Práctica 01

Aprendizaje por refuerzo y redes neuronales

Objetivo: Esta práctica consiste en la implementación, entrenamiento y validación de la solución encontrada de un *algoritmo por refuerzo* utilizando redes neuronales. Los objetivos concretos consisten en:

1.- El robot sea capaz de acercase a un objeto.

2.- El robot debe de ser capaz de seguir el objeto, independientemente de si este está fijo o en movimiento.

Al mismo tiempo, esto debe de cumplirse independientemente de la posición inicial de cada objeto.

Herramientas: Para ello, se hará uso de librerías y herramientas ampliamente utilizadas en la comunidad, tales como el entorno de trabajo de Gymnasium (https://gymnasium.farama.org/), la librería StableBaselines3 (https://stable-baselines3.readthedocs.io/en/master/). No obstante, estas son las principales, y cualquier otra librería puede ser utilizada/necesaria para llevar a cabo la práctica.

Como robot, se utilizará el robot Robobo, en su versión en simulación (está pendiente la implementación de confirmación la implementación en el robot real):

- Página principal del proyecto Robobo: https://therobobоproject.com
- Simulador RoboboSim: https://github.com/mintforpeople/robobo-programming/wiki/Unity
- Librería "Robobo.py" para programar el Robobo: https://github.com/mintforpeople/robobo-programming/wiki/python-doc https://mintforpeople.github.io/robobo.py/
- Librería "Robobosim.py" para utilizar funcionalidades exclusivas del simulador: https://github.com/mintforpeople/robobo-programming/wiki/robobosimpy

El escenario de trabajo que se utilizará con RoboboSim será el que se observa en la Figura 1.

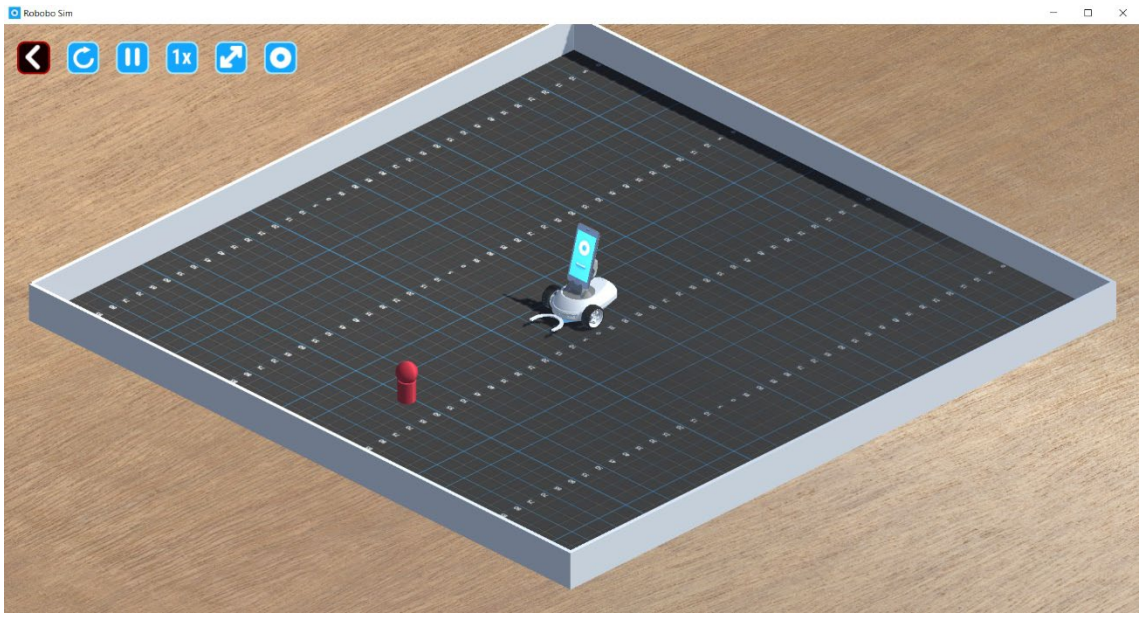

Figura 1. Escenario "cylinder" de RoboboSim

Detalles de la práctica:

El programa deberá seguir la estructura propia de los programas/scripts desarrollados en el marco de Gymnasium, pues hay que tener en cuenta que es la estructura que utiliza la librería StableBaselines3:

https://stable-baselines3.readthedocs.io/en/master/guide/custom_env.html

Por ello, deberá de crearse un entorno de trabajo (environment) adaptado de RoboboSim al marco de trabajo de Gymnasium.

Esto implica que, teniendo en cuenta que es una práctica de aprendizaje por refuerzo, habrá que definir los elementos básicos que se emplean en todo algoritmo por refuerzo:

- Espacio de observaciones/espacio de estados: Está definido por la información que el robot necesita y que recibe del entorno. En general, sensorización del entorno y del propio robot.
- Espacio de acciones: Está definido por el conjunto de acciones que el robot puede realizar en el entorno. Acciones que puede realizar el robot.
- Función de recompensa: Define el valor de recompensa que recibe el robot en base al nuevo estado alcanzado, gracias a la acción que ha realizado en el estado previo.

- Política y algoritmo de aprendizaje: Esta dupla dependerá de la selección que se haga en la librería StableBaselines3, pero debe de quedar clara y justificada. Como sugerencia, se recomienda utilizar PPO.

Esto servirá para definir los elementos básicos del sistema.

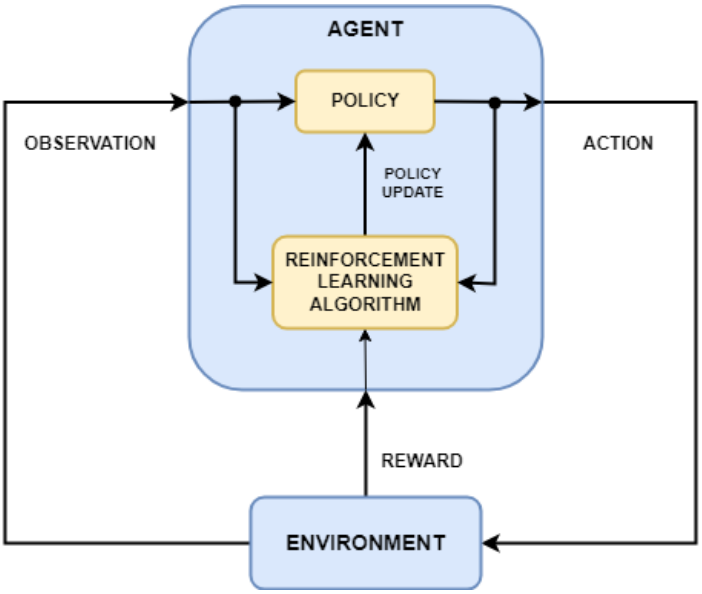

Figura 2. Estructura básica de un algoritmo por refuerzo, imagen extraída de MatlanHelp Center: https://www.mathworks.com/help/reinforcement-learning/ug/what-is-reinforcement-learning.html

Entrenamiento:

El proceso de entrenamiento consiste en ejecutar el bucle de la Figura 2 e ir actualizando la política (policy) para que el valor de la recompensa vaya aumentando con el paso de las iteraciones.

En el entrenamiento, habrá que establecer cuantos pasos de tiempo se entrena el modelo, cada cuanto se actualiza la política, etc.

NOTA: Se sugiere emplear la física simplificada y aumentar la velocidad de simulación a x10 para acelerar el aprendizaje.

Validación:

La política aprendida deberá de poder cargarse en un nuevo programa o script, para verificar su correcto funcionamiento.

Representación de los resultados:

Los resultados de entrenamiento, de métricas como “mean_reward” o “ep_rew_mean” aunque se guardan por defecto en tensorboard, deben de ser representadas en un archivo “.png”, “.jpeg” o similar, utilizando alguna otra librería de Python, tal como seaborn o similar.

Así como, representar en un plano 2D, las diferentes posiciones que ha ido tomando el robot.

Criterio de Evaluación:

- Definición del problema e implementación de los elementos básicos: (hasta 4 puntos):
    - Implementación del entorno de trabajo de RobobSim adaptado a Gymnasium.
    - Definición y complejidad de los espacios de estados, de acciones y función de recompensa. Cuanto mayor sea la complejidad y definición de los mismos, mayor será la puntuación. Ejemplo: la utilización de espacios continuos frente a espacios discretos es más compleja, pero se valorará más.
    - Se valorará el refinamiento de la función de recompensa, teniendo no sólo en cuenta las recompensas, sino también las posibles penalizaciones por incurrir en estados no deseados.
- Implementación del algoritmo de aprendizaje y calidad de la solución propuesta (hasta 4 puntos):
    - Rapidez con la que se obtiene una solución aceptable (en pasos de tiempo)
    - Calidad de la solución obtenida. Es decir, consistencia con la que se aproxima al objetivo y rapidez con la que se aproxima.
- Representación de la información en un formato distinto a tensorboard (hasta 2 puntos):
    - Representación de las métricas más relevantes dadas por el StableBaselines3 utilizando una librería diferente a tensorboard.
    - Representación en un plano 2D de las diferentes posiciones recorridas por el Robobo (y las posiciones que ha tomado el cilindro de ser el caso)

Se evaluará el desempeño del sistema, así como la originalidad y complejidad de la propuesta y la solución.

Entrega:
Viernes 10 de octubre. A las 23:59. La práctica se deberá de entregar en un archivo ".zip" que contenga:

- Pequeña memoria de 4 páginas máximo con fuente a tamaño 12.
- El código será entregado y deberá de ejecutarse sin problema. Se indicarán que librerías/dependencias son necesarias para su ejecución.

Entregar esto se entregará a través de **Moodle** (no correo).