Pierre Alain
alain@enssat.fr

Romain Heriteau
rheritea@enssat.fr

Florian Sautejeau
fsauteje@enssat.fr

Antoine Maurais
amauraisb@enssat.fr

Module :
Contenu Multimédia

Promotion :
Fisa Informatique 2

Projet « Labyrinthe aveugle »

ENSSAT
LANNION

Université de Rennes

1 Introduction

1.a Présentation du projet « Labyrinthe aveugle »

Le projet « Labyrinthe aveugle » consiste à la création d'une application de jeu, en équipe. Le jeu est un labyrinthe dans lequel le joueur doit s'orienter grâce au son. En effet, le joueur ne verra pas le labyrinthe ainsi que sa position dans celui-ci. Ce projet dispose de 3 niveaux d'avancement qui sont les suivants :

- **Echo-location simple** : Le joueur doit appuyer sur des touches pour savoir s'il y a un mur à droite/gauche/devant (/en haut/en bas pour la version en 3dimensions).
- **Echo-location avancée** : Le joueur obtient automatiquement un retour en fonction de son environnement.
- **Ajout d'un mode spectateur** : Pour des raisons de debug, vous pouvez avoir besoin d'afficher le labyrinthe dans une fenêtre et la rendre visible aux spectateurs

1.b Présentation des membres du groupe

Ce projet est à réaliser en groupe, notre équipe, la FAR est composée de 3 membres. Florian Sautejeau, Antoine Maurais et Romain Heriteau composent la FAR.

2 Le travail en équipe

Afin de mener à bien ce projet en groupe nous avons fonctionné sous la forme de mini-scrum. Un cycle correspondait à une séance de 3h30. Celui-ci débutait par un point d'équipe où chacun décidait les tâches à réaliser au cours de la séance. Et il finissait par une présentation du travail réalisé. Cela nous a permis de nous faire des feedbacks et d'adapter notre trajectoire sur le long terme. Nous avons utilisé les outils suivant :

- **Trello** pour la gestion de l'état de nos tâches et de notre reste à faire.
- **git** pour le développement collaboratif avec l'aide de son système de branches.

3 Mise en place technique

Le développement du jeu a été réalisé en C++. Le C++ est un langage de programmation orienté objet créé en 1983 par Bjarne Stroustrup. Aucun membre de notre équipe avait d'expérience sur ce langage de programmation nous avons du montée en compétence.

3.a Choix de la méthode de génération du labyrinthe

Nous avons décidé de réaliser un labyrinthe en 2 dimensions. La génération de celui-ci est assurée par la méthode « generate » présente dans le fichier MazeGenerator. Cette méthode crée une matrice représentant le labyrinthe. Celle-ci s'appuie sur l'algorithme de génération de labyrinthe « Sidewinder ». L'algorithme « Sidewinder » parcourt les cellules du labyrinthe ligne par ligne, créant des passages horizontaux en cassant les murs vers la droite lorsque la cellule actuelle n'est pas la dernière de la ligne. Pour les passages verticaux, il choisit une cellule aléatoire dans la course horizontale et casse le mur vers le haut s'il atteint la dernière cellule de la ligne ou si un nombre aléatoire modulo 2 est égal à 0. Ainsi, l'algorithme génère progressivement le labyrinthe avec des passages contrôlés, créant une matrice finale représentant le labyrinthe avec les murs cassés et intacts.

3.b Utilisation d'OpenAL pour la gestion du son

En ce qui concerne la gestion du son nous avons fait le choix d'OpenAL. OpenAL (Open Audio Library) est un bibliothèque pour la gestion de la lecture audio 3D, créée par Creative Labs en 2000. Cette bibliothèque était parfaite pour nous car elle permet de gérer le son en 3 dimensions, elle est open source et elle dispose d'une bonne documentation.

3.c Mise en place de l'écho-location simple

L'écho-location est assurée par une méthode au sein du code, « playSoundAround ». Celle-ci intervient à la suite du clique sur le bonton « a » par le joueur. Il pourra alors entendre la position des murs autour de lui (devant et côté) grâce à un son binaurale. Cette méthode procède donc en deux étapes : détecter les murs autours du joueur puis produire un son en conséquence. Deux paramètres sont donc nécessaire pour le fonctionnement de celle-ci, l'objet joueur et le labyrinthe.

3.d Utilisation de SDL2 pour l'interface

La bibliothèque SDL2 (Simple DirectMedia Layer 2) est utilisée pour créer une interface utilisateur dans notre projet. L'interface présente un design moderne avec un fond noir et des éléments blancs pour représenter les murs. L'activation de l'interface se fait en appuyant sur la touche « V » du clavier, offrant ainsi une interaction simple et intuitive pour les utilisateurs. Grâce à SDL2, nous bénéficions d'une interface visuellement attrayante et d'une expérience utilisateur fluide.

4 Architecture logicielle

Notre projet est découpé en 4 parties :

- **un dossier docs** qui regroupe les documents documentations. Nous avons fait le choix de faire de la docs as a code. Cela permet d'avoir une documentation collaboratives et présente au même endroit que le code avec un suivi de version grâce à des outils tels que git.
- **un dossier data** qui regroupe les fichiers audios nécessaire au bon fonctionnement du projet.
- **un dossier libs** qui regroupe les librairies utilisées pour le projet.
- **l'ensemble des fichiers assurant le fonctionnement du projet**, ceux ci sont situé à la racine.

5 Tests

A la fin de chaque itération de notre mini-scrum nous mettions en commun notre code et le testions. Ces tests etaient orientés côté utilisateur avec des tests utilisateurs. Ainsi chaque membre de l'équioe testait sur sa machine le bon fonctionnement du jeu.

6 Conclusion

En conclusion, le projet « Labyrinthe aveugle » a été une expérience enrichissante pour l'équipe FAR, composée de Florian Sautejeau, Antoine Maurais et Romain Heriteau. Ce projet a impliqué la création d'une application de jeu en équipe, dans laquelle le joueur doit s'orienter dans un labyrinthe grâce au son. Le projet a été réalisé en C++, un langage de programmation orienté objet, pour lequel l'équipe a dû monter en compétence. La génération du labyrinthe a été réalisée grâce à une méthode, tandis que la gestion du son a été assurée par la bibliothèque open-source OpenAL. L'interface utilisateur a été créée en utilisant SDL2. Le travail en équipe a été organisé sous la forme de mini-scrums, avec une gestion de tâches via Trello et un développement collaboratif avec git. Des tests utilisateurs ont été effectués à la fin de chaque itération. Le projet a ainsi permis à l'équipe FAR de développer ses compétences en C++, de travailler efficacement en équipe, de faire face à des difficultés environnementales et de relever le défi de créer une application de jeu originale et intéressante.