REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLAB DE BLIDA 1
FACULTE DE TECHNOLOGIE
DEPARTEMENT DE MECANIQUE

Signals & Systems Laboratory
University of Boumerdes

Projet de Fin d'Etudes
Pour l'obtention du Diplôme de Master en
Génie Mécanique
Option : Fabrication Mécanique et Productique

CONCEPTION ET RÉALISATION D'UN BRAS MANIPULATEUR [ALG.-M. O.- 02] COMMANDÉ PAR ARDUINO UNO

Proposé et encadré par :
BENMISRA Abdelkader

Réalisé par :
DRAISSIA Anis
MAZOUNI Fayçal
DJABER Said

Année Universitaire 2020/2021

Remerciements

Nous tenons à honorer toutes les personnes qui, de près ou de loin, ont collaboré à l'accomplissement de cette œuvre. Selon la tradition, et à travers cette page de remerciements, nos profondes reconnaissance sont exprimés à :

ALLAH, qui nous a donné la force dans les moments difficilesd'éditer ce mémoire.

Aucune étude n'est couronnée que celle effectuée avec l'assistance honnête des hommes qui nous sont proches.

Nos parents qui nous ont toujours entouré et motivé à sans cessedevenir meilleur ;

Nos frères et sœurs qui nous ont assisté dans ces moments difficiles etnous ont servi d'exemple ;

Nous remercions notre Monsieur Prof. TEMMAR Mustapha pour avoir accepté la soutenance de ce mémoire, au sein de Laboratoire Signaux & Systèmes de l'Université de Boumerdes dont l'aide précieuse a été indispensable sur le plan scientifique et humain ;

Nos très cordiaux remerciements vont aussi à tous nos enseignants pour tous les efforts conjugués tout au long de notre formation dans la Facultés Technologie de l'Université Blida 1, dont notre promoteur Mr. BENMISRA Abdelkader pour son aide et ses conseils pertinentes ;

Trouvez ici l'expression de nos profondes gratitudes et reconnaissances

Je dédie ce travail à :
Mes chers parents Mes frères et mes sœurs
Et mes nièces Mes chers amis …

Anis

Je dédie ce travail

A mes chers parents pour tous les efforts consentis pour ma formation ;

Frères, sœurs, cousins et cousines…

Que Dieu, me les préserve.

Saïd

Je dédie ce travail tout particulièrement aux personnes qui me sont les plus chères au monde ; Mes très chers parents pour leur patience, soutien et confiance, Je dédie ce travail également à tous les membres de ma famille, A tous ceux qui m'aiment…

M. Mazouni

Sommaire

Remerciements
Dédicaces
Table des Matières

Table des Matières

Table des Figures

Liste Des Tables

Abréviations et acronymes

S.M.A. / S.M.P.A. : Systèmes Mécaniques Poly-articulés
CPU : circuits intégrés et modules
Robots Anthropomorphes : 6R : 6 rotations, 6 **DDL :** 6 degrés de liberté
Model dynamique : $\Gamma = (q, \dot{q}, \ddot{q}, f_e)$

Γ: Vecteur des couples/forces des actionneurs **[N.m]**

q: Vecteur des positions articulaires **[rad]**

$\dot{q}$: Vecteur des vitesses articulaire **[rad/s]**

$\ddot{q}$: Vecteur des accélérations articulaires **[rad/s²]**

$\mathbf{f_e}$: **vecteurs représentant l'effort extérieur qu'exerce le robot sur l'environnement**

T : Matrice de transformation homogène
: Vecteur position **[m]**
A : Matrice d'orientation **[rad]**
J(q) : Matrice jacobienne
X: Vitesse des variables opérationnelles **[m/s]**
$\ddot{X}$**:** Accélération cartésiennes **[m/s²]**
Vn **:** Vitesse de translation **[m/s]**
: Vitesse de rotation [**rad/s]**
***E* :** Energie cinétique **[N.m]**
***U* :** Energie potentielle **[N.m]**
: Vecteur unitaire
: Masse du corps **[kg]**
I: Moments d'inertie des actionneurs **[N.m/rad/s²]**
Vj : Vecteur de vitesses linéaire **[m/s]**
ωj **:** Vecteur de vitesses angulaire **[rad/s]**
Γfj **:** Couple de frottements **[N.m]**
Fsj**:** Frottement sec **[N.m]**
Fvj **:** Frottement visqueux **[N.m/rad/s]**
$\overrightarrow{e_t}$**:** Vecteur tangent unitaire
$\overrightarrow{e_n}$**:** Vecteur normal unitaire
$\overrightarrow{e_b}$**:** Vecteur bi-normal
s : Abscisse curviligne
$\rho = \frac{1}{k}$:Rayon de courbure

k : La courbure .
φ : Position angulaire **[rad]**
$l1$: Distance $Z0$ et ZT **[m]**
$l2$: Distance $Y0$ et YT **[m]**
$l3$: Distance $X0$ et XT **[m]**

Q : Matrice d'orientation curviligne.

Arduino : Carte de Commande
UNO : La ***carte Arduino Uno*** est basée sur un ATMega328 cadencé à 16 MHz. C'est la plus simple et la plus économique ***carte*** à microcontrôleur d'***Arduino***
NANO : la carte Nano : L'Arduino Nano est essentiellement un Arduino UNO réduit, ce qui le rend très pratique pour les espaces restreints et les projets pouvant nécessiter une réduction de poids chaque fois que cela est possible
Due : L'Arduino Due est l'une des cartes les plus grandes et la première carte Arduino à être alimentée par un processeur ARM.

Résumé

Les robots poly-articulés sont des moyens de production fortement utilisés, en effet l'amélioration de la précision des robots industriels pour des applications de soudage, découpage, peinture et usinage à grande vitesse est indispensable. De nombreux robots existants possèdent déjà une sorte d'intelligence de perception, c'est-à-dire la capacité d'analyser et d'interpréter leur environnement (en particulier segmenter la scène environnante en objets de catégories identifiées, avec des distances et des déplacements estimés), on a qualifié les différentes souplesses des robots industriels afin de déduire une cartographie de rigidité dans l'espace de travail cartésien. Dans cette thèse, on a analysé des robots polyvalents. On a montré que la conception des modules de mouvement de ces robots peut être simplifiée par une analyse de modélisation, bien qu'ils ne possèdent pas la propriété de découplage cinématique agréable en tant que robots industriels. Néanmoins, il est probable que les premiers robots surpassant les humains dans un contexte bien délimité devraient bientôt apparaître. Notre positionnement de concerne l'utilisation des dispositifs de commande Arduino uno, pouvant par exemple être issue de projets antérieurs, ou d'études préliminaires. Finalement, une nouvelle plate-forme de levage et manutention, est optimisée et testée afin de valider la réalisation réelle pour atteindre le plein potentiel de l'architecture de robot [Algérie Machine Outils 02] réalisé.

Mots clés : Modélisation, ***l'Arduino uno***, Gamme de fabrication, Cartographie de C.A.O., Espace opérationnel.

Abstract

Poly-articulated robots are a widely used means of production, indeed improving the precision of industrial robots for welding, cutting, painting and high-speed machining applications is essential. Many existing robots already have a sort of perceptual intelligence, i.e. the ability to analyze and interpret their environment (in particular segment the surrounding scene into objects of identified categories, with distances and estimated displacements), we have qualified the different flexibilities of industrial robots in order to deduce a stiffness map in the Cartesian workspace. In this thesis, we analyzed multipurpose robots. It has been shown that the design of the motion modules of these robots can be simplified by modeling analysis, although they do not have the property of kinematic decoupling pleasing as industrial robots. Nevertheless, it is likely that the first robots to surpass humans in a well-defined context are expected to appear soon. Our positioning concerns the use of Arduino uno control devices, which may for example be the result of previous projects, or preliminary studies. Finally, a new lifting and handling platform will have to be optimized and tested in order to validate the actual realization to reach the full potential of the robot architecture [Algeria - Tool machines 02] made.

Keywords: Modeling, Arduino uno, Manufacturing range, C.A.O. mapping, Operational space

الملخص

الروبوتات متعددة المفاصل وسيلة إنتاج مستخدمة على نطاق واسع، إن تحسين دقة الروبوتات الصناعية للِحام والقطع والطِلاء وتطبيقات الآلات عالية السرعة أمر ضروري. تمتلك العديد من الروبوتات الموجودة بالفعل نوعًا من الذكاء الإدراكي، أي القدرة على تحليل وتفسير بيئتها (على وجه الخصوص تقسيم المشهد المحيط إلى أجسام من فئات محددة، مع مسافات وحالات انسحاب مقدّرة)، قمنا بتأهيل المرونة المختلفة للروبوتات الصناعية من أجل استنتاج خريطة الصلابة في مساحة العمل الديكارتية. في هذه الأطروحة، قمنا بتحليل الروبوتات متعددة الأغراض. لقد ثبت أن تصميم وحدات الحركة لهذه الروبوتات يمكن تبسيطه من خلال تحليل النمذجة، على الرغم من أنها لا تمتلك خاصية الفصل الحركي المثالي مثل الروبوتات الصناعية. ومع ذلك، فمن المحتمل أن تظهر قريبًا أولى روبوتات تتفوق على البشر

في سياق محدد جيدًا. يتعلق تحديد المواقع لدينا باستخدام أجهزة التحكم **(ضامن منصة الاستخدام [فريد])**، والذي قد يكون على سبيل المثال نتيجة لمشاريع سابقة أو دراسات أولية. أخيرًا، قمنا بتحسين منصة الرفع والمناولة الجديدة واختبارها من أجل التحقق من صحة الإدراك الفعلي للوصول إلى الإمكانات الكاملة لبنية الروبوت **[الجزائر–آلات العدد ٠٢] المُنجز**.

الكلمات الرئيسية: النمذجة، **ضامن منصة الاستخدام [فريد]**، نطاق التصنيع، رسم الخرائط، مساحة التشغيل.

Introduction Générale

Introduction

Le nouveau concept des systèmes mécaniques poly articulées a très rapidement suscité un vif intérêt dans le monde, ainsi que l'aide apportée par la CAO devient primordiale et la simulation du mécanisme avant fabrication est incontournable. Ceci conduit à l'émergence de concepts tels que les " Usines" intelligentes. Les difficultés de cette conception notamment liées à leur caractère pluridisciplinaire. Le concept est né, la nouveauté tenait à l'intégration des tribologies sous les notions métallurgiques. L'intégration de certains mécanismes avec leur configuration était déjà bien explorée.

De nouvelles approches basées sur des systèmes robotisés sont recherchées, le but est de proposer des outils généraux et des méthodes capables de soutenir le travail coopératif entre divers participants d'un projet de conception.

Dans un premier axe, les méthodes utilisées doivent s'appuyer sur des modèles de haut niveau, fonctionnels, exécutables, que nous pouvons appeler des Prototypes Virtuels. Les concepts d'ingénierie concourante, de cycle de conception en V ont été abordés comme moyens pour réduire les couts et les temps d'études au cours du processus de conception. Les méthodes de conception doivent aussi prendre en compte la conception coopérative et la réutilisation des acquis (retour d'expérience). Ensuite en deuxième axe, on s'oriente vers la synthèse de tolérance paramétrique et les méthodes existantes pour traiter en particulier de la démarche de conception qui doivent donc être basées sur des procédures et des langages normalisés ou standardisés facilitant les échanges. Puis dans le troisième axe, nous nous sommes intéresses aux grandes classes de conception : la conception robuste et la conception basée sur la fiabilité.

Dans ce mémoire de master : Particulièrement, il s'agit de concevoir un bras manipulateur pour laboratoire [200g, charge 50g, 320mm] sous le développement des techniques de prototypage, moulage plastique et impression 3D

Dans le premier chapitre, nous développons des généralités sur la robotique, on présente les parties des robots entre le mécanisme et la théorie associée. Entre autres nous citons illustrations de modèles enregistrés ISO de systèmes (mécanismes) pouvant être utilisés dans un environnement de simulation comprenant éventuellement plusieurs outils de CAO.

Cependant, la modélisation d'une structure, qu'elle soit parallèle ou bien sérielle, ne doit pas se limiter à une description purement géométrique, mais doit intégrer toutes les caractéristiques.

Un deuxième chapitre : .....modélisation mathématique. La part importante de notre travail se situe donc dans la définition et la mise au point de méthodes et d'outils de

conception mécanique en allant d'une partie mathématique d'un système.

La modélisation des différents types de composants, de différents domaines d'énergie mécanique/thermique ;

Sur **troisième chapitre**, nous dressons les limitations des outils électroniques pour mieux appréhender la partie carte de commande, et la problématique associées en intégrant simultanément les apports que nous nommerons par la suite du comportement des systèmes mécaniques, pour offrir de très bonnes performances en termes de précision, vitesse et d'accélération. En résumé ce n'est qu'une initiation à l'utilisation des composantes électroniques dans une structure mécanique.

Le **quatrième** traite les étapes de conception de bras réalisé avec des copies écran des dessins réalisés par Solidworks,

Avec une conclusion et références

Chapitre 1
Généralités sur la robotique

1 Chapitre 1 Généralités sur la robotique

1.1 Etat de l'art

Les concepts actuels liés à la robotique ne sont pas si éloignés des idées énoncées par Aristote il y a plus de 2000 ans. Le premier automate identifié comme tel date de 380 avant J.-C. Il s'agit d'un pigeon en bois fabriqué par Archytas de Tarente, un ami de Platon. Le terme de robotique quant à lui est apparu en **1942** dans le cycle universellement connu rédigé par *Isaac Asimov* et intitulé "**Les robots**". L'homme a depuis toujours été fasciné par la possibilité de pouvoir recréer la vie. Désir profond de jouer les créatrices, ou bien tentations de la mystification, ou encore simple jeu de copie de l'existant, les divers hommes d'airain, androïdes ou robots, crées dans la littérature, au cinéma ou dans la réalité, ont depuis la haute antiquité répondu à nos fantasmes.

Un passage des "Contes des mille et une nuits" nous entraîne chez la reine Abrizah où des statues parlaient et chantaient en gesticulant (49° nuit). De nombreux contes chinois font aussi intervenir des androïdes : ainsi l'histoire de cet inventeur qui construisit pour un roi "un homme mécanique, d'une intelligence sans égale, capable de chanter, de danser et de se mouvoir comme un être humain". Héron qui vivait à Alexandrie vers la fin du deuxième siècle, décrit plus de cent appareils mécaniques parmi lesquels un groupe de statues dans une scène de sacrifices. Il est difficile dans ces textes de faire la part entre le merveilleux et ce qui a pu être réalité[1]

Ce n'est qu'avec le développement de l'horlogerie au XVI° siècle que les automates ont pu acquérir une certaine complexité dans leurs mouvements. En effet tous les mécanismes précédents devaient fonctionner sur le principe des clepsydres, c'est à dire à partir d'un écoulement régulier d'eau (éventuellement remplacée par du mercure ou du sable). Les développements de la mécanique et de l'horlogerie entraînèrent la création de ces merveilleux automates qui font toujours l'admiration de tous, car quelques exemplaires ont pu survivre plusieurs siècles et arriver jusqu'à nous.

Il est cependant nécessaire de faire la distinction entre deux types d'automates : les automates cycliques qui exécutent un même mouvement de façon répétitive, et les automates à programme dont le mouvement est beaucoup plus riche. Dans le premier cas, le mouvement est obtenu par une mécanique figée qu'il est très difficile de modifier. Par contre, dans le deuxième cas les mouvements sont programmés à partir d'un dispositif central qui décrit leur séquence et éventuellement leur amplitude. Ces dispositifs débutèrent par des cylindres

[1] HAMDI HOCINE INTRODUCTION A LA ROBOTIQUE nouvelle édition revue augmentée et corrigée Les éditions de ll''université Mentourii Constantiine 2002-2003.

munis de picots ou de cames, pour évoluer vers les disques, rubans et cartes perforées.

Il est difficile de dire à quand remonte ce concept de programme, et en particulier s'il existait sur les automates de l'antiquité. Les premières applications connues avec certitude datent du XVI° siècle avec les horloges à personnages intervenant dans des scènes complexes et variables avec le temps : les Jacquemarts. Cette technique conduisit tout naturellement aux boîtes à musique (XVII°S), et aux automates musiciens et dessinateurs (XVIII°S), dont certains exemplaires sont encore conservés dans des musées. Les techniques introduites et leurs inventeurs ont largement participé à la mécanisation et l'automatisation de l'industrie, en particulier dans l'industrie textile on a vu apparaître des métiers à tisser entièrement automatiques.

Le XVIII° siècle a ainsi vu démarrer le processus de mécanisation de l'industrie, qui passa d'un stade artisanal à un stade de forte concentration en moyens physiques et humains, l'amélioration de la productivité restant un souci constant.

Malgré toutes les découvertes et inventions du XVIII° siècle à nos jours, les deux grandes étapes du processus d'automatisation furent d'abord l'utilisation de la puissance mécanique (avec d'abord la machine à vapeur, puis plus tard les machines électriques), puis très récemment l'introduction de l'informatique dans les machines (avec toutes ses possibilités de stockage de l'information, et surtout de traitement de signaux variés).

1.2 Introduction

La robotique en née, de la plume du romancier, Tchèque Karel Capek en 1921, le mot robot, été introduit dans langue courant, en l'adaptant du mot tchèque : robota (travailleur terme).

Dans sa pièce de théâtre Rossum's universel, les robots remplacent les ouvriers dans les usines, tandis que l'héroïne essaie, de sauver ces machine, moment parfois un supplément d'âmes.

Le problème du travail mécanique et de l'intelligence artificielle est aussi lancé, avant même la construction des premières machines. Plus le temps passe, plus elles ont de leur propre moyen d'analyse.

1.3 Définition

Un robot est un dispositif mécanique poly-articulé mus par des actionneurs et commandé par un contrôleur (calculateur) accomplissant automatiquement une grande variété des tâches qui sont généralement considérées comme dangereuses, pénibles, répétitives et impossibles pour les humains ou dans un but d'une plus grande efficacité. Le terme robot

vient du Tchèque robota, IL veut dire le travail forcé.

1.4 Vocabulaires reliés à la robotique

Le robot est une machine dont le mécanisme est généralement composé d'une série de segments, articulés ou coulissants l'un par rapport à l'autre, ayant pour but de saisir et/ou de déplacer des objets (pièces ou outils) généralement suivant plusieurs degrés de liberté. Un manipulateur peut être commandé par un opérateur, un automate programmable ou tout système logique (par exemple système à cames ou logique câblée). Un manipulateur n'inclut pas de terminal…voire figure 1. 1.

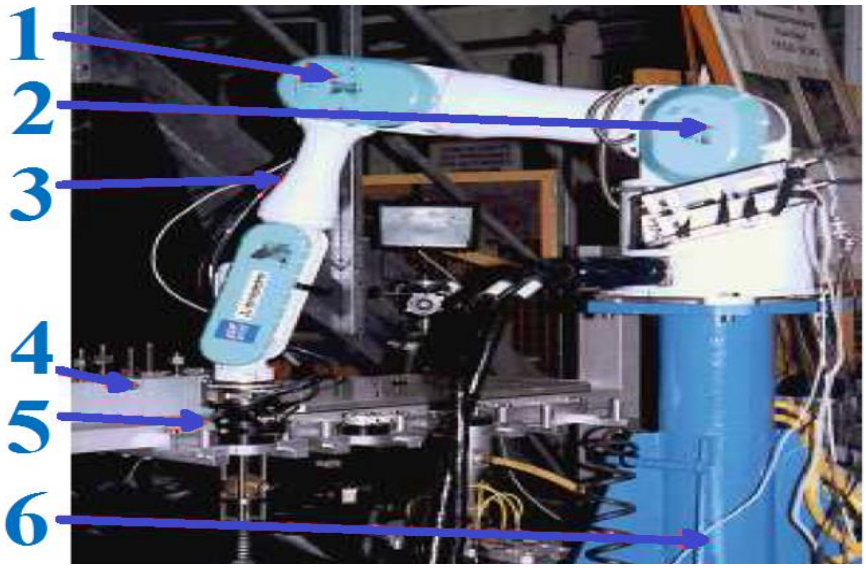


Figure 1:1 Vocabularies du robot

Table 1-1-1 Vocabularies du robot

N°	Vocabulaires	Synonymes	Chaînons/Couples
1	Actionneur	Moteur	Chaînon fixe (n_0)
2	Axe	Articulation	Couple cinématique
3	Corps	Segment	Chaînon mobile [menant] (n-1)
4	Organe terminal	Main	Chaînon mobile [mené] (n)
5	Effecteur	Outil	Chaînon mobile [mené] (n+1)
6	Base	Bâtis	Chaînon fixe (n_0)

***** Google/search ? **© Cours de Robotique, Université de Bouira & © Contrôle de bras manipulateur, Univ-Biskra, 2010.**

On distingue classiquement 4 parties principales dans un robot manipulateur

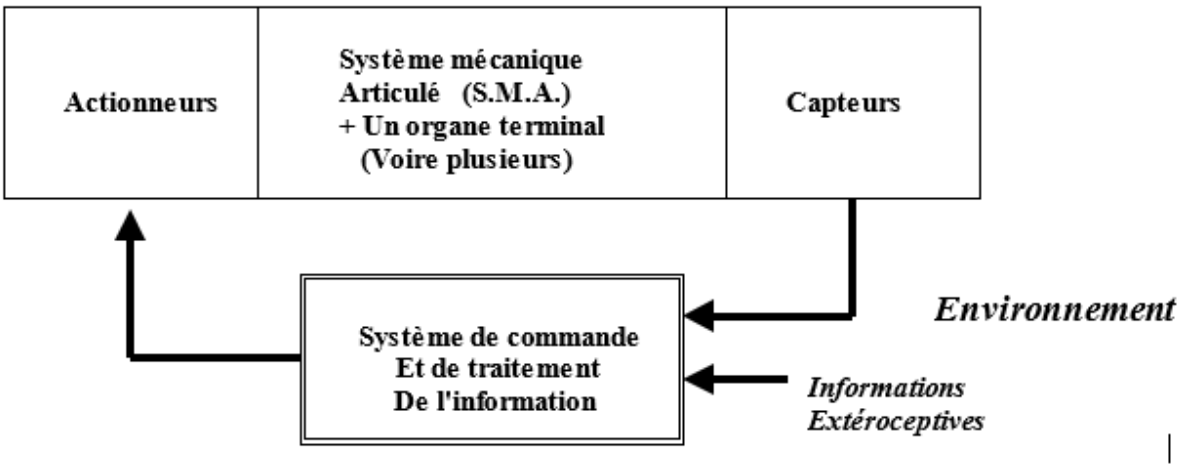


Figure 1:2 Parties principales dans un robot

1.4.1 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 multi-fonctionnel, au sens où il peut être équipé de plusieurs dispositifs ayant des fonctionnalités différentes. Il peut aussi être mono-fonctionnel, 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.

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

1.4.3 Actionneurs

Pour être animé, 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). Il existe trois types d'actionneurs à savoir : actionneurs hydrauliques, actionneurs pneumatiques et actionneurs électriques :

1.4.3.1 Actionneurs hydrauliques

Les actionneurs hydrauliques ce sont des systèmes qui transforment l'énergie hydraulique (l'huile) en énergie mécanique. Il existe deux types d'actionneurs hydrauliques les vérins doubles effets et simples effets pour les mouvements de translation et les moteurs pour des mouvements rotatifs.

Figure 1:3 Vérin hydraulique simple effet

Figure 1:4 Vérin hydraulique double effet.

1.4.3.2 Actionneurs pneumatiques

Tous les systèmes qui transforment l'énergie pneumatique (l'air) en énergie

mécanique sont des actionneurs pneumatiques. En robotique on parle généralement de vérins double et simple effet, qui sont considérés comme des actionneurs linéaires, et commandés en tout ou rien pour des automatismes de transfert.

Figure 1:5 Vérin pneumatique simple effet

Figure 1:6 Vérin pneumatique double effet

1.4.3.3 Actionneurs électriques

La plus part des robots proposés par des constructeurs sont à actionnement électrique, en raison aux nombreux avantages que présentent les actionneurs électriques. Il existe plusieurs types de ces derniers parmi eux : moteur à courant continu, moteur pas à pas, servomoteurs…etc.

Figure 1:7 Servomoteur.

1.4.4 Capteurs

La perception permet de gérer les relations entre le robot et son environnement. Les organes de perception sont des 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.

En robotique on utilise généralement deux types de capteurs :

1.4.4.1 Capteurs proprioceptifs

Ces capteurs mesurent l'état du robot lui-même (capteur de position (GPS), capteur de vitesse, capteur de charge de batteries,...)

Figure 1:8 Capteur de vitesse

Figure 1:9 Capteur de position

1.4.4.2 Capteurs extéroceptifs

Ces capteurs renseignent sur l'état de l'environnement (capteur de température, télémètre (RADAR, LIDAR), boussole, détecteur de chaleur/lumière, ...)

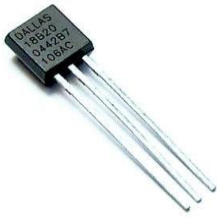


Figure 1:10 Capteur de lumière

Figure 1:11 Capteur de température

1.5 Catégories de robots industriels

Il existe trois types de robots : robots mobiles et robots humanoïdes et robots manipulateurs.

1.5.1 Robots mobiles

Même si l'industrie a besoin de mobilité, elle résout facilement ce problème avec les véhicules filoguidés ou sur rails etc., les conditions d'exploitation (atelier flexible par

exemple) permettant un aménagement de l'espace. Aussi, on ne saurait y parler de véritables robots mobiles, lesquels sont caractérisés par la recherche de trajectoires variables et non définies à l'avance, posant ainsi des problèmes bien plus complexes que le robot à poste fixe. En effet, ce dernier se trouve dans un environnement a priori figé, avec de faibles variations correspondant soit à la succession des opérations d'exécution de la tâche soit à des perturbations limitées. On peut instrumenter cet environnement et/ou l'aménager pour limiter ou supprimer l'information qu'y doit prélever le robot pour exécuter sa tâche, d'où la relative simplicité de son système de commande. Par ailleurs ce robot est accessible à l'homme qui l'a « sous la main » pour le contrôler, le surveiller ou le réparer.

À l'opposé, le robot mobile, par nature, se caractérise par son éloignement de l'homme et par un environnement en permanence évolutif, ce qui engendre deux classes de problèmes dont les solutions efficaces en mode automatique n'existent encore que dans des cas particuliers fortement contraints (ou fortement simplifiés par rapport au cas général).

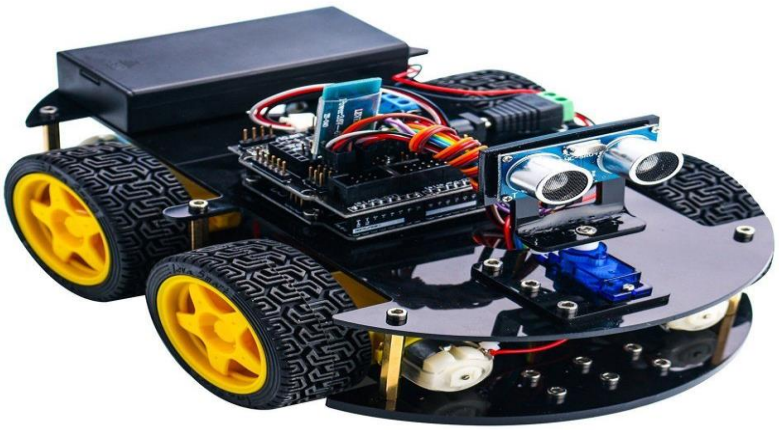


Figure 1:12 Robot mobile

1.5.2 Robots humanoïdes

Un robot humanoïde est un robot dont l'apparence générale rappelle celle d'un corps humain. Généralement, les robots humanoïdes ont un torse avec une tête, deux bras et deux jambes, bien que certains modèles ne représentent qu'une partie du corps, par exemple à partir de la taille. Certains robots humanoïdes peuvent avoir un « visage », avec des « yeux » et une « bouche ».

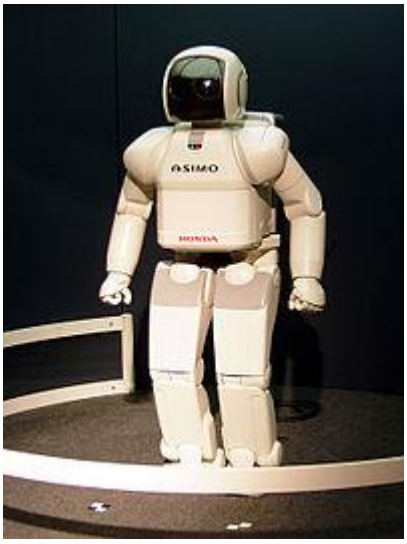


Figure 1:13 Robot humanoïde ASIMO présenté par Honda en 2005.

1.5.3 Robots manipulateurs

Selon la RIA (Robot Institute of America) c'est un manipulateur qui doit être reprogrammable multifonctionnel conçu pour déplacer des matériaux, des pièces, des outils ou tout autre dispositif spécialisé au moyen d'une série de mouvements programmés et d'accomplir une variété d'autres tâches. L'ISO (International Standard Organisation) l'a défini comme étant une machine mue par un mécanisme incluant plusieurs degrés de libertés, ayant souvent l'apparence d'un ou plusieurs bras se terminant par un poignet capable de tenir des outils, des pièces ou un dispositif d'inspection .

Figure 1:14 Robot manipulateur kuka, CE

1.6 Différents types de robots

1.6.1 Robots SCARA

SCARA : Sélective Compliance Articulated Robot for Assembly

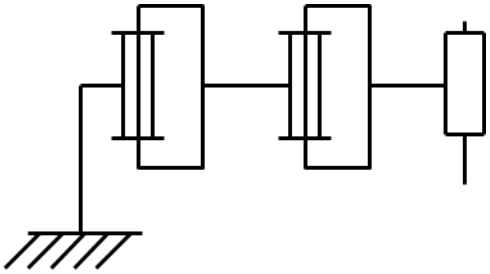


Figure 1:15 Schéma de robot SCARA

Caractéristiques :

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

Exemples :

Figure 1:16 Sankyo RS60

Figure 1:17 IAI série IX

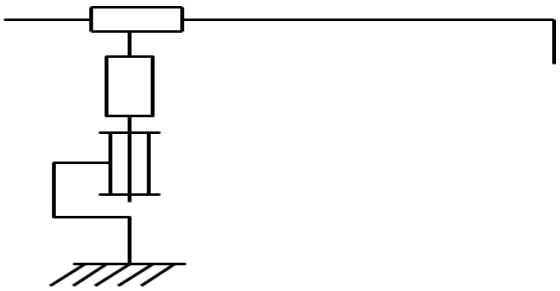


Figure 1:18 Robot cylindrique

Caractéristiques :

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

- Très rapide.

Exemple :

Figure 1:20 Denso CS4130A

Figure 1:19 ST- Robotics R19E

1.6.2 Robots sphériques

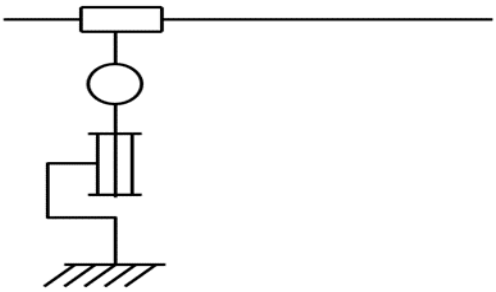


Figure 1:21 Robot sphérique

Caractéristiques :

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

Exemple :

Figure 1:23 Unimate 2000

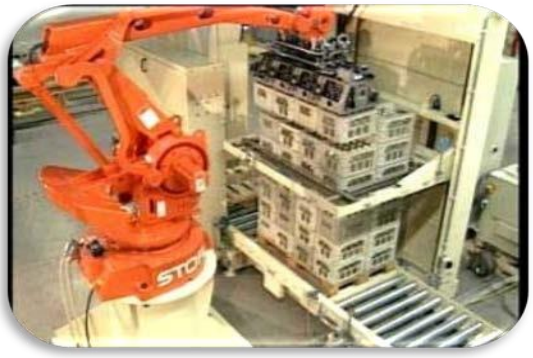


Figure 1:22 CSI C400i

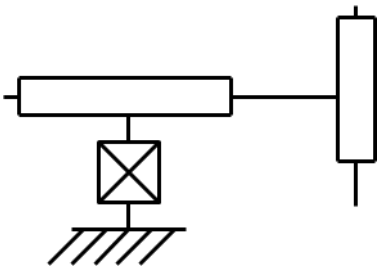


Figure 1:24 Robot cartésiens

1.6.3 Robots Cartésiens

Caractéristiques :

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

Exemple :

Figure 1:25 Fisnar FN9300N

Figure 1:26 Competella Spider

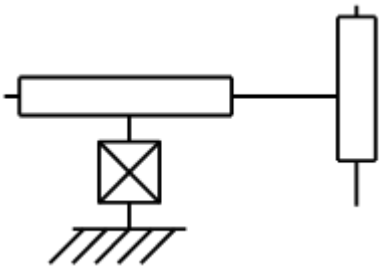


Figure 1:27 Robot cartésiens

Caractéristiques :

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

Exemple :

Figure 1:28 Fisnar FN9300N

Figure 1:29 Competella Spider

1.6.5 Robots parallèles

Caractéristiques :

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

Exemple :

Figure 1:31 ABB Flexpicker

Figure 1:30 Simulateur de vol CAE

1.6.6 Robots Anthropomorphes

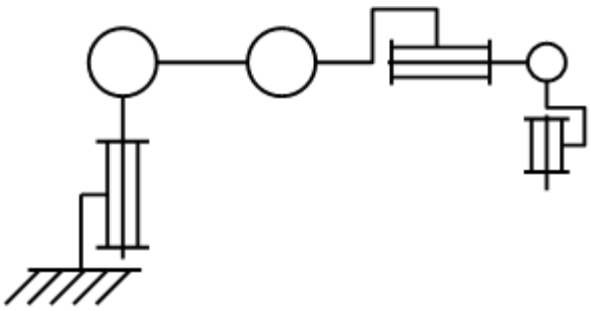


Figure 1:32 Robot Anthropomorphe.

Caractéristiques :

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

Exemples :

- Architecture standard :

Figure 1:33 Kawasaki FS03N

- Architecture à parallélogramm

Figure 1:34 Kuka KR SIXX R650

1.7 Utilisation des robots

1.7.1 Tâches simples :

La grande majorité des robots est utilisée pour des tâches simples et répétitives.

Les robots 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.
- Pénibilté de la tâche (peinture, charge lourde, environnement hostile, ...).

Exemple :

- **Robots Soudeurs**

Figure 1:35 Robot soudeur par point

Figure 1:36 : Robot soudeur l'arc

1.7.2 Tâches complexes

- **Robotique médicale**

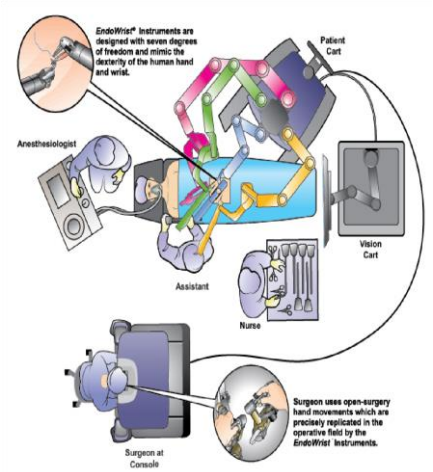


Figure 1:37 Intuitive Surgical Da Vinci.

1.8 Caractérisation d'un robot

1.8.1 Articulations

On admettra qu'un corps solide rigide isolé dans l'espace à trois dimensions possède six degrés de liberté : trois composantes d'un vecteur position et trois composantes d'orientation. Les divers formalismes pratiques utilisés pour quantifier ces degrés de liberté. Quand on relie un tel corps à un autre, au moyen de « liaisons » mécaniques, il perd de sa mobilité par rapport au second. On peut imaginer diverses combinaisons de translations et de rotations, dont quelques-unes sont représentées. Un robot est un système mécanique poly-articulé. On distingue principalement deux types d'articulations :

- Articulation prismatique, schématisée par :

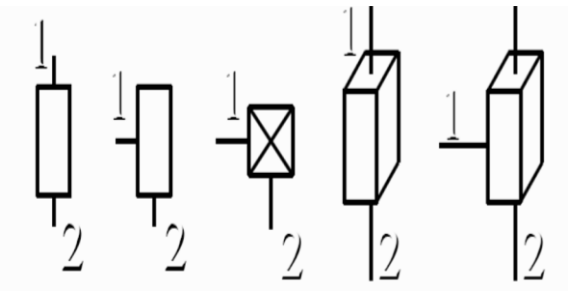


Figure 1:38 Articulation prismatique

- Articulation rotoïde, schématisée par :

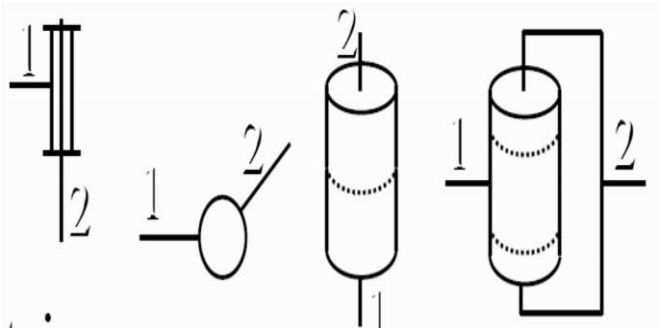
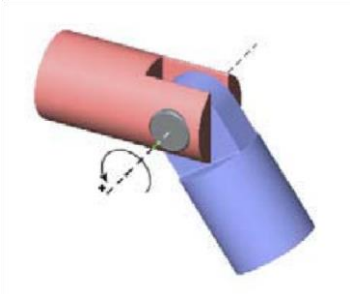
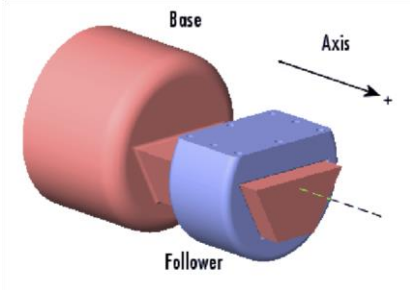


Figure 1:39 Articulation rotoïde

- Les différents types d'articulations les plus utilisées :



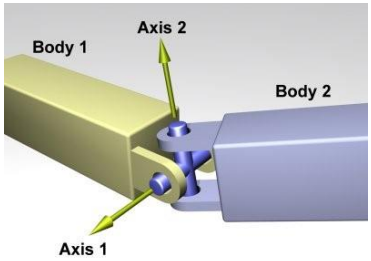
Rotoïde (R)

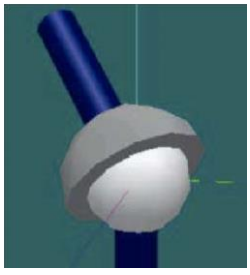


Prismatique (P)

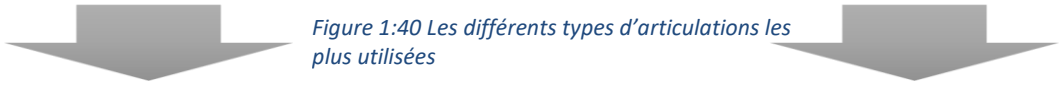
1 DDL (Rotation)

1 DDL (Translation)





Universelle (u) Sphérique (s)



2 DDL (rotations) 3 DDL (rotations)

Figure 1:40 Les différents types d'articulations les plus utilisées

1.8.2 Caractéristiques géométriques

- Nombre d'axes (mus par un actionneur)
- Architecture (série ou parallèle)
- Chaînage des articulations
- Nombre de degrés de liberté

Exemples :

- 3 axes, série, RRR, 3DL.
- 3 axes, série, PPP, 3DL.

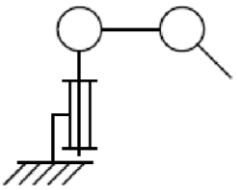


Figure 1:41 3 axes, série, RRR, 3DL

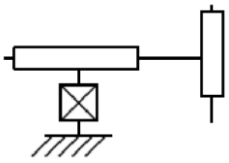


Figure 1:42 3 axes, série, PPP, 3DL

1.8.3 Espace de travail

Volume accessible par l'outil du robot.

Ce volume dépend :

- de la géométrie du robot.
- de la longueur des segments.
- du débattement des articulations (limité par des butées).

Exemples :

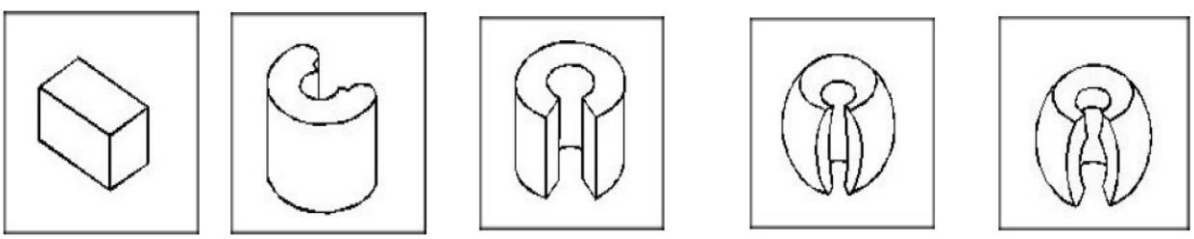


Figure 1:43 Volume accessible par l'outil du robot

1.8.4 Précision / Répétabilité

Positionnement absolu imprécis (>1 mm) :

- Erreurs de modèle géométrique.
- Erreurs de quantification de la mesure de position.
- Flexibilités.

Répétabilité : la répétabilité d'un robot est l'erreur maximale de positionnement répété de l'outil en tout point de son espace de travail

En général, la répétabilité < 0.1 mm.

1.8.5 Performances dynamiques

1.8.5.1 Vitesse maximale

- Vitesse maximale de translation ou de rotation de chaque axe
- Les constructeurs donnent souvent une vitesse de translation maximale de l'organe terminal

1.8.5.2 Accélération maximale

- Est donnée pour chaque axe dans la configuration la plus défavorable (inertie maximale, charge maximale)
- Dépend fortement de l'inertie donc de la position du robot.
- C'est la charge maximale que peut porter le robot sans dégrader la répétabilité et les performances dynamiques
- La charge utile est nettement inférieure à la charge maximale que peut porter le robot qui est directement dépendante des actionneurs.

1.8.5.3 Morphologie

La structure mécanique d'un robot manipulateur est composée de plusieurs corps Connectés les uns aux autres par des liaisons appelées articulations ou joints, à un seul d.d.l de translation ou de rotation. Cette structure mécanique peut constituer une Chaîne cinématique continue ouverte simple, une chaîne arborescente ou une chaîne Complexe.

1.9 Notions géométriques d'un manipulateur

1.9.1 Solides

Un solide S est dit indéformable si, pour toute paire de points de ce solide de Coordonné m et n, on a $||m(t) - n(t)|| = ||m(0) - n(0)|| =$ constante au cours du temps.

1.9.2 Degré de liberté (D.D.L) :

Le nombre de *d.d.l.* d'un mécanisme est le nombre de paramètres *indépendants* qui permettent de définir la position du *mécanisme* à un instant donné du mouvement.

1.9.3 Liaison

Une liaison entre deux solides indéformables limite le D.D.L d'un solide par rapport à l'autre. On appelle D.D.L e la liaison le nombre de paramètres indépendants Permettant de définir la localisation (position et orientation) d'un solide par rapport à L'autre dans tout déplacement.

Exemples :

- Un cube sur un plan a 3 d.d.l. : 2 pour fixer les coordonnées d'un point dans le plan, 1 pour déterminer son orientation dans le plan.
- Une sphère sur un plan a 5 d.d.l. : 2 pour fixer les coordonnées d'un point dans le plan, 3 pour déterminer son orientation dans le plan. - Une porte par rapport au mur a 1 d.d.l.

1.9.4 Mécanismes

On appelle mécanisme un ensemble de solides reliés 2 à 2 par des liaisons. On distingue 2 types de mécanismes :

- Les *mécanismes en chaîne simple ouverte* (ou *en série*). Lorsque l'on parcourt le mécanisme, on ne repasse jamais 2 fois sur la même liaison, ou sur le même solide. Ce type de système est le plus répandu
- Les mécanismes en chaîne complexe, i.e., tout ce qui n'est pas en série (au moins un solide avec plus de 2 liaisons). De tels systèmes se subdivisent en 2 groupes : les chaînes structurées en arbre, et les chaînes fermées (dont l'avantage est d'être a priori plus rigide, plus précis, capable de manipuler de lourdes charges). A titre d'exemple, le pantographe est un mécanisme en chaîne fermée.

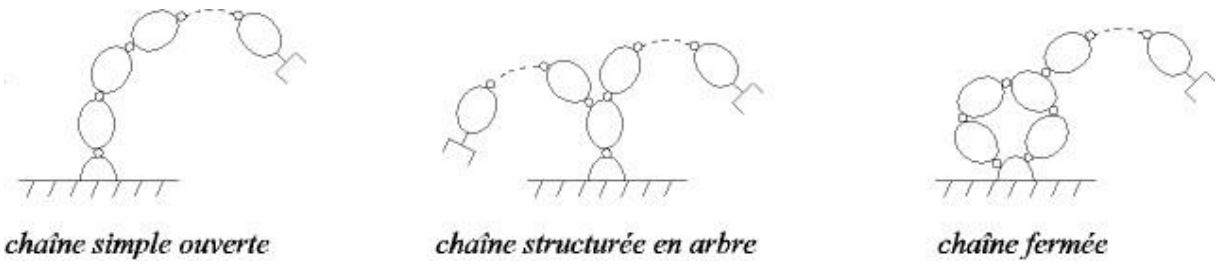


Figure 1:44 Diffèrent type de mécanismes

1.9.5 Morphologie des porteurs

La structure d'un porteur est caractérisée par des corps rigides, les segments susceptibles de se mouvoir par rapport à une base et par des articulations limitant le mouvement relatif entre deux segments adjacents. On désigne fréquemment les bras manipulateurs en accolant les lettres R (pour rotoïdes) et P (pour prismatique) pour décrire la succession des liaisons. Les porteurs sont souvent classés, du point de vue de la structure, suivant le système de coordonnées dans lequel ils opèrent. On définit ainsi cinq types de manipulateur plus utilisés :

1.9.5.1 Robots cartésiens (ppp)

Un manipulateur dont les trois premières articulations sont prismatiques est connu sous le nom de manipulateur cartésien. Il est montre dans la figure pour le manipulateur cartésien les variables articulaires sont les cartésiennes de l'organe terminal on respectant la base. Comme on peut le deviner la description cinématique de ce manipulateur est la plus simple de toutes les configurations.

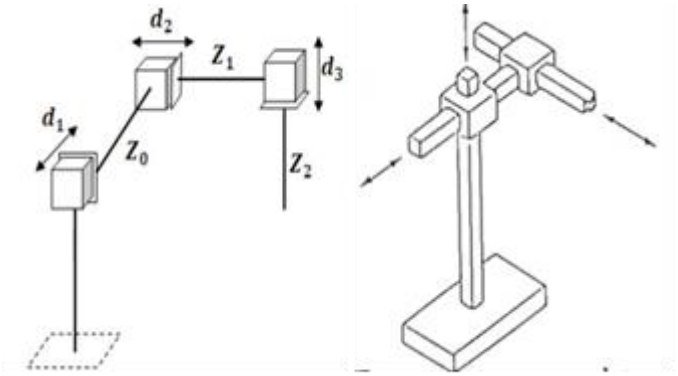


Figure 1:45 Exemple de configuration cartésienne (ppp).

1.9.5.2 Robots cylindriques (RPP)

La configuration cylindrique est montrée dans la figure, la première articulation est rotoide et produit une rotation autour de la base, alors que la seconde et la troisième sont prismatiques. Comme le nom le suggère, les variables articulaires sont les coordonnées cylindriques de l'organe terminal en prenant compte de la base.

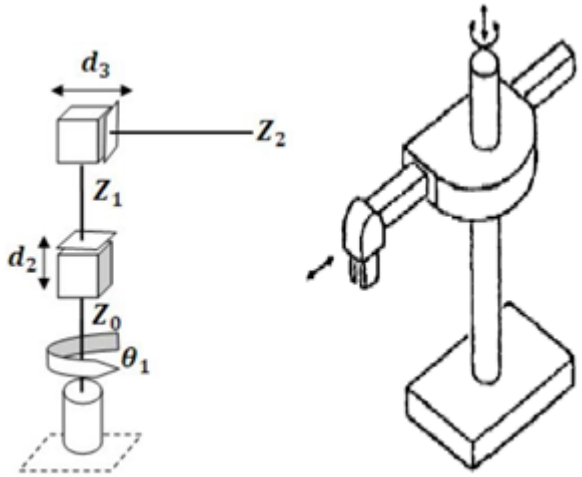


Figure 1:46 : Exemple de configuration cylindrique (RPP).

1.9.5.3 Robots sphériques (RRP)

Les porteurs travaillant en coordonnées sphériques ont une structure de base comportant deux rotations et une translation ; il s'agit alors d'un bras télescopique pouvant effectuer une rotation autour d'un axe vertical et une rotation autour d'un axe horizontal, ce qui donne à l'espace de travail du porteur la forme d'une coquille sphérique. Comme pour les porteurs cylindriques, l'existence des rotations réduit considérablement la résolution en bout de bras à cause de l'amplification des erreurs angulaires apparaissant au niveau des axes de rotation néanmoins, l'avantage d'une telle structure est qu'elle améliore la flexibilité dans l'utilisation.

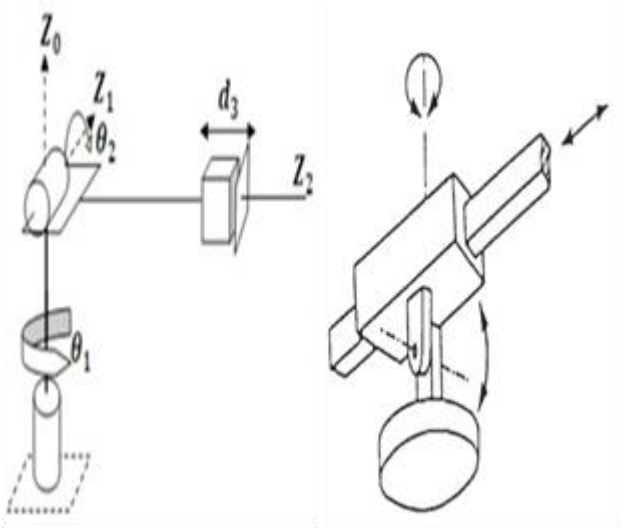


Figure 1:47 Exemple de configuration sphérique (RRP).

1.9.5.4 Robots articulés (RRR)

Les robots articules ont une structure qui rappelle la plupart du temps celle d'un bras humain. En général, ils comprennent quatre segments principaux que l'on peut assimiler

au tronc, au bras, a l'avant-bras et a la main ; toutes les articulations du porteur articule sont donc de type rotoides et vont souvent correspondre à l'épaule, au coude et au poignet. De ce fait, la résolution en bout de bras du porteur articule, quoiqu'elle dépende de la position de travail, est toujours très mauvaise. Cependant, de toutes les structures, la structure articulée est celle qui offre la meilleure flexibilité et permet une grande souplesse dans l'exécution des taches, d'où son utilisation fréquente dans les systèmes robotises de petite et moyenne envergure. De plus, grâce aux progrès technologiques en matière d'asservissement, les robots articules peuvent être utilisés pour des applications nécessitant une très grande précision.

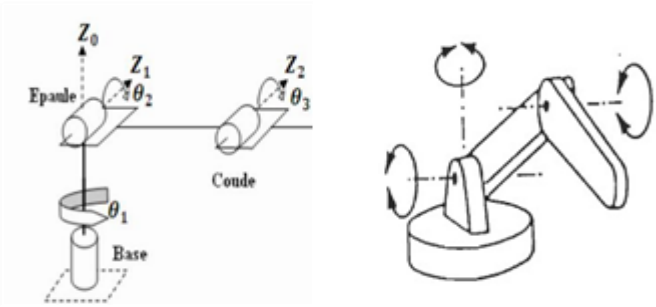


Figure 1:48 Exemple de configuration articulée (RRR).

1.10 Classification des manipulateurs

L'AFRI (Association Française de la Robotique Industrielle) distingue quatre classes de robots manipulateurs selon leur niveau d'autonomie :

1.10.1 Télémanipulateurs ou manipulateurs à commande manuelle

Ils sont commandés à distance et "en temps réel" par un opérateur humain. Cette télécommande se fait à plus ou moins longue distance par signaux mécaniques, hydrauliques, ou le plus souvent électriques. Ces manipulateurs sont employés en forge, fonderie, meulage-ébarbage, milieux "hostiles", etc., mais nécessitent toujours la présence et l'intervention constante d'un opérateur. Appelés aussi robots télécommandes ils ont aucune autonomie.

Figure 1:49 Manipulateur à commande manuelle

1.10.2 Manipulateurs automatiques à cycles préréglés

Leurs mouvements sont limités par des butées et cames réglables à la main. Ils sont commandés à l'aide de logiques à relais ou pneumatiques (séquences fixes), ou par automates programmables et cartes à microprocesseurs (séquences variables). Généralement modulaires, ces appareils sont conçus pour une application déterminée.

Figure 1:50 Manipulateur à cycle préréglé.

1.10.3 Robots programmables

Ils sont pilotés par des ordinateurs ou des armoires de commande numérique. Leurs mouvements continus dans l'espace sont alors programmés par apprentissage ou en langage symbolique par l'intermédiaire d'un clavier, ou encore sur l'écran d'un poste de CAO. Ils assurent des manipulations complexes, des opérations de soudage, usinage, découpe, peinture et pulvérisation, etc.

Figure 1:51 Robot programmable.

1.10.4 Robots dits "intelligents"

Equipés de capteurs (par exemple un système de vision artificielle ou de suivi de joint en soudage), ils peuvent analyser les modifications de leur environnement ou de leur trajectoire et réagir en conséquence. Ces machines appelées robots de "deuxième génération" commencent à être répandus dans l'industrie. La "troisième génération" disposant

de capacités de raisonnement grâce à l'intelligence artificielle qui fait aujourd'hui l'objet de recherches approfondies.

Figure 1:52 Robot intelligent.

1.10.5 Pinces

Elles sont largement utilisées dans l'industrie manufacturière. Leur forme dépend de l'élément à saisir et l'environnement dans lequel elles agissent. Elles sont généralement composées (voir figure Ⅰ.54) de :

1 : Corps de la pince
2 : Mâchoires
3 : Mors
4 : Rainure de capteur
5 : F = force de serrage sur un doigt seulement
6 : l'élément à saisir
7 : L = distance entre le centre de gravité de la charge et la surface de référence
8 : C = coursehj d'un doigt

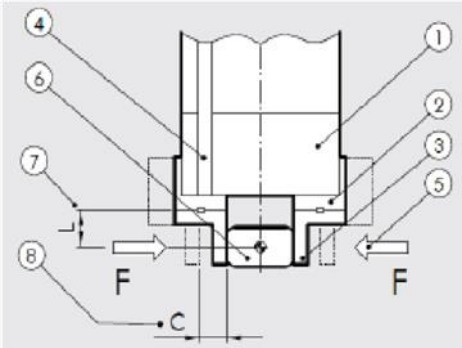
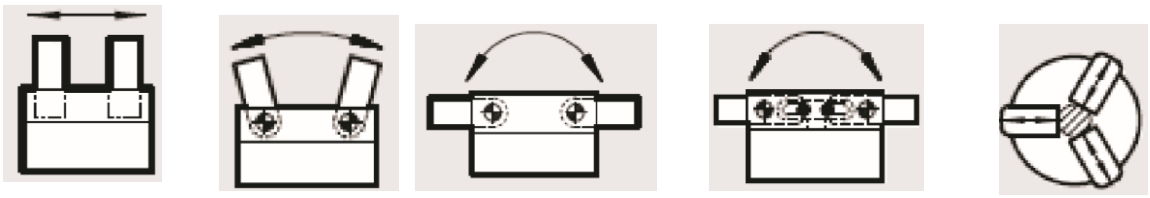


Figure 1:53 Nomenclature d'une pince

1.10.6 Type de pinces

On peut les classer en fonction du type de déplacement des doigts.



(a) (b) (c) (d) (e)

Figure 1:54 : Type de pinces

- **Pinces parallèles (figure1.55 (a)) :** Les doigts ont un déplacement linéaire. Ils peuvent être au nombre de 2, 3 ou parfois 4.
- **Pinces angulaires (figure 1.55(b)) :** Les doigts sont articulés et décrivent un mouvement en arc de cercle. Elles ont, en général un coût plus réduit que les pinces parallèles, mais avec quelques limitations (Figure 1.56. Avec ce type de pinces :

- Si la pièce a des dimensions variables, la surface de contact varie.
- Si la pièce est cylindrique avec des dimensions variables, la position de l'axe de la pièce varie.

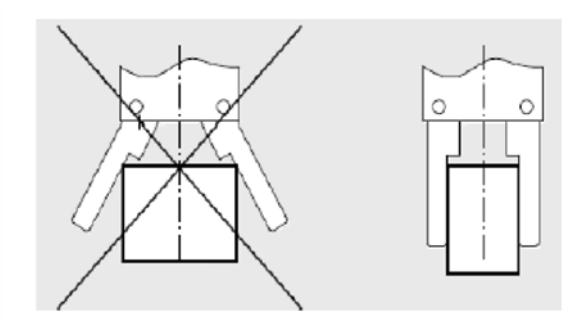


Figure 1:55 Limites d'utilisation des pinces à ouverture angulaire

1.10.6.1 Pinces angulaires avec doigts rétractables (figure Ⅰ.55 (c))

Les doigts ont un angle d'ouverture d'environ 90°. Les doigts de serrage peuvent s'effacer complètement sur la surface supérieure de la pince, permettant ainsi dans certains cas, d'éviter un mouvement linéaire de reculement.

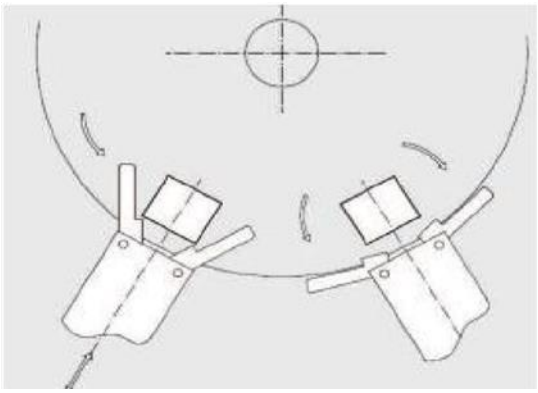


Figure 1:56 Pinces à ouverture angulaire.

1.10.6.2 Pinces à genouillère (figure 1.55 (d))

Pinces angulaires avec un mécanisme à genouillère, assurant une grande force de serrage. Le serrage est irréversible même lorsqu'il n'y a plus de pression, aussi la pièce ne peut être relâchée accidentellement. L'angle d'ouverture est de 90°, aussi elle agit comme une pince à doigts rétractables. La force de serrage n'est importante que sur une plage d'angle de rotation limitée.

1.10.6.3 Pince à serrage concentrique(e) (intérieur ou extérieur)

Les pinces à deux doigts sont utilisées pour les pièces de forme prismatique, ou cylindrique à un seul diamètre. Les pinces à trois doigts peuvent être utilisées sur des pièces cylindriques à différents diamètres.Les pinces parallèles, concentrique et angulaire peuvent travailler dans les deux sens pourréaliser des serrages sur l'extérieur de la pièce ou en intérieur, dans un alésage par exemple (Figure1.58)

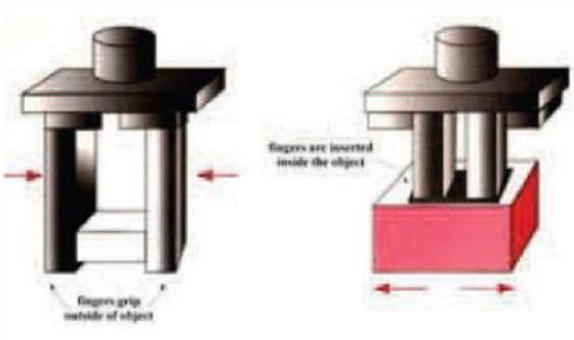


Figure 1:57 Serrage intérieur et extérieur

La pince angulaire à ouverture totale (180°), quant à elle, elle ne fonctionne qu'en serrage externe. En effet, elle a été conçue pour dégager totalement les doigts vers l'extérieur, afin de supprimer un mouvement de dégagement au manipulateur (Figure1.59).

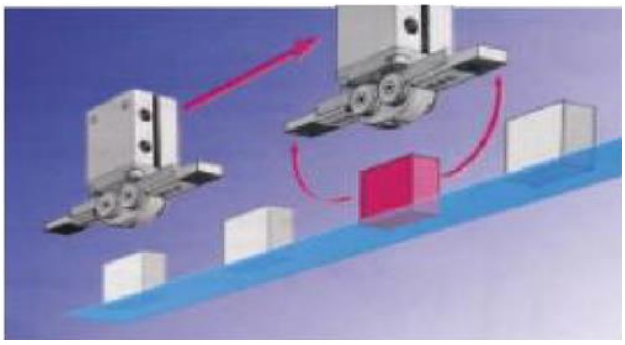


Figure 1:58 Dégagement complet des doigts de la pince à ouverture totale.

1.10.7 Dimensionnement des pinces

Il se fait en fonction des données du cahier des charges de l'application, qui sont :

- Fonction opérative de la pince ;
- Caractéristiques : forme, masse, matériau de la pièce à saisir ;
- Trajectoires et accélérations de la pince ;
- Course totale d'ouverture ;
- Temps d'action requis.

Le dimensionnement permettra de définir la taille de la pince, c'est-à-dire la force de serrage développée capable d'assurer l'équilibre de toutes les forces statiques et dynamique agissant sur la pièce à manipuler. Il faut ensuite, vérifier la capacité d'absorption de l'énergie cinétique en fin de course du préhenseur choisi. La dernière étape consiste à vérifier que tous les efforts (poids, force d'inertie, effort extérieur…) appliqués sur le guidage en rotation ou en translation des pré-mors sont compatibles avec les charges statique et dynamique maximales admissibles par la pince.

Le maintien d'un objet dans la pince se fait soit :

- **Par obstacle** : où le maintien est basé sur une constriction de l'objet entre les doigts (Figure1.50 (a)), ces derniers entourent l'objet, au moins en partie, et bloquent tout mouvement relatif de l'objet par rapport à la pince (lorsque celle-ci est serrée).

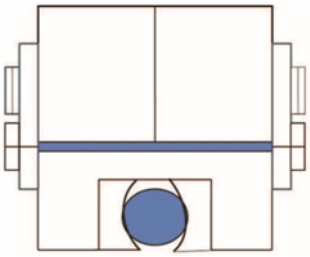


Figure 1:59 : (a) Préhension par obstacle

- **Par adhérence** : dans ce cas, le maintien est basé sur la force de frottement (sec) entre les doigts et l'objet (figure 1.61 (b)).

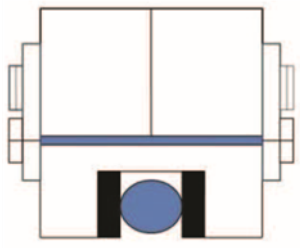


Figure 1:60 : (b) Préhension par adhérence.

II faut donc étudier la forme des doigts, au moins de la partie des doigts qui est en contact avec l'objet, et l'adapter à la forme de l'objet : cavités adaptées à la forme de l'objet, cavités en ***V*** pour pièces cylindriques. Dans le cas d'une pince (rotative) à deux doigts, avec cavités en ***V***, le diamètre maximum des objets que l'on peut saisir et manipuler, en fonction des caractéristiques de la pince est donné par (voir figure) :

$$R_{\max} = a + A'B' \sin\gamma = a + (L\cos\gamma - b)\sin\gamma$$

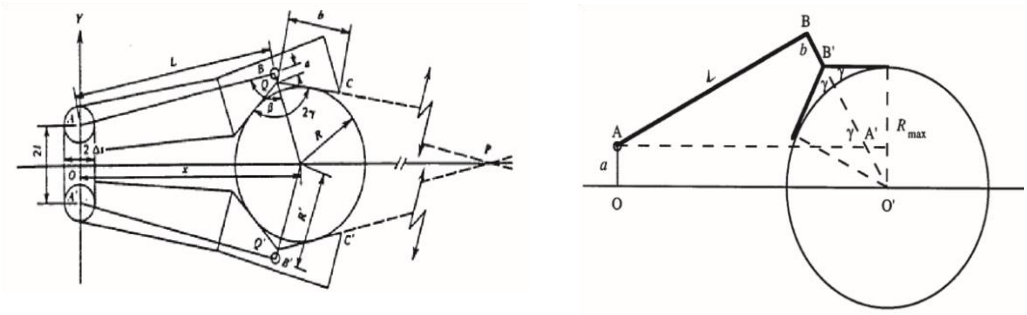
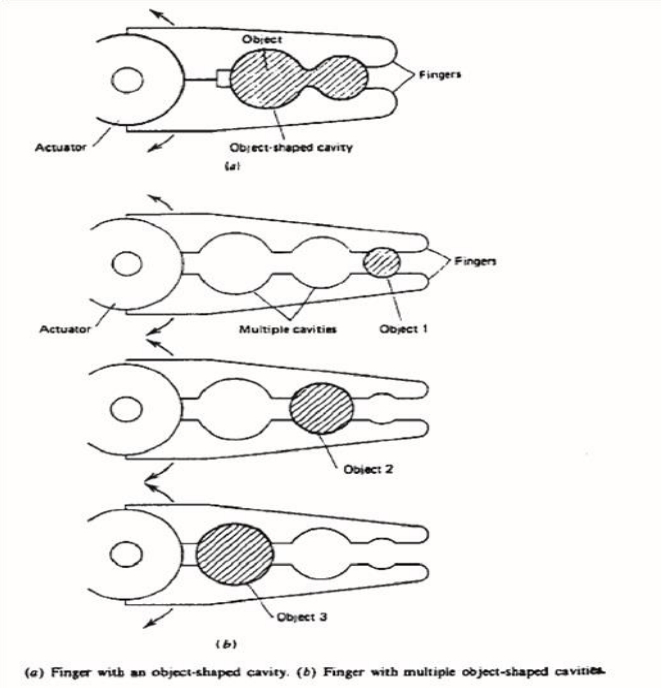


Figure 1:61 : (a) Préhension par constriction universelle. (b) Limite de la constriction.



(a) Finger with an object-shaped cavity. (b) Finger with multiple object-shaped cavities.

Figure 1:62 Préhension par constriction dédicacée

Lorsque l'objet est maintenu par des forces de frottement, il faut que celles-ci soient suffisantes pour retenir l'objet malgré les autres forces qui s'exercent sur lui (pesanteur, inertie, force de réaction due à la tâche exécutée, ...). Souvent, la partie de la pince en contact avec l'objet est faite d'une matière non dure, ce qui accroît le coefficient de frottement et peut aussi protéger la surface de l'objet manipulé de dommages tels que griffes, coups, ... En outre, il en résulte généralement aussi une déformation locale, qui entraîne un contact sur toute une ligne ou une surface (d'où la protection évoquée ci-dessus) et un certain effet de constriction.

1.11 Mécanismes de préhension

L'ouverture ou la fermeture de la pince peut être commandée par :

- un mouvement de rotation ou,
- un mouvement de translation

Le serrage peut être symétrique ou asymétrique ; la réalisation d'un serrage asymétrique est évidemment plus facile, mais ce type de serrage présente un inconvénient : si on veut éviter un déplacement de l'objet saisi, il faut tenir compte du diamètre de celui-ci lors de la programmation de la position de la pince du robot, de façon à ce qu'une des mâchoires de la pince vienne affleurer l'objet à saisir ; par contre, dans le cas où le serrage est symétrique, il suffit de programmer la position du centre de l'objet (quelque soit sa taille).

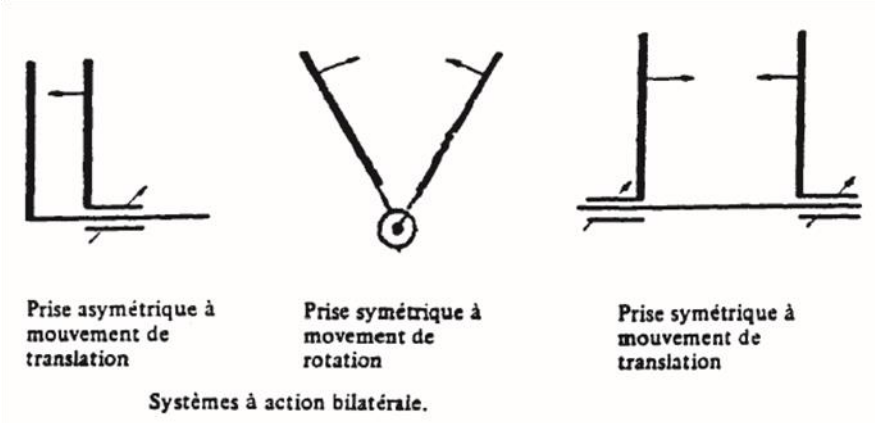


Figure 1:63 : Prises bilatérales

Le dispositif cinématique utilisé pour actionner la pince peut être :

- un mécanisme à plusieurs barres
- un dispositif à came
- un système à pignon crémaillère
- un système vis écrou (dispositif irréversible)
- un système à câbles et poulies, surtout utilisé quand les actionneurs sont placés assez loin des doigts, pour des raisons d'équilibrage, ou dans des organes à prise multilatérale.

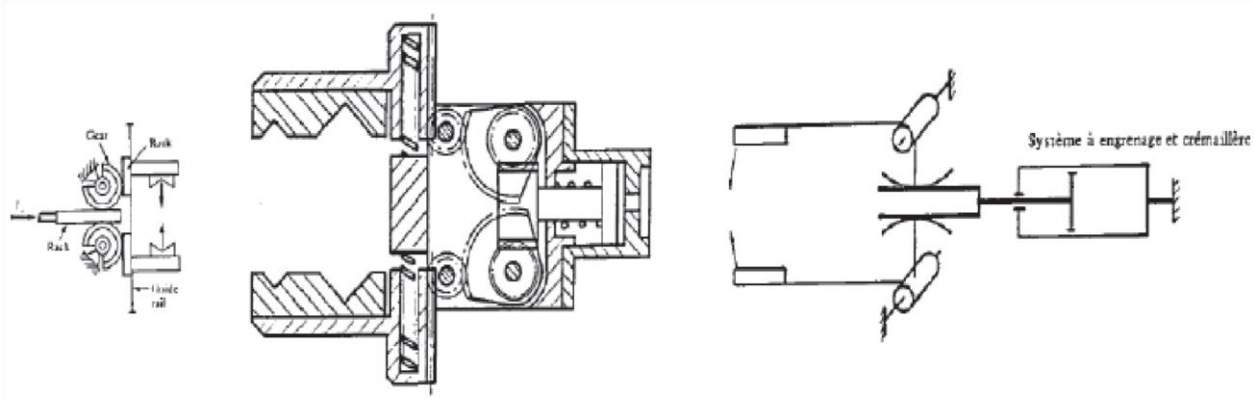


Figure 1:64 Système pignon crémaillère.

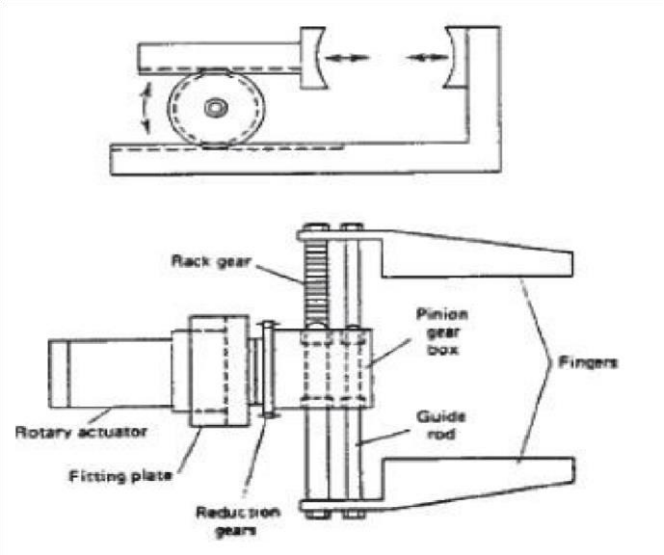


Figure 1:65 Système vis écrou.

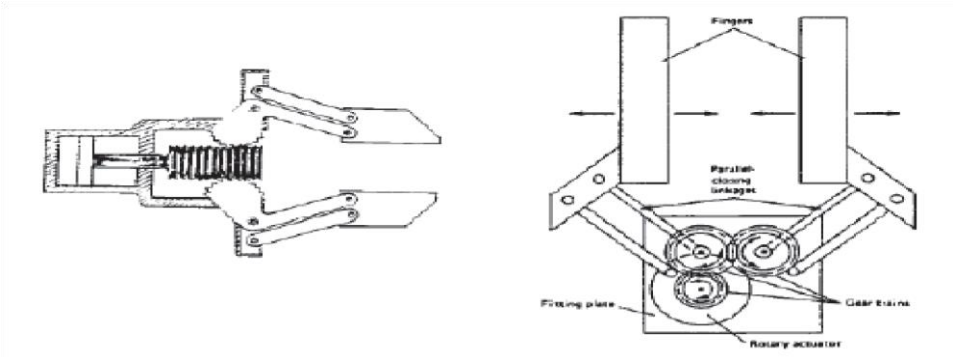


Figure 1:66 Déplacement des deux mâchoires de la pince.

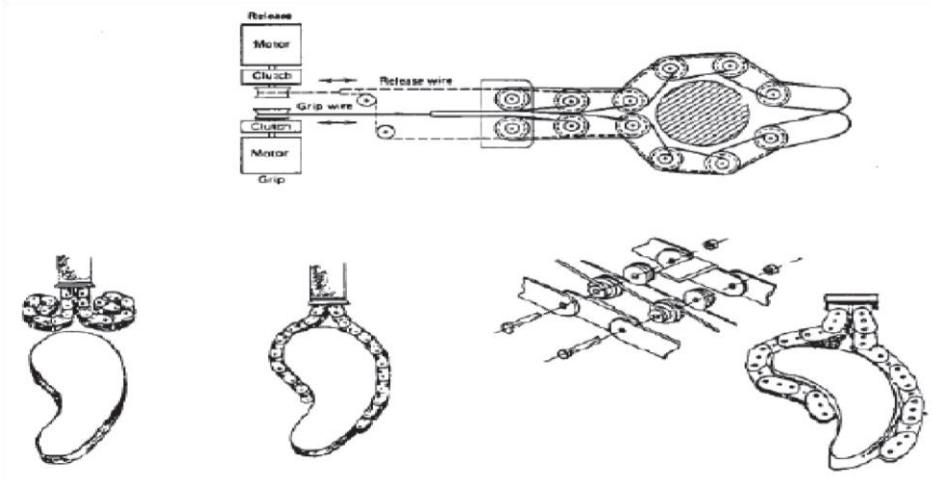


Figure 1:67 Système s'adaptant à la forme de l'objet.

1.12 Conclusion

La robotique est en constante evolution. Depuis ces debuts, chaque grande avancee est considere comme un pas vers le robot du futur. Les robots d'aujourdhui ne sont quasiment que des prototypes, mais comme tout avancee technologique, ils se perfectionnent avec le temps et leurs capacites vont certainement etre au-dessus de nos esperances. Le robot est un manipulateur à plusieurs degrés de liberté, à commande automatique, reprogrammable, multi-applications, mobile ou non, destiné à être utilisé dans les applications d'automatisation industrielle.

Chapitre 2
Modélisation Mathématique des Robots Industriels

2 Chapitre 2 Modélisation Mathématique des Robots Industriels

2.1 Introduction

La modélisation de système polyarticule a pour but de représenter au mieux le robot dans son environnement pour ensuite lui programmer des trajectoires avec la planification de mouvement.

La conception et la commande des robots nécessitent le calcul de certains modèles mathématiques, tels que :

Les modèles de transformation entre l'espace opérationnel (dans lequel est définie la situation de l'organe terminal) et l'espace articulaire (dans lequel est définie la configuration du robot). Parmi ces modèles, on distingue :

Les modèles géométriques direct et inverse qui expriment la situation de l'organe terminal en fonction de la configuration du mécanisme et inversement,

Les modèles cinématiques direct et inverse qui expriment la vitesse de l'organe terminal en fonction de la vitesse articulaire et inversement,

Les modèles dynamiques définissant les équations du mouvement du robot, qui permettent d'établir les relations entre les couples ou forces exercés par les actionneurs et les positions, vitesses et accélérations des articulations.

2.2 Modèle géométrique

Il permet de déterminer la configuration (position, orientation) de l'effecteur d'un robot en fonction de la configuration de ses liaisons, l'effecteur peut être une pince, une caméra, une pompe de peinture. La modélisation des robots de façon systématique et automatique exige une méthode adéquate pour la description de leur morphologie. Plusieurs méthodes et stations ont été proposées [Denavit, 55] [Sheth, 71] [Renaud, 75] [Khalil, 76] [Borrel, 79] [Craig, 86a].

2.3 Transformations homogènes

Soit $\left({}^{i}P_{x}, {}^{i}P_{y}, {}^{i}P_{z}\right)$ les coordonnées cartésiennes d'un point P arbitraire, mesure dans le repère $R_i\left(O_i, x_i, y_i, z_i\right)$, les coordonnées homogène du point P sont $\left(w.{}^{i}P_{x}, w.{}^{i}P_{y}, w.{}^{i}P_{z}, w\right)$ ou w est le facteur d'échelle, dans la robotique w=1. Les coordonnées homogènes du point P sont représentées par le vecteur colonne :

$$ {}^{i}P = \begin{bmatrix} {}^{i}P_{x} \\ {}^{i}P_{y} \\ {}^{i}P_{z} \\ 1 \end{bmatrix} $$

Les coordonnées homogènes d'un plan Q de l'équation :

$$^{i}\alpha x + {}^{i}\beta y + {}^{i}\gamma z + {}^{i}\delta = 0$$

Exprime selon le repère R_i, sont données par le vecteur ligne :

$$^{i}Q = \begin{bmatrix} ^{i}\alpha & ^{i}\beta & ^{i}\gamma & ^{i}\delta \end{bmatrix}$$

Si un point P appartient à un plan Q, le produit matriciel $^{i}Q.^{i}P$ est nul :

$$^{i}Q.^{i}P = \begin{bmatrix} ^{i}\alpha & ^{i}\beta & ^{i}\gamma & ^{i}\delta \end{bmatrix} ^{i}P = \begin{bmatrix} ^{i}P_x \\ ^{i}P_y \\ ^{i}P_z \\ 1 \end{bmatrix} = 0$$

La transformation (translation et/ou rotation) d'un repère R_i au repère R_j est représentée par la matrice de transformation homogène $^{i}T_j$ de dimension (4×4) ou :

$$^{i}T_j = \begin{bmatrix} ^{i}s_j & ^{i}n_j & ^{i}a_j & ^{i}P_j \end{bmatrix} = \begin{bmatrix} s_x & n_x & a_x & P_x \\ s_y & n_y & a_y & P_y \\ s_z & n_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ou, $^{i}s_j, ^{i}n_j, ^{i}a_j$ contiennent les composants des vecteurs unitaires suivant x_j, y_j et z_j respectivement du repère R_j exprimées dans le repère R_i et $^{i}P_j$ représente les coordonnées de l'origine O_j de repère R_j exprimées dans R_i.

On peut dire également que la matrice $^{i}T_j$ définit le repère R_j dans le repère R_i.

La matrice de transformation s'écrit aussi sous la forme :

$$^{i}T_j = \begin{bmatrix} & ^{i}A_j & & ^{i}P_j \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} ^{i}s_j & ^{i}n_j & ^{i}a_j & ^{i}P_j \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Finalement, la matrice de transformation $^{i}T_j$:

- Est considère comme la représentation du repère R_j dans le repère R_i.
- Permet le passage du repère R_i au repère R_j.

Soit Trans (a, b, c) la transformation d'une translation pure ou a, b, et c sont les translations Le long des axes x, y et z respectivement. Quand l'orientation est conservée, la matrice de Transformation de cette translation a la forme suivante :

$$ {}^{i}T_{j} = Trans(a,b,c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

La notation Trans (u, d) indique la translation d'une valeur d le long de l'axe u. Ainsi la matrice Trans (a, b, c) peut être décomposée en produit de trois matrice : Trans(x, a)Trans (y, b) Trans (z, c), par n'importe quel ordre de multiplication.

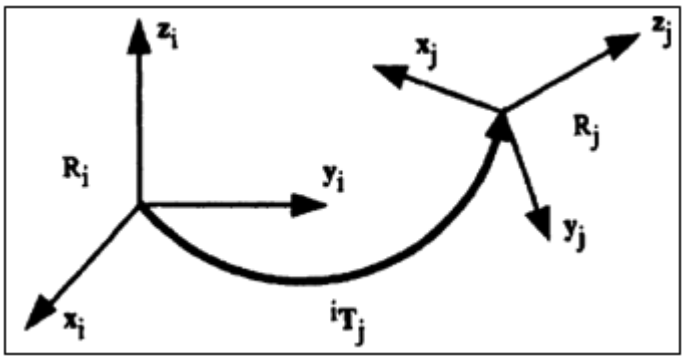


Figure 2:1 Transformation de repere

Soit Rot(x,θ) la transformation correspondante a une rotation pure d'angle θ autour de l'axe x, la matrice de transformation de cette rotation s'ecrit :

$$ {}^{i}T_{j} = Rot(x,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & rot(x,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

ou, rot(x,θ) designant la matrice d'orientation de dimension (3×3).

De la même façon on définit les matrice de transformation des rotations autour des axes y et z.

$$ {}^{i}T_{j} = Rot(y,\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & rot(x,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{i}T_{j} = Rot(z,\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & rot(x,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

2.4 Description de la structure géométrique d'un robot

2.4.1 Description des robots a chaine ouverte simple

Une structure ouverte simple est composée de n+1 corps C_0, C_1, C_2…C_n et de « n » articulations (voir la figure II.1).

- Le corps C_0 désigne la base du robot.
- Le corps C_n est celui qui porte l'organe terminal.
- L'articulation j connecte le corps C_j au corps C_{j-1}.

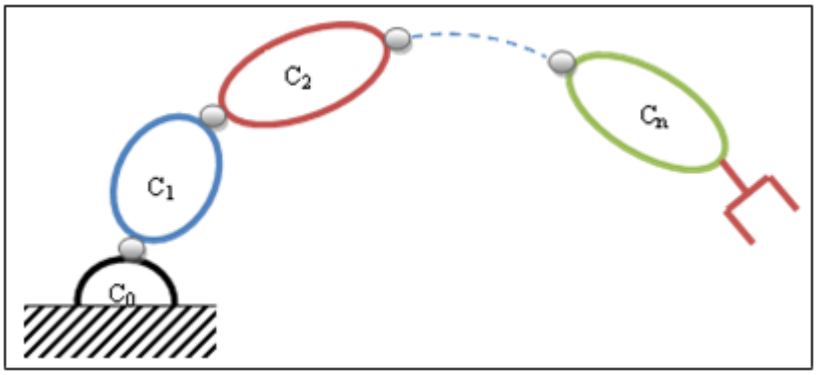


Figure 2:2 : Robot a structure ouverte simple

2.4.2 Paramétrage de Denavit -Hartenberg (DH standards)

Les paramètres de Denavit-Hartenberg (également appelés paramètres DH) sont les quatre paramètres associés à une convention particulière pour fixer les cadres de référence pour les liaisons d'une chaîne cinématique spatiale, ou d'un robot manipulateur.

Les paramètres de Denavit et Hartenberg sont quasi universellement adoptés par les roboticiens pour définir, avec un nombre minimum de paramètres, les matrices de transformations homogènes élémentaires qui permettent de passer du repère associé à un corps du robot au corps qui le suit dans la chaîne cinématique, les corps sont supposés parfaitement rigides et les articulations sont considérées comme idéales.

- L'axe $\mathbf{z_j}$ est porté par l'axe de l'articulation j+1.
- L'axe $\mathbf{x_j}$ est porté par la perpendiculaire commune aux axes $\mathbf{z_j}$ est $\mathbf{z_{j-1}}$ Si les axes $\mathbf{zj}$ et $\mathbf{zj-1}$ sont parallèles ou colinéaires, le choix n'est pas unique : des considérations de symétrie ou de simplicité permettent alors un choix rationnel.

Avec cette méthode nous avons besoin seulement de quatre paramètres géométriques (au lieu de six), pour définir la description d'un référentiel $\mathfrak{R}\mathbf{j}$ par rapport au référentiel $\mathfrak{R}\mathbf{j-1}$. Ces quatre paramètres sont appelés paramètres de Denavit-Hartenberg standards (DH).

1. rotation de θj autour de l'axe zj_{-1} (angle entre x_{j-1} et x_j)
2. translation de d_j le long de l'axe z_j-1.
3. rotation de α_j autour de l'axe x_j (angle entre z_{j-1} et z_j)

4. distance (perpendiculaire commune) entre z_{j-1} et z_j notée par a_j le long de l'axe x_j,

La figure ci-dessous représente la définition des paramètres de Denavit-Hartenberg standards

Il est à noter que les angles sont positifs quand la rotation est dans le sens inverse des aiguilles d'une montre où en utilisant la règle de la main droite.

Les paramètres $\boldsymbol{\alpha_j}$, $\mathbf{a_j}$, $\boldsymbol{\theta_j}$ et $\mathbf{d_j}$, sont les paramètres de Denavit et Hartenberg. On remarque que seul quatre paramètres sont nécessaires pour passer d'un repère $\mathfrak{R}_{j-1}$ au repère $\mathfrak{R}_j$, grâce notamment au choix de l'emplacement de ces derniers.

La variable articulaire $\mathbf{q_j}$ associée à la $\mathbf{j^{éme}}$ articulation se traduit par la relation :

$$\mathbf{q_j = \sigma_j \theta_j + \bar{\sigma}_j d_j}$$

Telle que : $\sigma_j = 0$ si l'articulation j est rotoïde. $\quad \bar{\sigma}_j = 1 - \sigma_j$.

$\sigma_j = 1$ si l'articulation j est prismatique.

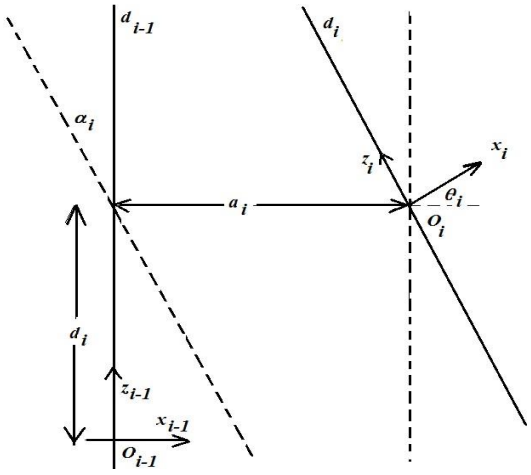


Figure 2:3 Paramètres de Denavit-Hartenberg standards.

La matrice $^{j-1}\mathbf{T_j}$ qui représente la description du référentiel $\mathfrak{R}_j$ par rapport au référentiel $\mathfrak{R}_{j-1}$ peut être obtenue en fonction des paramètres de Denavit-Hartenberg (DH) par :

$$T_j^{j-1} = \mathbf{Rot(Z,\theta i)Trans(Z, di)Rot(X, \alpha i)Trans(X, ai)}$$

$$^{j-1}T_j = \begin{bmatrix} C\theta_j & -S\theta_j C\alpha_j & S\theta_j S\alpha_j & a_j C\theta_j \\ S\theta_j & C\theta_j C\alpha_j & -C\theta_j S\alpha_j & a_j S\theta_j \\ 0 & S\alpha_j & C\alpha_j & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.4.3 Méthode de Denavit-Hartenberg modifiée :

La méthode de Denavit-Hartenberg est bien adaptée pour des structures ouvertes simples, mais présente des ambiguïtés lorsqu'elle est appliquée sur des robots à structures fermées ou arborescentes. Wisama Khalil propose une modification de cette méthode : méthode de Denavit-Hartenberg modifiée (méthode de Khalil). Cette méthode permet la description homogène en un nombre minimum de paramètres pour la représentation des différentes structures de robots généralement rencontrés.

Un repère de référence $\mathfrak{R}_j$ est assigné pour chaque corps C_j du robot à l'articulation **j** où elle rencontre le corps précédent C_{j-1}, ce repère est défini comme suit:

- L'axe z_j se dirige le long de l'axe de l'articulation **j**.
- L'axe **xj** est aligné suivant la direction de la perpendiculaire commune aux axes **zj** et **zj+1** .
- l'axe y_i , non représenté sur la figure, est choisi de manière à former un trièdre orthonormé direct avec x_j et z_j .

Les transformations élémentaires qui permettent d'exprimer le passage du repère $\mathfrak{R}j-1$ au repère $\mathfrak{R}j$ (Figure II.3) sont :

- une rotation d'angle α_j autour de l'axe x_{j-1} , α_j est l'angle entre z_{j-1} et z_j.
- une translation a_j suivant x_{j-1} égale à la perpendiculaire commune aux axes z_{j-1} et z_j.
- une rotation d'angle θj autour de l'axe **zj** , θj est l'angle entre l'axe **j−1** et **xj** .
- une translation suivant l'axe z_j , L'amplitude de cette translation, notée d_j. est donnée par la distance (signée) entre l'axe x_{j-1} avec l'axe x_j.

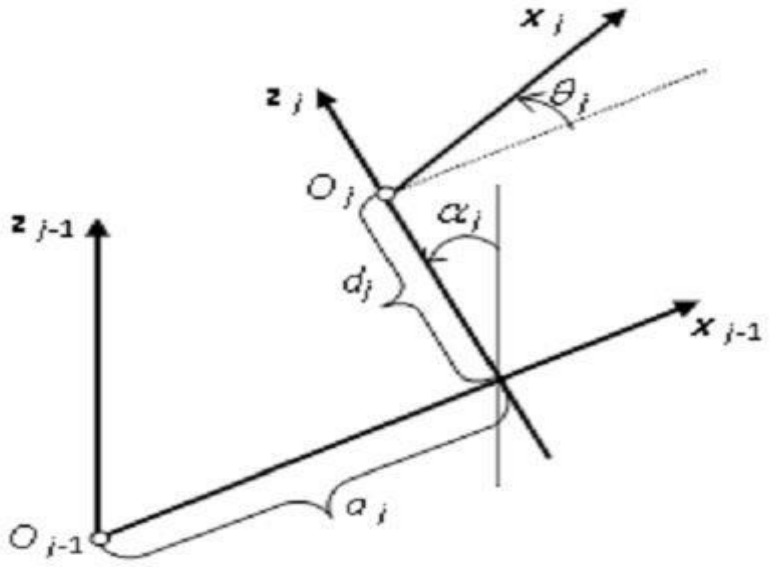


Figure 2:4 : Paramètres de Denavit et Hartenberg Modifiés

En termes de matrice de transformation homogène, les quatre transformations élémentaires donnent la matrice suivante :

$$T_j^{j-1} = \text{Rot(X, } \alpha_j\text{)Trans(X,}a_j\text{)Rot(Z, } \theta_j\text{)Trans(Z, } d_j\text{)}$$

Après son développement, on obtient :

$$T_j^{j-1} = \begin{bmatrix} C\theta_j & -S\theta_j & 0 & a_j \\ C\alpha_j S\theta_j & C\alpha_j C\theta_j & -S\alpha_j & -d_j S\alpha_j \\ S\alpha_j S\theta_j & S\alpha_j C\theta_j & C\alpha_j & d_j C\alpha_j \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Avec les notations : $C\theta_i = \cos(\theta_i)$ et $S\theta_i = \sin(\theta_i)$

La matrice de transformation homogène $T^{i}{}_{i}{}^{-1}$ est souvent notée sous la forme :

$$T_i^{i-1} = \begin{bmatrix} R_i^{i-1} & P_i^{i-1} \\ 0\ 0\ 0 & 1 \end{bmatrix} = \begin{bmatrix} n_i^{i-1} & o_i^{i-1} & a_i^{i-1} & P_i^{i-1} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Tel que :

R_j^{i-1} : Est la matrice de rotation (3x3), appelé aussi matrice d'orientation ou matrice des cosinus directeurs, elle représente la rotation entre les deux repères $\Re_{j-1}$ et $\Re_j$.

Les colonnes de la matrice $\mathbf{R_j^{j-1}}$ représentent les composantes des vecteurs unitaires du repère $\mathfrak{R}\mathbf{j}-\mathbf{1}$ dans le repère .

$\mathbf{P_j^{j-1}}$: Est la matrice de position (3×1) qui définit l'origine du repère $\mathfrak{R}$ j dans le repère $\mathfrak{R}_{\mathbf{j-1}}$.

2.5 Modèle géométrique direct (MGD)

Le modèle géométrique d'un mécanisme regroupe les contraintes géométriques qui doivent être respectées par les variables articulaires θ_i (ou bien q_i) afin d'établir la relation entre la configuration du mécanisme définie dans l'espace des coordonnées généralisées et la configuration du mécanisme décrite dans le repère cartésien [Cyril, 2009]. Le modèle géométrique direct est unique et est donne sous forme d'équations explicites.

$$X_i = f(\theta_i)$$

Ou :

$\theta = [\theta_1, \theta_2, ...\theta_n]^T$ □□Rⁿ : Vecteur des variables articulaires.

$X = [x_1, x_2, ...x_n]^T$ □□Rⁿ : Vecteur des variables opérationnelles.

Afin de calculer l'équation géométrique directe d'un bras manipulateur a` chaıne ouverte simple, une méthode systématique et générale doit être appliquée pour définir la position et l'orientation relative de deux corps consécutifs, Les méthodes de *Denavit-Hartenberg* et de *Khalil et Kleinfinger* sont les plus répandus.

Le problème est comment déterminer deux repères attaches aux deux corps?, et comment calculer les transformations entre eux ? Cependant, il est commode de placer quelques hypothèses pour la définition des repères du corps.

Se reportant a` la figure (Fig II.1), la structure est composée de *n* corps $C_i$, i=1 a` *n*, plus la base (C_0), relies entre eux par des liaisons rotoıde ou prismatique, l'articulation *i* relie le corps C_i au corps C_{i-1}. Afin de définir la relation entre les positions des corps on associe a` chaque corps C_i le repère R_i (O_i, x_i, y_i, z_i) tels que :

- L'axe z_i est porté par l'axe de l'articulation *i*.
- L'axe xi est porté par la perpendiculaire commune aux axes zi et zi+1.
- L'axe yi est formé par la règle droite pour compléter les coordonnées du système (x_i, y_i, z_i).

La matrice de transformation du repère R_{i-1} au repère R_i s'exprime en fonction des paramètres (*de Denavit-Hartenberg*) suivants (Fig II.4) :

α_i : L'angle de rotation entre l'axe z_{i-1} et l'axe z_i autour de x_{i-1}.

d_i : La distance entre l'axe z_{i-1} et l'axe z_i le long de l'axe x_{i-1}.

θ_i : L'angle de rotation entre les axes x_{i-1} et x_i autour de z_i.

r_i : la distance entre les axes x_{i-1} et x_i le long de l'axe z_i.

La variable q_i de l'articulation i, définissant l'orientation ou la position relative entre les articulations i−1 et i, est soit θ_i, soit r_i, selon le type d'articulation est rotoïde ou prismatique respectivement. Ceci est défini par la relation :

$$q_i = \overline{\sigma}_i \theta_i + \sigma_i r_i$$

avec :

- $\sigma_i = 0$ si l'articulation i est rotoïde.
- $\sigma_i = 1$ si l'articulation i est prismatique.
- $\overline{\sigma}_i = 1 - \sigma_i$.

De la même façon on définit la variable q_i :

$$\overline{q_i} = \sigma_i \theta_i + \overline{\sigma}_i r_i$$

La matrice de transformation définissant R_i dans R_{i-1} est donnée par :

$$^{i-1}T_i = Rot(x, \alpha_i)\ Trans(x, d_i)\ Rot(z, \theta_i)\ Trans(z, r_i)$$

$$^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & d_i \\ \cos\alpha_i \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i & -r_i \sin\alpha_i \\ \sin\alpha_i \sin\theta_i & \sin\alpha_i \cos\theta_i & \cos\alpha_i & r_i \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

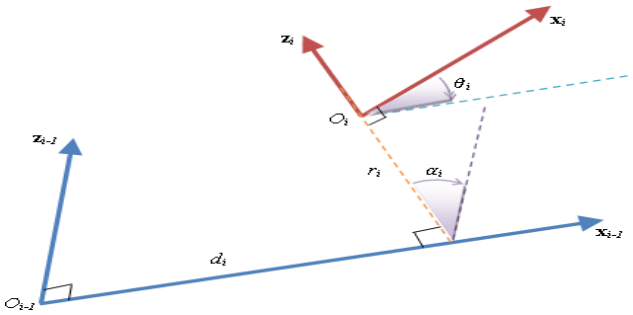


Figure 2:5 Parametres de Denavit-Hartenberg

$$ {}^{i-1}T_i = \begin{bmatrix} & & & d_i \\ & {}^{i-1}A_i & & -r_i \sin\alpha_i \\ & & & r_i \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Ou` la matrice de rotation ${}^{i-1}A_i = Rot(x, \alpha_i)\ Rot\ (z, \theta_i)$.

La matrice de transformation définissant R_{i-1} dans R_i est donnée par :

$$ {}^{i}T_{i-1} = \begin{bmatrix} & & & -d_i \cos\theta_i \\ & {}^{i-1}A_i^{\,T} & & d_i \sin\theta_i \\ & & & -r_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Le modelé géométrique *direct* (MGD) est l'ensemble des relatons permettent de définir la position de l'organe terminal du robot en fonction de ses coordonnées articulaires. Pour un robot `a chaıne ouverte simple le MGD est défini par la matrice de transformation ${}^0T_n$:

$$ {}^0T_n = {}^0T_1(q_1)\,{}^1T_2(q_2)\,{}^2T_3(q_3)\ldots{}^{n-1}T_n(q_n) \qquad (2.1) $$

Le MGD peut aussi être représente par la relation :

$$ X = f(q) $$

Ou q est le vecteur des variables articulaires tels que :

$$ q = \begin{bmatrix} q_1 & q_2 & q_3 & \ldots. & q_n \end{bmatrix}^T $$

2.6 Modèle géométrique inverse (MGI)

Le modelé géométrique direct fournit l'emplacement de l'organe terminal en fonction des coordonnées articulaires. Le modelé géométrique *inverse* (MGI) consiste à déterminer les variables articulaires correspondant à une situation spécifique de l'organe terminal. Lorsqu'elles existent.

Trois méthodes de calcul de MGI sont répandues [Kha04] :

- La méthode de Paul, qui convient pour la plupart des robots industriels.
- La méthode de Pieper, qui permet de résoudre le problème pour les robots `a six degrés de liberté avec trois articulations rotoıde ou trois articulation prismatiques.
- La méthode de raghavan et Roth, donnant la solution générale des robots à six articulations à partir d'un polynôme de degré au plus égale 16.

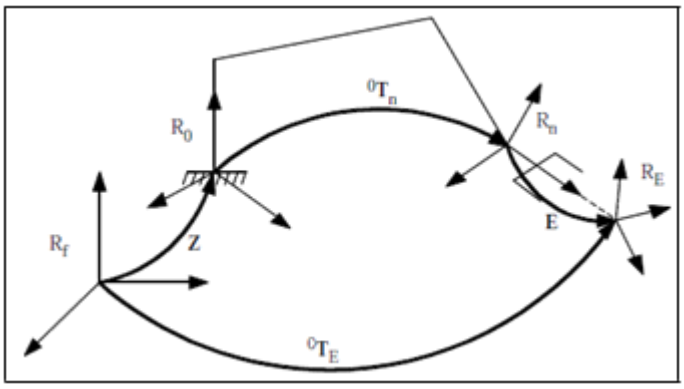


Figure 2:6 Transformations entre l'organe terminal et le repère atelier

Quand le modèle géométrique inverse ne peut pas être obtenu, des techniques numériques peuvent être employées. Ces techniques emploient la méthode de *Newton-Raphson* ou des méthodes fondes sur la transpose de la *matrice Jacobienne*.

Soit ${}^{i-1}T_E{}^d$ la matrice de transformation homogène représente la position désirée du repère *outil* R_E par rapport au repère *atelier* R_f. En général on peut exprimer ${}^{i-1}T_E{}^d$ sous la forme :

$$ {}^f T_E{}^d = Z\,{}^0T_n(q)E \qquad (2.2)$$

tels que :

- Z est la matrice de transformation définissant le repère de base du robot dans le repère atelier R_f.
- ${}^0T_n$ est la matrice de transformation du repère terminal dans le repère R_0.
- E est la matrice de transformation du repère outil R_E dans le repère terminal R_n.

Mettant tous les termes connus dans le membre gauche, on obtient :

$$U_0 = {}^0T_n(q)$$

Avec, $U_0 = Z^{-1}\,{}^f T_E{}^d E^{-1}$

Le MGD donnant $X = f(q),\ q = [q_1 q_2 \ldots . q_n]^T, X = [x_1 x_2 \ldots . x_m]^T$ ou *n* est le nombre de coordonnées opérationnelles et *m* le nombre de coordonnées articulaires, le problème au-dessus (2.2) s'agit de résoudre un système de *m* équations à *n* inconnues, ce système étant non linéaire. Le nombre de solutions dépend de l'architecture du robot manipulateur et de l'amplitude des articulations.

Trois cas se présentent pour calculer le MGI :

1. Solutions en nombre fini.
2. Absence de solution, lorsque la position de l'organe terminal désirée est en dehors de la zone accessible du robot.
3. Infinité de solutions lorsque :

- le robot est redondant vis-à-vis la tâche.
- le robot se trouve dans certaines configurations singulières.

Lorsqu'il est possible de calculer toutes les configurations permettant d'atteindre une situation données d'un robot manipulateur ce dernier est dit résoluble.

- **La méthode de Paul :**

Soit la matrice de transformation homogène d'un robot manipulateur (2.1) :

$$^{0}T_{n} = {}^{0}T_{1}(q_1)\,{}^{1}T_{2}(q_2)\,{}^{2}T_{3}(q_3)\ldots{}^{n-1}T_{n}(q_n)$$

Soit U_0 la situation désirée telle que :

$$U_0 = \begin{bmatrix} s_x & n_x & a_x & P_x \\ s_y & n_y & a_y & P_y \\ s_z & n_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

On cherche à résoudre le système d'équations suivant :

$$U_0 = {}^{0}T_{1}(q_1)\,{}^{1}T_{2}(q_2)\,{}^{2}T_{3}(q_3)\ldots{}^{n-1}T_{n}(q_n) \qquad (2.3)$$

Pour résoudre le système (2.3), Paul a proposé une méthode (*Méthode de Paul*) qui consiste à pré-multiplier successivement les deux membres de l'équation par les matrices $^{i}T_{i-1}$ pour i de 1 à $n-1$. Ces opérations permettent d'isoler les variables d'articulations l'une après l'autres.

$$U_0 = {}^{0}T_{1}\,{}^{1}T_{2}\,{}^{2}T_{3}\ldots{}^{n-1}T_{n}$$

$${}^{1}T_{0}U_0 = {}^{1}T_{2}\,{}^{2}T_{3}\,{}^{3}T_{4}\,{}^{4}T_{5}\ldots{}^{n-1}T_{n}$$

$${}^{2}T_{1}U_1 = {}^{2}T_{3}\,{}^{3}T_{4}\,{}^{4}T_{5}\ldots{}^{n-1}T_{n}$$

$${}^{3}T_{2}U_2 = {}^{3}T_{4}\,{}^{4}T_{5}\ldots{}^{n-1}T_{n}$$

$${}^{n-1}T_{n-2}U_{n-2} = {}^{n-1}T_{n}$$

Avec, $U_{j+1} = {}^{j+1}T_{n} = {}^{j+1}T_{j}U_{j}$, pour j = 0,…, n.

2.7 Modélisation cinématique

2.7.1 Modèle cinématique direct

Le modèle cinématique *direct* (MCD) d'un robot manipulateur décrit les vitesses des coordonnées opérationnelles en fonction des vitesses articulaires [Eti01]. Il s'écrit :

$$\dot{X} = J(q)\dot{q}$$

Ou, J(q) désigne la matrice *Jacobienne* de dimension ($m \times n$) du mécanisme, égale a $\frac{\partial X}{\partial q}$.

Le calcul de la matrice Jacobienne peut se faire en dérivant le MGD, $X = f(q)$, à partir de la relation suivante :

$$J_{ij} = \frac{\partial f_i(q)}{\partial q_i} i = 1,..., m; j = 1,..., n$$

Où, J_{ij} est l'élément (i, j) de la matrice Jacobienne J.

Cette méthode est pratique pour des robots simples avec un nombre réduit de degrés de liberté. Le calcul de la *matrice Jacobienne de base*, connu sous le nom de *matrice Jacobienne* cinématique, est plus pratique pour un robot général de degré de liberté égale n. La matrice Jacobienne obtenue (*sans calcul de la* dérivée *du MGD*) relie les vecteurs des vitesses de translation et de rotation V_n et ω_n et les vitesses articulaires [Kha04] :

$$\begin{bmatrix} V_n \\ \omega_n \end{bmatrix} = J_n \dot{q}$$

Ou, V_n et ω_n sont les vitesses linéaire et angulaire du repère R_n respectivement.

La vitesse $\dot{q}_k$ de l'articulation k produit une vitesse linéaire ($V_{k,n}$) et une vitesse angulaire (ω_{kn}) sur le repère terminal R_n. Deux cas se présentent :

1. Si l'articulation k est prismatique ($\sigma_k = 1$, Fig II.6) :

$$\begin{cases} V_{k,n} = a_k \dot{q}_k \\ \omega_{k,n} = 0 \end{cases}$$

Où, a_k est le vecteur unitaire porte par l'axe z_k de l'articulation k.

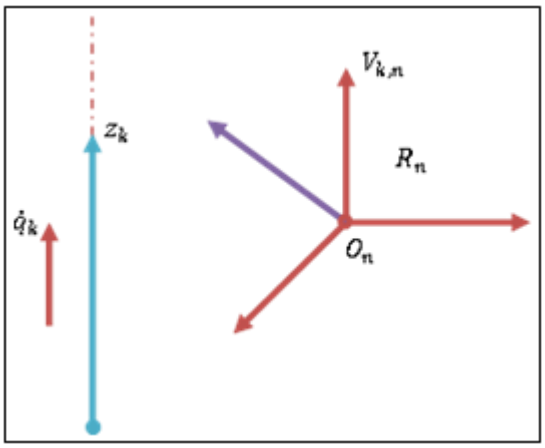


Figure 2:7 Cas d'articulation prismatique

2. Si l'articulation k est rotoıde (σ_k= 0, Fig II.7) :

$$\begin{cases} V_{k,n} = a_k \dot{q} \times L_{k,n} = (a_k \times L_{k,n})\dot{q}_k \\ \omega_{k,n} = a_k \dot{q} \end{cases} \quad (2.4)$$

Le terme $L_{k,n}$ désigne le vecteur $\overrightarrow{O_k O_n}$.

De façon générale, les vecteurs $V_{k,n}$ et $\omega_{k,n}$ s'écrivent sous la forme :

$$\begin{cases} V_{k,n} = \left[\sigma_k a_k + \overline{\sigma_k}(a_k \times L_{k,n})\right]\dot{q}_k \\ \omega_{k,n} = \overline{\sigma_k} a_k \dot{q}_k \end{cases}$$

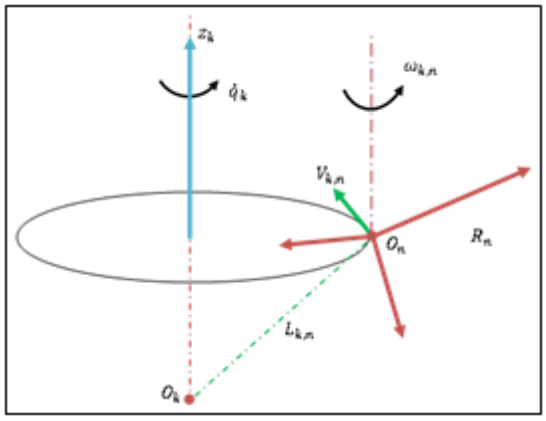


Figure 2:8 Cas d'articulation rotoıde

Les vitesses lineaires et angulaires de lorgane terminal peuvent etre ecrites comme :

$$\begin{cases} V_n = \sum_{k=1}^{n} V_{k,n} = \sum_{k=1}^{n} \left[\sigma_k a_k + \overline{\sigma_k} \left(a_k \times L_{k,n} \right) \right] \dot{q_k} \\ \omega_n = \sum_{k=1}^{n} \omega_{k,n} = \sum_{k=1}^{n} \overline{\sigma_k} a_k \dot{q_k} \end{cases}$$

L'ecriture de l'equation (2.5) sous forme d'une matrice en utilisant l'equation (2.4), donne :

$$J_n = \begin{bmatrix} \sigma_1 a_1 + \overline{\sigma_1} \left(a_1 \times L_{1,n} \right) & \cdots & \sigma_n a_n + \overline{\sigma_n} \left(a_n \times L_{n,n} \right) \\ \overline{\sigma_1} a_1 & \cdots & \overline{\sigma_n} a_n \end{bmatrix}$$

Se referant les vecteurs de J_n par rapport au repere R_i, on obtient la matrice jacobienne ${}^{i}J_n$, telle que :

$$ {}^{i} \begin{bmatrix} V_n \\ \omega_n \end{bmatrix}_n = {}^{i}J_n \, \dot{q} \qquad (2.5)$$

En general, on calcul V_n et ω_n dans R_n et R_0, la matrice jacobienne correspondante est ${}^{n}J_n$ ou ${}^{0}J_n$ respectivement. ces matrices peuvent etre aussi calculees en utilisant une matrice ${}^{i}J_n$, j=0,...,n, grace a` l'expression suivante :

$$ {}^{s}J_n = \begin{bmatrix} {}^{s}A_i & 0_3 \\ 0_3 & {}^{s}A_i \end{bmatrix} {}^{i}J_n $$

ou ${}^{s}A_i$ est la matrice d'orientation du rep`ere R_i dans R_s. En general on obtient la matrice simple ${}^{i}J_n$ lorsqu'on prend i=entier(n/2).On note que les matrices ${}^{i}J_n$ ayant les memes positions singulieres.

- **Calcul de la matrice ${}^{i}J_n$**

Le produit $a_k \times K_{k,n}$ peut etre calcule par $\hat{a}_k L_{k,n}$, la k^{ime} colonne de la matrice ${}^{i}J_n$ notee par ${}^{i}J_{n:k}$ devienne :

$$ {}^{i}J_{n:k} = \begin{bmatrix} \sigma_k \, {}^{i}a_k + \bar{\sigma}_k \, {}^{i}A_k \, {}^{k}\hat{a}_k \, {}^{k}L_{k,n} \\ \bar{\sigma}_k \, {}^{i}a_k \end{bmatrix} $$

On pose ${}^{k}a_k = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ et ${}^{k}L_{k,n} = {}^{k}P_n$, donc on obient :

$$ {}^{i}J_{n:k} = \begin{bmatrix} \sigma_k \, {}^{i}a_k + \overline{\sigma_k}(-{}^{k}P_{n_y} \, {}^{i}s_k + {}^{k}P_{n_x} \, {}^{i}n_k) \\ \overline{\sigma_k} \, {}^{i}a_k \end{bmatrix} $$

ou ${}^{k}P_{ny}$ et ${}^{k}P_{nx}$ sont les composants du vecteur ${}^{k}P_n$. a partir de cette expression, la k^{ime} colonne de la matrice ${}^{i}J_n$ est :

$$ {}^{i}J_{n:k} = \begin{bmatrix} \sigma_k \, {}^{i}a_k + \bar{\sigma}_k \, {}^{i}\hat{a}_k({}^{i}P_n - {}^{i}P_k) \\ \bar{\sigma}_k \, {}^{i}a_k \end{bmatrix} $$

qui donne pour $i = 0$:

$$ {}^{0}J_{n:k} = \begin{bmatrix} \sigma_k \, {}^{0}a_k + \bar{\sigma}_k \, {}^{0}\hat{a}_k({}^{0}P_n - {}^{0}P_k) \\ \bar{\sigma}_k \, {}^{0}a_k \end{bmatrix} $$

dans ce cas on est besoin de calculer les matrices ${}^{0}T_k$ pour k=1,...,n.

2.7.2 Modele cinematique inverse

Le modèle cinématique *inverse* (MCI) donne les vitesses articulaires $\dot{q}$ correspondants à une vitesse désire $\dot{X}$ de l'organe terminal. Le modèle cinématique inverse s'obtient par la solution d'un système d'équations linéaires soit analytiquement, soit numériquement. Les *solutions analytiques* diminuent le nombre d'opérations de façon remarquable par rapport aux solutions numériques, mais il faut traiter les cas singuliers séparément. Les *solutions* numériques sont plus générales et traitent tous les cas de la même façon [Eti01].

Soit $X = \begin{bmatrix} X_p^T & X_r^T \end{bmatrix}^T$ une représentation de la situation du repère R_n dans le repère R_0. Ou X_p^T et X_r^T désignent respectivement la position et l'orientation opérationnelles.

Les relations entre les vitesses $\dot{X}_P$ et $\dot{X}_r$ et les vecteurs ${}^{0}V_n$ et ${}^{0}\omega_n$ sont :

$$ \begin{bmatrix} \dot{X}_p \\ \dot{X}_r \end{bmatrix} = \begin{bmatrix} \Omega_p & O_3 \\ O_3 & \Omega_r \end{bmatrix} \begin{bmatrix} {}^{0}V_n \\ {}^{0}\omega_n \end{bmatrix} = \Omega \begin{bmatrix} {}^{0}V_n \\ {}^{0}\omega_n \end{bmatrix} $$

A partir de l'équation (2.4) le MCD s'écrit sous la forme : $\dot{X} = J\dot{q}$.

La méthode la plus générale consiste à calculer J^{-1} la matrice inverse de *J*, qui permet de déterminer les vitesses articulaires $\dot{q}$ grâce a` la relation :

$$\dot{q} = J^{-1}\dot{X}$$

Lorsque la matrice *J* a la forme : $J = \begin{bmatrix} A & 0 \\ B & C \end{bmatrix}$ les matrices A et C étant carrées inversibles, il est facile de montrer que l'inverse de cette matrice s'écrit :

$$J^{-1} = \begin{bmatrix} A^{-1} & 0 \\ -C^{-1}BA^{-1} & C^{-1} \end{bmatrix}$$

La résolution *du* problème *se* ramené *donc à* l'inversion *de la matrice J.*

2.8 Modélisation dynamique

Le modèle *dynamique* est la relation entre les couples (et/ou forces) appliques aux actionneurs et les positions, vitesses et accélérations articulaires [Abd04]. Il est représenté par la relation :

$$\Gamma = \left(q, \quad \dot{q}, \quad \ddot{q}, \quad f_e \right) \qquad (2.6)$$

Avec :

Γ : vecteur des couples des actionneurs, selon que l'articulation est rotoıde ou prismatique.

q : vecteur des positions articulaires.

$\dot{q}$: vecteurs des vitesses articulaires.

$\ddot{q}$: vecteurs des accélérations articulaires.

f_e : vecteurs représentant l'effort extérieur qu'exerce le robot sur l'environnement.

La relation (2.6) est appelée modèle dynamique inverse(ou modèle dynamique) parce qu'elle définit le système d'entre en fonction des variables sorties.

Le modèle dynamique direct décrit les accélérations articulaires en fonction des positions, vitesses et couples. Il est représenté par la relation suivante :

$$\ddot{q} = g\left(q, \quad \dot{q}, \quad \Gamma, \quad f_e \right)$$

Le modèle dynamique joue un rôle important dans la conception et le fonctionnement des robots. Pour la conception, le modèle dynamique inverse peut être utilise pour choisir les actionneurs, alors que le modèle dynamique direct est utilisé pour effectuer des simulations,

afin de tester les performances du robot. En ce qui concerne les fonctionnalités du robot, le modèle dynamique inverse est utilisée pour calculer les couples actionneurs, qui sont nécessaires pour réaliser un mouvement souhaite. Il est également utilise pour identifier les paramètres dynamiques qui sont nécessaires à la fois pour le contrôle et la simulation [Kha04].

Plusieurs approches sont proposées pour obtenir le modèle dynamique des robots. Les plus souvent utilise dans la robotique sont le formalisme de *Lagrange*, et le formalisme de *Newton- Euler*.

L'approche de Newton-Euler est basée sur les forces et les moments agissent entre les liens. La formulation de Newton-Euler peut être considère comme une approche basée sur l'équilibre *des forces*, la formulation de Lagrange est une approche basée sur l'énergie. Bien sûr, pour le même manipulateur, les deux donnent les mêmes équations du mouvement [Joh05].

Dans cette étude on présente le formalisme de Lagrange et on considère que les robots à chaıne ouverte simple.

➢ Formalisme de Lagrange :

On considère un système idéal sans frottement ou élasticité, exerçant ni des forces ni des moments sur l'environnement.

La formulation de Lagrange décrit le comportement d'un système dynamique en terme d'énergie [LB99]. Les équations de Lagrange sont généralement écrites sous la forme :

$$\Gamma = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q}$$

ou L est le *Lagrangien* du système, se définit comme la différence entre l'énergie cinétique K et l'énergie potentiel P :

$$L = K - P$$

Pour obtenir l'équation dynamique générale du robot, on détermine les énergies cinétiques et potentielles, et le lagrangien, puis les remplaces dans l'équation de Lagrange.

Dans notre usage, q sera le vecteur des variables articulaires, se composant de l'angle d'articulation θ_i et le déplacement de l'articulation d_i. Alors que Γ est un vecteur qui a comme composants les couples n_i correspondants aux angles θ_i, et les forces f_i correspondantes aux déplacements d_i.

Energie cinétique :

L'énergie cinétique k_i du i^{ieme} articulation est donnée par l'expression :

$$k_i = \frac{1}{2}\, m_i v_{C_i}^T\, v_{C_i} + \frac{1}{2}\, {}^i\omega_i^T\, {}^{C_i}I_i\, {}^i\omega_i$$

ou le premier terme est l'énergie cinétique due à la vitesse linéaire et le deuxième terme est l'énergie cinétique due à la vitesse angulaire. L'énergie cinétique totale du manipulateur est la somme de l'énergie cinétique des différentes articulations c'est-à-dire :

$$K = \sum_{i=1}^{n} k_i$$

Les vitesses $v_{C_i}^T$ et ${}^i\omega_i^T$ sont des fonctions de q et $\dot{q}$ respectivement, donc on voit que l'énergie cinétique d'un manipulateur peut être décrite par une formule en fonction de la position et de la vitesse, $k(q, \dot{q})$. En fait, l'énergie cinétique d'un manipulateur est donnée par :

$$K(q, \dot{q}) = \frac{1}{2}\, \dot{q}^T\, M(q)\, \dot{q}$$

où $M(q)$ est une matrice $n \times n$ de l'énergie cinétique appelée *matrice d'inertie*. Une expression de la forme ci-dessus est appelé la *forme quadratique*. L'énergie cinétique doit toujours être positive, pour cela la matrice d'inertie du manipulateur doit être une matrice définie positive.

Energie potentiel :

L'énergie potentiel du i^{ime} articulation, a l'expression suivantes :

$$p_i = m_i\, {}^0g^T\, {}^0Tc_i + p_{ref_i}$$

ou ${}^0g$ est le vecteur de gravite (3×1) , ${}^0T_{Ci}$ une transformation homogène localisant le centre de la $i^{ieme`}$ articulation, et p_{refi} est une constante choisie de sorte que la valeur minimum de p_i soit 0. L'énergie potentielle totale est la somme de l'énergie potentielle dans les différentes articulations, c'est-à-dire :

$$P = \sum_{i=1}^{n} p_i$$

Puisque ${}^0T_{Ci}$ sont en fonction de $q_1, q_2, \ldots, q_n$, l'énergie potentielle est en fonction de la position de l'articulation, $P(q)$. donc le Lagrangien sera :

$$L\left(q, \dot{q}\right) = K\left(q, \dot{q}\right) - p(q) \qquad (2.7)$$

2.9 Equation du mouvement d'un robot manipulateur

Se reportant à l'équation (2.7) le Lagrangien a la forme :

$$L(q,\dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} - P(q)$$

ou $M(q)$ est la matrice d'inertie du manipulateur et $P(q)$ est l'énergie potentielle due au gravite. Il est convenable décrire l'énergie cinétique sous forme d'une somme :

$$L(q,\dot{q}) = \frac{1}{2}\sum_{i,j=1}^{n} M_{ij}(q)\dot{q}_i\dot{q}_j - P(q) \quad (2.8)$$

en substituant dans l'équation de Lagrange, les équations de mouvements sont données par :

$$\Gamma_i = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i}$$

ou Γ_i sont les forces (couples) agissant sur l'articulation i, en utilisant l'équation (2.8) on a :

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} = \frac{d}{dt}(\sum_{j=1}^{n} M_{ij}(q)\dot{q}_j) = \sum_{j=1}^{n}(M_{ij}(q)\ddot{q}_j + \dot{M}_{ij}(q)\dot{q}_j$$

$$\frac{\partial L}{\partial q_i} = \frac{1}{2}\sum_{j,k=1}^{n}\frac{\partial M_{kj}(q)}{\partial q_j}\dot{q}_k\dot{q}_j - \frac{\partial P(q)}{\partial q_i}$$

Le terme $\dot{M}(q)$ peut être augmente en termes de dérivés partiels.

$$\Gamma_i = \sum_{j=1}^{n} M_{ij}(q)\ddot{q}_j + \sum_{j,k=1}^{n}(\frac{\partial M_{ij}(q)}{\partial q_k}\dot{q}_j\dot{q}_k - \frac{1}{2}\frac{\partial M_{kj}(q)}{\partial q_i}\dot{q}_k\dot{q}_j) + \frac{\partial P(q)}{\partial q_i}$$

Afin de mettre les équations du mouvement sous forme d'un vecteur, on définit la matrice $C\left(q,\dot{q}\right)$, tell que :

$$C_{ij}(q,\dot{q}) = \frac{1}{2}\sum_{k=1}^{n}(\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{kj}}{\partial q_i})\dot{q}_k$$

maintenant l'équation (2.9) peut être écrite comme :

$$\Gamma = M(q)\ddot{q} + C\left(q,\dot{q}\right)\dot{q} + G(q) \quad (2.9)$$

ou :

Γ : vecteur des couples appliquées aux articulations.

M(q) : matrice d'inertie du robot.

$C\left(q,\dot{q}\right)\dot{q}$: vecteur des forces de Coriolis et centrifuges.

G(q) : vecteur des forces de gravite.

Pour un système avec frottement ou élasticité, le modèle dynamique est :

$$\Gamma = M(q)\ddot{q} + C\left(q,\dot{q}\right)\dot{q} + G(q) + F_e$$

Il s'agit d'une équation différentielle du second ordre pour le mouvement du manipulateur en fonction des couples articulaires appliques [RLSS94].

Les matrices *M* et *C*, qui résument les propriétés d'inertie du manipulateur, ont les propriétés :

- La matrice d'inertie M(q) est symétrique et définie positive.
- La matrice *M*(*q*) est donnée comme suit :

$$M_m I \le M(q) \le M_M I$$

avec M_m et M_M des scalaires positives, et *I* la matrice d'inertie.

L'inverse de la matrice M(q) est borne :

$$\frac{1}{M_M} \le M^{-1}(q) \le \frac{1}{M_m}$$

La matrice N $(q,\dot{q}) = \dot{M}(q) - 2C(q,\dot{q})$ est antisymétrique.

La matrice $C(q,\dot{q})$ vérifier la relation suivante :

$$C(q, x)y = C(q, y)x$$

avec *x, y* les vecteurs de vitesse de dimension ($n \times 1$).

la matrice $C(q,\dot{q})\dot{q}$ peut être écrite sous la forme :

$$C(q,\dot{q})\dot{q} = B(q)\left[q\dot{q}\right] + C(q)\left[\dot{q}^2\right]$$

ou :

B(*q*) : une matrice de dimension $n \times n\,(n-1)$ des coefficients de Coriolis ses éléments sont égaux a :

$$b_{ijk} = \frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_i} - \frac{\partial m_{ij}}{\partial q_k}$$

et $C(q)$: est une matrice de dimension ($n \times n$) des coefficients centrifuges et [$\dot{q}^2$] est un vecteur de dimension ($n \times n$), les éléments de $C(q)$ sont données par :

$$C(q) = \frac{1}{2}\left(2\frac{mkj}{\partial q_k} - \frac{\partial m_{jj}}{\partial q_k}\right)$$

La norme de la matrice C ($q, \dot{q}$) vérifiée la relation suivante :

$$\left\| C(q,\dot{q}) \right\| \leq v_0 \left\| \dot{q} \right\|$$

ou, $v_0 > 0$ est indépendant de q.

La norme du vecteur de gravite est bornée supérieurement ce que veut dire :

$$\|G(q)\| \leq g_b(q)$$

où, g_b est une fonction scalaire.

2.10 Conclusion

Pour modélisé un système, c'est-à-dire gouverner ses sorties, il faut prévoir le comportement du système, en réponse aux différentes excitations d'entrer qui pourront lui être appliquées; la démarche est de représenter le comportement du système sous la forme d'un modèle, une telle démarche s'appelle **la modélisation**; d'une manière générale, on recherche toujours le modèle le plus simple qui permet d'expliquer, de manière satisfaisante, le comportement du processus dans son domaine d'application; les modèles de transformation entre l'espace opérationnel (dans lequel est définie la situation de l'organe terminal) et l'espace articulaire. (Dans lequel est définie la configuration du robot), On distingue :

Les modèles géométriques qui expriment la situation de l'organe terminal en fonction de la configuration du mécanisme.

Les modèles cinématiques permettent de contrôler la vitesse de déplacement du robot afin de connaître la durée d'exécution d'une tâche.

Les modèles dynamiques définissent les " équations du mouvement du robot qui permettent d'établir les relations entre les couples ou forces exercés par les actionneurs et positions, vitesses et accélérations.

Chapitre 3

Généralités et Applications Hard/Soft sur l'Environnement de Programmation 'ARDUINO' de Bras Manipulateur Conçu

3 Chapitre 3 Généralités et Applications Hard/Soft sur l'Environnement de Programmation 'ARDUINO' de Bras Manipulateur Conçu

3.1 Introduction

Depuis que l'électronique existe, sa croissance est fulgurante et continue encore aujourd'hui. Si bien que faire de l'électronique est devenu accessible à toutes personnes en ayant l'envie. Mais, le manque de cours simples sur le net ou en libraire empêche la satisfaction des futurs électroniciens amateurs ou professionnels et parfois empêche certains génies à se révéler.

Tout système automatisé est composé d'une installation (machine) et d'une partie commande constituée par l'appareillage d'automatisme. Cette dernière partie synthétise les consignes des asservissements pilotant les actionneurs à partir des ordres de l'utilisateur. Le domaine de la commande des robots est très vaste, il est encore en pleine évolution, différentes techniques ont proposé pour la commande des robots. Celle qu'on a choisie pour notre manipulateur est la commande par la carte Arduino.

3.2 Présentation de l'Arduino

3.2.1 Définition

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 le monde réel et le monde 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.

3.2.2 Applications

Le système Arduino nous permet de réaliser un grand nombre de choses, qui ont une application dans tous les domaines, nous pouvons donner quelques exemples :

- Contrôler les appareils domestiques
- Faire un jeu de lumières
- Communiquer avec l'ordinateur
- Télécommander un appareil mobile (modélisme) etc.

- Fabriquer votre propre robot.

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.

3.2.2.1 Pourquoi 'Arduino' ?

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

Le prix : En vue des performances qu'elles offrent, les cartes Arduino sont relativement peu couteuses, ce qui est un critère majeur pour le débutant.

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

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

Un environnement de programmation clair et simple : l'environnement de programmation Arduino (le logiciel Arduino) est facile à utiliser pour les débutants, tout en étant assez flexible pour que les utilisateurs avancés puissent en tirer profit également.

3.2.2.2 Le principe de fonctionnement de l'Arduino

Les différentes versions des Arduino fonctionnent sous le même principe général :

1. On conçoit ou on ouvre un programme existant avec le logiciel Arduino.
2. On vérifie ce programme avec le logiciel Arduino (compilation).
3. Si des erreurs sont signalées, on modifie le programme.
4. On charge le programme sur la carte.
5. On câble le montage électronique.
6. L'exécution de programme est automatique après quelques secondes.
7. On alimente la carte soit par le port USB, soit par une source d'alimentation autonome (pile 9 volts par exemple).
8. On vérifie que notre montage fonctionne.

3.2.3 Les types de la carte 'Arduino'

Des cartes Arduino il en existe 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).

3.2.4 Différents cartes

- **la carte uno :**

L'UNO est sans doute l'Arduino le plus populaire. Il est alimenté par un processeur Atmega328 fonctionnant à 16 MHz, comprend 32 Ko de mémoire programme, 1 Ko d'EEPROM, 2 Ko de RAM, 14 E / S numériques, 6 entrées analogiques et un rail d'alimentation de 5V et 3,3V.

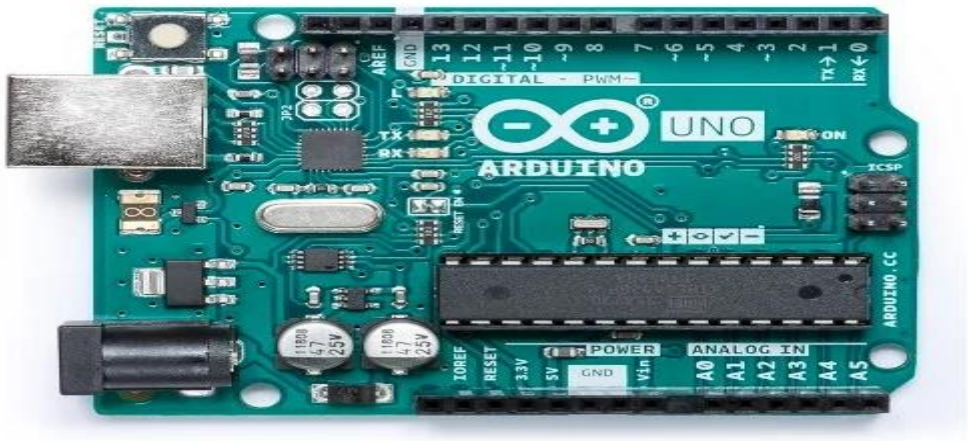


Figure 3:1 carte Arduino « uno »

- **la carte Nano :**

L'Arduino Nano est essentiellement un Arduino UNO réduit, ce qui le rend très pratique pour les espaces restreints et les projets pouvant nécessiter une réduction de poids chaque fois que cela est possible, comme le modélisme ou des projets DIY portable.

Comme l'UNO, le Nano est alimenté par un processeur Atmega328 fonctionnant à 16 MHz, comprend 32 Ko de mémoire programme, 1 Ko d'EEPROM, 2 Ko de RAM, 14

entrées-sorties numériques, 6 entrées analogiques et des rails d'alimentation 5V et 3,3V.

Contrairement au système UNO, le Nano ne peut pas se connecter aux platines de prototypages. Les cartes Arduino Nano sont souvent l'option de carte Arduino la moins chère, ce qui les rend rentables pour les grands projets.

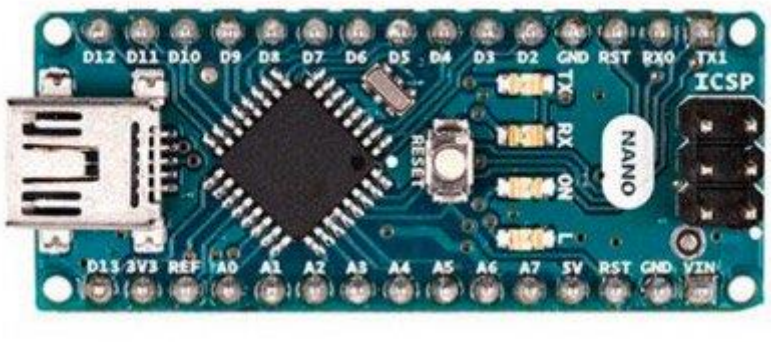


Figure 3:2 carte Arduino « Nano »

- **la carte Due :**

L'Arduino Due est l'une des cartes les plus grandes et la première carte Arduino à être alimentée par un processeur ARM.

Alors que l'UNO et Nano fonctionnent à 5V, la DUO fonctionne en 3,3V – il est important de le noter, car une surtension endommagerait irrémédiablement la carte. Alimenté par un Cortex-M3 ATSAM3X8E cadencé à 84 MHz, le Due dispose de 512 Ko de ROM et de 96 Ko de RAM, de 54 broches d'E / S numériques, de 12 canaux PWM, de 12 entrées analogiques et de 2 sorties analogiques.

La DUE n'a pas de mémoire EEPROM intégrée et est l'une des cartes Arduino les plus chères. Bien que le Due dispose d'un grand nombre d'en-têtes de broches pour la connexion aux nombreuses E / S numériques, il est également compatible avec les broches Arduino standard.

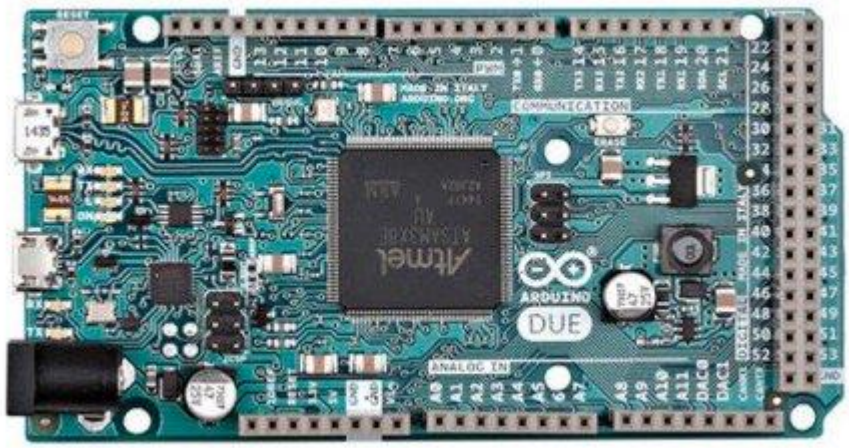


Figure 3:3 carte Arduino « Due »

- **la carte Mega 2560 :**

L'Arduino Mega est un peu similaire au Due en ce sens qu'il dispose également de 54 E / S. Cependant, au lieu d'être alimenté par un cœur ARM, il utilise plutôt un ATmega2560.

Le processeur est cadencé à 16 MHz et comprend 256 Ko de ROM, 8 Ko de RAM, 4 Ko d'EEPROM et fonctionne à 5 V, ce qui facilite son utilisation avec la plupart des appareils électroniques conviviaux.

L'Arduino Mega dispose de 16 entrées analogiques, de 15 canaux PWM, d'un brochage similaire à Due et d'un matériel compatible avec les shields Arduino. Comme pour Due, la compatibilité logicielle avec Mega ne peut pas toujours être garantie.

Figure 3:4 carte Arduino « Mega »

- **la carte Leonardo :**

La carte Arduino LEONARDO est basée sur un ATMega32u4 cadencé à 16 MHz permettant la gestion du port USB par un seul processeur. Des connecteurs situés sur les bords extérieurs du circuit imprimé permettent d'enficher une série de modules complémentaires.

Elle peut se programmer avec le logiciel Arduino. Le contrôleur ATMega32u4 permet la gestion du port, ce qui permet d'augmenter la flexibilité dans la communication avec l'ordinateur.

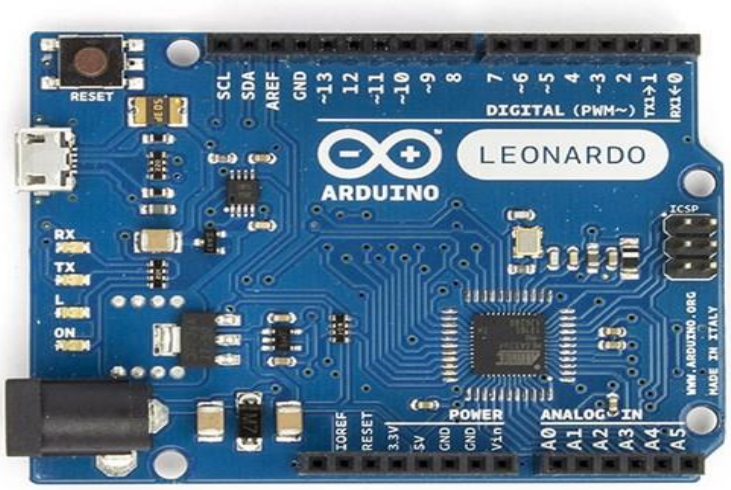


Figure 3:5 carte Arduino « Leonardo »

3.2.4.1 Présentation de l'Arduino UNO

Le système Arduino est une carte électronique basée autour d'un microcontrôleur et de composants minimum pour réaliser des fonctions plus ou moins évoluées à bas coût. Elle possède une interface usb pour la programmer. C'est une plateforme open-source qui est basée sur une simple carte à microcontrôleur (de la famille AVR), et un logiciel, véritable environnement de développement intégré, pour écrire, compiler et transférer le programme vers la carte à microcontrôleur.

Arduino peut être utilisé pour développer des applications matérielles industrielles légères ou des objets interactifs (création artistiques par exemple), et peut recevoir en entrées une très grande variété de capteurs. Arduino peut aussi contrôler une grande variété d'actionneurs (lumières, moteurs ou toutes autres sorties matériels). Les projets Arduino peuvent être autonomes, ou communiquer avec des logiciels sur un ordinateur (Flash, Processing ou MaxMSP). Les cartes électroniques peuvent être fabriquées manuellement ou

bien être achetées préassemblées ; le logiciel de développement open-source est téléchargeable gratuitement.

3.2.4.2 Synthèse des caractéristiques

Table 3-1 Synthèse des caractéristiques

Microcontrôleur	Atmega328
Tension de fonctionnement	5V
Tension d'alimentation (recommandée)	7-12V
Tension d'alimentation (limites)	6-20V
Broches E/S numériques	14 (dont 6 disposent d'une sortie PWM)
Broches d'entrées analogiques	6 (utilisables en broches E/S numériques)
Intensité maxi disponible par broche E/S (5V)	40 mA (ATTENTION : 200mA cumulé pour l'ensemble des broches E/S)
Intensité maxi disponible pour la sortie 3.3V	50 mA
Intensité maxi disponible pour la sortie 5V	Fonction de l'alimentation utilisée – 500 mA max si port USB utilisé seul
Mémoire Programme Flash	32 KB (Atmega328) dont **0.5 KB** sont utilisés par le bootloader
Mémoire SRAM (mémoire volatile)	2 KB (Atmega328)
Mémoire EEPROM (mémoire non volatile)	1 KB (Atmega328)
Vitesse d'horloge	16 MHz

3.2.4.3 Qu'est-ce qu'un microcontrôleur ?

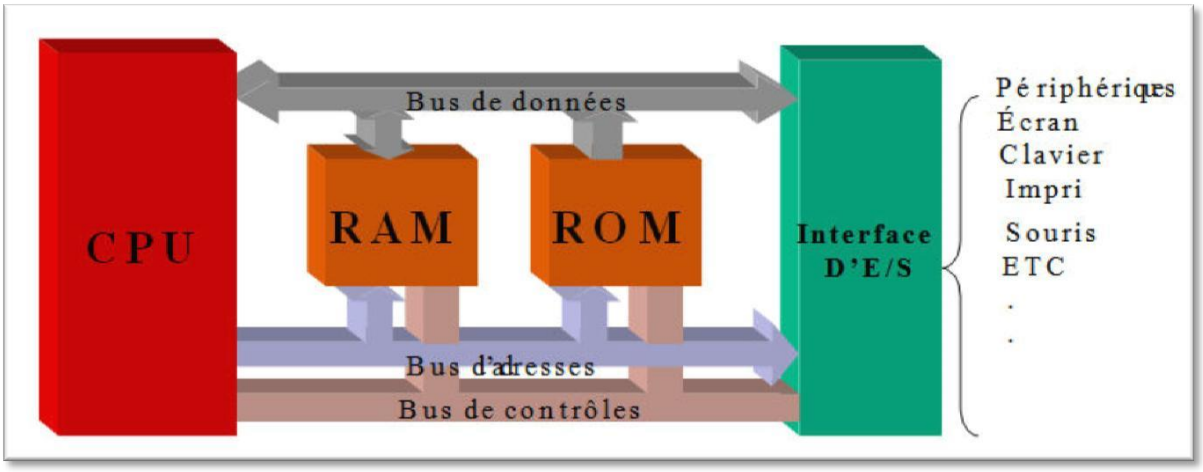


Figure 3:6 schéma simplifié du contenu type d'un microcontrôleur.

Un **microcontrôleur** est un circuit intégré qui rassemble les éléments essentiels

d'un ordinateur : processeur, mémoires (mémoire morte pour le programme, mémoire vive pour les données), unités périphériques et interfaces d'entrées-sorties. Les microcontrôleurs se caractérisent par un plus haut degré d'intégration, une plus faible consommation électrique (quelques milliwatts en fonctionnement, quelques nanowatts en veille), une vitesse de fonctionnement plus faible (quelques mégahertz à quelques centaines de mégahertz) et un coût réduit par rapport aux microprocesseurs polyvalents utilisés dans les ordinateurs personnels.

A. La partie logicielle :

Le logiciel de programmation des modules Arduino est une application Java, libre et multiplateformes, servant d'éditeur de code et de compilateur, et qui peut transférer le firmware et le programme au travers de la liaison série (RS232, Bluetooth ou USB selon le module). Il est également possible de se passer de l'interface Arduino, et de compiler les programmes en ligne de commande.

Le langage de programmation utilisé est le C++, compilé avec avr-g++, et lié à la bibliothèque de développement Arduino, permettant l'utilisation de la carte et de ses entrées/sorties. La mise en place de ce langage standard rend aisé le développement de programmes sur les plates-formes Arduino, à toute personne maitrisant le C ou le C++.

B. La partie matérielle :

Un module Arduino est généralement construit autour d'un microcontrôleur ATMEL AVR (Atmega328 ou Atmega2560 pour les versions récentes, Atmega168 ou Atmega8 pour les plus anciennes), et de composants complémentaires qui facilitent la programmation et l'interfaçage avec d'autres circuits. Chaque module possède au moins un régulateur linéaire 5V et un oscillateur à quartz 16 MHz (ou un résonateur céramique dans certains modèles). Le microcontrôleur est pré-programmé avec un boot loader de façon à ce qu'un programmateur dédié ne soit pas nécessaire.

Les modules sont programmés au travers une connexion série RS-232, mais les connexions permettant cette programmation diffèrent selon les modèles. Les premiers Arduino possédaient un port série, puis l'USB est apparu sur les modèles Diecimila, tandis que certains modules destinés à une utilisation portable se sont affranchis de l'interface de programmation, relocalisée sur un module USB-série dédié (sous forme de carte ou de câble).

L'Arduino utilise la plupart des entrées/sorties du microcontrôleur pour l'interfaçage avec les autres circuits. Le modèle Diecimila par exemple, possède 14 entrées/sorties numériques, dont 6 peuvent produire des signaux PWM, et 6 entrées analogiques. Les connexions sont établies au travers de connecteurs femelle HE14 situés sur le

dessus de la carte, les modules d'extension venant s'empiler sur l'Arduino. Plusieurs sortes d'extensions sont disponibles dans le commerce.

3.3 Alimentation

La carte Arduino UNO peut être alimentée par l'USB ou par une alimentation externe. La source est sélectionnée automatiquement.

La tension d'alimentation extérieure (hors USB) peut venir soit d'un adaptateur AC-DC ou de piles. L'adaptateur peut être connecté grâce à un 'jack' de 2.1mm positif au centre. Le raccordement vers un bloc de piles peut utiliser les bornes Gnd et Vin du connecteur d'alimentation (POWER). La carte peut fonctionner à l'aide d'une tension extérieure de 7 à 12 volts. Les broches (pins) d'alimentation sont les suivantes :

- VIN. La tension d'entrée positive lorsque la carte Arduino est utilisée avec une source de tension externe (à distinguer du 5V de la connexion USB ou autre source 5V régulée). Vous pouvez alimenter la carte à l'aide de cette broche, ou, si l'alimentation est fournie par le jack d'alimentation, accéder à la tension d'alimentation sur cette broche.
- 5V. La tension régulée utilisée pour faire fonctionner le microcontrôleur et les autres composants de la carte (pour info : les circuits électroniques numériques nécessitent une tension d'alimentation parfaitement stable dite « tension régulée » obtenue à l'aide d'un composant appelé un régulateur et qui est intégré à la carte Arduino). Le 5V régulé fourni par cette broche peut donc provenir soit de la tension d'alimentation VIN via le régulateur de la carte, ou bien de la connexion USB (qui fournit du 5V régulé) ou de tout autre source d'alimentation régulée.
- 3V3. Une alimentation de 3.3V fournie par le circuit intégré FTDI (circuit intégré faisant l'adaptation du signal entre le port USB de votre ordinateur et le port série de l'Atmega) de la carte est disponible : ceci est intéressant pour certains circuits externes nécessitant cette tension au lieu du 5V). L'intensité maximale disponible sur cette broche est de 50mA.
- GND. Broche de masse (ou 0V).

3.4 Mémoire

L'Atmega328 a 32 KB de mémoire (dont 0.5 KB pour le bootloader). Il a également 2 KB de SRAM et 1 KB de mémoire non volatile EPROM (qui peut être écrite et lue grâce à la librairie 'EEPROM').

3.5 Entrées et sorties

Chacune des 14 broches numériques de la Uno peut être utilisée en entrée (input) ou en sortie (output), en utilisant les fonctions pinMode (), digitalWrite (), et digitalRead ().

Elles fonctionnent en logique TTL (0V-5V) ; chacune pouvant fournir (source) ou recevoir un courant maximal de 40 mA et dispose si besoin est d'une résistance interne de 'pull-up'.

En outre, certaines broches ont des fonctions spécialisées :

- Serial : broche 0 (RX) et broche1 (TX). Permet de recevoir (RX) et de transmettre (TX) des données séries TTL. Ces broches sont raccordées à leurs homologues sur le chip Atmega8U2 spécialisé dans la conversion USB-to-TTL série.
- Interruptions externes 2 et 3. Ces broches peuvent être configurées pour déclencher une interruption sur une valeur LOW, sur un front montant ou descendant, ou encore sur le changement de valeur. (voir la fonction attachInterrupt () pour des détails).
- PWM : 3, 5, 6, 9, 10, and 11. Output 8-bit de PWM avec la fonction analogWrite ().
- SPI : 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Ces broches fournissent le support de communication SPI en utilisant la 'library' spécialisée
- LED : 13. Il y a une LED connectée à la broche numérique 13.

La carte Uno à 6 broches d'entrée analogiques, A0 à A5, chacune avec 10 bits de résolution (1024 valeurs différentes).

Par défaut les mesures sont effectuées de la masse à 5V (valeur de référence), mais il est possible de spécifier la valeur de référence en utilisant la broche VREF et la fonction analogReference ().

En outre, certaines broches ont des fonctions spécialisées :

- I2C : 4 (SDA) and 5 (SCL). Permettent le support du bus I2C (TWI) en utilisant le 'library' Wire.

Autres broches sur la carte :

- AREF. Tension de référence déjà mentionnée.
- Reset. Permet au niveau bas (LOW° de faire un reset du contrôleur. Elle est utilisée typiquement pour monter un bouton 'reset' aux cartes additionnelles ('shields') bloquant celui de la carte principale.

3.6 Communication

La carte Arduino Uno a de nombreuses possibilités de communications avec *l'extrieur*. L'Atmega328 possède une communication série UART TTL (5V), grâce aux broches numériques 0 (RX) et 1 (TX).

Un contrôleur Atmega8U2 sur la carte, gère cette communication série vers l'USB et apparaît comme un port de communication virtuel pour le logiciel sur l'ordinateur.

Le firmware de l'8U2 utilise le protocole USB, et aucun driver externe n'est nécessaire. Windows a cependant besoin d'un fichier .inf, à l'installation. Le logiciel Arduino possède un logiciel série (Telnet) intégré permettant l'envoi et la réception de texte. Les DELs RX et TX sur la carte clignoteront pour indiquer la transmission de données vers l'ordinateur.
Une librairie 'SoftwareSerial' permet la transmission de données série à partie de chacune des broches numériques du Uno.
L'Atmega328 supporte le bus I2C (TWI) et le protocole de communication synchrone maîtreesclave SPI. Le logiciel Arduino inclut un ensemble de fonctions pour mettre en œuvre l'un ou l'autre.

3.7 Programmation

La carte Arduino Uno peut être programmée directement avec « l'Arduino software ». L'Atmega328 sur la carte Uno est pré programmé avec un 'bootloader' qui permet de charger le code d'une nouvelle application sans utiliser un programmateur hardware externe. Il communique avec un ordinateur en utilisant le protocole STK500 d'ATMEL.
Mais vous pouvez programmer le contrôleur de la carte en utilisant le port ICSP (In-Circuit Serial Programming).
Le code source du firmware du contrôleur auxiliaire Atmega8U2 est disponible.

3.7.1 Reset automatique par Software

Il est possible d'effectuer un reset via le logiciel ARDIONO. En effet, la ligne DTR sur l'Atmega8U2 est connectée à la ligne Reset de l'Atmega328 à travers une capacité. Lorsque cette ligne est amenée à l'état logique 0, un signal Reset est envoyé au contrôleur.

3.7.2 Protection de surintensité USB

La carte Arduino Uno possède une protection par fusible pour le port USB si un courant de plus de 500mA est demandé. La déconnexion durera tant que la source de consommation excessive n'aura pas cessé.

3.7.3 Dimensions

Les longueurs et largeur maximales du PCB sont de 6,9 et 5,3 cm respectivement.

- **Schéma structurel :**

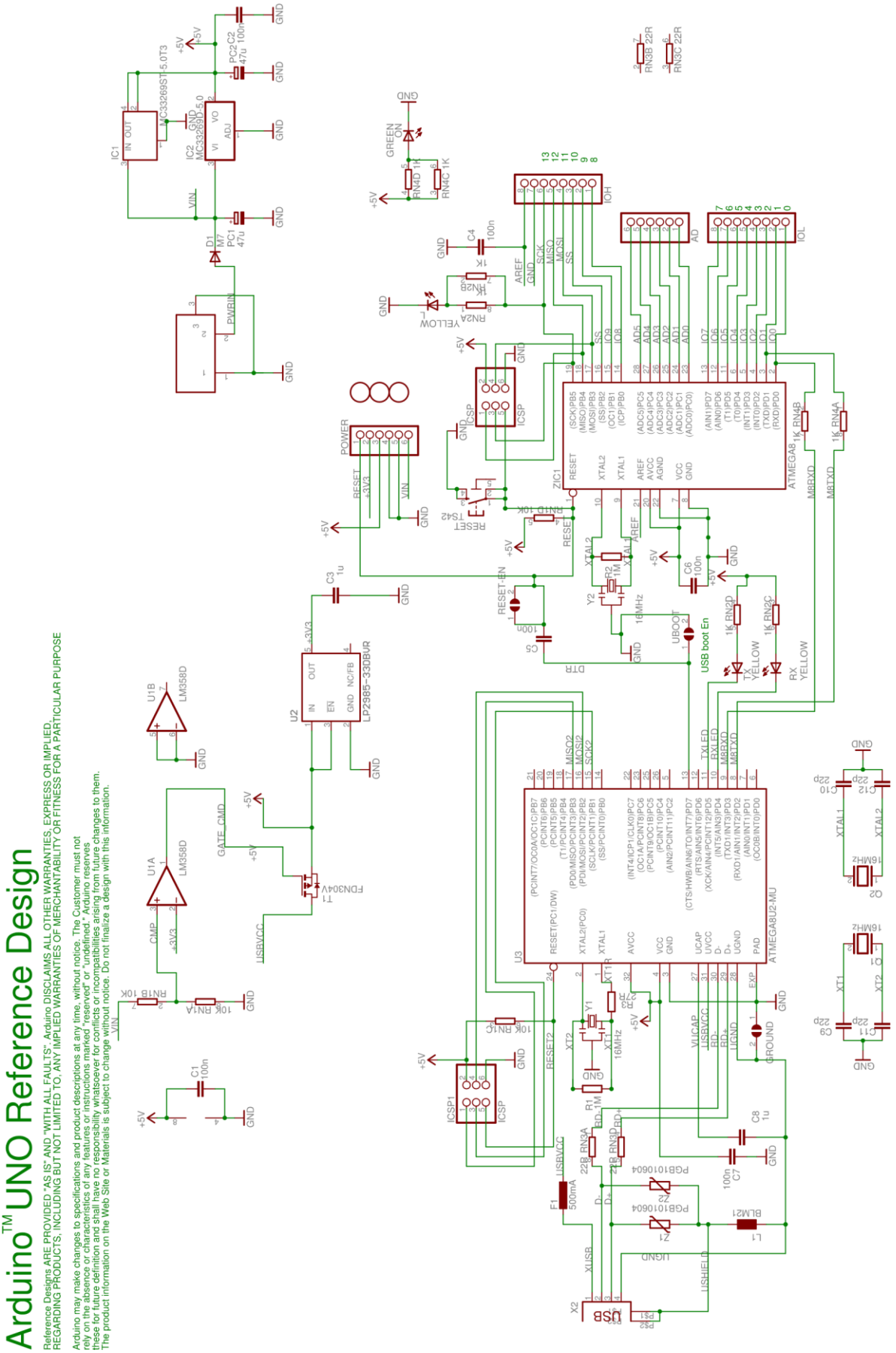


Figure 3:7 schéma structurel de l'ARDUINO UNO

3.7.4 Présentation de l'Espace de développement Intégré (EDI) Arduino

- **Description de l'interface :**

Le logiciel Arduino a pour fonctions principales :

- de pouvoir écrire et compiler des programmes pour la carte Arduino
- de se connecter avec la carte Arduino pour y transférer les programmes
- de communiquer avec la carte Arduino

Cet espace de développement intégré (EDI) dédié au langage Arduino et à la programmation des cartes Arduino comporte :

- une BARRE DE MENUS comme pour tout logiciel une interface graphique (GUI),
- une BARRE DE BOUTONS qui donne un accès direct aux fonctions essentielles du logiciel et fait toute sa simplicité d'utilisation,
- un EDITEUR (à coloration syntaxique) pour écrire le code de vos programme, avec onglets de navigation,
- une ZONE DE MESSAGES qui affiche indique l'état des actions en cours,
- une CONSOLE TEXTE qui affiche les messages concernant le résultat de la compilation du programme

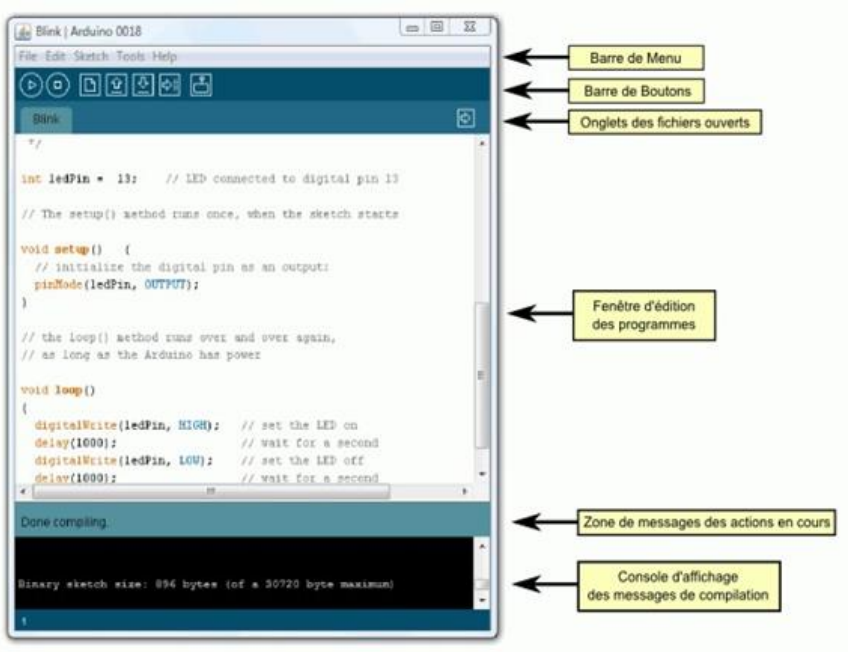


Figure 3:8 présentation des éléments de l'ARDUINO software

- un TERMINAL SERIE (fenêtre séparée) qui permet d'afficher des messages textes reçus de la carte Arduino et d'envoyer des caractères vers la carte Arduino. Cette fonctionnalité permet une mise au point facilitée des programmes, permettant d'afficher sur l'ordinateur l'état de variables, de résultats de calculs ou de conversions analogique-numérique : un élément essentiel pour améliorer, tester et corriger ses programmes.

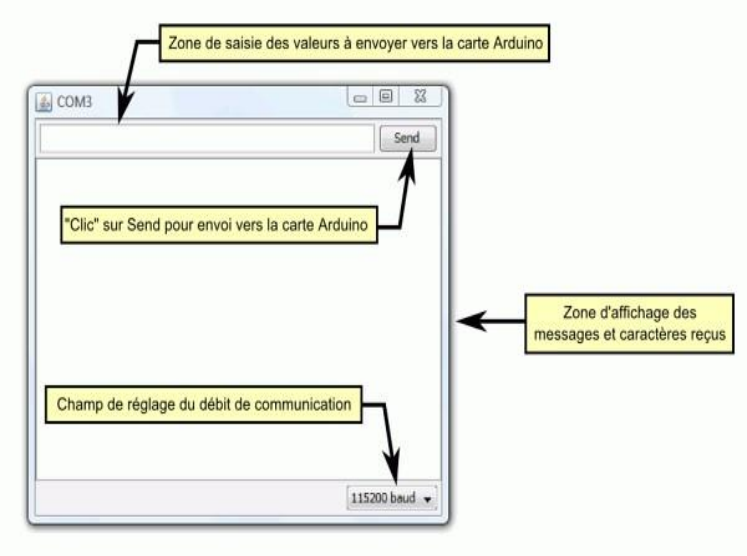


Figure 3:9 module TERMINAL SERIE

3.7.5 Description de la structure d'un programme

➢ Description générale des parties :

Un programme utilisateur Arduino est une suite d'instructions élémentaires sous forme textuelle, ligne par ligne. La carte lit puis effectue les instructions les unes après les autres, dans l'ordre défini par les lignes de code, comme mors d'une programmation classique. Cette structure se décompose en trois parties :

- Description des constantes et variables du programme
- Fonction principale : configuration des entrées/sorties et éléments à configurer (cette partie ne sera exécutée qu'une seule fois) dans le partie VOID SETUP()
- Fonction boucle : description du fonctionnement général du programme (gestion des interactions entre les entrées/sorties) dans la partie VOID LOOP()

Figure 3:10 : fenêtre graphique de l'EDI

- **Remarque :**

Il est possible d'ajouter des commentaires au programme. Pour cela on peut procéder de deux manière :

- à la fin de la ligne en ajoutant « // »
- en encadrant les commentaires entre « /* » et « */ »

3.7.6 Description détaillée des parties

3.7.6.1 Définition des variables et constantes

Dans cette partie, on déclare des éléments utilisés tout au long du programme : les constantes (statiques) et les variables (dynamiques). Ce sont des emplacements mémoire utilisés pour stocker des données (des valeurs) utilisables dans la suite du programme.

3.7.6.1.1 Variable

Une variable peut aussi bien représenter des données lues ou envoyées sur un des ports

analogiques ou numériques, une étape de calcule pour associer ou traiter des données, que le numéro 'physique' de ces entrées ou sorties. Une "variable" n'est donc pas exclusivement un paramètre variant dans le programme. On la déclare de la façon suivante :

TYPE_DE_LA_DONNEE NOM_DE_LA_DONNEE

- ✓ **Exemple :** int led

3.7.6.1.2 Constante

Une constante est une variable dont la valeur est inchangeable lors de l'exécution d'un programme. On la déclare de la façon suivante :

CONST TYPE_DE_LA_DONNEE NOM_DE_LA_DONNEE

- ✓ **Exemple :** const int led

3.7.6.1.3 Les différents types de données

En programmation informatique, un type de donnée, ou simplement type, définit les valeurs que peut prendre une donnée, ainsi que les opérateurs qui peuvent lui être appliqués.

Table 3-2 Types de données

NOM DU TYPE	VALEUR MIN/MAX	TAILLE EN MEMOIRE
VALEURS BINAIRES		
boolean	0/1	1 octet
VALEURS NUMERIQUES ENTIERES SIGNEES		
int	-32 768 / +32 767	2 octets
long	-2 147 483 648 / +2 147 483 647	4 octets
VALEURS NUMERIQUES ENTIERES NON SIGNEES		
byte	0 / +255	1 octet
unsigned int	0 / +65535	2 octets
word	0 / +65535	2 octets
unsigned long	0/ +4 294 967 295	4 octets

VALEURS NUMERIQUES A VIRGULE		
float	-3.4028235E+38 / +3.4028235E+38	4 octets
double	-3.4028235E+38 / +3.4028235E+38	4 octets
CARACTERES		
char	-128 / +127 (ASCII)	1 octet

3.7.6.2 Fonction principale : void setup ()

Cette fonction n'est exécutée qu'une seule fois au démarrage du programme. Elle permet la configuration des entrées et sorties de la carte. Les broches numériques de l'Arduino peuvent aussi bien être configurées en entrées qu'en sorties. Ici on a configuré LED_Pin_13 en sortie.

PinMode (nom, état) est une des quatre fonctions relatives aux entrées et sorties numériques que nous verrons plus bas.

```
void setup()
{
      ici se trouve la configuration des entrées et des sorties}
```

Figure 3:11 Instr. 3. 1

3.7.6.3 Fonction boucle : void loop()

Cette fonction loop() (boucle en anglais) fait exactement ce que son nom suggère et s'exécute en boucle sans fin, permettant à votre programme de s'exécuter. Dans cette boucle, on définit les opérations à effectuer.

La fonction loop() est obligatoire, même vide, dans tout programme.

```
void loop()
{
      ici se trouve la description générale du programme en boucle}
```

Figure 3:12 Instr. 3. 2

3.7.7 Compilation et programmation de l'ARDUINO

L'écriture d'un programme se déroule en plusieurs étapes.

3.7.7.1 Ecriture de l'algorithme

L'algorithme est une méthode pour résoudre un problème. L'algorithme est un moyen pour le programmeur de présenter son approche du problème à d'autres personnes. En

effet, un algorithme est l'énoncé dans un langage bien défini d'une suite d'opérations permettant de répondre au problème donné. Un algorithme doit donc être compréhensible même par un non-informaticien. Avant d'écrire un programme, il est donc nécessaire d'avoir un algorithme.

3.7.7.2 Ecriture du programme

La rédaction du programme se fait bien sur directement en rapport avec l'algorithme ci-dessus. Il faut absolument penser à mettre des commentaires compréhensifs par le non programmateur. Détaillé le programme et le partitionner en bloc logiques.

La rédaction du programme se fait dans la partie rayée ci-dessous :

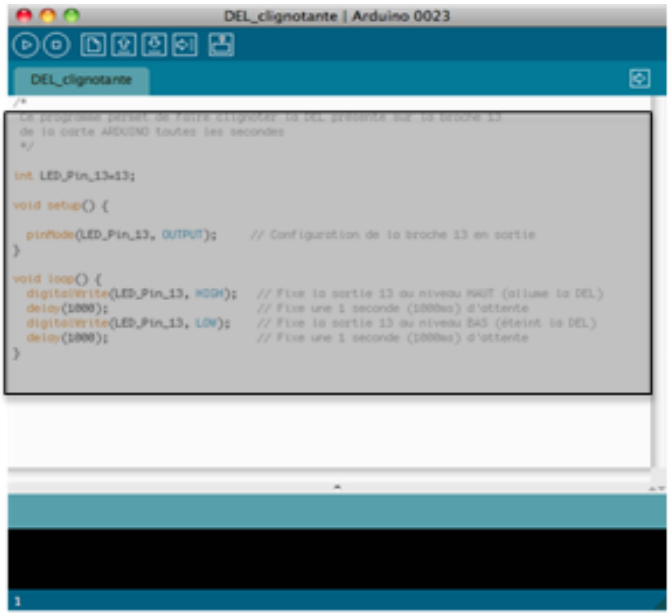


Figure 3:13 Ecriture du programme

3.7.7.3 Compilation du programme

Dans cette partie, on vérifie si le code contient des erreurs de syntaxes. En cas d'anomalie de compilation, le compilateur renseigne sur le type d'erreur et la ligne où elle se trouve.

Pour lancer la compilation, il faut appuyer sur.  A ce moment-là, le bouton devient jaune et la zone de message affiche « Compiling » indiquant que la compilation est en cours.

Si la compilation se déroule sans erreur, le message «Done compilling » apparaît, suivi de la taille du programme.

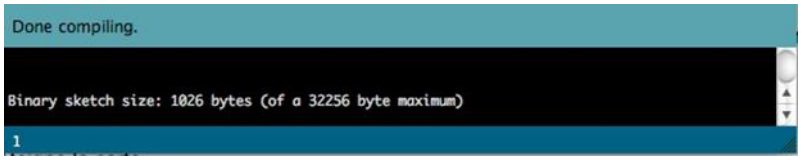


Figure 3:14 la compilation

Un compilateur est un programme informatique qui traduit un langage (appelé le langage source) en un autre (le langage cible), généralement dans le but de créer un fichier exécutable.

Un compilateur sert le plus souvent à traduire un code source écrit dans un langage de programmation en un autre langage, habituellement un langage d'assemblage ou un langage machine. Le programme en langage machine produit par un compilateur est appelé code objet.

3.7.7.4 Sélection de la cible et du port série

Avant de transférer le programme vers la carte Arduino, il faut, si ce n'est déjà fait, sélectionner la bonne carte Arduino (la bonne cible) depuis le menu **Tools>Board** (Outils>Carte).

La carte doit évidemment être connectée à l'ordinateur via un câble USB.

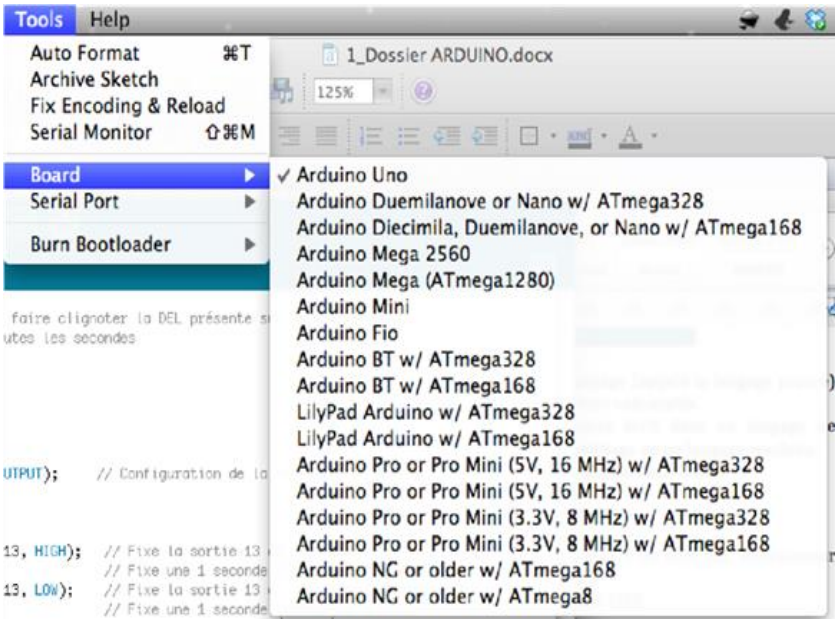


Figure 3:15 sélectionner le bon port série

Vous devez également sélectionner le bon port série depuis le menu **Tools > Serial Port** (Outils > Port Série).

- **Remarque :**

Selon le système d'exploitation le nom du port série peut différer :

- Sous Mac, sélectionner le port /dev/tty.usbserial-1B1 (pour une carte USB)
- Sous Windows, sélectionner le port COM1, COM2 (pour une carte série) ou COM4 ou supérieur (pour une carte USB)
- Sous Linux, sélectionner le port /dev/ttyUSB0, /dev/ttyUSB1 ou équivalent.

3.7.8 Transfert du programme vers la carte ARDUINO

Une fois que vous avez sélectionné le bon port série et la bonne carte Arduino, cliquez sur le bouton UPLOAD  (Transfert vers la carte) dans la barre d'outils, ou bien sélectionner le menu **File>Upload to I/O board** (Fichier > Transférer vers la carte).

Sur la plupart des cartes, vous devez voir les LEDs des lignes RX et TX clignoter rapidement, témoignant que le programme est bien transféré. Durant le transfert, le bouton devient jaune et le logiciel Arduino affiche un message indiquant que le transfert est en cours.

3.8 Conclusion

Dans ce chapitre nous avons d'abord présenté l'outil de commande de notre bras Manipulateur (Arduino uno), leurs principaux composants (matériel et logiciel). Le système Arduino, nous donne la possibilité d'allier les performances de la programmation à celles de L'électronique.

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 a l'informatique embarque. On pourra ainsi enrichir tout ces projets d'un microcontrôleur pour leur donner une plus value importante.

Chapitre 4

Conception et assemblage

4 Chapitre 4 conception et assemblage

4.1 Introduction

Dans ce chapitre, nous présentons quelques expérimentations qui visent à valider que l'arduino décrit au chapitre 3 répond bien aux problématiques soulevées au chapitre 1. Nous testerons le robot dans différentes versions : avec la manipulation uniquement, en ajoutant la matrice inertielle ou les forces, ou enfin en fusionnant facteurs dynamiques. Nous pourrons ainsi valider, à l'aide de différentes réalités terrain, l'estimation de la cinématique du repère local dans le repère globale et la cohérence des mesures entre ce dernier et les efforts estimés. Nous verrons aussi les performances que nous obtenons en utilisant la matrice inertielle pour estimer les forces aux contacts, et validerons ainsi expérimentalement ce que l'étude d'observabilité du chapitre 2 et 3 démontre. Enfin, la qualité de mouvement nous permettra de valider les hypothèses que nous avons faites pour développer le modèle dynamique.

4.1.1 Modélisation de la dynamique du robot

Comme nous l'avons vu dans l'état de l'art, une question qui revient régulièrement dans les différents travaux et le choix du modèle. Le modèle retenu doit être le plus complet possible pour engendrer le moins d'erreurs possible. Mais il doit aussi pouvoir être calculé suffisamment rapidement pour être utilisable en temps réel.

4.1.2 Dispositif expérimental

La première expérimentation consiste à tester la réponse de notre ARDUINO sans phase de mise à jour. Sur le robot, avec les tâches décrite en chapitre 3, nous demandons un centre de masse de référence qui suit un échelon de 2 cm. Ainsi, la tâche est générée. Les mouvements des articulations qui permettent d'obtenir la nouvelle position du centre de masse sont décrits. Nous regardons alors l'évolution de la position du poignet estimée à celle calculée à partir de la configuration initiale. Les résultats sont montrés dans la figure 4.4 Nous pouvons voir que, sans l'application de l'arduino, qui compense les défauts de modèle, nous pouvons reconstruire la position du l'organe terminal avec une bonne approximation. Cependant, Nous pouvons voir un biais de 0.5 mm entre les deux états stables sur le bras X (vers l'avant). Nous constatons aussi que les oscillations de l'ensemble ne sont pas considérables. Comme la description de la carte maire étendu en chapitre 3 le suggère, il existe deux méthodes pour améliorer ces résultats. Nous pouvons améliorer par changement de l'environnement de compilation et par utilisation de capteurs de compensation.

4.1.3 Caractéristique technique

Table 4-1 Caractéristique technique

DESCRIPTION	**COMPOSANTS :** - 4 SERVOMOTEURS DONT : - 3 SERVOMOTEURS TYPE : MG90S - 1 SERVOMOTEURS TYPE : SG90 - VIS + ECROU **DESIGNATIONS :** - M4-30*4 - M3-15*12 - M4-15*4 - ECROU : DIAMETRE 4*24 - DIAMETRE 3*4
DIMENSION. **TAILLE.POIDS**	**LONGUEUR :** 350 mm **LONGUEUR MAXIMALE DU BRAS ETENDU :** 200 mm s **POIDS :** 200g
PUISSANCE. **VOLTAGE**	6v

4.2 Modules du Robot [Algérie Machines- Outils 02 :(Alg.M.-O.02)]

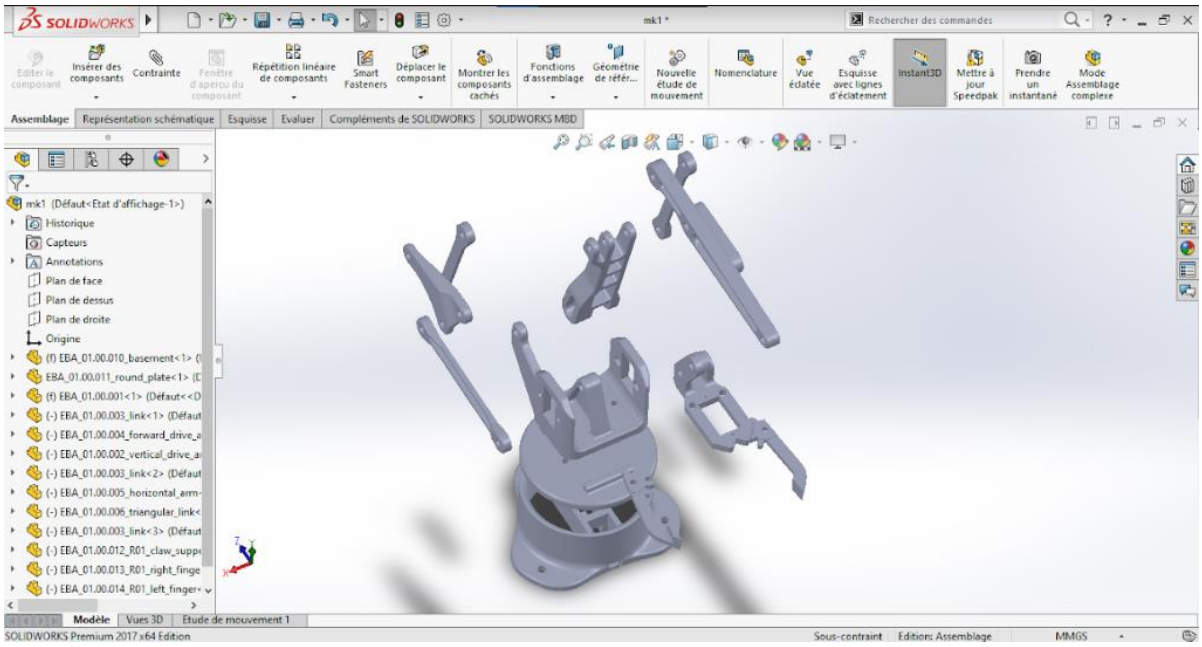


Figure 4:1 a : Robot [Algérie Machines-Outils -02-] Vue d'ensemble

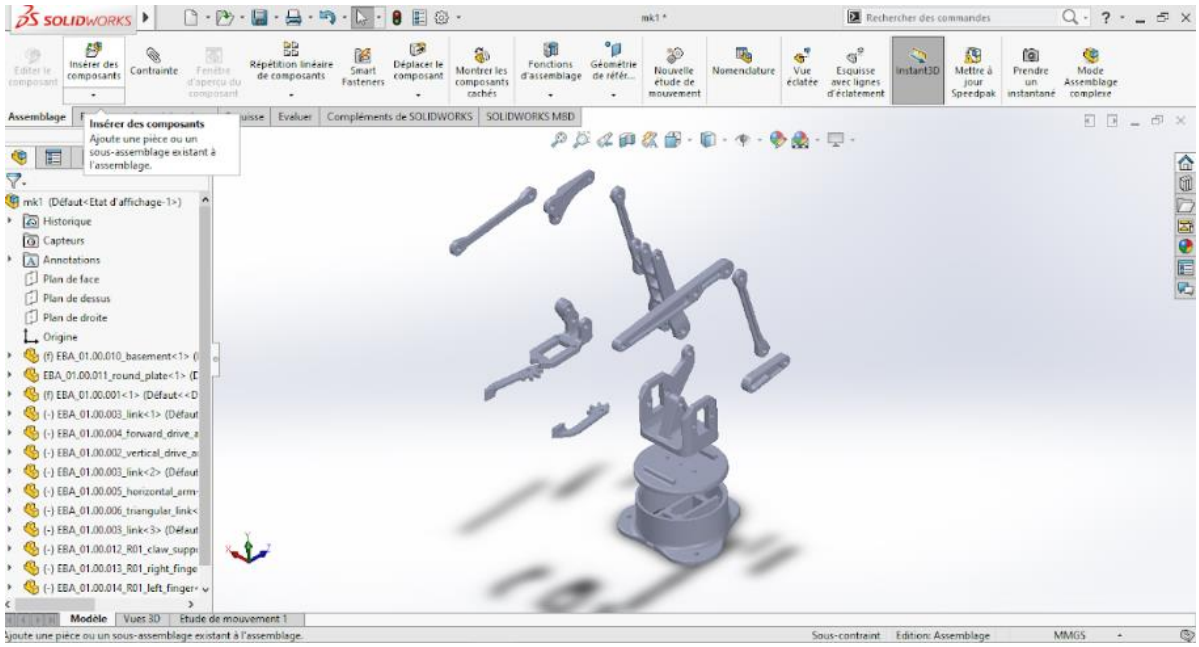


Figure 4:2 b : Robot [Algérie Machines-Outils -02-] Vue éclaté

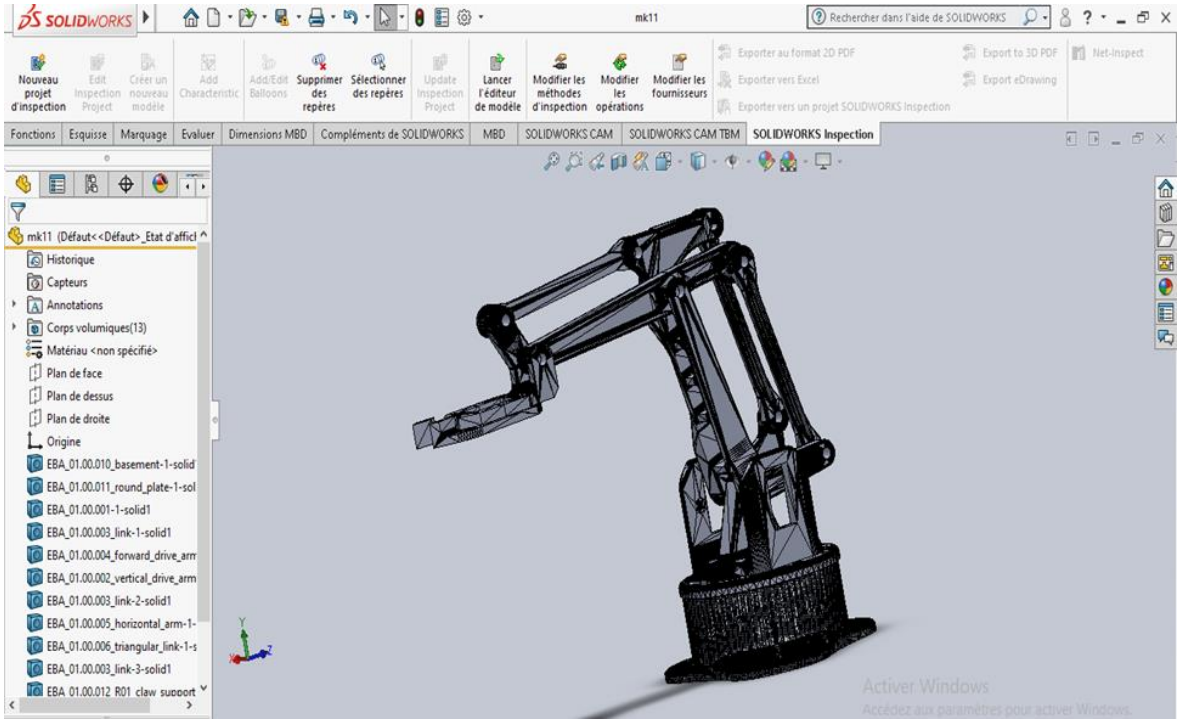


Figure 4:3 c : Robot [Algérie Machines-Outils -02-] Assemblage

4.2.1 Étapes de l'assemblage des Modules de Déplacement

4.2.1.1 Étape 1

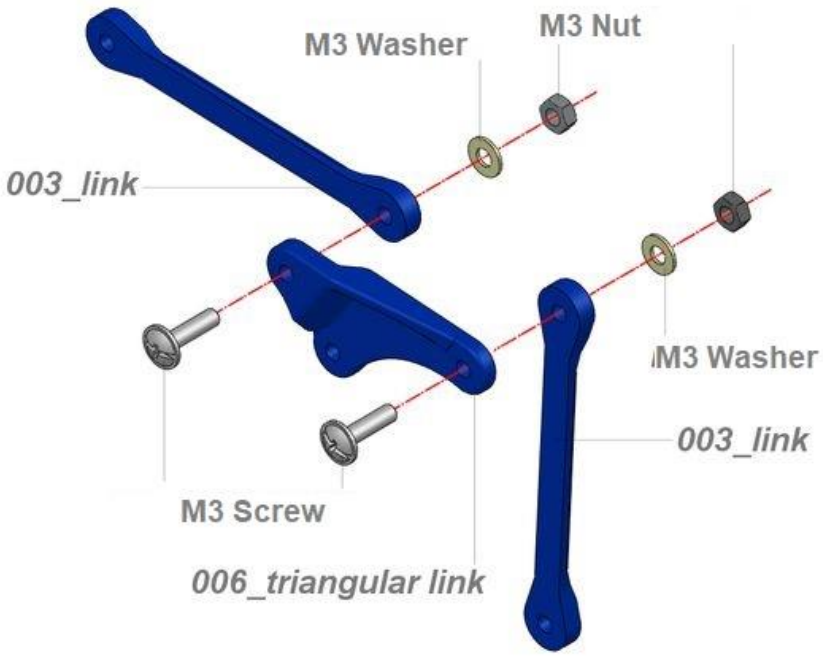


Figure 4:4 b : Assemblage vis-écroux

Connectez deux bras de liaison (003) à la liaison triangulaire (006). Gardez les vis à tête ronde

M3 sur le côté intérieur comme indiqué sur l'image et les écrous sur le côté extérieur.

IMPORTANT je conçois tous les trous de joints assez précis pour permettre de les rendre plus précis à l'aide d'un foret Les écrous sont à serrer jusqu'au blocage du joint, puis par conséquent vous devez les desserrer jusqu'à obtenir un mouvement fluide avec un jeu inférieur entre les composants. Cette règle est valable et doit être appliquée également pour le joint suivant qui implique l'utilisation d'écrous.

4.2.1.2 Étape 2

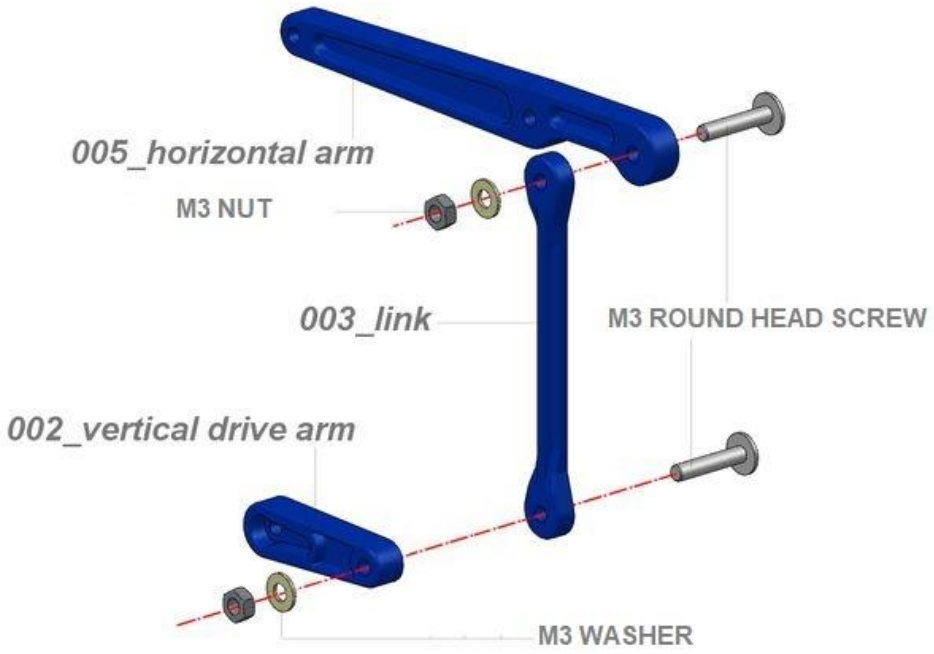


Figure 4:5 Assemblage vis-écroux Étape 2

Connectez le lien (003) au joint arrière du bras horizontal (005). La partie inférieure de la biellette (003) doit être reliée au bras d'entraînement vertical (002) comme illustré. Entre les deux maillons intercalent trois rondelles M3, ceci pour mieux les aligner avec le bras vertical Gardez les vis à tête ronde M3 sur le côté intérieur et les écrous à l'extérieur.

4.2.1.3 Étape 3

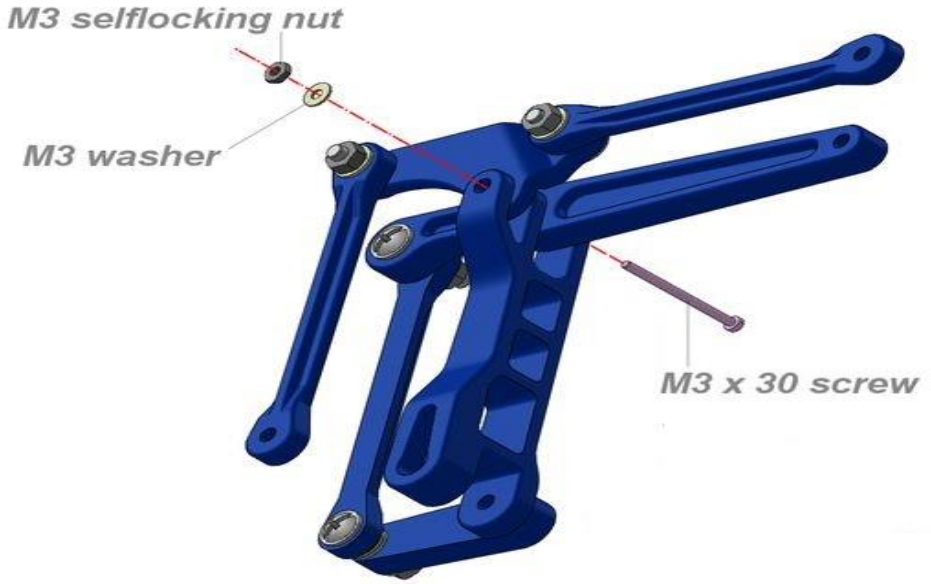


Figure 4:6 Assemblage vis-écroux Étape 3

Connectez maintenant les deux liaisons pré-assemblées au bras d'entraînement avant (004). Mettre en position bras horizontal (005) et biellette triangulaire (006) alignés avec la liaison supérieure du bras d'entraînement avant (004). Fixez toutes les pièces avec la vis M3x30, bloquée par l'écrou de l'autre côté. Vérifiez la liberté de mouvement et si tout va bien, passez à l'étape suivante.

4.2.1.4 Étape 4 : Assemblage de la base

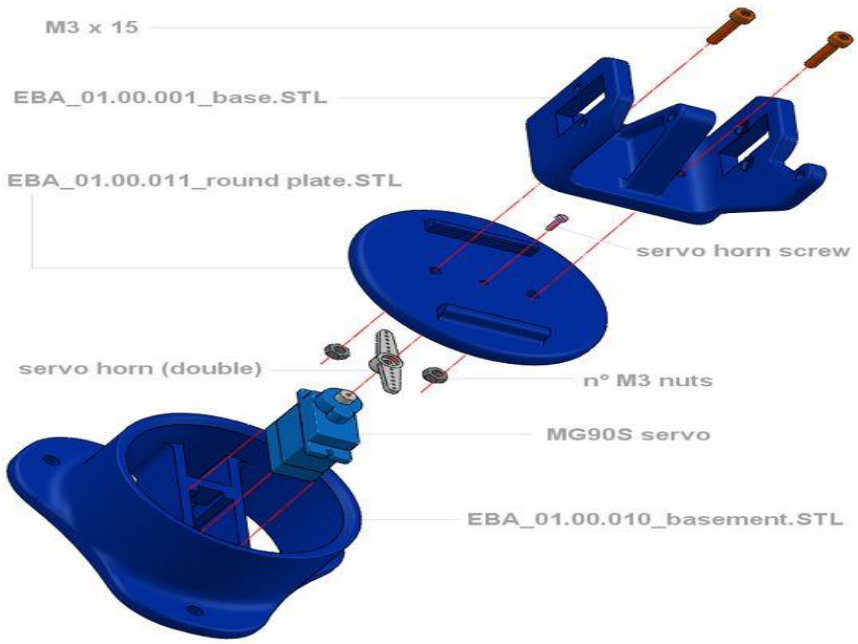


Figure 4:7 Assemblage de la base

Liste des pièces :

- n° 1 EBA_01.00.001_base.stl
- n° 1 EBA_01.00.011_round_plate.stl
- n° 1 EBA_01.0.010_sous-sol.stl
- n° 1 servo Tower Pro SG90 ou MG90S avec pavillon double bras
- n° 1 vis de fixation du palonnier
- n° 2 vis M3 x 15 (VTCEI)
- n° 3 écrous M3

4.2.1.5 Étape 5

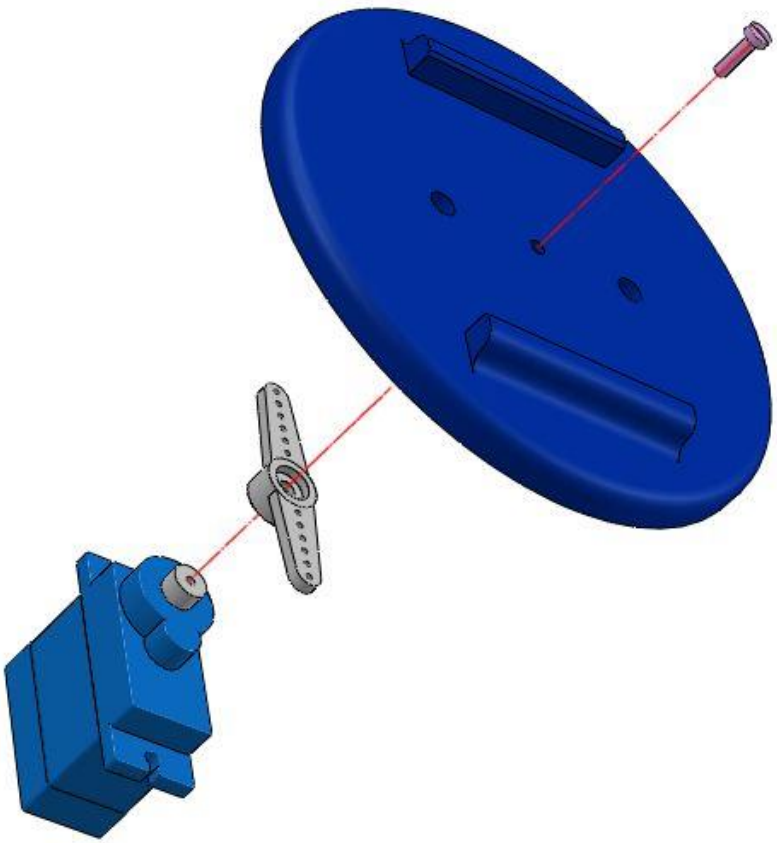


Figure 4:8 Assemblage de la base Étape 5 a

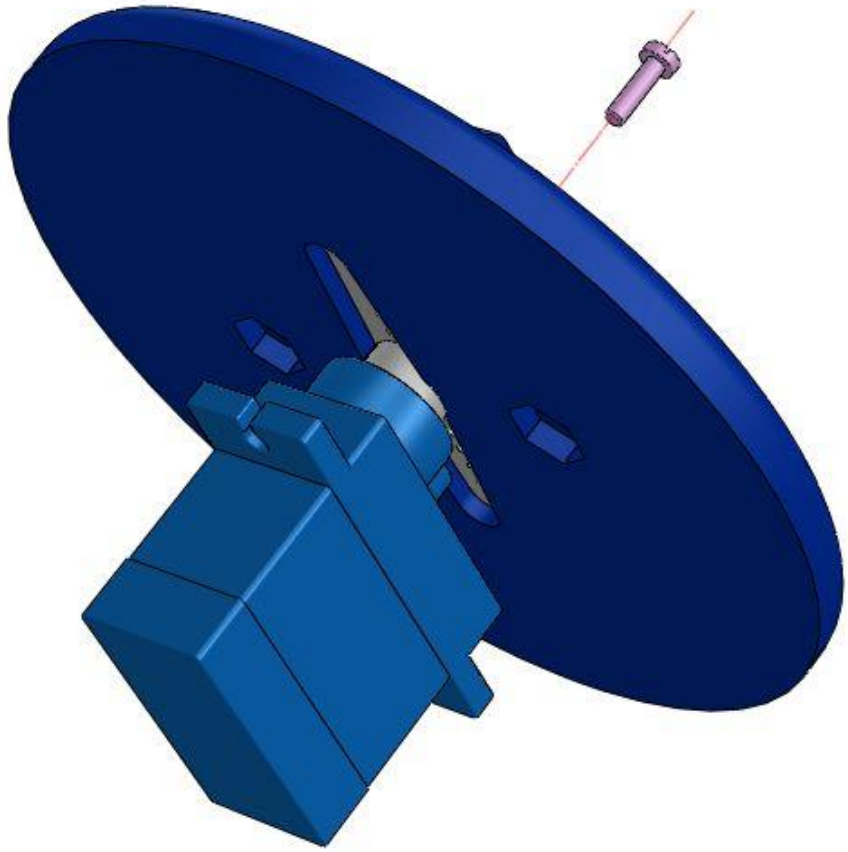


Figure 4:9 Assemblage de la base Étape 5 b

Assurez-vous que le servo est en position neutre, puis installez la corne à double bras sur l'arbre cannelé en gardant les bras parallèles au corps du servo Insérez le cornet à l'intérieur du boîtier sous la plaque ronde et fixez le servo à la plaque à l'aide d'une des deux vis longues fournies avec le servo (la petite est trop courte en raison de l'épaisseur de la plaque ronde)

4.2.1.6 Étape 6

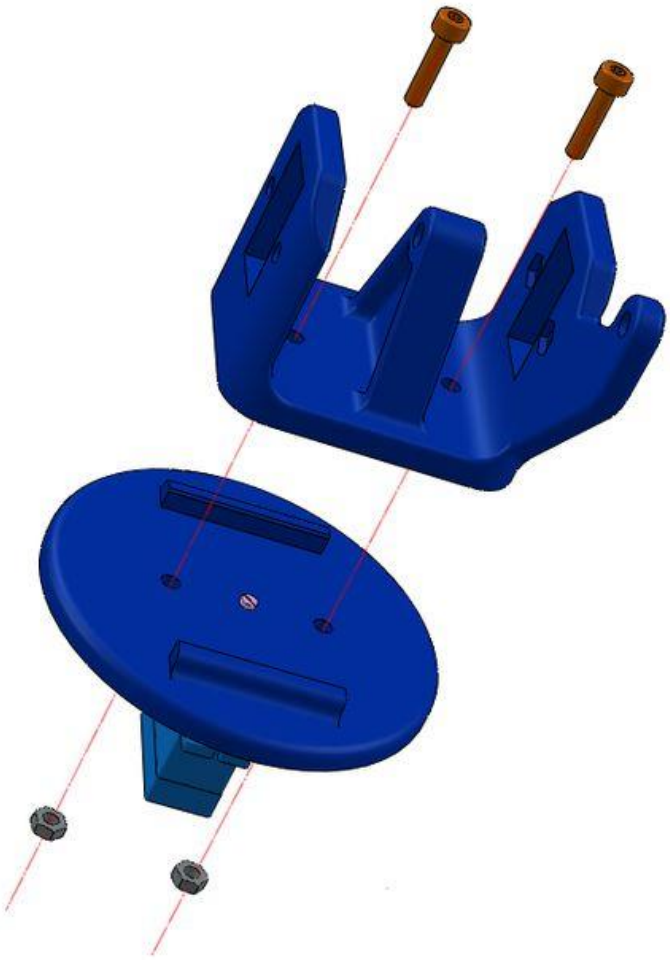


Figure 4:10 Assemblage de la base Étape 6 a

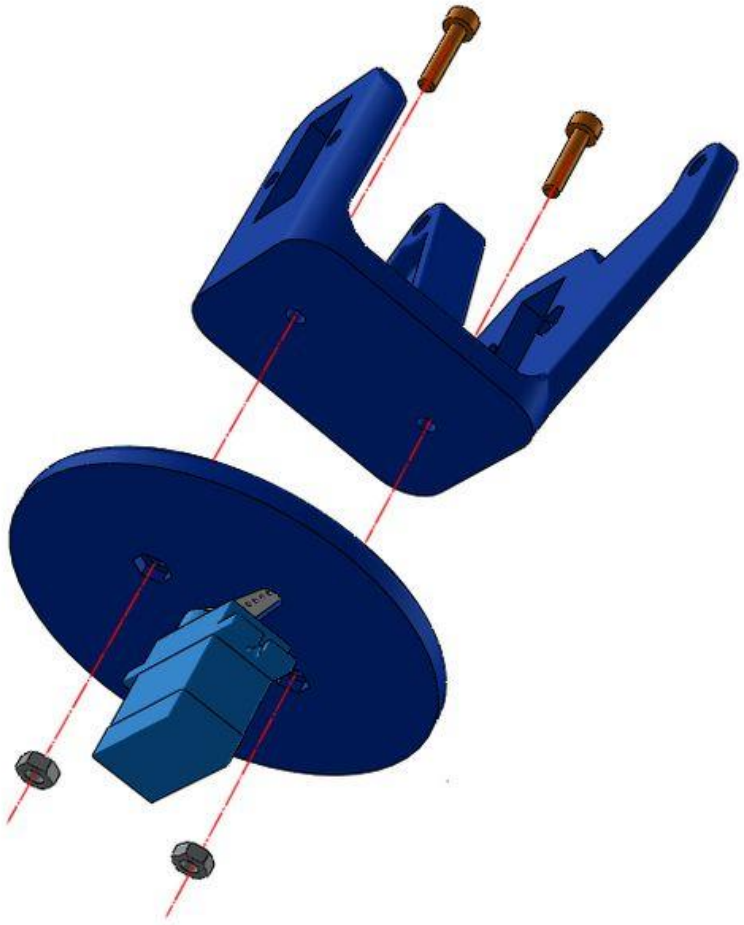


Figure 4:11 Assemblage de la base Étape 6 b

Mettre en place la base entre les deux épaulements de la plaque et fixer ensemble à l'aide des deux vis et écrous M3. Il y a deux logements hexagonaux en dessous, donc les écrous seront maintenus en position pendant le serrage

4.2.1.7 Étape 7

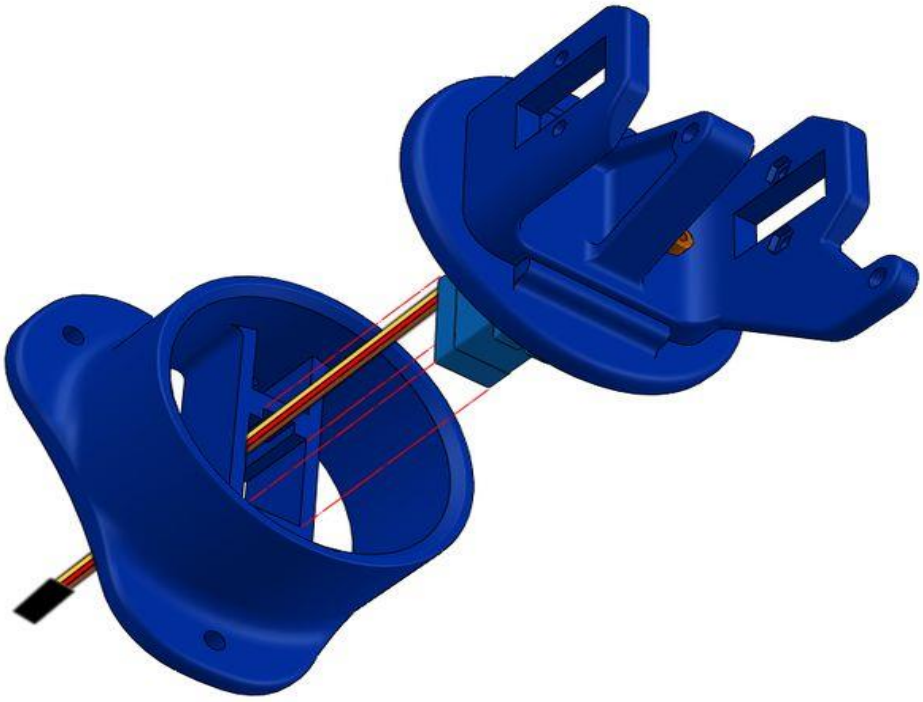


Figure 4:12 Assemblage de la base Étape 7 a

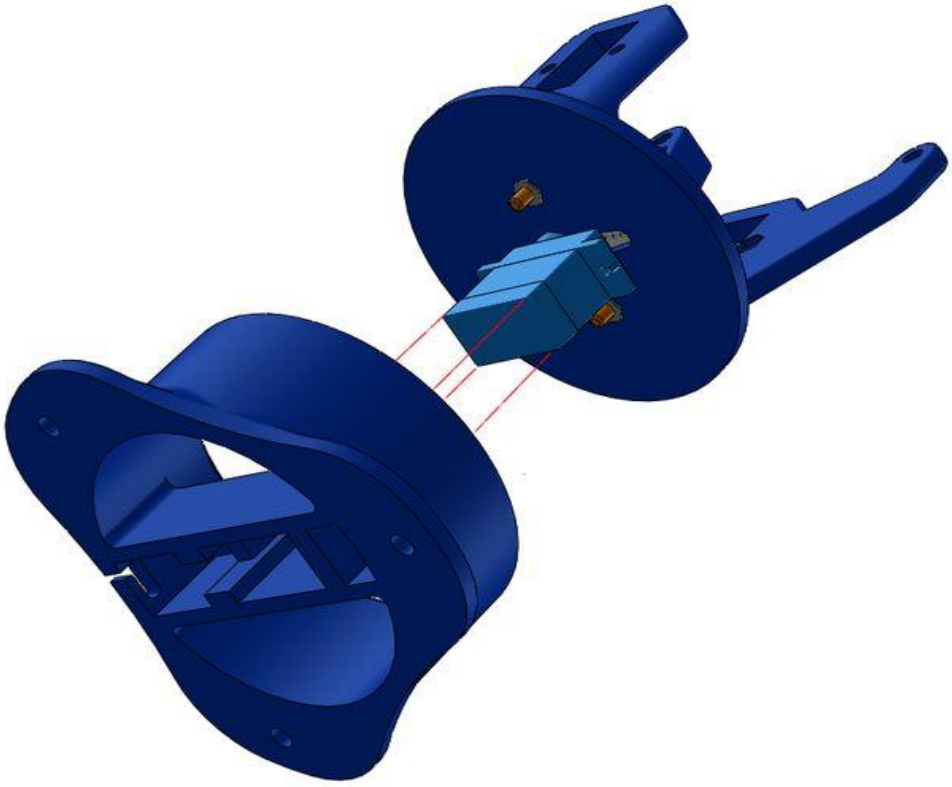


Figure 4:13 Assemblage de la base Étape 7 b

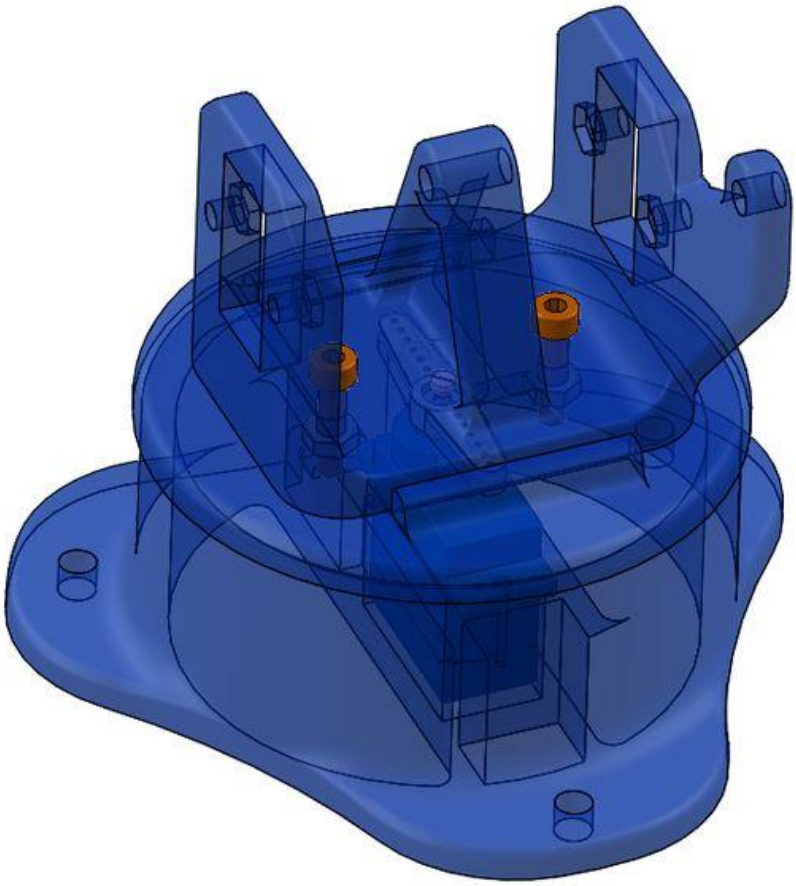


Figure 4:14 Assemblage de la base Étape 7 c

Alignez le servo et introduisez le câblage dans la partie centrale du sous-sol. Tirez doucement sur le fil pour le rendre droit tout en continuant à le pousser en logeant le servo Le fil est ensuite maintenu en position en le faisant passer par un trou frontal

4.2.1.8 Étape 8 : Assemblage de la pince

Figure 4:15 Assemblage de la pince Étape 8

Liste des pièces:

• n° 1 servo TowerPro MG90S ou SG90 avec pavillon monobras

• n° 1 vis de fixation du palonnier

• n° 1 EBA_01.00.012_claw support.stl

• n° 1 EBA_01.00.015_drive gear.stl

• n° 1 EBA_01.00.014_left finger.stl

• n° 1 EBA_01.00.016_driver gear.stl

• n° 1 EBA_01.00.013_right finger.stl

• n° 2 vis M3 x 20 (TCEI)

• n° 3 écrous M3

4.2.1.9 Étape 9

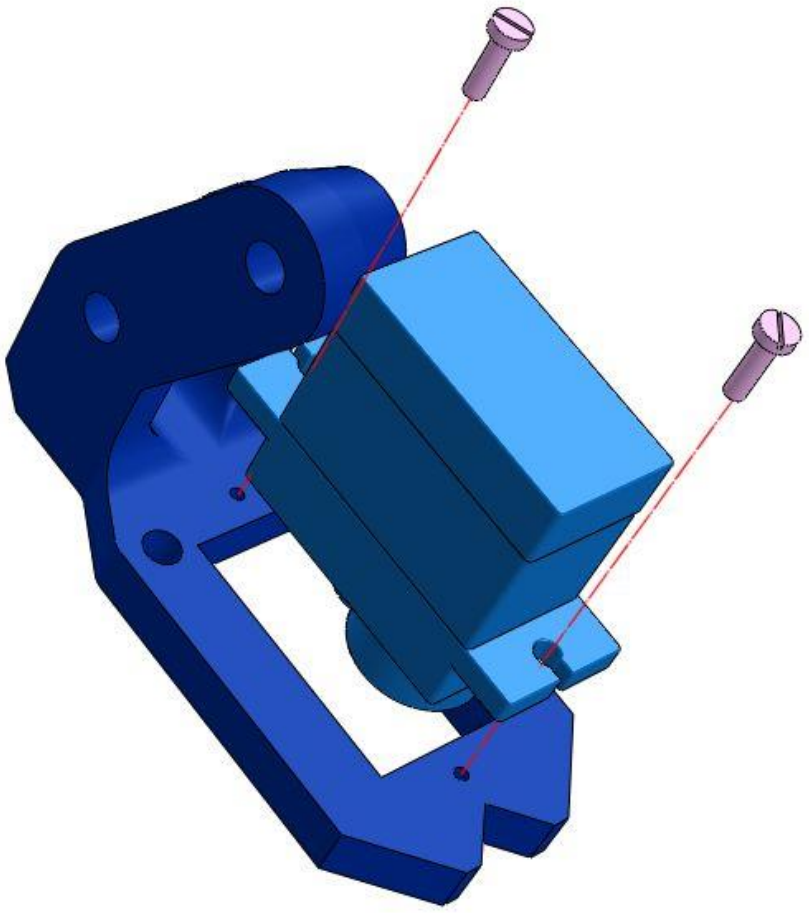


Figure 4:16 Assemblage de la pince Étape 9

Fixez le servo au support de griffes à l'aide des deux vis de fixation fournies avec le servo Gardez l'arbre de sortie vers l'avant.

4.2.1.10 Étape 10

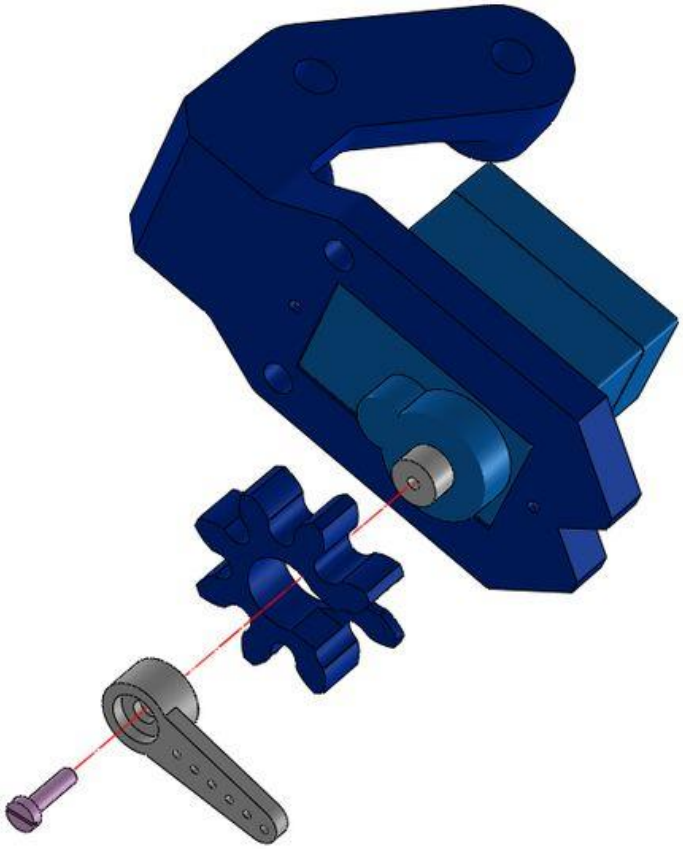


Figure 4:17 Assemblage de la pince Étape 10 a

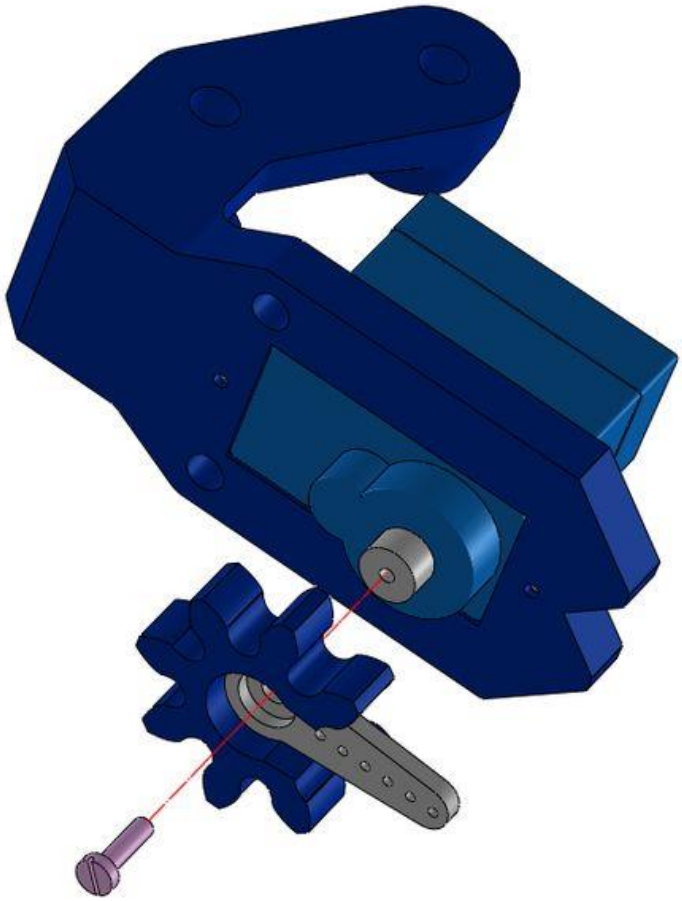


Figure 4:18 Assemblage de la pince Étape 10 b

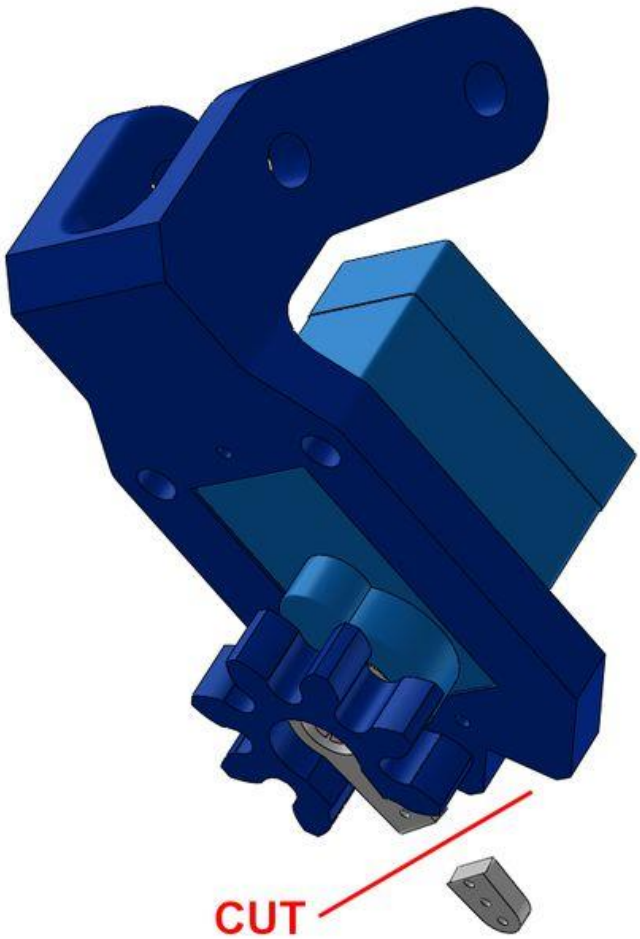


Figure 4:19 Assemblage de la pince Étape 10 c

Insérez le klaxon dans le pignon mené puis fixez le klaxon à l'arbre du servo à l'aide de la vis fournie Le klaxon doit être aligné vers l'avant avec le servo en position neutre. Couper la partie dépassant de la corne de l'engrenage à l'aide d'un cutter

4.2.1.11 Étape 11

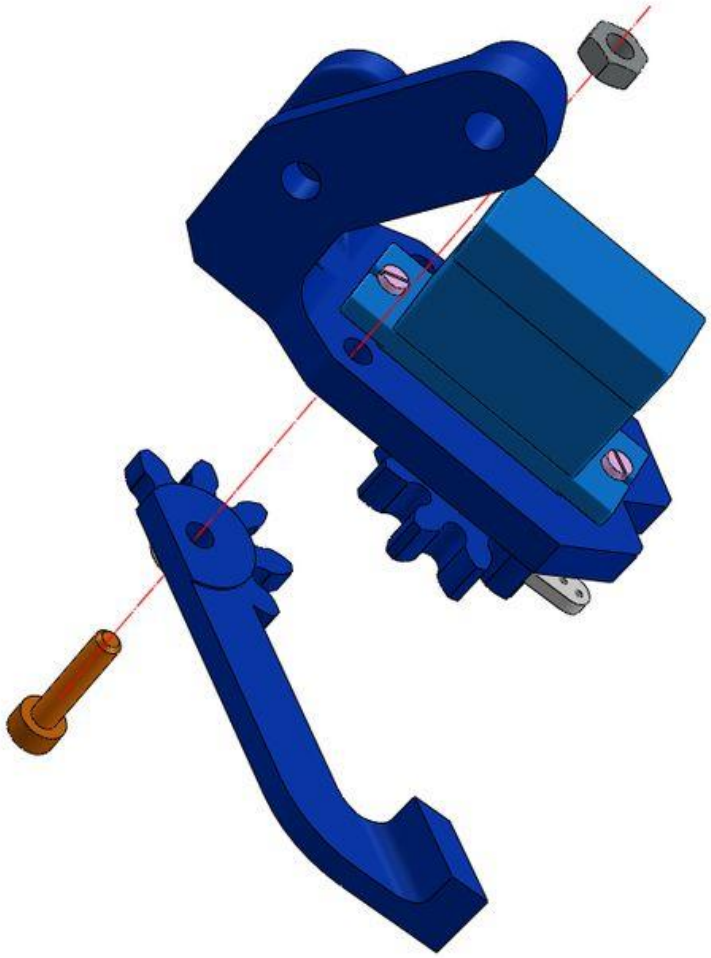


Figure 4:20 Assemblage de la pince Étape 11

Insérez une vis M3 dans le trou central connectez-la au support de griffe puis serrez l'écrou en vérifiant la liberté de mouvement

4.2.1.12 Étape 12

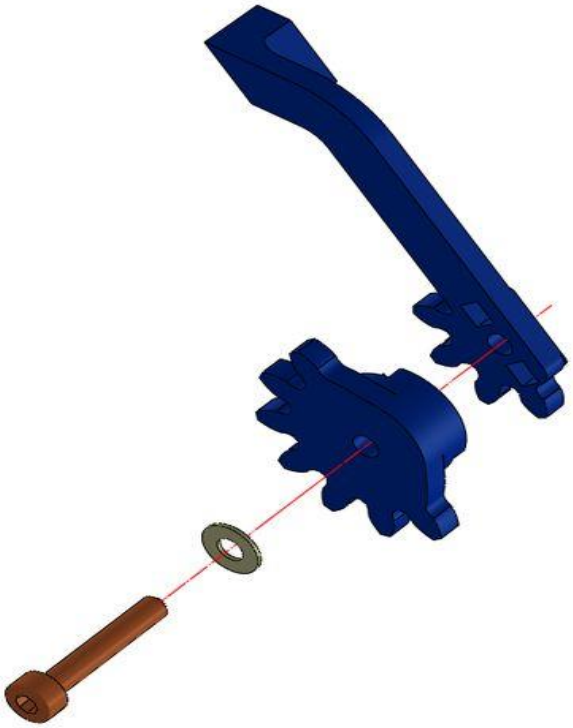


Figure 4:21 Assemblage de la pince Étape 12a

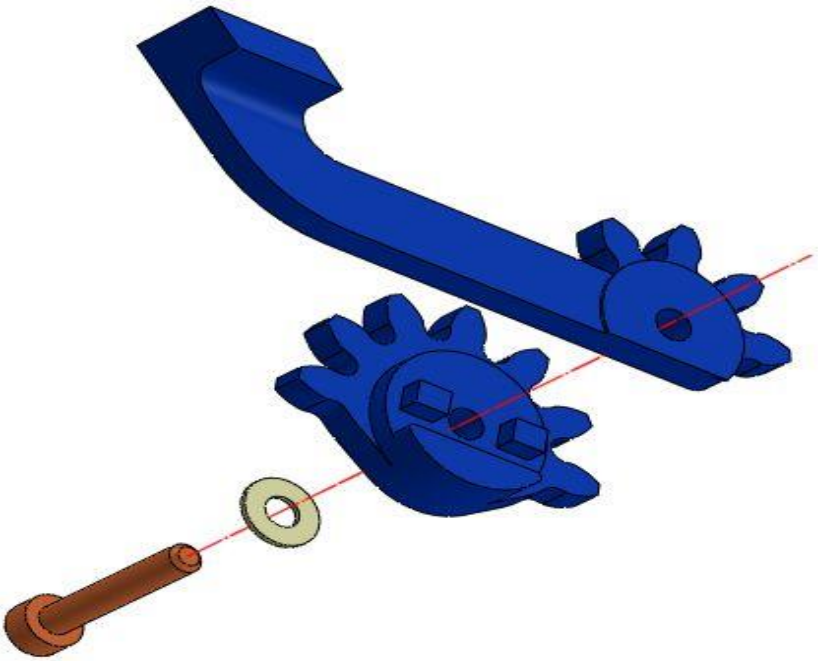


Figure 4:22 Assemblage de la pince Étape 12 b

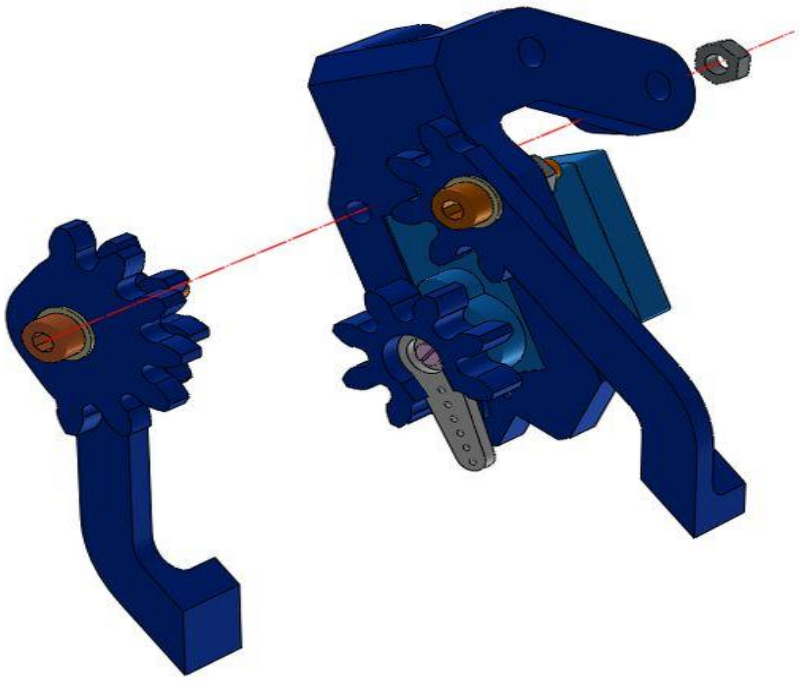


Figure 4:23 Assemblage de la pince Étape 12 c

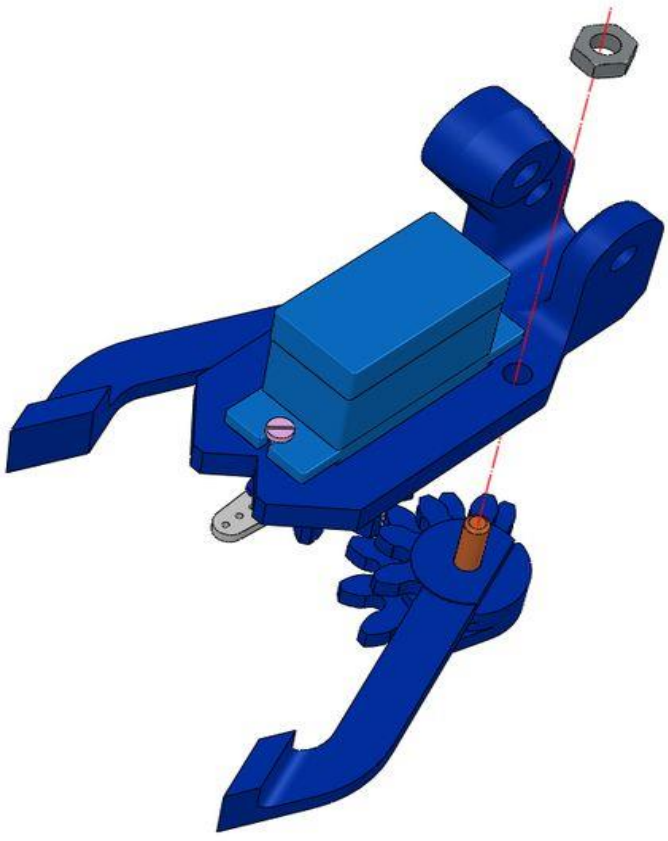


Figure 4:24 Assemblage de la pince Étape 12 d

Insérez les deux broches du pignon mené dans les trous dédiés sur le doigt gauche. Le pignon mené a également un épaulement qui doit être aligné avec le côté latéral du doigt. Si vous rencontrez des difficultés pour les coupler, réduisez les interférences à l'aide d'un fichier. Une fois accouplé insérez une vis M3 dans le trou central et fixez le doigt sur le support de griffe La pince est maintenant prête à être installée sur le bras horizontal de l'EEzybot Vérifiez la liberté de mouvement de la pince manuellement ou à l'aide d'un testeur d'asservissement.

4.2.1.13 Étape 13 : Assemblage final.

Figure 4:25 Étape 13 : Assemblage final.

Nous avons maintenant les trois sous-ensembles principaux prêts à être connectés les uns aux autres. Prochaine étape nous rejoindrons la base avec les bras principaux

4.2.1.14 Étape 14

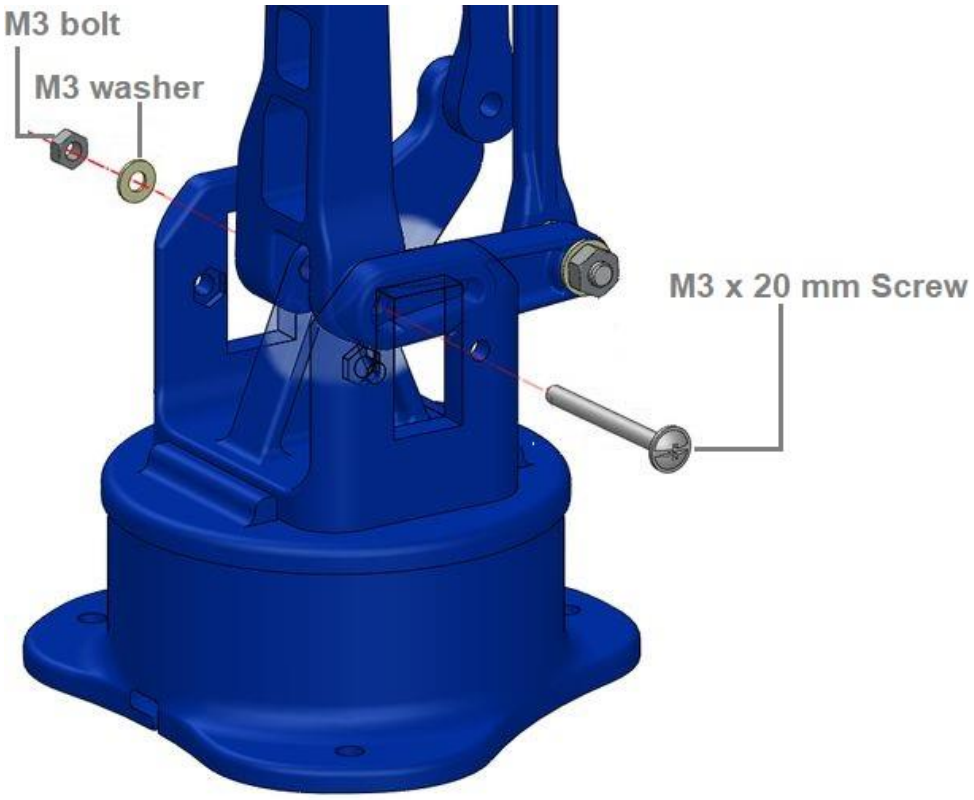


Figure 4:26 Assemblage final Étape 14 a

Figure 4:27 Assemblage final Étape 14 b

Pour joindre la base aux bras principaux, alignez l'axe des pièces et insérez d'un côté la vis M3 de 20 mm de long. De plus, le bras court du servo qui entraîne le mouvement vertical doit être inséré après la vis comme indiqué sur les photos. Vérifiez la liberté de mouvement.

4.2.1.15 Étape 15

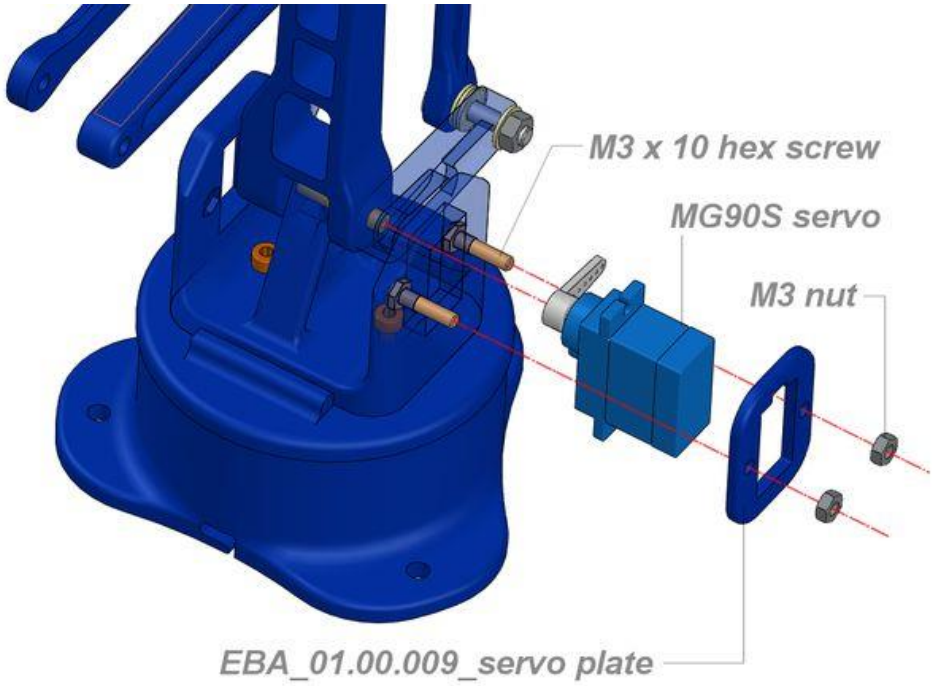


Figure 4:28 Assemblage final Étape 15 a

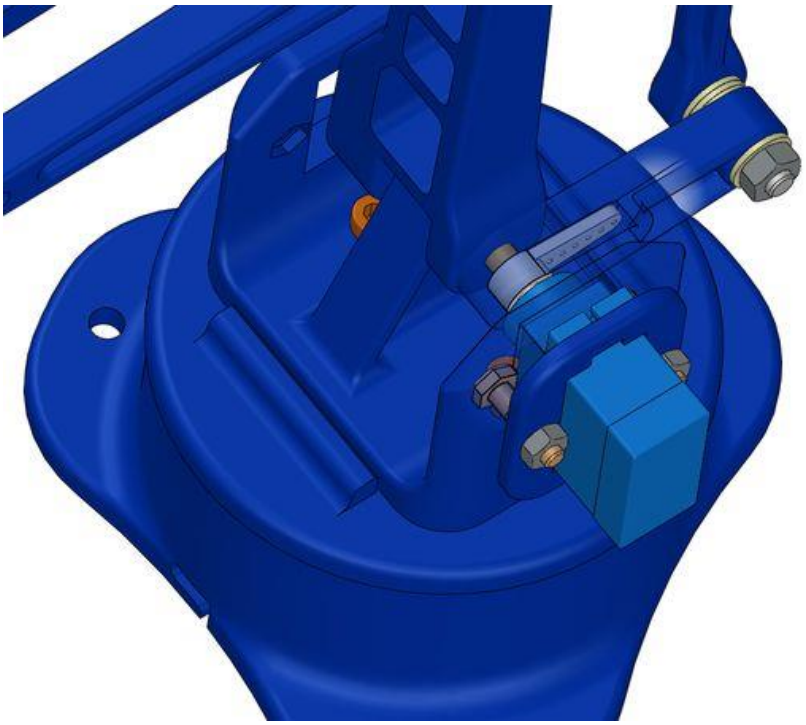


Figure 4:29 Assemblage final Étape 15 b

Il est maintenant temps d'installer le servo qui entraîne le mouvement vertical du bras. Mettez dans les réceptacles dédiés deux vis hexagonales M3x12. Le servo doit être en position neutre avec le klaxon à 90 degrés sur le côté droit avec la plaque de presse (009) installée (Faire passer le câblage à travers l'élargissement dédié). Introduire le servo coudé dans le siège carré sur la plaque de base et glisser le klaxon dans le logement profilé du bras qui entraîne le mouvement vertical. Fixez la plaque de presse contre le servo à l'aide de deux écrous M3

4.2.1.16 Étape 16 : Servo d'entraînement avant/arrière

.

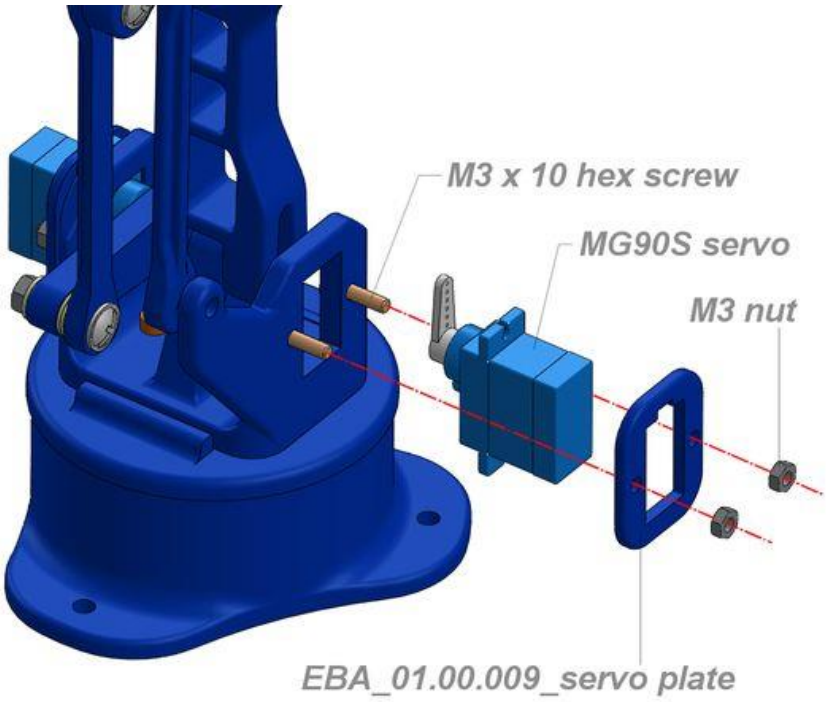


Figure 4:30 Étape 16 : Servo d'entraînement avant/arrière a

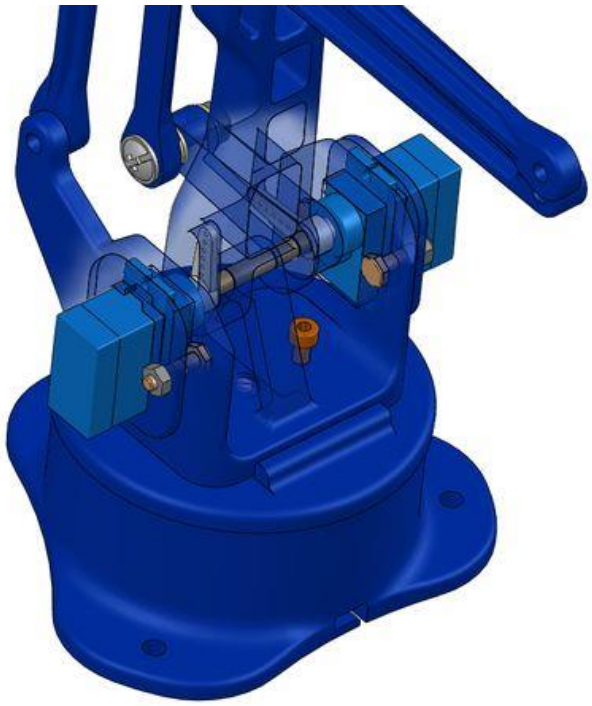


Figure 4:31 Étape 16 Servo d'entraînement avant/arrière b

La séquence pour le servo d'entraînement avant et arrière est similaire à la précédente. Dans ce cas, le palonnier doit être installé avec le servo au neutre aligné verticalement

4.2.1.17 Étape 17 : Dernier lien

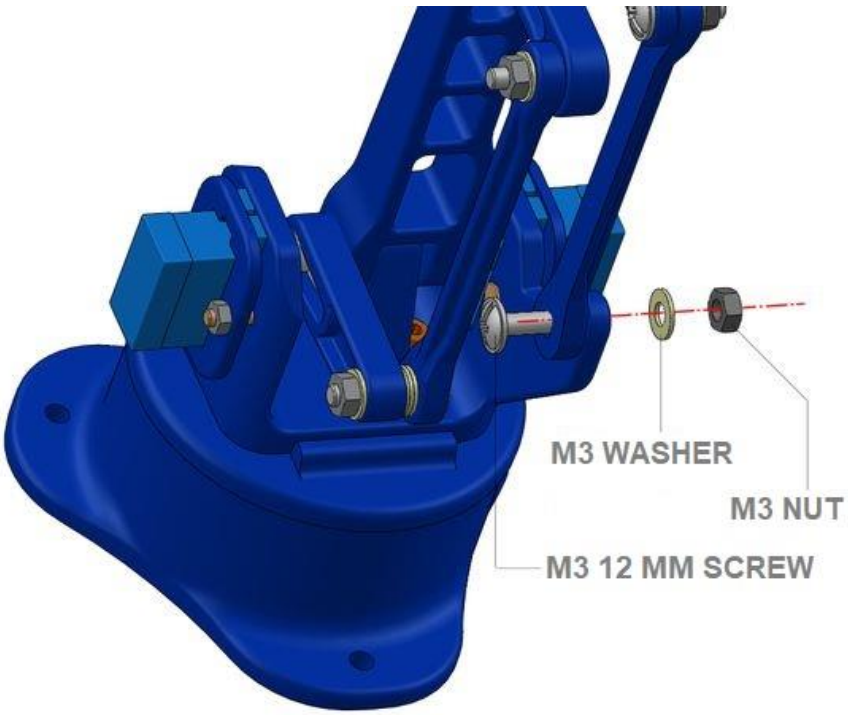


Figure 4:32 Étape 17 : Dernier lien

Attachez le dernier lien au bras fixe sur la face arrière de la base à l'aide d'une rondelle M3x12 et d'un écrou

4.2.1.18 Étape 18 : Fixation de la pince

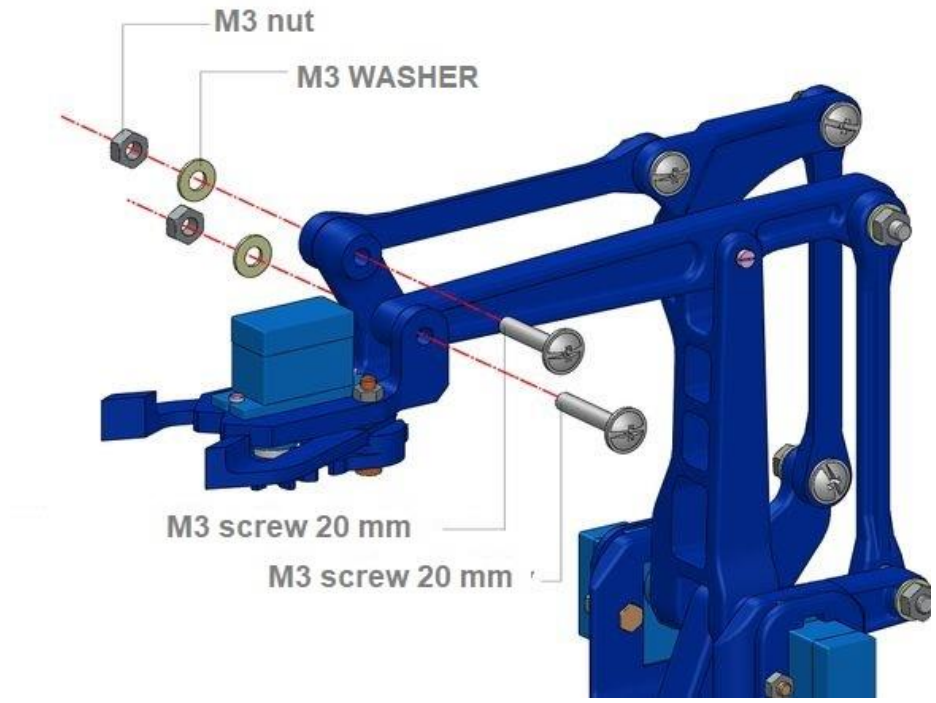


Figure 4:33 Étape 18 : Fixation de la pince

La dernière étape d'assemblage consiste à joindre la pince au bras horizontal comme indiqué sur l'image.

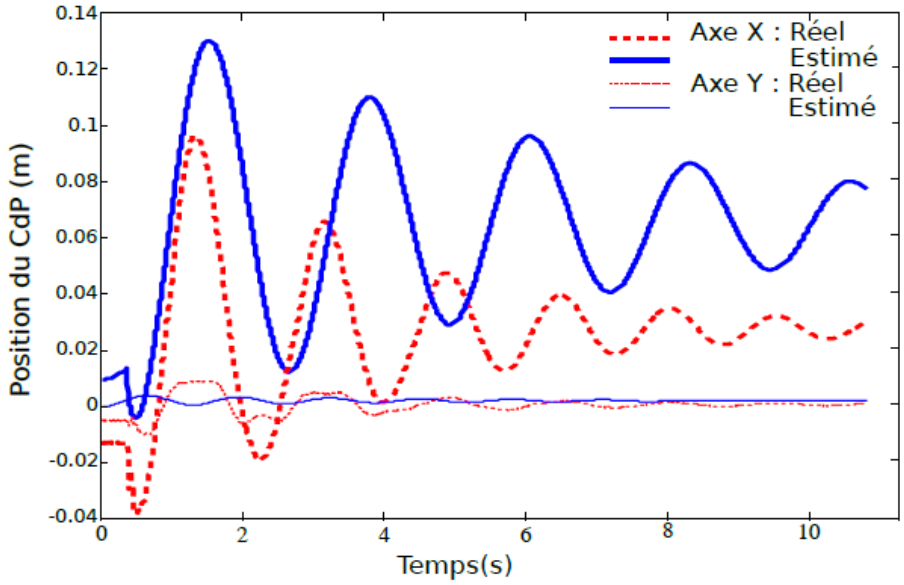


Figure 4:34 Sans correction des capteurs pour un échelon de 2 cm vers l'avant de la position de référence du centre de masse : en bleu le CdP reconstruit par notre estimation des efforts de contacts et en rouge pointillé le CdP reconstruit à partir des capteurs de

```
#include <Servo.h>

Servo myservo;  // create servo object to control a servo
Servo myservo2;
Servo myservo3;
Servo myservo4;
int potpin = A0;  // analog pin used to connect the potentiometer
int potpin2 = A1;
int potpin3 = A2;
int potpin4 = A3;
int val;    // variable to read the value from the analog pin
int val2;
int val3;
int val4;
void setup() {
  Serial.begin(9600);
  myservo.attach(2);  // attaches the servo on pin 9 to the servo object
  myservo2.attach(3);
  myservo3.attach(4);
  myservo4.attach(5);
}
```

Figure 4:35 Programme 4 1

```
void loop() {
  val = analogRead(potpin);            // reads the value of the potentiometer (value between 0 and 1023)
  Serial.println(val);
  val = map(val, 0, 1023, 750, 2400);     // scale it to use it with the servo (value between 0 and 180)
  myservo.writeMicroseconds(val);                  // sets the servo position according to the scaled value
                          // waits for the servo to get there
  val2 = analogRead(potpin2);            // reads the value of the potentiometer (value between 0 and 1023)
  val2 = map(val2, 0, 1023, 750, 2400);     // scale it to use it with the servo (value between 0 and 180)
  myservo2.writeMicroseconds(val2);
  val3 = analogRead(potpin3);            // reads the value of the potentiometer (value between 0 and 1023)
  val3 = map(val3, 0, 1023, 750, 2400);     // scale it to use it with the servo (value between 0 and 180)
  myservo3.writeMicroseconds(val3);
  val4 = analogRead(potpin4);            // reads the value of the potentiometer (value between 0 and 1023)
  val4 = map(val4, 0, 1023, 750, 2400);     // scale it to use it with the servo (value between 0 and 180)
  myservo4.writeMicroseconds(val4);
    delay(10);
}
```

Figure 4:36 Programme 4 2

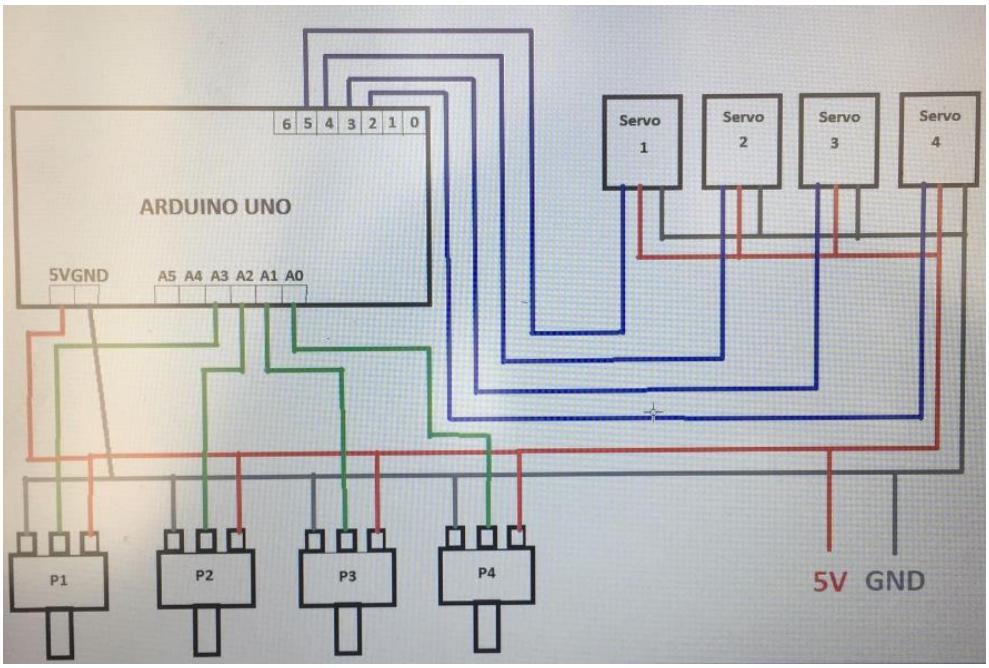
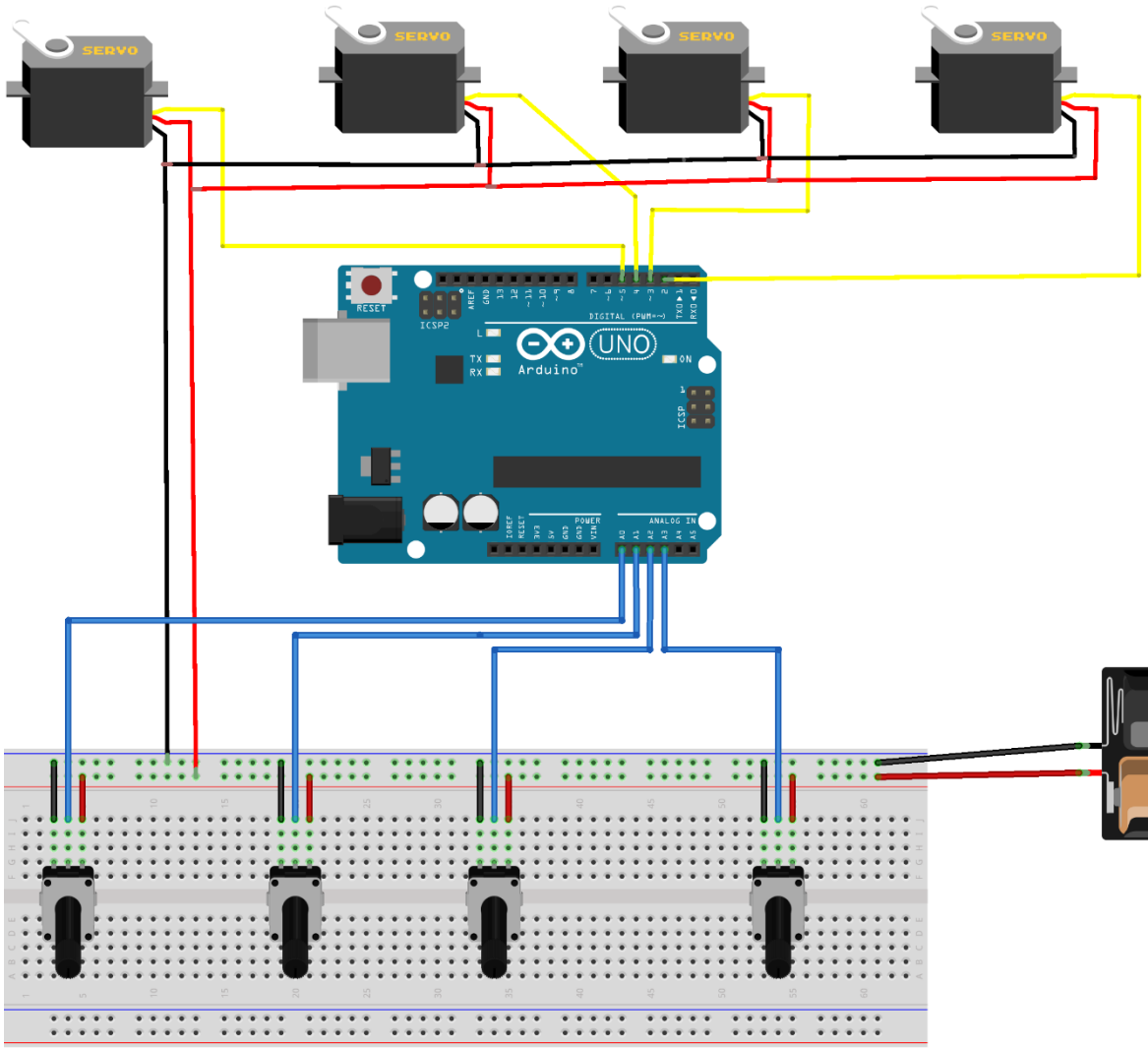


Figure 4:37 carte programme



4.3 Conclusion

Dans ce chapitre, nous avons pu décrire les différents éléments constitutifs du bras manipulateur, et injecter le programme dans la carte Arduino UNO, pour commander le bras manipulateur.

Conclusion générale

Conclusion générale

La robotique peut être définie comme l'ensemble des techniques et études tendant à concevoir des systèmes mécaniques, informatiques ou mixtes, capables de se substituer à l'homme dans ses fonctions motrices, sensorielles et intellectuelles.

Au terme de ce travail, nous avons pu concevoir et réaliser un bras manipulateur à quatre degrés de liberté qu'on a commandé avec une carte Arduino UNO.

Ce travail nous a permis de mettre en valeur nos connaissances théoriques et pratiques, que nous avons cumulées pendant toute la durée de nos études, il nous a permis aussi d'acquérir une expérience nouvelle dans la programmation des cartes Arduino.

L'étude et la réalisation de ce projet nous ont permis d'approfondir nos connaissances théoriques et pratiques acquises durant notre formation, notamment en robotique et en systèmes embarqués. Nous avons pu appréhender différents points importants comme le fait de travailler en équipe et de concevoir et mener à bien un projet.

Bien que ce modeste travail ait atteint ses objectifs, on estime qu'il pourrait être encore amélioré. En effet, les travaux décrits dans ce mémoire peuvent se poursuivre sur plusieurs voies. Tout d'abord, il serait intéressant d'envisager un environnement de travail avec obstacles, ce qui n'est pas le cas dans cette étude. Il serait également intéressant d'implémenter des commandes en boucle fermée, plus utiles en présence des perturbations. Cela constitue une bonne extension de ce projet. Pour la réalisation du bras, il serait avantageux d'optimiser la construction en utilisant un matériau plus solide. On pourrait utiliser des servomoteurs offrant un couple moteur plus important, dans le but de soulever des objets plus lourds.

Nous nous somme intéressé dans notre travail à l'aspect théorique de la robotique mais surtout à l'aspect pratique, ce qui a présenté la difficulté majeur de notre démarche, et cela parce que dans notre formation la théorie est prédominante. Et en s'intéressant à cet aspect nous avons pu mettre en relief les différentes notions théoriques que nous avons acquis. Il est évident que notre travail est loin d'être parfait, mais il peut constituer une base très intéressante pour les promotions à venir qui voudront travailler sur ce type de sujet.

Références

Références

[1] E. Renaudo, G. Benoît, R. Chatila et M. Khamassi , Design of a control architecture for habit learning in robots , In Biomimetic and Biohybrid Systems, Springer, p. 249– 260 (cf. p. 152), 2014.

[2] C. Martin, M. Alric, F. Chapelle, J.J. Lemaire, and G. Gogu, Trajectory planning for tumor resection: integrated design of a mini-invasive neurosurgical robot, In Proceedings of IDMME-Virtual Concept, Beijing, China, October 2008.

[3] A. Droniou, S. Ivaldi et O. Sigaud, Deep unsupervised network for multimodal perception: representation and classification, In Robotics and Autonomous Systems, 2014.

[4] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, Numerical recipes, The Art of Scientific Computing (3 rd edition), Cambridge University Press, 2007.

[5] B. Tondu, S. Ippolito, J. Guiochet, and A. Daidié, A seven-degrees-of-freedom robot-arm driven by pneumatic artificial muscles for humanoid robots, International Journal of Robotics Research, 24(4), 2005.

[6] A. Droniou, Apprentissage de représentations et robotique développementale : quelques apports de l'apprentissage profond pour la robotique autonome, Thèse de Doctorat en Informatique, École doctorale Informatique, Télécommunications et Électronique, Institut des Systèmes Intelligents et de Robotique, Université Pierre et Marie Curie - Paris VI, 2015.

[7] C. Pinto, E. Macho, V.Petuya, O. Altuzarra, A. Hernández, Logiciel de simulation pour la caractérisation cinématique de robots spatiaux, Department of Mechanical Engineering, University of the Basque Country, Alameda Urquijo s/n, 48013, Bilbao, charles.pinto@ehu.es, Spain.

[8] E. Macho, O. Altuzarra, E. Amezua, A. Hernandez, Obtaining configuration space and singularity maps for parallel manipulators, Mechanism and Machine Theory, Vol 44(11), pp 2110-2125, 2009.

[9] S Kostrzewski, J.M. Duff, C. Baur & M. Olszewski, Robotic system for cervical spine surgery, The International Journal of Medical Robotics and Computer Assisted Surgery, vol. 8, no. 2, pages 184–190, 2012.

[10]V. Françoise, A. Sahbani, A. Roby-Brami, G. Morel, Assistance to bone milling : tool mounted visual display improves the efficiency of robotic guidance, In IEEE Engineering in Medicine and Biology Society (EMBC'13), 2013.

[11] K. Harada, K. Tsubouchi, M. G. Fujie, and T. Chiba, Micro manipulators for intrauterine fetal surgery in an open mri, In IEEE International Conference on Robotics and Automation, Barcelona, Spain, April 2005.

[12] P. Martinet, B. Thuilot, F. Hao, M. Berducat, C. Cariou, C. Debain, R. Lenain, C. Tessier, A. Godin, F. B. Amar, F. Plumet, S. Lacroix, T. Peynot, C. Grand, et G. Besseron, R2M : Rover Multi-Modes pour une haute mobilité sur terrain accidenté, Dans Journées Bilan ROBEA, 2006.

[13] R. Oshima, T. Takayama, T. Omata, K. Kojima, K. Takase, and N. Tanaka, Assemblable three-fingered nine-degree of freedom hand for laparoscopic surgery, In IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, USA, October 2009.

[14] S. Kajita, H. Hirukawa, K. Harada, K. Yokoi, Introduction à la commande des robots humanoïdes : de la modélisation à la génération du mouvement, © SpringerVerlag, France 2009.

[15]W. Khalil, E. Dombre, Bases de la modélisation et de commande des robotsmanipulateurs de type série, IRCCyN Ecole Centrale de Nantes & LIRMM Université Montpellier, 08 mai 2012.

[16] Christian Tavernier, *Arduino Maîtriser sa programmation et ses cartes d'interface (shields)*, Paris, 2011, DUNOD

[17] RAO Michael « Introduction à l'Arduino », Cours, Université de Lyon,2016.

[18] DATASHEET « Arduino UNO », Components 101, 2018.

Annexes

Annexes

Conseil d'administration

La carte de microcontrôleur Due est basée sur le processeur Microchip SAM3X8E Arm® Cortex®-M3

L'Arduino Due est une carte de microcontrôleur basée sur le processeur Microchip SAM3X8E Arm Cortex-M3. Il s'agit de la première carte Arduino basée sur un microcontrôleur central ARM 32 bits. Le Due dispose de 54 broches d'entrée/sortie numériques (dont 12 peuvent être utilisées comme sorties PWM), 12 entrées analogiques, quatre UART (ports série matériels), une horloge 84 MHz, une connexion USB OTG, deux DAC (numérique vers analogique), deux TWI, une prise d'alimentation, un en-tête SPI, un en-tête JTAG, un bouton de réinitialisation et un bouton d'effacement.

La carte contient tout le nécessaire pour prendre en charge le microcontrôleur ; connectez-le simplement à un ordinateur avec un câble USB ou alimentez-le avec un adaptateur AC-DC ou une batterie pour commencer. Le Due est compatible avec tous les shields Arduino qui fonctionnent à 3,3 V et sont compatibles avec le brochage Arduino 1.0. Le Due possède également un noyau Arm 32 bits qui peut surpasser les cartes de microcontrôleur 8 bits typiques.

Le SAM3X dispose de 512 Ko (deux blocs de 256 Ko) de mémoire flash pour stocker le code. Le chargeur de démarrage est pré-gravé en usine à partir de Microchip et est stocké dans une mémoire ROM dédiée. La SRAM disponible est de 96 Ko dans deux banques contiguës de 64 Ko et 32 Ko. Toute la mémoire disponible (Flash, RAM et ROM) est accessible directement sous la forme d'un espace d'adressage plat. Il est possible d'effacer la mémoire flash du SAM3X avec le bouton d'effacement intégré. Cela supprimera l'esquisse actuellement chargée du MCU. Pour effacer, maintenez enfoncé le bouton d'effacement pendant quelques secondes pendant que la carte est sous tension.

Caractéristiques

• 54 broches d'E/S numériques

• 12 broches d'E/S numériques capables d'une sortie PWM 8 bits

• 12 broches d'entrée analogique

• Deux broches de sortie analogique (DAC 12 bits)

Applications

• Un cœur 32 bits, qui permet des opérations sur des données de 4 octets de large au sein d'une seule horloge CPU

• Fréquence du processeur à 84 MHz

• Port USB natif directement connecté au SAM3X

• 512 Ko de mémoire Flash disponible pour l'application de l'utilisateur

• 96 Ko de SRAM (deux banques : 64 Ko et 32 Ko)

Vitesse d'horloge de 84 MHz

• 96 Ko de SRAM

• 512 Ko de mémoire Flash pour le code

Un contrôleur DMA qui peut soulager le CPU des tâches gourmandes en mémoire