Memorias del XX Congreso Latinoamericano de Control Automático (CLCA 2025)
13-17 de Octubre, 2025. Cancún, Quintana Roo, México

Spherical Robot Control: Classical and Intelligent Approaches

Ernesto Fabregas * **Guelis Montenegro** ** **Karla Schröder** ***
Omar Escorza *** **Gonzalo Garcia** **** **Gonzalo Farias** ***

* *Departamento de Informática y Automática, Universidad Nacional de Educación a Distancia. Juan del Rosal, 16, Madrid, 28040, Spain, (e-mail: efabregas@dia.uned.es)*
** *Departamento de Electrotecnia e Informática, Universidad Técnica Federico Santa María, Av. Federico Santa María 6090, Viña del Mar 2520001, Chile, (e-mail: guelis.montenegro@usm.cl)*
*** *Escuela de Ingeniería Eléctrica, Pontificia Universidad Católica de Valparaíso, Av. Brasil 2147, Valparaíso, 2362804, Chile, (e-mail: gonzalo.farias@pucv.cl)*
**** *College of Engineering, Virginia Commonwealth University, Richmond, 23284, USA, (e-mail: garciaga3@vcu.edu)*

Resumen: Spherical robots offer omnidirectional mobility and mechanical robustness, making them ideal for exploration and surveillance in unstructured environments. However, their control is challenging due to their non-linear and coupled dynamics. This paper presents the design, modelling and control of a spherical robot with an internal pendulum. The approach combines realistic simulations in CoppeliaSim, experimental validation using a visual tracking platform, and advanced control strategies based on Deep Reinforcement Learning (DQN and DDPG) and traditional approaches. Results from both simulation and real-world experiments demonstrate the effectiveness of model-free learning techniques in achieving stable and accurate position control.

Keywords: Robótica Móvil; Control Clásico; Redes Neuronales

1. INTRODUCCIÓN

La robótica es un campo multidisciplinar que integra ingeniería mecánica, electrónica, informática y teoría del control para diseñar y construir máquinas capaces de realizar tareas de forma autónoma o semi-autónoma. En las últimas décadas, la robótica ha evolucionado rápidamente y ha dado lugar al desarrollo de sistemas cada vez más sofisticados capaces de operar en diversos entornos, desde la exploración de fondos marinos hasta las misiones planetarias (Greitans, 2025).

Entre los distintos tipos de robots, los móviles han ganado especial atención por su capacidad para navegar e interactuar con entornos dinámicos. Estos robots no están fijos en una única ubicación y pueden desplazarse utilizando distintos mecanismos de locomoción como ruedas, patas, orugas o incluso propulsión aérea (Siegwart et al., 2011). Los robots móviles se utilizan ampliamente en aplicaciones que van desde la automatización de almacenes y vehículos autónomos hasta las operaciones de búsqueda y rescate (Sharma et al., 2023).

Una subclase única y atractiva de robots móviles es el robot esférico. Caracterizados por una carcasa exterior esférica que encierra todos los componentes internos, estos robots se mueven rodando, ofreciendo movilidad omnidireccional y un alto grado de protección mecánica. Su diseño proporciona estabilidad y resistencia al vuelco, lo que los hace adecuados para terrenos accidentados y entornos peligrosos. Los robots esféricos se han explorado para aplicaciones de vigilancia, reconocimiento, exploración planetaria y robótica educativa (Diouf et al., 2024). Sin embargo, sus mecanismos internos de actuación y sus estrategias de control plantean importantes retos debido a la dinámica no lineal y al acoplamiento entre los movimientos de rotación y traslación.

Durante los últimos años, los robots esféricos han evolucionado hacia diseños con grados de libertad redundantes, lo que permite una mayor maniobrabilidad y resistencia a perturbaciones externas. Por ejemplo, el robot XK-III (Lin et al., 2024) incorpora un observador de perturbaciones no lineales y un controlador difuso PID para mejorar su desempeño en terrenos irregulares. Además, se han propuesto nuevos enfoques de control como el uso de controladores Riccati con observadores de velocidad basados en Kalman (Wang et al., 2023), que permiten un seguimiento más preciso de la velocidad en múltiples tipos de terreno.

Estos métodos han demostrado ser más eficaces que los controladores PID convencionales, especialmente en tareas que requieren cambios rápidos de dirección o adaptación a superficies variables (Vidaña et al., 2024). Estas innovaciones refuerzan el papel de los robots esféricos como plataformas versátiles para aplicaciones en exploración, vigilancia y asistencia en entornos no estructurados.

https://doi.org/10.58571/CNCA.AMCA.2025.015

Este trabajo aborda estos retos con una visión global del diseño, modelado, construcción y control de un robot esférico, presentando las siguientes contribuciones:

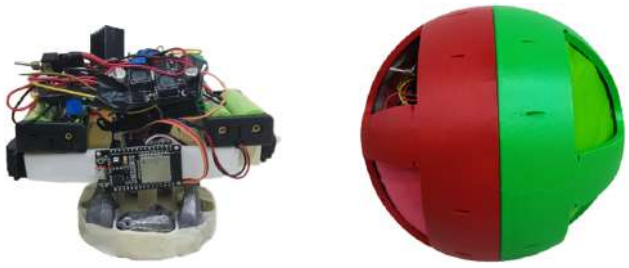
- **Simulaciones realistas:** Utilización de CoppeliaSim para modelar la dinámica no lineal y los efectos de fricción del robot en un entorno muy dinámico e interactivo (Montenegro et al., 2022).
- **Validación experimental:** Implementación del control de posición del robot esférico real en una plataforma de pruebas que utiliza el seguimiento visual con SwisTrack para obtener la posición global del robot (Schröder et al., 2023).
- **Control de posición del robot mediante Inteligencia Artificial:** Aplicación de algoritmos de Deep Reinforcement Learning DQN (por sus siglas en inglés, Deep Q-Network) y DDPG (por sus siglas en inglés, Deep Deterministic Policy Gradient), para optimizar las políticas de control sin depender de modelos lineales (Escorza et al., 2025).

El resto de este artículo se organiza como sigue. En la Sección 2 se presenta el diseño y modelado del robot esférico, así como las estrategias de control implementadas. La Sección 3 detalla la simulación y los resultados experimentales obtenidos utilizando plataformas virtuales y físicas. La Sección 4 presenta una discusión y evaluación de los resultados. Finalmente, en la Sección 5 se presentan las conclusiones y las líneas de trabajo futuras.

2. DISEÑO, MODELADO Y CONTROL DEL ROBOT

2.1 Diseño del Robot

El robot se compone de dos servomotores conectados horizontalmente a la carcasa esférica que permiten el movimiento hacia delante. Un motor de corriente continua conectado al péndulo para realizar el movimiento en sentido horario o antihorario. Los tres motores se accionan mediante señales PWM para modificar sus velocidades. La Figura 1-a muestra los componentes del robot y la Figura 1-b el robot armado.



(a) Componentes internos. (b) Robot esférico.

Figura 1. Diseño del robot.

2.2 Modelado del robot esférico en CoppeliaSim

El robot esférico modelado en este trabajo consiste en una carcasa esférica impresa en 3D que encierra todos los componentes internos y permite el movimiento de balanceo mediante accionamiento interno. El mecanismo interno se basa en un péndulo y tres motores: dos servomotores unidos a la carcasa mueven el péndulo para desplazar el centro de gravedad, mientras que un motor de corriente continua permite girar el péndulo para inducir la rotación. Esta configuración permite al robot moverse y mantener el equilibrio con un único punto de contacto con el suelo.

El robot se diseñó inicialmente con Autodesk Fusion 360 (Verma, 2018) y con Tinkercad - 3D Design (Kelly, 2014) y posteriormente se ensambló virtualmente en el simulador CoppeliaSim (Montenegro et al., 2022). El modelo de simulación incluye representaciones precisas de la masa, la inercia y la geometría, lo que garantiza un comportamiento dinámico realista. El diámetro total del robot es de 20 cm, y los componentes internos están dispuestos simétricamente para mantener el equilibrio. Para girar, el robot utiliza el principio del giroscopio de momento de control (CMG). La dinámica de rotación se rige por:

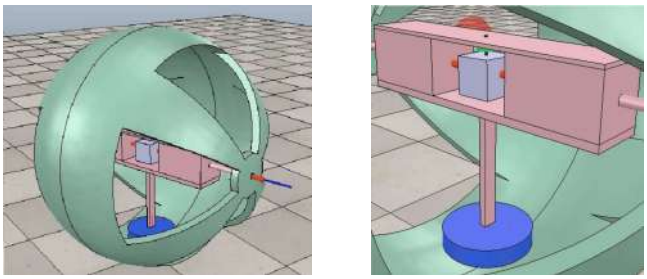
$$\dot{q} = \omega \tag{1}$$

$$\dot{\omega} = \frac{B_1}{J_1}\omega + \frac{B_2}{J_1}W + \frac{1}{J_1}\tau_m \tag{2}$$

$$\dot{W} = \frac{B_2}{J_{eq}}W + \frac{1}{J_{eq}}\tau_m \tag{3}$$

donde q es el ángulo de orientación del robot, ω es su velocidad angular. τ_m es el torque del péndulo, B_1 es el coeficiente de fricción del actuador interno, B_2 es el coeficiente de acoplamiento dinámico entre el actuador y la esfera, W es la velocidad angular del péndulo, J_1 es el momento de inercia de la carcasa del robot, J_2 es el momento de inercia del péndulo y J_{eq} es el momento de inercia equivalente.

Teniendo en cuenta factores como el coste, la viabilidad de los materiales, la baja complejidad en el diseño, el buen control del movimiento y la posibilidad de construcción en un corto periodo de tiempo, se eligió el mecanismo de péndulo accionado como método a utilizar. Por lo tanto, las pruebas de control de posición se basarán en los controles realizados en el artículo (Montenegro et al., 2022). El modelo simulado en el artículo está representado por la Figura 2 y considera al robot como un todo, formado por 3 partes. La carcasa (a), el péndulo y los circuitos electrónicos (b).



(a) Modelo del robot. (b) Péndulo del robot.

Figura 2. Modelo del Robot en CoppeliaSim.

Para la construcción del robot, en el caso de la carcasa, se imprimieron 2 piezas simétricas, una roja y otra verde. El diámetro de la circunferencia es de 20 cm y para unir ambas piezas se ideó un sistema de roscas que permite un acceso rápido y sencillo al circuito.

El mecanismo interno se compone de dos piezas, una que alberga los componentes electrónicos y otra que da

 https://doi.org/10.58571/CNCA.AMCA.2025.015

forma al péndulo. El péndulo alberga $0,627$ kg de peso distribuidos uniformemente entre sus aletas para generar el momento necesario para girar. En el centro, tiene una columna de $3,7$ cm que está conectada a un motor de corriente continua. En cuanto al soporte interior, se diseñó para contener los servomotores, el motor de corriente continua y el resto de componentes utilizados. Es una pieza simétrica lo suficientemente larga como para llegar a ambos extremos de la esfera.

2.3 Estrategias de Control (Enfoques Clásicos)

Para validar el modelo de robot y evaluar su rendimiento en distintas condiciones, se aplicaron y probaron varias estrategias de control. El experimento consiste en conducir un robot desde su posición actual, $C(x, y, \theta)$, hasta un punto objetivo, Target Point (T_p), por sus siglas en inglés. La Figura 3 muestra el diagrama de bloques de este experimento.

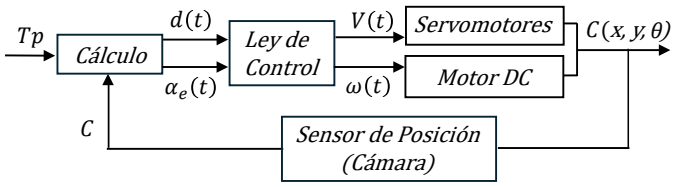


Figura 3. Diagrama en bloques del control del robot.

Las diferentes leyes de control clásico, que se implementan en el bloque Ley de Control, y entre las que se encuentran:

1.- Controlador Villela: Una ley de control clásica que calcula la velocidad lineal (V) y la angular (ω) en función de la distancia al objetivo y del error angular (α_e) (Gonzalez-Villela et al., 2004):

$$V = \begin{cases} V_{\max}, & \text{if } |d| > k_r \\ \dfrac{V_{\max}}{k_r} d, & \text{if } |d| \leq k_r \end{cases} \tag{4}$$

$$\omega = \omega_{\max} \sin(\alpha_e) \tag{5}$$

donde d es la distancia al objetivo y k_r es un radio umbral.

2.- Controlador Proporcional Integral (IPC): Una versión mejorada del controlador Villela que incluye un término integral para mejorar la convergencia. Las leyes de control son (Fabregas et al., 2020):

$$V = \text{mín}(K_v \cdot p(\alpha_e) \cdot d, V_{\max}) \tag{6}$$

$$\omega = K_p \sin(\alpha_e) + K_i \int_0^t \alpha_e(\tau) d\tau \tag{7}$$

donde es $p(\alpha_e) = \frac{\pi - |\alpha_e|}{\pi}$, y K_v, K_p, K_i son parámetros de ajuste.

Estas estrategias se evaluaron en escenarios que incluían la estabilización de puntos, el seguimiento de trayectorias y el control de formaciones multi-agente. El rendimiento se evaluó utilizando métricas estándar como el error integral absoluto (IAE), el error integral cuadrático (ISE) y sus variantes ponderadas en el tiempo (ITAE, ITSE).

2.4 Estrategia de control (Aprendizaje por Refuerzo)

Además de las estrategias de control clásicas, se implementó un enfoque de aprendizaje profundo por refuerzo (DRL por sus siglas en inglés, Deep Reinforcement Learning) para abordar la dinámica no lineal y acoplada del robot esférico. En concreto, se utilizó un algoritmo de aprendizaje profundo en base a redes neuronales para aprender una política de control óptima a través de la interacción con el entorno, basado en un modelo explícito del sistema (Mnih et al., 2016).

El proceso de aprendizaje se basó en la ecuación de Bellman, y la teoría de Q-learning (Schröder et al., 2023) que define la recompensa acumulada $Q(x_k, u_k)$ esperada de tomar una acción u_k en el estado x_k y seguir la política óptima a partir de entonces:

$$Q(x_k, u_k) = s(x_k, u_k) + \gamma \max_{u'} Q(x_{k+1}, u') \tag{8}$$

con γ, un factor de descuento para valores futuros. En Q-learning, los valores Q se actualizan iterativamente utilizando la regla de aprendizaje de la diferencia temporal:

$$\begin{aligned} Q_{i+1}(x_k, u_k) = Q_i(x_k, u_k) + \alpha \big[R(x_k, u_k) \\ + \gamma \max_{u'} Q_i(x_{k+1}, u') - Q_i(x_k, u_k) \big] \end{aligned} \tag{9}$$

donde α es la tasa de aprendizaje y $R(x_k, u_k)$ es la recompensa inmediata. La función de recompensa se diseñó para penalizar los errores angulares grandes y el esfuerzo de control excesivo, fomentando el movimiento suave y preciso hacia el objetivo. La técnica utilizada se diseñó sobre la base de la extensión desde Q-learning a DQN y DDPG, al utilizar redes neuronales llamadas Critic y Actor, (Escorza et al., 2025). El método basado en datos tabulados y de naturaleza discreta se modifica para proporcionar valores continuos. DQN reemplaza la estructura tabular por una continua, utilizando una red neuronal artificial (ANN). Además, resuelve dos problemas que surgen con el entrenamiento en tiempo real de redes neuronales: la falta de independencia e igualdad de distribución (IID) entre tuplas de experiencia consecutivas $(S_t, A_t, R_{t+1}, S_{t+1})$, y la no estacionariedad del objetivo numérico de búsqueda. DQN define la función de valor $Q^a(s; \theta)$ como las salidas de una red neuronal, parametrizada por el vector θ. Estos parámetros se obtienen minimizando el costo:

$$L_i(\theta_i) = \left(r + \gamma \cdot \max_{a'} Q^a(s'; \theta_i^-) - Q^a(s; \theta_i)\right)^2 \tag{10}$$

donde θ_i^- se mantiene artificialmente fijo durante un cierto intervalo de tiempo, o se hace variar lentamente mediante un filtro pasa-bajos, para reducir la no estacionariedad. Además, se selecciona aleatoriamente una tupla de experiencia desde un bufer deslizante para reducir la correlación. La Ecuación (10) comparte la misma estructura que (9) en términos del objetivo numérico de búsqueda y el error, donde r es la función de recompensa y s y a son el estado actual y la actuación, respectivamente. DQN sigue utilizando valores discretos para las acciones, teniendo una salida Q desde la red neuronal para cada valor discreto de acción. El operador max selecciona el valor de actuación óptimo.

DDPG por su parte, se deriva de DQN, buscando reemplazar la naturaleza discreta de las acciones por una continua. En este sentido, DDPG hereda la formulación de $Q(s, a; \theta)$ para la estimación de la recompensa, ahora llamada *Critic*, pero en lugar de obtener la acción mediante una maximización entre sus salidas discretas, implementa

 https://doi.org/10.58571/CNCA.AMCA.2025.015

una red neuronal adicional $\mu(s;\phi)$, llamada *Actor*, para generar la acción de control óptima. De forma similar, los parámetros del Critic θ se obtienen minimizando la siguiente función de costo:

$$L_i(\theta_i) = \left(r + \gamma \cdot Q(s', \mu(s';\phi_j^-);\theta_i^-) - Q(s,a;\theta_i)\right)^2 \quad (11)$$

Pero ahora, teniendo la acción $a = \mu(s;\phi)$ como entrada, se reemplaza la maximización sobre el espacio de actuación. Esta actuación se obtiene entonces maximizando la siguiente función de costo:

$$J_j(\phi_j) = Q(s, \mu(s;\phi_j);\theta_i) \quad (12)$$

Aquí, se adopta el mismo enfoque para el vector de parámetros del Actor ϕ con el fin de reducir el efecto de un objetivo no estacionario. El diagrama final de interconexión se puede ver en la Figura 4.

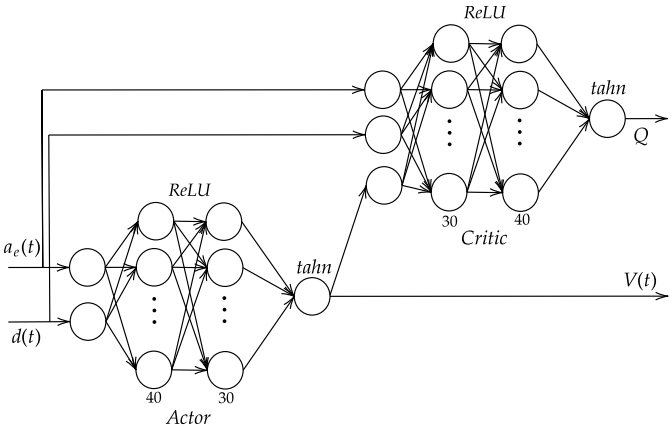


Figura 4. Redes neuronales en el controlador con 2 salidas.

El controlador utiliza el algoritmo DDPG (Sumiea et al., 2024) para calcular la señal de control V basada en el error angular α_e y la distancia al objetivo d, sintonizada con la función de recompensa $r = -10 \cdot \alpha_e^2 - 12 \cdot V_d$, siendo $V_d = \frac{d}{dt}d$ la tasa de cambio temporal de la distancia. Esta adición a la función de recompensa fue para forzar indirectamente al controlador a reducir el tiempo total para alcanzar el objetivo, recompensando mayores velocidades negativas de distancia. La señal Q es usada durante el entrenamiento para ajustar los pesos de las redes neuronales (tanto *Critic* como *Actor*), pero cuando ya está entrenado el controlador, solo se exporta al robot la red llamada Actor, que en este caso es la funcion f_{DDPG} que se muestra en la Figura 5. La señal de control ω se diseña como en los casos anteriores:

$$\begin{aligned} \omega(t) &= f^{\omega}_{\text{DDPG}}(\alpha_e(t), \alpha_e^{\text{int}}(t)) \\ V(t) &= f^{V}_{\text{DDPG}}(\alpha_e(t), d(t)) \end{aligned} \quad (13)$$

La estructura del controlador general es muy similar a la anterior DDPG, ahora el segundo actor salidas V. El diagrama en bloques del controlador se puede ver en la Figura 5.

3. RESULTADOS EXPERIMENTALES

3.1 Controlador Villela

Las primeras pruebas de simulación se realizaron con el controlador Villela. En este caso, el punto inicial del robot

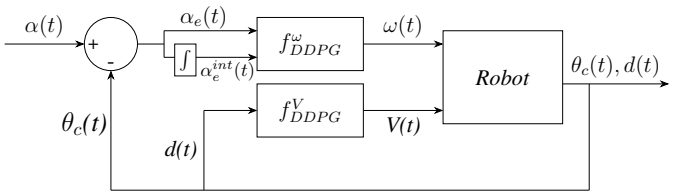
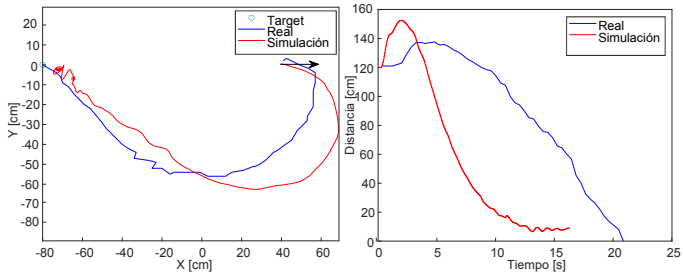


Figura 5. Diagrama en bloques del controlador DDPG.

es la posición $X_c = 40$ cm, $Y_c = 0$ cm y su objetivo está en $X_p = -80$ cm, $Y_p = 0$ cm. La flecha muestra la orientación inicial a 0 grados. La Figura 6 muestra la trayectoria del robot, así como el tiempo que tarda en llegar al destino.



(a) Posición. (b) Distancia.

Figura 6. Posición y distancia al objetivo $X_p = -80$ cm, $Y_p = 0$ cm con el controlador Villela.

Puede observarse en este caso que tanto la simulación como el comportamiento del robot, en la realidad, son similares en la trayectoria. Sin embargo, en términos de tiempo, la simulación tarda bastante menos, alcanzando el objetivo aproximadamente 6 segundos antes. Esta diferencia puede ser producida tanto por la superficie como por las consideraciones geométricas que difieren del modelo construido, ya que está basado en las simulaciones realizadas en el artículo (Schröder et al., 2023) y adaptadas para el robot construido en este artículo.

Finalmente, observando la Figura se puede determinar que el robot mantiene su comportamiento respecto a la trayectoria realizada, en términos de tiempo el robot real tarda más en alcanzar el objetivo. Esto se puede relacionar directamente con la velocidad elegida para el robot y la diferencia de dinámica entre el robot simulado y el real.

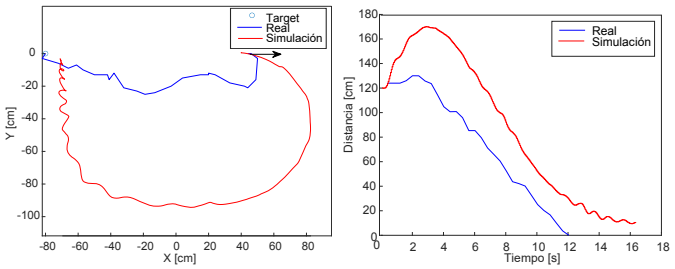
3.2 Controlador IPC

En este caso, el punto de partida del robot es la posición $X_c = 40$ cm, $Y_c = 0$ cm y su objetivo está en $X_p = -50$ cm, $Y_p = 0$ cm. La flecha muestra la orientación inicial a 0 grados. La Figura 7 muestra la trayectoria del robot y el tiempo que tarda en llegar al destino.

En el caso de la simulación, la trayectoria utilizada es mucho más amplia que en el caso real. Por tanto, podría decirse que la trayectoria no es la misma que en la simulación. Sin embargo, en términos de tiempo, el robot mantiene una relación distancia-tiempo similar, que se mantiene a medida que se acerca al objetivo. Esta diferencia en la trayectoria puede deberse a las mismas razones expuestas en el controlador desarrollado por Villela.

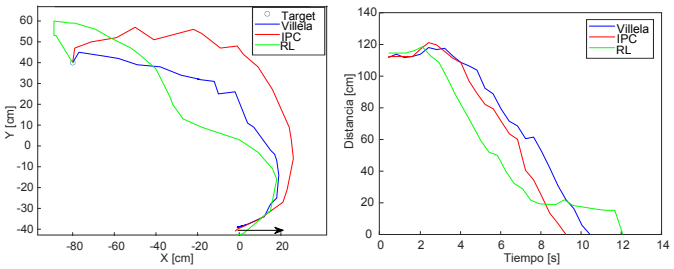
Finalmente, observando las gráficas, se puede determinar que existen diferencias de comportamiento tanto en el

 https://doi.org/10.58571/CNCA.AMCA.2025.015



(a) Posición. (b) Distancia.

Figura 7. Posición y distancia al objetivo $X_p = -80$ cm, $Y_p = 0$ cm con el controlador IPC.



(a) Posición. (b) Distancia.

Figura 8. Posición y distancia al objetivo $X_p = -80$ cm, $Y_p = 40$ cm con los tres controladores.

desplazamiento real como en el simulado, ya que el robot real disminuye su error angular más rápidamente que en la simulación, recorriendo menos distancia. Esto puede deberse a la diferencia entre el modelo matemático y la realidad, donde tanto las consideraciones de las superficies como la geometría del robot pueden no ser tan similares.

3.3 Aprendizaje Profundo por Refuerzo

Para comprobar el funcionamiento del robot accionado por diferentes controladores, se realizaron pruebas en tres escenarios diferentes que se explicarán a continuación en función de la ubicación del objetivo.

Para el algoritmo de RL Q-learning el robot necesitó un número de episodios del orden de los millones de iteraciones, pero para DDPG o DQN, la cantidad de iteraciones fue del orden de las decenas de miles. El aprendizaje se hizo en simulación solamente, y ese conocimiento se exportó al robot real, y no se continuó con aprendizaje en tiempo real con el robot. La Figura 8 muestra los resultados con el controlador Villela en azul, el IPC en rojo y el Aprendizaje por Refuerzo en verde. En el gráfico de la izquierda se muestra la trayectoria seguida por el robot para alcanzar el punto final, mientras que en el de la derecha, la disminución de la distancia en función del tiempo. Aquí se aplican las mismas consideraciones para la gestión de las señales PWM.

Para este caso, se utilizó $X_c = 0$, $Y_c = -40$ como punto de partida y $X_p = -80, Y_p = 40$ como objetivo. La flecha muestra la orientación inicial a 0 grados. A simple vista, se puede observar que el controlador IPC es el más rápido con un tiempo aproximado de 9 segundos para llegar al destino.

Por otro lado, el controlador de aprendizaje por refuerzo (RL, por sus siglas en inglés, Reinforcement Learning) es el que más tiempo tarda con 12 segundos, esto es debido a que al acercarse al objetivo realizó una desviación que le impidió llegar en menos tiempo, teniendo que corregir la orientación y por tanto tardando casi 3 segundos más.

Además, la calidad de cada algoritmo de control se evalúa utilizando métricas de rendimiento. Estas métricas utilizan la integral del error, que en este caso es la distancia al punto objetivo (Target). Las medidas de rendimiento consideradas en este documento son (1) Error Cuadrático Total (ISE), (2) Error Absoluto Total (IAE), (3) Error Cuadrático Total en Tiempo Total (ITSE) y (4) Error Absoluto en Tiempo Total (ITAE). En cuanto a los índices de rendimiento mostrados en la Tabla 1, se puede observar que los índices son menores en el control RL; sin embargo, al observar las gráficas de la Figura 8, el controlador es el que más tiempo tarda en alcanzar el objetivo.

Tabla 1. Índices de Desempeño.

Índices	Villela	IPC	RL
IAE	797	703	641
ISE	76977	68857	53879
ITSE	275800	219640	150900
ITAE	3307	2579	2517

Considerando lo anterior, se podría esperar que los índices ITSE e ITAE fueran mayores; sin embargo, la distancia que tienen los demás controladores genera una compensación, obteniendo finalmente un mejor índice RL. Aunque al principio de la trayectoria es el primero en reducir el error angular, durante la trayectoria próxima al objetivo se desvía, lo que produce este retraso en comparación con IPC, que es el más competitivo después de RL. En relación con Villela, es el que más tarda y, en consecuencia, tiene los peores índices de rendimiento.

4. DISCUSIÓN Y EVALUACIÓN

4.1 Evaluación del desempeño del robot

Para examinar más a fondo el comportamiento de los controladores DRL, se probaron diseños de control más antiguos, Q-learning RL (Schröder et al., 2023), Villela (Gonzalez-Villela et al., 2004), e IPC (Integral Proportional Controller) (Fabregas et al., 2020), para un escenario común. El punto de partida elegido es $(X_c, Y_c) = (40, 0)$, pero con el objetivo situado ahora en $(X_p, Y_p) = (-40, 0)$, lo que permite disponer de más tiempo para maniobrar. La Figura 9 muestra los resultados obtenidos.

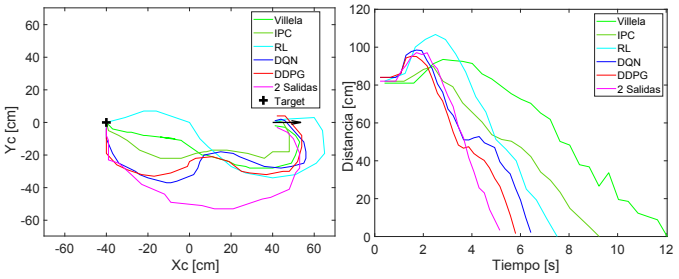


Figura 9. Posición y distancia al $T_p(X_p, Y_p) = (-40, 0)$.

En general, todos los controladores tomaron rutas diferentes cada vez que se realizó la prueba. En cuanto a

 https://doi.org/10.58571/CNCA.AMCA.2025.015

la distancia o el tiempo que tardaron los algoritmos en llegar al punto, se pueden hacer varias observaciones. La primera de ellas es que los tres controladores DRL obtuvieron mejores resultados en términos de tiempo. Otro punto importante es que esta prueba es coherente con la tendencia de las anteriores en el orden de llegada. Para los controladores no basados en DRL, el que menos tiempo tardó fue el controlador RL, seguido del IPC y el que más tardó fue el Villela, lo cual era de esperar, coherente con el (Montenegro et al., 2022). Para evaluar la calidad de cada algoritmo, se utilizaron los índices de rendimiento que se resumen en la Tabla 2, para el objetivo en el punto $(X_p, Y_p) = (-40, 0)$.

Tabla 2. Índices de Desempeño.

Índices	Villela	IPC	RL	DQN	DDPG	2 Salidas
IAE	694	477	474	389	342	325
ISE	51437	32462	39477	28706	25017	25753
ITSE	208130	96459	107430	63494	49322	49494
ITAE	3249	1697	1453	1019	799	693

Tras analizar todas las pruebas, se puede afirmar que el controlador con mejores resultados, tanto en los índices de rendimiento como en el tiempo empleado, es el controlador DDPG con 2 salidas (V, ω). Esto puede atribuirse a que se trata de un algoritmo de control en tiempo continuo, que gestiona tanto la velocidad lineal como la angular del robot de forma concurrente, basándose en las dos señales de medida.

5. CONCLUSIONES Y TRABAJO FUTURO

Este trabajo presenta el modelado, diseño y control de un robot esférico utilizando el entorno de simulación CoppeliaSim y una plataforma real. Se implementaron y evaluaron varias estrategias de control, incluidos enfoques clásicos (Villela e IPC) y un método de aprendizaje por refuerzo (Q-learning). Los controladores basados en DRL (DQN y DDPG) demostraron un rendimiento superior en términos de suavidad de trayectoria, tiempo de convergencia y adaptabilidad a escenarios dinámicos. Tanto los resultados de la simulación como los del robot real mostraron que las estrategias de control propuestas son eficaces para lograr un movimiento estable y preciso en diversas tareas.

El trabajo futuro se centrará la integración de un sistema de seguimiento visual en tiempo real y la validación de los resultados de la simulación en condiciones reales. Además, se explorarán nuevas mejoras en la eficacia y robustez del aprendizaje mediante el uso de algoritmos DRL avanzados y espacios de acción continuos.

AGRADECIMIENTOS

Esta investigación fue apoyada en parte por la Agencia Nacional de Investigación y Desarrollo de Chile (ANID) bajo el Proyecto FONDECYT 1191188. La financiación de la Universidad Nacional de Educación a Distancia (UNED) para la publicación de acceso abierto. El Ministerio de Ciencia e Innovación de España bajo el Proyecto PID2022-137680OB-C32. La Agencia Estatal de Investigación (AEI) bajo el Proyecto PID2022-139187OB-I00.

REFERENCIAS

Diouf, A., Belzile, B., Saad, M., and St-Onge, D. (2024). Spherical rolling robots—design, modeling, and control: A systematic literature review. *Robotics and Autonomous Systems*, 175, 104657.

Escorza, O., Garcia, G., Fabregas, E., Velastin, S.A., Eskandarian, A., and Farias, G. (2025). Deep reinforcement learning applied to a spherical robot for target tracking. *IEEE Transactions on Industrial Electronics*, 1–10.

Fabregas, E., Farias, G., Aranda-Escolástico, E., Garcia, G., Chaos, D., Dormido-Canto, S., and Dormido, S. (2020). Simulation and experimental results of a new control strategy for point stabilization of nonholonomic mobile robots. *IEEE Transactions on Industrial Electronics*, 67(8), 6679–6687.

Gonzalez-Villela, V., Parkin, R., Para, M., and Dorador-Gonzalez, J. (2004). A wheeled mobile robot with obstacle avoidance capability. *Mech. Technologia*, 1, 150–159.

Greitans, M. (2025). Automatic control and computer sciences: A multidisciplinary platform for robotics and control systems. *Automatic Control and Computer Sciences*, 58(8).

Kelly, J.F. (2014). *3D modeling and printing with Tinkercad: create and print your own 3D models*. Que Publishing.

Lin, R., Huo, J., Yang, X., Wang, Q., Yang, R., and Xu, J. (2024). XK-III: A spherical robot with redundant degrees of freedom. *Journal of Intelligent & Robotic Systems*, 110(104).

Mnih, V., Badia, A., Mirza, M., Graves, A., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. *International Conference on Machine Learning*, 1928–1937.

Montenegro, G., Chacón, R., Fabregas, E., et al. (2022). Modeling and control of a spherical robot in the coppeliasim simulator. *Sensors*, 22(16), 6020.

Schröder, K., Garcia, G., Chacón, R., et al. (2023). Development and control of a real spherical robot. *Sensors*, 23(8), 3895.

Sharma, N., Pandey, J.K., and Mondal, S. (2023). A review of mobile robots: Applications and future prospect. *International Journal of Precision Engineering and Manufacturing*, 24, 1695–1706.

Siegwart, R., Nourbakhsh, I.R., and Scaramuzza, D. (2011). *Introduction to Autonomous Mobile Robots*. MIT Press.

Sumiea, E.H., Abdulkadir, S.J., Alhussian, H.S., Al-Selwi, S.M., Alqushaibi, A., Ragab, M.G., and Fati, S.M. (2024). Deep deterministic policy gradient algorithm: A systematic review. *Heliyon*.

Verma, G. (2018). *Autodesk fusion 360 black book*. BPB Publications.

Vidaña, J.E.V., Garcia-Hernandez, R., Alvarez-Macias, C., and Murillo-Garcia, O.F. (2024). Pid control for a spherical robot applied as a solar tracking system. In *2024 XXVI Robotics Mexican Congress (COMRob)*, 123–128. IEEE.

Wang, Y., Wang, Y., Liu, Y., Guan, X., Hu, T., Zhang, Z., Hao, J., and Li, G. (2023). New riccati velocity controller of the spherical robot. *Journal of Intelligent & Robotic Systems*, 108(3), 51.

 https://doi.org/10.58571/CNCA.AMCA.2025.015