ISSN: 1692-7257 - Volumen 3 – Número Especial - 2020

Revista Colombiana de Tecnologías de Avanzada

Recibido: 3 de marzo de 2020
Aceptado: 3 agosto de 2020

IMPLEMENTACIÓN Y SIMULACIÓN DE UN ALGORITMO DE POSICIONAMIENTO ARTICULAR PARA UN ROBOT PLANAR CONTINUO UTILIZANDO TÉCNICAS DE INTELIGENCIA ARTIFICIA

IMPLEMENTATION AND SIMULATION OF A JOINT POSITION ALGORITHM FOR A CONTINUOUS PLANAR ROBOT USING ARTIFICIAL INTELLIGENCE TECHNIQUES

PhD. Andrés Ricardo Castillo[*], Ing. Fabián Camilo Castro[**]
Ing. Joseph Jonás Vogulys[*,]**

[*]**Universidad Militar Nueva Granada,** Facultad de Ingeniería, Programa de Mecatrónica.
Carrera 11 n.° 101-80, Bogotá, Colombia.
E-mail: {ricado.castillo, u3900228}@unimilitar.edu.co.
[**] **Universidad Militar Nueva Granada,** Facultad de Ingeniería, Grupo de investigación Davinchi.
Carrera 11 n.° 101-80, Bogotá, Colombia.
E-mail: {u3900234}@unimilitar.edu.co.

Resumen: En este trabajo se muestra la implementación de un algoritmo de Inteligencia Artificial para posicionar las articulaciones de un robot planar tipo continuo hiperrebundate para que el robot puede generar una curvatura y tenga la habilidad de esquivar obstáculo dinámicos. Este trabajo se desarrolló con el framework ROS y se simulo sobre una plataforma virtual de un entorno para robótica.

Palabras clave: Inteligencia Artificial**,** Robot Continuo, Hiperrrebundancia.

Abstract: This work shows the implementation of an Artificial Intelligence algorithm to position the joints of a hyperrebundate continuous type planar robot so that the robot can generate a curvature and have the ability to dodge dynamic obstacles. This work was developed with the ROS framework and was simulated on a virtual platform of an environment for robotics.

Keywords: Artificial Intelligence, Continuous Robot, Hyperabundance.

1. INTRODUCCIÓN

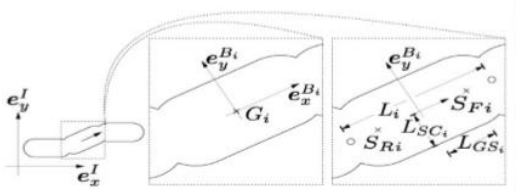
Los robots continuos son una nueva clase de robots que pueden definirse como sistemas adaptables a diferentes entornos, a diferencia de los robots convencionales, los robots continuos permiten una mayor flexibilidad y aplicación para llevar a cabo tareas con condiciones de restricción de espacio [1]. Estos robot poseen múltiples grados de libertad permitiendo que en su locomoción tengan un alto grado de manipulación y destreza en ambientes de difícil acceso y espacios muy cerrados donde la maniobrabilidad debe ser alta, estas características son muy similares a las exhibidas por la locomoción de las serpientes, trompas de elefantes y tentáculos de pulpo, capacidades más allá del alcance de manipuladores

tradicionales de enlace rígido [2]. Las posibles aplicaciones de este tipo de robots incluyen la navegación a través de entornos congestionados e impredecibles en los que un robot continuo puede usarse para la exploración subterránea o subacuática, los robots continuos con sus habilidades únicas puede llegar a lugares que generalmente son inaccesibles para robots de enlaces rígidos y hostiles para seres humanos.

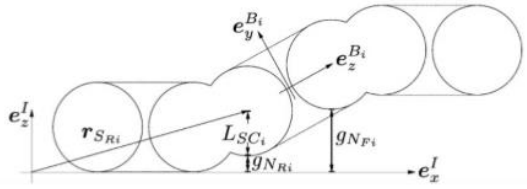
Un tema de investigación importante para estudiar los robots continuos es el cálculo de modelos cinemáticos. Los enfoques tradicionales de modelado, en donde los marcos de referencia están asociados con cada articulación, son inapropiados para este caso, debido a la ausencia de enlaces discretos y rígidos en su arquitectura, la complejidad del problema que reside en el acoplamiento de variables operacionales, la hiper-rebundancia no permite que pueden controlarse fácilmente al considerar un número alto de grados de libertad que para los robot de hiper-rebundancia son mayores a 20 grados. Esto genera el posicionamiento articular sea difícil de realizar y no sea óptima admitiendo así un conjunto reducido de soluciones físicas [3]. Este trabajo de investigación surge a la necesidad de generar una estrategia para el posicionamiento de este tipo de robot, con el fin de que el robot pueda esquivar obstáculos aprendiendo del entorno por medio de redes neuronales que pueden ser una herramienta poderosa para superar esas no linealidades que a menudo son difíciles de encontrar y capturar en un proceso de modelamiento cinemático.

2. CREACIÓN DEL MUNDO VIRTUAL

Se diseñó un esquemático de un robot planar continuo tipo discreto que consiste en n eslabones cilíndricos de igual tamaño que están conectados por $n-1$ articulaciones esféricas. Cada articulación posee dos grados de libertad que se mueven de acuerdo a un sistema de coordenadas cartesianas. La longitud de cada cilindro que forma los eslabones son L_i siendo i el número del eslabón, esta distancia es igual al tamaño de la distancia que hay entre centros de cada articulación cuando estos están de forma tangente, el radio de cada articulación es L_{SCi} y cada eslabón es modelado como un cilindro de diámetro $2L_{SCi}$, las dos esferas de radio L_{SCi} son fijas en los extremos de cada eslabón. La fuerza en los actuadores se simulo mediten un servo motor que controla el ángulo de giro. En la figura 1 se puede observar el marco de referencia con los parámetros seleccionados a través de una vista frontal y superior



a) Esquemático del robot desde una vista superior. (Plano x-y)



b) Esquemático del robot desde una vista frontal. (Plano z-x)

Fig. 1. Marco de referencia y diseño esquemático de un robot planar tipo continuo

El mundo virtual se desarrolló en el software de GAZEBO, el cual se puede conectar de manera fácil con ROS y de esta forma poder manipular señales de sensores y actuadores en tiempo real. En la figura 2 se puede observar la arquitectura del software implementado

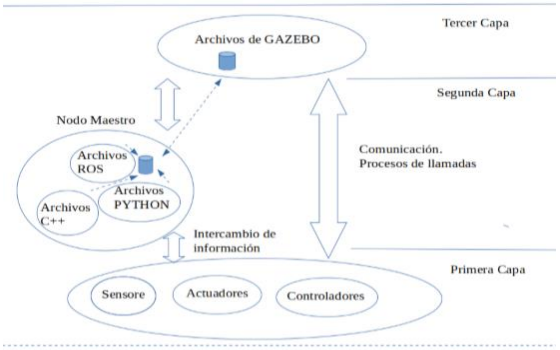


Fig. 2. Arquitectura del software implementado comunicación ROS Y GAZEBO

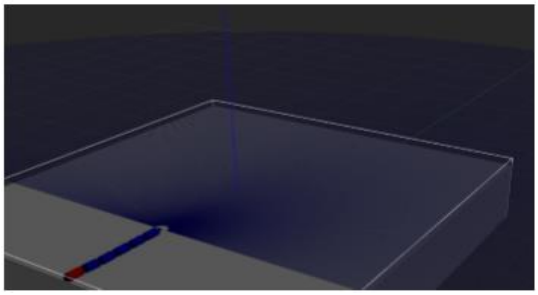
3. Algoritmo de Inteligencia Artificial: Arquitectura de la Red Neuronal

Se implementa un modelo perceptron multicapa (MPL), de aprendizaje tipo off- line, esta requiere de una fase previa de entrenamiento con un dataset de entrenamiento y un dataset de test o

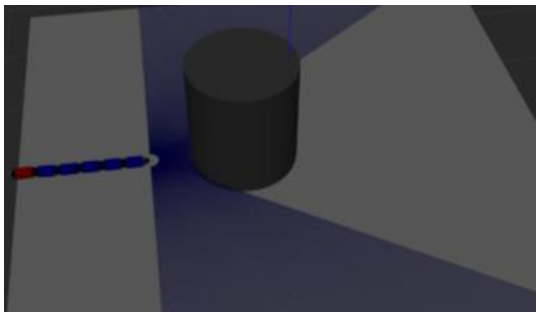
prueba; igualmente los pesos de las conexiones no se modifican después de terminar la etapa de entrenamiento de la red esto permite un fácil reconocimiento de situaciones presentadas por el vector de entrada y la evaluación de la proximidad a situaciones límite dados por el vector de prueba.

3.1 Obtención de los Datos en el Mundo Virtual

Para los datos de entrada en la red neuronal MPL, el robot continuo en el último eslabón tiene un sensor de tipo láser Lidar que captura 480 puntos en un área de trabajo de 180 grados del mundo virtual como se puede observar en la figura 4.



a) Ángulo de trabajo del sensor capturando 480 puntos



a) Obstáculo detectado dentro del espacio de trabajo del robot

Fig. 4. Simulación del robot continuo con un sensor Lidar

Los obstáculos detectados mediante la frontera de protección o espacio de trabajo del sensor se denominan burbuja, cuyas dimensiones dependen de la velocidad del robot y del intervalo de tiempo entre lecturas de láser. Cuando un obstáculo penetra en la frontera de la burbuja, el robot calcula el llamado ángulo de rebote, en la dirección en la que haya menor densidad de obstáculos, el método de cálculo del ángulo de rebote es mediante una media ponderada, en la que el ángulo es el dato y la distancia leída por el láser. En el ángulo de rebote se calcula:

$$\alpha_r = \frac{\sum_0^{180} \alpha_i D_i}{\sum_0^{180} D_i} \quad (1)$$

Donde Di es la lectura del láser correspondiente al ángulo αi. Uno de los mayores inconvenientes es el cálculo del ángulo de rebote, si resulta que hay un obstáculo bloqueando el camino y éste se encuentra a un lado del robot, el ángulo de rebote resultará ser un ángulo hacia el lado de menor densidad de obstáculos, que es el resultado esperado, el problema de este de este método surge cuando la distribución de obstáculos está bastante centrada. En este caso la media ponderada dará como resultado un ángulo que conducirá al robot directo hacia el obstáculo. En la figura 5 se puede observar el diagrama de medida ponderara del a distribución del sensor.

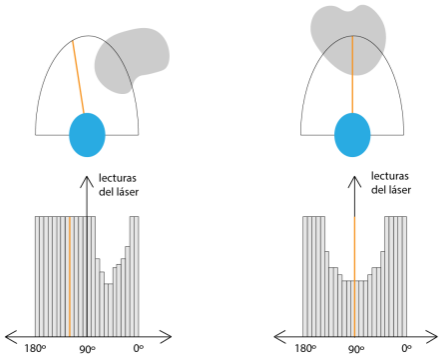


Fig. 5. Distribución pondera del sensor Lidar

Para la implementación de los valores de los ángulos a la red neuronal, se genera un vector con la información de cinco ángulos ponderados, ya que estos son los puntos que detecta el robot cuando esta de frente un obstáculo sin importar su forma, los datos de salida serán los ángulos de cada eslabón dados en un rango entre $(\frac{\Pi}{2}, -\frac{\Pi}{2})$.

3.2 Obtención de los Datos de Entrenamiento

Para obtener el dataset que se utilizó para entrenar la red, se generó previamente trayectorias de curvas con el algoritmo de Bézier [6], el número de puntos de control igual al número de eslabones, una vez generada la curva, se subdivide de acuerdo a los ángulos permitidos por cada articulación para

obtener los valores de entrenamiento. En la obtención del dataset de entrenamiento se generaron 50 curvas en el software de Matlab. En la figura 6 se puede ver cuatro curvas de Bézier generadas con ocho puntos de control de acuerdo al número de eslabones propuesto para la realización de la pruebas.

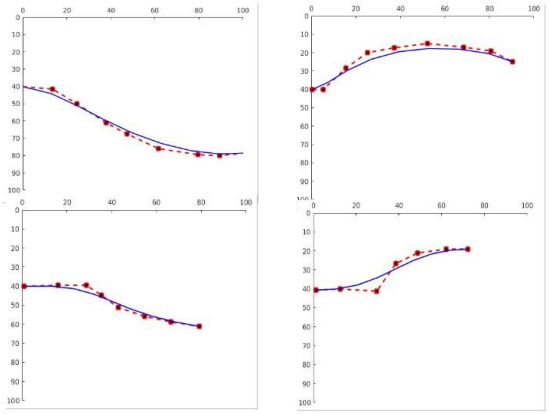


Fig. 6. Curvas de Bézier generadas con ocho puntos de control

Para obtener los datos de los sensores se realizó un obstáculo dinámico en forma de un robot móvil diferencial controlado a través de una interfaz gráfica. Los robots móviles son capaces de realizar movimientos en espacios de trabajo ilimitados en comparación a los robots manipuladores y uno de sus principales objetivos es efectuar de forma autónoma movimientos planificados con gran exactitud. El robot móvil tiene como función entrenar el manipulador robótico posicionándose en diferentes lugares y con diferente orientación. En la figura 7 se puede observar el robot diferencial en una posición inicial y en la figura 8 se puede observar al robot girado 20 grados a la izquierda

3.3 Fase de entrenamiento

En la fase de entrenamiento y en su modelo de aprendizaje se utilizó el método del gradiente descendiente y algoritmo Gauss Newton [6], este método también es conocido como Gradiente Conjugado que puede considerarse como un método intermedio entre el Descenso del Gradiente y el método de Newton. La ventaja de utilizar este algoritmo es poder acelerar la convergencia del entrenamiento, habitualmente lenta, obtenida con el Descenso del Gradiente, y a la vez evitar los requisitos de computación asociados a la evaluación, almacenamiento e inversión del Hessiano, como requiere el método de Newton. El modelo fue entrenado para tener un pequeño sobreajuste, esto con el fin de que el robot pudiera predecir de forma correcta la evasión de obstáculos a situaciones que nunca ha visto y no quede limitado solo a soluciones que se generaron en el entrenamiento. Para el aprendizaje se generaron tres arquitecturas de redes neuronales:

3.3.1 Primera Arquitectura

Este primer entrenamiento se realiza con una red neuronal de 20 neuronas con 4 capas; dos ocultas, una capa de entrada y una de salida, se seleccionó 20 neuronas porque es el mínimo de neuronas que la red tiene un comportamiento donde el error va convergiendo a cero. Se puede observar que la red tiene un comportamiento en la validación del entrenamiento del 50% y el error tiene un porcentaje del 1%. En la figura 7 los resultados de la red neuronal en la primera arquitectura

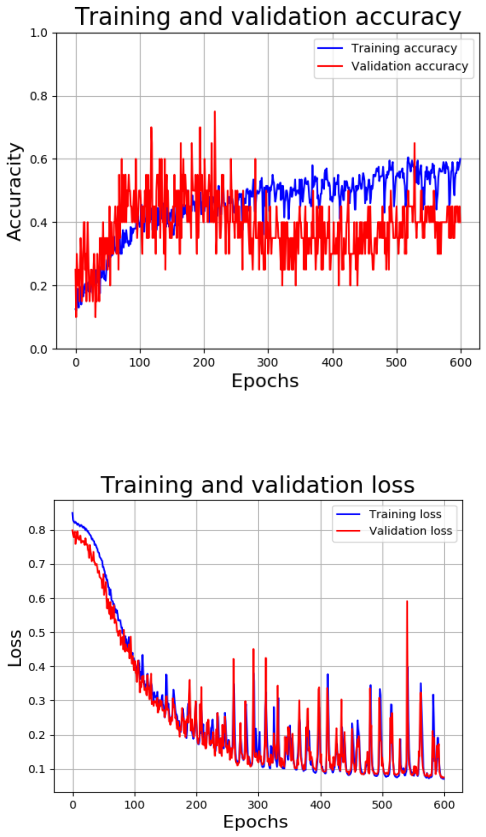


Fig. 7. Resultados de una red neuronal con 4 capas de 20 neuronas cada capa

3.3.2 Segunda Arquitectura

El segundo entrenamiento que se realiza con una red neuronal de 20 neuronas con 8 capas; 6 ocultas, una capa de entrada y una de capa de salida, se puede observar con respecto a la red anterior un comportamiento mejor en cuanto a entrenamiento y el error, tiene un comportamiento en la validación del entrenamiento en promedio con 90% ya que oscila entre el 82% y 100% el error tiene un porcentaje del 0.02%. En la figura 8 se puede observar el comportamiento de la segunda arquitectura de red neuronal.

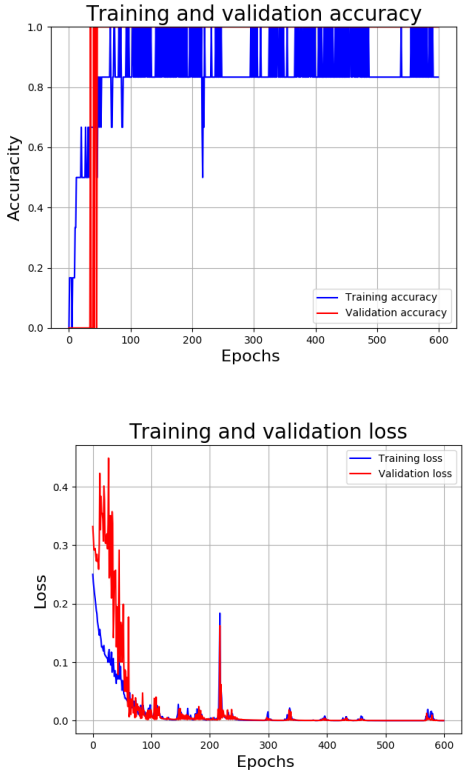


Fig. 8. Resultados de una red neuronal con 8 capas de 20 neuronas cada capa

3.3.3 Tercera Arquitectura

El tercer entrenamiento que se realiza con una red neuronal de 20 neuronas con 16 capas; 8 ocultas, una capa de entrada y una de capa de salida, se puede observar con respecto a la red anterior un comportamiento similar en cuanto a entrenamiento, pero esta arquitectura converge más lento a la arquitectura anterior y tiene una oscilación mayor, con respecto al error se puede decir que tiene error mayor al 0.02%. En la figura 9 se puede observar se puede observar el comportamiento de la Tercera arquitectura de red neuronal.

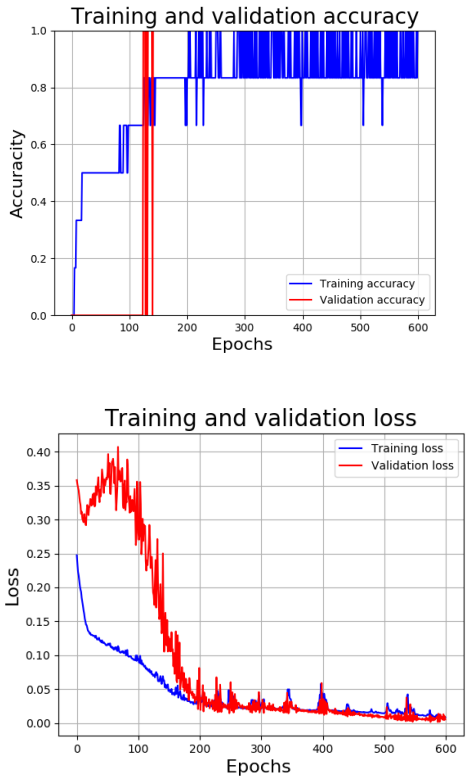


Fig. 9. Resultados de una red neuronal con 16 capas de 20 neuronas cada capa

4. Simulación y Resultados

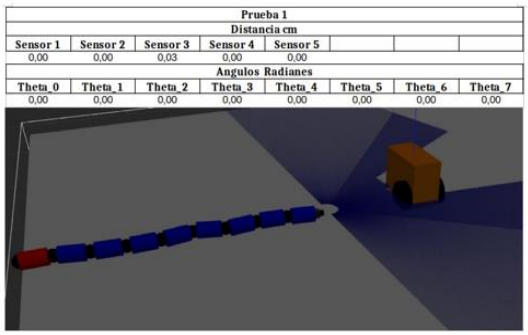
El objetivo de esta prueba es demostrar mediante simulación, la confiabilidad del algoritmo MPL para la planeación de trayectorias de un robot planar continuo tipo discreto evadiendo obstáculos una vez entrenado para este tipo de escenarios, la idea central es el que el robot sea capaz de auto configurarse evadiendo los obstáculos, en este caso el obstáculo es un robot móvil tipo diferencial en cual puede desplazarse por la zona de detección del sensor lidar.

4.1 Escenarios y pruebas elegidos

En las pruebas 1 hasta la 4 se utilizó un robot de ocho eslabones y como indicadores se observó la salida de la red que para este caso son los ángulos de cada eslabón.

4.1.1 Prueba 1 con un robot de ocho eslabones implementado algoritmo MPL

En la prueba 1 el robot móvil se coloca enfrente de sensor del robot en este caso solo el sensor 3 puede detectar el obstáculo como se puede apreciar en la figura 10, en este caso la salida se puede observar los ángulos de cada eslabón, el método de burbuja donde el obstáculo entra en el espacio del sensor, el robot debe irse de frente hacia el obstáculo, en este caso el robot como es un manipulador se queda en su posición inicial.

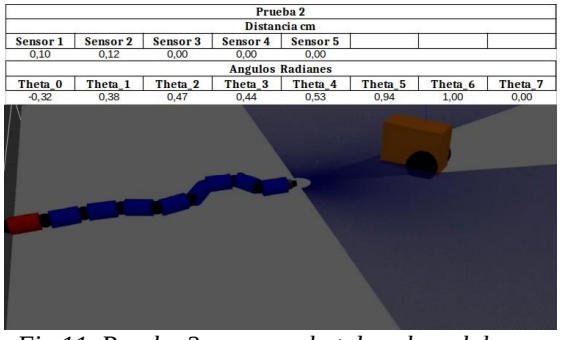


Prueba 1							
Distancia cm							
Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5			
0,00	0,00	0,03	0,00	0,00			
Angulos Radianes							
Theta_0	Theta_1	Theta_2	Theta_3	Theta_4	Theta_5	Theta_6	Theta_7
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

Fig.10. Prueba 1 con un robot de ocho eslabones implementado algoritmo MPL

4.1.2 Prueba 2 con un robot de ocho eslabones implementado algoritmo MPL

En la prueba 2 el robot móvil se inclina hacia la derecha a 20 grados donde solo los puntos 1 y 2 del sensor del manipulador pueden detectar los valores obtenidos en la figura 11 son los valores que el sensor registra en la posición donde se encuentra el robot. En esta prueba el obstáculo entra en el espacio de trabajo del sensor hacia el lado derecho y está siendo detectado por el método de burbuja, lo que hace que el robot intenta contraerse para esquivarlo esto simulando una compresión en robot planar continuo

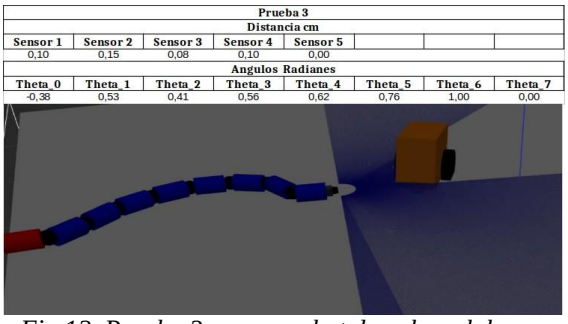


Prueba 2							
Distancia cm							
Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5			
0,10	0,12	0,00	0,00	0,00			
Angulos Radianes							
Theta_0	Theta_1	Theta_2	Theta_3	Theta_4	Theta_5	Theta_6	Theta_7
-0,32	0,38	0,47	0,44	0,53	0,94	1,00	0,00

Fig.11. Prueba 2 con un robot de ocho eslabones implementado algoritmo MPL

4.1.3 Prueba 3 con un robot de ocho eslabones implementado algoritmo MPL

En la prueba 3 el robot móvil se inclina hacia la derecha -20 grados donde solo los puntos 4 y 5 del sensor del manipulador pueden detectar. En esta prueba el obstáculo entra en el espacio de trabajo del robot y está siendo detectado por el método de burbuja que es la parte derecha del ángulo de trabajo del sensor lo que hace que el robot intenta contraerse para esquivarlo pero en contraste generó una curva de mayor grado con respecto a la prueba anterior. En la figura 12 se puede observar la curvatura que robot realiza.

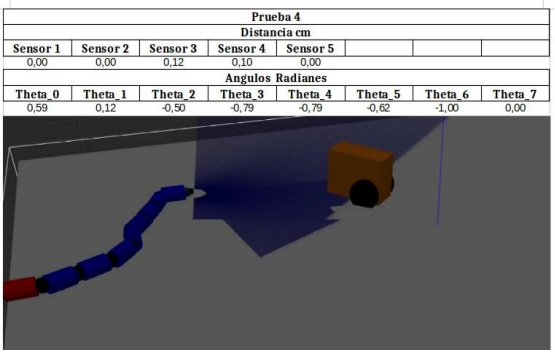


Prueba 3							
Distancia cm							
Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5			
0,10	0,15	0,08	0,10	0,00			
Angulos Radianes							
Theta_0	Theta_1	Theta_2	Theta_3	Theta_4	Theta_5	Theta_6	Theta_7
-0,38	0,53	0,41	0,56	0,62	0,76	1,00	0,00

Fig.12. Prueba 3 con un robot de ocho eslabones implementado algoritmo MPL

4.1.4 Prueba 2 con un robot de ocho eslabones implementado algoritmo MPL

En la prueba 4 el robot móvil se acerca manteniendo la inclinación hacia la derecha de 20 grados donde solo los puntos 1 y 2 del sensor pueden detectar. En la figura 13 se puede observar el comportamiento del robot planar mientras se acerca el robot móvil y este va generando una curvatura abierta para no tocar el obstáculo



Prueba 4							
Distancia cm							
Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5			
0,00	0,00	0,12	0,10	0,00			
Angulos Radianes							
Theta_0	Theta_1	Theta_2	Theta_3	Theta_4	Theta_5	Theta_6	Theta_7
0,59	0,12	-0,50	-0,79	-0,79	-0,62	-1,00	0,00

Fig.13. Prueba 4 con un robot de ocho eslabones implementado algoritmo MPL

4. CONCLUSIONES

Al observar el comportamiento de la red neuronal en un robot de ocho eslabones se puedo que generar curvas brusca y no siempre generaba una posicionamiento correcta ya que aunque se

aumenta el número de neuronas los datos de los sensores son muy parecidos en las diferentes posicionamientos lo cual no hace posible una correcta separación y clasificación por parte de la red neuronal.

El posicionamiento de robot ante la respuesta de la red neuronal también se ve afecta por la reducción de puntos de un espacio de 480 puntos a cinco puntos de ponderación lo cual son pocos datos para entrenar una red neuronal, sim embargo el objetivo de evadir el obstáculo y no tocarlo se cumplió pero con movimientos brucos generando curvaturas muy abiertas.

En este trabajo se puedo observar que una arquitectura tipo MPL no es la más adecuada y se deja abierta la posibilidad de implementar otros tipos de arquitectura como LSTM o algoritmos por reforzamiento y compara con el MPL obtenido en este trabajo.

REFERENCIAS

M.Cecilia. (Junio 2017). DESARROLLO DE UN ROBOT MANIPULADOR BLANDO E HÍPER-REDUNDANTE. PhD Escuela Técnica Superior de Ingenieros Industriales. Universidad Politécnica de Madrid.

Kapadia, A. D., Walker, I. D., Dawson, D. M., and Tatlicioglu, E. A model-based sliding mode controller for extensible continuum robots. In Proceedings of the 9th WSEAS International Conference on Signal Processing, Robotics and Automation, IEEE, 11(6), 45-52. Stevens Point,Wisconsin, (USA, 2010), ISPRA?10, World Scientific and Engineering Academy and Society (WSEAS), pp. 113−120

Hannan, M. W., and Walker, I. D. Analysis and experiments with an elephant's trunk robot.. Advanced Robotics 15, 8 (2001), 847−858..

Jones, B. A., and Walker, I. D. Practical kinematics for real-time implementation of continuum robots.IEEE Transactions on Robotics 22, 6 (2006), 1087?1099

Gokhale, D. P. Kinematic analysis and animation of a variable geometry truss robot.PhD thesis, Virginia Tech, 1987.

Barrio, A. M., Terrile, S., Barrientos, A., and del Cerro, J. Robots hiper-redundantes: Clasificación, estado del arte y problemática.Revista Iberoamericana de Automática e Informática industrial 15, 4 (2018), 351-362.

Mishra, A., Mondini, A., Del Dottore, E., Sadeghi, A., Tramacere, F., and Mazzolai, B. Modular continuum manipulator:analysis and characterization of its basic module. Biomimetics3, 1 (2018), 3

Zhao, L., Xiao, Q., Cao, Z., Huang, R., and Fu, Y. daptive neural network tracking control of snake-like robots via a deterministic learning approach.Revista n 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO) (Dec 2017), pp. 2710-2715.

León, J. F. Estudio de neuro-controladores evolutivos para navega- ción de robots autónomos. Master of Systems Engineering, UNCPBA, Argentina (2005).

Wu, X., and Ma, S. Cpg-based control of serpentine locomotion of a snake-like robot. Mechatronics 20, 2 (2010), 326-334

Brockett, R. W obotic manipulators and the product of exponentials formula. In Mathematical theory of networks and systems.(1984), Springer, pp. 120-129.

Walker, I. D. Continuous backbone continuum robot manipula- tors.Isrn robotics 2013 (2013).

Transeth, A., Leine, R., Glocker, C., and Pettersen, K. 3-D Snake Robot Motion: Nonsmooth Modeling, Simulations, and Experiments. IEEE Transactions on Robotics 24, 2 (Apr. 2008), 361?376..

Crespi, A., Badertscher, A., Guignard, A., and Ijspeert, A. J. Amphibot i: an amphibious snake-like robot. Robotics and Autonomous Systems 50, 4 (2005), 163-175.

Chollet, F. Deep Learning mit Python und Keras: Das Praxis- Handbuch vom Entwickler der Keras-Bibliothek. MITP-Verlags GmbH and Co. KG, 2018.

Banik, M. S., and Couvillon Jr, L. A. Robotic endoscope with wireless interface, Aug. 3 2004. US Patent 6,770,027..

Nishihara, T., Osuka, K., and Tamura, I.

Development of a simulation model for inner-gas-pipe inspection robot: Spring. In Pro- ceedings of SICE Annual Conference 2010 (2010), IEEE, pp. 902-904..

Kitamura, S., and Oka, K. Recognition and cutting system of sweet pepper for picking robot in greenhouse horticulture. In IEEE International Conference Mechatronics and Automation, 2005 (2005), vol. 4, IEEE, pp. 1807?1812.

Singer, P. W. Military robots and the laws of war. The New Atlantis, 23 (2009), 25?45.

Garcia, A., Gonzalez, I., Colomo-Palacios, R., Lopez, J. L., and Ruiz, B. Methodology for software development estimation optimization based on neural networks. IEEE Latin America Transactions 9, 3 (2011), 384?398..

Florez Vergara, D. E., Castro Riveros, F. C., and Casti- llo Estepa, R. A. Planeacion y ejecucion de trayectorias en un robot Delta. Scientia et technica 22, 2 (June 2017), 186.

Guo, Y., Kang, R., Chen, L., and Dai, J. Dynamic modeling for a continuum robot with compliant structure. In ASME 2015 In- ternational Design Engineering Technical Conferences and Computers and Information in Engineering Conference (2015), American Society of Mechanical Engineers Digital Collection.

J. Minguez, J. Osuna y L. Montano, «A "divide and conquer" strategy based on situations to achieve reactive collision avoidance in troublesome scenarios» de IEEE International Conference on Robotics and Automation, New Orleans, 2004..

Amari, S.-i. Backpropagation and stochastic gradient descent method. Neurocomputing 5, 4-5 (1993), 185?196.

López Garc?á, D. A., et al. Nuevas aportaciones en algoritmos de planificación para la ejecución de maniobras en robots autónomos no holónomos..