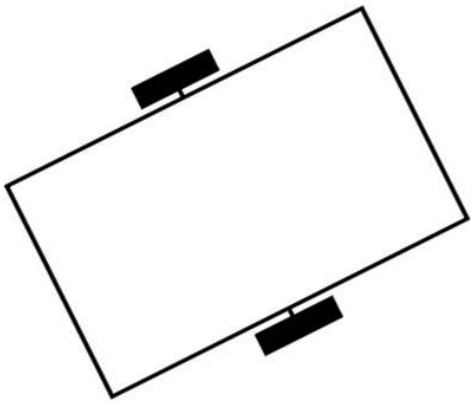
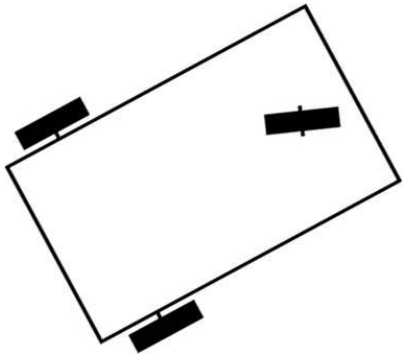


Figure 1.2 : Robot de type unicycle

1.3.3. Robot tricycle

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



. **Figure 1.3 : Robot de type tricycle**

1.3.4. Robot voiture

Un robot de type voiture est semblable au tricycle, il est constitué de deux roues fixes placées sur un meme axe et de deux roues centrées orientables placées elles aussi sur un meme axe. Le robot de type voiture est cependant plus stable puisqu'il possède un point d'appui supplémentaire.

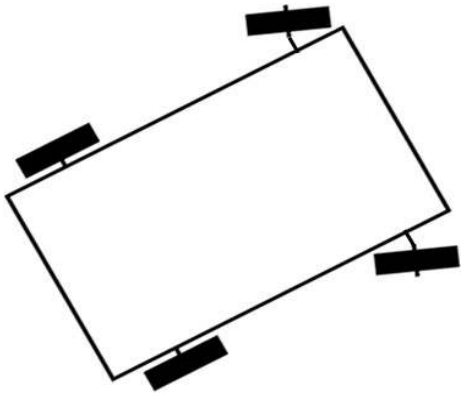


Figure 1.4 : Robot de type voiture

1.3.5. Robot omnidirectionnel

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

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

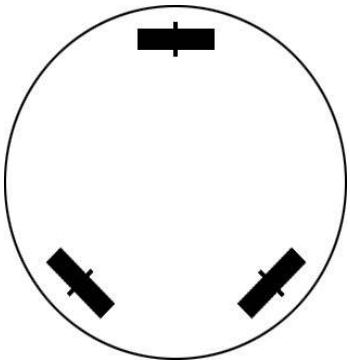


Figure 1.5: Robot de type omnidirectionnel

1.3.6. Comparaison des différents types

Nous pouvons observer dans le tableau ci-dessous un récapitulatif des avantages et des inconvénients des différents types de robots à roues.

Robot unicycle	- non-holonome + stable + rotation sur soi-meme + complexité mécanique faible
Robot tricycle	- non-holonome - peu stable - pas de rotation sur soi-meme + complexité mécanique modérée
Robot voiture	- non-holonome + stable - pas de rotation sur soi-meme +complexité mécanique modérée
Robot omnidirectionnel	+ holonome + stable + rotation sur soi-meme - complexité mécanique modérée

Tableau1.1 : Comparaison des différents types de robot à roue

1.3.7. Les robots à chenilles

Les robots avec des chenilles sont pratiques pour de nombreuses fonctions comme la capacité à tondre au milieu de toutes sortes d'obstacles, comme des rochers, des fossés et des nids de poule. Compte tenu de la matière des chenilles, les robots à chenilles peuvent se déplacer sur des surfaces glissantes comme la neige, le béton humide, ou un carrelage de cuisine propre.

Figure 1.6: Robot a chenille

1.3.8. Les robots à pattes

Le robot à pattes se déplace la plupart du temps selon une marche qui peut être quasi-statique ou dynamique. Il devient dès lors '**robot marcheur**'. Il n'en demeure pas moins que certains robots à pattes se déplacent sous la forme de 'sauts'

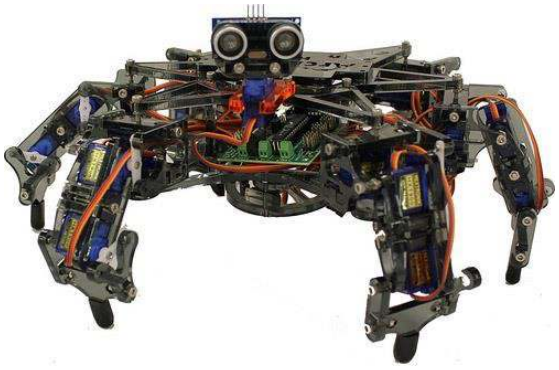


Figure1.7.a : Robot marcheur quasi-statique

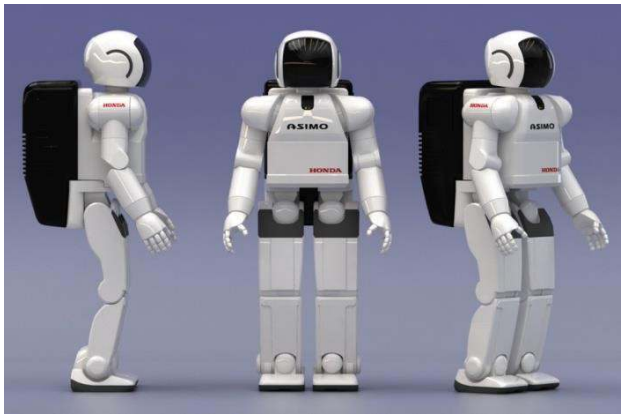


figure 1.7.b :Robot marcheur dynamique

1.3.9. Les robots rampant

On peut citer tout d'abord un robot dont le déplacement copie celui des amibes qui font circuler des fluides à l'intérieur de leur corps. L'appareil créé par l'homme est doté d'anneaux se contractant et se décontractant provocant le déplacement sans pour autant être une vrai reptation

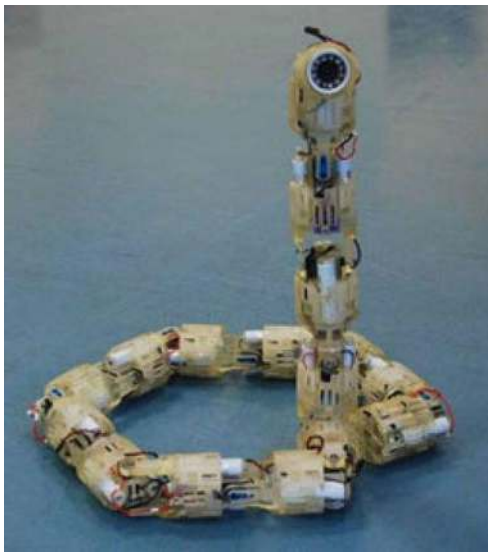


Figure 1.8:Robot rampant

1.4. Système 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 auteur du 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…). Ce sont des dispositifs redondants par rapport aux capteurs précédents. L'organisation de la sécurité est représentée sur le schéma de la Figure I.9.

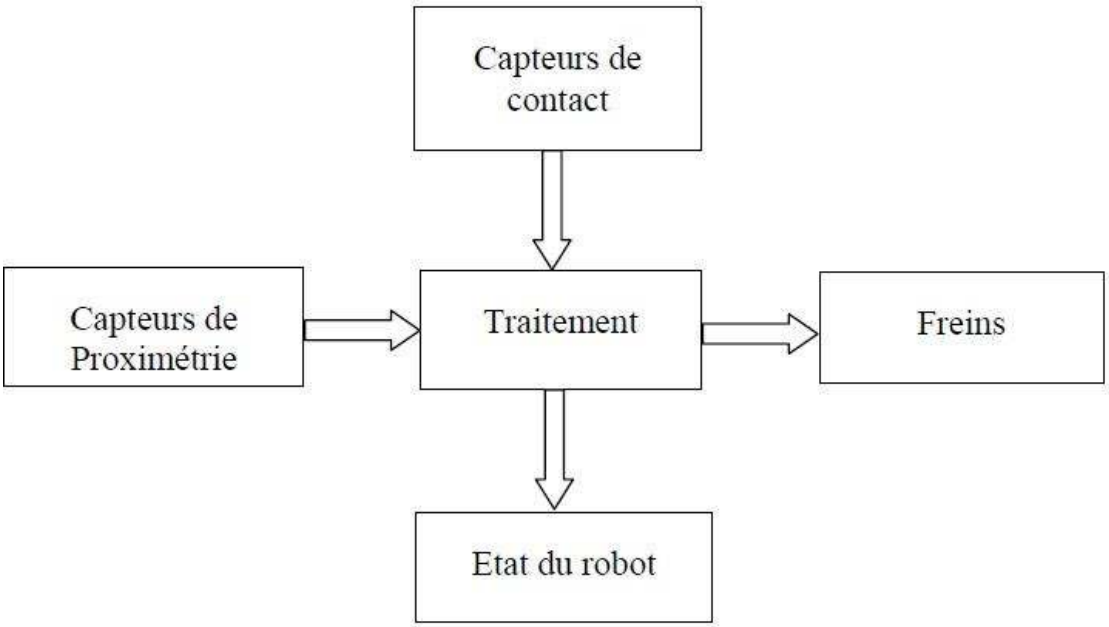


Figure I.9 : Synoptique de la sécurité

1.5. 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. Nous nous limiterons au problème de génération de plan qui consiste à établir la manière dont le robot se déplace par rapport à des connaissances à priori (statiques) ou obtenues en cours d'évolution (dynamiques). La génération de plus repose sur trois concepts :

- La stratégie denavigation
- La modélisation del'espace
- La planification.

1.6. Applications

On cite ici quelque exemple des applications des robots mobiles :

Les robots industriels

sont des robots utilisés dans un environnement de fabrication industrielle. Ils sont utilisés dans la fabrication des automobiles, des composants et des pièces électroniques, des médicaments et de nombreux produits

Figure I.10 : Robot industriel (bras manipulateur)

1.7. Robots domestiques ou ménagers

Robots utilisés à la maison. Ce type de robots comprend de nombreux appareils très différents, tels que les aspirateurs robotiques, robots nettoyeurs de piscines, balayeuses, nettoyeurs gouttières et autres robots qui peuvent faire différentes tâches. En outre, certains robots de surveillance et de téléprésence pouvaient être considérées comme des robots ménagers se il est utilisé dans cet environnement.

Figure I.11 : Robot domestique (aspirateur)

1.8. Robots en médecine et chirurgie

Les robots semblent avoir de l'avenir à l'hôpital. Robodoc aide à réaliser certaines opérations de chirurgie. Le robot infirmier est encore en projet. Le cybersquelette HAL aide les personnes à se déplacer. Et le robot patient permet aux futurs chirurgiens dentistes d'apprendre à soigner sans faire de dégâts...

Figure I.12 : Robot medical (Da Vinci)

1.9. Avantages

1.9.1. Calculatoire

L'avantage premier a l'IA est la limitation des erreurs de calcul. En effet grâce à tous les algorithmes il est plus facile d'utiliser un ordinateur pour résoudre des calculs. Cela est plus rapide et plus efficace. Alors que pour un calcul très long l'homme risque de se tromper à multiples reprises et de prendre beaucoup de temps, l'ordinateur va donner la réponse rapidement.

1.9.2. Travail

Un autre avantage est celui des machines à la chaîne, par exemple, pour remplacer l'homme dans des taches pénibles ou dangereuses, Ces machines permettent tout de même une fois de plus d'optimiser le travail qui aurait pu être réaliser par les hommes. De plus, elles possèdent un autre avantage non négligeable, elles n'ont pas de contraintes physiques. En effet, elles n'ont ni besoin de manger ni besoin de repos et elles ne peuvent pas être malade.

1.9.3. Au quotidien

Au quotidien, des robots sous forme d'humanoïde se développent, ils vont permettre de faire le ménage, faire les courses, la cuisine etc. C'est un avantage surtout pour les personnages âgées qui pourront avoir une aide à leur coté pour leur facilité le quotidien devenu difficile.

1.9.4. Déplacement

Aujourd'hui il existe déjà des véhicules pouvant se déplacer seul à l'aide de caméra et de capteurs répartit sur celui-ci. On peut considérer ces véhicules comme intelligent car il s'adapte à leur environnement , peuvent réagir seul et dissocient la route des obstacles.

1.9.5. Explorer

Dans le domaine spatiale, les robots et l'IA sont très utilisés car ils permettent d'envoyer, à la place des hommes, des robots en partie autonome là où les hommes ne pourraient pas aller en raison des pressions ou des températures trop forte ou trop faible.

1.9.6. Etudier

Dans le domaine de la médecine, l'IA a permis l'élaboration de robots patients qui présentent un avantage pour les étudiants en médecine. Grâce à ces robots l'étudiant va pouvoir s'entrainer sans risque. Il sera sequ'il a bien ou mal fait et pourra ainsi s'améliorer. C'est un réel avantage car avec

le robot on peut faire des erreurs, c'est beaucoup moins grave que si elles avaient lieux sur un humain.

1.9.7. Médecine

Le développement des prothèses va permettre aux gens ayant perdu l'usage d'un membre de retrouver une vie normale comme ils avaient avant. En effet,ces membres artificiels possèdent une intelligence car ils vont devoir s'adapter à la personne sur laquelle ils ont été greffé. Par exemple, lorsque la personne voudra saisir un verre, il faudra que la force du bras soit adapté pour ne pas casser le verre ni le faire tomber.

1.9.8. Sauver

Cette partie a été vu dans le domaine militaire qui présente comme avantage de pouvoir sauver des personnes en danger sans se mettre en danger car c'est le robot qui va le faire.

1.9.9. Nouvelle forme humanoïde

Les robots sont maintenant représentés sous forme d'humain appelé humanoïde. L'avantage face à cette nouvelle forme est le fait que la personne se trouvant en face n'a pas l'impression d'avoir à faire à une machine et cela donne un côté plus agréable surtout quand cette machine est en permanence dans votre quotidien

1.10. Conclusion :

Dans ce chapitre nous avons présenté et définis les différents types de robots mobiles ainsi que les avantages et les inconvénients de chaque structure et leurs principales caractéristiques.

2.1. INTRODUCTION

L'espace qui nous entoure a une structure tri dimensionnelle (3D).Lorsque l'on demande à une personne de décrire ce qu'elle voit, elle n'éprouve aucune difficulté a nommer les objets qui l'entourent téléphone, table, livre Et pourtant l'information qui est réellement disponible sur la rétine des ses yeux n'est, ni plus ni moins, une collection de points(environ un million!) .En chaque point ou pixel (Picture élément) il y a tout simplement une information qui donne une indication quant à la quantité de lumière et la couleur qui proviennent de l'espace environnant et qui ont été projetées à cet endroit de la rétine, Le téléphone, la table ou le livre n'existent pas sur la rétine.

Guidé à la fois par l'information codée dans l'image ,ou la rétine, et par ses propres connaissances, le processus visuel construit des percepts, Le téléphone ou le livre sont le réponses finales, résultant d'un processus d'interprétation qui fait partie intégrante du système de vision, De plus, il n'y a pas de correspondance terme à terme entre l'information sensorielle, la lumière et la couleur, et la réponse finale (des objets 3D), Le système de vision doit fournir les connaissances nécessaires afin de permettre une interprétation non ambigüe.

2.2. Une théorie de la vision

L'élaboration d'une théorie scientifique demande trois étapes :

a. énoncer la théorie, spécifier et élaborer les concepts de base ces concepts doivent exprimer le cadre formel qui est à la base de la théorie.
b. exprimer ces concepts sous forme mathématique.
c. réaliser un ensemble expérimental qui permette de verrier la théorie. Voici comment la vision par ordinateur peut s'énoncer brièvement dans les termes de ce paradigme. La vision est un processus de traitement de l'information, Elle utilise des stratégies bien définies afin d'atteindre ses buts. L'entrée d'un système de vision est constituée par une séquence d'images, Le système lui même apporte un certain nombre de connaissances qui interviennent à tous les niveaux. La sortie est une description de l'entrée en termes d'objets et de relations entre ces objets.

Deux types de stratégies sont mis en jeu ascendantes et descendantes, Les stratégies ascendantes tentent de construire à partir de l'information sensorielle une représentation la plus abstraite possible, par exemple, un ensemble de primitives géométriques (3D). Les stratégies descendantes déduisent à partir de l'ensemble d'objets connus par le système une description compatible avec les primitives extraites de l'image. Il est alors possible de mettre en correspondance la représentation extraite

de l'image avec les descriptions des objets afin de décrire les données sensorielles en termes de ces objets.

2.3. Capteurs d'image

Le but des capteurs d'images est de retranscrire, le plus fidèlement possible, l'image d'un objet éclairé, ou d'une source lumineuse, formée à leur surface par un système optique adéquat.

Nous nous attacherons principalement à décrire les deux grandes familles de capteurs optico-électoniques utilisés de nos jours : les capteurs **CCD** et les capteurs **CMOS**.

2.3.1. CCD

Les capteurs **CCD** (Charge Coupled Device : systèmes à transfert de charges) font référence à une architecture semiconductrice dans laquelle la charge est transférée via une aire de stockage. La plupart des détecteurs opérant dans la région du visible utilise une architecture **CCD** pour déplacer le paquet de charge, ils sont communément appelés matrices **CCD**. Leur architecture a trois fonctions de base, outre la création de la charge:

a. collection de la charge,

b. transfert de la charge,

c. conversion de la charge en une tension mesurable.

Comme les matrices opérant dans le visible sont des systèmes monolithiques, la génération de la charge est souvent considérée comme la fonction initiale du **CCD**. La charge est créée en un pixel (pixel est la contraction de « picture element » : le plus petit morceau d'image) proportionnellement au niveau de lumière incidente en ce site.

2.3.2. CMOS

Les capteurs **CMOS** sont apparus au début des années 1990.

Ils permettent notamment une conversion de la charge directement sur la photosite de génération par la présence d'un amplificateur sur le pixel. Cette particularité permet également de supprimer de nombreux transferts et d'accroitre la vitesse de lecture. Leurs principaux avantages sont issus de leur fabrication.

a. Fabrication identique (90%) aux chips informatiques (et plus particulièrement la **DRAM** Dynamical Random Access Memory)

b. Production de masse à bas coût

c. Conversion directe de la charge sans transfert : pas de blooming ni de smearing

d. Chaque pixel a son propre ampli, pas de shift register : Active Pixel Sensor

e. Chaque pixel est adressable individuellement

f. Pas d'horloges compliquées

i. Faible consommation électrique (100 x moins que **CCD**)

j. Cadence de lecture élevée

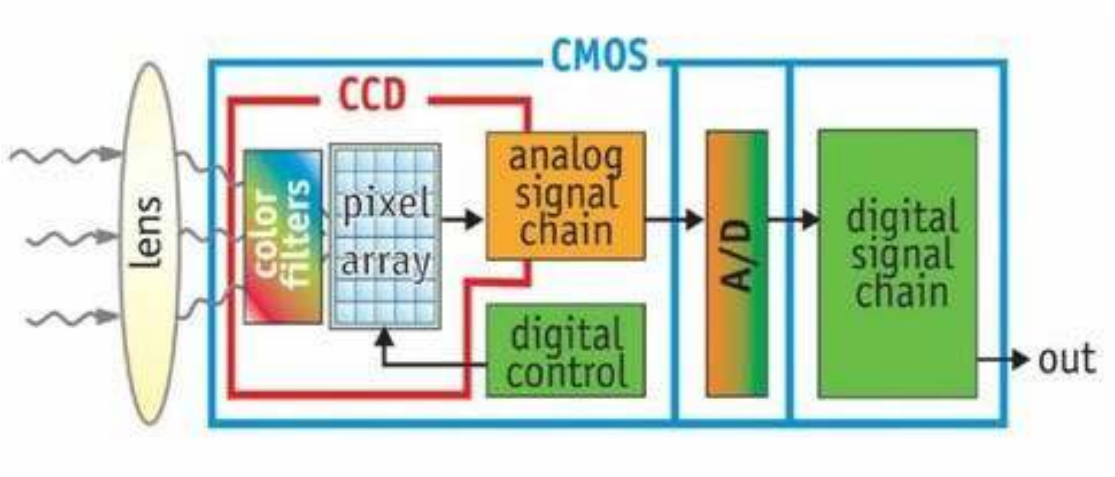


Figure 2.1 fonctionnement des capteur ccd et cmos

2.3.3. Comparatif des CCD et CMOS

Les principaux critères de comparaison sont repris ci-dessous, et sont en forte évolution...

a. Facteur de remplissage

CMOS : 30% à 50%

CCD:50% à80%

b. Exposition de saturation

CMOS : 0,2 µJ/cm²

CCD:0,05 µJ/cm²

c. Courant d'obscurité:

CMOS > **CCD**

CMOS ne permet pas un temps d'exposition long

d. Dispersion de la sensibilité des pixels (Pattern Noise) :

CMOS :5% à20%

CCD:1 à 2%

Pour le **CMOS**, une correction possible après étalonnage de la sensibilité des pixels (shading)Défaut linéaire **CMOS** non linéaire avec flux reçu

Figure 2.2 comparaison entre un capteur CCD et un capteur CMOS

2.4. image numérique

2.4.1. Définition

Une image numérique est composée d'unités élémentaires (appelé pixel) qui représentent chacun une portion de l'image.

Une image est définie par :

a. Le nombre de pixels qui la compose en largeur et en hauteur (qui peut varier presque à l'infini),
b. l'étendu des teintes de gris ou des couleurs que peut prendre chaque pixel (on parle de dynamique de l'image).

Toutes les données correspondant aux informations contenues dans l'image sont structurées d'une certaine façon afin de permettre leur stockage. Il existe un grand nombre de formats d'images, tous ces formats ne correspondent ni plus ni moins qu'à une structuration particulière des données concernant l'image.

Une image numérique en elle même est en fait un concept tout à fait abstrait (des données numériques) qui ne trouve une signification à nos yeux qu'à la visualisation lorsque l'on utilise un logiciel adéquat.

2.4.2. La qualité de L'image

La qualité d'une image numérique s'exprime en termes de rendu chromatique de résolution et de netteté, bref de sa fidélité par rapport au sujet. Elle dépend de plusieurs facteurs:

a. La résolution optique du système photographique. Elle varie en fonction de la résolution du capteur photosensible, de la qualité de l'optique, en particulier des aberrations résiduelles, du bruit électronique et de la capacité de Shannon, elle-même dépendant du bruit et du niveau de contraste
b. La fidélité chromatique qui est influencée par l'espace de couleur de travail, la dynamique de l'appareil, le bruit des différents canaux L,R,G,B, et les performances des conversions A/D et RAW.
c. Le vignetage qui dépend des caractéristiques de l'optique, de la taille du capteur photosensible et éventuellement des accessoires (pare-soleil, etc.). Son intensité varie en fonction de l'ouverture du diaphragme.

2.5. LA VISION ROBOTIQUE

Un Système de Vision par ordinateur doit posséder deux parties, une correspondant à la phase de perception, il s'agit de remplacer l'œil par un système similaire qui permette de photographier» des images avec le maximum de renseignements subtiles, l'autre, l'interprétation et la décision

2.5.1. Traitement d'image

En vision les traitements se composent de quatre phases:

a. la numérisation,
b. les prétraitements,
c. la segmentation,
d. l'interprétation.

2.5.2. La Numérisation

La numérisation consiste à réaliser la transformation du signal analogique issu du capteur en un signal numérique exploitable par l'ordinateur. Cette transformation est électronique, elle s'effectue en temps réel. Dans le cas d'un système de vision matriciel. la numérisation forme une matrice de pixels. A chaque pixel est associée une (ou plusieurs pour la couleur RVB) grandeur caractérisque.

2.6. ARDUINO

2.6.1. Introduction

Le concept de matériel libre vient d'abord du monde de l'informatique, mais il s'étend à tout autre domaine touchant la vie humaine.

Si le matériel que nous utilisons dans notre quotidien est inaccessible, non documenté, si l'on ne peut le modifier en changeant une puce, en soudant un élément supplémentaire relativement facilement, si toute modification est interdite par l'entreprise qui le vends ou que cela casse la garantie de la totalité des pièces du produit, alors c'est le matériel (notamment par le biais des firmwares en matière informatique, mais pas seulement) qui "piégera" et contournera le logiciel libre et le monde du libre.

Appréhender le matériel libre, c'est appréhender la possibilité de devenir sa propre usine, son propre fabricant d'objets, en se libérant des contraintes de la consommation obligatoire, de la publicité, de l'avant et après-vente.

2.6.2. Qu'est-ce qu'Arduino?

Le système Arduino est un outil pour fabriquer de petits ordinateurs qui peuvent capter et contrôler davantage de choses du monde matériel que votre ordinateur de bureau. C'est une plateforme open-source d'électronique programmée qui est basée sur une simple carte à microcontrôleur (de la famille AVR), et un logiciel, véritable environnement de développement intégré, pour écrire, compiler et transférer le programme vers la carte à microcontrôleur.

2.6.3. L'application d'Arduino

Arduino peut être utilisé pour développer des objets interactifs, pouvant recevoir des entrées d'une grande variété d'interrupteurs ou de capteurs, et pouvant contrôler une grande variété de lumières, moteurs ou toutes autres sorties matérielles. Les projets Arduino peuvent être autonomes, ou bien ils peuvent communiquer avec des logiciels tournant sur notre ordinateur. Les cartes électroniques peuvent être fabriquées manuellement ou bien être achetées pré- assemblées; le logiciel de développement open-source peut être téléchargé gratuitement.

2.6.4. Les différentes cartes Arduino:

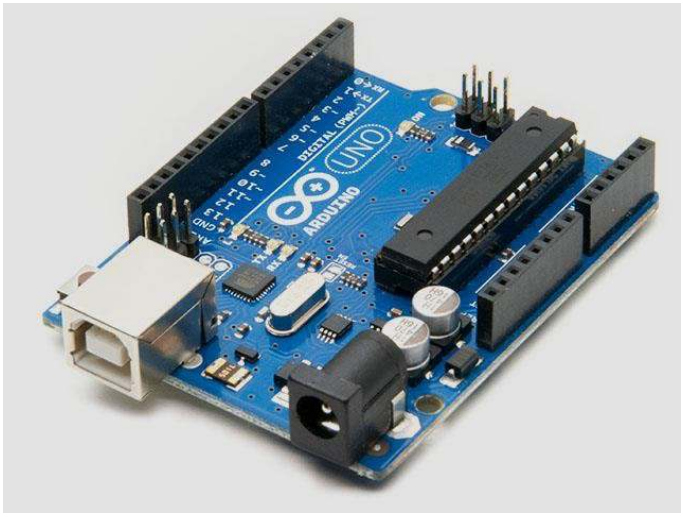


Figure 2.3 La carte Arduino UNO

Figure 2.4 La carte Arduino LEONARDO

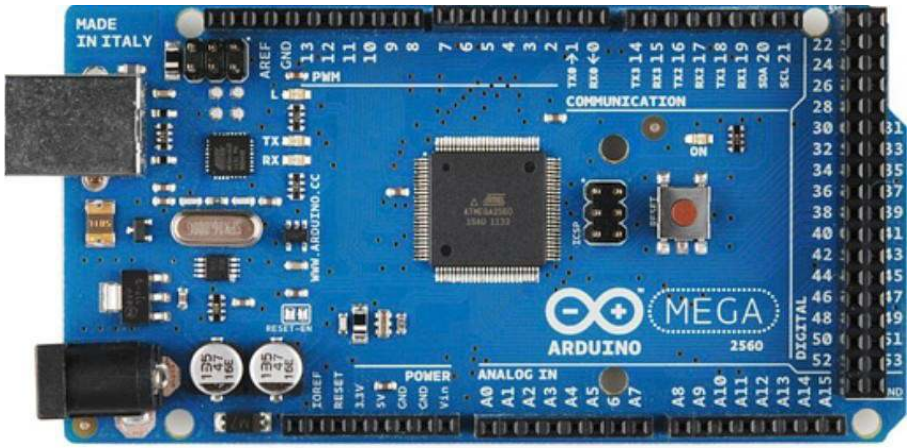


Figure 2.5 La carte ArduinoMEGA

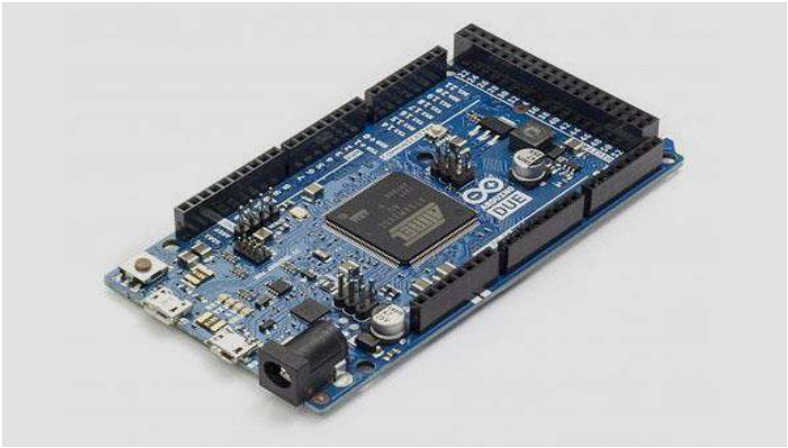


Figure 2.6 La carte Arduino DUE

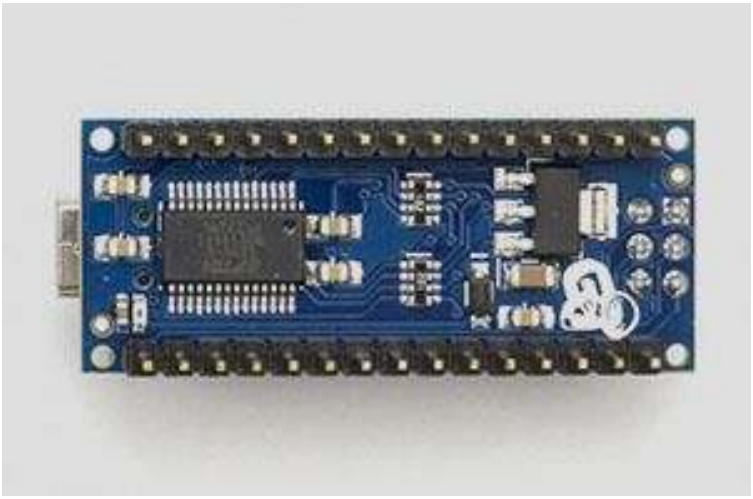


Figure 2.7 La carte Arduino NANO

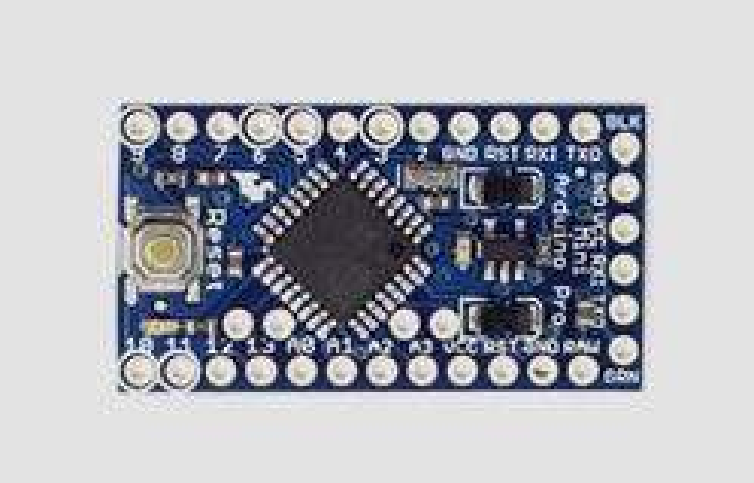


Figure 2.8 La carte Arduino MINI PRO

Figure 2.9 La carte Arduino YUN

2.6.5. Tableau comparatif des différentes cartes d'Arduino

Name	Processor	Operating/Input Voltage	CPU Speed	Analog In/Out	Digital IO/PWM	EEPROM [kB]	SRAM [kB]	Flash [kB]	USB	UART
Due	ATSAM3X8 E	3.3 V / 7-12 V	84 MHz	12/2	54/12	-	96	512	2 Micro	4
UNO	ATmega328P	5 V / 7-12 V	16 MHz	6/0	14/6	1				
Leonardo	ATmega32U4	5 V / 7-12 V	16 MHz	12/0	20/7	1	2.5	32	Micro	1
Mega ADK	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
Mini PRO	ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8MHZ 16MHz	6/0	14/6	14/6	1	32	-	-
Nano	ATmega168 ATmega328P	5 V / 7-9 V	16 MHz	8/0	14/6	0.512 1	1 2	16 32	Mini	1
Yùn	ATmega32U4 AR9331 Linux	5 V	16 MHz 400MHz	12/0	20/7	1	2.5 16MB	32 64MB	Micro	1

Tableau 2.1 comparaison des différentes cartes d'Arduino

2.7. Arduino MEGA2560

La carte Arduino Mega 2560 est une carte à microcontrôleur basée sur un ATmega2560

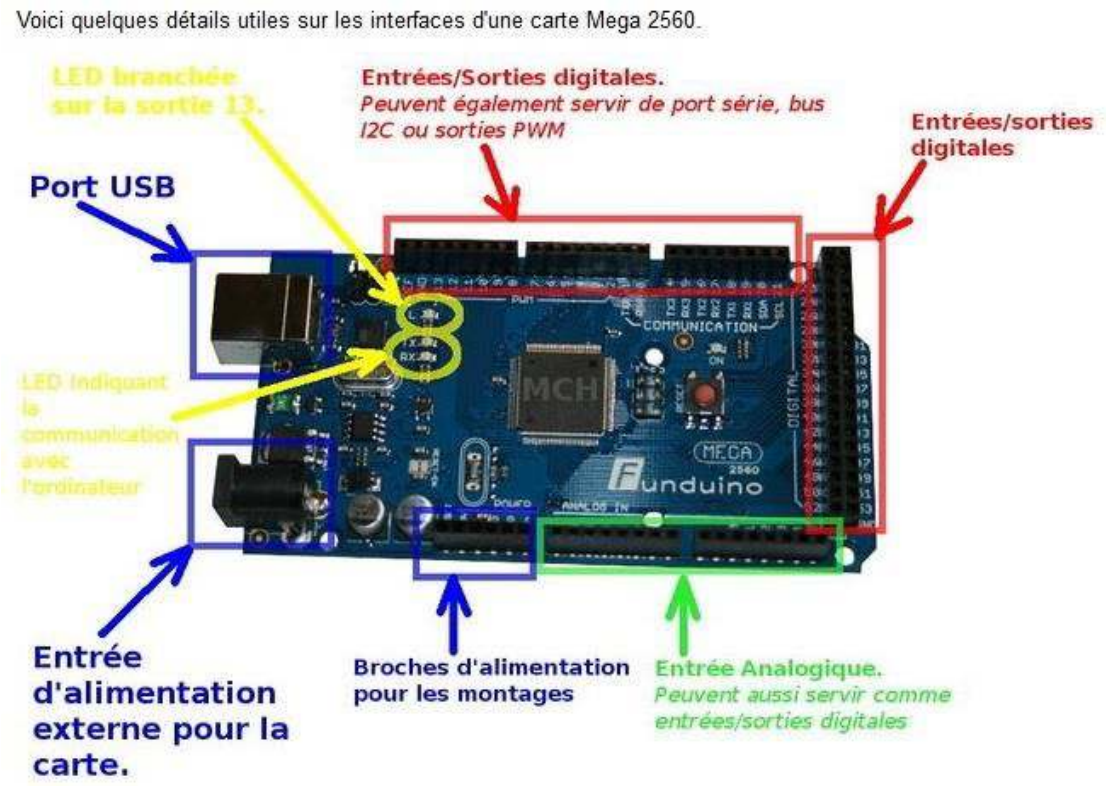


Figure 2.10 Arduino Mega 2560

2.7.1. Synthèse des caractéristiques :

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 bootloader
Mémoire SRAM (mémoire volatile)	8 KB
Mémoire EEPROM (mémoire non volatile)	4 KB
Vitesse d'horloge	16 MHz

Tableau 2.2 Synthèse des caractéristiques

2.7.2. Entrées et sorties numerique

Certaines broches ont des fonctions spécialisées :

a. Communication Serie

PortSerieSerial:0(RX)and1(TX); Port

SerieSerial1:19(RX)and18(TX);

PortSerieSerial2:17(RX)and16(TX);

Port Serie Serial 3: 15 (RX) and 14 (TX). Utilisées pour recevoir (RX) et transmettre (TX) les données séries de niveau TTL. Les broches 0 (RX) and 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.

b. 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. Voir l'instruction attachInterrupt() pour plus de détails.

c. Impulsion PWM (largeur d'impulsion modulée)

Broches 0 à 13.

Fournissent une impulsion PWM 8-bits à l'aide de l'instruction analogWrite().

d. 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 etDiecimila.

e.I2C

Broches 20 (SDA) et 21 (SCL).

Supportent les communications de protocole I2C (ou interface TWI (Two Wire Interface - Interface "2 fils"), disponible en utilisant la librairie Wire/I2C (ou TWI - Two- Wire interface - interface "2 fils") . Noter que ces broches n'ont pas le même emplacement que sur les cartes Uno, Duemilanove ou Diecimila.

f. 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.

2.7.3.Broches analogiques

La carte Mega2560 dispose de 16 entrées analogiques, chacune pouvant fournir une mesure d'une résolution de 10 bits (càd 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.

2.7.4.Autres broches

Il y a deux autres broches disponibles sur la carte :

a.AREF :

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

b.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.

2.7.5.Arduino ide

L'Arduino IDE est un Software open-source qui permet d'écrire et de compiler des programmes pour la carte Arduino. C'est une inteface assez simple qui permet aux connaisseurs du langage C++ d'effectuer des scripts et de les transférer par la suite sur la carte via un port USB. Ce logiciel permet aussi de communiquer en temps réel avec la carte Arduino.

2.7.6.Fenetre de l'application Arduino IDE:

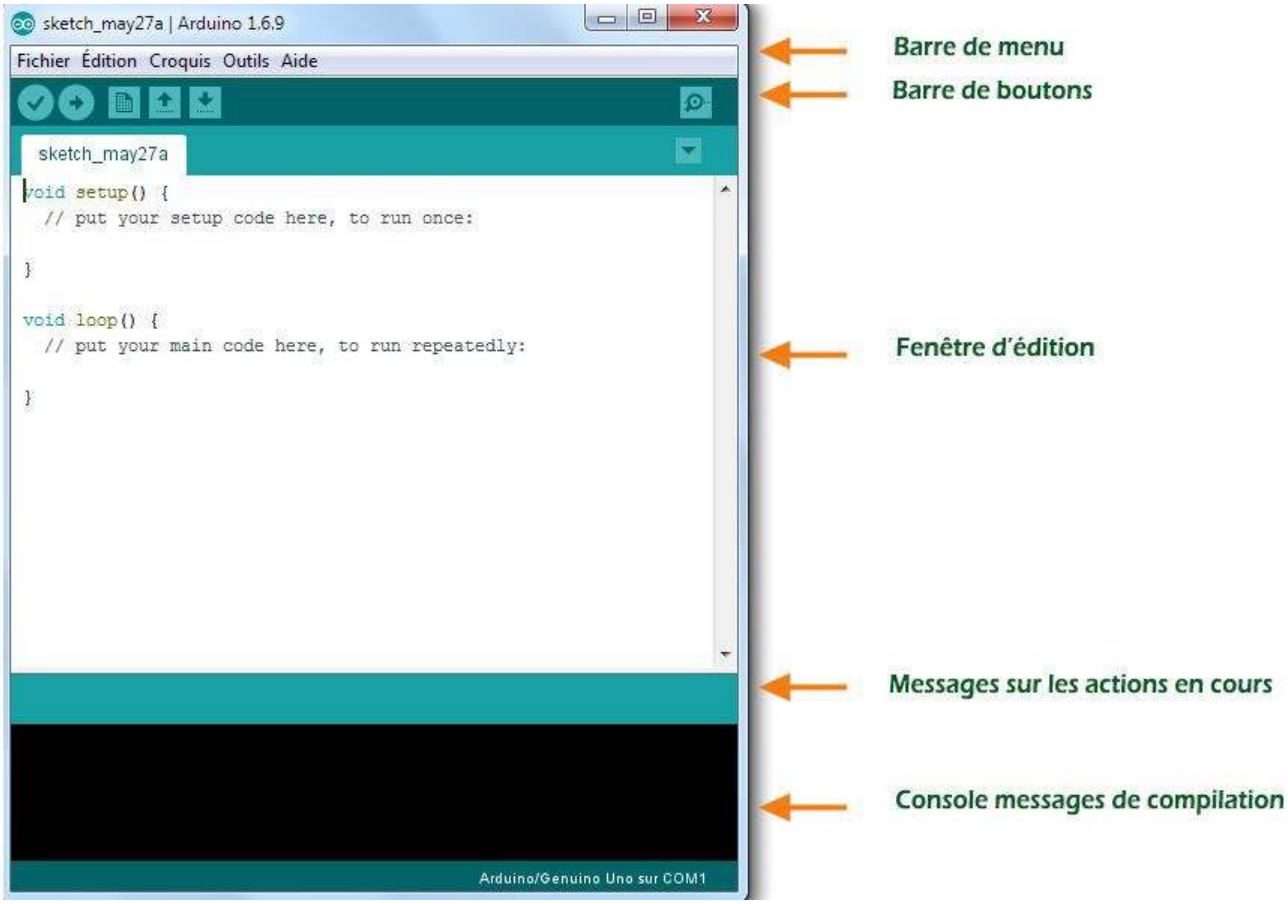


Figure 2.11 Menu Arduino

2.7.7.Fenetre duprogramme

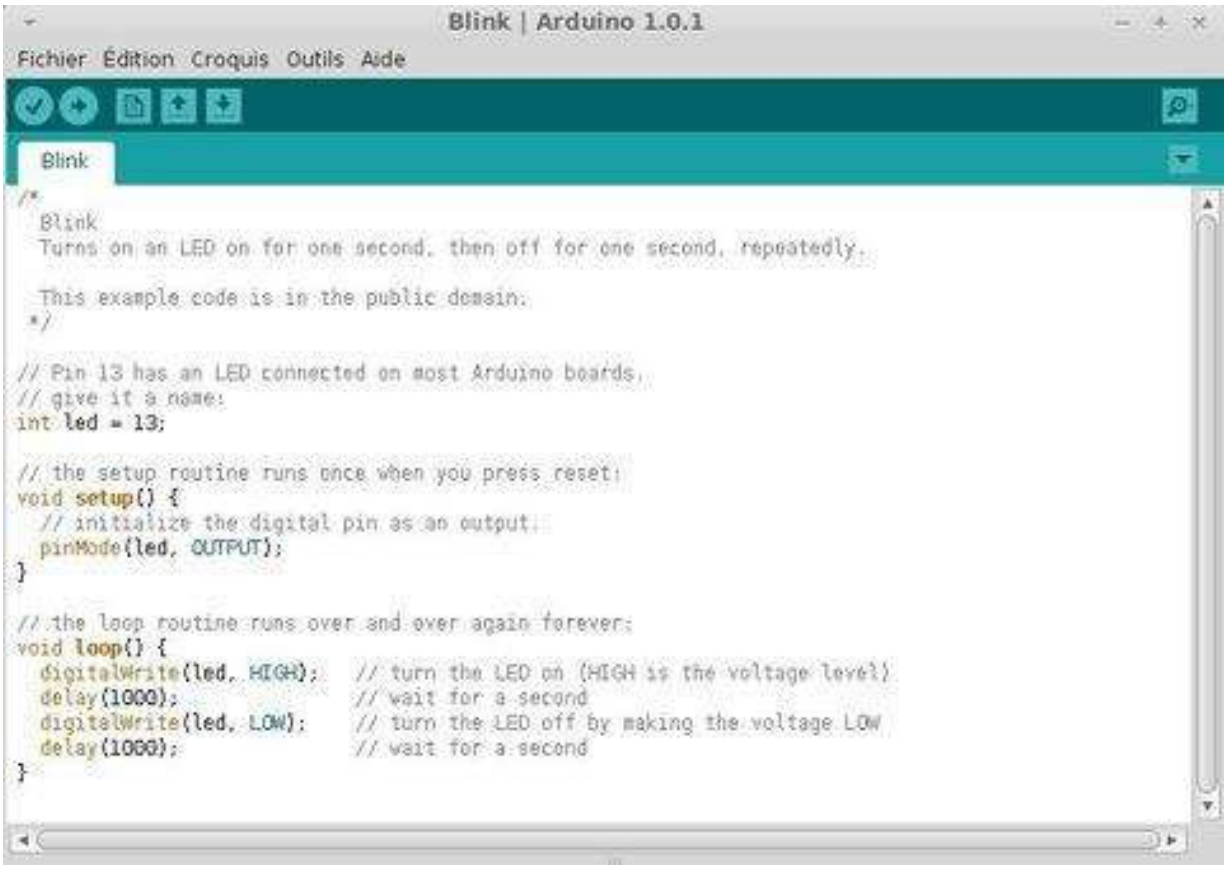


Figure 2.12 Fenêtre du programme

a.Moniteur série

Le moniteur sériel est utilisé pour afficher l'information qui est envoyée par la carte Arduino vers l'application (habituellement par le câble USB).

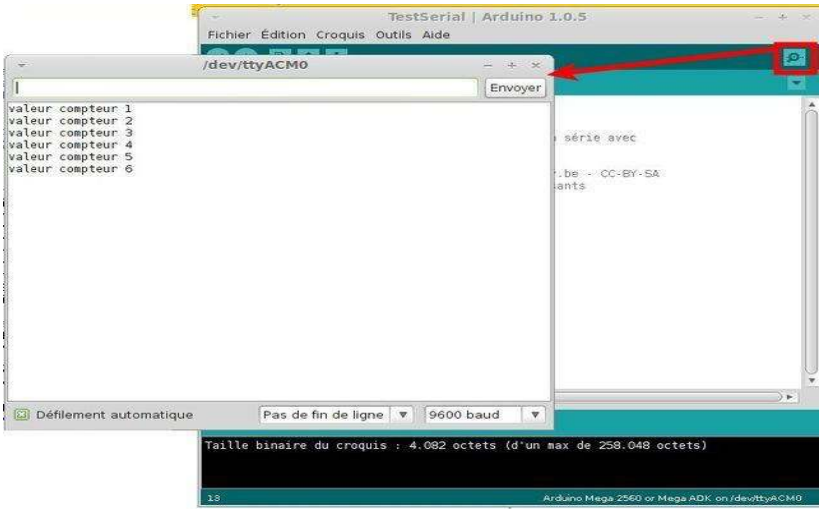


Figure 2.13 Moniteur Serie

2.8.TRANSMISSION SÉRIE : LE BUS I2C

2.8.1.INTRODUCTION

Les normes I2C (Inter-Integrated Circuit) et SPI (Serial Peripheral Interface) ont été créées pour fournir un moyen simple de transférer des informations numériques entre des capteurs et des microcontrôleurs. Les bibliothèques Arduino pour I2C et SPI facilitent l'utilisation de ces deux protocoles. Le choix entre I2C et SPI est en général déterminé par les périphériques que l'on souhaite connecter. Certains appareils offrent les deux standards, mais habituellement un périphérique ou une puce ne supporte qu'une seule des deux normes. I2C à l'avantage de n'avoir besoin que de deux connexions de signalisation (l'emploi de plusieurs périphériques sur les deux connexions est assez simple et on reçoit une confirmation que les signaux ont été correctement reçus). L'inconvénient est que la vitesse des données est inférieure à celle de SPI et qu'elles ne peuvent voyager que dans un seul sens à la fois (Simplex), ce qui abaisse encore plus le débit si des communications bidirectionnelles sont nécessaires (Half Duplex). Il faut aussi connecter des résistances « pull-up » aux connexions pour s'assurer de la fiabilité de la transmission des signaux. L'avantage de SPI est qu'il a un meilleur débit et qu'il a des connexions d'entrée et de sorties séparées, si bien qu'il peut envoyer et recevoir en même temps (Full Duplex). Il utilise une ligne supplémentaire par appareil pour sélectionner le périphérique actif et il faut donc plus de connexions si on a de nombreux appareils à connecter.

2.8.2.LE BUSI2C

Les deux connexions du bus I2C se nomment SCL (Serial Clock Line) et SDA (Serial Data line). Elles sont disponibles sur une carte standard Arduino en employant la broche analogique 5 pour SCL qui fournit un signal d'horloge, et la broche analogique 4 pour SDA, qui s'occupe du transfert des données (sur la Mega, il faut utiliser la broche 20 pour SDA et la broche 21 pour SCL). La carte Uno rev 3 a des broches supplémentaires qui dupliquent les broches 4 et 5. Pour le Raspberry Pi type B, il s'agit respectivement des broches GPIO2 et GPIO3 pour SDA et SCL repérées sur le « Pi Cobbler ». Un périphérique sur le bus I2C est considéré comme le périphérique maître. Son travail consiste à coordonner le transfert des informations entre les autres périphériques (esclaves) qui sont connectés. Il ne doit y avoir qu'un seul maître qui contrôle les autres composants auxquels il est connecté. La figure ci-dessous montre un maître I2C avec plusieurs esclaves I2C.

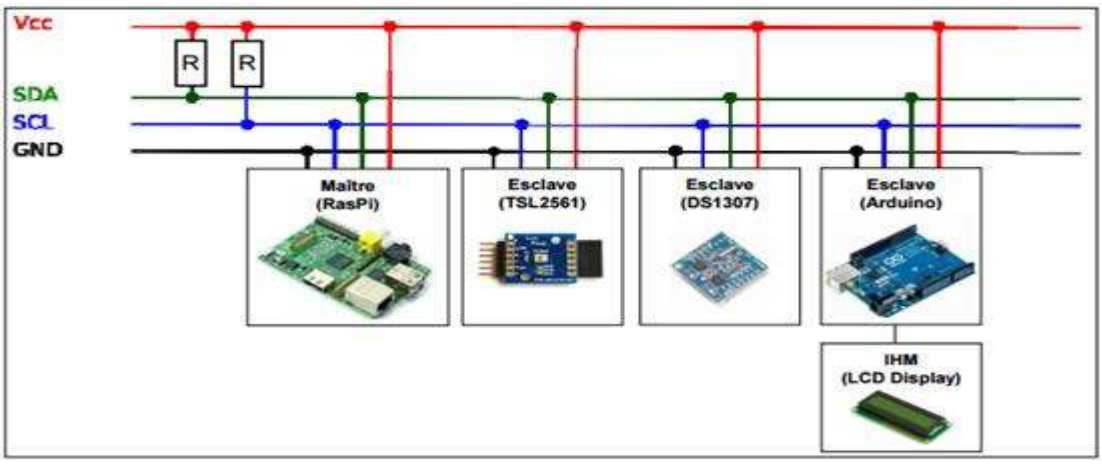


Figure 2.14: maitre i2c avec plusieurs esclaves i2c

Les périphériques I2C ont besoin d'une masse commune pour communiquer. Les périphériques esclaves sont identifiés par leur numéro d'adresse. Chaque esclave doit avoir une adresse unique. Certains appareils I2C ont une adresse fixe alors que d'autres permettent que l'on configure leur adresse en définissant les broches à HIGH ou LOW ou en envoyant des commandes d'initialisation.

2.8.3.Remarques :

Le bus I2C permet d'échanger des informations sous forme série avec un débit pouvant atteindre 100 kilobits/s ou 400 kilobits/s pour les versions les plus récentes. Ses points forts sont les suivants :

a. c'est un bus série bifilaire utilisant une ligne de donnée SDA et une ligne d'horloge SCL ;
b. le bus est multi maîtres ;
c. chaque abonné dispose d'une adresse codée sur 7 bits, on peut donc connecter simultanément 128 abonnés d'adresses différentes sur le même bus (sous réserve de ne pas le surcharger électriquement = la charge);
d. un acquittement est généré pour chaque octet de donnée transféré ;
e. un procédé permet de ralentir l'équipement le plus rapide pour s'adapter à la vitesse de l'équipement le plus lent lors d'un transfert ;
f. le nombre maximal d'abonné n'est limité que par la charge capacitive maximale du bus qui peut être de 400 pF ;
g. les niveaux électriques permettent l'utilisation des circuits en technologies CMOS, NMOS ou TTL.

2.8.4.PROTOCOLE

La figure ci-dessous montre le principe adopté au niveau des étages d'entrées/sorties des circuits d'interfaces du bus I2C.

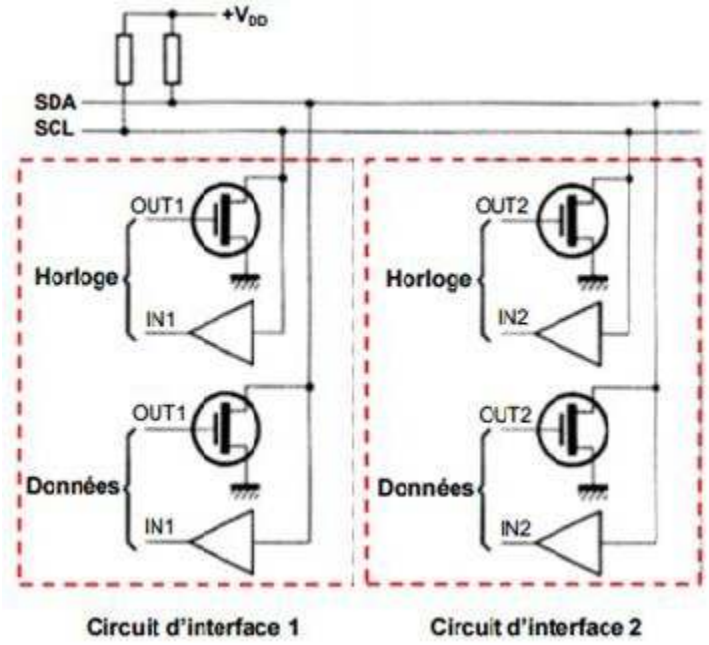


Figure 2.15 Schema de l'interfece matérielle des cicuits compatibles I2C

La partie entrée n'appelle aucune remarque particulière, en revanche la partie sortie fait appel à une configuration à drain ouvert (l'équivalent en MOS du classique collecteur ouvert) et qui permet de réaliser des ET câblés par simple connexion sur la ligne SDA ou SCL, des sorties de tous les circuits. Aucune charge n'étant prévue dans ces derniers, une résistance de rappel (pull-up) à une tension positive doit être mise en place. Étant entendu que nous travaillons en logique positive, c'est-à-dire qu'un niveau haut correspond à une tension plus élevée qu'un niveau bas.

Lorsque aucun abonné n'émet sur le bus, les lignes SDA et SCL sont au niveau haut qui est leur état de repos. Quand une ligne (SDA ou SCL) est au repos (niveau 1), on peut la forcer à 0. Quand une ligne (SDA ou SCL) est au niveau 0, on ne peut pas la forcer à 1.

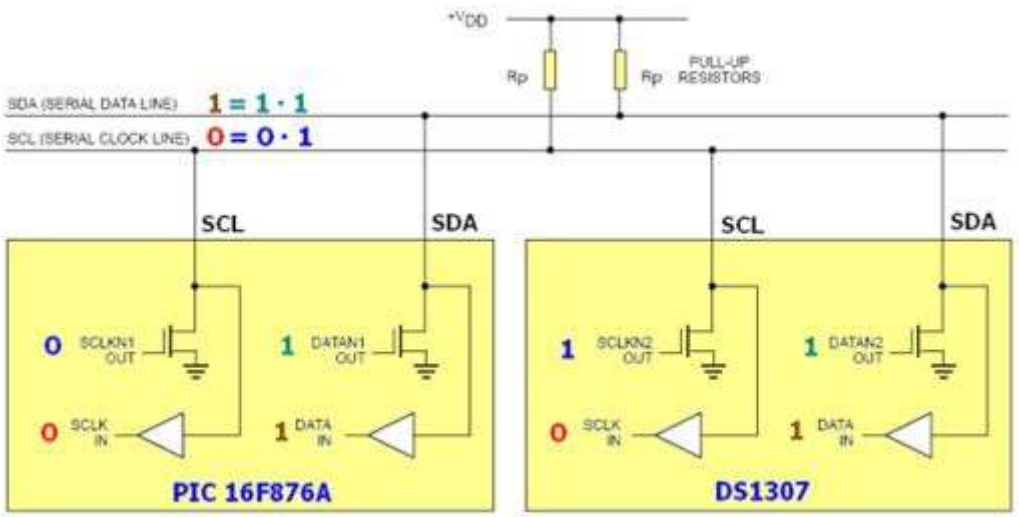


Figure 2.16 Exemple avec PIC 16F876A (master) et DS7307 (slave)

Dans l'exemple ci-dessus, l'horloge SCL est au niveau 0 car c'est le PIC 16F876A (maître) qui force cette ligne à 0.

2.8.5.Remarque :

Le DS1307 (module RTC : Real Time Clock) fonctionne toujours en esclave : ce n'est donc jamais lui qui agit sur l'état de l'horloge (SCL).

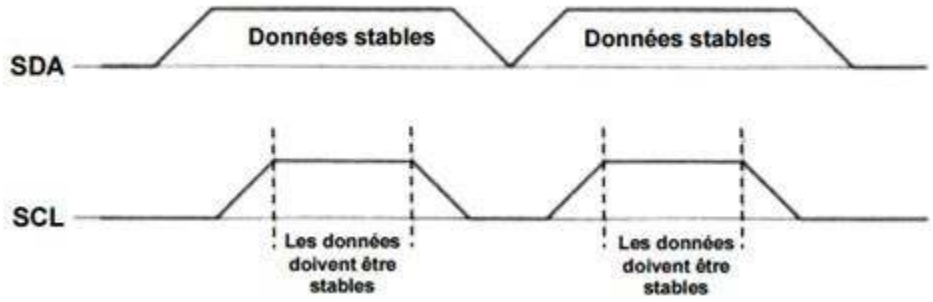


Figure 2.17 Chronogramme fondamental de bus I2C

Le chronogramme ci-dessus résume le principe fondamental d'un transfert de données : Une donnée n'est considérée comme valide sur le bus que lorsque le signal SCL est à l'état haut. L'émetteur doit donc positionner la donnée à émettre lorsque SCL est à l'état bas et la maintenir tant que SCL est à l'état haut. Comme la transmission s'effectue sous forme série, une information de début et de fin doit être prévue. L'information de début se nomme START et l'information de fin STOP.

Une condition de départ est réalisée lorsque la ligne SDA passe du niveau haut au niveau bas alors que SCL est au niveau haut. Réciproquement, une condition d'arrêt

est réalisée lorsque SDA passe du niveau bas au niveau haut alors que SCL est au niveau haut.

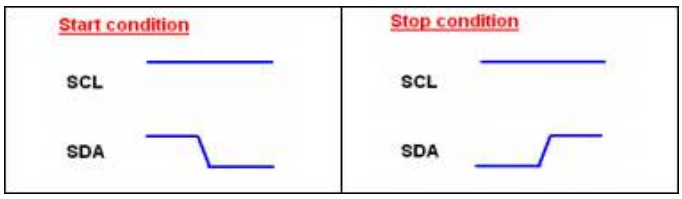


Figure 2.18 Condition de départ

Les données sont envoyées par paquets de huit bits (ou octets). Le bit de poids fort est envoyé le premier, chaque octet est suivi par un bit d'acquittement (ACK) de la part du destinataire.

Le processus fonctionne de la façon ci-dessous.

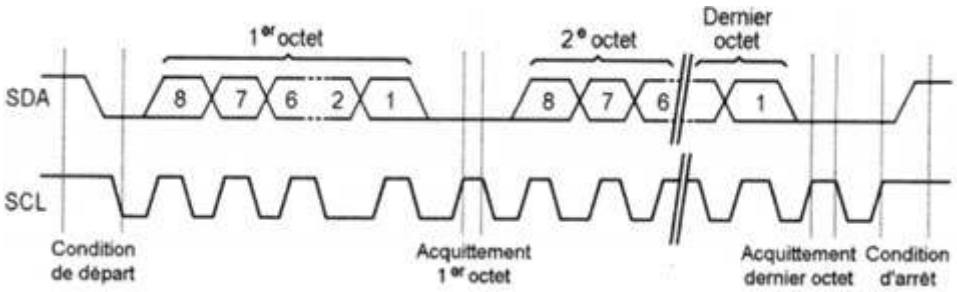


Figure 2.19 Chronogramme d'un échange sur un bus I2C

La ligne SCL est pilotée par l'initiateur de l'échange ou maître. Le chronogramme ci-dessus montre d'abord une condition de départ, généré par le maître du bus à cet instant. Elle est suivie par le premier octet de données, poids fort en tête. Après le huitième bit, l'émetteur qui est aussi le maître dans ce cas met sa ligne SDA au niveau haut, c'est-à-dire au repos mais continue à générer l'horloge sur SCL. Pour acquérir l'octet, le récepteur doit alors forcer la ligne SDA au niveau bas pendant l'état haut de SCL qui correspond à cet acquittement, prenant en quelque sorte la place d'un neuvième bit. Le processus peut alors continuer avec l'octet suivant et se répéter autant de fois que nécessaire pour réaliser un échange d'informations complexes. Lorsque cet échange est terminé, le maître génére une condition d'arrêt.

.

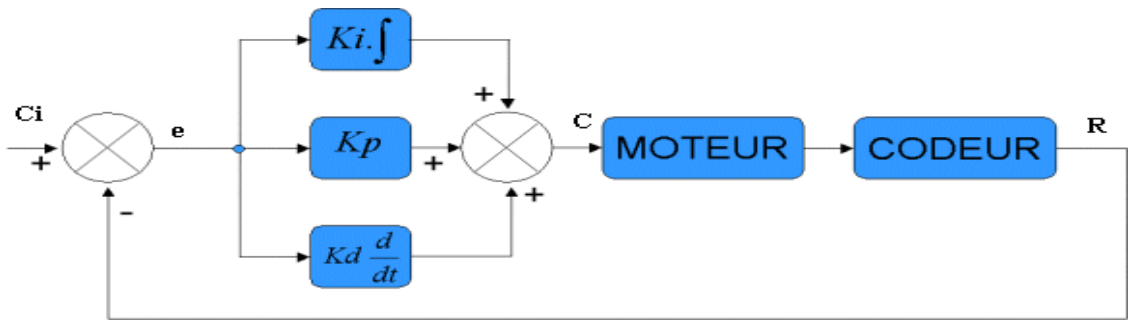
2.9. Introduction

L'histoire commence avec une action à réaliser, par exemple déplacer un robot de 0.5 m en avant. L'asservissement par PID s'applique à de nombreux domaines, mais je préfère vous présenter directement l'utilisation concrète que nous en avons dans le robot. Pour avancer droit, on peut intuitivement penser qu'il faut appliquer une tension constante sur les deux moteurs pendant le délai nécessaire. On obtient ce délai par un petit calcul prenant en compte le diamètre et la vitesse de rotation des roues, sans oublier le temps de montée (accélération) et le temps de descente (décélération)... le tour est joué... ou pas! Malheureusement, cela ne fonctionne pas. Même avec des moteurs de bonne qualité, la vitesse dépend souvent de la charge des accumulateurs, du poids du robot et d'un tas d'autres paramètres qui varient selon les circonstances. Du coup les résultats obtenus sont vite erronés. Dans notre cas, on souhaite que le robot avance, mais les deux moteurs ne répondent pas de la même façon, le robot dévie immédiatement de sa trajectoire. Donc, on ordonne au robot de se déplacer de façon précise (avancer de 0.5m) et il suit une trajectoire différente, et donc atteint une destination différente!! Ce qu'il fait n'est pas trop loin de la consigne, mais des imperfections apparaissent. Du coup, nous allons simplement dire qu'il a commis une **erreur**. Et l'idée ici c'est de rendre notre robot conscient de cette erreur qu'il a commis ou mieux est en train de commettre pour qu'il corrige automatiquement sa trajectoire et sa vitesse. On dit alors qu'on effectue un asservissement.

2.10. Principe d'asservissement

Après cette introduction, il est temps de passer aux choses sérieuses.
L'asservissement consiste tout simplement en la récupération d'une information sur la vitesse du 0moteur (ici grâce aux roues codeuses) puis en son utilisation pour ajuster la tension de commande. Il existe de nombreuses méthodes d'asservissement, nous présentons ici celle que nous utilisons dans notre robot (et la plus connue de toutes) : le PID (Proportionnel Intégrale Dérivée).

2.11. Principe d'asservissement PID



a. Ci : Consigne initiale (ce qu'on veut qu'il fasse)
b. e : erreur entre la consigne initiale et la réalité
c. C : Consigne appliquée au moteur
d. R : Grandeur réelle mesurée (réalité)

Le principe de base de tout asservissement est de prendre des mesures à la sortie, de les réinjecter à l'entrée pour comparer à la consigne afin d'obtenir l'erreur (ie la différence entre la mesure réelle et la consigne demandée). Quand elle est nulle, le robot a atteint sa destination et donc il n'y a plus rien à actionner. Ce système est dit en boucle fermée, puisque la sortie du système est réinjectée dans l'entrée.

Maintenant que vous connaissez le principe général, caractérisons le PID, c'est à dire le Proportionnel, Intégral, Dérivé. Ces trois blocs bleus correspondent chacun à une fonctionnalité : Ki est le coefficient intégral, Kp le coefficient proportionnel et Kd le coefficient dérivé. Commençons par le plus simple : l'asservissement Proportionnel.

2.11.1.Asservissement PID

L'idée est à présent de dériver l'erreur et d'ajouter cette valeur à celle obtenue précédemment afin de limiter les dépassements. En faisant varier le terme dérivé, on obtient les résultats suivants :

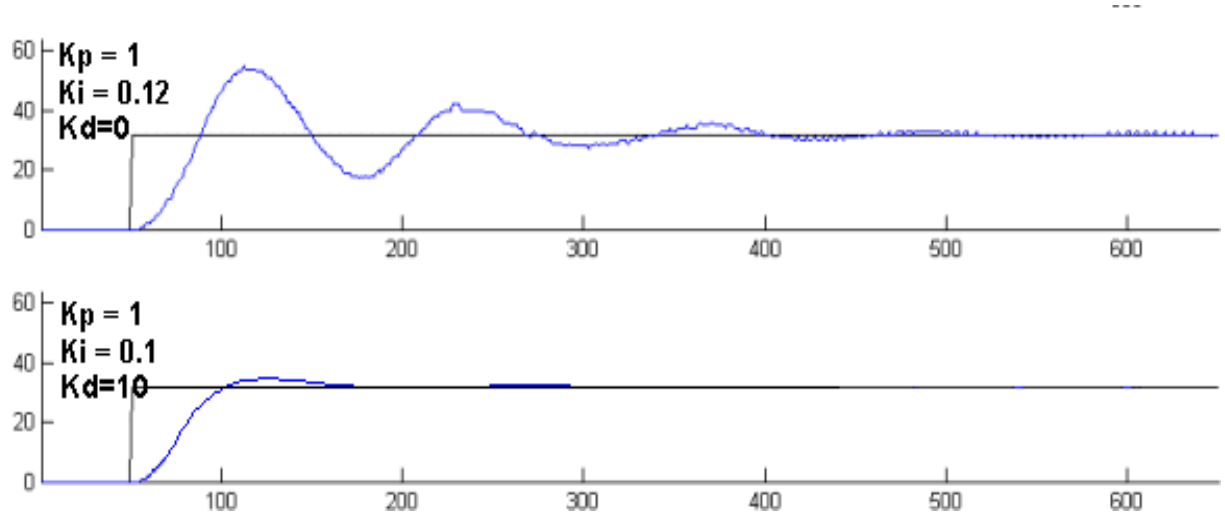


Figure 2.9.2 :courbes d'asservissement Proportionnel Intégrale Dérivée

Comme vous pouvez le voir sur les courbes, en ajoutant le terme dérivé, on diminue le dépassement (en anglais overshoot).

2.11.2. Résumé

Pour résumer, le régulateur PID est un régulateur à boucle fermée qui utilise la différence entre l'entrée et la sortie pour modifier la consigne dans le but d'atteindre une réponse égale à la valeur d'entrée. Pour ceci il utilise 3 termes:

- Le terme Proportionnel qui permet d'augmenter la vitesse de montée (atteint la consigne le plus rapidement possible).
- Le terme Intégral qui réduit l'erreur statique.
- Le terme Dérivé qui réduit le dépassement.

2.12. Implémentation d'un PID sur un robot

Un asservissement PID consiste à mémoriser l'erreur, la somme des erreurs et la différence de l'erreur courante avec l'erreur précédente.

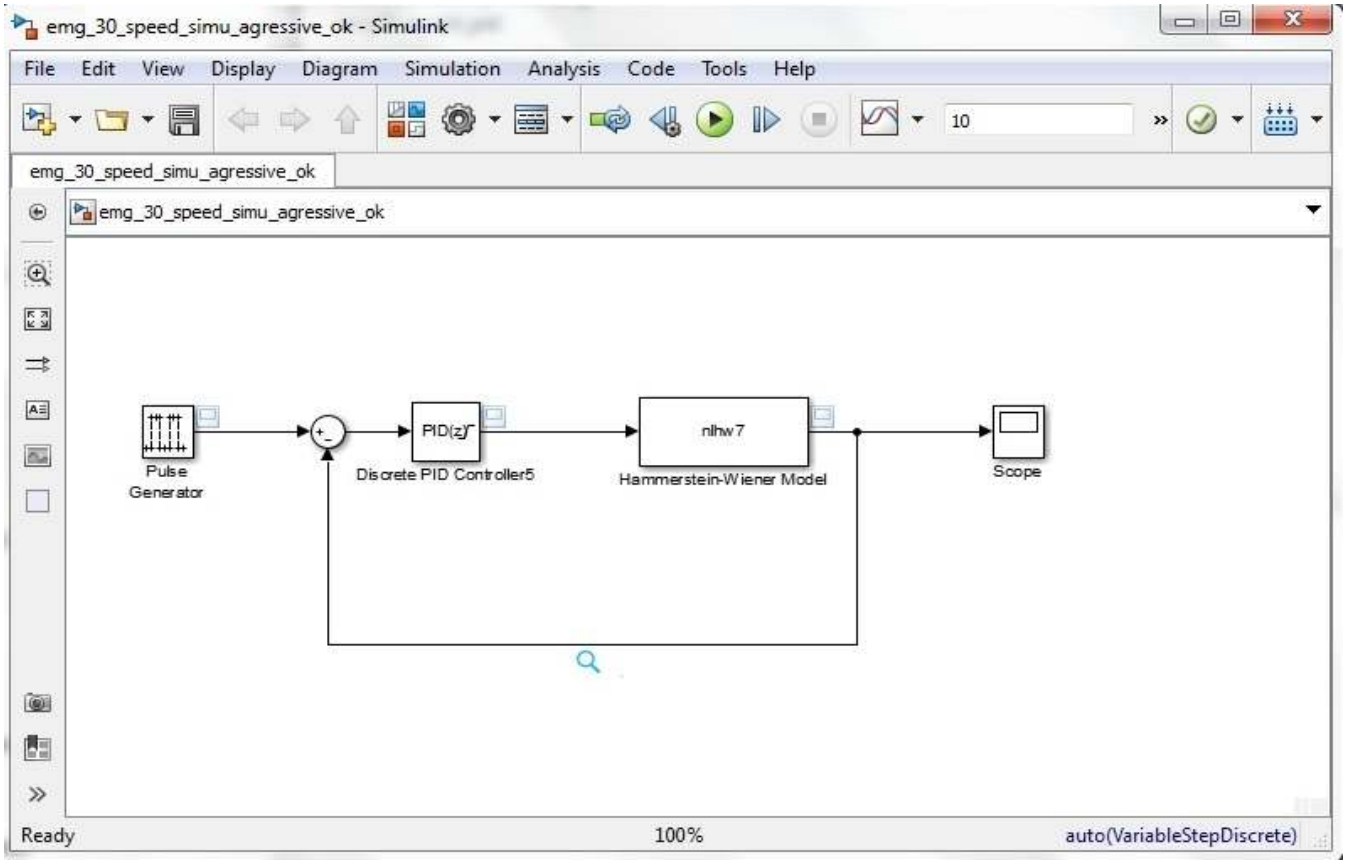


Figure : 2.9.3 un model de simulation du robot

2.13.Régler les coefficients d'un PID

Le réglage des coefficients Kp, Ki et Kd d'un PID peut se faire expérimentalement par essais/erreurs. Tout d'abord, sachez qu'il ne sert à rien de vouloir régler les trois coefficients en même temps ! Il y a trop combinaisons possibles et trouver un triplet performant relèverait de l'exploit. Il vaut mieux y aller par étape.

a. Tout d'abord, il faut mettre en place un simple régulateur proportionnel (les coefficients Ki et Kd sont donc nuls). Par essais/erreurs, il faut régler le coefficient Kp afin d'améliorer le temps de réponse du système. C'est-à-dire qu'il faut trouver un Kp qui permette au système de se rapprocher très vite de la consigne tout en faisant attention de garder la stabilité du système : il ne faut pas que le système réponde très vite tout en oscillant beaucoup !

b. Une fois ce coefficient réglé, on peut passer au coefficient Ki. Celui-là va permettre d'annuler l'erreur finale du système afin que celui-ci respecte exactement la consigne. Il faut donc régler Ki pour avoir une réponse exacte en peu de temps tout en essayant de minimiser les oscillations apportées par l'intégrateur !

c. Enfin, on peut passer au dernier coefficient Kd qui permet de rendre le système plus stable. Son réglage permet donc de diminuer les oscillations.

En général, pour régler ces coefficients, on donne au système une consigne fixe (exemple : pour un moteur : tourne à 3 tours par seconde) et on observe la réponse du système (exemple : l'évolution du nombre de tours par seconde du moteur au cours du temps). Le graphe résultant possède donc cette forme

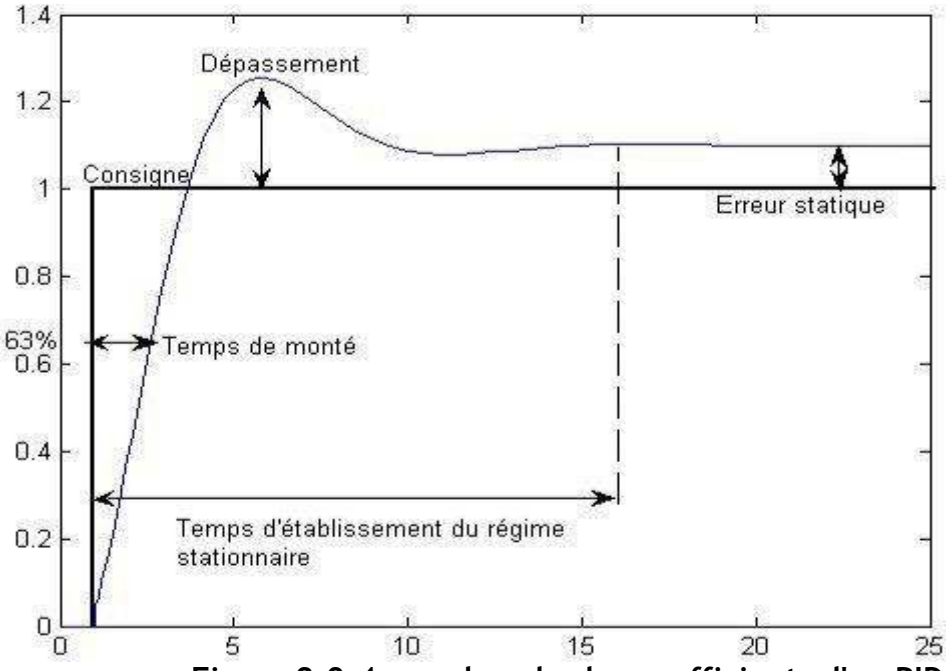


Figure 2.9.4: courbes des les coefficients d'un PID

Le PID parfait n'existe pas, tout est une question de compromis. Certaines applications

autoriseront un dépassement afin d'améliorer le temps de stabilisation, alors que d'autres ne l'autoriseront pas (exemple, contrôler un stylo pour écrire sur une feuille. S'il y a dépassement dans le PID, le stylo traversera la feuille).
Tout dépend donc du cahier des charges. Chacun des coefficients à un rôle à jouer sur la réponse à une consigne :

- L'erreur statique, c'est l'erreur finale une fois que le système est stabilité. Cette erreur doit être nulle. Pour diminuer l'erreur statique, il faut augmenter Kp et Ki.
- Le dépassement, c'est le rapport entre le premier pic et la consigne. Ce dépassement diminue si Kp ou Ki diminuent ou si Kd augmente.
- Le temps de montée correspond au temps qu'il faut pour arriver ou dépasser à la consigne. Le temps de montée diminue si Kp ou Ki augmentent ou si Kd diminue.
- Le temps de stabilisation, c'est le temps qu'il faut pour que le signal commette une erreur inférieure à 5% de la consigne. Ce temps de stabilisation diminue quand Kp et Ki augmentent.

2.14. Asservissement en vitesse d'un moteur avec Arduino

2.14.1. Le moteur et la codeuse

Le moteur que l'on va utiliser est un motoréducteur 29:1 avec une roue codeuse monté sur l'arbre moteur.

Les caractéristiques intéressantes à noter sont :

- 350 tours de roue minutes
- 10150 tours d'arbre moteur minute (environ 170 par seconde)

- 32 transitions montante et descendante de la codeuse par tour d'arbre moteur
- 928 transitions montante ou descendante de la codeuse par tour de roue

2.14.2.Câblage du moteur à l'Arduino

Si on ne dispose pas de shield Arduino pour contrôler un moteur à courant continue, il suffit de câbler une petite interface de puissance très basique.

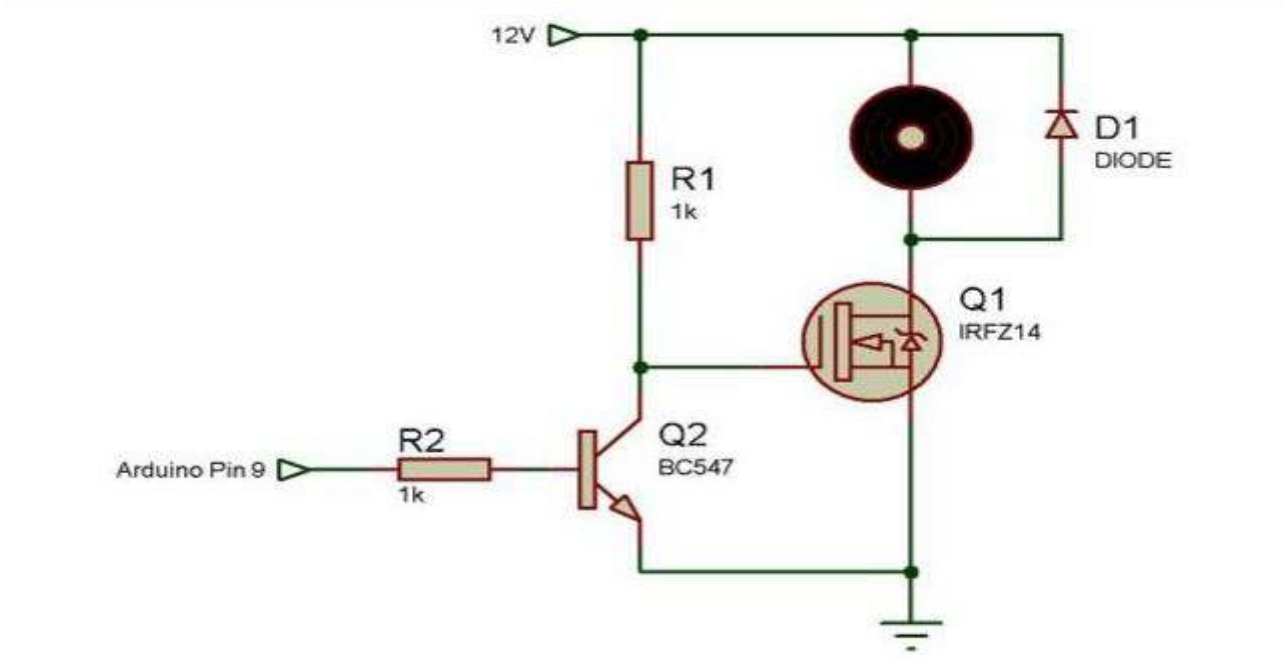


Figure 2.9.5 Câblage du moteur à l'Arduino

Transistor Q2 pilote le MOSFET de puissance. Lorsque la sortie 9 de l'arduino est à l'état haut, Q1 est bloqué et le moteur ne tourne pas. A l'inverse, lorsque la sortie 9 de l'arduino est à l'état bas, alors Q1 devient passant et le moteur tourne à plein régime. Le moteur se contrôle donc en "tout ou rien", ce qui tombe bien, car la sortie "analogique" de l'Arduino est en réalité une sortie PWM de rapport cyclique ajustable.

La sortie 9 de l'Arduino commande le moteur. La codeuse (on n'utilise qu'une seule des deux sorties de la codeuse) est branchée sur la pin 2 qui correspond à l'interruption 0 de l'Arduino (ex : pin n°2 pour une arduino mega).

2.15. Sortie de l'asservissement final PID

Afin de mettre en place un asservissement PID complet, on va rajouter le terme dérivateur, bien que dans notre cas, celui-ci n'ai pas beaucoup d'influence car un simple asservissement PI nous permet d'atteindre une réponse quasi parfaite.

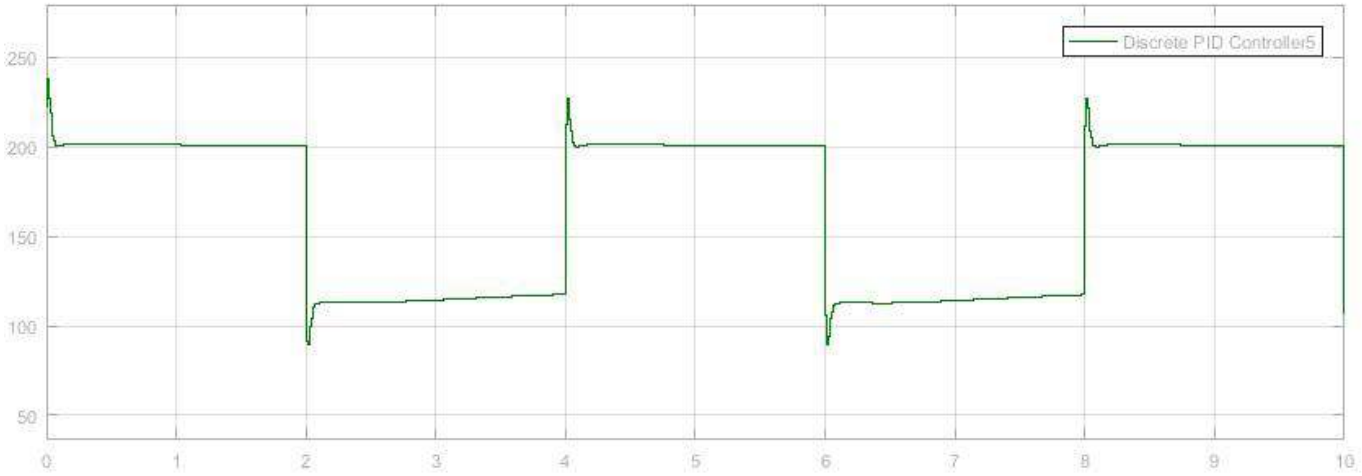


Figure 2.9.6. l'allur de la sortie PID.

Il faut faire attention de ne pas prendre un coefficient dérivateur kd trop grand car le temps de réponse augmente.

Après plusieurs essais, on arrive à un triple assez performant. Voici l'allure de la vitesse

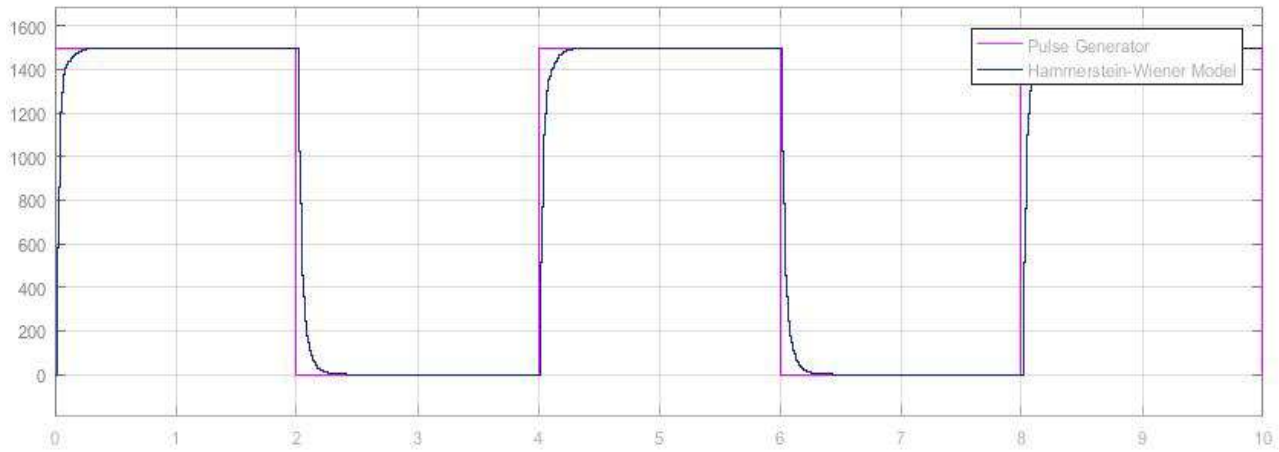


Figure 2.9.7.l'allure de sortie vitesse modele

2.16.Conclusion

En conclusion, il est difficile de préconiser une forme particulière pour la mise en œuvre des régulateurs PID, chacune présentant des avantages et des inconvénients. La forme standard est la plus utilisée dans l'industrie. Il est important de savoir que toutes ces formes sont finalement très similaires et qu'il existe des formules pour passer d'une forme à l'autre. Les différences portent principalement sur l'effet des paramètres de réglage sur le comportement de la boucle de régulation ce qui s'avère fondamental dans le cadre de la mise au point manuelle.

3.1 Introduction

Dans ce chapitre nous allons présenter le fonctionnement de notre robot, et pour bien expliquer cela on va diviser ce chapitre on 3 grandes parties.

a. **Présentation du modèle cmucam5.**
b. **La partie électronique du robot**

3.2 Aperçu générale durobot.

Figure 3.1 aperçu général du robot

3.3 Présentation du module cmucam5

Les caméras vidéos à sortie numérique disponibles sur le marché offrent des possibilités étonnantes. Ainsi associés à un microcontrôleur doté d'un algorithme approprié, ces dernières pourront être utilisées pour détecter pratiquement n'importe quoi dans leur champ de vision.

Toutefois leur utilisation pose généralement des problèmes récurrents. En premier lieu, celui de devoir disposer d'un microcontrôleur assez puissant et rapide pour pouvoir gérer le très grand nombre de données que ces dernières fournissent (pouvant aller jusqu'à plusieurs dizaines de méga-octets par seconde).

Quand bien même nous disposerions de ce type de microcontrôleur, ce dernier serait alors monopolisé en grande partie pour la gestion de l'algorithme de reconnaissance vidéo, nous laissant alors peu de ressources pour les autres tâches de notre application.

La caméra CMUcam5 Pixy contourne ces problèmes à l'aide d'un capteur prêt à l'emploi pouvant être utilisé (pour des applications ludiques) avec la plupart des microcontrôleurs (performants ou non). Ce module est architecturé sur la base d'un capteur vidéo numérique Omni vision OV9715, 1/4", 1280 x 800 pixels associé à un très puissant microcontrôleur NXP™ LPC4330 - 204 MHz - dual core.

Lorsqu'on considère un système a vision embraqué on a un nombre d'information plus grand par rapport aux autres capteurs qui sont utilisés pour recevoir l'information de l'environnement, le capteur de vision fournit une perception de l'environnement alors que le robot fournit l'action réelle.

Le système de vision permet de faire la détection des objets, l'identification, ainsi que la localisation (les angles par rapport à la face du robot, l'estimation de la distance)

Figure 3.2 Le module pixy cmucam5.

3.4 Présentation de la cmucam5

La CMUcam5 se présente sous la forme d'une petite platine électronique principalement composée d'un processeur très haute performance "NXP LPC4330, 204 MHz, dual core" associé à un module capteur/ caméra CMOS "OV9715Omni vision, ¼'', 1280 x 800,L'ensemble est spécialement conçu pour extraire et traiter simplement les données en provenance des images vidéo captées. La CMUcam5 peut être pilotée via un port qui contient de 6 files (UART Série, SPI, I2C, USB, digitale, analogique)

Compatible avec " Arduino" et" Raspberry Pi", le capteur caméra Pixy CMUcam5 réussit là où de nombreux autres capteurs d'images échouent généralement : il est en effet bien difficile d'utiliser ce genre de capteur avec le simple processeur d'une carte de type Arduino, sans que celui-ci ne se trouve complètement saturé.

Avec ce capteur caméra Open-Source proposé par Ada fruit, un grand nombre est possible, il pourra vraiment repérer, suivrez, localiser des objets.

Tout est possible parce que notre processeur disposera encore de tous les moyens nécessaires pour faire tourner d'autres capteurs et actionneurs : caméra de vidéo surveillance, caméra intelligente pour vidéos animalières, robots autonomes, il suffit de laisser parler notre imagination.

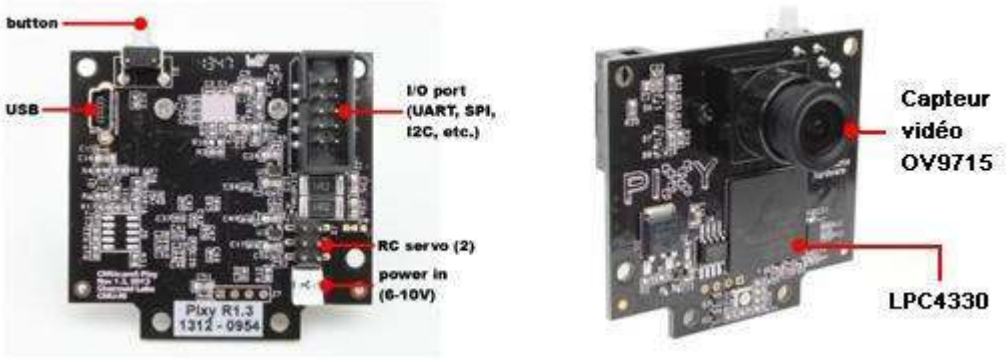


Figure 3.3 le module CMUCAM5 (la face et le font).

3.4.1. Les différentes caractéristiques et spécifications de la cmucam5.

Processeur : NXP LPC4330, 204 MHz, dual core.

Capteur d'image : OV9715 Omni vision, ¼'', 1280 x 800.

Champ de vision de la lentille : 75° à l'horizontale, 47° à la verticale. Type

de lentille : M12 Standard (différents autres types compatibles).

Consommation : 140 mA typiques.

Alimentation : Entrée USB (5 V) ou entrée non régulée (6 à 10 V).

RAM : 264 KB.

Flash : 1 MB.

Sorties de données disponibles : UART Série, SPI, I2C, USB, digitale, analogique.

Dimensions : 53,34 x 50,8 x 35,56 mm.

Poids : 27 g.

50 images par seconde.

Pour toutes les bibliothèques Arduino, Raspberry Pi, Banana Pi, PcDuino et Python

Compatibilité C/C++.

3.4.2. Principe de fonctionnement

Pixy fonctionne selon un algorithme de filtrage des couleurs pour détecter des objets, le filtrage par couleur est apprécié car il est une méthode simple et efficace, la plupart des personnes connaissent **RGB** (rouge, vert, bleu) pour représenter les couleurs, Pixy calcule la couleur (nuance) et la saturation de chaque pixel **RGB**.

Le système de caméra fonctionne avec un algorithme permettant de déterminer quand un objet commence ou arrête de se mouvoir, Pixy compile les différentes tailles et positions de chaque objet (**X, Y,** hauteur, largeur)et les enregistre via l'une de ses interfaces (par exemple : SPI), Par exemple une balle orange, il suffit de la placer devant Pixy et appuyer sur la touche. C'est simple et rapide, après cela le robot suit la balle.

Le cmucam5 utilise les paramètres (**X, Y**, Hauteur, largeur) pour reconnaitre la position de chaque objet.

3.4.3. La détection des objets

a. Des centaines d'objàets détectables par la CMUCAM5

La CMUcam5 Pixy peut déceler des centaines d'objets à la fois dans son champ de vision.

Pour ce faire elle utilise un algorithme de composants connectés pour déterminer où un objet commence et où il se termine. Elle compile ensuite les dimensions et l'emplacement de chaque objet et les signale par une de ses interfaces (par exemple via un bus SPI).

Le capteur cmucam5 est un capteur très rapide et performent, donc il peut détecter et enregistrer jusqu'à 7 couleurs différentes, numérotées de 1 à 7 (signature) et la peut voir cela à partir de l'interface de programmation le logiciel (pixy Mon).

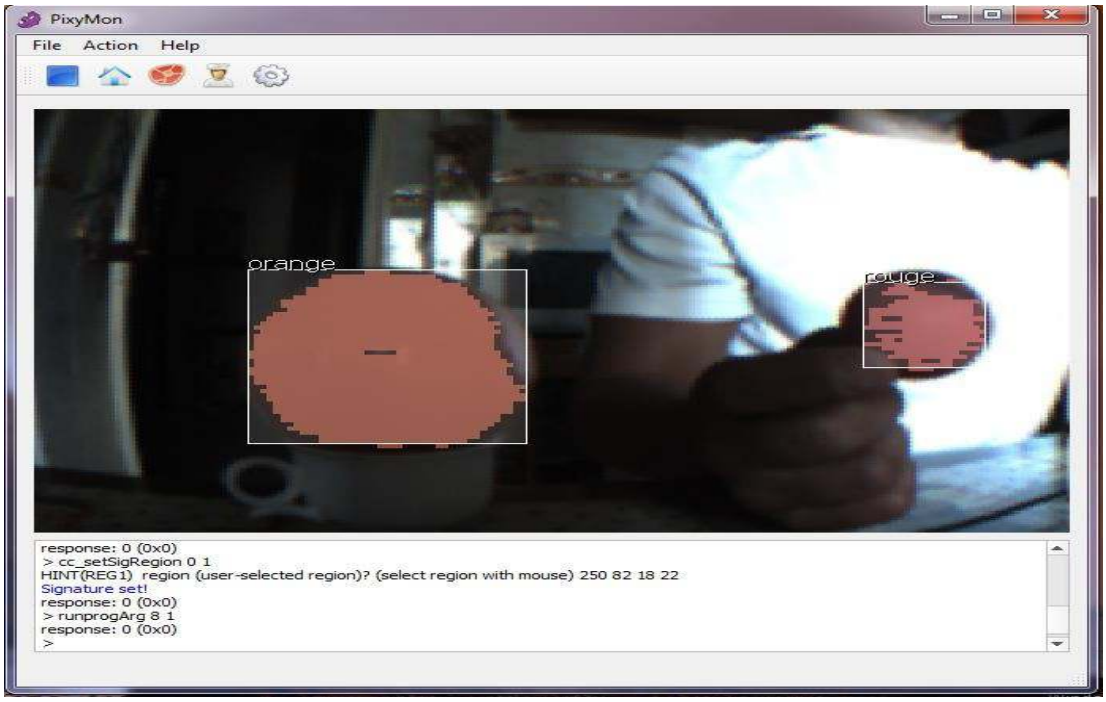


Figure 3.4 l'interface de programmation de cmucam5.

Pour détecter et identifier les objets il faut sélectionner « action » de la barre de fonctions, puis choisir « set signature 1.2…etc.», et sélectionner l'objet qu'on veut suivre.

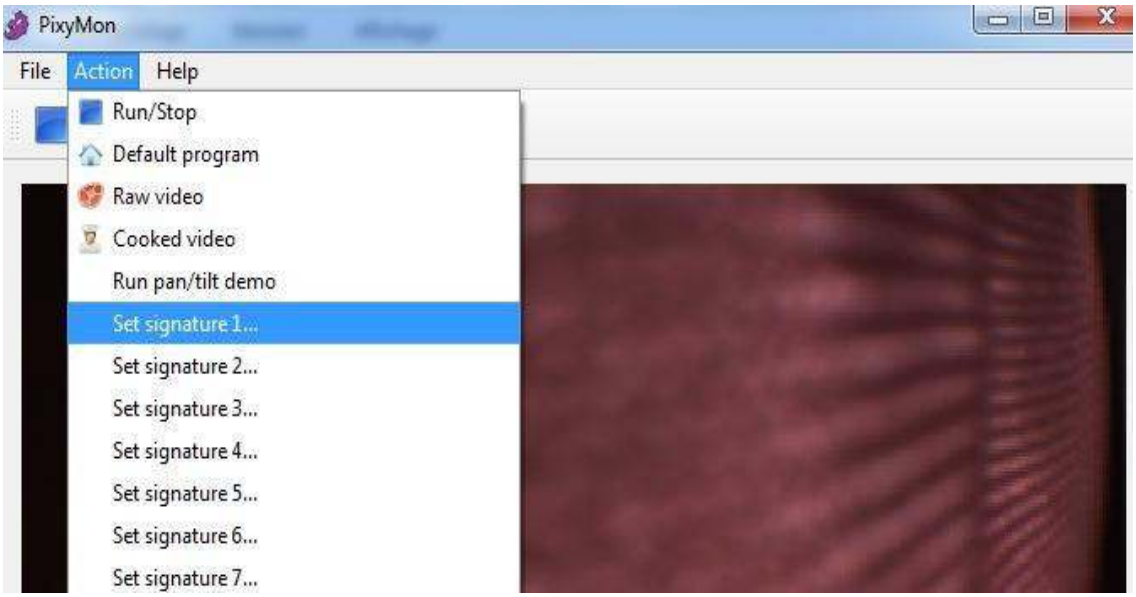


Figure 3.5 sélectionné des objets

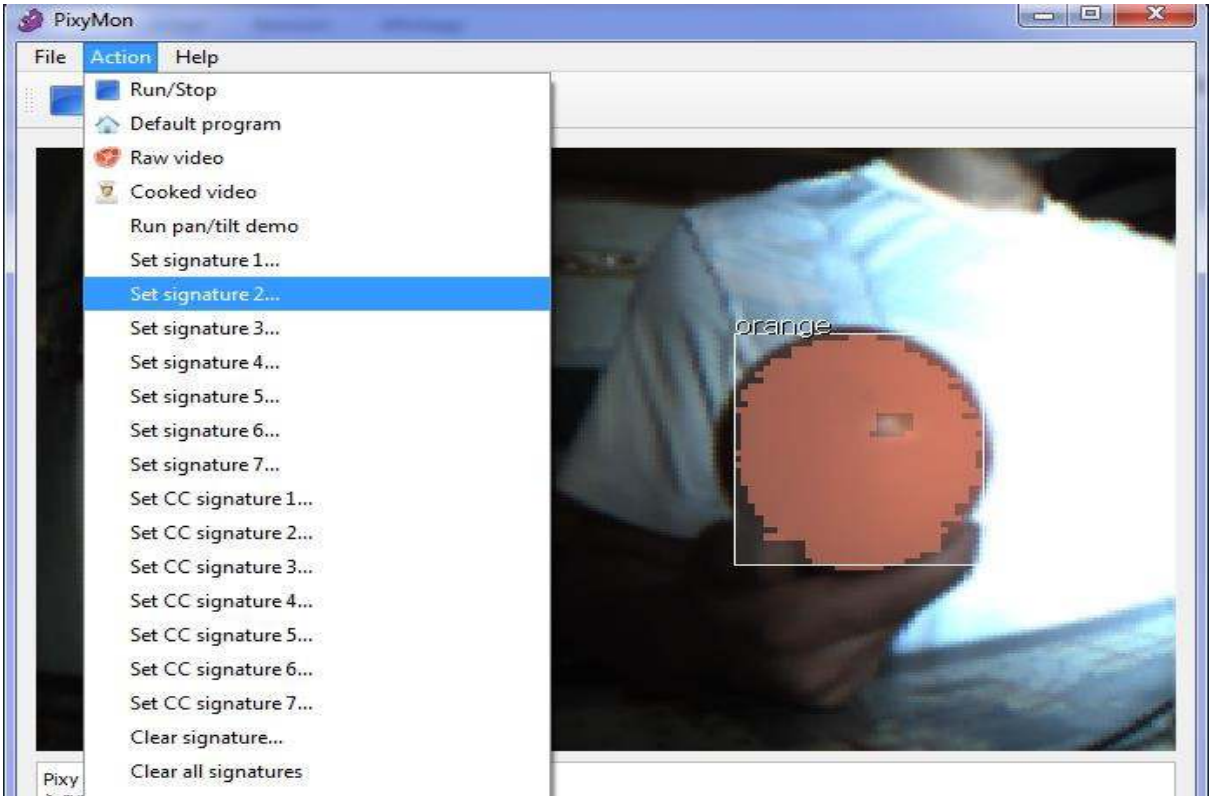


Figure 3.6 l'objet capté signature 2

b. vitesse de laCMUCAM5

La CMUcam5 est très rapide. Cette dernière traite toute une trame d'image 640 x 400 chaque 1/50emede seconde (20 millisecondes). Cela signifie que nous obtenons une mise à jour complète de tous les objets détectés toutes les 20 ms. À ce rythme, le suivi de la trajectoire d'une chute d'une balle bondissante est possible.

Figure 3.7 la vitesse de détection de la cmucam5

3.4.4.Domaines d'applications

La cmucam5 détermine la position centrale et la taille de chaque objet pour offrir de nombreuses possibilités.

a. Technique robotisée

Choisir un objet à enregistrer pour le robot à l'aide de l'un des sept codes couleurs, Pixy déterminera sa position, son orientation et sa taille. Comme nous allons le faire dans notre projet. Alors on va découvrir un angle presque de 75 ° horizontal et de 47° vertical.

b. Système d'alarme

Pixy détecte l'arrivée d'un objet directement via la sortie numérique.

c. Automatisation

Nous souhaitons différencier plusieurs objets en fonction de leur couleur et de leur taille, Pixy détermine la position et la taille de chaque objet en 20 millisecondes, Il est possible de mémoriser 7 objets différents aux couleurs différentes.

d. Jeu de balles

Pixy détermine la position d'un objet avec une cadence de 50 Hz. Il est possible de déterminer la vitesse et l'orientation de chaque balle

3.4.5.Les avantages et les inconvénients de la cmucam5

a.Points forts

Petite, rapide, facile à utiliser et économique, ce système de visualisation délivre 50 images par seconde Pour toutes les bibliothèques pour Arduino, Raspberry Pi, Banana PI, PcDuino, etc. Prend en charge C / C++ et Python.

b.point de faible

L'utilisation de capteur de vision présente néanmoins deux inconvénients: de nombreux processeurs n'ont pas une capacité suffisante pour traiter cette quantité de données.

3.5.Partie électronique du robot

La partie électronique joue un rôle très important quand nous allons commencer à réaliser un système dans domaine de technologie moderne, car elle constitue la base de fonctionnement de ce système.

Pour la mise en marche de tous les systèmes mécaniques, notre robot doit disposer de plusieurs cartes électroniques qui assurent le bon fonctionnement de celui-ci tel que la carte de commande (Arduino méga 2560) et (Arduino uno) puis la carte de puissance (L298), ainsi que le capteur de couleurs (cmucam5) et les autres capteurs.

3.5.1.Structure générale du programme

Comme tous les projets d'électronique la machine est constituée d'une carte de commande pour piloter et vérifier et pour assurer le bon fonctionnement de ce système, alors dans notre projet on va utiliser la carte arduino de type méga 2560 de la famille °ATMEL°.

.*Structure générale du programme

Faire l'apprentissage pour informer le robot pour suivi un objet. La caméra détecte l'objet.

les vitesses des roues gauches et droites seront calculées à partir des encodeurs.

la position absolue du robot sera mise-à-jour.

la vitesse angulaire et la vitesse d'avance du robot seront calculées, à partir de l'emplacement de l'objet que nous voulons suivre.

on change la position de l'objet.

les nouvelles vitesses pour chaque roue seront calculées.

les nouvelles vitesses pour chaque roue seront fournies comme consigne à deux régulateurs PID assurant la commande des moteurs.

Le reste du temps sera réservé à la communication par l'UART (nouvelles Consignes, transmission de la position actuelle, faire l'action (suivi l'objet)).

3.5.1.La carte de puissance

La carte de puissance est le deuxième élément important dans le circuit, c'est la partie qui lie la carte commande aux deux moteurs du robot.

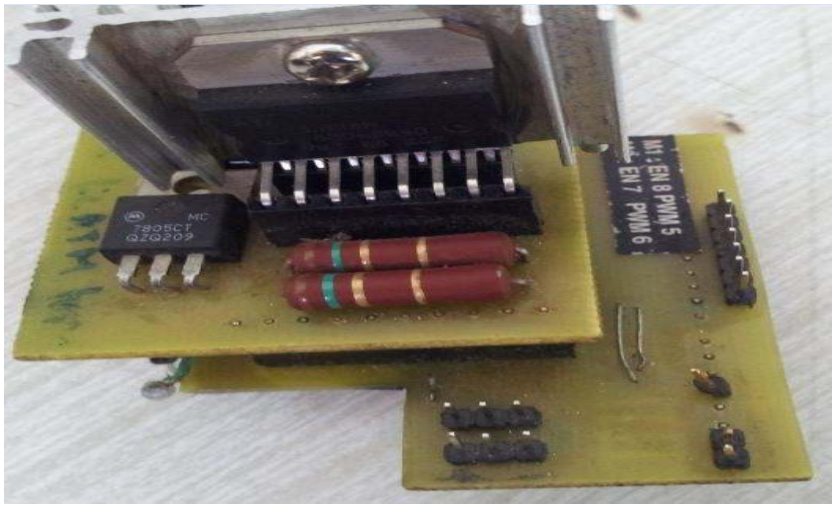


Figure 3.8 la carte de puissance.

a. La carte de puissance à base de L298

Dans notre cas, en à utilisé des moteurs à courant continu dont la vitesse de rotation (donc la vitesse du robot) est proportionnelle à la tension d'alimentation, il va donc falloir être capable de produire une tension variable et commandée. Pour cela il existe un montage appelé montage pont en H (H- BRIDGE en English), ce montage est constitué de 4 transistors montés en H.

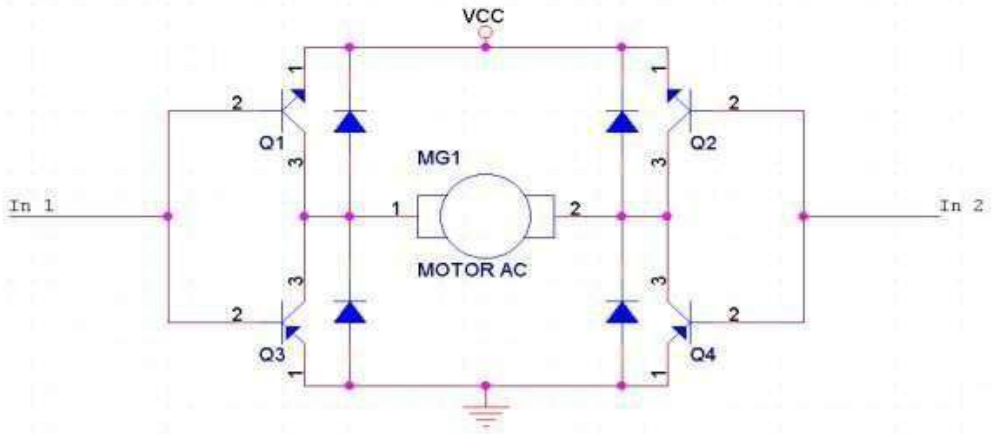


Figure 3.9 schéma bloc de pont H.

b. Les caractéristiques du L298 sont les suivantes

Intensité maximale : 2A par pont.
Alimentation de puissance : 5.5V à 50V.
Type de boîtier : Multiwatt15.
Dissipation puissance total : 25w.
Trois entrées par pont : In1, In2 et ENABLE.

c. utilisation avec les deux moteurs

La carte de puissance est constituée de deux ponts en H et d'une partie d'alimentation, le L298 fonctionne avec une tension d'alimentation logique (VCC) de 5V, et avec une tension d'alimentation du moteur de 12V.

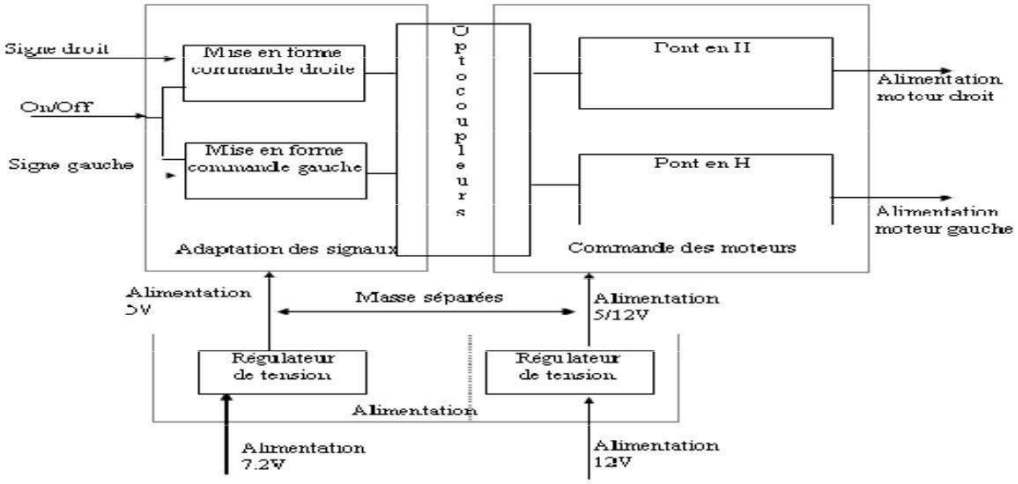


Figure 3.10 schéma principale de la carte de puissance.

d. principe de fonctionnement

Le moteur à besoin de tourner dans les deux sens de rotation, on utilise alors un dispositif nommé pont en H.

Le pont H est une structure électronique servant à contrôler la polarité aux bornes d'un dipôle, il est composé de quatre éléments de commutation généralement disposés schématiquement en une forme de H d'où le nom.

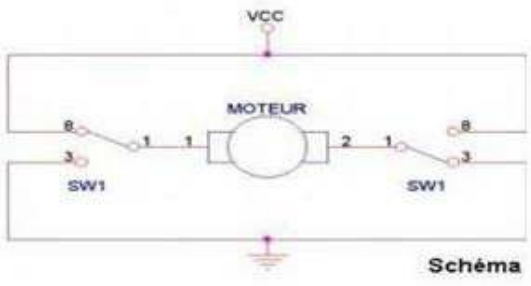


Figure 3.11.a la 1er sens de rotation.

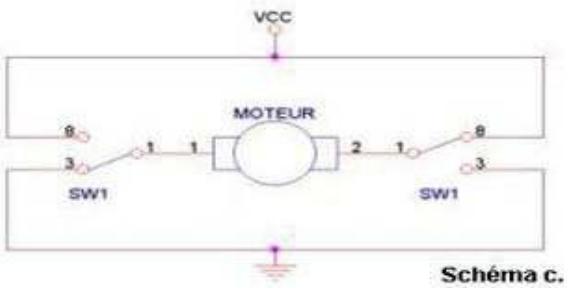


Figure3.11.b le deuxième sens de rotation.

e. Remarque:

Le pont en H permet d'effectuer un freinage magnétique s'il est capable d'en dissiper la puissance générée. Cette opération s'effectue en actionnant soit les deux commutateurs supérieurs ou inférieurs en même temps, ce qui court-circuite. Le tableau ci dessue illustre la circulation du courant.

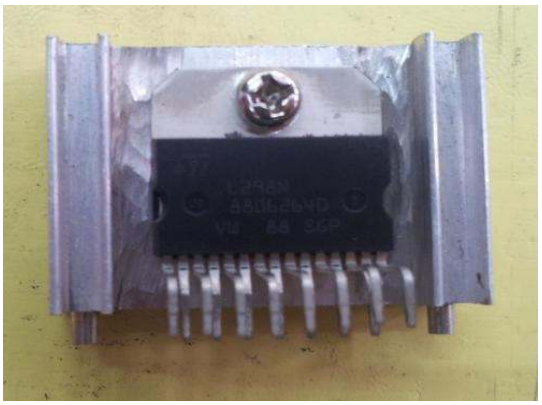


Figure4.12 le circuit L 289 (pont H).

3.5.3.Le moteur à CC(EMG30)

L'EMG30 (**moto réducteur)** est un moteur à courant continue de 12v totalement équipé avec encodeur et une boite à engrenages à réduction Il est idéal pour les applications robotiques, fournissent un retour d'informations et une commande efficaces a l'utilisateur.

Il comprend également un condensateur de suppression de bruit.

Figure 4.13 Moteurà CC (EMG 30)

Tension : 12v.

Démultiplication : 30 ,1.

Couple nominale : 1,5Kg/cm.

Vitesse : 170tr/mn (216 à vide).

Courant : 530mA (150 à vide).

Courant de décrochage : 2,5A.

Puissance nominale : 4,22w.

Nombres d'encodeurs par tour : 360 impulsions.

Dimensions : Φ28, 5×86,6mm (axe de 10mm inclus).

3.6.Conclusion

Dans ce chapitre on a présenté l'essentiel de la partie matérielle du robot, à savoir les parties électronique

4.1 Introduction

Après avoir fait l'étude théorique de l'ensemble des organes de notre robot, nous allons passer aux différents tests afin d'évaluer les résultats obtenus dans notre travail. D'après les différents modèles de robots existants nous avons opté pour une architecture de type unicycle.

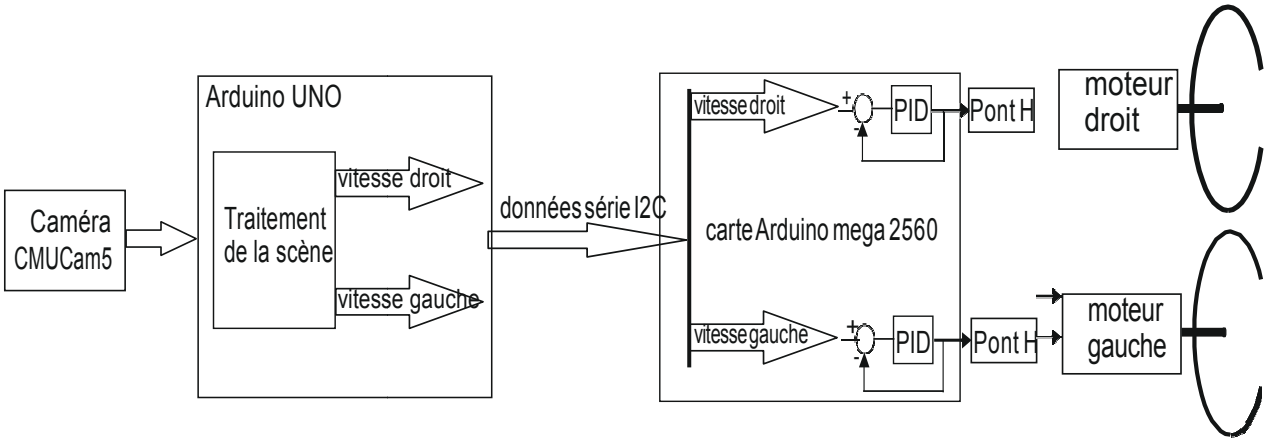


Figure 4.1. aperçu final et synoptique global de notre robot.

4.2. Partie programmation

D'une manière générale et particulièrement lorsqu'il s'agit d'un système embarqué le langage le plus utilisé est le C/C++. L'environnement de programmation arduino IDE est à base de C++, ce dernier intègre des fonctionnalités propres aux cartes arduinos, tels que l'accès aux différents périphériques d'entrées/sorties numériques et analogiques ou encore les différents protocoles de communications séries (RS232, I2C, SPI).

4.2.1. Outils de programmation de la CMUcam5 (Pixy)

Pixy peut détecter littéralement des centaines d'objets à la fois. Il utilise un algorithme de composants connectés pour déterminer un objet et leur extrémité. Pixy compile ensuite les tailles et les emplacements de chaque objet et les signale à travers l'une de ses interfaces (par exemple SPI).

4.2.2. Implémentations de CMUcam5 avec le PC

On peut configurer notre caméra avec le PC à l'aide du logiciel «PixyMon». Ce dernier nous permet de configurer la caméra pour choisir et suivre l'objet désiré tout en tenant compte de l'environnement (lumineux) dans lequel cet objet se situe. Une fois que l'objet en question est délimité par sa couleur, l'appui sur le bouton 'apply' permet de mémoriser cette information sur le module Pixy.

La figure 5.2 illustre un exemple de choix d'une balle orange comme cible à suivre.

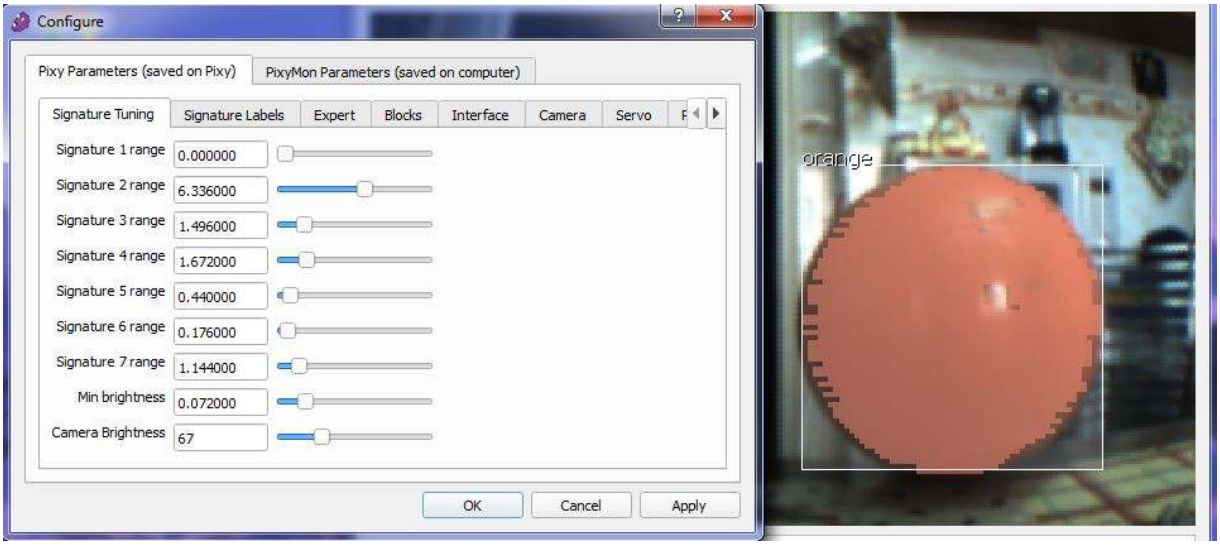


Figure 4.2. Exemple de configuration du module Pixy à l'aide de 'Pixymon'.

4.2.3. Implémentations de la cmucam5 avec le robot

Une fois la couleur de l'objet et la surface de celui si choisies nous connectons le robot avec la carte arduino et on visualise les paramètres de déplacement de l'objet détecté (X et Y) ainsi que sa hauteur et sa largeur, la figure 5.5 illustre ces paramètres.

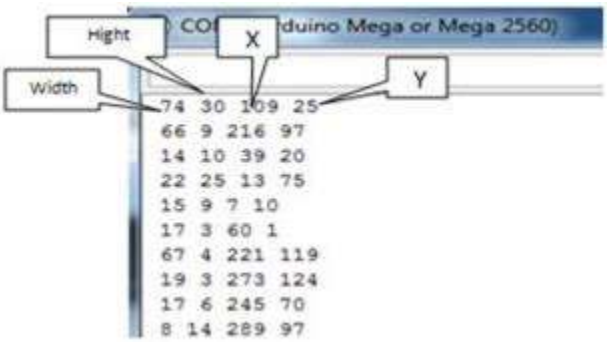


Figure 4.3. l'affichage sur le sériel moniteur.

a. Communication du module CMUcam5 (Pixy) avec l'arduino

L'exploitation de cette caméra par notre robot se fait à l'aide d'un microcontrôleur, ce dernier permet la communication entre la caméra CMUcam5 et le robot, il permet aussi de recevoir l'information acquise par la caméra et de donner l'ordre au robot de faire des actions déjà sélectionnés par notre programme.

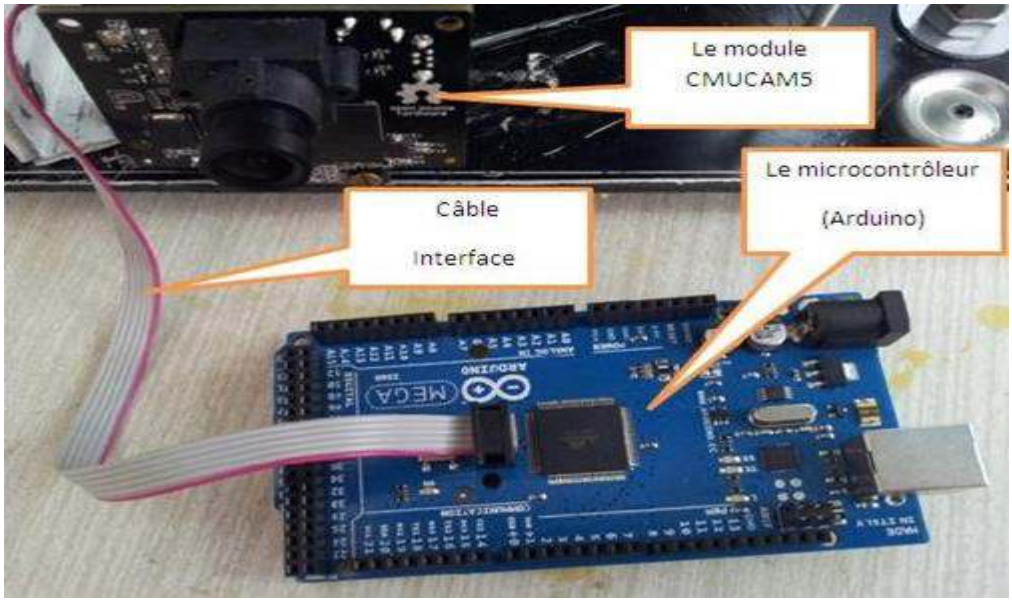


Figure 4.4. L'emplacement de la cmucam5 avec l'arduino

4.3. Structure des tâche du robot

Dans cette partie nous allons montrer le fonctionnement et les tâches que va accomplir notre robot

- Reconnaître les couleurs des objets dans l'image (Traitement de l'image (première partie)).
- Calculer la position des objets par rapport au robot étalonnage géométrique de la caméra et du système (deuxième partie).

4.3.1. Réaction du robot par rapport à la distance et la surface

Pour notre travail nous avons besoin d'une balle colorée, orange dans notre cas, pour laquelle nous avons choisi la signature 2 et nous avons ajusté la luminosité et la surface de l'objet, cette configuration s'effectue à l'aide du logiciel 'PixyMon'.

En premier lieu nous sélectionnons la balle puis nous définissons une distance par rapport au robot, afin de déterminer la trajectoire suivant la surface de la balle. Dans notre exemple la surface de notre balle et de (4000 pixels) cette valeur correspond à une distance du robot d'environ 40 cm par rapport au robot, il s'agit d'un choix expérimental, qui peut facilement être modifié.

Afin de tester le fonctionnement de notre système nous avons élaboré un programme d'application qui va faire en sorte que le robot ait le comportement illustré par l'organigramme ci-dessous.

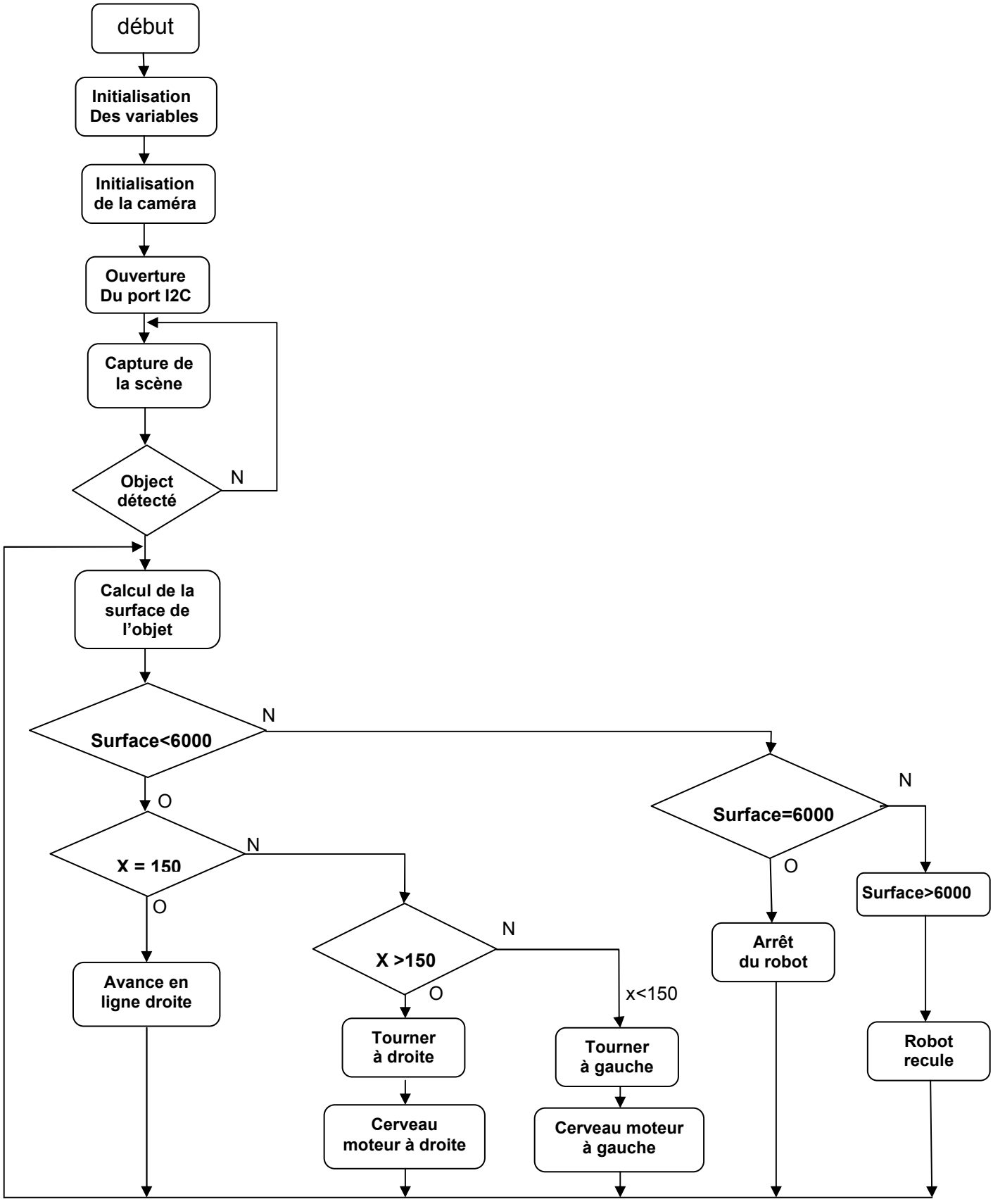


Figure 4.6 : Organigramme décrivant les mouvements du robot

Si la balle et dans la signature 2 le programme calcule la surface de l'objet et suit les instructions suivante :

Surface = ancienne surface + la hauteur*la largeur.

Nous avons pris la valeur de la surface initiale =0.

Si la surface et inférieur à 6000 le robot se déplace vers l'avant, et si la surface et supérieur à 6000 le robot de déplace vers l'arrière, et enfin quand la surface et égale à 4000 pas de mouvement pour le robot.

a. Distance = 60 cm

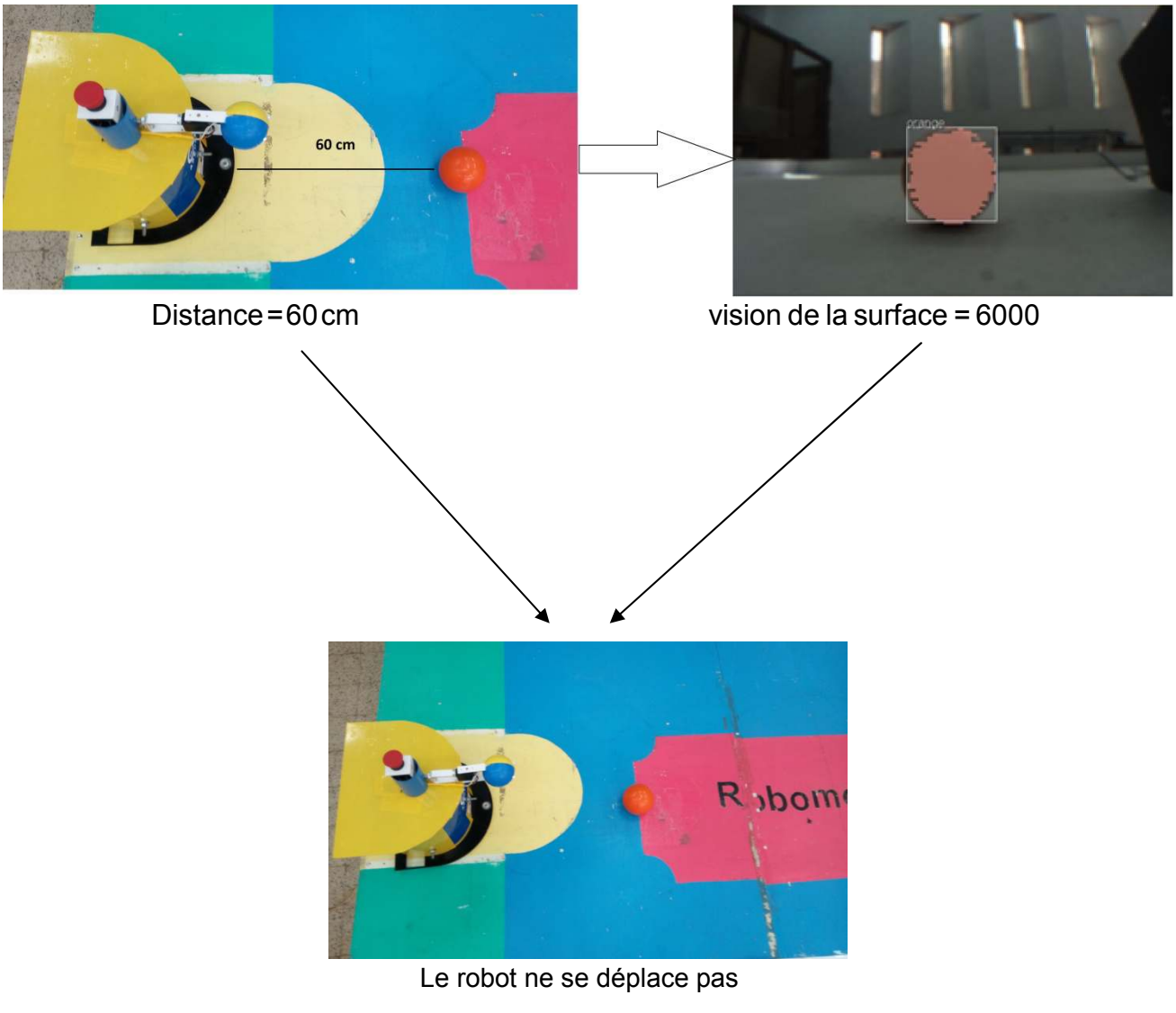
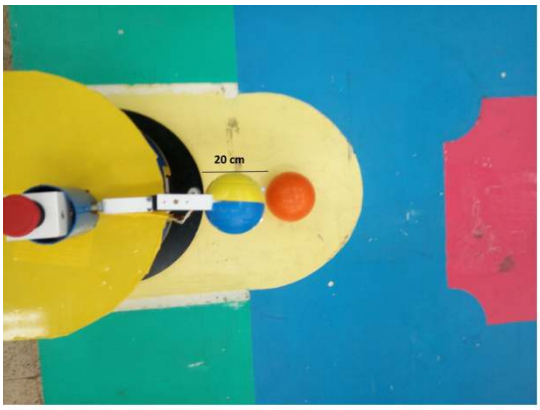


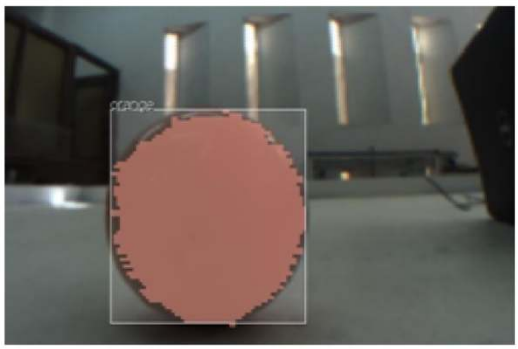
Figure 4.7. réaction du robot à une distance de 60cm et une surface égalant 6000

La vision du robot à cette distance et à une surface perçue de 6000 pixels, le robot ne réagit pas (pas de mouvement).

b. Distance = 20 cm



Distance = 20 cm



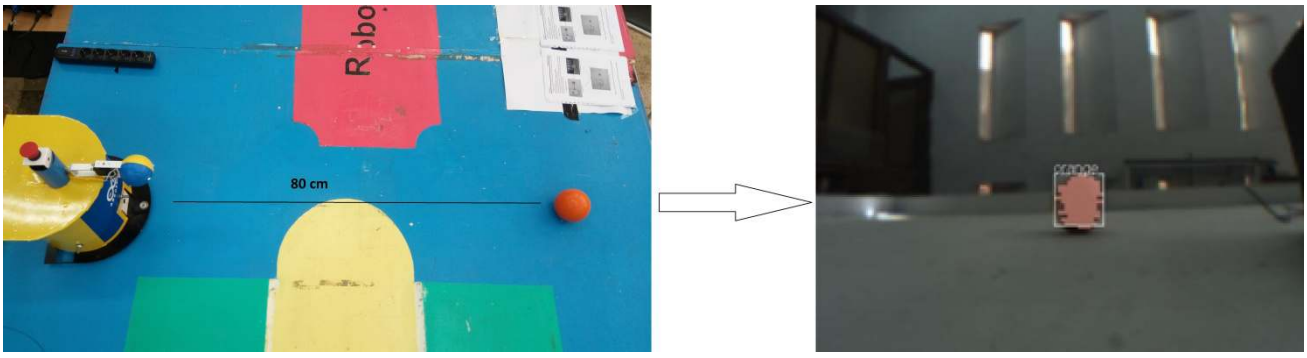
vision de la surface >6000

Le robot fouit la balle

Figure 4.8. réaction du robot à une distance de 20 cm et une surface supérieur a 6000

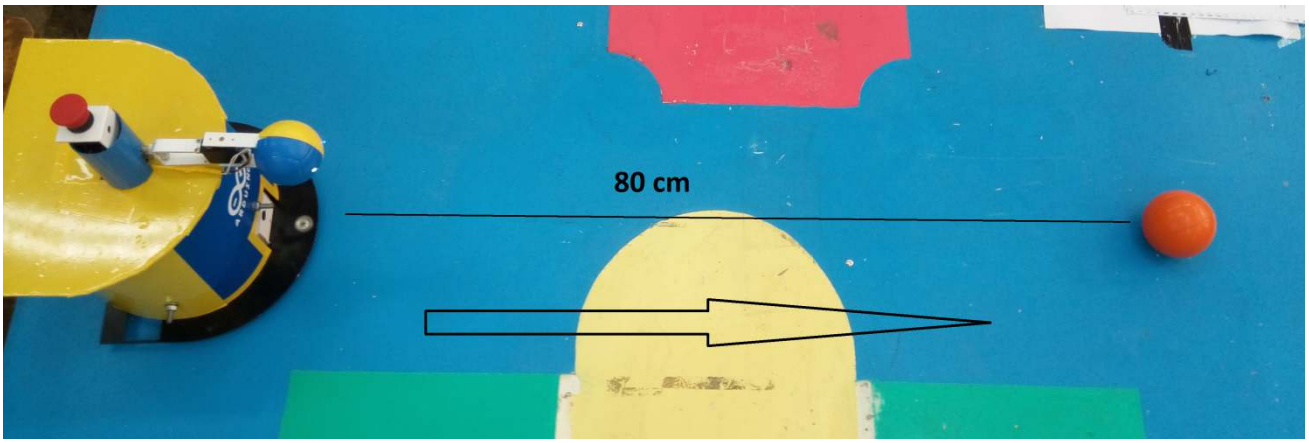
Par contre nous remarquons que la vision du robot à cette distance et à une surface supérieur à 6000 change, le robot se met en mouvement et recule fouillant la balle.

c. Distance = 80 cm



Distance = 80 cm

vision de la surface < 6000



Le robot traque la balle

Figure 4.9. réaction du robot à une distance de 80cm et une surface inférieur à 6000

Enfin la vision du robot à une distance de 80cm (>60cm) et une surface inférieur à 6000 le robot réagit en avançant en direction de la balle.

4.3.2. Trajectoire et la vitesse de déplacement du robot

Suivant notre programme mise en œuvre pour le robot, la trajectoire se définit par rapport à la vision de la CMUcam5 qui communique les informations (coordonnées (X et Y) surface) via le bus I2C à la carte d'Arduino qui à son tour asservi et synchronise les moteur gauche et droit pour une trajectoire optimale.

a. Objet à droite

Par rapport à la vision de la CMUcam5 le robot augmente la vitesse de rotation du moteur gauche par rapport au moteur droit en vue d'ajuster sa trajectoire.

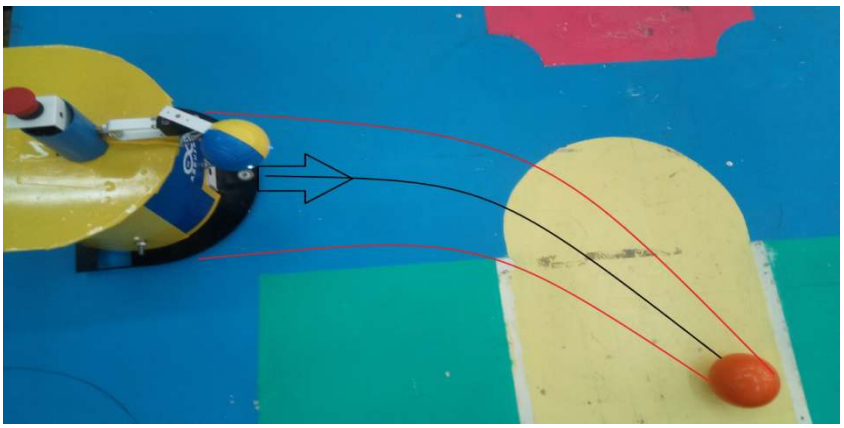


Figure 4.10 trajectoire du robot quand la balle et à droite

b. Objet à gauche

Par rapport à la vision de la CMUcam5 le robot augmente la vitesse de rotation du moteur droit par rapport au moteur gauche, le but étant de s'approcher de la balle le plus vite possible.

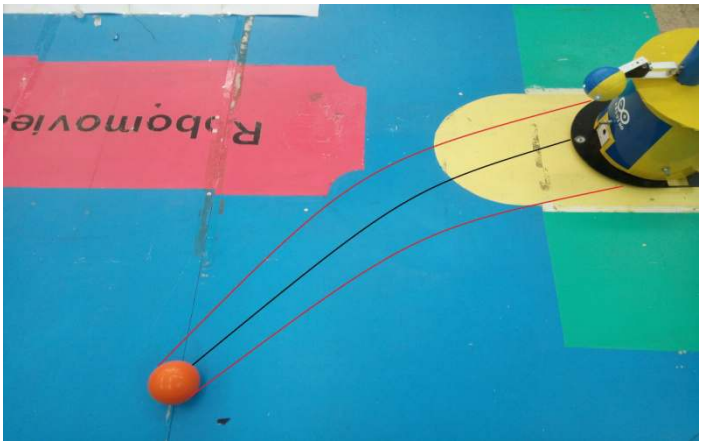


Figure 4.11 trajectoire du robot quand la balle et à gauche

c. Objet (à droite ou à gauche) et surface inférieur à 6000

Suivant l'emplacement (valeur des coordonnées (X et Y) et la surface inférieure à 6000) de l'objet capter par la CMUcam5 le robot accélère et ajuste sa trajectoire instantanément par rapport aux données reçues en direction de l'objet et décélère à nouveau suivant l'augmentation de la surface.

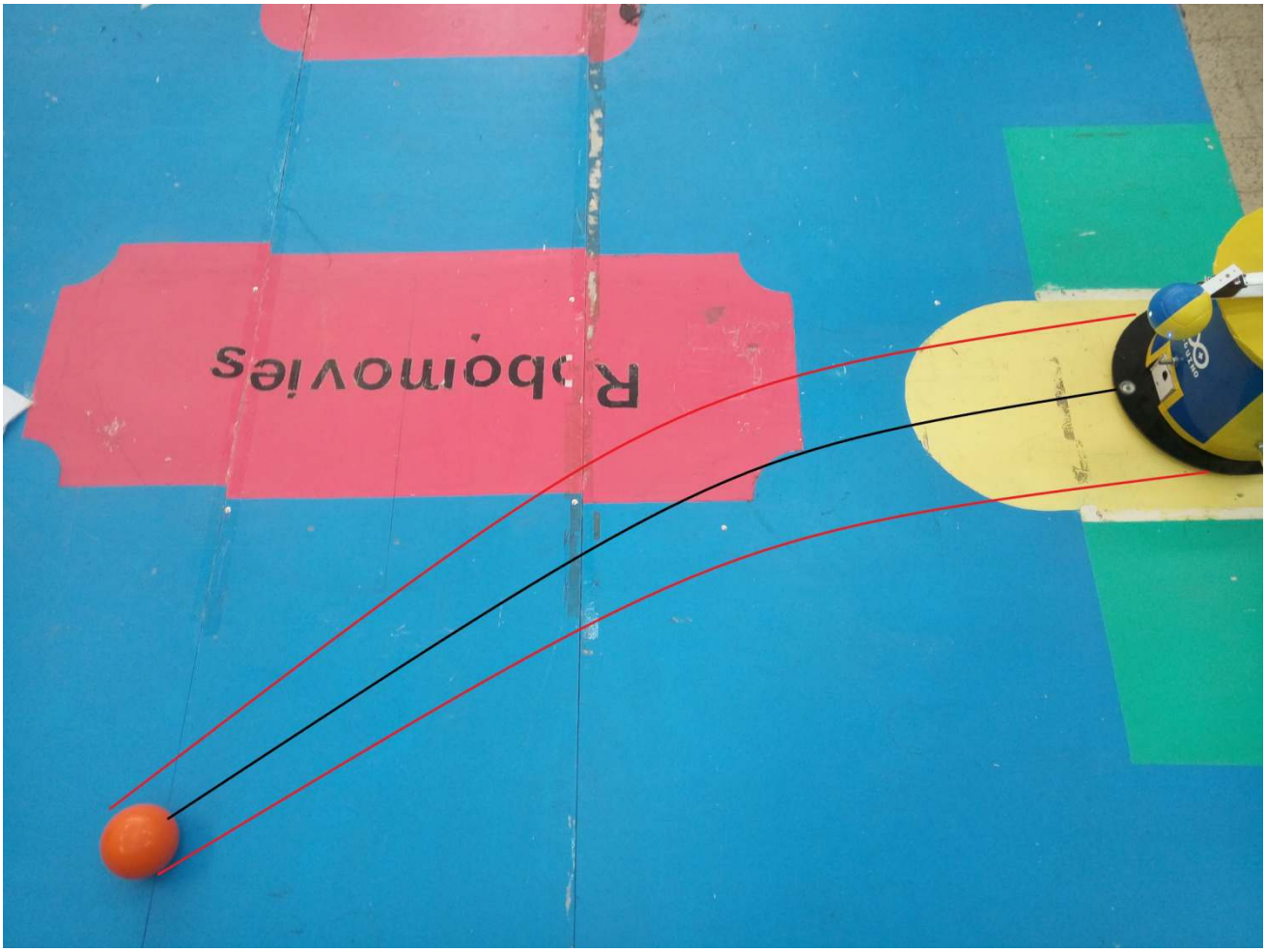


Figure 4.12 trajectoire du robot quand la balle est

à gauche et surface très inférieure à 6000 (balle très loin)

4.4. Conclusion

Dans ce chapitre nous donné l'essentiel des constituants et du comportement de notre robot en fonction de la position de l'objet qu'il doit suivre. Nous avons pris .quelques cas types qui permettent de juger d'une manière assez significative sur le fonctionnement du système proposé.