Escuela de Ingeniería en Computadores

Algoritmos y Estructura de Datos I

Proyecto 1

Tron

Profesor:

Leonardo Araya

Estudiante:

Andy López Mora

Grupo 2

Septiembre 2024

Índice

1. Introducción **2**

2. Descripción del problema **2**

3. Descripción de la solución **2**

3.1. Requerimientos 2

3.2. Implementación 3

3.3. Limitaciones 4

3.4. Problemas encontrados 4

4. Diseño General **4**

1. Introducción

Este documento proporciona una descripción detallada del proyecto de un juego desarrollado en C# utilizando Windows Forms. El juego es una adaptación del clásico "Tron", en el que los jugadores controlan motocicletas que dejan estelas en un mapa delimitado. El objetivo del juego consiste en evitar colisiones, ya sea con las estelas de otros jugadores o los límites del mapa y ser el último en ruedas.

2. Descripción del problema

El juego requiere una implementación donde el jugador y los enemigos se muevan a lo largo del mapa, evitando colisiones con las estelas de otros motociclistas y también con los límites del mapa. Además, se debe gestionar el combustible del jugador, que se reduce con el movimiento, el comportamiento de los enemigos, el cual es impredecible y que también cuentan con un combustible ilimitado. El juego cuenta con algunos "poderes" que pueden ayudar al jugador a mantenerse durante la partida, como el escudo y la hipervelocidad, también aparece aleatoriamente una pequeña cantidad de combustible por el mapa al igual que los poderes para seguir en el juego.

Inicialmente, tanto el jugador como las motos enemigas aparecen en posiciones aleatorias por todo el mapa, la moto del jugador al principio tiene un tamaño definido de 3 celdas, el cual se puede ir extendiendo a lo largo de la partida hasta tener un tamaño de 10. Los enemigos también son capaces de recoger los poderes y usarlos, si alguna de las motos es destruida, los poderes que tenía guardados sin usar serán repartidos a lo largo del mapa de manera aleatoria. Las maneras en que el juego termina son: colisión con una moto enemiga o su estela, colisión con un límite del mapa o quedar sin combustible

3. Descripción de la solución

3.1. Requerimientos

- **Movimientos del jugador:** el jugador debe poder moverse en cuatro direcciones (arriba, abajo, izquierda, derecha), debe tener un control y cuidado sobre el combustible, que se reduce cada cinco movimientos.

- **Movimiento de los enemigos:** los enemigos deben moverse aleatoriamente en el mapa y tener combustible infinito, ya que deben mantenerse para agregarle dificultad al jugador.

- **Colisiones:** el juego debe detectar las colisiones entre las motocicletas, sus estelas y los límites del mapa.

- **Poderes:** a lo largo del mapa se deben repartir poderes aleatoriamente cada cierto tiempo, los cuales pueden ayudar al jugador contra los enemigos, al igual que una pequeña cantidad de combustible.

- **Visualización del mapa:** los límites del mapa deben ser visibles y los elementos del juego como las estelas, las motos y los poderes deben ser claramente distinguidos.

3.2. Implementación

- **Movimientos del jugador:** implementado mediante eventos de teclado en la clase *GameForm*, donde se actualiza la dirección del jugador y se controla el consumo de combustible.

- **Movimiento de los enemigos:** utiliza una dirección aleatoria en la clase *Motorcycle*, con un comportamiento de cambio de dirección aleatorio cada cierto tiempo.

- **Colisiones:** implementado en el método *Move* de la clase *Motorcycle*, que verifica colisiones con estelas y límites del mapa.

- **Visualización del mapa:** los límites y elementos del mapa se dibujan en el método *Draw* de la clase *Game*.

3.3. Limitaciones

- **IA de enemigos:** la IA de los enemigos es básica, moviéndose aleatoriamente sin estrategias avanzadas.

- **Rendimiento:** el rendimiento del juego puede verse afectado con un gran número de motocicletas y elementos en el mapa.

3.4. Problemas encontrados

- **Movimiento estancado:** algunos enemigos y el jugador se quedaban estancados al principio del juego; se resolvió ajustando la lógica de movimiento y dirección.

- **Colisiones no detectadas:** problemas iniciales con la detección de colisiones fueron corregidos revisando la lógica de verificación en el método *Move*.

- **Error al crear la ventana del juego:** problema al ejecutar el código se abría una ventana de Windows Form completamente en blanco o simplemente no se ejecutaba debido a un error.

4. Diseño General