



ESCUELA POLITÉCNICA  
SUPERIOR DE CÓRDOBA  
Universidad de Córdoba



# UNIVERSIDAD DE CÓRDOBA ESCUELA POLITÉCNICA SUPERIOR

GRADO EN INGENIERÍA INFORMÁTICA  
ESPECIALIDAD EN COMPUTACIÓN

TRABAJO DE FIN DE GRADO

---

## Entrenamiento de una IA mediante aprendizaje por refuerzo para un juego hecho en Unreal Engine

---

- MANUAL DE USUARIO -

**Autor:**

Francisco David Castejón Soto

**Directores:**

Dr. Manuel Jesús Marín Jiménez

Dr. Javier Sánchez Monedero

Córdoba, 9 de junio de 2024

# Índice general

<b>Índice de figuras</b>	<b>II</b>
<b>1. Introducción</b>	<b>1</b>
<b>2. Prerequisitos</b>	<b>2</b>
2.1. Windows . . . . .	2
2.2. Unreal Engine . . . . .	2
2.3. MindMaker . . . . .	3
2.4. Python . . . . .	3
2.5. Stable Baselines 3 . . . . .	4
2.6. SB3 Zoo . . . . .	4
2.7. Gymnasium . . . . .	4
2.8. Otras bibliotecas . . . . .	4
2.9. Editores . . . . .	5
2.10. Proyecto . . . . .	5
<b>3. Instalación de los entornos</b>	<b>7</b>
<b>4. Guía de uso</b>	<b>8</b>

# Índice de figuras

2.1. Descarga del proyecto desde Gitlab . . . . .	6
4.1. Ajustes del episodio desde UE . . . . .	9
4.2. Juego en el editor de Unreal Engine . . . . .	9
4.3. Botón para ejecutar el juego en el editor de UE . . . . .	10

# Capítulo 1

## Introducción

Como bien indica el título de este Trabajo de Fin de Grado, el objetivo de este proyecto es el desarrollo de un pequeño videojuego hecho en Unreal Engine que haga uso de una inteligencia artificial entrenada mediante aprendizaje por refuerzo.

Con motivo de asistir en la correcta instalación y uso del software desarrollado, se ha elaborado este manual de usuario. En él se detallan los pasos necesarios para el correcto funcionamiento del mismo, así como una guía de uso para facilitar su manejo.

## Capítulo 2

# Prerequisitos

Para probar el funcionamiento del videojuego y la IA, son necesarias una serie de programas, bibliotecas y herramientas. En el manual técnico puede encontrar más información acerca de cada una de ellas. A continuación se detallan cada una de ellas, junto a diferentes guías de instalación.

### 2.1. Windows

Aunque existen versiones de Mac y Linux de Unreal Engine, el videojuego ha sido desarrollado en Windows 11, por lo que se recomienda usar este sistema operativo para evitar problemas de compatibilidad.

### 2.2. Unreal Engine

El videojuego ha sido desarrollado en Unreal Engine 5.3.2. Teóricamente es compatible con versiones posteriores, pero no se ha probado su funcionamiento en ellas. Para instalar Unreal Engine, puedes seguir la guía oficial en el siguiente enlace:

[https://dev.epicgames.com/documentation/en-us/unreal-engine/  
installing-unreal-engine](https://dev.epicgames.com/documentation/en-us/unreal-engine/installing-unreal-engine)

Probablemente esta sea la parte más pesada de la instalación, ya que Unreal Engine es un programa muy pesado y puede tardar varias horas en descargarse e instalarse completamente. Además, requiere de otros programas para funcionar correctamente. Todo ello está explicado en el enlace anterior.

Si no sabes si tu PC cumple con los requisitos mínimos para instalar Unreal Engine, puedes consultarlos en el siguiente enlace:

<https://dev.epicgames.com/documentation/en-us/unreal-engine/hardware-and-software-specifications-for-unreal-engine>

## 2.3. MindMaker

Para que este proyecto funcione, es necesario que tengas instalado un plugin en Unreal Engine llamado MindMaker. Este plugin incluye la biblioteca de Socket.IO, que es necesaria para la comunicación entre Unreal Engine y Python. Puedes descargarlo este software gratuito desde el siguiente enlace:

<https://www.unrealengine.com/marketplace/en-US/product/mindmaker-ai-plugin>

## 2.4. Python

Todo el código relativo a la IA se ha desarrollado en Python 3.11, aunque cualquier versión posterior también servirá. Para instalar Python, solo tienes que descargarte el ejecutable desde el siguiente enlace y seguir el instalador. Asegúrate de marcar la casilla de verificación que dice "Add Python to PATH" para poder ejecutar Python desde la terminal de Windows.

<https://www.python.org/downloads/>

### 2.5. Stable Baselines 3

Stable Baselines 3 se ha utilizado para entrenar la IA mediante sus algoritmos de aprendizaje por refuerzo. Para instalarlo, solo tienes que ejecutar el siguiente comando en la terminal de Windows después de haber instalado Python:

```
pip install stable-baselines3
```

### 2.6. SB3 Zoo

SB3 Zoo es una biblioteca auxiliar de Stable Baselines 3 que contiene diferentes agentes preentrenados y funcionalidad de ajuste de hiperparámetros mediante Optuna. Para instalarlo, solo tienes que ejecutar el siguiente comando en la terminal de Windows:

```
pip install rl_zoo3
```

### 2.7. Gymnasium

Gymnasium es una biblioteca que contiene herramientas para diseñar entornos de aprendizaje por refuerzo. Para instalarlo, solo tienes que ejecutar el siguiente comando en la terminal de Windows:

```
pip install gymnasium
```

Es importante no confundir Gymnasium con Gym, ya que al ser una versión más moderna, el código de Gymnasium es ligeramente diferente al de Gym.

### 2.8. Otras bibliotecas

Es posible que al intentar ejecutar el código en Python, te aparezcan errores relacionados con la falta de bibliotecas. Por ejemplo, puede ser que

te falte NumPy, Matplotlib, TensorFlow, etc. Para instalar estas bibliotecas, solo tienes buscar su nombre en tu buscador favorito y seguir las instrucciones de instalación. Generalmente, usar la versión de PIP es la más sencilla.

## 2.9. Editores

Para revisar el código de C++ y Python, es muy aconsejable usar un editor de texto pensado para ello. Para el código de Unreal Engine en C++ se recomienda Rider, mientras que para el código de Python se recomienda Visual Studio Code (VSCode).

Aunque Rider es la herramienta estándar para programar en Unreal Engine, no es gratuita. Por lo que si sólo quieres revisar el código, también puedes usar VSCode para ello. Puedes descargarlos desde los siguientes enlaces:

<https://www.jetbrains.com/es-es/rider/download/#section=windows>

<https://code.visualstudio.com/download>

## 2.10. Proyecto

Por último, es necesario descargar el proyecto de este Trabajo de Fin de Grado. Puedes hacerlo desde el siguiente enlace en Gitlab:

<https://gitlab.com/Silver812/tfg>

Para descargar el proyecto desde Gitlab, puedes simplemente hacer clic en el botón azul que dice Code y luego en en Zip. Otra forma de hacerlo es clonar el repositorio con Git. En la siguiente Figura 2.1 se muestra la interfaz de descarga.



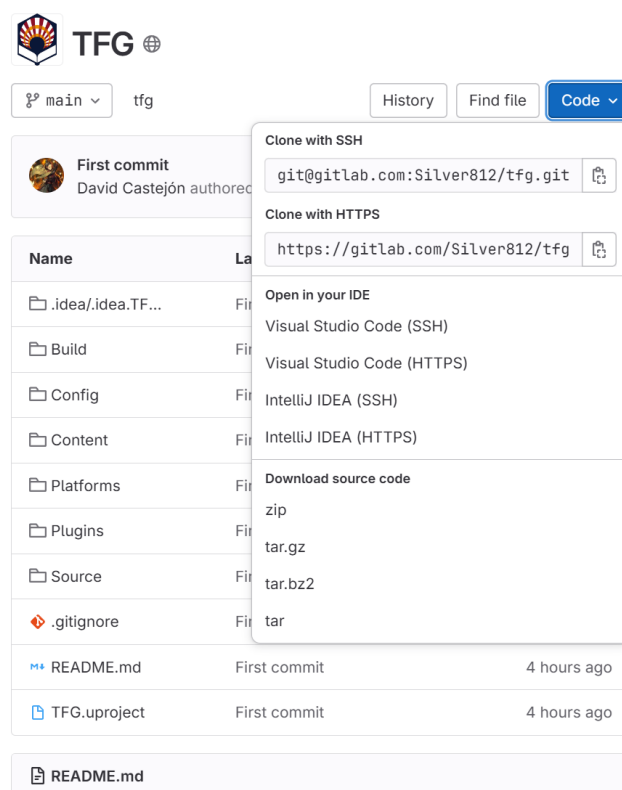


Figura 2.1: Descarga del proyecto desde Gitlab

## Capítulo 3

# Instalación de los entornos

Una vez descargados e instalados todos los programas y bibliotecas necesarios, es hora de instalar los entornos de aprendizaje por refuerzo desarrollados durante este proyecto. Para ello, solo tienes que seguir los siguientes pasos:

1. Abre una terminal en la siguiente ruta:

```
<ruta_del_proyecto>\TFG\Plugins\GameFeatures\FiringRange\  
Source\rl_code.
```

2. Ejecuta el siguiente comando:

```
pip install -e .
```

De esta forma, podrás acceder a los tres entornos de Gymnasium como si fueran bibliotecas de Python. Esto es necesario para realizar los entrenamientos, evaluaciones y pruebas de los modelos de aprendizaje por refuerzo.

Para comprobar que se han instalado correctamente, puede ejecutar el *script* que se encuentra en la siguiente ruta:

```
<ruta_del_proyecto>\TFG\Plugins\GameFeatures\FiringRange\  
Source\rl_code\utils\zoo.py.
```

## Capítulo 4

# Guía de uso

En este capítulo se detallan los pasos necesarios para poder ejecutar el juego en el editor de Unreal Engine e iniciar el servidor local de Python para que la IA pueda jugar. Se asume que el lector tiene conocimientos básicos de Unreal Engine y Python y que ya ha instalado los programas necesarios para ejecutar el proyecto.

1. Abrir la clase de Blueprint *B\_LyraGameMode* en UE y seleccionar la variable *RLEngineStruct*. Esta variable tiene todos los parámetros necesarios para configurar el episodio de entrenamiento. En la figura [4.1](#) se pueden ver los parámetros que se pueden ajustar. Entre ellos, se destacan:

- a) *MaxTimesteps*: número máximo de pasos que puede realizar la IA en un episodio.
- b) *DistancePercentage*: ajusta cuánto espacio la IA mueve la cámara en cada paso.
- c) *UpdateTime*: cambia cada cuánto tiempo el *script* de Python comprueba si hay una nueva observación en UE.
- d) *RLAlgorithm*: indica el algoritmo de aprendizaje por refuerzo con el que se ha entrenado el agente que se quiere utilizar.

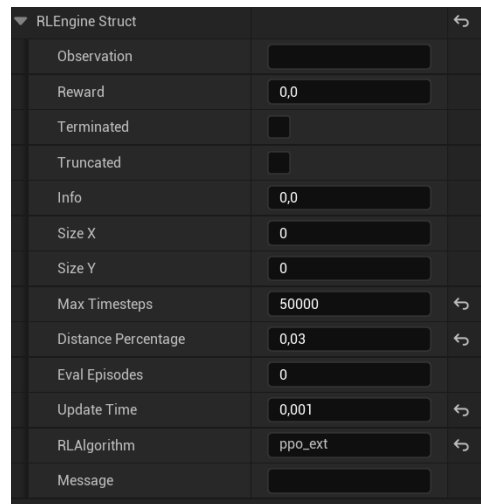


Figura 4.1: Ajustes del episodio desde UE

2. Ejecutar el *script* de Python `rl_engine.py` en la terminal de Windows.

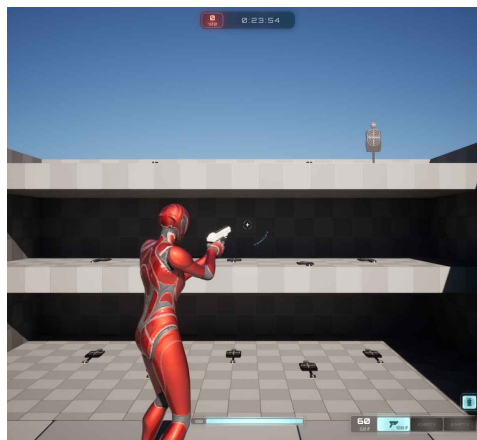


Figura 4.2: Juego en el editor de Unreal Engine

3. Ajustar la pantalla del editor para que se vea relativamente cuadrada.  
Esto no es estrictamente necesario, pero permite que el movimiento de la sea más consistente, ya que esta toma las dimensiones de la pantalla para calcular la distancia que se puede mover. En la figura 4.2 se aprecia esta forma cuadrada en juego.

4. Ejecutar el juego en el editor de Unreal Engine. Para ello, solo hay que pulsar el botón verde de *Play* en la parte superior de la pantalla, como se muestra en la figura 4.3.



Figura 4.3: Botón para ejecutar el juego en el editor de UE

Con estos pasos, se debería empezar a mover sola la cámara en el juego hacia un maniquí. Si no se moviera, algunos de los pasos anteriores se habrá realizado incorrectamente. En el caso de todo haya funcionado correctamente, debería de verse como en el siguiente vídeo <https://youtu.be/GV67p5SrTTc>.

Por último, si en lugar de dejar que la IA juegue, se quiere jugar al juego, con no ejecutar el *script* de Python será suficiente. En este caso, simplemente hay pulsar el botón de *Play* y empezar a jugar.