XLIV Jornadas de Automática 2023

https://jautomatica.es

Diseño de un Control de Velocidad mediante Redes Neuronales y Algoritmos Genéticos para Vehículos Autónomos

Argente-Mena, J.[a, *], Santos-Peñas, M.[b], García-Sierra, J.E.[c]

[a] *Facultad de Informática, Universidad Complutense de Madrid, Calle del Prof. José García Santesmases, nº9, 28040 Madrid, España.*
[b] *Instituto de Tecnología del Conocimiento, Universidad Complutense de Madrid, Calle del Prof. José García Santesmases, nº9, 28040 Madrid, España.*
[c] *Departamento de Ingeniería Electromecánica, Universidad de Burgos, Campus Río Vena. Edificio A, Avda. Cantabria s/n, 09006 Burgos, España.*

To cite this article: Argente-Mena, J., Santos-Peñas, M., García-Sierra, J.E. 2023. *Design of an Adaptive Speed Control using Neural Networks and Genetic Algorithms for Autonomous Vehicles* XLIV Jornadas de Automática, 121-126. https://doi.org/10.17979/spudc.9788497498609.121

Resumen

Los Vehículos Autónomos Guiados (AGV) son cada vez más populares en lo que a logística interna de las fábricas se refiere debido a su capacidad para transportar cargas pesadas y su alto grado de autonomía. No obstante, la dinámica de estos robots puede sufrir cambios debido a variaciones en la carga que transportan y/o a desgaste mecánico, lo cual implica una mayor complejidad en el control de velocidad. En muchas ocasiones se emplean controladores de tipo Proporcional Integral (PI) para dicho control. Sin embargo, este controlador exige un ajuste fino y carece de suficiente robustez ante variaciones de sus condiciones de trabajo. Con el fin de mejorar el rendimiento del control de velocidad, en este artículo se presenta el diseño de un neuro-controlador. Dado que encontrar unos valores óptimos para los hiperparámetros de aprendizaje puede ser difícil y requiere múltiples pruebas y ajustes, se opta por utilizar un Algoritmo Genético (AG) para buscar una solución válida de entre las óptimas.

Palabras clave: Redes Neuronales, Control Inteligente, Algoritmos Evolutivos, Modelado, Robots Móviles, Vehículos Autónomos Guiados.

Design of a Speed Control using Neural Networks and Genetic Algorithms for Autonomous Vehicles.

Abstract

Autonomous Guided Vehicles (AGVs) are becoming increasingly popular in terms of internal factory logistics due to their ability to transport heavy loads and their high degree of autonomy. Nevertheless, the dynamics of these robots can undergo changes due to variations in their load and/or mechanical wear, which involves greater complexity in their speed control. Proportional Integral (PI) controllers are often used for this control. However, this controller requires fine tuning and lacks enough robustness against variations in working conditions. In order to improve the speed control performance, this article presents the design of a neuro-controller. Since finding optimal values for the learning hyperparameters can be difficult and requires multiple tests and adjustments, a Genetic Algorithm (GA) is used to find a valid solution among all the optimal.

Keywords: Neural networks, Intelligent control, Evolutionary algorithms, Modelling, Mobile robots, Autonomous Guided Vehicles

1. Introducción

Los Vehículos de Guiado Automático ofrecen versatilidad para una amplia gama de tareas y entornos, permitiendo ser operados manualmente cuando sea necesario (García et al., 2022). Sin embargo, la dinámica de estos sistemas añade complejidad a la tarea de ajuste del controlador y afecta significativamente tanto al seguimiento de la trayectoria como

*Autor para correspondencia: msantos@ucm.es; jesierra@ubu.es

a la precisión de la velocidad de crucero, debido a los cambios de las condiciones de trabajo como consecuencia de variaciones en la carga transportada o de los parámetros mecánicos por fallas específicas o desgaste. Por ello, es esencial desarrollar controladores robustos con capacidad de adaptación a condiciones de trabajo en continuo cambio o, al menos, minimizar las influencias negativas que estas generan sobre el funcionamiento del control.

En (Sierra y Santos, 2020) se emplean técnicas de aprendizaje por refuerzo para el control de seguimiento de un AGV, debido al desgaste de los elementos mecánicos que provoca desviaciones en la trayectoria. Los resultados obtenidos para trayectorias complejas superan de manera significativa a los logrados por un controlador Proporcional Integral Derivativo (PID). En (Abajo *et al*., 2021) se emplea un algoritmo genético para la sintonización de un controlador PID para el trazado de trayectorias, en el que podemos ver cuan relevante es conseguir un ajuste fino.

En este trabajo comparamos el control de velocidad de un robot diferencial usando dos técnicas de control diferentes: Proporcional Integral (PI) y Neuronal sintonizado con un Algoritmo Genético (AG). El objetivo es obtener un control adaptativo de manera que el robot se mantenga en la consigna de velocidad a pesar de variaciones paramétricas.

El resto del artículo se organiza de la siguiente manera: la Sección 2 describe las características dinámicas del AGV y sus ecuaciones del movimiento. En la 3 se explica la arquitectura de un controlador neuronal con modelo de referencia. La sección 4 se enfoca a la configuración de un algoritmo genético para sintonizar los hiperparámetros de aprendizaje del controlador. La Sección 5 muestra los resultados obtenidos y una comparación entre las técnicas de control usadas. Finalmente, la última sección reúne las conclusiones y los trabajos futuros.

2. Modelado del robot

Con el propósito de evaluar las técnicas de control comentadas anteriormente, nos hemos basado en un AGV con una configuración diferencial. La Figura 1 muestra una representación del robot a modelar junto a sus vectores de velocidad.

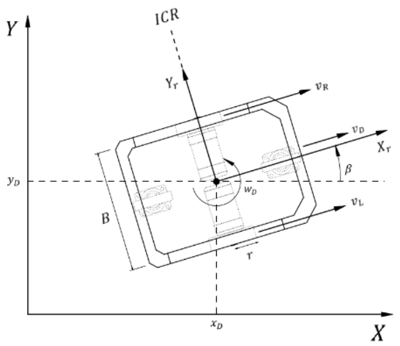

Figura 1: Configuración robot móvil diferencial

donde w_L, w_R, v_L y v_R son las velocidades angulares (rad/s) y tangenciales (m/s) de las ruedas izquierda (L) y derecha (R) respectivamente, β es el ángulo (rad) del robot respecto al sistema de coordenadas generalizadas y w_D es la velocidad de giro (rad/s) del robot alrededor de su centro instantáneo de rotación ICR.

Por simplicidad, las fuerzas de rozamiento no son tenidas en cuenta, los parámetros de ambos conjuntos motor-rueda se consideran iguales, con momentos de inercia nulos, y la masa de estos se asumen como parte del cuerpo del robot. Las ecuaciones del movimiento se obtuvieron a partir del *Lagrangiano* del sistema, considerando rodadura perfecta de las ruedas y la restricción no holonómica (1) para el deslizamiento lateral

$$\Lambda = [-\dot{x}\sin(\beta) \quad \dot{y}\cos(\beta) \quad 0] \tag{1}$$

El sistema de ecuaciones resultantes del robot móvil, únicamente para velocidades, y tras añadir los dos motores de corriente continua (DC) viene dado por las ecuaciones (2) a (6)

$$\frac{di_L}{dt} = \frac{u_L - R_m i_L - K_b w_L}{L_m} \tag{2}$$

$$\frac{di_R}{dt} = \frac{u_R - R_m i_R - K_b w_R}{L_m} \tag{3}$$

$$\frac{dw_R}{dt} = \frac{a_1(K_m i_L - B_m w_L) + b_1(K_m i_R - B_m w_R)}{2Jm_D r^2} \tag{4}$$

$$\frac{dw_L}{dt} = \frac{b_1(K_m i_L - B_m w_L) + a_1(K_m i_R - B_m w_R)}{2Jm_D r^2} \tag{5}$$

siendo

$$a_1 = (4J - B^2 m_D) \quad y \quad b_1 = (4J + B^2 m_D) \tag{6}$$

donde m_D es la masa total (Kg); i_L, i_R, u_L y u_R son las corrientes (A) y voltajes (V) de los motores de las ruedas izquierda y derecha respectivamente; K_m, K_b y B_m son la constante del par motor (Nm/A), la constante de la fuerza contra electromotriz (Vs/rad) y un coeficiente de fricción viscosa (Nms/rad), respectivamente; w_L y w_R son las velocidades angulares (rad/s) de las ruedas izquierda y derecha respectivamente; y J, r y B son el momento de inercia del cuerpo del robot $(Kg\ m^2)$, el radio de las ruedas (m) y la distancia entre los ejes de las ruedas (m).

La velocidad de crucero v_D (m/s) se obtiene de la cinemática acorde a la ecuación (7)

$$v_D = r\frac{w_R + w_L}{2} \tag{7}$$

3. Controlador neuronal

Las redes neuronales artificiales (RNA) son de gran utilidad para la resolución de problemas complejos los cuales no pueden ser resueltos de una manera sencilla mediante un enfoque algorítmico tradicional. La principal característica de estas redes es su capacidad de trabajar eficazmente con las incertidumbres e imprecisiones del mundo real. En este trabajo, se opta por utilizar un Controlador Neuronal con Modelo de Referencia (MRNNC por sus siglas en ingles) para el control de velocidad de crucero, con el objetivo de que el controlador adapte su respuesta ante variaciones paramétricas del modelo y perturbaciones externas.

3.1 Controlador Neuronal con Modelo de Referencia

La estructura del controlador se muestra en la Figura 2. Su arquitectura está compuesta por un modelo neuronal del AGV, un modelo de referencia y un neurocontrolador. El modelo neuronal tiene como objetivo identificar la dinámica del sistema, mientras que el modelo de referencia es empleado para imponer una dinámica deseada a la respuesta del robot. Por último, el controlador neuronal genera las acciones de control necesarias para que el sistema siga la salida del modelo de referencia.

Dado que las redes neuronales *feedforward*, con al menos una capa oculta y funciones de activación lineal en la capa de salida y no lineal en la oculta, pueden aproximar cualquier función no lineal continua y arbitraria, esta topología se utiliza tanto para el modelo como para el controlador neuronal.

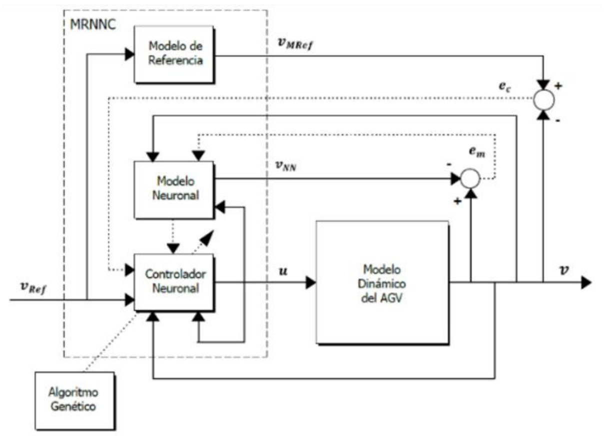

Figura 2: Diagrama de bloques de la estrategia de control

3.1.1 Red neuronal del modelo

El modelo neuronal toma como entradas las tensiones de los motores DC (u_R, u_L) y valores pasados de las velocidades de las ruedas del sistema dinámico del AGV (v_R, v_L), siendo sus salidas una predicción sobre las velocidades tangenciales de dichas ruedas (v_{NNR}, v_{NNL}). Esta red tiene una estructura *feedforward* y configuración Serie-Paralelo NNARX (*Neural Network ARX*), con 18 entradas, una capa oculta de 25 neuronas con funciones de activación *tangente sigmoide* y una capa de salida de 2 neuronas, cuyas funciones de activación son *lineales*. El número de neuronas en la capa oculta se determinó mediante prueba y error. Los vectores de entrada a la red son los mostrados en (8).

$$\begin{aligned} U_R &= [u_R(k), \dots, u_R(k-3)] \\ U_L &= [u_L(k), \dots, u_L(k-3)] \\ V_R &= [v_R(k-1), \dots, v_R(k-5)] \\ V_L &= [v_L(k-1), \dots, v_L(k-5)] \end{aligned} \qquad (8)$$

El modelo es entrenado *offline* y sobre este no se realiza ninguna actualización durante la evaluación del control. Para la actualización de sus parámetros durante el entrenamiento, se usó el algoritmo de *Levenberg–Marquardt.* El funcionamiento es evaluado con la media del error de predicción E_m al cuadrado (***mse***). El entrenamiento es llevado a cabo sobre un conjunto de *80000* muestras obtenidas tras aplicar a los motores unas tensiones de entrada aleatorias con amplitud variable entre $\pm 48V$. Tras una ejecución de *100* épocas, el resultado es de *2e-8* (*mse*)

3.1.2 Modelo de referencia

El propósito del modelo de referencia es definir la dinámica deseada para el sistema controlado. Mediante este, se pueden imponer especificaciones temporales deseadas a las velocidades de cada rueda. En este trabajo, se escogió un tiempo de establecimiento $t_{e98\%} = 2{,}5s$, y respuesta sobreamortiguada ante una entrada escalón. Para ello, se optó por una función de transferencia de primer orden con ganancia $K = 1$ y constante de tiempo $\tau = 0{,}6391$.

3.1.3 Controlador neuronal

La red neuronal del controlador es de tipo recurrente. Está formada por 28 entradas, una capa oculta de 35 neuronas y una capa de salida con 2 neuronas. Tiene una configuración de señales de entrada que clasificamos como hibrida entre los modelos Serie-Paralelo NNARX y NNOE (*Neural Network Output Error*). Las salidas de la red representan los voltajes de los motores u_R y u_L y las funciones de activación utilizadas para la capa oculta son *tangente sigmoide*, mientras que para la capa de salida se usan funciones *lineales*. El número de neuronas elegidas se determinó mediante prueba y error. La tupla de entrada a esta red la forman los vectores dados en (9)

$$\begin{aligned} V_{Rref} &= [v_{Rref}(k), \dots, v_{Rref}(k-3)] \\ V_{Lref} &= [v_{Lref}(k), \dots, v_{Lref}(k-3)] \\ U_R &= [u_R(k-1), \dots, u_R(k-5)] \\ U_L &= [u_L(k-1), \dots, u_L(k-5)] \\ V_R &= [v_R(k-1), \dots, v_R(k-5)] \\ V_L &= [v_L(k-1), \dots, v_L(k-5)] \end{aligned} \qquad (9)$$

donde v_{Lref} y v_{Rref} son las consignas de velocidad para la rueda izquierda y derecha respectivamente. La función de coste usada en esta red es también el ***mse***.

Asumiendo un buen funcionamiento del modelo neuronal, es decir $v \approx v_{NN}$, el error de control E_c es propagado hacia atrás por la red neuronal. Una vez calculado el gradiente del error, los parámetros del controlador son actualizados. El algoritmo de aprendizaje desarrollado es el *Descenso por el Gradiente con Momento* y *Regularización L2*.

Hay que destacar que la red neuronal del controlador tiene que ser actualizada *online* en cada iteración y que el algoritmo de propagación del error hacia atrás usado es estático (Argente-Mena et al, 2023).

4. Optimización con algoritmos genéticos

Los algoritmos genéticos forman parte de las técnicas heurísticas, siendo empleados para resolver problemas de búsqueda y optimización complejos. Se basan en el concepto de selección natural y evolución propuesto por Charles Darwin y los descubrimientos de George Mendel en el campo de la herencia genética. Realizan una búsqueda heurística aportando soluciones validas con un coste computacional y tiempo razonables. A pesar de que no garantizan obtener una solución óptima absoluta, sus resultados suelen ser suficientemente aceptables para muchas aplicaciones.

En nuestro caso, y dado tiempo que supone el proceso de seleccionar los valores de los hiperparámetros de aprendizaje *learning rate* (α), *momento* (β) y *factor de regularización L2* (λ) mediante prueba y error, se optó por agilizar este proceso empleando un algoritmo genético. Su configuración se realizó de la siguiente manera:

- ✓ *Número de genes:* 3 (hiperparámetros aprendizaje α, β y λ)
- ✓ *Cromosomas:* 70
- ✓ *Generaciones:* 80
- ✓ *Espacio de búsqueda de los parámetros*: acotados por los siguientes valores (10)

$$0 \leq \beta \leq 0.9 \quad 0 \leq \alpha \leq 1 \quad 0 \leq \lambda \leq 0.1 \tag{10}$$

- ✓ *Método de selección:* Torneo.
- ✓ *Probabilidad de cruce:* 80%
- ✓ *Elitismo:* 3
- ✓ *Cantidad de mutación:* Función *@mutationadaptfeasible*
- ✓ *Convergencia:* tolerancia de la función de coste a $1e^{-10}$
- ✓ *Función de coste a minimizar:* error medio absoluto (11), como resultado de evaluar los cromosomas durante un intervalo de tiempo.

$$F_{cost} = \sum_{i}^{n} |v_{MRef\,i} - v_i| \tag{11}$$

donde $v_{MRef\,i}$ es la velocidad de salida del modelo de referencia del neurocontrolador para la rueda i y v_i es la velocidad de la rueda *i* del modelo dinámico del AGV.

Hay que mencionar que este algoritmo genético no forma parte de la estructura del controlador y únicamente es usado como método de optimización en la configuración de la red neuronal del controlador.

5. Resultados

Las simulaciones se realizaron utilizando el software *Matlab* en un ordenador HP Victus 16.1 con procesador i7-12700H a 4.70GHz y sistema operativo Windows 11. El modelo del AGV fue sometido a un perfil de velocidad de referencia de onda cuadrada y de distinta amplitud y frecuencia para cada una de las ruedas. Las simulaciones duran 150s y, durante su transcurso, se introducen variaciones en la masa del robot y en los parámetros de fricción viscosa de los motores DC. Dichas variaciones se aplican siguiendo una función Gaussiana con el 560% como valor máximo sobre sus valores iniciales. Ambos controladores se comparan y analizan tanto grafica como numéricamente, utilizando para ello las métricas ***mse*** (error cuadrático medio), ***mae*** (error absoluto medio), ***R²*** (coeficiente de determinación), ***var*** (varianza) y tiempo de cómputo.

Los valores de los hiperparámetros del controlador neuronal, obtenidos como resultado de la ejecución del algoritmo genético, se fijan según (12)

$$\beta = 0.015 \qquad \alpha = 0{,}72 \qquad \lambda = 0.0001 \tag{12}$$

Dado que los resultados del AG también dependen de los valores de la población inicial, se realizaron varias ejecuciones del AG con distintos valores iniciales, siendo éste de entre todos ellos, el conjunto de valores que mejores resultados dieron.

Para el controlador PI usado como técnica alternativa, en (13) se muestra la ecuación implementada:

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau\right) \tag{13}$$

donde $u(t)$ es la acción de control (V) para el motor correspondiente, $e(t)$ es el error de velocidad $v_{Ref}(t) - v(t)$, K es la ganancia proporcional y T_i es el tiempo integral. La sintonización de este controlador se escoge de manera que la respuesta de velocidad de cada rueda tenga un tiempo de establecimiento $t_{e98\%} \approx 2.5\ s$.

Las Figuras 4(a) y 4(b) ilustran cómo el control PI (línea azul) alcanza la velocidad de referencia (verde) para cada rueda cuando los parámetros permanecen invariables. Sin embargo, cuando se somete a una variación de masa, con valor máximo en *t=50s* (Figura 4(c)), el rendimiento del controlador se deteriora, apareciendo desviaciones de su respuesta ideal (rojo). Estas desviaciones se reflejan en la velocidad de crucero v_D (Figura 4(d)) y, aparte de no mantener la velocidad de referencia, provocan una sobre-elongación que varía entre el 4 y el 26%.

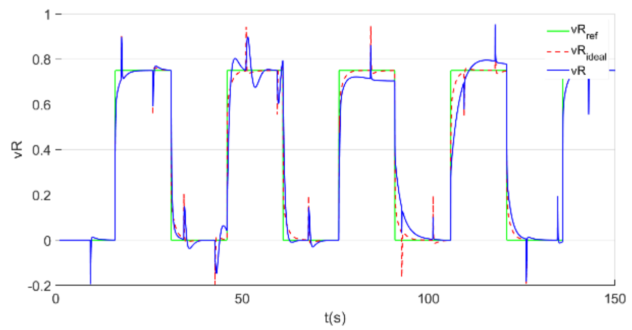

Figura 4(a): Control PI. Velocidad rueda derecha

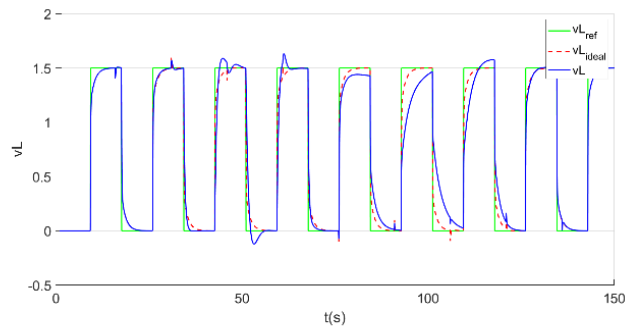

Figura 4(b): Control PI. Velocidad rueda izquierda

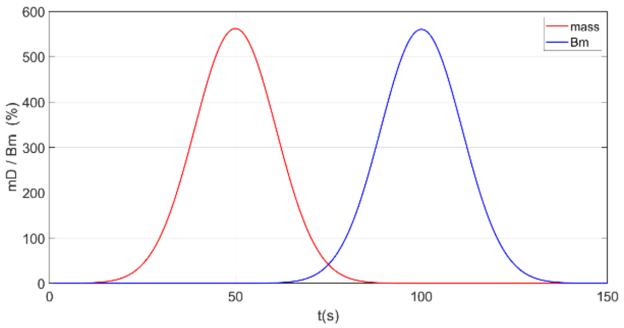

Figura 4(c): Control PI. Variación paramétrica porcentual

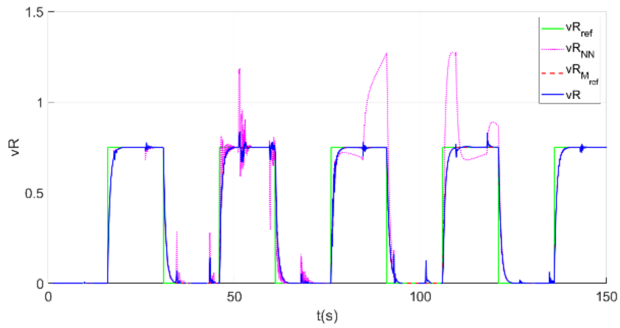

Figura 5(a): Control MRNN. Velocidad rueda derecha

Respecto al período de variación del coeficiente de fricción viscosa, con valor máximo en *t=100s*, el rendimiento empeora considerablemente. Observando la gráfica de v_D, la sobre-elongación no excede del 6% en *t=116s*; sin embargo, el controlador no es capaz de mantenerse en la consigna de velocidad durante el periodo que dicha perturbación dura.

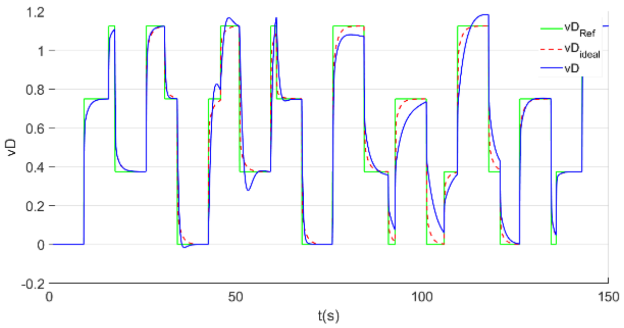

Figura 4(d): Control PI. Velocidad de crucero

Fijándonos ahora en las Figuras 5(a) y 5(b), estas muestran, en azul, la respuesta temporal del AGV controlado, en magenta la del modelo neuronal y verde para el perfil de velocidad.

La respuesta del modelo de referencia se representa con una línea roja definida a trazos. Dado el buen funcionamiento del controlador, dicha línea no es fácilmente observable ya que queda solapada por la respuesta del sistema. Durante la variación de la masa se produce un sobrepaso de alrededor del 10% de la señal de referencia en la rueda derecha. Por el contrario, en la rueda izquierda obtenemos una respuesta muy satisfactoria para este tipo de variación. Únicamente aparecen leves oscilaciones que no superan el 1.5%. Respecto al periodo de variación de los parámetros de fricción, se aprecia una respuesta similar a la del momento de variación de masa, pero con oscilaciones de menor amplitud (1%) que sólo se dan durante los transitorios. Con todo ello, la velocidad de crucero se mantiene bastante estable en su velocidad de referencia durante toda la simulación y con una dinámica dictada por el bloque del modelo de referencia, tal y como se muestra en la Figura 5(d).

Si nos fijamos detenidamente en las Figuras 4(a) y 4(b), se puede apreciar el acoplamiento de este sistema multivariable (MIMO). Cada vez que se produce un salto en la velocidad de una rueda, se observa un pico en sentido contrario en la otra rueda, incluso cuando apenas existe variación paramétrica. Al utilizar el controlador funciones de transferencia (f.d.t) de primer orden para la dinámica deseada de cada rueda y, dado que las respuestas de estas f.d.t son completamente independientes entre sí y de la dinámica del sistema, se logra reducir la influencia de una rueda sobre la otra al forzar al controlador de cada rueda a seguir la respuesta de su modelo de referencia correspondiente. Si nos fijamos en el primer escalón de velocidad en las Figuras 5(a) y 5(b) se ve como la interacción entre las ruedas disminuye muy notablemente en comparación con las Figuras 4(a) y 4(b), aunque no es un desacoplamiento perfecto.

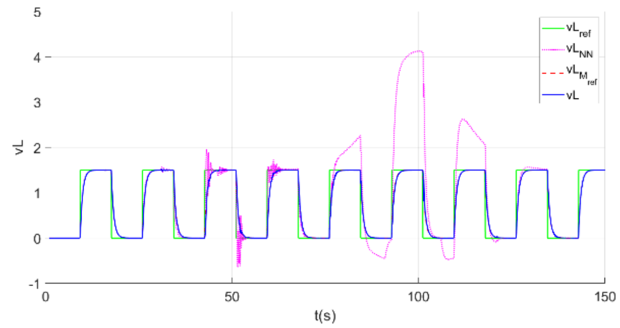

Figura 5(b): Control MRNN. Velocidad rueda izquierda

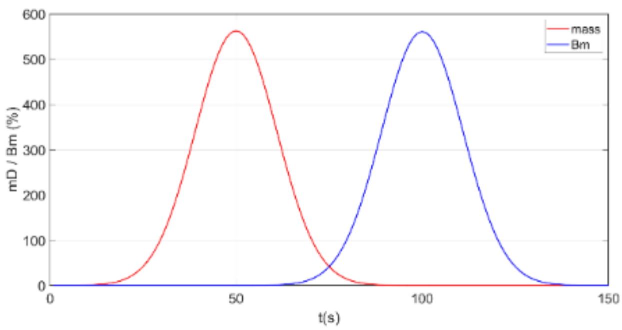

Figura 5(c): Control MRNN. Variación paramétrica

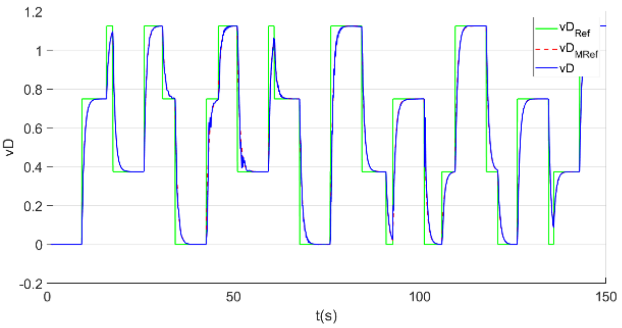

Figura 5(d): Control MRNN. Velocidad de crucero

Hay que destacar que en este trabajo estamos poniendo a prueba únicamente el controlador neuronal y su sintonización con un AG. Es por ello por lo que el modelo neuronal no es actualizado *online* a pesar de que una buena precisión en su predicción es bastante importante para la actualización de los parámetros del neurocontrolador. El efecto de esta no-actualización *online* se puede ver en la respuesta de la red del modelo (línea magenta) en las Figuras 5(a) y 5(b). Idealmente, ante una buena identificación del modelo dinámico por parte del modelo neuronal, las trazas magenta y azul (neuronal y dinámico respectivamente) deberían ser iguales. Esto se cumple siempre y cuando no haya variación de los parámetros. Cuando se dan las variaciones paramétricas, se ve que ambas respuestas divergen considerablemente como consecuencia de que la red neuronal del modelo no ha sido entrenada para identificar el modelo dinámico bajo tales condiciones de trabajo. A pesar de ello el control tiene una respuesta muy buena en lo que respecta a la velocidad de crucero (Figura 5(d)), consiguiendo pues, actualizar sus pesos y sesgos de manera óptima.

Por último, la Tabla 1 recoge los resultados de las métricas mencionadas al inicio de este apartado. Los valores mostrados corroboran numéricamente la mayor eficiencia del control neuronal sobre el PI, con el tiempo de cómputo como única excepción.

Tabla 1. Métricas del error

	PI	MRNN
mse	0.0045	0.0002
mae	0.0357	0.0056
1-R² (%)	98.74%	99.96%
var	0.0045	0.0002
time	1.9 μs	4.6 ms

6. Conclusiones y trabajos futuros

En este trabajo se llevó a cabo el desarrollo y simulación de un sistema de control de velocidad basado en redes neuronales para un Vehículo de Guiado Autónomo (AGV). Se construyó un modelo matemático que describe la dinámica del robot, y se diseñó un controlador neuronal con modelo de referencia. Sus hiperparámetros de aprendizaje fueron sintonizados mediante un algoritmo genético.

Una vez finalizada la estrategia de control, se llevaron a cabo simulaciones con el fin de evaluar y comparar los controladores implementados. Cada uno de ellos fue sometido a diferentes señales de referencia. La robustez de los controladores se puso a prueba modificando los valores de los parámetros de masa y fricción viscosa. Los resultados obtenidos demuestran una mayor eficiencia del controlador neuronal en comparación con el controlador convencional PI. Esto se debido gracias a su adaptabilidad y mayor capacidad para rechazar perturbaciones, lo que le brinda una mayor robustez. Los resultados de los experimentos han sido obtenidos en simulación bajo condiciones ideales; para la implementación en un caso práctico habría que considerar también la presencia de ruido en el sistema.

Como trabajo futuro, proponemos utilizar un modelo dinámico más sofisticado, probando otras configuraciones de red neuronal, y llevando a cabo también una optimización más exhaustiva del controlador mediante un algoritmo de búsqueda evolutiva con una configuración más compleja.

Referencias

Sánchez, R., Sierra-García, J. E., Santos, M.: Modelado de un AGV híbrido triciclo-diferencial. Revista Iberoamericana de Automática e Informática Industrial, 19(1), 84-95, (2022).

Sierra-García, J. Enrique, Santos, M.: Mechatronic Modelling of Industrial AGVs: A Complex System Architecture. Complexity, vol. 2020, 21 pages, 2020.

Abajo, M. R., Sierra-García, J. E., Santos, M.: Evolutive tuning optimization of a PID controller for autonomous path-following robot. In Int. Soft Computing Models in Industrial and Environmental Applications (pp. 451-460). Springer (2021).

Argente-Mena, J., Sierra-García, J.E., Santos Peñas, M. (2023). Robust velocity control of an automated guided vehicle using artificial neural networks. In: 17th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2022). Lecture Notes in Networks and Systems, vol 531. Springer. https://doi.org/10.1007/978-3-031-18050-7_44

Klancar, G., Zdesar, A., et al: Wheeled Mobile Robotics 1st Edition", Butterworth-Heinemann, Oxford, UK, (2017).

Espinosa, F., Santos, C., Sierra-García, J. E.: Transporte multi-AGV de una carga: estado del arte y propuesta centralizada, Revista Iberoamericana de Automática e Informática Industrial, 18(1), 82-91, (2021).

Sierra-García, J.E., Santos, M.: Control of Industrial AGV Based on Reinforcement Learning. International Workshop on Soft Computing Models in Industrial and Environmental Applications. Springer, Cham, (2020).

Hagan, M. T, B Demuth, H. et al: Neural Network Design 2nd Edtion, Martin Hagan, (2014).

Slama, A., Errachdi, A., et al: A Neural Model Reference Adaptive Controller Algorithm for Nonlinear Systems, Automation Research Laboratory, Tunis El Manar University, ENIT, Department of Electrical Engineering, Tunissa, (2017).

Whitley, D. A genetic algorithm tutorial. Stat Comput 4, 65–85 (1994). https://doi.org/10.1007/BF00175354.

Darrell W.:An overview of evolutionary algorithms: practical issues and common pitfalls, Information and Software Technology, Volume 43, Issue 14 (2001) https://doi.org/10.1016/S0950-5849(01)00188-4.

Sierra-Garcia, J. E., Santos, M.: Combining reinforcement learning and conventional control to improve automatic guided vehicles tracking of complex trajectories. Expert Systems, e13076, (2022). https://doi.org/10.1111/exsy.13076

Chang GC, Luh JJ, Liao GD, Lai JS, Cheng CK, Kuo BL, Kuo TS. A neuro-control system for the knee joint position control with quadriceps stimulation. IEEE Trans Rehabil Eng. 1997 Mar;5(1):2-11. PMID: 9086380

Sierra-García, J. E., Santos, M.: Switched learning adaptive neuro-control strategy, Neurocomputing, Volume 452, Pages 450-464, (2021).

K. KrishnaKumar, S. Rickard, S. Bartholomew,:Adaptive neuro-control for spacecraft attitude control, Neurocomputing, Volume 9, Issue 2 (1995) https://doi.org/10.1016/0925-2312(94)00062-W.