DESARROLLO DE UN ALGORITMO DE NAVEGACIÓN AUTÓNOMA BASADO EN TÉCNICAS DE APRENDIZAJE POR REFUERZO USANDO INFORMACIÓN VISUAL

DANIEL FELIPE APONTE VARGAS
ERIKA DAYANNA MARTÍNEZ MÉNDEZ

UNIVERSIDAD SANTO TOMÁS
INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C
2022

DESARROLLO DE UN ALGORITMO DE NAVEGACIÓN AUTÓNOMA BASADO EN TÉCNICAS DE APRENDIZAJE POR REFUERZO USANDO INFORMACIÓN VISUAL

DANIEL FELIPE APONTE VARGAS
ERIKA DAYANNA MARTÍNEZ MÉNDEZ

Proyecto de grado presentado como requisito para optar al título de
INGENIERO ELECTRÓNICO

UNIVERSIDAD SANTO TOMÁS
INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C
2022

Resumen

Palabras clave: Aprendizaje Profundo por Refuerzo, Redes Q Profundas, Replay Memory, Navegación Autónoma, Información Visual.

En este proyecto se realiza la implementación de un algoritmo de navegación autónoma basado en información visual, usando aprendizaje profundo por refuerzo (DRL, por sus siglas en inglés Deep Reinforcement Learning). El algoritmo le enseña a un agente a identificar patrones visuales para navegar hacia un objetivo en un entorno cerrado y desconocido.
El proceso de aprendizaje se compone de tres etapas: clasificación, imitación y entrenamiento, y un sistema de Replay Memory. Las etapas de aprendizaje brindan al agente diferentes herramientas para categorizar la información y tomar una decisión, transfiriendo el conocimiento adquirido en cada una. Por su parte, el sistema de Replay Memory le proveé información al agente de experiencias pasadas para entender y resolver entornos desconocidos. A su vez, el algoritmo se basa en un modelo de entrenamiento redes Q profundas (DQN, por sus siglas en inglés Deep Q Network), con una recompensa hacia el agente en cada interacción con el entorno. La evaluación del algoritmo se realiza a través de experimentos basados en la interacción con entornos simulados de diferentes tamaños, rutas y caracteracterísticas.

Abstract

Keywords:Deep Reinforcement Learning, Deep Q Networks, Replay Memory, Autonomous Navigation, Visual Information.

This project proposes the implementation of an algorithm autonomous navigation based on visual information using deep reinforcement learning. The algorithm aims to teach an agent to identify visual patterns to navigate to a goal in closed and unknown environments.

The learning process is made out of three stages: Classification, Imitation and Training, and a Replay Memory system. The Learning stages provide the agent with different tools to classify the information and make a decision, transferring the knowledge acquired in each one. Meanwhile, the replay memory provides the agent information from past experiences to understand and solve unfamiliar environments. At the same time, the algorithm is based on a Deep Q Network (DQN) model, with a reward to the agent in each interaction with the environment. The evaluation of the algorithm is performed through experiments based on the interaction with simulated environments of different sizes, routes and features.

Contenido

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

1.1. Planteamiento Del Problema

En el campo de la robótica, existen agentes móviles programados, que suplen las necesidades básicas requeridas como asistentes, sin embargo, estos se ven limitados ante cambios externos. Supóngase un agente para movilizarse en un ambiente cerrado, las decisiones que toma el robot programado están previamente descritas y basadas en un mapeo del escenario, al momento de presentarse una reestructuración o alguna modificación sencilla dentro del escenario, deberá volverse a programar el asistente considerando las nuevas modificaciones, convirtiéndose en una tarea tediosa para el dueño del asistente y para el programador, ya que debe diseñar por sí mismo las rutas alternativas. En el caso de los robots de asistencia personal, se presentan dificultades cuando el dueño quiere movilizarse hacia un espacio que el agente no conoce previamente, lo que conduce a hacer uso de él únicamente en su hogar o aquellos lugares conocidos por el agente.

Como solución a esta problemática, se hace uso de la técnica de aprendizaje por refuerzo, la cual procesa la información capturada con los sensores del robot, sobre el entorno y su ubicación espacial, para definir las acciones que serán realizadas. La ejecución de estas acciones generan una recompensa que busca ser maximizada a largo plazo, sin embargo,

esta se caracteriza por ser escasa, ruidosa y/o demorada[1]. Por otro lado, existe el dilema, exploración vs. explotación, en el cual se debe evaluar si se quiere que el robot explore nuevas zonas del escenario donde podría encontrar mejores recompensas o por el contrario, explotar aquellas opciones que le ofrecen recompensas positivas, todo esto teniendo en cuenta que el robot se desenvuelve en un ambiente dinámico [2].

Luego de exponer estos dilemas, se formula la pregunta de investigación: ¿Cómo enseñarle a un robot móvil a navegar de forma autónoma por medio de aprendizaje por refuerzo en espacios desconocidos a partir de información visual?

1.2. Objetivos

1.2.1. General

Diseñar un algoritmo de navegación autónoma para un agente móvil, basado en información visual mediante el uso de técnicas de aprendizaje por refuerzo.

1.2.2. Específicos

- Definir las variables que darán origen a los estados del sistema considerando que el robot usa información visual para su navegación.
- Diseñar un sistema de estados basado en redes neuronales que reemplace al clásico sistema basado en la matriz Q.
- Establecer políticas de recompensa para generar el proceso de aprendizaje del agente.
- Implementar el sistema propuesto en un ambiente virtual para la simulación de robots reales (Coppelia Sim).
- Evaluar el desempeño del algoritmo diseñado, realizando pruebas en diferentes ambientes de simulación, e implementado de acuerdo a la arquitectura establecida para la generación de los estados.

1.3. Justificación

Las técnicas de aprendizaje de máquina aplicadas a la robótica son de especial importancia considerando que, el secreto utilizado por varias empresas a nivel mundial para expandirse está resumido en dos áreas de investigación, la inteligencia artificial (IA) y el aprendizaje automatizado [3]. Estas ramas de investigación, buscan pronósticos confiables, sistemas que se adapten a sus intereses y aprendan de las necesidades de sus consumidores.

El aprendizaje por refuerzo, al ser un derivado de IA se convierte en un punto destacable e importante para el desarrollo industrial de un país. No es desconocido el hecho de que el emprendimiento en países en desarrollo como Colombia, ha tomado bastante fuerza desde hace algunos años, lo cual se convierte en un desafío para ingenieros e investigadores de estos campos [4].

Actualmente, existen estrategias de navegación que están bien implementadas y son ampliamente usadas como el SLAM [5], una técnica basada en modelos que permite la localización y mapeo de entornos, sin embargo, en ambientes dinámicos y con caracteristicas complejas, su desempeño se ve afectado[6]. Así, el uso de técnicas de aprendizaje como el aprendizaje por refuerzo profundo (por sus siglas en ingles DRL) se convierte en una estrategia efectiva y útil cuando se debe planificar rutas de navegación en entornos desconocidos o variables, como áreas de desastres: Allí es necesario que el agente posea autonomía y tenga la capacidad de aprender para lograr adaptarse.

Adicional a la técnica de aprendizaje, es importante seleccionar datos de entrada que le permitan al robot conocer el estado real de su entorno e identificar patrones de guía o señales para tomar la mejor decisión en cada posición. Por esta razón, en este proyecto se seleccionó una cámara como sensor principal.

Por esto, los robots autónomos y con gran capacidad de adaptabilidad son altamente requeridos para tareas de alto riesgo y/o muy poco deseadas para los seres humanos, tales como búsqueda y rescate de personas en zonas de desastre, cuidado de personas de la tercera edad, inspección y vigilancia en zonas de contaminación.

1.4. Impacto Social

Este proyecto le apunta a diseñar un algoritmo de navegación autónoma para agentes móviles en entornos cerrados, como es el caso de los robots de asistencia, los cuales requieren de un algoritmo capaz de aprender y adaptarse a cualquier entorno señalizado para llegar a un destino específico.

El fomento de la robótica asistencial como una estrategia de mejora social aplicada a los campos de cuidado de mayores, enfermería y sanidad, puede alivianar el impacto ocasionado por la disminución de la tasa de natalidad y el envejecimiento de la población, un ejemplo de ello es Japón, quienes han estado a la vanguardia en temas de robótica desde hace décadas y en el 2015 implementó el plan 'New Robot Strategy'[7] siendo uno de sus objetivos llegar al 2024 con un sistema que reduzca o elimine los riesgos sufridos por los cuidadores debido a la carga de trabajo que tienen al cumplir con las actividades de cuidado de mayores.

Los hombres y mujeres mayores de 65 años poseen autonomía limitada y están expuestos a sufrir distintos problemas asociados como síntomas de depresión, falta de relaciones sociales, entre otros. Para reducir estos problemas de dependencia se plantea el uso de robots de asistencia, que según [8], ofrecen apoyo a personas con discapacidades en actividades como transporte de objetos, limpieza y quehaceres del hogar. De acuerdo con [9], la población de adultos mayores estaría dispuesta a que un manipulador móvil les colabore en dichas actividades con el fin de mantener su independencia.

Actualmente en Colombia hay aproximadamente 5.8 millones personas mayores de 60 años, de las cuales el 14.2 % viven solas (equivalente a 821 mil personas), un dato preocupante al observar que un 18.7 % del total de personas mayores sufre al menos una discapacidad, el 55,8 % reportó tener dificultades para ver de cerca, de lejos o alrededor, el 40.7 % tiene dificultades para mover el cuerpo y/o caminar y el 15.2 % tiene dificultades para agarrar o mover objetos con las manos. Según el DANE, en Colombia se estima que para el 2030 la población mayor será de 9.739.701 personas, es decir, representará el 17,5 % de la población total del país para ese momento.[10]

Capítulo 2

Estado Del arte

Teniendo en cuenta que este proyecto está orientado hacia el uso de aprendizaje por refuerzo o el aprendizaje de máquina, se revisaron diferentes propuestas con este enfoque. En el artículo [1], se presenta un algoritmo de apredizaje de juegos llamado "TD-Gammon" el cual esta basado en una red neuronal que se entrena a partir los resultados obtenidos de jugar el juego de mesa "Backgammon" contra sí mismo. En [1] se propone el uso de TD (Temporal Differrence), una clase de aprendizaje por refuerzo, para predecir acciones dependiendo valores futuros dada una señal. En dicho documento se plantea la dificultad de implementar soluciones con aprendizaje por refuerzo en problemas de gran escala o complejos, entendiendo que estas soluciones se limitan a tener tablas de búsqueda o funciones de evaluación lineales, problemas que son solventados al implementar árboles de decisión o perceptrones multicapa (NN).

Posterior al éxito de los resultados obtenidos con el algoritmo [1], surgieron avances teóricos y prácticos entorno al aprendizaje por refuerzo. Un ejemplo de ello es [11], trabajo de grado presentado en 2018 en la Universidad Santo Tomás con el objetivo de desarrollar y simular un algoritmo que brinde un comportamiento de enjambre en robots, como resultado el autor propone dos soluciones que usan aprendizaje Q. En la primera, el agente genera estados obtenidos a partir de las distacias entre sus vecinos. En la segunda, el agente determina la cantidad de agentes

vecinos cercanos a sus cuadrantes, basado en un radio de repulsión y otro de atracción. Un segundo ejemplo del uso de aprendizaje por refuerzo para navegación autónoma es [12], donde se adicionan tareas auxiliares para predecir la profundidad y la clasificación de cierre de lazo, con el objetivo de mejorar el tratamiento de datos. Todas las tareas utilizan aprendizaje por refuerzo con una memoria de corto y largo plazo, pero las tareas auxiliares son también entrenadas mediante aprendizaje supervisado.

En el artículo [13], los autores ponen a prueba el desempeño de una red DQN (Deep Q-Network), en un ambiente gráfico virtual que simula los juegos de Atari 2600. La recompensa obtenida por el agente en la interacción con el entorno virtual, indica su avance en el juego. La interacción se evalúa a través de la red Q, con una imagen de entrada obtenida del entorno y una acción predecida de salida. El proceso de aprendizaje utiliza el método de Experience Replay, almacenando en memoria N cantidad de transiciones, que serán puestas a prueba aleatoriamente por el algoritmo para actualizar los pesos de la red basados en la tasa de aprendizaje. Los resultados del algoritmo evidencian la eficiencia del uso combinado de aprendizaje profundo y aprendizaje Q, sobrepasando con creces la capacidad de un humano y algoritmos previos.

En este punto, el aprendizaje por refuerzo profundo ha presentado un desempeño favorable y eficiente en tareas específicas. Sin embargo, el artículo [2] expone las limitaciones que presenta esta técnica cuando debe desenvolverse en tareas generales. Como solución, en el documento plantean dividir una tarea general en tareas específicas y aplicar aprendizaje por refuerzo profundo en ellas. Por ejemplo, en Hierarchical RL se implementa aprendizaje por refuerzo profundo para desarrollar multi-timestep “actions” asignándole las acciones primitivas de sub-políticas que permiten conformar una política, logrando descubrir y alcanzar metas, los cuales son estados específicos del entorno. En Multi-agent RL, el enfoque del aprendizaje por refuerzo profundo ha sido permitir la comunicación entre agentes, con el fin de cooperar entre ellos a través de la identificación espacial. Otro reto propuesto está en

Model-base RL, donde se pretende llegar a la navegación de un agente en un entorno desconocido sin la interacción con el entorno real, el agente aprenderá del entorno utilizando uno simulado y para este aprendizaje los algoritmos de DRL aprenderán usando información captada en pixeles.

Por último, este trabajo se desarrollará en colaboración y apoyo del Grupo de Investigación y Desarrollo en Robótica de la Universidad Santo Tomás (GED). El equipo de investigación ha desarrollado y publicado trabajos relacionados con Navegación Autónoma y Visión Artificial, temas afines al enfoque de este proyecto. En el artículo [14] publicado en 2013, se propone un algoritmo de detección de bordes basado en lógica difusa como parte de un sistema de visión local que ayudaría a reconocer marcas visibles en un juego de campo diseñado para la liga humanoide de robots de RoboCup, además en el artículo [15] se propone un algoritmo capaz de detectar el camino por donde transita un UGV (Unmanned Ground Vehicle). Por otro lado, en la rama de navegación autónoma, el grupo GED posee publicaciones como [16], [17], donde se enfocan en la navegación de un sistema multiagente a través de la teoría de enjambre donde los robots son los agentes del enjambre, se establecen fuerzas de repulsión y atracción las cuales son usadas para navegar objetivamente, es decir, evitar obstáculos y mantener el grupo compacto, el objetivo de lograr esto va orientado hacia operaciones de rescate de víctimas, para lo cual establecieron un algoritmo de consenso en el que cada agente por separado identifica una posible víctima, una vez estén de acuerdo los agentes de que determinada víctima existe, una parte los agentes se separa para formar un sub-enjambre que se encargará de adquirir información de la posible víctima y así facilitar la búsqueda y rescate, como también el artículo [18], se propone la navegación en espacios desconocidos para un agente móvil, en el cual, se usan sensores propios del robot en vez de sensores de localización, esto con el fin de implementar la navegación y la evasión de obstáculos.

Capítulo 3

Marco Teórico

Para el entendimiento y ejecución de este proyecto se deben conocer conceptos relacionados con el aprendizaje por refuerzo, como lo son:

3.1. Agente

Se define como agente a la entidad encargada de percibir su entorno mediante sensores y utilizar actuadores para interactuar con él, como se ilustra en la figura 3.2.

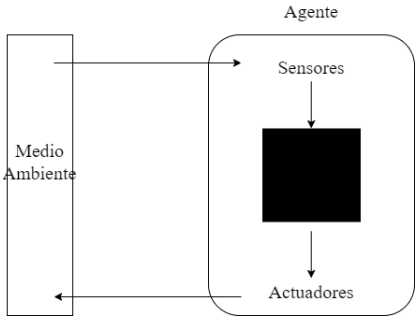


Figura 3.1: Interacción agente y entorno

Un ser vivo posee órganos sensoriales que le ayudan a percibir el mundo y posee extremidades o músculos que lo ayudan a movilizarse, de la misma forma, un agente robot recibe información del entorno mediante sensores que miden matemáticamente las características a su alrededor. Toda esta

información es enviada en paquetes para procesarla y ejecutar una acción con almenos uno de sus actuadores. [3]

3.2. Aprendizaje de máquina

Es un rama de la inteligencia artificial que busca dotar a las máquinas con la capacidad de aprender mediante la generalización de comportamientos y el reconocimiento de diferentes patrones. La experiencia es la acumulación de información que le permitirá a un agente determinar la acción a ejecutar.[19]

3.2.1. Aprendizaje Supervisado

Consiste en el aprendizaje a partir de un conjunto de ejemplos conocidos y etiquetados que son dados por un supervisor externo con el objetivo de extrapolar o generalizar la respuesta frente a la entrada de datos desconocidos con los que no se entrenó.[20]

3.2.2. Aprendizaje no supervisado

Consiste en el aprendizaje dado un conjunto de datos de los cuales no se conoce su clasificación con el objetivo de que el sistema encuentre patrones que le ayuden a etiquetar de manera correcta los datos de entrada. [20]

3.3. Aprendizaje por Refuerzo

Es una rama del aprendizaje de máquina en la cual el agente es el encargado de guiar su aprendizaje basado en las interacciones con el ambiente y la recompensa que recibe por ejecutarlas. Se enfoca en tomar acciones aleatorias que le permitan conocer la forma adecuada de interactuar con el ambiente, replicar las acciones con recompensa positiva y encontrar argumentos para tomar una acción Y en un estado X. En muchos dominios complejos, donde el agente desconoce el entorno y no es viable realizar un mapeo o una discretización, el aprendizaje por refuerzo es un camino efectivo para entrenar un agente, por ejemplo, es un reto

enseñar a un agente a conducir un helicóptero; sin embargo, se puede establecer un sistema de recompensa que castigue cuando se estrelle, oscile o desvíe de la ruta objetivo. [20][3]

Algunos elementos importantes dentro del aprendizaje por refuerzo son:

- Función de recompensa: Es una retroalimentacion numérica generada por el entorno que le indica al agente si la acción tomada fue correcta.
- Política: Es el método que determina el comportamiento del agente en determinado estado, aquí se asignan las acciones que puede tomar frente a cada estado.
- Modelo: Es aquel que realiza las predicciones sobre un conjunto de transiciones. Una transición es la relación establecida entre el estado actual, la acción por ejecutar, la recompensa y el estado al que llegará.
- Función de valor: Esta es una estimación sobre la recompensa al ejecutar cierta acción y determina qué acciones son convenientes para el agente en el futuro.

3.4. Q-learning

Es una aplicación del aprendizaje por refuerzo dado la funcion *Q(s,a)*:

$$Q(s_t, a_t) = max(R_{t+1}) \tag{3.1}$$

De manera que *Q(s,a)* representa máxima recompensa futura que se recibe al tomar una acción a en el estado s, es llamada $Q - function$ porque representa la calidad "Quality" de cierta acción en cierto estado.

La política para la función Q puede ser definida como:

$$\pi(s) = argmax_a Q(s', a') \tag{3.2}$$

Donde π representa la política. Así, la función Q se puede expresar en función del valor Q del siguiente estado s' de la siguiente manera:

$$Q(s, a) = r + \gamma max Q(s', a') \tag{3.3}$$

La ecuación expuesta anteriormente es conocida como la ecuación de Bellman, que indica que la máxima recompensa futura para el estado s y la acción a es la recompensa inmediata sumada con la máxima recompensa futura del estado siguiente. [21] Finalmente, la función de Q-learnig se puede aproximar iterativamente usando la ecuación de Bellman como sigue[20]:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[R_{t+1} + \gamma max Q(s_{t+1}, a) - Q(s_t, a_t)] \qquad (3.4)$$

Donde α es la velocidad de aprendizaje que indica qué tanto se tiene en cuenta el valor Q anterior con respecto al nuevo.

3.5. Deep Q Networks

El concepto teórico de Deep Q Network (DQN) combina el algoritmo de aprendizaje por refuerzo Q-Learning con redes neuronales para abordar entornos con enormes cantidades de estados y acciones, reemplazando la tabla por una red neuronal que aproxima la función Q. El algoritmo hace uso de dos redes neurales para ajustar y actualizar el proceso de aprendizaje; la red A, se utiliza para estimar los valores Q del estado y la acción presente, mientras que la red B, se utiliza para estimar los valores Q del estado y la acción siguiente, evitando que la funcion Q cambie muy rápido y termine retroalimentandose al actualizar sus propios valores Q. En la red A se produce el aprendizaje durante todas las iteraciones, mientras que, la red B actualiza sus parámetros cada N iteraciones, donde se transfiere el aprendizaje de la red A hacia la red B.[13].

3.5.1. Replay Memory

La aproximación de los valores obtenidos por la *función Q* (denomidados valores q) usando funciones no lineales no es estable y tiende a ser un proceso demorado para el aprendizaje, una de la mejores soluciones para este problema es el *experience replay*, una memoria donde se almacenan las transiciones que ejecuta el agente siendo una transicion <s,a,r,s'> compuesta por el estado actual, la acción tomada, la recompensa obtenida y el estado futuro. Cuando se entrena la red, se

toman muestras aleatorias de la memoria y se usan en lugar de la transición actual.[21][22]

3.5.2. Exploración vs. Explotación

La elección de acciones óptimas puede llevar a resultados subóptimos, debido a la limitación de experiencia o falta de conocimiento. Por ello, el agente debe contar con una política que le permita iterar entre explotar para maximizar su recompensa y explorar para maximizar su comportamiento en estados desconocidos a largo plazo. Esto se define mediante una función que le permite al agente explorar lo suficiente en estados desconocidos pero también explotar aquellas acciones que le otorgan recompensas positivas.[20][3]

3.6. Aprendizaje por transferencia

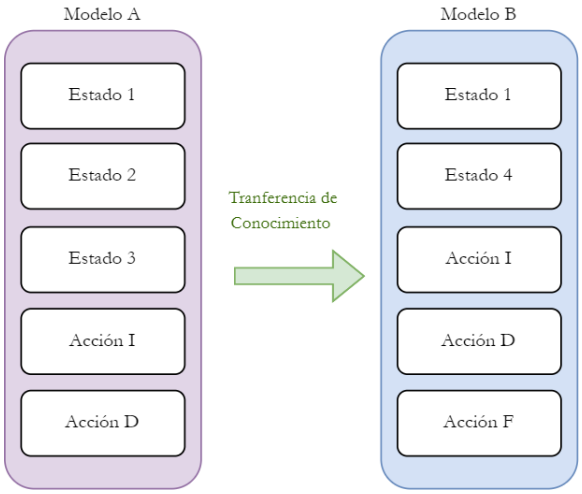


Figura 3.2: Diagrama de aprendizaje por transferencia entre modelos

La técnica de aprendizaje por transferencia busca mejorar el desempeño de las decisiones del modelo A en un ambiente objetivo, transfiriendo el conocimiento adquirido previamente del modelo B en un ambiente primario, cuando el ambiente primario contiene parcialmente estados y acciones del

ambiente objetivo.[23][24]

3.7. Aprendizaje por imitación

La técnica de aprendizaje por imitación se basa en la premisa de un ambiente con un set de estados S, un set de acciones A, un modelo $P(s'|s,a)$, una función de recompensa dada por el mismo ambiente $R(s,a)$ y una trayectoria τ, la cuál es una demostración de un experto, donde las acciones que toma están basadas en una política óptima, con el fin de que el agente entrene con buenas muestras en vez de muestras aleatorias erróneas permitiendo que la eficiencia de aprendizaje mejore sustancialmente. [25][24]

3.8. Red Neuronal

Una red neuronal es un método que permite a una computadora procesar datos, utiliza neuronas interconectadas en una estructura de capas que se asemeja al cerebro humano. Consta de una capa de entrada, una capa de salida y al menos dos capas ocultas, ubicadas entre la entrada y la salida.

- La capa de entrada recibe información del ambiente y se encarga de procesarla, analizarla, clasificarla y entregarla a la siguiente etapa, por medio de nodos.
- Cada capa oculta en la red analiza la salida de la capa anterior, la procesa aún más y la entrega a la siguiente capa.
- La capa de salida entrega el resultado final de todo el procesamiento de datos que se realiza en la red neuronal.

Para el realizar procesamiento, análisis y clasificación de información, la red utiliza el proceso de convolución, en el cual se define una matriz (kernel) que recorre la imagen y la opera matemáticamente, destacando cuantitativamente características en cada paso. Además de la función de filtrado, la convolución permite extraer características propias de la imagen comprimiendola para reducir su tamaño inicial, a través de métodos de

muestreo como el $Max - Pooling$. Finalmente se hace uso del proceso de Flatten el cual transforma la salida de una capa convolucional que es multidimensional a linear para poder pasarla a una capa dense la cual toma sus entradas de manera linear y empieza a asociar valores y características comunes.[26]

Capítulo 4

Diseño Metodológico

En este capítulo se especificarán las etapas del diseño metodológico planteado para la elaboración del presente proyecto, resumidas en la Fig.4.1.

4.1. Revisión bibliográfica

Se realiza un entendimiento, estudio y clasificación de diferentes documentos (usados como material bibliográfico) con temas afines al proyecto contextualizando la problemática actual, verificando y examinando avances presentados con anterioridad. Basados en la información recolectada se realiza la redacción del actual documento:

1. **Definición y planteamiento del problema.** Se expuso el asunto que esta investigación tiene como objeto aclarar, exponiendo el contexto global y finalizando con la pregunta que orienta el curso de la implementación.

2. **Redacción de justificación.** Se plantean los argumentos que sustentan el desarrollo del proyecto.

3. **Búsqueda y definición del impacto social.** Se presenta el pilar ético de la investigación, exponiendo la población que se verá beneficiada con el desarrollo del proyecto.

4. **Definición de objetivos.** Se determina un objetivo general, definido como la meta y el propósito del proyecto y se trazan los objetivos específicos a seguir durante el proyecto, que contribuyen a la realización del objetivo general.

4.2. Desarrollo práctico.

Posterior a las definiciones de la etapa anterior, se implementa el algoritmo, se definen los escenarios de prueba para evaluar el desempeño, se realizan pruebas y experimentos de funcionamiento en un entorno simulado. El detalle de cada actividad realizada para terminar esta etapa se detallan a continuación:

1. **Diseño del algoritmo.** Esta tarea tiene como objetivo determinar la arquitectura de la red, la entrada, recompensa, políticas y acciones del algoritmo $DeepQ - learning$ para el aprendizaje del agente.

2. **Simulación del algoritmo diseñado.** Se realiza la definición del robot y el sensor que se va a utilizar como agente durante la simulación del algoritmo. La construcción de los mapas de entrenamiento y evaluación, con sus respectivas características. Toda la simulación será implementada utilizando el software Coppelia Sim.

3. **Ejecución y pruebas de simulación.** Se simula el algoritmo en los mapas creados para analizar su proceso de aprendizaje.

4.3. Verificación de resultados.

En esta etapa se analizan y evaluan los resultados obtenidos en cada simulación, realizando las mejoras identificadas sobre el algoritmo hasta obtener el desempeño deseado.

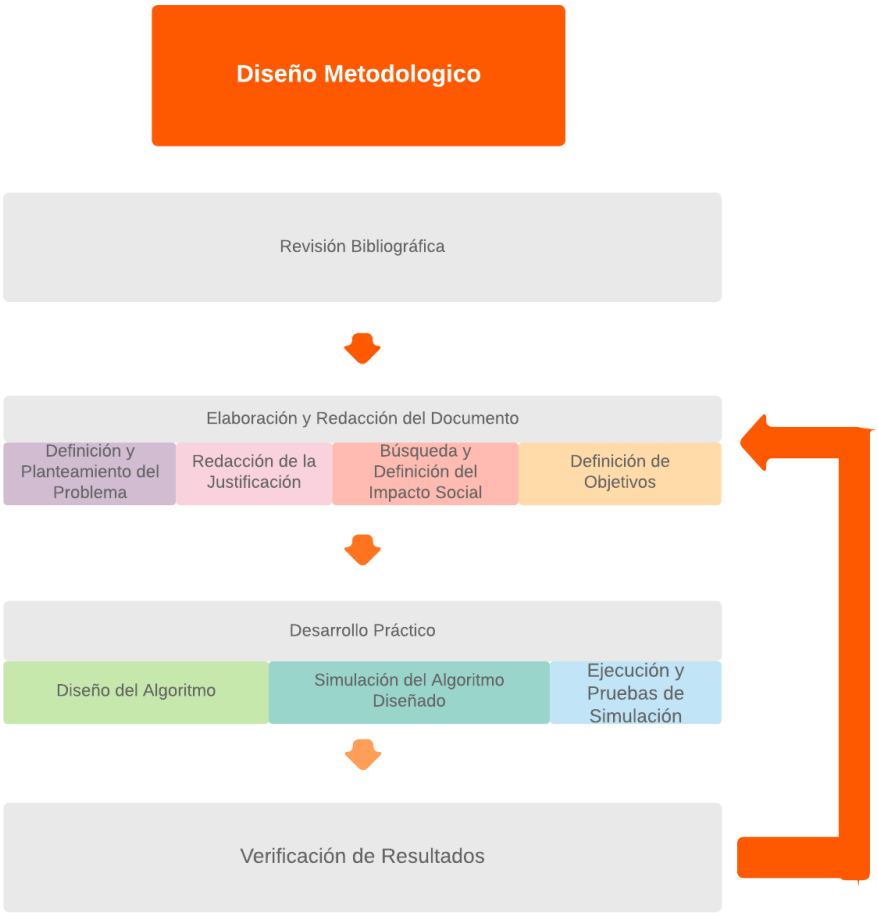


Figura 4.1: Diagrama de metodología

Capítulo 5

Desarrollo Conceptual

Para resolver el problema planteado se propuso crear un algoritmo basado en la definición conceptual del DQN, en el cual, el agente recibe una recompensa por cada acción que toma 5.1. La relación entre el estado actual, la acción, recompensa y estado siguiente, se almacena en una memoria para posteriormente entrenar la red utilizando la función 5.1. Adicionalmente, se propusieron tres (3) etapas de aprendizaje dentro del entrenamiento, utilizando los métodos de transferencia e imitación definidos en el *Capítulo 3*, con el objetivo de potenciar y acelerar el proceso de aprendizaje. Los componentes principales del algoritmo se discuten y profundizan a continuación:

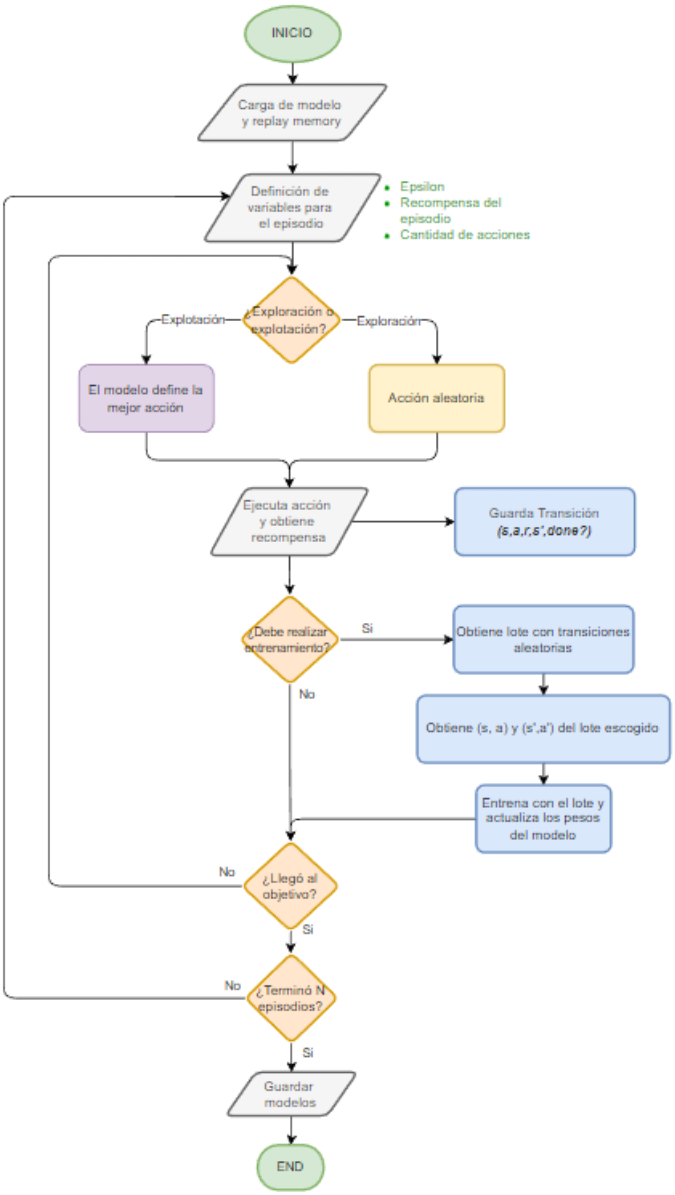


Figura 5.1: Algoritmo de entrenamiento

5.1. Modelo de Red Neuronal

Para el modelo de red neuronal se contempla una entrada de tamaño (64x64x3), correspondiente a la resolución de la imagen capturada por el sensor en el agente, y una salida de tamaño 3, debido a la cantidad de acciones que realiza el agente. La arquitectura, se compone de tres (3) capas convolucionales, cada una con subcapa de $Max - Ppooling$, y una serie de cinco (5) capas ocultas.

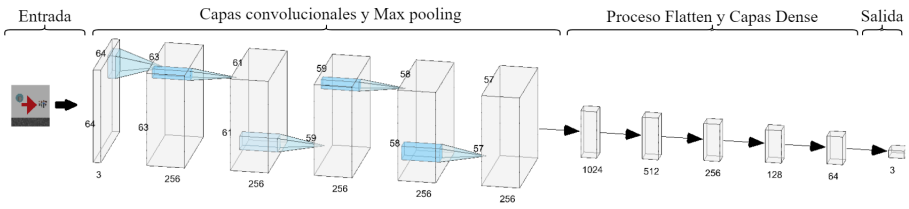


Figura 5.2: Visualización del modelo de la red neuronal

5.2. Estados

Los estados del sistema están definidos como las imágenes obtenidas por el sensor del agente después de ejecutar una acción, como se muestra en la figura 5.3

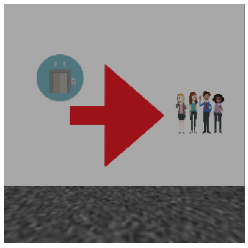


Figura 5.3: Estado del agente en el entorno

5.3. Acciones

El set de acciones definidos para el agente se compone tres (3) acciones discretas:

- l: Movimiento rotacional de noventa (90) grados en dirección izquierda
- r: Movimiento rotacional de noventa (90) grados en dirección derecha
- f: Movimiento de 1,4 metros con velocidad constante hacia el frente

Estos movimientos le permiten al robot explorar, conocer y navegar el entorno. Se excluye el movimiento en reversa considerando que el ángulo de apertura del sensor no supera los 80 grados y solo se tiene información de lo que está en frente.

5.4. Etapas de Aprendizaje

Las etapas de aprendizaje propuestas son:

- Clasificación: En esta etapa se busca que el modelo clasifique imágenes de un conjunto de datos construido en tres categorías: izquierda (left), derecha (right), al frente (forward).
- Imitación: En esta etapa se busca que el agente aprenda a relacionar la información de entrada con la acción que lo acercará al objetivo. Para ello, se añadió un replay memory con la trayectoria ideal para resolver el mapa, se virtualizó la simulación y se planteó utilizar el aprendizaje por imitación, recompensando positivamente al agente solo cuando sigue la ruta ideal.
- Entrenamiento: En esta etapa se busca brindar al agente todas las herramientas necesarias para resolver cualquier mapa que contenga información visual descriptiva. Aquí, el agente interactúa directamente con el simulador, donde tiene total libertad en sus movimientos y será recompensado por cada acción correcta que tome. Adicionalmente, en esta etapa, se incluye una definición más para el agente que le enseñará a distinguir información visual poco descriptiva y girar en busca de información que le acerque al objetivo.

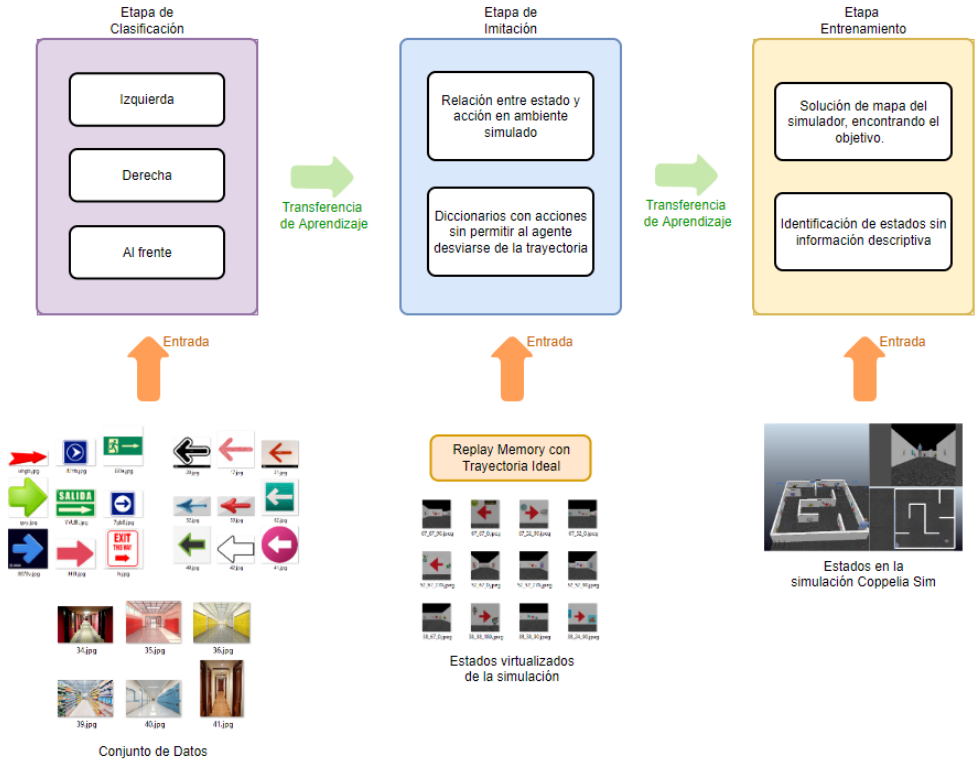


Figura 5.4: Diagrama de relación entre las etapas de aprendizaje

5.5. Política de Recompensa

Con el objetivo de recompensar al agente cuando reconoce y sigue la instrucción de una señal en el entorno, o interpreta el estado en el que se encuentra y toma la mejor decisión para buscar alguna señal, se propuso una recompensa que oscila entre 1 y -1, evaluando parámetros estáticos y dinámicos del episodio. Se asignó un porcentaje de peso para cada parámetro evaluado (en el mejor escenario suman 100 %) como se muestra en la ecuación:

$$R = (Ae * 0{,}9) + (Co * 0{,}3) - (Lp * 0{,}2) \tag{5.1}$$

El parámetro Lp (Living Penalty) es un castigo por cada acción tomada para que el agente busque disminuir la cantidad de acciones tomadas a

largo plazo, Ae es la evaluación de la acción tomada, si la acción tomada está dentro de la trayecoria ideal será positiva de lo contrario es negativa, y por ultimo Co es una recompensa dinámica que, irá incrementando a medida que el agente se encuentre más cerca del objetivo.

5.6. Política de Exploración

Esta política le da la opción al agente de tomar la mejor acción posible encontrada hasta el momento o explorar una acción aleatoria que le permitirá encontrar nuevos estados con recompensas diferentes. Teniendo en cuenta las fases de aprendizaje mencionadas anteriormente, se propuso un balance entre la explotación del conocimiento transferido y la toma de acciones aleatorias para exploración, definiendo un valor inicial del 50 % que va decrementando linealmente en cada episodio, hasta llegar a 0. Lo anterior se expresa con la ecuación:

$$\epsilon = \frac{max_e - e}{max_e} * max_\epsilon \tag{5.2}$$

Donde el cambio de la política de exploración ϵ está definido por la máxima cantidad de episodios max_e, el episodio actual e y el valor máximo máx$_\epsilon$. Adicionalmente se realiza una evaluación del desempeño del agente cada N episodios colocando el ϵ en cero (0), es decir, el agente realizará solo explotación de las mejores acciones para cada estado.

5.7. Agente

Para este desarrollo se propuso como agente el robot Pioneer 3-DX, el cual posee una tracción diferencial. Como sensor de entrada se añadió una cámara en la parte frontal - superior del robot.

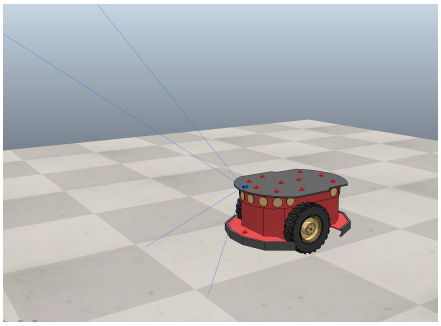


Figura 5.5: Pioneer 3-DX

5.8. Entorno de Simulación

Para la simulación del agente en un entorno desconocido se hizo uso del software Coppelia Sim, en el cual se diseñaron tres (3) mapas de entrenamiento y test. Todos son espacios cerrados, provistos de información visual descriptiva (como señales) que indican la ruta hacia el objetivo, información visual no descriptiva donde existen imágenes u objetos que no se relacionan con la ruta hacia el objetivo y pasillos por los cuales el agente puede avanzar libremente.

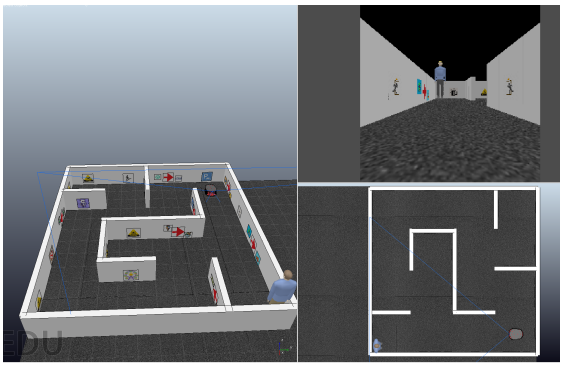


Figura 5.6: Visualización de entorno de simulación para entrenamiento

Los estados del agente son interpretados y las acciones ejecutadas mediante dos scripts escritos en *python*. El primer script, controla la interacción del robot con el entorno con métodos como:

- Obtención de imágenes
- Definición de política de recompensa
- Ejecución de acciones

El segundo script, se encarga del proceso de aprendizaje con métodos como:

- Entenamiento de red, basado en DQN
- Definición de política de exploración
- Almacenamiento de *Replay Memory*

La implementación de esta solución se puede encontrar en el repositorio: `https://github.com/DanielAponte/DRL-AutonomousNavigation-Daniel-Erika`

Capítulo 6

Experimentos

El algoritmo de navegación autónoma con información visual se evalúa realizando tres experimentos donde el agente debe encontrar la ruta al objetivo de cada escenario iniciando siempre desde una posición diferente. La evaluación define un máximo de tiempo igual a 180 segundos y un máximo de acciones igual a 60, si el agente supera estos límites la prueba se determinará como fallida.

Las tablas 6.1,6.2,6.3 muestran la información de cada episodio en los experimentos. En la sección *Información del episodio*, el campo *Duración* corresponde al tiempo, en segundos, que le toma al agente llegar al objetivo desde el punto de inicio; el campo *No. de Acciones*, corresponde a la cantidad de acciones que tomó el agente para resolver el mapa. Adicionalmente, en la sección *Resultado Esperado*, el campo *No. de Acciones* se muestra el número de acciones de la mejor ruta posible para llegar al objetivo en cada episodio. Finalmente, el campo *Porcentaje de desviación* compara el número de acciones tomadas por el agente, con el número de acciones de la mejor ruta y representa en porcentaje, la desviación entre los campos.

6.1. Experimento 1

El primer experimento se realiza en un mapa de 16 posibles posiciones, con dos (2) rutas que llegan al objetivo. Se ejecutan doce (12) episodios con posiciones de partida diferentes, destacadas en la imagen 6.1, todas con el objetivo final en la misma posición.

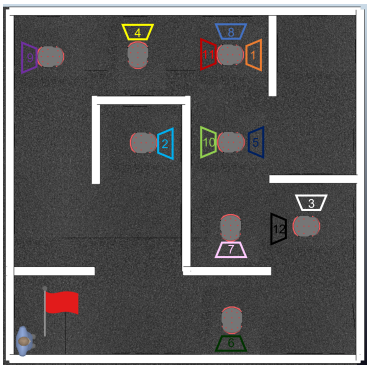
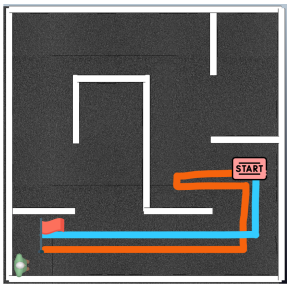


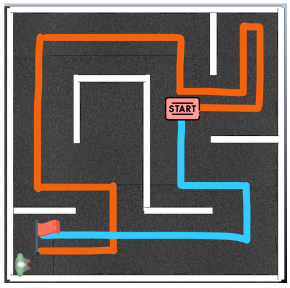
Figura 6.1: Posiciones inciales del robot en el mapa

Episodio	Información del Episodio		Resultado Esperado	Porcentaje de desviación
	Duración [s]	No. Acciones	No. Acciones	
1	27.1	13	11	18.2 %
2	11.8	7	5	40 %
3	22.8	11	7	57.1 %
4	22.6	11	11	0 %
5	38.6	18	10	80 %
6	30.3	15	3	400 %
7	16.2	8	8	0 %
8	24.8	12	12	0 %
9	17.8	9	9	0 %
10	20.1	11	10	10 %
11	23.4	11	11	0 %
12	20.3	10	6	66.7 %

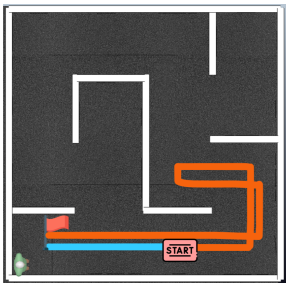
Tabla 6.1: Resultados primer experimento



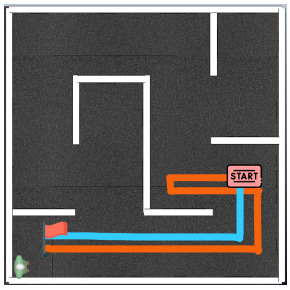
(a) Episodio 3



(b) Episodio 5



(c) Episodio 6



(d) Episodio 12

Figura 6.2: Rutas de episodios con desviación alta

6.2. Experimento 2

El primer experimento se realiza en un mapa de 36 posiciones, con múltiples rutas que llegan al objetivo. Se ejecutan diez (10) episodios con posiciones de partida diferentes, destacadas en la imagen 6.3, todas con el objetivo final en la misma posición.

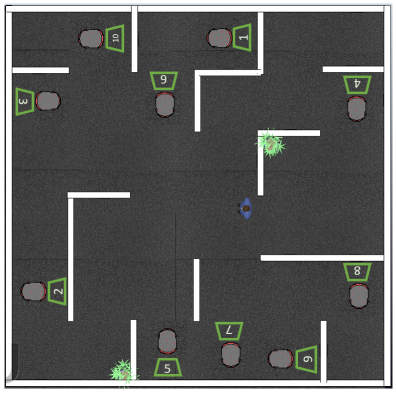
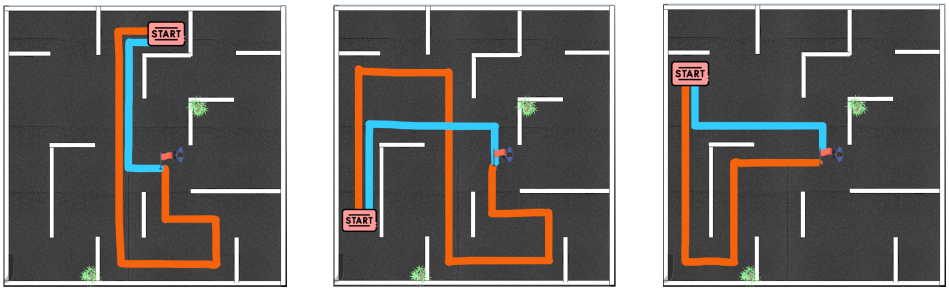


Figura 6.3: Posiciones inciales del robot en el mapa

Episodio	Información del Episodio		Resultado Esperado	Porcentaje de desviación
	Duración [s]	No. Acciones	No. Acciones	
1	62	19	9	111.1 %
2	75.1	22	9	144.4 %
3	47.1	14	8	75 %
4	22.3	7	7	0 %
5	30.5	10	6	66.7 %
6	59.6	19	6	216.6 %
7	8.5	3	3	0 %
8	16.9	6	6	0 %
9	19.5	6	6	0 %
10	22.0	8	8	0 %

Tabla 6.2: Resultados segundo experimento



(a) **Episodio 1** (b) **Episodio 2** (c) **Episodio 3**

(d) **Episodio 5** (e) **Episodio 6**

Figura 6.4: Rutas de episodios con desviación alta

6.3. Experimento 3

El primer experimento se realiza en un mapa de 81 posiciones, con múltiples rutas que llegan al objetivo. Se ejecutan siete (7) episodios con posiciones de partida diferentes, destacadas en la imagen 6.3, todas con el objetivo final en la misma posición.

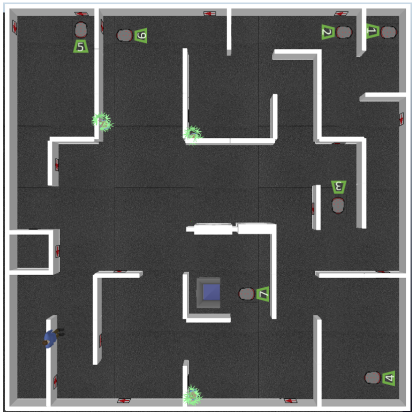
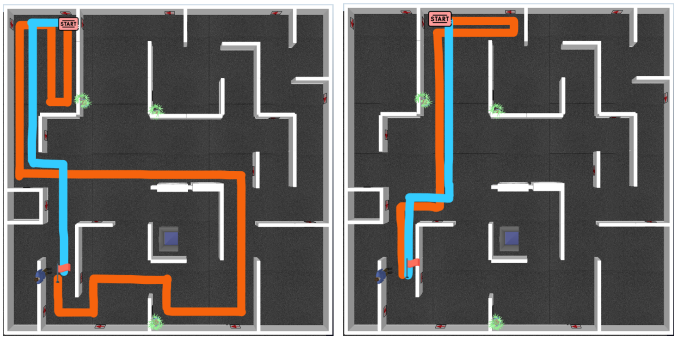


Figura 6.5: Posiciones inciales del robot en el mapa

Episodio	Información del Episodio		Resultado Esperado	Porcentaje de desviación
	Duración [s]	No. Acciones	No. Acciones	
1	102	31	31	0 %
2	77.4	23	23	0 %
3	47.9	13	13	0 %
4	73.2	22	22	0 %
5	100.4	36	12	200 %
6	67.3	19	12	58.3 %
7	32.7	12	12	0 %

Tabla 6.3: Resultados tercer experimento



(a) Episodio 5 **(b) Episodio 6**

Figura 6.6: Rutas de episodios con desviación alta

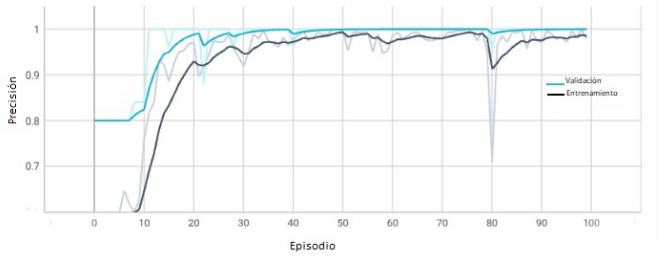
Capítulo 7

Resultados

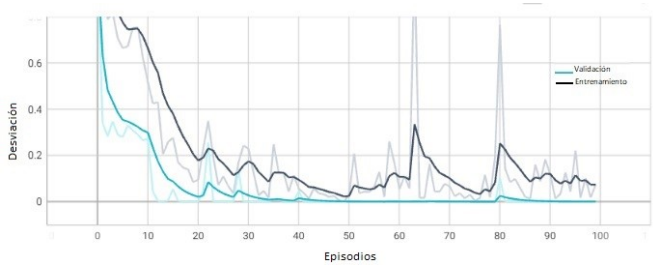
En este apartado se presentan los resultados del proceso de aprendizaje propuesto en el capítulo 5 y los experimentos planteados en el capítulo 6. El proceso de aprendizaje se realizó sobre el mapa del primer experimento.

7.1. Etapa de Clasificación

Se obtuvieron y graficaron los parámetros *accuracy* (determina la precisión de la red cuando clasifica una imagen del conjunto de datos) y *loss* (determina la desviación en la respuesta de la red). En la gráfica 7.1 se observa como el modelo aumenta progresivamente la precisión mientras que disminuye la desviación de sus decisiones, un indicativo de que cada vez sus clasificaciones son más acertadas. Los picos aislados que se presentan en las gráficas de desempeño, se pueden atribuir a imágenes particulares que la red encontró confusas o poco descriptivas.



(a) Precisión



(b) Desviación

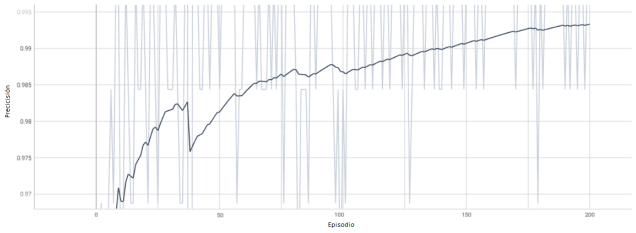
Figura 7.1: Gráficas de desempeño durante la etapa de clasificación

7.2. Etapa de Imitación

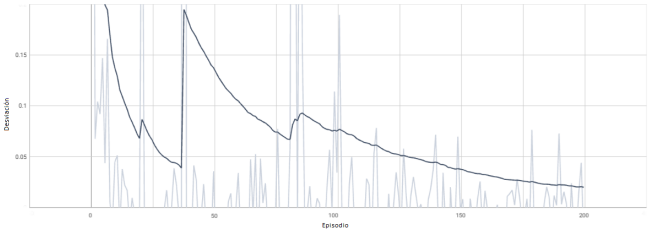
El desempeño del modelo en esta etapa tuvo un comportamiento positivo y similar a la etapa anterior 7.2. Sin embargo, aquí podemos analizar la influencia de las técnicas de aprendizaje por transferencia y aprendizaje por imitación aplicadas.

Según la tendencia de la gráfica de precisión 7.2a, podemos concluir que la red tuvo la capacidad de explotar el conocimiento adquirido en la etapa anterior, utilizando la trayectoria ideal asignada en el *replay memory*.
En la misma gráfica, se observa que la red inició con un porcentaje de precisión sobre el 90 %, atribuido al conocimiento transferido de la red de la etapa anterior.

Los picos en la desviación y la precisión se deben a las acciones aletorias que toma el agente cuando está explorando, un comportamiento esperado y de utilidad en el proceso de aprendizaje, para que el agente maximice la



(a) Precisión



(b) Desviación

Figura 7.2: Gráficas de desempeño durante la etapa de imitación

calidad (*Quality*) de sus acciones.

Adicionalmente, en esta etapa se graficaron otros parámetros obtenidos en las interacciones del agente con el entorno simulado:

La línea de tendencia en la gráfica 7.3 evidencia la intención del algoritmo por maximizar la recompensa individual de cada paso.

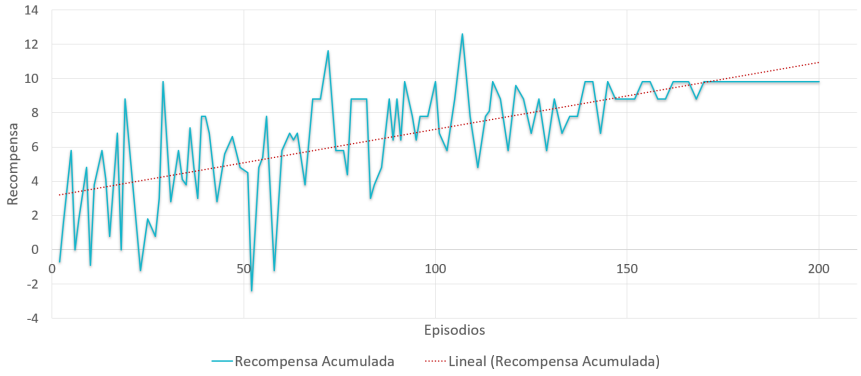


Figura 7.3: Recompensa Acumulada vs. Episodios durante la etapa de imitación

La disminución de acciones tomadas por episodio en la gráfica 7.4 evidencia el logro del algoritmo, al minimizar la cantidad de acciones erróneas que toma el agente para encontrar el objetivo.

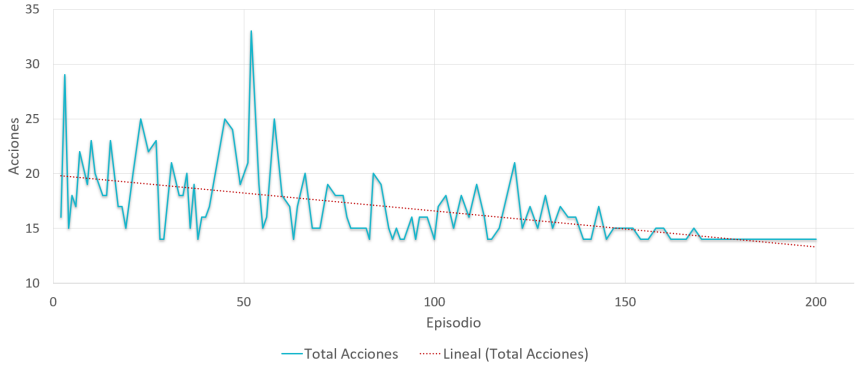


Figura 7.4: Numero de acciones tomadas por episodio durante la etapa de imitación

Las acciones erróneas son definidas como los eventos en los que el agente, usando la predicción de la red, no sigue la instrucción de la flecha o no interpreta la información visual para avanzar al frente. En la gráfica 7.5 se observa como la cantidad de acciones erróneas disminuye a medida que la explotación aumenta, de lo cual podemos inferir que el algoritmo le permite a la red identificar también las acciones que no tienen buena recompensa.

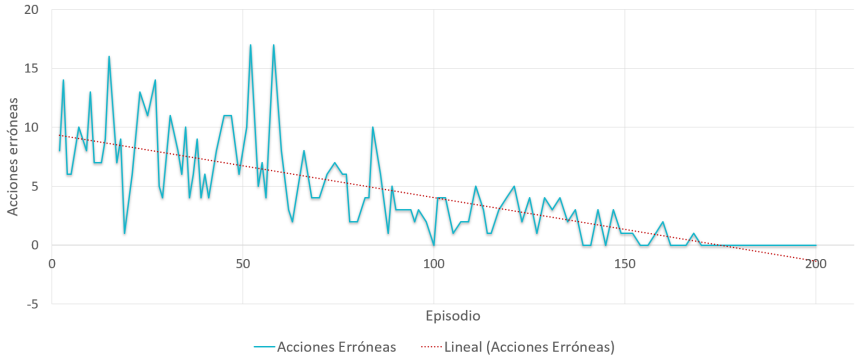
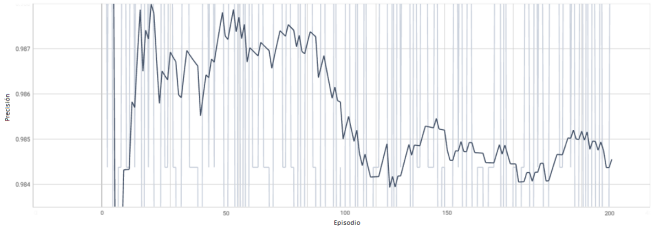


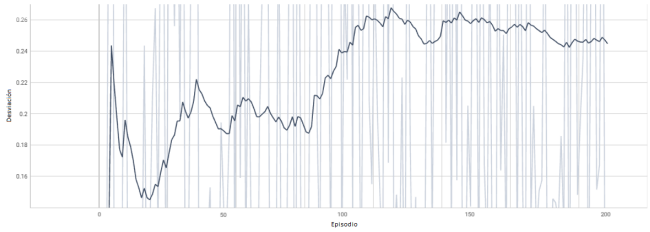
Figura 7.5: Total de acciones erróneas tomadas por episodio durante la etapa de imitación

7.3. Etapa de Entrenamiento

Los parámetros de precisión y desviación 7.6 tienen un comportamiento contrario a lo evidenciado en las gráficas 7.1, 7.2. La precisión en la acción que toma el agente inicia en un punto muy alto (casi 1) y va disminuyendo con cada episodio, un indicativo claro del efecto que tiene la exploración. Sin embargo, la precisión no decae totalmente, llega a un punto de estabilidad que le permite al agente seguir navegando de forma correcta mientras explora rutas desconocidas y se encuentra con información visual poco descriptiva.



(a) Precisión



(b) Desviación

Figura 7.6: Gráficas de desempeño durante la etapa de imitación

Para esta etapa, la gráfica de recompensa acumulada por episodio 7.7 tiene una línea de tendencia positiva pero menos determinante, comparada con la etapa de imitación.

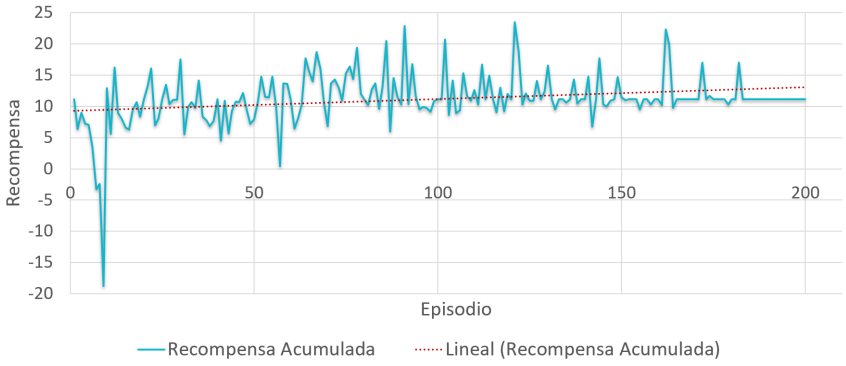


Figura 7.7: Recompensa acumulada por episodio durante la etapa de entrenamiento

Como característica de esta etapa, el agente puede perderse, explorar rutas desconocidas y encontrar en estas nuevas rutas la información para llegar al objetivo. El impacto de esto, se ve reflejado en las fluctuaciones en la cantidad de acciones tomadas por episodio son bastante altas debido al aumento de información que el agente obtuvo del entorno.

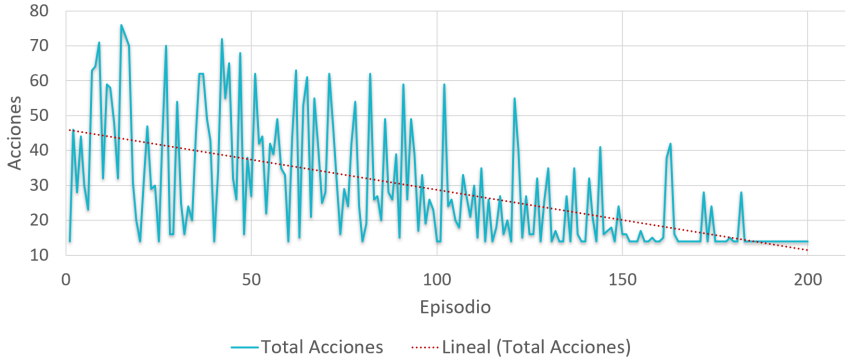


Figura 7.8: Total de acciones tomadas por episodio durante la etapa de entrenamiento

En comparación con la gráfica 7.5 de la etapa anterior, la cantidad de acciones erróneas tomadas por la red son mucho mayores y le toma más tiempo eliminarlas, sin embargo, en esta etapa el agente también logra llegar a un punto de estabilización.

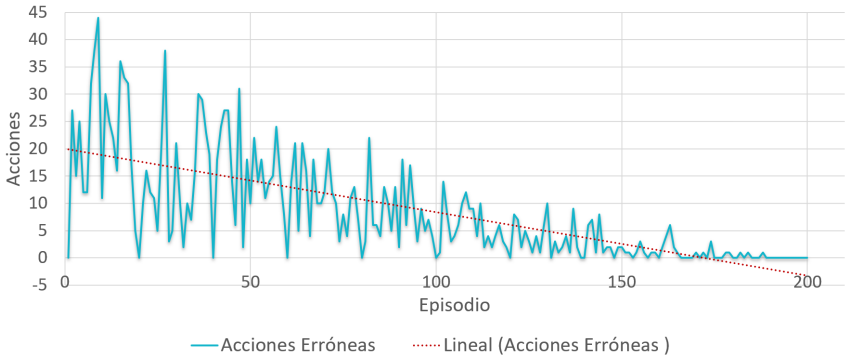


Figura 7.9: Total de acciones erróneas tomadas por episodio durante la etapa de entrenamiento

7.4. Experimentos

Poniendo a prueba el sistema en los tres (3) mapas planteados 6.16.36.5, el porcentaje de éxito en la solución en los mapas es del 100 %, es decir, para los 29 casos de prueba el agente logró encontrar el objetivo dentro de los estándares de evaluación definidos.

El 62 % de los casos evaluados posee alta efectividad para encontrar el objetivo, analizando algunos episodios del 38 % restante, se pudo observar:

- En el mapa 1, el episodio 6 tiene la trayectoría más corta de las doce (12) que se plantearon, y también el mayor porcentaje de desviación, esto se debe a que el agente inició en una posición con información no descriptiva y las decisiones que tomó a partir de ella tampoco le brindaban información de la ubicación del objetivo, por esto tuvo que recorrer gran parte del mapa hasta encontrar las señales que le indicaran el camino hacia el objetivo.

- En el mapa 2, el episodio 2 tuvo el segundo peor desempeño, lo cual se debe a una característica adquirida por el agente, en la que solo va a girar cuando encuentre un obstáculo. Esto quiere decir que en un escenario sin obstáculos, el agente va a navegar de extremo a extremo, como ocurre también en el episodio 5 del mapa 3.

Capítulo 8

Conclusiones y Trabajos Futuros

Basados en los resultados expuestos en el capítulo anterior, se evidencia que el agente interpreta la información visual para buscar señales de la ruta hacia el objetivo, generando así, una relacion entre la eficiencia de las decisiones tomadas por el agente y la información visual que encuentra en su trayectoria; las decisiones serán más acertadas a medida que la información sea más descriptiva.

En conclusión, el algoritmo diseñado en este proyecto le permite a un agente móvil navegar de forma autónoma basado únicamente en información visual. Adicionalmente, se destacan dos contribuciones diferentes: La inclusión de etapas de aprendizaje que le permiten al modelo converger de manera efectiva y eficiente. Y por las características del algoritmo implementado, un sistema de navegación con memoria a corto y largo plazo para mejorar la toma de decisiones en un entorno nuevo.

Por otra parte, se propone como mejora la inclusión de sensores que le proporcionen al agente un mayor campo de visión para diversificar las decisiones tomadas frente a los nuevos estados. El los trabajos futuros se contempla ampliar el sistema a una implementación con un modelo real que permita ayudar a la población mencionada en el capítulo de Impacto Social.

Bibliografía

[1] G. Tesauro, "Temporal difference learning and td-gammon," *Commun. ACM*, vol. 38, no. 3, p. 58–68, mar 1995. [Online]. Available: https://doi.org/10.1145/203330.203343

[2] K. Arulkumaran, M. Deisenroth, M. Brundage, and A. Bharath, "A brief survey of deep reinforcement learning," *IEEE Signal Processing Magazine*, vol. 34, 08 2017.

[3] S. J. Russell, P. Norvig, M. C. R. Juan, and J. L. Aguilar. Pearson Educacion, 2011.

[4]

[5] J. Zhong, C. Ling, A. Cangelosi, A. Lotfi, and X. Liu, "On the gap between domestic robotic applications and computational intelligence," *Electronics*, vol. 10, no. 7, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/7/793

[6] F. Zeng, C. Wang, and S. Ge, "A survey on visual navigation for artificial agents with deep reinforcement learning," *IEEE Access*, vol. PP, 07 2020.

[7] G. o. J. METI, "Japan's new robot strategy," p. 6, 04 2018.

[8] M. D. G. V. A. L. P. E. N. G. C. S. F. N. T. J. CRISTINA URDIALES GARCÍA, JUAN ANTONIO FERNÁNDEZ BERNAT, "https://sd2.ugr.es/wp-content/uploads/2019/10/losrobotsparaelcuidadodelosmayores.pdf," p. 13, 2017.

[9] C.-A. Smarr, T. Mitzner, J. Beer, A. Prakash, T. Chen, C. Kemp, and W. Rogers, "Domestic robots for older adults: Attitudes, preferences, and potential," *International journal of social robotics*, vol. 6, pp. 229–247, 04 2014.

[10] DANE', "Personas mayores en colombia." [Online]. Available: https://www.dane.gov.co/files/investigaciones/notas-estadisticas/nov-2021-nota-estadistica-personas-mayores-en-colombia.pdf

[11] W. Quesada, "GeneraciÓn de comportamientos de enjambre en robots mÓviles a travÉs del uso del aprendizaje por refuerzo." 03 2019.

[12] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell, "Learning to navigate in complex environments," 11 2016.

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 12 2013.

[14] A. Perez, A. Gomez Garcia, E. Rojas-Martínez, C. Rodríguez-Rojas, J. Lopez-Jimenez, and J. Calderon, "Edge detection algorithm based on fuzzy logic theory for a local vision system of robocup humanoid league," *Tecno Lógicas*, vol. 30, pp. 33–50, 06 2013.

[15] J. Calderon, A. Obando, and D. Jaimes, "Road detection algorithm for an autonomous ugv based on monocular vision," in *Proceedings of the Electronics, Robotics and Automotive Mechanics Conference*, ser. CERMA '07. USA: IEEE Computer Society, 2007, p. 253–259.

[16] G. Cardona and J. Calderon, "Robot swarm navigation and victim detection using rendezvous consensus in search and rescue operations," *Applied Sciences*, vol. 9, p. 1702, 04 2019.

[17] J. Leon Leon, G. Cardona, A. Botello, and J. Calderon, "Robot swarms theory applicable to seek and rescue operation," 12 2016.

[18] G. A. Cardona, C. Bravo, W. Quesada, D. Ruiz, M. Obeng, X. Wu, and J. M. Calderon, "Autonomous navigation for exploration of

unknown environments and collision avoidance in mobile robots using reinforcement learning," in *2019 SoutheastCon*, 2019, pp. 1–7.

[19] F. S. Caparrini and W. W. Work, "Introducción al aprendizaje automático." [Online]. Available: http://www.cs.us.es/~fsancho/?e=75

[20] R. S. Sutton, F. Bach, and A. G. Barto, *1.* MIT Press Ltd, 2018.

[21] T. Matiisen, "Demystifying deep reinforcement learning," Dec 2015. [Online]. Available: https://neuro.cs.ut.ee/demystifying-deep-reinforcement-learning/

[22] M. Vallejo del Moral, 2021. [Online]. Available: https://academica-e.unavarra.es/bitstream/handle/2454/40521/TFG_Mikel_Vallejo.pdf?sequence=1&isAllowed=y

[23] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. PP, pp. 1–34, 07 2020.

[24] J. Hua, L. Zeng, G. Li, and Z. Ju, "Learning for a robot: Deep reinforcement learning, imitation learning, transfer learning," *Sensors*, vol. 21, no. 4, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/4/1278

[25] Z. Lőrincz, "A brief overview of imitation learning," Sep 2019. [Online]. Available: https://smartlabai.medium.com/a-brief-overview-of-imitation-learning-8a8a75c44a9c

[26] M. Lahtela and P. P. Kaplan, "¿qué es una red neuronal?" 1966. [Online]. Available: https://aws.amazon.com/es/what-is/neural-network/