Projet : Création d'un jeu vidéo 2D Snake

Version du document : 3

Propriétaire du document : Equipe 22

Date de Création : 26/11/2023

Document de Spécification de moteurs de Jeux Vidéos.

Introduction

Ce projet consiste en le développement d'un jeu vidéo en deux dimensions (2D), mettant en place les moteurs physique et graphique. Ces moteurs permettent de simuler le déplacement des objets du jeu et d'afficher les éléments visuels du jeu, en utilisant des principes physiques tels que la position, la vélocité, et l'accélération.

Ce document a pour but de définir les spécifications détaillées du moteur physique, graphique, entrée et noyau du jeu vidéo 2D, ainsi que la couche GamePlay dédiée spécialement au jeu Snake.

I. Moteur physique :

Un moteur physique dans le développement d'un jeu est un composant logiciel qui simule les lois de la physique dans le contexte du jeu. Pour gérer les mouvements, les collisions et les réactions des objets et des entités du jeu, en tenant compte de principes physiques tels que la gravité, la vélocité, l'accélération, et les forces.

1. Espace du jeu :

L'espace du jeu est un monde en deux dimensions où les événements se déroulent en suivant les règles.

2. Objets et Entités :

Les objets et les entités représentent les éléments du jeu, tels que les personnages, les ennemis, les objets, murs, obstables. Chaque objet est caractérisé par sa position dans un espace bidimensionnel, sa vitesse, son accélération, sa largeur, sa hauteur et sa masse. Les objets peuvent se déplacer en mettant à jour leur position en fonction de leur vitesse et de leur accélération.

3. Dynamique des Objets :

Le moteur doit permettre la simulation de mouvement des objets, en prenant en compte la position, la vitesse, l'accélération et les dimensions.

- **Position :** L'objet a une position dimensionnelle (x, y) dans l'espace du jeu, représentée par un vecteur.
- **Vitesse :** Il possède également une vitesse (x, y),présentée par un vecteur.
- **Accélération** : L'accélération (x, y) est utilisée pour modifier la vitesse de l'objet et est stockée dans un vecteur.
- **Dimensions :** L'objet a une largeur et une hauteur qui définissent sa taille dans l'espace du jeu.
- **Masse :** Une masse (non visible dans la classe actuelle) peut être ajoutée pour des calculs liés à la physique).

Le déplacement des objets s'effectue en suivant un processus simple :

Mise à Jour de la Vitesse :

Une méthode ajuste la vitesse de l'objet en fonction de l'accélération. Elle ajoute la composante x de l'accélération à la vitesse x de l'objet et la composante y de l'accélération à la vitesse y de l'objet.

Mathématiquement : `velocity$_x$ += acceleration$_x$` et `velocity.y += acceleration.y`.

Mise à Jour de la Position :

La méthode met à jour la position de l'objet en fonction de sa vitesse. Elle ajoute la composante x de la vitesse à la position x de l'objet et la composante y de la vitesse à la position y de l'objet.

Mathématiquement : position.x += velocity.x et position.y += velocity.y.

Ces opérations sont effectuées à chaque itération du moteur du jeu, permettant ainsi de simuler le mouvement de l'objet en fonction de sa vitesse et de son accélération.

Cela signifie que l'accélération influe sur la vitesse, qui à son tour influence la position. En d'autres termes, l'accélération représente le changement de vitesse au fil du temps. Ce processus est itératif, ce qui signifie qu'il se répète à chaque itération du jeu, permettant ainsi aux objets de se déplacer de manière dans l'espace 2D.

4. Réaction aux Forces Extérieures :

Les objets doivent réagir aux forces extérieures, par exemple une rafale de vents, forçant les objets à changer de direction.

5. Gestion de Collision :

Le moteur doit gérer efficacement les collisions entre objets. Lorsque deux entités se superposent ou entrent en contact :

- *Détection de Collision :* Le moteur doit être capable de détecter les collisions entre les objets du jeu, tels que des objets traversable ou pas.

- *Réponse aux Collisions :* Le moteur doit fournir des mécanismes pour gérer la réaction aux collisions, y compris la résolution de la collision (rebond, pénétration minimale) et la gestion des dégâts.

6. Gestion de la gravité:

La gravité est un élément clé dans la simulation physique. Le moteur applique la force de gravité pour faire en sorte que les objets tombent vers le bas, créant ainsi un réalisme dans le mouvement.

II. Moteur graphique :

Le moteur graphique a pour objectif principal de convertir les données du jeu en une représentation visuelle attrayante et cohérente. Il se charge de traduire les informations sur les objets, les décors, et les actions du jeu en images affichées à l'écran, contribuant ainsi à créer un univers visuel captivant pour les joueurs.

1. **Création d'objets graphiques:** Le moteur permet la création et la gestion d'objets graphiques, définis par leurs dimensions, positions, couleurs, et éventuellement des textures.

2. **Animations :** Le moteur doit prendre en charge des systèmes pour la création d'effets spéciaux. Elles sont utilisées dans une variété de contextes, tels que le mouvement des personnages, les actions spéciales, les réactions aux événements.

3. **Affichage :** Ce moteur se charge d'afficher les objets sur une fenêtre.

4. **Gestion des textures :** Le moteur prend en charge l'application de textures aux objets graphiques, permettant d'ajouter des détails visuels réalistes.

III. Moteur d'Entrées :

Le module d'entrée a pour objectif de gérer les interactions utilisateur avec le jeu en capturant et traitant les entrées, principalement à partir du clavier. Il offre une interface pour mapper des événements d'entrée à des actions spécifiques dans le jeu.

1. **Gestion des Événements Clavier :** Le module prend en charge la gestion des événements liés au clavier, notamment les pressions de touches.

2. **Mappage d'Événements :** Il permet le mappage d'événements clavier spécifiques à des actions du jeu, facilitant la personnalisation des commandes.

3. **Structure d'Événements :** Le module utilise une structure d'événements pour encapsuler les informations sur les entrées utilisateur, fournissant des détails tels que la touche pressée.

IV. Moteur de Son :

Le moteur sonore du jeu a pour but de gérer les éléments audio, renforçant ainsi l'aspect immersif de l'expérience de jeu. Cette composante spécifique est responsable du chargement, de la lecture, de la pause, de la reprise, de la réinitialisation et de l'arrêt des pistes sonores.

V. Noyau :

Le noyau (Kernel) a pour objectif de coordonner les différents composants du jeu, notamment les moteurs graphique et physique, le gestionnaire d'entrée, et les objets de jeu, pour assurer le bon fonctionnement et l'interaction cohérente entre ces éléments.

Le noyau donc est responsable de la gestion des aspects essentiels du jeu, la physique, le graphique et la gestion des entrées du jeu pour :

- Fournir une infrastructure solide pour la création d'objets.
- Faciliter l'intégration de la logique du jeu, des interactions utilisateur et des graphismes.

1. Composants:

Le noyau d'un jeu est composé des éléments suivants :

- Moteur physique
- Moteur graphique
- **InputHandler :** Un gestionnaire d'entrées pour gérer les actions de l'utilisateur.

2. Fonctionnalités:

Le noyau permet :

- La création et la gestion d'objets.
- La mise à jour de l'état du jeu.
- D'assurer la gestion des entrées utilisateur, incluant la prise en charge des commandes pour déplacer un objet.
- De mettre en pause et de reprendre le jeu en modifiant l'échelle temporelle.

Mais également :

- ✓ *Gestion des Composants :* Le noyau gère les instances des moteurs graphique et physique, le gestionnaire d'entrée, ainsi que les objets de jeu.
- ✓ *Intégration des Moteurs :* Il intègre les moteurs graphique et physique pour synchroniser les aspects visuels et physiques du jeu.
- ✓ *Gestion des Entrées :* Le noyau prend en charge la gestion des entrées utilisateur en utilisant le gestionnaire d'entrée, permettant la liaison d'événements à des actions spécifiques.
- ✓ *Coordination des Objets de Jeu :* Il coordonne la création, la mise à jour et la gestion des objets de jeu, assurant une interaction harmonieuse entre les composants.

3. Couche Game Play – Le jeu Serpent :

La couche gameplay du jeu Snake a pour objectif de mettre en œuvre la logique spécifique du jeu, notamment le déplacement du serpent, la gestion du corps du serpent, les interactions avec l'utilisateur, et le lancement du jeu dans le contexte du noyau.

- ✓ *Initialisation du Serpent :* La couche gameplay initialise le serpent avec une tête colorée, des textures spécifiques, et une vélocité initiale.
- ✓ *Gestion du Corps du Serpent :* Elle gère la croissance du serpent en ajoutant de nouveaux segments au corps à chaque mise à jour.
- ✓ *Gestion des collisions du Serpent :* Le serpent doit réagir à la collision avec d'autres objets, tels que lui-même, qui forcera le jeu à se terminer, ou un objet à points, qui lui permettra de gagner des points.

Conclusion :

Cette spécification fournit un aperçu des caractéristiques, de l'architecture, du fonctionnement des différents moteurs d'un jeu en 2D ainsi que la couche gameplay du jeu Snake, détaillant sa contribution à l'expérience de jeu globale.