Synthèse D’image

Rapport de Projet

Barros Mikel-ange & Beddalia Zacharia

I) Présentation du projet

Dans le cadre de notre projet de synthèse d’image, il nous a été demandé de réaliser un jeu faisant intervenir l’ombre et la lumière comme base du gameplay. Dans ce sens, nous avons décidé de réaliser un jeu d’horreur/suspense.

Le but du jeu est de se rendre à divers endroits clés afin d’y récupérer des indices. Le joueur gagne lorsque les 6 indices sont récupérés. Le joueur perd si il ne récupère pas les indices à la fin du temps imparti (240 secondes).

Le personnage joueur est au centre de l’écran. Il possède une lanterne qui illumine en face de lui sur une distance définie. Le reste de la scène est sombre. On ne discerne rien à part des formes très vague.

Toutes les 60 secondes, un éventement se déclenche pendant 3 secondes :

- le protagoniste apparaît
- le joueur ne peut plus bouger
- un effet sonore de tonnerre se lance
- la lanterne s’éteint
- la lumière de la foudre (bleutée) apparaît dans la scène
- un dialogue apparaît

Quand cet événement a eu lieu 4 fois, la partie est perdue.

La caméra est fixe sur l’axe Y, et suit le joueur avec une vue en plongée.

Le joueur ne peut pas sortir de la scène, ni rentrer à l’intérieur des différents objets constituants la scène grâce à l’utilisation d’un système de collision.

II) Mise en place technique

II.1) la gestion des shaders

Afin de mettre en place notre jeu d’horreur le plus important fut de lui définir une ambiance pour cela l’utilisation de shader nous a semblé primordial.
Dans notre projet nous avions deux idées majeures concernant l’éclairage, une lampe à bougie pour les déplacements et un éclair arrivant indiquant le temps. Pour modéliser cela, nous utilisons un seul shader. Basé sur les méthodes vues en cours, ce shader va se comporter différemment en fonction du type d’éclairage que l’on va lui demander.

II.1.a) La lampe

Ainsi pour la lampe nous calculons la lumière en fonction de la distance à l'axe de direction de la lampe et de la distance au point d'origine de la lumière afin de créer un cône atténué sur les côtés. De plus nous faisons varier la distance d'éclairage selon un cosinus afin de simuler le scintillement d'une bougie.

Les formules utilisées pour simuler cela sont montrées sur la figure 1:

```
float flameVar= a*cos(2*PI*f*time);
float cone_dist = dot(p - source,direction);
float cone_radius = (cone_dist / h) * r;
float orth_distance = length((p - source) - cone_dist * direction);

float i=0.08;
if(orth_distance<2*cone_radius+0.005)
{
        i=1.0-(orth_distance-cone_radius)/(2*cone_radius);
        if(i>=1.0)
        {
            i=1.0;
        }
}
float center_dist=length(p-source);
if(center_dist>2+flameVar)
{
        i=i-(center_dist-(2+flameVar))/(2+flameVar);
        if(i<0.08)
        {
            i=0.08;
        }
}
```

Figure 1: Code de la lampe

II.1.b) L'éclair

Afin de simuler un éclair, nous éclairons l'ensemble de la scène sur une durée de deux secondes avec une lumière bleutée. Cette lumière variera d'intensité en fonction d'un cosinus afin de simuler les scintillements des éclairs.
Les autres utilisations de l'éclair seront détaillés dans la partie interactions.

La méthode mise en place est présentée dans la figure 2 :

```
float flameVar= a*cos(2*PI*f*time);
lighting = (5* flameVar) * (ambient + (diffuse + specular)) * colore*i*vec3(0.5,0.5,1);
```

Figure 2 : code de l'éclair

II.2) La gestion du son

Une autre partie importante de l'ambiance d'un jeu d'horreur est le son. Afin de le mettre en place nous avons choisi d'utiliser la librairie Soloud. Le choix de cette librairie, c'est fait car elle nous permettait de jouer de nombreux sons en même temps et ce avec un haut niveau d'abstraction.

Afin d'implémenter la gestion du son avec Soloud une classe, Audio a été mise en place afin de faciliter la gestion des bruitages.

II.3) La scène

Tout bon jeu ne serait rien sans une scène adaptée à son ambiance. Et c'est pourquoi nous avons passé de nombreuses heures à modéliser des éléments étranges , déformés ou cassés sur blender afin d'obtenir un décor correspondant à nos envies. Une fois que nous avons obtenu des objets nous convenant nous avons pu les positionner sur notre scène à l'aide de différentes transformations et ainsi recréer une scène réaliste.

II.4) Les interactions

II.4.a) Les boîtes englobantes

Afin de pouvoir profiter d'un système de collision permettant de limiter les déplacements du joueur à la seule pièce qui constitue le niveau du jeu et pour l'empêcher de pouvoir chevaucher des éléments du décor, nous avons utilisé des boites englobantes avec une détection de collision.
Lorsque le joueur rencontre un obstacle, il est repoussé vers sa précédente position. Ceci crée un effet de saccade mais dont l'impact visuel est moindre étant donné la très faible luminosité de la scène. La méthode implémenté et celle des boites non orientées basées sur les AABB.

Nous nous sommes resservi de ce que nous avons réalisé pendant le semestre même si, après réflexion, un système qui ne prenant en compte que les axes x et z aurait été moins gourmand pour des résultats identiques. De la même manière les murs étant des barrières, une méthode de détection basé sur un plan plutôt que des boites aurait été un choix intéressant mais in-explorer par manque de temps.

II.4.b) Les interfaces et les dialogues

Afin de mettre en place des interfaces et des dialogues nous avons décidé d'utiliser la librairie ImGui. Simple et permettant un positionnement dans l'espace nous avons pu mettre en place un menu ainsi que différents dialogues à l'intérieur de notre jeu. Ces dialogues permettent une meilleure immersion du joueur et permettent au joueur de comprendre les objectifs du jeu.

II.4.c) La gestion du personnage

Le personnage est géré directement avec la sdl2 et la détection de touches.
Ainsi le personnage peut se déplacer en avant (z) et en arrière (s) et tourner sur lui-même (q & d).
De plus, dans certains cas il aura la possibilité si il est assez près de ramasser un objet (e).

II.4.d) La foudre

La foudre à une importance capitale dans le jeu. En effet, en plus de définir le timer du jeu, il va aussi permettre le déclenchement de certaines phases de dialogue ou de certains événements.
Il va ainsi permettre le déclenchement d'une musique et d'un effet sonore.
De plus lorsque l'éclair se déclenche le personnage se retrouve dans l'incapacité de bouger.

II.4.e) Les événements de positions

Enfin en fonction de votre position dans l'espace certains événements tels que le déclenchement d'un son peuvent se produire

III) Conclusion

Ce projet aura été une expérience enrichissante sur la synthèse d'image. Nous avons pu mettre en pratique une grande partie ce qui a été vu pendant ce semestre notamment l'utilisation de shader, les boites englobantes et l'utilisation d'OpenGL.
Avec plus de temps, nous aurions aimé intégrer plus de fonctionnalité à notre jeu, en particulier l'utilisation de shadowmap pour laquelle nous avons réalisé des recherches avant de nous raviser par manque de temps. Les lightmaps ont aussi été considérées notamment pour afficher des ombres directement sur le sol. Par exemple, l'ombre d'élément n'apparaissant pas dans la scène comme des arbres hors champ.