MAHREZ
Ali
GROUPE 3

Rapport de projet
Spirale d'Ulam en 3 dimensions

I- Introduction :

L'objectif de ce projet consiste à réaliser un modèle en 3D à partir de formes géométriques. J'ai alors choisi de créer une pyramide à partir de cubes, chacun ayant un numéro spécifique le caractérisant et une couleur.

Déroulement :

Au tout début du projet, j'ai créé une classe objet cube en lui donnant comme attribut la taille de chaque cube et son identifiant. Ensuite j'ai mis en place une méthode reliant les différents vertex représentant les côtés de chaque face du cube et correspondant à l'objet cube, puis des méthodes nécessaires à la spécification de ce dernier, notamment sd() qui calcule la somme des diviseurs de chaque cube par rapport à son identifiant et qui va servir par la suite à faire une classification des cubes en fonction de leur couleur.

Cependant, en arrivant à la dernière partie du projet (picking), je me suis aperçu que je ne pouvais pas manipuler les vertex et les fragments, ce qui m'empêchait de poursuivre le projet et ainsi manipuler les shaders. J'ai alors décidé de changer la structure des cubes en passant d'un codage par classe à une PShape, tout en gardant les méthodes qui m'ont permis de spécifier et caractériser chaque cube.

Mes choix :

1-Formes :

J'ai choisi de créer une pyramide. Je me suis alors inspiré de la spirale d'Ulam pour dessiner le modèle. En effet, j'ai tout d'abord commencé par dessiner l'ensemble de mes cubes sans donner un volume au modèle (c'est à dire en 2D) et pour cela il a fallu déclarer une variable correspondant au nombre de déplacements autorisés. Une fois le code en place, j'obtiens une représentation similaire à celle de la figure 1.

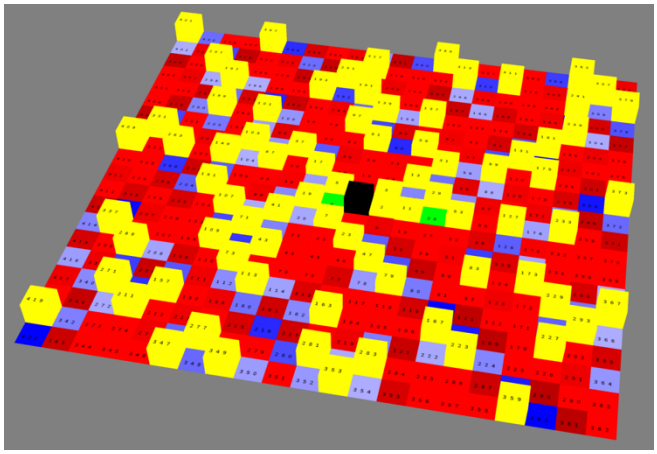

Figure 1 : spirale D'Ulam de l'énoncé avec des cubes

MAHREZ
Ali
GROUPE 3

Il a fallu ensuite rajouter une troisième dimension et bien placer les cubes pour ainsi donner du volume et créer la pyramide.

En partant du sommet de la pyramide, on peut constater qu'à partir du 2ème étage le nombre de cubes de l'étage suivant augmente de 4. En rajoutant un compteur et lorsqu'on arrive à un multiple de 4, le z augmente et on baisse d'un étage.

Enfin pour mieux placer les cubes dans la pyramide, on retrouve deux cas différents :

- Si le cube est aux extrémités de chaque ligne, il devra être placé de sorte à ce que le quart de ce cube soit caché par le cube du dessus.
- Si le cube n'est pas aux extrémités, il est placé en ligne avec les autres cubes et c'est sa moitié qui est cachée par le cube supérieur.

En suivant ce raisonnement, on obtient le résultat de la figure 2.

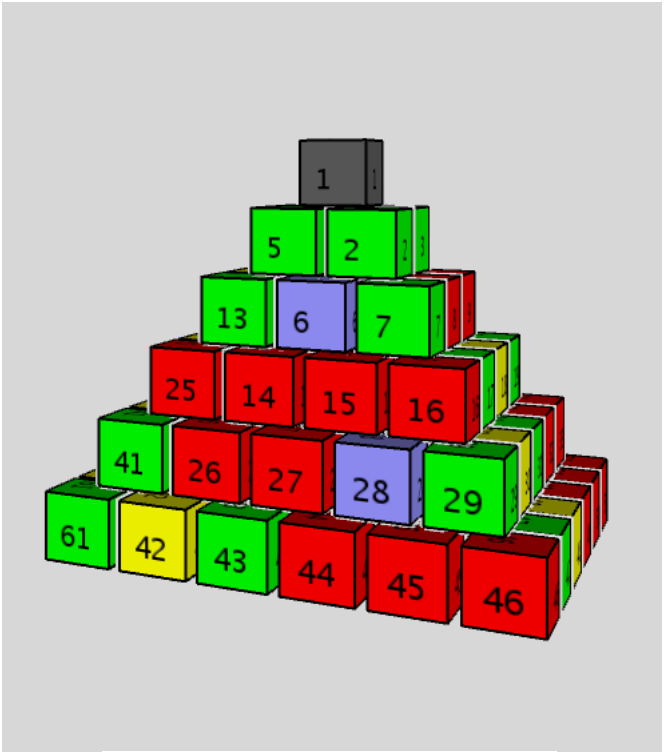

Figure 2 : Pyramide de cubes

2-Les textures :

Chaque cube est identifié par un numéro spécifique en utilisant un PGraphique. Pour cela j'ai créé une fonction prenant comme attribut un entier et retournant un PGraphique avec un background transparent pour laisser apparaître la couleur du cube et un texte au milieu représentant l'identifiant du cube. J'ai ensuite translate chaque PGraphique en face de chaque face du cube (tailleCube +1) pour qu'il soit positionné juste par-dessus la face du cube le représentant.

3- Les nombres :

J'ai ensuite créé 3 attributs (a, b, c) représentant les différents coefficients d'un polynôme, et un fonction pour calculer le nombre de cubes appartenant au polynôme. De cette manière on ne peut afficher que les cubes dont les identifiants sont divisibles par le polynôme (ex : comme le cube numéro 19 est divisible par 6x +1, alors il est affiché). Il ne restait plus qu'à afficher le polynôme sur la Frame, pour cela j'ai dessiné un texte représentant les différents coefficients du polynôme et rajouté des rectangles qui en cliquant dessus, permettent de changer la valeur du coefficient

du nombre. Grâce à la fonction prédéfinie mouseCliqued() si la position cliquée X et Y de la souris est à l'intérieur du rectangle, alors le coefficient correspondant change.

4- L'animation :

En ce qui concerne l'animation, j'ai utilisé une fonction rotate pour chaque pyramide ce qui me permet de parcourir tous les cubes.
N.B: J'ai aussi rajouté la possibilité de parcourir la pyramide en déplaçant la souris.

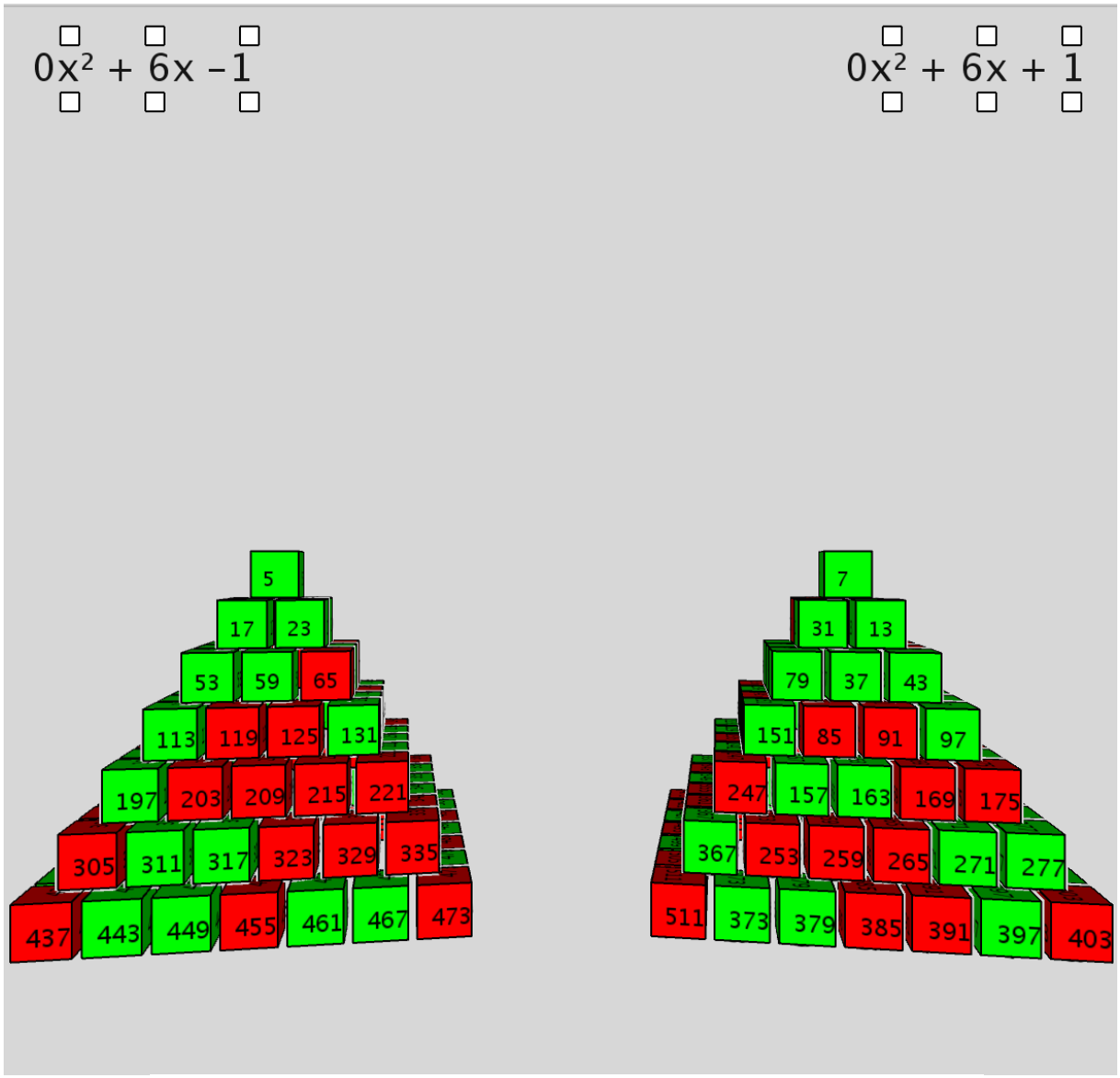

Figure 3 : Version final qui compare le polynôme 6x+1 et 6x-1

MAHREZ
Ali
GROUPE 3

5- Picking :

Cette partie était très difficile à comprendre et donc à réaliser. Pour le rendre fonctionnel, j'ai tout d'abord créé un vertex shader qui avait pour but d'interpoler les vertices de chaque point avec l'identifiant du cube, ce qui permet, une fois cliqué sur la surface du cube, d'obtenir le numéro du cube cliqué, puis un fragment shader qui avait pour but de créer une couleur en fonction du nombre passé en argument. Pour réussir à envoyer l'identifiant du cube, j'ai attribué le numéro du cube à chacune de ses vertex lors de sa création. Ensuite j'ai rajouté dans mes shaders une variable qui permet de lire l'attribut de chaque cube et donc le placer dans chaque vertice. Ensuite dans ma fonction mousePressed(), j'ai créé un PGraphique qui reproduit la pyramide en 2D et qui a pour seul et unique but de lire les pixels cliqués par la souris. Une fois cliqué, grâce à une fonction prédéfinie get(mouseX, mouseY) qui prend la couleur en dessous de la souris, je peux recréer le nombre cliqué en additionnant le résultat de chacune des couleurs(rouge vert et bleu). Une fois l'identifiant identifié, j'ai rajouté un Array booléen qui pour chaque indice correspond au numéro du cube, indique si un cube est cliqué. Si ce dernier est cliqué il s'affichera dans les deux modèle en couleur magenta.

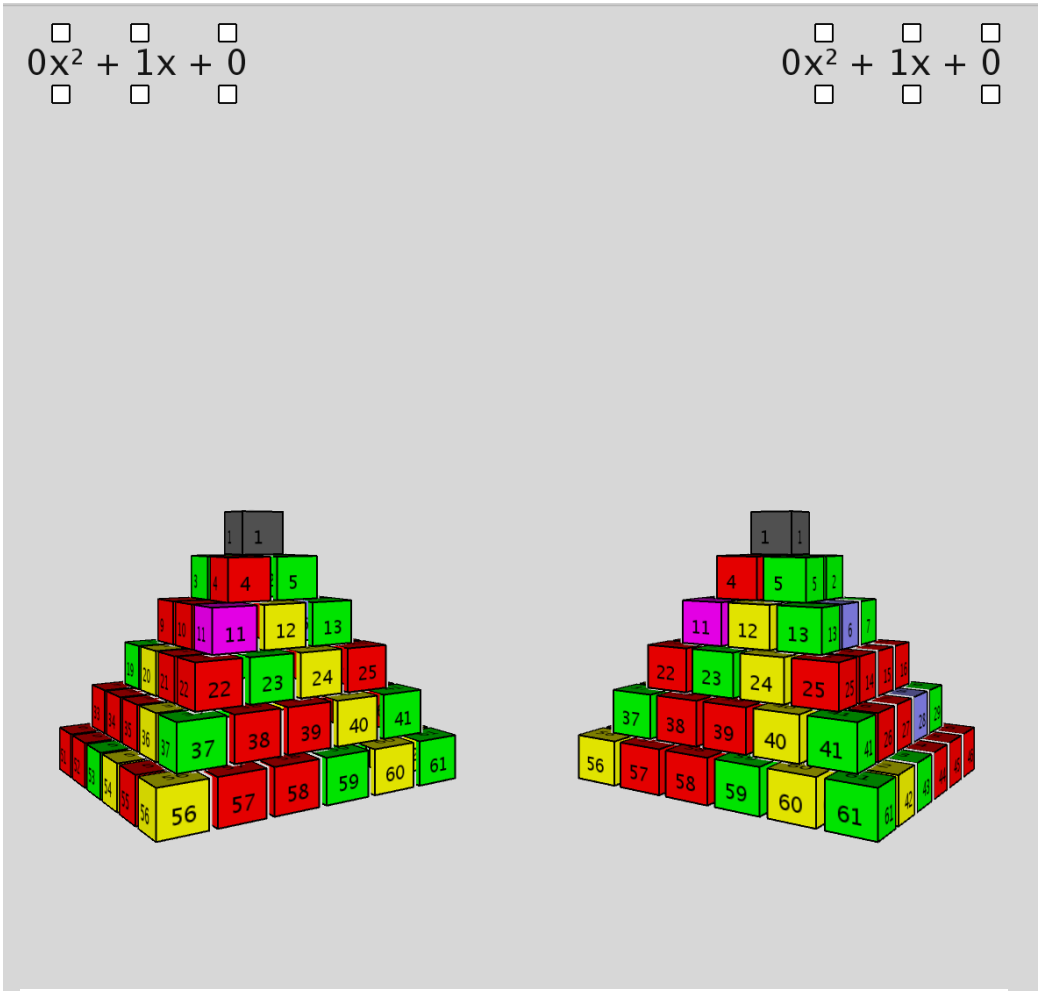

Figure 4 : Picking dans le numéro 11 avec comme couleur magenta

MAHREZ
Ali
GROUPE 3

LIMITES :

Il n’y a pas eu vraiment de limite bien que la réalisation du projet était assez difficile.