REPUBLIQUE ALGERIENNE DEMOCRATIQUE
ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT
SUPÉRIEUR ET DE LA
RECHERCHE SCIENTIFIQUE
Université Abou Bekr Belkaïd de Tlemcen
Faculté de Technologie

MEMOIRE DE FIN D'ETUDE POUR L'OBTENTION DU DIPLOME

DE MASTER ACADEMIQUE
Filière : Electrotechnique

Spécialité : Commande des machines électriques

Préparé au Département de Génie Electrique et Electronique (GEE)

Et présenté par :

ABDELLAOUI Zakarya et BENKHELIFA Badreddine

Intitulé du mémoire

Contrôle d'un robot mobile

Sous la direction du Dr. Lotfi BAGHLI
Soutenu publiquement le 20 Juin 2016 devant la commission d'examen
Composée de :

MELIANI Sidi Mohamed	Président	MCA, Univ de Tlemcen
BOUMEDIENE Abdelmadjid	Examinateur	Professeur,Univ de Tlemcen
BENYAHIA Boumediene	Examinateur	MCA, Univ de Tlemcen
BAGHLI Lotfi	Encadreur	Professeur,Univ de Tlemcen

Année universitaire 2015 - 2016

Remerciements

Nous tenons en premier lieu remercier Dieu tout puissant de nous avoir accordé la force et le courage pour mener ce travail à terme.

Nous tenons à remercier sincèrement les membres de jury qui nous ont fait l'honneur d'accepter de juger ce modeste travail.

Nous tenons à adresser nos sincères remerciements à notre encadreur le Dr L.Baghli, Professeur à l'Université Abou Bekr Belkaid de Tlemcen pour sa disponibilité, ses suggestions et remarques et aussi pour nous avoir accordé sa confiance tout au long de ce projet de recherche. Nous le prions de bien vouloir agréer le témoignage de notre plus vive reconnaissance et notre profond respect.

Nous remerciements également nos amis nos collègues de la promotion 2015-2016 de Commande de Machines, qui nous ont toujours soutenu et encouragé au cours de ce PFE.

Dédicaces

Je dédie ce modeste travail :

A ma grande mère et mon grand père

A mon père et ma mère

A mes Sœurs et mes Proches

A ma promo de Commande des Machines Electriques

Et à ceux avec qui je partage de bons souvenirs

A.Zakarya

Dédicaces

Du profond de mon cœur, je dédie ce travail à tous ceux qui me sont chers,

A mon père ABDELHAMID, pour son soutien inconditionnel, ses encouragements, et pour m'avoir permis de réaliser mes études dans les meilleures conditions.

A ma mère HOURIA pour m'avoir soutenu, accompagné et surtout encouragé tout au long de ce travail.

A mon frère ABDELHAK, et toutes mes sœurs,

A tous mes professeurs et ceux qui ont contribué à

la réussite de ce travail.

A tout la promo de Commande des Machines Electriques 2016.

B.Badreddine

Table des matières

Liste des figures

CHAPITRE I : ***Généralités sur les robots mobiles.***

CHAPITRE II : ***Développement d'une application Android***

CHAPITRE III : ***Application mobile de Mahfoud III***

CHAPITRE IV : ***Détecteur d'obstacles***

CHAPITRE V : Conduite du robot

Liste des tableaux

Glossaire

USB:	Universal Serial Bus
GPS:	Global Positioning System
OTG:	On-The-Go
UART:	Universal Asynchronous Receiver Transmitter
ARM:	architectures matérielles RISC
API:	Application programming interface
APK:	Android application package
OpenCV:	Open Computer Vision
RGB:	Red-Green-Bleu
ROI:	region of Interest
IC:	Input Capture
ICI :	Input Capture Interrupt
ICM:	Input Capture Mode
FCY:	Instruction Cycle Rate
PWM:	Pulse Width Modulation
MCU :	multipoint control unit
ISR :	Interrept Service Routine
IVT :	Interrupt Vector Table
US :	Ultra-sonic

Introduction générale

De manière générale, on regroupe sous l'appellation robots mobiles l'ensemble des robots à base mobile, par contradiction 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.

On peut estimer que les robots mobiles à roues constituent la plupart des robots mobiles. Historiquement, leur Étude est venue assez tôt, suivant celle des robots manipulateurs, au milieu des années 70. Leur faible complexité en a fait de bons premiers sujets d'étude pour les roboticiens intéressés par les systèmes autonomes. Cependant, malgré leur simplicité apparente (mécanismes plans, à actionneurs linéaires), ces systèmes ont soulevé un grand nombre des problèmes difficiles. Nombre de ceux-ci ne sont d'ailleurs toujours pas résolus. Ainsi, alors que les robots manipulateurs se sont aujourd'hui généralisés dans l'industrie, rares sont les applications, industrielles qui utilisent des robots mobiles. Si l'on a vu depuis peu apparaitre quelques produits manufacturiers (chariots guidés) ou grand public (aspirateurs autonomes), l'industrialisation de ces systèmes bute sur divers problèmes délicats. Ceux-ci viennent essentiellement du fait que, contrairement aux robots manipulateurs prévus pour travailler exclusivement dans des espaces connus et de manière répétitive, les robots mobiles sont destinés à évoluer de manière autonome dans des environnements peu ou pas structurés. [1]

Nous avons choisi de travailler sur le sujet proposé par Monsieur Baghli pour continue le travail de PFE de l'année dernière précédente. Le nom de ce projet est MAHFOUD III et il correspond à la continuité d'un des premiers projets de développement dans le domaine de la robotique mobile à l'université Abou Bekr Belkaid, et c'est une nouvelle version de l'ancien version Mahfoud II, et il comporte maintenant :

- Une seule carte pour la commande du robot
- Une structure mécanique
- Un capteur ultrason
- Deux codeurs incrémentaux
- Une batterie d'alimentation
- Deux moteurs à courant continu
- Il est connecté à un smartphone Android et une application mobile pour le superviser

Le robot Mahfoud III il à une configuration tricycle à commande différentielle, composé de deux roues motrices et une roue libre.

Mahfoud III, est construit avec deux moteurs à courant continu, un pour chaque roue motrice, il est alimenté par une batterie au plomb rechargeable de 12V.

La carte de commande comporte le microcontrôleur dsPIC programmable en langage, et elle comporte aussi le module de puissance, et le module de liaison série USB.

Le microcontrôleur utilisé est un *dsPIC33FJ128MC802* de Microchip, c'est grâce à lui qu'on peut commander le robot.

Pour notre robot Mahfoud III, on veut commander ses deux moteurs à courant continu, on est donc obligé d'inverser sa polarité, et aussi faire varier sa vitesse, afin de faire cela on a choisi le double pont en H, le L298.

C'est un module avec 2 hacheurs 4 quadrants dans un circuit intégré. Ce circuit intégré permet de piloter des charges inductives tel que les relais, les moteurs pas à pas et aussi des moteur à courant continu ce qui le cas pour nous, il permet de piloter directement sa vitesse dans les deux sens de rotation, et ce qu'il a d'avantage de plus c'est que pour contrôler les deux moteurs à courant continu un seul L298 suffit. [2]

Une façon pour rendre Mahfoud III plus intelligent était de lui associer un smartphone avec lequel il va communiquer par liaison série USB, cette connexion entre le robot et le Smartphone doit assurer l'échange d'informations entre les deux, elle va aussi permettre au robot communiquer à un base de données par le biais de l'application mobile en envoyant ses données internes à notre site web.

Cette application, qu'on va développer, va être utilisée pour repérer les données de positionnement GPS du robot Mahfoud III, et aussi pour la détection et suivi de ligne via la caméra du smartphone. Elle enverra les informations au dspic correspondant à la distance à un obstacle et par rapport à la ligne qu'il doit suivre. Le dspic décidera comment agir pour faire tourner et avancer le robot et lui faire suivre la ligne.

La communication série entre le robot et le smartphone est assuré par le biais d'un câble USB OTG qui permet d'émuler un port série UART sur dspic ou port COM sur PC.

Le mémoire est organisé en 5 chapitres :

- Dans le premier chapitre, nous présentons quelques généralités concernant la robotique mobile en général, ainsi que son historique et on va présenter notre projet Mahfoud III.
- Dans le deuxième chapitre nous traiterons les outils nécessaires au développer d'une application Android.
- Et pour le troisième chapitre, on va te montrer comment on a construit cette application Android et comment faire un échange des données entre le Smartphone et la carte électronique.
- Le quatrième chapitre, sera consacré sur le détecteur d'obstacle le capteur ultrasonique et son utilisation.
- Enfin, le cinquième chapitre sera consacré au conduit du Mahfoud III et son régulation bien sure.
- Et à la fin Nous terminerons par une conclusion ou nous résumerons l'essentiel des travaux exposés dans cette mémoire et les perspectives futures de ce travail.

Et notre objectif est comme suites :

- Objectif 1 : Prendre le programme de l'année passée et le développer pour que Mahfoud 3 soit capable d'éviter les obstacles.
- Objectif 2 : Élaborer un programme en OpenCV qui permet de détecter les différentes lignes en utilisant la vidéo captée par le Smartphone, bien que la partie de contrôle de différents robots peut être différent, la partie de la vision et une partie de prise de décision pourrait être compatible avec tout autre type de robots. Ainsi nous nous efforçons de développer un programme avec l'université qui pourrait être mis à profit par les différents robots.
- Objectif 3 : On prendra la ligne blanche comme la ligne de référence que Mahfoud 3 doit suivre et on voit comment il réagit.

Chapitre 1 :

Généralités sur la robotique mobile

1. INTRODUCTION

La robotique est un ensemble de disciplines (mécanique, électronique, automatique, informatique), elle se subdivise en deux types : les robots industriels et les robots mobiles. Les robots industriels sont généralement fixes, ils sont utilisés dans des nombreuses applications industrielles: l'assemblage mécanique, la soudure, la peinture... Les robots mobiles ne sont pas fixes, ils sont classifies selon la locomotion en robots marcheurs, à roues, à chenilles... comme ils peuvent être classifié selon le domaine d'application en robots militaires, de laboratoire, industriels et de services.

Les robots mobiles sont largement utilisés dans les environnements industriels, le plus souvent pour des tâches répétitives en suivant un chemin bien défini matérialisé parfois par des lignes sur le sol ou par l'utilisation d'amers artificiels. Cependant actuellement il y'a une forte tendance à élargir les milieux où évoluent les robots à des environnements domestiques. Les types d'applications possibles sont innombrables. Cela peut aller des tâches de nettoyage et d'entretien, à une assistance à une personne handicapée dans des tâches d'exploration et de préhension. On parle alors, de façon générale, de robotique d'intérieur.

Un tel cadre d'utilisation requiert que le système robotisé dispose d'un niveau minimum d'autonomie et de facilités de navigation. Pour cela, le système doit obligatoirement accomplir trois tâches de base qui sont la localisation, la planification et la navigation. Il faut toute fois noter que les deux dernières tâches sont tributaires de la bonne exécution de la première. [5]

En effet la localisation est l'une des fonctions essentielles qui permet aux robots mobiles de se mouvoir en respectant les règles élémentaires de sécurité et d'évoluer, de façon générale, vers une autonomie totale. C'est pourquoi, depuis l'origine des travaux entrepris en matière de robotique mobile, la localisation de l'engin dans son environnement a constitué une voie de recherche privilégiée qui conditionne les résultats expérimentaux et les applications en milieu industriel et/ou domestique. Elle consiste à calculer et à maintenir à jour la connaissance de la position et de l'orientation du robot dans un repère absolu lié à l'environnement. On se restreint dans notre travail au cas de systèmes navigant sur un plan.

Actuellement la localisation des robots mobiles est un thème de recherche ouvert puisque aucune méthode globale n'est susceptible de générer des algorithmes suffisamment robustes, rapides et fiables pour être appliqués à tous types de problèmes.

2. HISTORIQUE :

Le concept de robot mobile autonome est apparu vers la fin des années soixante, de deux sources totalement différentes : tout d'abord des recherches menées au Stanford Research Institute sur les possibilités d'équiper des machines de capacités de déduction et de réaction logique à des événements

extérieurs. On a ainsi construit Shakey, machine à roues reliée à un ordinateur et équipée d'une caméra lui permettant d'acquérir des images de son environnement. Elle évolue dans un univers de cubes et de pyramides de tailles et de couleurs différentes. Shakey a pour mission de prendre un objet et de le porter ailleurs, quelque soit sa position ; chaque mission dure près de cinquante minutes.

D'autre part, l'industrie nucléaire a besoin de machines permettant d'agir à distance dans des environnements encombrés et inaccessibles à l'homme. L'entreprise américaine Général Electric développe alors un quadrupède pour essayer de résoudre ce problème, tandis que les projets Luna et Mars Rover s'échafaudent dans le but d'explorer des planètes sans que l'homme ne prenne part au voyage.

Pendant plusieurs années, laboratoires, industriels, informaticiens et mécaniciens vont continuer leurs travaux en parallèle. On accède ainsi côté industriel à la télé-opération et à une partie de la robotique classique, tandis que du côté informatique, on assiste à de grands progrès dans le domaine de l'intelligence artificielle. Ainsi, vers la fin des années soixante-dix, trois pôles géographiques principaux se distinguent (France, Japon, Etats-Unis). La synthèse de tous les travaux réalisés jusqu'alors donne enfin naissance aux robots mobiles autonomes (du robot domestique au robot militaire).

L'industrie de production, les sociétés d'exploitation minière, les expéditions de recherche sous-marine... Les domaines d'utilisation de robots autonomes sont très variés, allant de la production en chaîne dans une usine de voitures à l'exploration d'autres planètes, comme c'est le cas avec Mars Pathfinder, robot mobile autonome d'exploration de la planète Mars. [5]

C'est dans cet environnement de plus en plus automatisé que se fait sentir le besoin d'outils capables, non seulement d'effectuer des tâches répétitives ou encore impossibles à l'homme (porter des charges lourdes, découpage ultra précis, …), mais aussi de manifester une certaine autonomie de déplacement dans des milieux hostiles à l'homme. On en voit désormais les applications sur des chantiers tels que le désamiantage d'immeubles, la décontamination radioactive, les expériences en milieu dangereux… Aussi a-t-on besoin de robots mobiles autonomes capables de se déplacer d'un point à un autre sur une simple demande de l'utilisateur, qui n'a ainsi plus besoin d'être un expert en pilotage. Le meilleur exemple de planification complexe de la trajectoire est fourni par les drones, ces robots volants destinés aussi bien à l'espionnage militaire qu'aux études botaniques nécessitant des prises de vues aériennes.

3. DEFINITION :

3.1 Origine des termes [3]

Robot a été utilisé pour la première fois en 1921 par Karel Capek dans sa pièce R.U.R. (Rossums Universal Robots).

Le mot robot vient du tchèque "robota" qui signifie corvée, travail obligatoire

Le terme robotique a été utilisé pour la première fois par Asimov en 1941

3.2 Définition d'un robot [4]

Le Petit Larousse définit un robot comme étant un appareil automatique capable de manipuler des objets, ou d'exécuter des opérations selon un programme fixe ou modifiable. En fait, l'image que chacun se fait d'un robot est généralement vague, souvent un robot est défini comme un manipulateur automatique à cycles programmables.

Pour « mériter » le nom de robot, un système doit posséder une certaine flexibilité, caractérisée par les propriétés suivantes :

- La versatilité : Un robot doit avoir la capacité de pouvoir exécuter une variété de tâches, ou la même tâche de différente manière ;

-L'auto-adaptativité : Un robot doit pouvoir s'adapter à un environnement changeant au cours de l'exécution de ses tâches.

L'Association Française de Normalisation (A.F.N.O.R.) définit un robot comme étant un système mécanique de type manipulateur commandé en position, reprogrammable, polyvalent (i.e., à usages multiples), à plusieurs degrés de liberté, capable de manipuler des matériaux, des pièces, des outils et des dispositifs spécialisés, au cours de mouvements variables et programmés pour l'exécution d'une variété de tâches. Il a souvent l'apparence d'un, ou plusieurs, bras se terminant par un poignet. Son unité de commande utilise, notamment, un dispositif de mémoire et éventuellement de perception et d'adaptation à l'environnement et aux circonstances. Ces machines polyvalentes sont généralement étudiées pour effectuer la même fonction de façon cyclique et peuvent être adaptées à d'autres fonctions sans modification permanente du matériel.

4. PRESENTATION GENERALE DES ROBOTS MOBILES :

Contrairement au robot industriel qui est généralement fixé, le robot mobile est doté de moyens qui lui permettent de se déplacer dans son espace de travail. Suivant son degré d'autonomie ou degré d'intelligence, il peut être dote de moyens de perception et de raisonnement. Certains sont capables,

sous contrôle humain réduit, de modéliser leur espace de travail et de planifier un chemin dans un environnement qu'ils ne connaissent pas forcément d'avance.

Actuellement, les robots mobiles les plus sophistiqués sont essentiellement orientés vers des applications dans des environnements variables ou incertains, souvent peuplés d'obstacles, nécessitant une adaptabilité à la tâche. La figure (1.1) illustre la structure d'un tel robot.

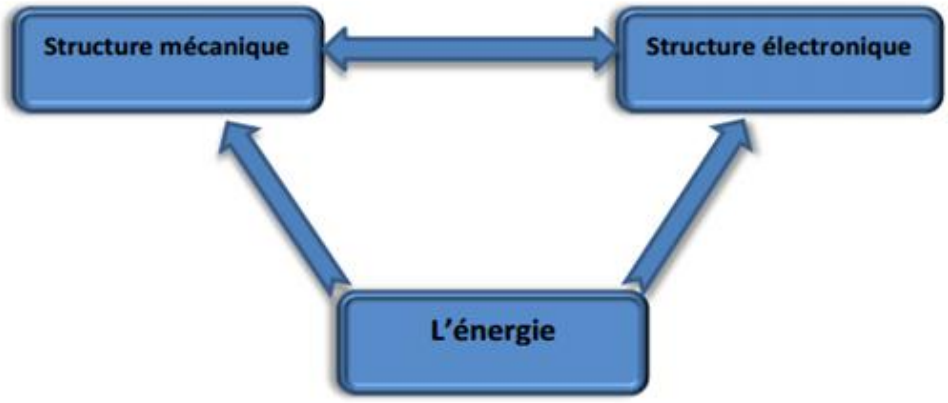


Figure 1-1 : Structure d'un robot mobile

Les robots mobiles ont une place particulière en robotique. Leur intérêt réside dans leur mobilité qui ouvre des applications dans de nombreux domaines. Comme les robots manipulateurs, ils sont destinés à assister l'homme dans les tâches pénibles (transport de charges lourdes), monotones ou en ambiance hostile (nucléaire, marine, spatiale, lutte contre l'incendie, surveillance…). [5]

L'aspect particulier de la mobilité impose une complexité technologique et méthodologique qui s'ajoute en général aux problèmes rencontrés par les robots manipulateurs. La résolution de ces problèmes passe par l'emploi de toutes les ressources disponibles tant au niveau technologique (capteurs, motricité, énergie) qu'à celui du traitement des informations par l'utilisation des techniques de l'intelligence artificielle ou de processeurs particuliers (vectoriel, cellulaires). L'autonomie du robot mobile est une faculté qui lui permet de s'adapter ou de prendre une décision dans le but de réaliser une tâche malgré un manque d'informations préliminaires ou éventuellement erronées. Dans d'autres cas d'utilisation, comme celui des véhicules d'exploration de planètes, l'autonomie est un point fondamental puisque la télécommande est alors impossible par le fait de la durée du temps de transmission des informations.

Figure 1.2 : "MobBob" - the intelligent Android Mobile Phone Robot

5. CLASSIFICATION : [3]

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

- Véhicule télécommandé par un opérateur qui lui impose chaque tâche élémentaire à réaliser.
- Véhicule télécommandé au sens de la tâche à réaliser. Le véhicule contrôle automatiquement ses actions.
- Véhicule semi-autonome réalisant sans l'aide de l'opérateur des tâches prédéfinies.
- Véhicule autonome qui réalise des tâches semi-définies. Ce type de véhicule pose des problèmes d'un niveau de complexité élevé de représentation des connaissances, de capacité décisionnelle et de génération de plans qui sont résolus à bord dans la mesure du possible.

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

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

6. APPLICATIONS : [6]

Le tableau ci-après résume de manière non exhaustive les diverses applications des robots mobiles.

Industrie nucléaire	- Surveillance de site - Manipulation de matériaux radio-actifs - Démantèlement de centrale
Sécurité civile	- Pose d'explosif - Déminage - Neutralisation d'activité terroriste - Surveillance de munition
Militaire	- Surveillance, patrouille - Pose d'explosifs - Manipulation de munition
Chimique	- Surveillance de site - Manipulation de matériaux toxique
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 modules - Recherche de navires immergés - Inspection des fonds marins
Agricole	- Cueillette de fruits - Traite, moisson, traitement des vignes…
Construction BTP	- Projection mortier - Lisage du béton
Nettoyage	- Coque de navire - Nettoyage industriel
Espace	- Exploration
Industriel	- Convoyage - Surveillance

Tableau 1-1 : Applications des robots mobiles

7. ARCHITECTURE DES ROBOTS MOBILES : [6]

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

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

- Le système de localisation

L'architecture des robots mobiles est représentée sur la figure 1-3

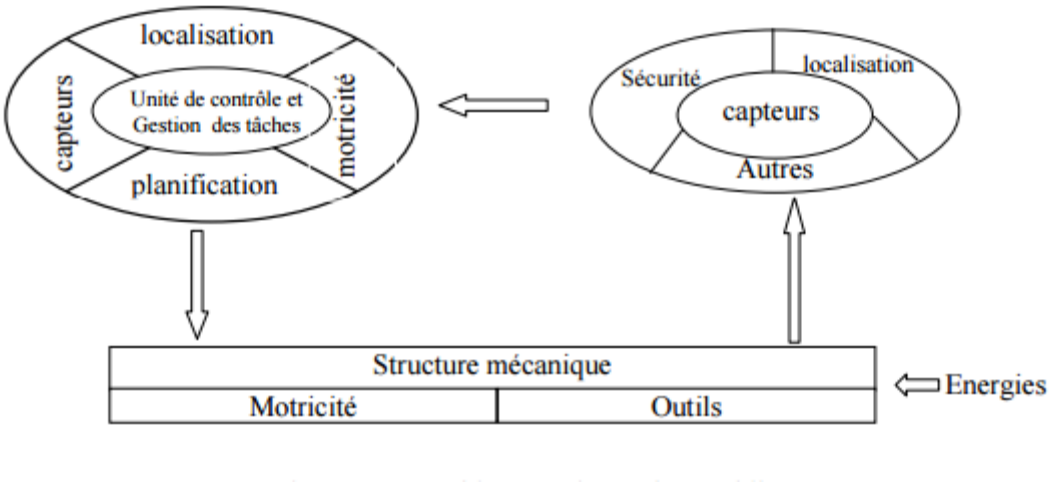


Figure 1-3 : Architecture d'un robot mobile

7.1 -La structure mécanique et la motricité :

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

7.1.1 -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. Le franchissement d'obstacles ou l'escalade de marches d'escalier est possible dans une certaine mesure. Toutes les configurations (nombre, agencement, fonction) des roues sont appliquées.

7.1.2 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…).

7.1.3 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. [4]

L'adaptation au support est un problème spécifique aux marcheurs. Il consiste à choisir le meilleur emplacement de contact en alliant l'avance et la stabilité avec l'aide de capteurs de proximité, de contact ou de vision.

7.1.4 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.

- Le type scolopendre constitue une structure inextensible articulée selon deux axes orthogonaux.
- Le type lombric comprend trois articulations, deux rotations orthogonales et une translation dans le sens du mouvement principal.
- Le type péristaltique consiste à réaliser un déplacement relatif d'un module par rapport aux voisins.

7.2 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.

L'énergie thermique est essentiellement employée par des véhicules de forte puissance comme énergie de base pour la traction ou pour activer un compresseur hydraulique. [5]

7.3-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 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 1-4.

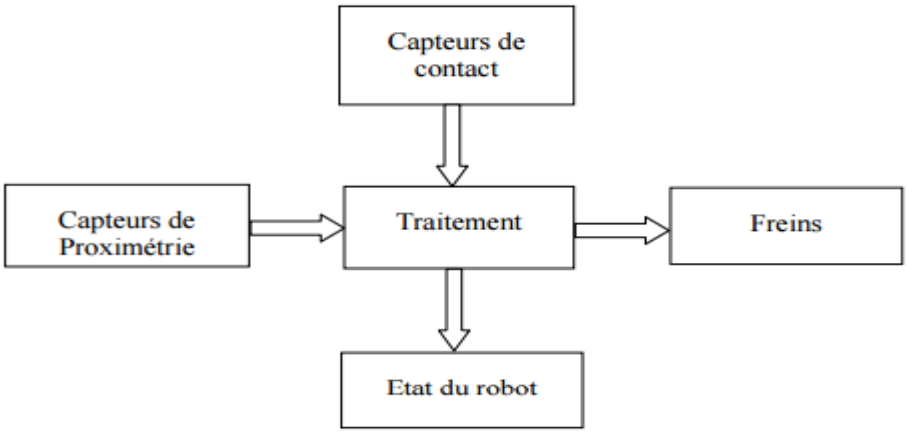


Figure 1-4:Synoptique de la sécurité.

Il comporte également un système de vérification permanent de l'état de fonctionnement des autres organes.

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.

7.4 Traitement des informations et gestion des taches : [5]

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 de navigation
- La modélisation de l'espace
- La planification.

7.5. La navigation :

La navigation est une étape très importante en robotique mobile. Bien entreprise, elle permet une large autonomie à un robot mobile. Le système de navigation comporte plusieurs modules qui peuvent être traités différemment et parmi lesquels on distingue celui de la localisation et celui de l'évitement d'obstacles. La détection et l'évitement des obstacles est l'étape fondamentale de l'évolution d'un robot en territoire inconnu. On dispose à cet effet de plusieurs types de capteurs : caméras; un programme d'analyse des images étant alors nécessaire, capteurs laser, capteurs infrarouge et capteurs à ultrasons.

On utilise en général un capteur à ultrasons qui permet de renseigner sur la présence d'un obstacle sur le chemin d'évolution. Une fois les obstacles repérés, le robot peut effectuer plusieurs actions, par exemple : cartographier le site sur lequel il évolue, vérifier si sa distance à l'obstacle est supérieure ou non à une distance limite, et dans le cas contraire, éviter l'obstacle. Un exemple d'évitement d'obstacles est présenté sur la figure 1-5.

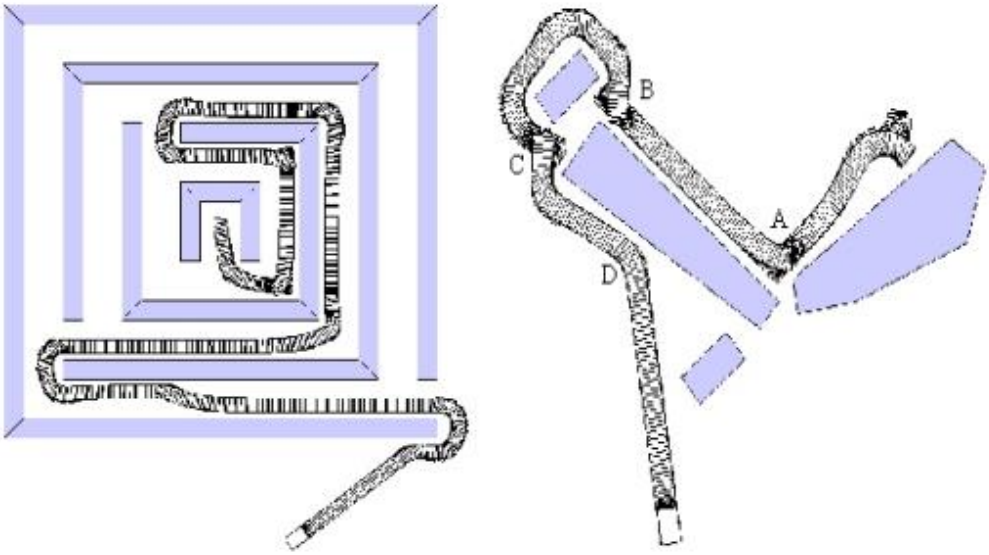


Figure 1-5 : Navigation de robot mobile en environnement encombré.

7.6 La modélisation de l'environnement :

La connaissance du milieu dans lequel évolue le robot mobile n'est établie en général qu'après avoir effectué une campagne de mesure de l'ensemble des éléments constituant l'environnement. Cette procédure fastidieuse peut être évitée si le robot construit lui-même son modèle d'environnement de manière dynamique. Par contre, la planification de trajectoire n'est pas utilisable tant que le robot ne dispose pas d'un modèle de l'espace d'évolution ce qui handicape très fortement l'utilisation du robot. A partir de cette base d'informations et d'une loi évaluant les erreurs de représentation, le planificateur peut générer des sous-trajectoires faisables dans certaines parties et modifier les soustrajectoires dans d'autres parties à l'aide des informations locales issues de la mesure des capteurs d'environnement. Lors de l'exécution d'une trajectoire, le robot acquiert des informations qui vont permettre de reconstituer le plus fidèlement possible le modèle de l'environnement de manière récursive à l'aide d'un algorithme approprié. [5]

7.7 La planification de trajectoire :

On voit ainsi au travers de cette première approche assez théorique apparaître un problème essentiel : la planification de la trajectoire. Différentes approches sont envisageables selon que le robot évolue en milieu connu ou inconnu :

- L'évolution en territoire cartographié simplifie évidemment la tâche des concepteurs : une fois la carte de la zone d'évolution rentrée dans la mémoire d'un ordinateur communiquant avec le robot ou bien dans une mémoire intégrée au robot lui-même, des algorithmes de routage permettent de diriger le robot.
- Il en va tout autrement dans le cas de l'évolution en territoire inconnu. Le robot doit alors analyser son environnement au moyen de différents capteurs, détecter sa position par rapport à son but, et décider de sa trajectoire. Cette localisation peut s'effectuer par différentes méthodes : triangulation de signaux émis par des balises déposées au cours du déplacement ou/et repérage d'obstacles à distance et construction d'une carte du site, mesures odométriques et estimation de la position.

On applique ensuite des algorithmes complexes pour diriger le robot. Ceux-ci peuvent amener des résultats plus ou moins heureux, le principal problème étant la nonconvergence de certaines boucles de déplacement. Si aucun algorithme de secours n'a été prévu, l'intervention humaine est alors nécessaire.[6]

8-Mahfoud III :

8.1-Description :

Le robot Mahfoud III est un robot mobile constituée d'une seule carte de commande, cette carte fait au même temps le contrôle et la commande du robot, , ce qui montre la puissance du dsPIC33F, qui nous a permis de bien exploiter le robot avec un minimum de composants, on a réussi a commander le robot, a lui associer des capteurs et des codeurs, desquels les mesures sont lus dans le programme du dsPIC.

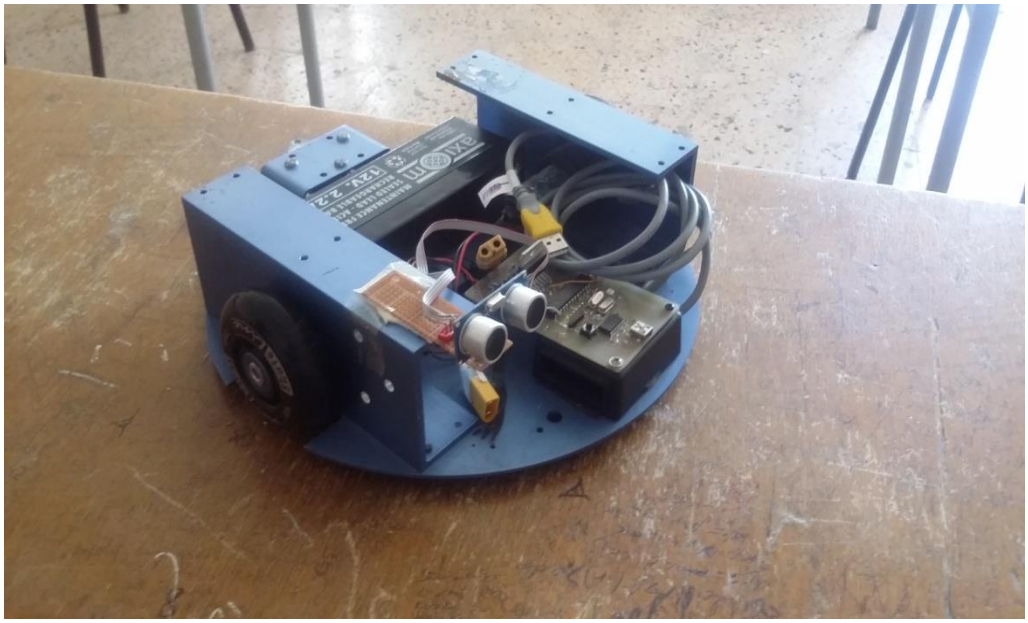


Figure 1-6 : Mahfoud III

Et comme on voit dans la figure ci-dessous la configuration de Mahfoud III il a une configuration tricycle à commande différentielle, composé de deux roues motrices et une roue libre.

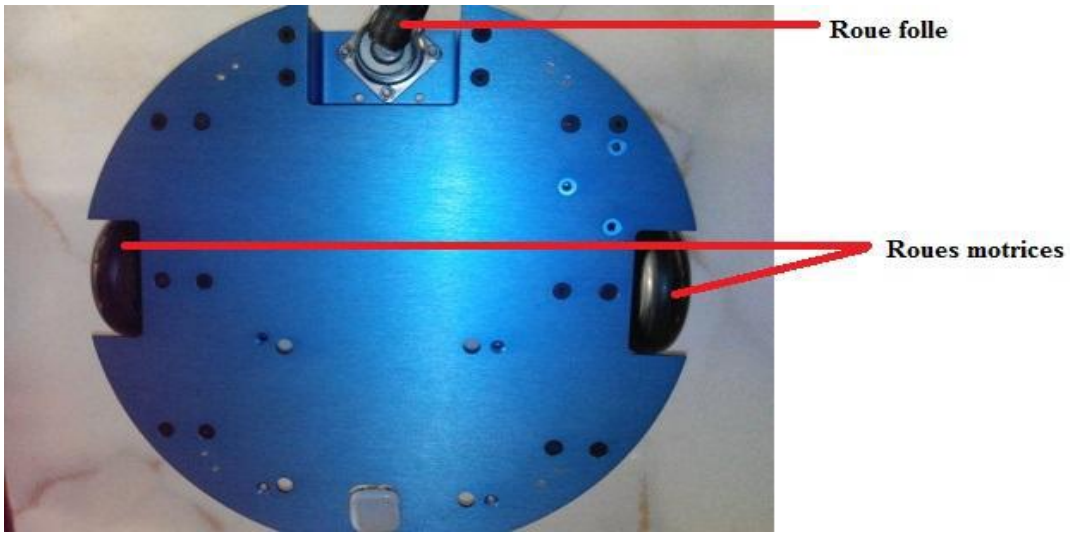


Figure 1.7 : configuration du robot Mahfoud III [2]

Pour chaque roue motrice on à utiliser un moteur à courant continu. Les moteurs sont des moteurs A-max de 12V.

8.2-Carte de Commande de Mahfoud III :

Pour la Catre de commande comporte le microcontrôleur dsPIC programmable en langage C et flash able à l'aide d'un programmateur PICkit2, et elle comporte aussi les modules de puissance, et le module de liaison série USB, voir Figure (1-8) :

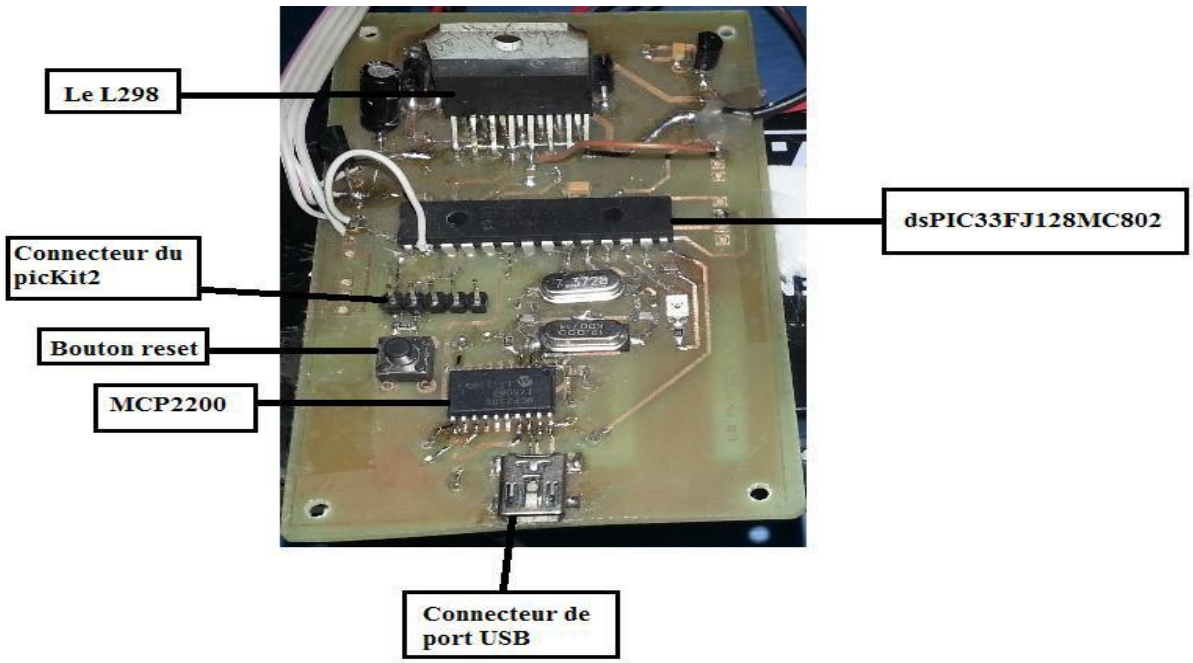


Figure (1-8) : carte de commande de Mahfoud III [2]

8.2.1- DSPIC 33FJ : [2]

Le microcontrôleur utilisé est un *dsPIC33FJ128MC802* fabriqué par Microchip, dispose de 28 pins dont 21 pins pour les E/S, tous ses 28 pins sont entièrement utilisés, on l'utilise a 29Mhz ce qui correspond à 29 MIPS pour un maximum autorisé de 40 MIPS

Le *dsPIC33FJ128MC802* est un microcontrôleur, 16 bits, possédant un jeu de 83 instructions, chacune stockée dans un seul mot (16 bits) de programme et est exécutée en 1 cycle.

8.2.2- Le L298 : [2]

Le L298 de chez STMicroelectronics©, est un module à 2 hacheurs 4 quadrants dans un circuit intégré ce circuit intégré permet de piloter des charges inductives tel que les relais, les moteurs pas à

pas et aussi des moteur à courant continu ce qui le cas pour nous, il permet de piloter directement sa vitesse dans les deux sens de rotation, et ce qu'il a d'avantage de plus c'est que pour contrôler les deux moteurs à courant continu un seul L298 suffit.

Et pour le fonctionnement de L298 il possède deux branches d'alimentation :

- Une pour l'alimentation du moteur Vs
- Et une autre pour l'alimentation de la logique interne Vss

8.2.3 - Convertisseur USB vers série MCP2200 :

Afin de pouvoir connecter le Smartphone a la carte pour réaliser certaines fonctions, il nous a fallu un module de connexion USB, celui qu'on a utilisé c'est le MCP2200 de chez Microchip.

Le MCP2200 est un convertisseur série USB vers UART qui permet une connectivité USB dans les dispositifs qui disposent d'une interface UART. Le MCP2200 comporte également 256 octets d'EEPROM intégrée.

8.2.4 Critères de choix des composants :

La carte nous a été imposée par le PFE de l'année précédente, Ces composants nous permettent de continuer le développement et de modifier légèrement les branchements pour associer le capteur ultrason. Il reste aussi de quoi connecter les interfaces des codeurs incrémentaux QEP.

9. CONCLUSION :

Les robots d'aujourd'hui ne sont plus que des prototypes, mais réalisent bien des tâches précises.

Nous constatons que les robots auront une place importante dans un futur proche et que les liens entre Homme-Machine se tisseront de plus en plus facilement, mais jusqu'à quelles limites ?

Les robots possèdent de nombreux avantages, ils améliorent réellement la vie de l'Homme. Tout devient réalisable avec un robot.

Il est bien de montrer que la robotique transformera nos vies, mais il faut bien prendre en compte que de telles avancées devront être contrôlées, car il existe toujours des abus, en particulier dans le domaine militaire. On nous le montre trop souvent dans les films de science-fiction où les robots prennent une place trop importante dans le monde humain, et deviennent incontrôlables. Ce n'est bien

sûr que l'imagination des réalisateurs mais il ne faut pas négliger la sécurité des êtres humains. Nous devons prendre garde à toujours rester maître de nos inventions.

Dans la dernière partie de ce chapitre on a vu tous ce que nos camarades de l'année dernière ont réussi à faire et ce qui nous reste à faire, c'est à dire d'utiliser le même programme et ajouter une partie de capteur ultrasonic pour que Mahfoud III soit capable d'évider les obstacles et à développer une application Android pour qu'il suive une ligne sur le sol donc qu'il soit un robot mobile autonome.

Chapitre 2 :

Développement d'une application Android

Android doit son nom à la startup éponyme spécialisée dans le développement d'applications mobiles rachetée par Google en août 2005, nom venant lui-même d'« androïde » qui désigne un robot construit à l'image d'un être humain. Le logiciel, qui avait été surnommé *gPhone* par les rumeurs de marchés et qui selon un de ses concepteurs Andy Rubin était initialement prévu pour être un système d'exploitation pour appareil photo, est proposé de façon gratuite et librement modifiable aux fabricants de téléphones mobiles, ce qui facilite son adoption. Le gPhone a été lancé en octobre 2008 aux États-Unis dans un partenariat de distribution exclusif entre Google et T-Mobile. Anticipant les annonces officielles, les marchés financiers se ruent massivement sur les actions Google les faisant monter jusqu'au plus haut historique de 724 dollars le5 novembre 2007 (le vendredi 18 octobre 2013, les actions Google franchissent les 1 000 dollars).[12]

Lors de l'été 2015, Android doit faire face à plusieurs crises. La première concerne la faille Stagefright. Elaborée par un chercheur en sécurité, elle peut perturber 95% des terminaux fonctionnant avec l'OS de Google par un simple MMS. Le pirate peut ainsi avoir accès à quasiment toutes les données sur le téléphone. La deuxième est un bug découvert par des chercheurs Trend Micro qui paralyse téléphones et tablettes.

1- Introduction :

L'Android est parmi les derniers systèmes d'exploitation qui développent les exigences des téléphones intelligents. La plateforme Android de smart phone devient de plus en plus importante pour les réalisateurs de logiciel, en raison de ses puissantes possibilités et open source.

Figure 2-1 : Logo d'Android

Lors des années précédentes, le traitement des données informatiques se fait par des ordinateurs ; en revanche le smart phone a des avantages qui ont les mêmes fonctions que l'outil informatique ; ce dernier porte l'intérêt de l'ordinateur grâce à Android.

La téléphonie mobile a connu une explosion dans les années 2000 mais aucune révolution n'a semblé arriver depuis que les appareils se ressemblent. Les innovations n'avaient plus vraiment de saveur ; les applications étaient difficiles d'accès de par leur mode de distribution et souvent peu performantes à cause des faibles capacités des appareils. [12]

Depuis quelques mois, les smart phones sont dotés d'une puissance plus importante et d'espaces de stockages conséquents. Les téléphones tendent à devenir des objets artistiques, presque de reconnaissance sociale, et possèdent des fonctionnalités qu'aucun téléphone ne pouvait espérer auparavant : connexion haut débit, localisation GPS, boussole, accéléromètre, écran tactile souvent multipoint, marché d'applications en ligne. Autant de qualités permettant de créer des applications innovantes et de les distribuer en toute simplicité.

La plate-forme Android apporte tout cela au consommateur, mais surtout, elle affranchit le développeur de nombreuses contraintes. Par son ouverture; elle permet à n'importe quel développeur de créer ses applications avec un ticket d'entrée quasi nul. Le framework et le système d'exploitation et outils associés ont un code source ouvert, leur accès est gratuit et illimité. Plus besoin de négocier avec le constructeur du téléphone pour qu'il vous laisse développer sur sa plate-forme. Tous les développeurs sont ainsi sur un même pied d'égalité, tous peuvent ajouter de la mobilité à des applications existantes.

Version	Codename	API	Distribution
2.2	Froyo	8	0.3%
2.3.3 - 2.3.7	Gingerbread	10	4.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	4.1%
4.1.x	Jelly Bean	16	13.0%
4.2.x		17	15.9%
4.3		18	4.7%
4.4	KitKat	19	39.3%
5.0	Lollipop	21	15.5%
5.1		22	2.6%

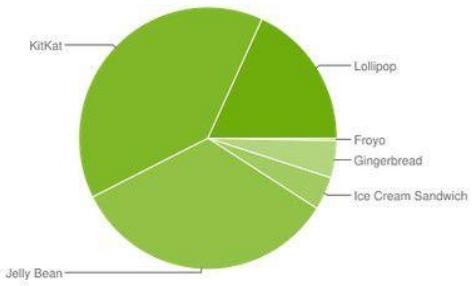


Figure 2.2 : le nombre des versions d'Android et le pourcentage de terminaux mobiles

Ce tableau et ce graphique illustrent le nombre des versions d'Android et le pourcentage de terminaux mobiles qui en sont équipés (aout 2015) © Google. [12]

2-Système Android :

La société Android a été rachetée en 2007 par Google. Mais aujourd'hui, l'Android est utilisé dans de nombreux appareils mobiles (smart phones). Les applications sont exécutées par un processeur de type ARM à travers un interpréteur JAVA. En plus de cela Android concurrence l'opérateur système d'Apple qu'il tend à dépasser en nombre d'utilisateurs. Android évolue pour mieux gérer l'hétérogénéité des appareils qu'il utilise.

Qu'est ce que ca veut dire Android ?

Android est un système d'exploitation développé initialement pour les Smart phones. Il utilise un noyau Linux qui est un système d'exploitation libre pour PC et intègre tous les utilitaires et les périphériques nécessaires à un smart phone. Il est optimisé pour les outils Gmail. Aussi, Android est libre et gratuit et a été ainsi rapidement adopté par des fabricants. [12]

2-1 La philosophie et les avantages d'Android :

2.1.1- Open source

Le contrat de licence pour Android respecte les principes de l'open source, c'est-à-dire que l'on peut à tout moment télécharger les sources et les modifier. Android utilise des bibliothèques open source puissantes, comme par exemple SQLite pour les bases de données et OpenGL pour la gestion d'images 2D et 3D.

2.1.2- Gratuit (ou presque)

Android est gratuit pour les constructeurs. En revanche, pour poster vos applications sur le Play Store, il vous en coûtera de payer 25$. Ces 25$ une seule fois, ce qui permet de publier autant d'applications à vie.

2.1.3- Facile à développer

Toutes les API mises à disposition facilitent et accélèrent grandement le travail. Ces APIs sont très complètes et très faciles d'accès. De manière un peu caricaturale, on peut dire que vous pouvez envoyer un SMS en seulement deux lignes de code (concrètement, il y a un peu d'enrobage autour de ce code, mais pas tellement).

Qu'est ce que ca veut dire une API ?

Une API, ou « interface de programmation » en français, est un ensemble de règles à suivre pour pouvoir dialoguer avec d'autres applications. Dans le cas de Google API, il permet en particulier de communiquer avec Google Maps.

2.1.4- Facile à vendre

Le Play Store (anciennement Android Market) est une plateforme immense et très visitée qui regroupe les applications publiées pour Android et vérifiées par Google pour la sécurité d'accès à votre smartphone.

2.1.5- Flexible

Le système est extrêmement portable, il s'adapte à beaucoup de structures différentes. Les Smartphones, les tablettes, la présence ou l'absence de clavier ou de trackball, différents processeurs… On trouve même des fours à micro-ondes qui fonctionnent à l'aide d'Android ! Non seulement c'est une immense chance d'avoir autant d'opportunités, mais en plus Android est construit de manière à faciliter le développement et la distribution en fonction des composants en présence dans le terminal (si votre application nécessite d'utiliser le Bluetooth, seuls les terminaux équipés de Bluetooth pourront la voir sur le Play Store). [12]

Cette flexibilité, qui contraste avec la maîtrise totale qu'Apple exerce sur iOS, fait à la fois la force et la faiblesse d'Android. Force de par son ouverture, sa compatibilité avec des langages de programmation répandus (Java Development Kit) et sa capacité à toucher un très large public en équipant aussi bien des téléphones haut de gamme que premier prix. Faiblesse car Android est fragmenté en de multiples versions, avec pas moins de 16 itérations publiées entre 2007 et 2015.

Ceci pose de nombreux problèmes de compatibilité des applications et de sécurité. En 2015, on dénombrait plus de 24.000 terminaux Android distincts proposés par 1.294 marques (source : OpenSignal). Cette année-là, une importante faille de sécurité baptisée Stagefright, qui touchait potentiellement 95% des mobiles sous Android, est venue illustrer avec fracas cette situation.

À partir de 2014, Google a commencé à décliner Android pour partir à la conquête de nouveaux marchés : Android TV pour les téléviseurs connectés, Android Auto pour les voitures et Android Wear pour les montres connectées.

2.1.6- Ingénieux

L'architecture d'Android est inspirée par les applications composites, et encourage par ailleurs leur développement. Ces applications se trouvent essentiellement sur internet et leur principe est que vous pouvez combiner plusieurs composants totalement différents pour obtenir un résultat surpuissant. Par exemple, si on combine l'appareil photo avec le GPS, on peut poster les coordonnées GPS des photos prises.

L'architecture d'Android est inspirée par les applications composites, et encourage par ailleurs leur développement. Ces applications se trouvent essentiellement sur internet et leur principe est que vous pouvez combiner plusieurs composants totalement différents pour obtenir un résultat surpuissant. Par exemple, si on combine l'appareil photo avec le GPS, on peut poster les coordonnées GPS des photos prises. [12]

2.2- Structuration :

Chaque application est attachée à un fichier qui la représente techniquement (il s'agit de l'application en elle-même). Lorsque le client d'un terminal achète et/ou souhaite utiliser une application, il doit télécharge un fichier sous une extension estampillée apk. Un fichier apk est en réalité un fichier compressé qui contient tous les éléments nécessaires/optionnels comme des fichiers de configuration ou le code source de l'application.

Un fichier apk contient principalement :

- Un manifeste qui va regrouper toutes les informations essentielles à l'application comme son nom, sa version, sa stratégie de sécurité, le nom du 'package' de l'application etc ... Le manifeste est un élément essentiel qui doit être présent dans toute application Androïd.
- Le code source : sans lui l'application ne pourrait pas fonctionner, il s'agit en fait de la base de toute application. Le code source représente le code métier de l'application (c'est à dire le contexte et les fonctionnalités de l'application) et couvre ses aspects fonctionnels. Souvent les différences de versions se situent uniquement à la modification du code source.
- Les ressources de l'application : une ressource représente tout élément non exécutable présent dans une application. Une ressource peut très bien être un ensemble de chaînes pour la réalisation d'une application multilingue (comme fichiers binaires, documents XML ou textuels) ou tout simplement des images. La présence de ressources n'est pas spécialement nécessaire mais plutôt recommandé.

Les fichiers apk ont la particularité unique de pouvoir être signés numériquement. C'est à dire que l'on peut appliquer une signature unique à l'archive. Ceci permet entre autre de garantir le fait que l'application fournie a bien été livrée par l'auteur présumé.

Prenons le cas par exemple où une entreprise créé une application de gestion de mails. La signature numérique de son application permet de garantir à ses clients que l'application a effectivement été compilée et réalisé par cette société.

Toutes les applications présentes sur le Play Store sont obligatoirement signées car Android n'installera pas d'application non signées pour des raisons de sécurité.

Conclusion :

Les applications distribuées sous Android sont représentées par une seule et unique archive apk qui contient tous les éléments essentielles nécessaires à son fonctionnement. Pour résumer le contenu de l'archive apk, des éléments présents tels que le nom de l'application, sa version, des ressources telles que des images ou fichiers et le code source de l'application encapsulé dans un fichier sous extension tex. Elles sont également signés numérique pour éviter toute usurpation d'application par un tiers non autorisé.

2.3-Les éléments d'une application :

- Activities (les Activités) : c'est la composante principale d'une application Android, elle représente une partie de l'application présentant une vue à l'utilisateur, importé par le paquet (android.app.Activity).
- Services (Des services) : à la différence d'une activité, le service ne possède pas de vue, c'est plutôt une activité tache de fond, importé par le paquet (android.app.Service), elle permet l'exécution d'un algorithme sur un temps indéfini qui ne s'arrête que lorsque la tache est finie ou son exécution est arrêtée, et il peut être lancé à différents moment :
    - Au démarrage du Smartphone
    - Au lancement de l'application
    - Par une action particulière sur le Smartphone
    - Lors un événement (par exemple, l'arrivé d'un appel, un sms, etc.…)

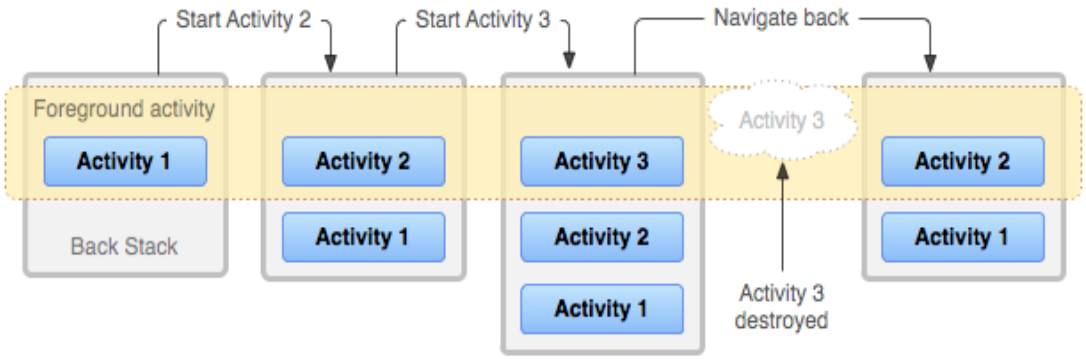
- Intents (les intentions) : ça permet d'envoyer un message pour un composant externe sans le nommer explicitement importé par le paquet (android.app.Intent)

- Broadcast receiver (les récepteurs des intentions) : il permet d'écouter ce qui se passe sur le système ou sur l'application et de déclencher une action prédéfinie, ça permet de déclarer la capacité de répondre à des intentions, il est importé par le paquet (android.app.BroadcastReceiver)
- Content providers (les fournisseurs de contenus) : permet de partager des informations au sein ou entre applications importé par le paquet (android.app.ContentProvider), ça nous permet d'accéder à :
  - contacts stockés dans le téléphone
  - L'agenda
  - Les photos de la galerie

2.4. Le cycle de vie d'une application Android :

Avant de rentrer dans le vif du sujet du développement d'une application, il est important de connaître le cycle de vie des éléments qui constituent cette application et notamment des activités. Comme nous avons pu le voir, une activité fournit un écran avec lequel les utilisateurs peuvent interagir. Une application Android est constituée d'un ensemble d'activités liées entre elles.

Chaque fois qu'une activité est lancée, elle est placée dans une pile appelée back stack. Chaque fois que vous lancez une nouvelle activité pour exécuter une nouvelle action, l'activité en premier plan est stoppée, la nouvelle est démarrée et placée en haut de la pile. Quand vous revenez en arrière, l'activité en premier plan est détruite de la pile et le précédent est réactivée.



En fait le cycle de vie des activités, est clairement défini pour éviter de surcharger le système. En tant que développeur vous devez faire attention à respecter les ressources du appareil de l'utilisateur et à bien les libérer lorsque votre application est stoppée.

Le diagramme ci dessous illustre les différents états que peut prendre une activité : les différents états sont symbolisés pas les rectangles colorés et arrondis, tandis que les rectangles grisés représentent les méthodes de callback que vous pouvez implémenter dans les phases de transition entre deux états.

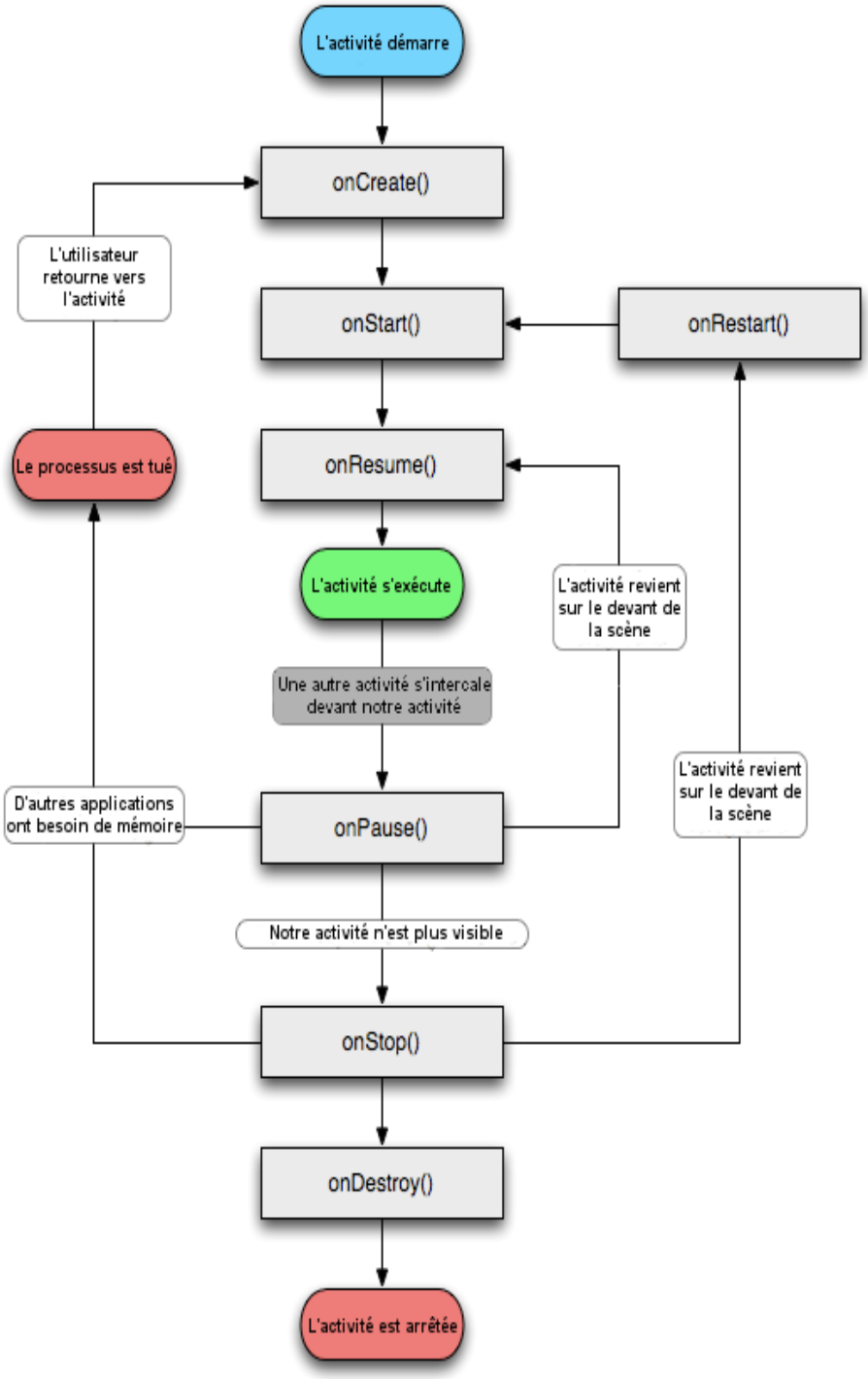
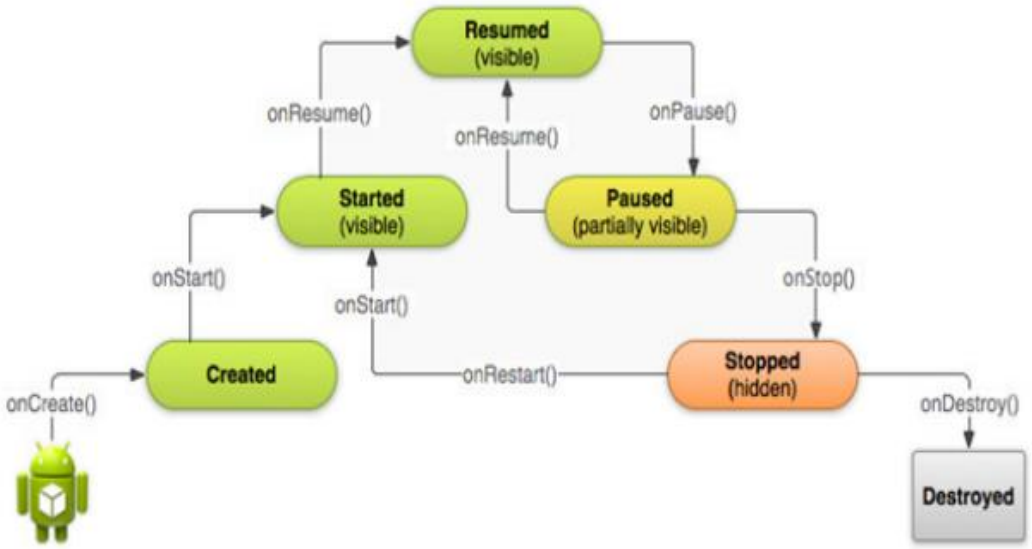


Figure 2.3 : cycle de vie d'une application Android

- onCreate (Bundle) : appelée à la création de l'activité, sert à initialiser l'activité mais aussi tout les données nécessaires pour cette dernière. On associe à cette méthode un Bundle en argument, ce dernier contient l'état de sauvegarde enregistré lors du dernier enregistrement de l'activité.
- onStart ( ) : cette méthode est appelée quand l'activité démarre (début du passage au premier plan). Si l'activité ne peut pas aller en premier plan quelque soit la raison, l'activité sera transférée à onStop ( ).
- onResume ( ) : appelée après onStart (), quand cette méthode est appelée l'application se trouve au premier plan et reçoit les interactions d'utilisateur
- onPause ( ) : cette méthode est appelée quand une autre activité passe au premier plan
- onStop ( ) : quand l'activité n'est plus visible cette méthode est appelée
- onFinish ( ) : permet de finir l'activité
- onDestroy ( ) : appelée quand l'application se termine, quand elle est totalement fermée, dans cette méthode, si les données ne sont pas sauvegarder ils seront perdus.



Au cours de la vie d'une activité, le système appelle un ensemble de méthodes de cycle de vie dans une séquence semblable à une pyramide à degrés qui signifie que chaque étape du cycle de vie de l'activité est une étape distincte sur la pyramide. Dès que le système crée une nouvelle instance d'activité, chaque méthode de rappel déplace l'activité Etat un pas vers le haut. Le sommet de la

pyramide correspond au point où l'activité est en cours d'exécution dans le premier plan et l'utilisateur peut interagir avec lui. [7].

Lorsque l'utilisateur commence à sortir de l'activité, le système appelle d'autres méthodes qui se déplacent l'état de l'activité vers le bas de la pyramide afin de démonter ou de détruire l'activité. Dans certains scénarios, l'activité se déplace en partie seulement en bas de la pyramide pour permettre à l'utilisateur de CV où il / elle a quitté. L'activité appelle une méthode spécifique comme onResume () ou OnStart () pour repousser l'activité vers le haut de la pyramide. Les événements entiers de méthode de rappel sont illustrés dans la figure 2-3.

3-OpenCV:

3.1-Introduction :

La vision humaine est extrêmement complexe. Chaque pixel contient des informations sur la lumière (quantité et contenu spectral/couleur) reçue par l'œil. Les objets (téléphone, voiture...) n'existent pas sur la rétine, et pourtant on les voit ; leur interprétation est le résultat du processus visuel. La vision par ordinateur ne cherche pas à comprendre ou à reproduire la vision humaine, mais à construire un modèle algorithmique qui, vu de l'extérieur, possède des propriétés semblables. L'ensemble du processus se présente de la manière suivante :

- Extraction de primitives à partir des images.
- Représentation des connaissances. (Modèle)
- Mise en correspondance image/connaissance. (Reconnaissance).

3.2-Librairie OpenCV :

OpenCV (Open Computer Vision) est une bibliothèque optimisée, de fonctions dédiées à la programmation pour la vision en temps réel.et elle est aussi une bibliothèque qui implémente de nombreux algorithmes couramment utilisés dans le domaine de la vision par ordinateur. Vision par ordinateur est le domaine de la science informatique qui met l'accent sur l'extraction des informations structurées à partir d'images. Images comme ils sont stockés sur les ordinateurs sont grandes, non structurées, deux réseaux bidimensionnels de pixels. Techniques de vision par ordinateur peuvent également être appliquées à des vidéos, qui sont stockés sous forme de séquences d'images. OpenCV fournit des algorithmes qui peuvent être utilisés pour des tâches telles que la localisation des visages dans une image, reconnaître des objets et des formes prédéfinies et la détection de mouvement dans une vidéo. OpenCV fournit également l'infrastructure nécessaire pour travailler avec des images et des

vidéos. OpenCV est régulièrement mis à jour, avec une documentation en plusieurs langues dont l'anglais et le chinois, le russe ... [7]

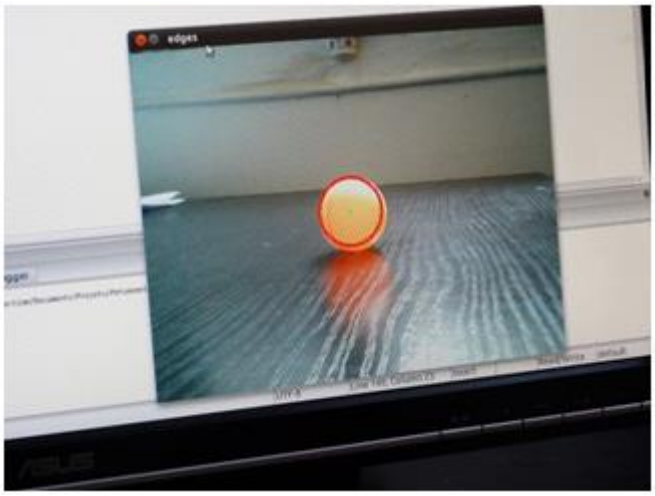


Figure 2.4 : Utilisation d'OpenCV pour suivre une balle [21]

3.3-Qui utilise OpenCV?

La plupart des scientifiques et des programmeurs informatiques pratiques sont eux qui utilisent OpenCV. Mais peu de gens sont au courant de toutes les façons dont la vision est utilisée. Par exemple, la plupart des gens sont un peu au courant de son utilisation dans la surveillance, et beaucoup le savent aussi qu'on l'utilise pour les images et la vidéo sur le Web. Quelques-uns ont vu une certaine utilisation de la vision par ordinateur dans les interfaces de jeu. Pourtant, peu de gens savent que la plupart des images aériennes et de Google Street View font un usage intensif de l'étalonnage de l'appareil et des techniques d'assemblage d'images. [11]

La licence open source pour OpenCV a été structuré de telle sorte que vous pouvez construire un produit commercial en utilisant tout ou partie de OpenCV.

Depuis sa première version alpha en Janvier 1999, OpenCV a été utilisé dans des nombreuses applications, produits et recherche. Ces applications sont utilisées dans plusieurs domaines tels que les satellites, détection d'objet, calibrage des cameras, des applications militaire, ct, OpenCV était un élément clé du système de vision dans le robot de Stanford, "Stanley", qui a remporté 2M $ DARPA en gagnant la course de grand prix de désert pour le robot. [10]

3.4-Applications d'OpenCV :

De nombreux algorithmes de vision sont exécutés par OpenCV, avec un large éventail d'applications. Au lieu d'essayer de lister tous les algorithmes et donner une explication technique détaillée, nous allons citer quelques problèmes de vision qu'OpenCV peut être utilisé pour résoudre.

OpenCV offre de nombreuses fonctions utiles pour reconnaître des objets dans une image. Ceux-ci vont de la simple tâche de trouver un objet d'une couleur sur un fond d'une autre au processus plus compliqué de trouver les visages des gens dans une image. OpenCV fournit également des algorithmes d'apprentissage machine, où le programme est montré un très grand ensemble d'images d'un type d'objet et utilise ces informations pour détecter les cas de cet objet dans d'autres images. Par exemple, un programme peut être formé pour reconnaître les lettres et les chiffres dans une image. Un tel programme pourrait alors être utilisé pour transformer les pages numérisées d'un livre en fichiers texte.

OpenCV fournit également des fonctions pour analyser le mouvement entre les images d'une vidéo. Ces fonctions sont principalement concernées par la détermination de quelles parties de l'image entre les trames déplacées, ainsi que la direction et la distance de ce mouvement. Une application de ces fonctions consiste à séparer les parties de l'image qui sont en mouvement à partir de celles qui ne sont pas. Cela à des applications dans la sécurité automatisée - un appareil photo pourrait être programmé pour envoyer une alerte seulement après qu'il détecte un mouvement passé un certain seuil. [9]

OpenCV peut également être utilisé pour extraire des informations en trois dimensions à partir de deux ou plusieurs vues d'un objet. Cette technique est appelée vision stéréo lorsqu'il est utilisé avec deux caméras. Ceci est utile pour les robots mobiles, qui doivent souvent naviguer dans un environnement inconnu. Plusieurs chercheurs de la NASA, par exemple, ont utilisé la vision stéréo pour naviguer sur la surface inconnue des autres planètes.

OpenCV fournit également l'infrastructure nécessaire pour travailler avec les images. Il fournit des objets de code pour gérer la capture d'images à partir de webcams, des images et des matrices. Il fournit également des fonctions qui mettent en œuvre les opérations courantes telles que l'inversion et le lissage des images. [9]

3.5- Langage de programmation :

OpenCV est écrit en C ++ et son interface principale est en C ++, mais il reste étendu à son ancienne interface C. Il a des liaisons en Python, Java et MATLAB / OCTAVE. L'API pour ces interfaces peut être trouvée dans la documentation en ligne

3.5.1- OpenCV-Python :

Le Python est un langage de programmation de haut niveau. Il a de nombreux modules disponibles qui traitent des tâches allant de l'analyse des images à des jeux vidéo de programmation. Cela permet aux programmeurs de se concentrent sur le problème qu'ils tentent de résoudre au lieu de les obliger à réinventer la roue.

Cependant, les langages de haut niveau comme Python consomment généralement plus de ressources système, mais il permet aussi aux programmeurs d'écrire un code lisible et plus concis. Les deux langages de haut et de bas niveau sont largement utilisés pour différentes applications.

Un autre avantage de Python est qu'il a un très grand nombre de modules open-source. Ces modules peuvent être librement téléchargés et utilisés, et fournissent des fonctions pour des applications allant de la lecture et l'écriture de fichiers de CAO (DXF-reader), à l'écriture de jeux vidéo (Pygame), pour interpréter et analyser les images (OpenCV). Les pages Web peuvent également être écrits en Python (Django). Python lui-même est également open-source, ce qui signifie qu'il peut être utilisé librement, sans paiement de droits de licence. Ces modules, combinées à la flexibilité de Python, permettent aux programmeurs l'utilisent pour un large éventail d'applications. Ils se cachent également de nombreux détails compliqués derrière des couches d'abstraction. Par exemple, on n'a pas besoin de savoir précisément comment les formes sont stockées dans un fichier CAO à la sortie dans ce format. [10]

3.5.2-Java :

Java est un langage de programmation orienté objet, développé par Sun Microsystems et destiné à fonctionner dans une machine virtuelle, il permet de créer des logiciels compatibles avec des nombreux systèmes d'exploitation.

Java et non seulement un langage de programmation puissant conçu pour être sûr, inter plateformes et international, mais aussi un environnement de développement qui est continuellement étendu pour fournir des nouvelles caractéristiques et des bibliothèques permettant de gérer de manière élégante des problèmes traditionnellement complexes dans les langages de programmation classiques, tels que le multithreading, les accès aux bases des données, la programmation réseau, l'informatique répartie.

En plus java est considéré comme un langage adaptable aux plusieurs domaines puisque une application web implémentée par celle-ci peut avoir des extensions ou des modifications dans le futur.

De plus, java permet de réduire le temps de développement d'une application grâce à la réutilisation du code développé.

Il a été choisi pour développer des applications pour Android avec le JDK (Java Development Kit).

3.5.3- C++ :

C++ est un langage de programmation compilé, permettant la programmation sous de multiples paradigmes comme la programmation procédurale, la programmation orientée objet et la programmation générique. Le langage C++ n'appartient à personne et par conséquent n'importe qui peut l'utiliser sans besoin d'une autorisation ou obligation de payer pour avoir le droit d'utilisation. C++ est l'un des langages de programmation les plus populaires, avec une grande variété de plates-formes matérielles et de systèmes d'exploitation.

En langage C, ++ est l'opérateur d'incrémentation, c'est-à-dire l'augmentation de la valeur d'une variable de 1. C'est pourquoi C++ porte ce nom : cela signifie que C++ est un niveau au-dessus du C.

3.6-Structure d'openCV :

OpenCV est structuré en cinq composantes principales, dont quatre sont représentés dans la figure 2.5 :

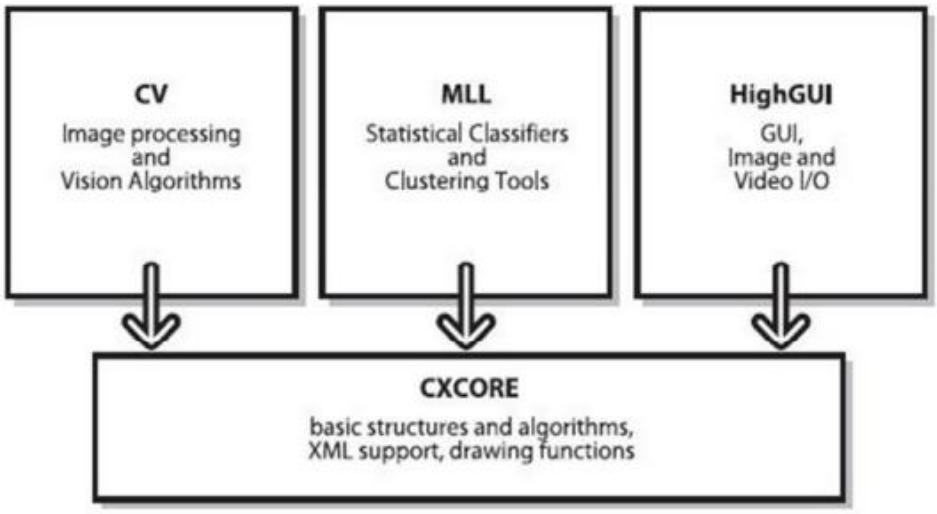


Figure 2.5 : La structure de base OpenCV

La composante CV contient les algorithmes de base de traitement d'image et de vision de plus haut niveau de l'ordinateur. Le MLL est une bibliothèque d'apprentissage de la machine, qui comprend de

nombreux classificateurs statistiques et les outils de regroupement. HighGUI contient des routines et des fonctions I/O pour le stockage et le chargement de la vidéo et des images. CXCore contient les structures de données de base et le contenu. Un autre composant est CvAux, qui contient les deux zones abandonnées (HMM intégré de reconnaissance des visages) et algorithmes expérimentaux (fond / segmentation de premier plan) [8].

3.7- Avis sur Android et OpenCV :

Si OpenCV est utilisable en Java en programmation Android, c'est grâce à son portage via un wrapper (encapsuleur) Java. Du coup, il n'existe pas de réelles classes implémentées en Java derrière ce qu'on utilise dans notre programme, et la documentation de la librairie est celle du code C++. [9]

4-Conclusion :

Dans ce chapitre nous avons présenté vue un bref aperçu des éléments essentiels pour développer une application mobile et on peut conclure que le traitement vision par Smartphone/ image est un domaine relativement nouveau, il va jouer un rôle très important dans le marché futur de la robotique mobile.

Vision par ordinateur/Smartphone est un domaine en croissance rapide, à cause des caméras et Smartphone qui sont moins chèrs et plus capables, et ils ont des puissances de traitement très importantes, et en partie parce que les algorithmes de vision commencent à être plus simple à comprendre. OpenCV joué un rôle très important dans ce domaine, il permet à des milliers de personnes à faire un travail plus productif dans la vision. En mettant l'accent sur la vision en temps réel, OpenCV aide les étudiants et les professionnels à mettre en œuvre efficacement les projets et relancer la recherche en leur fournissant une infrastructure de vision.

Chapitre 3 :

L'application mobile de Mahfoud III

1-Introduction :

Ce troisième chapitre est consacré à la conception de l'application mobile qui permettra à Mahfoud III d'être plus intelligent (smarter) et communicant. Cette connexion entre le robot et le smartphone doit assurer l'échange d'informations dans les deux sens, elle va aussi permettre au robot d'interagir avec le monde réel par le biais de l'application mobile en envoyant ses données internes à un site web tout en suivant une ligne blanche.

Pour cela, nous sommes partis de l'application développée de l'année dernière comme elle est montrée dans la Figure (3.1), puis nous y avons apporté les changements, notamment la librairie OpenCV pour la détection de la ligne blanche et la communication des informations.

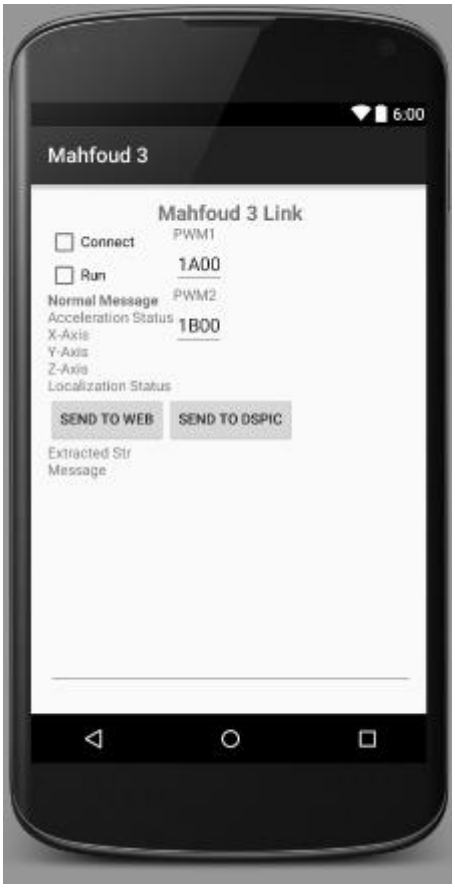


Figure3.1-L'application mobile initiale de Mahfoud III

Donc l'objectif de ce chapitre sera comme suit :

- Ajouter la partie de la camera à ce programme
- Recherche la zone d'intérêt
- Envoi et réception des informations.

2-Construction de l'application Mahfoud III :

2.1-La Caméra :

La classe de l'appareil photo est utilisé pour définir les paramètres de capture d'image, démarrer / arrêter aperçu, prendre des photos, et de récupérer des images pour l'encodage pour la vidéo. Cette classe est un client pour le service de l'appareil, qui gère le matériel de la caméra réelle.

Pour accéder à la caméra de l'appareil, vous devez déclarer la permission de CAMERA dans votre manifeste Android. Veillez également à inclure les <utilisations-fonction> élément manifeste de déclarer les caractéristiques de l'appareil utilisé par votre application. Par exemple, si vous utilisez la fonction appareil photo et auto-focus, votre Manifest devrait inclure les éléments suivants:

```
<uses-permission android:name="android.permission.CAMERA"/>
<uses-feature android:name="android.hardware.camera"
android:required="false"/>
<uses-feature android:name="android.hardware.camera.autofocus"
android:required="false"/>
<uses-feature android:name="android.hardware.camera.front"
android:required="false"/>
<uses-feature android:name="android.hardware.camera.front.autofocus"
android:required="false"/>
```

Et pour l'orientation :

```
android:screenOrientation="portrait"
```

Tout d'abord, nous allons avoir besoin d'une surface pour pré visualiser l'image que nous renvoie la camera. Il faut ajouter une SurfaceView à notre layout, comme ceci :

```
<org.opencv.android.JavaCameraView
    android:layout_width="match_parent"
    android:layout_height="230dp"
    android:id="@+id/color_blob_detection_activity_surface_view"
    android:layout_alignParentLeft="true"
    android:layout_alignParentBottom="true"
    android:layout_marginBottom="1dp"
    android:layout_marginTop="3dp"/>
```

A cette étape, notre application ressemble à :

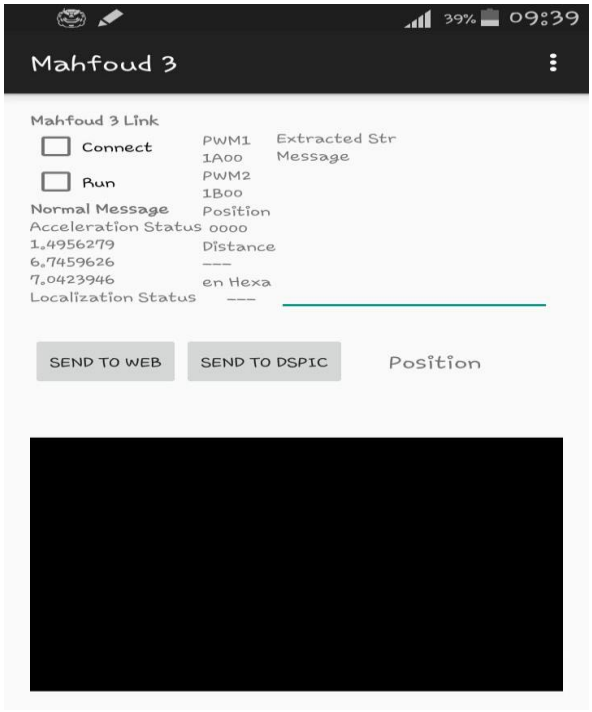


Figure 3.2 : Mettre une surfaceView à Layout

Pour cela il faut aller sur le fichier main.java et ajouter le code concernant de la caméra.

Toute d'abord nous allons implémenter une interface nommée CameraBridgeViewBase pour permettre à notre activité d'avoir les retours sur notre application :

```
public class MainActivity extends ActionBarActivity implements
CameraBridgeViewBase
```

On crée l'objet de Camera par :

```
private CameraBridgeViewBase mOpenCvCameraView;
```

```
mOpenCvCameraView.enableView();
mOpenCvCameraView.setOnClickListener(MainActivity.this);
```

Pour voir l'image sur l'écran il faut ajouter certain méthodes :

```
public void onDestroy() {
    super.onDestroy();
    if (mOpenCvCameraView != null)
        mOpenCvCameraView.disableView();
}
```

Cette méthode appelée quand l'application se termine, quand elle est totalement fermée.

```
@Override
protected void onPause() {
    // Ideally a game should implement onResume() and onPause()
    // to take appropriate action when the activity looses focus
    super.onPause();
    sensorManager.unregisterListener(this);
    unregisterReceiver(mUsbReceiver);
    unbindService(usbConnection);
    if (mOpenCvCameraView != null)
        mOpenCvCameraView.disableView();
}
```

Cette méthode est appelée quand une autre activité passe au premier plan

```
@Override
public void onStart() {
    super.onStart();

// ATTENTION: This was auto-generated to implement the App Indexing API.
// See https://g.co/AppIndexing/AndroidStudio for more information.
    client2.connect();
    Action viewAction = Action.newAction(
    Action.TYPE_VIEW, // TODO: choose an action type.
    "Main Page", // TODO: Define a title for the content shown.
// make sure this auto-generated web page URL is correct.
// Otherwise, set the URL to null.
            Uri.parse("http://host/path"),
// TODO: Make sure this auto-generated app deep link URI is correct.
Uri.parse("android-app://com.embesystems.mahfoud/http/host/path")
    );
    AppIndex.AppIndexApi.start(client2, viewAction);
}
```

On l'appelle cette activité quand l'application est démarrée

```
@Override
public void onStop() {
    super.onStop();

    // ATTENTION: This was auto-generated to implement the App Indexing
API.
    // See https://g.co/AppIndexing/AndroidStudio for more information.
    Action viewAction = Action.newAction(
            Action.TYPE_VIEW, // TODO: choose an action type.
            "Main Page", // TODO: Define a title for the content shown.
            // TODO: If you have web page content that matches this app
activity's content,
            // make sure this auto-generated web page URL is correct.
            // Otherwise, set the URL to null.
            Uri.parse("http://host/path"),
            // TODO: Make sure this auto-generated app deep link URI is
correct.
            Uri.parse("android-
app://com.embesystems.mahfoud/http/host/path")
    );
    AppIndex.AppIndexApi.end(client2, viewAction);
    client2.disconnect();
}
```

Cette fonction est appelée quand l'activité n'est plus visible.

A cette étape on à :

Figure 3.3 : L'application mobile après mettre les activités

Et maintenant pour la rotation du camera de 90 degrés

```
mRgba = inputFrame.rgba();
Mat mRgbaT = mRgba.t();
Core.flip(mRgbaT, mRgbaT, 1);
Imgproc.resize(mRgbaT, mRgbaT, mRgba.size());
```

Dans cette partie on à créer une nouvelle matrice mRgbaT et après on à faites une rotation de 90 degrés par la fonction Core.*flip* et à la fin on à utilisés la fonction Imgproc.*resize* pour redimensionner notre surface de camera.

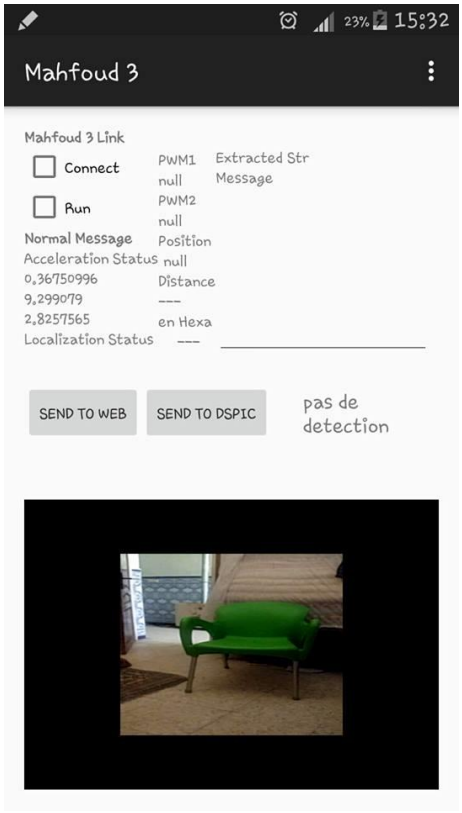


Figure 3.4 : rotation de la camera de 90 degrés

2.2-traitement d'image :

```
Imgproc.cvtColor(roi, mono, Imgproc.COLOR_BGR2GRAY);
```

La fonction cvtColor converti une image d'entrée d'un espace colorimétrique à un autre. Dans le cas d'une transformation à partir de l'espace de couleur RGB, l'ordre des canaux doit être spécifié explicitement (RGB ou BGR). Notez que le format de couleur par défaut dans OpenCV est souvent désigné comme RGB, mais il est en fait BGR (les octets sont inversés). Ainsi, le premier octet dans une image en couleur standard (24 bits) sera un 8-bit Bleu composant, le second octet sera vert, et le troisième octet sera rouge. Les quatrième, cinquième et sixième octets seraient alors le deuxième pixel (bleu, puis vert, puis rouge), et ainsi de suite.

Donc dans notre cas nous avons converti une image d'entrée roi en une autre image de sortie mono quelle est sous la forme noir et blanc seulement.

```
Imgproc.GaussianBlur(mono, blur, new Size(9, 9), 2, 2);
```

La fonction GaussianBlur convoluer l'image source avec le noyau gaussien spécifié. Afin d'appliqué le filtrage.

```
Imgproc.threshold(blur, thresh, 128, 255, Imgproc.THRESH_BINARY |
Imgproc.THRESH_OTSU);    // cherche  les zones sombres
```

La fonction threshold applique à un seuil de niveau fixe à un tableau à un seul canal. La fonction est généralement utilisée pour obtenir un bi-niveau d'image (binaire) sur une image en niveaux de gris (comparer () pourrait également être utilisé à cet effet) ou pour la suppression d'un bruit, qui est, le filtrage des pixels avec une trop petite ou trop grande valeur.

Qu'est-ce que Thresholding?

- La méthode de segmentation simple,
- Exemple d'application : séparer les régions d'une image correspondant à des objets que nous voulons analyser. Cette séparation est basée sur la variation d'intensité entre des pixels d'objet et des pixels de fond.
- Pour différencier les pixels, nous sommes intéressés par le reste (qui sera finalement rejeté), nous effectuons une comparaison de chaque valeur d'intensité de pixel par rapport à un seuil (déterminé en fonction du problème à résoudre).
- Une fois que nous avons séparé correctement les pixels importants, nous pouvons les définir avec une valeur déterminée pour les identifier (par exemple, nous pouvons leur attribuer une valeur de 0 (noir), 255 (blanc) ou toute valeur qui convient à vos besoins).

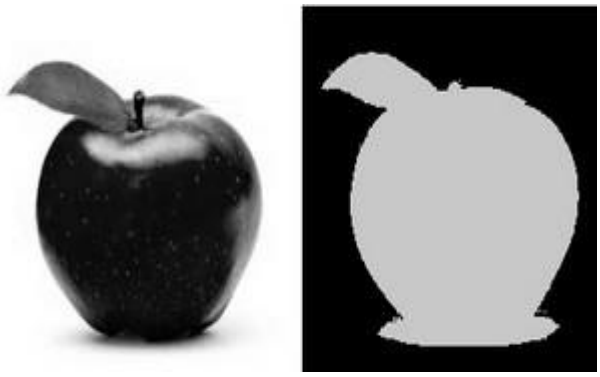


Figure 3.5 : Exemple sur le principe de la fonction Treshold

Dans cette figure on voit deux photos la première c'est la photo originale et la deuxième photos c'est après l'utilisation de la fonction treshold, On à utiliser la fonction treshold pour séparer les deux couleur (Le noir et le blanc).

```
Imgproc.erode(thresh, erodeImg, erode);
```

La fonction erode part de l'image source en utilisant l'élément structurant spécifié qui détermine la forme d'un quartier de pixel sur lequel le minimum est pris.

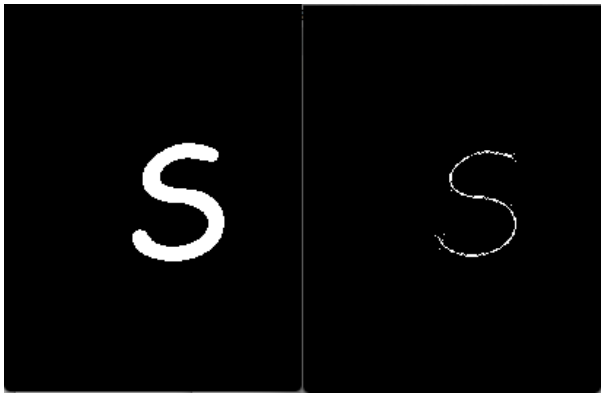


Figure 3.6 : Exemple sur le principe de la fonction erode

```
Imgproc.dilate(erodeImg, dilateImg, dilate);
```

La fonction dilate part de l'image source en utilisant l'élément structurant spécifié qui détermine la forme d'un quartier de pixel sur lequel le maximum est pris ; le contraire.

Figure 3.7 : Exemple sur le principe de la fonction dilate

Après avoir procéder à améliorer les contours et les isoler adéquatement du fond de l'image, nous sommes maintenant prêt à trouver les contours des objets dans l'image.

Trouver les contours des objets (findContours)

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.

L'utilisation de la fonction findContours pour trouver tous les contours dans une image.

```
Imgproc.findContours(dilateImg, contours, notused, Imgproc.RETR_LIST,
Imgproc.CHAIN_APPROX_SIMPLE);
```

On utilise ***RETR_LIST*** pour récupère tous les contours

L'utilisation de ***CHAIN_APPROX_SIMPLE*** c'est pour comprime les segments horizontaux, verticaux et diagonaux et ne laisse que leurs points d'extrémité. Par exemple, un contour rectangulaire en haut à droite est codé avec 4 points.

2.3-La zone d'intérêt :

```
Imgproc.rectangle(mRgbaT, new Point(R.x, R.y), new Point(R.x + R.width, R.y
+ R.height), new Scalar(25, 50, 50, 0), 2);
```

Pour dessiner un rectangle, vous avez besoin du coin supérieur gauche et du coin inférieur droit du rectangle. Cette fois, nous allons dessiner un rectangle vert dans le coin supérieur droit de l'image.

Et pour trouver et dessiner le plus grand rectangle :

```
for (MatOfPoint cont : contours) {
    int largest_area = 0;
    int largest_contour_index = 0;
    Rect r = boundingRect(cont);
    for (int i = 0; i < contours.size(); i++) // iterate through each contour.
    {
        double area = (contours.get(i));  //  Find the area of contour

        if (area > largest_area) {
            largest_area = (int) area;
            largest_contour_index = i;          //Store the index of largest contour
            r = boundingRect(contours.get(i)); // Find the bounding rectangle for
biggest contour
        }
    }
    Imgproc.rectangle(mRgbaT, new Point(R.x + r.x, R.y + r.y), new Point(R.x + r.x
+ r.width, R.y + r.y + r.height), new Scalar(00, 200, 00, 0), 2);
```

La fonction ***contourArea*** est utilisée pour calculer une zone de contour

Pour déterminer le centre de plus grand rectangle on utilise :

```
//find the mass center of largest contour area
Point center = new Point(r.x + r.width / 2, r.y + r.height / 2);
Imgproc.rectangle(roi, center, center, new Scalar(0, 200, 0, 0), 5);
```

Avant d'utiliser cette méthode pour trouver le centre de plus grand rectangle on à essayer plusieurs méthode et l'un des méthodes la plus utilisé dans OpenCV c'est la fonction **moment ()**, cette fonction elle trouve directement le centre de rectangle, mais pour notre cas elle n'a pas marché à cause de problème de openCV 3.0.0 qu'on à travailler avec, mais cette fonction marche très bien pour les autres versions d'openCV.

Et pour dessiner les coins de ce rectangle :

```
//déssinet les cordonneés du rectangle en bleu
Point ab = new Point(r.x + r.width, r.y + r.height);
Imgproc.rectangle(roi, ab, ab, new Scalar(0, 0, 255), 10);
Point ad = new Point(r.x, r.y);
Imgproc.rectangle(roi, ad, ad, new Scalar(0, 0, 255), 10);
Point ac = new Point(r.x + r.width, r.y);
Imgproc.rectangle(roi, ac, ac, new Scalar(0, 0, 255), 10);
Point aa = new Point(r.x, r.y + r.height);
Imgproc.rectangle(roi, aa, aa, new Scalar(0, 0, 255), 10);
```

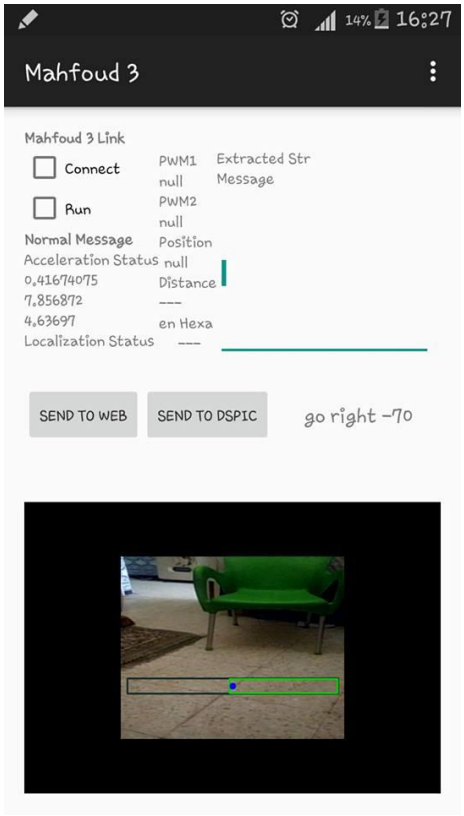


Figure 3.8 : Trouver la zone d'intérêt

2.4-Conversion d'un nombre décimal vers hexadécimale :

Avant de faire la conversion nous avons déclaré les chiffres hexadécimaux dans un tableau :

```
private final static char[] DIGITS = { '0', '1', '2', '3', '4', '5', '6',
'7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f' };
```

```
private static final char[] UPPER_CASE_DIGITS = {
        '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
        'A', 'B', 'C', 'D', 'E', 'F'
};
```

Cette déclaration nous donne la possibilité d'utiliser tous les nombres hexadécimaux (les 16 nombres) est l'affichage sera en majuscules.

Et la méthode de conversion sera comme suit :

```
public static String intToHexString(int distance,boolean upperCase, int
minWidth) {
    int bufLen = 8;  // Max number of hex digits in an int
    char[] buf = new char[bufLen];
    int cursor = bufLen;
    char[] digits = upperCase ? UPPER_CASE_DIGITS : DIGITS;
    do {
        buf[--cursor] = digits[distance & 0xf];
    } while ((distance >>>= 4) != 0 || (bufLen - cursor < minWidth));
    return new String(buf, cursor, bufLen - cursor);
}
```

```
if (distance >=0)
        {
    hexa= intToHexString(distance, true, 4);

        }
else
        {
    hexa= intToHexString(distance, true, 4).substring(4,8);

        }
```

```
Hexa.setText(hexa);
```

Cette partie concerne l'affichage du message dans le smartphone en hexadécimal

- Si le nombre décimal est positif, alors nous allons afficher tous les 4 caractères
- Et si le nombre est négatif donc il a 8 caractères et nous allons afficher seulement les 4 derniers, c'est pour cela nous avons utilisé **substring (4,8)**

Et pour afficher ce nombre on utilise la fonction **setText**

Figure 3.9 : L'application mobile de Mahfoud III

2.5- Renvoi des informations :

En règle générale, la communication avec des ports série comprend les étapes suivantes (sans ordre particulier):

- Recherche de ports série
- Connexion au port série
- Démarrage des flux de sortie d'entrée
- Ajout d'un event Listener pour écouter les données entrantes
- Déconnexion du port série
- Envoi de données
- Réception de données

2.5.1- Au niveau du l'Android, à la réception :

L'affichage de données reçus du robot se fait en utilisant un h*andle*

```
@Override
public void handleMessage(Message msg) {
    //Toast.makeText(mActivity.get(), "received",
Toast.LENGTH_SHORT).show();
```

```
    //Log.d(TAG, "msg RX");
    switch (msg.what) {
        case UsbService.MESSAGE_FROM_SERIAL_PORT:
            String data = (String) msg.obj;

            RXbuffer += data;
            int debut = RXbuffer.indexOf("$");
            int fin = RXbuffer.indexOf("\r", debut);
            if (fin != -1) {
                String ExtractedStr = RXbuffer.substring(debut, fin);
                RXbuffer = RXbuffer.substring(fin, RXbuffer.length());

                TextView Pwm1 = (TextView) findViewById(R.id.edPWM1);
                String pwm1 = ExtractedStr.substring(6, 10);
                Pwm1.setText(pwm1);

                TextView Pwm2 = (TextView) findViewById(R.id.edPWM2);
                String pwm2 = ExtractedStr.substring(15, 19);
                Pwm2.setText(pwm2);

                TextView POsRx = (TextView) findViewById(R.id.PosRx);
                String PosRx = ExtractedStr.substring(33, 37);
                POsRx.setText(PosRx);

                mActivity.get().ExtStr.setText(ExtractedStr);
                Log.d(TAG, "msg:" + ExtractedStr);
                Log.d(TAG, "buf:" + RXbuffer);
            }
            break;
    }
}
```

Et pour envoyer la position de camera vers le dspic :

```
// envoie la position ref
            String data = "CX";
            String hexVal = hexa;
            data += hexVal;
            if (usbService != null) // if UsbService was correctly binded,
Send data
            {
//                          byte[] Strbyte = new byte[data.length()+2];
//                          System.arraycopy(data.getBytes(), 0, Strbyte,
0, data.length());
//                          Strbyte[data.length()]=13;  // \n
//                          usbService.write(Strbyte);

                data += "\n";  // \n
                usbService.write(data.getBytes());
            }
```

2.5.2- Au niveau du dspic, à la réception :

```
void ReceiveData()
{
        Flags.CheckRX=0;
        if (InData[0] != 'C')            return;
        switch (InData[1])
                {
                case 'E' :      if (InData[2] == '1')
        Flags.Running=1;


        EnableL298=Flags.Running;        // enable or not L6234 principal
        PowerLED = Flags.Running;        // set or clear running LED

        Flags.RefreshAll=1;
                                                        break;
                case 'W' :      if (InData[2] == '1')           PWM1
        =ConvHexaToInt( &InData[3]);
                                                                if (InData[2]
== '2')         PWM2    =ConvHexaToInt( &InData[3]);
                                                        break;
                case 'M' :      if (InData[2] == '1')
        Flags.UARTIRef=1;


        Flags.RefreshAll=1;
                                                        break;
                case 'R' :      IRef_UART       =ConvHexaToInt( &InData[2]);
                                                        break;
                case 'X' :      Deviation       =ConvHexaToInt( &InData[2]);
                                                        break;

                }
}
```

3-Le site web et la base de données : [1]

Le smartphone qui est connecté au robot par un câble USB envoi des données de positionnement au site web, la connexion entre les trois est donnée par la figure suivante :

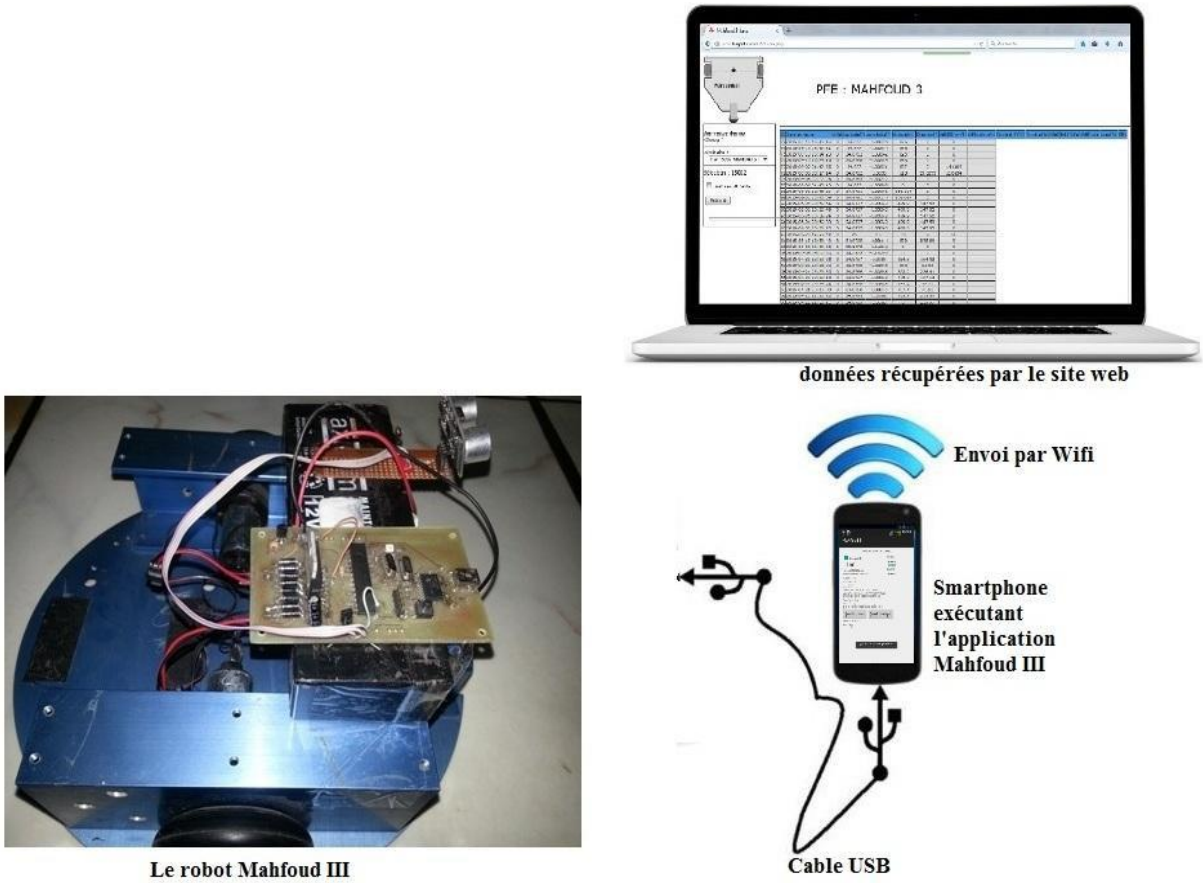
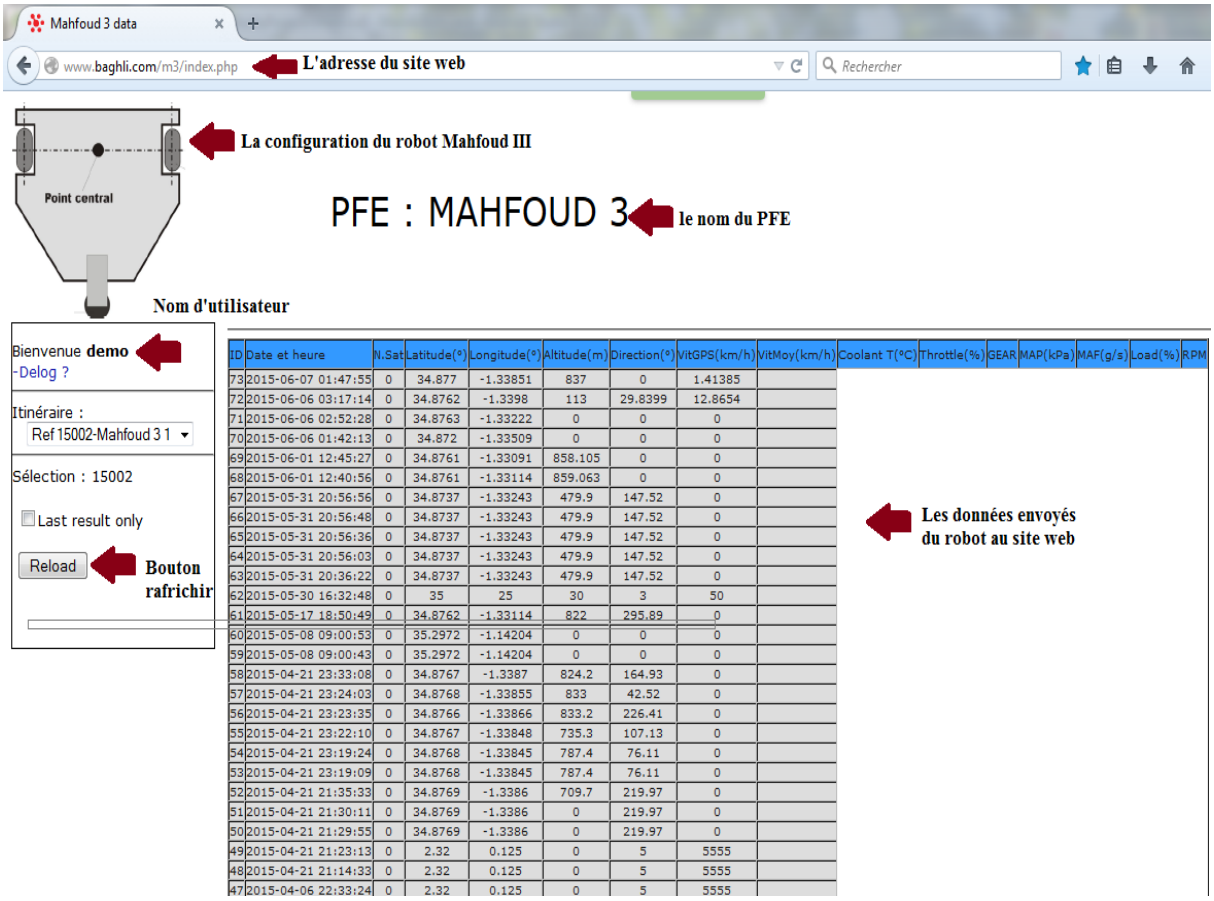


Figure 3.10 : liaison entre le robot et le smartphone et envoi des données

Les données récupérées par le smartphone du robot sont ensuite envoyés à une base de données (figure 3.11), dans un site web, cette dernière un login par identification, un bouton rafraichir et affiche un tableau dans lequel les données envoyées par le smartphone apparaissent.



ID	Date et heure	N.Sat	Latitude(°)	Longitude(°)	Altitude(m)	Direction(°)	VitGPS(km/h)	VitMoy(km/h)	Coolant T(°C)	Throttle(%)	GEAR	MAP(kPa)	MAF(g/s)	Load(%)	RPM
73	2015-06-07 01:47:55	0	34.877	-1.33851	837	0	1.41385								
72	2015-06-06 03:17:14	0	34.8762	-1.3398	113	29.8399	12.8654								
71	2015-06-06 02:52:28	0	34.8763	-1.33222	0	0	0								
70	2015-06-06 01:42:13	0	34.872	-1.33509	0	0	0								
69	2015-06-01 12:45:27	0	34.8761	-1.33091	858.105	0	0								
68	2015-06-01 12:40:56	0	34.8761	-1.33114	859.063	0	0								
67	2015-05-31 20:56:56	0	34.8737	-1.33243	479.9	147.52	0								
66	2015-05-31 20:56:48	0	34.8737	-1.33243	479.9	147.52	0								
65	2015-05-31 20:56:36	0	34.8737	-1.33243	479.9	147.52	0								
64	2015-05-31 20:56:03	0	34.8737	-1.33243	479.9	147.52	0								
63	2015-05-31 20:36:22	0	34.8737	-1.33243	479.9	147.52	0								
62	2015-05-30 16:32:48	0	35	25	30	3	50								
61	2015-05-17 18:50:49	0	34.8762	-1.33114	822	295.89	0								
60	2015-05-08 09:00:53	0	35.2972	-1.14204	0	0	0								
59	2015-05-08 09:00:43	0	35.2972	-1.14204	0	0	0								
58	2015-04-21 23:33:08	0	34.8767	-1.3387	824.2	164.93	0								
57	2015-04-21 23:24:03	0	34.8768	-1.33855	833	42.52	0								
56	2015-04-21 23:23:35	0	34.8766	-1.33866	833.2	226.41	0								
55	2015-04-21 23:22:10	0	34.8767	-1.33848	735.3	107.13	0								
54	2015-04-21 23:19:24	0	34.8768	-1.33845	787.4	76.11	0								
53	2015-04-21 23:19:09	0	34.8768	-1.33845	787.4	76.11	0								
52	2015-04-21 21:35:33	0	34.8769	-1.3386	709.7	219.97	0								
51	2015-04-21 21:30:11	0	34.8769	-1.3386	0	219.97	0								
50	2015-04-21 21:29:55	0	34.8769	-1.3386	0	219.97	0								
49	2015-04-21 21:23:13	0	2.32	0.125	0	5	5555								
48	2015-04-21 21:14:33	0	2.32	0.125	0	5	5555								
47	2015-04-06 22:33:24	0	2.32	0.125	0	5	5555								

Figure 3.11 : aperçu sur le site web

4-Conclusion :

Dans ce chapitre nous avons détaillé tous ce qui est de la partie de développement de l'application Android. Nous avons surtout parlé de traitement d'image et de la zone d'intérêt (ROI) pour que Mahfoud III soit capable de réceptionner puis transmettre sur Internet.

Chapitre 4 : Détecteur d'obstacles

Chapitre 4 : Détecteur d'obstacles

1. Introduction:

Les capteurs ultrasonores utilisent des ondes sonores de fréquence non perceptible par l'oreille humaine, généralement dans la fourchette 20 kHz - 200 kHz. De la même manière que les télémètres laser, ils sont basés sur le principe de la mesure du temps aller-retour lors de la réflexion sur un obstacle. C'est la méthode employée par certains animaux pour percevoir leur environnement, comme les chauves-souris ou les dauphins : l'écholocalisation. Un avantage de ces capteurs est que contrairement aux télémètres, l'onde qu'ils émettent n'étant pas focalisée, ils perçoivent beaucoup plus facilement des éléments filiformes comme des pieds de chaises ou des grillages. Par contre leur portée est faible, et ils sont moins adaptés aux milieux de propagation non isotropes comme l'air [17] .

2.Capteur ultrason HC-SR04:

Le capteur HC-SR04 utilise les ultrasons pour déterminer la distance d'un objet. Il offre une excellente plage de détection sans contact, avec des mesures de haute précision et stables. Son fonctionnement n'est pas influencé par la lumière du soleil ou des matériaux sombres, bien que des matériaux comme les vêtements puissent être difficiles à détecter. [13]

3. Principes de fonctionnement des ultrasons :

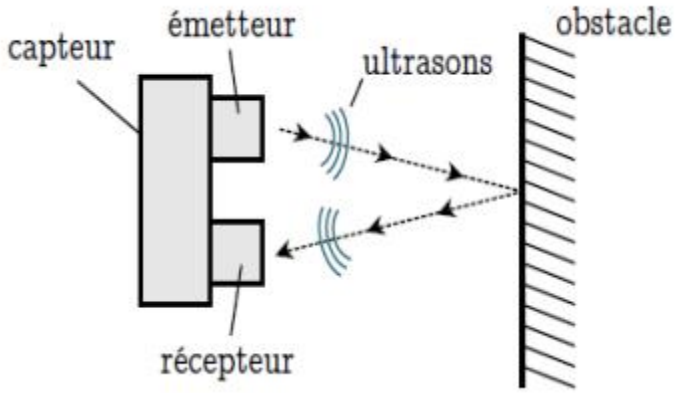


Figure4.1: principe de fonctionnement d'un capteur ultrason

Les capteurs ultrasons fonctionnent en mesurant le temps de retour d'une onde sonore inaudible par l'homme émise par le capteur. La vitesse du son étant à peu près stable, on en déduit la distance à l'obstacle.

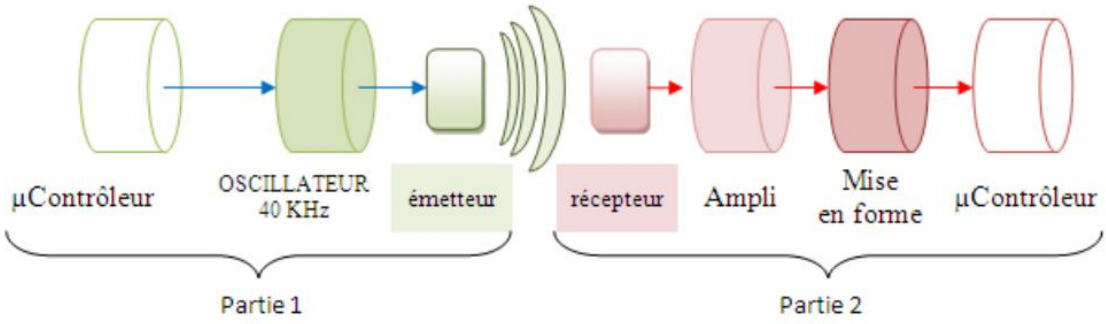
Les ultrasons sont des ondes infra-acoustique, qui oscillent à des fréquences supérieures au seuil acoustique.
Ceci est important à plusieurs niveaux :

- ✓ Le premier est que cette fréquence n'interfère pas avec le bruit audible produit par la majorité des corps physiques de notre environnement.

- ✓ La deuxième est que ces fréquences sont très précises, rapides et peuvent être personnalisées.

Les capteurs fournis ont souvent la forme d'une paire car il y a deux parties essentielles :

- L'émetteur
- Le récepteur



L'émetteur émet un son à une fréquence définie et le récepteur collecte le son répercuté par les obstacles. La distance aux objets est calculée par le temps mis par le son pour revenir au récepteur.

Première partie : l'émission:

La fréquence du signal de l'émetteur couramment utilisé est égale à 40 KHz, sa production nécessite l'un des TIMER du microcontrôleur.

Cette fréquence n'est efficace que si l'émetteur ultrason est alimenté par une tension au moins égale à 4.5 V.

Deuxième partie : la réception:

- Le capteur transforme les variations de pression acoustique en signal électrique.

-Ce signal électrique analogique de quelques millivolts est amplifié.

-Le signal amplifié est mis en forme pour pouvoir être traité par microcontrôleur.

Le signal fourni par le capteur a une amplitude de l'ordre de 10 mV, celle-ci diminue avec la distance.

Son exploitation au niveau du microcontrôleur nécessite une amplitude de quelques volts qui varie de 0 à 5 V donc nécessite son amplification. [14]

Nous avons choisi d'utiliser le module le plus disponible sur le marché : **HC-SR04.**

4. Caractéristiques de HC-SR04:

- Dimensions : 45 mm x 20 mm x 15 mm
- Plage de mesure : 2 cm à 400 cm
- Résolution de la mesure : 0.3 cm
- Angle de mesure efficace : 15 °
- Largeur d'impulsion sur l'entrée de déclenchement : 10 µs (Trigger Input Pulse width)[13].

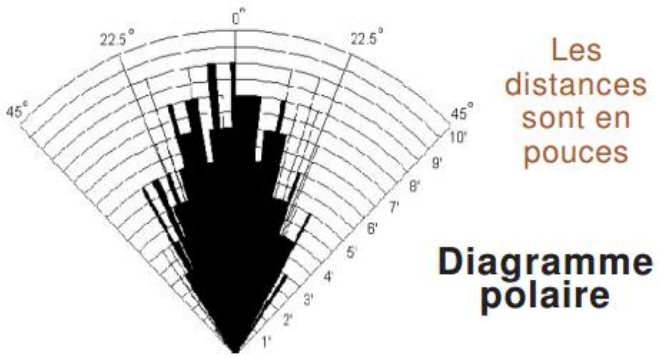


Figure 4.2: Test pratique de performance, meilleur angle de 30 degrés [13]

5. Broches de connexion:

- Vcc = Alimentation +5 V DC
- Trig = Entrée de déclenchement de la mesure (Trigger input)
- Echo = Sortie de mesure donnée en écho (Echo output)
- GND = Masse de l'alimentation [13]

6. Spécifications et limites :

Paramètre	Min	Type	Max	Unité
Tension d'alimentation	4.5	5.0	5.5	V
Courant de repos	1.5	2.0	2.5	mA
Courant de fonctionnement	10	15	20	mA
Fréquence des ultrasons	-	40	-	kHz

Tableau 4.1 : spécification et limites d'un capteur ultrasons

Figure 4.3:capteur ultrason HC-SR04

7. Interface le HC-SR04 avec le dspic 33FJ :

Pour détecter la distance aux obstacles, un module de capteur HC-SR04 est un bon compromis qualité/prix. Il donne la distance à tout obstacle devant lui. Ceci peut être utilisé dans plusieurs projets simples.

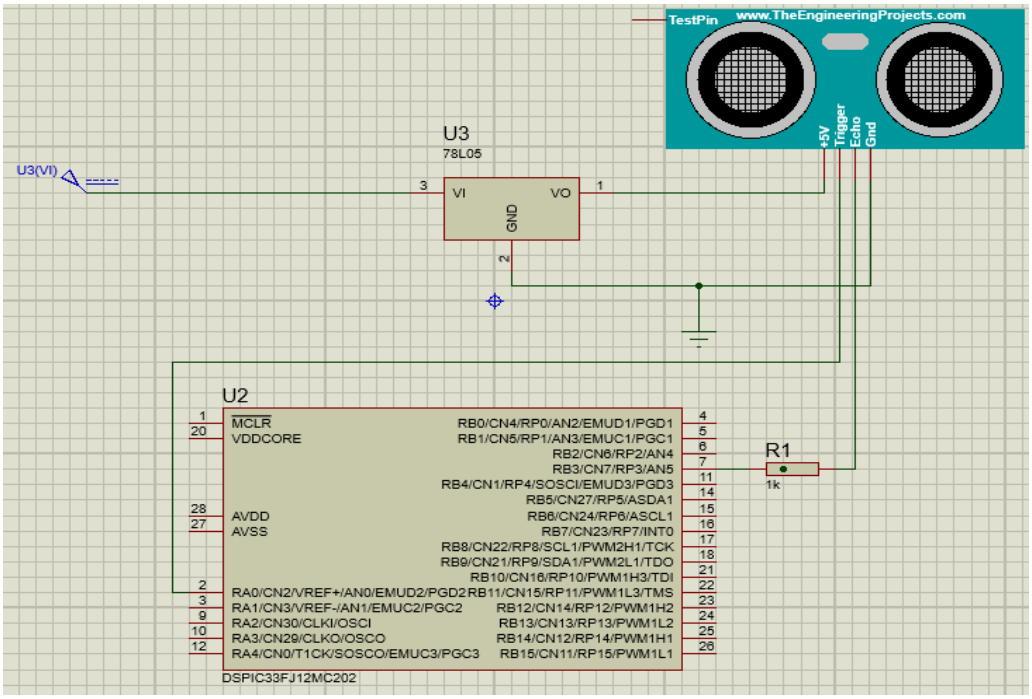


Figure4.4: Interfacer le capteur ultrasonic HC-SR04 avec le dspic33FJ Microcontrôleur

Il dispose d'un transducteur ultrasonore qui génère les ondes ultra-sonic, un récepteur et des circuits de commande ultra-sonic construit sur un petit PCB.

Figure4.5:vue de face de HC-SR04

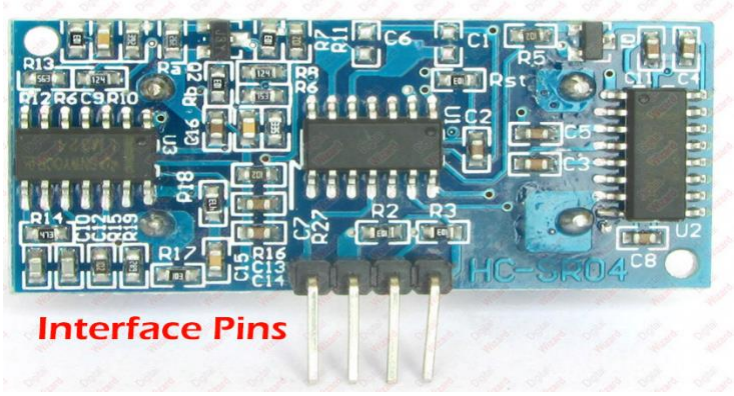
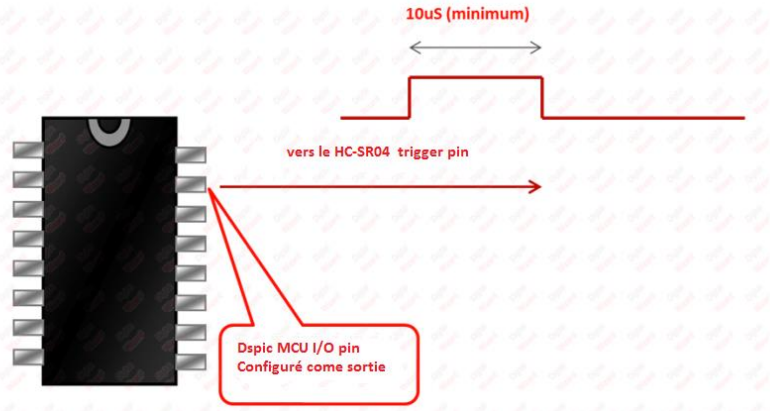


Figure4.6: vue arrière de HC-SR04

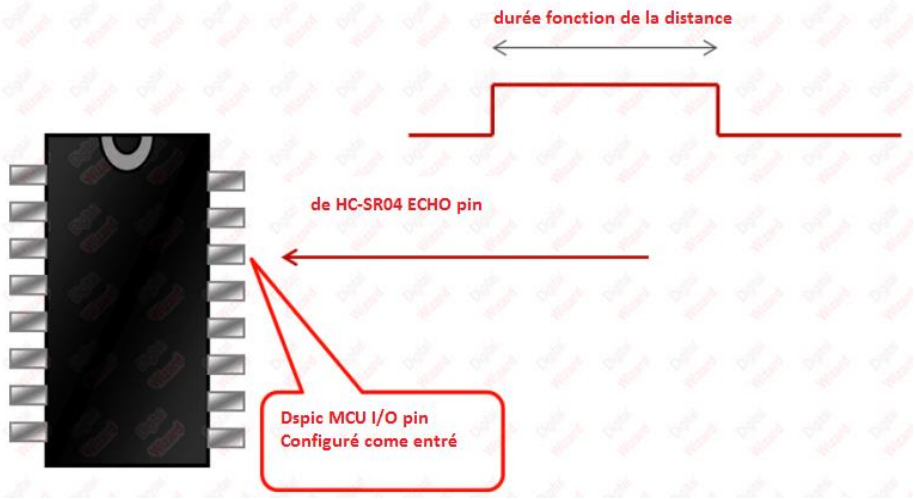
Pour l'interfaçage avec le microcontrôleur, il fournit deux lignes à savoir TRIGGER et ECHO.

La bronche de déclenchement (TRIGGER) connecté sur la pin RA0 est une broche de sortie, le MCU envoie une impulsion de 10us sur cette ligne pour dire au HC-SR04 pour commencer une prise de mesure.



Dès que le HC-SR04 reçoit cette impulsion, il envoie des ondes ultrasoniques et attend qu'elles aillent à l'obstacle et reviennent au capteur. Le capteur émet alors une

impulsion sur la ligne ECHO qui est connecté sur le pin RB3 (Input capture pin) dont la largeur est égale à ce moment. Par simple calcul, nous pouvons trouver la distance à obstacle.



Voici une représentation graphique de la séquence de fonctionnement du module :

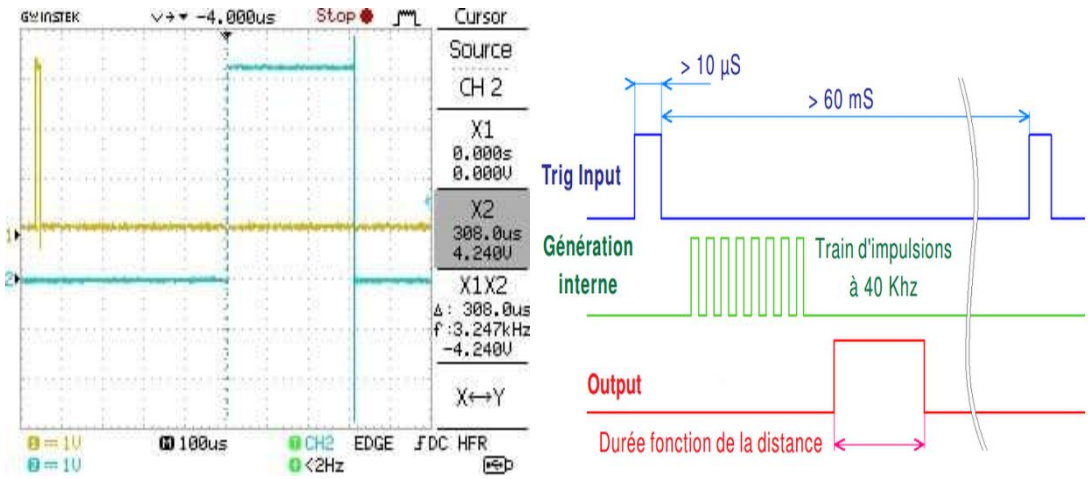


Figure 4.7: Chronogramme relevé sur les voies Trig Input et Output du module us HC-SR04

8. Calcul de distance de l'objet:

La distance parcourue par un son se calcule en multipliant la vitesse du son, environ 340 m/s dans l'air par le temps de propagation, soit :

$$\mathbf{d = v \cdot t}$$

Le HC-SR04 donne une durée d'impulsion en dizaines de µs. Il faut donc multiplier la valeur

obtenue par 10 µs pour obtenir le temps t. On sait aussi que le son fait un aller-retour. La distance vaut donc la moitié.

$$d = 340\frac{m}{s}.10\mu s.\frac{valeur}{2}$$

Finalement, $\boldsymbol{d\ (cm)\ =\ 17/100\ cm\ \cdot\ valeur}$

La formule d = durée/58 cm figure aussi dans le manuel d'utilisation du HC-SR04 car la fraction 17/1000 est égale à 1/58.8235. Elle donne cependant des résultats moins précis.[13]

On a fait des tests avec un capteur ultrason et le graphe suivant illustre la distance vu par le capteur en fonction de la distance mesurée d'un obstacle qui se trouve a diffèrent distances :

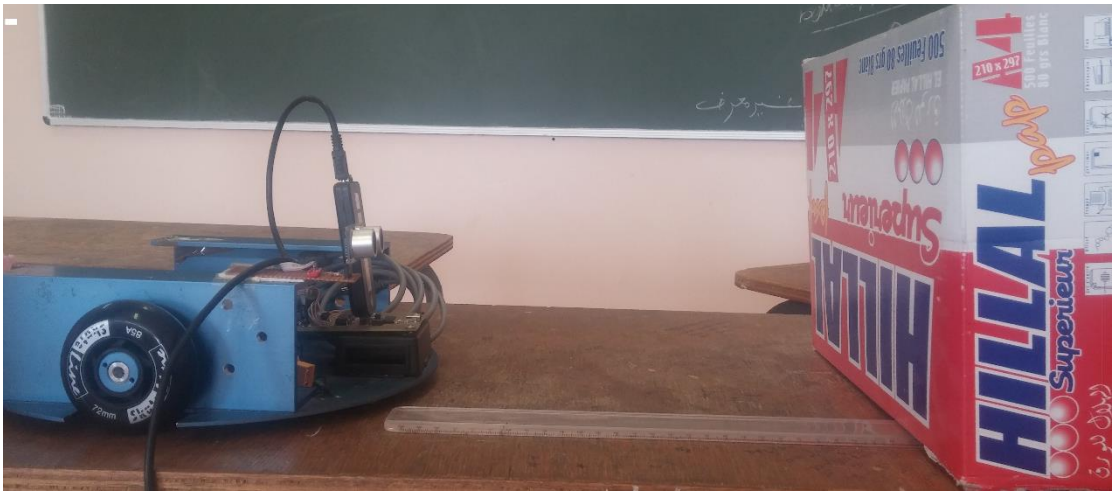


Figure 4.8:essai expérimental de Capteur US

Relevé expérimental :

Distance réelle (cm)	**Distance mesurée (cm)**	**Erreur (cm)**	**Erreur (%)**
6.2	5.6	0.6	9.67
10.6	10.2	0.4	3.77
14.81	14.8	0.01	0.06
18.07	18.7	0.63	3.48
22.07	21.7	0.37	1.67
27.125	27.5	0.375	1.69
37.00	38	1.00	2.70

Tableau 4.2 : La différence entre la distance réelle et la distance mesurée

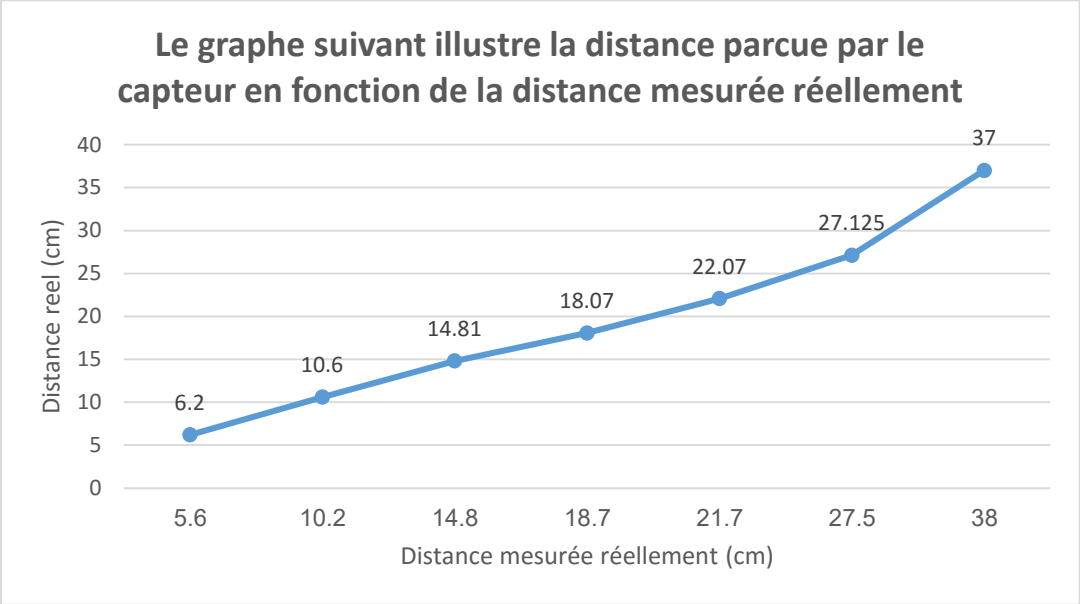


Figure 4.9 : *la distance* perçue *par le capteur en fonction de la distance* mesurée réellement.

Les mesures sont quelque peu décalées, mais cela semble être une erreur statique stable. Mais Globalement le capteur est assez linéaire et les résultats convenables.

9.1 Input Capture :

Les fonctions qui réagissent à un changement d'état d'une entrée du microcontrôleur en générant une interruption sont :

- Input capture : via un couplage au timer, cette fonction permet de mesurer précisément le temps écoulé entre deux changements d'état. (voir figure 4.9)
- Interruptions externes : utilisés principalement pour le comptage d'occurrence de changement d'état.

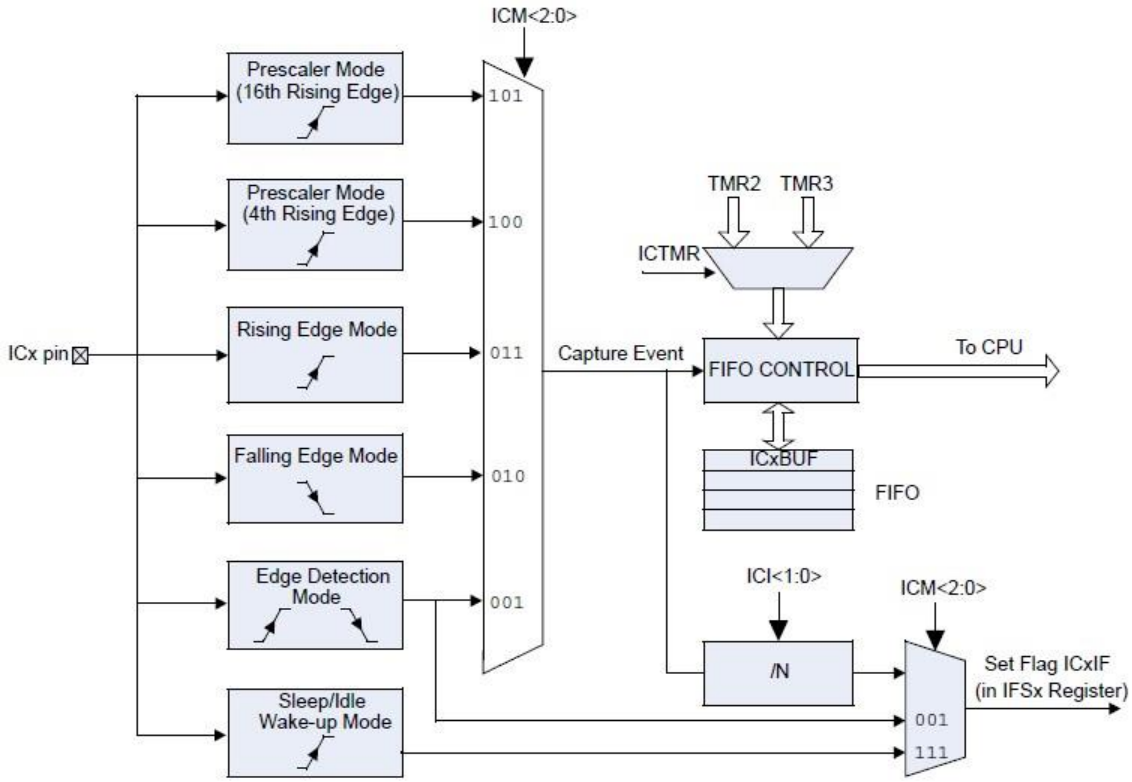


Fig4.10 : diagramme synoptique du circuit interne input capture

On a configuré l'input capture dans notre programme comme suit :

```
void InitInputCapture()
{
// Initialize Capture Module
        IC1CONbits.ICM = 0b000; // Disable Input Capture 1 module
        IC1CONbits.ICTMR = 1;                  // Select Timer2 as the IC1 Time base
        IC1CONbits.ICI = 0b01;          // Interrupt on every second capture event
        IC1CONbits.ICM = 0b001;          // Generate capture event on 001 Edge Capture moc
        // Enable Capture Interrupt And Timer2
        _IC1IP = 1; // Setup IC1 interrupt priority level (higher than Timer1)
        _IC1IF = 0; // Clear IC1 Interrupt Status Flag
        _IC1IE = 1; // Enable IC1 interrupt
}
```

Le registre *ICxCONbits* permet le paramétrage du module d'input capture, dans notre cas on a pris les paramètres suivants :

IC1CONbits.ICM =0b000 ; avant de changer le mode capture on doit en premier le désactiver

IC1CONbits. ICTMR =1 ; cela signifie que lorsque l'interruption survient, le contenu du timer 2 est recopié dans le registre associé à l'input capture.

IC1CONbits. ICI =0b01 ; décale la génération de l'interruption de 1 à 4 survenue, de l'évènement scruté. Dans notre cas l'ICI est mis à 2, l'interruption de l'IC sera générée au deuxième événement non à chaque fois, comme le montre la figure 4.10

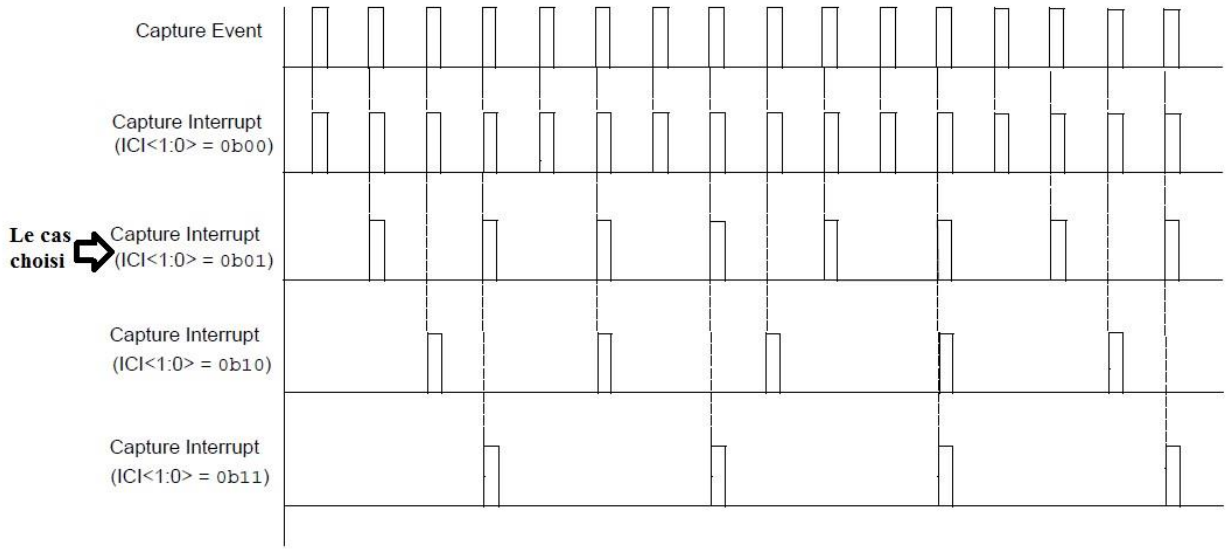


Figure4.11; génération d'interruption de l'input capture

IC1CONbits.ICM =0b001 ; précise l'évènement scruté. On a pris une génération d'interruption à chaque front montant et descendant comme le montre la figure suivante :

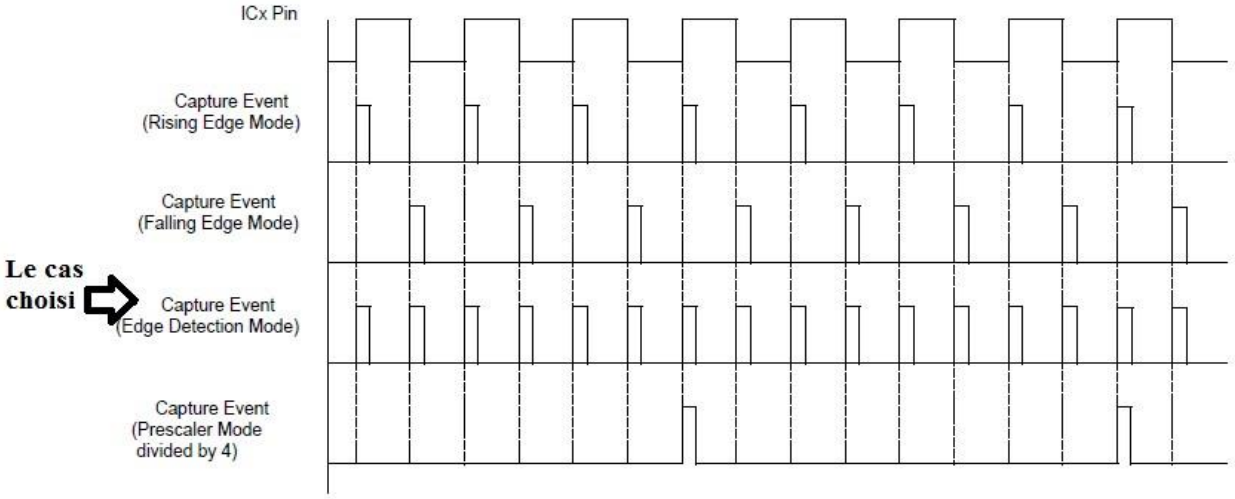


Figure4.12 : génération de l'événement d'input capture

Mais puisque *ICM=001 ;* dans ce cas le décalage de L'<ICI> ne fonctionne pas, et ses bits seront ignoré

Comme pour un timer, après avoir configuré les bits, on active les interruptions de l'IC par :

- _IC1IP = 1; La priorité de l'input capture est supérieure à celle du timer
- _IC1IF = 1; Après s'être assuré au préalable que l'interruption n'est pas en cours, sans quoi le code ne sera jamais appelé
- _IC1IE = 1; On lance l'IC [2].

9.2 Le Timer: [2]

La fonction la plus utile du microcontrôleur est le timer, c'est grâce à ce dernier qu'on peut réaliser une tache avec une fréquence particulière.

Son principe c'est d'incrémenter un registre régulièrement et proportionnellement a la fréquence Fcy, jusqu'à atteindre une certaine valeur, une fois atteinte l'interruption du timer se déclenche permettant de réaliser l'action voulue.

Le dsPIC qu'on a utilisé dispose de 5 timers, mais on n'a utilisé que 3 timers dans la programmation de notre robot,

- ✓ Le timer 2 est utilisé pour la lecture de l'input capture,
- ✓ Le timer 3 est utilisé pour la génération du capteur ultrasonique réglé à $10\mu s$ comme le montre le code suivant :

```
//  Read_US lis les capteurs UltraSons
//-----------------------------------------------------------------------------
void          Read_US()
{
        TMR3= 0;
        TRIG_US = 1;                                    // start the US
        T3CONbits.TON = 1;          // enable Timer 1 and start the count
}
//-----------------------------------------------------------------------------
// Timer3 ISR compte 10 us
//-----------------------------------------------------------------------------
void __attribute__((interrupt, auto_psv)) _T3Interrupt( void )
{
        _T1IF = 0;
        T3CONbits.TON = 0;          // Stop Timer 1 and start the count
        TRIG_US = 0;                                    // stop the US starting signa
}
```

9.3 ROUTINE DE SERVICE DE l'iNTERRUPTION (ISR):

La méthode utilisée pour déclarer un ISR et initialiser l'IVT (Interrupt Vector Table) avec l'adresse de vecteur correcte dépend du langage de programmation (C ou assembleur) et la langue suite d'outils de développement utilisés pour développer l'application.

En général, l'utilisateur de l'application doit effacer l'indicateur d'interruption dans le registre approprié IFSx pour la source d'interruption que le ISR gère. Autrement, le programme revient à nouveau dans la région qu'il a quitté au moment de l'interruption.. [16]

9.4 Le code:

Le programme sera le suivant:

```
 //Capture Interrupt Service Routine
-------------------------------------------------------------------//
----------
void Read_Capture_value()
}
    if(!Int_flag) return;     // measure two falling edge time.
    dist_us = timeDistance;   // ou ajouter un coefficient...
    Interrupt_Count = 0;
    Int_flag = 0  ;
{
   void __attribute__((interrupt, auto_psv)) _IC1Interrupt(void)
}
          Interrupt_Count;++
    if( Interrupt_Count == 1 )
}
        t1 = IC1BUF;    //InputCapture_first_edge
{
    else if( Interrupt_Count == 2 )
}
        t2 = IC1BUF;    //InputCapture_second_edge
        timeDistance = (t2 - t1);
       Int_flag = 1;
        TMR2  = 0;
{
      _IC1IF=0;               // Clear IC1 Interrupt Status Flag
{
```

10. Conclusion:

Le capteur HC-SR04 est intéressant. Pour un coût très bas, il donne des résultats étonnants de précision. L'écart est d'environ 3 cm avec un objet placé à 2 m, ce qui représente une erreur inférieure à 2 %.

Chapitre 5 : Conduite du robot

Chapitre 5 : Conduite du robot

1. introduction :

L'objectif de ce chapitre et d'expliquer comment rendre notre robot mobile capable de suivre visuellement une ligne dans un terrain.

Ce chapitre est composé de deux parties :

- Dans la première partie on a présenté le principe d'un robot mobile suiveur de ligne basé sur deux méthodes :
  - ✓ Suivie de ligne basée sur des capteurs IR (infra-rouge).
  - ✓ Suivie de ligne basée sur la vision de la ligne et recherche de la ligne.
- .La deuxième partie est consacrée à la partie de la Régulation et comment en implante un régulateur PI.

2. Principe d'un robot suiveur de ligne :

Une ligne Robot suiveur est un robot de base conçu pour suivre une ligne ou un chemin prédéterminé.

La construction de cette ligne varie de la simple ligne noire sur le sol à des lignes complexes intégrées, des lignes magnétiques, etc. Afin de détecter ou de tracer le chemin, différents types de systèmes de navigation ou des systèmes de vision sont employés. Ces systèmes de navigation comprennent les capteurs IR et les capteurs plus complexe et coûteux comportant des circuits de vision. Le choix du type de système de navigation dépend des besoins de l'utilisateur. Du point de vue industriel ces robots suiveurs de lignes sont utilisés comme support de marchandises pour transporter les charges utiles d'un site à un autre où des rails ou des bandes transporteuses ne sont pas utilisables. [18]

2. Robot suiveur de ligne basé sur des capteurs infra-rouge :

2.1.1 Les Capteurs infrarouges :

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 dégradé 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. [19]

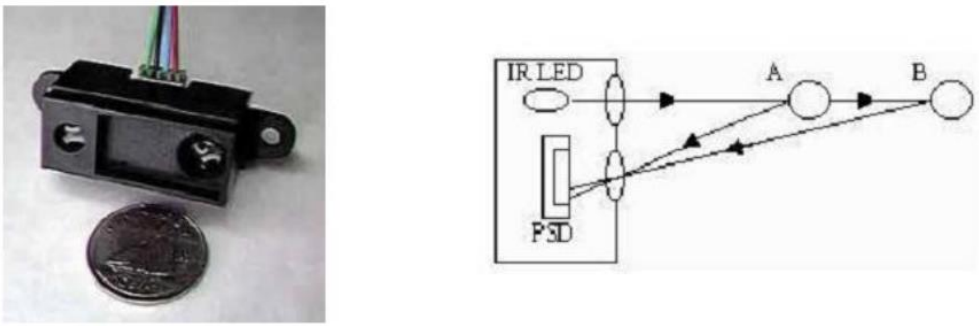
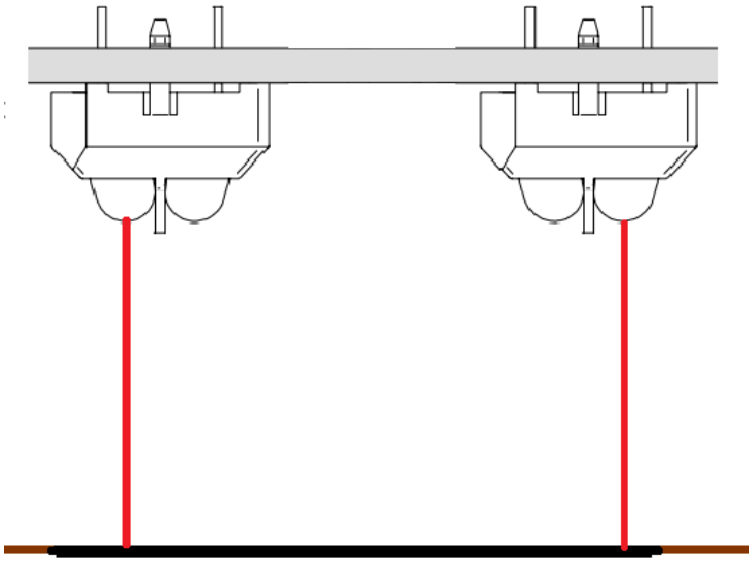


Figure 5.1 : Télémètres infrarouges.

2.1.2. Suiveur de ligne :

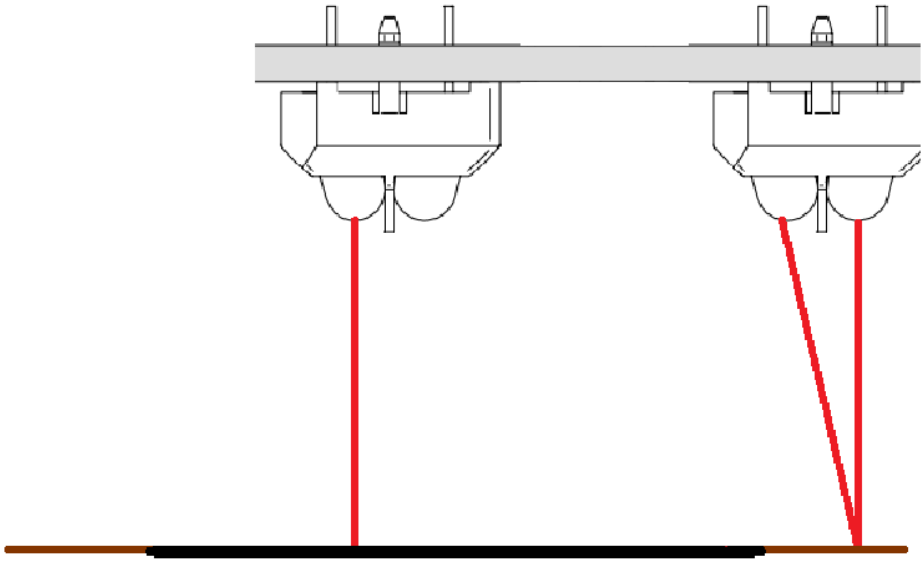
Le suiveur de ligne a un fonctionnement similaire au capteur de distance. La différence est que l'énergie lumineuse sera nulle (ou presque) lorsque la radiation va rencontrer une ligne noir elle ne va pas réfléchir, et lorsqu'elle va rencontrer autre chose qu'une ligne noir (on sera dévié par rapport à la ligne à suivre), le phototransistor va être « activé » et va donc envoyer un courant, qui sera le déclenchement du processus de remise sur le bon chemin de robot. Voilà deux schémas qui expliquent le principe de fonctionnement :



Ici nous voyons bien que le rayonnement n'est pas réfléchi par la ligne noire, le phototransistor ne reçoit rien et le robot ne doit du coup pas modifier sa trajectoire.

Lorsque la ligne n'est plus une ligne droite mais courbe, le capteur droit par exemple n'enverra plus son rayonnement dessus, et la lumière va donc réfléchir et atteindre le

phototransistor : le robot aura donc pour instruction de tourner du bon côté (ici vers la gauche) pour faire en sorte que le phototransistor ne reçoive plus de rayonnement, c'est à dire que les deux capteurs soient directement au-dessus de la ligne noir. [20]



2.2 Robot suiveur de ligne Basé sur la vision de la ligne :

Un robot suiveur de ligne est un robot qui suit une certaine trajectoire contrôlée par un mécanisme de rétroaction. Le chemin peut être visible comme une ligne noire sur une surface blanche (ou vice versa) ou il peut être invisible comme un champ magnétique de détection d'une ligne et de guidage du robot pour rester dans la course. [21]

a. Présentation de système :

Figure5.2 : robot suiveur de ligne

Notre robot est constitué de trois composantes principales : un smartphone Android, une carte d'interface USB, un contrôleur de moteur.

Étant donné la difficulté de protéger les capteurs infrarouges contre la lumière du soleil, nous avons choisi une approche basée sur la caméra. Cela a également été influencé par l'accès facile à l'appareil photo intégré sur notre smartphone. Compte tenu de notre connaissance du traitement du signal, nous étions convaincus que nous pouvions surmonter tous les défis que le traitement de l'image peut présenter.

Nous avons choisi de miser sur la plate-forme Android pour plusieurs raisons. Tout d'abord, comme la plupart des plates-formes de smartphones, les appareils Android offrent un processeur puissant jumelé avec une variété de capteurs dans un format compact, économe en énergie. Pour nos besoins, cela signifiait que nous aurions seulement besoin de matériel externe pour gérer la sortie, pas l'entrée. Android fournit un environnement de développement ouvert et facile à utiliser et est compatible avec la puissante bibliothèque de vision par ordinateur OpenCV.

Android fournit plusieurs options pour l'interfaçage avec le matériel. Jusqu'à récemment, ils ont été limités à des technologies sans fil comme le Bluetooth et le WiFi. Pour notre robot nous avons préféré une solution USB. Finalement, avec un câble OTG de la sortie de smartphone s'est avéré être le système le plus simple et le plus facilement disponible qui a répondu à nos besoins d'interface.

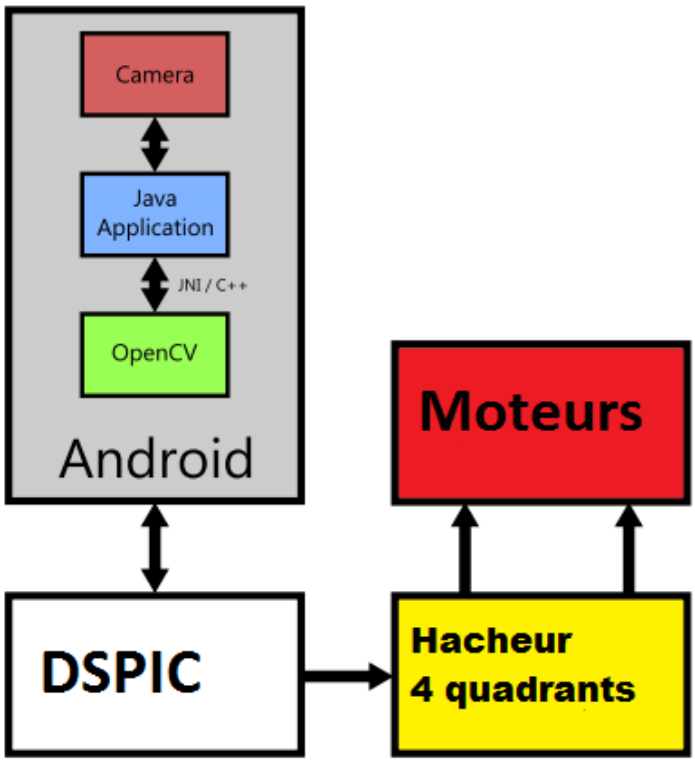


Figure5.3 : Schéma de l'architecture du système

b. Vision :

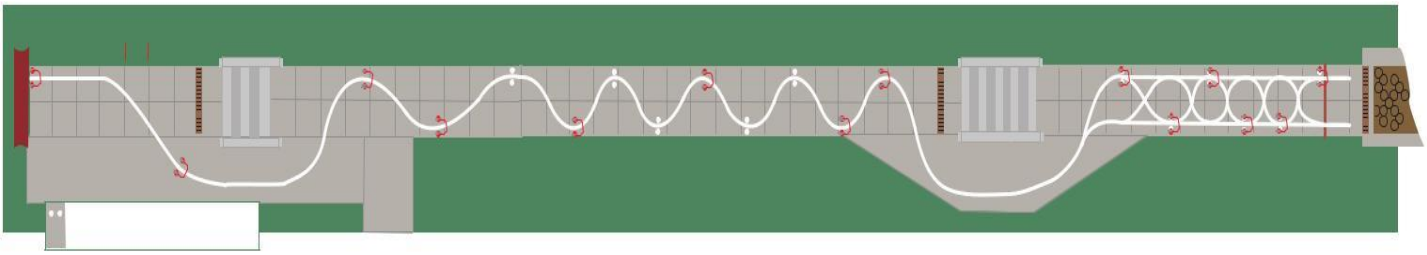


figure5.4 : Trajectoire d'un robot avec une ligne blanche

Lorsque l'appareil Android est prêt à traiter les images reçues de Smartphone, la tâche est maintenant de savoir comment les images sont traitées pour identifier la nouvelle position du robot. Dans notre projet, nous avons utilisé la méthode de suivi sur la base de la couleur.

Donc le principe de suivi de la ligne sera comme suit :

- ✓ on désigne un rectangle R1 sur la partie basse de l'image.
- ✓ convertir l'image RGB vers GRAY.
- ✓ choisir la couleur blanche comme la ligne quand veut la suivre on utilise la fonction threshold.
- ✓ L'image GRAY est dilatée.
- ✓ définir un rectangle et définir son région d'intérêt (ROI).
- ✓ Trouver les contours de la région d'intérêt.
- ✓ Déterminer le plus grand rectangle
- ✓ calculer le centre des deux rectangles (R1 et plus grand rectangle) et faire la différence entre eux.

  - si la différence ≥ 0 tourner à droite
  - sinon tourner à gauche

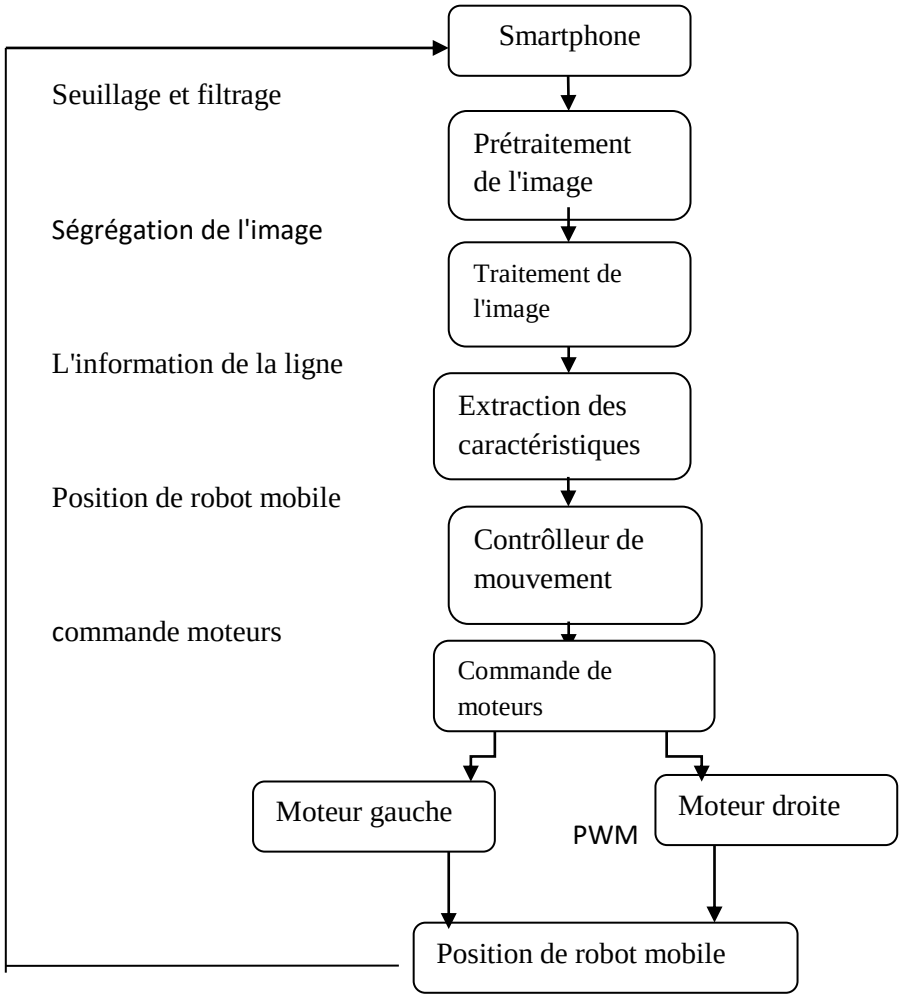


Figure 5.5 : diagramme d'un Robot suiveur de ligne [21]

3. Implantation d'un régulateur P pour un robot suiveur de ligne :

3.1. Régulateur P :

Proportionnel : cette action P Il nous dit à quel point le robot est loin de la ligne comme à droite ou à l'extrême droite, à gauche ou un peu vers la gauche. Proportionnel est le terme fondamental pour calculer les deux autres actions (action I et action D).

Le programme de la régulation P sera le suivant :

```
void Regul()
{
// pas de regul, voir V1 pour le regulateur

PWM1 = 200 +deviation;
PWM2 = 200 -deviation;


        Vref = PWM1 + HalfDUTY;  // passage en manuel
        if (Vref<Vrefmin)        Vref=Vrefmin;
        if (Vref>Vrefmax)        Vref=Vrefmax;
        PDC1 = Vref;
        Vref = PWM2 + HalfDUTY;  // passage en manuel
        if (Vref<Vrefmin)        Vref=Vrefmin;
        if (Vref>Vrefmax)        Vref=Vrefmax;
        PDC2 = Vref;


}
```

3.2. Régulateur PI :

Pour améliorer la réponse de notre contrôleur P, nous avons ajouté un nouveau terme à l'équation. Ce terme est appelé l'action intégrale, le rôle de cette action est d'élimine l'erreur entre le robot et la ligne blanche, mais à cause du manque de temps on n'a pas pu implémenter la régulation dans notre programme.

4. Conclusion :

D’après les résultats obtenus et au cours de la réalisation de ce robot on peut conclure que ce travail peut être divisé en deux grandes parties, la partie de l'application Android qui est basée essentiellement sur la programmation en langage java, et la partie microcontrôleur dspic qui implémente le programme de contrôle rapproché écrit en C. Nous devons faire communiquer les 2 parties pour la réalisation du robot capable de suivre en toute autonomie la ligne tracée les résultats obtenus montrent que le robot peut détecter et poursuivre la ligne détecté.

Conclusion générale :

La poursuite d'une trajectoire est une tâche importante que doit exécuter un robot mobile avec le minimum d'erreurs. Elle constitue la base de toute mission du robot. Pour les robots à commande différentielle, le maintien du robot sur son chemin revient à régler les vitesses des roues motrices de façon à donner au robot l'orientation désirée.

En robotique mobile la poursuite de trajectoire est une tâche élémentaire, qui représente la base de toute autre tâche du robot. Nous avons choisi de développer un système basé sur la vision. L'objet de notre application est le robot mobile Mahfoud 3. La vision est gérée par un smartphone Android avec sa caméra et l'API OpenCV. Les ordres de commande du robot se font avec le dspic qui communique avec le smartphone via une liaison UART sur USB.

En perspectives, il faudrait faire la modélisation sur Matlab du mouvement du robot par rapport à la ligne virtuelle, de manière à copier ce que fait Mahfoud 3 dans la réalité.

Il faut aussi implanter la régulation avec intégrateur basée sur la vision et l'écart du trait trouvé sur l'écran par rapport au centre de l'écran. Pour l'instant, c'est un régulateur proportionnel qui est implanté.

Pour la partie de la communication entre le robot et le smartphone, nous suggérons de remplacer la communication USB par une communication wifi direct ou Bluetooth parce qu'on avait eu des problèmes avec la liaison série USB.

Enfin, utiliser les algorithmes de reconnaissance pour venir repérer et trouver les panneaux de signalisation routière sur la partie droite de l'image (limitation de vitesse, STOP,…).

Pour conclure, ce projet nous a permis, d'une part, d'améliorer grandement nos compétences scientifiques, en électronique embarqués, en capteurs et du côté programmation en C pour dspic et en Java pour Android. D'autre part ce projet nous a permis de nous mettre en situation en tant qu'ingénieur puisque nous avons dû travailler en groupe que ce soit pour la répartition du travail, la programmation du robot ou résoudre les nombreux problèmes rencontrés.

Bibliographie

[1] : Etude et réalisation d'un robot mobile tricycle a commande différentielle. Samir Bouabdallah à université Abou Bekr Belkaid Tlemcen en 2001

[2] : Contrôle d'un robot mobile. Par KHIREDDINE Med el Amine et DRIHEM Nadia, Université Abou Bekr Belkaïd de Tlemcen, en 2015

[3] : Introduction à la robotique par Laetitia Matignon Université de Caen, France

[4] : ROBOTIQUE, ISTIA, Université Angers Jean-Louis Boimond

[5] : Etude et conception d'un Robot marchant par HEDDOUCHE Kamel Université Université Mohamed Khider Biskra, année 2014

[6] : SYSTEME DE LOCALISATION POUR ROBOTS MOBILES par Dr. SLIMANE Noureddine UNIVERSITE DE BATNA en 2005

[7] Starting an Activity
URL:http://developer.android.com/training/basics/activitylifecycle/starting.html

(Last Consulted: 24.06.2013)

[8] Gary Bradski, Adrian Kaehler: Learning OpenCV, 2nd Edition. Computer Vision with the OpenCV Library; O'Reilly Media,Sebastopol, 2012.

[9] Bradski, Gary, and Adrian Kaehler. Learning OpenCV: Computer Vision with the OpenCVLibrary.O'Reilly Media, 2008. Print.

[10] Paul Kinsky, and Quan Zhou. Obstacle Avoidance Robot, WORCESTER POLYTECHNIC INSTITUTE, 2011

[11] Daniel Lélis,and Baggio Shervin,and Emami David,and Millán Escrivá,and Khvedchenia Ievgen,and Naureen Mahmood,and Jason Saragih and Roy Shilkrot . Mastering OpenCV with Practical Computer Vision Projects, 2012 Packt Publishing

[12] : M. Dalmau, « Développement d'applications pour Android »

[13]: Lucien Bachelard "HC-SR04 - Module de détection aux ultrasons - Utilisation avec Picaxe", Le 28 novembre 2015.

[14]: Détecteur d'obstacles et distance avec Ultrason

http://www.technologuepro.com/montageselectroniques/capteurdistanceultrason21.html 2/10

[15] : dsPIC33FJ32MC302/304, dsPIC33FJ64MCX02/X04 AND dsPIC33FJ128MCX02/X04, datasheet documents disponible en ligne sur : http://ww1.microchip.com/downloads/en/DeviceDoc/70291G.pdf

[16] : Bouali Abdelmalek "Planification de trajectoire pour un robot mobile",30 /06/2012.

[17]: L.G, R.V, C.S "Developing Manual Control for a Line Follower Robot ", Department of Electronics and Communication Engineering Amity University, Noida, India, Number 3 (2013).

[18]: "Conception et réalisation de robot suiveur de ligne", FSR_2008

[19]: François Desprez,Jérémy Tachel"PROJET ISN 2014 Voiture robot",PROJET ISN 2014.

[20]: Nolan Hergert, William Keyes, and Chao Wang," Smartphone-based Mobile Robot Navigation", 18-551 FINAL REPORT, SPRING 2012

[21]: http://artiom-fedorov.blogspot.com/2012/12/robot-suiveur-de-balles-opencv-projet.html

Résumé :

Notre projet reprend un PFE précédent pour commander un robot mobile à deux roues motrices pilotés par hacheur 4-quadrants et commandées par dspic. Ce dernier est interfacé avec un smartphone Android via une liaison série bidirectionnelle UART sur USB.

Nous développons un algorithme de suivi de la ligne au sol, basé sur l'analyse d'images du flux vidéo de la caméra du smartphone grâce à la librairie OpenCV. L'information est ensuite communiquée au dspic qui assure le pilotage. Une détection de distance est également ajoutée grâce à un capteur ultrason couplé aux entrées de capture du dspic.

Mots clés : Robot, capteur ultrasonique, contrôle, dspic, MLI, Android, OpenCV

Abstract:

Our project continue a previous PFE to control a 2-wheel driven mobile robot with 4-quadrant choppers and controlled by dsPIC. It is interfaced with an Android smartphone thanks to a bidirectional serial link UART over USB.

We develop a line tracking algorithm based on image analysis of video streams of the smartphone's camera using the OpenCV library. The information is then communicated to the dsPIC that provides control. Range sensing is also implemented using an ultrasonic sensor connected to the dsPIC input capture port.

Keywords : Robot, ultrasonic sensor, control, dspic, PWM, Android, OpenCV

ملخص

يستمر مشروع السنة الماضية للسيطرة على عجلتين مدفوعتين للروبوت المتحرك مع مقطع رباعي والذي يتحكم فيه dsPIC. هذا الاخير مربوط مع الهاتف الذكي اندرويد عن طريق ارتباط تسلسلي ثنائي الاتجاه usb.

طورنا خوارزميه للروبوت لتتبع الخط في الارض وهي قائمه على معالجه الصوره المتدفقه من الكاميرا الخاصه بالهاتف الذكي على برمجيه الرؤيا الحاسوبيه. بعد هذا المعلومه تتواصل مع المعالج الذي يضمن التحكم. وإلحاق اضافه الاستشعار عن بعد ايضا تم اضيفت بفضل متحسس فوق صوتي مرتبط في مدخل القبض في المعالج dspic.

الكلمات المفتاحية: روبوت، جهاز الاستشعار بالموجات فوق الصوتية، مراقبة ،المعالج dsPIC، ، التعديل النبضي MLI،PWM، اندرويد ، مكتبة برمجية مفتوحة للرؤية الحاسوبية OpenCV