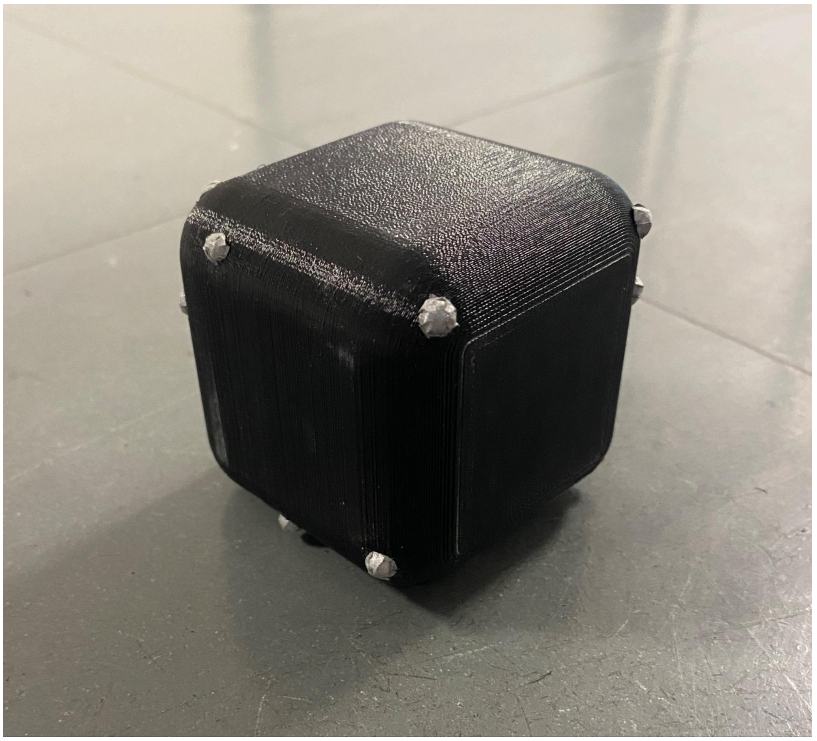
RAPPORT PROJET RVRA PostWIMP
2023 - 2024
3A MMIS

Cube Mouse



Par RENIAUD Felix, BOUBERBACHENE Nassim, BOUREZZA Mehdi, MENG Fei

Encadrés par M. Berard François

Sommaire

I. Introduction

1 - Contexte

Manipuler des modèles dans une scène 3D dans un logiciel tel que Unity n'est pas toujours simple et intuitif. Pour appliquer la translation parfaite il faut bien souvent faire une composition de plusieurs translation et la rotation non plus se fait en plusieurs étapes en général. Cette difficulté est due au fait qu'une souris est manipulée sur un plan 2D et que ce n'est pas assez de degré de libertés pour permettre des transformations à trois degrés de liberté. De plus, pour déplacer un objet en translation, le déplacer en rotation, déplacer la caméra en translation ou en rotation, il y a souvent deux possibilités pour choisir une transformation:

- Utiliser des raccourcis claviers en plus de la manipulation de la souris qui peuvent changer d'un logiciel à un autre.

- Cliquer sur des boutons du logiciel.

La première méthode n'est pas toujours intuitive tandis que la deuxième n'est pas très efficace puisqu'il faut naviguer dans l'interface avant chaque transformation. Enfin, avec ces méthodes il n'est pas possible d'effectuer à la fois une translation et une rotation ce qui constitue une limite pour l'utilisateur.

2 - Objectifs

L'idée du projet est de réaliser un prototype de souris "augmenté". On utilise un objet physique, un cube, à manipuler dans les trois dimensions de l'espace pour remplacer la souris d'ordinateur, dans l'optique d'avoir une meilleure maniabilité des objets ou caméras dans l'interface 3D Unity. Les objectifs étaient donc que le système permette dans Unity:

- Le déplacement du cube contre un plan pour l'utiliser comme une souris classique.

- La manipulation dans trois dimensions pour que les objets sélectionnés subissent les mêmes transformations que celles appliquées au cube réel.

- De déplacer et tourner la caméra de la scène si aucun objet n'est sélectionné.

Après implémentation nous avons alors testé et comparé l'utilisation de notre technique par rapport à l'utilisation d'une souris classique afin d'évaluer la qualité de l'interaction.

II. Solution envisagée

1 - Tracking et objet choisi

Pour récupérer les informations de positions et d'orientation nous avons choisi d'utiliser Optitrack et de créer un rigidbody associé à l'objet réel. Pour l'objet nous avons choisi d'utiliser un cube pour les raisons suivantes:

- Le cube est facile à attraper à la main et facile à manipuler à une seule main.

- C'est aussi une forme dont il est facile de suivre la rotation à l'oeil nu et de d'estimer visuellement l'ampleur des rotations que l'on applique et les rotations importantes telles que les multiples de 90 degrés.

- Les faces plates du cubes permettent de poser et de déplacer efficacement le cube contre un plan.

- Les cibles nécessaires pour le tracking du cube peuvent être posées sur les différents sommets et arêtes du cube et le tracking est efficace.

Figure 1 : une façon de prendre le cube

Pour récupérer la position de cube, il a suffi d'ajouter des cibles sur le cube pour générer un rigidbody et utiliser optitrack pour le détecter et nous renvoyer les informations de coordonnées et d'orientations 3D.

2 - Mode souris 2D

Dans ce mode, le cube fonctionne exactement comme une souris standard.

1. Navigation 2D

L'utilisateur déplace le cube face contre le plan du bureau sur lequel il est posé et peut déplacer le curseur à l'écran exactement comme une souris classique. Cela demande simplement de tracer les coordonnées du rigidbody et de prendre le contrôle du curseur dès que le cube est assez proche du plan du bureau.

2. Clic gauche et droite

Nous avions initialement envisagé de permettre les clics droits et clics gauche directement sur le cube avec les doigts associés à ces clics et indépendamment de l'orientation du cube en traquant les mouvement des doigts et en repérant les mouvements brusques de ces derniers mais nous n'avons pas pu l'implémenter dans les temps.

3. Scroll molette

Pour le scroll de la molette nous avons envisagé de l'effectuer par un glissement des doigts sur une face du cube mais nous n'avons pas pu l'implémenter non plus.

3 - Mode transformation 3D

Dès lors que le cube est levé trop haut par rapport au bureau, le système passe en mode souris 3D. Ce mode permet de déplacer le modèle dans la scène 3D, et ses déplacements n'affecterons plus du tout le curseur à l'écran. Une fois un objet sélectionné (ou la scène entière si aucun objet n'est sélectionné) en cliquant (fonctionnement similaire à celui du mode souris), on pourra choisir de faire des translation et/ou rotation de la sélection.

1. Translation d'objets et de la caméra

Une fois un objet sélectionné puis le choix de la translation comme transformation, il suffit de déplacer le cube pour que les transformations dans le monde réel soit appliquées à l'objet dans la scène 3D. Aussi ne pas sélectionner d'objets permet d'appliquer les transformations à la caméra de la scène et modifier son point de vue.

2. Rotation d'objets et de la caméra

Une fois un objet sélectionné puis le choix de la rotation comme transformation, il suffit de faire tourner le cube pour que les transformations dans le monde réel soit appliquées à l'objet dans la scène 3D. Aussi ne pas sélectionner d'objets permet de faire tourner la caméra de manière. Faire tourner le cube fait tourner la caméra exactement comme si toute la scène subissait la même rotation.

III. Problèmes rencontrés et solutions

1 - Le choix du cube et la disposition des caméras d'Optitrack

La disposition des caméras du système optitrack peut influencer la performance de détecter le rigidbody du cube. Pendant ce projet, nous avons changé deux fois la disposition de caméra de optitrack. Initialement, la distance entre le bureau et les caméras était trop grande pour pouvoir traquer les cibles, il a donc fallu imprimer un nouveau cube avec de plus grandes cibles.

2 - Mode souris

Quand nous avons testé l'utilisation des coordonnées 3D d'Optitrack pour calculer des coordonnées 2D pour bouger la souris, le premier problème a été la conversion. En effet, les translations du cube en mode souris se font selon les axes +x et -z et celles du pointeur de la souris selon +x et +y, ainsi il a fallu faire correspondre les axes.

De plus, il a fallu jouer sur l'échelle des mouvements, ceux réalisés par l'utilisateur et envoyés par optitrack ne sont pas à la même échelle que ceux souhaités pour la souris. Cela revient à changer la sensibilité du système.

3 - Mode Translation

La difficulté principale pour la translation était de s'assurer d'effectuer les translations dans les bons repères et selon les axes voulus avec la bonne échelle.

Pour la translation des objets, nous voulions que l'utilisateur ait l'impression de manipuler l'objet comme si la vision qu'il a de la scène était la sienne. Il fallait donc s'assurer qu'une translation vers la droite dans le monde réel résulte en une translation vers la droite à l'écran. Cela signifie donc qu'il fallait appliquer les translations selon le référentiel de la caméra de la scène et non pas selon le référentiel de l'objet ou référentiel monde.

Il a fallu faire la même chose pour la translation de la caméra en utilisant bien son référentiel.

Aussi pour l'application des translations, cela fonctionne en appliquant en permanence des petites translations qui correspondent aux différences de position entre l'instant précédent et l'instant actuel.

4 - Mode Rotation

La principale difficulté a été la manipulation de quaternions et notamment comment calculer les différences de rotations entre chaque instant et donc les petites rotations à effectuer à chaque instant. La solution retenue a donc été de retenir le quaternion initial du cube à l'instant ou la rotation commence puis ensuite à chaque instant de calculer la rotation à effectuer avec le delta_quaternion égal à l'inverse du quaternion initial multiplié par le quaternion actuel associé au cube.

5 - Problèmes généraux

Dans toutes les applications précédentes, une difficulté récurrentes était que l'application en continu de petites transformations résultait en une accumulation de bruit et par exemple la souris pouvait avoir tendance à bouger toute seule vers le coin en haut à gauche même quand le cube était immobile. Pour contrer ce phénomène au maximum nous avons fait en sorte qu'en cas de transformations bien trop faibles (déplacement ou angles en dessous d'un certain threshold) elles ne soient tout simplement pas appliquées, cela nous a permis de gagner en stabilité, cependant, il est aussi important de conserver une partie des petit déplacement pour que l'utilisateur puisse faire des petits ajustement. Un autre problème rencontré est lié à la rotation, en effet la manipulation des quaternions par Optitrack et par unity n'est pas la même, ce qui nous conduit à faire des transformations qui conduisent aussi à l'ajout de bruit et d'imprécision lors de celle-ci.

IV. état des lieux

Tout d'abord pour utiliser notre système il faut installer le set up Optitrack une première fois et le calibrer dans le logiciel Motive.

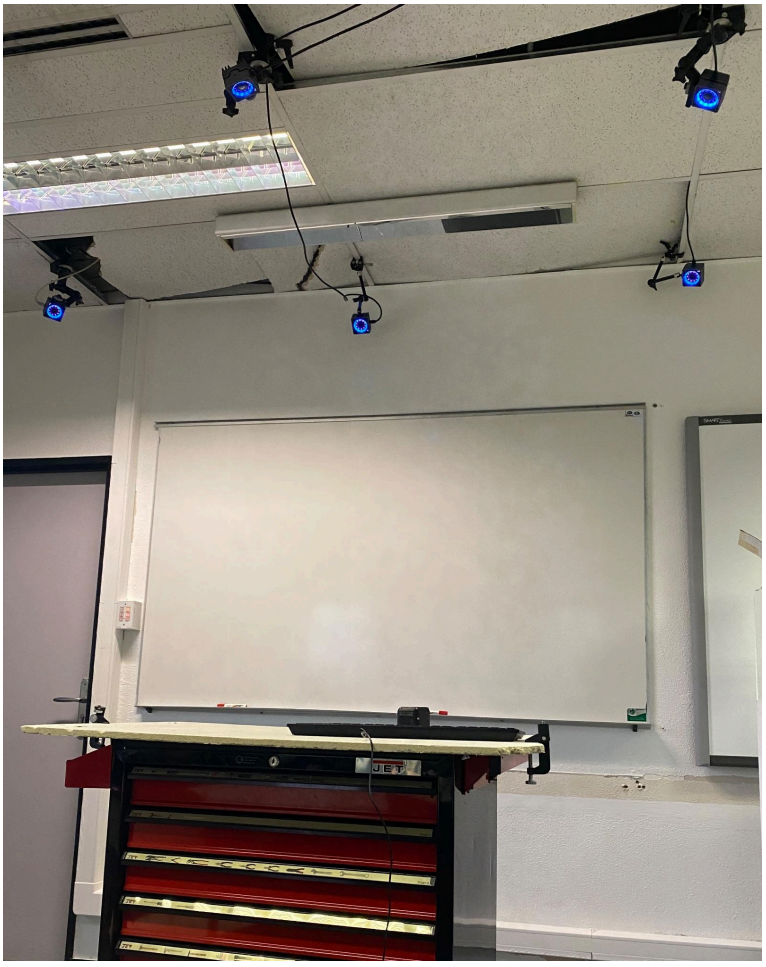


Figure 2 : set up de optitrack, clavier et cube

Un premier programme en C++ se charge à la fois:

- De lire les paquets envoyés par Optitrack, en déduire la hauteur du cube pour déterminer le mode d'utilisation du cube.
- D'appliquer les déplacements du curseur et de traiter les informations relatives au clavier grâce aux fonctions de la bibliothèque "Windows.h".
- De communiquer à un second fichier en C# les informations nécessaires pour appliquer les transformations 3D.

Ce second fichier en C# est un programme qui est utilisé par Unity pour appliquer des transformations sur les objets de la scène. Ce programme permet de manipuler d'interroger Unity sur la scène actuelle (les informations sur les objets de la scène, les sélections, les différents onglets de scène et leurs propres caméras, les autres caméras etc…) Une fois qu'il reçoit les informations du programme précédent, ce

second programme applique les transformations aux bons objets avec les bons paramètres et dans les référentiels adaptés.

La communication entre les deux se fait de la manière suivante: Le programme C++ est tourné en librairie DLL et sa fonction principale (qui traite les infos d'Optitrack) est appelée par le programme C#.

Après l'ouverture de Unity et et le lancement du programme "CubeMouse" dans le menu "Window", l'utilisateur prend directement le contrôle du curseur avec le cube si le il est posé contre le plan du bureau. Les clics peuvent être faits en appuyant sur la touche **V** du clavier.

Une fois que l'utilisateur sélectionne un objet dans la scène, il peut déplacer cet objet. Il lui suffit d'appuyer directement sur **shift+T** pour activer le mode translation ou **shift + R** pour activer le mode de rotation.

Démonstration vidéo pour la translation:
https://drive.google.com/file/d/1Ye60wS7szC-UGFMN1MwDyINfUYqUvIe4/view?usp=sharing

Démonstration vidéo pour la rotation:
https://drive.google.com/file/d/18Zb-ap44RhuT2JbShuM8r_BlwxU2wzVW/view?usp=sharing

V. évaluation de l'interaction

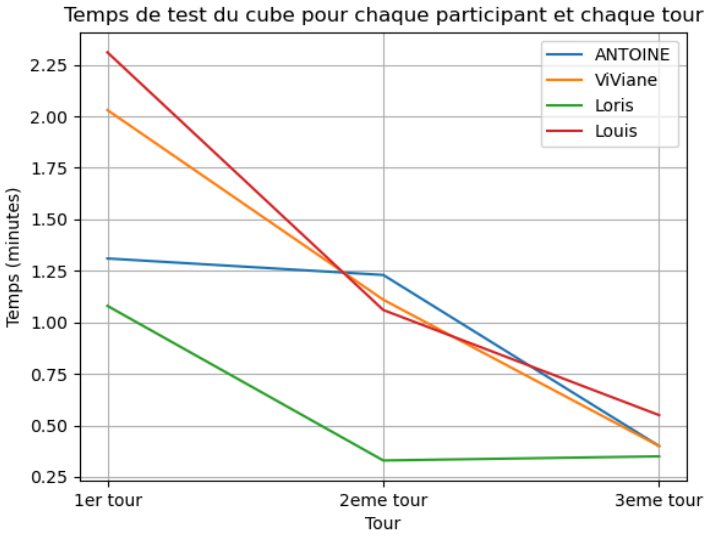
Nous n'avons pas pu remplir tous les objectifs notamment l'implémentation des clics mais l'interaction qui nous intéresse principalement était la manipulation du cube dans l'espace et nous avons donc pu créer un test pour pouvoir comparer les utilisations de la souris traditionnelle et du cube dans la manipulation d'objet 3D.

1 - Test effectué

À cause des imprécisions du cube en rotation, notre test n'est effectué que sur la translation. Le but de l'utilisateur est de placer trois cubes à leur position attendue en partant de positions prédéfinies. Les différents participants (certains ayant travaillé sur le projet et d'autres non) ont effectué plusieurs fois le test à la souris classique puis avec le cube et nous avons retenu les temps pour atteindre l'objectif. (Les cubes sont verts si correctement placés et rouge sinon)

(https://drive.google.com/file/d/1Ye60wS7szC-UGFMN1MwDyINfUYqUvIe4/view?usp=sharing).

Pour les participants n'ayant pas travaillé sur le projet, les meilleurs temps ont donné les résultats suivants:



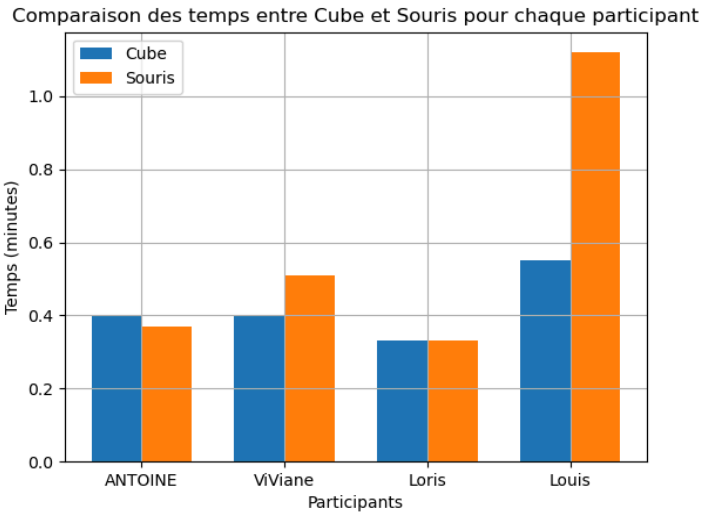
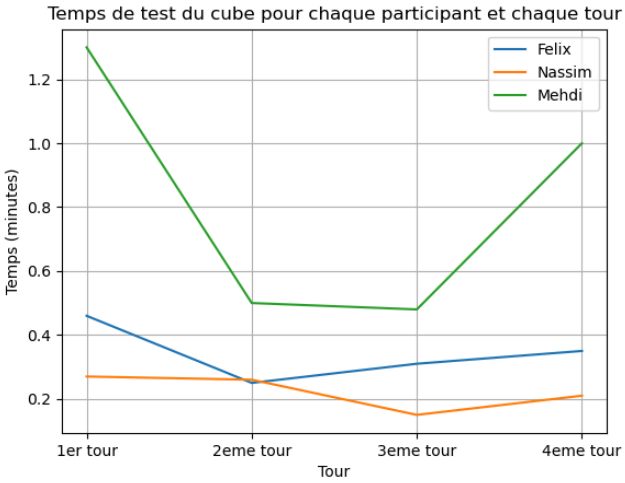


Figure 3 : temps de test du cube pour chaque participant et chaque tour
Figure 4 : Comparaison des temps entre Cube et Souris pour chaque participant

Pour les membres du projet ayant testé participé au test:



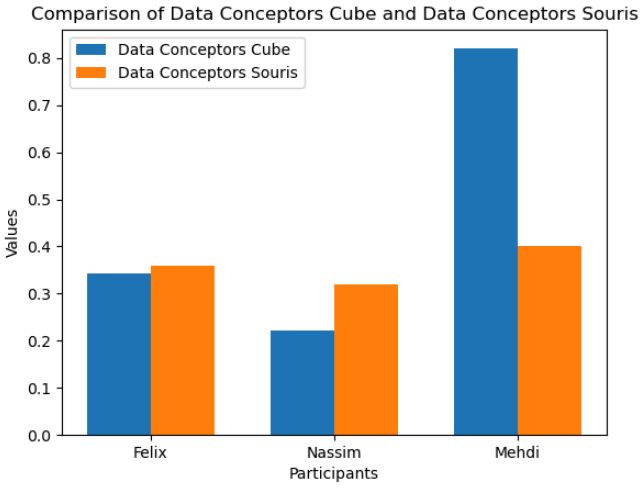


Figure 5: Temps de test du cube pour chaque participant et chaque tour
Figure 6: Comparison of Data Conceptors Cube and Data Conceptors Souris

Il faut prendre en compte la distance entre l'écran et le set up de manipulation du cube était très grande (environ 5 mètres) car nous ne pouvions pas rapprocher les caméras des ordinateurs en communication avec Optitrack, il est raisonnable de considérer que tous les participants ont été handicapés dans l'utilisation du cube car trop loin de l'écran et ce phénomène s'est d'autant plus fait ressentir par les personnes avec des problèmes de vue.

En termes de ressentis, il ressort que l'ensemble des participants préféraient la manipulation du cube plutôt que l'utilisation de la souris pour la tâche test.

2 - Éventuels tests supplémentaires

Pour tester l'ensemble des interactions nous aurions pu effectuer d'autres tests:

- Tester la rotation (après correction des imprécisions du programme) des objets en demandant d'effectuer le même test que précédemment mais avec des objets 3D ayant subi des rotations de départ.

- Tester la translation et la rotation de la caméra en demandant aux participants de retrouver le point de vue d'une scène à partir d'une image donnée.

- Tester le mode souris 2D uniquement en demandant aux participants de cliquer sur des cibles à l'écran avec le curseur.

VI. Conclusion

Malgrés les imperfection du prototype (vibration de la souris, réglage de la sensibilité approximatif, threshold peu calibré), les résultats semble indiquer que le prototype permet un contrôle aussi efficace qu'avec une souris traditionnelle, ainsi il est raisonnable de penser qu'avec des améliorations pour pallier au problème actuel, le projet remplit bien son rôle et permettrait bien un contrôle meilleur des logiciels 3D tel que Unity. Pour améliorer la performance, nous pourrions:

- Trouver une meilleure disposition des caméras d'Optitrack.

- Ajuster les sensibilités de la souris 2D et des translations 3D.

- Ajouter des cibles aux doigts des utilisateurs pour pouvoir simuler le click et la molette de la souris et complètement remplacer la souris.
  Il serait alors possible de choisir entre translation et rotations avec des mouvements des doigts plutôt qu'avec des raccourcis claviers (même s'il est probable que cela ne constitue pas une amélioration notable).

- Trouver une autre façon de mettre les cibles sur le cube afin d'éviter de les endommager à cause des frottements contre le bureau.

- Implémenter d'autres plugins pour pouvoir utiliser plus d'applications comme Maya, Meshlab, Blender, etc..