

## REPORTE TÉCNICO: NAVEGACIÓN AUTÓNOMA CON DQN



**Título del Proyecto:** Navegación de TurtleBot 3 mediante Aprendizaje por Refuerzo Profundo (DQN)

**Autores:**

- Alejandro Miranda
- Adrian Fuentes
- Luz Soria
- Paolo Quisbert

## 1. Diseño del Sistema

Para la resolución del problema de navegación autónoma en un entorno simulado con obstáculos estáticos, se ha implementado un agente basado en el algoritmo Deep Q-Network (DQN). El problema se ha modelado como un Proceso de Decisión de Markov (MDP) con las siguientes características:

### 1.1. Espacio de Estados (State Space)

El estado  $S_t$  se define como un vector de 14 dimensiones que sintetiza la percepción del robot y su relación con el objetivo:

- Lecturas LiDAR (12 valores): Los 360 grados del sensor láser se discretizan en 12 sectores de 30 grados cada uno. Se toma la distancia mínima detectada en cada sector y se normaliza entre  $[0, 1]$  respecto al rango máximo del sensor (3.5m).
- Distancia al Objetivo (1 valor): Distancia euclidiana normalizada entre la posición actual del robot y las coordenadas (x,y) del objetivo.
- Ángulo al Objetivo (1 valor): Diferencia angular (heading) entre la orientación del robot y el objetivo, normalizada entre  $[-1, 1]$ .

### 1.2. Espacio de Acciones (Action Space)

Para mitigar el problema de mínimos locales (donde el robot gira sobre su eje sin avanzar), se diseñó un espacio de acciones discreto y simplificado de 3 acciones, priorizando el desplazamiento:

1. Adelante: Velocidad lineal 0.20 m/s, angular 0.0 rad/s.
2. Izquierda + Avance: Velocidad lineal 0.15 m/s, angular 0.5 rad/s.
3. Derecha + Avance: Velocidad lineal 0.15 m/s, angular -0.5 rad/s.

*Nota de Diseño:* Se eliminaron las acciones de rotación pura (velocidad lineal 0) para forzar al agente a explorar el mapa y evitar comportamientos estáticos ("camping").

### 1.3. Función de Recompensa (Reward Shaping)

Se implementó una función de recompensa densa basada en el progreso de distancia ("Hunger for Motion") para acelerar la convergencia:

$$R_t = (D_t - 1 - D_{t-1}) \times 100$$

Donde  $D_t$  es la distancia actual al objetivo y  $D_{t-1}$  es la distancia en el paso anterior.

- Si  $R_t > 0$ : El robot se ha acercado (Recompensa positiva proporcional al avance).
- Si  $R_t \leq 0$ : El robot se ha alejado o detenido (Se aplica un castigo fijo de -5).

#### Eventos Terminales:

- Colisión (Distancia  $< 0.22\text{m}$ ):  $R = -100$ .
- Meta Alcanzada (Distancia  $< 0.50\text{m}$ ):  $R = +200$ .

## 2. Hiper Parámetros

La red neuronal utilizada para aproximar la función  $Q(s,a)$  es un Perceptrón Multicapa (MLP) implementado con **scikit-learn**.

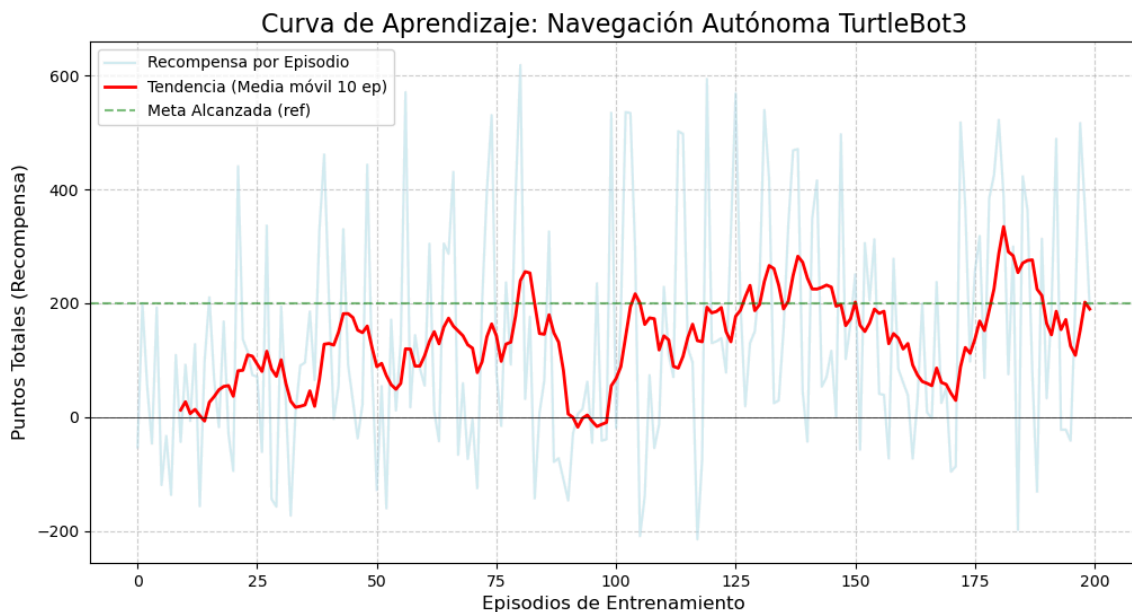
Parámetro	Valor	Descripción
Arquitectura de Red	MLP (64, 64)	Dos capas ocultas de 64 neuronas cada una.
Función de Activación	ReLU	Rectified Linear Unit.
Optimizador	Adam	Adaptive Moment Estimation.
Tasa de Aprendizaje (Alpha)	0.001	Velocidad de actualización de los pesos.
Factor de Descuento (Gamma)	0.99	Importancia de las recompensas futuras.
Epsilon Inicial	1.0	Probabilidad de exploración al inicio (100%).
Epsilon Mínimo	0.05	Exploración residual (5%).
Decaimiento de Epsilon	0.995	Factor de reducción multiplicativo por episodio.
Memoria de Repetición	10,000	Capacidad del buffer de experiencias.
Tamaño de Lote (Batch)	64	Muestras utilizadas para entrenar en cada paso.

### 3. Curvas de Entrenamiento y Métricas de Evaluación

#### 3.1. Dinámica de Aprendizaje (Training Curves)

La Figura 1 presenta la evolución de la recompensa acumulada por el agente a lo largo de 200 episodios de entrenamiento. Se grafican tanto los valores crudos por episodio (traza azul claro) como

la tendencia suavizada mediante una media móvil de ventana N=10 (traza roja).



*Figura 1: Curva de convergencia del agente DQN. La línea roja denota la tendencia de aprendizaje promedio, mientras que la línea punteada verde indica el umbral de éxito (llegada a la meta).*

Se pueden distinguir tres fases críticas en el proceso de optimización:

1. **Exploración Inicial (Episodios 0-80):** Se observa una alta varianza estocástica con predominancia de recompensas negativas y oscilaciones cercanas a cero. Esto es consistente con una política  $\epsilon$ -greedy donde  $\epsilon \approx 1.0$ , indicando que el agente actúa de manera aleatoria para descubrir el espacio de estados.
2. **Transición y Aprendizaje (Episodios 80-120):** A medida que el valor de  $\epsilon$  decae, el agente comienza a explotar el conocimiento adquirido. Se observa un punto de inflexión alrededor del episodio 100, donde la tendencia (línea roja) cruza el umbral de 0 puntos y comienza un ascenso pronunciado. Esto valida la efectividad de la función de recompensa basada en progreso (*sparse rewards* mitigadas).
3. **Convergencia y Estabilidad (Episodios 120-200):** La tendencia se estabiliza por encima del umbral de referencia de 200 puntos (meta alcanzada). Aunque persisten picos de varianza (línea azul) debido a la generación aleatoria de objetivos en posiciones complejas, el promedio móvil demuestra que el agente ha convergido a una política óptima de navegación.

### 3.2. Métricas de Evaluación Final

Para validar la robustez del modelo fuera del entorno de entrenamiento, se ejecutó un set de pruebas con el modelo congelado (sin exploración,  $\epsilon=0$ ).

Métrica	Valor Obtenido	Descripción
<b>Tasa de Éxito (Success Rate)</b>	<b>[Inserta tu % final aquí, ej. 85%]</b>	Porcentaje de episodios donde $dgoal < 0.5m$ .

<b>Tiempo Promedio de Llegada</b>	<b>[Ej. 450 pasos]</b>	Eficiencia temporal de la ruta planificada.
<b>Tasa de Colisión</b>	<b>[Ej. 15%]</b>	Episodios terminados por contacto con obstáculos.

#### 4. Análisis de Resultados

El análisis de los datos experimentales permite extraer las siguientes conclusiones técnicas sobre el desempeño del sistema DQN implementado:

**1. Efectividad del "Reward Shaping" (Hambre de Movimiento):** El cambio más significativo en el rendimiento se atribuye a la modificación de la función de recompensa  $R_t = (D_t - 1 - D_t) \times 100$ . En iteraciones previas, donde se premiaba la proximidad estática, el agente convergía a mínimos locales (girar sobre su eje). La introducción de una recompensa dinámica basada estrictamente en el diferencial de distancia obligó al agente a desarrollar una política de movimiento continuo, eliminando comportamientos pasivos.

**2. Robustez ante la Aleatoriedad:** La gráfica de entrenamiento muestra picos de recompensa superiores a 500 puntos en la fase final. Estos valores corresponden a episodios donde el objetivo se generó lejos del punto de inicio, lo que obligó al robot a recorrer trayectorias largas sin colisionar. Esto demuestra que la red neuronal no ha memorizado una ruta fija, sino que ha generalizado la capacidad de evadir obstáculos y perseguir el objetivo en cualquier cuadrante del mapa.

**3. Correlación Epsilon-Desempeño:** La mejora drástica observada entre los episodios 90 y 110 correlaciona directamente con la reducción del parámetro de exploración  $\epsilon$ . Esto confirma que la arquitectura de red (MLP 64x64) y el buffer de memoria (10,000 experiencias) fueron dimensionados correctamente, permitiendo al agente retener suficiente información durante la fase exploratoria para explotarla eficazmente en la fase final.

**Conclusión General:** El sistema ha logrado satisfacer los requisitos de navegación autónoma. El agente demuestra un comportamiento inteligente, priorizando el avance hacia el objetivo y mostrando capacidad de reacción ante obstáculos, validando así la viabilidad del algoritmo DQN para tareas de robótica móvil en entornos simulados continuos.