Rapport De Stage

Détection et reconnaissance de doigts

Par caméra « RGB-D » pour télé-opération gestuelle

D'une main robotique

KHALED Belgacem

N°étudiant : 11291653

N° Lit : C620

2014

Enseignant tuteur :

M. Nicolas JOUANDEAU

M. Adrien REVAULT D'ALLONES

A l'attention de M. Jean-Noël VITTAUT

Responsable du stage

Université Paris 8, Vincennes Saint-Denis
Laboratoire d'Informatique Avancée de Saint-Denis
2, rue de la Liberté
FR-93526 Saint-Denis Cedex
France

Rapport De Stage

Détection et reconnaissance de doigts

Par caméra « RGB-D » pour télé-opération gestuelle

D'une main robotique

KHALED Belgacem

N°étudiant : 11291653

N° Lit : C620

Juin/Juillet 2014

Tuteurs de Stage :

M. Nicolas JOUANDEAU

Email : n@ai.univ-paris8.fr

M. Adrien REVAULT D'ALLONES

Email : adrien.revault-d_allonnes@univ-paris8.fr

Numéro du Laboratoire : 01.49.40.70.44

Remerciements

Je ne saurai exprimer avec de simples mots toute ma gratitude et tout mon ressenti suite à l'accomplissement de ce stage, à mes mentors, aux organisateurs et aux collègues et collaborateurs qui en ont fait une expérience marquante.

Je tiens particulièrement à remercier M. Nicolas JOUANDEAU ainsi que M. Adrien REVAULT D'ALLONES pour m'avoir guidé et orienté tout au long de ce stage, en étant constamment présent lors des périodes de doutes ou de découragement, en sachant répondre à toutes mes interrogations et surtout en m'apprenant les bases de ce que doit être un informaticien .

J'ai appris, grâce à eux, l'importance de l'autonomie dans ce secteur ainsi que la façon dont doivent se faire les choses en termes de recherche, de présentation et de cohérence du travail effectué. J'en ressors plus mur, ayant bénéficié d'une bonne expérience du monde professionnel grâce au modèle que j'ai eu en leur personne. Un grand merci à eux pour la confiance qu'ils m'ont accordée et l'opportunité qu'ils m'ont offerte.

Je saurai gré également à tous mes collègues et collaborateurs dans ce stage pour avoir créé une excellente ambiance de travail qui plus est, est chaleureuse et où l'entraide et la coopération étaient les mots d'ordre.

Je tiens, pour finir, à remercier chaleureusement les secrétaires Mme Emmanuelle Najjar et Mme Sarah BEN NACEF d'avoir veillé à ce que nous travaillions dans d'excellentes conditions.

Sommaire

Présentation du stage

Contexte et environnement

Dans le cadre de la troisième année de Licence informatique à l'Université Paris8, un stage de 1 à 3 mois est prévu pour mettre en pratique les compétences acquises lors de ces trois dernières années d'étude.

Dans ce cadre, j'ai eu la chance et l'immense plaisir d'intégrer le Laboratoire d'Informatique Avancée de Saint-Denis (LIASD) au sein de l'Université Paris8 pour réaliser le projet proposé par l'équipe des laboratoires de recherche de l'Université de Paris Ouest Nanterre la Défense : « **RECONNAISSANCE ET SUIVI DES DOIGTS POUR TELE-OPERATION GESTUELLE DE LA MAIN ROBOT INMOOV** ».

Le domaine de la robotique est en constante évolution, se caractérisant essentiellement par l'apparition d'un nombre croissant de nouvelles technologies, originales et innovantes, et s'entoure d'une grande communauté très active. C'est donc avec un grand plaisir que je vous présente mon initiation au domaine de la robotique et mon évolution en son sein à travers ce projet de stage qui s'inscrit dans le cadre de la recherche susceptible de contribuer au renouveau de la robotique.

Pour expliquer ma contribution à ce projet, je vais consacrer la première partie de mon propos à la présentation générale du projet dans lequel s'inscrit mon stage et des objectifs de mon stage. J'y présenterai ensuite les outils mis à ma disposition dans le cadre de ce travail puis des quelques projets existants de reconnaissance de doigts. La partie suivante de mon propos sera consacrée à l'explication détaillée des algorithmes de détection et de reconnaissance des doigts que j'ai implémentés. Je présenterai, en dernier lieu, les expérimentations effectuées, pour terminer ce présent rapport par un bilan technique et personnel de mon stage.

Problématique et objectif final du stage

Afin d'être membrede la communauté active autour du projet inMoov - *le robot, fait à partir d'une imprimante 3D – qui sera présentée avec de plus amples détails dans la section II.3–* , le Laboratoire de Recherche Informatique et Electro Numérique de l'Université Paris Ouest Nanterre la Défense a proposé au LIASD de Paris8 de se charger du sujet de Contrôle à distance de la main du robot inMoov.

Le sujet en question s'articule en deux parties. La première s'articule autour du développement d'un module de contrôle des servomoteurs qui contrôle les doigts robot. Cette partie constitue le sujet de stage de mon collègue et sera expliquée et détaillée dans la suite de mon propos. La seconde s'articule autour du développement d'un module de télé-opération gestuelle par asservissement visuel via caméra de type rgb-d.

Ce dernier point constitue l'objet de mon sujet de stage qui consiste à détecter dans un premier temps la main de l'utilisateur, puis ses doigts et enfin, de contrôler la main robotique à partir des mouvements de ces derniers, le tout grâce à la caméra Asus Xtion live qui sera présentée dans la section II.2.

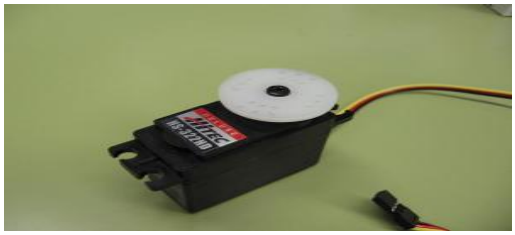
Lien avec les deux autres stages effectués dans le même laboratoire

Le sujet de stage étant divisé en plusieurs parties, nous étions trois informaticiens en dernière année de licence, à travailler autour. Si les parties composant le sujet de ce stage restent indépendantes les unes des autres, elles sont néanmoins étroitement liées dans leurs finalités.

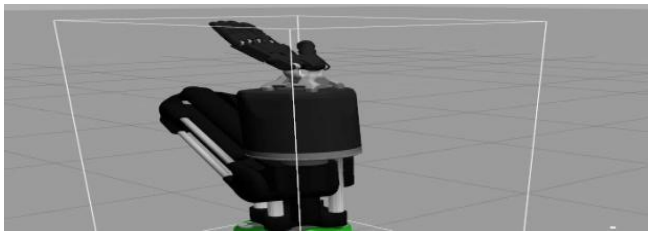
Mon travail et celui de mon collègue qui s'occupe de la programmation des modules de contrôle des servomoteurs responsables des mouvements des doigts de la main robotique grâce au fil reliant ces derniers, tel un engrenage, sont liés dans la mesure où, une fois le mouvement des doigts de l'utilisateurs détectés , mon application communiquera avec celle de mon collègue en lui envoyant la commande de mouvement que son application interprétera afin de bouger les servomoteurs.

Enfin le dernier sujet, traité par ma collègue, est de rendre certains modules de ROS (Robot Operating System – *qui sera présenté dans la prochaine section*) portables vers les nouvelles distributions de ROS. L'un de ces modules est lié à mon sujet de stage : il s'agit des

simulateurs de main robotique « shadow arm » et « shadow hand » qui permet de tester le projet, au préalable, sur un simulateur, afin d'éviter l'endommagement du matériel (la main robot).



Servomoteur



le simulateur shadow arm

Environnement et outils de travail

Présentation de ROS

Comme son nom l'indique, ROS (Robot Operating System) est un système d'exploitation pour robots. De même que les systèmes d'exploitation pour PC, serveurs ou appareils autonomes, ROS est un système d'exploitation complet pour la robotique de service. ROS est un méta-système d'exploitation, se situant, d'une certaine manière, entre le système d'exploitation et le middleware.

Il fournit des services proches d'un système d'exploitation (abstraction du matériel, gestion de la concurrence, des processus…) mais également des fonctionnalités de haut niveau (appels asynchrones, appels synchrones, base centralisée de données, système de paramétrage du robot…).

Avant les OS robotiques, chaque concepteur de robot, chaque chercheur en robotique consacrait un temps non négligeable à concevoir matériellement son robot ainsi que le logiciel embarqué associé.L'idée principale d'un OS robotique est d'éviter de réinventer la roue à chaque fois et de proposer des fonctionnalités standardisées faisant abstraction du matériel, tout comme un OS classique pour PC, d'où l'analogie de nom.

ROS est développé et maintenu par une société californienne, Willow Garage, fondée en 2006 par Scott Hassan, un des premiers employés de Google qui a participé au développement de la technologie du moteur de recherche et qui est aussi à l'origine des Yahoo Groups. Le PDG de Willow Garage est Steeve Cousin, un ancien d'IBM.

La philosophie de ROS se résume dans les cinq grands principes suivants :

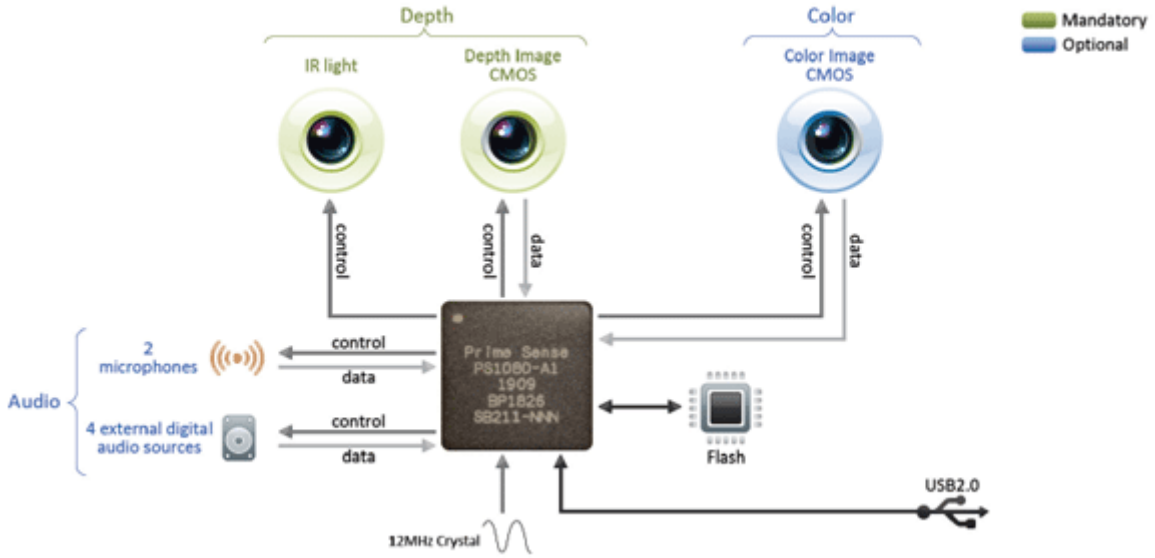
- Peer to Peer : permet à chacune des composantes d'un robot de dialoguer en direct avec une autre composante, de manière synchrone ou asynchrone en fonction des besoins.
- basée sur des outils (microkernel).
- multi langages : ROS est neutre d'un point de vue langage et peut être programmé en différents langages.
- léger
- gratuit et open source.

Ci-dessous les différents logosdes différentes distributions de ROS par ordre d'apparition :

Présentation de la camera Asus Xtion live pro

Présentation de la camera Asus Xtion live pro la Webcam Asus Xtion Pro Live est dotée d'un capteur couleur RVB (portée 0,8 à 3,5 mètres) ainsi que de capteurs infrarouges et de profondeur de champ permettant de suivre en temps réel les mouvements du corps. Adaptée notamment aux jeux, elle profite d'algorithmes complexes et dispose d'un micro stéréo intégrant une reconnaissance vocale capable d'offrir des possibilités encore plus étendues. Son champ d'utilisation est de 58° en mode horizontal, 45° en vertical et 70° en diagonale. Adoptant les langages de programmation C++/C# et C/C++, ce modèle est compatible avec les systèmes d'exploitation Windows (XP/Vista/7), mais aussi avec Linux Ubuntu (10.10 - X86) en 32 et 64-bits. La webcam offre une résolution de 640×480 et, en comparaison à la Kinect, l'autre capteur sensoriel de même type (utilisant la même technologie infrarouge crée

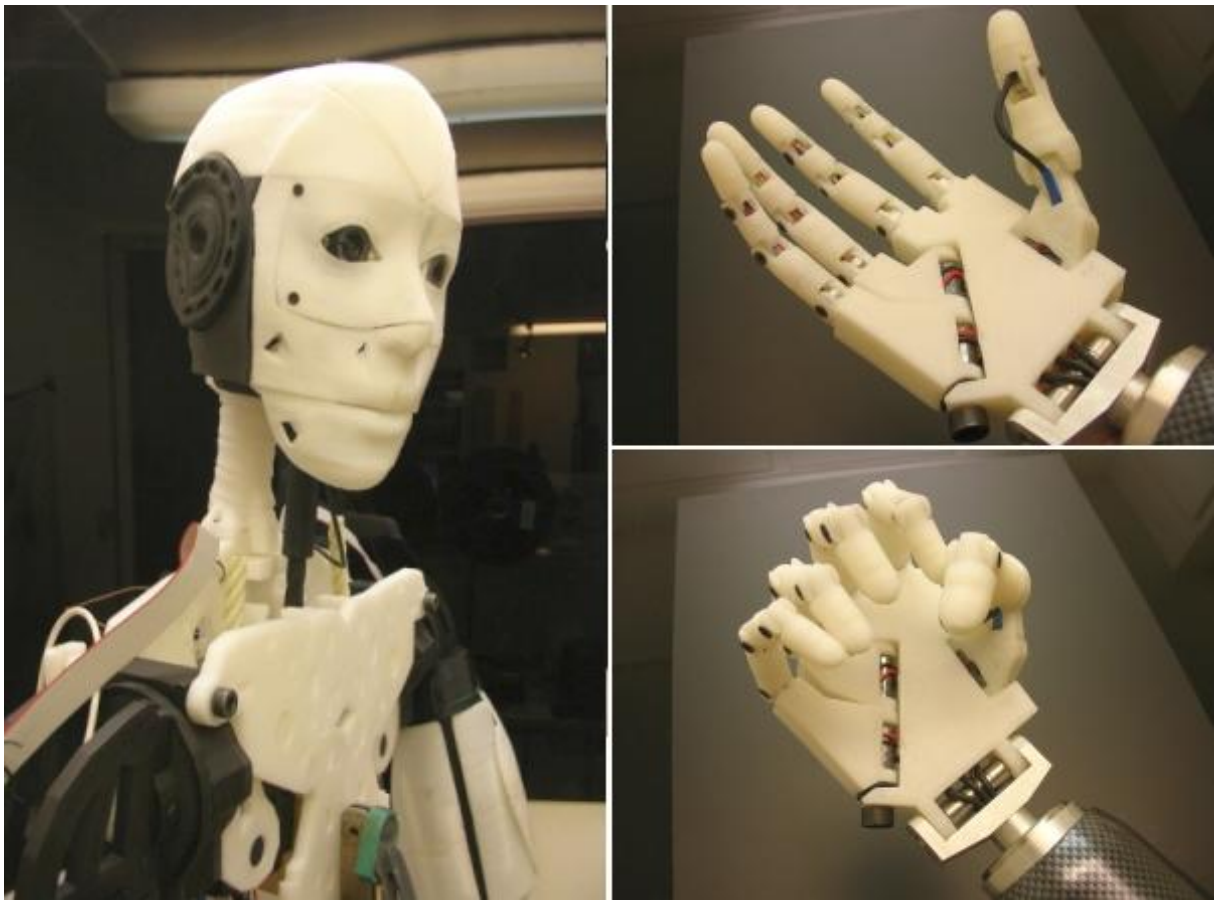
par la même entreprise primeSense), la Xtion pro Live est plus compact, légère et ergonomique. Elle ne nécessite pas d'une alimentation supplémentaire, excepté l'USB.



Présentation du robot inMoov plus particulièrement la main

InMoov est le premier robot Open Source à taille humaine entièrement imprimé en 3d. Réplicable sur n'importe quelle petite imprimante 3D avec une surface d'impression de 12x12x12cm, celui-ci est conçu comme une plateforme de développement autant pour les Universités, les Laboratoires que pour les passionnés de robotique. Son concept, basé sur le partage et la communauté, fait qu'il est déjà répliqué pour un bon nombre de projets partout à travers le monde. Fruit de plus d'un an de travail acharné du français, Gaël Langevin, ce grand robot humanoïde peut aujourd'hui être fabriqué par quiconque, ayant quelques compétences en électronique et/ou en programmation, grâce à un manuel de fabrication

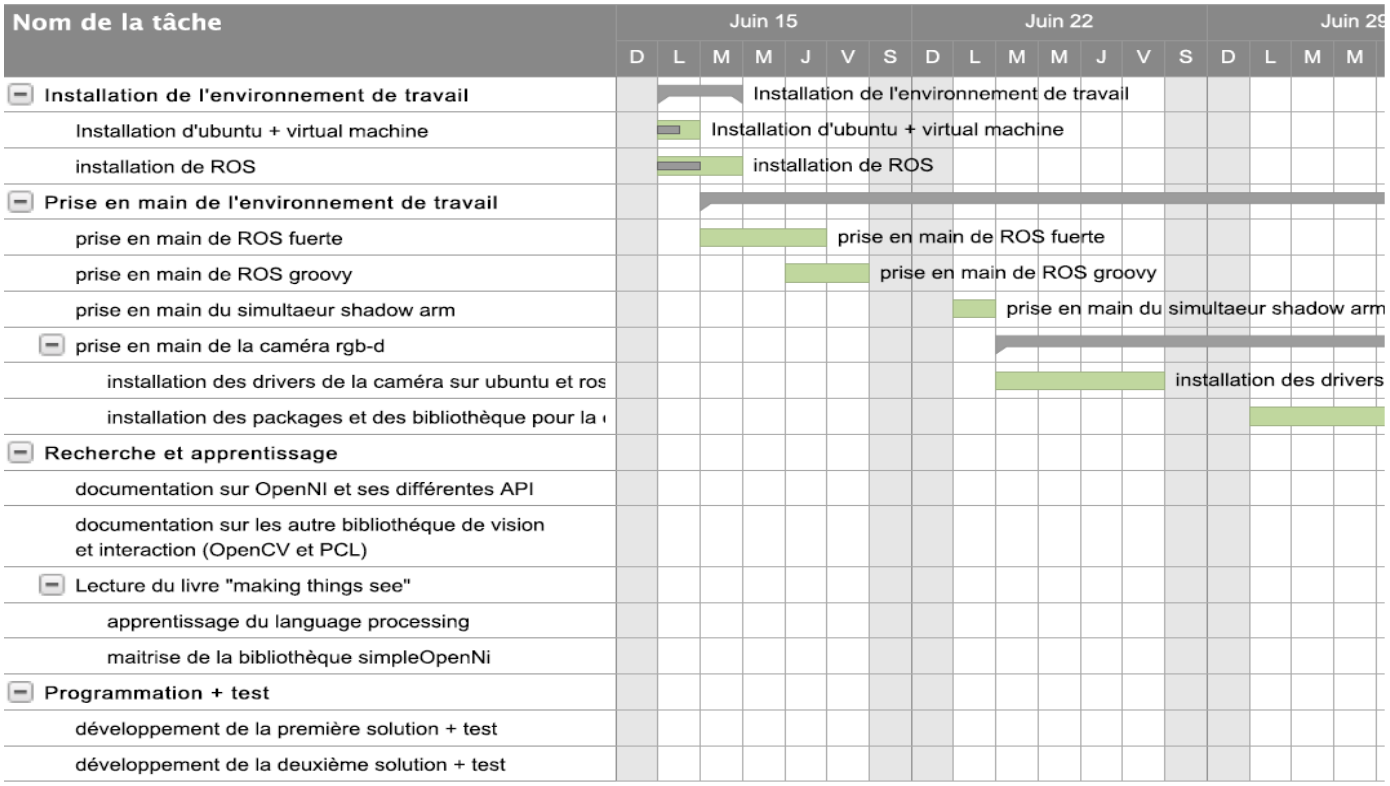
nommé « Do It Yourself». Le sujet de stage se focalise sur la main du robot inMoov, constituée de plusieurs servomoteurs servant d'articulation à cette dernière.

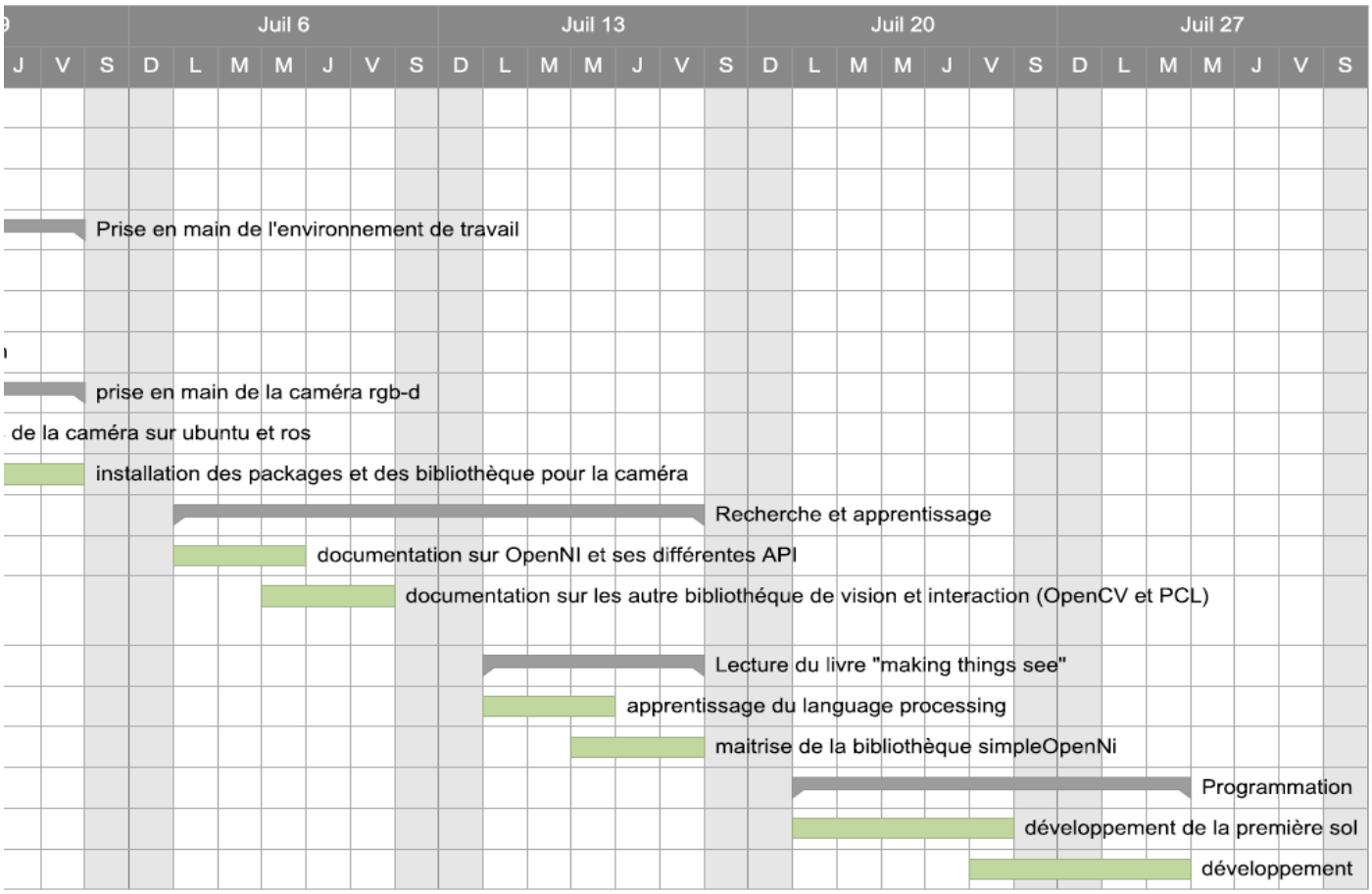


Déroulement du stage

*S'ajoutent, à ceci, deux réunions professionnelles. La première avait pour objet de présenter notre travail à nos collaborateurs et la seconde de leur en montrer l'évolution.

Diagramme de Gantt


Nom de la tâche
Juin 15
Juin 22
Juin 29
D L M M J V S D L M M J V S D L M M
Installation de l'environnement de travail
Installation de l'environnement de travail
Installation d'ubuntu + virtual machine
Installation d'ubuntu + virtual machine
installation de ROS
installation de ROS
Prise en main de l'environnement de travail
prise en main de ROS fuerte
prise en main de ROS fuerte
prise en main de ROS groovy
prise en main de ROS groovy
prise en main du simultaeur shadow arm
prise en main du simultaeur shadow arm
prise en main de la caméra rgb-d
installation des drivers de la caméra sur ubuntu et ros
installation des drivers
installation des packages et des bibliothèque pour la
Recherche et apprentissage
documentation sur OpenNI et ses différentes API
documentation sur les autre bibliothéque de vision et interaction (OpenCV et PCL)
Lecture du livre "making things see"
apprentissage du language processing
maitrise de la bibliothèque simpleOpenNi
Programmation + test
développement de la première solution + test
développement de la deuxième solution + test


Juil 6
Juil 13
Juil 20
Juil 27
J V S D L M M J V S D L M M J V S D L M M J V S D L M M J V S
Prise en main de l'environnement de travail
prise en main de la caméra rgb-d
de la caméra sur ubuntu et ros
installation des packages et des bibliothèque pour la caméra
Recherche et apprentissage
documentation sur OpenNI et ses différentes API
documentation sur les autre bibliothéque de vision et interaction (OpenCV et PCL)
Lecture du livre "making things see"
apprentissage du language processing
maitrise de la bibliothèque simpleOpenNi
Programmation
développement de la première sol
développement

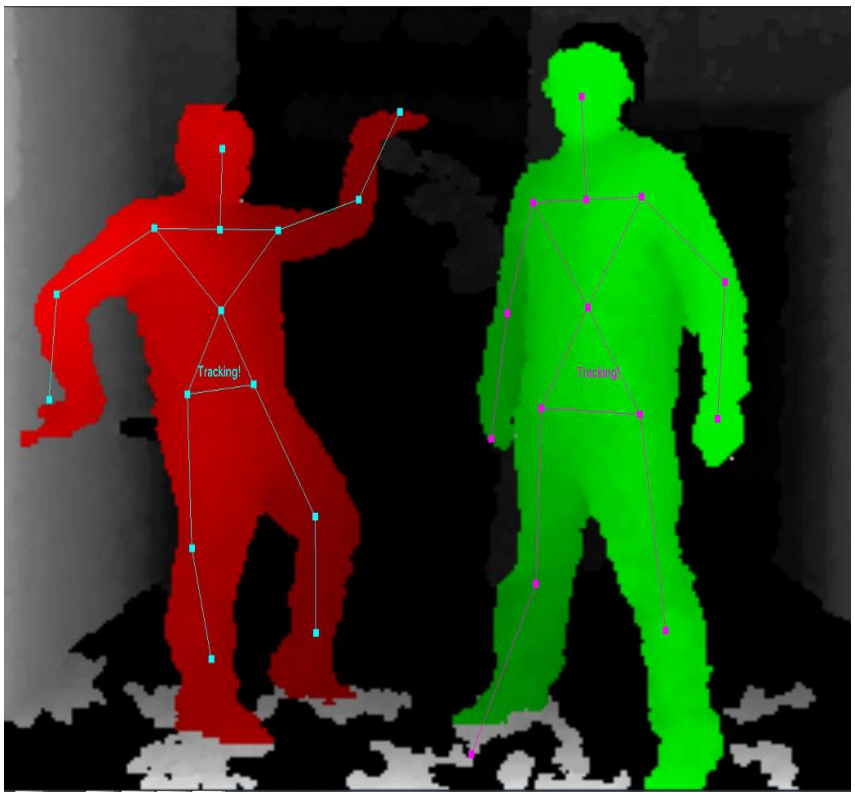
Existant

Les API OpenNi de reconnaissance visuelle

Plusieurs drivers et kits de développement permettent de programmer avec les cameras à capteur sensoriel. L'un des plus répandus et des plus connus es tOpenNI (pour open Natural Interaction) et ceci car OpenNI tente de standardiser, unir et simplifier la programmation pour tous les types de caméra sensorielle. Elle n'est pas spécifique à un seul matériel de reconnaissance et de capture de mouvement.

La force d'OpenNI réside dans les API proposées (les Interfaces de Programmation d'Application) : des outils pour développeurs pré-faits comme la reconnaissance du squelette humain et de ses articulations, la détection d'une main et son suivi, la reconnaissance vocale etc.

La solution apportée au sujet de stage sera programmée à l'aide de la bibliothèque SimpleOpenNi qui n'est rien de plus qu'OpenNI avec une syntaxe légèrement différente faite pour le langage« processing » utilisé.

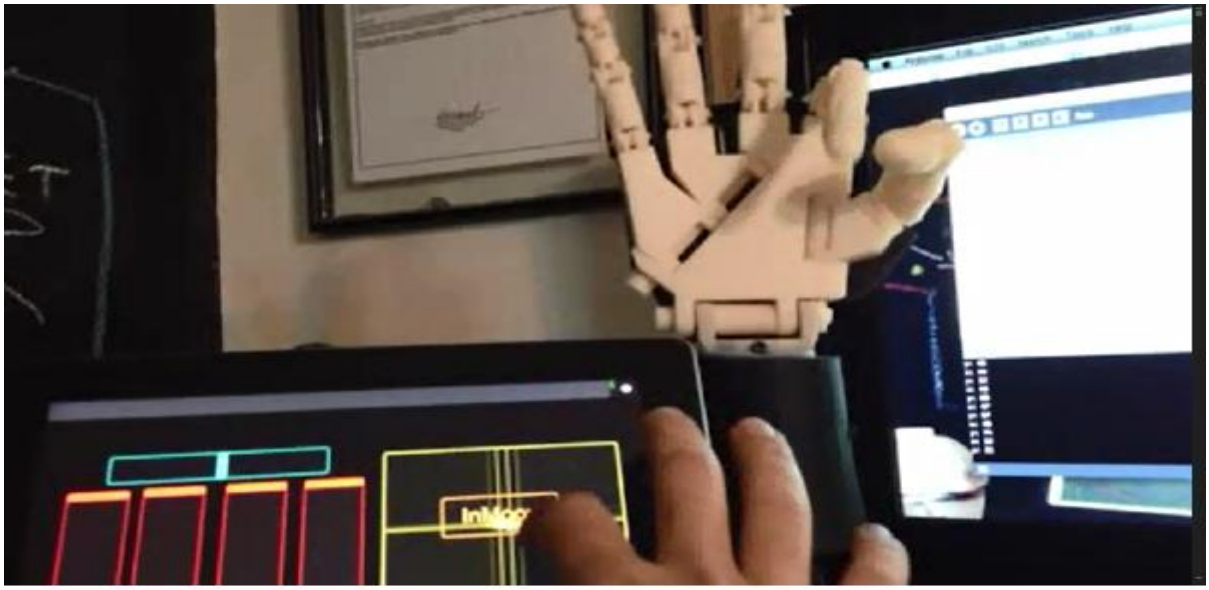


Projet similaire avec d'autres types de matériaux

Plusieurs recherches dans le domaine de l'interaction naturelle – *qui ont pour but de contrôler à distance une main robotique* – ont été effectuées, et cela un peu partout dans le monde. Trois d'entre elles ont particulièrement attiré mon attention dans la mesure où il s'agit de recherches qui touchent de près à ce que je dois faire.

Contrôle de la main InMoov avec un iPad touch :

Le contrôle de la main du robot inMoov se fait, ici, avec un ipad touch ayant une interface spéciale avec des barres dédiées au doigt et un carré dédié à l'action de plis de la paume.



Contrôle d'une main robotique grâce à un gant spécial :

Dans ce cas, la télé-opération de la main robotique se fait grâce à un gant multi sensoriel qui permet à la main robot de reproduire les gestes de l'opérateur humain.

Contrôle de la main inMoov grâce au capteur leap motion

Leap motion est un capteur sensoriel qui se place sous les mains à la hauteur du clavier dédié à la reconnaissance des mains et seulement les mains. Le principe est plus ou moins le même que celui des caméras sensorielles comme la Xtion Live mais il est spécifique aux mains et est donc beaucoup plus précis. On peut voir ici l'utilisateur contrôler la main du robot inMoov avec des gestes main libre ce qui se rapproche le plus de mon sujet de stage.

Solutions Apportées

Langage utilisé

Etant donnée la multitude de choix en termes de langages de programmation, notamment dans la programmation sur ROS ou avec OpenNI, il n'a pas été aisé d'en choisir un.

Le choix du langage de programmation « Processing » (basé sur java) s'est fait à partir de plusieurs critères de sélection malgré le fait que ce langage ne soit pas compatible avec ROS.

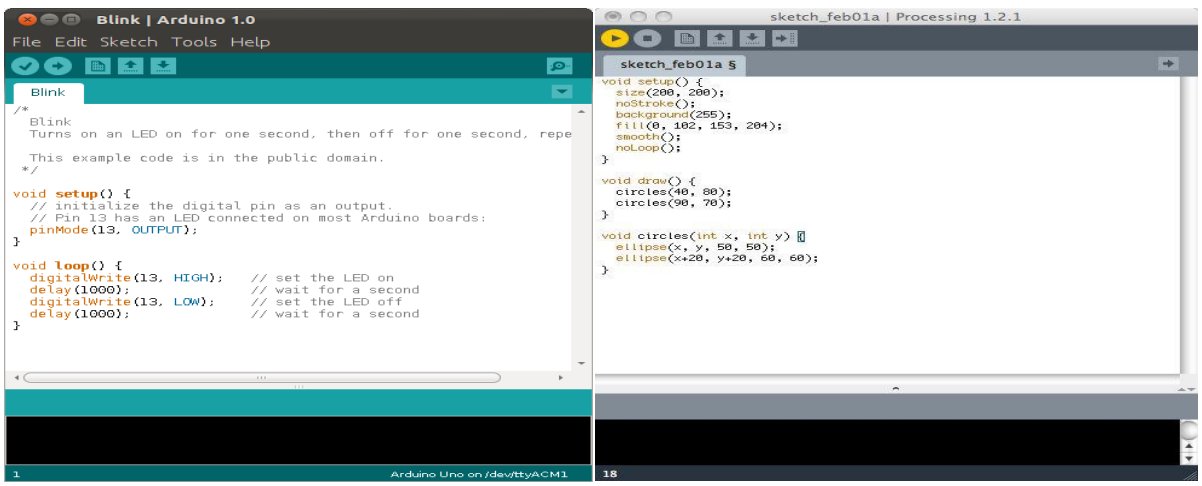
Premièrement, « Arduino » - la carte et le langage qu'utilise mon collègue pour programmer les servomoteurs des doigts du robot -, directement lié à la finalité de mon sujet de stage, communique facilement avec « Processing » par un simple import de bibliothèque. Ils ont la même interface de programmation.

Il y a ensuite, l'utilisation de la bibliothèque « SimpleOpenNI », qui existe seulement sur « processing » et qui est une version plus simple d'utilisation d'OpenNI.

En outre, le fait d'utiliser OpenNI– qui existe sous plusieurs langages de programmation dont le 'C' et le ' C++ ' compatible avec ROS – ainsi que la portabilité de l'application vers le ' C++ ' - autre langage objet comme 'Processing' - est, bien évidemment, plus facile à gérer.

Et enfin, la principale justification de ce choix est d'ordre bibliographique : la carence en matière de documentation et de cours pour programmer avec OpenNI. L'un des rares livres (voir ressources), si ce n'est le seul – à mon humble connaissance – qui aborde le sujet avec brio, est écrit en « processing ».

A gauche arduino **A droite processing**



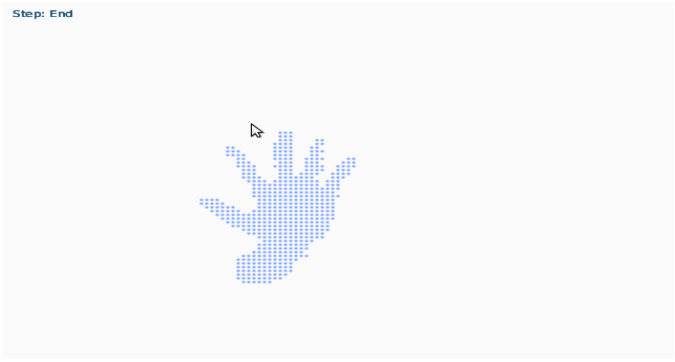
Détection de la main avec seuil

1ére méthode : seuil fixe

Grâce à la caméra de profondeur, se trouvant dans l'Asus Xtion Live Pro, il est possible d'obtenir la matrice de profondeur d'une image, cette dernière représentant chaque pixel de l'image par un entier allant de 0 a 255 et définissant sa profondeur.255 étant le point le plus proche et 0 le point le plus loin et sachant que le champ de vision de la Xtion live pro est entre 0.8m et 3.5m, on peut en déduire la distance réel avec quelques calculs.

La première méthode utilise cette carte de profondeur pour définir le seuil. Elle parcourt la carte, pixel par pixel, cherchant le point le plus proche, donc celui avec la plus grande valeur, en partant du principe que la main est la partie la plus proche de la caméra. Une fois ce point trouvé, le seuil est défini comme étant un peu plus loin de ce point afin de couvrir toute la main.

En pratique, une fois la main entrée dans le seuil, elle change de couleur afin de le signaler. Cette méthode a, néanmoins, comme contrainte, le fait d'obliger l'utilisateur à tendre la main vers l'avant sans qu'il y ait d'objet entre cette dernière et la caméra.



Seuil selon les points les plus proche

2éme méthode : seuil automatique

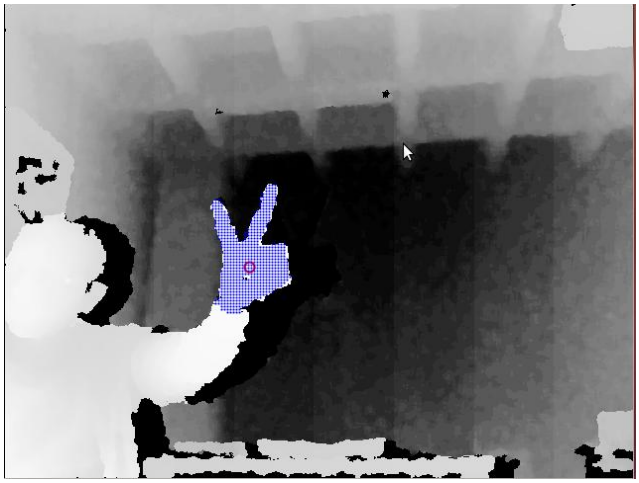
Afin d'éviter les contraintes imposées par la 1ére méthode, une 2éme solution fut mise en place. Cette dernière utilise l'une des API d'OpenNI, celle qui permet de détecter la main et de la suivre. Cette API dessine un point sur la paume de la main qui est presque à son milieu, et la suit en dessinant un trait dans la direction du mouvement de la main. En partant de cette API, en supprimant le trait qui suit la main et en agrandissant le point, on obtient le pixel représentant le milieu de la main (ni son centre ni son barycentre mais une approximation).

A partir de là, le seuil est détecté automatiquement et en 3D, en utilisant la position du point central et sa profondeur. Le seuil couvre un périmètre de 30cm environ autour du point afin de recouvrir toute la main (valeur agrandie intentionnellement afin de recouvrir différentes tailles de main) et est bornée par une profondeur maximum et minimum définie par la profondeur du point central (une vingtaine de centimètre devant et derrière la main).

Tout point se trouvant dans le seuil sera donc colorié en bleu afin de les différencier.

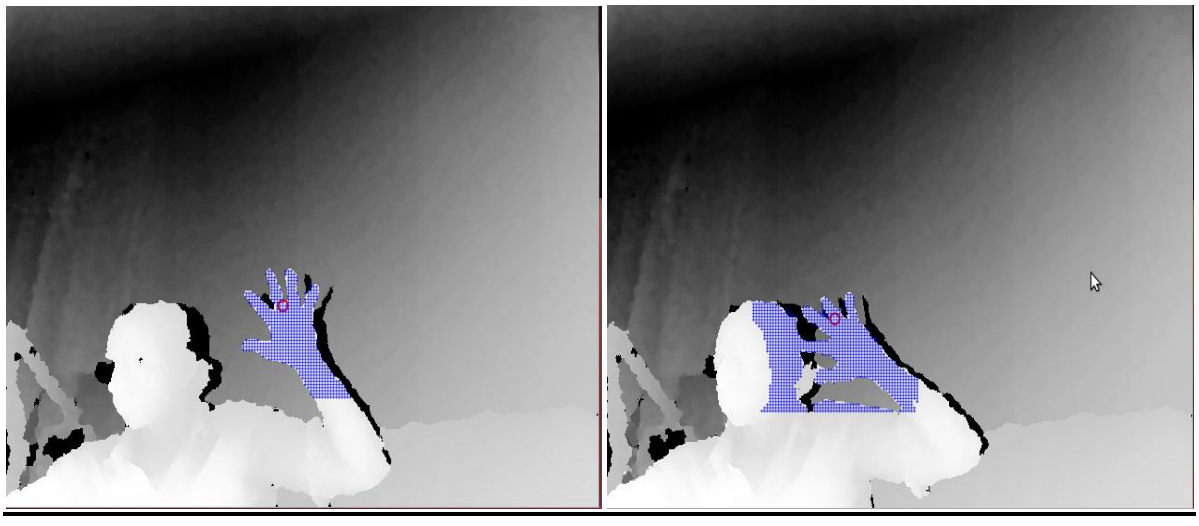
La seule contrainte restante est d'éviter le contact de la main et d'une autre partie du corps car, si celles-ci sont collées, certains points indiscernables seront considérés comme une partie de la main.

Détection par seuil automatique



Détection par seuil automatique : contraintes

A gauche : main espacé du corps **A droite : main collée au corps**



Détection des bouts des doigts

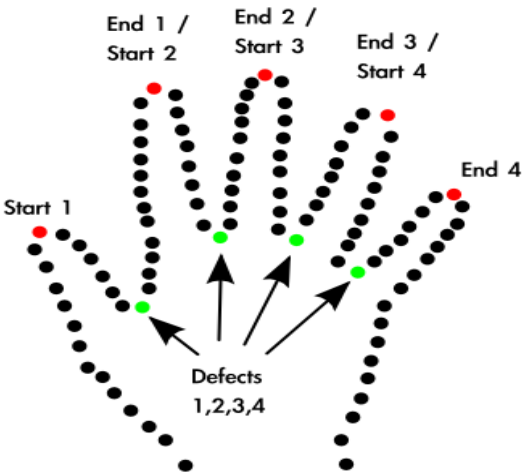
1ere méthode : « Convexhull »

Une fois la main détectée, grâce au seuil, on dispose d'un nuage de points afin d'effectuer le traitement d'image. La première méthode utilise l'algorithme du « convex hull » afin de déterminer le bout des doigts. Le « convex hull » est la coque qui recouvre un nuage de point en reliant les points les plus éloignés de ce dernier. On peut le considérer comme le plus grand périmètre d'un ensemble de points.

Ensuite, 2 méthodes permettent de trouver les extrémités des doigts : la première « convexity defects » trouve les points où la courbe du convex change de direction à partir d'un repère qui consiste à un point de départ et à point de fin qui correspondent au bout des doigts. Cette méthode a été abandonnée car, si un seul doigt est levé, l'algorithme en question ne trouvera pas de « convexity defect ». La deuxième consiste à trouver les « K-curvature », les angles entre deux lignes du convex qui sont inférieurs à un certain angle prédéfini qui correspondront à l'extrémité. Or avoir un angle prédéfini n'est pas la meilleure solution et cette méthode a donc également été abandonnée.

Ces méthodes restent en 2D : si on pointe un doigt vers la caméra, la détection sera plus dure. Les différentes méthodes de détection de doigts sont loin d'être parfaites à ce jour, c'est pourquoi le choix a été porté sur la deuxième solution pour la détection des bouts des doigts. Elle semblait être la plus avantageuse et la plus compatible à la forme des mains.

A gauche : Convex hull d'une main **A droite : convexity defects**



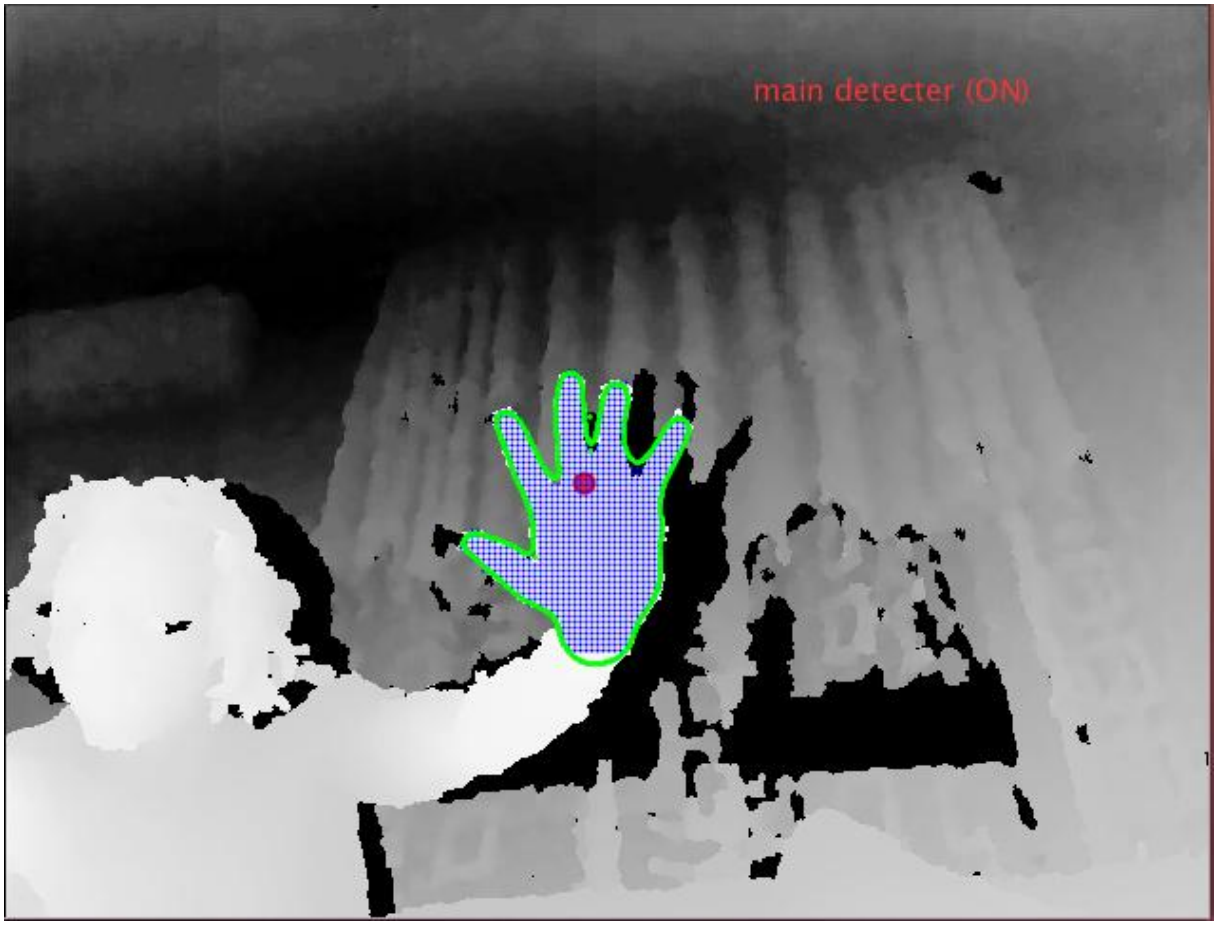
2éme méthode : Finger Tracker Library

La bibliothèque « finger tracker library » est une bibliothèque open source se basant sur l'algorithme de « fast marching squares » pour trouver le contour de la main et pour ensuite détecter les bouts des doigts en calculant les courbures et les inflexions du contour.

Cette méthode utilise un seuil prédéfini pour trouver le contour de la main, elle a donc été modifiée pour que le seuil soit défini automatiquement, en correspondant au seuil de la main détectée précédemment. Quelques détails visuels ont été modifiés aussi afin de voir plus clairement les bouts des doigts.

La contrainte de cette méthode est qu'il faille garder le bras en face du corps car, si la détection automatique du seuil de la main est indépendante de la position de la main par rapport au corps, le seuil du contour de la main pour détecter le bout des doigts, lui, commence là ou se trouve la main mais continue jusqu'au bout (Limiter de devant mais pas de derrière contrairement au seuil de la main limité des 2 cotés).

Un message à l'écran nous signale si une main est détectée ou pas (ON/OFF).



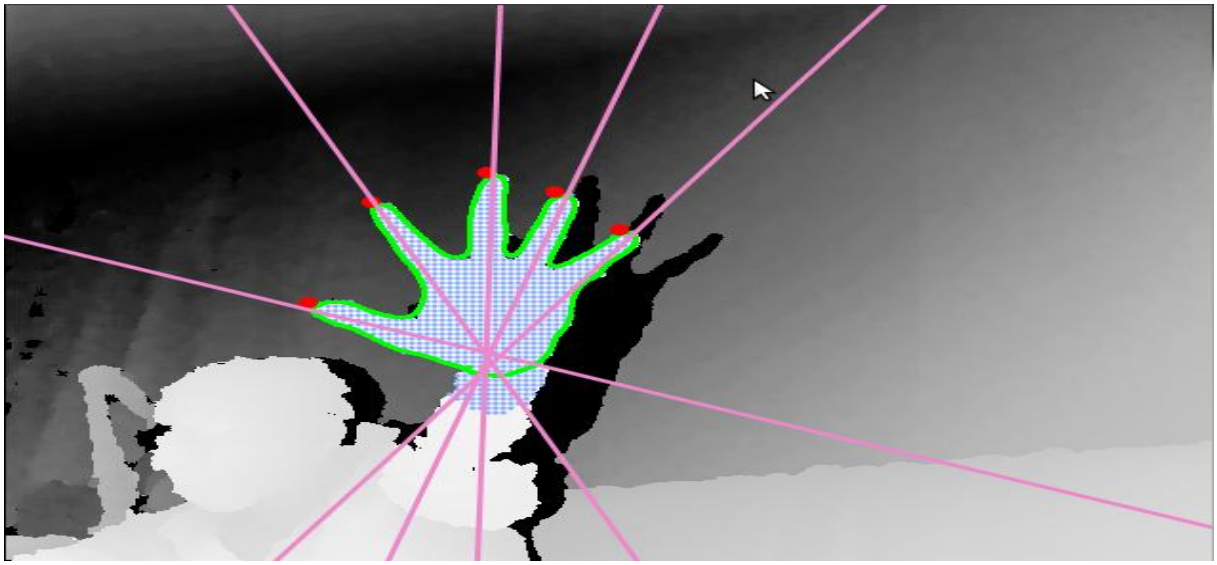
Détection des droites représentant les doigts

1ére méthode : Algorithme de RANSAC

Pour trouver les droites représentant les doigts de la main, l'utilisation de l'algorithme de RANSAC semblait être une bonne option. RANSAC pour « RANdom SAmple Consensus » est une méthode mathématique qui permet de trouver la droite régressive passant par le plus grand nombre de points dans un nuage de points donné. Plus l'algorithme tourne longtemps plus il y a de chances de trouver la meilleure droite (la plus longue passant par le maximum de points).

La première méthode consiste donc à utiliser cet algorithme et le faire tourner 5 fois par image pour avoir 5 droites régressives, le nuage de points donnés étant le nuage de points de la main détectée plus tôt (en bleu). Pour avoir 5 droites distinctes, une condition fut rajoutée : chaque droite doit passer par un bout de doigt et sans répétition. C'est de cette manière que les doigts sont détectés avec cette méthode.

Or, pour avoir les meilleurs droites et les plus précises, il fallait faire tourner l'algorithme plus longtemps pour chaque doigts, et pour un traitement en temps réel, le temps de latence était inacceptable. Une amélioration de ce temps de latence a été faite mais en dépit de la qualité des droites trouvées. C'est ainsi que les droites passant par le bout des doigts et étant les plus longues n'étaient pas forcément les droites représentant les doigts, particulièrement en ce qui concerne le petit doigt et le pouce. C'est la raison pour laquelle cette méthode a été abandonnée.



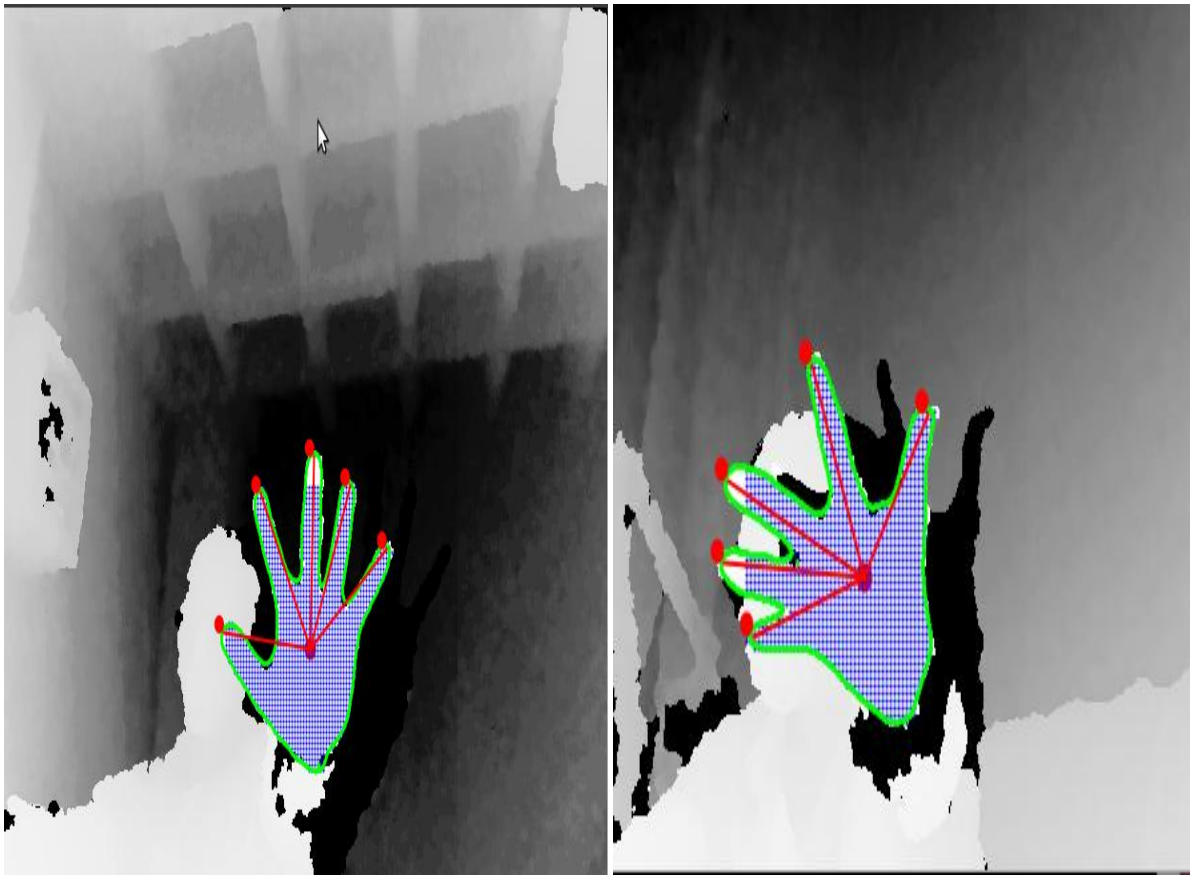
2éme méthode : du centre au bout des doigts

Cette méthode simple consiste à relier par une droite chaque bout de doigt au centre de la main, or le centre en question, comme mentionné précédemment, n'est pas vraiment le centre de la main. C'est alors qu'un traitement intervient sur le point censé représenter le centre, le déplacer légèrement selon la position de la main afin qu'il corresponde au milieu de manière plus précise.

Ainsi, si la main est dirigée vers le haut, le point sera repositionné plus bas, si la main est orientée vers le bas, le point sera repositionné plus haut etc.

L'orientation de la main est définie selon la position des bouts des doigts par rapport au centre.

Les contraintes et les différentes positions seront expliquées dans la partie Expérimentation.



Module de contrôle ROS et Communication inter module

Avant le début de développement des solutions et après négociation avec mon collègue qui s'occupe de la programmation des servomoteurs des doigts du robot, la phase de programmation à commencer par la communication inter module de ros.

J'ai alors crée les 2 nœuds (qui correspondent à des applications) dont le serveur est un type de message qui correspond à une structure de données, afin de communiquer le résultat de mon sujet principale à l'application gérer par mon collègue.

Ces derniers ont été écrits en C++. Or les solutions ont été écrites en « processing » et en « arduino », qui eux peuvent communiquer directement entre eux. C'est pourquoi ces modules de communication ros ont été mis de côté.

Mais malheureusement, la fin du stage m'ayant rattrapé avant l'accomplissement de mon sujet de stage, cette partie n'a pas été complétée et est restée inachevée.

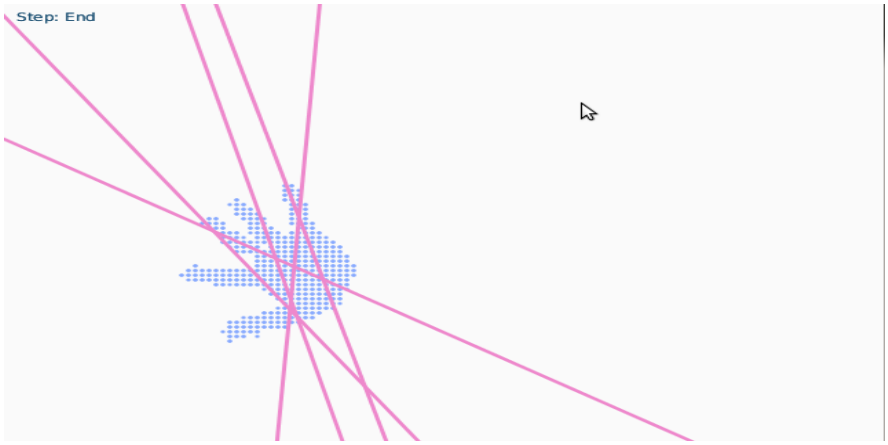
Expérimentation

Test des résultats avec plusieurs positions et orientations différentes de la main selon la première solution

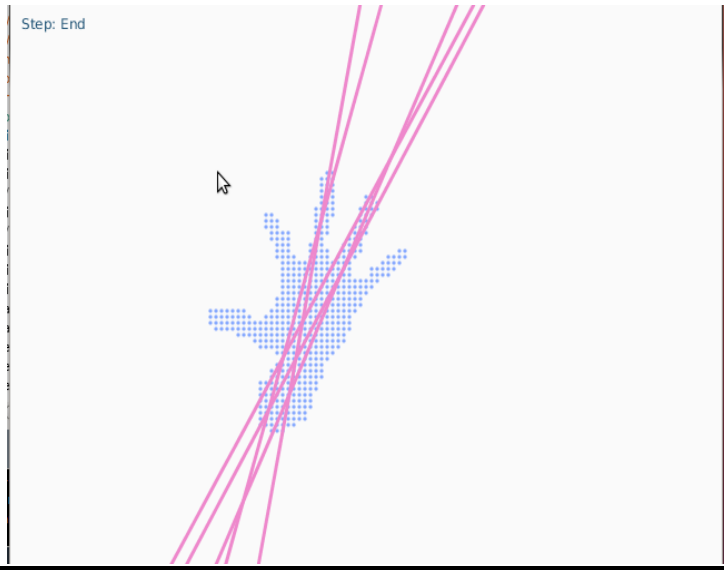
La première méthode de détection des doigts, avec ou sans seuillage automatique, présentait des inconvénients qui revenaient souvent lors des phases de test et d'expérimentation.

Par exemple, avant l'intégration de la condition selon laquelle les droites régressives trouvées par la méthode RANSAC passent forcément par l'un des bout des doigts, les résultats étaient souvent de deux types :

En désordre totale :

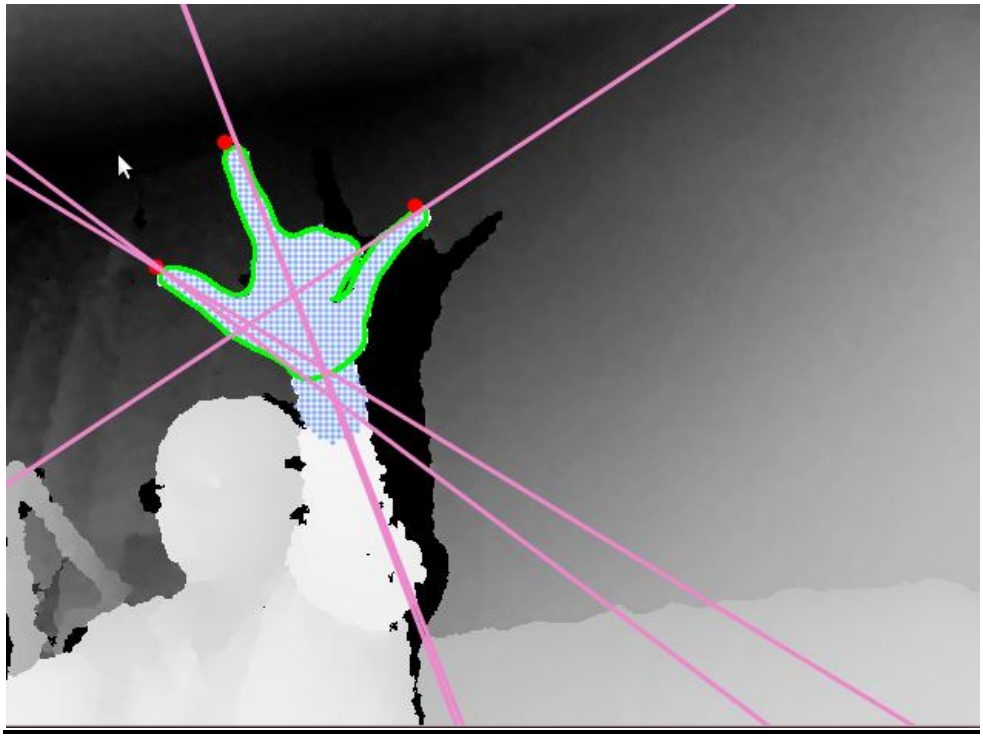


Ou rassembler sur le même doigt :

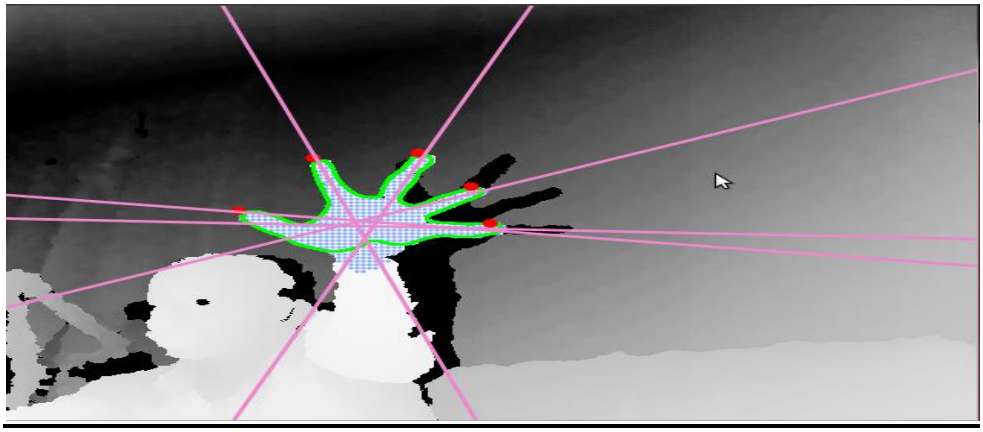


Après l'ajout de la condition pour que les droites passent par le bout des doigts, les tests ont été établis par rapport à la position de la main, son orientation ainsi que le nombre de doigts soulevés. C'est alors qu'une autre forme d'inconvénient est apparue :

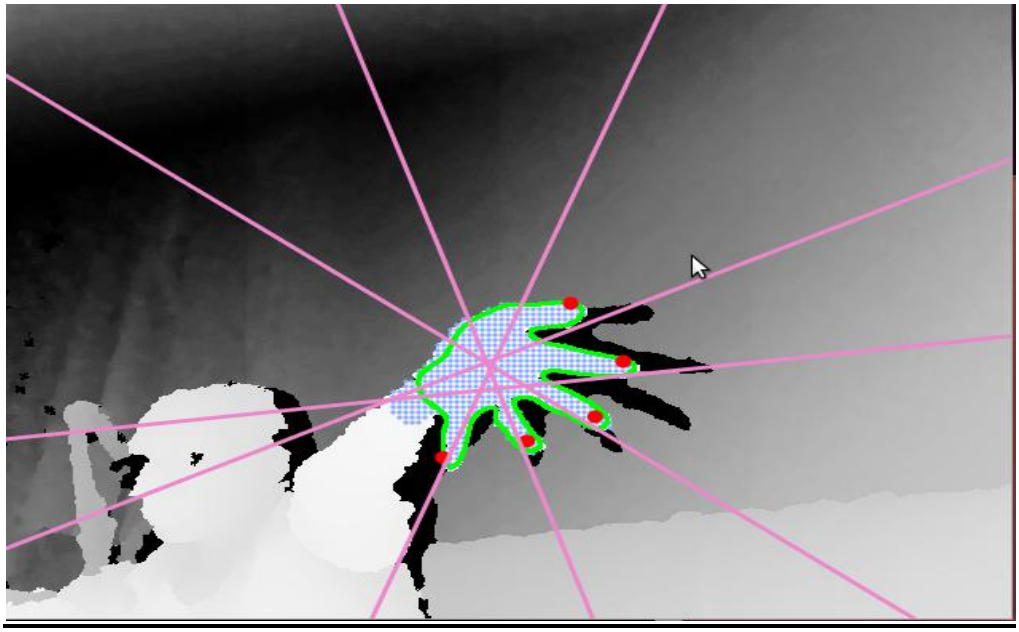
Le petit doigt qui n'est pas droit



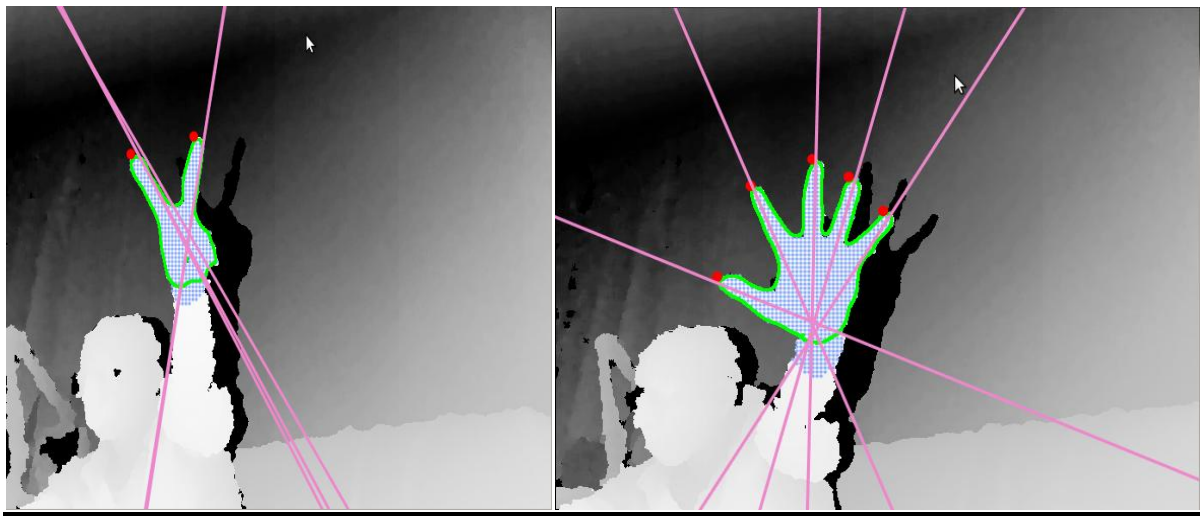
La droite qui devrait représenter le pouce est loin de le faire



Avec une orientation et une position tendue , le résultat n'est plus très fiable



Mais donne parfois de bons résultats :



Test des résultats avec plusieurs positions et orientations différentes de la main selon la deuxième solution + seuil automatique

Détection de la main par seuil automatique

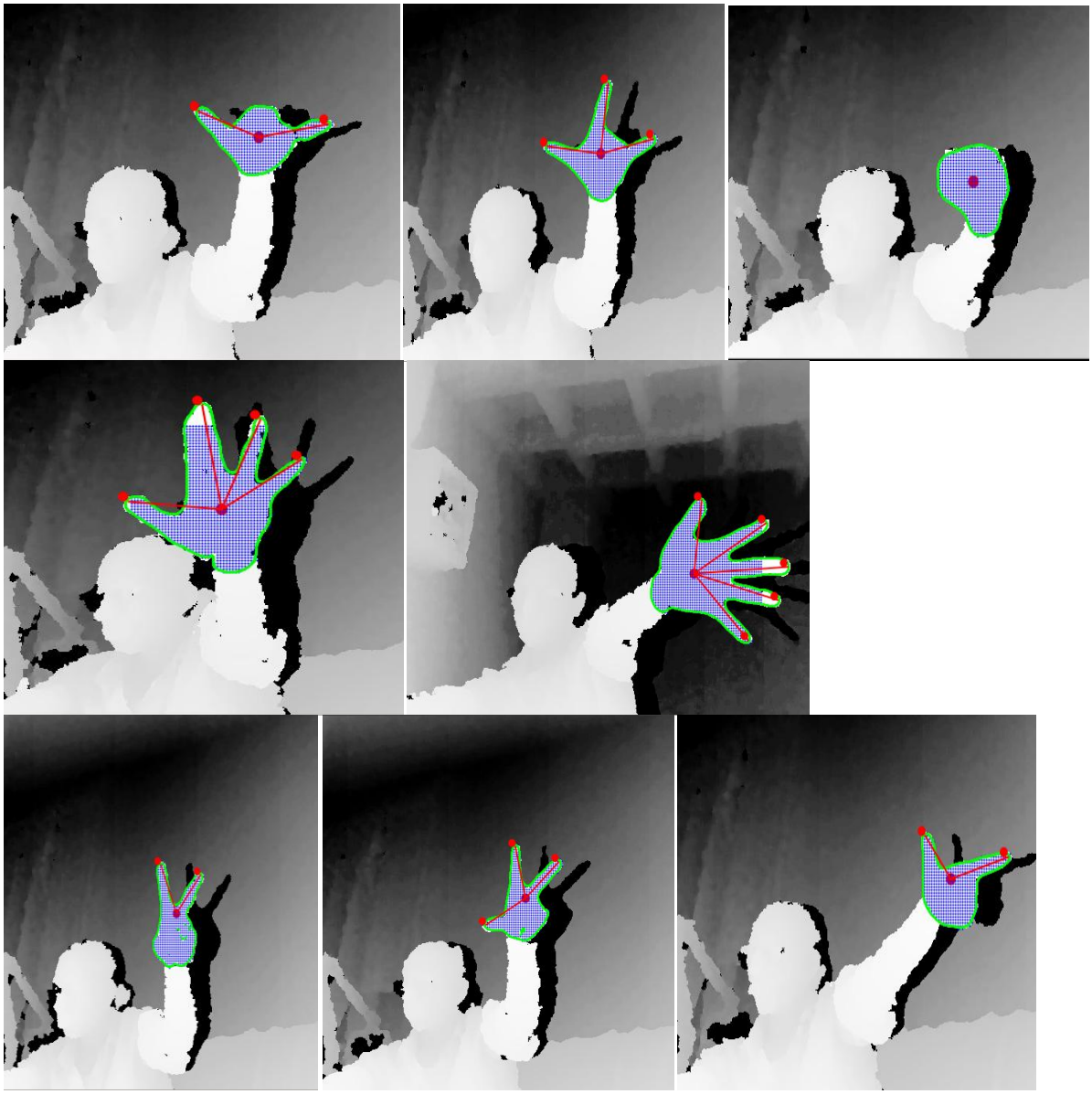
Le test de la détection de la main par seuillage automatique à été effectué avec de nombreuses positions et avec des orientations et formes différentes. En évitant la contrainte qui est de coller la main à une partie du corps, la détection marche bien. En voici un aperçu avec, en ordre de lecture, à l'envers, derrière, devant et en haut.

Détection des doigts en reliant leurs bouts au centre de la main

Après application de la deuxième méthode de détection des doigts, la phase d'expérimentation a porté sur le test de différentes positions, orientations et nombres de doigts de la main. Résultat satisfaisant dans la plupart des cas.

Contrainte: ne marche pas bien quand deux doigts ou plus sont collés, le bout des doigts est alors uni, on aura donc des doigts en moins.

Aperçu des résultats :



Conclusion

En collaboration avec le laboratoire informatique de l'Université Paris Ouest Nanterre la Défense, j'ai effectué un stage dans le Laboratoire d'informatique avancé de Saint-Denis portant sur le sujet de télé-opération gestuelle d'une main robotique grâce à une caméra rgb-d (3D) détectant la profondeur et les mouvements. Le stage s'est effectué dans le laboratoire de LIASD à l'Université Paris8 avec deux collègues travaillant sur le même thème mais avec des sujets différents. Après plusieurs phases de prise en main de l'environnement de travail, d'apprentissage de nouvelles technologies, de développement de solutions et de phases de tests le stage s'est terminé, sans avoir atteint l'objectif final mais avec des résultats prometteurs, laissant derrière lui une belle expérience très enrichissante.

Bilan de stage

Malgré les interrogations et les inquiétudes que j'avais avant de passer le stage de fin de licence, ce dernier se solde par leur dissolution. Au fur et à mesure que le stage avançait, je m'apercevais à quel point les études et le monde professionnel en informatique ce ressemblaient. Je ne saurais dire si ce sont les cours qui nous ont préparés aux métiers, ou si le métier était lui-même un constant apprentissage mais ce qui est certain, c'est que la façon de faire est identique.Je pourrais, éventuellement, comparer mon sujet de stage à plusieurs projets effectués lors de mes études, mais ce dernier est, à mon humble avis, de plus grande ampleur et de plus grande envergure, ouvrant de nouveaux horizons et de nouvelles perspectives.

Ceci dit, le sujet de stage en question me paraissait de plus en plus intéressant, au fur et à mesure que je comprenais le thème et le sujet de la robotique ainsi que de la vision par

ordinateur et les caméras 3D. Toutes les possibilités de ce domaine qui se sont offertes à moi m'ont inspiré pour mes futurs projets, et cela par le fait de m'encourager à élargir mes horizons, d'une part, ou par cette passion qu'elles ont fait naître en moi pour la robotique d'autre part. En y ajoutant l'acquis professionnel et les expériences gagnées en termes de vision par ordinateur et manipulation d'image 3D, les entreprises maitrisant cette technologie ne sont plus une porte fermée à mes yeux.

Pour ce qui est de l'ambiance au travail, elle pourrait se résumer en un seul mot : excellente. L'ambiance était chaleureuse, formée de personnes toutes aussi passionnées les unes que les autres qui s'aidaient mutuellement et qui continuaient à parler programmation même lors des pauses. Nos tuteurs de stages étaient accueillants, présents pour nous expliquer, pour nous guider et répondre à toutes nos interrogations. La communication était également très satisfaisante, se caractérisant essentiellement par la réactivité et l'efficacité des uns et des autres, en direct ou par mail. Nos tuteurs nous ont fortement encouragés à murir dans ce métier. Un grand merci à eux.

Ressources

Livre utilisé pour l'apprentissage : « Making things see » :

- Consulter en pdf

- Nom complet: Making Things See: 3D vision with Kinect, Processing, Arduino, and MakerBot (Make: Books)
- Auteur : Greg Borenstein
- Date de sortie : 06/02/2012
- Nombre de pages : 437

Algorithme trouvé et utilisé et/ou modifier :

- convex hull algorithme
- la library finger tracker
- RANSAC algorithme

Les ressources webographiques :

- Ros.wiki.org
- Le site officiel de la documentation pour OpenNI