Rapport Kinect

1. Gestes reconnus

Notre projet contient les 6 gestes suivants :

- Geste d'aller-retour de la main droite vers la droite, puis retour vers la gauche
- Geste d'aller-retour de la main droite vers la gauche, puis retour vers la droite
- Geste d'aller-retour des deux mains vers le haut, puis retour vers le bas
- Déplacement de la main droite vers l'avant puis retour vers l'arrière
- Simuler la course avec les bras (Mouvements de va-et-vient de haut en bas, dans un sens différent pour les deux mains)
- Geste d'aller-retour de la main droite vers le bas, puis retour vers le haut

2. Solutions algorithmiques

Notre solution consiste à considérer chaque vecteur de déplacement (entre chaque frame) afin de déterminer une direction. Nos mouvements sont déterminés à partir d'une succession spécifique de changements de direction. Par exemple pour le premier geste, nous enregistrons une position de départ du mouvement lors d'un mouvement vers la droite. Lorsqu'un mouvement vers la gauche est détecté, nous le suivons jusqu'à arriver à une certaine distance du point de départ enregistré précédemment. Chaque mouvement est en fait une mini machine à état indépendante des autres, ce qui permet de détecter plusieurs mouvements contenant les même directions en même temps.

3. Fichiers d'évaluation

Légende: Pour chaque mouvement et chaque personne, il y a quatre chiffre binaire représentant les 4 exécution d'un mouvement où 1 signifie que le geste a été reconnu et 0 signifie qu'il ne l'a pas été.

Nom	Mouvement 1	Mouvement 2	Mouvement 3	Mouvement 4	Mouvement 5	Mouvement 6
Shems	1000	1101	0010	0000	0010	1111
Guillian	0010	0000	1010	0000	0000	1011
% moyen de détection	25%	37.5%	37.5%	0%	12.5%	87.5%

4. Manuel d'utilisation du package

Intégration du package: L'intégration du package dans un nouveau projet se fait assez simplement en allant dans "Assets > Import Package > Custom Package" et en sélectionnant notre package. Une fois tous les fichiers importé, ajoutez le préfab "KinectController" (situé dans le dossier "Prefabs") à votre scène. Aussi, il est possible d'ajouter un objet Text dans le paramètre "Countdown Text Object" du script "MouvementHandler" qui, après avoir détecté un mouvement, désactive la détection pendant 3 sec (le temps de vous replacer correctement) et la réactive mais cela est optionnel. A présent, vous pouvez utiliser notre "KinectController".

Ajout de nouveaux mouvements: Pour commencer, une scène de test "Kinect Sample" est disponible dans le dossier "Scenes" et vous permet de tester 6 mouvements déjà programmé dans cette scène. Ses mouvements sont des scripts contenu dans le dossier "Scripts" et dont le nom est "Mouvement1", "Mouvement2", etc… Pour ajouter de nouveaux mouvements vous pouvez vous inspirer de ses scripts. Comme dit plus haut, chaque mouvement sont programmés comme une machine à état: On commence par récupérer la direction de déplacement d'une main. Pour la main droite, se sera dans le tableau goingRight du script "MouvementHandler" qui contient l'actuel direction de déplacement de la main droite pour chaque axe (X, Y et Z). Par exemple, pour l'axe X, la valeur de direction pourra être "1" (déplacement vers la droite), "-1" (déplacement vers la gauche) ou "0" (aucun déplacement).

Pour détecter un geste de balayage de la main droite vers la droite, on détecte si l'on l'a main droite se dirige vers la droite (avec goingRight[MouvementHandler.AXE_X] == 1) et, si le mouvement est bien détecté, on utilise la fonction "startMovement", qui prend en paramètre le numéro (ou l'id) du mouvement, pour signaler au script "MouvementHandler" que l'on souhaite démarrer un mouvement, ce qui bloquera le démarrage d'un autre mouvement en parallèle. La fonction "startMovement" renvoie "false" si il n'est pas possible de démarrer un mouvement. A la fin d'un mouvement, on signale la fin de celui-ci avec la fonction "endMovement" de "MovementHandler" qui prend en paramètre l'id du mouvement à terminé et une valeur booléen qui à "true" signifie que le mouvement c'est finalisé avec succès. Une fonction "startTimeoutCountdown" est disponible pour démarrer une compte à rebour qui, une fois arrivé à termes, termine un mouvement avec "erreur" (ce n'est pas une erreur à proprement parler mais c'est simplement que le mouvement ne s'est pas finalisé correctement). Pour savoir si le compte à rebour c'est terminé, il faut utiliser la fonction "GetMouvementTimeout" de "MouvementHandler" qui renvoie "true" si le compte à rebour est finie.