Point d'avancement

Projet informatique individuel : DataVRiz

Pour rendre compte des fonctionnalités implémentées et futures de DataVRiz, j'utiliserai les tableaux de la version 2 du cahier des charges, améliorée grâce aux retours de mon tuteur et mise à jour selon les dernières recherches et expériences de développement sur le projet.

Fonctionnalités implémentées

Voici les fonctionnalitées présentes en ce jour dans l'application. Un test réussi signifie que la fonctionnalité est vérifiée au niveau utilisateur, sur le casque Oculus de l'ENSC et l'ordinateur associé, sur un build PC du projet Unity (export).

F1_DATA_V1	Les données affichées sont issues d'un fichier .csv ou .txt avec un séparateur spécifique enregistrant des données a 3 dimensions quantitatives et présent dans le répertoire de l'application.	Testé V1
F2_DATA_V1	Les données sont affichées sont la forme d'un nuage de points 3D centré réduit.	Testé V1
F1_CTRL_V1	L'utilisateur peut, par une action sur le contrôleur, quitter l'application.	En cours de dev.
F2_CTRL_V1	Lorsque l'utilisateur tourne la tête, sa projection dans l'environnement 3D tourne accordément le regard.	Testé V1
F3_CTRL_V1	L'utilisateur équipé d'un casque VR peut se déplacer dans l'environnement 3D virtuel.	En cours de dev.
F_PORT_V1	L'application fonctionne pendant plus de 5 minutes sur le matériel VR du hall technologique de l'école pour des données avec moins de 100 points.	Testé V1

Ayant décidé de travailler principalement le Vendredi après midi, je n'ai pas encore développé les 2 fonctionnalités de CTRL, mais ayant déjà regardé les parties de l'API correspondantes, la version 1 devrait pouvoir être livrée cette semaine : le risque de retard est faible.

Pour les fonctionnalités de PORT, un test réussi signifie que l'application tourne toujours à plus de 90 FPS, pour correspondre au taux de rafraîchissement de l'Oculus et ne pas favoriser la cybersickness.

Scripts Unity

La partie la plus importante du développement est la programmation des scripts pour l'application Unity. Actuellement, il y en a 5, correspondant à 5 classes, dont seulement 2 utilisent l'API Unity et sont placées directement sur des GameObjects.

StatsHelper

Cette classe contient des fonctions de stats pour pouvoir centrer-réduire les données (variance, moyenne).

DataLine

Cette classe permet de construire les objets contenant les données pour un individu du set de données (valeurs standardisées, valeurs initiales).

TxtReader

Cette classe permet de lire le fichier de données (.txt; .csv principalement) et d'en sortir une liste de *DataLine* exploitable.

PlotPoints

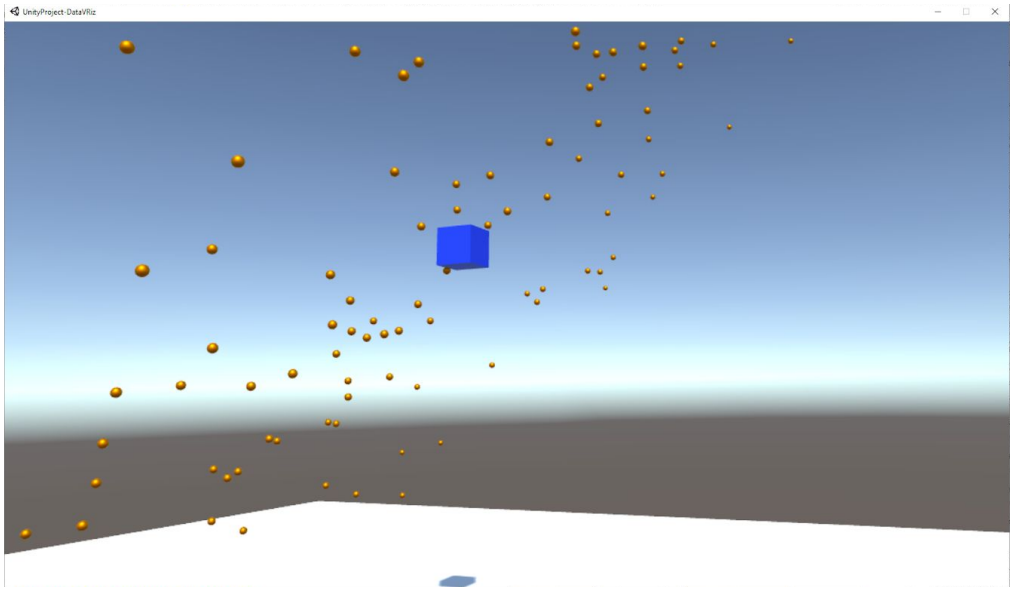
Cette classe Unity, qui s'exécute au lancement, permet l'affichage des données dans un nuage de points 3D centré réduit en utilisant toutes les classes ci-dessus.

InputManager

Cette classe Unity est prévue pour gérer les interactions du joueur (contrôleurs, déplacements).

Capture d'écran

Même si un screenshot ne permet pas très bien de rendre compte de la réalité virtuelle, voici une illustration de ce que l'application donne pour l'instant sur des données d'exemple. Ces données ont été bruitées et deux dimensions sont intentionnellement corrélées fortement.



Le cube bleu représente l'origine alors que chaque point orange est un individu, on voit ici apparaître le plan de corrélation prévu.

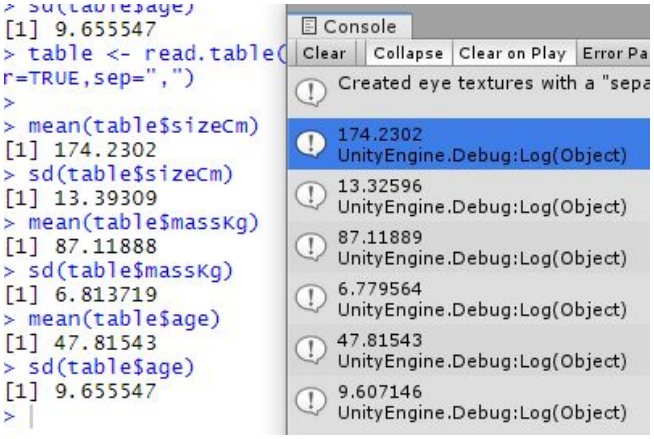
Fonctionnalités restantes à développer

Voici les fonctionnalités qu'il est prévu de développer, dans l'ordre de priorité, avec d'abord les deux dernières de la version 1.

Version 2		
F1_DATA_V2	Les données sont affichées avec des axes visibles en indiquant la signification et l'échelle.	
F_CTRL_V2	L'utilisateur peut pointer un point et ainsi obtenir l'affichage de ses coordonnées, de son écart à la moyenne, de ses valeurs dans les unités des données.	
F2_DATA_V2	L'application peut aussi exploiter des données à 2 dimensions.	
F_PORT_V2	L'application fonctionne pour des données de moins de 1000 points.	
Version 3		
F_DATA_V3	Les données affichées peuvent représenter une 4ème dimension qualitative (ou quantitative en tranche) par une coloration distincte et non graduée des points.	
F1_CTRL_V3	L'utilisateur peut, sans quitter l'application, lancer l'affichage de n'importe lequel fichier de données présent dans le répertoire. Il a un retour si le fichier n'est pas valide.	
F2_CTRL_V3	En activant un mode jeu, les points deviennent soumis à une gravité et sont poussables par l'utilisateur.	
F3_CTRL_V3	L'utilisateur peut choisir les variables à afficher et sur quels axes les placer.	
F_PORT_V3	L'application fonctionne sur au moins un autre dispositif matériel.	

A améliorer

J'ai généré les données de test avec R pour pouvoir facilement deviner à quoi ressemblerait leur nuage 3D mais aussi pour pouvoir comparer les précisions. Or, parce que Unity utilise des *float* (32 bits) pour ses coordonnées d'objets, on perd effectivement en précision, comme on le voit sur la capture d'écran ci-dessous.



Mais en gérant mieux les conversions de type, en utilisant des *double (64 bits)* partout où c'est possible, je devrais pouvoir récupérer de la précision.

Planning mis à jour

Comme j'avais oublié de ne pas compter la semaine de vacances, voilà ce que donne le nouveau planning, avec toujours la version 3 comme objectif :