République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique

2014/2015

I.A.E.S

Université de Blida 1, SAAD DAHLEB
Institut d'Aéronautique et des Etudes Spatiales

En Vue de l'Obtention du Diplôme Master
Option : Avionique-CNS/ATM

Thème :

Commande d'un Bras manipulateur par Pc

Réalisé par :

Seghir Narimane

Zeddam Samia

Encadré par :

Mr: Z. Banselma

Mr: A. Bencharchali

Dédicace

Je dédie ce modeste travail

A ma très chère mère,

A mon très cher père,

A mes sœurs IBTISSAM, YASMINE, SARAH,

A ADEM

A toute ma famille SEGHIR et HAMDI,

A mon binôme SAMIA,

A tous mes collègues

HOUDA, HAYAT, KAWTHER, HAMID, ISMAIL, SABER, DJALAL,

*Marhoum *NABIL**

Et A tous mes amis et connaissances, proche et lointains.

NARIMANE

Dédicace

Je dédie ce modeste travail

A ma très chère mère,

A mon très cher père,

A ma très chère grande mère,

A mes sœurs CHAFIA, KHADIDJA, MERIEM,

A mes frères HOUSSIN, MOUHAMED

A SIRINE, RAYANE,

A toute ma famille,

A mon binôme NARIMANE,

A tous mes collègues

HOUDA, KAWTHER, HAYAT, HAMID, ISMAIL,

MOUHAMED, SABER, DJALAL

Et A tous mes amis et connaissances, proche et lointains.

SAMIA

Remerciements

Tout d’abord, Nous tiens à remercier ***DIEU*** le miséricordieux de nos avoir donné lapossibilité de réaliser notre projet, d’arriver à notre souhaits et d’atteindre notre objectifs.

Nous aimerons dans ces quelques lignes remercier toutes les personnes qui d’unemanière ou d’une autre, ont contribué au bon déroulement de notre travail, tout au niveau humain qu’au niveau scientifique.

Nous tenons tout d’abord à remercier notre encadreur MONSIEUR, ***Zoubir Benselama***, on a pu bénéficier à la fois de ses compétences, et de sa grandedisponibilité, tant pour résoudre les difficultés rencontrées lors de notre réalisation, de répandre à nos questions.

Nous ajoutons en particulière sa patience et ses encouragements, nous a permis detravailler dans bonnes conditions.

Nous exprimons notre gratitude à MONSIEUR, ***Mohamed Amin Bencharchali*** qui a contribué notre travaille, on les prie de bien vouloir croire à notre gratitude en espérant que cet humble travail fera crédibilité de leurs efforts.

Nos remerciements s’adressent également à tous membres de Jury, qui ont accepté denous honorer de leur présence et de juger notre travail Merci.

Et à toute personne ayant contribué de près ou de loin à notre soutien moral.

ملخص

الهدف من مشروع نهاية الدراسة انجاز ذراع تحكم عن طريق جهاز كمبيوتر ولهذا المشروع يتطلب جزءين هامين :

الجزء الاول هو تطبيق محض يتمحور حول انجاز واجهة جهاز كمبيوتر مشتركة مع soft word الذي يعمل على معالجة المعطيات الاحدثيات التي تسمح بالمرور من موضع ابتداءي الى موضع نهاءي من خلال عدد التذبذبات لكل محور وكذلك تحويل المعطيات عن طريق تسلسلي

الجزء الثاني يحتوي على بطاقة التحكم على اساس الاردوينو لتسيير المحركات بفضل حامل مزدوج و **PWM لتمويل مداخل المقاطع ورد فعل مواضع المحركات**

المخطط هذا يتبع المسار التالي :

نبدا بمقدمة عامة تضعنا في الصورة حول الروبوتات بصفة خاصة ذراع التحكم الذي يسمح بالاطلاع على الوساءل الهامة للوصول الى تطبيق عملي،وفيمايلي نعرض بطاقة التحكم والمعالجة لمعرفة بطاقة الاردوينو والواج لجعل عنصر التحكم سهلة الاستخدام المنجزة لننهي مشروعنا بخاتمة شاملة

Abstract

The objective of the final project study and implementation ARM MANIPULATOR PC controlled. For this, the project will comprise two parts, the first purely Soft will focus on the realization of a PC software interface associated with making the treatment of Cartesian allowing passage of an initial position to a final position terms of number of pulses for each joint as well as data transmission through the serial port.

The second part will include a control board and microcontroller-based control for managing engines through binary respective ports and PWM for servo interrupt inputs and the feedback position of the motors.

The manuscript will follow this path: We begin with a general introduction to situate our application, this will be followed generalities on Robotics and especially the manipulator arms. The latter will allow us to see the tools to realize our application that aims to be practical and user-friendly. Later we will present the card processing and control namely Arduino prototyping board and interface to achieve to make the user-friendly control from the PC and we will finish with a conclusion

RESUME

L'objectif du projet de fin d'étude et de réalisation BRAS MANIPULATEUR commandé par PC. Pour cela, le projet va comporter deux grandes parties, la première purement Soft s'articulera autour de la réalisation d'une interface PC associée à un software faisant le traitement des données cartésiennes permettant le passage d'une position initiale vers une position finale en terme de nombre d'impulsions pour chaque articulation ainsi que la transmission des données à travers le port série.

La deuxième partie comportera, une carte de commande et de contrôle à base de microcontrôleur pour la gestion des moteurs grâce aux ports respectifs binaire et PWM pour l'asservissement des entrées d'interruption et pour le feedback des positions des moteurs.

Le manuscrit suivra le cheminement suivant : Nous commencerons par une introduction générale pour situer notre application, celle-ci sera suivie des généralités sur la robotique et en particulier les bras manipulateurs. Cette dernière nous permettra de voir les outils nécessaires pour réaliser notre application qui se veut être pratique et conviviale. Par la suite nous présenterons la carte traitement et de commande à savoir la carte de prototypage ARDUINO et de l'interface à réaliser pour rendre la commande convivial à partir du PC et nous terminerons par une conclusion.

LISTE DES FIGURE

DEDECACES

REMERCIEMENT

RESUME

INTRODUCTION GENERALE

I *ROBOTIQUE*

II *LA CARTE ARDUINO*

III *Réalisation*

Figure I

Figure II

Figure III

Depuis le milieu des années 70, la robotique est devenue une science extrêmement populaire dans les milieux universitaire. Alliant un grand intérêt pédagogique et industriel, Cette nouvelle science demande beaucoup de créativité et des connaissances pluridisciplinaires (Mécanique, Electronique numérique et analogique, électrotechnique, Programmation, Intelligence artificielle, Temps réel, Automatique…..)

Nous avons choisi la commande d'un bras manipulateur par Pc, ce mémoire se base sur le traitement de l'étude théorique (cinématique) sur le bras et études théorique et pratique sur le microcontrôleur (carte de commande), et on a programmé finalement cette dernière. Enfin on a créé une interface graphique sur le micro-ordinateur

Notons que la commande sera par PC ce qui nous donne la possibilité de modifier et développer l'interface suivant la méthode d'asservissement et de commande qu'on la veut utiliser , et le signal PWM généré par le microcontrôleur permet de commander la tension des moteurs sans de changer l'amplitude et la fréquence ce qui nous permet aussi d'appliquer les différents lois de commande (PID, réseau de neurones, logique floue,…).

Organisation du mémoire :

Ce travail est divisé en trois chapitres : Dans le premier chapitre, nous commençant par une généralité sur les robots (ses constituants et classifications), puis on passe à la description du bras manipulateur. Le deuxième chapitre est consacré pour les différents types de carte Arduino, leurs compositions et leurs spécifications et on va baser sur le type Arduino uno. Enfin le contenu du troisième chapitre présente à la réalisation hardware et software et les autres composants constituants la carte.

I.1 Introduction

La robotique est un ensemble de disciplines techniques (mécanique, électronique, automatique, informatique) articulées autour d'un objectif et d'un objet communs. Cet objectif est l'automatisation flexible de nombreux secteurs de l'activité humaine réputés jusqu'à très récemment comme ne pouvant se passer de la présence de l'homme, et l'objet est le robot, sorte de machine universelle dont l'homme rêve depuis toujours (mais qui n'a pas encore la forme de ce rêve !). Historiquement, le terme «robot» a été introduit en 1920 par l′écrivain tchèque Karel Čapek dans sa pièce de théâtre RUR (Rossum′s Universel Robots). Ce terme, provenant du tchèque robot signifie «travail forcé », désigne à l′origine une machine androïde capable de remplacer l′être humain dans toutes ses tâches. Ensuite, dans les années quarante, les progrès de l′électronique permettent de miniaturiser les circuits électriques (inventions des transistors et circuits intégrés), ouvrant ainsi de nouvelles horizons à la fabrication de robots. Dans les premiers temps de la robotique, le robot est considéré comme une imitation de l′homme, aussi bien fonctionnelle que physique. Aujourd′hui, les constructeurs ne tentent plus de reproduire l′aspect humain sur un robot, privilégiant avant tout sa fonctionnalité. Actuellement, les robots sont très répandus dans le secteur industriel, notamment en construction automobile et chez la plupart des fabricants d′ordinateurs. Leurs capacités d′effectuer rapidement des travaux répétitifs ne cessent de croitre. Ils sont notamment utilisés dans les chaînes de montage et de fabrication. On les emploie également dans des environnements difficilement supportables par l′homme caractérisés par des conditions extrêmes de température ou de pression, radioactivité élevée, etc... L′industrie du nucléaire a ainsi largement contribué au développement de la robotique, notamment dans la conception de bras télémanipulateurs.

I.2 Définitions

La définition que l'on donne actuellement du robot industriel diffère quelque peu selon les pays.

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.

C'est au Japon que la définition est la plus vague. Où le rôle du robot y est essentiellement de servir d'intermédiaire entre l'homme et la machine. On le qualifie de :

« Tout mécanisme permettant d'effectuer, en tout ou en partie, une tâche normalement réalisée par l'homme »

Le rôle du robot y est essentiellement de servir d'intermédiaire entre l'homme et la machine. Il permet de changer le système de production d'un système à interaction directe entre la machine et l'homme en un système où l'homme gère la machine par l'intermédiaire du robot. Le peu de précision de la définition japonaise du robot a une incidence immédiate sur les statistiques correspondantes : Le JIRA (Japan Industrial Robot Association) a recensé en 1981 77000 robots industriels dans l'industrie japonaise. Toutefois, plus de 50% de ces robots sont des manipulateurs à séquence fixe qui ne seraient pas recensés comme tels aux USA ou en France, où la définition d'un robot est beaucoup plus restrictive. La définition américaine du robot (Robot Institue of America) est beaucoup plus spécifique :

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

C'est la définition de l'Association Française de Robotique Industrielle (*A.F.R.I*) qui est la plus explicite, et aussi la plus proche de la technologie actuelle des robots :

« Un robot industriel est une machine formée de divers mécanismes comportant divers degrés de liberté, ayant souvent l'apparence d'un ou de plusieurs bras se terminant par un poignet capable de maintenir un outil, une pièce ou un instrument de contrôle. En particulier, son unité de contrôle doit contenir un système de mémorisation, et il peut parfois utiliser des accessoires sensitifs et adaptables qui tiennent compte de l'environnement et des circonstances. Ces

En plus de la définition américaine, la définition française, du fait qu'elle envisage la perception de l'environnement par le robot, implique une certaine prise de décision. Elle annonce la génération des robots dits « intelligents ».

I.3 Constituants d'un robot

La Figure **I.1** représente les vocabulaires du robot :

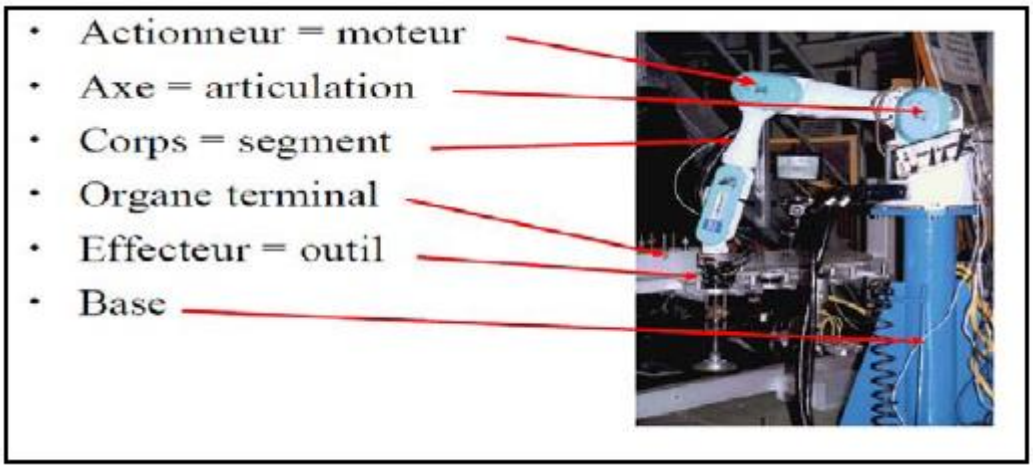


Figure I.1: *vocabulaire du robot*

On distingue classiquement 4 parties principales dans un robot manipulateur :

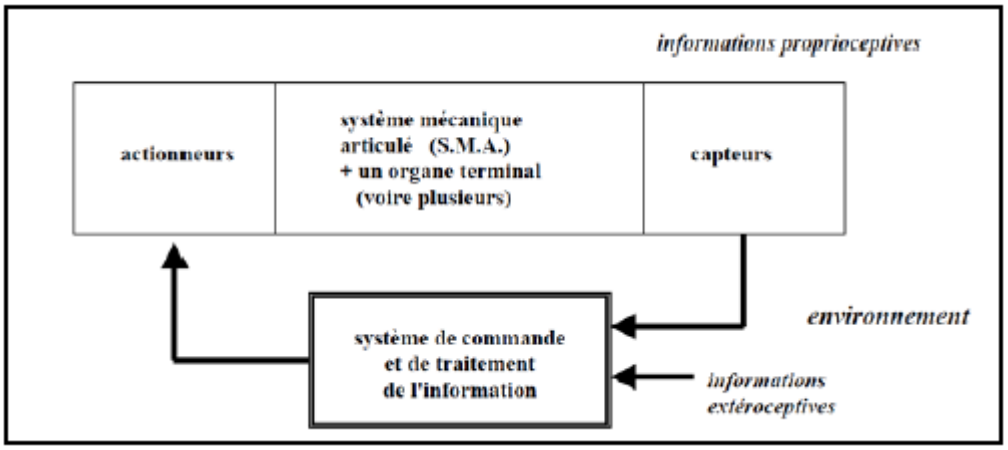


Figure I.2: *parties principales dans un robot*

a) Organe terminal : tout dispositif destiné à manipuler des objets (dispositifs de serrage, dispositifs magnétiques, à dépression, …), ou à les transformer (outils, torche de soudage, pistolet de peinture, …). En d'autres termes, il s'agit d'une interface permettant au robot d'interagir avec son environnement. Un organe terminal peut être multifonctionnel, au sens où il peut être équipé de plusieurs dispositifs ayant des fonctionnalités différentes. Il peut aussi être monofonctionnel, mais interchangeable. Un robot, enfin, peut-être multi-bras, chacun des bras portant un organe terminal différent. On utilisera indifféremment le terme organe terminal, préhenseur, outil ou effecteur pour nommer le dispositif d'interaction fixé à l'extrémité mobile de la structure mécanique.

b) Le système mécanique articulé (S.M.A.) : est un mécanisme ayant une structure plus ou moins proche de celle du bras humain. Il permet de remplacer, ou de prolonger, son action (le terme "manipulateur" exclut implicitement les robots mobiles autonomes). Son rôle est d'amener l'organe terminal dans une situation (position et orientation) donnée, selon des caractéristiques de vitesse et d'accélération données. Son architecture est une chaîne cinématique de corps, généralement rigides (ou supposés comme tels), assemblés par des liaisons appelées

c) articulations. Sa motorisation est réalisée par des actionneurs électriques, pneumatiques ou hydrauliques qui transmettent leurs mouvements aux articulations par des systèmes appropriés.

d) **Les actionneurs :** le S.M.A. comporte des moteurs le plus souvent avec des transmissions (courroies crantées), l'ensemble constitue les actionneurs. Les actionneurs utilisent fréquemment des moteurs électriques à aimant permanent, à courant continu, à commande par l'induit (la tension n'est continue qu'en moyenne car en général l'alimentation est un hacheur de tension à fréquence élevée ; bien souvent la vitesse de régime élevée du moteur fait qu'il est suivi d'un réducteur, ce qui permet d'amplifier le couple moteur). On trouve de plus en plus de moteurs à commutation électronique (sans balais), ou, pour de petits robots, des moteurs pas à pas. Pour les robots devant manipuler de très lourdes charges (par exemple, une pelle mécanique), les actionneurs sont le plus souvent hydrauliques, agissant en translation (vérin hydraulique) ou en rotation (moteur hydraulique). Les actionneurs pneumatiques sont d'un usage général pour les manipulateurs à cycles (robots tout ou rien). Un manipulateur à cycles est un S.M.A. avec un nombre limité de degrés de liberté permettant une succession de mouvements contrôlés uniquement par des capteurs de fin de course réglables manuellement à la course désirée (asservissement en position difficile dû à la compressibilité de l'air).

e) **Les capteurs :** Les organes de perception permettent de gérer les relations entre le robot et son environnement. Les capteurs Dits proprioceptifs lorsqu'ils mesurent l'état interne du robot (positions et vitesses des articulations) et extéroceptifs lorsqu'ils recueillent des informations sur l'environnement (détection de présence, de contact, mesure de distance, vision artificielle).

- La partie commande **:** synthétise les consignes des asservissements pilotant les actionneurs, à partir de la fonction de perception et des ordres de l'utilisateur. S'ajoutent à cela :

- L'interface homme-machine à travers laquelle l'utilisateur programme les tâches que le robot doit exécuter,
- Le poste de travail, ou l'environnement dans lequel évolue le robot.

I.4 Classification des robots

On peut classer les robots d'un point de vue fonctionnel ou d'après leur structure géométrique.

I.4.1 Classification fonctionnelle

Le nombre de classe et les distinctions entre celles-ci varient de pays à pays (6 classes au Japon, 4 en France). L'A.F.R.I. distingue 4 classes illustrées ci-dessous :

I.4.1.a manipulateurs à commande manuelle ou télécommande

La Figure **I.3** représente les manipulateurs à commande manuelle :

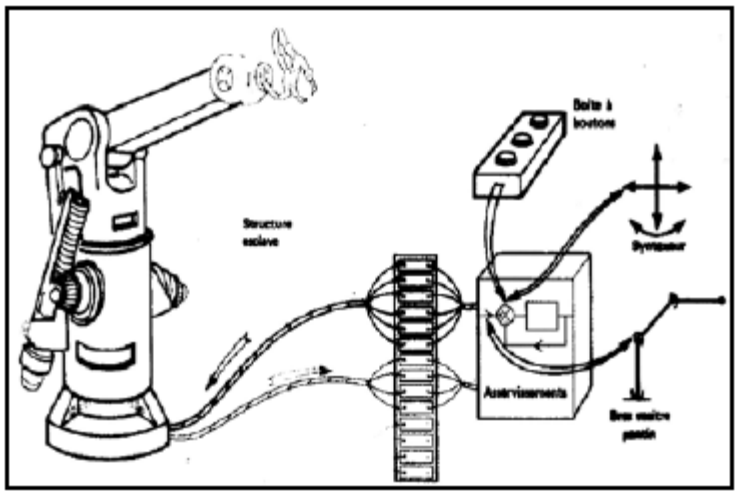


Figure I.3: *manipulateurs à commande manuelle*

I.4.1.b manipulateurs automatiques à cycles préréglés

Le réglage se fait mécaniquement par cames, butées comme le montre la Figure I.4, la commande peut se faire par automate programmable on peut distinguer entre manipulateurs à cycle fixe et manipulateurs à cycle programmable.

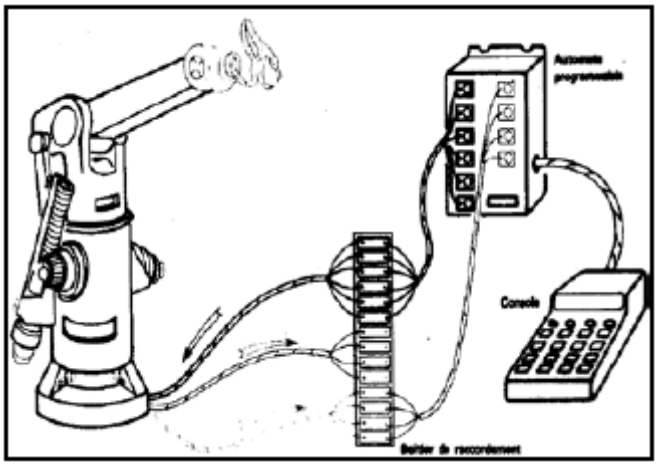


Figure I.4: *Manipulateur à cycle préréglé*

I.4.1.c robots programmables

C'est la première génération de robots industriels ; ils répètent les mouvements qu'on leur a appris ou programmés, sans informations sur l'environnement ou la tâche effectuée. On peut aussi faire la distinction entre robots « play-back » qui reproduisent la tâche apprise et robots à commande numérique qui peuvent être programmés hors-ligne. Pour de nombreux robots, l'apprentissage de la tâche se fait à l'aide d'un « syntaxeur » (« boite à boutons », « teach pendant ») qui permet à un opérateur d'amener le robot en un certain nombre de points, qui sont ensuite mémorisés , lors de l'exécution de la tâche, le robot suivra une trajectoire passant successivement par tous les points programmés, le passage d'un point au suivant se faisant suivant un profil de vitesse en fonction du temps qui est prédéfini (triangulaire ou trapézoïdal), l'opérateur n'ayant qu'à choisir la fraction de la vitesse maximum à laquelle il souhaite que le robot effectue la tâche. Pour certains robots, par exemple les robots de peinture, qui doivent suivre une trajectoire complexe qu'il est difficile d'exprimer mathématiquement, un opérateur humain spécialiste de la tâche effectue la trajectoire en guidant le bras du robot à l'aide d'un « pantin », et l'entièreté de la trajectoire est mémorisée par le robot et la figure **I.5** représente les robots programmable :

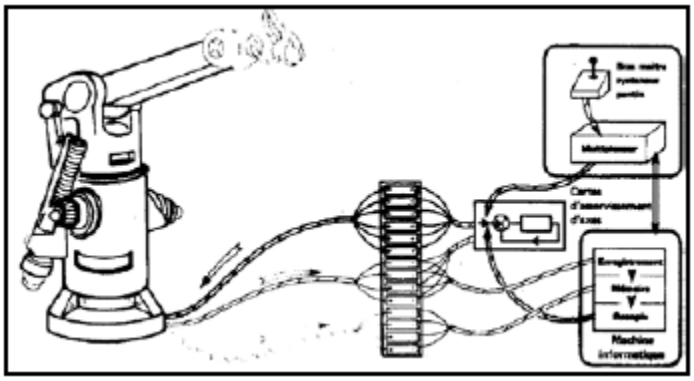


Figure I.5: *Robot programmable*

I.4.1.d robots intelligents

On trouve actuellement des robots de seconde génération qui sont capables d'acquérir et d'utiliser certaines informations sur leur environnement (systèmes de vision, détecteurs de proximité, capteurs d'efforts,...) comme le montre la Figure I.6. On étudie des robots de troisième génération, capables de comprendre un langage oral proche du langage naturel et de se débrouiller de façon autonome dans un environnement complexe, grâce à l'utilisation de l'intelligence artificielle.

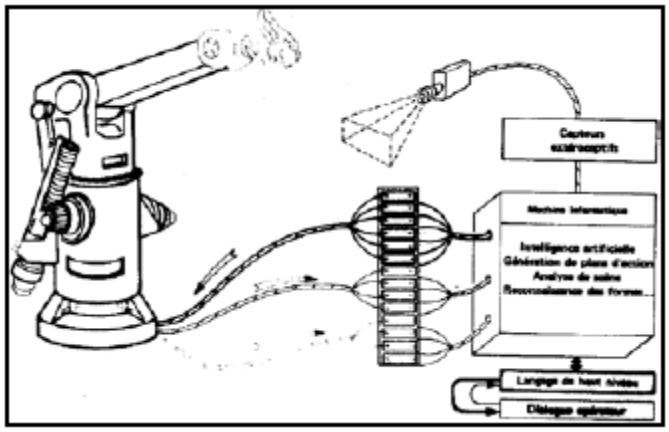


Figure I.6: *Robot intelligent*

I.4.2 Classification géométrique

On peut aussi classer les robots suivant leur configuration géométrique, autrement dit l'architecture de leur porteur. Les 3 premiers ddm d'un robot peuvent être réalisés avec un grand nombre de combinaisons de translations (max. 3T) et de rotations (max. 3R), autrement dit par des articulations **prismatiques** (P) ou **rotoïdes** (R) ; en pratique, on n'utilise que 4 ou 5 d'entre elles :

- porteur cartésien (TTT ou PPP) : les 3 axes sont animés d'un mouvement de translation.
- porteur en coordonnées cylindriques (RTT ou RPP) : un mouvement de rotation et une translation axiale, complétées par une translation radiale 3(voir figure I.7).
- porteur en coordonnées polaires ou sphériques (RRT ou RRP) : deux rotations (longitude et latitude) autour d'axes orthogonaux, complétées par une translation radiale comme le montre la figure I.8.
- porteur en coordonnées universelles, appelé aussi configuration poly articulée ou anthropomorphe (RRR), trois rotations dont les deux dernières se font autour d'axes parallèles orthogonaux au premier, les trois articulations correspondant respectivement au tronc (base), à l'épaule et au coude d'un être humain.

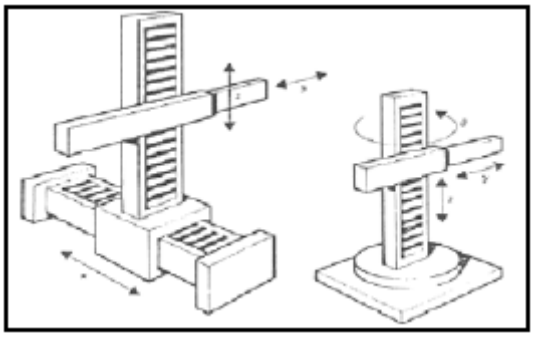


Figure I.7: *Coordonnées cartésiennes et cylindriques*

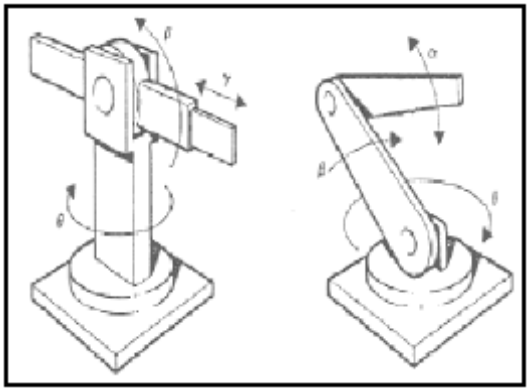


Figure I.8: *Coordonnées polaires et universelles*

Une cinquième architecture comprend deux rotations autour de deux axes parallèles, précédées ou suivies d'une translation dans la même direction (éventuellement celle-ci peut être reportée au niveau du poignet, qui peut aussi tourner autour du même axe, soit au total 4 ddm). Cette architecture est celle des robots SCARA (Selective Compliance Arm for Robotic Assembly) que l'on utilise dans des opérations d'assemblage et la figure I.9 représente le robot scara.

Figure I.9: *Robot SCARA*

I .5 Les différents types des robots

I .5.1 Les robots SCARA :

SCARA = Selective Compliance Articulated Robot for Assembly.

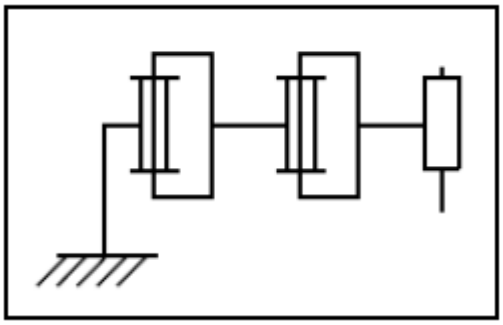


Figure I.10:*schéma de Les robots SCARA*

Caractéristiques

- 3 axes, série, RRP, 3 DDL.
- Espace de travail cylindrique.
- Précis.
- Très rapide.

Exemples

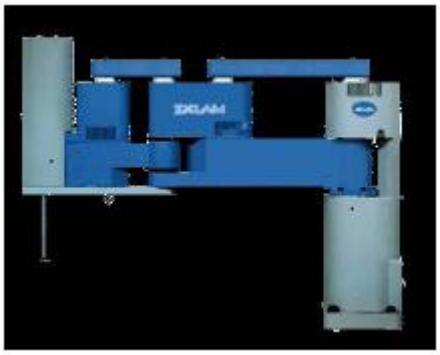


Figure I.11:*Robot Sankyo*

Figure I.12: *robot Adept*

I .5.2 Les robots cylindriques

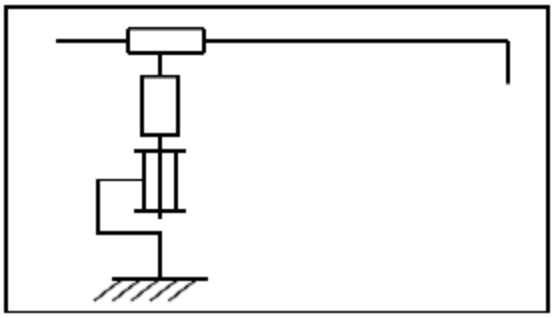


Figure I.13: *Le robot cylindrique*

Caractéristiques

- 3 axes, série, RPP, 3 DDL.
- Espace de travail cylindrique.
- Très rapide.

Exemples

Figure I.14: *Robot Seiko*

I .5.3 Les robots sphériques

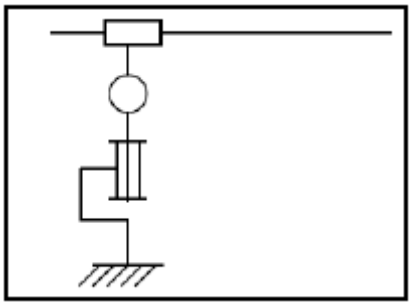


Figure I.15: *robot sphérique*

Caractéristiques

- 3 axes, série, RRT, 3 DDL.
- Espace de travail sphérique.
- Grande charge utile.

Exemples

Figure I.16 *: ROBOT FANUC*

I .5.4 Les robots Cartésiens

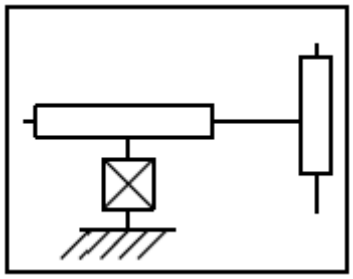


Figure I. 17 : *robot Cartésien*

Caractéristiques

- 3 axes ⊥ 2 à 2, série, PPP, 3 DDL.
- Très bonne précision.
- Lent.

Exemples

Figure I.18 : *robot Toshiba*

I .5.5 Les robots parallèles

Caractéristiques

- Plusieurs chaînes cinématiques en parallèle.
- Espace de travail réduit.
- Précis (grande rigidité de la structure).
- Rapide.

Exemples

Figure I.19 : *robot COMAU*

I .5.6 Les robots anthropomorphe

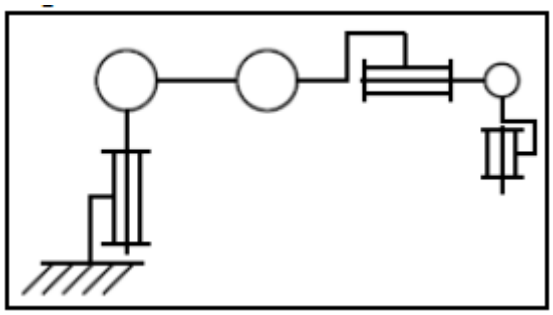


Figure I.20 : *robot anthropomorphe*

Caractéristiques

- Reproduisent la structure d'un bras humain.
- 6 axes, série, 6R, 6 DDL.

Exemples

- Architecture standard :

Figure I.21 : *robot Kawasaki*

- Architecture à parallélogramme

Figure I.22 : *robot ABB*

I .6 Utilisation des robots

I .6.1 Tâches simples

La grande majorité des robots est utilisée pour des tâches simples et répétitives. Ils sont programmés une fois pour toute au cours de la procédure *d'apprentissage*. - Critères de choix de la solution robotique :

- La tâche est assez simple pour être robotisée.
- Les critères de qualité sur la tâche sont importants.
- Pénibilité de la tâche (peinture, charge lourde, environnement hostile,..)

Exemples

• Robot soudeurs

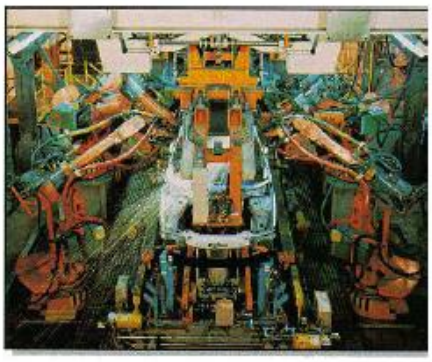


Figure I.23: *Robot soudeurs Par points*

Figure I.24: *Robot soudeurs a l'arc*

I .6.2. Tâches complexes

• Robotique de service

Figure I.25: *Robot pompiste*

Figure I.26: *Robot de construction*

Figure I.27: *robot Computer motion personnes*

Figure I.28 : *robot Assistance aux Handicapées*

I .7. Avenir de la robotique

- Stagnation du nombre de robots utilisés pour des tâches simples.
- Forte croissance du nombre des robots utilisés pour des tâches complexes :
  - robotique de service
  - robotique d'assistance aux manipulations dans la recherche biologique et génétique.

- robotique médicale.

I.7.1 Les robots manipulateurs

Les robots manipulateurs sont actuellement d'une très large utilisation dans les applications

Industrielles et spatiales. Ils sont d'une importance majeure, surtout dans les travaux dangereux fastidieux et monotones [9].

En 1979 l'institut américain de robotique a donné une définition pour un robot manipulateur :

{Un manipulateur multifonctionnel reprogrammable conçu pour déplacer des matériaux, des pièces, des outils ou des appareils spécialisés et ceci à travers des mouvements programmables et variés pour la performance d'une variété de tâches}

Ces manipulateurs sont composés de liaisons rigides interconnectées par le moyen d'articulations, et un organe effecteur se trouvant à l'extrémité de la dernière liaison. Le mouvement de ces liaisons est assuré par des actionneurs et l'état du manipulateur est donné par des mesures issues des capteurs.

Le mouvement désiré du manipulateur est achevé en utilisant un système de contrôle qui fournit des commandes aux actionneurs des articulations dépendant sur la méthodologie de commande implémentée.

La plupart de ces robots manipulateurs ont été conçus d'une manière à maximiser leur raideurs ainsi donc augmenter leur rigidités, afin de minimiser les vibrations de l'organe effecteur pour achever son bon positionnement. Ceci est obtenu en choisissant un matériau lourd pour la conception de manipulateur.

Cependant leur capacité de port des charges est seulement cinq à dix pour cent de leur propre poids.

Vu leur dimensions relativement énormes, ces manipulateurs nécessitent des grands actionneurs et par conséquent une grande consommation d'énergie et une vitesse d'opération qui est généralement lente.

Le nombre des applications de ces manipulateurs a augmenté notamment lors de ces dernières années, par exemple les applications dans l'espace nécessitent les mêmes manipulateurs mais avec des poids légers les rendant ainsi des manipulateurs à liaisons flexibles.

On peut retrouver deux formes de manipulateurs en fonction de la manière dont les liaisons sont connectées : une forme sérielle et une autre parallèle

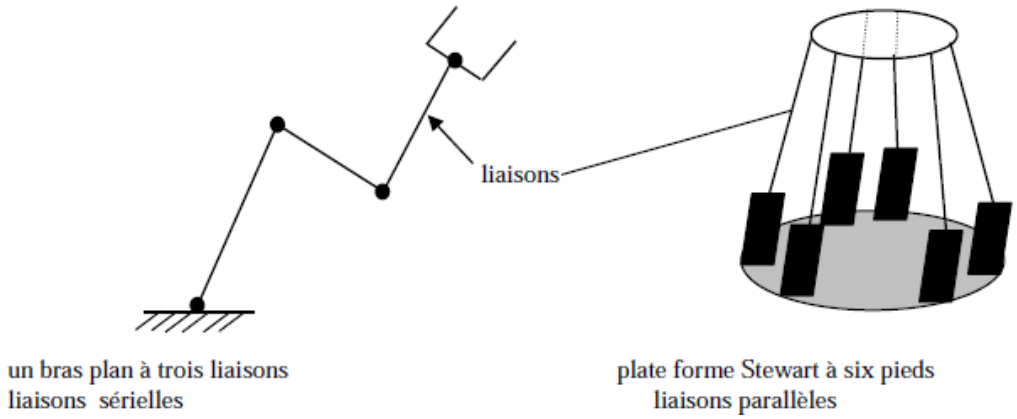


Figure I.29 : *Manipulateurs à liaisons sérielles et parallèles*

Les bras robotiques peuvent être cinématiquement conçus redondants en leur conférant des Degrés de libertés supplémentaires dans l'espace articulaire, ainsi un manipulateur est redondant quand le nombre n de ses articulations est supérieur à la dimension de l'espace de la tâche m.

Un manipulateur redondant est caractérisé par le fait qu'il peut avoir un nombre infini de configurations au niveau articulaire correspondant à plus de positions de l'organe effecteur dans l'espace du travail.

Alors que ce surplus de degrés de libertés complique la programmation et les stratégies de contrôle par contre il augmente considérablement l'utilité du robot.

Mathématiquement, un bras robotique est décrit par ses équations cinématiques et dynamiques, la cinématique d'un bras introduit l'étude des relations entre les positions, vitesses et les Accélérations de ses différentes parties; l'analyse cinématique est nécessaire pour la planification et l'exécution des mouvements désirés du manipulateur aussi bien que par des calculs dynamiques.

Les équations dynamiques d'un bras décrivent son évolution dans le temps en réponse à des forces externes, et des couples agissant sur ses actionneurs. Cependant un système robotique n'est pas seulement un bras manipulateur, en plus du bras, le système renferme aussi une source d'énergie externe, un outillage de l'extrémité du bras, des capteurs externes et internes, des servomécanismes, un ordinateur interface et le contrôleur (**Fig. I.29).**

Le contrôleur d'un robot peut être pris comme le cerveau qui commande les mouvements mécaniques du bras : il est responsable, en se basant sur les modèles cinématiques et dynamiques du bras et les mesures captées, de la génération de directives contrôlant les actionneurs des articulations, nécessaire pour la génération du mouvement désiré.

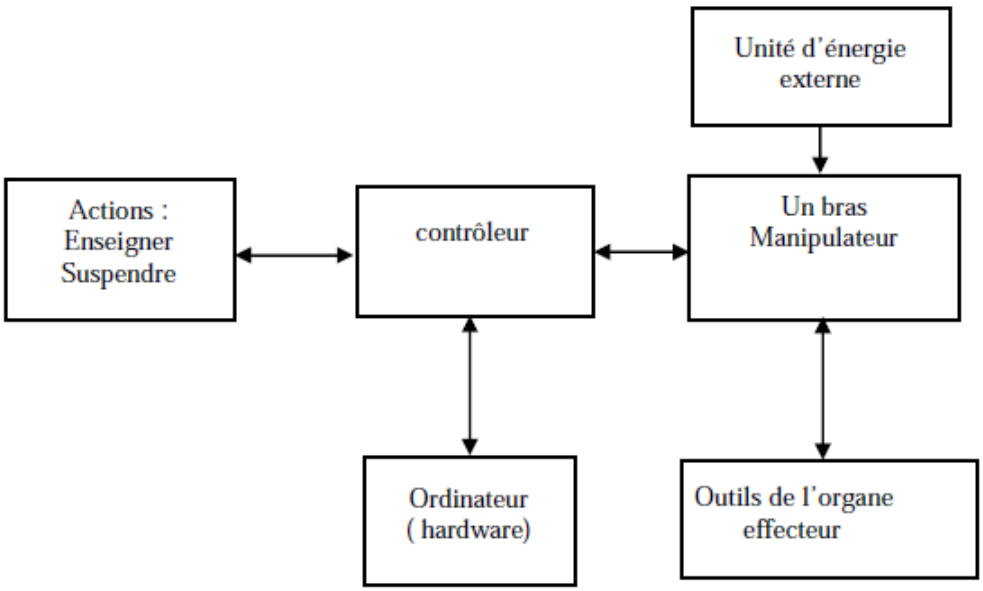


Figure I.30 : *composantes d'un système robotique à un seul bras*

I.7.2 Modélisations des robots manipulateurs

a) Modèle cinématique

Considérons un manipulateur semblable à celui de la figure 2.2, ce manipulateur est composé de deux liaisons rigides de masses m_1 et m_2 de longueurs l_1 et l_2 ou les angles des articulations sont respectivement

θ_1 et θ_2 . Le problème cinématique direct est posé comme ce qui suit :

Etant donné deux angles articulaires θ_1 et $\theta_2$déterminer la position de l'organe effecteur (Coordonnées) et son orientation par rapport au repère de base.

Soit le vecteur indiquant la position et l'orientation de l'organe effecteur dans le repère de base, représenté par $x \in \mathbb{R}^m$ et le vecteur des positions articulaires par $q \in \mathbb{R}^n$ ou x= [x y] et q= [θ_1 θ_2] pour n=m=2

La fonction cinématique directe f d'un robot manipulateur liant q à x :

$$x = f(q) \qquad \text{Eq (I.1)}$$

L'expression de f est obtenue en fixant des repères à chaque liaison et en utilisant les matrices de transformations entre ces repères on peut obtenir les équations cinématiques directes.

Une méthode systématique pour définir les repères des liaisons et obtenir les matrices de transformations est celle proposée par Denavit et Hartenberg et qui est utilisée universellement dans l'analyse des bras manipulateurs [12] [13].

On peut exprimer la position de l'organe effecteur en fonction des angles articulaires [14] :

$$x = I_1 cos\theta_1 + I_2 \cos(\theta_1 + \theta_2) \qquad \text{Eq (I.2)}$$

$$y = I_1 sin\theta_1 + I_2 \sin(\theta_1 + \theta_2) \qquad \text{Eq (I.3)}$$

Et l'orientation du repère outil est donnée :

$$i_1 i_0 = \cos(\theta_1 + \theta_2) \quad i_2 i_0 = -\sin(\theta_1 + \theta_2)$$
$$j_2 i_0 = \sin(\theta_1 + \theta_2) \quad j_2 i_0 = \cos(\theta_1 + \theta_2)$$ Eq (I.4)

$$\begin{bmatrix} i_2 i_0 & j_2 i_0 \\ i_2 j_0 & j_2 j_0 \end{bmatrix} = \begin{bmatrix} cos(\theta_1 + \theta_2) & -sin(\theta_1 + \theta_2) \\ sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \end{bmatrix}$$ Eq (I.5)

On remarque que *f* est une fonction non linéaire des positions articulaires *q*, pour tout robot manipulateur la fonction cinématique directe *f* existe toujours et peut être calculée sous une forme simple, notons que la solution cinématique est unique autrement dit pour tout vecteur *q* donné il existe une position de l'organe effecteur unique *x*

La procédure inverse c'est à dire le calcul des positions articulaires pour un vecteur position donné de l'organe effecteur, appelée le modèle cinématique inverse qui s'énonce comme suit :

Etant donné une position et une orientation de l'organe effecteur d'un manipulateur, on détermine un ensemble d'angles articulaires qui permettent d'obtenir cette position et orientation désirées.

La solution au problème cinématique inverse est très importante dans le but de transformer les spécifications du mouvement de l'organe effecteur qui sont donnés dans l'espace opérationnel en leurs spécifications correspondantes dans l'espace articulaire du mouvement.

Contrairement à la fonction cinématique directe *f*, la fonction cinématique inverse pour la plupart des manipulateurs n'est pas facile à avoir sous forme analytique précise : le plus importantes que la solution cinématique inverse pour une position donnée de l'organe effecteur n'est pas unique .Du fait de la

Structure des équations précédentes on voit qu'on peut trouver deux solutions pour une position donnée qui sont appelées solutions pour coude haut et coude bas **[8].**

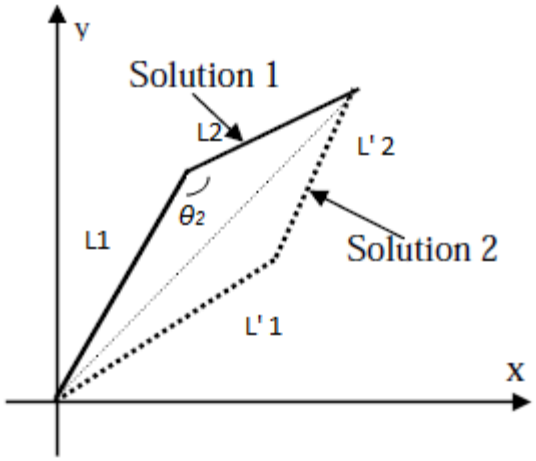


Figure I.31 : *Solutions multiples pour le modèle cinématique inverse d'un manipulateur*

$$cos\theta_2 = \frac{x^2 + y^2 - I_1^2 - I_2^2}{2I_1I_2} = D \qquad \text{Eq (I.7)}$$

Ceci implique

$$\theta_2 = cos^{-1}\, D \qquad \text{Eq (I.8)}$$

Par conséquent on retrouve

$$\theta_2 = tg^{-1}\left(\pm\frac{\sqrt{1 - D^2}}{D}\right) \qquad \text{Eq (I.9)}$$

Le signe ' ± ' signifie qu'on a deux solutions différentes, on obtient ensuite :

$$\theta_1 = tg^{-1}\left(\frac{y}{x}\right) - tg^{-1}\left(\frac{I_2 sin\theta_2}{I_1 + I_2 sin\theta_2}\right) \qquad \text{Eq (I.10)}$$

Les deux solutions pour le modèle cinématique inverse sont :

$$\begin{cases} \theta_1 = tg^{-1}\left(\frac{y}{x}\right) - tg^{-1}\left(\frac{I_2 sin\theta_2}{I_1 + I_2 sin\theta_2}\right) \\ \theta_2 = tg^{-1}\left(\pm\frac{\sqrt{1-D^2}}{D}\right) \end{cases} \qquad \text{Eq (I.11)}$$

Il peut exister plusieurs ou une infinité de vecteurs de positions articulaires correspondant à un vecteur *x* donné, cependant la résolution du problème cinématique inverse est fondamentale pour la commande des robots manipulateurs, depuis qu'une tache de manipulation est spécifiée sous la forme de positions, vitesses et accélérations de l'organe effecteur au moment ou on a besoin de spécifier les entrées de commandes au niveau articulaire.

La différentiation de l'équation (I.1) génère la relation de vitesse suivante :

$$\dot{X} = J\dot{q} \qquad J(q) \triangleq \frac{\partial f}{\partial q} \epsilon R^{m \times n} \qquad \text{Eq (I.12)}$$

Où $J(q)$ est la matrice du Jacobien de la fonction cinématique directe $f(q)$.

Le calcul de la solution cinématique inverse au niveau de la vitesse requiert le calcul du vecteur

vitesse articulaire $\dot{q}$ pour une vitesse donnée $\dot{x}$ de l'organe effecteur à une configuration articulaire *q*

L'équation (I.4) donne une définition analytique de la matrice du Jacobien J, du point de vue calcul il est très efficace de construire cette matrice utilisant les vecteurs définis par la notation de Denavit-Hartenberg [6] [7]

Similairement les équations cinématiques du second ordre reliant les accélérations de l'organe effecteur et celles des articulations peuvent être obtenues en différentiant l'équation (I.12).

$$\ddot{x} = J\ddot{q} + \dot{J}\dot{q} \qquad \text{Eq (I.13)}$$

II.3 Commande des bras manipulateurs

La résolution du problème de la commande des robots manipulateurs nécessite la détermination d'un ensemble d'entrées articulaires (les couples □) qui résulte par le suivi de l'organe effecteur d'une trajectoire désirée, spécifiée typiquement par des séquences de positions et de vecteurs d'orientation de l'organe effecteur *x* ou par une trajectoire continue.

Plusieurs types de commandes ont été étudiées pour les robots manipulateurs, la méthode la plus simple et qui reste toujours employée pour les manipulateurs industriels est la commande articulaire indépendante ou chaque articulation du manipulateur est commandée comme un système à une seule entrée et une seule sortie (S.I.S.O) [5]

La stratégie de cette commande est que chaque actionneur d'une articulation est contrôlé

Indépendamment : tous les effets de couplage entre les deux articulations sont ignorées ou traitées comme des perturbations. Cette commande à donné des résultats satisfaisants pour les simples déplacements, mais n'est pas convenable pour les déplacements rapides et ceux avec une large variation.

La méthode de la commande articulaire indépendante peut être considérée comme une forme simplifiée de la méthode du couple calculé qui est la technique commune la plus utilisée pour les robots manipulateurs, la plupart des méthodes de commandes des manipulateurs peuvent être considérés comme des cas spéciaux de cette technique.

La méthode du couple calculé en elle même est une application de la technique de linéarisation du 'Feed-Back' pour les systèmes non linéaires.

b) Méthode de la commande par couple calculé

La stratégie de la commande par couple calculé peut être considérée comme un contrôleur en deux parties, une section étant basée modèle alors que l'autre étant la portion de la loi de

Servocommande **(Fig. I.31) [5].**

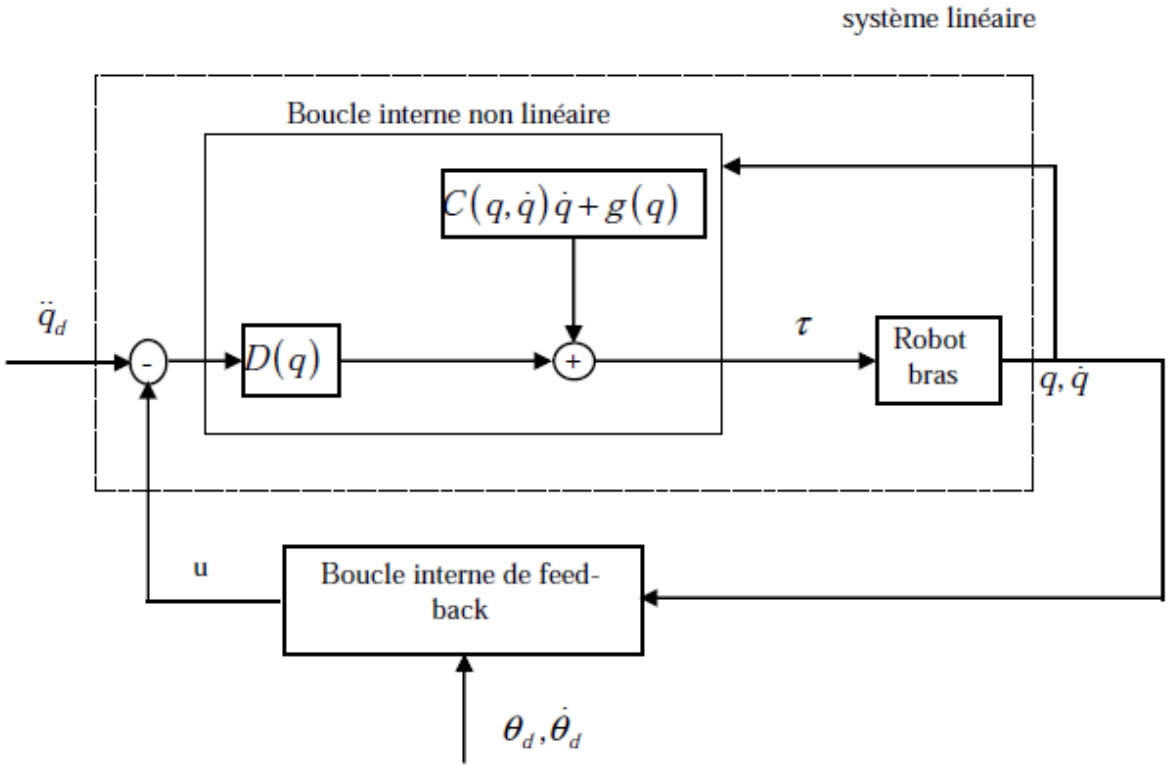


Figure I.32 : *Schéma de la commande par couple calculé*

Considérons un manipulateur dont son organe effecteur est libre dans l'espace, et ayant comme modèle dynamique

$$\tau = [D(q)]\ddot{q} + C(q,\dot{q})\dot{q} + g(q) \qquad \text{Eq (I.14)}$$

La portion basée modèle de la loi de commande est

$$\tau = [D(q)](\ddot{q}_d - u) + C(q, \dot{q})\dot{q} + g(q) \qquad \text{Eq (I.15)}$$

Où $\ddot{q}_d$ est le vecteur d'accélération articulaire désirée et u est le vecteur d'entrée de la commande qui est déterminé par une loi de servocommande. En général une loi de commande de type P.D est utilisée pour les robots manipulateurs.

Ainsi si $q_d \; et \; \dot{q}_d$ et sont les vecteurs des positions et vitesses articulaires désirées avec q et $\dot{q}$ les vecteurs des positions et des vitesses articulaires actuelles (ou mesurées).

Les erreurs dans l'espace articulaire sont calculées comme suit :

$$e = q_d - q \quad \text{et} \quad \dot{e} = \dot{q}_d - \dot{q} \qquad \text{Eq (I.16)}$$

Ensuite la loi de commande peut être choisie comme suit :

$$u = -K_v\dot{e} - K_p e \qquad \text{Eq (I.17)}$$

Faisant une substitution à partir de l'équation (I.17) dans l'équation (I.15), l'équation de l'erreur pour le système en boucle fermée est obtenue :

$$\ddot{e} + K_v\dot{e} + K_p e = 0 \qquad \text{Eq (I.18)}$$

En choisissant les matrices de gain K_v et K_p comme des matrices diagonales avec des valeurs positives le long de la diagonale, cette erreur du système peut

être rendue asymptotiquement stable. Il est important de noter que malgré la

Sélection des matrices diagonales de gain, il en résulte un découplage de la commande au niveau de la boucle externe mais ceci n'implique pas une stratégie de commande articulaire découplée.

Parce que la multiplication par la matrice d'inertie et l'addition de termes non linéaires dans la loi basée modèle laisse la loi de commande u affecter toutes les articulations.

Pour calculer l'entrée couple de n'importe quelle articulation, les positions et vitesses des autres articulations sont nécessaires.

Différents choix de la loi de servocommande résulte dans la technique de commande de base décrite précédemment, (on peut avoir la commande articulaire indépendante quand D(q)=I et $C(q,\dot{q})\dot{q} + g(q) = -\ddot{q}_d$

L'idée de base est de supprimer les non-linéarités dans le modèle (utilisant la loi basée modèle) et ensuite traiter le système comme un système linéaire. Le grand inconvénient de ce schéma de commande est que les paramètres et la structure du système doivent être connus afin de calculer l'équation (I.15), toutefois quand les erreurs dans les paramètres ne sont pas aussi larges ; ce schéma de commande donne une grande performance satisfaisante [5].

c) La méthode de commande P.D

Soit un manipulateur ayant pour équation du mouvement :

$$[D(q)]\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau \qquad \text{Eq (I.19)}$$

L'objectif de la commande est de déterminer les n composantes des couples agissant sur des articulations rotoïdes permettant de déplacer l'organe effecteur à une position finale donnée

$$q_d = [q_{d1} \quad q_{d2}]^T \qquad \text{Eq (I.20)}$$

Afin d'achever une position désirée, on peut utiliser un contrôleur à action proportionnelle et dérivée P.D

La loi de commande est basée sur les mesures locales des erreurs de positions $\tilde{q}_j = q_J - q_{dJ}$et la vitesse articulaire $\dot{q}_i$.

Premièrement considérons le cas ou $g\left(q\right) = 0$ (c'est-à-dire qu'il n y a pas de forces de gravitation), alors la loi de commande est donnée par :

$\tau_J = -K_{pJ}\tilde{q}_J - K_{dJ}\dot{q}$ ou $K_{pJ} et\ K_{dJ}$ sont des constantes strictement positives.

Pour montrer que cette loi de commande est stable et qu'elle assure une erreur nulle en régime permanent, considérons la fonction de Lyapounov candidate :

$$V = \frac{1}{2}\left[\dot{q}^T D(q)\dot{q} + \tilde{q}^T K_p \tilde{q}\right] \qquad \text{Eq (I.21)}$$

On peut réécrire la loi de conservation de l'énergie sous la forme suivante :

$$\frac{1}{2}\frac{d}{dt}(\dot{q}^T D(q)\dot{q}) = \dot{q}\tau \qquad \text{Eq (I.22)}$$

Le terme de droite représente la puissance d'entrée fournie par les actionneurs et le terme de gauche représente la dérivée de l'énergie cinétique.

On a $\tau_J = -K_{pJ}\tilde{q}_J - K_{dJ}\dot{q}$ ou K_p et K_d sont des matrices symétriques définies positives constantes (Usuellement c'est des matrices diagonales).

On peut écrire la dérivée temporelle de V comme ce qui suit :

$$\dot{V} = \dot{q}^T\left[\tau + K_p\tilde{q}\right] \qquad \text{Eq (I.23)}$$

Utilisant la relation :

$$\tau_J = -K_{pJ}\tilde{q}_J - K_{dJ}\dot{q} \qquad \text{Eq (I.24)}$$

On aura alors

$$\dot{V} = -\dot{q}^T \mathrm{K_d q} \dot{\leq} 0 \qquad \text{Eq (I.25)}$$

En considérant $V = 0$ ceci va engendrer

$$\dot{q} = 0 \Rightarrow \ddot{q} = 0 \Rightarrow \ddot{q} = D^{-1}K_p\tilde{q} \Rightarrow \tilde{q} = 0 \qquad \text{Eq (I.26)}$$

Alors dans ce cas la trajectoire du système converge vers l'état désiré.

Supposons maintenant que $g(q)$ n'est pas égal à zéro, alors la loi de commande peut s'écrire sous la forme suivante :

$$\tau_J = -K_{pJ}\tilde{q}_J - K_{dJ}\dot{q} + g(q) \qquad \text{Eq (I.27)}$$

Dans ce cas la loi de commande supprime les effets des termes de gravités, par conséquent cette loi de commande exige le calcul à chaque instant de $g(q)$ à partir des équations du Lagrangien **[8].**

Conclusion

Dans ce chapitre on a donné la description sur le domaine de la robotique en mettant l'accent sur les domaines applicatifs des robots industriels. Ensuite la classification des robots est abordée et quelques notions sur les bras manipulateurs rigides, et sur les modélisations utilisées et ainsi que quelques lois de commandes employées dans le domaine de la robotique.

II.1 Introduction

Les cartes Arduino sont conçues pour réaliser des prototypes et des maquettes de cartes électroniques pour l'informatique embarquée.

Ces cartes permettent un accès simple et peu couteux à l'informatique embarquée. De plus, elles sont entièrement libres de droit, autant sur l'aspect du code source (Open Source) que sur l'aspect matériel (Open Hardware). Ainsi, il est possible de refaire sa propre carte Arduino dans le but de l'améliorer ou d'enlever des fonctionnalités inutiles au projet.

Le langage Arduino se distingue des langages utilisés dans l'industrie de l'informatique embarquée de par sa simplicité. En effet, beaucoup de librairies et de fonctionnalités de base occulte certains aspects de la programmation de logiciel embarquée afin de gagner en simplicité. Cela en fait un langage parfait pour réaliser des prototypes ou des petites applications dans le cadre de hobby.

Les possibilités des cartes Arduino sont énormes, un grand nombre d'application ont déjà été réalisée et testées par bon nombre d'internautes.

II.2 Définition et Historique

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

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

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

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

II.3 La composition de carte Arduino

En la carte Arduino se compose de plusieurs composants mais en principe se compose de 8 parties principales comme la figure nous montre

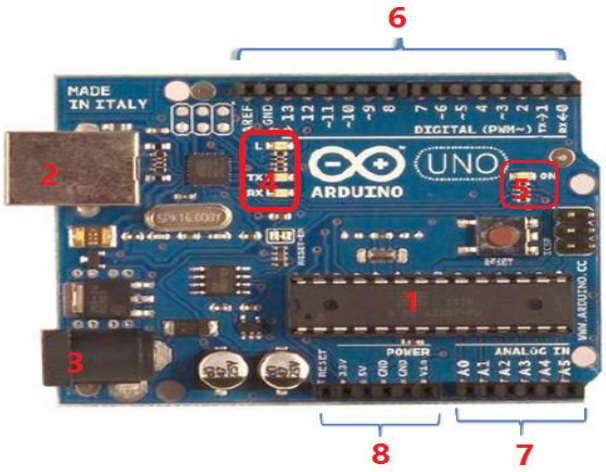


Figure II.1: *les composants principaux de la carte Arduino Uno.*

II.3. 1. Le microcontrôleur :

Voilà le cerveau de notre carte (en **1**). Le microcontrôleur est un composant électronique programmable, C'est lui qui va recevoir le programme que nous aurons créé et qui va le stocker dans sa mémoire puis l'exécuter. Grâce à ce programme, il va savoir faire des choses, qui peuvent être : faire clignoter une LED, afficher des caractères sur un écran, envoyer des données à un ordinateur.

Figure II.2: *Le microcontrôleur*.

II.3.2 PORT USB

Ce port USB nous permettre de faire connecter notre carte ARDUINO avec un ordinateur. Pour changer les donner entre la carte et l'ordinateur.

Figure II.3: *Porte USB de la carte Arduino Uno*

II.3.3. Alimentation :

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

Figure II.4: *Entrée de l'alimentation externe pour la carte.*

II.3.4. Visualisation

Les trois "points blancs" entourés en rouge (**4**) sont en fait des LED dont la taille est de l'ordre du Millimètre. Ces LED servent à deux choses :

- Celle tout en haut du cadre : elle est connectée à une broche du microcontrôleur et va servir pour tester le matériel.

Note : quand on branche la carte au PC, elle clignote quelques secondes.

- Les deux LED du bas du cadre : servent à visualiser l'activité sur la voie série (une pour l'émission et l'autre pour la réception). Le téléchargement du programme dans le microcontrôleur se faisant par cette voie, on peut les voir clignoter lors du chargement.

Figure II.5: *Les LED de la carte Arduino Uno*

II.3.5 .Tester le matériel

Avant de commencer à programmer, il faut, avant toutes choses, tester le bon fonctionnement de la carte. Car ce serait idiot de programmer la carte et chercher les erreurs dans le programme alors que le problème vient de la carte, nous allons tester notre matériel en chargeant un programme qui fonctionne dans la carte.

Et parfois la carte va y avoir un défaut dans la composition de la carte, pour tester ça il suffit juste de broncher notre carte à l'alimentation et on va voir une LED verte qui clignoté sur la carte ce qui signifier que la carte est bonne.

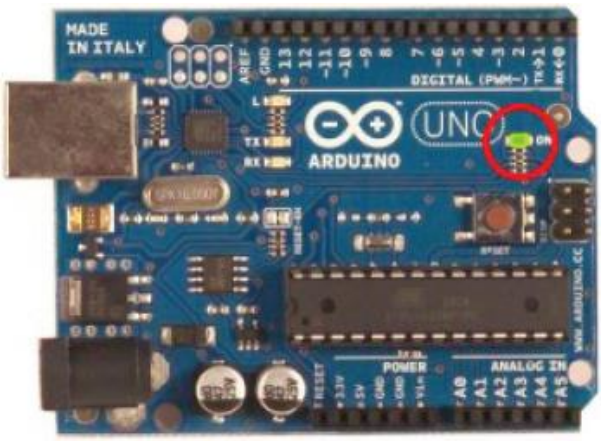


Figure II.6: *la carte Arduino connecter et alimenter*.

II.3.6. Les entrées-sorties

- **Broches analogiques**

La carte Uno dispose de 6 entrées analogiques (numérotées de 0 à 5), chacune pouvant fournir une mesure d'une résolution de 10 bits (c.à.d. sur 1024 niveaux soit de 0 à 1023) à l'aide de la très utile fonction analogread () du langage Arduino. Par défaut, ces broches mesurent entre le 0V (valeur 0) et le 5V (valeur 1023), mais il est possible de modifier la référence supérieure de la plage de mesure en utilisant la broche AREF et l'instruction analogreference () du langage Arduino.

Note : les broches analogiques peuvent être utilisées en tant que broches numériques : elles sont numérotées en tant que broches numériques de 14 à 19.

Figure II.7: *Les entrées analogiques*.

Autres broches: Il y a deux autres broches disponibles sur la carte :

- AREF : Tension de référence pour les entrées analogiques (si différent du 5V).
- Reset : Mettre cette broche au niveau BAS entraîne la réinitialisation (= le redémarrage) du microcontrôleur. Typiquement, cette broche est utilisée pour ajouter un bouton de réinitialisation sur le circuit qui bloque celui présent sur la carte.

II.4 Les Caractéristiques de la carte Arduino

- Micro contrôleur : ATmega328
- Tension d'alimentation interne = 5V
- Tension d'alimentation (recommandée)= 7 à 12V, limites =6 à 20 V
- Entrées/sorties numériques : 14 dont 6 sorties PWM
- Entrées analogiques = 6

- Courant max par broches E/S = 40 mA
- Courant max sur sortie 3,3V = 50mA
- Mémoire Flash 32 KB dont 0.5 KB utilisée par le boot loader
- Mémoire SRAM 2 KB
- Mémoire EEPROM 1 KB
- Fréquence horloge = 16 MHz
- Dimensions = 68.6mm x 53.3mm

II.5 Les types de la carte Arduino

Des cartes Arduino ils en existent beaucoup, Peut-être une centaine toutes différentes, nous avions vous montrer les quelles on peut utiliser et celle que nous utiliserions dans ce mémoire.

Il existe beaucoup type des cartes Arduino mais on peut classer les cartes Arduino en trois grandes familles:

- Les cartes Arduino officielles ou classique, compatible hardware et software avec le "forme factor" et l'ide Arduino.

- Les cartes Arduino « compatibles » qui ne sont pas fabriqués par smart projects, mais qui sont totalement compatibles avec les Arduino officielles.
- Les cartes dérivées d'Arduino, compatible avec les Shields Arduino classique (mais pas avec l'ide Arduino de base).

II.6 Les différentes cartes

- **La carte Uno et Duemilanov**

Nous choisirons d'utiliser la carte portant le nom de « Uno » ou « Duemilanove ». Ces deux versions sont presque identiques.

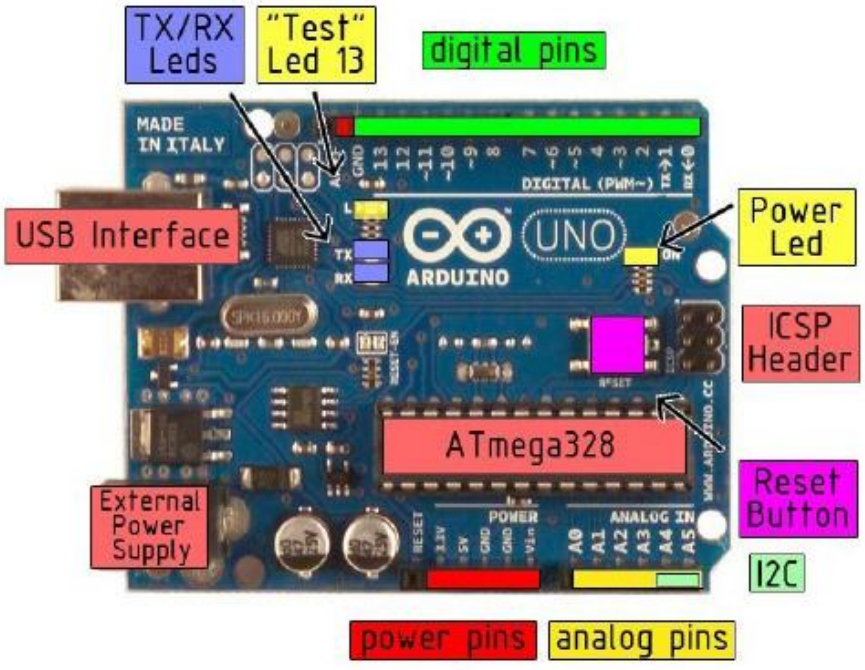


Figure II.8 *: Carte Arduino "Uno"*

- **La carte *Mega***

La carte Arduino Mega est une autre carte qui offre toutes les fonctionnalités des précédentes, mais avec des options en plus.

On retrouve notamment un nombre d’entrées et de sorties plus importantes ainsi que plusieurs liaisons séries.

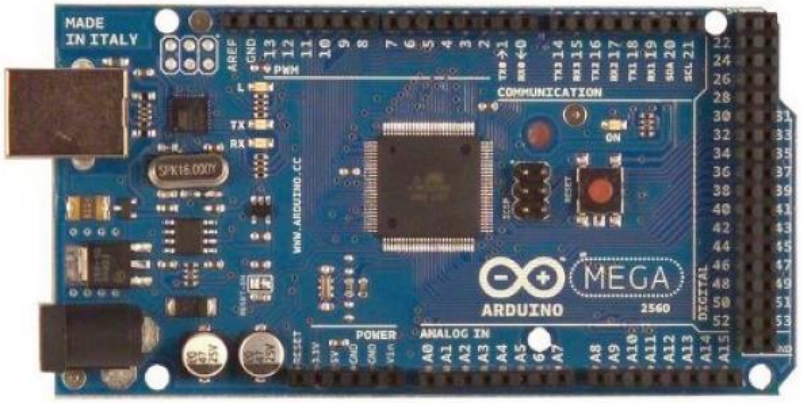


Figure II.9 : *Carte Arduino "Mega"*

II.7 Applications

Le système Arduino, nous donne la possibilité d'allier les performances de la programmation à celles de l'électronique. Plus précisément, nous allons programmer des systèmes électroniques. Le gros avantage de l'électronique programmée c'est qu'elle simplifie grandement les schémas électroniques et par conséquent, le coût de la réalisation, mais aussi la charge de travail à la conception d'une carte électronique.

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

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

Avec Arduino, nous allons faire des systèmes électroniques tels qu'une bougie électronique, une calculatrice simplifiée, un synthétiseur, etc. Tous ces systèmes

Seront conçus avec pour base une carte Arduino et un panel assez large de composants électroniques

II.8 Le logiciel Arduino

II.8.1 Présentation du logiciel

La page principale de logiciel Arduino se compose principalement de quatre zones voire la figure :

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

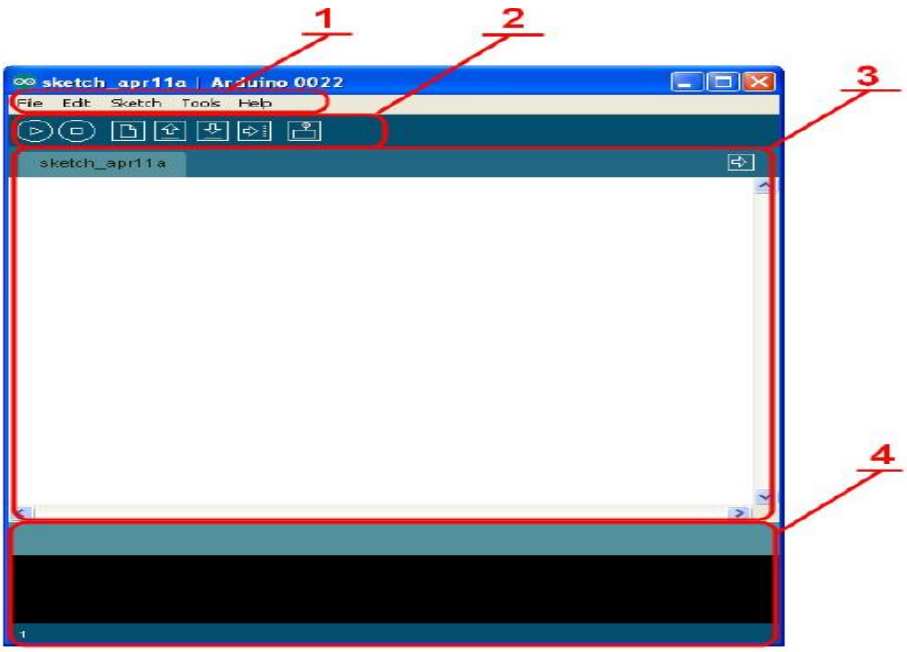


Figure II.10 : *l'interface de logiciel Arduino*.

➢ **Le menu FILE :**

C'est principalement ce menu que l'on va utiliser le plus. Il dispose d'un certain nombre de choses qui vont nous être très utiles :

- New (nouveau) : va permettre de créer un nouveau programme. Quand on appuie sur ce bouton, une nouvelle fenêtre, identique à celle-ci, s'affiche à l'écran
- Open... (ouvrir) : avec cette commande, nous allons pouvoir ouvrir un programme existant
- Save / Save as... (enregistrer / enregistrer sous...) : enregistre le document en cours / demande où enregistrer le document en cours
- Examples (exemples) : ceci est important, toute une liste se déroule pour afficher les noms d'exemples de programmes existants ; avec ça, vous pourrez vous aider pour créer vos propres programmes

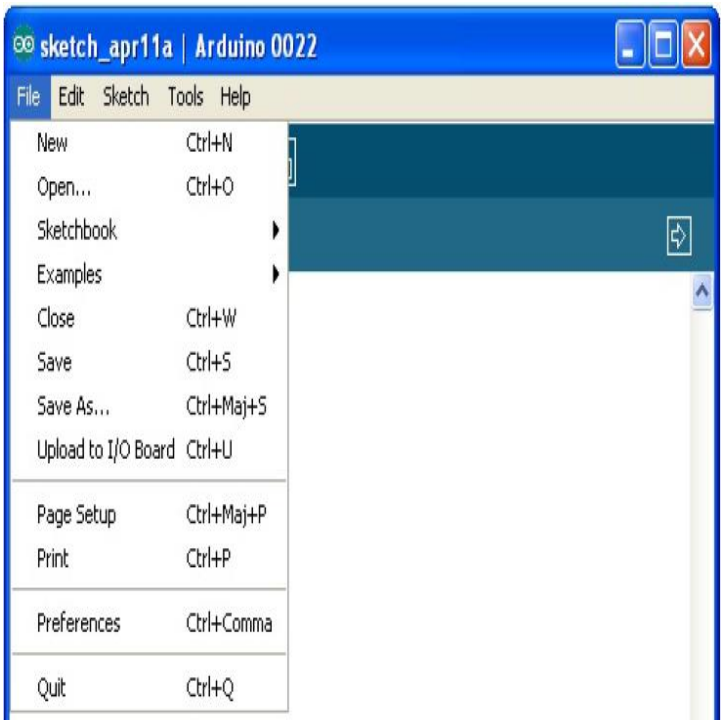


Figure II.11: *Le menu File*

- **Les boutons**

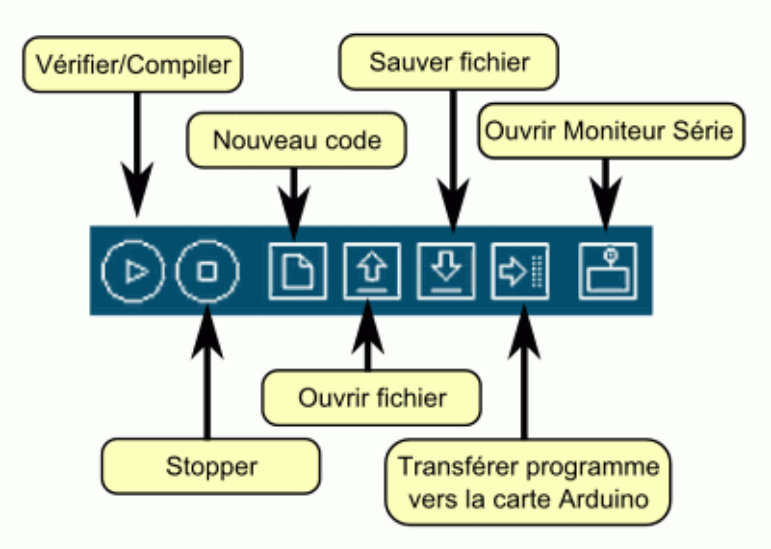


Figure II.12 : *Détail de la barre de boutons*

A noter : le logiciel comprends aussi un moniteur série (équivalent à hyperterminal) qui permet de d'afficher des messages textes émis par la carte Arduino et d'envoyer des caractères vers la carte Arduino (en phase de fonctionnement).

- **Choisir la carte Arduino Uno**

Le choix de la carte est important car chaque carte a des spécifications par rapport à une outre, et parfois il va y avoir une incompatibilité quand on va lancer le programme dans la carte et pour notre mémoire on va utiliser la carte Arduino Uno et ça se fait avec le clique sur le menu Tools /Board :''Arduino Uno'' /Arduino Uno

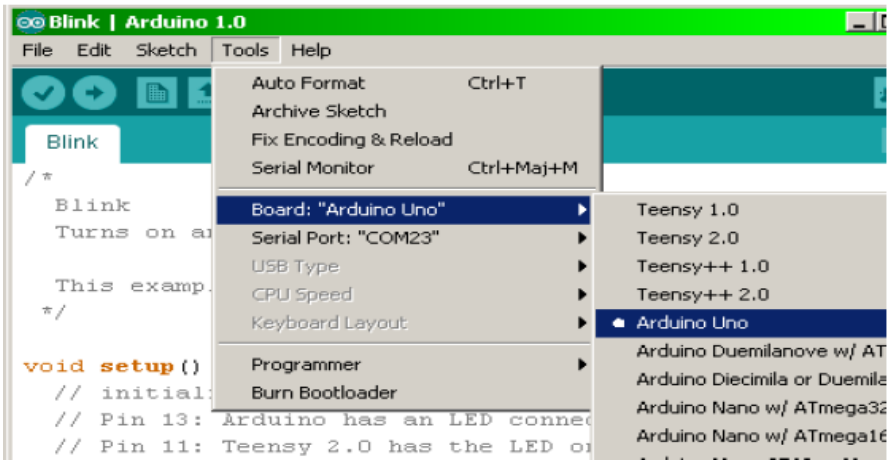


Figure II.13: *la sélection de la carte Arduino Uno.*

II.9 La programmation

Le programme c'est l'ensemble des variables et des constantes et des instructions bien organiser dédier à réaliser une tâche bien définie et avant de réaliser le programme on va voir une vision brève sur l'ensemble des instructions, les variables, les constants, les structures, les fonctions utiliser dans le langage Arduino.

- **Structure**

Fonctions de base	**Structures de contrôle**	**Syntaxe de base**
Ces deux fonctions sont Obligatoires dans tout Programme en langage Arduino : • void setup () • voidloop()	• i f • if...else • for • Switch case • while • do... While • break • continue • return • goto	• ; (point-virgule) • {} (accolades) • / / (commentaire sur une Ligne) • /* * / (commentaire sur Plusieurs lignes) • #define • #include
Opérateurs arithmétiques • = (égalité) • + (addition) • - (soustraction) • * (multiplication) • / (division) • % (modulo)	**Opérateurs de comparaison** • == (égal à) • != (différent de) • < (inférieur à) • > (supérieur à) • <= (inférieur ou égal à) • >= (supérieur ou égal à)	**Opérateurs booléens** • && (ET booléen) • | | (OU booléen) • ! (NON booléen)
Pointeurs • * pointeur • & pointeur Voir également : • Manipulation des Ports	**Opérateurs bit à bit** • & (ET bit à bit) • |(OU bit à bit) • ^(OU EXCLUSIF bit à bit) • ˜(NON bit à bit) • << (décalage à gauche) • >> (décalage à droite)	

- **Variables et constantes**

Les variables sont des expressions que vous pouvez utiliser dans les programmes pour stocker des valeurs, telles que la tension de sortie d'un capteur présente sur une broche analogique.

Constantes prédéfinies	**Conversion des types De données**
Les constantes prédéfinies du langage Arduino sont des valeurs particulières ayant une signification spécifique.	• char () • byte () • int () • long () • float () • word ()
• HIGH | LOW • INPUT | OUTPUT • true | false A ajouter : constantes Décimales prédéfinies	**Portée des variables et Qualificateurs** • Portée des variables • static • volatile • cons t
Expressions numériques • Expressions numériques entières • Expressions numériques à virgule	**Utilitaires** • sizeof () (opérateur Sizeof) **Référence** • Code ASCI I

➢ Fonctions

Les fonctions sont des petites programmes enregistrer sur le toolbox de dans le but les utiliser directement dans le besoin.

Entrées/Sorties Numériques

• pinmode (broche, mode)
• digitalwrite (broche, valeur)
• intdigitalread(broche)

Entrées analogiques

• intanalogread (broche)
• analogreference (type)

Sorties "analogiques" (génération d'impulsion)

• analogwrite (broche, Valeur) – PWM

Entrées/Sorties Avancées

• tone ()
• notone ()
• shiftout(broche, Brochehorloge, ordrebit, valeur)
• unsigned long Pulsein (broche, valeur)

Communication

• Seria l

Temps

• unsigned long millis ()
• unsigned long micros
• delay (ms)
• delaymicroseconds (us)

Math

• min (x, y)
• max (x, y)
• abs (x)
• constrain (x, a, b)
• map (valeur, tolow, Fromhigh, tolow, Tohigh)
• pow (base, exposant)
• sq (x)
• sqr t(x)

Pour davantage de fonctions mathématiques, voir aussi la librairie math.h : log, log10, Asin, atan, acos, etc...

Nombres randomisés (hasard)

• randomseed (seed)
• long random(max)
• long random (min, max)

Trigonométrie

• sin (rad)
• cos (rad)
• tan (rad)

Bits et Octets

• lowbyte ()
• highbyte ()
• bitread ()
• bitwrite ()
• bitse t ()
• bitclear ()
• bi t ()

Interruptions Externes

• attachinterrupt(interrupti On, fonction, mode)
• detachinterrup (interruption)

Interruptions

• interrupts ()
• nointerrupts ()

Voir également la librairie Interrupt.h.

II.10 Les Avantages des cartes Arduino

• Pas cher

• Environnement de programmation clair et simple.

• Multiplateforme : tourne sous Windows, Macintosh et Linux.

• Nombreuses librairies disponibles avec diverses fonctions implémentées.

• Logiciel et matériel open source et extensible.

• Nombreux conseils, tutoriaux et exemples en ligne (forums, site perso etc...)

• Existence de « shield » (boucliers en français) : ce sont des cartes supplémentaires qui se connectent sur le module Arduino pour augmenter les possibilités comme par exemple : afficheur graphique couleur, interface et hernet, GPS, etc...

Par sa simplicité d'utilisation, Arduino est utilisé dans beaucoup d'applications comme l'électronique industrielle et embarquée, le modélisme, la domotique mais aussi dans des domaines différents comme l'art contemporain ou le spectacle

II.11 Conclusion

On peut conclure sur le fait que les cartes Arduino sont un puissant outil de prototypage pour les cartes électroniques. Mais aussi, elles permettent un accès facile et intuitif à l'informatique embarqué. On pourra ainsi enrichir tout ces projets d'un microcontrôleur pour leurs donner une plus value importante.

III.1 Introduction

Ce chapitre traitera de la communication hard et soft pour le contrôle des sens de rotations de deux moteurs à courant continu, correspondant aux articulations d'un dispositif manipulateur destiné à la réalisation d'une certaine opération (***figure III.1).***

L'objectif, étant de contraindre la structure manipulatrice à la modification de sa position au travers d'un changement de coordonnées, imposé via une interface, par un utilisateur. Comme, il est possible de transmettre des instructions sous forme de signaux numériques, dans un langage approprié, vers une carte (Arduino Uno dans notre cas), un programmeur pourra stocker des données afin d'imposer un protocole adéquat de communication.

III.2 Description du Bras manipulateur et schéma synoptique

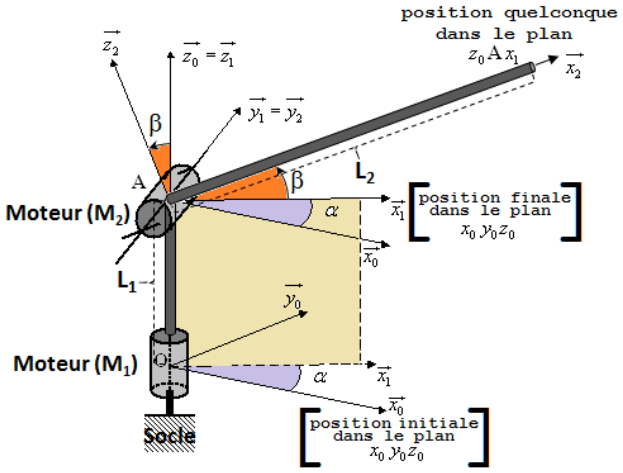


Figure III.1 : *Description du bras manipulateur*

Le système est constitué de deux bras métalliques de longueurs$\mathbf{L_1}$et $\mathbf{L_2}$, caractérisant l'ossature du robot manipulateur (Figure I.1).Destiné à être programmé, pour réaliser des fonctions similaires à un bras humain, il se compose deux moteurs M_1 et M_2constituant les deux articulations du bras. Les liens de ce manipulateur sont représentés par des axes liés à des repères $Ox_oy_oz_o$et $Ax_1y_1z_1$ considérés en rotation l'un par rapport à l'autre, permettant le mouvement nécessaire au

Déplacement recherché. Il peut être autonome ou contrôlé manuellement et peut être utilisé pour Effectuer une variété de tâches avec une grande précision. Comme les bras manipulateurs peuvent être fixes ou mobiles, chaque moteurs constitué d'un jeu de réducteur-encodeur, doit être géré à travers une carte de puissance via un pont H, par un microcontrôleur de type Arduino Uno permettant de piloter le dispositif. Cet ensemble sera gérer selon le schéma synoptique de la figure III.2

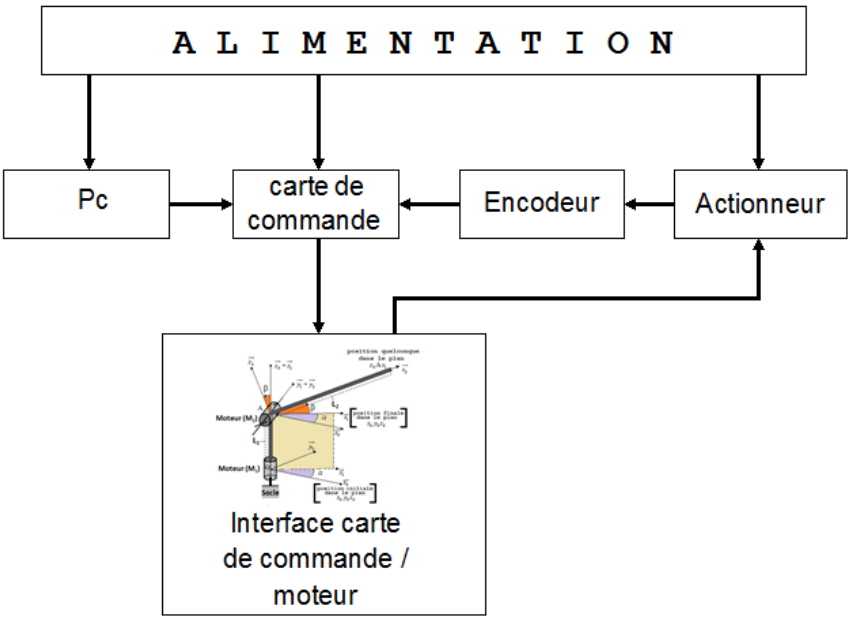


Figure III.2 : *Schéma synoptique*

III.3 Développement de chaque bloc

III.3.1 Bloc alimentation

Le dispositif nécessitant une alimentation permettant de fournir l'énergie nécessaire aux différents blocs, doit débiter des tensions de 5, 6, 7 et 12 V. Les 5V nécessaires au microcontrôleur peuvent être modifiés selon les circonstances d'utilisation. En effet, la carte pourrait être alimentée en 5 V par le port USB ou en mais 6V, si une batterie est utilisée et enfin, entre 7V et 12V si une alimentation externe est employée. Cette tension devant être continue pourrait être, par exemple, fournie par

une pile 9V, associé à un régulateur charger de la réduire à 5V pour un fonctionnement correct de la carte.

III.3.2 Carte de commande

Le microcontrôleur est l'élément "intelligent" du système. En effet, il est le convertisseur d'instructions informatiques, introduites par l'utilisateur pour assurer un tant soit peu, l'autonomie du robot. L'intérêt de l'utilisation de l'environnement Arduino Uno est d'assurer une aisance de mise en œuvre d'instructions bien définies. Arduino fournit un environnement de développement s'appuyant sur des outils open source.

Le chargement du programme dans la mémoire du microcontrôleur se fait par port USB. En outre, des bibliothèques de fonctions sont également fournies pour l'exploitation d'entrées-sorties courantes : gestion des E/S TOR, gestion des convertisseurs ADC, génération de signaux PWM, exploitation de bus TWI/I2C, exploitation de servomoteurs ...etc.

III.3.3 **Bloc Interface carte de commande/moteur**

III.3.3.1 Le pont en H

Afin de faire tourner les moteurs DC dans les 2 sens, un pont H est nécessaire, en l'occurrence le L293D, pour les contrôler. En effet, ce composant fonctionne sous des tensions de 4.5V à 36V et est capable de délivrer 600 mA par canaux (dans notre cas cela fera 1,2A par moteur puisque nous utiliserons les demi ponts par paire pour tourner dans les deux sens).

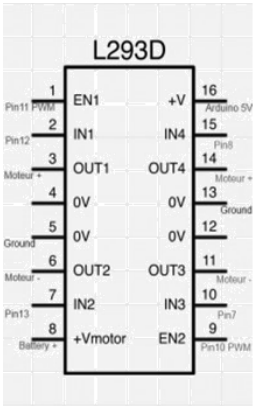


Figure (a)

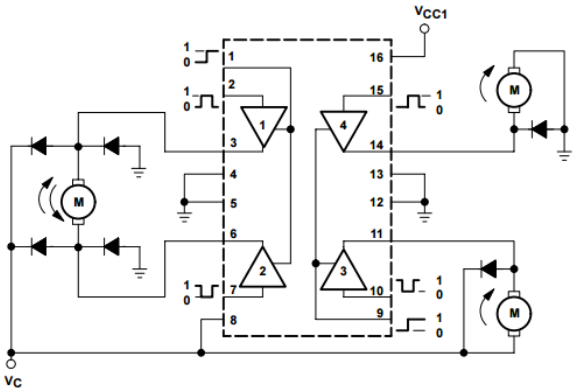


Figure (b)

Figure III.3 : *Pont H (L293D) figure (a) : vue externe, figure (b) : vue interne*

Ce composant existant en deux versions, le L293 et le L293D, le courant de pic toléré peut aller jusqu'à 1,2A par canal, soit 2,4A dans notre cas. La différence entre les deux versions de ponts est que le L293D intègre des diodes en parallèle. Ce qui implique des concessions sur les caractéristiques (le courant max passe à 1A par canaux et 2A pic pour la version sans les diodes). Pour des raisons de design du circuit, le composant L293Dintègre deux ponts en H dans une seule et même puce. Le circuit permet alors, de contrôler soit quartes moteurs dans un seul sens de rotation, soit deux moteurs dans les deux sens.

III.3.3.2 Utilisation du L293D avec Arduino et 2 moteurs DC

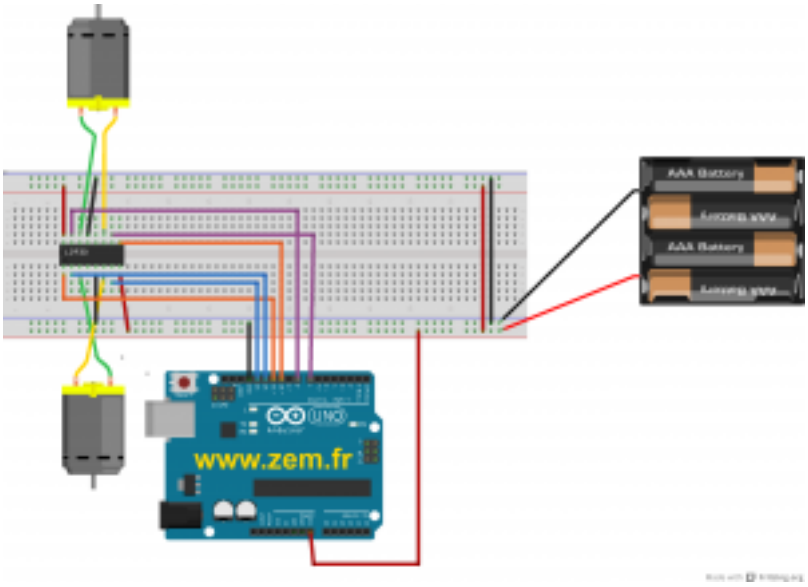


Figure III.4 *: Allure du montage moteurs-Arduino Uno*

L'utilisation du L293D, dotera l'Arduino d'une capacité de pilotage de deux moteurs DC indépendamment l'un de l'autre. Si la puissance de ces derniers est faible, il sera possible d'utiliser les 5V en sortie de notre Arduino pour les alimenter.

III.3.4 Les actionneurs

Un **actionneur** est un organe qui transforme l'énergie qui lui est fournie en un phénomène physique utilisable. Les interfaces hépatiques permettent au robot de saisir des objets. Les moteurs permettent à des éléments mobiles de bouger suivant un ou plusieurs degrés de liberté. Dans le cadre de notre travail, les actionneurs utilisés se limiteront aux moteurs à courant continu.

III.3.4.1 Moteurs à courant continu

La plupart des robots mobiles sont ainsi actionnés par des moteurs électriques à courant continu avec ou sans collecteur, alimentés par des convertisseurs de puissance fonctionnant sur batterie.

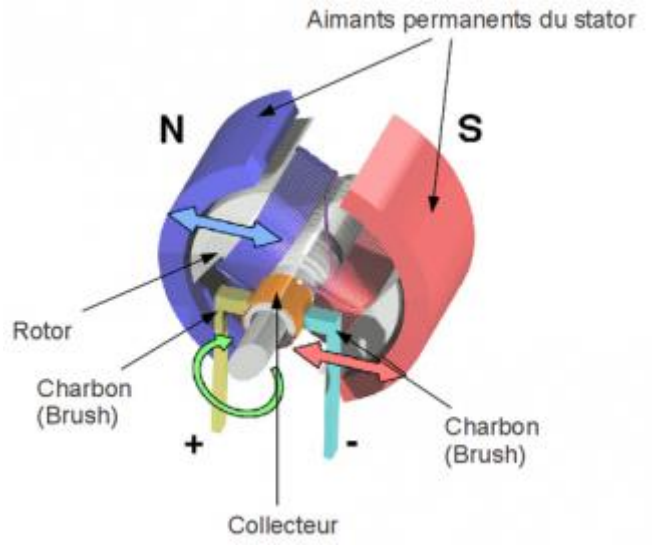


Figure III.5 : *Structure d'un moteur a courant continu*.

Le moteur à courant continu est composé d'un stator ou inducteur (partie fixe) d'un rotor ou induit (partie mobile).Le rotor composé de fils de cuivre enroulés sur un support lui-même monté sur un axe. Cet axe, constitue l'arbre de sortie du moteur. C'est lui qui va transmettre le mouvement à l'ensemble mécanique (pignons, chaîne, actionneur…) qui lui est associé en aval. Dans le cas d'un robot sur roues, par exemple, la roue est disposée sur cet axe, par l'intermédiaire d'un réducteur qui diminue la vitesse de rotation afin d'en augmenter le couple.

III.3.4.2 Vitesse angulaire

La vitesse de rotation est mesurée par rapport à l'axe de rotation du moteur. Si nous supposons que le moteur entraîne son axe, lorsqu'il est alimenté par un courant, ce dernier va avoir une certaine vitesse angulaire. Il peut tourner lentement ou rapidement. On mesure une vitesse de rotation en mesurant l'angle en radians parcourus par cet axe pendant une seconde.

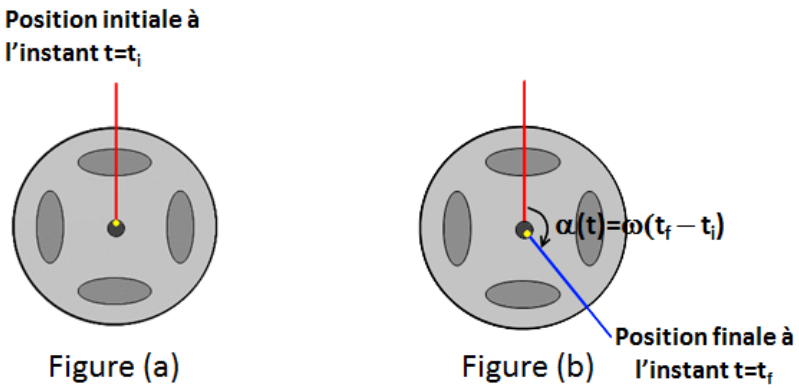


Figure III.6 **:** *Position de la roue à deux instants*.

Marquage de l'axe du moteur par un point jaune (gauche). Au bout d'une seconde (droite), mesure de l'angle αentre la position de départ et d'arrivée du point jaune. On obtient alors la vitesse de rotation de l'axe du moteur. Cette mesure est exprimée rd/s.

III.3.4.3 Les réducteurs

Un moteur électrique est bien souvent très rapide en rotation. Afin qu'un moteur ne tourne pas trop rapidement, un réducteur de vitesse est nécessaire. Ce dernier se comporte comme un "frein" pour imposer des vitesses de rotations réduites indépendamment de la charge imposée.

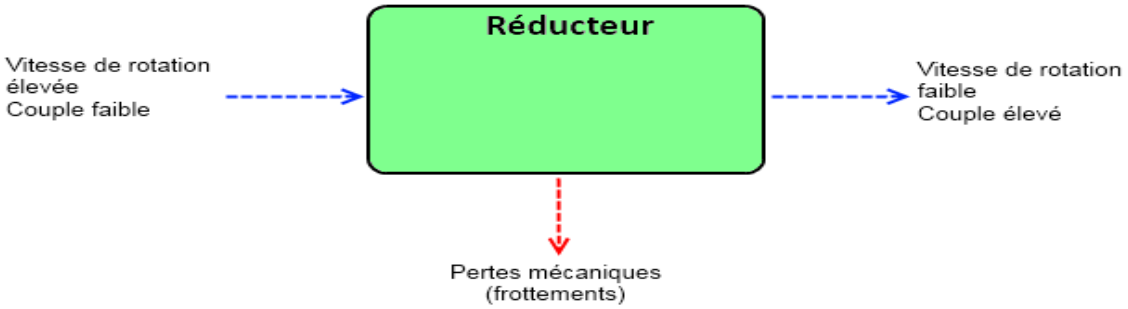


Figure III.7**:** *Procédure d'une réduction de la vitesse*.

III.3.4.4 Alimentation du moteur

Connecter un moteur sur une source d'énergie : la pile

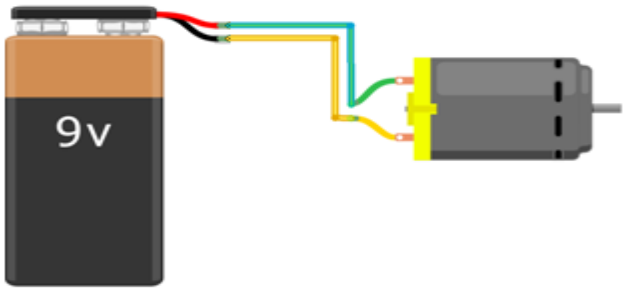


Figure III.8 : *Alimentation du moteur avec batterie.*

- **Caractéristiques du moteur à courant continu**

- Plage de variation de vitesse très grande (> 1000 en boucled'asservissement)
- Couple de démarrage important, idéal pour l'entraînement de charges à forte inertie.
- Rapport volume/puissance très supérieur à toutes les autres technologies
- Rendement élevé
- Linéarité tension/vitesse, couple/courant

Mais :

- Prix élevé
- Maintenance coûteuse (remplacement des balais en graphite, usure du collecteur)
- Source importante de parasites.

III.3.4.5 Environnement Arduino avec le moteur

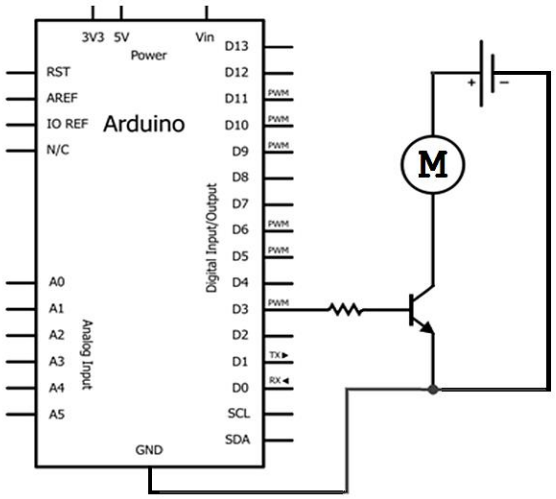


Figure III.9 : *Alimentation ensemble Arduino-moteur*

Le transistor est commandé par une sortie de la carte Arduino via la résistance sur la base. Lorsque l'état de la sortie est au niveau 0, le transistor est bloqué et le courant ne le traverse pas. Le moteur ne tourne pas. Lorsque la sortie vaut 1, le transistor est commandé et devient saturé, c'est-à-dire qu'il laisse passer le courant et le moteur se met à tourner. L'inconvénient vient du transistor qui :

- parcouru par un grand courant, peut chauffer et griller s'il n'est pas refroidi
- est sensible aux parasites et risque d'être endommagé
- ne supporte pas des tensions importantes.

La réponse à ces contraintes passe par l'adjonction d'un dissipateur thermique (ou

Radiateur) que le transistor puisse refroidir. L'installation un condensateur de filtrage pour réduire ou éliminer les parasites et enfin l'utilisation de diode pour écrêter les tensions importantes.

III.3.5 Bloc encodeur

Un **encodeur** est un composant matériel ou logiciel qui transforme une information en un code. Généralement associé à un dispositif électromécanique, il devient un instrument capable d'estimer une position linéaire ou angulaire de l'élément mesuré.

Un encodeur audio/vidéo logiciel, aussi appelé codec, transforme les informations en données informatiques compressées tandis qu'un encodeur électromécanique, linéaire ou rotatif, génère un signal électrique selon la position ou le déplacement.

Il existe plusieurs types d'encodeurs dont les principaux sont :

- Encodeur a capteurs optiques
- Encodeur a effet hall

➢ ***Encodeur optiques***

Les Encodeurs optiques sont des capteurs qui fournissent des signaux électriques sous forme de trains d'impulsions, lesquels peuvent être interprétés comme mouvement, direction ou position. Les encodeurs rotatifs servent à mesurer le déplacement **rotatif** d'un axe.

La figure III.2 montre un exemple de composants de base d'un encodeur rotatif. En effet, une diode électroluminescente (LED), un disque et un capteur de lumière de l'autre côté du disque.

Le disque, monté sur l'axe rotatif, comporte des motifs encodés de secteurs opaques et transparents. Lorsque le disque tourne, les segments opaques bloquent la lumière alors que ceux où le verre est clair la laissent passer. Ceci génère des impulsions d'onde carrée qui peuvent ensuite être interprétées comme position ou mouvement. En général, les encodeurs ont entre 100 et 6 000 segments par révolution. Ceci correspond à une résolution de 3,6 degrés pour un encodeur à 100 segments et à une résolution de 0,06 degrés pour un encodeur à 6 000 segments.

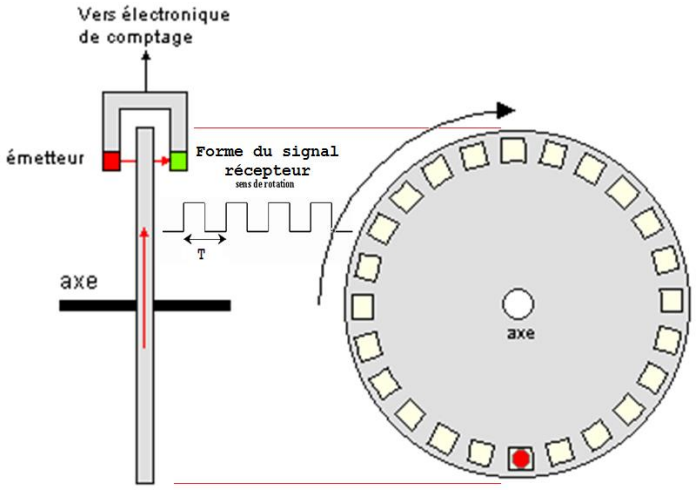


Figure III.10 : *Encodeur optique*

Le même principe de fonctionnement est valable pour les encodeurs linéaires, si ce n'est que le disque rotatif est remplacé par une bande opaque stationnaire dont la surface comporte des fentes et que l'assemblage capteur/LED est fixé sur la partie mobile.

Un encodeur ne générant qu'une série d'impulsions ne serait pas très utile puisqu'il ne pourrait pas indiquer la direction de la rotation. L'encodeur en quadrature comporte deux pistes de code dont les secteurs sont décalés de 90 degrés d'une piste à l'autre (**Figure III.10**) ; ceci permet d'avoir deux sorties de voie qui indiquent la position et la direction de la rotation. Si A devance B, par exemple, le disque tourne dans le sens des aiguilles d'une montre. Si B devance A, le disque tourne dans le sens contraire. Par conséquent, en surveillant à la fois le nombre d'impulsions et les phases relatives des signaux A et B, vous pouvez suivre la position et la direction de la rotation.

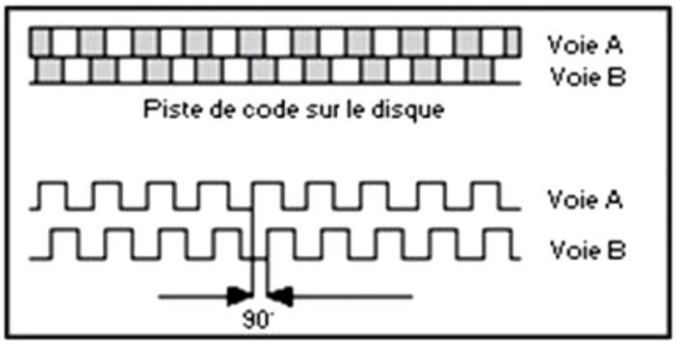


Figure III.11 : *Signaux de sortie A et B d'un encodeur en quadrature*

- ***Encodeur a effet hall***

Leur principe est basé sur une découverte déjà ancienne. En 1879, le physicien américain Edwin Herbert Hall découvre que si un courant Io traverse un barreau en matériau conducteur ou semi-conducteur, et si un champ magnétique d'induction B est appliqué perpendiculairement au sens de passage du courant, une tension V_h, proportionnelle au champ magnétique et au courant I_o, apparaît sur les faces latérales du barreau.

Les électrons sont déviés par le champ magnétique, créant une différence de potentiel appelée tension de Hall .Le champ magnétique déforme la trajectoire des électrons car il engendre une force de LORENTZ.

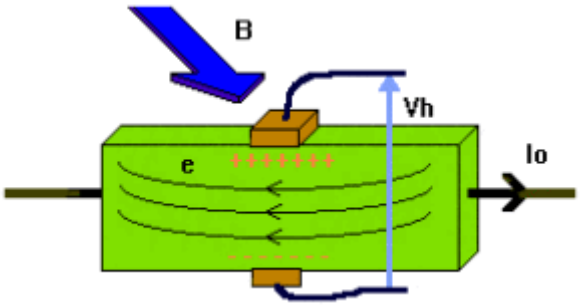


Figure III.12 : *Capteur à effet hall*

Un capteur à effet hall donne donc un signal lorsqu'il détecte un champ magnétique. La tension de Hall est alors amplifiée dans le capteur.

Le codeur incrémental

Le codeur incrémental (figure III.13), associé au moteur, est un générateur d'impulsions qui fournit 2 voies en quadrature et un top Zéro.

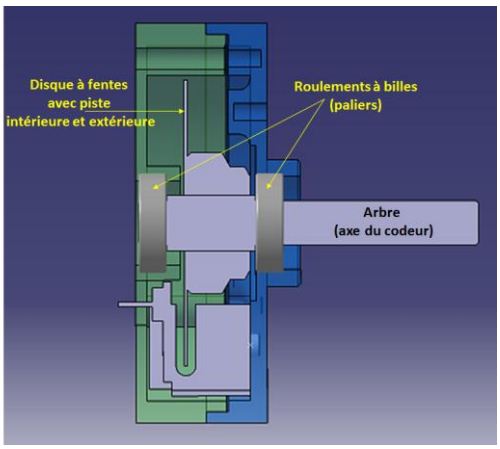


Figure III.13 : *Vue en coupe d'un codeur incrémental*

Il est destiné à des applications où l'information de position est obtenue par mesure du déplacement de l'objet. Le codeur délivre un train d'impulsions dont le nombre permet de déduire la valeur du déplacement ainsi que la vitesse car ce dernier est proportionnel à la fréquence des impulsions.

Constitué d'un disque comportant deux à trois pistes : A et Z.

Piste extérieure (A), divisée en intervalles d'angles égaux, alternativement opaques et transparents. C'est le nombre de fenêtres ainsi créées qui détermine la résolution du capteur.

Piste intérieure (Z: top zéro), qui ne comporte qu'une seule fenêtre et qui délivre qu'un signal par tour du disque. Ce « top zéro » permet de réinitialiser la partie commande et de connaître une position d'origine.

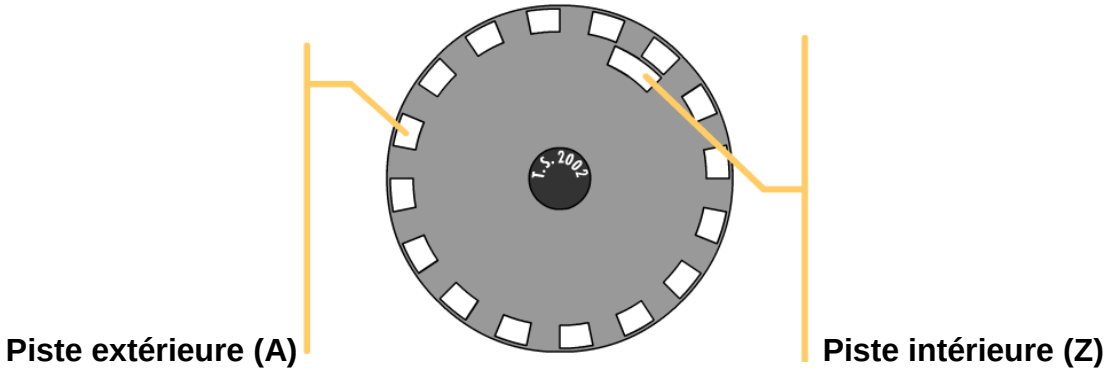


Figure III.14: *Vue du disque d'un codeur incrémental*

Pour un tour complet de l'axe du codeur, la partie commande reçoit autant d'impulsions électriques qu'il y a de fenêtres, dont la durée dépend de la vitesse de rotation du disque.

- Particularités de fonctionnement

Un codeur incrémental possède trois têtes de lecture :

Une tête de lecture est affectée à la piste intérieure et délivre une impulsion par tour, permettant à la commande de compter le nombre d'impulsions reçues.

Deux têtes de lecture sont placées sur la piste extérieure. Chaque tête, prise isolement, permet à la partie commande de déterminer l'angle de rotation du disque en comptant le nombre d'impulsions reçues.

Les deux têtes sont décalées l'une par rapport à l'autre d'un quart de largeur de fente. Ainsi, les signaux émis sont décalés dans le temps. La partie commande, en détectant quelle voie change d'état en premier peut déterminer le sens de rotation du disque.

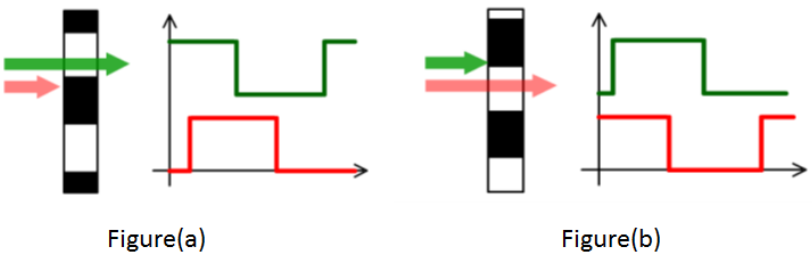


Figure III.15: *Chronogrammes*

- (Figure. a) Le front montant de la voie verte se présente avant celui de la voie rouge.
- (Figure .b) Le front montant de la voie rouge se présente avant celui de la voie verte.

III.3.6 Guide

- ***Interfaces graphiques sous MATLAB***

Les interfaces graphiques (ou interfaces homme machine) sont appelées GUI (pour Graphical User Interface) sous MATLAB. Elles permettent à l'utilisateur d'interagir avec un programme informatique, grâce à différents objets graphiques

(Boutons, menus, cases à cocher…). Ces objets sont généralement actionnés à l'aide de la souris ou du clavier.

- **L'outil GUIDE**

Depuis la version 5.0 (1997), MATLAB possède un outil dédié à la création des interfaces graphiques appelé GUIDE (pour Graphical User Interface Development Environment).

- **Présentation**

Le GUIDE est un constructeur d'interface graphique qui regroupe tous les outils dont le programmeur à besoin pour créer une interface graphique de façon intuitive. Il s'ouvre, soit en cliquant sur l'icône  , soit en tapant **guide** dans le Command Window de MATLAB.

Le placement des objets est réalisé par sélection dans une boite à outils. Leur mise en place et leur dimensionnement se font à l'aide de la souris.

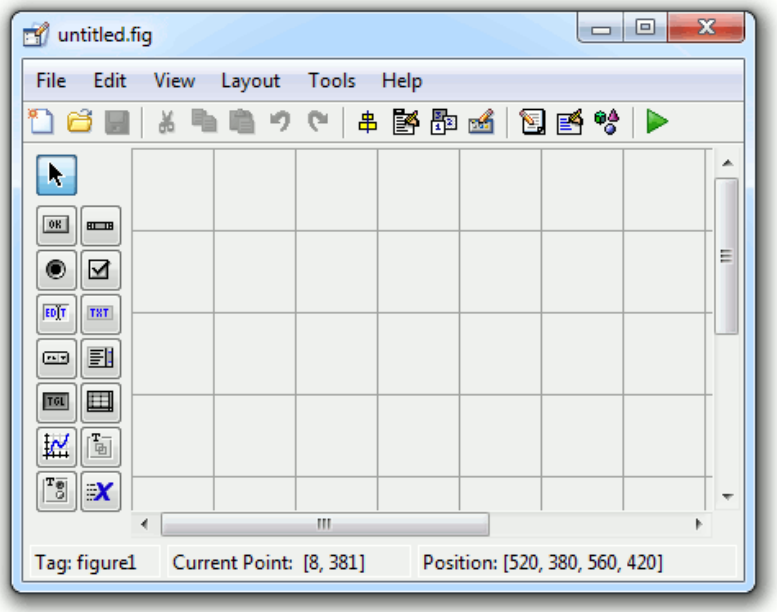


Figure III.16: Fenêtre principale du GUIDE

Un double-clique sur un objet permet de faire apparaître le Property Inspector où les propriétés des objets sont facilement éditables. Leurs modifications et la visualisation de ces modifications sont immédiates.

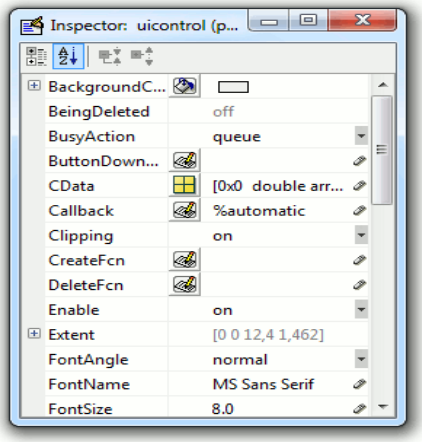


Figure III.17: Property Inspector

Le GUIDE possède également des outils pour gérer l'alignement des objets et pour créer des barres d'outils ou des menus.

Une fois l'interface graphique terminée, son enregistrement donne deux fichiers portant le même nom mais dont les deux extensions sont .fig. et .m.

Le fichier .fig contient la définition des objets graphiques (positions et propriétés). Ce fichier peut être ouvert ultérieurement avec le GUIDE pour modifier les objets graphiques.
Le fichier .m contient les lignes de code qui assurent le fonctionnement de l'interface graphique (actions des objets). Ce fichier peut être édité dans le MATLAB Editor pour y ajouter des actions à la main. C'est ce fichier qui doit être lancé pour utiliser l'interface graphique.

Lors de l'enregistrement, le GUIDE génère deux fichiers :

- un fichier.fig (non éditable) contenant les objets graphiques Figure, Axes et Push button ;
- un fichier .m contenant le code du fonctionnement de l'interface graphique

Et voilà pour un titre d'exemple notre travail :

```
250    % eventdata  reserved - to be defined in a future version of MATLAB
251    % handles    empty - handles not created until after all CreateFcns called
252
253    % Hint: edit controls usually have a white background on Windows.
254    %       See ISPC and COMPUTER.
255 -  if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
256 -      set(hObject,'BackgroundColor','white');
257 -  end
258
259
260    % --- Executes on button press in pushbutton1.
261    function pushbutton1_Callback(hObject, eventdata, handles)
262    % hObject    handle to pushbutton1 (see GCBO)
263    % eventdata  reserved - to be defined in a future version of MATLAB
264    % handles    structure with handles and user data (see GUIDATA)
265 -  L1=3;
266 -  L2=4;
267 -  Yo=str2num(get(handles.Yo,'string'));
268 -  Xo=str2num(get(handles.Xo,'string'));
269 -  Y=str2num(get(handles.Y,'string'));
270 -  X=str2num(get(handles.X,'string'));
271 -  BCO=sqrt(Yo^2+Xo^2);
272 -  rO=sqrt(abs((L1^2)-(((BCO^2)+(L1^2)-(L2^2))/(2*BCO))^2));
273 -  ALPHAO=acosd(rO/L1)+acosd(rO/L2);
```

pfe / Yo_CreateFcn Ln 256 Col 26 OVR

Figure III.18 : Fichier matleb

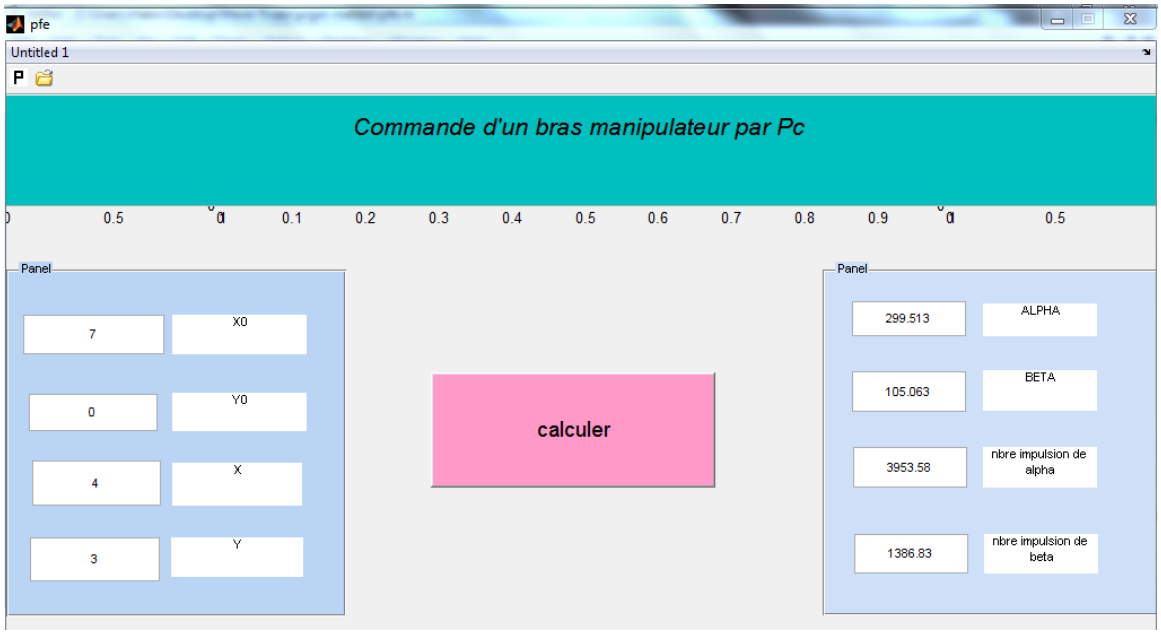


Figure III.19*:* Fichier.fig

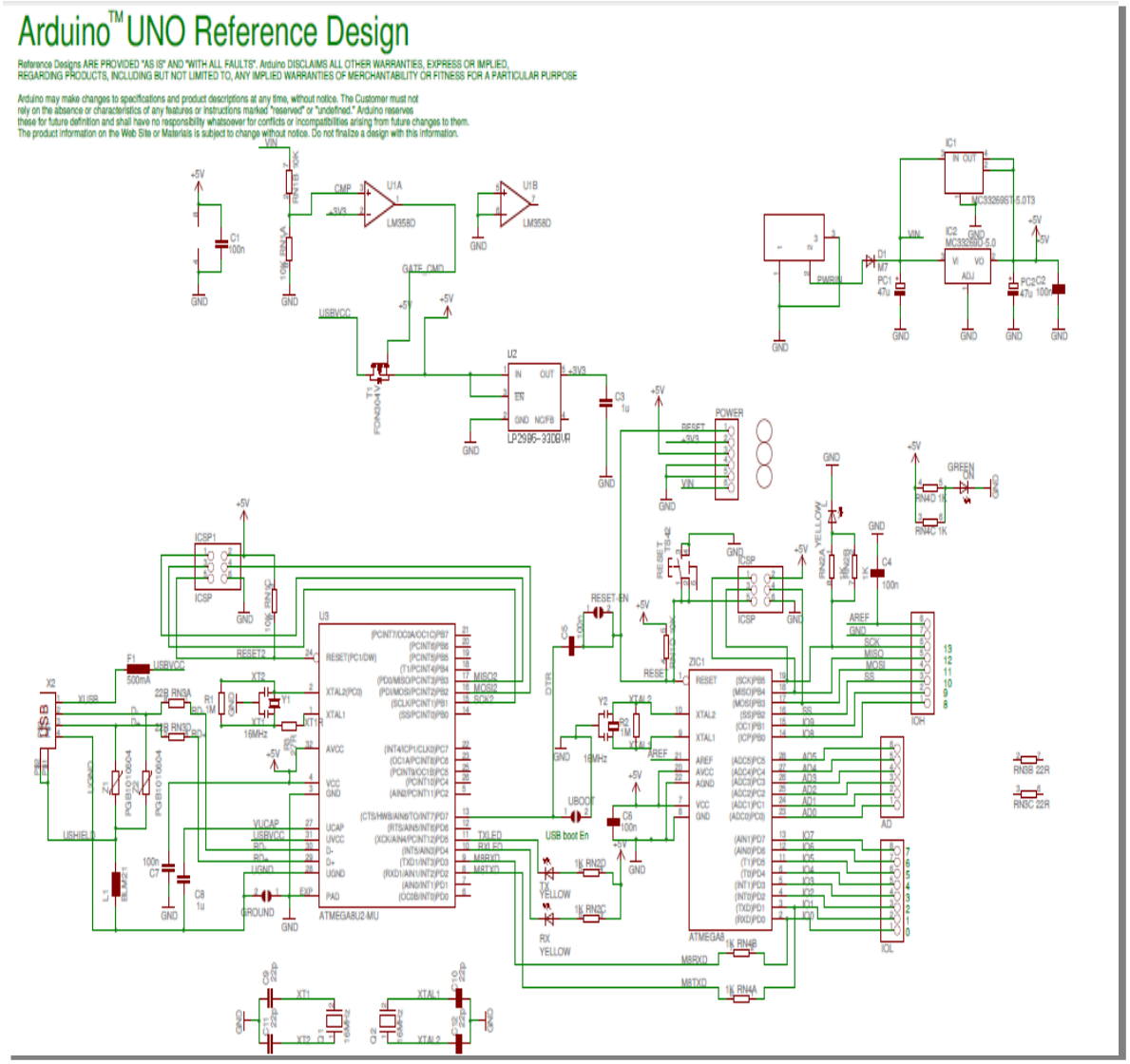
III.3.7 Conclusion

L'objectif de cette étude était la réalisation d'un bras manipulateur à deux degré de liberté. Les résultats expérimentaux obtenus sont satisfaisants sur les trois aspects abordés (mécanique, électronique et informatique).

Nous avons pu constater, ce qui prime dans Arduino, c'est sa simplicité qui permet de mettre en œuvre de nombreux objets numériques à moindre coût sans être un spécialiste du fer à souder ou de la programmation des microcontrôleurs. C'est cette qualité-là qui donne à Arduino le succès planétaire qu'on lui connaît.

Conclusion Générale :

L'objectif assigné à ce travail était d'élaborer une commande de position d'un bras manipulateur à deux degré de liberté, grâce à un outil informatique (language arduino et matlab) et d'une carte électronique pour assurer la possibilité à un ordinateur de communiquer avec des périphériques extérieurs. Les objectifs fixés par le cahier de charge ayant été atteint, malgré les difficultés techniques et fiduciares, la persepective d'élargir l'horizon de la commande dans les domaines de l'automatique serait souhaitable. En effet, la programmation de véritable structures analytiques (équation différentielles, loi de commande linéaire ou non, optimisation, etc) pour les systèmes embarqués, serait un véritables challenge dans les dommaines qu'intègrerait l'automatique. Le sujet reste donc ouvert, pour des oxygénations accadémiques qui trouveraient, preneurs.


Arduino™ UNO Reference Design
Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS". Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE
Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Arduino reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.
ATMEGA8U2-MU
ATMEGA8
LM358D
LP2985-330BVR
MC33269D-5.0T3
MC33269D-5.0
FDN304P
16MHz
USB boot En
RESET-EN
POWER
ICSP
ICSP1
USB
GREEN ON
YELLOW
RX YELLOW
TX YELLOW

[1] **:** P. Fissette, H.Buyse, J.C.Samin « Introduction à la robotique », 29 juin 2009
>>**http://meca2732.pdf**
[2] : Jean-Louis Boimond « ROBOTIQUE », ISTIA, Université Angers
>> www.istia.univ-angers.fr/~boimond/Cours_robotique.pdf
[3] : Jacques Gangloff « Cours de Robotique »
>> http://eavr.u-strasbg.fr/wiki/upload/a/a4/Cours_rob_intro.pdf
[4]: O.Ahcene, NEMMICHE Belkacem, BELHAMITI hadj Afif, BERRAHO Ismail, « Commande en 3D d'un bras robot à 5 DL par positionnement en temps réels par deux caméras », Université ABDELHAMID IBN BADIS De Mostaganem.
[5] Arato.S.Deo thèse de PhD "Inverse Kinematics and Dynamic Control Methods for Robotic Systems" Houston .Texas.1995
[6] Philippe Coiffet "La Robotique : Principes et Applications " Editions Hermes Paris 1992.
[7] Wissama Khalil, Etienne Dombre "Modélisation identification et commande des robots "
Editions Hermes science publications Paris 1999.

[8] Victor Gavriloiu Thèse de Master "Design of Dynamic Non-Linear Control Techniques for Flexible-Link Manipulators " Université de Concordia Canada 2005.

[9] S.K.Dwivedy, P.Eberhard, Mechanism and Machine Theory 41 ELSEVIER Science Direct 2006.
[10] L.Sciavico, B.Siciliano, modelling and control of robot manipulators: The McGraw-Hill
Companies Inc, 1996.
[11] E.A.Miranda Thèse de Master "Direct Adaptive control Of a Two-Link Flexible Manipulator" Texas A &M University-Kingsville 2004.
[12] Le site Arduino : http://www.arduino.cc.
[13]Le lien: http://www.semageek.com/category/electronique/arduino-electronique/