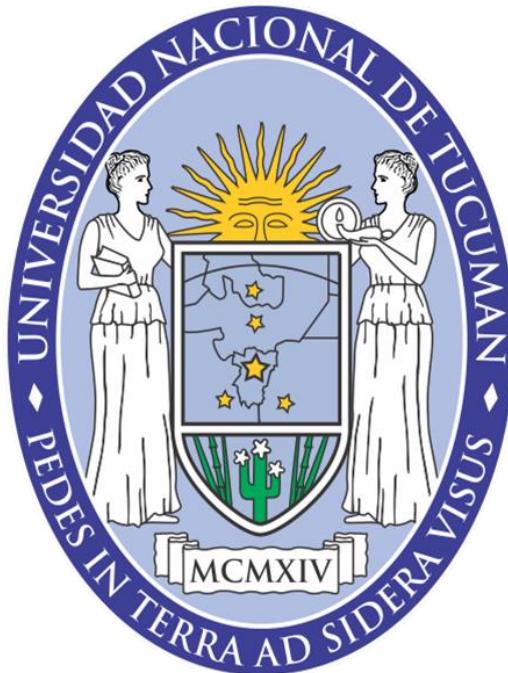


Facultad de Ciencias Exactas y Tecnología
Universidad Nacional de Tucumán



Carrera de Ingeniería Electrónica

Informe de Trabajo de Graduación

**Optimización de Parámetros de Radar OTH mediante Aprendizaje
por Refuerzo Profundo**

Autor: Leonardo Leandro Díaz

Director: Dr. Ing. Zenón Saavedra

San Miguel de Tucumán, Diciembre 2024

Contenido

1	Introducción	1
2	Marco Teórico.....	2
2.1	Radar OTH por Onda de Cielo.....	2
2.2	Simulador OTHR	3
2.2.1	Funcionalidad del simulador	4
2.3	Aprendizaje por Refuerzo Profundo.....	4
2.3.1	Radar Cognitivo	4
2.3.2	Aprendizaje por Refuerzo	5
2.3.3	Componentes del Aprendizaje por Refuerzo.....	5
2.3.4	Política y Función Valor	6
2.3.5	Integración de Redes Neuronales en DRL.....	8
3	Metodología	10
3.1	Modelo particular	10
3.2	Modelo general	11
3.3	Agente	13
3.3.1	Acciones	15
3.3.2	Red Neuronal.....	15
3.4	Ambiente o Entorno	18
3.4.1	Clasificación de Objetivos	18
3.4.2	Definición de Estados	19
3.4.3	Definición de Recompensa	20
3.5	Protocolo de evaluación del modelo	22
4	Resultados.....	24
4.1	Primer ejemplo.....	24
4.2	Segundo ejemplo.....	29
4.3	Tercer ejemplo.....	34
4.4	Ejemplo modelo general	39
4.5	Ánalisis de resultados	44
5	Conclusión	46
6	Bibliografía	47
7	Anexos.....	49

1 Introducción

En la era actual, caracterizada por avances rápidos en tecnología de detección y procesamiento de señales, los sistemas de radar desempeñan un papel importante en numerosas aplicaciones, desde la vigilancia aérea y marítima hasta la meteorología y la gestión del tráfico [1]. Su desempeño eficaz depende no solo de su precisión para detectar y rastrear objetivos, sino también de su capacidad para adaptarse a entornos operativos complejos y dinámicos [2]. En este contexto, las técnicas avanzadas de aprendizaje automático, en particular el aprendizaje por refuerzo profundo, emergen como una alternativa prometedora para potenciar la adaptabilidad y la eficacia de estos sistemas, permitiendo optimizaciones que van más allá de los métodos tradicionales.

El aprendizaje por refuerzo (RL) se basa en el concepto de agentes que aprenden a tomar decisiones óptimas a través de la interacción continua con su entorno. Integrar el aprendizaje por refuerzo profundo con la simulación de radar permite desarrollar modelos que pueden aprender dinámicamente y ajustar sus estrategias de detección en tiempo real.

El objetivo de la tesis es obtener un modelo/herramienta de software que seleccione de forma automática un conjunto de parámetros del radar que, en este caso, optimicen la calidad y cantidad de detecciones cercanas al objetivo. Para realizar dicha tarea se utilizó un entorno de simulación de radar que representa de manera efectiva las interacciones entre las señales de radar y diversos tipos de objetivos bajo diferentes condiciones. El simulador radar fue facilitado por el Laboratorio de Telecomunicaciones de la Facet - UNT. Por otro lado, se implementa y entrena un agente de RL que interactúe con el entorno del simulador para optimizar los parámetros operativos del radar, tales como la potencia de transmisión y la frecuencia de repetición de pulsos, en función de las respuestas obtenidas en exploraciones anteriores.

Para llevar a cabo este proyecto, se utiliza Python, un lenguaje de programación versátil y especialmente valorado por su eficacia en el procesamiento de datos y la implementación de algoritmos de aprendizaje automático.

2 Marco Teórico

En este capítulo, se describe el funcionamiento de un Radar Sobre Horizonte por Onda de Cielo, sus principales características técnicas y su proceso de detección, el cual comienza con la emisión de ondas que, al reflejarse en la ionosfera, iluminan un área específica donde se identifican los objetivos presentes. Se explican los principios básicos del radar OTHR, incluyendo su frecuencia portadora, potencia, rango y configuración de antenas, así como el proceso de reflexión y captación de señales. También se presenta el simulador OTHR, que permite configurar parámetros clave como la señal transmitida, las características de los objetivos, y las condiciones del entorno. Además, se introduce el Aprendizaje por Refuerzo Profundo (DRL), con sus componentes principales (agente, entorno, recompensa y estado), el uso de la política Epsilon-Greedy, y la aplicación de la Ecuación de Bellman para optimizar las decisiones del agente. Finalmente, se describe la integración de redes neuronales en el DRL, destacando su capacidad para manejar datos complejos y estimar valores de acción-estado.

2.1 Radar OTH por Onda de Cielo

Un Radar Sobre Horizonte (OTHR por sus siglas en inglés) por Onda de Cielo es un tipo de radar que trabaja en la banda de frecuencia de HF y aprovecha la refracción de ondas electromagnéticas en la ionosfera para detectar objetivos ubicados a cientos o incluso miles de kilómetros de distancia (figura 1) [3].

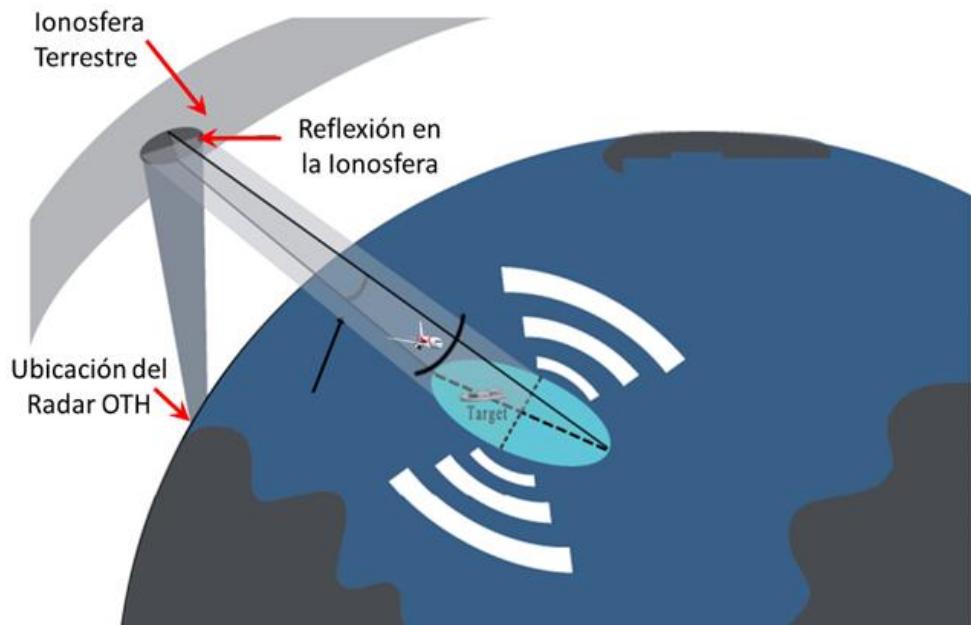


Figura 1: Radar Sobre Horizonte por Onda de Cielo

El proceso de búsqueda de objetivos de un sistema OTHR inicia con la determinación del área en donde se realizará la búsqueda, posteriormente el arreglo de antenas de transmisión se dirige hacia la ionosfera terrestre y se emite una onda electromagnética con un cierto valor de frecuencia. La onda electromagnética se propaga por el medio hasta ingresar en la ionosfera

en donde se producirá el efecto de reflexión de esta, provocando que la onda retorne hacia la tierra/mar. La energía de la onda ocupará una determinada región del espacio en función del ancho de haz provisto por el arreglo de antenas. Al alcanzar la superficie del mar/tierra, la onda iluminara una determinada región denominada región iluminada por el radar (DIR Dwell Illumination Region). Todos los objetos presentes dentro de esta región DIR son alcanzados por energía electromagnética y como consecuencia de esto, se produce un fenómeno de dispersión en estos objetos en donde parte de la energía electromagnética se propaga de regreso al radar y será captada por el arreglo de antenas de recepción (figura 2). Luego de la recepción en antena, la señal pasa por un sistema de recepción el cual adecua la señal para su posterior procesamiento digital. Finalmente, gracias al procesamiento de señales y de datos se logra extraer desde las señales recibidas información de los objetivos presentes en la región DIR. La mayoría de los sistemas de radar OTH obtienen información de posición geográfica (latitud, longitud y en algunos casos altitud) junto a la velocidad radial de los objetivos detectados dentro del área de búsqueda.

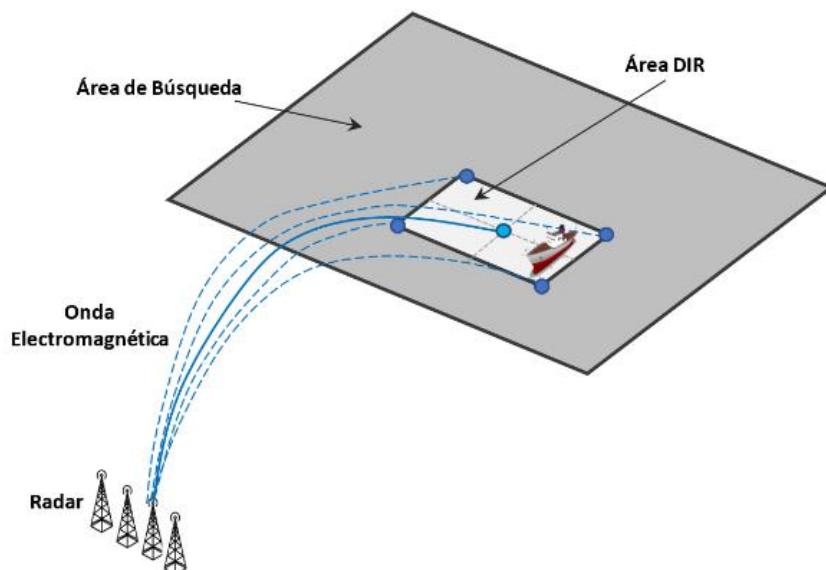


Figura 2: Esquema representativo del proceso de búsqueda de un radar OTH por onda de cielo.

Las principales características de estos sistemas de radar OTHR son:

- Frecuencia de Portadora: 3 – 30 MHz.
- Potencia del sistema: 55 – 60 dBW.
- Profundidad de Rango del área de búsqueda: 500 – 4000 km.
- Rango cruzado del área de búsqueda: 1000 – 18000 km.

2.2 Simulador OTHR

El simulador permite recrear el funcionamiento de un radar OTH operativo y con esto observar el desempeño de este en distintos escenarios de búsqueda, configurando elementos clave como los arreglos de antenas, señal a transmitir, características de los objetivos, y condiciones del entorno. Esto facilita analizar su efectividad en situaciones variadas y realistas.

2.2.1 Funcionalidad del simulador

La funcionalidad principal del simulador es evaluar el comportamiento del sistema de radar frente a diferentes escenarios de búsquedas. La evaluación toma como principal criterio de decisión la presencia de detecciones positivas de objetivos en el escenario ficticio establecido [4].

El simulador OTHR por Onda de Cielo permite al usuario realizar las siguientes configuraciones (Figura 3) referidas al sistema de radar como así también al escenario de búsqueda:

- Parametrización de los arreglos de antenas Tx/Rx.
 - distribución espacial
 - cantidad de elementos
 - tipo de antena patrón.
- Parametrización de la señal a transmitir:
 - Nivel de potencia
 - Tipo de modulación
 - Frecuencia de portadora
 - Dirección de apuntamiento
- Parametrización de Objetivos:
 - Cantidad
 - Tipo y tamaño
 - Estado de movimiento de objetivos.
- Parametrización del medio:
 - Estado del mar
 - La ionosfera terrestre
 - Niveles de ruido
- Parametrización de ubicación geográfica:
 - Región en donde se realiza la búsqueda de objetivos.
 - Sistema de Radar.

2.3 Aprendizaje por Refuerzo Profundo

El radar cognitivo es un sistema avanzado que utiliza inteligencia artificial para adaptarse a entornos cambiantes, ajustando automáticamente parámetros del radar para mejorar la detección. Basado en RL, el radar actúa como un agente que toma decisiones para maximizar su efectividad, aprendiendo de las recompensas obtenidas en diferentes condiciones. A continuación, se describen los diferentes tópicos involucrados.

2.3.1 Radar Cognitivo

Un radar cognitivo es un tipo de radar que incorpora métodos de inteligencia artificial y aprendizaje automático para mejorar su rendimiento y adaptabilidad en entornos cambiantes. Este tipo de radar analiza su entorno operativo, identificando no solo los objetivos, sino también las características del entorno que afectan su rendimiento, como el clima, los obstáculos terrestres y las señales interferentes. Posteriormente pasa a una etapa de aprendizaje y

adaptación donde el radar cognitivo aprende de la experiencia y adapta sus parámetros, como la potencia de transmisión, la banda de frecuencia, y los patrones de escaneo, para optimizar la detección y el seguimiento de objetivos bajo condiciones cambiantes. Finalmente toma decisiones de forma autónoma con base en su análisis y aprendizaje sobre cómo y cuándo cambiar sus estrategias operativas para mejorar la detección y minimizar las interferencias y el ruido [5].

2.3.2 Aprendizaje por Refuerzo

El aprendizaje por refuerzo (Reinforcement Learning, RL) es una técnica de aprendizaje automático en la que un agente aprende a tomar decisiones mediante la interacción con un entorno dinámico [6]. A diferencia de otros enfoques (Figura 3) de aprendizaje supervisado, donde el modelo aprende a partir de ejemplos etiquetados, el aprendizaje por refuerzo se basa en la experiencia del agente.



Figura 3: Tipos de aprendizaje en Inteligencia Artificial

2.3.3 Componentes del Aprendizaje por Refuerzo

A continuación, se describen los principales componentes en el Aprendizaje por Refuerzo [7]:

- El **agente** es una entidad que aprende a tomar decisiones inteligentes. Podemos decir que un agente es un aprendiz en el entorno de RL. En nuestro contexto, el transmisor del radar puede considerarse un agente ya que aprende a seleccionar acciones basadas en los estados o señales que recibe del entorno.
- El **entorno o ambiente** es el mundo del agente, en nuestro caso es el simulador del radar OTH. El entorno es fundamental en el aprendizaje por refuerzo, ya que proporciona los estados y las recompensas al agente basados en las acciones que éste realiza.
- La **recompensa** define el objetivo de un problema de RL. En cada paso discreto de tiempo, el entorno devuelve al agente un número real llamado “recompensa”. El único objetivo del agente es maximizar la recompensa total que recibe a largo plazo. Esta define cuales son los eventos buenos y malos para el agente. En un sistema biológico, podríamos pensar que las recompensas son análogas a las experiencias de placer o dolor. Son las características inmediatas y definitorias del problema que enfrenta el agente.

- El **estado** representa una descripción completa de la situación en la que se encuentra el agente en un momento dado. Es una captura instantánea de todos los factores relevantes que necesitan ser considerados para tomar una decisión sobre qué acción realizar a continuación.

Cada uno de los componentes serán descriptos en detalle en los próximos capítulos.

El ciclo de interacción en un sistema de aprendizaje por refuerzo, que consta de dos componentes principales: Agente y Entorno se visualiza en la Figura 4. La interacción se realiza de la siguiente manera:

- Estado S_t : en un instante de tiempo t , el agente recibe el estado actual del entorno S_t , que representa toda la información relevante del sistema en ese momento.
- Acción A_t : con base en el estado S_t , el agente selecciona una acción A_t . Esta acción es la decisión del agente para tratar de maximizar la recompensa a largo plazo.
- Entorno: cuando el entorno recibe la acción A_t del agente, este reacciona modificando su estado y proporcionando una recompensa R_t como respuesta a la acción tomada por el agente. La recompensa representa una señal de retroalimentación para el agente sobre la "calidad" de la acción elegida.
- Recompensa R_t : es un valor que evalúa el efecto de la acción A_t sobre el entorno. Una recompensa positiva indica que la acción fue favorable para alcanzar el objetivo, mientras que una recompensa negativa indica lo contrario.
- Nuevo Estado S_{t+1} : el entorno también proporciona el nuevo estado S_{t+1} que refleja los cambios en el entorno después de la acción. Este nuevo estado será la base para la siguiente acción en el ciclo.

Este proceso se repite en cada paso de tiempo t , formando un ciclo de retroalimentación continua entre el agente y el entorno. El agente aprende a través de este ciclo de prueba y error, ajustando sus decisiones para maximizar las recompensas acumuladas a lo largo del tiempo.

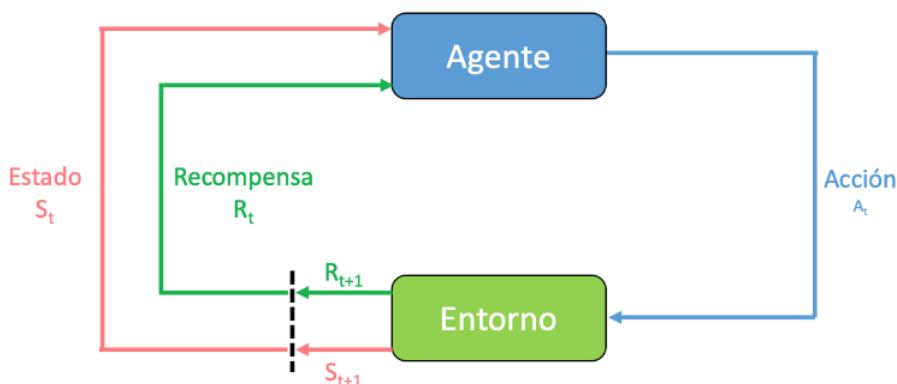


Figura 4: Diagrama de Aprendizaje por Refuerzo

2.3.4 Política y Función Valor

Una política es una estrategia que el agente utiliza para decidir qué acción tomar dado un estado particular. La política determina el comportamiento del agente al asignar a cada estado una

acción, de manera que maximice la recompensa acumulada esperada a lo largo del tiempo. En nuestro caso, aplicaremos la política de Epsilon-Greedy (Figura 5), que es un método simple para equilibrar la exploración y la explotación, eligiendo entre ambas opciones al azar.

Explorar significa que el agente toma acciones que no conoce bien, probando nuevas opciones para descubrir si pueden resultar mejores en el futuro. Este enfoque ayuda al agente a aprender más sobre el entorno y potencialmente encontrar mejores estrategias. Por otro lado, Explotar, significa que el agente elige la mejor acción conocida hasta el momento, aquella que se espera que le otorgue la mayor recompensa inmediata en base a lo que ha aprendido.

El parámetro ϵ (Epsilon) refiere a la probabilidad de elegir explorar con una acción nueva en lugar de explotar la mejor acción conocida. En general, el agente explota la mayor parte del tiempo, pero existe una pequeña posibilidad de que explore para seguir aprendiendo sobre otras posibles estrategias [8].

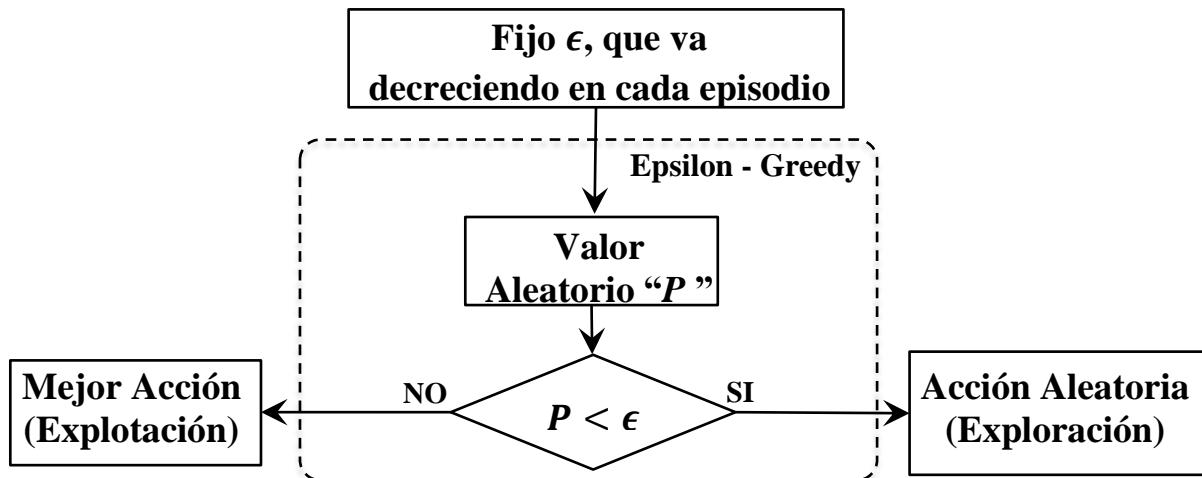


Figura 5: Lógica política Epsilon-Greedy

Por otro lado, mientras que la señal de recompensa indica lo que es bueno en un sentido inmediato, la **Función de Valor** específica lo que es bueno a largo plazo. En términos generales, la función de valor de un estado es la cantidad total de recompensa que un agente podría esperar acumular en el futuro, comenzando en ese estado. Para hacer una analogía con el ser humano, las recompensas son algo como el placer (si es alto) y dolor (si es bajo), mientras que las funciones de valor corresponden a un juicio más refinado y con visión de futuro de lo contentos o disgustados que estamos de que nuestro entorno se encuentre en un estado en particular. Las recompensas son, en cierto sentido, primarias; mientras que las funciones de valor, como predicciones de recompensas, son secundarias. Sin embargo, son las funciones de valor las que más nos interesan para la evaluación y toma de decisiones [8]. Desafortunadamente, es mucho más difícil determinar las funciones de valor que determinar las recompensas. Las recompensas son básicamente dadas directamente por el entorno, pero las funciones de valor deben estimarse y reestimarse a partir de las secuencias de observaciones que hace un agente durante todo un largo intervalo. De hecho, es el componente más importante

de casi todos los algoritmos de aprendizaje por refuerzo donde necesitamos estimar valores de manera eficiente.

La estimación de las funciones de valor es un proceso complejo que requiere actualizar y reajustar los valores de forma iterativa, aprovechando las observaciones del agente durante sus interacciones prolongadas con el entorno. A diferencia de las recompensas, que son observaciones directas, las funciones de valor deben ser calculadas y refinadas a lo largo del tiempo.

Por otro lado, un análisis ligeramente diferente y útil se realiza al utilizar la **Función Acción-Valor Q(S,A)**. Este valor representa la utilidad o calidad de tomar una acción en un estado determinado, teniendo en cuenta tanto la recompensa inmediata como el potencial de futuras recompensas acumuladas [9].

Un ajuste iterativo de los valores Q se realiza aplicando la Ecuación de Bellman, Ec. 2.1, que permite actualizar los valores de $Q(S_t, A_t)$ para cada acción A_t en un estado S_t , utilizando información sobre las recompensas inmediatas y el valor futuro esperado desde el siguiente estado.

$$\hat{Q}(S_t, A_t) = Q(S_t, A_t) + \alpha[R + (\lambda \cdot \max[Q(S_{t+1}, A_{t+1})]) - Q(S_t, A_t)] \quad \text{Ec 2.1}$$

- $\hat{Q}(S_t, A_t)$: Nuevo valor estimado después de considerar la recompensa obtenida y el valor óptimo esperado del siguiente estado.
- $Q(S_t, A_t)$: Es el valor actual de la función de acción-valor para el estado S_t y la acción A_t .
- α : Es la razón de aprendizaje, que determina cuánto se actualiza Q en cada paso.
- R : Es la recompensa inmediata obtenida al realizar la acción A_t en el estado S_t y llegar al estado siguiente S_{t+1} .
- λ : Es la tasa de descuento, que equilibra la importancia de las recompensas inmediatas respecto a las futuras.
- $\max [Q(S_{t+1}, A_{t+1})]$: Es el valor máximo de Q para el próximo estado S_{t+1} , considerando todas las posibles acciones A_{t+1} . Esto refleja el mejor valor de acción futuro que se puede obtener desde el estado siguiente.

Finalmente, para obtener la mejor acción A_t a partir de los valores de $Q(S_t, A_t)$, lo que se hace es elegir la acción con el valor de $Q(S_t, A_t)$ más alto en el estado S_t actual. Este valor representa la estimación de la recompensa futura que se espera al tomar esa acción en ese estado, de acuerdo con la política actual del agente.

2.3.5 Integración de Redes Neuronales en DRL

El aprendizaje por refuerzo profundo (Deep Reinforcement Learning, DRL) es una subdisciplina del aprendizaje automático que combina los principios del aprendizaje por refuerzo con el poder de representación de las redes neuronales profundas. Esta combinación permite al DRL abordar problemas complejos y de alta dimensionalidad que son intratables o inefficientes de resolver utilizando las técnicas tradicionales de RL, ampliando así su aplicabilidad a dominios más desafiantes.

En el contexto de DRL, la red neuronal funciona como un aproximador de funciones que estima el valor de $Q(S_t, A_t)$, donde cada acción A_t tiene un valor específico en un estado S_t . Una de las principales desventajas del aprendizaje Q es que se vuelve inviable cuando se trabaja con espacios de estados grandes, ya que el tamaño de la tabla Q crece exponencialmente con el número de estados y acciones. Para abordar este desafío, un enfoque alternativo es combinar el aprendizaje Q con redes neuronales profundas. [10]. Para mejorar la precisión de las predicciones, la red neuronal en DRL suele estructurarse en varias capas ocultas, cada una de las cuales se compone de múltiples neuronas. En cada capa, se aplican funciones de activación que permiten a la red capturar relaciones no lineales complejas, lo cual es fundamental para interpretar estados altamente variables y ricos en información, lo cual hace más eficiente al momento de predecir en vez de usar una tabla (Figura 6).

Las redes neuronales, debido a su capacidad de manejar características de alta dimensión y aprender representaciones complejas, son ideales para interpretar el estado del entorno, que en casos como el radar OTH puede ser altamente dinámico y rico en información sensorial. El entrenamiento de la red neuronal en el contexto de DRL se realiza mediante la actualización de sus pesos para minimizar la diferencia entre las predicciones de Q y los valores objetivo-calculados según la Ecuación de Bellman.

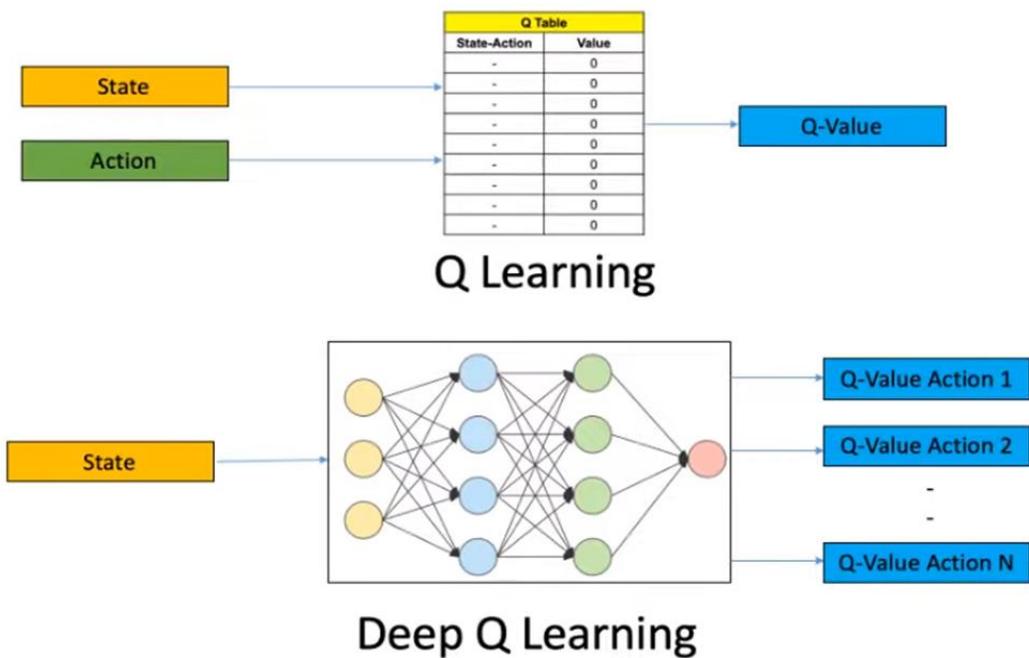


Figura 6: Diferencia entre Q Learning y Deep Q Learning

3 Metodología

En este capítulo se presenta un modelo de Aprendizaje por Refuerzo Profundo capaz de aprender y tomar decisiones en un entorno simulado de radar de manera efectiva y optimizada. Como primer paso se describen dos modelos, el Particular y General. El primero de ellos es un modelo entrenado para un escenario de búsqueda en particular mientras que el segundo es un modelo que es entrenado unificando todos los escenarios particulares definidos.

Posteriormente se describen en detalle los componentes presentes en un modelo DRL, que serán los utilizados para obtener los modelos DRL particulares y generales.

3.1 Modelo particular

Se definieron 13 ejemplos (escenarios), donde cada uno de ellos define un modelo particular que fue entrenado con la misma arquitectura de red neuronal. En cada ejemplo se modifican los parámetros de configuración del simulador OTH, los cuales pueden ser; de la señal a transmitir, parámetros de escenarios de búsquedas y del estado de movimiento de los objetivos.

Para cada ejemplo se eligió una secuencia de 500 episodios, donde cada episodio consta de 10 escaneos. Los escaneos muestran cómo se mueve cada objetivo en un determinado intervalo de tiempo. Entre escaneo se modifica el estado y la recompensa del agente, acorde a las acciones que se elija, por lo que el simulador se ejecutará entre cada escaneo y usará la ecuación de Bellman (que es nuestra función de valor) para entrenar la red neuronal.

Cada ejemplo guardará la información del entrenamiento en conjuntos de transiciones, donde cada una contiene la siguiente información: estado actual, estado siguiente, recompensa, acción y una variable booleana que indicará si con esa transición finaliza el episodio (útil ya que el modelo sabrá cómo aplicar la ecuación de Bellman en esa instancia).

Por otro lado, cada ejemplo está conformado de 500 episodios y 9 transiciones por episodio dando un total de 4500 transiciones en total.

Al finalizar la etapa de entrenamiento, se obtendrá un modelo DRL por cada ejemplo, los cuales tendrán la capacidad de encontrar los parámetros óptimos de la señal a transmitir para tener una mejora en la detección de dos objetivos en cada uno de los escenarios presentes en los 13 ejemplos.

Los ejemplos fueron configurados de la siguiente forma en las Tabla.1 y Tabla.2.

	Ejemplo 1	Ejemplo 2	Ejemplo 3	Ejemplo 4	Ejemplo 5	Ejemplo 6	Ejemplo 7
Tipo de Modulación	LFM	LFM	LFM	BPSK	BPSK	BPSK	BPSK
AB [Hz]	12000	10000	9500	10000	9000	10000	9500
PRF [Hz]	24	30	30	26	23	25	25
Potencia [dBW]	42	44	48	42	40	42	42
T [s]	0.001	0.002	0.0015	0.0011	0.001	0.013	0.001368
N	250	250	250	250	280	200	200
Fs [Hz]	5000	3000	7000	30000	35000	27000	25000
Fc [MHz]	14.4407	13.5254	14.4407	13.5254	14.4407	14.4407	13.5254
Tipo de Código	Barker11*	Barker11*	Barker 11*	Barker 11	Barker 11	Barker 13	Barker 13

Tabla.1: Ejemplos de entrenamiento de 1 a 7

	Ejemplo 8	Ejemplo 9	Ejemplo 10	Ejemplo 11	Ejemplo 12	Ejemplo 13
Tipo de Modulación	BPSK	BPSK	BPSK	BPSK	BPSK	BPSK
AB [Hz]	10000	11000	9400	11000	9500	9000
PRF [Hz]	40	28	30	35	23	29
Potencia [dBW]	40	42	42	41	42	42
T [s]	0.0007	0.0006	0.0017	0.001455	0.0025	0.0027
N	250	250	270	250	250	230
Fs [Hz]	30000	28000	24000	25000	23000	22000
Fc [MHz]	13.525	14.441	13.5254	14.4407	13.5254	14.4407
Tipo de Código	Barker 7	Barker 7	Complem. 1	Complem. 1	Complem. 2	Complem. 2

Tabla.2: Ejemplos de entrenamiento de 8 a 13

* Tener en cuenta que la modulación LFM no tiene en cuenta el tipo de código

El cálculo detallado de la memoria RAM utilizada durante el entrenamiento de un ejemplo puede encontrarse en el [Anexo A](#).

3.2 Modelo general

En el modelo general se combinan los 13 ejemplos particulares, en busca de lograr una generalización del modelo DRL. El modelo fue entrenado utilizando todo el conjunto de transiciones obtenido a partir de 13 ejemplos distintos. El proceso de entrenamiento se observa en la Figura 7. Cada uno de los 13 ejemplos representa un escenario de entrenamiento con características y patrones únicos, diseñados para proporcionar al modelo una diversidad de experiencias. Para maximizar la capacidad de generalización del modelo y evitar el sobreajuste a un único patrón, las transiciones de todos los ejemplos fueron extraídas y mezcladas aleatoriamente.

Este enfoque aleatorio implica que en cada lote de entrenamiento del modelo recibe transiciones de diferentes ejemplos, lo cual le permite aprender de manera conjunta múltiples

situaciones. Al exponer el modelo a una mezcla de datos diversos en cada etapa del entrenamiento, se busca que el modelo identifique patrones generales y constantes a lo largo de todos los ejemplos, en lugar de adaptarse específicamente a las características de un solo ejemplo.

El tamaño del lote tendrá influencia en la estabilidad, velocidad y capacidad de generalización del modelo. Aunque los tamaños grandes aceleraron el entrenamiento y aprovecharon mejor los recursos de la GPU, los tamaños intermedios (128 y 256) ofrecieron un buen balance entre precisión y eficiencia, sin comprometer la capacidad de generalización del modelo de manera significativa.

El número de lotes se determina por la Ec. 3.1, el cual indica cuántas veces deberá actualizarse los pesos de la red neuronal en el entrenamiento, el resto del cociente entero se lo entrenará al final como un lote más chico. Se adoptó una cantidad de 128 transiciones por lote.

$$n^{\circ} \text{ lotes} = \frac{4500 \frac{\text{transiciones}}{\text{ejemplos}} * 13 \text{ ejemplos}}{128} = 457,031 \quad \text{Ec 3.1}$$

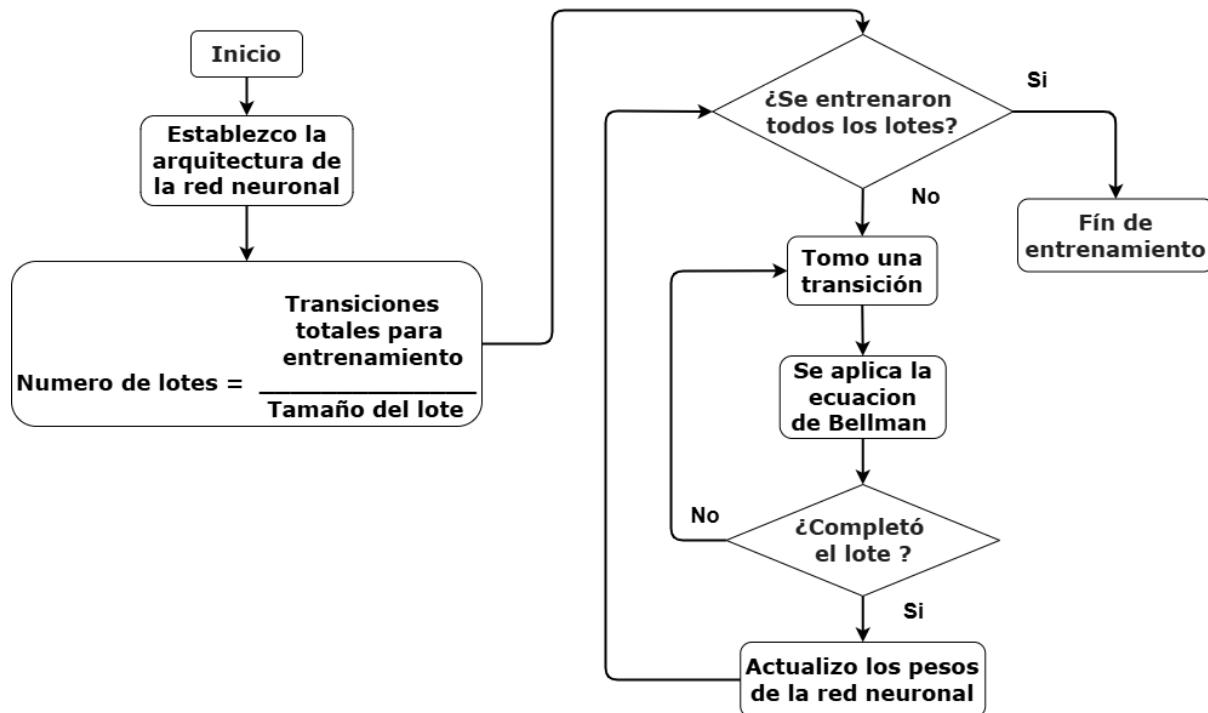


Figura 7: Secuencia de entrenamiento de modelo general

3.3 Agente

El diagrama en la Figura 8 muestra el proceso que sigue el agente para aprender y tomar decisiones en el entorno de simulación, usando una combinación de la política de Epsilon-Greedy, la ecuación de Bellman, y una red neuronal.

Todo comienza con una configuración inicial, que establece los parámetros básicos para un episodio específico. Esta configuración es el punto de partida para el agente, y desde ahí se realiza una secuencia de simulación. Durante el primer escaneo, el agente usa los parámetros establecidos inicialmente. En el segundo escaneo, sin embargo, estos parámetros son modificados en función de la acción seleccionada por la política de Epsilon-Greedy.

La política de Epsilon-Greedy juega un papel fundamental en la selección de acciones del agente. Al inicio del entrenamiento, el valor de Epsilon es alto (por ejemplo, 1), lo que hace que el agente elija acciones al azar para explorar el entorno. Con el avance del entrenamiento, el valor de Epsilon se reduce progresivamente multiplicándose por un factor (0.998 en este caso), de modo que el agente empieza a depender más de su experiencia acumulada (en la Figura 9 se observa como decrece el valor de épsilon a medida que avanzan los episodios). Esto significa que, en lugar de elegir acciones aleatorias, comenzará a elegir aquellas acciones que la red neuronal identifica como más beneficiosas, en función de los valores Q estimados.

Una vez seleccionada la acción, el agente usa la ecuación de Bellman para calcular el valor Q de dicha acción. La ecuación de Bellman combina la recompensa inmediata obtenida por esa acción con el valor Q estimado del siguiente estado, ajustado por un factor de descuento (0.95), lo que permite estimar el beneficio esperado a largo plazo. Con esta información, el agente entrena su red neuronal, que aprende a predecir los valores Q ajustando sus pesos para minimizar la diferencia entre el valor Q calculado y el valor Q predicho. La tasa de aprendizaje (0.2) controla el grado de ajuste en cada iteración, evitando cambios bruscos y promoviendo un aprendizaje más estable.

Con cada paso, el agente actualiza los pesos de la red que utiliza para aproximar los valores Q asociados a los estados observados. Durante este proceso, el estado siguiente se convierte en el estado actual para el próximo escaneo, permitiendo que el agente repita iterativamente la selección de acciones y la actualización de los pesos de la red neuronal. Este ciclo continúa hasta completar todos los escaneos de la secuencia, conformando un episodio. En el último escaneo, donde ya no existe un estado siguiente, el valor Q se simplifica y se calcula exclusivamente a partir de la recompensa inmediata, considerando la tasa de aprendizaje para ajustar los pesos correspondientes.

Al final de cada episodio, el agente reinicia sus parámetros para dar inicio a un nuevo ciclo de aprendizaje. Durante este nuevo episodio, el agente reitera la secuencia de aprendizaje, pero con una política de selección de acciones ajustada, dado que la estrategia Epsilon-Greedy reduce progresivamente la aleatoriedad en favor de una mayor explotación de acciones previamente exitosas (ver Figura 9). A medida que acumula experiencia, el agente mejora su

capacidad de seleccionar acciones que maximizan la recompensa acumulada, tendiendo hacia una solución óptima para el entorno específico en el que opera.

Este proceso de aprendizaje continuo, basado en la exploración estratégica, la actualización iterativa de los valores Q y la explotación de acciones previamente evaluadas, permite al agente optimizar sus decisiones y adaptarse dinámicamente a los cambios del entorno. Esto contribuye al desarrollo de un modelo más robusto y con mayor capacidad de generalización.

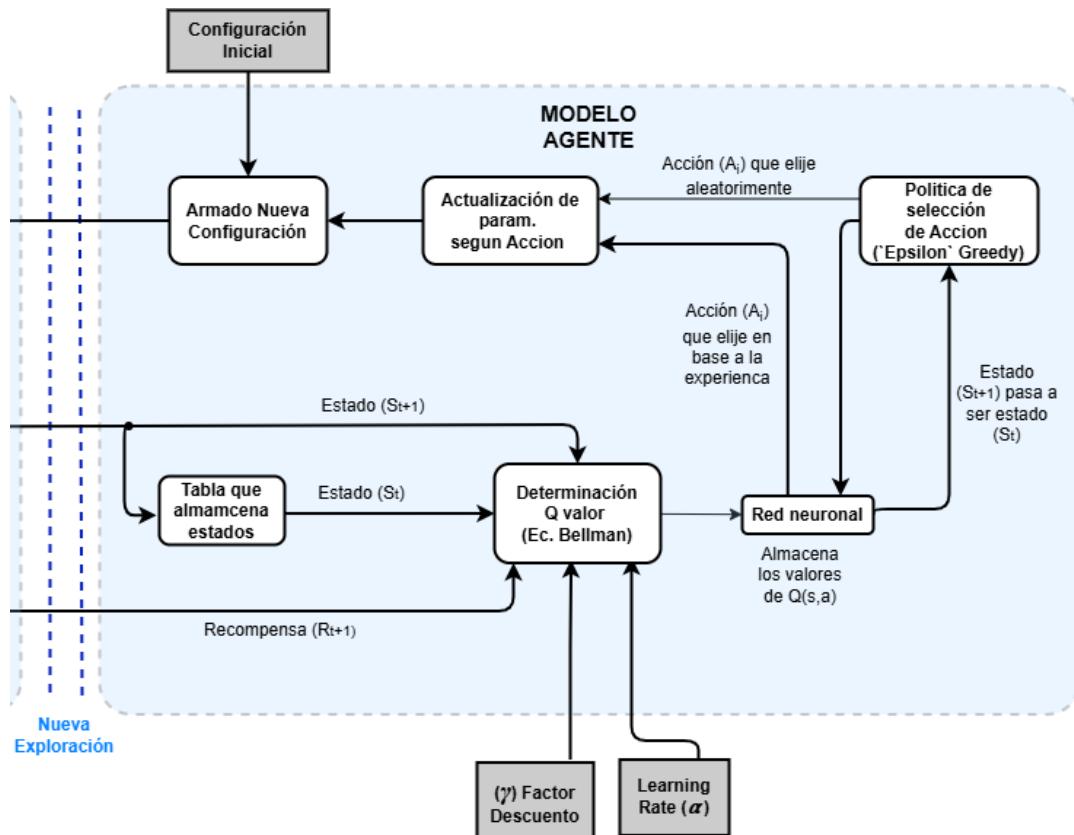


Figura 8: Esquema del modelo del agente, a la izquierda (imagen cortada) el esquema del modelo del entorno.

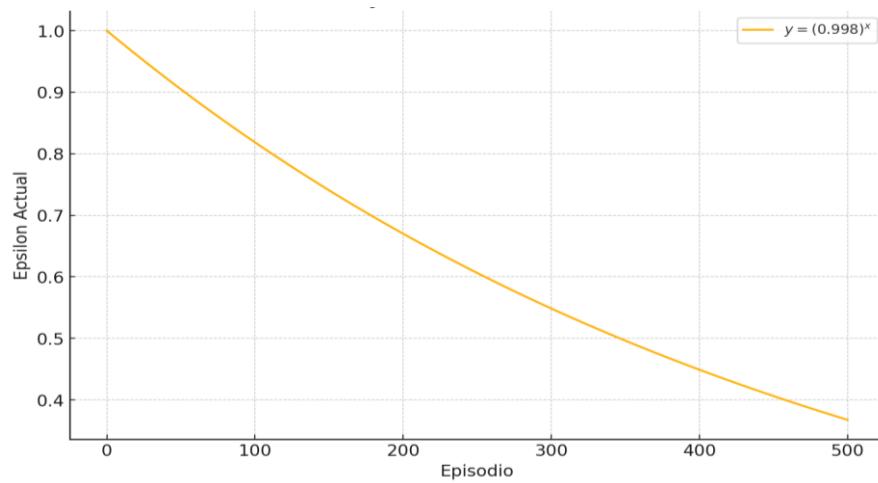


Figura 9: Variación de Epsilon en función de cada episodio

3.3.1 Acciones

Las acciones que puede tomar en cada estado el agente son discretas ya que es un espacio de acciones finito y claramente definidas. Estas acciones modifican parámetros del radar, involucrados en la forma de onda transmitida por este. Las acciones posibles de realizar por el agente son las siguientes:

- Acción 1: incrementar la PRF en 2Hz respecto a la última simulación.
- Acción 2: decrementar la PRF en 2Hz respecto la última simulación.
- Acción 3: incrementa la potencia del sistema transmisor en 1 dBW (1.26W) respecto la última simulación.
- Acción 4: decremente la potencia del sistema transmisor en 1 dBW (1.26W) respecto la última simulación.
- Acción 5: no modifica ningún parámetro

3.3.2 Red Neuronal

Al utilizar un enfoque de DRL, se necesita implementar una red neuronal para estimar los valores Q en cada estado. A continuación, se describe la arquitectura de la red neuronal utilizada.

En una red neuronal, el peso es un coeficiente que multiplica el valor de salida de una neurona antes de ser enviado a la siguiente neurona. Cada conexión entre neuronas tiene un peso asociado, que se ajusta durante el proceso de entrenamiento para minimizar la función de pérdida. El sesgo también es un valor adicional que permite que las neuronas ajusten su activación sin depender estrictamente de las entradas, esto otorga mayor flexibilidad a la red para modelar relaciones complejas en los datos [11].

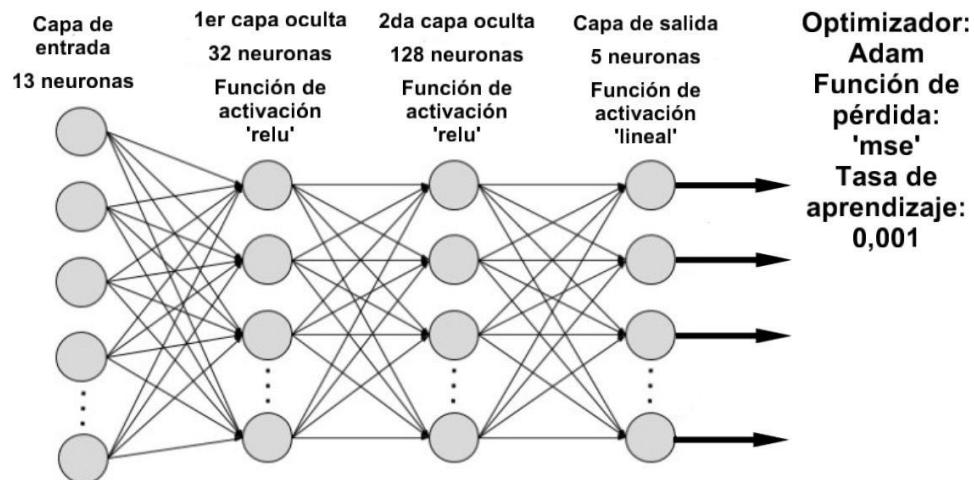


Figura 10: Arquitectura de red neuronal del modelo particular

Matemáticamente, si tenemos una neurona n_i con una entrada x_i , su salida y_j , está determinada por Ec. 3.2:

$$Y_j = f \left(\sum_{i=1}^n w_{ij} \cdot x_i + b_j \right) \quad \text{Ec 3.2}$$

Donde:

- w_{ij} es el peso que conecta la neurona i con la neurona j.
- x_i es la salida de la neurona i.
- b_j es el sesgo asociado a la neurona j.
- $f()$ es la función de activación aplicada al resultado.

Con respecto a la red neuronal que se utilizó (ver Figura 10) para entrenar cada ejemplo particular, es una arquitectura secuencial con capas densas y activaciones no lineales (ReLU) que permite que las capas ocultas (una capa con 32 y otra con 64 neuronas) aprendan relaciones complejas entre las características de entrada y las decisiones del agente para capturar patrones complejos en los datos, combinada con una capa de salida lineal que asegura que la salida no esté limitada y que el modelo pueda predecir con precisión los valores de Q.

Durante el entrenamiento, el modelo minimiza la función de pérdida error cuadrático medio (MSE por sus siglas en inglés), ajustando sus pesos para que el valor $Q(s,a)$ estimado se aproxime al valor objetivo calculado. Este proceso de ajuste se realiza mediante retropropagación, donde los gradientes de la pérdida con respecto a los pesos se calculan y utilizan para actualizar los pesos del modelo. El uso de MSE definido por la Ec. 3.3 ayuda a suavizar el proceso de aprendizaje, dado que minimiza las variaciones extremas en los valores de Q al enfocarse en reducir el error acumulado de las predicciones [13].

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad \text{Ec 3.3}$$

- y_i es el valor real o el valor objetivo
- \hat{y}_i es el valor predicho por el modelo para la muestra i

Por otro lado, también se utiliza el optimizador Adam (Adaptive Moment Estimation) que es un método basado en gradientes, ampliamente utilizado en redes neuronales profundas debido a su eficiencia y capacidad de ajuste dinámico de la tasa de aprendizaje para cada peso del modelo. Adam combina las ventajas de dos algoritmos populares:

- Momentum: Este enfoque mantiene una "velocidad" para cada parámetro, acumulando gradientes previos, lo que permite un ajuste más rápido y estable.
- RMSprop: Utiliza una tasa de aprendizaje adaptativa que considera el cuadrado de gradientes pasados para estabilizar la actualización de cada peso.

Adam calcula dos promedios móviles, el promedio móvil del gradiente, que actúa como el "momentum" y el promedio móvil del cuadrado del gradiente, que se usa para la normalización.

Además, Adam utiliza los gradientes del MSE para hacer ajustes en los pesos, ayudando al modelo a mejorar sus predicciones. La combinación de MSE y Adam permite un entrenamiento eficiente y preciso, ideal para redes neuronales profundas en tareas complejas.

Esta arquitectura anterior de la red está bien alineada con la toma de decisiones óptima en un sistema de radar, permitiendo al agente mejorar su rendimiento a medida que se entrena.

Por otro lado, luego de probar diferentes arquitecturas de red y diferentes tamaños de lotes para el modelo general, se logró armar la red neuronal de la Figura 11 que modela correctamente la cantidad de lotes explicado en la [sección 3.2](#). A diferencia del modelo particular, se utilizaron tres capas ocultas y más neuronas en cada capa, con el objetivo de mejorar la capacidad de aprendizaje y generalización.

Las capas ocultas utilizaron Dropout que es una técnica de regularización que desactiva aleatoriamente un porcentaje de neuronas durante el entrenamiento. Se utilizó una tasa del 10% para que pueda prevenir el sobreajuste y mejorar la generalización, permitiendo al modelo aprender características distribuidas y robustas frente a datos no vistos, lo que es crucial dado el volumen limitado de transiciones disponibles. También se usó regularización L2 con un coeficiente 0.001, lo que ayuda reducir la varianza en el modelo y también contribuye a mejorar la generalización penalizando los pesos altos en la red, evitando que el modelo se sobreajuste a los datos disponibles. Para garantizar la estabilidad del entrenamiento, se utilizó el optimizador Adam con un learning rate inicial de 0.0005 y un parámetro clipvalue en 1, que limita los gradientes a un máximo absoluto de ± 1 , evitando problemas de gradientes explosivos, es decir, que no haga actualizaciones muy grandes en los pesos.

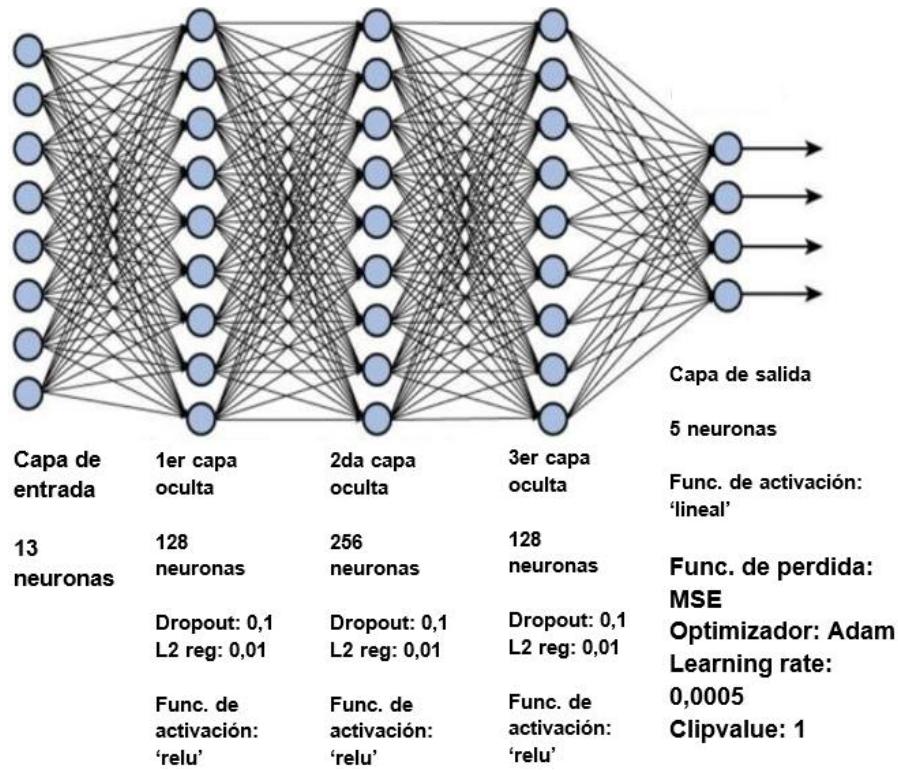


Figura 11: Arquitectura de red neuronal del modelo general

3.4 Ambiente o Entorno

El entorno representa la interacción entre el simulador OTHR y el agente, estructurando el flujo de información mediante las detecciones obtenidas y las configuraciones de parámetros seleccionadas. Este modelo realiza varias etapas clave: en primer lugar, determina las observaciones a partir de las detecciones (latitud, longitud, y velocidad) asignándolas al objetivo que corresponde y calcula otros parámetros adicionales como la tasa de detección TD, distancia promedio y la rapidez promedio para cada objetivo detectado. Estos parámetros conformarán el estado, los cuales se incorporan a los parámetros de la señal de transmisión.

Con cada acción seleccionada por el agente, se actualizan las configuraciones del radar para el siguiente paso. El modelo utilizará ciertos parámetros de la señal de transmisión que junto con la tasa de detección servirán para determinar una recompensa basada en la calidad de las detecciones al final de cada escaneo. Este ciclo iterativo permite que el agente optimice las configuraciones del radar, mejorando progresivamente la precisión de las detecciones. La Figura 12 ilustra la relación entre las etapas principales del modelo de entorno, destacando cómo las observaciones y estados son utilizados para determinar las recompensas y retroalimentar al agente.

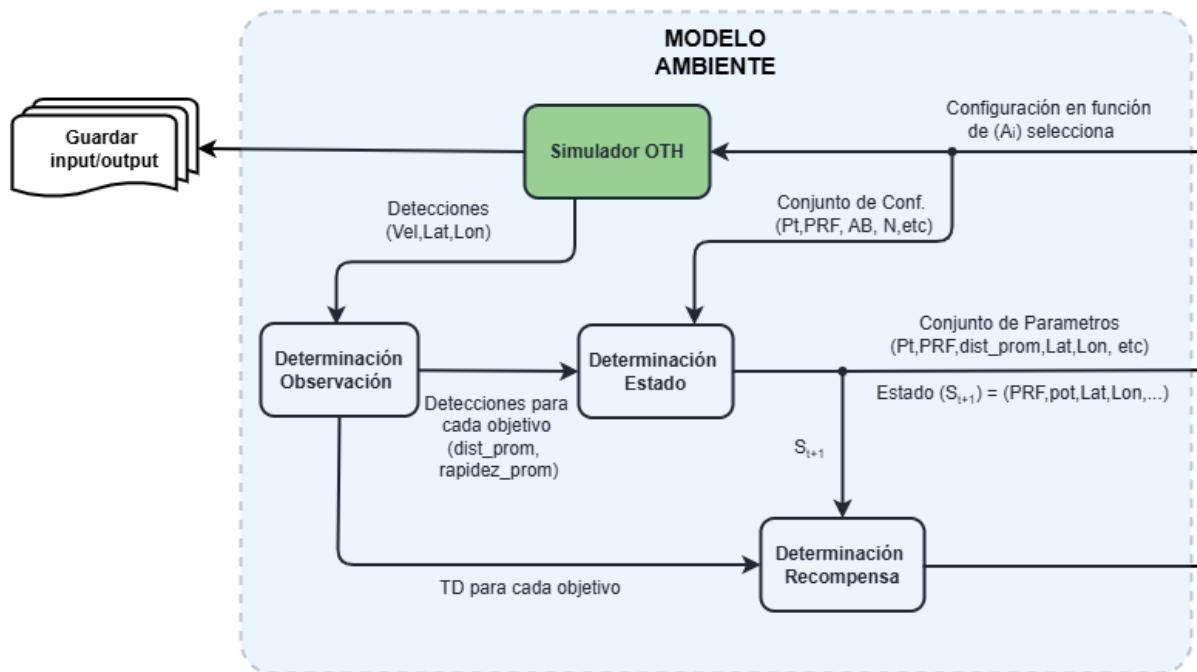


Figura 12: Esquema del modelo del entorno

3.4.1 Clasificación de Objetivos

Durante la simulación de cada escaneo, el simulador proporciona información clave sobre los objetivos reales, como su posición actual, velocidad, altitud y dirección de desplazamiento. Además, genera un conjunto de detecciones sin indicar específicamente a cuál objetivo real pertenecen, lo que introduce variabilidad debido a las características del medio y la configuración del detector CFAR.

Para asociar estas detecciones a cada objetivo, se utiliza la técnica de agrupamiento no supervisado K-means, la cual será aplicada siempre y cuando haya más de una detección porque se fijará dos agrupamientos dada la cantidad de objetivos. Es un modelo que analiza datos sin etiquetas o resultados predefinidos y divide un conjunto de observaciones en k grupos, asignando cada observación al grupo cuyo valor promedio es más cercano.

Dado un conjunto de observaciones $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$, donde cada observación es un vector real de d dimensiones, k-medias construye una partición de las observaciones en k conjuntos ($k \leq n$) a fin de minimizar la suma de los cuadrados dentro de cada grupo Ec. 3.4 (WCSS): $\mathbf{S} = \{S_1, S_2, \dots, S_k\}$ [12].

$$\arg \min_s \sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - \mu_i\|^2 \quad \text{Ec 3.4}$$

donde μ_i es la media de puntos en S_i .

Esta técnica permite diferenciar las detecciones correspondientes a cada objetivo, asignando las posiciones de los objetivos reales a centroides que representan el centro de cada agrupamiento y se calculan como la media de las detecciones en dicho grupo.

3.4.2 Definición de Estados

El estado se encuentra conformado por un conjunto de parámetros, los cuales brindan una descripción completa de la situación en la que se encuentra el agente en un momento dado. Es en base al estado actual que el agente debe tomar una decisión sobre qué acción realizar a continuación. Cada estado estará definido por un arreglo que contiene los siguientes 13 parámetros:

- Distancia promedio 1: es el promedio de las distancias entre cada detección positiva del primer objetivo y el centroide que modela dicho objetivo, considerando únicamente aquellas detecciones que se encuentran a menos de 10 km del centroide.
- Rapidez promedio 1: es el promedio de la rapidez de cada detección positiva del primer objetivo, que se encuentran a menos de 10 km del centroide.
- Distancia promedio 2: es el promedio de las distancias entre cada detección positiva del segundo objetivo y el centroide que modela dicho objetivo, considerando únicamente aquellas detecciones que se encuentran a menos de 10 km del centroide.
- Rapidez promedio 2: es el promedio de la rapidez de cada detección positiva del segundo objetivo, que se encuentran a menos de 10 km del centroide.
- Tipo de modulación: define el tipo de modulación aplicada sobre la onda portadora a transmitir por el radar. Se considera un radar del tipo Pulsado. Las modulaciones disponibles son Pulse-BPSK (Modulación con Corrimiento de Fase) o Pulse-LFM (Modulación Lineal en Frecuencia)

- Ancho de banda (AB): la porción del espectro en frecuencias, en la cual se concentra la mayor potencia/energía de la señal.
- Frecuencia de repetición de pulso (PRF): es la frecuencia con la cual se realizan los disparos con el transmisor dentro de un IIC (Intervalo de Integración Coherente), sobre un área DIR.
- Potencia: es el valor de potencia en la salida del sistema transmisor, antes de ingresar al sistema de antenas transmisoras.
- Ancho de pulso (T): es la duración temporal del pulso transmitido (Tiempo en ON).
- Cantidad de integraciones (N): cantidad de disparos/transmisiones que realiza el transmisor durante un IIC (Intervalo de Integración Coherente, sobre un área DIR).
- Frecuencia de muestreo (Fs): es la frecuencia de conversión utilizada con el CAD (Conversor Analógico Digital) en el receptor.
- Frecuencia de portadora (Fc): Es el valor de frecuencia que posee la onda electromagnética, emitida/recibida por el radar.
- Tipo de código: define el tipo de código a utilizar en la modulación BPSK, sobre la portadora. Los tipos de código de modulación disponibles son: Barker 7, Barker 11, Barker 12, complementario 1 (Golay), complementario 2 (Golay).

3.4.3 Definición de Recompensa

La recompensa define el objetivo de un problema de RL. En cada paso discreto de tiempo, el entorno devuelve al agente un número real llamado “recompensa”. El único objetivo del agente es maximizar la recompensa total que recibe a largo plazo. El objetivo es disminuir la dispersión de las detecciones, logrando detecciones más precisas mientras se obtienen valores de potencia y PRF razonables para este tipo de radar.

La recompensa obtenida luego de cada escaneo se define por la Ec. 3.5.

$$R(TD, \text{potencia}, PRF) = R_{TD} + R_{\text{potencia}} + R_{PRF} \quad \text{Ec 3.5}$$

Donde

- R_{TD} : Recompensa basada en la métrica de detección del objetivo
- R_{potencia} : Ajuste de la recompensa basado en la potencia
- R_{PRF} : Ajuste de la recompensa basado en el PRF

A continuación, se describe cada uno de los términos presentes en la fórmula de cálculo de la recompensa.

Para determinar el valor R_{TD} , se necesita como paso previo determinar el estado actual del agente. Dentro del estado se encuentran la distancia promedio Ec. 3.6 y rapidez promedio Ec. 3.7 para cada objetivo en base a las características de las detecciones obtenidas en función al centroide obtenido de la técnica de agrupamiento de cada objetivo.

$$\text{distancia promedio} = \frac{\sum_{i=1}^n d(P_i, C_i)}{n} \quad \text{Ec 3.6}$$

$$\text{rapidez promedio} = \frac{\sum_{i=1}^n \text{rapidez de cada detección a menos de 10km del centroide}}{n} \quad \text{Ec 3.7}$$

donde $d(P_i, C_i)$ es la distancia euclíadiana de cada detección al centroide, siempre que dicha detección se encuentre a una distancia de 10 km o menos del centroide. Y además n es la cantidad de detecciones a 10 km o menos del centroide. Se tomo la distancia de 10 km ya que, en base a pruebas del simulador, es una distancia que logrará apreciar el funcionamiento del agente.

Posteriormente se calcula una Tasa de Detección (TD) Ec. 3.8 para cada objetivo con la finalidad de dimensionar la ubicación de las detecciones con respecto al centroide y así poder obtener la recompensa.

$$TD = \frac{\text{cantidad de detecciones que están a 10 km del centroide}}{\text{total de detecciones de ese agrupamiento}} \quad \text{Ec 3.8}$$

Finalmente, la recompensa total se obtiene sumando la recompensa para cada objetivo. La recompensa para cada objetivo Ec. 3.9 se calculará de la siguiente manera:

$$R_{TD} \left\{ \begin{array}{ll} 0 & \text{si } TD = 0 \text{ o } TD = \infty \\ 1 & \text{si } 0.1 < TD \leq 0.3 \\ 2 & \text{si } 0.3 < TD \leq 0.6 \\ 3 & \text{si } 0.6 < TD \leq 0.8 \\ 4 & \text{si } 0.8 < TD \leq 1 \\ 0 & \text{en cualquier otro caso} \end{array} \right. \quad \text{Ec 3.9}$$

Por otro lado, para determinar $R_{potencia}$ (ver Ec. 3.10), se eligió el siguiente intervalo de análisis para premiar la potencia de transmisión ya que en esos valores se obtienen cantidad de detecciones aceptables y además el hecho de tener mayor potencia de transmisión genera un costo adicional en las características de un radar.

$$R_{Potencia} \left\{ \begin{array}{ll} 1.5 - \left(\frac{1.5 - 0.5}{60 - 40} \right) \cdot (potencia - 40) & \text{si } 40 \leq potencia \leq 60 \\ 0 & \text{en cualquier otro caso} \end{array} \right. \quad \text{Ec 3.10}$$

Finalmente, para el caso de la recompensa de la frecuencia de repetición de pulso (PRF) (ver Ec. 3.11) se necesita en caso de la detección de aviones valores más altos porque permite un seguimiento preciso de objetivos rápidos en un rango más corto y dinámico mientras que para barcos al desplazarse más lento no sería necesario un valor tan alto por lo que se eligió un intervalo que contemple ambos casos.

$$R_{PRF} \left\{ \begin{array}{ll} 1 & \text{si } 0 < PRF \leq 58 \\ 1 - \frac{PRF - 58}{2} & \text{si } PRF > 58 \\ 0 & \text{en cualquier otro caso} \end{array} \right. \quad \text{Ec 3.11}$$

Por último, en la Fig. 13 se encuentran las graficadas las funciones R_{TD} , $R_{potencia}$ y R_{PRF} .

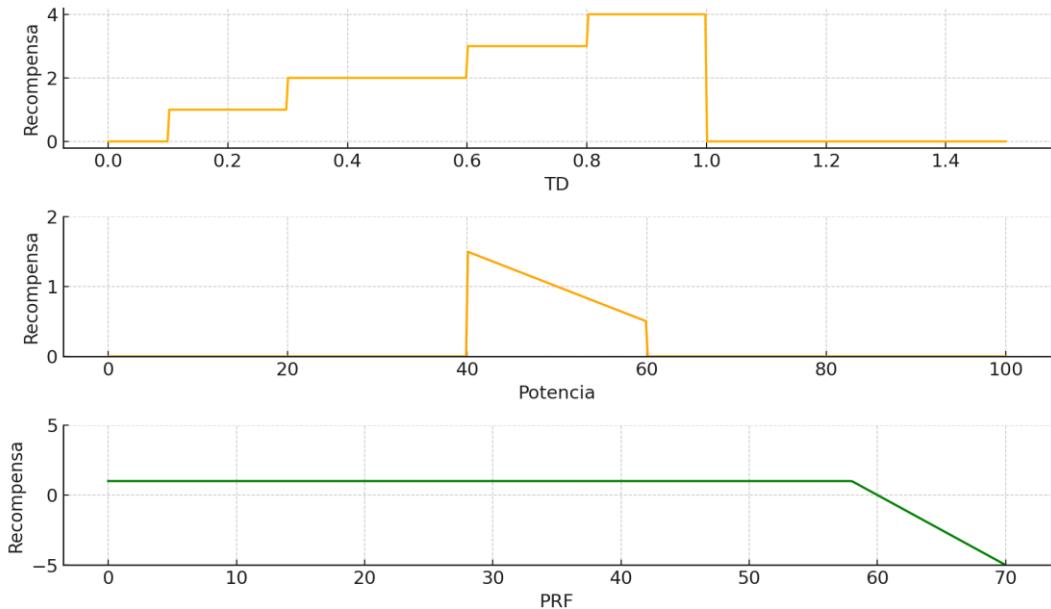


Figura 13: Gráficas de recompensas

3.5 Protocolo de evaluación del modelo

Una vez entrenado el modelo de aprendizaje por refuerzo, se procedió a realizar pruebas siguiendo el diagrama de flujo presente en la Figura 14, para evaluar su capacidad de generalización y su rendimiento en escenarios no incluidos en la etapa de entrenamiento.

La lógica para el análisis es similar al de entrenamiento con la diferencia que ya no funciona la política de Epsilon-Greedy, sino que directamente siempre explotara lo que aprendió en el entrenamiento y además no se precisa de la ecuación de Bellman ya que no modificaremos los pesos de la red, sólo predice la acción en función del estado armado y va reconfigurando los parámetros del radar.

Para el modelo particular de cada ejemplo se logró hacer las pruebas variando el PRF y potencia con respecto al entrenamiento y viendo si las acciones que me daba el modelo efectivamente mejoraban las detecciones analizándolo las gráficas de detecciones de objetivos finales.

Para el modelo general, se aplicó una estrategia de evaluación similar a la del modelo particular, pero con un enfoque adicional en la variación de los parámetros que forman los estados de los datos de prueba con respecto a los de entrenamiento. Esta variación en los datos de prueba es esencial, ya que permite observar cómo responde el modelo ante situaciones que no se encuentran exactamente en los datos de entrenamiento, evaluando así su capacidad de generalización. Este enfoque asegura que el modelo no dependa de características específicas de los datos de entrenamiento, sino que pueda reconocer patrones amplios aplicables a casos no vistos. Esto es fundamental para lograr un modelo robusto y generalizable, capaz de funcionar adecuadamente en escenarios diversos.

Se considera que el modelo DRL es el apropiado si el mismo logra converger a los parámetros de radar óptimos para un dado escenario de búsqueda, en un numero pequeño de iteraciones.

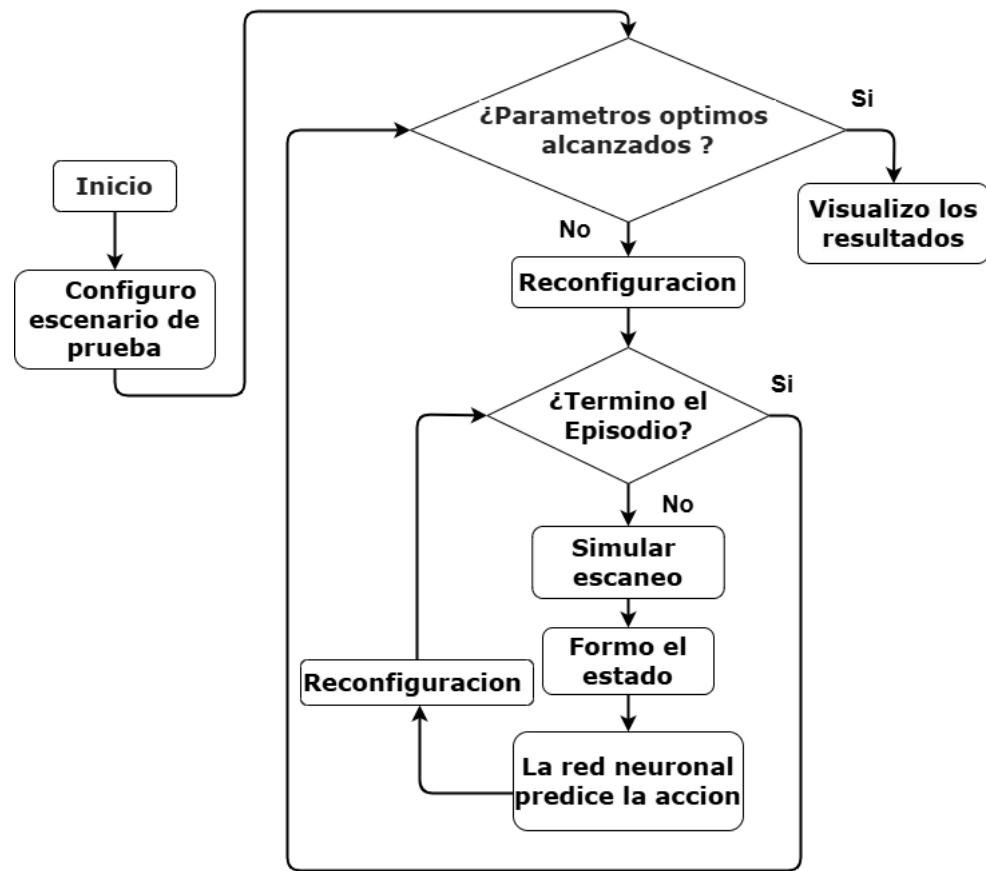


Figura 14: Procedimiento de pruebas

4 Resultados

En este capítulo, se presentan y analizan algunos resultados obtenidos tras el entrenamiento y prueba del modelo obtenido. Estos resultados permiten evaluar la efectividad del enfoque propuesto y su capacidad para aprender y tomar decisiones de manera óptima por parte del agente.

Se realizaron 13 ejemplos, donde cada uno define un determinado escenario de búsqueda. Los escenarios se definen parametrizando las entradas del simulador OTH descriptas en la sección 2.2.1. A continuación, se describen solamente 3 de los ejemplos.

4.1 Primer ejemplo

En este caso se configuraron los parámetros de partida del transmisor de la siguiente forma para el entrenamiento:

- Tipo de Modulation: BPSK
- Ancho de banda: 10000 Hz
- PRF: 26 Hz
- Potencia: 42 dBW
- T: 0.0011 s
- N: 250
- Fs: 30000 Hz
- Fc: 13.52 MHz
- Tipo de código: Barker 11

Al entrenar el modelo se obtuvieron las siguientes gráficas (Figura 15 y 16) que muestran como a medida que pasan los escaneos el modelo va evolucionando, teniendo en cuenta que cada 10 escaneos tenemos un episodio:

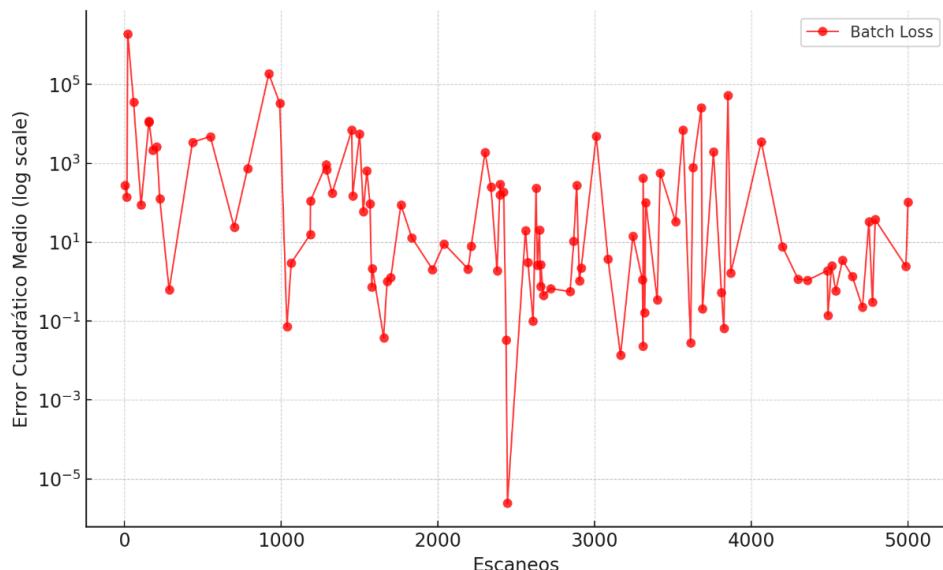


Figura 15: Error cuadrático medio en función de los escaneos

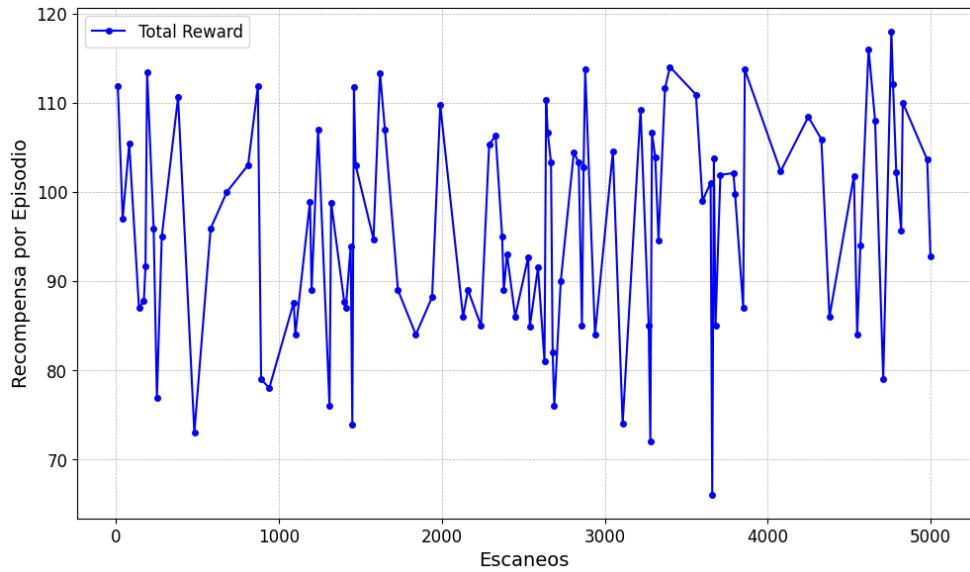


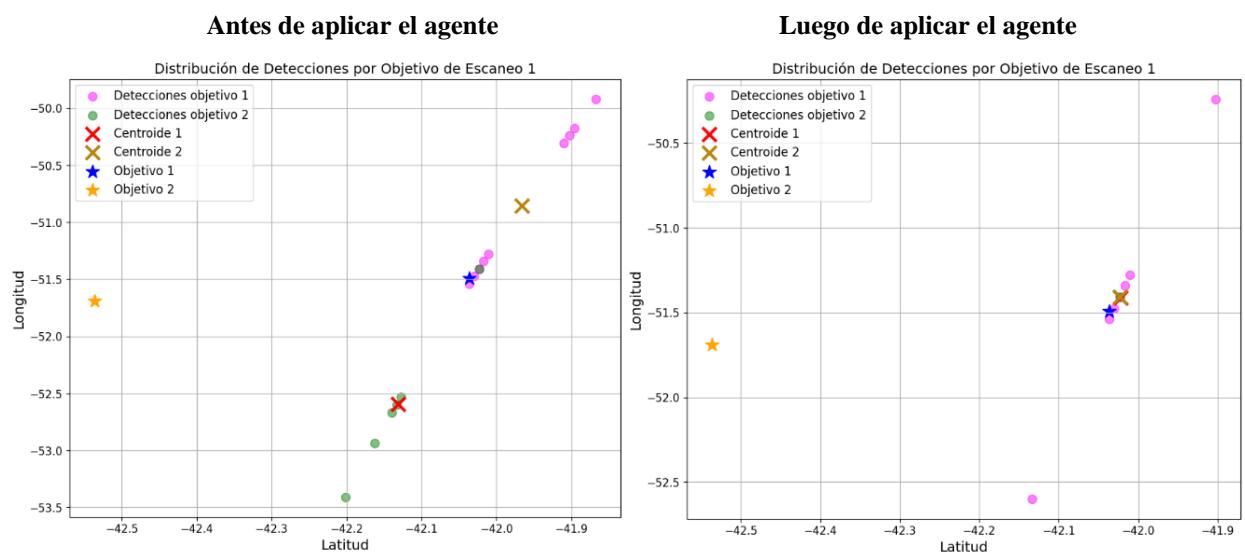
Figura 16: Recompensa acumulada por episodio en función escaneos

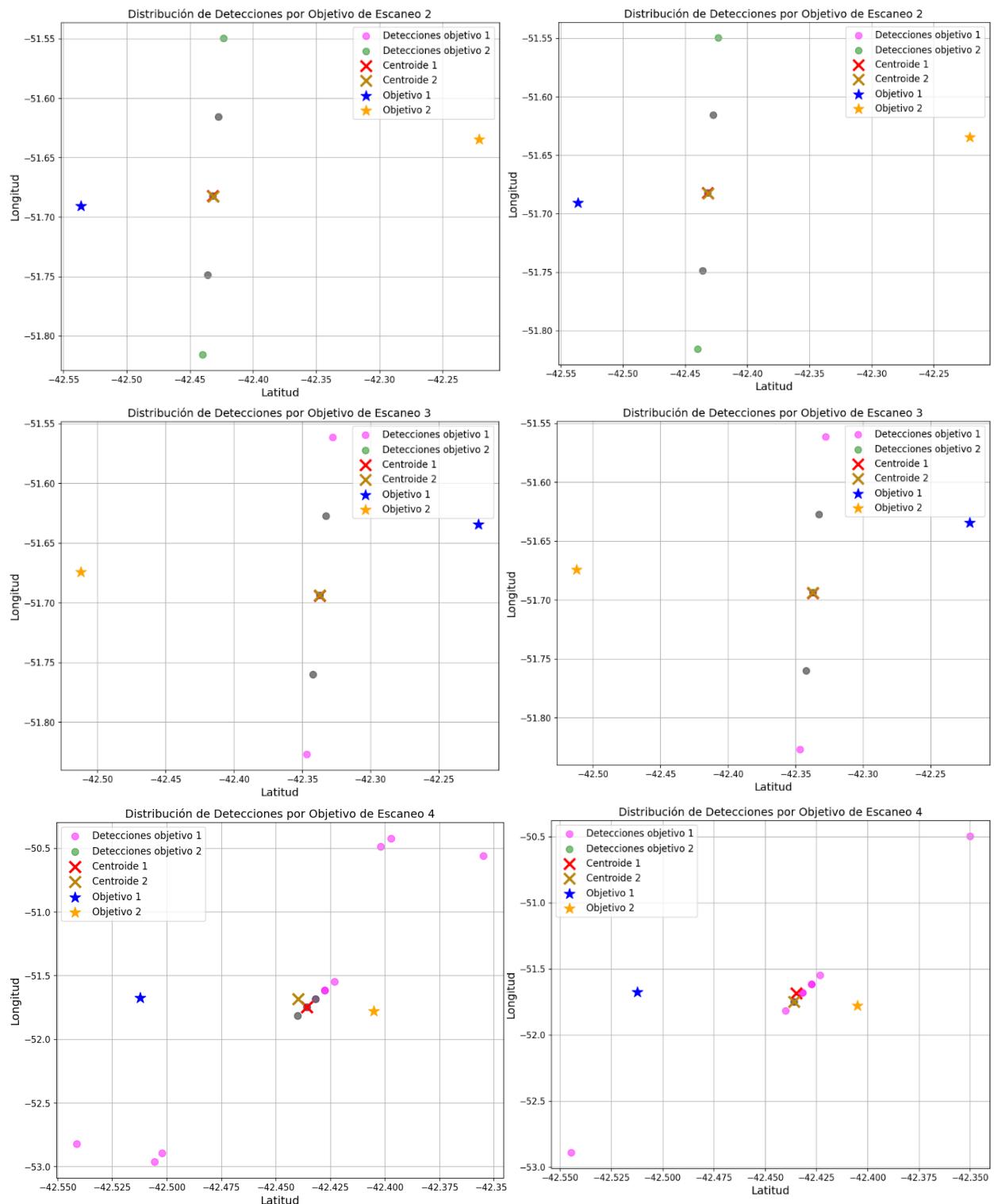
Finalizada la etapa de entrenamiento se obtiene un modelo DRL correspondiente al primer ejemplo. Para este ejemplo se tiene 10 escaneos y 500 episodios.

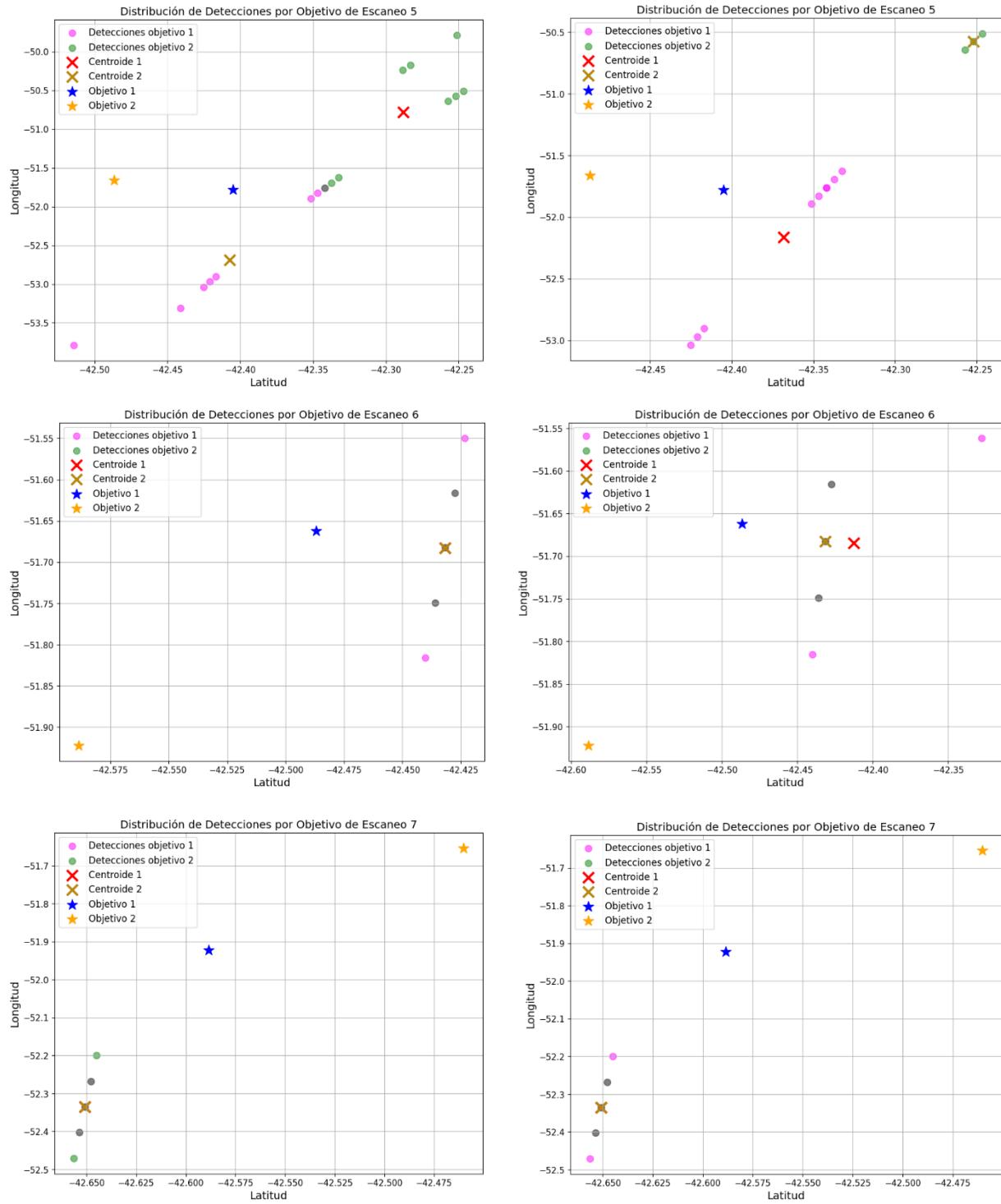
Para la prueba de este modelo se configuraron los parámetros de partida del transmisor de la misma forma que en el entrenamiento, pero con PRF=30 Hz y potencia=39 dBW.

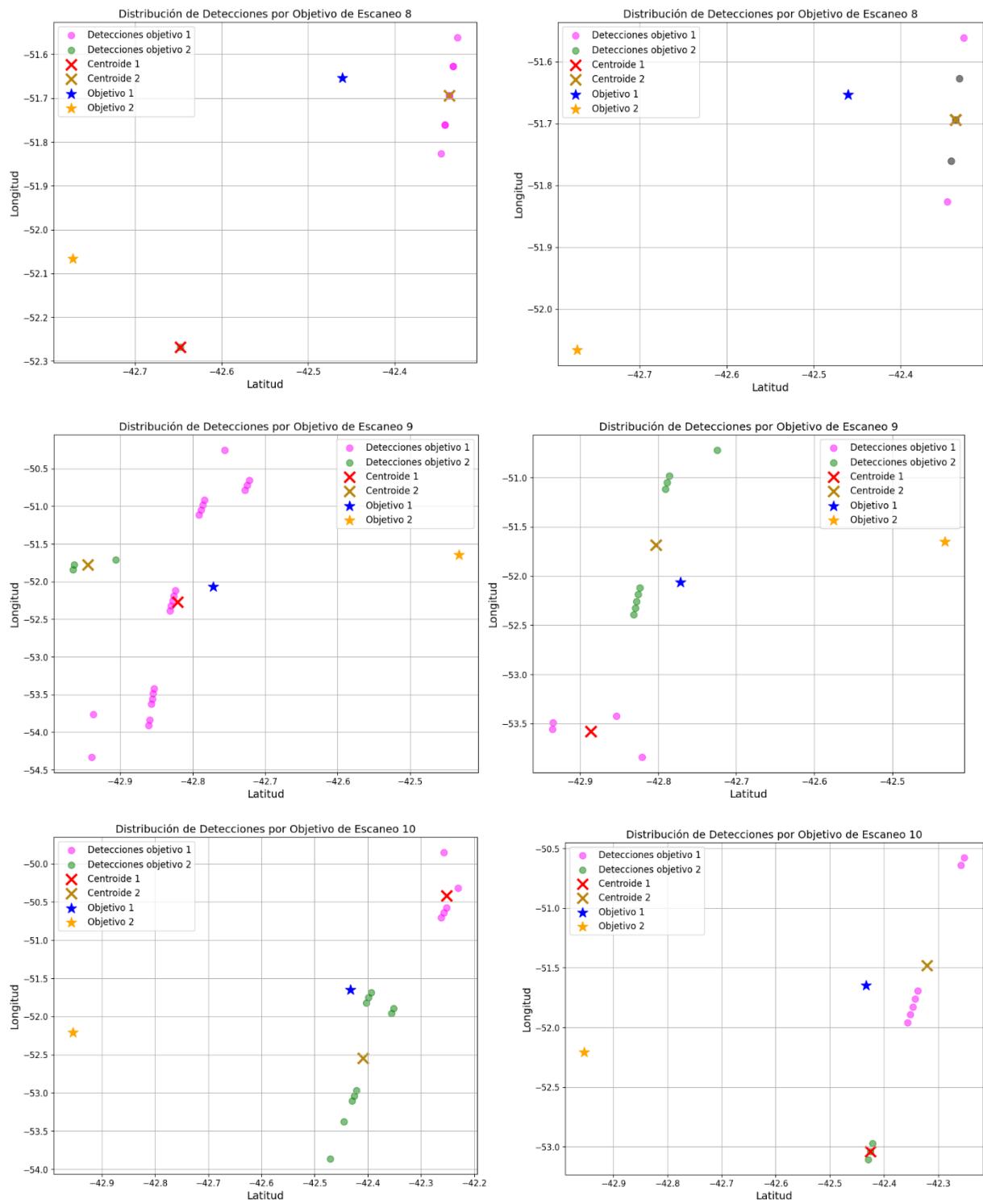
Luego de aplicar 5 veces el predictor del modelo del agente, en donde cada vez se obtiene como resultado la aplicación de la acción “3” de decrementar la potencia 1 dBW y al final terminó con 34 dBW. Posterior a esto ya se obtuvo como respuesta la acción 4 de no modificar ningún parámetro más.

La diferencia presente entre aplicar y no aplicar el agente es el siguiente, se presenta en las siguientes figuras.









4.2 Segundo ejemplo

En este caso se configuraron los parámetros de partida del transmisor de la siguiente forma para el entrenamiento:

- Tipo de Modulation: LFM
- Ancho de banda: 12000 Hz
- PRF: 24 Hz
- Potencia: 42 dBW
- T: 0.001 s
- N: 250
- Fs: 5000 Hz
- Fc: 14.4 MHz
- Tipo de código: Barker 11

Al entrenar el modelo se obtuvieron las siguientes graficas (Figura 17 y 18) que muestran como a medida que pasan los escaneos el modelo va evolucionando, teniendo en cuenta que cada 10 escaneos tenemos un episodio:

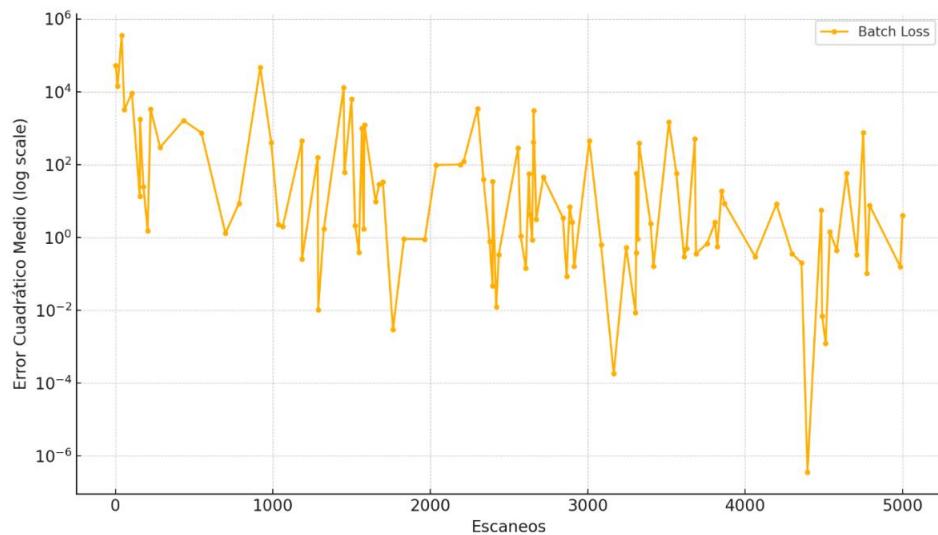


Figura 17: Error cuadrático medio en función de los escaneos

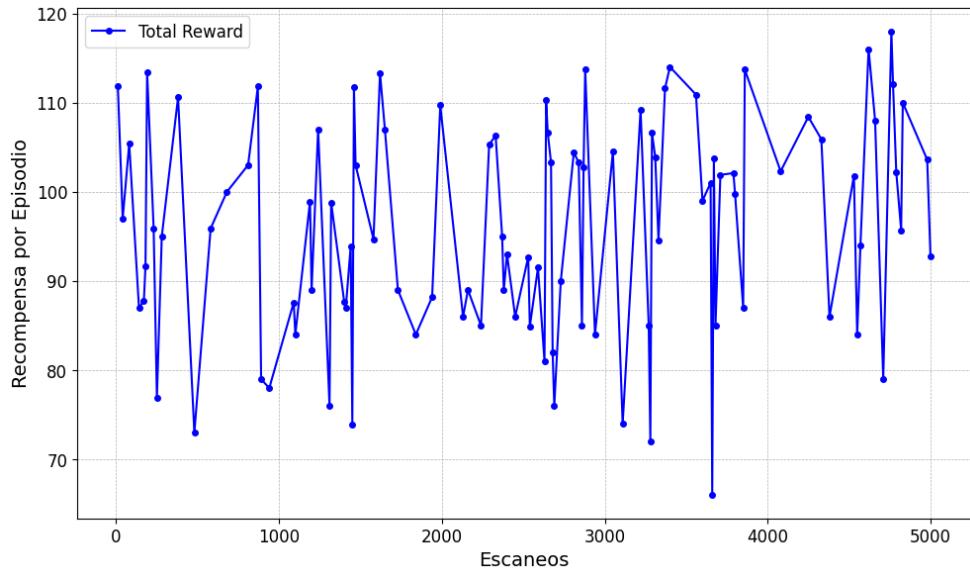


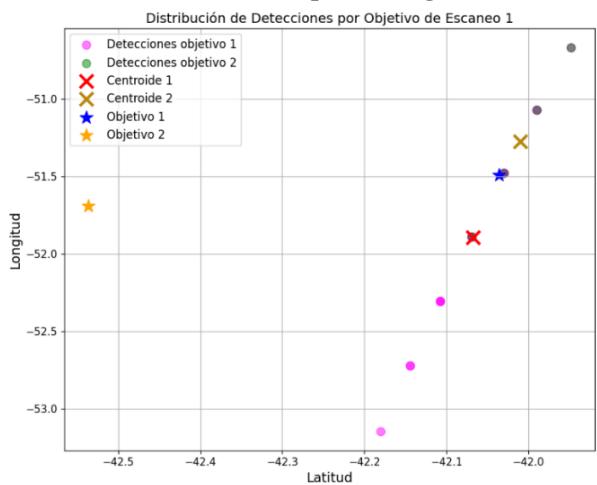
Figura 18: Recompensa acumulada por episodio en función escaneos

Finalizada la etapa de entrenamiento se obtiene un modelo DRL correspondiente al primer ejemplo. Para este ejemplo se tiene 10 escaneos y 500 episodios.

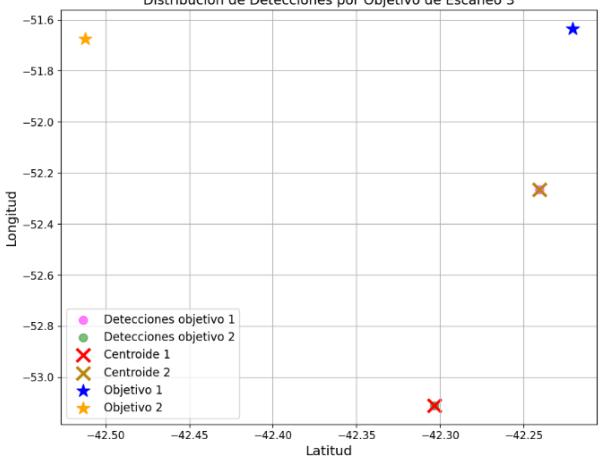
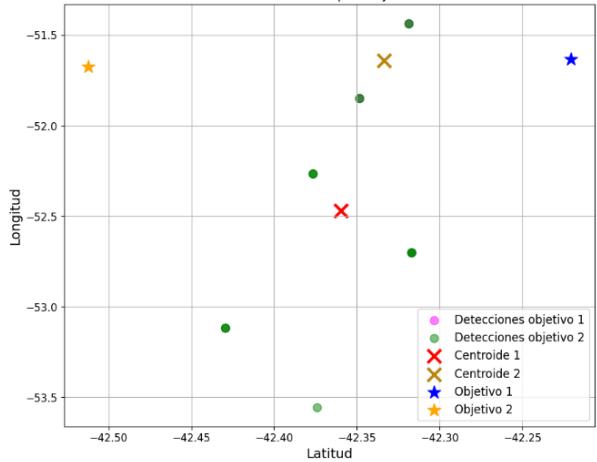
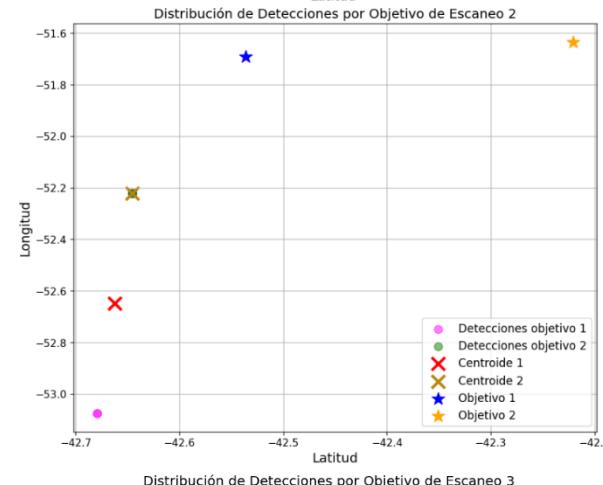
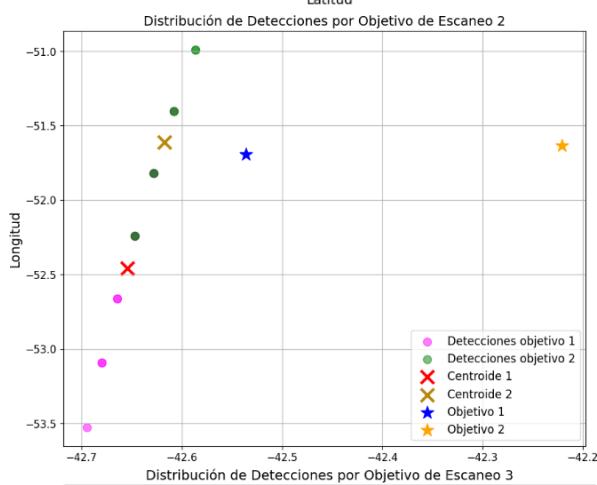
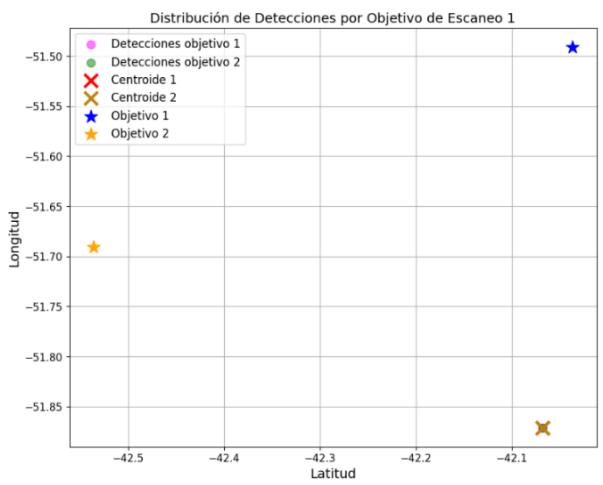
Para la prueba de este modelo se configuraron los parámetros de partida del transmisor de la misma forma que en el entrenamiento, pero con PRF=28 Hz y potencia=46 dBW.

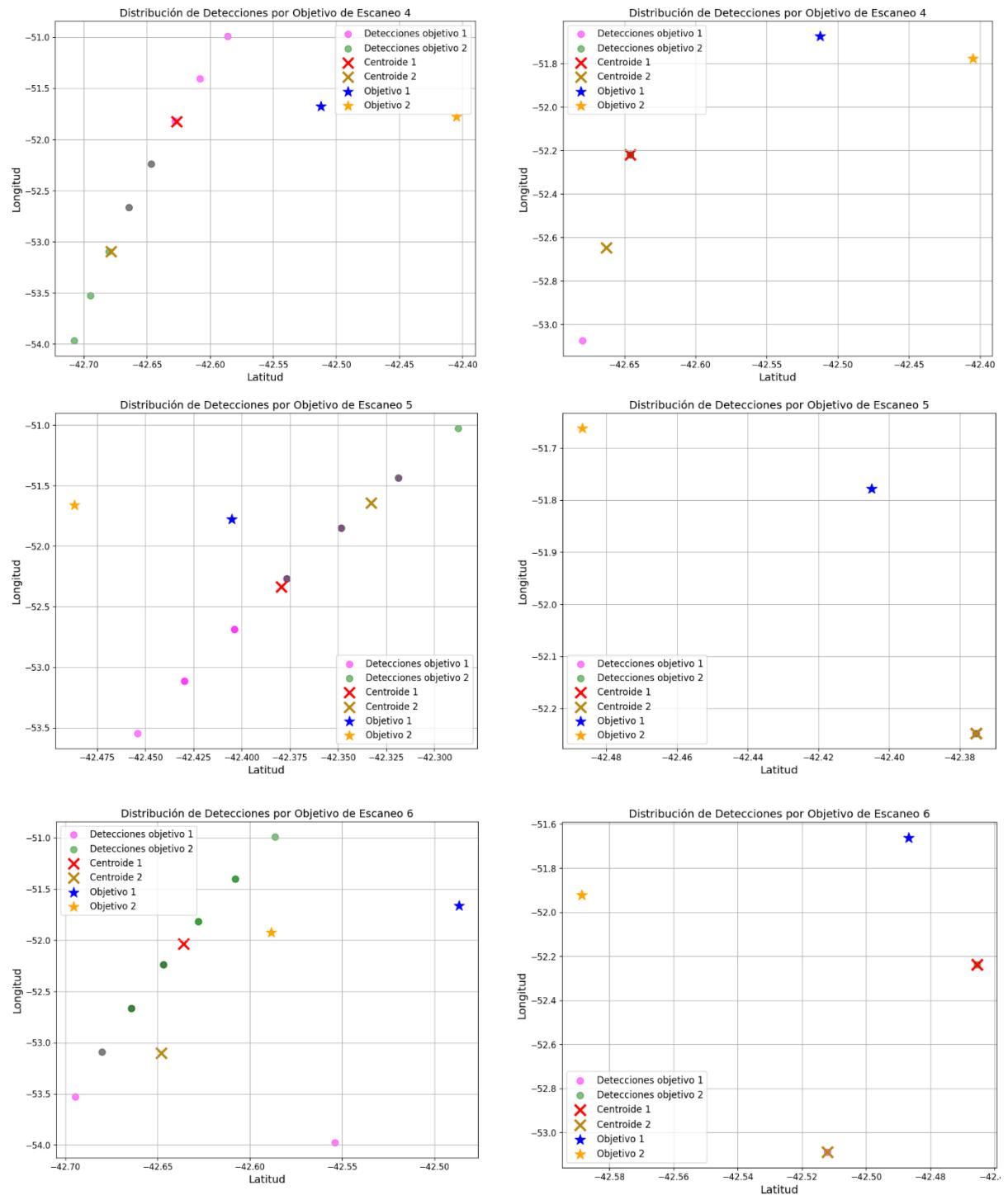
Luego de aplicar 2 veces el predictor del modelo del agente, en donde cada vez se obtiene como resultado la aplicación de la acción “2” de aumentar la potencia 1dBW y la acción “0” de aumentar la el PRF 2 Hz, al final termino con una PRF de 30 Hz y la potencia de 47 dBW. Posterior a esto ya se obtuvo como respuesta la acción 4 de no modificar ningún parámetro más. La diferencia presente entre aplicar y no aplicar el agente es el siguiente, se presenta en las siguientes figuras.

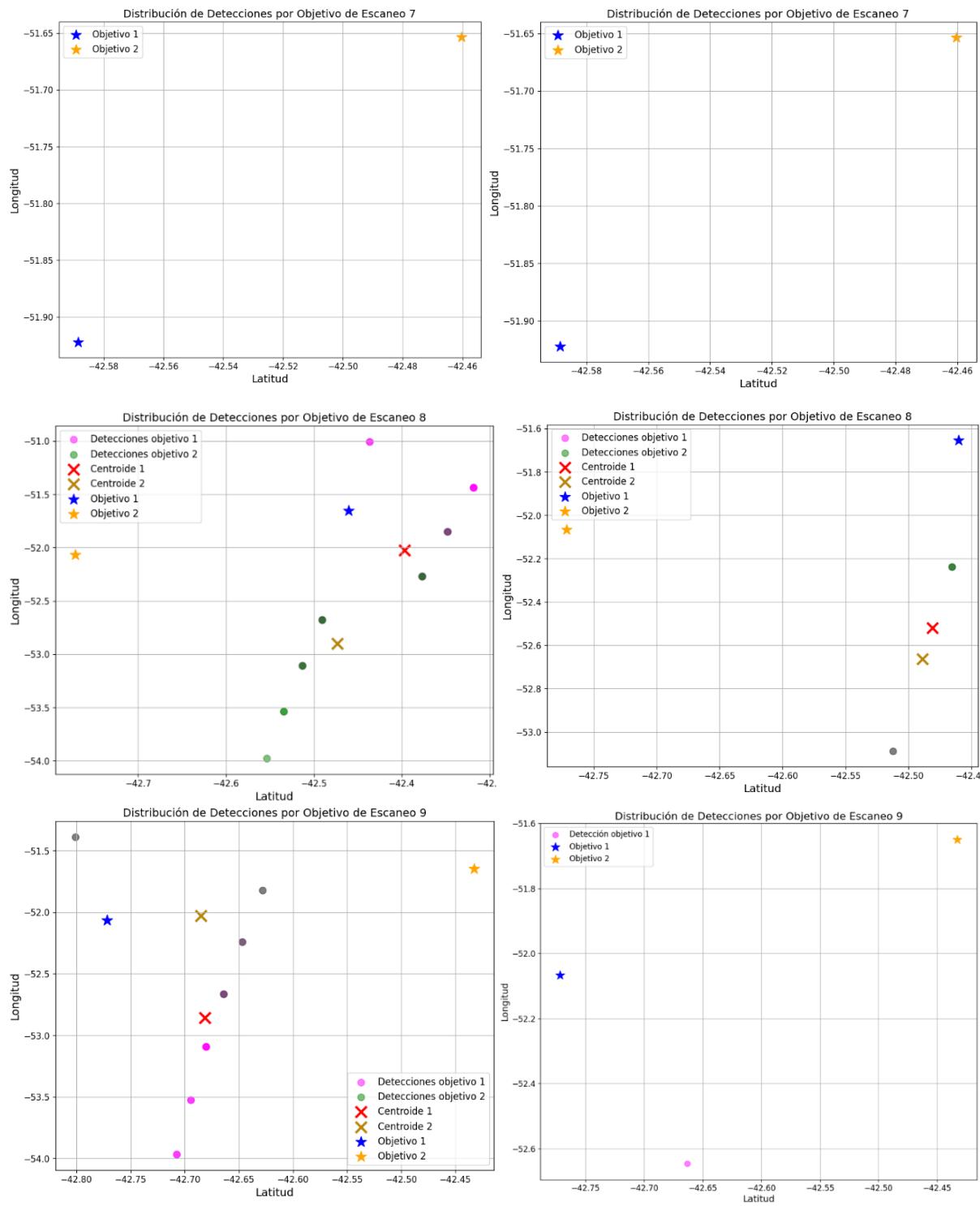
Antes de aplicar el agente

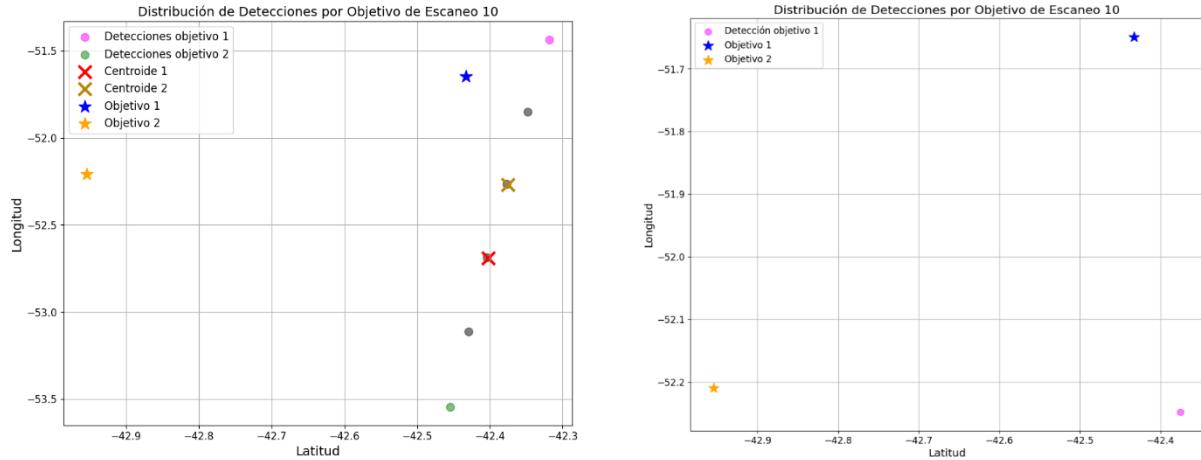


Luego de aplicar el agente









4.3 Tercer ejemplo

En este caso se configuraron los parámetros de partida del transmisor de la siguiente forma para el entrenamiento:

- Tipo de Modulation: BPSK
- Ancho de banda: 9400 Hz
- PRF: 30 Hz
- Potencia: 42 dBW
- T: 0.0017 s
- N: 270
- Fs: 24000 Hz
- Fc: 13.5 MHz
- Tipo de código: Complemento 1

Al entrenar el modelo se obtuvieron las siguientes gráficas (Figura 19 y 20) que muestran como a medida que pasan los escaneos el modelo va evolucionando, teniendo en cuenta que cada 10 escaneos tenemos un episodio:

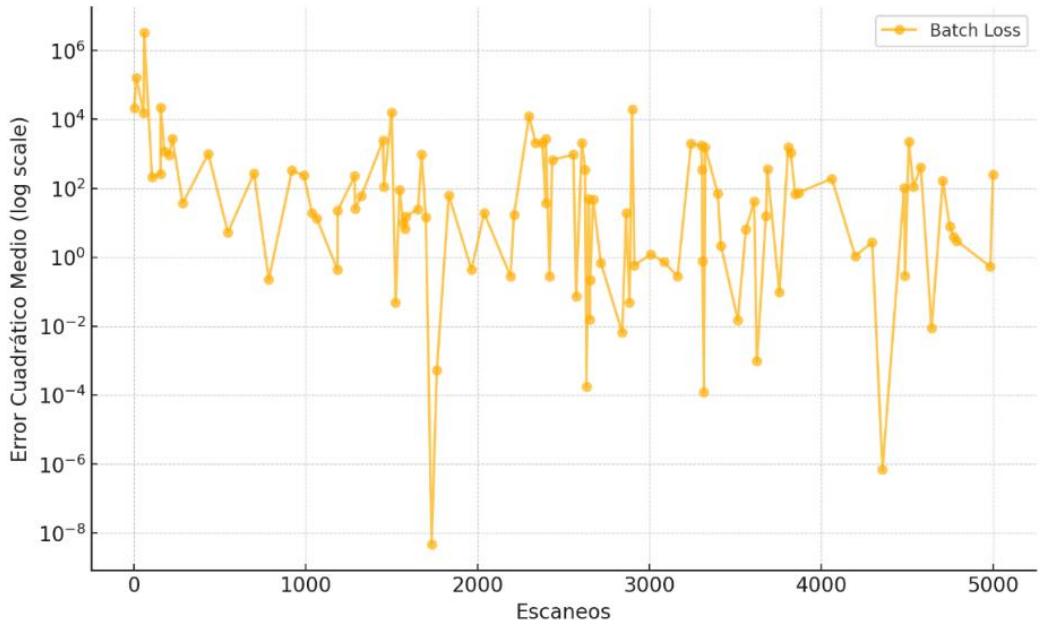


Figura 19: Error cuadrático medio en función de los escaneos

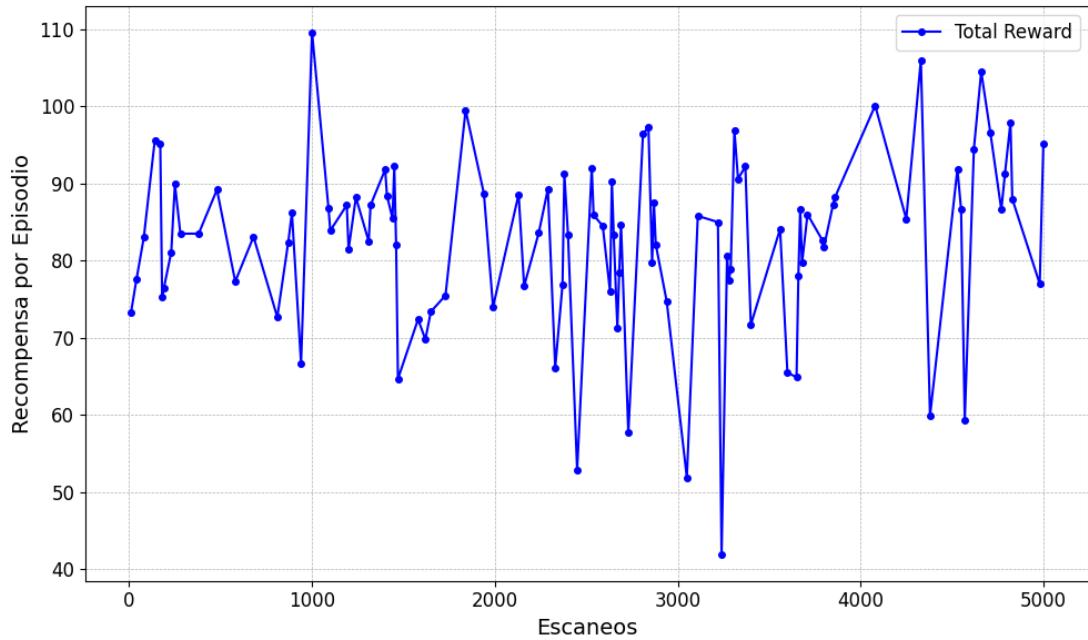


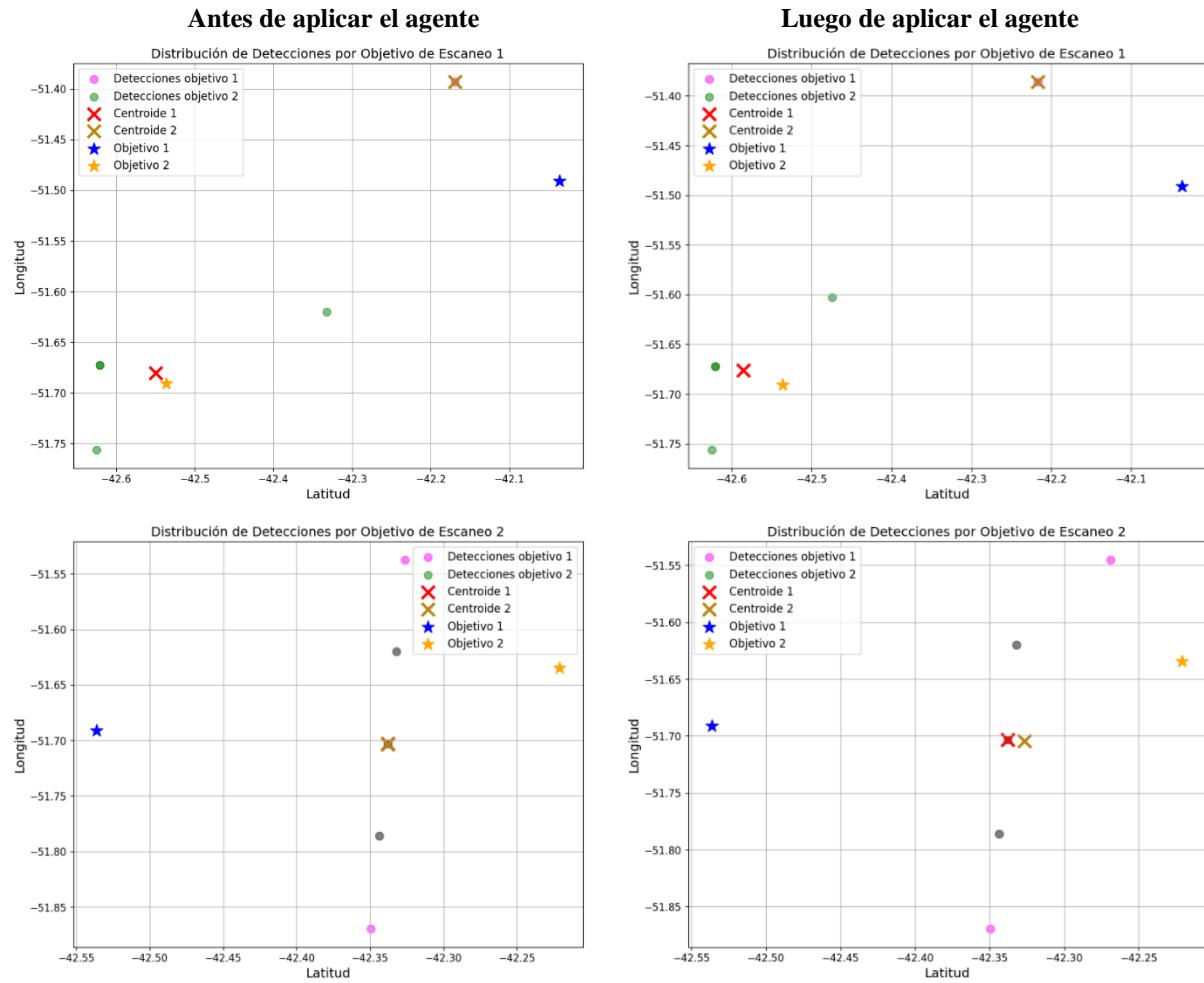
Figura 20: Recompensa acumulada por episodio en función escaneos

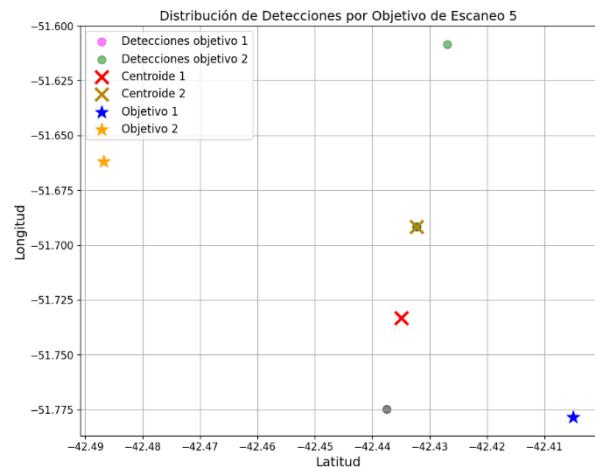
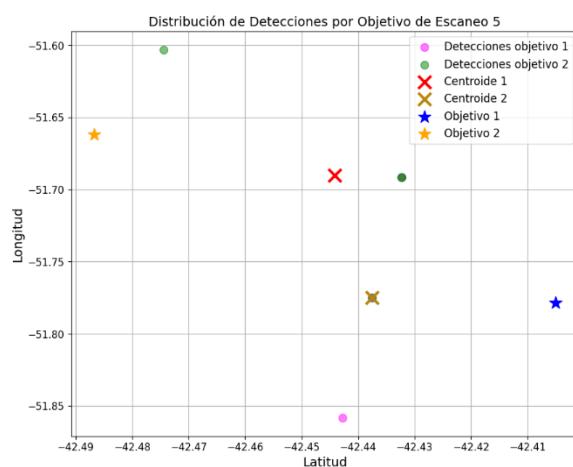
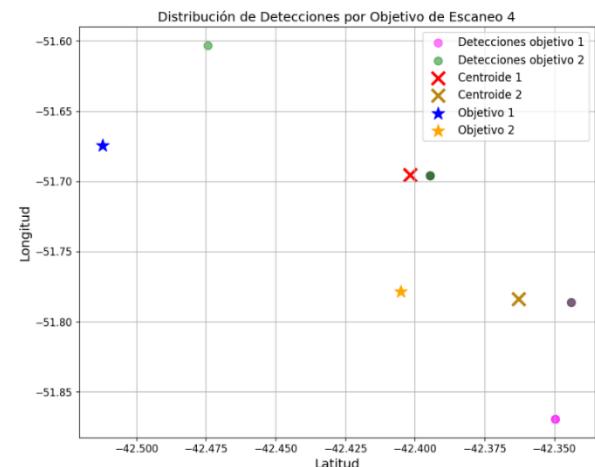
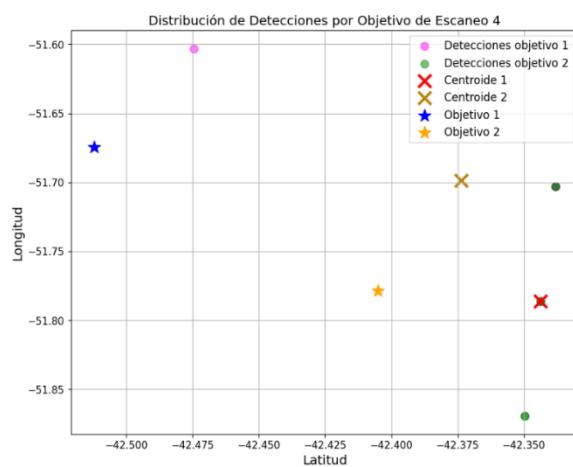
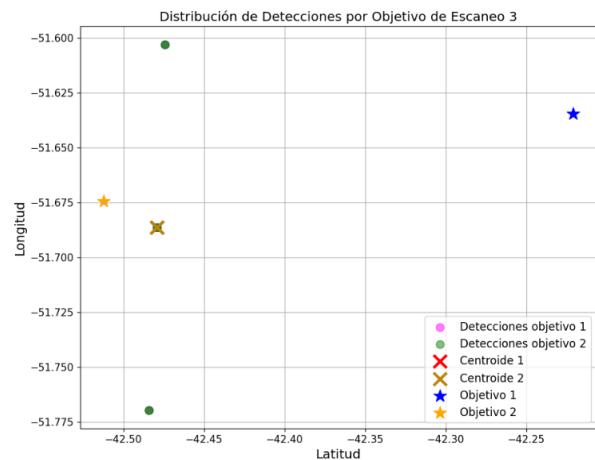
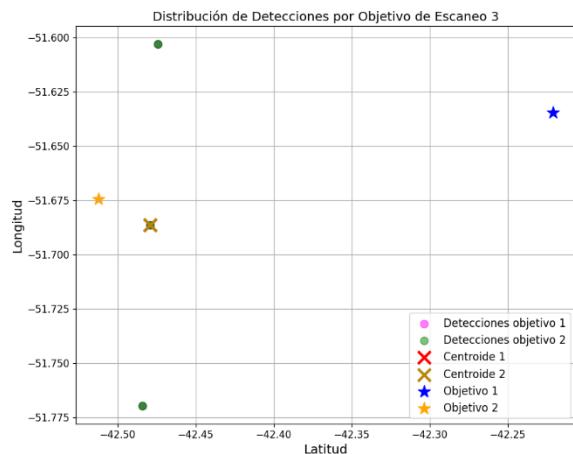
Finalizada la etapa de entrenamiento se obtiene un modelo DRL correspondiente al primer ejemplo. Para este ejemplo se tiene 10 escaneos y 500 episodios.

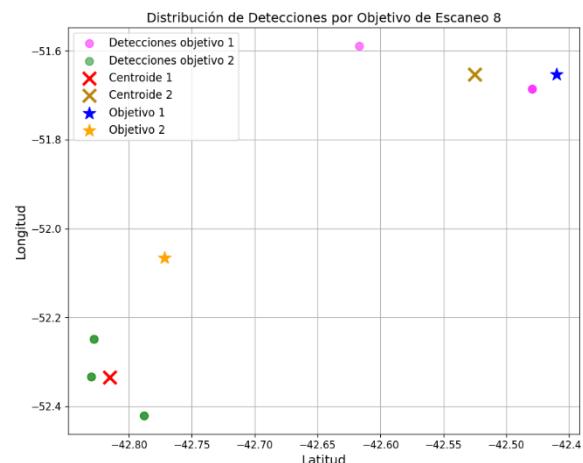
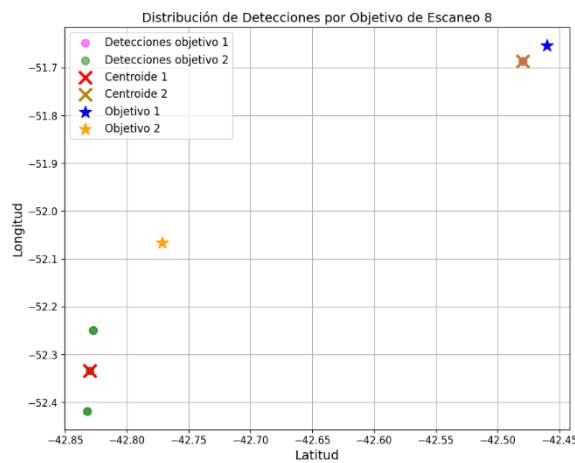
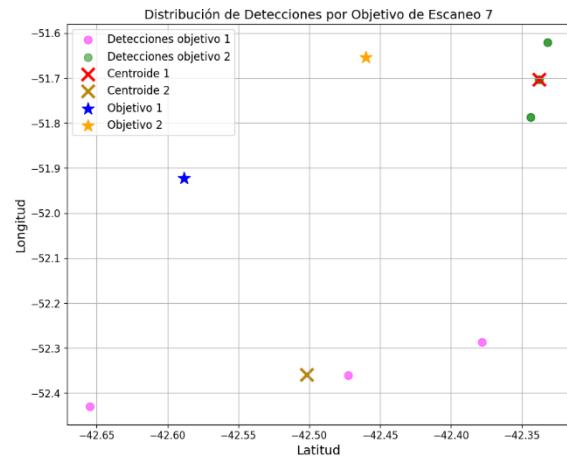
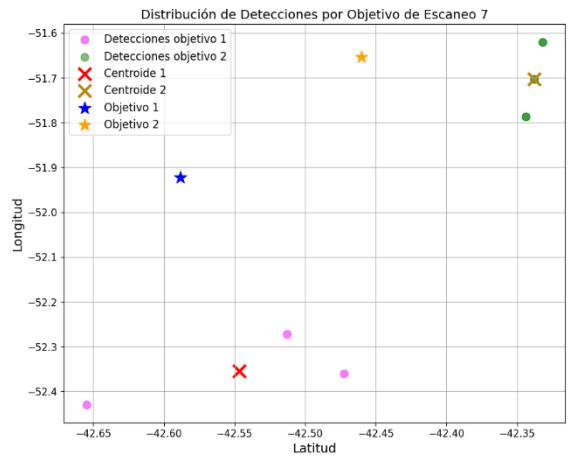
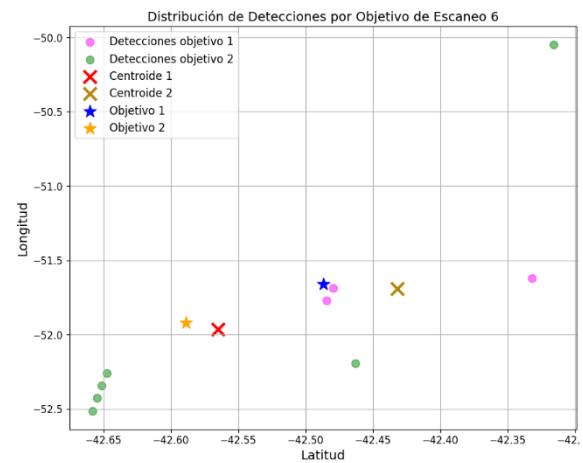
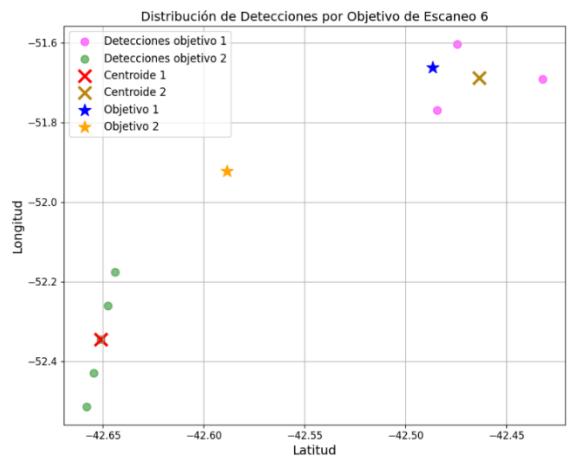
Para la prueba de este modelo se configuro los parámetros de partida del transmisor de la misma forma que en el entrenamiento, pero con PRF=34 Hz y potencia=44 dBW.

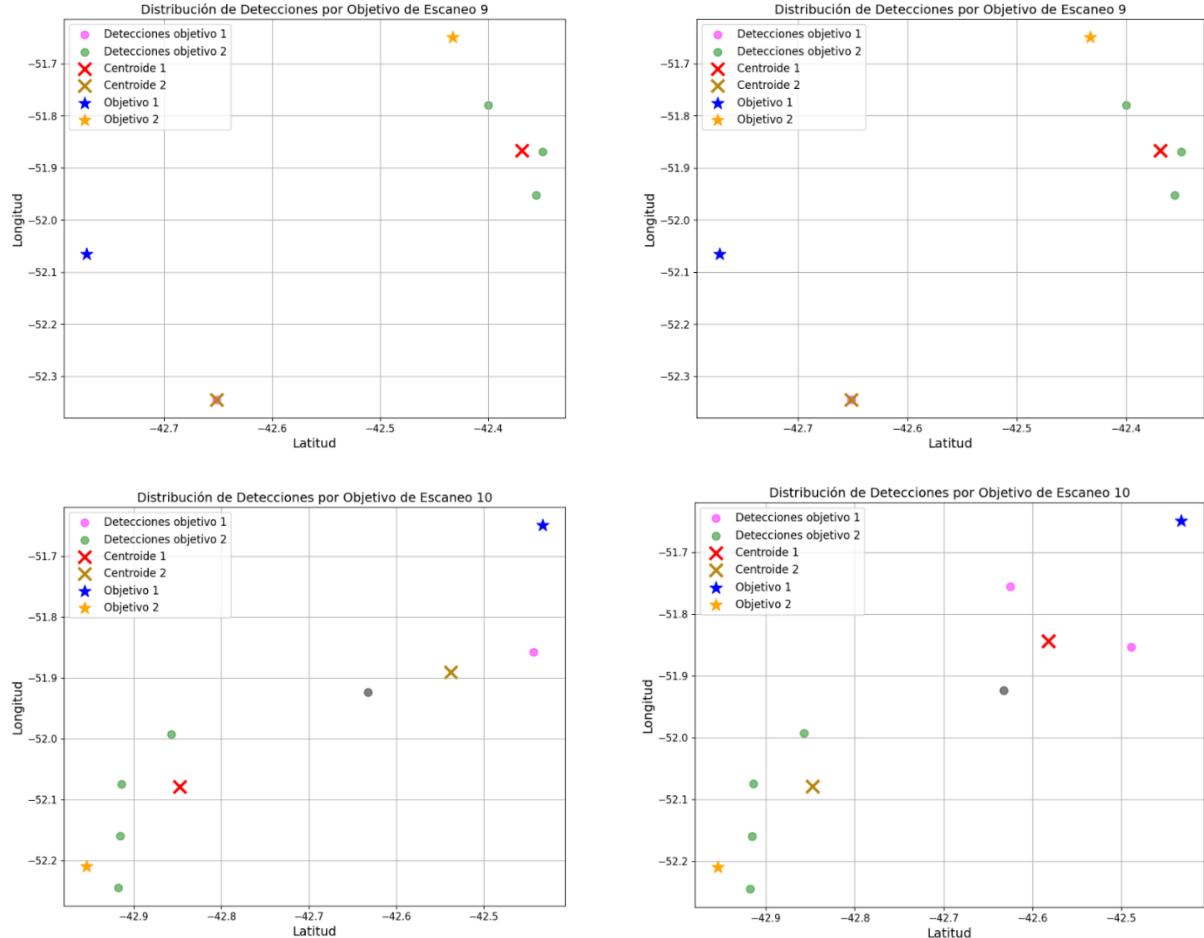
Luego de aplicar una vez el predictor del modelo del agente, se obtiene como resultado la aplicación de la acción “2” de aumentar la potencia 1 dBW. Luego el predictor recomendó aplicar las acciones ‘0’ y ‘1’, que es aumentar el PRF 2 Hz y luego disminuirlo en una misma cantidad respectivamente, por lo que su aumento neto es 0Hz. Finalmente terminó con PRF en

35 Hz y la potencia en 45 dBW. La diferencia presente entre aplicar y no aplicar el agente es el siguiente, se presenta en las siguientes figuras.









4.4 Ejemplo modelo general

Luego de obtener las transiciones de los modelos particulares, se procedió a armar un modelo general explicados en las secciones [3.2](#) y [3.3.2](#). Los resultados del entrenamiento ilustran la evolución del error cuadrático medio (Figura 21) durante el proceso de entrenamiento, considerando lotes de 128 transiciones cada uno (denominados batches en inglés).

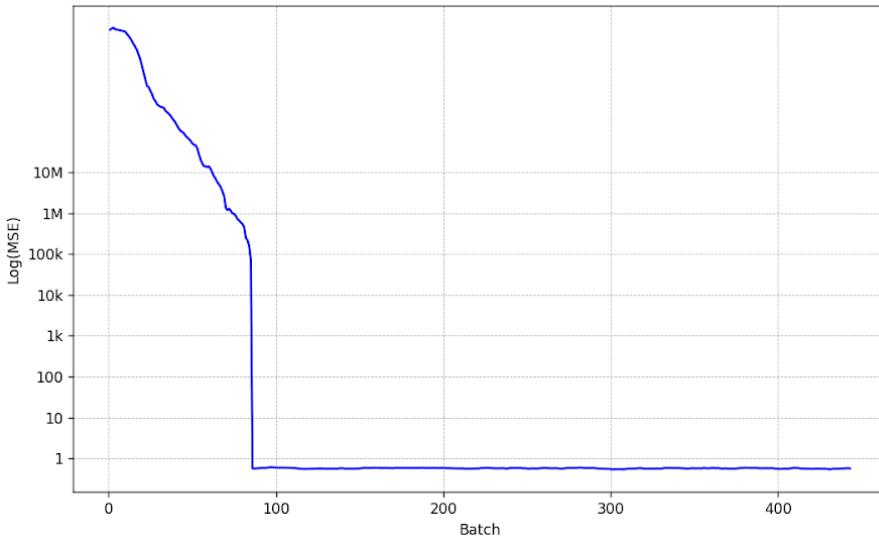


Figura 21: Error cuadrático medio en función de los lotes

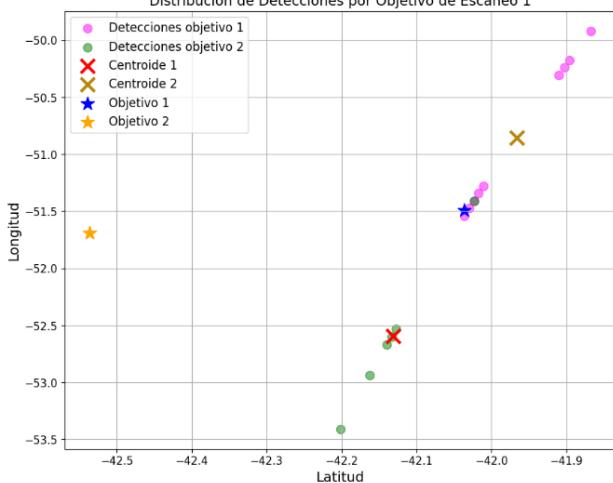
Tomando como referencia los parámetros de prueba del ejemplo 1, se configuraron los parámetros del simulador de la siguiente manera:

- Tipo de Modulation: BPSK
- Ancho de banda: 10000 Hz
- PRF: 30 Hz
- Potencia: 39 dBW
- T: 0.0011 s
- N: 250
- Fs: 30000 Hz
- Fc: 13.52 MHz
- Tipo de código: Barker 11

Luego de aplicar 3 veces el predictor del modelo del agente, que pide aplicar la acción “1” de decrementar PRF 2Hz. Luego el predictor recomendó aplicar las acciones ‘2’ y ‘3’, que es aumentar la potencia 1 dBW y luego disminuirlo en una misma cantidad respectivamente, por lo que su aumento neto es 0dBW. Finalmente terminó con PRF en 28Hz y la potencia en 39dBW. La diferencia presente entre aplicar y no aplicar el agente es el siguiente, se presenta en las siguientes figuras.

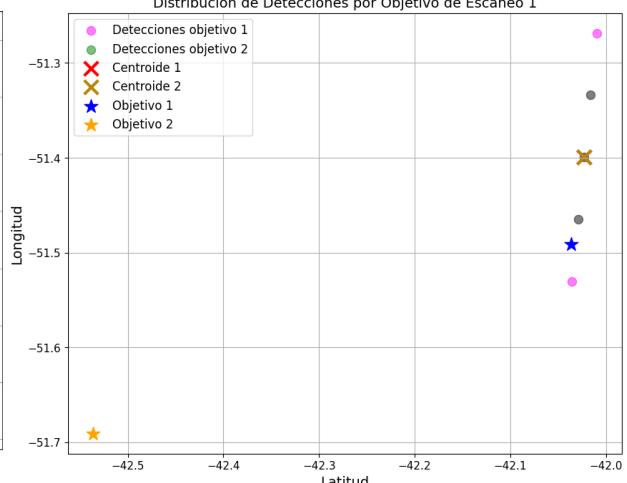
Antes de aplicar el agente

Distribución de Detecciones por Objetivo de Escaneo 1

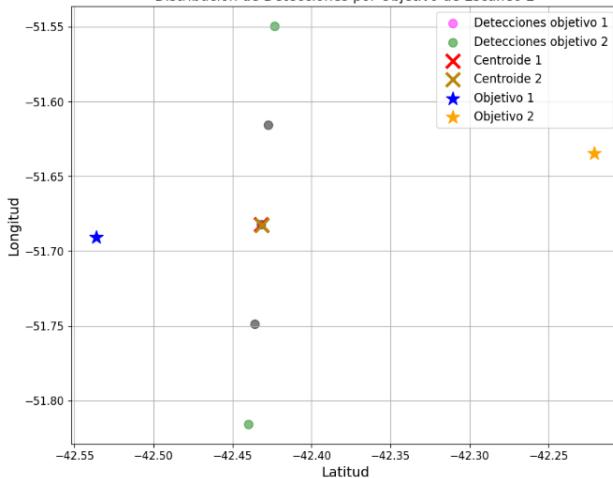


Luego de aplicar el agente

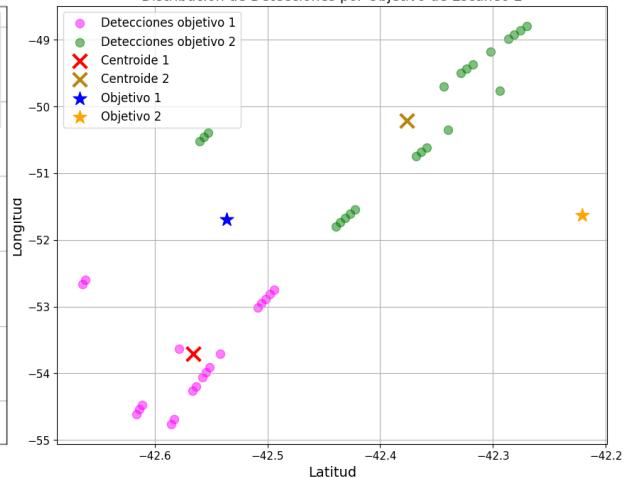
Distribución de Detecciones por Objetivo de Escaneo 1



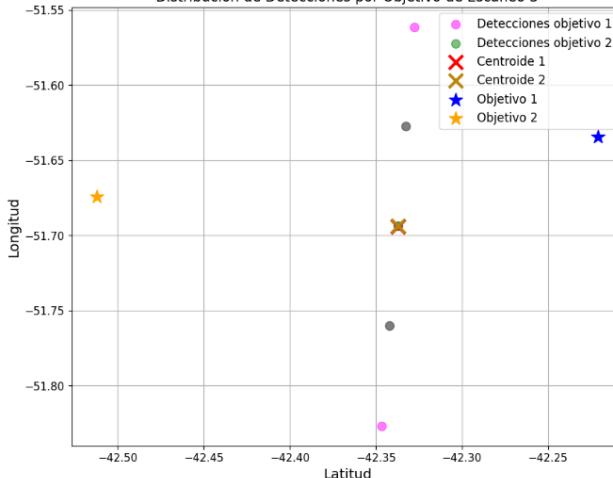
Distribución de Detecciones por Objetivo de Escaneo 2



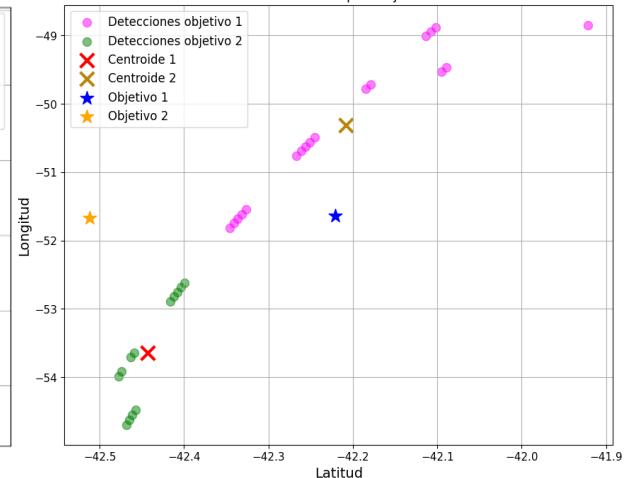
Distribución de Detecciones por Objetivo de Escaneo 2

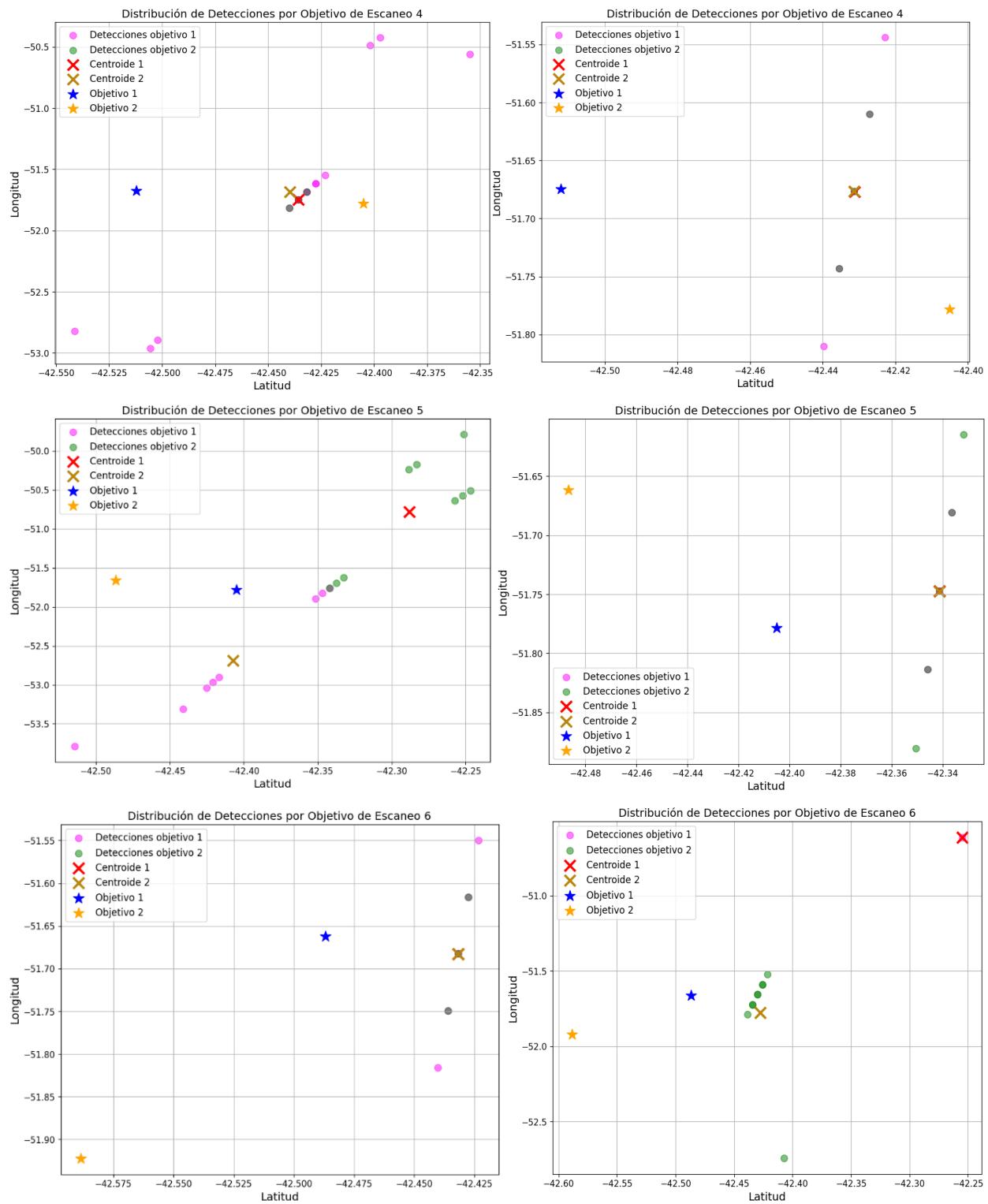


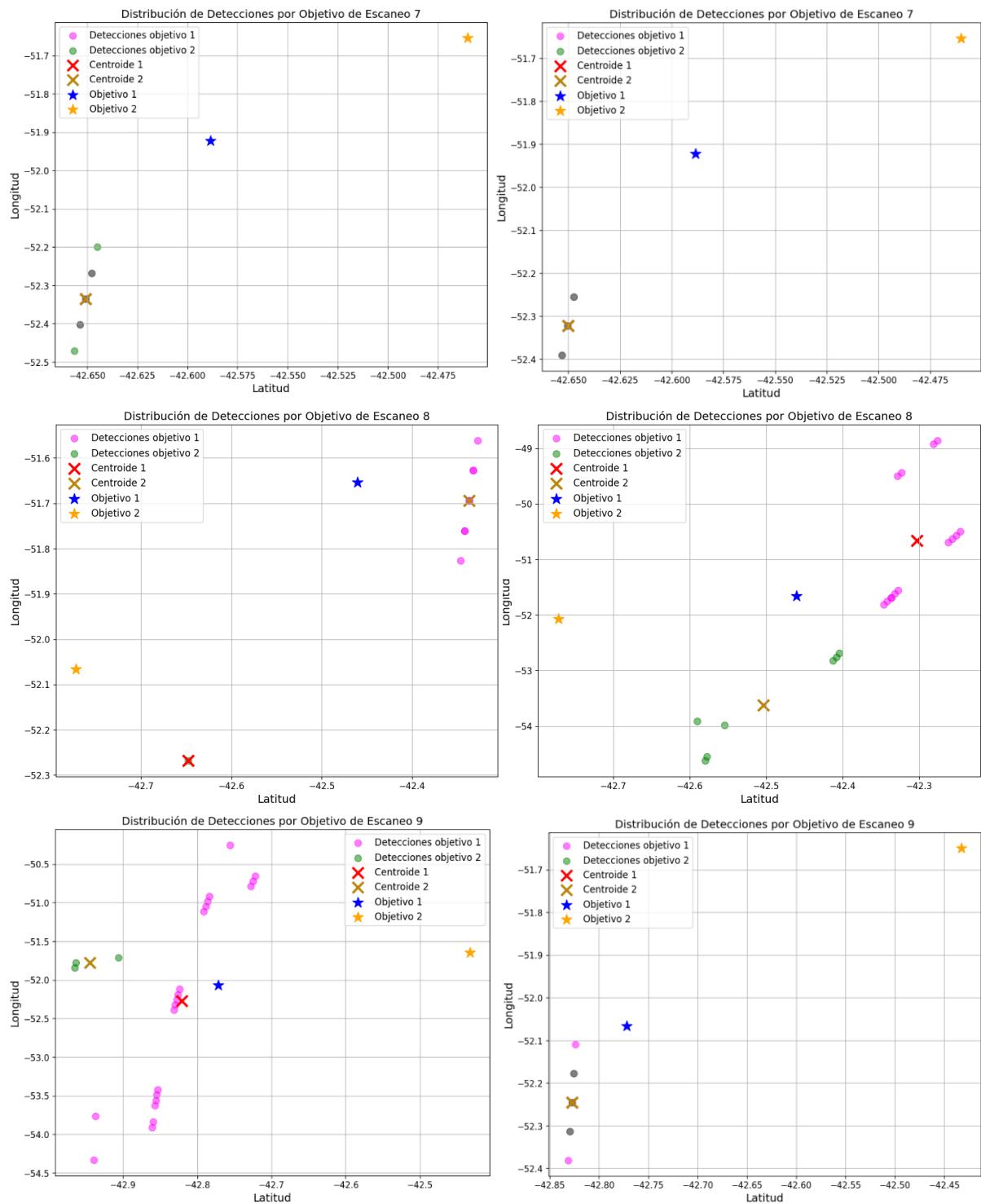
Distribución de Detecciones por Objetivo de Escaneo 3

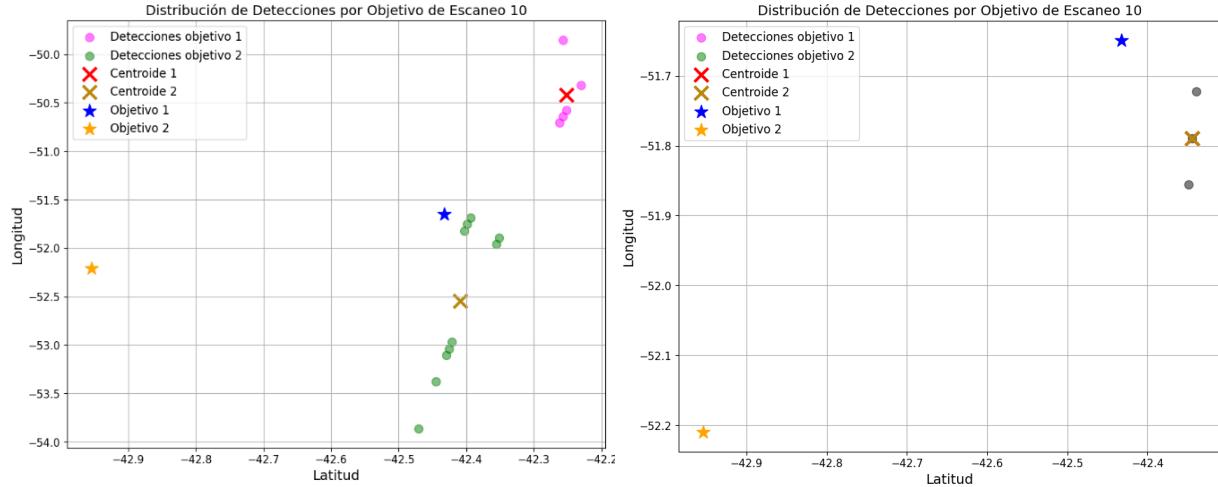


Distribución de Detecciones por Objetivo de Escaneo 3









4.5 Análisis de resultados

Los resultados del entrenamiento presentan, para los tres casos analizados, las gráficas correspondientes a la función de pérdida (error cuadrático medio) (Fig. 15 a 20) y las recompensas obtenidas por episodio. Estas gráficas reflejan que el modelo aprende de manera progresiva durante el entrenamiento. Esto se observa en la disminución del error cuadrático medio, lo que indica un ajuste continuo de los pesos para minimizar el error en las predicciones, y en el aumento de las recompensas por episodio, que demuestra que el modelo está optimizando su política para mejorar el rendimiento en la tarea.

Sin embargo, las fluctuaciones en ambas gráficas sugieren que el modelo enfrenta dificultades para converger hacia una política completamente óptima. Estas oscilaciones pueden estar relacionadas con la exploración de nuevas estrategias o la complejidad del entorno y los datos. A pesar de esto, hacia el final del entrenamiento se observa una tendencia a la estabilización, especialmente en las recompensas, lo que indica que el modelo está consolidando una política más consistente.

En los ejemplos analizados, los puntos grises representan detecciones solapadas de diferentes agrupamientos, mientras que los puntos en rosa o verde intenso indican detecciones solapadas dentro de un mismo agrupamiento. En los primeros dos casos, el agente, al ajustar los parámetros de PRF y potencia, tiende a reducir la cantidad de detecciones, enfocándose en las más cercanas al objetivo real. En contraste, en el tercer ejemplo, las detecciones muestran un mayor desplazamiento, lo que hace que el centroide generado por K-Means esté más próximo al objetivo real.

En el modelo general, los resultados del entrenamiento (Fig. 21) muestran una disminución consistente del error cuadrático medio a medida que se procesan lotes de 128 transiciones, lo que confirma que el modelo está aprendiendo. Al comparar los resultados del modelo con agente y sin agente, se observa que las detecciones con agente son menos numerosas, más compactas, y tienden a agruparse cerca del objetivo, aumentando la tasa de detección y generando recompensas más altas. Por ejemplo, al comparar el modelo general con el particular

en el ejemplo 1, la recompensa fue ligeramente mayor en el modelo general (91.5 frente a 89). Esto sugiere que el modelo general, al menos en este caso, mejora el rendimiento al predecir acciones óptimas. Esta mejora también es evidente en los gráficos, donde las detecciones tienden a encontrarse a una distancia inferior a 10 km del centroide correspondiente a cada agrupamiento.

Con el modelo general obtenido también se probó variar algunos parámetros de transmisión acorde a los 13 ejemplos con los que se entrenó, como el ancho de pulso T o el número de integraciones. Al hacer esto, el modelo mostró un comportamiento consistente de sobreajuste, manifestando su tendencia a devolver la misma acción independientemente del estado. Este patrón indica que, en lugar de aprender una política generalizable, el modelo memorizó patrones específicos de los datos de entrenamiento.

Una posible causa de este sobreajuste podría ser la falta de ejemplos suficientemente variados y representativos que permitan al modelo generalizar mejor. Esto sugiere que los datos de entrenamiento no abarcan adecuadamente la diversidad de situaciones necesarias para que el modelo capture patrones más amplios y robustos. Como resultado, el desempeño en los datos de prueba fue inferior al esperado, mostrando una fuerte dependencia hacia el conjunto de entrenamiento.

Finalmente, la técnica de agrupamiento K-Means introduce variaciones en la asignación de detecciones a los clusters, lo que añade un grado de error que se propaga al proceso de toma de decisiones del agente. Esto contribuye a la variabilidad observada en las recompensas, reflejando la complejidad del desafío de generalización en este contexto.

5 Conclusión

A lo largo de este trabajo, se demostró que el modelo desarrollado tiene la capacidad de ajustar dinámicamente los parámetros del radar para optimizar la detección de objetivos en diferentes escenarios simulados. En general, el modelo mostró un desempeño consistente al reducir detecciones irrelevantes y priorizar las más cercanas al objetivo real, aunque se observaron variaciones en los resultados debido al uso de técnicas como K-Means, que introducen pequeños errores en las asignaciones.

A pesar de los buenos resultados, queda claro que mejorar el modelo implicaría aumentar la cantidad de episodios o los escaneos por episodio. Sin embargo, esto también significaría un incremento significativo en los tiempos de simulación, lo que plantea un desafío práctico en términos de recursos y eficiencia.

Hacia el futuro, sería interesante explorar cómo adaptar el modelo para trabajar con una cantidad variable de objetivos, diseñando estados que no dependan puntualmente de los valores específicos de cada uno o que se logren interconectar redes neuronales con una cantidad variable de parámetros de entrada. Además, el enfoque podría aplicarse a otros parámetros del simulador, ampliando su alcance a distintos tipos de escenarios operativos.

En resumen, este modelo representa una base sólida para seguir investigando y refinando el uso del aprendizaje por refuerzo profundo en la optimización de sistemas complejos como los radares, con la posibilidad de hacerlo más eficiente y flexible para enfrentar nuevos desafíos.

6 Bibliografia

- (1) **Skolnik, M. I. (2008).** *Radar Handbook*. McGraw-Hill Education. “Applications of radar” McGraw-Hill Education, pp. 22
- (2) **Sutton, R. S., & Barto, A. G. (2018).** *Reinforcement Learning: An Introduction* (2^a ed., pp. 3-5). MIT Press.
- (3) **Saavedra, Z. (2020).** *Desarrollo de un sistema de radar cognitivo basado en aprendizaje por refuerzo*. Universidad Nacional de Tucumán.
https://www.facet.unt.edu.ar/posgrado/wp-content/uploads/sites/54/2022/11/Tesis_ZSaavedra_2020.pdf
- (4) **Saavedra, Z., Zimmerman, D., Cabrera, M. A., & Elías, A. G. (2020).** Herramienta de simulación de radar de ondas ionosféricas sobre el horizonte. *IET Radar, Sonar & Navigation*. <https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/iet-rsn.2020.0158>
- (5) **SZ Gurbuz, HD Griffiths, A. Charlish, M. Rangaswamy, MS Greco y K. Bell**, "Una descripción general del radar cognitivo: pasado, presente y futuro", en *IEEE Aerospace and Electronic Systems Magazine* , vol. 34, núm. 12, págs. 6-18, 1 de diciembre de 2019, doi: 10.1109/MAES.2019.2953762.
https://ieeexplore.ieee.org/document/8961364?utm_source=chatgpt.com
- (6) **Sutton, R. S., & Barto, A. G. (2018).** *Deep Reinforcement Learning* (2^a ed., pp. 322). MIT Press.
- (7) **Moya, R. (s.f.).** *Aprendizaje por refuerzo con Python*. GitHub.
https://github.com/RicardoMoya/Reinforcement_Learning_with_Python/blob/master/Aprendizaje_Por_Refuerzo.ipynb
- (8) **Argota, N. (2022).** *Diseño y Modelado de un Sistema Digital de Comunicaciones por incidencia Cuasi-vertical*. Universidad Nacional de Tucumán.
<https://www.facet.unt.edu.ar/posgrado/wp-content/uploads/sites/54/2022/10/ARGOTA-Nicolas.pdf>
- (9) **Sutton, R. S., & Barto, A. G. (2018).** *Reinforcement Learning: An Introduction* (2^a ed., pp. 49-52). MIT Press.
- (10) **Luu, Q. T. (2024, 18 de marzo)**. Aprendizaje Q vs. Aprendizaje Q profundo vs. Red Q profunda. *Baeldung*. <https://www.baeldung.com/cs/q-learning-vs-deep-q-learning-vs-deep-q-network>

(11) **EITC.** (2023, 15 de agosto). ¿Qué son los pesos y los sesgos en la IA? *EITC/AI/GCML Google Cloud Machine Learning*. <https://es.eitca.org/artificial-intelligence/eitc-ai-gcml-google-cloud-machine-learning/introduction/what-is-machine-learning/explain-weights-and-biases/>

(12) **Wikipedia.** (s.f.). K-medias. *Wikipedia*. <https://es.wikipedia.org/wiki/K-medias>

(13) **Wikipedia.** (s.f.). Error cuadrático medio. *Wikipedia*.
https://es.wikipedia.org/wiki/Error_cuadr%C3%A1tico_medio

7 Anexos

Anexo A: Estimación del uso de memoria en el entrenamiento

Los valores de las transiciones están almacenados en formato de punto flotante. En Python, el tipo de dato float es generalmente un double precision float (doble precisión) según el estándar IEEE 754, que ocupa 8 bytes (64 bits) por cada valor de punto flotante. Por lo que se estima que el tamaño de la memoria ram aproximada a usar durante este entrenamiento es el siguiente:

Estado: 13 floats \times 8 bytes/float \rightarrow 104 bytes.

Acción: Un entero \rightarrow 1 byte (valores de 0 a 4).

Recompensa: Un float \rightarrow 8 bytes.

Next state: 13 floats \rightarrow 104 bytes.

Done: Un booleano \rightarrow 1 byte.

$$\text{Memoria total por ejemplo} = 4500 \text{ transicion} \times 218 \frac{\text{bytes}}{\text{transición}} \cong 1 \text{ MByte}$$

Esta cifra no es significativa para nuestro caso, pero es útil tener una idea a futuro en caso que se quiera agrandar el tamaño del estado, entrenar más episodios o agregar más acciones.