CONTRÔLE D'UN BRAS ROBOTIQUE À BASE DE MOTION CAPTURE

RÉALISÉ PAR :

BEJI MARIAM

KAANICHE KHALIL

BAZZAZI MOHAMED TAHER

BEN SLIMENE IMED EDDINE

II A 3

PROFESSEUR ENCADRANT: MED SAHBI BELLAMINE

Table des matières

Table des figures

Introduction

Dans le cadre de notre projet personnel professionnel en 3ème année informatique industrielle et automatique à l'INSAT, nous allons étudier et développer un système de contrôle d'un bras robotique à base de motion capture.

Cette méthode de contrôle spécifique du bras peut être utilisée dans le domaine des loisirs vu l'aspect ludique qu'offre cette technique ainsi que dans d'autres domaines un peu plus délicats comme la chimie industrielle où certains opérateurs sont couramment exposés à des matériaux dangereux. Ainsi la motion capture offre une immersion vue l'aspect naturel de ce type de contrôle.

Le travail consistera à traiter deux grands axes de degrés d'importances différents : Le bras robotique en question ainsi que le principe de motion capture. Nous procéderons en premier lieu à une recherche bibliographique sur les bras robotisés. Cela englobera une brève présentation, un aperçu global sur leurs différents domaines d'application ainsi que les différents types de bras manipulateurs. Ensuite nous procéderons à une présentation du principe de la motion capture, ses avantages et inconvénients, ses domaines d'applications ainsi que les différentes méthodes utilisées.

Cette partie purement théorique convergera par la suite vers une approche de plus en plus pratique. Effectivement, on détaillera en premier lieu de façon globale l'approche utilisée pour la réalisation du projet. Ensuite, on présentera la conception mécanique du prototype du bras robotisé réalisé pour ce projet. De plus, Par la suite, on procédera à la présentation de la partie traitement d'image.

Ceci étant fait, on présentera un prototype (ou communément appelé proof of concept) qui mettra en valeur les différentes fonctionnalités détaillées précédemment.

Enfin, le rapport se clôturera par un récapitulatif résumant le travail effectué, expliquant ses limites le tout dans le but à ouvrir des perspectives.

I- Recherche Bibliographique sur les bras robotiques

1- Présentation

Par définition, un bras manipulateur est une partie d'un robot généralement programmable, semblable à un bras humain, permettant selon son degré de complexité de réaliser des tâches plus ou moins délicates (cela va de transporter des objets de façon grossière, à peindre un tableau à titre d'exemple). Il est soit autonome comme c'est le cas de bras manipulateurs utilisés dans certaines chaines industrielles, ou manuel. Tel est le cas notamment dans certaines maquettes éducatives. Un bras manipulateur peut être fixe ou mobile, cela dépend de l'utilisation de ce dernier.

2- Utilité

Les bras robotiques depuis leurs inventions ont étés exploités dans plusieurs domaines (le plus courant reste le domaine industriel) et au fur et à mesure de la réduction de leurs coûts, augmentation de leur précision ainsi que leur fiabilité, leur utilisation devient de plus en plus courante.

A- Domaine industriel

L'organisation internationale de la normalisation (International Organisation for Standardization ISO en anglais) définie la robotique industrielle comme un système commandé automatiquement, multi-applicatif, reprogrammable, polyvalent, manipulateur et reprogrammable sur 3 axes ou plus.

Les applications les plus courantes restent des robots d'assemblage, de soudure ou de peinture. Cela permet en premier lieu, une rapidité d'exécution, une haute précision, ainsi qu'une répétition dans la durée.

Figure 1 : 1er Robot industriel : Unimate

On trouve ci-dessus la photo du premier robot industriel nommé Unimate utilisé dans une ligne d'assemblage de General Motors

B- Domaine Médical

Par définition, un robot médical est un système robotique utilisé lors d'une application thérapeutique. Par respect aux contraintes de sécurités assez strictes, ce genre de bras robotisés est généralement commandé manuellement et son autonomie reste bridée.

Dans ce domaine, les robots sont classés en deux grandes catégories : les robots interventionnels utilisés notamment en chirurgie ainsi que ceux utilisés pour la rééducation des patients. Notons qu'ils sont utilisés dans une multitude de domaines médicaux comme la chirurgie générale et cardiothoracique, la gynécologie, la neurochirurgie, la radiologie etc…

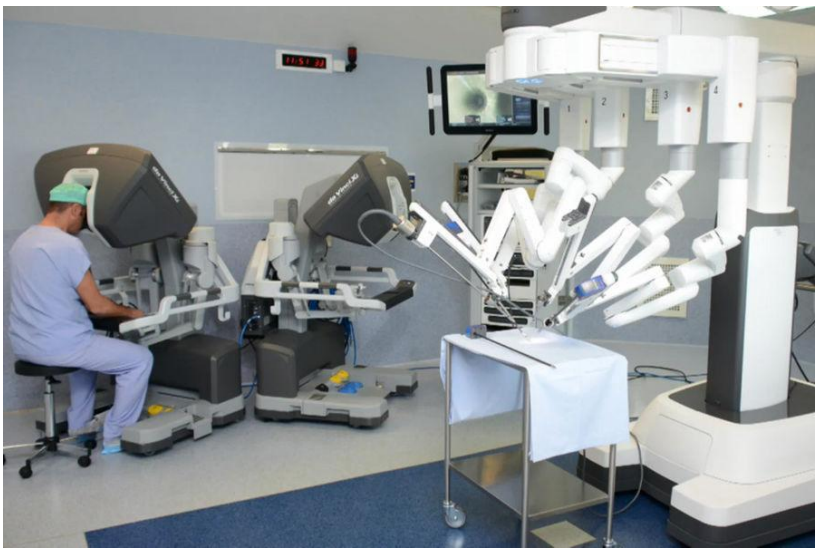


Figure 2 : Robot Medical : Da vinci

Ci-dessus l'image du robot Da Vinci (le plus utilisé de nos jours) il est constitué d'un ensemble de bras manipulateurs.

C- Culture et loisir

Les bras robotiques sont utilisés à des fins de divertissement pur et dur dans certains cas. On peut citer à titre d'exemple, les kits de construction de bras destinés à un public large, ou dans un autre genre, les « Robocoaster » qui ne sont en réalité qu'un ensemble de sièges montés sur un bras manipulateur.

Figure 3 : Robocoaster

3- Types de bras manipulateur

A- Les robots SCARA (Selective Compliance Articulated)

Ce type de robots est un robot à 3 axes série (RRT) et à 3 degrés de liberté. Ils sont généralement précis et rapides.

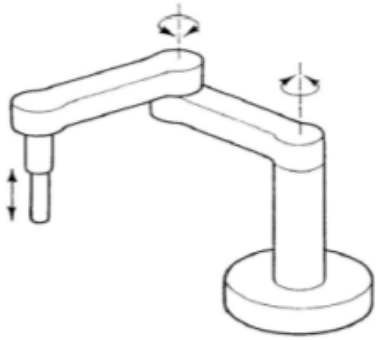


Figure 4 : Schéma simplifié d'un robot SCARA

B- Les robots cylindriques

C'est un robot à 3 axes, série (RTT) et 3 degrés de liberté, son espace de travail est cylindrique.

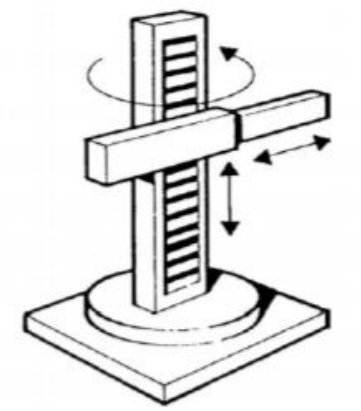


Figure 5 : Schéma simplifié d'un robot cylindrique

C- Les robots sphériques (polaires)

Ce sont des robots à 3 axes, série RRT, et 3 degrés de liberté, leurs espaces de travail est sphérique. Ils sont caractérisés par leur grande charge utile.

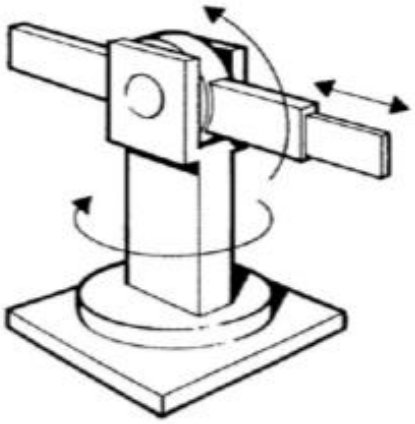


Figure 6 : Schéma simplifié d'un robot polaire

D- Les robots Cartésiens

Ce sont des robots à 3 axes perpendiculaires 2 à 2, série, TTT, et 3 degrés de liberté et caractérisés par leur très bonne précision, mais ils sont lents.

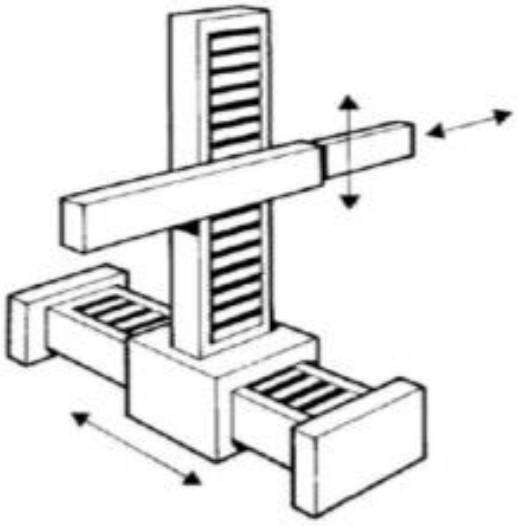


Figure 7 : Schéma simplifié d'un robot cartésien

E- Les robots parallèles

Ce sont des robots à Plusieurs chaînes cinématiques en parallèles et un espace de travail réduit. Ils sont caractérisés par la rapidité et la précision (grande rigidité de la structure)

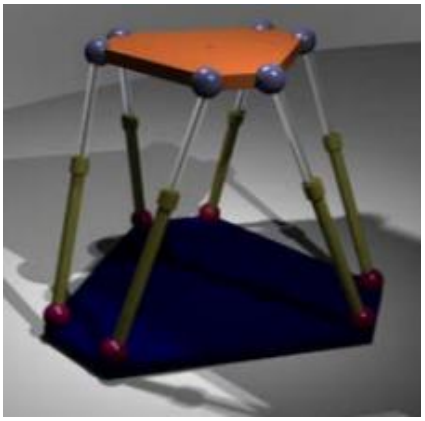


Figure 8 : image représentative d'un robot parallèle

F- Les robots anthropomorphes

Ce sont des robots à 6 axes, série, 6 rotations, et 6 degrés de liberté, ils reproduisent la structure d'un bras humain.

Figure 9 : Schéma simplifié d'un robot antropomorphe

II- Présentation de la motion capture :

1- Introduction

Capture de mouvement ou motion capture en anglais, abrégé en « mo-cap » est la procédure d'enregistrer le mouvement d'objets ou de personnes. C'est une technique utilisée dans le domaine militaire, sportif, médical ou encore dans un but de divertissement pur.

La motion capture pour résumer est l'utilisation d'une technologie de détection (sensing technology) pour détecter une cible et de suivre l'évolution de son mouvement au cours du temps.

En effet, elle peut être divisée en trois grandes parties :

- Détection du mouvement (sensing)
- Traitement des données du/des capteur(s) (Processing)
- Sauvegarde des données acquises et traitées (Storing)

La première et deuxième partie sont regroupés en tant que Motion Tracking. Les données en résultants, au lieu d'être sauvegardées, peuvent être « directement » utilisées dans des applications en temps réel nécessitant une certaine interaction avec l'utilisateur.

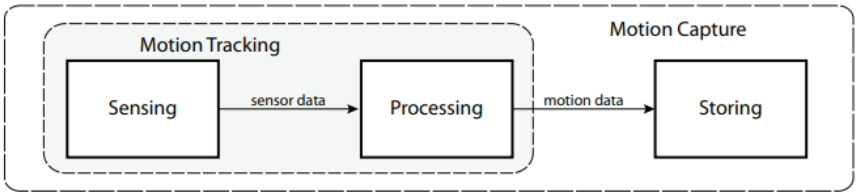


Figure 10 : Schéma explicatif de la motion capture

La partie Détection de mouvement (sensing) de la motion capture implique la mesure du mouvement. Ceci peut être réalisé par une variété de capteurs allant du plus simple, comme un potentiomètre à titre d'exemple, au plus complexe (un ensemble de caméra). Cependant, les données récoltées en elles même ne sont pas directement exploitables et nécessitent un certain traitement (allant du simple filtrage pour enlever les bruits résultants lors de l'acquisition des données à l'interprétation des données).

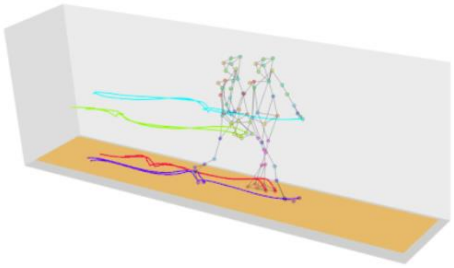


Figure 11 : figure représentative des donnés de la motion capture

2- Avantages et inconvénients

A- Avantages de la motion capture

- Une faible latence (quasi égale à zéro)
- Une méthode de travail homogène ne dépendant pas de la complexité du mouvement effectué
- Une façon fiable, facile et relativement précise de représenter la variation de position et de rotation d'un objet au cours du temps
- L'acquisition d'un grand nombre de données exploitables dans un lapse de temps relativement court
- Peut ne pas être une solution couteuse vu le développement des technologies utilisées et leurs démocratisations dans le marché

B- Inconvénients de la motion capture

- Requiert une combinaison Software-Hardware pas évidente à fournir et à mettre en place
- Des coûts relativement élevés pour avoir une certaine précision, obtenir une certaine performance et pouvoir suivre un nombre élevé de points
- Des restrictions liées aux techniques utilisées (par exemple un niveau de luminosité et d'éclairage spécifique pour obtenir de meilleurs résultats à la caméra si on utilise cette approche-là lors de la motion capture
- Des limitations liées à l'unité de traitement des données

3- Domaines d'applications

La « motion capture » est de plus en plus utilisée dans plusieurs domaines d'activités vu son rôle lors de la création des CGI (Computer-generated imagery) et sa fiabilité.

A- Le cinéma :

En 1999, George Lucas, le réalisateur de ' Star Wars, épisode I : La Menace fantôme', a utilisé pour la première fois la capture de mouvement dans le monde du cinéma. L'utilisation était partielle et limitée à la création de quelques mouvements des personnages. Mais en 2004, Robert Zemeckis a présenté 'Le Pôle Express ' qui a été le premier film totalement basé sur la motion capture.

De nos jours, la motion capture est indispensable dans la filmographie et on trouve son utilisation dans tous les films couronnés de succès (Avatar, MCU, Le Hobbit, Jurassic World …).

Figure 12 : La motion capture et le cinéma

B- Les jeux vidéos

Contrairement au cinéma, L'utilisation de capture de mouvement en jeux vidéo a été adoptée en 1988 sous un format 2D pour animer le caractère principal du jeu 'Vixen'. En 1994, la technologie a été utilisé pour des modèles 3D pour la première fois dans le jeu d'arcade 'Virtua Fighter 2'. La motion capture est de plus en plus développés dans les jeux d'aujourd'hui qui ont comme but d'être plus réaliste (les jeux de sports par exemple (FIFA, NBA2K...))

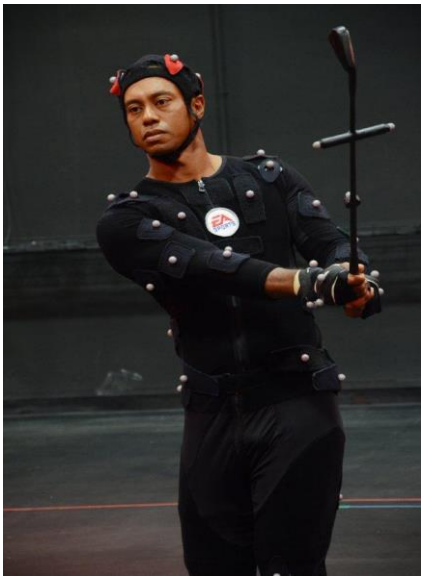


Figure 13 : Recréation des mouvements de joueurs à travers la motion capture

C- Le monde médical

La capture de mouvement devient plus adoptée dans le monde médical pour faciliter les analyses des mouvements des os, et de l'état des articulations. Cette technique est efficace pour surveiller et repérer, par exemple, les endroits de tension, de frottements ou autres complications. Ainsi, elle serait utile pour éviter les problèmes dans l'avenir.

Figure 14 : Utilisation de la motion capture dans la rééducation

4- Méthodes utilisées :

Il existe différents types de technologies.

Technologies non optiques (Non-visual tracking) :

Dans cette section, nous pourrions mentionner :

- Les systèmes magnétiques : qui fournissent la position et l'orientation des capteurs fixés sur la structure en utilisant l'orientation d'un champ magnétique de basse fréquence créé par un émetteur. Les positions des capteurs dans le champ magnétique sont captées et un logiciel va ensuite retranscrire la position et les mouvements des capteurs dans un espace en 3D. Si cette méthode est utilisée sur plus qu'un seul corps à la fois, les capteurs vont interférer les uns avec les autres. Les données enregistrées seront faussées et donc inutilisables. reportées

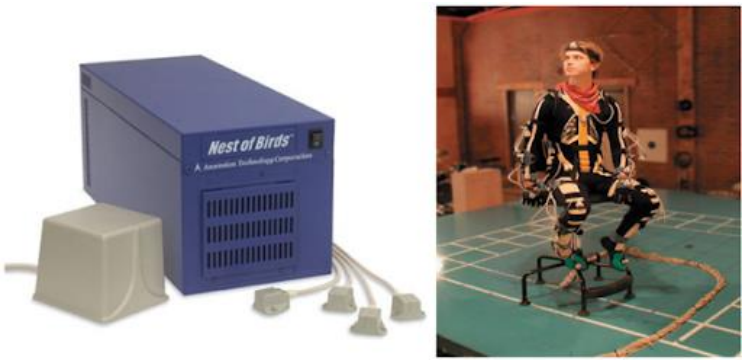


Figure 15 : Motion capture à base de systèmes magnétiques

- Les capteurs inertiels (IMU - Inertial Measurement Unit) / gyroscopiques ont récemment été introduits dans ce type d'application. Ce type de capteurs fournit en temps réel l'orientation 3D (Roll, Yaw, Pitch) de l'endroit (par exemple la tête) où ils se trouvent et transmet les informations directement à l'ordinateur. Ils sont beaucoup plus faciles à placer que les optiques, et plus pratiques avec une précision maximale.

Figure 16 : Motion Capture à base de capteurs intertiels

- La capture mécanique : un exosquelette est placé sur le sujet et indique son mouvement à l'aide de potentiomètres ou d'autres mécanismes. Ces capteurs localisés dans les articulations sont reliés par des fils à un microcontrôleur. Chaque capteur a sa position par rapport aux autres, ce qui

permet au microcontrôleur de savoir les mouvements de l'ensemble. Cette méthode est très précise et le calcul des données est très rapide. Mais le rayon d'action est réduit à cause de sa liaison par fils à l'ordinateur.

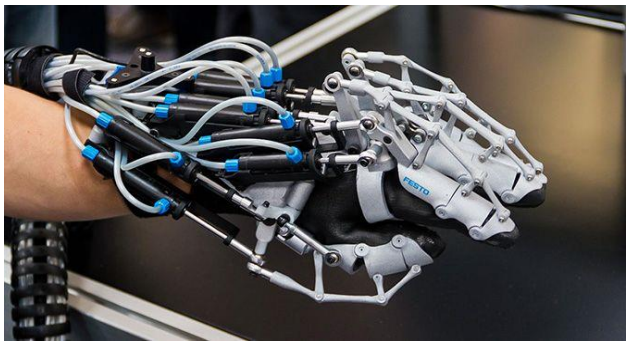


Figure 17 : Motion Capture à base de capteurs mécaniques

- Les systèmes de suivi acoustique (ultrasons) calculent la position sur la longueur d'onde d'un signal acoustique et la vitesse du son. Les systèmes basés sur le temps de vol mesurent le temps entre l'envoi d'un signal d'un émetteur et de sa captation par un récepteur, et des systèmes basés sur la cohérence de phase mesurer la différence de phase entre le signal à l'extrémité de l'émetteur et l'extrémité du récepteur. Un seul émetteur associé à un seul récepteur donne la position du récepteur dans une sphère autour de l'émetteur. En ajoutant plus d'émetteurs la position 3D du récepteur peut être trouvée. La vitesse du son dans l'air (~343 m/s à 20 C) peut varier avec la pression atmosphérique et la température ce qui affecte la précision de la capture.

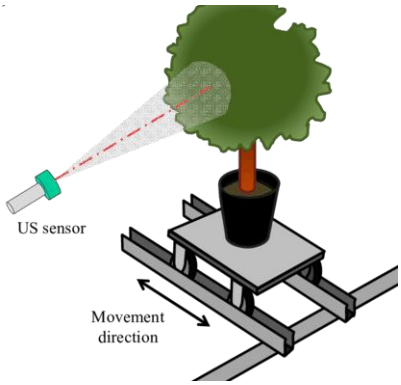


Figure 18 : Motion Capture à base de systèmes de suivi acoustique

Technologie optique (suivi visuel) :

Dans ce type de technologie, des repères actifs ou passifs sont fixés sur le sujet à suivre et la position des repères est mémorisée. Le détecteur (caméra) reflète uniquement la position (du faisceau) de chaque marque individuelle et stocke les coordonnées x et y ; les images ne sont pas stockées. Ces résultats sont traités

mathématiquement a posteriori et ainsi tous les paramètres cinématiques du mouvement sont calculés.

Les capteurs sont placés sur les articulations et les principales masses du corps, telles que la tête, les bras et les jambes. Pour un enregistrement correct des mouvements du corps (bras inclus), au moins 30 capteurs sont nécessaires.

Dans ces méthodes de capture visuelles, un montage très spécifique des différentes caméras est requis et plus nous plaçons de caméras, meilleures sont les données de position que nous pouvons obtenir.

Le problème principal de cette méthode, c'est sa sensibilité aux obstacles entre les capteurs et les caméras.

- Optique-passif : Les marquages passifs sont éclairés par des sources de rayonnement (infrarouge, voire ultrasonore) et les rayons sont réfléchis vers un détecteur : des capteurs rétro réfléchissants. C'est la méthode la plus utilisée dans l'industrie.

  Un objet avec des marqueurs connectés à des positions connues est utilisé pour calibrer les caméras et obtenir les positions. Typiquement, un système optique passif se composera d'environ 2 à 48 caméras. Il existe des systèmes de plus de 300 caméras pour tenter de réduire les échanges de marqueurs. Des caméras supplémentaires sont nécessaires pour une couverture complète autour du sujet de prise de vue et de plusieurs sujets.

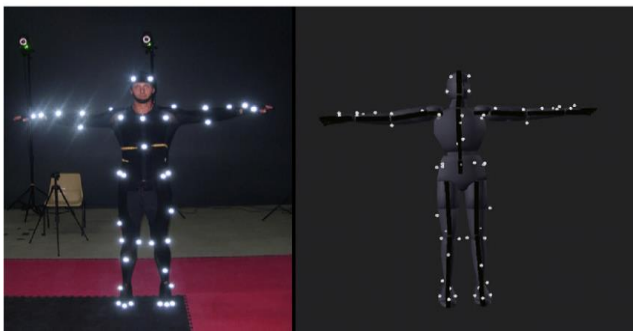


Figure 19 : Modélisation 3D via Motion Capture

- Optique-actif : Cette technique utilise des capteurs LED connectés par des câbles à la combinaison de capture de mouvement. Les systèmes optiques actifs déduisent les positions en éclairant très rapidement une LED à la fois ou plusieurs LED avec un logiciel pour les identifier par leurs positions relatives, pareil aux méthodes utilisées dans la navigation astronomique. Au lieu de refléter la lumière générée de l'extérieur, les marqueurs sont alimentés en émettant leur propre lumière.

Il existe également des possibilités de trouver la position à l'aide de marqueurs LED de couleur. Dans ces systèmes, chaque couleur est attribuée à un point précis du corps.

Figure 20 : Motion capture en utilisant une technique optique-actif

- Méthode vidéo ou sans capteurs : Cette technique ne repose que sur un logiciel pour suivre les mouvements du corps étudié. Des algorithmes très avancés sont utilisés pour réaliser la capture, en particulier lors de la capture de mouvement en temps réel.

La technologie sans capteurs utilise le concept de 'depthmap', une surface à niveaux reconstruite suivant divers procédés optiques, qui n'est pas à proprement parler un modèle 3D classique (fait de sommets, d'arêtes et de polygones) mais de voxels, objets hybrides entre le pixel et le vecteur, sorte de nuage de cubes de couleur positionnés dans l'espace. Appliquée au corps, on utilise ce volume approximatif pour en extraire une configuration du squelette virtuel.

L'absence de capteurs en fait la méthode la plus imprécise. Il peut s'agir d'une méthode peu coûteuse, puisqu'elle est facilement enregistrable avec les caméras Kinect, celles utilisées par Microsoft dans les jeux vidéo, à la fois sur XBOX et Windows. Ceux-ci sont relativement peu coûteux, et ils rendent la méthode plus accessible à tous.

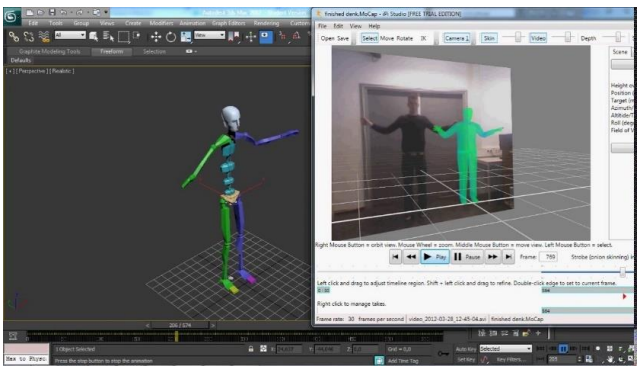


Figure 21 : Motion Capture à base de vidéo

III- Etude pratique et implémentation

Nous procéderons à l'étude du bras robotique, une explication de la logique utilisée pour la motion capture ainsi qu'à l'explication de la communication utilisée pour la communication Ordinateur-Microcontrôleur. Ceci se clôturera par une mise en place de notre solution étudiée.

1- Conception mécanique du bras robotique

A- Schéma cinématique

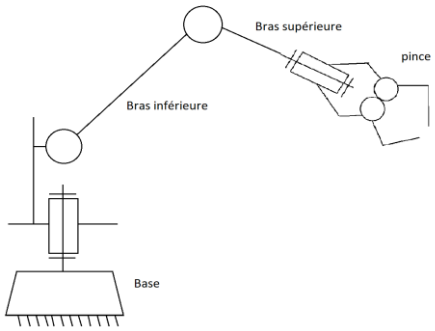


Figure 22 : schéma cinématique

B- Dimensionnement, calcul et solutions mécaniques

Calcul du volume de travail :

Le calcul du volume de travail d'un bras robotique est obtenue par étude dans un repère sphérique puisqu'on peut le modéliser par une sphère.

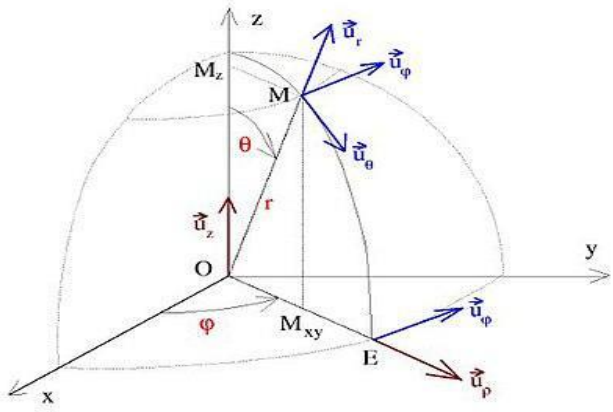


Figure 23 : repère sphérique

Le calcule de volume est obtenu par l'intégration des coordonnées du système successivement :

r de 0 à R

θ de 0 à π

φ de 0 à 2π

Dans notre cas, on trouve deux limites d'intégration :

Pour le coordonné θ, la zone morte du bras, dû à la présence d'une base, limette l'intégration à $\frac{3}{4}\pi$

Pour le coordonnéφ, la rotation du servo-moteur placé dans la base est limitée (pour notre cas l'angle limite est 240°). Ça implique une intégration jusqu'à $\frac{4}{3}\pi$

R présente le rayon limite que notre bras peut atteindre ; celui est égal à la distance entre la première articulation et la pince

$$R = l2 + l3$$

$$R = 17 + 23 = 40cm$$

Tous ces paramètres nous donnent le volume présenté ci-dessous:

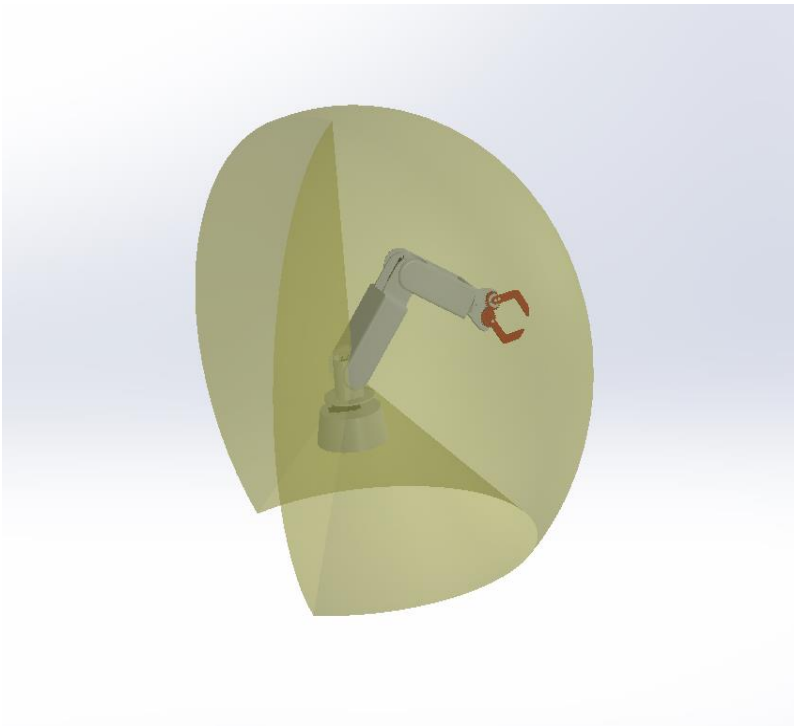


Figure 24 : volume de tarvail

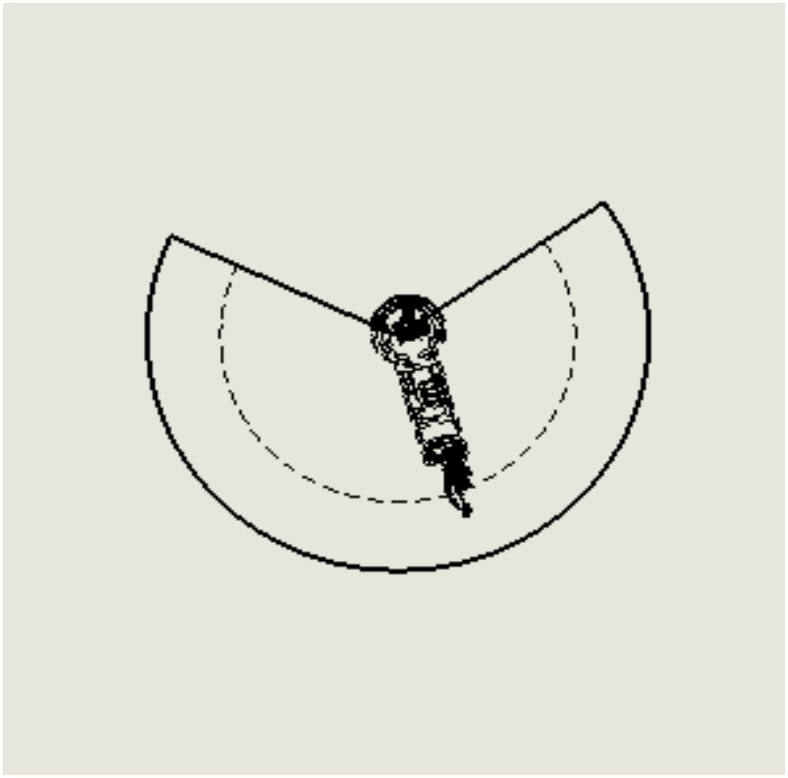


Figure 25 : vue de dessous (limitation de φ)

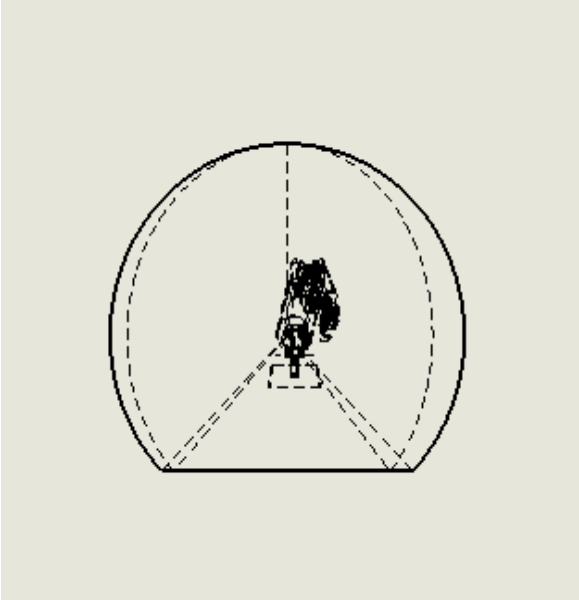


Figure 26 : vue de face (limitation de ϑ)

Le volume est exprimé alors par :

$$V = \int_0^R \int_0^{\frac{3}{4}\pi} \int_0^{\frac{4}{3}\pi} r^2 \sin\theta \, dr \, d\theta \, d\varphi$$

$$V = \int_0^R r^2 dr \int_0^{\frac{3}{4}\pi} \sin\theta \, d\theta \int_0^{\frac{4}{3}\pi} d\varphi$$

$$V = \left[r^3/3\right]_0^{40} \times \left[-\cos\theta\right]_0^{\frac{3}{4}\pi} \times \left[\varphi\right]_0^{\frac{4}{3}\pi}$$

$$V = \frac{40^3}{3} \times \left(\frac{\sqrt{2}}{2} + 1\right) \times \frac{4}{3}\pi$$

$$V = 151248.52\ cm^3 = 0.15\ m^3$$

Etude de l'engrenage au niveau de la pince :

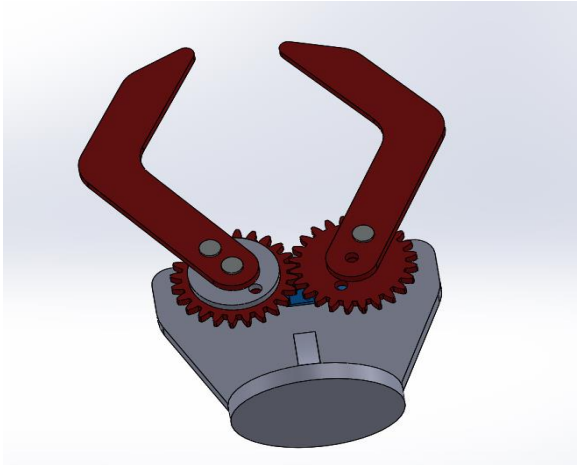


Figure 27 : vue 3D de la pince

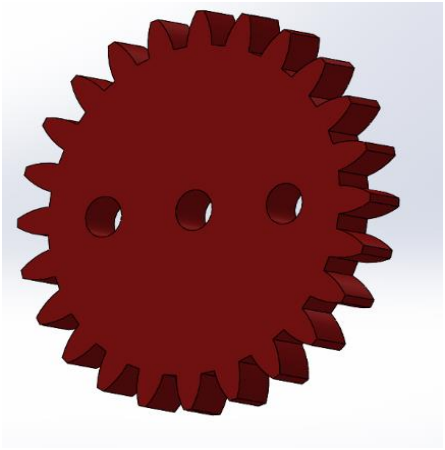


Figure 28 : Roue Dentée

Pour cette pince on utilise un engrenage pour la transmission de mouvement avec inversion du sens de rotation.

On a donc la roue motrice entraînée par le servo moteur entraîne à son tour la deuxième roue de l'engrenage qui lui est identique.

Le rapport de transmission r=1 ; puisque le but de l'utilisation de l'engrenage est de faire tourner les deux mâchoires de la pince à la même vitesse mais dans des sens opposés

On souhaite avoir un entraxe entre les deux roues dentées (a) égal à 27 mm

$a = \frac{d1+d2}{2}$ Avec d1=d2

⇒ a=d primitif= 27 mm

Le nombre de dents de ces roues est Z=24,

D'où m=1,115

On peut alors, calculer les diamètres de tête et de pied :

–Diamètre de tête : da = d + 2m = 29.23 mm ⇒ Ra= 14.615mm

–Diamètre de pied : df = d – 2,5m = 24.2125 mm ⇒ Rf=12.1mm

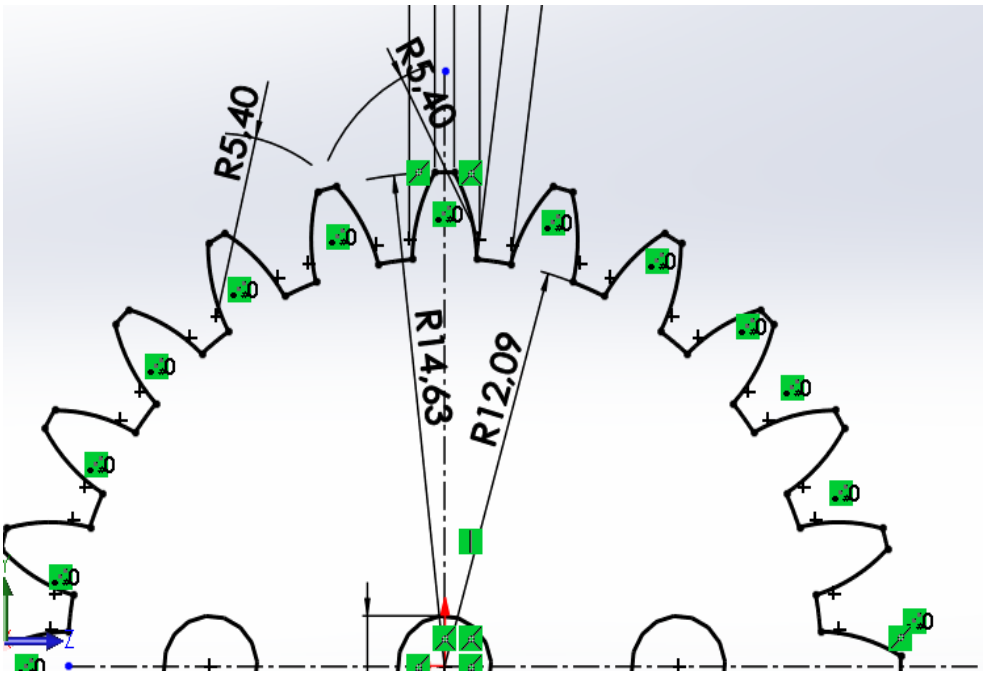


Figure 29 : Représentation de l'engrenage étudié sur SolidWorks

Conception

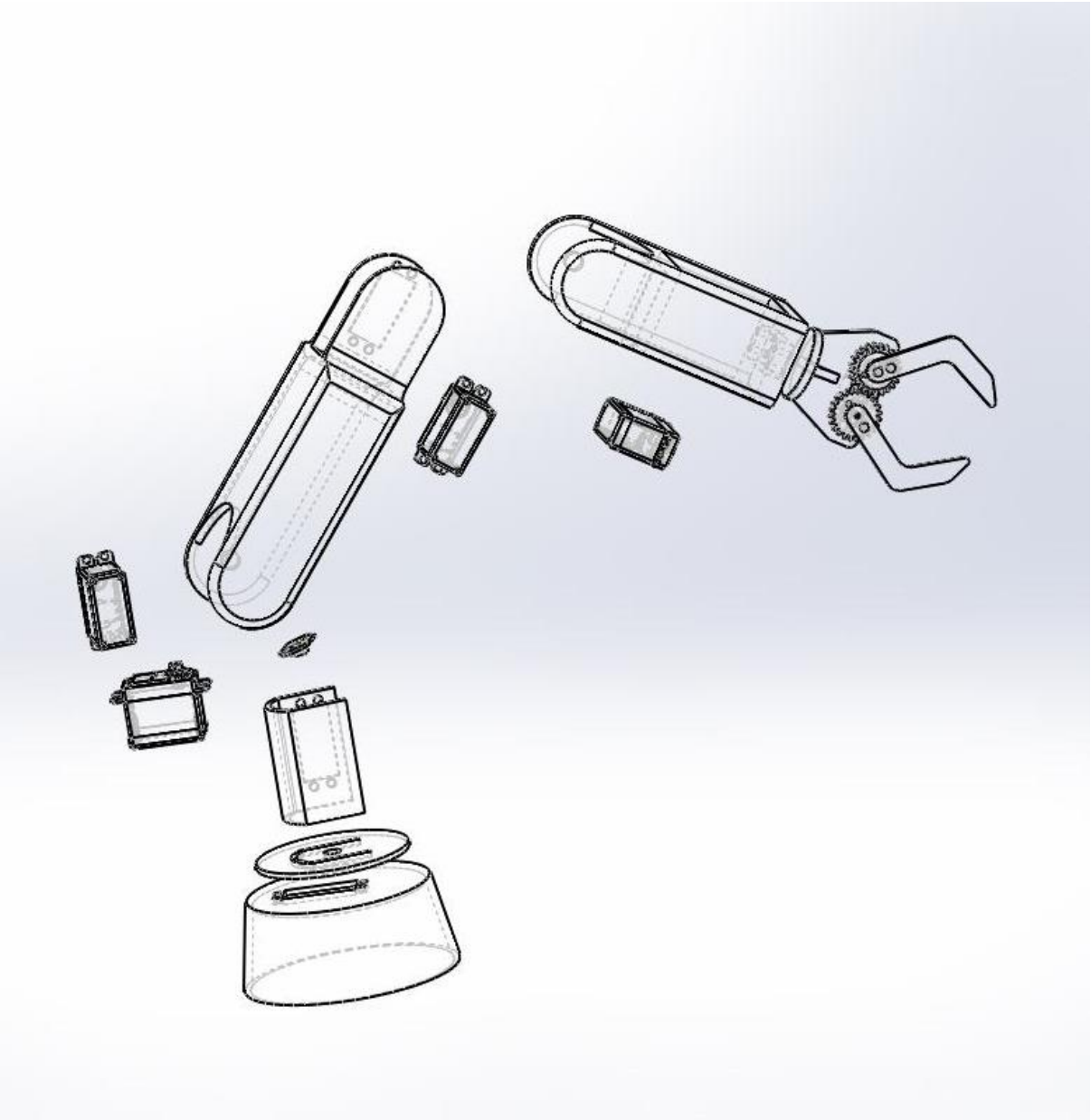
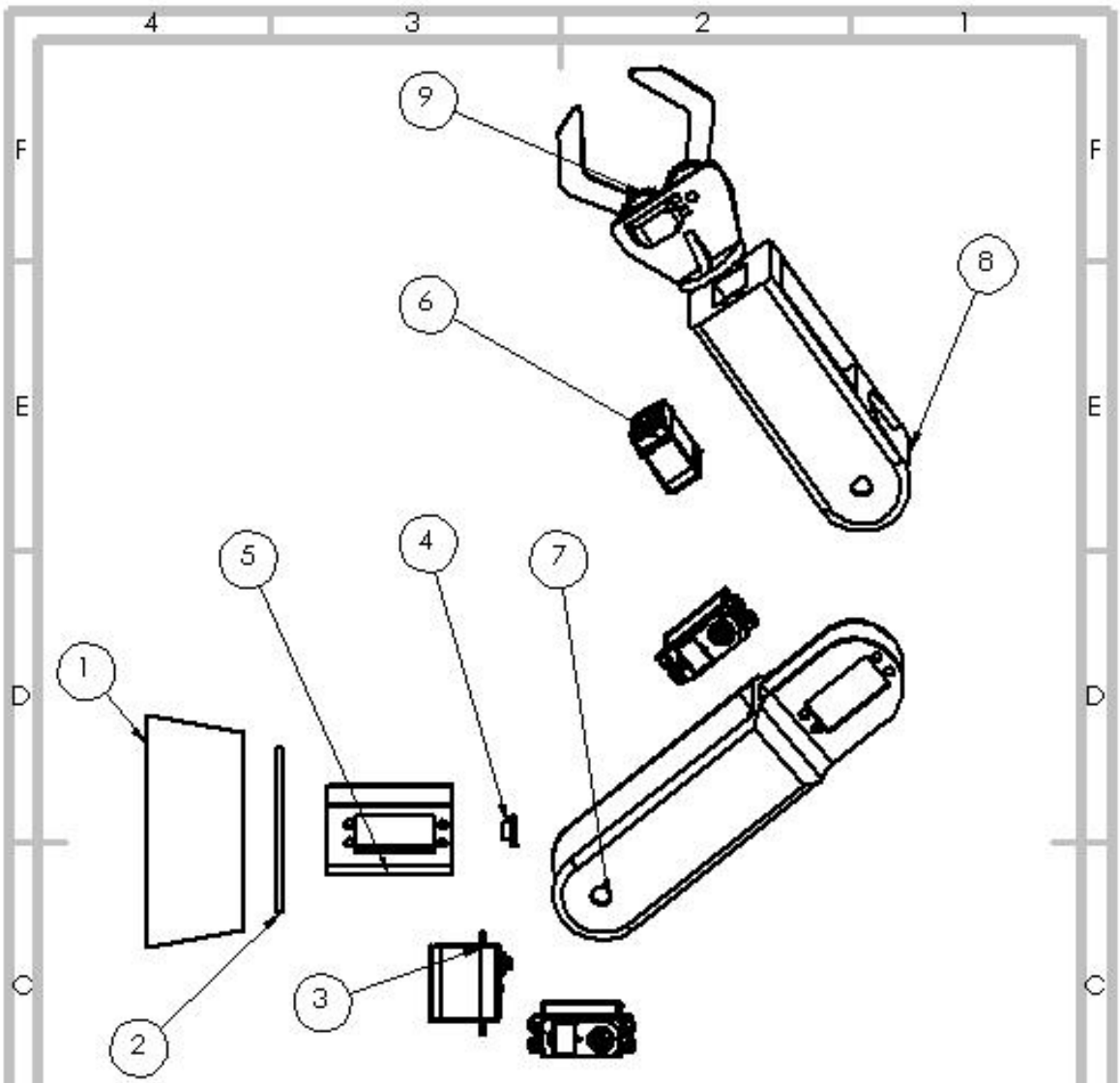
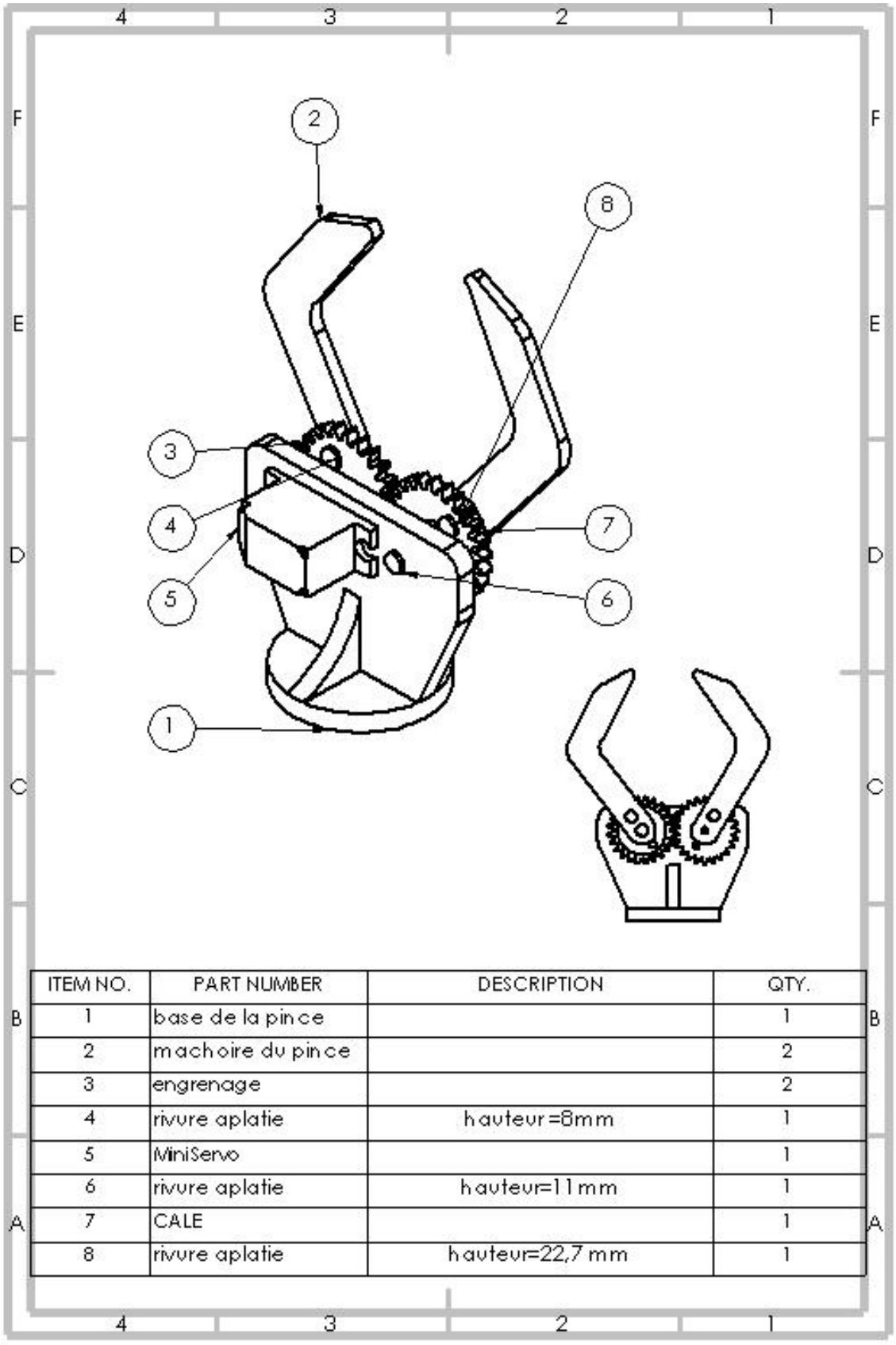


Figure 30 : Vue éclatée en 3D du bras robotique



ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	Base		1
2	Plateau tournant		1
3	SERVO		1
4	attache servo		1
5	support servo		1
6	STD Servomotor.step	servo moteur de puissance inférieur à celle du servo (3)	3
7	ARM1		1
8	ARM2		1
9	pince		1

Figure 31 : Vue éclatée avec nomenclature du bras robotique échelle 1 :3



ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	base de la pince		1
2	machoire du pince		2
3	engrenage		2
4	rivure aplatie	hauteur=8mm	1
5	MiniServo		1
6	rivure aplatie	hauteur=11mm	1
7	CALE		1
8	rivure aplatie	hauteur=22,7 mm	1

Figure 32 : Vue 3D avec nomenclature de la pince au bout du bras échelle 1 :1 et vue de face échelle 1 :3

2- Capture du mouvement

La reconnaissance des gestes de la main (ou Hand gesture recognition en anglais) est très importante pour l'interaction homme-machine. Par exemple, il peut constituer la base de la compréhension de la langue des signes et du contrôle des gestes de la main, et peut également permettre la superposition de contenu numérique et d'informations du monde physique dans la réalité augmentée.

Dans notre projet, nous utiliserons la « Motion Capture » pour contrôler un bras robotique, Cette technique sera utile lorsque quelqu'un travaille dans un environnement dangereux (peu accessible) ou pour manipuler des objets avec une meilleure précision.

A- L'aperçu de la méthode :

La solution de suivi manuel utilise un pipeline ML composé de plusieurs modèles travaillant ensemble :

- Un modèle de détecteur de paume qui fonctionne sur l'image entière et renvoie une boîte de sélection orientée.
- Un modèle de repère de main qui fonctionne sur la région d'image recadrée définie par le détecteur de paume et renvoie des points clés de main 3D haute-fidélité.
- Un outil de reconnaissance de gestes qui classe la configuration de point-clé précédemment calculée en un ensemble discret de gestes.

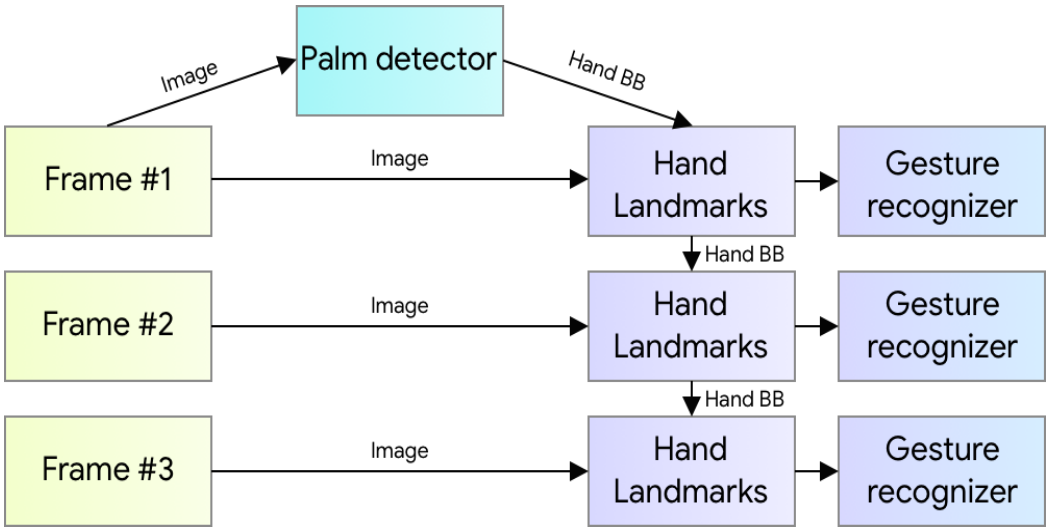


Figure 33 : Discrétisation des gestes enregistrés

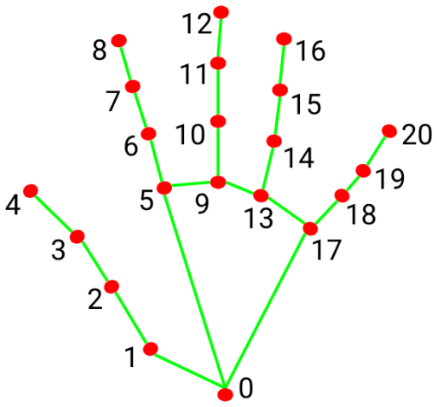
B- Détection de la paume de la main :

Le principe de cette méthode est de soustraire l'arrière-plan de l'image, si l'arrière-plan est statique, il est facile et efficace de détecter la région de la main de l'image originale en utilisant la méthode de soustraction d'arrière-plan. Cependant, dans certains cas, d'autres objets en mouvement sont inclus dans le résultat de la soustraction d'arrière-plan. La couleur de la peau peut être utilisée pour distinguer la région de la main des autres objets en mouvement. La couleur de la peau est mesurée avec le modèle HSV.

La mesure de la couleur se fait facilement en utilisant OpenCV en python.

C- Modèle de repérage de main :

Après la détection de la paume sur toute l'image, notre modèle de repère de main ultérieur effectue une localisation précise des points-clés de 21 coordonnées d'articulation 3D à l'intérieur des régions de main détectées par régression, c'est-à-dire prédiction directe des coordonnées.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 34 : Points de la main

Pour obtenir les données, ils ont annoté manuellement ~ 30K images du monde réel avec 21 coordonnées 3D, comme indiqué ci-dessous (ils prennent la valeur Z de la carte de profondeur d'image, si elle existe par coordonnée correspondante). Pour mieux couvrir les poses de main possibles et fournir une supervision supplémentaire sur la nature de la géométrie de la main, ils rendent également un modèle de main synthétique de haute qualité sur divers arrière-plans et le mappent aux coordonnées 3D correspondantes.

D- Reconnaissance des gestes :

Après la détection et le suivi de la main nous allons développer des algorithmes qui vont pouvoir nous donner en premier lieu l'abscisse et l'ordonné d'un point choisi et ayant comme référence la fenêtre ouverte lors de l'exécution du programme , puisqu'on a les coordonnés de la main on peut calculer la position exacte de la main pour commander le bras robotique, ou encore savoir la direction de déplacement de la main en calculant la différence entre la dernière position et la position courante, en second lieu on va pouvoir savoir si la main est ouverte ou fermée, cela va se faire en étudiant les différentes positions des points sur la main traquée.

E- Pourquoi utiliser la bibliothèque Mediapipe dans notre projet ?

Mediapipe est une bibliothèque puissante et complexe développée par Google et elle est livrée avec des algorithmes de suivi pré-entraînés que nous utilisons car il est vraiment difficile de former un algorithme d'apprentissage automatique avec 30000 images ou plus en utilisant nos ordinateurs et nous n'avons pas assez de temps pour faire tout le travail par nous-mêmes, nous avons essayé d'utiliser uniquement OpenCV mais les résultats ne sont pas aussi précis que les résultats obtenus avec mediapipe car nous n'avons utilisé que la détection de couleur sans ML.

3- Communication ordinateur-Microcontrôleur

La communication entre ordinateur et microcontrôleur doit respecter deux contraintes majeures :

- La première est que le code de traitement d'image est un code python, par conséquent la librairie d'utilisation de ce projet doit être compatible avec ce langage de programmation
- La deuxième est que le microcontrôleur utilisé (Arduino dans notre cas vu son accessibilité au niveau du prix ainsi que sa facilité d'utilisation) doit pouvoir communiquer via le protocole choisi.

Par conséquent, on se penchera sur une utilisation d'une interface UART/USB intégrée dans la carte Arduino ce qui permettra de communiquer de manière simple et efficace les données de sortie de la partie traitement d'image de notre solution.

A- Port Serie Arduino

Le port série pour une Arduino peut varier d'une carte à une autre. On aura besoin que d'un seul port dans notre cas. On utilisera une carte Arduino Uno dont les pins 0 et 1 représentent respectivement les pins RX et TX de l'UART.

Pour Rappel, L'UART/USART (Universal Asynchronous Receiver Transmitter) est un périphérique qui convertit les octets de données entrants et sortant en un flux de données série. Au niveau de la trame de données, le début est marqué par un bit niveau bas appelé starter et la fin est signalée un bit de stop ou deux (niveau haut). Un bit de parité (Parity Bit) est ajouté éventuellement afin de détecter des erreurs de transmission. De plus, la vitesse de transmission doit être configurée pour les périphériques communiquant en UART. Au niveau du câblage utilisé, l'UART utilise une seule ligne de données pour la transmission et une deuxième pour la réception.

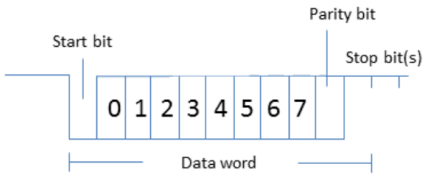


Figure 35 : Structure d'une trame de données envoyé par UART

Notons bien qu'au niveau de l'Arduino, une utilisation des pins 0 et 1 peut entrainer des erreurs lors du transfert des données.

B- PySerial

Les données récoltées par l'Arduino et utilisées pour le control du bras robotique ont été générées à la base grâce à l'ordinateur et du résultat des images de la caméra. Elles seront transférées par code python à travers la librairie PySerial et utiliseront le câble USB utilisé déjà pour alimenter la carte.

Par définition le module PySerial sert à exploiter les ports séries à disposition (par conséquent, elles rendent les données accessibles)

Les fonctions utilisées dans notre code :

```
Ser = Serial.Serial(''COM3'', 9600) ;
```

C'est une initialisation du port USB utilisé sur ordinateur connectant ce dernier à la carte Arduino.

COM3 est le port de communication et est trouvable dans le gestionnaire des périphériques après avoir connecté la carte.

9600 représente le baud rate c'est-à-dire la vitesse de transmission de données exprimée en baud correspondant à un symbole par seconde.

D'autres paramètres peuvent êtres configurés, mais on gardera les valeurs initiales par soucis de simplification.

`__init__`(*port=None, baudrate=9600, bytesize=EIGHTBITS, parity=PARITY_NONE, stopbits=STOPBITS_ONE, timeout=None, xonxoff=False, rtscts=False, write_timeout=None, dsrdtr=False, inter_byte_timeout=None, exclusive=None*)

Parameters:

- **port** – Device name or `None`.
- **baudrate** (*int*) – Baud rate such as 9600 or 115200 etc.
- **bytesize** – Number of data bits. Possible values: `FIVEBITS`, `SIXBITS`, `SEVENBITS`, `EIGHTBITS`
- **parity** – Enable parity checking. Possible values: `PARITY_NONE`, `PARITY_EVEN`, `PARITY_ODD` `PARITY_MARK`, `PARITY_SPACE`
- **stopbits** – Number of stop bits. Possible values: `STOPBITS_ONE`, `STOPBITS_ONE_POINT_FIVE`, `STOPBITS_TWO`
- **timeout** (*float*) – Set a read timeout value.
- **xonxoff** (*bool*) – Enable software flow control.
- **rtscts** (*bool*) – Enable hardware (RTS/CTS) flow control.
- **dsrdtr** (*bool*) – Enable hardware (DSR/DTR) flow control.
- **write_timeout** (*float*) – Set a write timeout value.
- **inter_byte_timeout** (*float*) – Inter-character timeout, `None` to disable (default).
- **exclusive** (*bool*) – Set exclusive access mode (POSIX only). A port cannot be opened in exclusive access mode if it is already open in exclusive access mode.

Figure 36 : Les paramètres de la méthode Serial dans le packet PySerial

:

Ser.write(bytes('S','utf-8')

Cette fonction permet d'envoyer des trames de données en série via le port déjà sélectionné sous forme d'octets. Dans notre cas, on envoie des unicode string et encodés sous le format 'utf-8' (Un codage de caractères).

4- Communication Carte Servo et Arduino

La communication établiée entre la carte servo utilisée dans notre cas ainsi que l'Arduino est une communication I2C (Inter-Integrated circuit). C'est un protocole

de communication half-duplex c'est-à-dire qui opère dans les deux sens mais pas de façon simultanée. Le bus I2C permet de connecter plusieurs esclaves et plusieurs maîtres.

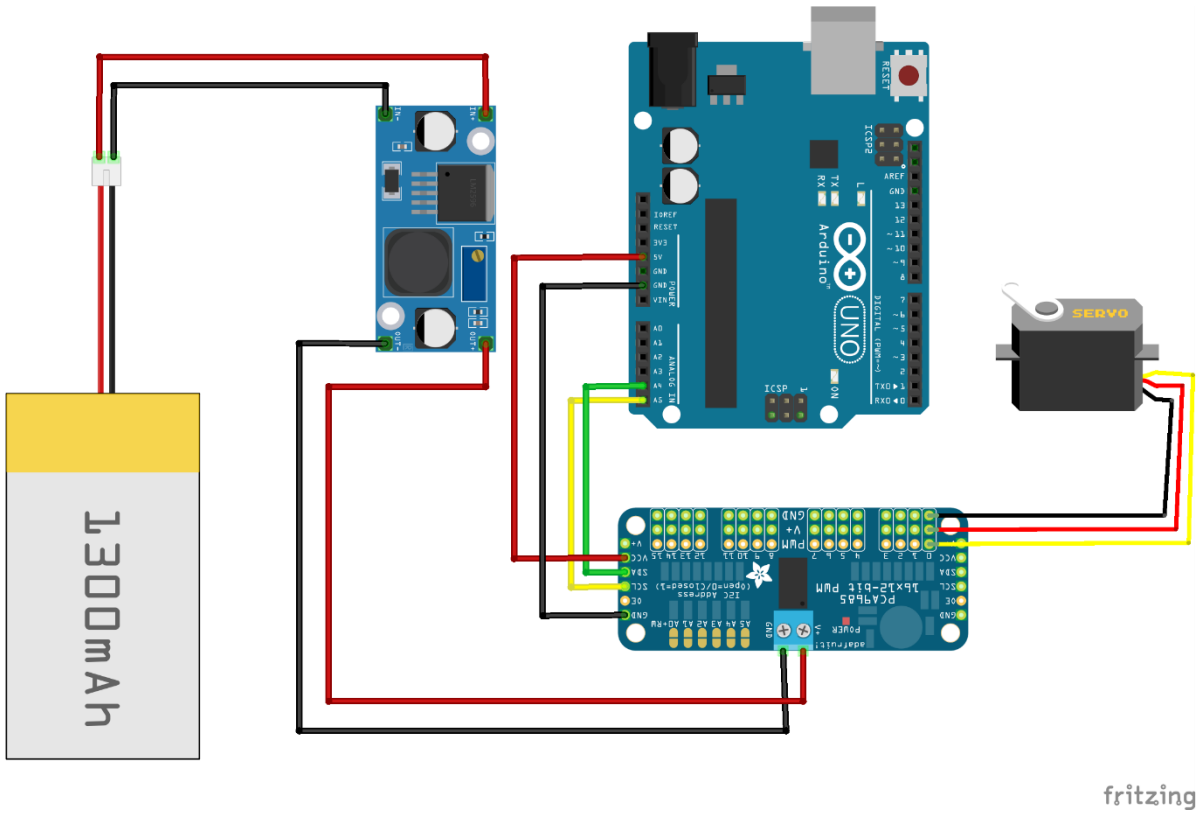
La connexion est réalisée à l'aide de deux cables : SDA permettant de transmettre les données et SCL permettant de synchroniser le maitre et l'esclave.

5- Mise en place

A- Présentation du matériel

- Carte Arduino Uno
- Ordinateur muni d'un WebCam
- 16-Channel 12-bit PWM/Servo Driver - PCA9685
- Jumper Wires
- Cables
- Régulateur de tension
- 2 Piles lithium 3.6V

B- Câblage



C- Code Arduino (partie microcontrôleur)

```
/*Cablage de la carte servo
PCA9685.....Uno/Nano
GND.....GND
OE.....N/A
SCL.....A5
SDA.....A4
VCC.....5V*/
#include <HCPCA9685.h>                // imporation de la bibliothéque pour la carte de commande des servo
#define  I2CAdd 0x40
HCPCA9685 HCPCA9685(I2CAdd);
char x;    // Variable ou on va stocker les données transmises par communication série

// Les fonctions G,D,H et B contiennent des configurations spécifiques pour les servos
// Elles sont executés selon l'ordre reçu par communication série avec l'ordinateur
 void G(){
 int Pos3=180;
 int Pos2=0 ;
 int Pos1=150;
 int Pos0=200;
 HCPCA9685.Servo(0,Pos0);
 HCPCA9685.Servo(1,Pos1);
 HCPCA9685.Servo(2,Pos2);
 HCPCA9685.Servo(4,Pos3);

 }
 void D(){
 int Pos3=180;
 int Pos2=0 ;
 int Pos1=150;
 int Pos0=0;
 HCPCA9685.Servo(0,Pos0);
 HCPCA9685.Servo(1,Pos1);
 HCPCA9685.Servo(2,Pos2);
 HCPCA9685.Servo(4,Pos3);

 }
 void H(){

 int Pos3=180;
 int Pos2=80 ;
 int Pos1=200;
 int Pos0=100;

 HCPCA9685.Servo(0,Pos0);
 HCPCA9685.Servo(1,Pos1);
 HCPCA9685.Servo(2,Pos2);
 HCPCA9685.Servo(4,Pos3);
 }
```

```
 void B(){
   int Pos3=240;
 int Pos2=0 ;
 int Pos1=240;
 int Pos0=100;
 HCPCA9685.Servo(0,Pos0);
 HCPCA9685.Servo(1,Pos1);
 HCPCA9685.Servo(2,Pos2);
 }
void setup()
{
  /* Initialisation de la librairie en mode Servo */
HCPCA9685.Init(SERVO_MODE);
  /* Etablir un communication série */
Serial.begin(9600);
  /* Définition du temps maximal d'attente , si dépassé on ne lit plus les donnés reçues */
Serial.setTimeout(1);

  /* Sortir la carte servo du mode veille */
  HCPCA9685.Sleep(false);


}
```

```
void loop()
{
  /* Première instruction pour voir si le port est disponible ou non et éviter les interférences  */
while (!Serial.available());
/* Stockage du caractère envoyé par l'ordinateur via communication série */
x = Serial.read();
/* la fonction char(x) permet de convertir la trame (octet) reçue en caractère */
if (char(x)=='B') {
B();
Serial.flush() ; // Elle permet de nettoyer la mémoire tampon
}
  if (char(x)=='H') {
  H();
   Serial.flush() ;
   }
   if (char(x)=='G') {
  G();
   Serial.flush() ;
   }
   if (char(x)=='D') {
  D();
   Serial.flush() ;
  }
}
```

Code Python (partie ordinateur)

Après l'installation des différentes bibliothèques pour le code « python » la première chose à faire est la création du nouveau script et l'importation des différentes bibliothèques et modules.

```
# -*- coding: utf-8 -*-
"""
Created on Tue May  4 09:50:58 2021

@author: imedb
"""

import cv2
import mediapipe as mp
import time
import HandTrackingModule as htm
import serial
```

L'étape suivante est l'initialisation de toutes les variables à utiliser et la définition du port pour la communication entre Arduino et python.

```
x= 0
y = 0
pTime = 0
cTime = 0
cap = cv2.VideoCapture(0)
detector = htm.handDetector()
tipIds = [4, 8, 12, 16, 20]
ser = serial.Serial("COM3",9600)
```

Dans la section principale du code on va lancer la camera puis l'algorithme de détection de la main et son « tracking », ensuite on va définir une liste dans laquelle on va mettre les valeurs des coordonnées **x** et **y** du point de la main qu'on veut suivre.

Dans la suite on va développer un algorithme qui nous permettra de savoir la direction dans laquelle la main se déplace (dans notre cas « haut », « bas », « gauche », « droite » et « stationnaire »).

```
while True:
    success, img = cap.read()
    img = detector.findHands(img, draw=True )
    lmList = detector.findPosition(img, draw=False)
    if len(lmList) != 0:
        #print(lmList[0])

        if int(lmList[0][1])- x  < -6 :
            print("droite")
            ser.write(bytes('D', 'utf-8'))
        elif int(lmList[0][1])  - x > 6 :
            print("gauche")
            ser.write(bytes("G", 'utf-8'))
        '''else :
            print('stationnaire')'''
        x =  int(lmList[0][1])

        if int(lmList[0][2])- y  < -6 :
            print("haut")
            ser.write(bytes('H', 'utf-8'))
        elif int(lmList[0][2])  - y > 6 :
            print("bas")
            ser.write(bytes('B', 'utf-8'))

        else :
            print('stationnaire')
            ser.write(bytes('S', 'utf-8'))
        y =  int(lmList[0][2])
```

L'étape suivante du code est l'implémentation d'un algorithme qui nous permettra de savoir la position du « pouce » car lors de la fermeture de la main le pouce prend une position différente que celle des autres doigts (tous les doigts se replient et descendent sauf le pouce il va de droite à gauche lors de la position main fermée), la seconde partie vas nous permettre de compter les doigts levés qui va être utile pour une implémentation ultérieure.

```
fingers = []

# Thumb
if lmList[tipIds[0]][1] > lmList[tipIds[0] - 1][1]:
    fingers.append(1)
else:
    fingers.append(0)

# 4 Fingers
for id in range(1, 5):
    if lmList[tipIds[id]][2] < lmList[tipIds[id] - 2][2]:
        fingers.append(1)
    else:
        fingers.append(0)

print(fingers)
totalFingers = fingers.count(1)
print(totalFingers)



if totalFingers == 0 :
    print('closed')
    ser.write(bytes('C', 'utf-8'))
else  :
   print('open')
   ser.write(bytes('O', 'utf-8'))
   print(bytes('O', 'utf-8'))
```

La partie suivante présente l'implémentation d'un compteur FPS pour visualiser l'impact du code sur les performances de la machine.

Enfin, on va fermer proprement la caméra de la machine pour éviter les crashs du code.

Conclusion :

Ce rapport portant sur le contrôle des bras robotiques en utilisant la capture de mouvements, menée dans le cadre du projet personnel professionnel inclut les quatre grandes parties suivantes : Les bras robotiques, la motion capture, une étude pratique, et l'implémentation du système de capture sur le bras robotique.

Nous avons commencé par énumérer l'utilité, les domaines d'application et les différents types des bras. Dans une deuxième partie on a présenté la motion capture, nous avons cité ses avantages et ses inconvénients, ses domaines d'applications et ses techniques.

Ensuite, nous avons étudié le bras de point de vue mécanique, puis on à développer les étapes nécessaires à la motion capture et les solutions de communication ordinateur-microcontrôleur et Carte Servo-microcontrôleur utilisées.

Enfin, on a présenté la mise en place réalisée pour mettre à l'application tout ce qui a été développé au paravent.

D'après les résultats obtenus, on peut conclure que le projet de contrôler un bras robotique à base de Motion Capture ne peut être destiné qu'au domaine des loisirs à l'état actuel vu qu'il manque de précision et qu'il est gourmand en termes de ressources matérielles. De plus, pour un contrôle plus précis, un développement de nos propres modèles de machine learning est envisageable et peut être un sujet d'un projet ultérieurement.

Références :

Brownridge, A. (2014, juin). *Real-Time Motion Capture for Analysis and Presentation Within Virtual*. https://e-space.mmu.ac.uk/326218/1/Thesis_Final.pdf

Google. (s. d.). *Hands Detection*. Mediapipe. Consulté le 8 mai 2021, à l'adresse https://google.github.io/mediapipe/solutions/hands.html

Renaud, A. (s. d.). *Tout savoir sur la motion capture, une technologie en plein boom !* Réalité-Virtuelle.com. Consulté le 10 juin 2021, à l'adresse https://www.realite-virtuelle.com/tout-savoir-motion-capture/

Wikipedia contributors. (2019, 2 octobre). *Bras manipulateur*. Wikipedia. https://fr.wikipedia.org/wiki/Bras_manipulateur

Wikipedia contributors. (2020a, mai 20). *Robot articulé*. Wikipedia. https://fr.wikipedia.org/wiki/Robot_articul%C3%A9

Wikipedia contributors. (2020b, décembre 24). *Robocoaster*. Wikipedia. https://fr.wikipedia.org/wiki/Robocoaster

Wikipedia contributors. (2021a, mars 23). *Robot médical*. Wikipedia. https://fr.wikipedia.org/wiki/Robot_m%C3%A9dical

Wikipedia contributors. (2021b, mai 2). *Robotique industrielle*. Wikipedia. https://fr.wikipedia.org/wiki/Robotique_industrielle

Wikipedia contributors. (2021c, mai 26). *Motion capture*. Wikipedia. https://en.wikipedia.org/wiki/Motion_capture