Fi-Info5 – Bibliothèques de Développement Multimédia

Projet 2018-2019 – **Tetris 3D**

1 Consignes générales

1.1 Objectifs pédagogiques du projet

L'objectif est de renforcer et de mettre en pratique les compétences du bloc "Bibliothèques de développement multimédia" à travers le développement d'un petit projet. Le développement se fera en C++ sous l'environnement `QtCreator` en utilisant les bibliothèques Qt, OpenGL et OpenCV. A l'issue de ce projet vous devrez être capable de :

- Paramétrer l'environnement de développement pour utiliser des bibliothèques open source.
- Utiliser les principales fonctions des bibliothèques Qt, OpenGL et OpenCV.
- Gérer la création et la visualisation d'une scène 3D dynamique.
- Concevoir et réaliser une interface utilisateur conviviale.
- Mettre en place une interaction avec l'utilisateur à partir de l'acquisition et du traitement des images issues d'une WebCam.

1.2 Travail demandé

Le travail demandé concerne la conception et le développement d'un jeu de tetris 3D dans lequel les pièces glissent un plan incliné comme sur la figure 1.

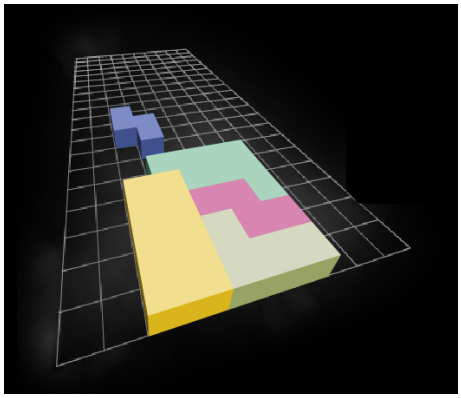

FIGURE 1 – Plateau de jeu du Tetris 3D

L'application est décrite plus en détails dans la partie 2 de ce document. Le développement devra être réalisé en C++. Vous devrez rendre un rapport au format PDF ainsi qu'une archive ZIP contenant le programme exécutable, les codes sources, le fichier projet et tous les fichiers nécessaires à la compilation.

Le rapport devra comporter :

- 1 page maximum de spécifications présentant l'interface utilisateur et décrivant les principales interactions possibles avec l'utilisateur,
- 2 pages maximum de conception présentant les principales classes que vous avez utilisées en précisant le rôle de ces classes et leurs relations (faire un diagramme des classes),
- 1 page maximum précisant l'état de finalisation de l'application : les fonctions qui ont été validées, celles qui ne sont pas finalisées, les bogues qui subsistent,...
- Les fichiers d'entête des classes (déclaration des classes et des fonctions) qui devra contenir des commentaires sur le rôle des champs et des méthodes utilisés (rôle de la méthode, paramètres d'entrée, de sortie, de retour et, le cas échéant, l'algorithme utilisé) ainsi que le nom de l'auteur.

Soyez clair et synthétique. Merci de respecter le nombre de pages.

1.3 Evaluation

La note finale du projet est collective et vient s'ajouter aux autres notes individuelles du module "Bibliothèque de développement multimédia". L'évaluation du projet est notée sur 20. Elle est calculée à partir des 3 éléments suivants :

— Evaluation du rapport (30%),
— Fonctionnement de l'application évalué pendant la démonstration (45%)
— Evaluation des fichiers sources et exécutable (25%),

Le fonctionnement de l'application sera évalué pour chaque binôme lors d'une séance de démonstration qui aura lieu le **10 avril matin** (horaire différent suivant les groupes). Lors de cette séance, il faudra présenter chacune des 8 fonctionnalités décrites dans le cahier des charges (cf. paragraphe 2.2).

1.4 Remise du projet

Le projet (rapport + sources + exécutable) est à remettre au plus tard le **9 avril** à 23:59:59. Il devra être déposé sur le portail MOOTSE, dans l'espace de dépôt correspondant à votre groupe de TD, le nom du fichier déposé comportant les noms des auteurs du projet et le nom du groupe de TD.

Ex: Nom1-Nom2-TDA.zip

1.5 Organisation

La réalisation du projet doit se faire impérativement en binômes. Lorsque le nombre d'étudiants d'un groupe de TD est impair, un étudiant doit travailler seul (on ne peut avoir qu'un seul étudiant travaillant seul). La constitution des binômes est laissée au choix des étudiants.

5 séances de TD sont prévues pour apporter une aide au développement du projet, mais il est indispensable de travailler également en dehors de ces séances. Les séances sont composées de :

— 2 séances de 3h pour la mise en place de l'interaction avec l'utilisateur par l'intermédiaire de la WebCam (OpenCV).
— 2 séances de 3h pour la mise en place des éléments d'interface graphique 3D (OpenGL).
— 1 séance de finalisation de l'application.

1.6 Pénalités

Si le projet est rendu en retard ou s'il manque des documents, une pénalité de 2 points par jour de retard sera appliquée (10% de la note).

2 Cahier des charges de l'application

2.1 Description générale

Vous devez réaliser un jeu de Tetris 3D qui sera contrôlé avec une WebCam. Dans cette version 3D, on reprend le même principe que le jeu original, mais avec un plateau et des pièces représentés en 3D. Au lieu d'être vertical, le plateau est incliné et les pièces glissent sur ce plan en s'approchant de l'utilisateur (cf. figure 1 et voir ce site [1]). Les différents éléments de la scène sont donc :

1. http://www.gomaths.ch/jeux/tetris3D.html

— Une grille de 10×20 cellules représentant le plateau de jeu représentée en perspective comme sur la figure 1.
— Des pièces de 7 types différents [2] apparaissent en haut du plateau et gissent sur le plan incliné pour venir s'empiler en bas du plateau. Ces pièces sont les tétriminios.
— Pendant la descente d'une pièce, l'utilisateur peut la déplacer latéralement et la faire tourner dans le plan du plateau tant qu'elle n'est pas arrivée en bas.
— Dès qu'une ligne est complétée, elle disparaît et les blocs placés au dessus glissent d'un rang. Le joueur marque alors 1 point.
— Si le plateau est rempli, le jeu s'arrête.

Les mouvements d'une pièce sont contrôlés par la WebCam en fonction des gestes effectués par le joueur. L'interface utilisateur devra comprendre une zone d'affichage de la zone de la WebCam utilisée pour détecter les gestes du joueur et une zone pour l'affichage du score.

2.2 Fonctionnalités à implémenter

Les fonctionnalités suivantes doivent être implémentées dans le projet :

1. Affichage en perspective du plateau constitué d'une grille de 10×20 cellules représentant le plateau.
2. Apparition d'un tétriminos et descente de ce tétriminos en glissant sur le plateau incliné.
3. Déplacement latéral du tétriminos commandé par un geste de rotation des mains du joueur vers la droite ou vers la gauche.
4. Rotation du tétriminos commandé lorsque les poings du joueur sont frappés l'un contre l'autre.
5. Arrêt du tétriminos lorsqu'il arrive en bas et apparition du tétriminos suivant en haut.
6. Détection d'une ligne remplie et incrémentation des points.
7. Disparition des blocs associés à la ligne remplie et descente des blocs situés au dessus.
8. Détection de la fin de la partie et ré-initialisation pour la partie suivante.

Les fonctionnalités suivantes constitueront un bonus sur la notation du projet :

— Ajout de niveaux de jeux correspondant à une accélération de la descente des tétriminos.
— Descente rapide de du tétriminos commandée par un geste de déplacement des 2 poings vers le bas.

2.3 Précisions sur les éléments graphiques 3D

Le jeu devra être dessiné avec OpenGL, selon les principes vus lors des TD.

2.4 Précisions sur l'architecture de votre projet

Une architecture de données adaptée devra être utilisée pour la gestion des objets. Il vous est conseillé de créer des objets tétriminos de différents types dérivés d'une classe virtuelle.

2.5 Précisions sur l'interaction avec la webcam

Il s'agit de mettre en place des fonctionnalités d'interaction entre l'utilisateur et l'application par l'intermédiaire d'une WebCam. L'interface proposera une zone d'interaction présentant une partie du champ de la caméra. L'utilisateur devra commander le déplacement d'un objet en réalisant des gestes en déplaçant ses deux mains devant la zone active de la caméra (figure 2).

2. https://fr.wikipedia.org/wiki/Tetris

Les gestes seront effectués les deux poings fermés l'un à coté de l'autre (cf. figure 2). En position neutre, les deux mains sont l'une à coté de l'autre à la même hauteur et séparées d'une quinzaine de centimètres. Trois gestes de base seront possibles :

1. déplacer à droite : dans un geste de rotation vers la droite, la main gauche monte alors que la main droite descend, le geste est actif tant que la main gauche est plus haute que la main droite,
2. déplacer à gauche : dans un geste de rotation vers la gauche, la main droite monte alors que la main gauche descend, le geste est actif tant que la main droite est plus haute que la main gauche,
3. remettre l'objet au centre : les deux poings se rapprochent jusqu'à rentrer en contact, l'action est réalisée dès que les poings se touchent.

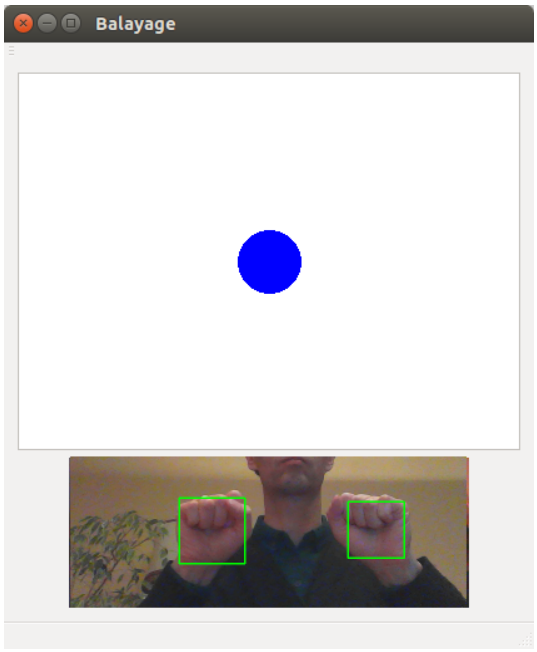

FIGURE 2 – Mise au point de l'interface par *WebCam*

Dans un premier temps, la gestion de l'interaction webcam/utilisateur devra se faire dans une fenêtre séparée de la fenêtre de rendu OpenGL.

Les étapes à franchir sont les suivantes :

1. Installer la bibliothèque OpenCV dans son environnement de développement.
2. Mettre en place une application pour la mise au point de l'interface. Cette application aura une zone de captation du geste et une zone de dessin de l'objet en mouvement (ici une sphère pour la mise au point). On pourra partir de l'exemple *TestWebCamQt* fourni.
3. Etudier la fonction `detectMultiScale` et son application à la détection de visage dans l'exemple *TestDetectMultiScale* donné sur Mootse et sur la page de documentation. Chercher sur internet un fichier de classifieur permettant de détecter les poings.
4. Ajouter la détection des gestes et la visualisation du mouvement de la sphère à votre application. On pourra passer par les sous-étapes suivantes :
5. Détection des gestes de déplacement à droite et à gauche.
6. Détection du geste pour remettre l'objet au centre.
7. Optimisation des paramètres pour augmenter la fiabilité et la fluidité (on pourra utiliser les fonctions de multi-threading de Qt comme `QtConcurrent::run` et la classe `QFutureWatcher` au besoin).