TRABAJO FIN DE GRADO

INGENIERÍA EN INFORMÁTICA

Titulo del Proyecto

Subtitulo del Proyecto

Autor
Andrés Merlo Trujillo

Directores
Luis López Escudero

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, mes de 2023

Título del proyecto

Subtítulo del proyecto.

Autor
Andrés Merlo Trujillo

Directores
Andrés Merlo Trujillo

Título del Proyecto: Subtítulo del proyecto

Andrés Merlo Trujillo

Palabras clave: unreal, simulador, ia,

Resumen

Se pretende desarrollar una aplicación gráfica utilizando el motor gráfico *Unreal Engine* cuyo objetivo es de poder simular carreras de coches. Uno de los objetivos de esta simulación es la de dotar a los pilotos virtuales de la capacidad de tomar decisiones realistas y de cometer errores, todo esto basado en las condiciones de cada piloto durante la carrera, que pueden variar dependiendo de las condiciones externas. El simulador también permitirá la modificación de parámetros antes y durante la carrera. Algunos de estos parámetros son: número de coches, número de vueltas, capacidad de los pilotos y modificación de las condiciones de cada piloto en tiempo real.

Project Title: Project Subtitle

First name, Family name (student)

Keywords: Keyword1, Keyword2, Keyword3, ...

Abstract

Write here the abstract in English.

Yo, **Andrés Merlo Trujillo**, alumno de la titulación TITULACIÓN de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 77147239H, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Andrés Merlo Trujillo

Granada a X de mes de 2023.

D. **Luis López Escudero**, Profesor del Área de XXXX del Departamento LSI de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Título del proyecto, Subtítulo del proyecto***, ha sido realizado bajo su supervisión por **Andrés Merlo Trujillo**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a X de mes de 2023 .

Los directores:

Andrés Merlo Trujillo

Agradecimientos

Poner aquí agradecimientos...

Índice general

Capítulo 1

Introducción

1.1. Introducción y Motivación

El mundo del motor siempre ha sido un deporte que despierta la curiosidad en muchas personas alrededor del mundo, ya sea por la innovación tecnológica continua que se produce cada año en los vehículos, por sentir la adrenalina de conducir un coche de carreras o incluso por la satisfacción personal de ser capaz de dar órdenes a un piloto para llevarlo a la victoria, seleccionando las mejores estrategias en cada momento. Este proyecto se centrará en esto último.

Gracias al avance de la informática, se puede realizar simulaciones de las carreras y de los vehículos para saber cuál es la mejor estrategia a seguir para llevar al equipo y al piloto a la victoria. Además, no son solo los profesionales los que recurren a este tipo de aplicaciones para realizar sus estrategias, sino que hay entusiastas de este deporte que compran este tipo de *software* para obtener la satisfacción personal de haber llevado un equipo virtual a la victoria y en el trayecto haber aprendido estrategias cada vez más complejas.

No obstante, el número de simuladores de gestión de carreras es relativamente pequeño y poco variado, normalmente centrándose en ciertas disciplinas concretas y en un solo equipo de carreras, sin poder interaccionar con los demás en ningún momento.

Los dos ejemplos más notables de este tipo de aplicación son:

- *Gran Turismo B-Spec:* Parte de la saga para consolas *PlayStation* que se centra en dar órdenes a un piloto en la carrera. Dependiendo de estas decisiones y de otros factores tales como la posición en la que se encuentre, puede variar su aguante físico, mental y su agresividad, pudiendo así cometer errores. Permite correr en una gran variedad de vehículos.

- *F1 Manager 22:* Simulador de gestión de carreras centrado exclusivamente en la *Formula 1*, similar al anterior simulador. La desventaja que tiene es que se centra nada más que en una disciplina particular, pero consigue un nivel alto de profundidad en los sistemas.

En este proyecto se pretende desarrollar un simulador multidisciplinar y personalizable mediante el uso de parámetros, incluyendo algunos similares a los mencionados anteriormente, y que además ofrece la posibilidad de modificar dichos parámetros de todos los pilotos en tiempo real, aparte de ser modificados por las condiciones de la carrera en la que se encuentre cada piloto.

Además, la aplicación permitirá ajustar las aptitudes, el número de pilotos, la velocidad máxima de los coches, el número de los mismos y el número de vueltas y la hora del día a la que se va a disputar la carrera. Cabe decir que el *software* permitirá también modificar otros aspectos ajenos a los que puedan modificar el resultado de la carrera, tales como la velocidad de simulación.

Todo lo mencionado permitirá a los entusiastas experimentar una simulación en tiempo real y sobre todo con el factor de la personalización superior a los ejemplos mencionados, al permitir simular condiciones muy específicas, y hacer que los usuarios comprendan como afectan dichas variables a los pilotos y su equipo, incluso en la vida real.

1.2. Objetivos

Como se ha mencionado en el apartado anterior, el objetivo principal de este proyecto es de ofrecer un simulador de carreras multidisciplinar y con un grado alto de personalización en las variables de cada piloto.

Además, el proyecto debe cumplir los siguientes objetivos mínimos relativos a cambios antes de ejecutar la simulación:

- Cambio de número de pilotos.
- Cambio de número de vueltas.
- Cambios de las aptitudes de los pilotos; es decir, el nivel máximo de cada condición que puede alcanzar.
- Cambio de la velocidad máxima de los vehículos.
- Permitir (opcionalmente) que haya variación de prestaciones entre vehículos.

En cuanto a cambios durante la simulación, debe cumplir estos objetivos:

- Seleccionar piloto de la lista de pilotos.
- Obtener las condiciones actuales del piloto seleccionado.
- Modificar las condiciones actuales del piloto seleccionado.

Cabe destacar que todos estos objetivos y algunos opcionales serán explicados en detalle en apartados siguientes.

1.3. Estado del arte

Como ya se ha descrito anteriormente, este proyecto hará uso de *Unreal Engine* como motor gráfico y como *framework* para la realización del simulador especificado. En cuanto a la forma en la que los pilotos se moverán por el circuito, he decidido usar el algoritmo de navegación **PONER AQUÍ EL ALGORITMO DE NAVEGACIÓN**.

A continuación, voy a explicar las distintas opciones que hay disponibles en motores gráficos y en algoritmos de navegación y explicar el motivo de la elección realizada. Separaré esto en dos subsecciones.

1.3.1. Motor gráfico

En el mercado hay gran variedad de motores gráficos de videojuegos, los más conocidos son los siguientes:

Unreal Engine es una plataforma de desarrollo que permite, entre otras cosas: el desarrollo de videojuegos, creación de contenido para programas y películas, simulación, etc [https://www.unrealengine.com/es-ES/faq]. Posee un soporte para un gran número de plataformas, tanto consolas como móviles.

Unreal además ofrece *Blueprint Visual Scripting*, el cual es un sistema de *scripting* visual que hace uso de nodos para programar las distintas partes del sistema. Permite programar de manera visual sin la necesidad de conocer la *API* de C++

Además, a partir de la versión 5.0, Unreal incluye diversas tecnologías como:

- TSR *(Temporal Super Resolution)*, mejorando el rendimiento haciendo que el juego se renderice internamente a una resolución menor y luego sea reescalado al tamaño deseado.

- Nanite, que permite tener objetos con una gran cantidad de polígonos en la pantalla,
- Lumen, que genera una iluminación más realista.

Todo esto permite tener un resultado a nivel gráfico muy convincente sin necesitar dedicar demasiado trabajo en este apartado y sin requerir un hardware demasiado potente.

Asimismo, Unreal incluye un bazar con una gran cantidad de recursos, permitiendo simplificar mucho el desarrollo de videojuegos.

Sin embargo, no tiene una licencia de código abierto, pero su código fuente es accesible a través de un repositorio de GitHub

Unity es una plataforma de desarrollo que permite: desarrollar videojuegos, visualizar construcciones en el ámbito de la arquitectura, para uso en la industria automotriz y para la creación de películas y series.

Entre sus características se encuentra:

- Uso de C# como lenguaje de programación.
- Motor de físicas 2D separado del de físicas en 3D.
- Posee un *Scriptable Rendering Pipeline* altamente personalizable, permitiendo mediante el uso de scripts escritos en C# personalizar sombras, iluminación, oclusión ambiental, entre otras.
- Soporte para un gran número de plataformas, incluyendo consolas y móviles, aparte de Windows, macOS y Linux.
- Incluye un lenguaje de programación visual, similar a *Unreal* que permite programar sin necesitar conocer la *API*.

Además, Unity tiene varias versiones del programa, incluyendo uno gratuito y los demás de pago, con prioridad en la asistencia y un mayor abanico de herramientas.

Godot es un motor gráfico gratuito y multiplataforma de código abierto que permite crear videojuegos en 2D y 3D.
Entre las características más notables se encuentran las siguientes:

- Motor gráfico 2D dedicado,

- Programación usando GDScript, C# o C++ de manera oficial. No obstante, se puede programar usando otro lenguaje como Rust, Python o JavaScript.
- Soporte para un gran número de plataformas, incluyendo móviles y consolas.
- Incluye un motor de físicas (solo para el motor gráfico 3D), aunque es algo más simple que el de sus competidores.

A partir de la versión 4, Godot ha dejado de incluir el lenguaje visual, debido a su falta de uso, por lo que solo es posible utilizar lenguajes de programación convencionales.

Además, Godot tiene una biblioteca con una gran cantidad de recursos para utilizar durante el desarrollo de un videojuego.

Al final he decidido usar Unreal Engine debido a que la versión gratuita incluye toda la funcionalidad, ofrece unos buenos resultados a nivel visual sin requerir demasiado esfuerzo, es el más potente en cuanto a programación visual y es con el que más familiarizado estoy de todas las opciones citadas anteriormente.

1.3.2. Algoritmo de navegación

En cuanto a algoritmos de navegación para los vehículos, los más destacados son:

- Aprendizaje por refuerzo (Q-Learning): Es una técnica de aprendizaje automático en la que el agente aprende a tomar decisiones en un entorno mediante la interacción a través de acciones (en este caso es: acelerador, freno y volante) y recibiendo una recompensa en caso de acercarse al objetivo deseado o de hacerlo bien.

  Es importante obtener un equilibrio entre exploración y explotación, para que el agente sea capaz de actuar ante situaciones desconocidas de manera correcta. En caso de estar desequilibrado, puede darse la situación en el que se produzca un máximo local y no sea capaz de descubrir la mejor estrategia.

  Para evitar esto, se suelen utilizar métodos de equilibrio. Uno de ellos es el denominado *epsilon-greedy*, que permite controlar la cantidad de exploración mediante un parámetro *epsilon*, que determina la aleatoriedad en la selección de acciones permitiendo elegir en función de dicho parámetro una acción aleatoria o la mejor hasta el momento.

Una desventaja que tiene es la necesidad de entrenar el modelo, cosa que puede llevar demasiado tiempo, pero con el suficiente tiempo y con un conjunto de recompensas y castigos bien realizado, es uno de los mejores algoritmos para resolver este tipo de problema.

- Mediante reglas: Otra forma de resolver este problema es mediante reglas, que como su nombre indican, son un conjunto de reglas que se accionarán cuando se dé la condición deseada y normalmente están priorizados. Suelen tener forma de árbol, al hacer preguntas en cadena hasta llegar con la solución.

  Esto tiene la ventaja de ser más sencillo de implementar que el de aprendizaje por refuerzo (sin usar ninguna biblioteca ya implementada), pero suele tener peores resultados, al poder darse el caso de no haber priorizado bien las reglas o no haber tenido en cuenta algún caso especial.

- Máquina de estados: Este algoritmo implementar una máquina de estados finito mediante una estructura de datos, donde cada nodo representa un estado en el que se encuentra el piloto y cada arco representa la transición a otro estado.

  Una de las ventajas que tiene es que suele ser más simple describir el comportamiento en agentes más complejos y además, suelen ser más fáciles de modificar que los basados en reglas.