Hes·so VALAIS WALLIS
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences Western Switzerland

Travail de Bachelor 2013

Filière Informatique de gestion

Ki-Mouse

Une application Windows permettant de contrôler sa souris en utilisant les membres de son corps.

Étudiant : Romain Paccolat

Professeur : Jean-Pierre Rey

Déposé, le : 26 juillet 2013

www.hevs.ch

RÉSUMÉ

La Fondation Suisse pour les Téléthèses a pour but de mettre la technologie au service des personnes en situation de handicap. La fondation met donc à disposition de chaque personne défavorisée l'aide électronique (téléthèse) nécessaire à ses besoins.

La fondation (FST) m'a demandé dans le cadre de ce travail de Bachelor de réaliser une application Windows en utilisant le capteur Kinect. Cette application permet à l'utilisateur de contrôler son ordinateur à l'aide des membres de son corps. En effet, un paramétrage permet à l'utilisateur de choisir les actions souris à effectuer et de les associer aux parties du corps.

Pour réaliser ce travail, un apprentissage du Kinect et du kit de développement (SDK) mis à disposition par Microsoft a été nécessaire afin d'assimiler leur fonctionnement. Comme il y a très peu de documents et d'exemples à disposition, la recherche d'informations a été essentielle.

L'application développée permet de configurer les actions et les mouvements à effectuer par chaque utilisateur. La gestion de profil permet de récupérer sa propre configuration lors d'une utilisation ultérieure.

Mots-clés : Kinect, tracking, mouvement, souris, kit de développement (SDK)

AVANT-PROPOS

La Fédération Suisse pour les Téléthèses (FST) a développé une pré-analyse basée sur une interface Kinect en réalisant une petite application prenant en charge le capteur Kinect de Microsoft et permettant de déplacer un objet à l'écran à l'aide d'une main.

La FST avait mis à disposition de la Castalie de Monthey cette application afin d'effectuer des tests avec de réels utilisateurs, mais ceux-ci ne se sont pas révélés très concluants.

Dans le cadre de ce travail de Bachelor mandaté par Julien Torrent de la FST, ma mission consiste à reprendre ce qui a déjà été fait dans le but de le faire évoluer vers une application fonctionnelle sur le système d'exploitation Windows. Depuis la réalisation de l'application développée par la FST, Microsoft a mis à disposition des développeurs un SDK pour la Kinect afin de faciliter le développement d'applications pour Windows.

Dans un premier temps, une analyse de l'application ainsi que du SDK Kinect de Microsoft a été effectuée afin de déterminer quelles sont les possibilités offertes. Puis, l'objectif du projet « Ki-Mouse » consiste à élaborer une nouvelle application permettant à l'utilisateur d'effectuer toutes les actions souris à l'aide des membres de son corps et cela dans n'importe quelle application fonctionnant sur le système d'exploitation Windows.

Ce travail prend place dans la volonté de la FST de mettre à disposition, dès la fin de ce projet, l'application développée en test à la Castalie à Monthey.

La démarche adoptée pour réaliser ce travail était constituée de trois phases distinctes. La première comprenait la partie d'analyse de l'existant et des nouvelles possibilités offertes par les outils mis à disposition par Microsoft. La seconde phase consistait en l'apprentissage du capteur Kinect et de ses outils. La dernière phase, la plus importante, était le développement de l'application.

Au travers de ce travail, j'ai été confronté à de nombreuses difficultés. En effet, j'étais plongé dans l'inconnu total, car je devais me familiariser avec un nouveau domaine et ses spécificités, à savoir le capteur Kinect et son SDK. De plus, la documentation quasiment inexistante et le peu d'exemples à disposition n'ont pas facilité mon travail.

Ce rapport donne toutes les informations principales sur les possibilités offertes par le SDK Kinect v1.7 de Microsoft. De plus, les étapes importantes du développement de l'application sont expliquées dans le présent document.

Romain Paccolat

REMERCIEMENTS

Je tiens à remercier toutes les personnes qui m'ont soutenu et aidé tout au long de ce travail de Bachelor :

M. Jean-Pierre Rey, professeur à la HES-SO Valais, pour m'avoir soutenu et conseillé lorsque j'en avais besoin.

M. Julien Torrent, responsable R&D et conseiller à la Fondation Suisse pour les Téléthèses, pour sa disponibilité et ses conseils sur l'utilisation de l'application.

Mme Pauline Pillet, Mme Francine Paccolat, Mme Séverine Paccolat, ayant participé à la relecture du rapport final.

TABLES DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS

Fondation Suisse pour les Téléthèses (FST). La FST a pour but de mettre la technologie au service des personnes en situation de handicap.

Microsoft a mis à disposition des développeurs un kit de développement (SDK) spécialement conçu pour le Kinect Windows.

Windows Presentation Foundation (WPF) est un système de présentation pour construire des applications client Windows.

Natural User Interface ou interface utilisateur naturelle (NUI). Les NUI sont faites pour rendre l'informatique encore plus accessible et utilisable dans les tâches quotidiennes.

Infrarouge (IR). L'IR est un « rayonnement électromagnétique d'une longueur d'onde comprise entre 0.8 micromètre (lumière rouge) et 1 mm ». p. 572 (Larousse, 2012).

Une Application Programming Interface (API). Une API permet à son utilisateur d'interagir avec un logiciel en masquant les détails de son fonctionnement.

Travail de Bachelor (TB). Le TB est le travail effectué par chaque étudiant pour terminer ses trois années d'études à la HES-SO Valais.

INTRODUCTION

Contexte du projet

Le travail de Bachelor marque la fin de trois années passées à la HES-SO Valais à Sierre et a pour but de mettre en relation les connaissances théoriques et pratiques acquises tout au long de la formation. De plus, il permet de mettre en œuvre les savoirs professionnels développés dans un projet réel. Après avoir donné mon ordre de préférence parmi les sujets mis à disposition, je me suis vu attribuer un travail sur le développement d'une application Windows en utilisant le capteur Kinect de Microsoft pour le compte de la FST. Ce projet a rapidement attiré mon attention, car il me permettait de relever un nouveau défi dans un domaine inconnu.

Fondation Suisse pour les Téléthèses

La Fondation Suisse pour les Téléthèses (FST) (2011), créée en 1982 par Jean-Claude Gabus, a pour but de mettre la technologie au service des personnes en situation de handicap. La fondation met à disposition de chaque personne défavorisée l'aide électronique (téléthèse) nécessaire à ses besoins.

La FST croit aux ressources cachées de chaque être humain et met tout en œuvre pour l'aider à gagner en autonomie. Elle cultive l'interdisciplinarité, la transparence et la responsabilité individuelle.

Imagination et innovation dans les techniques, qualité dans l'exécution et rigueur dans la gestion, sont les missions de la FST. Dans son activité quotidienne, la FST touche une large palette d'individus : handicapé(e)s physiques, polyhandicapé(e)s ou handicapé(e)s mentaux.

La FST est le partenaire officiel de l'Office fédéral des assurances sociales (OFAS). Ses prestations incluent l'information, la formation, le suivi et toutes activités liées à la maintenance des technologies utilisées.

Description du travail

Le but de ce travail est de créer une application Windows fonctionnant avec le capteur Kinect. Celle-ci permettra à ses utilisateurs de prendre le contrôle de la souris et d'effectuer

toutes les actions qui s'y rapportent à savoir clic gauche, double clic, clic droit, drag & drop avec les membres de leurs corps. De plus, l'application comporte un mode collaboratif à savoir deux utilisateurs simultanément. L'application offre également la possibilité de configurer le membre du corps qui effectuera l'action. Les configurations peuvent être sauvées sous forme de profil et rechargées lors de l'utilisation ultérieure de l'application. Afin de réaliser le projet et de déterminer comment utiliser correctement le SDK, une analyse des possibilités offertes ainsi qu'une analyse de l'existant ont été effectuées. Dans un deuxième temps, j'ai appris et testé le fonctionnement du Kinect et de son SDK par moi-même au travers d'un livre. À la suite de ces deux étapes, j'ai réalisé l'application Windows en utilisant les connaissances que j'ai acquises.

Délivrables du projet

Au terme de ce travail, deux délivrables ont été fournis à savoir l'application Windows et le rapport final.

Le code source de l'application est mis à disposition de la FST afin qu'elle puisse l'améliorer et aussi la proposer en test à la Castalie dès la fin de ce travail.

1 PRÉSENTATION DE L'ENVIRONNEMENT KINECT

Selon Jana (2012), Kinect était connu initialement sous le nom de code "Project Natal". Il s'agit d'un dispositif de détection de mouvement qui a été initialement développé pour la console de jeu Xbox 360. L'un des facteurs distinctifs qui permet à cet appareil de se démarquer des autres appareils de ce genre est le dispositif permettant de détecter la position, le mouvement ainsi que la voix. En effet, le Kinect ne possède pas de commande manuelle, mais un détecteur de mouvement.

Kinect fournit une interface utilisateur naturelle (NUI). Les gens utilisent de plus en plus les gestes et la parole pour interagir avec leur ordinateur et ses périphériques. Ces NUI sont faites pour rendre l'informatique encore plus accessible et utilisable dans les tâches quotidiennes. Kinect a apporté une nouvelle révolution dans le monde du jeu, et il a complètement changé la perception d'un dispositif de jeux. Depuis sa création, il a établi plusieurs records dans le domaine du matériel de jeux, dont le record du monde du "dispositif électronique le plus rapidement vendu". L'un des principaux arguments de vente du Kinect était l'idée du « contrôle mains-libres », qui a attiré l'attention des joueurs et amateurs de technologie semblable et catapulté le dispositif dans la célébrité instantanée.

Le capteur Kinect ne se limite plus aux jeux. Le Kinect pour Windows est un capteur spécialement conçu pour PC et permet aux développeurs d'écrire leur propre code et ainsi de développer des applications de la vie réelle avec des gestes humains et les mouvements du corps. Avec le lancement du Kinect pour les périphériques Windows, l'intérêt pour le développement de logiciels de détection de mouvement a atteint un nouveau sommet.

Étant donné que le Kinect est encore relativement nouveau sur le marché, les ressources pour apprendre et pour développer des applications avec ce produit sont rares.

Avant d'entrer dans le processus de développement, nous avons besoin d'une bonne compréhension du dispositif et il est très important de connaitre les composantes et comment interagir avec.

1.1 Présentation du capteur Kinect pour Windows

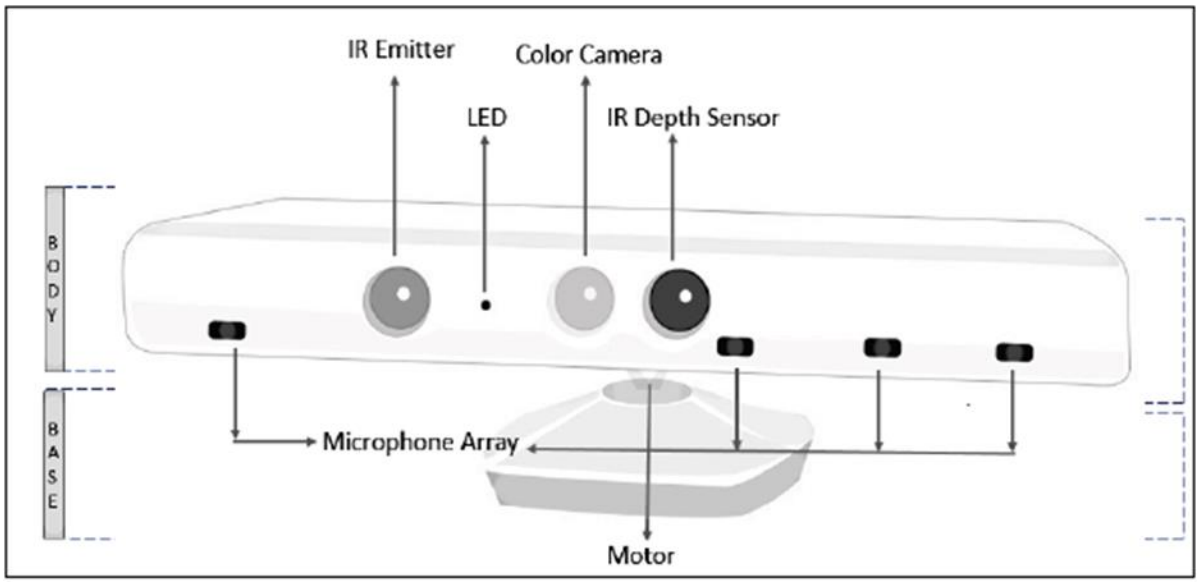
Selon Jana (2012), le Kinect est un périphérique horizontal avec des capteurs de profondeur, une caméra couleur et un ensemble de microphones. Tous ces éléments sont fixés à l'intérieur d'une petite boîte plate. Un petit moteur permet l'inclinaison horizontale du capteur. Le capteur Kinect comprend les éléments ci-dessous qui sont développés individuellement dans les pages suivantes :

- Caméra couleur
- Émetteur infrarouge
- Capteur de profondeur infrarouge (IR)
- Moteur d'inclinaison
- Micro
- LED

Outre les composants précédemment mentionnés, le dispositif comporte également une alimentation externe ainsi qu'un adaptateur USB qui permet de le connecter à l'ordinateur.

La figure 1 montre les différents composants d'un capteur Kinect:

Figure 1 - Composants d'un capteur Kinect



Source : Kinect for Windows SDK Programming Guide, p. 9

À l'intérieur du capteur

Selon Jana (2012), de l'extérieur, le capteur Kinect semble être un boîtier en plastique avec trois caméras visibles, mais il comporte des éléments très sophistiqués, des circuits et des algorithmes embarqués.

La figure 2 montre une vue de face de l'intérieur d'un capteur Kinect. Nous pouvons y voir (de gauche à droite) : son émetteur, sa caméra couleur IR et son capteur IR de profondeur.

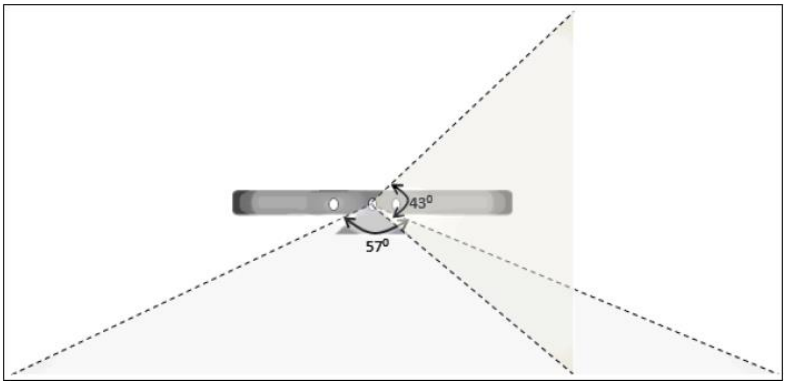
Figure 2 - Vue de face de l'intérieur du capteur Kinect

Source : Kinect for Windows SDK Programming Guide, p. 9

La caméra couleur

Selon Jana (2012), sa fonction est de détecter les couleurs rouge, bleu et vert de la source. Le flux de données renvoyées par la caméra est une succession d'images fixes. Il prend en charge une vitesse de 30 images par seconde (FPS) à une résolution de 640 x 480 pixels, et une résolution maximale de 1280 x 960 pixels jusqu'à 12 images par seconde. La valeur du nombre d'images par seconde peut varier en fonction de la résolution choisie par l'utilisateur. Le champ visible pour les caméras du capteur Kinect est de 43 degrés à la verticale par 57 degrés à l'horizontale.

Figure 3 - Champ visible pour la caméra

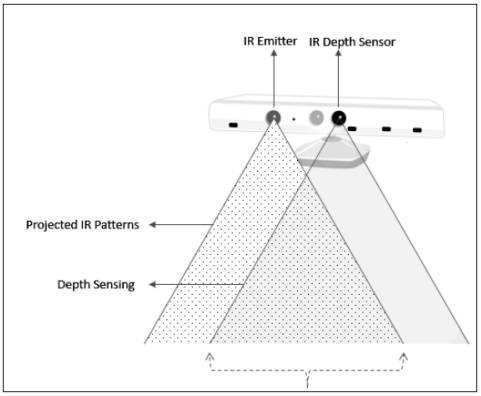


Source : Kinect for Windows SDK Programming Guide, p. 10

Émetteur infrarouge et capteur de profondeur IR

Selon Jana (2012), les capteurs de profondeur du Kinect sont constitués d'un émetteur infrarouge et d'un capteur de profondeur IR qui travaillent tous les deux ensemble. L'émetteur IR peut ressembler à un appareil photo de l'extérieur, mais c'est un projecteur infrarouge qui émet en permanence de la lumière infrarouge sur tous les objets qui se trouvent devant lui. Ces points sont invisibles pour l'être humain, mais il est possible de saisir leurs informations grâce à la profondeur IR. La lumière projetée sous forme de pointillés reflète sur les objets des informations que le capteur de profondeur IR lit. Ces dernières sont converties en des informations de profondeur. Cette opération permet de mesurer la distance entre le capteur et l'objet à partir du point où l'IR a été lu.

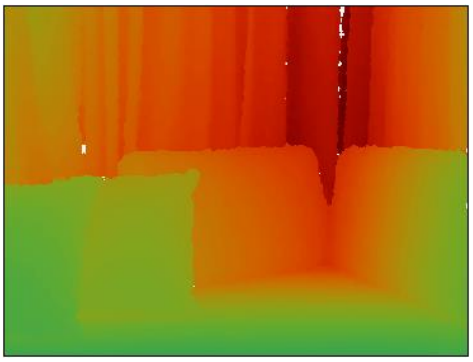
Figure 4 - Détection du capteur de profondeur



Source : Kinect for Windows SDK Programming Guide, p. 11

Le flux de données de profondeur supporte une résolution de 640 x 480 pixels, 320 x 240 pixels, et 80 x 60 pixels, et la gamme visible par le capteur reste la même que la caméra couleur. La figure 5 montre ce qui est capturé à partir de la profondeur du flux d'images:

Figure 5 - Profondeur du flux d'images

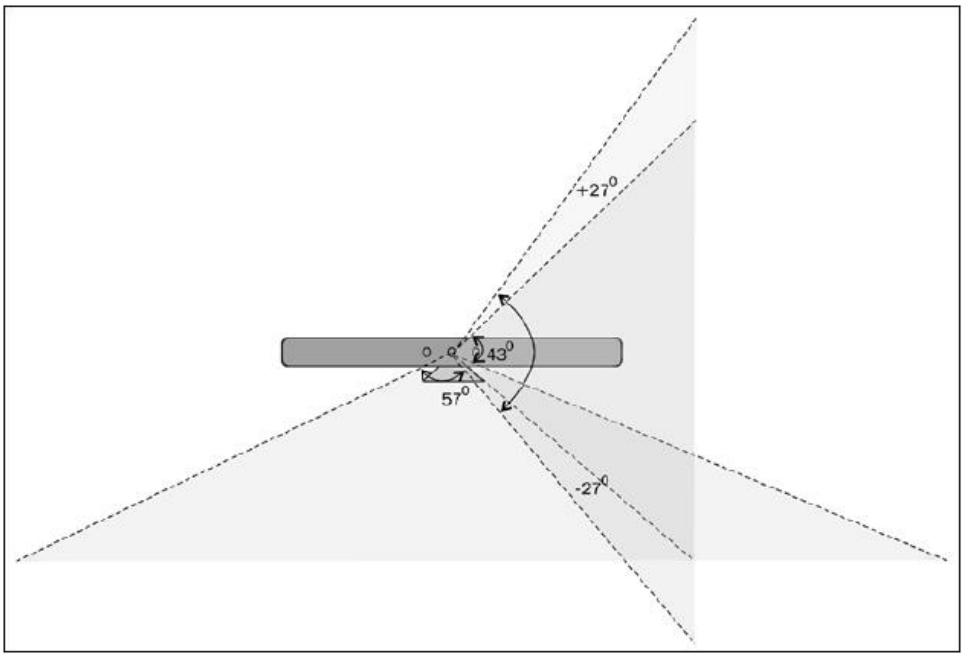


Source : Kinect for Windows SDK Programming Guide, p. 12

Moteur d'inclinaison

Selon Jana (2012), la base et le corps du capteur sont reliés par un petit moteur qui est utilisé pour changer l'angle du capteur. Il permet d'incliner verticalement jusqu'à un maximum de -27 ou de + 27 degrés.

Figure 6 - Angle d'inclinaison du capteur



Source : Kinect for Windows SDK Programming Guide, p. 14

Le micro

Selon Jana (2012), le capteur Kinect est équipé de quatre microphones qui sont disposés à des endroits différents dans un ordre linéaire (trois d'entre eux sont répartis sur le côté droit et l'autre est placé sur le côté gauche, comme le montre l'image ci-dessous).

Figure 7 - Microphones du capteur

Source : Kinect for Windows SDK Programming Guide, p. 14

LED

Selon Jana(2012), un voyant est placé entre la caméra et le projecteur IR. Il est utilisé pour indiquer l'état de l'appareil Kinect. La couleur verte de la LED indique que le Kinect et les pilotes de périphériques sont chargés correctement. Si vous branchez le Kinect à un ordinateur, la LED va devenir verte une fois que votre système aura détecté le périphérique. Cependant, pour obtenir toutes les fonctionnalités du capteur, il faut le brancher à une source d'alimentation externe.

1.2 Différence entre le capteur Kinect Windows et Xbox 360

Selon Jana (2012), le capteur Kinect pour Windows a été testé et est supporté par Windows avec des fonctionnalités telles que « Near mode », contrôle du suivi du squelette, amélioration de l'interface de programmation (API), amélioration de la prise en charge de l'USB sur une gamme d'ordinateurs Windows.

Le capteur a été spécialement conçu pour être utilisé avec les ordinateurs et est équipé d'un câble USB raccourci pour assurer une compatibilité maximum avec une large gamme d'ordinateurs.

Le Kinect pour Xbox 360 a été fabriqué, conçu et testé pour une utilisation avec la Xbox 360 uniquement. Il n'est pas fiable et n'est pas certifiée pour une utilisation commerciale.

Le prix du Kinect Xbox 360 est d'environ 95 CHF alors que celui conçu pour Windows est d'environ 235 CHF.

Le « Near mode » ou « mode de près » est une nouveauté sur le Kinect pour Windows. Cette fonction du Kinect pour Windows permet à la caméra de voir des objets situés à 40 centimètres devant le capteur tout en conservant sa précision.

Le Kinect pour Windows offre la même qualité de suivi du squelette et la même reconnaissance que le Kinect pour Xbox 360.

2 ÉTAT DE L'ART

La première partie de mon travail était d'analyser l'existant fourni par la FST et le SDK pour Kinect afin de faire ressortir les points forts et les points faibles entre ces deux choses à l'aide d'un tableau comparatif pour déterminer sur quelle voie continuer.

2.1 Analyse de l'existant

Cette analyse s'est basée sur un rapport réalisé par Raphaël Guye dans le cadre d'un projet de Master à la HES-SO. Ce projet se prénomme GMouse.

Dans cette pré-étude (Guye, 2011), le Kinect utilisé était celui adapté à la console de jeu Xbox 360. À l'heure actuelle, Microsoft propose un Kinect adapté pour des applications Windows et compatible avec son SDK.

Travail réalisé dans la pré-étude

Une application *kiMouseController* permettant de piloter un système d'exploitation par le geste a été développée durant le travail cité. À cette période, il y avait plusieurs librairies disponibles sur internet comme OpenKinect ou encore Open NI + Nite. Cependant, il n'y avait pas encore de SDK officiel pour le Kinect mais uniquement une version beta disponible sur invitation.

Le choix s'était porté sur l'utilisation d'Open NI avec le middleware NITE car le suivi de la main était déjà implémenté. Open NI ne permettait pas par exemple d'utiliser pour la fonction « clic » la manipulation de serrer le poing puis d'ouvrir à nouveau la main. Selon le rapport GMouse (Guye, 2011), les possibilités offertes par la librairie étaient très limitées.

Durant ce travail, une librairie nommée kiTrackerNI a été développée pour permettre d'utiliser les fonctionnalités de base du Kinect. Toutes les fonctionnalités de base comme la détection de la main, le suivi de la main, la détection du clic et d'autres ont dû être développées.

kiMouseControl

Connaître la position de la main de l'utilisateur à tout moment.

Simulation d'un clic

La détection du clic en serrant le poing puis en ouvrant à nouveau la main n'était pas possible avec l'api OpenNI.

La stratégie choisie est la suivante : avancer la main en direction du Kinect puis l'éloigner à nouveau, comme si nous appuyions sur un bouton. À savoir la détection par « push ». Une fois que les coordonnées de la main sont connues, par le tracking, nous connaissons les coordonnées en Z de la main à tout moment. Dès lors il est facile de détecter si l'utilisateur a avancé sa main ou l'a reculée.

Conception : points importants

Une librairie personnelle a été créée pour le projet « kiTrackerNI » en utilisant les listeners de JAVA. Voici ce que nous pouvons ressortir de cette librairie :

- Il y a la possibilité de définir une zone de travail.
- La fonction kiNoiseTracking n'est pas utilisable en situation réelle.
- La librairie et les méthodes développées pour la détection du nez ne sont pas utilisables.
- Si l'on approche sa main trop brusquement du capteur, le logiciel prend cela pour une simulation de clic.
- La librairie ne gère pas le zoom à deux doigts, le scrolling à deux mains, le clic droit.
- Lorsqu'un utilisateur est à 20 cm ou 2 m du capteur, la détection n'est pas fiable et beaucoup d'erreurs sont retournées.

Le logiciel développé n'est pas facile à prendre en main pour un nouvel utilisateur. De plus, il n'est pas très précis et il est souvent difficile de faire des clics sur la cible désirée.

2.2 Compte rendu de la Castalie sur le projet GMouse

À la suite du travail réalisé par Raphaël Guye, l'application a été installée en test à la Castalie de Monthey et un compte rendu a été rédigé par M. Alain Saudan responsable Alphalogic dans l'établissement.

Selon Alain Saudan (2011) « L'évaluation pourrait être plus efficiente si un outil de feedback des réglages était intégré. On n'était pas sûr dans certain cas, d'avoir les réglages optimums, d'où certains petits dysfonctionnements qui auraient pu être évités. ». (p. 1)

En prenant en compte cette remarque de la part de la Castalie, il faudrait un document d'aide ou une aide intégrée à l'application qui permet de définir si oui ou non les réglages sont corrects.

> La prise en main est très bonne comparée aux systèmes ultérieurement testés. Toutefois le calibrage sur une main verticale ouverte, avec oscillation de celle-ci, n'est pas un mouvement évident pour nos pensionnaires. Il serait donc intéressant d'étudier d'autres pistes pour le calibrage, comme un placement de la main sur une zone définie. (Saudan, 2011, p. 2)

En prenant en compte cette remarque, il sera important de définir avec la Castalie quels sont les gestes les plus appropriés aux différentes actions (clic, double clic,…) pour les utilisateurs.

Selon Alain Saudan (2011) « Il serait intéressant d'avoir un système qui garde la calibration mémorisée, même si la main a quitté le champ de vision. ». (p. 2)

Cette remarque est une proposition faite par la Castalie qui mentionne qu'il est important de ne pas avoir à calibrer la Kinect quand un membre du corps quitte et revient dans le champ de vision.

Selon Alain Saudan (2011) « Une détection au niveau de la tête serait souhaitée. Une détection de deux mains serait également intéressante. ». (p. 2)

Selon Alain Saudan (2011), il serait également très intéressant d'utiliser d'autres membres du corps comme la tête, les deux mains, un pied, etc.

À la suite de la lecture de ce compte rendu, j'ai également relevé qu'il est important d'avoir un curseur bien visible par les pensionnaires afin de le situer facilement sur l'écran. Le curseur original est trop petit et difficilement reconnaissable. Pour confirmer cette hypothèse, je me suis renseigné auprès de M. Torrent qui m'a confirmé l'information.

2.3 Analyse du Kinect SDK pour Windows de Microsoft

Selon Microsoft (2013), le SDK Kinect pour Windows prend en charge jusqu'à quatre capteurs sur un ordinateur, et peut être utilisé sur n'importe quelle machine virtuelle dont le système d'exploitation natif prend en charge l'exécution de Windows.

La dernière mise à jour du SDK Kinect pour Windows ajoute de nouvelles interactions comprenant des gestes naturels tels que le « grip » et « push », et comprend un nouvel outil prénommé Kinect Fusion qui permet de créer une reconstruction 3D des personnes et des objets en temps réel.

Le SDK offre une multitude de nouveaux outils et de fonctionnalités pour permettre aux développeurs de simplifier le développement d'applications et de créer des applications plus intelligentes. En effet les applications utilisent la voix humaine et les gestes de la même manière que les gens communiquent les uns avec les autres. De plus, il est entièrement compatible avec toutes les versions commerciales précédentes.

Kinect interactions

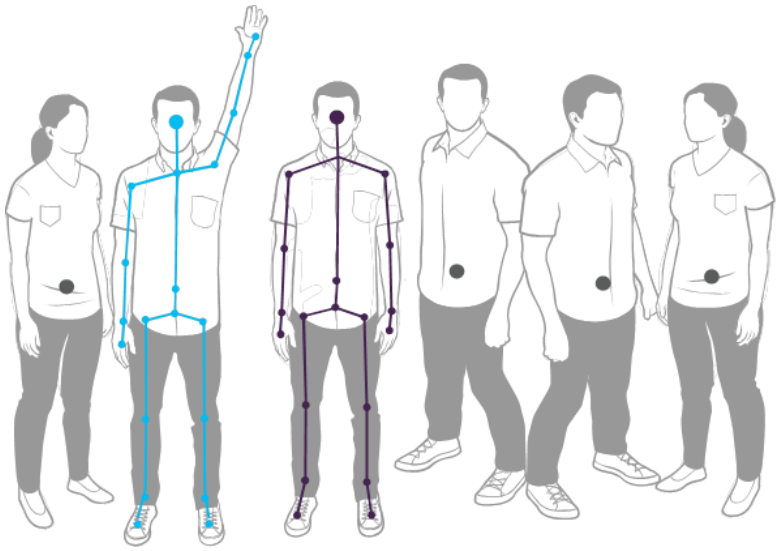
Une combinaison de nouveaux contrôles permet aux utilisateurs d'interagir plus facilement grâce à la technologie informatique naturelle. Des nouveaux contrôles WPF sont apparus dans cette version 1.7 : « Push », « grip ».

Le capteur Kinect pour Windows peut reconnaître jusqu'à quatre pointeurs de la main. Cela permet à deux personnes d'interagir avec les deux mains simultanément et aux développeurs de créer des interactions plus complexes, y compris la capacité de «zoomer».

Suivi de squelette (Skeletal Tracking)

La caméra infrarouge peut reconnaitre jusqu'à six utilisateurs dans le champ de vision du capteur. Parmi ceux-ci, jusqu'à deux utilisateurs peuvent être suivi en détail.

Figure 8 - Le Kinect peut reconnaitre six personnes et en suivre deux

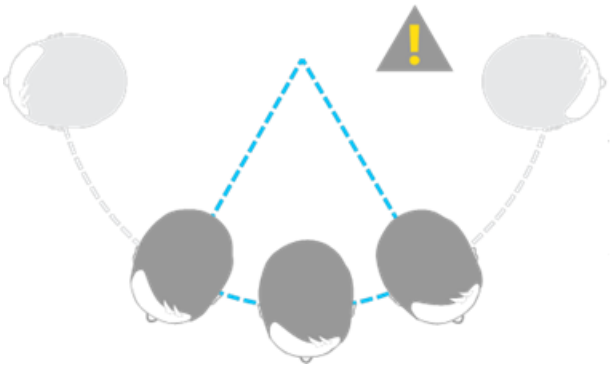


Source : Human Interface Guidelines v1.7, p 131

Le suivi du squelette est optimisé pour reconnaître un utilisateur assis ou debout, et face au Kinect.

Pour être reconnus, les utilisateurs ont simplement besoin d'être en face du capteur, en s'assurant que le capteur peut voir la tête et le haut du corps. Pas de pose spécifique et aucune action d'étalonnage ne sont nécessaires pour qu'un utilisateur soit suivi.

Figure 9 - Skeleton tracking est conçu pour reconnaitre les utilisateurs face à la caméra



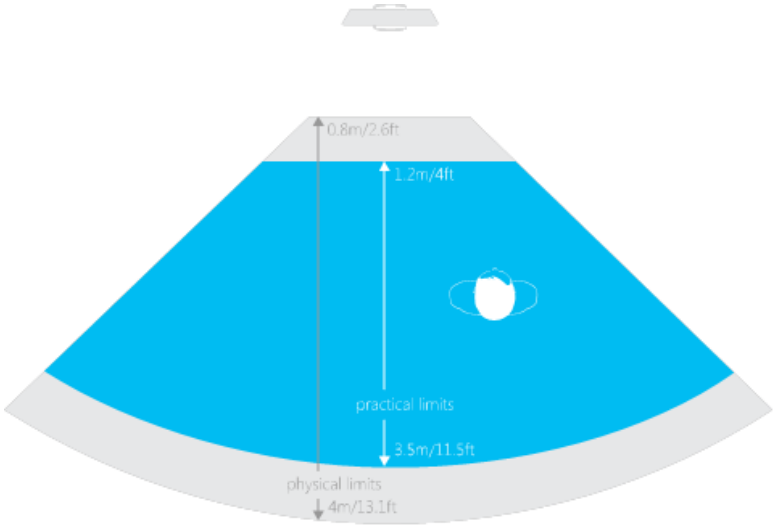
Source : http://msdn.microsoft.com/en-us/library/hh973074.aspx

Champ vision

Le champ de vision de la caméra est déterminé par des paramètres infrarouges et est défini par « DepthRange Enumeration ».

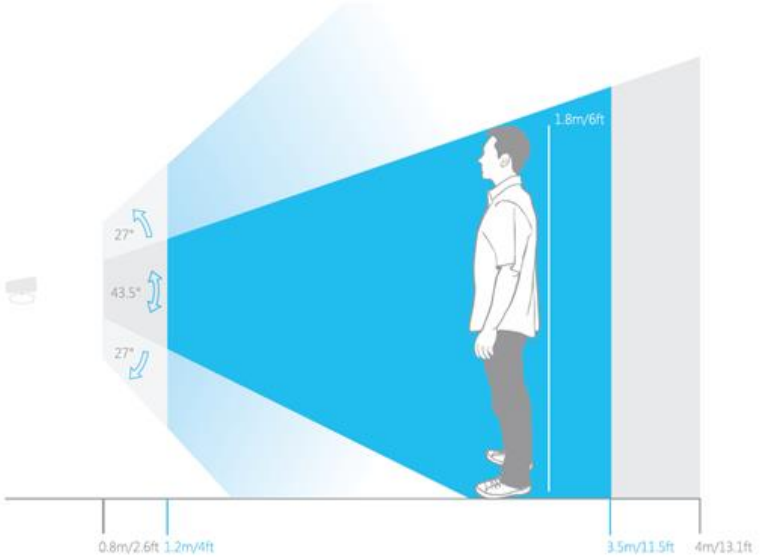
En mode par défaut, le Kinect peut voir des utilisateurs debout à une distance allant de 0.8 mètres à 4.0 mètres depuis le capteur. Cependant, il est recommandé d'être à une distance allant de 1.2 mètres à 3.5 mètres.

Figure 10 - Champ de vision horizontal par défaut du capteur Kinect



Source : http://msdn.microsoft.com/en-us/library/hh973074.aspx

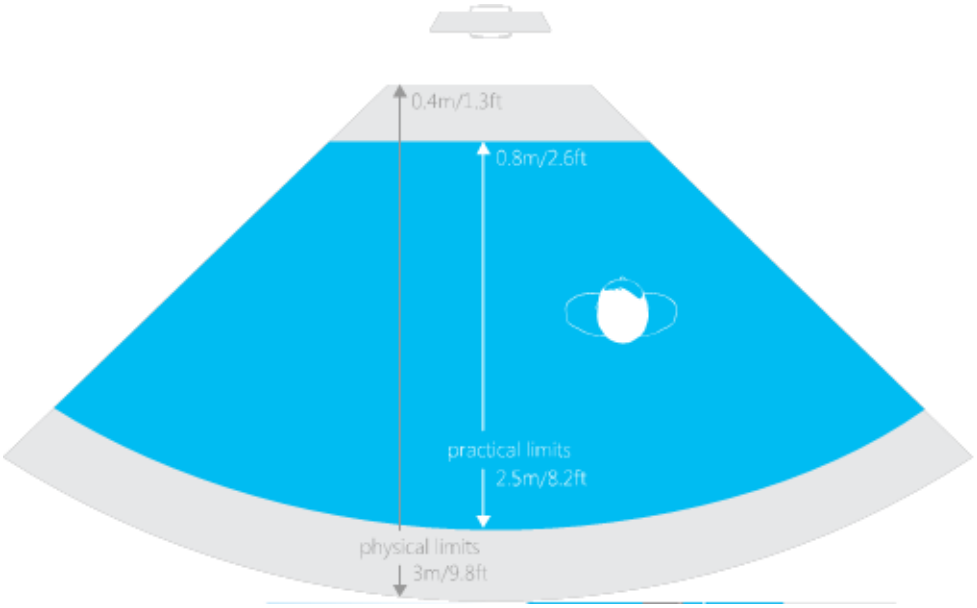
Figure 11 - Champ de vision vertical par défaut du capteur Kinect



Source : http://msdn.microsoft.com/en-us/library/hh973074.aspx

En « Near Range Mode », la Kinect est capable de voir les gens debout entre 0.4 mètres et 3.0 mètres. En pratique afin d'obtenir le meilleur résultat, il est conseillé de se situer entre 0.8 mètres et 2.5 mètres.

Figure 12 - Champ de vision du capteur Kinect en Near mode

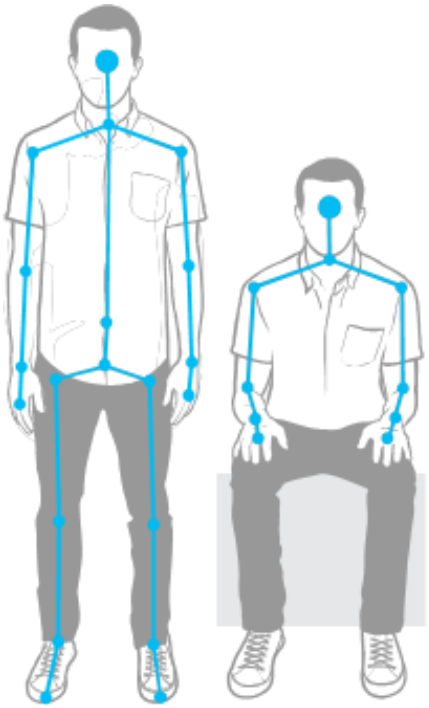


Source : http://msdn.microsoft.com/en-us/library/hh973074.aspx

L'émetteur infrarouge du capteur projette un rayon de lumière infrarouge. Ce rayon de lumière est utilisé pour calculer la profondeur des personnes dans le champ de vision et ainsi permettre la reconnaissance des différentes personnes et différentes parties du corps. Si vous utilisez plusieurs capteurs Kinect pour améliorer la zone cible, vous remarquerez peut-être une réduction de l'exactitude et de la précision du suivi du squelette due à une interférence avec les autres rayons de lumière infrarouge. Pour réduire le risque d'interférences, il est recommandé de ne pas avoir plus d'un capteur Kinect (ou source de lumière infrarouge) pointant vers un même champ de vision où le suivi d'un squelette est en cours.

Mode de suivi

Figure 13 - Champ de vision du capteur Kinect en Near mode



Source : http://msdn.microsoft.com/en-us/library/hh973077.aspx

Selon Microsoft (2013), le mode de suivi assis est conçu pour suivre les gens qui sont assis sur une chaise ou un canapé, ou dont la partie inférieure du corps n'est pas entièrement visible sur le capteur. Le mode d'alignement par défaut, en revanche, est optimisé pour reconnaître et suivre les personnes qui sont debout et entièrement visibles sur le capteur.

Seul un mode de suivi peut être utilisé à la fois. En d'autres termes, vous ne pouvez pas suivre un utilisateur en mode assis et un autre utilisateur en mode par défaut en utilisant un seul capteur.

Human interface Guidelines v 1.7

Selon Microsoft (2013), le Human Interface Guidelines permet d'obtenir un grand nombre d'informations sur le fonctionnement du capteur ainsi que sur ses performances. De plus, il peut être utilisé comme une aide lors du développement. À la suite de la lecture de ce document, j'ai ressorti tous les éléments importants de mon point de vue.

Comment le capteur Kinect pour Windows, le SDK, et le Toolkit fonctionnent ensemble

Capteur Kinect pour Windows

Il fournit une image couleur depuis la caméra RGB, des données audio depuis le microphone.

Kinect pour Windows SDK

Il traite les données brutes du capteur pour fournir des informations telles que le suivi du squelette pour deux personnes, le suivi du joueur pour six personnes et la reconnaissance de la parole à partir de données audio pour une langue donnée.

Le Windows Developer Toolkit pour Kinect

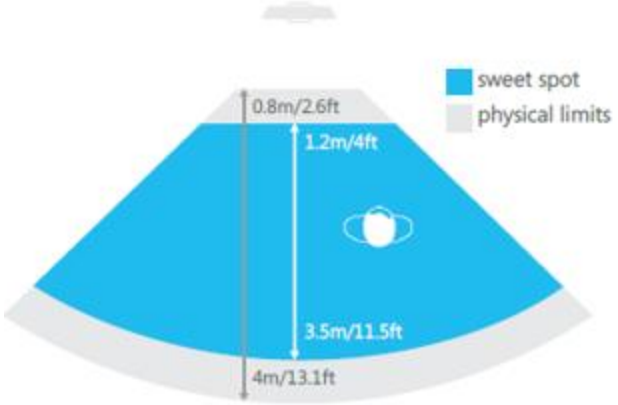
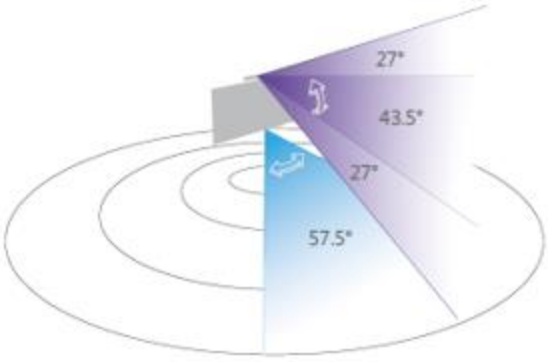
Cet outil fournit des exemples de codes qui démontrent comment utiliser les fonctionnalités du SDK, ainsi que des composants tels que le bouton Kinect ou encore le curseur Kinect, qui vous aident à construire des interfaces plus rapides.

Comment fonctionne le capteur Kinect ?

Le Kinect pour Windows est polyvalent. Six personnes peuvent être suivies, mais uniquement deux squelettes en détails. Le capteur est équipé d'une caméra RGB et d'un émetteur infrarouge qui permettent de mesurer la profondeur. Ces mesures sont effectuées en millimètres. On distingue deux zones :

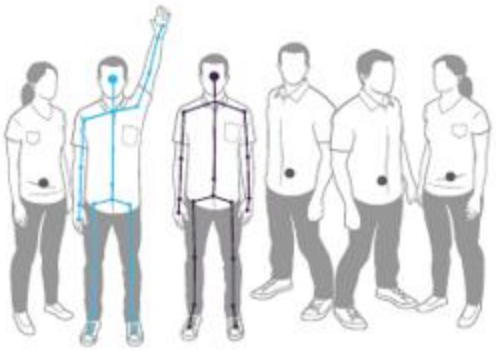
- Limites physiques : les capacités réelles du capteur et ce qu'il est capable de voir
- Sweet Spot : zone où les personnes auront une interaction maximale avec le capteur.

Tableau 1 - Comparaison limite physique et sweet pot

Gammes de profondeur : Near mode
Limites physiques : 0.4m à 3m Sweet spot : 0.8m à 2.5m 
Gammes de profondeur : Default mode
Limites physiques : 0.8m à 4m Sweet spot : 1.2m à 3.5m 
Angle de vision (profondeur et RGB)
Horizontal : 57.5 degrés Vertical : 43.5 degrés, avec une inclinaison de -27 à +27 degrés 

Source :Human Interface Guidelines v1.7, p. 7

Tableau 2 - Comparaison des différents modes de suivi

Suivi squelette	
Kinect pour Windows peut suivre jusqu'à six personnes. Il peut suivre le squelette en détail d'uniquement deux personnes.	
Mode debout (complet)	
Kinect pour Windows peut suivre les squelettes en mode debout avec 20 points de jointures.	
Mode assis	
Kinect pour Windows peut également suivre les squelettes assis avec seulement 10 points de jointures.	

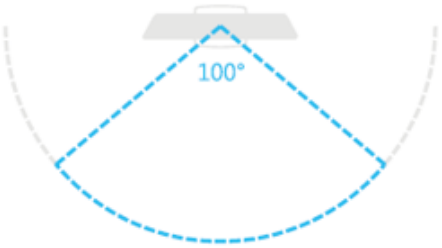
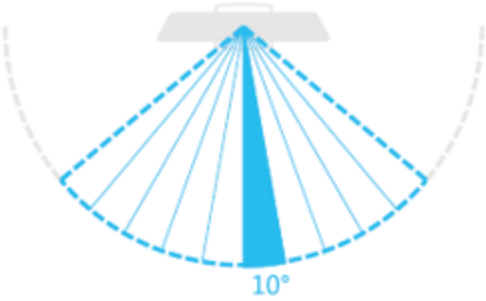
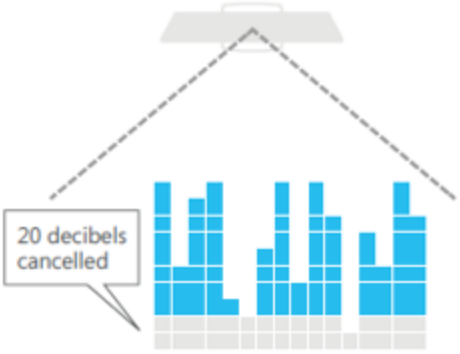
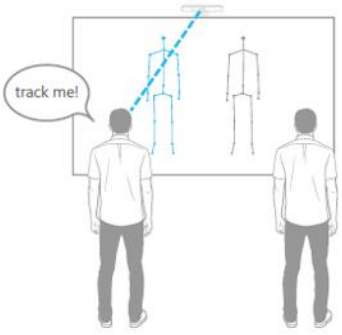
Source :Human Interface Guidelines v1.7, p. 8

Qu'est-ce que le Kinect peut entendre ?

Le capteur comporte quatre microphones qui permettent à l'application de répondre aux instructions données, en plus de répondre aux mouvements de l'utilisateur.

Par défaut la Kinect prend en compte le son le plus fort.

Tableau 3 – Qu'est-ce que le Kinect peut entendre ?

Entrée audio
Le capteur Kinect pour Windows détecte l'entrée audio de + à - 50 degrés en avant du capteur. 
Étendue des microphones
L'étendue des microphones peut être faite par incréments de 10 degrés jusqu'à 100 degrés. Cela peut être utilisé pour être plus précis sur la direction des sons, quand une personne parle. Cependant, cela ne fera pas disparaître complètement les autres bruits ambiants. 
Seuil sonore
Les microphones peuvent annuler 20dB (décibels) de bruit ambiant, ce qui améliore la qualité audio. C'est à peu près le niveau sonore d'un chuchotement. Le son venant de derrière le capteur reçoit une suppression de 6dB supplémentaire. 
Direction des microphones
Il est possible de diriger l'étendue des microphones. Par exemple, vers un emplacement défini, ou sur un squelette suivi. 

Source capturée

Par défaut, le capteur Kinect capture le son le plus fort.

Source :Human Interface Guidelines v1.7, p. 10

Lors de la réalisation de l'application, il est important de prendre en compte certains points :

- Il faut être vigilant pour qu'aucune personne n'entre dans le champ d'utilisation du capteur.
- Lorsque l'on souhaite utiliser la voix, un environnement calme est nécessaire pour avoir des résultats fiables.
- Les utilisateurs qui sont plus âgés ou avec des facultés visuelles affaiblies peuvent avoir besoin de graphiques ou de polices plus grandes.
- L'idéal est que la lumière ou l'éclairage viennent de derrière le capteur.
- Des vêtements noirs, ainsi que des éléments réfléchissants, peuvent interférer avec la caméra infrarouge et faire perdre de la fiabilité au suivi du squelette.
- Le capteur doit être placé au-dessus ou au-dessous de l'écran.
- Pour avoir le meilleur résultat, la position idéale depuis le capteur est de, 1,5 à 2 mètres.

Les gestes innés

Les gestes les plus appropriés sont :

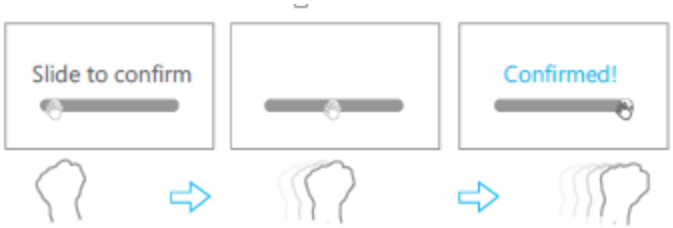
- Main ouverte pour déplacer le curseur.
- Fermer le poing pour prendre un élément.
- Avancer la main ouverte pour sélectionner.

Gestes à apprendre

- Une pose spécifique pour annuler une action par exemple
- Agiter la main pour confirmer

Geste dynamique

Figure 14 – Exemple de geste dynamique



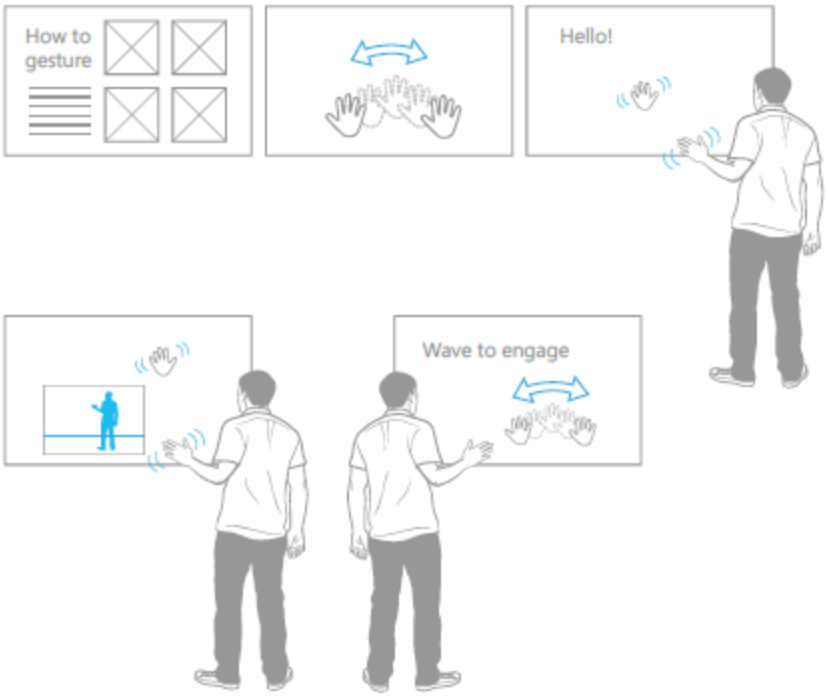
Source : Human Interface Guidelines v1.7, p. 24

Quelques conseils sont à prendre en compte :

- Il est important que l'application donne un feedback à l'utilisateur sous forme de message quand un geste est réussi.
- Il faut également définir un mouvement qui détermine que l'utilisateur commence à utiliser l'application comme agiter la main de gauche à droite.
- Il est nécessaire de donner aux utilisateurs des conseils clairs sur le geste exact dont on a besoin.
- Il est adéquat d'utiliser des mouvements logiques qui sont faciles à apprendre et à mémoriser.
- Il est plus facile d'utiliser les gestes similaires pour les actions de même type : déplacer à gauche et déplacer à droite.
- Il est préférable pour toutes les tâches essentielles d'utiliser une seule main.
- Pour des tâches comme le zoom et des tâches avancées, il est conseillé d'utiliser deux mains.

Méthode d'apprentissage des gestes

Figure 15 – Exemple des méthodes d'apprentissage des gestes



Source : Human Interface Guidelines v1.7, p. 42

- Tutoriel rapide : un écran d'accueil permet d'obtenir les indications nécessaires pour chaque mouvement.
- Indication du geste : tout au long de l'opération, le geste à effectuer est affiché.
- Image statique : une image statique montre le geste à effectuer.
- Une animation : une petite animation vidéo démontre le mouvement à effectuer.
- Un message ou une notification : un message textuel s'affiche pour donner des indications sur le mouvement à effectuer

Utilisation de la voix

Utilisable avec des phrases courtes :

- Page précédente, page suivante
- Mets dans le panier
- …

Sélection des utilisateurs à suivre

L'application peut prendre le contrôle du suivi des utilisateurs en activant AppChoosesSkeletons. Pour spécifier l'utilisateur ou les utilisateurs à suivre, il suffit d'utiliser la méthode SkeletonStream.ChooseSkeletons et passer l'ID de suivi d'un ou deux squelettes que l'on souhaite suivre (ou pas de paramètres si aucun squelette ne doit être suivi).

2.4 Comparaison des deux solutions

Tableau 4 – Comparaison de la pré-étude GMouse et du SDK Kinect

Quoi	GMouse	Microsoft SDK pour Kinect
Aide	Aucune aide intégrée	Possibilité d'ajouter de l'aide sous différentes formes : animation, tutoriel, image,…
Calibrage	Calibrage nécessaire lors de l'utilisation	Pas de calibrage nécessaire, il faut signaler au capteur que l'on souhaite l'utiliser à l'aide d'un geste défini.
Membres	Utilisation d'une seule main	Possibilité d'utiliser une ou deux mains, les jambes, les pieds => squelette. Pas de possibilité de contrôler le curseur avec la tête.
Librairie	Librairie **KiTrackerNI** développée pour le projet, mais aucune mise à jour depuis son développement	Toutes les fonctionnalités nécessaires sont dans le SDK et sont mises à jour à chaque nouvelle version du SDK.
Gestes	Problème de précision dans les gestes et conflit en cas de gestes trop brusques	Bonne précision et bonne détection des gestes.

Fonctionnalités	Pas de zoom, pas de scroll	Zoom et scroll disponibles
Distance	Problème à 20cm ou 2m du capteur	Dans la zone d'utilisation définie, aucun problème n'est à signaler
Précision	Pas très précis	Très précis dans la zone définie
Mode collaboratif	L'utilisation en mode collaboratif n'est pas précise et de nombreux conflits sont détectés	Le mode collaboratif est disponible et paramétrable.
Mise à jour et support	Aucune mise à jour régulière n'est effectuée et aucun support n'est disponible.	Des mises à jour régulières sont effectuées par Microsoft avec à chaque fois le support sur les nouvelles fonctionnalités.

Source : Données de l'auteur

2.5 Conclusion des recherches

À la suite de l'analyse de l'existant et du Kinect SDK pour Windows, j'ai rédigé un tableau comparatif (cf Tableau 4) en me basant sur les mêmes critères. En accord avec le client et grâce à cette comparaison, j'ai pu déterminer quelle était la meilleure voie à suivre pour le développement de l'application.

Il en est ressorti que la reprise de l'existant ne correspondait pas au souhait du client concernant l'utilisation de l'application. De plus, le SDK possède les mêmes fonctionnalités de base que le GMouse mais comporte de nombreuses possibilités supplémentaires. Le point le plus important est que le SDK est une solution stable, avec un support officiel et des mises à jour régulières de la part de Microsoft.

Le développement de l'application a été effectué en utilisant la version 1.7 du SDK Kinect pour Windows. Dans le chapitre suivant, nous allons voir comment mettre en place cet environnement et quels sont les outils nécessaires.

3 GESTION DE PROJET

3.1 Méthodologie

Afin d'être le plus flexible et de pouvoir répondre aux besoins et aux souhaits du client tout au long du développement, une gestion de projet Agile a été choisie. Avant de démarrer ce travail, un Product Backlog (cf. Annexe II) a été créé comme cahier des charges. Ce document contient toutes les fonctionnalités que va comporter le produit à réaliser. Ces fonctionnalités sont représentées sous forme de « User Stories »qui ont été définies selon les demandes du client concernant l'application. Puis, le client les a classées par priorité.

Les différentes fonctionnalités à développer ont été réparties sur la durée du travail de Bachelor à savoir 360h.

3.2 Planification

Le travail de Bachelor est réparti sur trois mois à savoir du 29 avril au 29 juillet. Les deux premiers mois, trois jours par semaine sont consacrés au travail de Bachelor. Depuis le 1er juillet, je travaille sur le TB à plein temps. J'ai choisi de répartir mon travail de la manière suivante. Pour débuter, j'ai consacré cinq semaines à découvrir et à m'approprier le Kinect afin de mettre en place mon environnement de travail et me familiariser avec le thème. Puis, sept semaines ont été consacrées à la réalisation de l'application. Pour finir, la dernière semaine a été utilisée pour la rédaction et la relecture du rapport ainsi qu'aux tests de l'application finale. Cependant, en parallèle des différentes phases, j'ai rédigé quelques parties de ce document.

3.3 Organisation des sprints

Pour la réalisation de ce travail, j'ai réparti les trois mois à disposition en quatre sprints distinctifs. Lorsqu'on utilise la méthode agile, le projet est composé d'une série d'itérations, d'une durée d'un mois ou moins, qui sont appelées des sprints. Dans chacun d'entre eux, les fonctionnalités ont été réparties en fonction de leur priorité et l'effort à fournir.

Le premier sprint a été consacré essentiellement à l'établissement du cahier des charges, la recherche, l'apprentissage et la mise en place de l'environnement de travail. Le deuxième sprint a été consacré à la réalisation de l'interface de l'application, le déplacement du curseur.

Le troisième sprint a été consacré à la détection distinctive des squelettes, à l'utilisation de l'application en mode collaboratif et aux liaisons des éléments graphiques entre eux comme les listes déroulantes. Le quatrième sprint a été consacré à l'attribution des actions aux membres du corps. Durant ce dernier sprint, je me suis également consacré à la finalisation du produit et aux tests de l'application ainsi qu'à la rédaction et la relecture du rapport final. Cette première répartition a évolué tout au long du projet en fonction des évènements.

Certaines pratiques utilisées lors d'une gestion de projet agile n'ont pas été effectuées du fait que j'ai été le seul à travailler sur le projet. J'entends par là par exemple : rétrospective.

La durée des sprints était variable étant donné que le temps à disposition n'était pas toujours le même. En effet, les neuf premières semaines consacrées au TB comportaient des heures de cours et des heures d'examen. Les sprints ont été planifiés de la manière suivante :

N° du sprint	Date de début	Date de fin	Nombre d'heures
0	29.04.2013	31.05.2013	110h
1	03.06.2013	21.06.2013	56h
2	24.06.2013	12.07.2013	103h
3	13.07.2013	25.07.2013	95h
Total			**364h**

3.4 Séances

Durant toute la durée du projet, des séances régulières ont été organisées avec le client M. Julien Torrent et dans la mesure du possible avec la participation de M. Jean-Pierre Rey, afin d'avoir un suivi régulier de l'avancement du travail. Celles-ci étaient en général fixées à la fin des sprints ou dès qu'une fonctionnalité essentielle était terminée, et cela en fonction des disponibilités des personnes concernées. Durant ces entretiens, un compte rendu de la séance contenant les éléments suivants a été rédigé :

- Retour sur les fonctionnalités qui devaient être développées
- Sprint review : présentation du travail effectué
- Sprint planning : présentation des tâches à réaliser lors du prochain sprint
- Date de la prochaine séance

Ces séances étaient bénéfiques, car elles me permettaient de prendre note des remarques du client et du professeur. À la suite de ces dernières, j'apportais les améliorations et j'adaptais l'application selon les besoins du client. Finalement, je corrigeais les éventuelles erreurs de l'application.

3.5 Écart de temps

Au terme de ce travail, le nombre d'heures réalisées ne diffère pas beaucoup de celui établi dans la planification initiale. Ce faible écart peut se justifier par le fait que l'apprentissage du SDK m'a demandé plus de temps que prévu. En effet, sa compréhension n'était pas aussi simple que ce que j'avais pensé.

Au final, une différence de quatre heures supplémentaires est à relever. La durée totale de ce projet se monte à 364 heures.

Le détail des heures réalisées se trouve en annexe de ce document.

4 APPLICATION KI-MOUSE

4.1 Environnement et outils de développement

Afin de pouvoir développer des applications Kinect pour Windows, un environnement spécifique est nécessaire. Le premier élément qui doit être installé sur l'ordinateur est une version de Microsoft Visual Studio 2010 express ou plus récente. Pour réaliser ce travail, j'ai utilisé la version Ultimate 2012 fournie sur le site MSDN de la HES-SO.

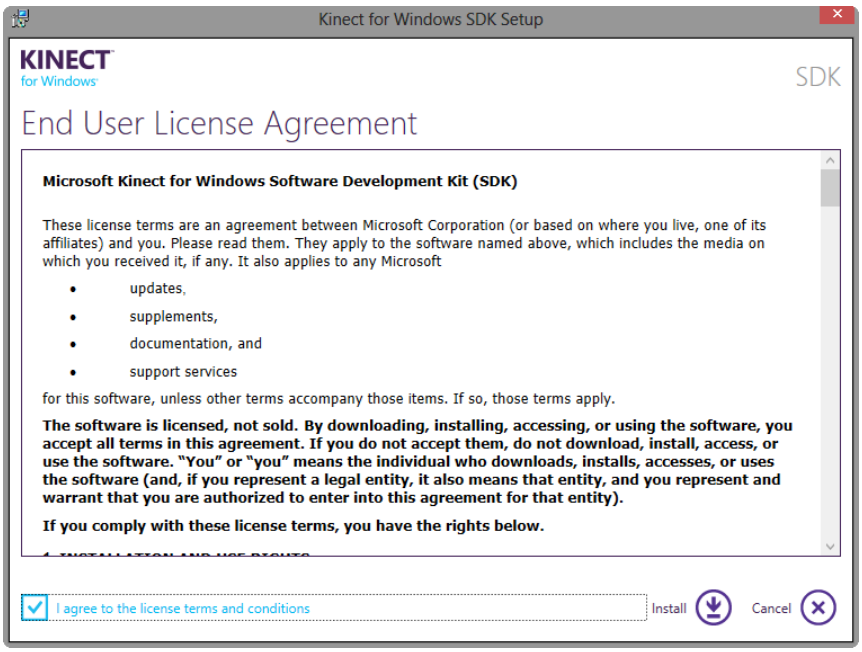
Le second élément est le SDK Kinect pour Windows accompagné du Toolkit. Ces deux installateurs sont disponibles à l'adresse suivante : http://www.microsoft.com/en-us/kinectforwindows/develop/developer-downloads.aspx

Le SDK est obligatoire pour pouvoir développer des applications pour Windows. Le Toolkit contient une série d'exemples d'applications ou d'utilitaires qu'il est possible de réaliser avec le SDK.

Il faut noter que les systèmes supportés par le SDK sont Windows 7 et Windows 8.

Installation du SDK

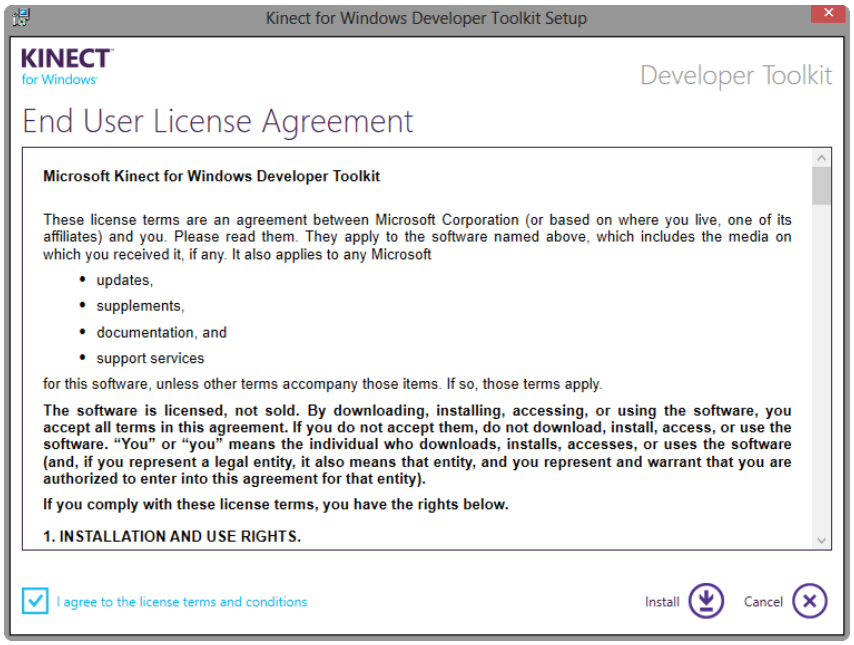
Figure 16 – Fenêtre d'installation du SDK



Source : Installateur du SDK de Microsoft

Installation du Toolkit

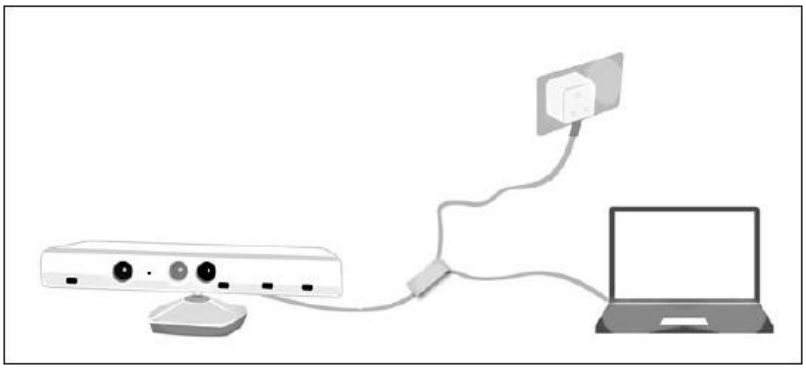
Figure 17 – Fenêtre d'installation du Toolkit



Source : Installateur du Toolkit de Microsoft

Après avoir installé l'environnement de développement, l'étape suivante consiste à brancher le périphérique Kinect à l'ordinateur. La première fois que l'on branche l'appareil au système, la LED du capteur Kinect est rouge, car le système démarre l'installation automatique des pilotes. L'emplacement par défaut des pilotes est %Program Files%\Microsoft Kinect Drivers\Drivers. La figure 18 montre comment le capteur doit être installé :

Figure 18 – Installation et branchement du capteur Kinect



Source : Kinect for Windows SDK Programming Guide, p. 29

4.2 Architectures et technologies

Les applications développées pour le Kinect sont des projets de type WPF application. Il y a la possibilité de choisir entre le langage C# ou le langage C++. Afin de pouvoir utiliser la librairie fournie par le SDK, et cela lors de la création de chaque projet, il faut ajouter une référence vers le fichier Microsoft.Kinect.dll. Il se trouve dans le répertoire du SDK, créé lors de l'installation, dans le dossier « Assemblies ». Celui-ci permet ensuite d'avoir accès à toutes les méthodes nécessaires au bon développement de l'application.

Les deux principaux éléments utilisés lorsque l'on développe une application sont l'objet KinectSensor et la classe Skeleton. Voici quelques explications sur ces éléments :

Le KinectSensor

L'objet KinectSensor se définit comme n'importe quel autre objet se référant à une classe. On définit cet objet de la manière suivante :

```
public partial class MainWindow : Window
{
KinectSensor sensor;
// remaining code goes here
}
```

Les objets de type KinectSensor ont besoin d'avoir une référence au dispositif Kinect qui est connecté au système afin de pouvoir être utilisé par l'application. Il n'est pas possible d'instancier l'objet KinectSensor car il ne possède pas de constructeur public. Cela est fait automatiquement par le SDK qui s'en occupe quand il détecte un périphérique Kinect connecté au système.

Activation des flux

Une fois que l'objet Kinect est instancié, on peut lui activer les différents flux : flux de couleur, flux de profondeur et suivi de squelette.

Figure 19 – Activation des flux du capteur

```
// Enable the color stream on the sensor
_sensor.ColorStream.Enable(ColorImageFormat.RgbResolution640x480Fps30);

// Enable the depth stream with the default resolution
_sensor.DepthStream.Enable(DepthImageFormat.Resolution320x240Fps30);

// Enable the skeleton stream
_sensor.SkeletonStream.Enable();
```

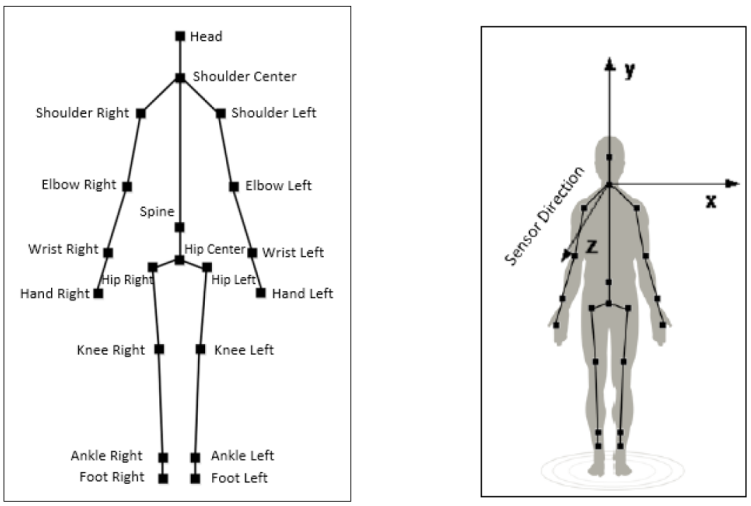
Source : Application Ki-Mouse

La classe Skeleton

C'est une des classes les plus utilisées lors du développement de l'application. En effet, elle permet de capturer la position des personnes suivies. Grâce à ses méthodes, il est possible d'obtenir les informations relatives aux squelettes. Actuellement, et comme cité précédemment dans ce document, le suivi est limité à six personnes. Chaque squelette possède différents éléments :

- Un identifiant unique
- Un état de suivi : Tracked (suivi), Not Tracked (non suivi), Position Only (non suivi totalement)
- Ses positions en X, Y, Z dans le plan
- L'accès aux 20 membres du corps qui sont suivis

Figure 20 – Suivi du squelette



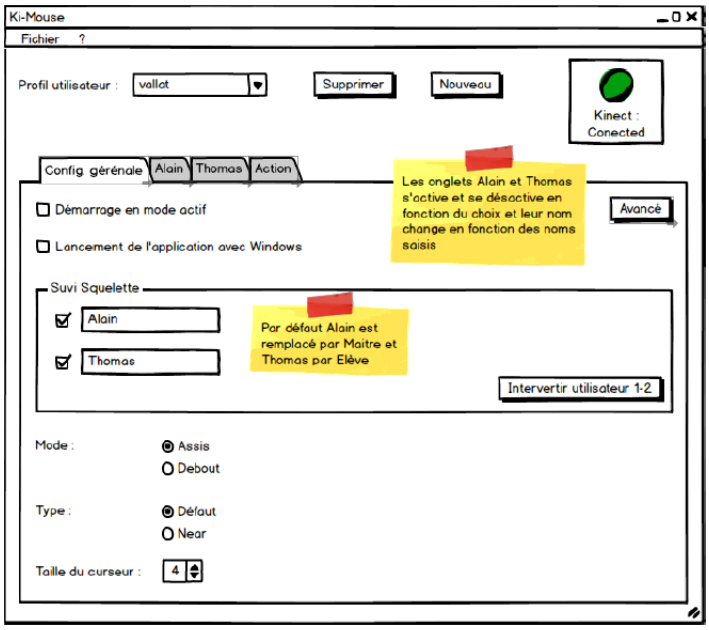
Source : Kinect for Windows SDK Programming Guide, p. 163 et 188

4.3 Mock-up de l'application

Le développement demandé pour cette application contenait une contrainte importante : les utilisateurs. En effet, comme expliqué au début de ce document, cette application est destinée dans un premier temps à la Castalie et à leurs employés ainsi qu'aux pensionnaires. Il a donc fallu rendre l'application très ergonomique et simple d'utilisation afin qu'elle soit accessible à cette catégorie d'utilisateurs. Pour ce faire, la réalisation de l'application a été répartie en différentes étapes (Sprint). Cette répartition a permis de valider les fonctionnalités régulièrement avec M. Torrent.

Dans un premier temps, sur la base des informations fournies, j'ai réalisé un mock-up de l'application à l'aide du logiciel Balsamiq[1]. Grâce à cet outil, les différents écrans créés ont été transmis au client (cf. Annexe IV). Le mandant a pu les analyser et me transmettre ses commentaires grâce auxquels j'ai pu effectuer les modifications nécessaires. Une fois arrivé au résultat qui correspondait le mieux à la vision du client, celui-ci les a validées et le développement a pu commencer. La figure 21 montre un exemple du mock-up de l'écran principal de l'application :

Figure 21 – Dessin écran de l'application Ki-Mouse



Source : Logiciel Balsamiq

[1] http://balsamiq.com/

4.4 Fonctionnalités développées

4.4.1 Écran principal

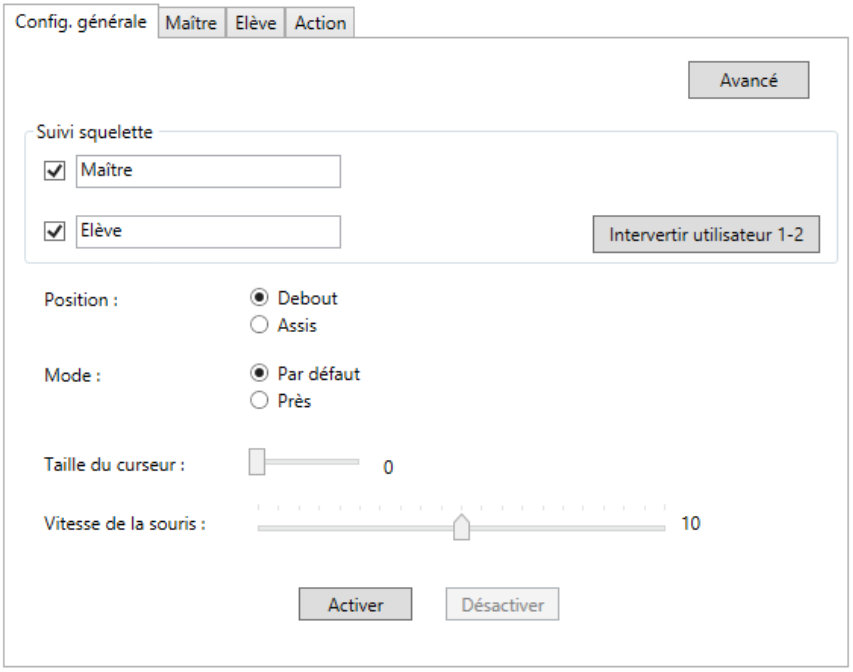
L'écran principal de l'application comporte au sommet les éléments se rapportant aux profils utilisateurs. La partie centrale composée d'onglets permet d'effectuer toute la configuration des utilisateurs. Pour finir, au bas de la fenêtre, une barre de statut indique l'état du capteur.

Figure 22 : Barre de statuts du capteur

Source : Application Ki-Mouse

4.4.2 Configuration générale

Figure 23 : Onglet configuration général



Source : Application Ki-Mouse

Ce premier onglet s'affiche lorsque l'application est lancée. Il permet à l'utilisateur d'effectuer les configurations de bases du capteur et de l'application.

Tout d'abord on y trouve deux cases à cocher accompagnées de deux zones de texte. Ces éléments permettent de choisir qui va utiliser l'application et quel est le nom des personnes concernées. Dans le cas où les deux cases sont cochées, on entre dans le mode collaboratif. Si à la détection des personnes, le maître et l'élève ont été inversés, un bouton permet d'intervertir les deux squelettes pour corriger le problème.

L'option position permet de choisir si l'on souhaite utiliser l'application de manière debout ou assise. L'option mode permet de sélectionner si l'on est positionné à une distance normale ou proche du capteur (cf. Tableau 1).

Les deux éléments suivants interagissent directement sur le curseur physique de l'ordinateur. Le premier élément permet de changer la taille de celui-ci et le second élément permet de changer la vitesse de déplacement.

Pour terminer cet onglet, deux boutons sont à disposition pour activer ou désactiver le suivi des squelettes. Dans le cas où le suivi et le contrôle de la souris sont activés, pour faciliter la désactivation, il est possible d'utiliser la touche CTRL gauche.

4.4.3 Bouton avancé

En cliquant sur le bouton avancé, une nouvelle fenêtre apparaît. Elle permet de visualiser le flux d'images capté par le Kinect. Dans la zone de gauche, le flux normal capturé est affiché alors que dans la zone de droite le flux d'images est affiché en noir et blanc avec les informations de profondeur.

Cette fenêtre offre aussi la possibilité de changer l'inclinaison du capteur en sélectionnant le nouvel angle que l'on souhaite lui donner à l'aide du slider. La dernière information disponible est le nombre de squelettes suivi par le Kinect.

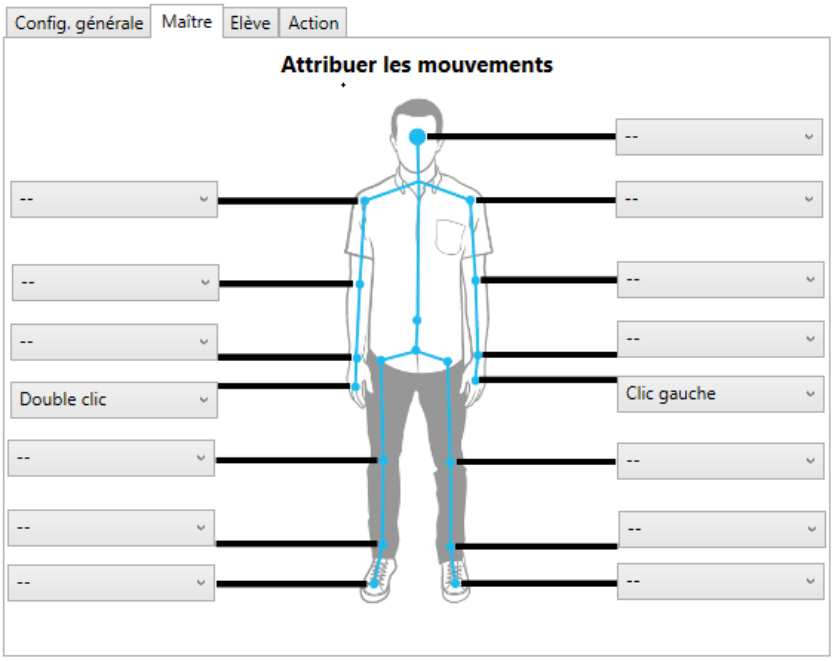
Figure 24 : Aperçu de la fenêtre Avancé

Source : Application Ki-Mouse

4.4.4 Choix des actions à effectuer

Par défaut, ces deux onglets se prénomment maître et élève. Ils permettent de sélectionner avec quels membres du corps on souhaite effectuer les actions souris. Pour chacun des membres, une liste déroulante comprenant les actions disponibles permet d'effectuer sa propre configuration. Une action ne peut être sélectionnée qu'une seule fois soit pour le maître, soit pour l'élève.

Figure 25 : Onglet de sélection des actions



Source : Application Ki-Mouse

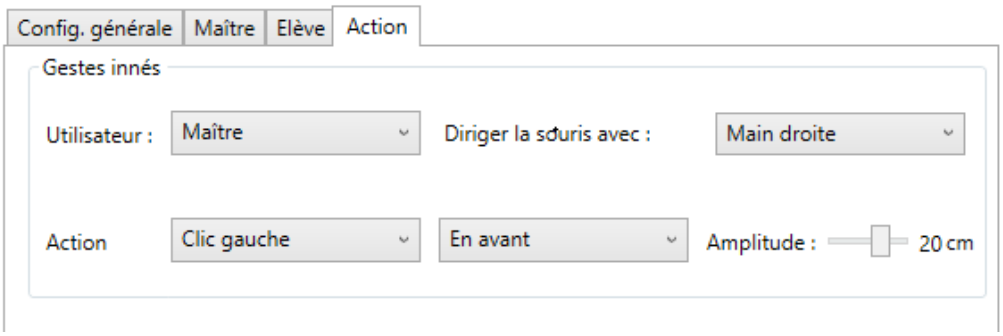
4.4.5 Paramétrages des actions

Ce dernier onglet permet de paramétrer chacune des actions sélectionnées précédemment. Une liste déroulante permet de choisir l'utilisateur sur lequel on souhaite agir. Une fois que l'utilisateur est sélectionné, la liste des actions se rapportant à celui-ci est disponible. Ce dernier peut alors configurer chacune d'entre elles. Les deux options disponibles sont le mouvement à effectuer et son amplitude.

Une liste déroulante spécifique permet de sélectionner la partie du corps qui contrôlera le curseur de la souris.

Quand ces différentes opérations ont été effectuées, le suivi des squelettes peut être activé et les configurations choisies sont prises en compte

Figure 26 : Onglet de paramétrage des actions



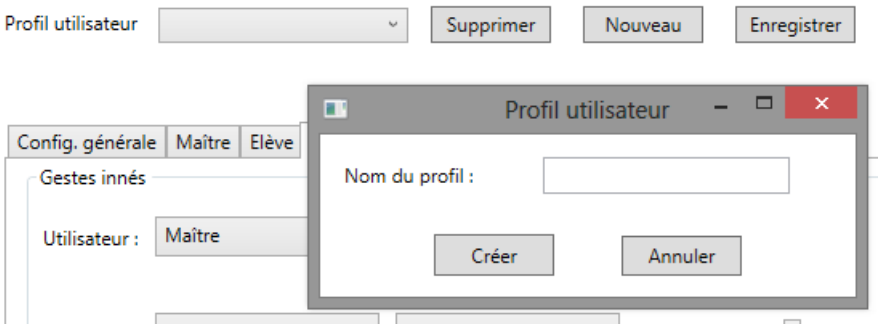
Source : Application Ki-Mouse

4.4.6 Gestion des profils utilisateurs

Une des fonctionnalités très importantes de l'application est la gestion des profils utilisateurs. Elle permet de récupérer sa configuration complète lors de l'utilisation ultérieure de celle-ci. Une liste déroulante comportant tous les profils existants est à disposition. Dans le cas contraire, le bouton nouveau permet de créer son profil qui sera par la suite disponible dans la liste déroulante.

Lors de la sélection d'un profil, la configuration complète est rechargée dans l'application. En cliquant sur le bouton Sauver, les modifications effectuées sont enregistrées dans le profil.

Figure 27 : Fenêtre de création d'un nouveau profil



Source : Application Ki-Mouse

4.4.7 Suivi des squelettes

Le suivi des squelettes est la fonctionnalité principale de l'application, car sans elle l'utilisateur ne peut rien faire. En effet, cette fonction développée totalement permet tout d'abord d'identifier le maître et/ou l'élève. Puis, elle s'occupe pour chaque squelette suivi de détecter les mouvements effectués par les membres du corps. En fonction des configurations faites dans les différents onglets, elle vérifie si le mouvement correspond à ce qui a été défini et si celui-ci est correct, puis elle effectue l'action souris correspondante.

Cette fonction est exécutée en permanence dès que le suivi est activé par l'utilisateur.

```
SensorSkeletonFrameReady{
    1. Détection du nombre d'utilisateurs dans le champ
    2. Détection du maître et de l'élève
    3. Récupération du squelette du maître et de l'élève
    4. Sélection du mode à utiliser : 1 squelette ou 2 squelettes à
       suivre
    5. Détecter et exécuter pour chaque squelette les actions
       définies
}
```

4.5 Problèmes rencontrés

Le travail s'est bien déroulé dans son ensemble. Cependant, j'ai dû faire face à quelques difficultés qui ont surgi tout au long du projet. Les points suivants décrivent les problèmes rencontrés et les solutions trouvées.

4.5.1 Apprentissage du SDK

L'apprentissage du capteur Kinect, du SDK et des outils mis à disposition par Microsoft a pris plus de temps que ce qui avait été planifié, car celui-ci ne s'est pas révélé aussi aisé que prévu.

L'apprentissage du capteur Kinect, du SDK ainsi que des outils mis à disposition par Microsoft m'a demandé un gros investissement en terme de temps. En effet, la prise en main de ces outils s'est avérée plus difficile que prévu.

Pour découvrir ces différents outils, j'ai utilisé le livre Kinect for Windows SDK Programming Guide. Il s'agit de la seule documentation complète que j'ai trouvée. Les fonctionnalités du SDK étaient bien expliquées au travers d'exemples, mais à plusieurs reprises, certaines d'entre elles n'étaient pas démontrées dans leur intégralité. Il était indispensable de comprendre le code et son fonctionnement pour assimiler le concept expliqué. Ce processus s'est révélé long et pas évident, car je n'avais aucune aide à disposition lorsque je ne parvenais pas à comprendre ce qui était démontré.

La solution trouvée pour pallier à ce manque d'explication et de compréhension a été de chercher des vidéos explicatives. Ces dernières ont éclairci les quelques concepts de base expliqués dans le livre.

4.5.2 Développement des fonctionnalités

Le problème majeur qui est apparu à plusieurs reprises durant la phase de développement de l'application, a été le manque d'exemples concrets sur lesquels s'appuyer.

En effet, dans le livre utilisé et dans les documentations proposées par Microsoft, les applications fournies sont spécifiquement dédiées à l'environnement Kinect et ses éléments graphiques. Dans mon cas, il fallait sortir de cet environnement puisque le but était d'interagir avec le système Windows. À de nombreuses reprises, pour certaines fonctionnalités que j'avais à développer, je ne pouvais me baser sur aucun des exemples à ma disposition. À l'heure actuelle, aucune application similaire à celle que j'ai développée n'existe. De plus, les exemples sont rares.

La seule solution qui s'offrait à moi était de faire de nombreuses recherches et d'explorer plusieurs pistes avant de trouver la solution à chaque problème.

4.6 Améliorations futures

Bien que l'application soit à présent fonctionnelle, ceci est une première version du projet. Le premier but de ce travail était de miser sur l'utilisabilité de l'application. Plusieurs améliorations graphiques peuvent être apportées et quelques fonctionnalités supplémentaires comme l'utilisation de la voix, le contrôle de la souris avec le visage ou encore la mise à jour automatique de l'application pourraient encore être implémentées. Ces éléments ont été laissés de côté par manque de temps, mais mériteraient d'être ajoutés afin d'avoir une application optimale.

4.6.1 Paramétrages des actions

Actuellement, après avoir sélectionné les actions sur les différentes parties du corps, l'utilisateur doit se rendre dans l'onglet action et paramétrer chaque action en la sélectionnant dans la liste déroulante. Avec le temps, cet aspect ne se révèlera pas forcément très pratique pour l'utilisateur.

Deux possibilités peuvent être envisagées pour rendre l'application plus ergonomique. La première solution serait de permettre à l'utilisateur au moment de la sélection de l'action sur le corps humain de définir directement la direction et l'amplitude. La deuxième solution est d'afficher dans l'onglet action toutes les actions les unes sous les autres afin d'éviter d'utiliser une liste déroulante.

4.6.2 Utilisation de la voix

Pour le moment, l'application ne permet pas d'utiliser la voix pour contrôler l'ordinateur. Cette fonctionnalité supplémentaire serait à développer et à rajouter dans l'onglet action en offrant la possibilité à l'utilisateur de choisir les mots à utiliser pour effectuer les différentes actions souris.

4.6.3 Package d'installation de l'application

Actuellement, pour utiliser l'application, l'utilisateur doit installer manuellement le SDK, les drivers et l'exécuter à l'aide de Visual Studio. Cette manipulation devra être remplacée par un

package d'installation qui permettra à n'importe quel utilisateur d'installer facilement l'application sur n'importe quel ordinateur. Ce pack d'installation comprendra tous les éléments nécessaires au fonctionnement de l'application.

4.6.4 Démarrage automatique de l'application

Dans l'écran de configuration de l'application, une case à cocher permet de spécifier si l'application doit être lancée au démarrage de Windows. Pour le moment, cette fonctionnalité n'est pas implémentée, mais se révèlera très pratique pour les utilisateurs.

4.6.5 Mise à jour automatique

Une fonctionnalité supplémentaire essentielle à l'avenir est la mise à jour automatique de l'application chez chaque utilisateur. Grâce à celle-ci, lorsque des modifications sont effectuées et validées par les développeurs, l'application offre la possibilité de la mettre à jour afin d'obtenir les dernières fonctionnalités ou les dernières corrections.

CONCLUSION

Bilan et critique

Pour terminer, il faut noter que ce travail s'est déroulé dans de très bonnes conditions. Grâce à la liberté qui m'a été offerte pendant toute la durée du projet, j'ai pu réaliser une application qui correspond aux attentes du client. M. Jean-Pierre Rey m'a accordé sa confiance, ce qui m'a permis d'avoir une grande autonomie et d'avancer le mieux possible.

Toutes les principales fonctionnalités mentionnées dans le cahier des charges ont été développées et toutes les User Stories, à l'exception d'une, ont été implémentées. Certaines demandes supplémentaires du client, comme l'utilisation de la voix, ont été écartées en raison du manque de temps à disposition.

D'un point de vue personnel, je suis satisfait du travail que j'ai effectué. J'ai eu beaucoup de plaisir à accomplir cette tâche et cela pour plusieurs raisons :

Tout d'abord, j'ai pu découvrir un nouveau produit qui pour moi avant ce travail était un périphérique de jeu et rien d'autre. Les possibilités offertes par le Kinect sont impressionnantes et l'on n'imagine pas forcément tout ce qu'il est possible de réaliser avec celui-ci. J'ai à présent pris connaissance d'une grande partie des applications exploitables avec ce produit.

Ensuite, ce projet m'a permis d'acquérir de très bonnes connaissances en C# et WPF, domaines dans lesquels je n'avais que de faibles notions. L'utilisation du SDK m'a permis de découvrir une nouvelle manière de travailler en se basant sur un outil très puissant. À présent, je connais ces langages et je n'aurais aucun problème à transférer les connaissances acquises dans d'autres travaux similaires.

Durant le projet, à plusieurs reprises, j'ai été victime d'une démotivation pour avancer et arriver au résultat demandé. Cela était dû au manque de documentation et à la difficulté de compréhension du SDK. Il n'est pas facile de travailler sur un sujet où aucun exemple concret n'existe et où rien ne peut servir d'aide pour le développement.

Finalement, au fur et à mesure de l'avancement du projet, j'ai eu l'impression de mettre un pied dans le monde professionnel en travaillant à plein temps sur le projet en relation avec

un client. Ce travail fut pour moi une très bonne préparation avant d'entrer dans la vie active et de débuter une carrière professionnelle.

RÉFÉRENCES

Fernandez, D. (2012, Février 1). *Skeletal Tracking Fundamentals*. Récupéré sur Channel 9: http://channel9.msdn.com/Series/KinectQuickstart/Skeletal-Tracking-Fundamentals

Fondation Suisse pour les téléthèses. (2011). *Fondation FST*. Récupéré sur http://www.fst.ch/fr/fondation-fst.html

Guye, R. (2011). *Pré-étude GMouse*. Fribourg: Haute École Spécialisée de Suisse Occidentale.

Jana, A. (2012). *Kinect for Windows SDK Programming Guide*. Birmingham: Packt Publishing Ltd.

Larousse. (2012). *Le Petit Larousse illustré 2012*. Paris: Edition Larousse.

Microsoft. (2013). *Human Interface Guidelines*. Récupéré sur http://download.microsoft.com/download/B/0/7/B070724E-52B4-4B1A-BD1B-05CC28D07899/Human_Interface_Guidelines_v1.7.0.pdf

Microsoft. (2013). *Kinect FAQ*. Récupéré sur http://www.microsoft.com/en-us/kinectforwindows/news/faq.aspx

Microsoft. (2013). *Kinect for Windows features*. Récupéré sur http://www.microsoft.com/en-us/kinectforwindows/discover/features.aspx

Microsoft. (2013). *New features*. Récupéré sur http://www.microsoft.com/en-us/kinectforwindows/Develop/New.aspx

Microsoft. (2013). *Skeletal Tracking*. Récupéré sur http://msdn.microsoft.com/en-us/library/hh973074.aspx

Microsoft. (2013, Mai 7). *Tracking Modes*. Récupéré sur http://msdn.microsoft.com/en-us/library/hh973077.aspx

Microsoft. (2013). *Tracking Modes*. Récupéré sur http://msdn.microsoft.com/en-us/library/hh973077.aspx

Saudan, A. (2011). *Pré-étude 1ère évaluation*. Monthey: La Castalie.

Smith, W. (2012, Octobre 3). *Easy Kinect Mouse Controller for Windows*. Récupéré sur Microsoft Dev: http://code.msdn.microsoft.com/windowsdesktop/Easy-Kinect-Mouse-09233c52#content

GLOSSAIRE

Agile : *« Une méthode agile est une approche itérative et incrémentale, qui est menée dans un esprit collaboratif avec juste ce qu'il faut de formalisme. Elle génère un produit de haute qualité tout en prenant en compte l'évolution des besoins des clients ».* Veronique Messager Rota. (7 mai 2009).

AppChoosesSkeletons : Fonction comprise dans le SDK qui permet d'activer le choix manuel des squelettes à suivre.

Caméra RGB : *« Une caméra RGB offre les trois composants de base des couleurs (rouge, vert et bleu) sur trois fils différents. Ce type d'appareil utilise souvent trois capteurs CCD indépendants pour l'acquisition des trois signaux de couleur.»* National Instruments. (10 octobre 1997).

GMouse : Nom de l'application développée par Rapahel Guye

Grip : Terme utilisé pour le défilement ou encore la reconnaissance de quatre mains simultanément

ID : Identifiant utilisé pour différencier chaque squelette capturé par le Kinect

Interface utilisateur naturelle (NUI) : Nouveau type d'interface permettant d'intégrer l'informatique aux actions du quotidien.

Kinect : Capteur de Microsoft conçu pour PC et utilisé pour le développement de l'application.

Application Programming Interface (API) : Interface de programmation spécifiant comment les composants logiciels interagissent avec chacun des autres.

Light-Emitting Diode (LED) : Composant électronique permettant d'informer l'état du capteur Kinect.

Geste inné : Geste naturel pouvant être effectué par n'importe quel être humain.

Machine virtuelle : « Une machine virtuelle est un conteneur de logiciels totalement isolé, capable d'exécuter ses propres systèmes d'exploitation et applications, à l'instar d'un ordinateur physique. » VMWare. (2013).

Mockup : Prototype d'écran utilisateur permettant de présenter les fonctionnalités d'un logiciel.

Mode collaboratif : Mode permettant d'utiliser le capteur et l'application avec deux personnes simultanément.

Near mode : Mode permettant de voir des objets situés à 40 cm devant le capteur.

Product backlog : Liste de toutes les fonctionnalités de l'application.

Push : Action permettant de sélectionner les objets virtuels

Software development kit (SDK) : Kit d'outil permettant aux développeurs de créer des applications.

SkeletonStream.ChooseSkeletons : Fonction du SDK permettant de sélectionner le squelette à suivre

Sprint : Période de temps d'une durée fixe durant laquelle une personne ou un groupe de personnes travaillent sur une série de tâches.

Sweet Spot : Zone où les personnes auront une interaction maximale avec le capteur

Toolkit : Boîte à outils permettant d'obtenir des éléments graphiques et des exemples pour le développement d'applications.

Tracking : Terme anglais utilisé pour définir qu'un squelette ou un membre du corps est suivi par le capteur.

User Story : Terme anglais utilisé pour déterminer une fonctionnalité dans le langage Agile.

Windows Presentation Foundation (WPF) : Système de présentation pour construire des applications client Windows

Annexe I : **Tableau des heures**

Sprint 0

Date fin	Travail effectué	Durée (h)
29.04.2013	Mise en place environnement de travail (modèle, documents,…)	8,5
30.04.2103	Lecture évaluation de la Castalie	1
30.04.2103	Ressortir les remarques importantes	1
06.05.2013	Lecture du document de pré-étude : Cahier des charges + GMouse Rapport	3
06.05.2013	Ressortir les éléments importants du GMouse Rapport	5
08.05.2013	Recherche/lecture de la documentation sur le SDK	20,5
08.05.2013	Document qui ressort les possibilités du SDK	4
09.05.2013	Créer un tableau comparatif par rapport aux éléments ressortis pour la pré-étude et le SDK	1
09.05.2013	Chercher comment s'effectue la calibration de la Kinect sur le pc	1
09.05.2013	Chercher si la calibration est obligatoire	1
13.05.2013	Chercher comment on peut gérer plusieurs personnes avec la Kinect : entrée multipoint	3
13.05.2013	Mettre en place l'environnement de développement	1
28.05.2013	Lecture du livre pour apprendre le fonctionnement du SDK	48
24.05.2013	Mockup de l'application	4
27.05.2013	Réalisation d'un prototype	8
	TOTAL	110

Sprint 1

Date fin	Travail effectué	Durée (h)
04.06.2013	Créer la base de l'application dans Visual Studio	20
04.06.2013	Afficher l'état de la Kinect (connecté,…)	2
07.06.2013	Afficher la caméra sur l'écran avancé	6
13.06.2013	Déplacer la souris et effectuer un clic	6
-	Détecter les 2 squelettes distinctement	6
18.06.2013	Effectuer des actions simultanées avec les 2 squelettes	10
21.06.2013	Définir une méthode pour chaque action souris possible	2
21.06.2013	Recherche	4
	TOTAL	56

Sprint 2

Date fin	Travail effectué	Durée (h)
12.07.2013	Lier les actions sélectionnées dans les listes déroulantes aux parties du corps	5
10.07.2013	Paramétrer le mouvement à effectuer pour un clic et tester (onglet action)	10
12.07.2013	L'application répond aux actions sélectionnées pour chaque utilisateur	15
09.07.2013	Détecter les 2 squelettes distinctement	12
05.07.2013	Rédaction du rapport	45
08.07.2013	Lier les listes déroulantes entre les onglets	6
12.07.2013	Recherche	7
19.07.2013	Séance avec le client	3
	TOTAL	103

Sprint 3

Date fin	Travail effectué	Durée (h)
24.07.2013	Mise à jour des actions entre les onglets	7
13.07.2013	Créer une interface pour créer un profil	1,5
13.07.2013	Un fichier par profil contenant les configurations	4,5
13.07.2013	Afficher la liste des profils existants	1
19.07.2013	Recharger les configurations du profil sélectionné	12
13.07.2013	Supprimer un profil	1
22.07.2013	Recherche	5
23.07.2013	Amélioration du code	6
23.07.2013	Tests de l'application	6
22.07.2013	Finalisation de l'application	10
25.07.2013	Rédaction du rapport + impression, reliure, rendu	45
	TOTAL	95

Nombre d'heures total : **364**

Annexe II : **Product Backlog**

US Nr.	En tant que	Je veux…	De sorte que…	Story points	Status
1	Développeur	Mettre en place mon environnement de travail	Installer les logiciels nécessaires et créer les documents de base pour le travail	2	🟢
2	Développeur	Ressortir les éléments de la pré-étude réalisée (Cahier des charges + GMouse Rapport)	Analyser les documents fournis et ressortir les éléments importants	2	🟢
3	Développeur	Ressortir les possibilités et les fonctionnalités offertes par le SDK de Microsoft	Analyser les documentations trouvées	4	🟢
4	Développeur	Comparer les deux solutions (pré-étude et SDK) et choisir celle qui correspond le mieux	Ressortir les avantages et inconvénients des deux solutions	2	🟢
5	Développeur	Ne pas devoir calibrer la Kinect lors de chaque utilisation	Garder la même précision sans la calibration à chaque utilisation	2	🟢
6	Développeur	Gérer plusieurs personnes	Avoir la possibilité de choisir le nombre de personnes (2 mains utilisateur, 1 main utilisateur et 1 thérapeute,…)	5	🟢
7	Développeur	Apprendre le fonctionnement du SDK	Lire et comprendre le livre "…"	8	🟢
8	Utilisateur	Avoir une interface ergonomique et facile d'utilisation	L'utilisateur puisse facilement comprendre et utiliser l'application	5	🟢
9	Utilisateur	Attribuer les membres du corps à utiliser	Sélectionner la partie à utiliser : main, tête, pied, zone définie à l'écran.	8	🟢
10	Utilisateur	Aucune calibration nécessaire	Utiliser le Kinect avec n'importe quel membre du corps	3	🔴
11	Utilisateur	Sauvegarder mes paramètres	Pouvoir réutiliser ma configuration lors d'une utilisation ultérieure	5	🟢

12	Utilisateur	Gérer les paramètres de plusieurs utilisateurs	Conserver les configurations de chaque utilisateur	5	
13	Utilisateur	Configurer les fonctions souris	Définir le clic, double clic, glissé déplacé,…)	5	
14	Utilisateur	Programme facilement accessible	Avoir une icône permettant de lancer un panneau de configuration + menu contextuel relatif	3	
15	Utilisateur	Pouvoir utiliser l'application en mode collaboratif	Gérer l'utilisation de l'application par plusieurs personnes simultanément	8	
16	Utilisateur	Pouvoir déplacer le curseur de la souris et pouvoir régler la vitesse	Le curseur de la souris se déplace en même temps que la main bouge	5	
17	Développeur	Rédaction du rapport	Avoir un rapport professionnel du travail réalisé	8	
18	Développeur	Finalisation et tests de l'application	Avoir une application optimisée et tester les principaux cas	5	

Annexe III : **Procès-verbaux**

Procès-verbal #1

Date	**26.04.2013**
Personnes présentes	Jean-Pierre Rey (Professeur) Julien Torrent (FST) Romain Paccolat (Réalisateur du projet)
Personnes absentes	-
Cadre de la séance	• Séance de démarrage • Discussion sur les fonctionnalités souhaitée par Julien Torrent • Documents souhaités pour la période d'analyse.
Prochaine séance	• 3 mai 2013

Procès-verbal #2

Date	**03.05.2013**
Personnes présentes	Jean-Pierre Rey (Professeur) Romain Paccolat (Réalisateur du projet)
Personnes absentes	-
Cadre de la séance	• Discussion du cahier des charges établi
Prochaine séance	• 17 mai 2013

Procès-verbal #3

Date	**17.05.2013**
Personnes présentes	Romain Paccolat (Réalisateur du projet) Julien Torrent (FST)
Personnes absentes	Jean-Pierre Rey (Professeur)

Cadre de la séance	• Présentation de l'analyse effectuée selon la demande de Julien Torrent
	• Présentation des premiers résultats de l'apprentissage du SDK
	• Présentation de la suite du travail :
	  • Terminer l'apprentissage du SDK
	  • Présenter les possibilités offertes par le SDK au travers d'un ou deux exemples
	  • Créer un mock-up de l'application

Prochaine séance	• 3 juin 2013

Procès-verbal #4

Date	**03.06.2013**
Personnes présentes	Romain Paccolat (Réalisateur du projet) Julien Torrent (FST)
Personnes absentes	Jean-Pierre Rey (Professeur)

Cadre de la séance	• Présentation des possibilités offertes par le SDK grâce à des exemples réalisés avec le livre.
	• Discussion du mock-up de l'application et corrections de certains éléments selon les remarques de Julien Torrent
	• Sprint review :
	  • Discussion de l'état actuel du projet et de ce qui a été réalisé
	  • Fixation des priorités de Julien

	• Présentation de la suite du travail : • Créer l'aspect graphique de l'application • Pouvoir effectuer les réglages de base du capteur • Pouvoir déplacer le curseur à l'aide la main
Prochaine séance	• Non définie

Procès-verbal #5

Date	**13.06.2013**
Personnes présentes	Romain Paccolat (Réalisateur du projet) Julien Torrent (FST) Jean-Pierre Rey (Professeur)
Personnes absentes	-
Cadre de la séance	
	• Discussion de l'état actuel du projet • Discussion sur les problèmes qui empêchent le projet d'avancer • Fixation de nouvelles priorités aux fonctionnalités de l'application par Julien et Jean-Pierre • Le prochain objectif est de déterminer comment le mode collaboratif peut être implémenté car c'est la priorité principale de Julien
Prochaine séance	• Dès que le mode collaboratif sera implémenté

Procès-verbal #5

Date	**01.07.2013**
Personnes présentes	Romain Paccolat (Réalisateur du projet) Julien Torrent (FST)
Personnes absentes	Jean-Pierre Rey (Professeur)
Cadre de la séance	• Présentation des nouveautés de l'interface graphique de l'application : • Les actions sélectionnées sont reprises dans l'onglet action • Les actions de chaque utilisateur sont reprises séparément • Intervertir les utilisateurs si la détection est fausse • Test du mode collaboratif : • Problème de détection des utilisateurs, il faut être les 2 dans le champ pour que cela fonctionne • L'utilisation de l'application par 2 personnes simultanément a fonctionné donc Julien est content • La détection du Maitre et de l'élève lorsqu'on quitte et revient dans le champ n'est pas au point. Il faut corriger cette partie
Prochaine séance	• Dès que les corrections mentionnées seront implémentées

Procès-verbal #6

Date	**11.07.2013**
Personnes présentes	Romain Paccolat (Réalisateur du projet) Julien Torrent (FST)
Personnes absentes	Jean-Pierre Rey (Professeur)
Cadre de la séance	• Présentation des améliorations pour le mode collaboratif et la détection : • Détection du maître et de l'élève fonctionne même lorsqu'on quitte le champ • La sélection des actions pour chaque utilisateur fonctionne • Le paramétrage de chaque action avec la direction et l'amplitude fonctionne • L'utilisation en mode collaboratif fonctionne • Exécution des actions : • Problème concernant la durée d'exécution de l'action. Un timer de 1 seconde a dû être défini • Trouver s'il y a la possibilité d'améliorer ce temps d'attente • Définition de la suite du travail : • La dernière fonctionnalité à implémenter à la gestion des profils
Prochaine séance	• 19 juillet 2013

Romain Paccolat

Procès-verbal #7

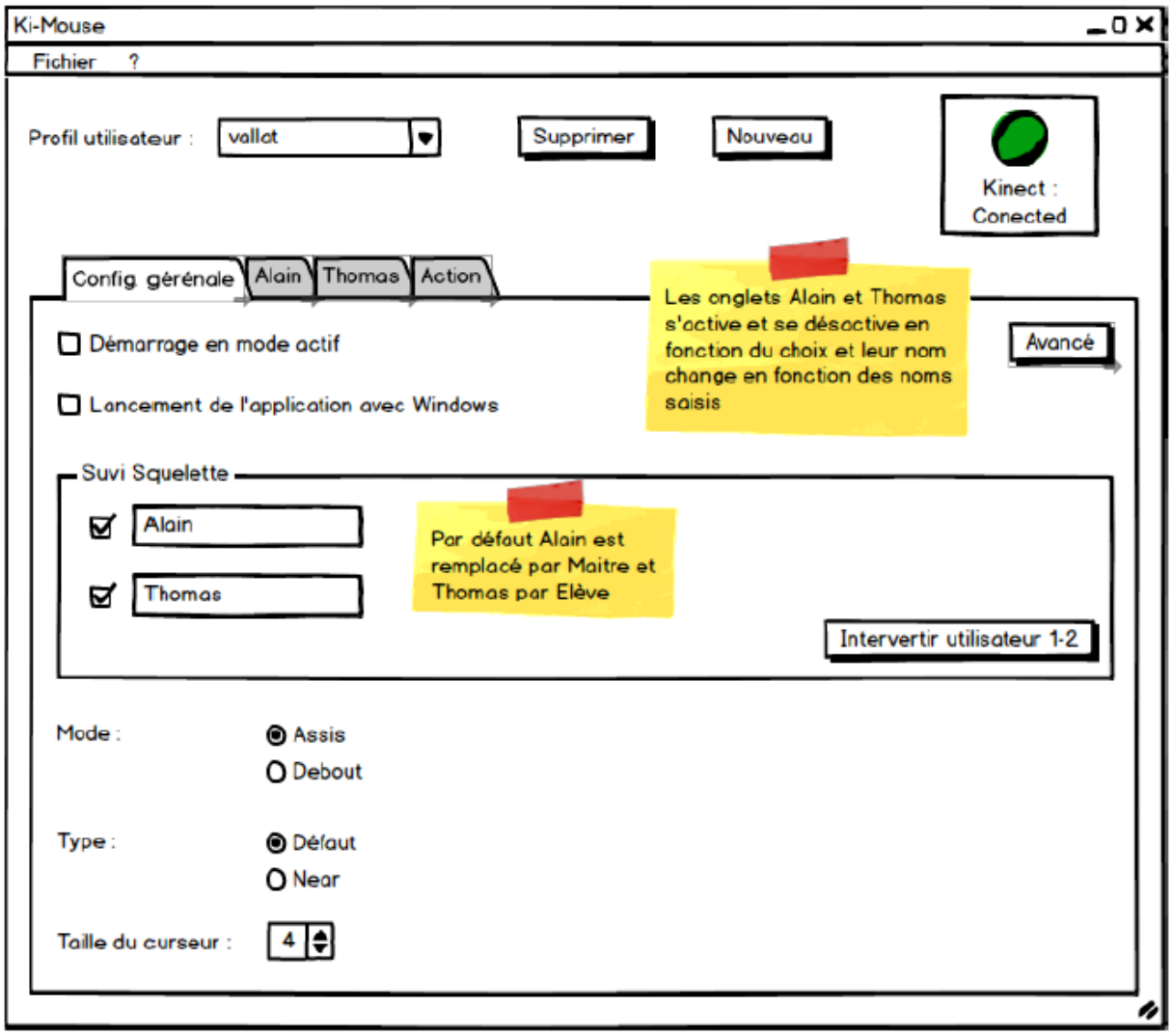
Date	**18.07.2013**
Personnes présentes	Romain Paccolat (Réalisateur du projet) Julien Torrent (FST)
Personnes absentes	Jean-Pierre Rey (Professeur)

Cadre de la séance	• Présentation de la gestion des profils :
	  • Créer un nouveau profil : un fichier xml se crée avec les configurations
	  • Charger un profil existant : les configurations du fichier de l'utilisateur sont rechargées dans l'application
	  • Suppression d'un profil
	• Mode collaboratif et simple :
	  • Paramétrage des actions pour le maitre et l'élève et test de l'application à l'aide du site poissonrouge.com pour avoir un exemple concret
	• Suite du travail :
	  • Corriger les bugs graphiques
	  • Optimiser le code
	  • Optimiser l'application

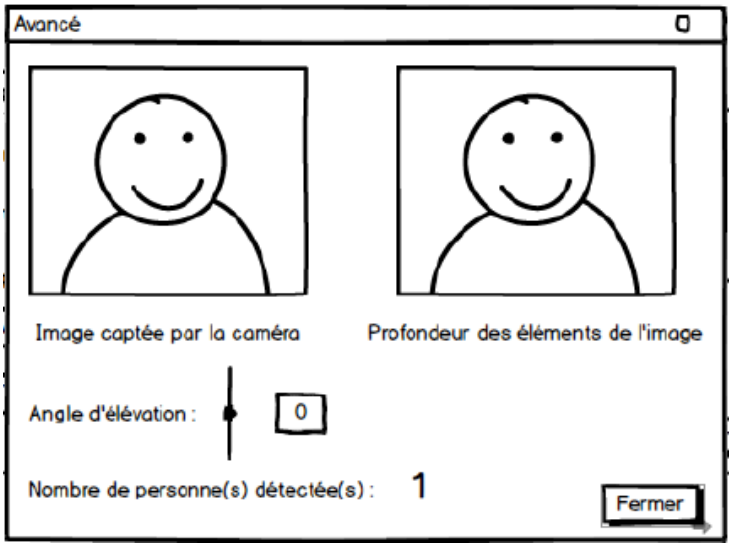
Prochaine séance	• -

Annexe IV : **Mock-up de l'application**

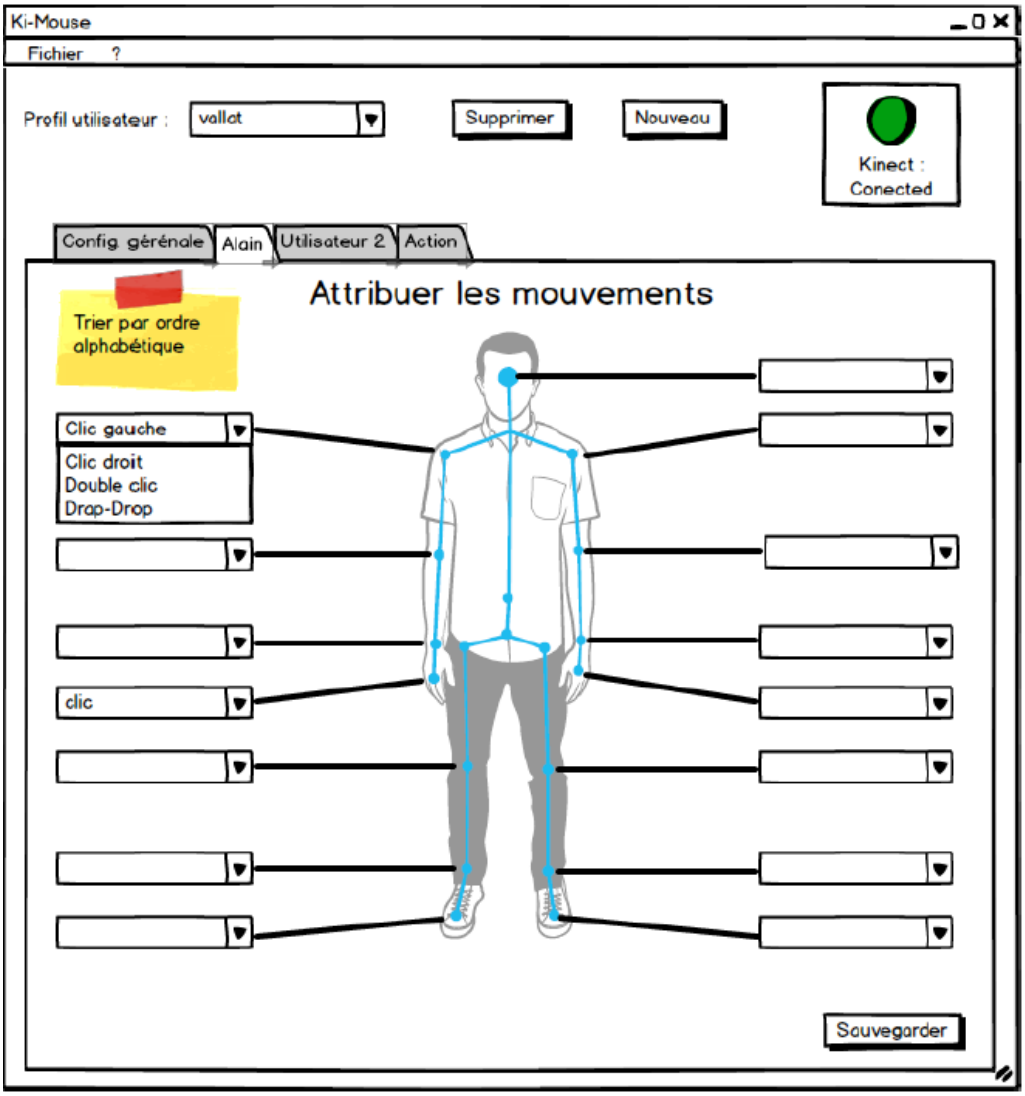
Fenêtre principale



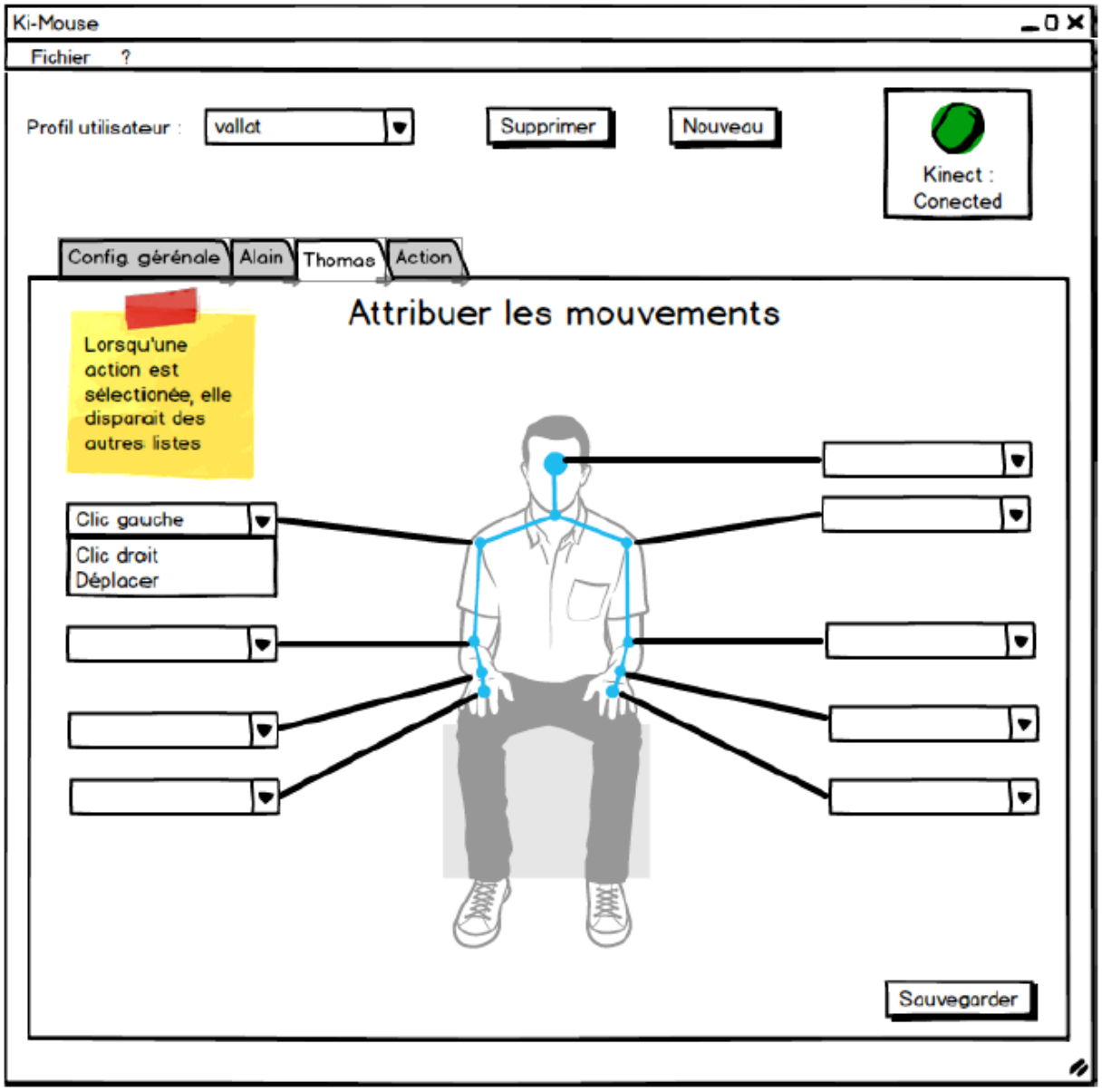
Fenêtre « Avancé »



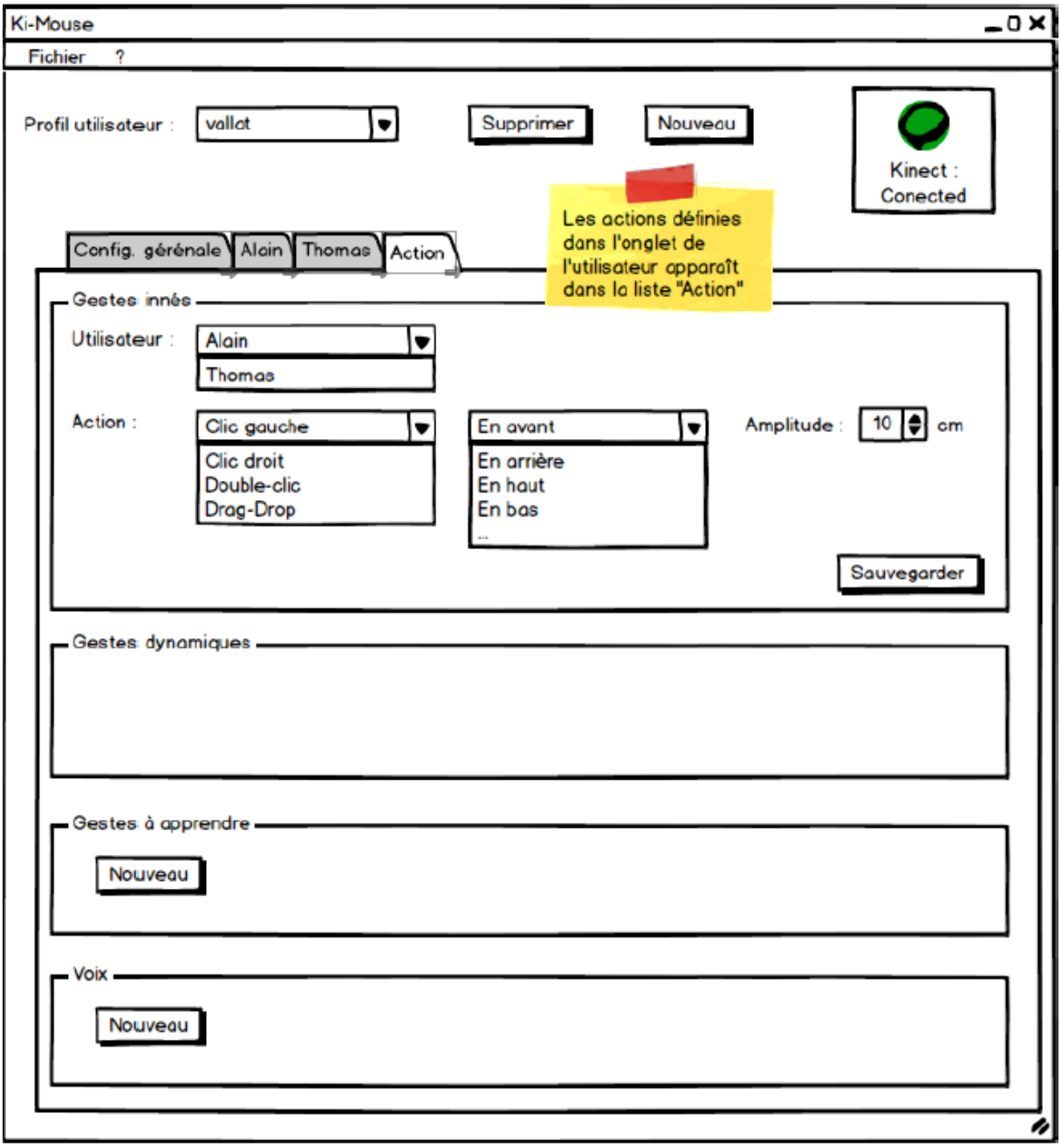
Fenêtre d'attribution des actions : mode debout



Fenêtre d'attribution des actions : mode assis



Fenêtre de paramétrages des actions



DÉCLARATION DE L'AUTEUR

Je déclare, par ce document, que j'ai effectué le travail de Bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du RF et du professeur chargé du suivi du travail de Bachelor, y compris au partenaire de recherche appliquée avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après :

- M. Jean-Pierre Rey
- M. Julien Torrent