FOOTBALL AUTÓNOMO EN PYTHON

Velasco Reyes Juan Ramón, Gallegos Alvarez Jovani, García Canteros Ángel Josue
{*jjf, gaaj010320, garciacanterosangeljosue057*}*@gs.utm.mx*
Profesor: M.C. Juan Juárez Fuentes

***Abstract*—En este artículo se presenta un proyecto basado en Python y Pygame, que simula un juego de fútbol automatizado, donde los jugadores y el balón son controlados por lógica de autómatas. El código está conformado por tres módulos principales, automatas_game, cons y entities, cada uno responsable de una parte clave del juego. Se analiza cómo cada componente interactúa para crear una experiencia de juego coherente y dinámica, donde los jugadores, porteros y el balón tienen comportamientos definidos por reglas matemáticas y aleatorias. Este enfoque ofrece una interesante mezcla de programación orientada a objetos y simulación de sistemas dinámicos simples.**

***Palabras clave*—Pygame, Simulación de Fútbol, Autómatas, Python, Desarrollo de Videojuegos.**

I. Introducción

El desarrollo de videojuegos es una disciplina compleja que involucra la creación de entornos interactivos, simulación de comportamientos y control de objetos dentro de un espacio. En este proyecto, se presenta un juego sencillo de fútbol automatizado utilizando Python y la biblioteca Pygame. El objetivo de este trabajo es simular el comportamiento de jugadores, porteros y un balón, controlados por autómatas que siguen reglas predefinidas.

El código está organizado en tres archivos fundamentales, automatas_game, que configura y maneja la interfaz de usuario y la ventana del juego; cons, que contiene las constantes y configuraciones necesarias para la lógica del juego, y entities, que define las entidades del juego como los jugadores y el balón, y las interacciones entre ellas.

II. Estructura del Código

El código está compuesto por tres archivos principales, cada uno con una responsabilidad clave para el funcionamiento del juego. A continuación, se describen los componentes de cada archivo.

A. automatas_game.py

Este archivo contiene la configuración inicial del juego, incluidas las dimensiones de la ventana, los colores y las áreas de juego. Además, maneja la lógica para la actualización de la ventana y la interacción con el usuario. Las principales características de este archivo son:

- Definición de los colores utilizados en el juego, como blanco, negro, rojo, azul, entre otros.
- Configuración de las dimensiones de la ventana del juego y de las áreas clave, como las porterías y el campo.
- Creación y manejo de la ventana de Pygame, que incluye la actualización y renderización de los elementos visuales.

B. cons.py

Este archivo se encarga de almacenar las constantes del juego, como las dimensiones de la cancha y la posición de las áreas de gol. En él se definen los valores utilizados para la creación de los objetos de juego en automatas_game y entities.py

Algunas de las definiciones más importantes en este archivo incluyen:

- Las posiciones y dimensiones de las áreas de gol y del campo.
- Las constantes que definen el tamaño de la cancha y los límites de los objetos dentro del juego.

C. entities

Este archivo contiene las clases que definen las entidades del juego, como los jugadores, porteros y el balón. Las clases se describen de la siguiente manera:

- **Jugador**: Define el comportamiento de los jugadores, tanto controlados por el usuario como por autómatas. Se implementan funciones para

moverlos en la cancha, ya sea de manera aleatoria o siguiendo comandos.

- **Portero**: Hereda de la clase ‘Jugador‘ y redefine el comportamiento de los porteros, que se mueven dentro de su área de gol.
- **Balón**: Controla el movimiento del balón, incluyendo su velocidad, dirección y colisiones con los jugadores y las paredes. También se simulan los goles y los rebotes.

III. Simulación de Jugadores y Comportamiento Aleatorio

Uno de los aspectos interesantes de este proyecto es el comportamiento de los jugadores y porteros, que puede ser controlado de forma aleatoria. Los jugadores automatizados siguen una lógica que calcula su velocidad y dirección en función de un modelo aleatorio. Para lograr esto, se utiliza el módulo ‘random‘ de Python para generar valores aleatorios y simular movimientos impredecibles dentrode los límites de la cancha.

Por ejemplo, los porteros se mueven dentro de su área de gol, utilizando un modelo de movimiento aleatorio basado en una distribución gaussiana. De esta manera, se crea una simulación en la que los jugadores siguen reglas simples pero efectivas para simular un partido de fútbol.

IV. Conclusiones

El proyecto presentado en este artículo ofrece una visión introductoria al desarrollo de juegos con Pygame, utilizando modelos simples de autómatas para simular un partido de fútbol. A través de la implementación de clases para los jugadores, porteros y balón, se crea un entorno interactivo que permite experimentar con la simulación de movimientos aleatorios y lógicos dentro de un juego de deportes.

Aunque el código está diseñado para ser sencillo, puede expandirse fácilmente para incluir más características, como la detección de goles, la gestión de equipos o incluso una interfaz de usuario más avanzada. Este proyecto demuestra cómo la programación orientada a objetos y la simulación matemática pueden aplicarse para crear juegos automatizados interesantes y dinámicos.

V. Referencias

References

[1] Pygame. (2024). *Pygame documentation*. Retrieved from https://www.pygame.org/docs/

[2] Python Software Foundation. (2024). *Python documentation*. Retrieved from https://docs.python.org/