

IDENTIFICACIÓN DE UN SISTEMA ROBÓTICO MEDIANTE REDES NEURONALES DINÁMICAS

S. Torres*, V.M. Becerra**, J.A. Méndez*, L. Acosta*, G.N. Marichal*, A.F. Hamilton*, M. Sigut*.

* Grupo de Computadoras y Control. Departamento de Física, Electrónica y Sistemas.

Universidad de La Laguna. 38271 La Laguna, Tenerife, ESPAÑA.

E-mail: {storres,jamendez,lacosta,nicomar,albham,marsigut}@ull.es

** Department of Cybernetics,

The University of Reading, Whiteknights, Reading RG6 6AY, Berkshire, UNITED KINGDOM.

E-mail: v.m.becerra@reading.ac.uk

Resumen

El objetivo de este trabajo es la identificación de un sistema robótico mediante el uso de redes neuronales dinámicas. El sistema en cuestión es una pierna formada por dos articulaciones de revolución. El modelo teórico del sistema corresponde por tanto al de una cadena cinemática de dos articulaciones, pero el gran número de no linealidades no contempladas en dicho modelo hacen que existan bastantes discrepancias con el modelo real. Por esta razón se ha optado por obtener un modelo empírico del mismo, empleándose para ello redes neuronales dinámicas recurrentes. Se han aplicado diversas técnicas novedosas de entrenamiento y validación, que aseguran un buen comportamiento del modelo empírico buscado, consistentes en la inclusión del estado de las neuronas escondidas de la red en el vector de decisión -en el caso de entrenamiento-, así como la resolución de un problema de optimización previo para la inicialización de dichos estados en validación. Los datos experimentales han sido obtenidos de un sistema real.

Palabras Clave: Identificación de Sistemas, Redes Neuronales Dinámicas, Robótica.

1 INTRODUCCIÓN

Las ecuaciones de la dinámica de un robot formado por una cadena articulada de elementos rígidos unidos por articulaciones se obtienen habitualmente de la formulación matricial de Lagrange-Euler o de la recurrente de Newton-Euler. Otras formulaciones como la recursiva de Lagrange o la generalizada de d'Alambert-Lee pueden ser utilizadas para el mismo fin. De cualquiera de las formas, el resultado de las mismas es un conjunto de ecuaciones no lineales y fuertemente acopladas, denominadas ecuaciones de la dinámica inversa del robot [1].

Dichas ecuaciones relacionan el par de fuerzas o torque aplicado sobre cada una de las articulaciones con las coordenadas generalizadas del robot, ya sean ángulos –en el caso de articulaciones de revolución– o desplazamientos –en el caso de articulaciones prismáticas–. Los términos que dependen de dichas coordenadas se refieren a términos de inercia (pares de reacción inducidos por la aceleración que una articulación provoca sobre otra), a las aceleraciones centrífuga y de Coriolis, y a términos gravitatorios debido a los elementos que componen la cadena articulada.

Dichas ecuaciones también pueden contener términos que reflejan los rozamientos y fricciones presentes en el sistema. Estos términos dependen de parámetros de difícil evaluación, como pueden ser coeficientes de rozamiento viscoso o coulombiano, a diferencia de los términos anteriores, que precisan el conocimiento de las masas, longitudes y momentos de inercia de cada uno de los elementos. Las fricciones, pues, añaden incertidumbre en el modelo calculado. Además, existen otras no linealidades que solo aparecen en la construcción del prototipo mecánico, como pueden ser holguras en las articulaciones, zonas muertas de actuación de los motores, etc. Estas características son difícilmente modelables de forma teórica por lo que se recurre a modelos empíricos para llevar a cabo la identificación completa del sistema.

El uso de redes neuronales es una herramienta habitual en el campo de la identificación de sistemas dinámicos. Una red de estructura determinada es entrenada con datos de entrada-salida del sistema real a identificar para que aprenda el comportamiento dinámico del sistema. La red neuronal, por definición, realiza un mapeo no lineal entre los datos de entrada y salida, por lo que resultan muy convenientes para aprender la dinámica de sistemas no lineales, tal y como se han venido utilizando en las últimas dos décadas.

En un principio, se emplearon redes estáticas tales como el perceptrón multicapa (MLP) [2],[3]. Las entradas de estas redes corresponden a valores pasados de las entradas y salidas del sistema. Sin embargo, esta aproximación presenta algunos inconvenientes. El primero de ellos es la dificultad para elegir la estructura de entrada a la red. Otro inconveniente es que la estructura debe ser reentrenada cada vez que se cambie el período de muestreo del sistema. Además, el problema de control no lineal se entiende mejor cuando se formula en tiempo continuo que en la versión discreta.

Las redes neuronales dinámicas (DNN) de Hopfield [4], y sus variantes [5][6][7][8], no presentan estos inconvenientes y son capaces de aproximar el comportamiento de sistemas dinámicos multi-variables. Las DNNs pueden aproximar el comportamiento de sistemas no lineales autónomos [9][10], sistemas afines a control multivariable [11] y sistemas no lineales en general [12]. Las condiciones de estabilidad para las DNN se han analizado en [13] y [14].

Recientemente, Becerra *et al* [15] han introducido algunas técnicas para la inicialización de los estados de las neuronas escondidas, para propósitos de entrenamiento y validación de redes neuronales con esta estructura. En particular, dicho trabajo (sobre el que se basa el presente artículo) propone, en el caso de entrenamiento, incluir dichos estados en el vector de parámetros de la DNN a optimizar, así como inicializar estos estados de forma eficaz para el caso de validación. Así, se propone resolver un problema de optimización consistente en calcular los estados de dichas neuronas que conducen a igualar las derivadas de los estados de salida a las derivadas medidas empíricamente.

Con estas técnicas se resuelve uno de los problemas que plantea este tipo de estructuras neuronales para el caso de la identificación de sistemas, ya que la inicialización de los estados de las neuronas escondidas se ha hecho de forma aleatoria o, más habitualmente, igualándolos a cero.

En el presente artículo se desarrolla esta idea para la inicialización de la DNN, aplicada a la identificación de un sistema robótico consistente en una pierna mecánica con dos grados de libertad. Para que la identificación del sistema mejore, se ha añadido un término a la función de costo a optimizar de forma que impida que los estados de las neuronas escondidas adquieran valores elevados.

El artículo está organizado como sigue. En la sección 2 se introducen las DNNs. La sección 3 describe el problema de entrenamiento de la DNN, mientras que en la sección 4 se explican los aspectos de

inicialización de las mismas. En la sección 5 se describe el sistema a identificar y en la sección 6 se exponen los resultados obtenidos. Finalmente, en la sección 7 se incluyen las principales conclusiones de este trabajo.

2 REDES NEURONALES DINÁMICAS

Las redes neuronales dinámicas (DNN) en su forma recurrente fueron introducidas en el contexto de las memorias asociativas [5][16]. La realimentación añadida a la arquitectura de una red neuronal *feedforward* permite obtener un modelo dinámico en el espacio de los estados.

Dicho modelo está compuesto por un vector unidimensional de neuronas, cada una de las cuales viene descrita por:

$$\dot{x}_i = -\beta_i x_i + \sum_{j=1}^n \omega_{ij} \sigma(x_j) + \sum_{j=1}^p \gamma_{ij} u_j \quad (1)$$

donde β_i , ω_{ij} y γ_{ij} son pesos ajustables, con $1/\beta_i$ constantes de tiempo positivas, x_i el estado de activación de la neurona i , $\{u_i\}_{i=1,\dots,m}$ las entradas a la red, con $p \leq n$, siendo n el número de neuronas de la red. La función $\sigma(\cdot)$ es típicamente una función no lineal tipo sigmoide, por ejemplo, la tangente hiperbólica, aplicada sobre el argumento (\cdot) . Esta disposición puede observarse en la figura 1.

La red está formada por n neuronas, de las cuales se escogen las p primeras como salidas, dejando las $n-p$ restantes como neuronas escondidas. La red puede entonces expresarse de manera matricial como sigue:

$$\begin{aligned} \dot{x} &= -\beta \cdot x + \omega \cdot \sigma(x) + \gamma \cdot u \\ \hat{y} &= C_n x \end{aligned} \quad (2)$$

donde $x \in \mathfrak{R}^n$ es el vector de estado, $u \in \mathfrak{R}^m$ es el vector de entradas, $\omega \in \mathfrak{R}^{n \times n}$, $\sigma(x) = [\sigma(x_1), \dots, \sigma(x_n)]$, $\gamma \in \mathfrak{R}^{n \times m}$, $y \in \mathfrak{R}^p$ es el vector de salida, $C_n = \begin{bmatrix} I_{p \times p} & \emptyset_{p \times (n-p)} \end{bmatrix}$ y $\beta \in \mathfrak{R}^{n \times n}$ es una matriz diagonal con elementos $[\beta_1, \dots, \beta_n]$. Esta estructura matricial puede observarse en la figura 2. Si $n > p$, se tienen $n-p$ neuronas escondidas, las cuales aumentan la capacidad de modelar la dinámica de un sistema [17][18]. El estado de la DNN puede entonces considerarse dividido en los estados de salida $x_o \in \mathfrak{R}^p$ y los estados escondidos $x_h \in \mathfrak{R}^{n-p}$:

$$x = \begin{bmatrix} x_o \\ x_h \end{bmatrix} \quad (3)$$

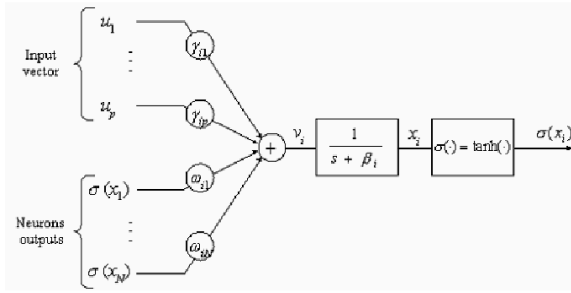


figura 1: Diagrama de una neurona dinámica.

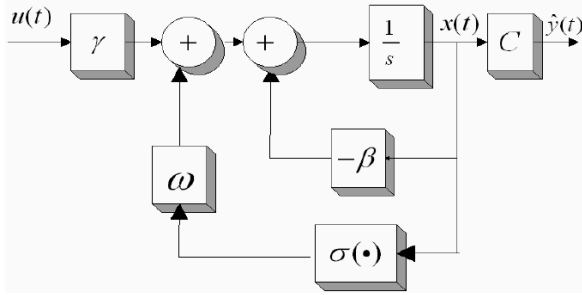


figura 2: Diagrama de la red neuronal dinámica.

3 ENTRENAMIENTO DE LA DNN

Para el ajuste de los parámetros de la DNN se dispone de una serie de datos obtenidos del sistema real que va a ser modelado. Supongamos un conjunto de entrenamiento compuesto por N pares de entrada-salida y tiempo de muestreo T_s :

$$Z_N = [y(t_k), u(t_k)]_{k=1, \dots, N} \quad (4)$$

donde $y \in \mathfrak{R}^p$ es la salida medida, $u \in \mathfrak{R}^m$ es la entrada y k un índice temporal. El problema de entrenamiento puede ser formulado como un problema de optimización. Sea $\theta \in \mathfrak{R}^{n_\theta}$ el vector de decisión, donde n_θ es el número de parámetros a estimar. Si escogemos la función de costo dada por el error cuadrático medio,

$$V_N(\theta, Z_N) = \frac{1}{2N} \sum_{k=1}^N [y(t_k) - \hat{y}(t_k|\theta)]^2 \quad (5)$$

donde $\hat{y}(t_k|\theta)$ es la salida de la DNN dada por (2), el problema de identificación no lineal puede ser descrito por el siguiente problema de optimización sin restricciones:

$$\min_{\theta} V_N(\theta, Z_N) \quad (6)$$

El trabajo de Becerra *et al* [15] propone la inclusión de los valores iniciales de los estados escondidos $x_h(t_0)$ en el vector de decisión θ formado como sigue:

$$\theta = \begin{bmatrix} \beta_d \\ \text{vec}(\omega) \\ \text{vec}(\gamma) \\ x_h(t_0) \end{bmatrix} \quad (7)$$

donde β_d es el vector columna de los elementos diagonales de la matriz β , y $\text{vec}(\cdot)$ es un vector formado por las columnas de la matriz argumento (\cdot) .

De esta forma se evita una incorrecta identificación de los parámetros de la DNN. Tal y como se demuestra en [15], la inicialización de los valores de los estados escondidos a cero (tal y como se hace habitualmente) conduce a un reajuste de los parámetros de la red, para tratar de igualar las derivadas de los estados de salida de la red a las derivadas medidas experimentalmente. Esto conduce a la aparición de bias en dichos parámetros y, consecuentemente, a una mala identificación del sistema.

Una variante del problema de identificación, que ayuda al entrenamiento de la DNN, es escoger la siguiente función de costo modificada

$$V'_N(\theta, Z_N) = \frac{1}{2N} \sum_{k=1}^N \left\{ [y(t_k) - \hat{y}(t_k|\theta)]^2 + \rho |x_h(t_0)| \right\} \quad (8)$$

donde ρ es una constante positiva que pesa los valores iniciales de los estados escondidos. De esta forma se evita que dichos valores sean elevados.

Una técnica eficaz que evita que el problema de optimización alcance mínimos locales es comenzar el entrenamiento desde distintas soluciones iniciales, escogidas de forma aleatoria. Otra solución alternativa que impide el estancamiento en un mínimo local es el uso de algoritmos genéticos, pero el costo computacional de los mismos es muy elevado.

4 INICIALIZACIÓN DE LA DNN

Para la validación de la DNN entrenada se emplea un conjunto de pares de entrada-salida diferente al de entrenamiento. Los valores iniciales de los estados escondidos obtenidos en entrenamiento no son válidos para este nuevo conjunto, por lo que se recurre a una estrategia para recalcular los mismos, consistente en inicializar los estados de las neuronas de salida con los valores medidos para la salida real del sistema,

$$x_0(t_0) = y(t_0) \quad (9)$$

y los estados de las neuronas escondidas resolviendo el siguiente problema de optimización:

$$x_h(t_0) = \arg \min_{x_h(t_0)} (y(t_0) - x_0(t_0))^2 \quad (10)$$

donde $y(t_0)$ puede ser calculado de los datos obtenidos del sistema real y $x_0(t_0)$ de la ecuación (1), una vez conocidos $x_0(t_0)$, $x_h(t_0)$, β , ω y γ . De esta forma, se asegura que las derivadas de las salidas de la DNN, $\dot{x}_0(t_0)$, son aproximadamente iguales al valor real $\dot{y}(t_0)$.

5 DESCRIPCIÓN DEL SISTEMA

El sistema real sobre el que se aplica este método de identificación mediante DNN, haciendo uso de los aspectos de entrenamiento y validación anteriormente expuestos, es la pierna robótica mostrada en la figura 3 (el sistema completo consta de un par de piernas, del cual haremos los experimentos de identificación en una de ellas).

Dicho sistema está compuesto por dos elementos, muslo y parte baja de la rodilla, accionados por sendos motores de corriente continua situados en las articulaciones, cadera y rodilla, respectivamente. Dichas articulaciones son de revolución. El elemento final de la pierna es el pie, pero actualmente el sistema no dispone de motor para accionar dicha articulación, por lo que el sistema cuenta únicamente con dos grados de libertad.

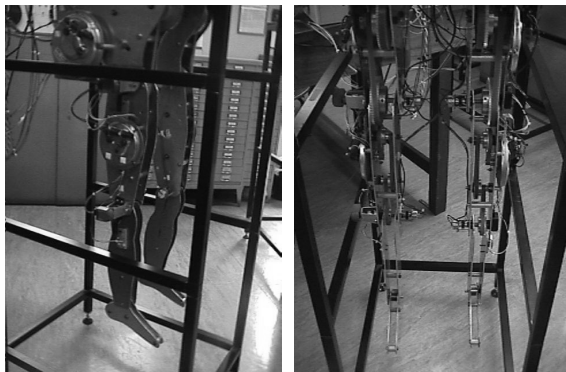


figura 3: Diferentes vistas de las piernas andadoras.

Para la medición de los ángulos de las articulaciones se dispone de sendos potenciómetros situados en la parte interior de la pierna. La medida de las velocidades la hemos realizado por diferenciación de las posiciones. Sin embargo, el sistema posee un diseño electrónico, basado en un conversor frecuencia-voltaje, capaz de dar las velocidades de

los elementos de la pierna. Una comparación entre las velocidades medidas por uno y otro método muestran que son iguales salvo para velocidades pequeñas, donde el circuito electrónico presenta algunos problemas. Por esta razón hemos escogido la diferenciación como base para el cálculo de las velocidades.

Aún cuando pareciese no muy costoso obtener el modelo teórico de este sistema, ya que corresponde inicialmente al de una cadena articulada de dos elementos unida por articulaciones de revolución, la existencia de no linealidades difíciles de modelar hacen que la identificación mediante un modelo neuronal sea más apropiada para la obtención del modelo del sistema real en estudio. En particular, aparecen en el sistema distintos tipos de fricciones, zonas muertas y también algunas holguras en las articulaciones. Por esta razón se persigue obtener el modelo de forma empírica y no teórica.

6 RESULTADOS

El interés de este trabajo es la identificación del sistema robótico de dos articulaciones explicado anteriormente mediante una DNN, de forma que contenga las no linealidades presentes en el sistema y que son difíciles de modelar teóricamente. La DNN utilizada es de tipo Hopfield y tiene la forma dada en la ecuación (2). Se aplican los aspectos de entrenamiento e inicialización para el caso de validación explicados en las secciones anteriores.

Se han obtenido de la planta real sendos conjuntos de 1595 pares de entrada-salida para entrenamiento y validación. Se han escogido las entradas como sinusoidales de distintas amplitudes y frecuencias, cubriendo, tanto las entradas como las salidas obtenidas, un amplio rango de valores en los que puede moverse el sistema real. Dichas señales han sido escaladas convenientemente. El problema de optimización fue resuelto utilizando la función *fminunc* de la toolbox de optimización de Matlab® versión 6.0 R.12, la cual utiliza el algoritmo de optimización sin restricciones Quasi-Newton.

La DNN provee como salidas los ángulos de las articulaciones, por lo que sus dos primeras neuronas se escogen como salidas, quedando el resto como neuronas escondidas. Inicialmente cabe pensar en que la estructura ideal para la DNN está constituida, como mínimo, por cuatro neuronas, ya que el sistema tiene cuatro estados: posiciones y velocidades de ambas articulaciones. Aun así, se ha hecho un estudio del error cuadrático medio para estructuras conteniendo entre tres y ocho neuronas, obteniéndose el mejor resultado para una estructura de cuatro neuronas, confirmando así la predicción anterior.

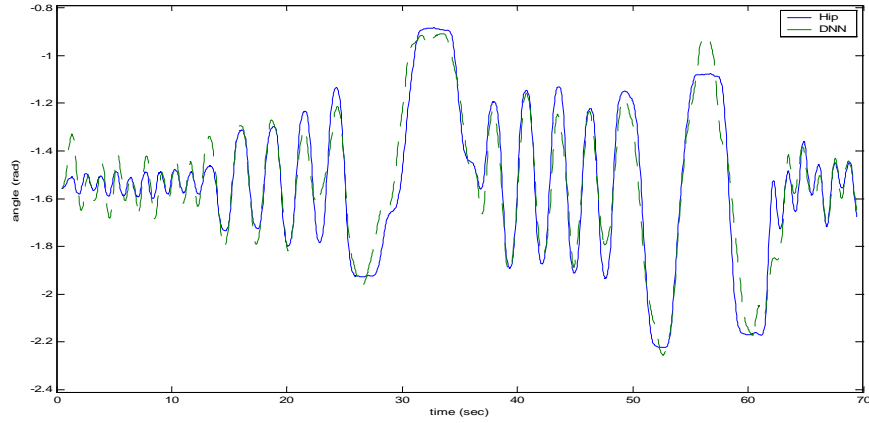


figura 4: Ángulos medidos para la primera articulación y salida de la DNN. Datos de entrenamiento.

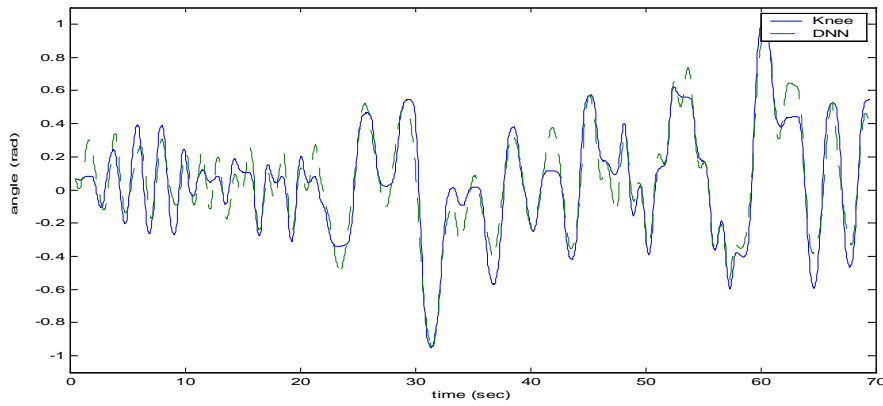


figura 5: Ángulos medidos para la segunda articulación y salida de la DNN. Datos de entrenamiento.

Los valores de los parámetros obtenidos son los siguientes:

$$\beta_d = \begin{bmatrix} 5.0856 \\ 2.9197 \\ -3.3450 \\ 2.0491 \end{bmatrix} \quad (11)$$

$$\omega = \begin{bmatrix} 3.6484 & -4.1867 & 6.1201 & -4.0089 \\ -1.2547 & 1.7313 & -3.2064 & -4.3606 \\ -0.4706 & -0.9042 & -5.7358 & -2.5743 \\ 0.7214 & 0.3866 & 0.5750 & 1.7928 \end{bmatrix} \quad (12)$$

$$\gamma = \begin{bmatrix} -0.3534 & 0.0060 \\ -0.3835 & -0.7599 \\ 0.0249 & -1.0831 \\ -0.3994 & -0.8308 \end{bmatrix} \quad (13)$$

Los resultados obtenidos para ambas articulaciones, en el caso de entrenamiento, son los mostrados en la figura 4 y en la figura 5 respectivamente. Como

podemos observar, el comportamiento de la DNN es muy similar al del sistema real. Las situaciones en las que se aprecian no linealidades son bien resueltas por la red. El incluir los estados iniciales de las neuronas escondidas en el vector de entrenamiento evita la aparición de bias en los parámetros. Además se observa que el entrenamiento se realiza en un tiempo menor.

Para el caso de validación, se obtienen los resultados mostrados en las figuras 6 y 7. Podemos observar el buen comportamiento de la DNN respecto a los datos reales. La elección de los valores de los estados iniciales de las neuronas escondidas es crucial para este caso, ya que la resolución previa del problema de optimización que minimiza el error en las velocidades, hace que la DNN se sitúe inicialmente en un estado que le permite seguir eficazmente el comportamiento del sistema real.

7 CONCLUSIONES

El objetivo de este trabajo es la identificación de un sistema robótico mediante una red neuronal

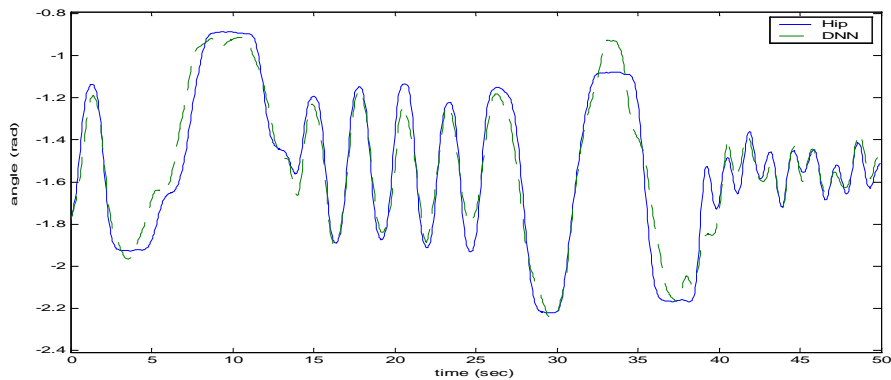


figura 6: Ángulos de la primera articulación y salida de la DNN. Datos de validación.

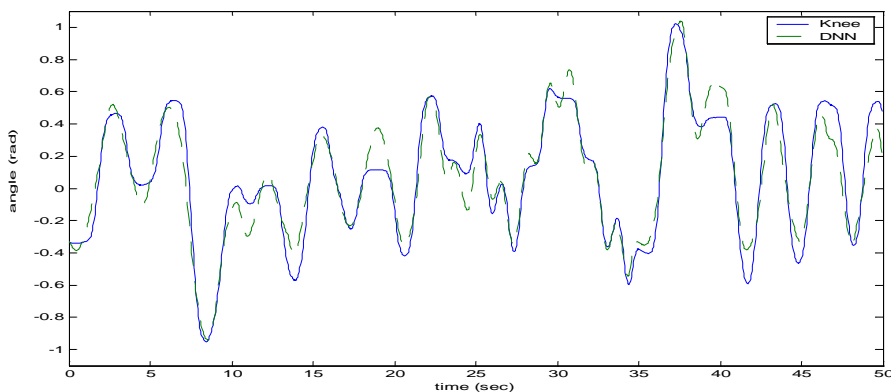


figura 7: Ángulos de la segunda articulación y salida de la DNN. Datos de validación.

dinámica, haciendo hincapié en aspectos de entrenamiento e inicialización de la misma que contribuyen a una mejor identificación del sistema, evitando la aparición de bias en los parámetros de la red como consecuencia de una errónea inicialización de los estados de las neuronas escondidas. El sistema es una pierna robótica de dos articulaciones, el cual posee bastantes no linealidades difíciles de modelar teóricamente. Los resultados obtenidos tras la identificación muestran que una DNN de cuatro estados identifica bastante bien el comportamiento real del sistema.

La red neuronal entrenada puede ser utilizada para la predicción de estados futuros del sistema utilizada por un controlador predictivo no lineal basado en modelo (NNMPC). Éste precisamente es el trabajo que se está realizando actualmente en la planta. Se han obtenido algunos resultados de control consistentes en el seguimiento de trayectorias utilizando el NNMPC. Asimismo se pretende implementar la estrategia de control predictivo con interpolación aplicada en [19]. Dichos resultados se pretenden comparar con una estrategia clásica basada en controladores PID.

Referencias

- [1] Angeles, J., (1997) Fundamentals of robotic Mechanical Systems. Theory, Methods and Algorithms. Springer-Verlag New York, Inc., pp. 216-225.
- [2] Rumelhart, D.E. and J.L. McClelland, (1986). *Parallel distributed processing: explorations in the microstructure of cognition*. MIT Press. Cambridge, Mass.
- [3] Broomhead, D.S. and Lowe, D., (1988) *Multi-variable functional interpolation and adaptive networks*. Complex Systems 2, 321-355.
- [4] Hopfield, J.J., (1982) *Neural networks and physical systems with emergent collective computational abilities*. Proceedings of the National Academy of Sciences of the United States of America – Biological Sciences 79(8), 2554-2558.
- [5] Hopfield, J.J., (1984) *Neurons with graded response have collective computational properties like those of 2-state neurons*. Proceedings of the National Academy of Sciences of the United States of America – Biological Sciences 81(10), 3088-3092.

-
- [6] Hopfield, J.J. and Tank, D.W., (1985) *Neural computation of decisions in optimizations problems*. Biological Cybernetics 52(3), 141-152.
- [7] Hopfield, J.J. and Tank, D.W., (1986) *Computing with neural circuits – a model*. Science 233(4764), 625-633.
- [8] Koiran, P. (1994). *Dynamics of discrete-time, continuous state Hopfield networks*. Neural Computation 6(3), 459-468.
- [9] Funahashi, K. and Nakamura, Y., (1993) *Approximation of dynamical-systems by continuous-time recurrent neural networks*. Neural Networks 6(6), 801-806.
- [10] Kimura, M. and Nakano, R., (1998) *Learning dynamical systems by recurrent neural networks from orbits*. Neural Networks 11(9), 1589-1599.
- [11] Garces, F., Kambhampati, C. and Warwick, K. (1999) *Dynamic recurrent neural networks for identification of a multivariable nonlinear evaporator system*. International conference on Dynamic Control Systems, DYCONS99. World Scientific. Ottawa.
- [12] Garces, F., (2000) *Dynamic neural networks for approximate input-output linearisation-decoupling of dynamic systems*. PhD Thesis. University of Reading.
- [13] Matsuoka, K., (1992) *Stability conditions for nonlinear continuous neural networks with asymmetric connection weights*. Neural Networks 5(3), 495-500.
- [14] Sánchez, E.N. and Pérez, J.P., (1999) *Input-to-state stability (ISS) analysis for dynamic neural networks*. IEEE Transactions on Circuits and Systems I-Fundamental Theory and Applications 46(11), 1395-1398.
- [15] Becerra, V.M., Calado, J.M.F., Silva P.M. and Garces, F., (2002) *System identification using dynamic neural networks: training and initialization aspects*. Proceedings of the 15th World Congress of the IFAC. Barcelona, Spain.
- [16] Kohonen, T., (1989) *Self-organization and associative memory*. 3rd ed. Springer-Verlag. Berlin; New York.
- [17] Hinton, G.E., (1986) *Learning distributed representations of concepts*. 8th Annual Conference of the Cognitive Science Society. Amherst. Mass.
- [18] Sejnowski, T.J., Kienker, P.K. and Hinton, G.E., (1986) *Learning symmetry groups with hidden units – beyond the perceptron*. Physica D 22(1-3), 260-275.
- [19] Torres, S., Méndez, J.A., Acosta, L., Sigut, M. and Marichal, G.N., (2002) *Disturbance rejection on a robot arm using an efficient predictive controller*. Proceedings of the 15th World Congress of the IFAC. Barcelona, Spain.