Aprendizaje por refuerzo

PRA1: Implementación de un agente para la robótica espacial

Andrés Esteban Merino Toapanta

23 de enero de 2023

Universitat Oberta de Catalunya

PRA1: Implementación de un agente para la robótica espacial

Índice

Introducción

En el presente trabajo, se propondrá, entrenará y comparará dos agentes que tienen como objetivo solucionar un problema de robótica espacial: aterrizaje autónomo. Para realizar la interacción, se utilizó el entorno **lunar-lander** de la librería `gym` de OpenAI.

En la primera sección se realiza la exploración del entorno, junto con pruebas básicas del mismo para un agente aleatorio. En la segunda sección se propone y entrena un agente DQN, se muestran sus resultados de entrenamiento y una prueba del agente. Finalmente, en la tercera sección, se propone y entrena un agente A2C con la finalidad de controlar la variabilidad del aprendizaje, se muestran sus resultados de entrenamiento y se compara con el agente de la segunda sección.

1 Entorno

1.1 Exploración del entorno

El entorno presenta un espacio de 4 acciones, con una dimensión para el espacio de observaciones de 8. El número máximo de pasos en cada episodio es 1000 y el rango de recompensas es todos los números reales.

Al realizar una ejecución aleatoria en el entorno, se obtuvo una recompensa de $-145{,}2$ y se pueden apreciar algunas capturas de la misma en la Figura 1.

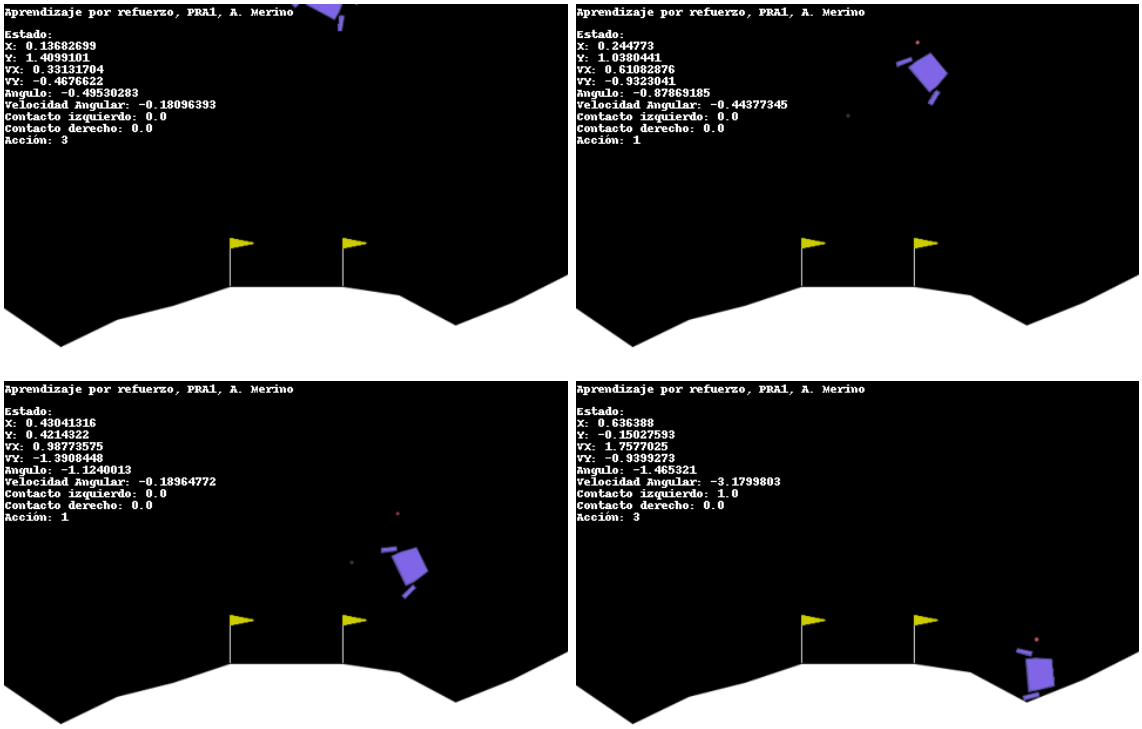

Figura 1: Ejecución aleatoria en el entorno.

Para tener una mejor apreciación del comportamiento del entorno bajo un agente aleatorio, se realizó una ejecución de 500 episodios, obteniendo una recompensa promedio de $-178{,}94$. En la Figura 2 se muestran los histogramas de las recompensas obtenidas y los pasos realizados en cada episodio. Se puede apreciar que la todos los episodios se terminan en menos de 150 pasos, por otro lado, casi todos los episodios tienen una recompensa negativa, llegando, incluso, a tener recompensas menores a -400.

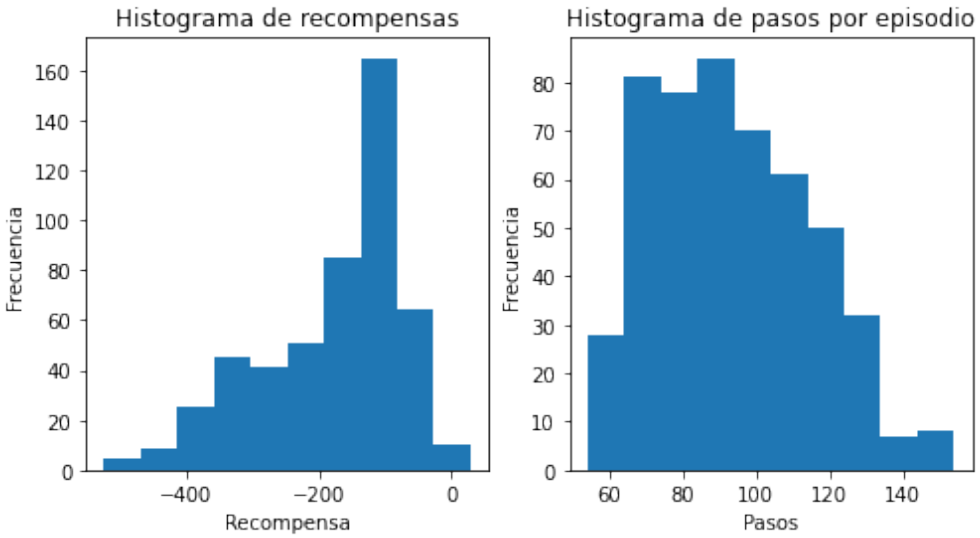

Figura 2: Histogramas de recompensas y pasos realizados en 500 episodios con agente aleatorio.

1.2 Espacio de observaciones y de acciones

El espacio de observaciones del entorno es de 8 dimensiones, las cuales son:

- Posición horizontal del módulo lunar: $[-1{,}5,\ 1{,}5]$.
- Posición vertical del módulo lunar: $[-1{,}5,\ 1{,}5]$.
- Velocidad horizontal del módulo lunar: $[-5,\ 5]$.
- Velocidad vertical del módulo lunar: $[-5,\ 5]$.
- Ángulo del módulo lunar: $[-3{,}14,\ 3{,}14]$.
- Velocidad angular del módulo lunar: $[-5,\ 5]$.
- Indicador de contacto con el suelo del lado izquierdo: $\{0,\ 1\}$.
- Indicador de contacto con el suelo del lado derecho: $\{0,\ 1\}$.

El espacio de acciones del entorno posee las siguientes 4:

- 0: No hacer nada.
- 1: Encender el motor izquierdo.
- 2: Encender el motor derecho.
- 3: Encender el motor principal.

Podemos apreciar estos valores en la esquina superior izquierda de la Figura 3.

Figura 3: Ejecución aleatoria en el entorno.

Dado que el espacio de observaciones es continuo (salvo las dos últimas componentes), no es posible aplicar soluciones tabulares para este problema; por lo tanto, se aplicarán soluciones aproximadas.

2 Agente de referencia

2.1 Implementación agente de referencia

Como referencia, se utilizó un agente DQN con replay buffer y target network. Para la red se utilizó la siguiente configuración:

- Una capa de entrada con 8 entradas y 64 salidas, con activación ReLU.
- Una capa oculta con 64 entradas y 64 salidas, con activación ReLU.
- Una capa de salida con 64 entradas y 4 salidas.

Para el entrenamiento, en el agente se utilizó la función de pérdida MSE, con un optimizador Adam. El código fuente del agente se encuentra en el archivo adjunto.

2.2 Entrenamiento agente de referencia

Para buscar los hiperparámetros del agente, se realizó un entrenamiento con 500 episodios. Se analizaron los siguientes hiperparámetros: tasa de aprendizaje, gamma (factor de descuento) y decaimiento del épsilon (factor de exploración).

Con respecto a la tasa de aprendizaje, se realizaron 3 experimentos con tasa de aprendizaje de 0,05, 0,005 y 0,0005. En todos los casos, el agente logró realizar aprendizaje, pero con una tasa de aprendizaje de 0,005 se obtuvieron los mejores resultados, como se puede apreciar en la Figura 4.

Figura 4: Evolución de la recompensa promedio del agente DQN en 500 episodios con tasa de aprendizaje de 0,05, 0,005 y 0,0005.

Por otro lado, se realizaron 3 experimentos con gamma de 0,99, 0,999 y 0,9999. De igual manera, en todos los casos, el agente logró realizar aprendizaje, pero con gamma de 0,9999 se obtuvieron los mejores resultados, a pesar de que, inicialmente, no se aprecia un buen comportamiento, a partir de los 250 episodios, tiene cierto nivel de ventaja. Esto se puede apreciar en la Figura 5.

Figura 5: Evolución de la recompensa promedio del agente DQN en 500 episodios con gamma de 0,99, 0,999 y 0,9999.

Finalmente, se realizaron 3 experimentos con decaimiento del épsilon de 0,99, 0,999 y 0,9999. En todos los casos, el agente logró realizar aprendizaje, sin embargo, con 0,99 se obtuvo un rendimiento significativamente superior, pues en el resto de casos, el aprendizaje parece estancarse. Esto se puede apreciar en la Figura 6.

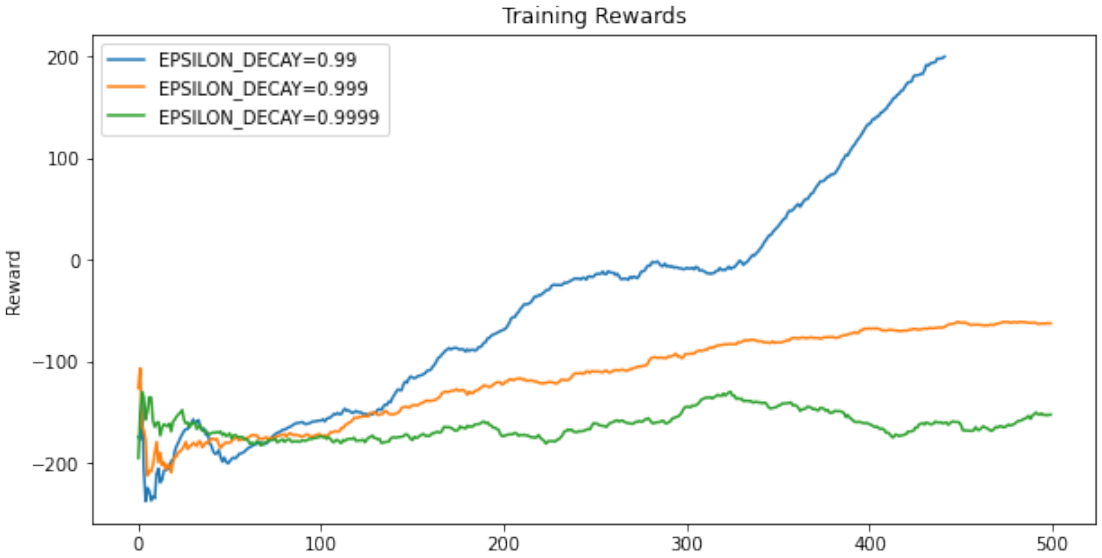

Figura 6: Evolución de la recompensa promedio del agente DQN en 500 episodios con decaimiento del epsilon de 0,99, 0,999 y 0,9999.

De esta manera, se obtuvieron los siguientes hiperparámetros: tasa de apren-

dizaje de 0,005, gamma de 0,9999 y decaimiento del épsilon de 0,99. Con estos hiperparámetros, se realizó un entrenamiento con 2500 episodios, sin embargo, se resolvió el entorno (se obtuvo una media de recompensa superior a 200) en 443 episodios, utilizando 353 segundos. El comportamiento presentado durante el aprendizaje se puede apreciar en las Figura 7 y 8.

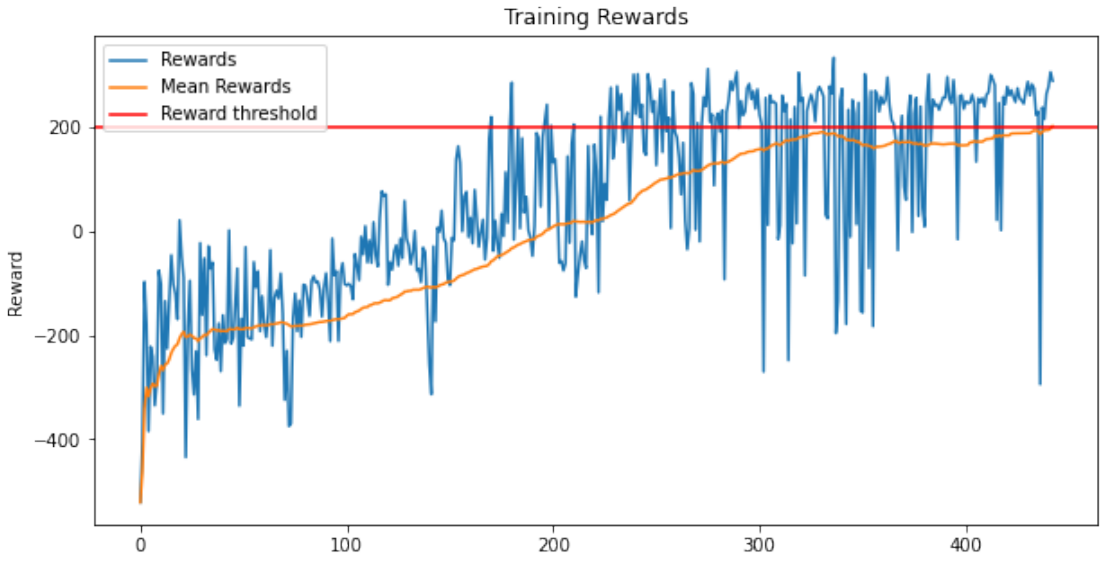

Figura 7: Evolución de recompensas durante el entrenamiento con un agente DQN.

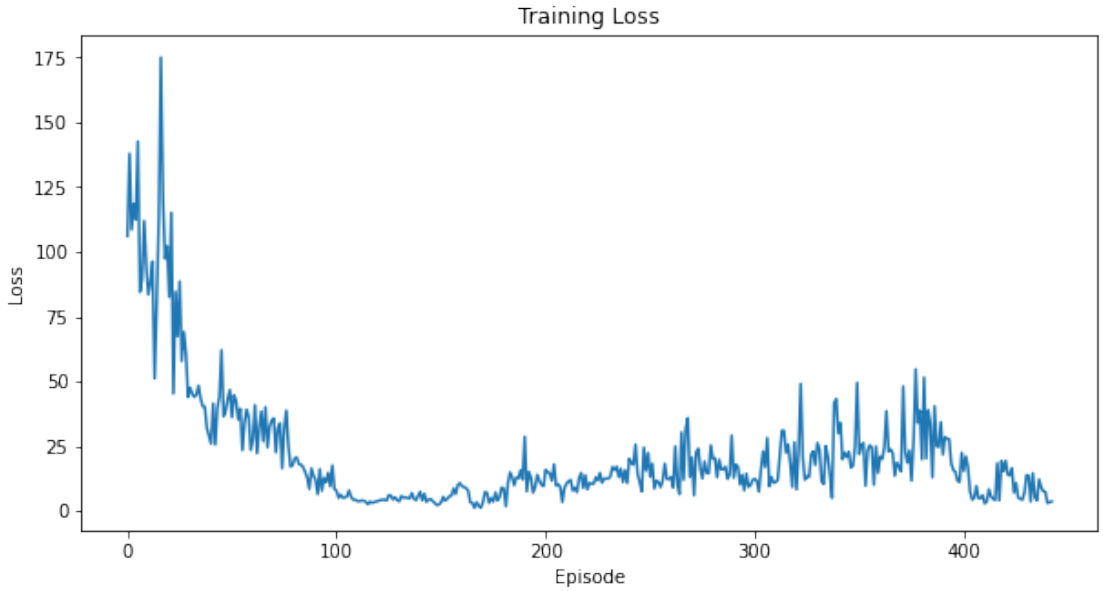

Figura 8: Evolución de la función de pérdida durante el entrenamiento con un agente DQN.

Como se puede apreciar, a pesar de alcanzar una media de aprendizaje que

supera el umbral de 200, las recompensas de cada episodio son fluctuantes, lo que indica que el agente no ha aprendido completamente el entorno. Por otro lado, la función de pérdida decrece de manera estable hasta los 150 episodios, luego de lo cual comienza a fluctuar, lo que señala un problema de convergencia en el aprendizaje.

2.3 Prueba agente de referencia

Al realizar una ejecución del agente entrenado, se obtuvo una recompensa de 188,03 y se pueden apreciar algunas capturas de la misma en la Figura 9.

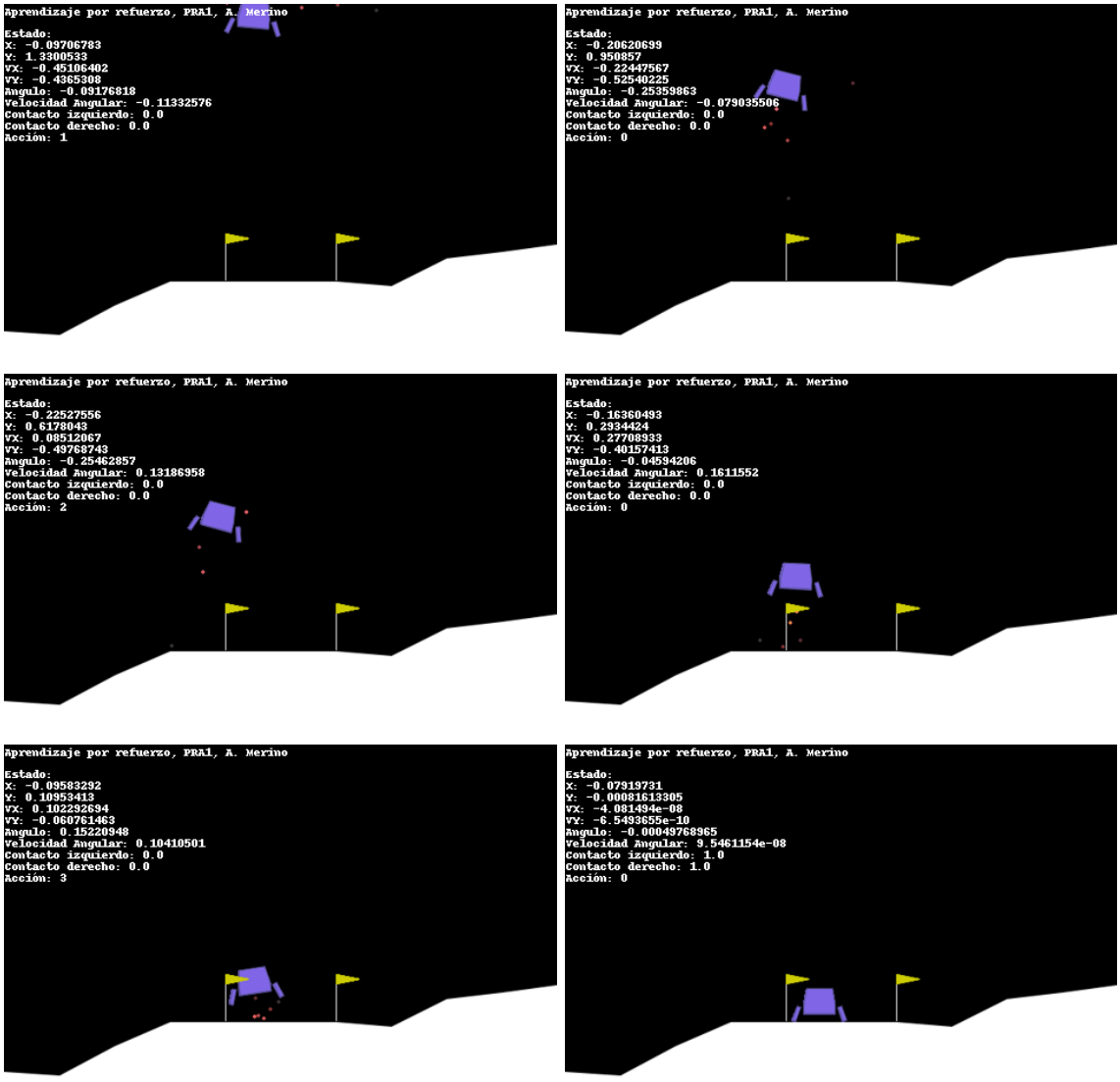

Figura 9: Ejecución con el agente DQN entrenado.

Para apreciar mejor el comportamiento del agente, se realizaron 500 ejecu-

ciones del agente entrenado, obteniendo una media de recompensa de 237,79, con una desviación estándar de 67,76. En la Figura 10 se puede apreciar la distribución de las recompensas obtenidas y del número de pasos en cada episodio. Podemos apreciar que aunque la gran mayoría de las ejecuciones obtuvieron una recompensa superior a 200, existen episodios que terminaron con recompensa negativa. De igual manera, existen episodios que terminaron con en 1000 pasos, lo que indica que, posiblemente, no tocaron el suelo.

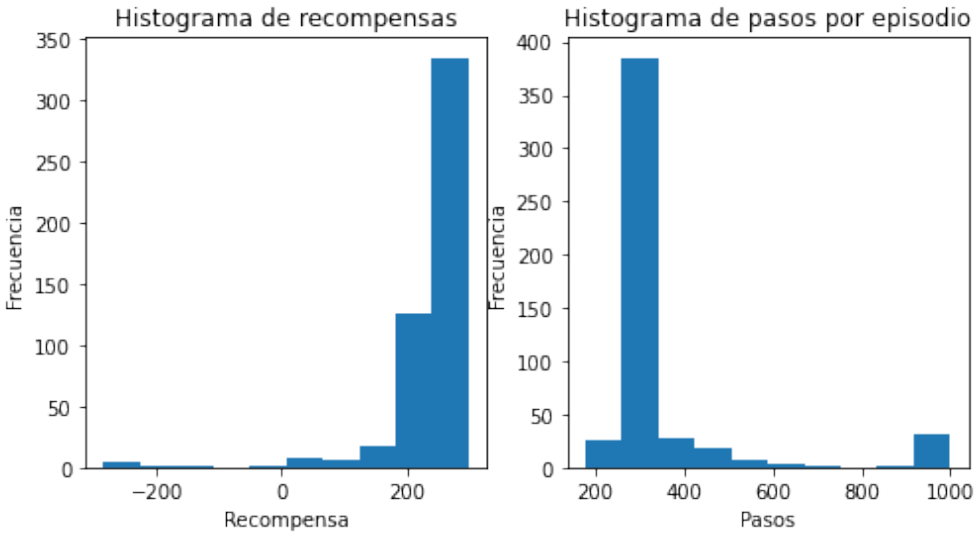

Figura 10: Distribución de recompensas y número de pasos en 500 ejecuciones del agente DQN entrenado.

3 Propuesta de mejora

3.1 Implementación agente identificado

Con el objetivo de reducir la variabilidad en las recompensas, se procede a implementar un agente A2C, el cual, con la red del crítico, evitaría que, durante el aprendizaje, se tomen decisiones erróneas que fomente la variabilidad de los resultados. Para la red del agente se utilizó la siguiente configuración:

- Una capa de entrada con 8 entradas y 64 salidas, con activación ReLU.
- Una capa oculta con 64 entradas y 64 salidas, con activación ReLU.
- Una capa de salida con activación softmax.

Para la red del crítico se utilizó la misma configuración, pero sin la activación softmax en la capa de salida. Para ambas redes se utilizó la función de pérdida que suma las pérdidas de cada red, es decir, la pérdida del agente y la pérdida del crítico. La función de pérdida del agente se usó MSE, mientras que para el crítico se utilizó la media de la suma de las acciones por la probabilidad de cada acción dada por el actor, multiplicado por las ventajas. Finalmente, se aplicó a cada red un optimizador del tipo RMSprop. El código de la implementación se encuentra en el archivo adjunto.

3.2 Entrenamiento agente identificado

Para buscar los hiperparámetros óptimos, se realizó un entrenamiento con 500 episodios. Se estudiaron los siguientes hiperparámetros:tasa de aprendizaje y factor de descuento (gamma). Para la tasa de aprendizaje se utilizaron los valores 0,005, 0,0005 y 0,00005, mientras que para el factor de descuento se utilizaron los valores 0,99, 0,999 y 0,9999.

Con respecto a la tasa de aprendizaje, se obtuvieron los resultados de la Figura 11. Se puede apreciar que, a pesar de que con una tasa de 0,005 se consiguen mejores resultados hasta antes de los 200 episodios, se tiene una caída abrupta, mientras que con una tasa de 0,0005 se tiene una mejora constante en el aprendizaje.

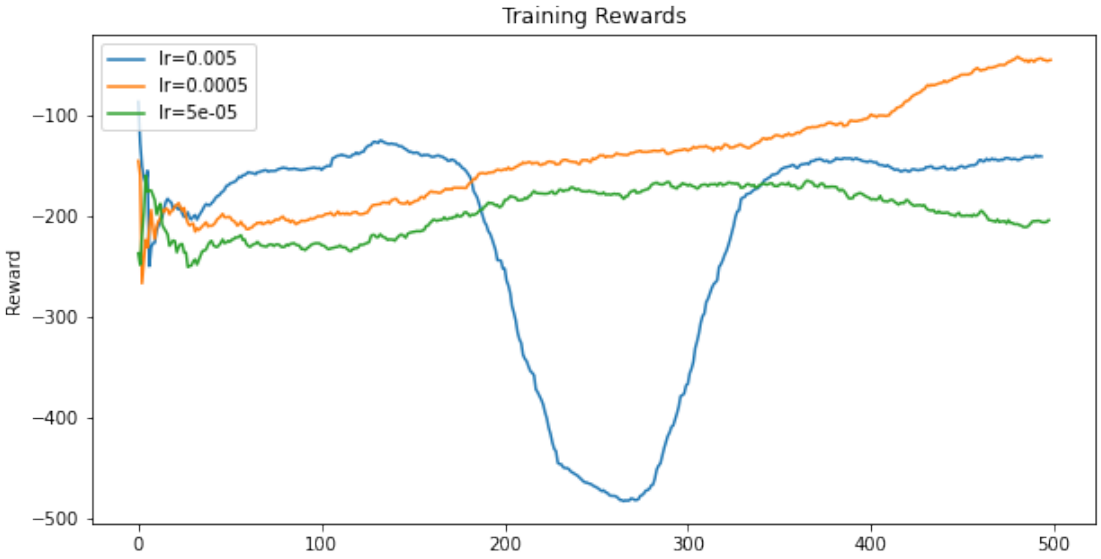

Figura 11: Evolución de la recompensa promedio del agente A2C en 500 episodios con tasa de aprendizaje de 0,005, 0,0005 y 0,00005.

Por otro lado, con respecto al factor de descuento, se obtuvieron los resultados de la Figura 12. Se puede apreciar que el valor de 0,999 produce un comportamiento siempre ascendente, mientras que los valores de 0,99 y 0,9999 producen comportamientos en los cuales se pierde el aprendizaje.

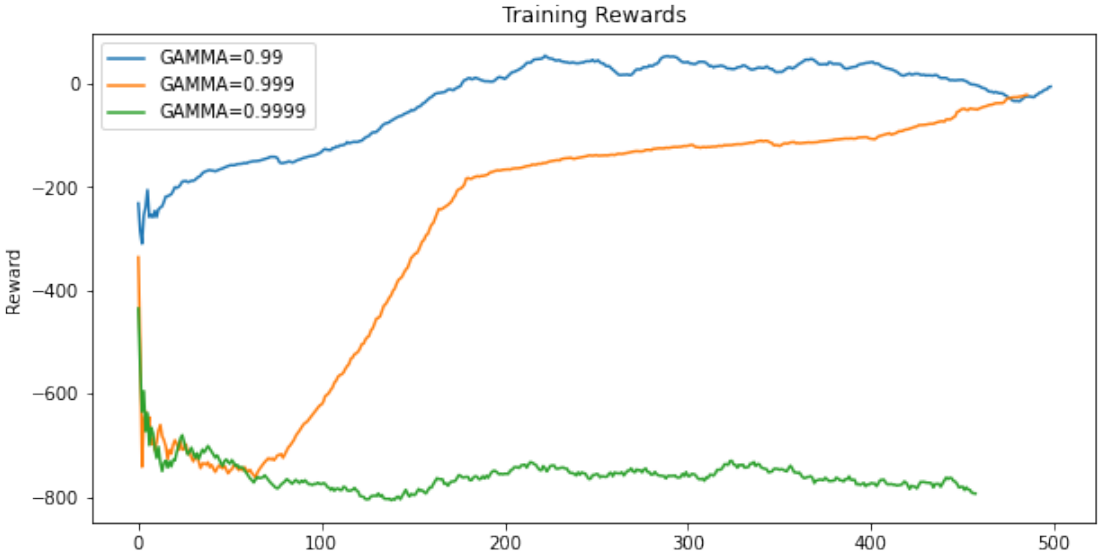

Figura 12: Evolución de la recompensa promedio del agente A2C en 500 episodios con factor de descuento de 0,99, 0,999 y 0,9999.

De esta manera, se obtuvieron los siguientes hiperparámetros: tasa de aprendizaje de 0,0005 y factor de descuento de 0,999. Con estos hiperparámetros se

realizó un entrenamiento con 4000 episodios, sin embargo, se resolvió el entorno en 1517 episodios, utilizando 796 segundos. El comportamiento presentado durante el aprendizaje se puede apreciar en las Figuras 17 y 14.

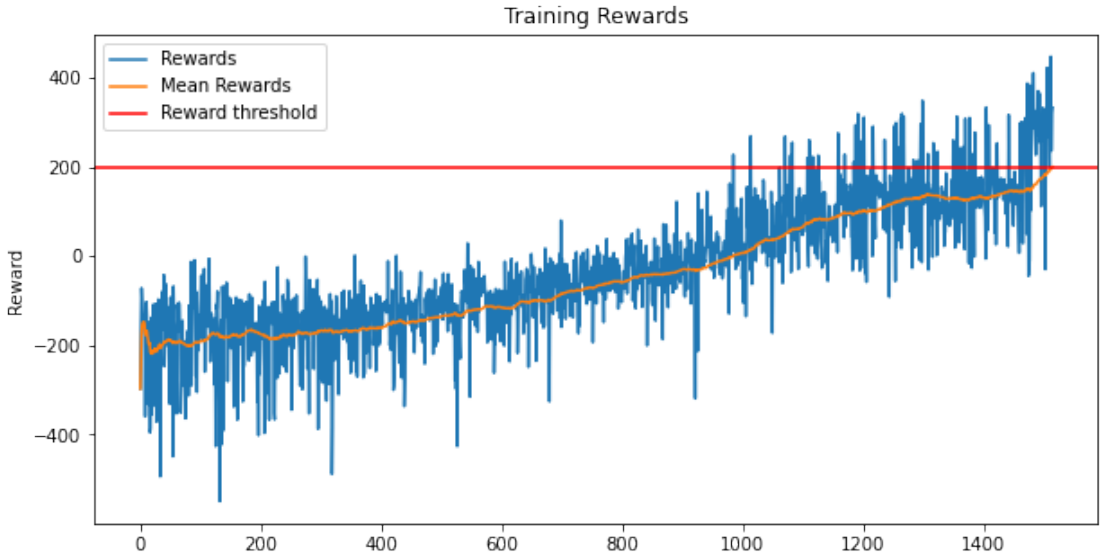

Figura 13: Evolución de la recompensa durante el entrenamiento con un agente A2C.

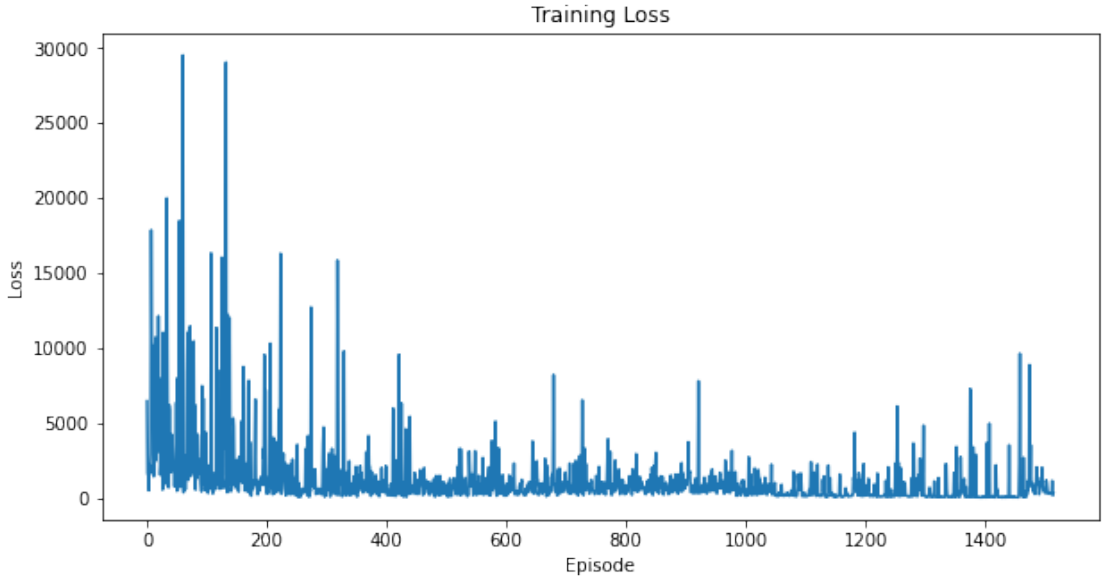

Figura 14: Evolución de la función de pérdida durante el entrenamiento con un agente A2C.

Como se puede apreciar, se tiene un aprendizaje más estable aunque más lento, con una posibilidad aparente de seguir aprendiendo. Respecto a la función de pérdida, se observa un decrecimiento abrupto inicialmente, pero aún mantie-

ne una fluctuación.

3.3 Prueba del agente identificado y comparación

Para apreciar el comportamiento del agente, se realizaron 500 ejecuciones con el agente entrenado, obteniendo una recompensa promedio de 237,89 y una desviación estándar de 75,08. En la Figura 15 se puede apreciar la distribución de las recompensas obtenidas y del número de pasos de cada episodio. Se aprecia que se obtienen recompensas más altas en comparación al agente DQN, de igual manera se tiene una menor cantidad de episodios que llegan a los mil pasos.

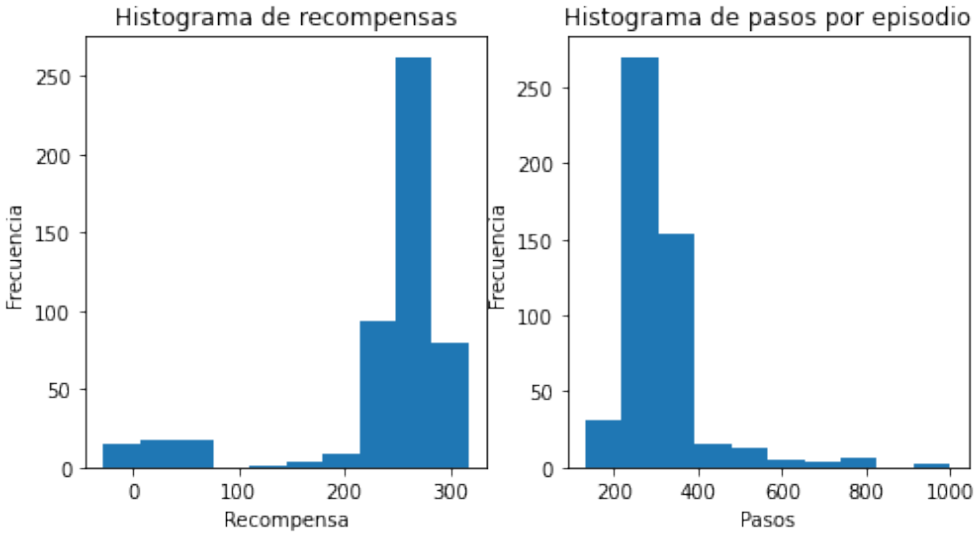

Figura 15: Distribución de las recompensas obtenidas y del número de pasos en 500 ejecuciones del agente A2C entrenado.

También, se realizó una ejecución del agente identificado, se obtuvo una recompensa de 232,41 y se puede apreciar algunas capturas de la misma en la Figura 16.

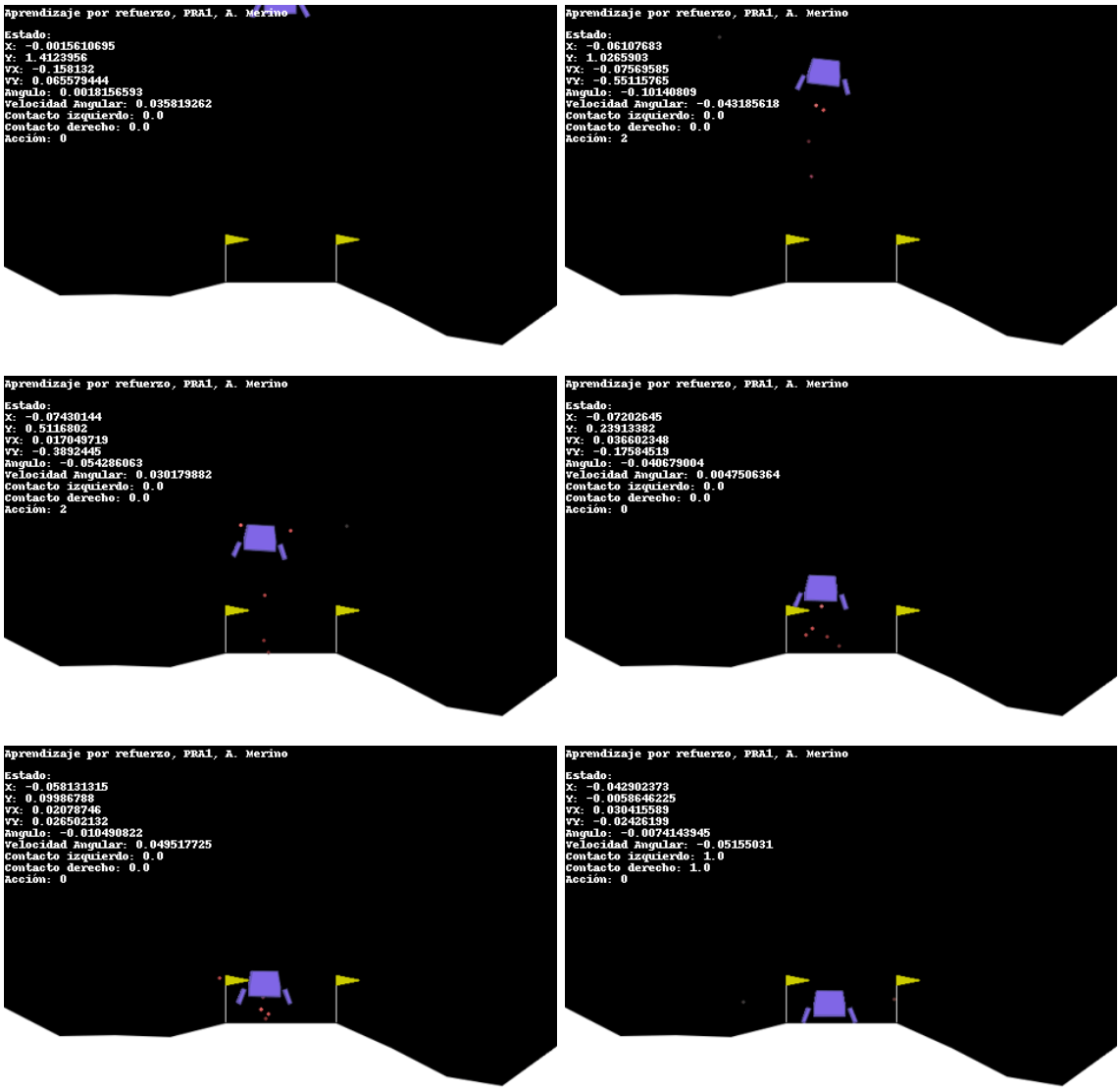

Figura 16: Ejecución con el agente A2C entrenado.

Podemos apreciar que, con la propuesta de agente (A2C), no se obtienen resultados significativamente mejores respecto al agente de referencia (DQN), dado que fue necesario más episodios y tiempo de entrenamiento para resolver el entorno, sin embargo, sí se obtiene una mejor estabilidad en el aprendizaje, sin tener pérdidas abruptas como las que se reflejaban en el entrenamiento del agente DQN.

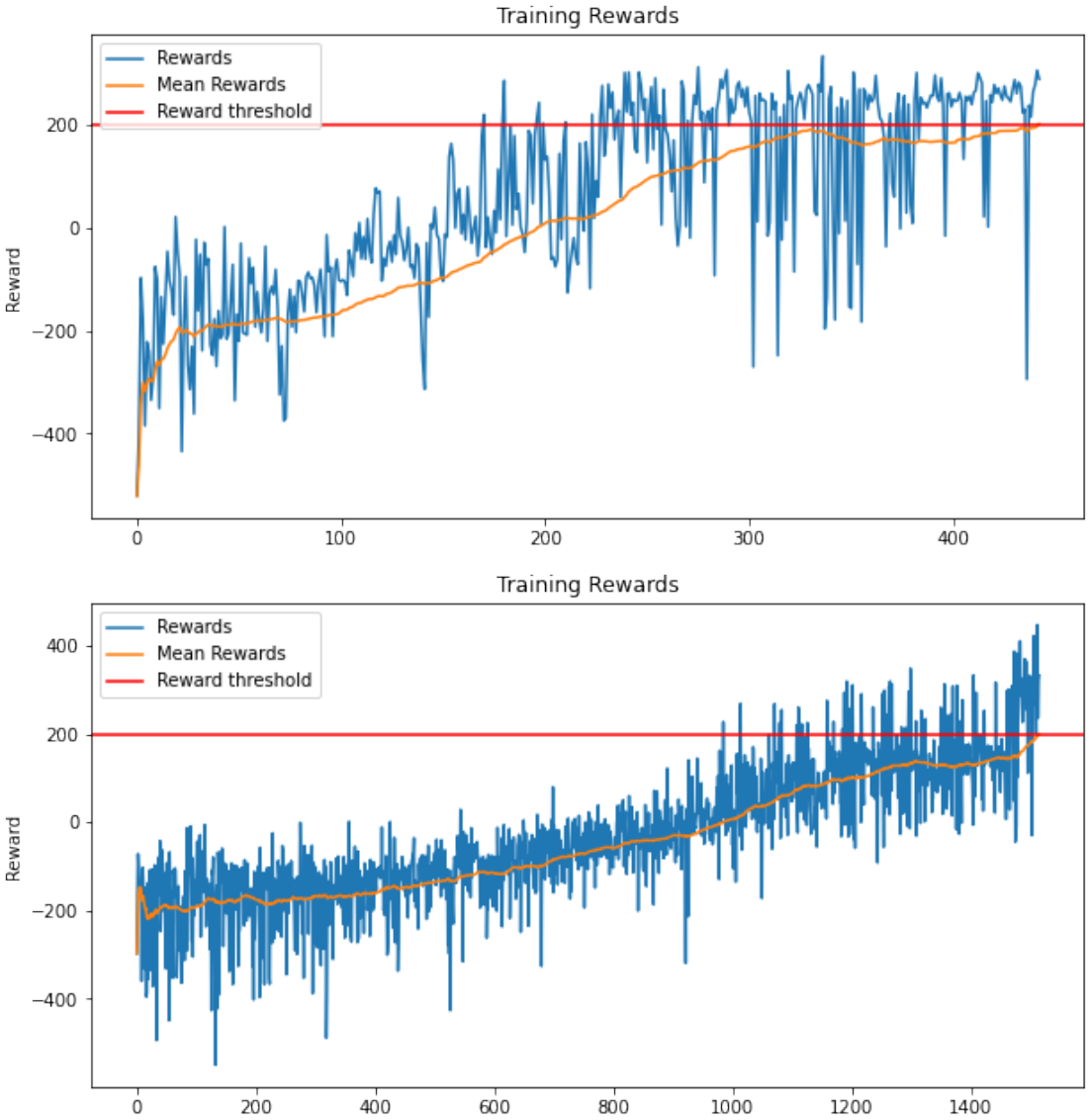

Figura 17: Evolución de la recompensa durante el entrenamiento, arriba DQN, abajo A2C.