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, 10 de junio de 2024

Índice general

Índice de figuras

Capítulo 1

Introducción

Como bien indica el titulo 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 presentan enlaces de descarga y 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

Durante el proyecto se ha usado 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

Probablemente esta sea la parte más pesada de la instalación, ya que Unreal Engine es un programa muy complejo 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 descargar 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 bibloteca 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.

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, Pytorch, etc. Para instalar estas bibliotecas, solo tienes buscar su nombre en tu buscador favorito y seguir las instrucciones de instalación. Personalmente, usar la versión de PIP suele ser la forma más sencilla de instalación.

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.

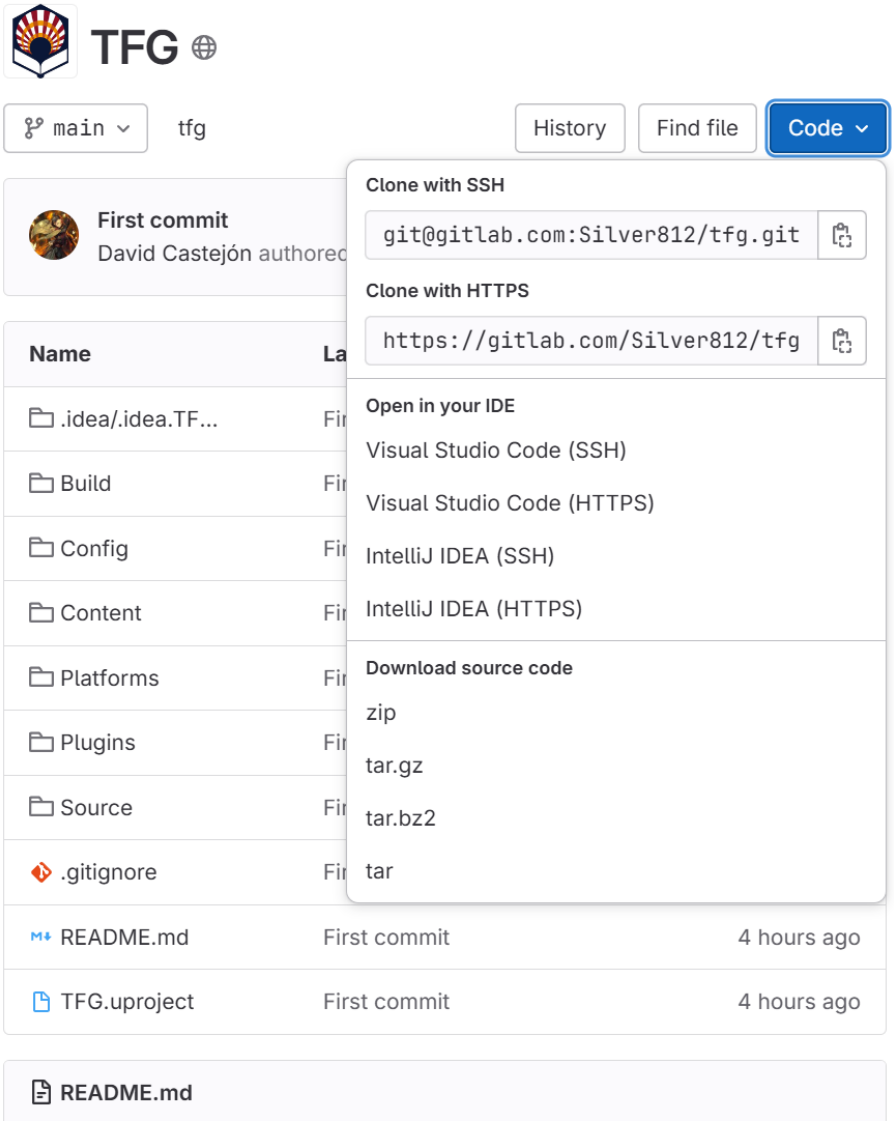

Figura 2.1: Descarga del proyecto desde Gitlab

Capítulo 3

Instalación y desinstalación 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.

El siguiente paso requiere ir a la ruta donde se instaló SB3 Zoo. En el caso de haberlo hecho con PIP, esta es la ruta:

`C:\Users\<usuario>\AppData\Local\Programs\Python\<version_de_python>\Lib\site-packages\rl_zoo3\hyperparams\ppo.yml`.

Allí, se tendrán que añadir los hiperparámetros con los que se quieren entrenar nuevos agentes (si no fuera el caso, este paso no es necesario). En la Figura 3.1 se muestra un ejemplo de cómo debería de quedar. Esta imagen está tomada dentro del archivo llamado *ppo.yml*, pero se puede hacer lo mismo en *a2c.yml* en caso de querer hacer pruebas de entrenamiento, evaluación o ajuste de hiperparámetros en los entornos de este proyecto usando SB3 Zoo (hiperparámetros optimizados en el manual técnico). Para el caso de PPO con las acciones extendidas, solo hay que copiar el mismo texto excepto con *CenteringWorld-v1* en la cabecera, ya que es una version diferente del mismo entorno.

```
CenteringWorld-v0:
  n_envs: 16
  n_timesteps: !!float 1e5
  policy: 'MlpPolicy'
  n_steps: 128
  batch_size: 256
  gae_lambda: 0.9
  gamma: 0.98
  n_epochs: 5
  ent_coef: 0.00017416557234567777
  learning_rate: 0.006738815018609934
  clip_range: 0.4
  max_grad_norm: 0.8
  vf_coef: 0.5419630056693383
  policy_kwargs: "dict(
                  activation_fn=nn.ReLU,
                  net_arch=dict(pi=[64], vf=[64])
                  )"
```

Figura 3.1: Parámetros en el caso de PPO por defecto

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

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

Para desinstalar los entornos simplemente se tienen que borrar los archivos creados tras el comando:

```
pip install -e .
```

El paquete con los entornos debería estar en la siguiente ruta:

`C:\Users\<usuario>\AppData\Local\Programs\Python\<version_de_python>\Lib\site-packages\rl_algorithm`.

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 el proyecto de Unreal Engine. Con hacer doble clic sobre *TFG.uproject* debería ser suficiente.

2. 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.

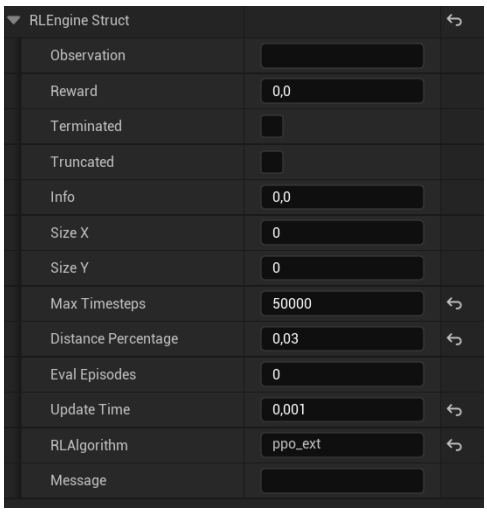

Figura 4.1: Ajustes del episodio desde UE

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

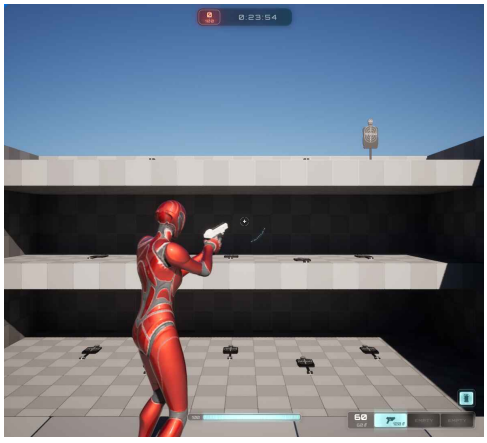

Figura 4.2: Juego en el editor de Unreal Engine

4. Redimensionar la pantalla del editor (PIE) para que se vea relativa-

mente 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 el juego del editor.

5. 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.

Si se quisiera cancelar el episodio, con darle al botón de Escape en el teclado sería suficiente.

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 con las teclas A, W, S, D para el movimiento del personaje y el ratón para mover la cámara y disparar.