Virtual Squash 2

Réalisé par : Samuel Bétrisey

Professeur : Thibault Schönmann

1 Table des matières

2 Cahier des charges

2.1 Description générale

Amélioration d'un jeu précédemment développé dans l'école. Il s'agit d'un jeu de squash qui sera jouable à deux grâce à la Kinect. Chaque joueur utilise sa main comme raquette et dois faire rebondir la balle avant qu'elle ne touche le mur derrière lui.

J'essaierai pendant une journée de voir si j'arrive à créer le jeu dans le navigateur en JavaScript. Si j'arrive sans trop de problèmes, je continuerai comme ça sinon le jeu sera développé en Java en utilisant la bibliothèque KinectPV2.

2.2 Infrastructure

2.2.1 Hardware

- Un ordinateur avec Windows 8 ou plus récent
- Un capteur Kinect 2.0
- Un adaptateur Kinect pour Windows

2.2.2 Software

- Java 8
- IntelliJ IDEA 15 (https://www.jetbrains.com/idea/), licence gratuite pour les étudiants (IDE Java)
- KinectPV2 0.7.5 (https://github.com/ThomasLengeling/KinectPV2), open source (licence MIT)
    - Librairie Java permettant de récupérer le squelette depuis la Kinect
    - Intègre aussi Processing pour l'affichage graphique en temps réel.
- Git pour gérer le code aussi l'envoyer sur GitHub (https://github.com/betrisey/Virtual-Squash)

2.2.3 Finance

- Un capteur Kinect 2.0 149.00 CHF
- Un adaptateur Kinect pour Windows 59.00 CHF

Total : 208.00 CHF

2.3 Détails sur la réalisation

2.3.1 Besoins du client

- Créer un squash jouable à la Kinect en se basant sur le projet crée il y a 2 ans
    - Simulation de la balle qui rebondit sur les murs, le plafond, le sol et au contact de la raquette
- Corriger un bug qui rendait la raquette beaucoup trop grande
    - La taille de la raquette doit correspondre à celle de la main.
- Doit fonctionner avec la Kinect V2
- Possibilité de jouer à 1 ou 2 joueurs
    - A 1 joueur compter le nombre de rebonds, augmentation de la vitesse de la balle tous les 10 rebonds
    - A 2 joueurs compter un point à chaque faute de l'adversaire, jusqu'à 21 avec 2 points d'écart

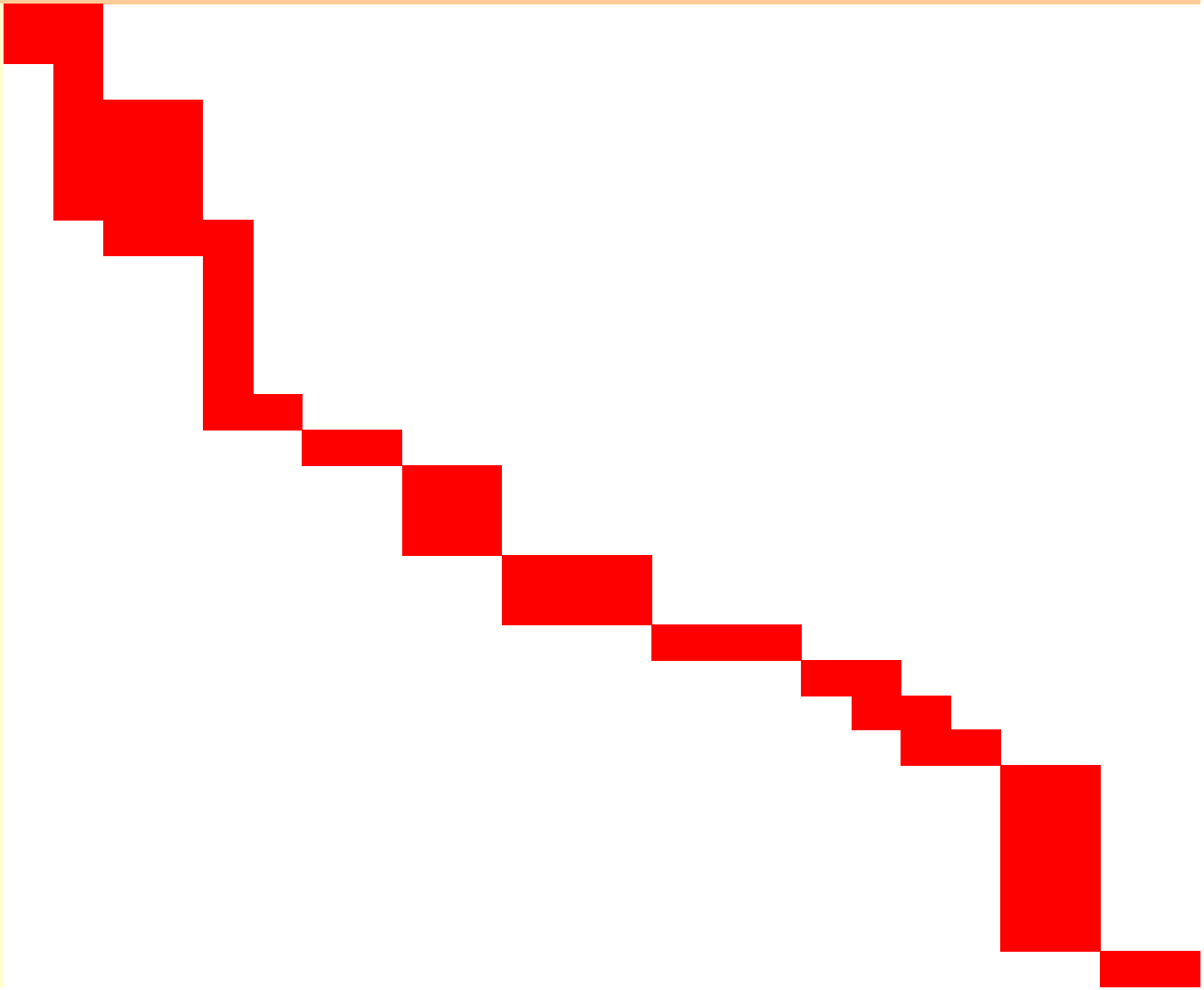
2.4 Contraintes et risques

- Pour l'instant je n'ai pas encore reçu le nouveau modèle de la Kinect donc je ne l'ai pas encore testé. Mais il devrait être compatible avec KinectPV2.
- J'aimerai essayer de faire le jeu dans le navigateur mais si c'est trop compliqué, je le ferai en Java.

3 Planification

3.1 Initiale

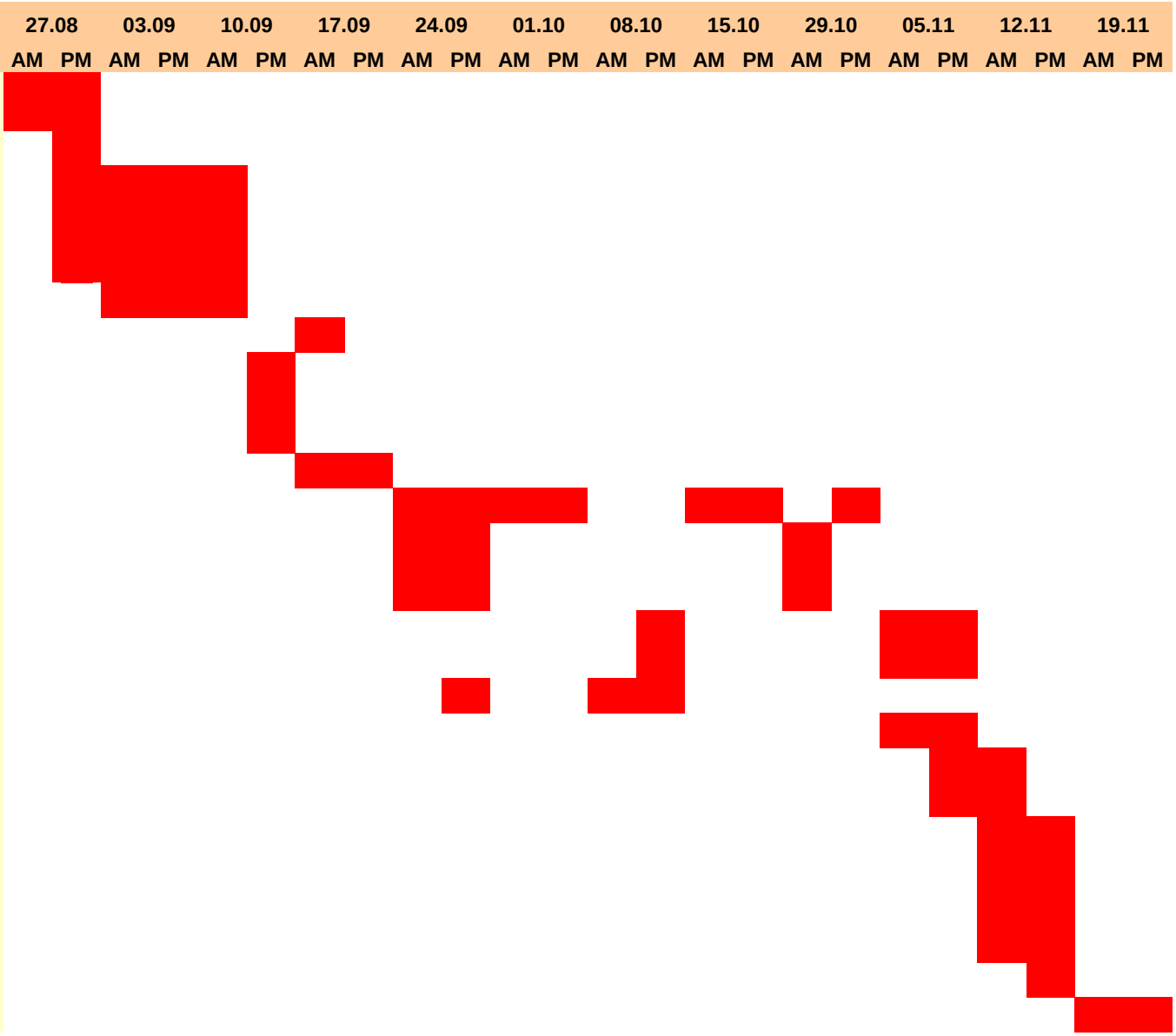
Méta-tâches	Tâches
Cahier des charges	Détails sur les tâches à réaliser
Planification	Planifier le travail
Recherche	Recherche sur les différentes possibilités de réalisation du projet, information sur les librairies utilisées
Analyse du code	
Réception de la Kinect	
Test	Déterminer si le jeu pourrait être développé dans le navigateur en JavaScript
Développement	Suivi de la main
	Affichage de la raquette
	Gestion des déplacements de la balle en se basant sur le code existant
	Faire rebondir la balle avec la main dans la bonne direction
	Deuxième joueur
Gestion de la partie	Scores, tour des joueurs
	Ecran affichant le gagnant
Finalisation	Design
	Ajout des dernières fonctionnalités, test et correction d'éventuels bugs
Documentation	Faire une documentation complète
Préparation	Préparer la présentation
Présentation	Présentation du projet



Sion, le 12.11.2015
Projet Virtual Squash 2

3.2 Réelle

Méta-tâches	Tâches
Cahier des charges	Détails sur les tâches à réaliser
Planification	Planifier le travail
Recherche	Recherche sur les différentes possibilités de réalisation du projet, information sur les librairies utilisées
Analyse du code	
Réception de la Kinect	
Test	Déterminer si le jeu pourrait être développé dans le navigateur en JavaScript
Développement	Suivi de la main
	Affichage de la raquette
	Gestion des déplacements de la balle en se basant sur le code existant
	Faire rebondir la balle avec la main dans la bonne direction
	Deuxième joueur
Gestion de la partie	Scores, tour des joueurs
	Ecran affichant le gagnant
Finalisation	Design
	Ajout des dernières fonctionnalités, test et correction d'éventuels bugs
Documentation	Faire une documentation complète
Préparation	Préparer la présentation
Présentation	Présentation du projet



3.3 Remarque sur la planification

La livraison de la Kinect a eu une semaine de retard, donc j'ai pris un peu de retard sur la planification initiale et je n'ai pas essayé de développer le jeu en JavaScript dans le navigateur. J'ai tout de même fais quelques recherches.

Une autre tâche qui a pris un peu plus long que prévu est l'affichage de la raquette. J'ai passé beaucoup de temps pour afficher la raquette correctement. (Trigonométrie pour calculer les angles de la raquette)

Malgré ces contretemps, j'ai réussi à remplir tous les objectifs du cahier des charges.

4 Documentation

4.1 Organisation du code

4.1.1 Classe Main

Classe de la fenêtre principale. Elle fait le lien entre la Kinect et les classes Player, Ball et Game. En cas d'utilisation d'un autre appareil, il y aura seulement cette classe à modifier.

- Mets à jour les positions des joueurs (classe *Player*)
- Instancie les classes *Ball* et *Game*
- Affiche l'interface graphique (scores, vainqueur, bouton recommencer)

4.1.2 Classe Player

- Stocke les positions du joueur et son score
- Affiche la raquette
- Calcule la direction de la frappe (détaillé dans Simulation des rebonds – raquette)

4.1.3 Classe Balle

Permet d'afficher une balle qui rebondit contre les murs.

On peut la faire rebondir dans une direction avec la méthode *bounce* et la faire accélérer avec la méthode *accelerate*.

On peut changer sa couleur (attribut *color*).

4.1.4 Classe Game

- Gère les tours des joueurs
- Calcule des scores
- Vérifie si le joueur frappe la balle et fait bouger la balle en conséquence
- Désigne le vainqueur et si la partie est finie
- Choisi le mode de jeu en fonction du nombre de joueurs détectés

4.2 Fonctionnalités

4.2.1 Multi-Joueurs

1 joueur : Gagne 1 point à chaque rebond de la balle. S'il n'arrive pas à la renvoyer assez vite (voir timeout) son score est remis à zéro. Son meilleur score est affiché.

2 joueurs : Les joueurs font rebondir la balle l'un après l'autre. Si un joueur n'arrive pas à la frapper à son tour, son adversaire marque 1 point. La partie se termine à 21 points avec 2 points d'écart.

Plus de 2 joueurs : Ce sont les mêmes règles que pour 2 joueurs. Pour activer ce mode, il faudra augmenter la constante limite de joueurs (NOMBRE_JOUEURS).

Le jeu passe automatiquement d'un mode à l'autre en fonction du nombre de joueurs détectés.

4.2.2 Accélération de la balle

Après 10 points en solo, la balle accélère de 20%.

4.2.3 Simulation des rebonds – murs et fond

Lorsque la balle touche un bord de l'écran ou le fond de la salle, elle rebondit dans la direction inverse (Par exemple, si elle atteint la position maximale sur l'axe *z*, son vecteur de déplacement *z* est inversé).

C'est la seule partie pour laquelle j'ai gardé pas mal de code du projet fait il y a 2 ans.

4.2.4 Simulation des rebonds – raquette

Lorsque le joueur frappe la balle, elle part dans la direction du mouvement de sa main.

Pour donner un effet de rebond plus réaliste, j'ai calculé la direction de la frappe avec les 15 dernières positions, normalisé ce vecteur et multiplié par la norme du vecteur du déplacement de la balle pour conserver la vitesse de la balle.

Donc la balle prend la direction de la frappe tout en conservant sa vitesse.

4.2.5 Taille de la raquette

La raquette a une taille de base (définie par les constantes WIDTH et HEIGHT) ensuite elle varie lorsque la main s'approche ou s'éloigne de la Kinect.

4.2.6 Gaucher ou droitier

Au début de la partie, le joueur met en avant la main qu'il veut utiliser. La raquette sera dans cette main tout au long de la partie.

4.2.7 Couleur des raquettes et de la balle

Chaque joueur a une raquette de couleur différente. Lorsque c'est à son tour de frapper la balle, cette dernière devient de la même couleur que la balle.

4.2.8 Bouton recommencer

A la fin de la partie, un bouton recommencer est affiché et le joueur peut facilement le sélectionner avec sa main au lieu de devoir appuyer sur une touche du clavier.

4.2.9 Affichage de la raquette

L'inclinaison du bras est reproduite par la raquette dans le jeu.

4.3 Réglages de constantes possibles

4.3.1 LONGUEUR_ECRAN et LARGEUR_ECRAN (Main.java)

Résolution de l'écran

4.3.2 NOMBRE_JOUEURS (Main.java)

Permet de limiter le nombre de joueurs

Modes de jeu selon Multi-Joueurs

4.3.3 FACTEUR_HORIZONTAL et FACTEUR_VERTICAL (Main.java)

Utilisés lors de la conversion des données de la Kinect (mètres → pixels).

Pour que la main soit alignée à la raquette virtuelle, il faut faire varier ces valeurs qui dépendent de la taille de l'écran et de la distance du joueur.

4.3.4 TIMEOUT (Game.java)

Temps que le joueur a pour frapper la balle lorsqu'elle arrive vers lui.

4.3.5 WIDTH et HEIGHT (Player.java)

Taille de la raquette à 1 mètre. Elle variera automatiquement si la main se rapproche ou s'éloigne.

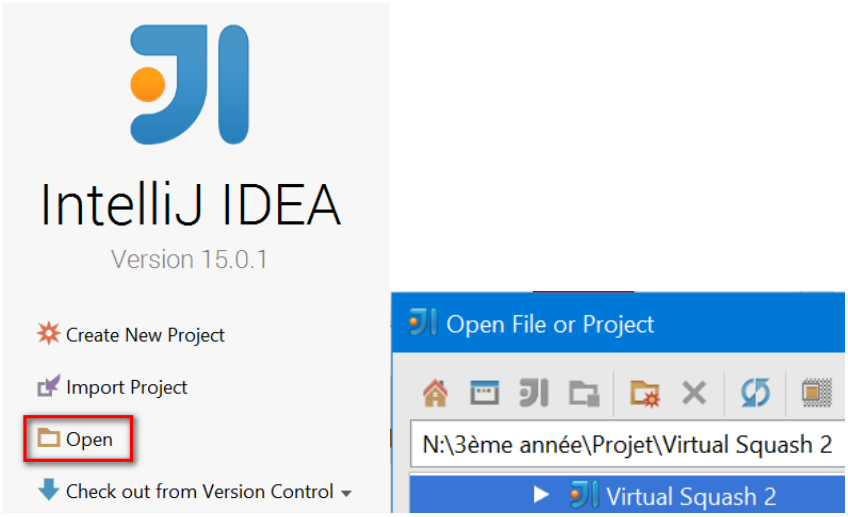
4.4 Installation du projet

Installer les SDK Kinect (2.0+) https://dev.windows.com/en-us/kinect

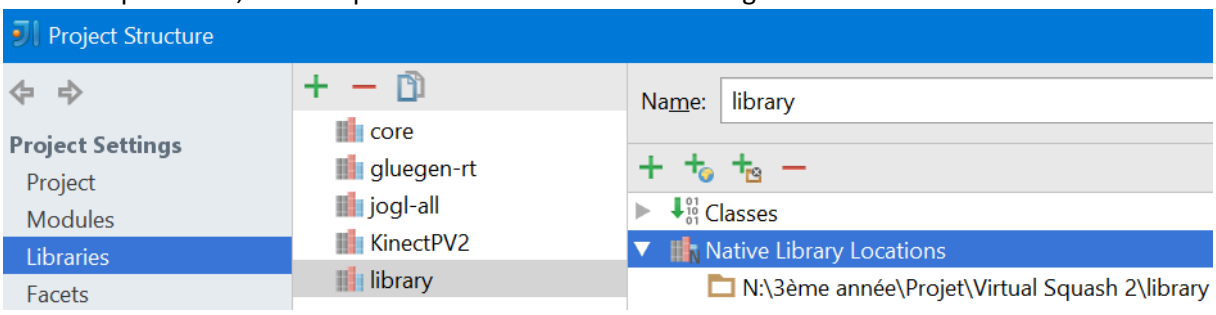
Installer Java JDK 8+ http://www.oracle.com/technetwork/java/javase/downloads/index.html

4.4.1 IntelliJ

Le projet et ses dépendances sont automatiquement détectés.

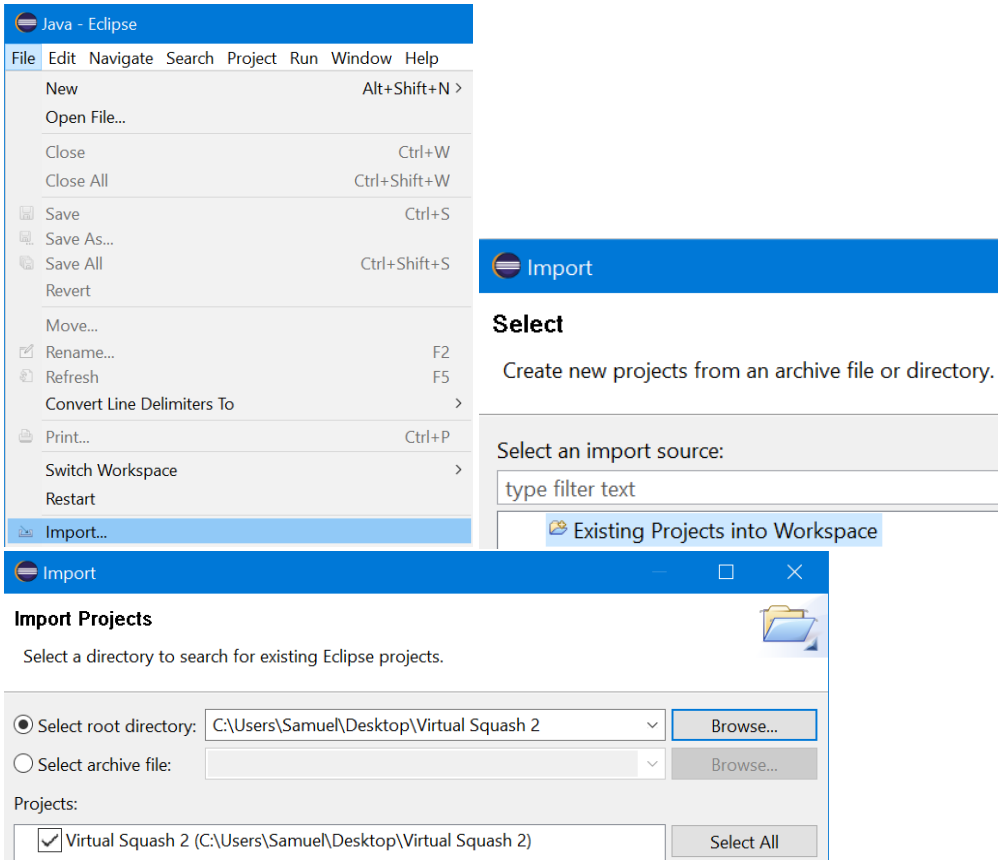


En cas de problème, vérifiez que ces librairies soient bien chargées.

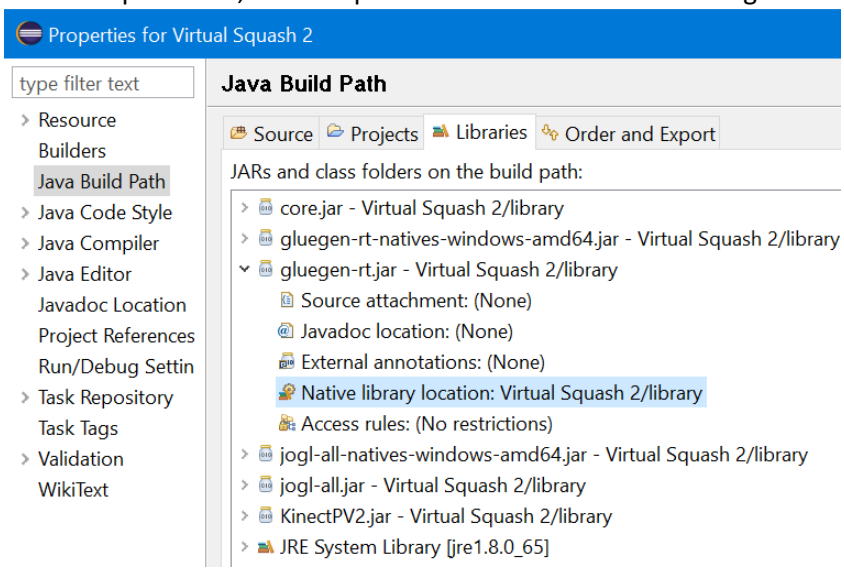


4.4.2 Eclipse

Le projet a été développé avec IntelliJ mais j'ai créé les fichiers nécessaires pour qu'il soit importable directement dans Eclipse.



En cas de problème, vérifiez que ces librairies soient bien chargées.



5 Tests

5.1 Importation et exécution du projet sur plusieurs plateformes

Le projet a été testé sur Windows 8.1 et Windows 10 avec IntelliJ IDEA et Eclipse. L'importation sur chaque plateforme se passe sans problème, les librairies (jar et dll) sont automatiquement chargées.

5.2 Fonctionnalités

Chaque fonctionnalité a été testée lors de son implémentation et à la fin le déroulement complet d'une partie.

Toutes les fonctionnalités décrites dans ce rapport fonctionnent comme prévu.

6 Améliorations possibles

6.1 Multi-joueurs en réseau

Un mode multi-joueurs en réseau serait tout à fait réalisable avec quelques jours de plus et une deuxième Kinect.

Connexion socket entre 2 PCs :

- Le PC « serveur » calcule les rebonds, scores et un joueur joue sur celui-ci
- Le PC « client » envoie la position de son joueur au serveur (objet Player sérialisé) et récupère la position de la balle, le score etc…

6.2 Utilisation d'un moteur 3D et physique

Pour une simulation plus réaliste des rebonds, de la gravité et de la profondeur

Par exemple avec Unity (https://unity3d.com/) ou Unreal Engine (https://www.unrealengine.com/), tous deux gratuits pour notre type d'usage.

7 Évaluation

7.1 Objectifs

Les objectifs fixés dans le cahier est charges ont été atteints.

J'ai pu rajouter des fonctionnalités en plus que je trouvais utiles tel que la détection gaucher/droitier, la couleur de la balle et des raquettes, la taille de la balle (perspective), un bouton recommencer accessible avec la main et la reproduction de l'orientation du bras.

7.2 Difficultés

Je passé beaucoup de temps pour reproduire la position du bras dans le jeu. Lors des calculs trigonométriques des angles du bras, il y a plusieurs cas à gérer. J'y suis finalement arrivé et j'ai ajouté des schéma pour aider la compréhension du code. Voici le code qui calcule un de ces angles :

```
// Calcul de l'angle de la raquette (angle coude-main)
float angleX; // α sur les schémas
/* Si la main se trouve plus en avant et plus haut que le coude
+-------------------------------------------------------+
|                                                       |
|              hand                                     |
|                X                                      |
|               /|                                      |
|              / |                                      |
|             /  |                                      |
|            /   |                                      |
|           /    |                           Kinect     |
|          / \α  |                            +-+       |
|         X------+                            +-+       |
|       elbow                                           |
|                                                       |
| <------|------|-----------------------------|---------- |
| Z     1.2     1                               0       |
|                                                       |
+-------------------------------------------------------*/
```

```
if (hand.getZ() < elbow.getZ() && hand.getY() < elbow.getY())
    angleX = (float) Math.atan((hand.getY() - elbow.getY()) / (elbow.getZ() -
hand.getZ()));

/* Si la main se trouve plus en avant et plus bas que le coude
+-------------------------------------------------------+
|                                                       |
|      elbow                                            |
|        X------+                                       |
|         \ /α  |                                       |
|          \    |                                       |
|           \   |                                       |
|            \  |                                       |
|             \ |                            Kinect     |
|              \|                             +-+       |
|               X                             +-+       |
|             hand                                      |
|                                                       |
| <------|------|---------------------------|---------- |
| Z     1.2     1                            0          |
|                                                       |
+-------------------------------------------------------*/
else if (hand.getZ() < elbow.getZ() && hand.getY() > elbow.getY())
    angleX = (float) -Math.atan((elbow.getY() - hand.getY()) / (elbow.getZ() -
hand.getZ()));
/* Si la main se trouve plus en arrière et plus haut que le coude
+-------------------------------------------------------+
|      hand                                             |
|        X                                              |
|        |\                                             |
|        | \                                            |
|        |  \                                           |
|        |   \                                          |
|        |    \                           Kinect        |
|        |  B/ \ \α=180-B                  +-+          |
|        +------X                          +-+          |
|             elbow                                     |
|                                                       |
| <-----------------------------------------------+     |
| Z     1.2     1                            0          |
|                                                       |
+-------------------------------------------------------*/
else if(hand.getZ() > elbow.getZ() && hand.getY() > elbow.getY())
    angleX = (float) (Math.PI - Math.atan((hand.getY() - elbow.getY()) /
(hand.getZ() - elbow.getZ())));
/* Si la main se trouve plus en arrière et plus bas que le coude
+-------------------------------------------------------+
|             elbow                                     |
|        +------X                                       |
|        |  B\ / /α=180-α                               |
|        |    /                                         |
|        |   /                                          |
|        |  /                                           |
|        | /                              Kinect        |
|        |/                                +-+          |
|        X                                 +-+          |
|      hand                                             |
|                                                       |
| <-----------------------------------------------+     |
| Z     1.2     1                            0          |
|                                                       |
+-------------------------------------------------------+
*/
else if (hand.getZ() > elbow.getZ() && hand.getY() < elbow.getY())
    angleX = (float)-(Math.PI - atan((elbow.getY() - hand.getY()) / (hand.getZ() -
elbow.getZ())));

angleX -= Math.PI / 2;
```

J'ai aussi eu un autre petit problème lorsque j'affichais le deuxième joueur, il n'apparaissait pas.

Pour le premier joueur, j'utilisais translate pour se positionner au centre de sa raquette et à la fin, je croyais revenir à la position (0, 0) avec translate(0, 0) mais en fait cette méthode faisait une translation relative donc un déplacement nul.

```
parent.translate(center.x, center.y); // Je me place au centre de la raquette
parent.rotateX(angleX);
parent.rotateZ(angleZ);
parent.fill(color);
parent.noStroke();
parent.box(width, height, 15);
parent.translate(0, 0); // Retour à la position initiale
```

Correction du problème : En utilisant pushMatrix et popMatrix, on se repositionne à l'emplacement initial.

```
parent.pushMatrix();
parent.translate(center.x, center.y);
parent.rotateX(angleX);
parent.rotateZ(angleZ);
parent.fill(color);
parent.noStroke();
parent.box(width, height, 15);
parent.popMatrix();
```

Au tout début, j'ai dû comprendre comment fonctionne l'ajout de librairies Java compilées (jar) et librairies natives Windows (dll).

8 Conclusion

J'avais peu eu d'expérience avec Java et jamais utilité Processing avant ce projet mais n'ai pas eu de problème à m'adapter puisque je me débrouille bien en plusieurs autres langages, la logique reste la même.

Utiliser Java pour ce projet m'a donné envie d'apprendre plus à ce sujet. Je suis en train de suivre un cours en ligne de l'EPFL sur la POO en Java. (https://www.coursera.org/course/intropoojava)

J'ai aussi trouvé très intéressant de programmer avec la Kinect, apprendre à traiter des données plus complexes que d'habitude lorsqu'on créer un site internet ou une application desktop.

Au cours de ce projet, j'ai appris à utiliser Git pour faire du versioning. C'est très pratique pour gérer son code, ajouter une fonctionnalité et revenir en arrière en cas de problème.

9 Références

Documentation Processing https://processing.org/reference/
IDE IntellliJ https://www.jetbrains.com/idea/
Git pour Windows https://git-scm.com/

10 Fichiers annexe

- Journal de bord
- Code source (aussi disponible ici https://github.com/betrisey/Virtual-Squash)