CONTROL ESPACIAL DE UN DEDO ANTROPOMORFO

Francisco García-Córdova, y Juan López Coronado
Departamento de Ingeniería de Sistemas y Automática. Universidad Politécnica de Cartagena.
Campus Muralla del Mar. C/Doctor Fleming S/N, 30202, Cartagena, Murcia, España.
(francisco.garcia/ jl.coronado/ @upct.es)

Resumen

Los problemas de la cinemática inversa de manipuladores redundantes han sido investigados desde hace muchos años. La cinemática inversa es computacionalmente complicada y puede traer retardos significativos en el control en tiempo real. Para un robot redundante, cálculos adicionales son requeridos para la solución de la cinemática inversa a través de esquemas de optimización. Recientemente, las redes neuronales han sido ampliamente usadas en el control de robots, debido a que son rápidas, toleran fallos y son capaces de aprender. Basado en el hecho que los humanos no calculan exactamente la cinemática inversa, pero pueden hacer posicionamientos precisos de forma heurística, nosotros desarrollamos un mapeo de la cinemática inversa a través de una red neuronal difusa. La implementación del esquema propuesto ha demostrado que es factible que cuando existen los casos de redundancia y no redundancia, sea muy eficientemente computacional. El algoritmo presentado realiza las transformaciones espacio-motoras y las motoras-espacial. Simulaciones sobre un dedo antropomorfo se llevaron a acabo para evaluar el desarrollo de la red neurodifusa.

Palabras clave: Neurodifuso, control espacial, cinemática inversa., transformaciones espaciales, dedo antropomorfo.

1 Introducción

Los manipuladores robóticos son utilizados en una gran variedad de industrias. Estos realizan aplicaciones en la soldadura, en la pintura, en el ensamble, en el manejo de materiales nucleares, exploración en el espacio y mar, etc.

El control de un manipulador involucra la planeación de la trayectoria, la dinámica inversa y la cinemática inversa (punto de interés de este artículo). Uno de mayores problemas hoy en día en el control de manipuladores robóticos es el de calcular la cinemática inversa (CI) en tiempo real. Calcular la CI es computacionalmente costosa y generalmente consume un largo porcentaje de tiempo en el control en tiempo real de manipuladores robóticos.

Manipuladores con cinemática redundante son de especial interés debido a que pueden ser utilizados para evitar singularidades, obstáculos, trabajar en entornos limitados e incrementar la capacidad a la falta de tolerancia.

El problema de la cinemática inversa es el de determinar las variables angulares a partir de la orientación y posición del extremo del manipulador. Resumiendo, dado a una trayectoria deseada en el espacio de trabajo, calcular las trayectorias correspondientes en el espacio articular del manipulador. Este es un problema complejo para manipuladores redundantes, debido a que tienen más grados de libertad (GdL) de los necesarios, presentando múltiples o infinitas soluciones. Además las ecuaciones para resolver son usualmente no-lineales y por eso se dificulta encontrar una solución cerrada. En general, la cinemática inversa no cumple en asignar uno a uno el espacio articular y el Cartesiano, y soluciones en forma cerrada al problema de la CI existe solamente para una clase muy pequeña de manipuladores cinemáticamente simples [1]. La cinemática inversa de manipuladores redundantes requieren de esquemas de optimizaciones para escoger de un conjunto de soluciones posibles la más apropiada. Pero la aplicación de los esquemas de optimización es complicada y los tiempos consumidos para los cálculos puede resultar en retardos significativos para el control.

Convencionalmente, el algoritmo de la Pseudoinversa [2] ha sido aplicado para resolver los problemas de la cinemática inversa debido a su habilidad para satisfacer restricciones adicionales a través de asignar las velocidades correspondientes a las restricciones adicionales dentro del espacio nulo del jacobiano ($\mathbf{J}$) mientras sigue la trayectoria deseada en el espacio de trabajo. Sin embargo, el algoritmo de la pseudoinversa del jacobiano involucra la inversión del la matriz ($\mathbf{J}$) la cual puede representar serios inconvenientes no solamente en las singularidades, sino también en la vecindad de una singularidad, aunque la inversa de mínimos cuadrados puede mejorar la inversión condicionada desde el punto de vista numérico.

En los recientes años, las redes neuronales artificiales (RNA) y la lógica difusa [3], [4], [5], han sido suficientemente aplicadas en el control de robots. Su capacidad de generalización y su estructura hacen de estas robustas y falta de tolerancia en algoritmos. Para resolver el problema de la cinemática inversa, aplicaciones de diferentes RNA han sido investigadas por muchos investigadores [6], [7], [8], [9], [10] . Otras RNA son de inspiraciones neurobiológicas tal como el modelo DIRECT [11], [12], [13]. El DIRECT es el modelo neuronal del sistema de control sensori-motor de los animales y da una explicación de cómo los animales realizan el control de sus miembros. El resultado de este controlador da una solución al problema de la equivalencia motora, teniendo la capacidad de adaptación a perdida de grados de libertad manteniendo su desarrollo en el espacio a cambios internos súbitos.

Los humanos no calculan exactamente la cinemática inversa cada vez que mueven un brazo o una pierna. La experiencia y el conocimiento, en lugar de cálculos complejos, permiten al humano moverse eficazmente con facilidad. En este artículo, proponemos caracterizar este conocimiento humano para proporcionar un método general del cálculo de la cinemática inversa para un arbitrario manipular de n grados de libertad a través de un mapeo cinemático inverso difuso (MCID) expresado por una red neuronal difusa (RND). Para evaluar el desarrollo del esquema de control usando un MCID se llevaron a cabo simulaciones con un dedo robótico antropomorfo de 4 GdL.

Este artículo esta organizado de la siguiente manera. En la sección 2 se describe el sistema de control espacial para un dedo antropomorfo utilizando un MCID. Resultados de simulación muestran el desempeño de la arquitectura de control para el movimiento en el espacio de un dedo, presentado en la Sección 3. Finalmente, en la Sección 4 se presentan conclusiones sobre el esquema de control propuesto y se describen trabajos futuros.

2 Esquema de control espacial de un dedo antropomorfo

2.1 Cinemática inversa convencional de un manipulador

Para un manipulador redundante cinemáticamente, la posición del extremo del manipulador es una función de las variables articulares, las cuales pueden ser expresada con la siguiente ecuación cinemática:

$$\mathbf{r} = f(\boldsymbol{\theta}) \tag{1}$$

donde $\mathbf{r}$ es un vector de la posición en el espacio del extremo del manipulador $(m \times 1)$ y $\boldsymbol{\theta}$ es el vector de los angulos articulares $(n \times 1)$, $n > m$ en caso de manipuladores redundantes.

La Figura 1 muestra la transformación de la cinemática inversa. Ahora, la trayectoria deseada en el espacio de trabajo es dada como $\mathbf{r}_d$ y nosotros necesitamos encontrar la trayectoria de los ángulos articulares $(\boldsymbol{\theta})$ correspondientes a $\mathbf{r}_d$.

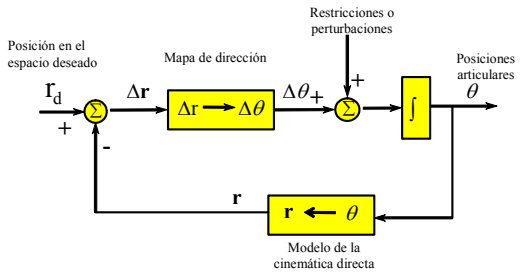


Figura 1: Diagrama de bloque simplificado de la transformación de la cinemática inversa.

Esencialmente, podemos resolver el problema inverso de la ecuación (1), quedando de la siguiente forma:

$$\dot{\mathbf{r}} = \mathbf{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{2}$$

donde $\dot{\mathbf{r}}$ es el vector de velocidad del extremo del manipulador $(m \times 1)$, $\dot{\boldsymbol{\theta}}$ es el vector de velocidad de los angulos articulares $(n \times 1)$, y $\mathbf{J}(\boldsymbol{\theta})$ es la matriz jacobiana $(m \times n)$.

Resolviendo la ecuación (2) para $\dot{\boldsymbol{\theta}}$ tenemos:

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{-1}(\boldsymbol{\theta})\dot{\mathbf{r}} \tag{3}$$

Cuando el manipulador es redundante $(n > m)$, la matriz jacobiana no es cuadrada. Usualmente la ecuación (3) se resuelve usando la pseudoinversa del jacobiano que localmente minimiza las normas de las velocidades de las articulaciones [2]. La ecuación (3) se vuelve:

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{\dagger}(\boldsymbol{\theta})\dot{\mathbf{r}} \tag{4}$$

donde

$$\mathbf{J}^{\dagger} = \mathbf{J}^{T}(\mathbf{J}\mathbf{J}^{T})^{-1} \tag{5}$$

Además, por medio

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{\dagger}(\boldsymbol{\theta})\dot{\mathbf{r}} + (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\dot{\boldsymbol{\theta}}_{a} \tag{6}$$

donde $\dot{\boldsymbol{\theta}}_a$ es un vector de velocidades de articulares arbitrario proyectado en el espacio nulo de $\mathbf{J}$, la

redundancia se resuelve especificándole a $\dot{\boldsymbol{\theta}}_a$ que satisfaga una restricción adicional.

Como se puede observar, el esquema de control con la pseudoinversa involucra la inversa de **J** y la solución de la ecuación requiere mucho calculo, debido a esto, no es conveniente para un control en tiempo real. Nosotros proponemos usar una red neuronal difusa, presentando un aprendizaje rápido del sistema a controlar, de estructura simple y propiedades de absorción a fallos tolerantes. La Figura 2 muestra el esquema del control para resolver la cinemática inversa de un manipulador. Por otra parte, la Figura 3 muestra la función que tiene el controlador cuando partimos de una posición inicial hacia una posición deseada en el espacio. En la estructura de control, el mapeo cinemático inverso difuso (MCID), expresado por una RND1, toma como entradas los incrementos de posición en el espacio (posición actual menos la deseada) y los valores de las variables articulares medidas. De estas entradas el MCID genera como salidas las trayectorias necesarias para las variables articulares, para que las posiciones del extremo del manipulador deseadas y actuales converjan a un error de estado estable.

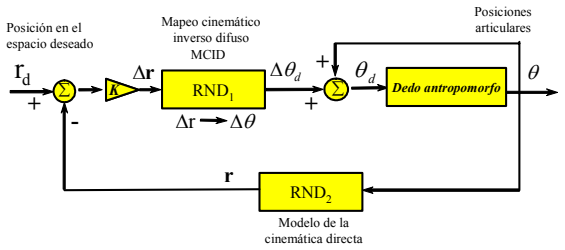


Figura 2: Controlador difuso para la transformación de la cinemática inversa de un dedo antropomorfo.

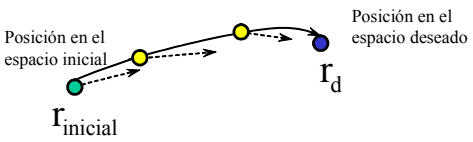


Figura 3: Robustez del controlador al error en el mapa de dirección para una posición en el espacio deseado. La dirección del movimiento en el espacio hacia la posición deseada en cada punto a lo largo de la trayectoria es indicado por flechas punteadas.

Una segunda RND2 es aplicada para calcular la cinemática directa del manipulador, teniendo como entrada las variables articulares actuales y obteniendo las posición actual en el espacio del extremo del manipulador.

2.2 Red neuronal difusa

Un sistema de reglas difusas ([5], [14], [15]) puede ser representado en forma de una red neuronal de varias maneras que dependen de la manera en que se realiza la implicación y la desdifusificación. En esta sección, se explica una forma de ver un sistema difuso como una red neuronal, y la apliacación de una regla de aprendizaje similar a retropropagación a esta red. Finalmente se presenta una implementación de esta red en donde los patrones se presentan en forma aleatoria en épocas, y utiliza la idea de *momentum* en el aprendizaje.

La red que consideraremos es la que se muestra en la Figura 4. Las salidas $y_1, y_2, \ldots, y_m$ pueden organizarse en un vector

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}, \tag{7}$$

Las entradas se denotan como $x_1, x_2, ..., x_n$, pueden ser ordenadas en un vector de entrada **x**:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}. \tag{8}$$

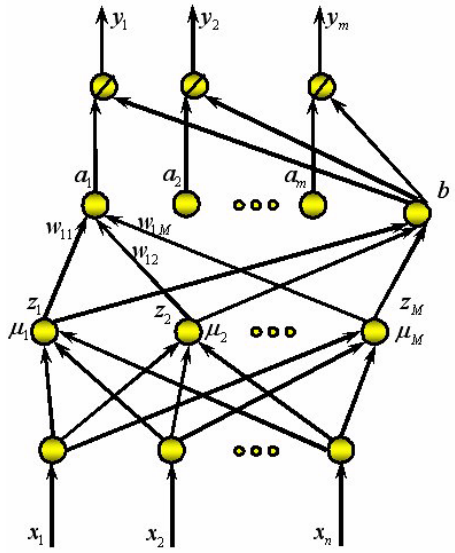


Figura 4: Red neuronal difusa de múltiples salidas. Los pesos no indicados son igual a uno.

Sobre estas entradas, se define M conjuntos difusos. La función de membresía del conjunto k tiene la siguiente forma general:

$$\mu_k(\mathbf{x}) = \exp\left[-\frac{(x_1 - h_{k1})^2}{\sigma_{k1}^2}\right] \exp\left[-\frac{(x_2 - h_{k2})^2}{\sigma_{k2}^2}\right]$$

$$\cdots \exp\left[-\frac{(x_n - h_{kn})^2}{\sigma_{kn}^2}\right]; \quad (9)$$

$$= \prod_{i=1}^{n} \exp\left[-\frac{(x_i - h_{ki})^2}{\sigma_{ki}^2}\right];$$

$$= \exp\left[-\sum_{i=1}^{n} \frac{(x_i - h_{ki})^2}{\sigma_{ki}^2}\right]; \quad (10)$$

donde $k = 1, 2, \ldots, M$. La función $\mu_k(\mathbf{x})$ tiene la forma de una gausiana en n dimensiones. En la Figura 5 se muestra la forma de esta función para $n = 2$.

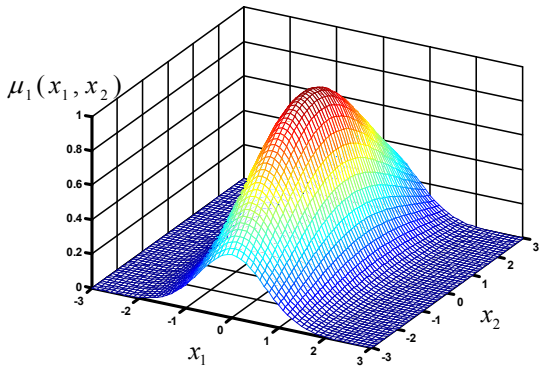


Figura 5: Función de membresía $\mu_1(x_1, x_2) = exp(-x_1^2).exp(-x_2^2/9)$.

El vector $\mathbf{h}_k$ representa las coordenadas del centro (o pico) del conjunto difuso $\mu_k(\mathbf{x})$ y que se denota con los valores $h_{k1}, h_{k2}, \ldots, h_{kM}$, se pueden ordenar en un vector $\mathbf{h}_k$:

$$\mathbf{h}_k = \begin{bmatrix} h_{k1} \\ h_{k2} \\ \vdots \\ h_{kM} \end{bmatrix}. \quad (11)$$

De la misma manera, el vector

$$\boldsymbol{\sigma}_k = \begin{bmatrix} \sigma_{k1} \\ \sigma_{k2} \\ \vdots \\ \sigma_{kM} \end{bmatrix} \quad (12)$$

contiene información acerca del ancho del conjunto difuso $\mu_k(\mathbf{x})$ sobre cada componente de $\mathbf{x}$.

Nótese que la segunda y la tercera capa están completamente interconectadas, de manera que las reglas difusas tienen la forma *si* $\mathbf{x}$ *pertenece a* μ_K *entonces* $\mathbf{y}$ *debe ser* $\mathbf{w}_k$, es decir, cada regla tiene una condición e indica acciones sobre las m salidas. Las reglas indican directamente acciones nítidas. La combinación de las acciones nítidas se obtiene haciendo un promedio ponderado de las salidas indicadas. El grado de satisfacción de cada regla se utiliza como factor de ponderación. Estos se puede formalizar mediante las siguientes ecuaciones:

$$y_i = \frac{a_i}{b}; \quad (13)$$

$$a_i = \sum_{k=1}^{M} w_{ik} z_k; \quad (14)$$

$$b = \sum_{k=1}^{M} z_k; \quad (15)$$

$$z_k = \mu_k(\mathbf{x}). \quad (16)$$

donde z_k representa el grado de satisfacción de la condición de la regla k.

Esta red tiene cuatro capas. La primera capa únicamente sirve para distribuir las antradas a la segunda capa. Los pesos entre la primera capa y la segunda están fijos en 1.0, esto significa, que no se modificarán durante el aprendizaje. Cada unidad de la segunda capa implementa una de las funciones de membresía $\mu_k(\mathbf{x})$. La salida de estas unidades es el vector

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_M \end{bmatrix}. \quad (17)$$

La tercera capa tiene $m+1$ unidades. Los pesos de la segunda capa a las m unidad de la tercera capa se organizan en una matriz de la siguiente manera:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1M} \\ w_{21} & w_{22} & \cdots & w_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mM} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1' \\ \mathbf{w}_2' \\ \vdots \\ \mathbf{w}_m' \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_m \end{bmatrix}', \quad (18)$$

donde $\mathbf{w}_i$ es el vector de pesos que entra a la unidad i de la tercera capa.

Esta unidad da como salida su entrada neta $a_i = \mathbf{w}_i'\mathbf{z}$. Los pesos de la segunda capa a la unidad b de la tercera capa están fijos en 1.0. Esta unidad da como salida su entrada neta $b = \sum_{k=1}^{M} z_k$. La capa de salida tiene m unidades. Los pesos de la tercera capa a la unidad de salida están fijos en 1.0. La unidad de salida da como salida $y_i = a_i/b$.

2.2.1 Descenso por gradiente en la red de múltiples salidas

Definimos ahora el error ante el patrón p como

$$E_p = \frac{1}{2} \sum_{i=1}^{m} (y_i^p - d_i^p)^2, \quad (19)$$

de esta manera, el error E_p es el promedio de los errores cuadráticos sobre todas las salidas.

La regla de aprendizaje para w_{jk} es

$$\Delta w_{jk} = -\alpha \left(\frac{y_j - d_j}{b_j} \right) z_k. \qquad (20)$$

La regla de aprendizaje para h_{ki} es

$$\Delta h_{ki} = -\alpha z_k \left(\frac{x_i - h_{ki}}{\sigma_{ki}^2} \right) \sum_{j=1}^{m} 2(y_j - d_j) \left(\frac{w_{jk} - y_j}{b_j} \right). \qquad (21)$$

La regla de parendizaje para σ_{ki} es:

$$\Delta \sigma_{ki} = -\alpha 2 z_k \left(\frac{(x_i - h_{ki})^2}{\sigma_{ki}^3} \right) \sum_{j=1}^{m} (y_j - d_j) \left(\frac{w_{jk} - y_j}{b_j} \right)$$

Implementación de las reglas de aprendizaje

Las reglas de aprendizaje anteriores pueden implementarse eficientemente de la siguiente manera:

$$\begin{aligned} \Delta w_{jk} &= \alpha \left(\frac{d_j - y_j}{b} \right) z_k; \\ \Delta h_{ki} &= 2 \left(\frac{x_i - h_{ki}}{\sigma_{ki}^2} \right) \sum_{j=1}^{m} \Delta w_{jk} \left(w_{jk} - y_j \right); \\ \Delta \sigma_{ki} &= \Delta h_{ki} \left(\frac{x_i - h_{ki}}{\sigma_{ki}} \right). \end{aligned}$$

Para incluir un término de *momentum* y el concepto de *época*, las anteriores reglas se implementarán de la siguiente manera. Después de cada presentación de un patrón, se calculan incrementos $\delta w_{jk}, \delta h_{ki}$, y $\delta \sigma_{ki}$.

$$\begin{aligned} \delta w_{jk} &= \alpha \left(\frac{d_j - y_j}{b} \right) z_k; \\ \delta h_{ki}(t+1) &= 2 \left(\frac{x_i - h_{ki}}{\sigma_{ki}^2} \right) \sum_{j=1}^{m} \delta w_{jk} \left(w_{jk} - y_j \right); \\ \delta \sigma_{ki}(t+1) &= \delta h_{ki} \left(\frac{x_i - h_{ki}}{\sigma_{ki}} \right). \end{aligned}$$

Al final de la época se calculan los incrementos $\Delta w_{jk}(t+1), \Delta h_{ki}(t+1)$, y $\Delta \sigma_{ki}(t+1)$ tomando en cuenta el *momentum*.

$$\Delta w_{jk}(t+1) = \beta \Delta w_{jk}(t) + \sum_{epoca} \delta w_{jk}; \qquad (22)$$

$$\Delta h_{ki}(t+1) = \beta \Delta h_{ki}(t) + \sum_{epoca} \delta h_{ki}; \qquad (23)$$

$$\Delta \sigma_{ki}(t+1) = \beta \Delta \sigma_{ki}(t) + \sum_{epoca} \delta \sigma_{ki}. \qquad (24)$$

3 Resultados de simulación

Para evaluar el sistema de control mostrado en la Figura 2, se llevaron a cabo simulaciones sobre un dedo antropomorfo. La Figura 6 representa la asignación del sistemas de coordenadas de los eslabones del dedo robótico antropomorfo y la Tabla 1 muestra los parámetros D-H del dedo.

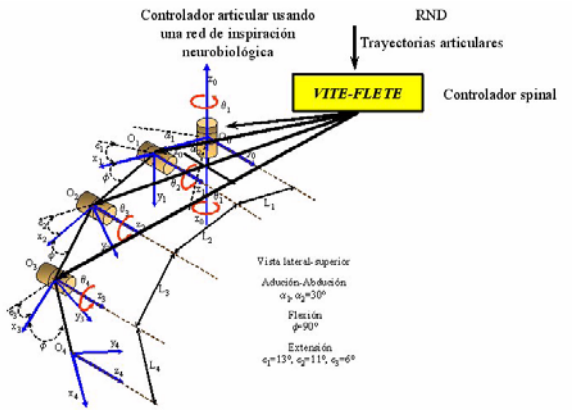


Figura 6: Representación de la asignación del sistema de coordenadas de referencias a los eslabones de la cadena cinemática del dedo índice de la mano antropomorfa

Tabla 1: Representacin de Denavit-Hartenberg (D-H) de la cadena cinemática del dedo antropomorfo.

Parámetros de coordenadas de los eslabones del dedo antropomorfo propuesto.					
Articulación	θ_n	α_n	a_n	d_n	Rangos de las articulaciones
1	θ_1	$-\pi/2$	L_1	0	-30° a $+30^\circ$
2	θ_2	0	L_2	0	-13° a $+90^\circ$
3	θ_3	0	L_3	0	-11° a $+90^\circ$
4	θ_4	0	L_4	0	-6° a $+90^\circ$

El vector de posición de la yema del dedo es:

$$\begin{aligned} x &= L_1 \cos(\theta_1) + \cos(\theta_1)[L_2 \cos(\theta_2) + L_3 \cos(\theta_2 + \theta_3) \\ &\quad + L_4 \cos(\theta_2 + \theta_3 + \theta_4)] \\ y &= L_1 \sin(\theta_1) + \sin(\theta_1)[L_2 \cos(\theta_2) + L_3 \cos(\theta_2 + \theta_3) \\ &\quad + L_4 \cos(\theta_2 + \theta_3 + \theta_4)] \\ z &= -L_2 \sin(\theta_2) - L_3 \sin(\theta_2 + \theta_3) \\ &\quad - L_4 \sin(\theta_2 + \theta_3 + \theta_4) \qquad (25) \end{aligned}$$

Las expresión (25) determina la posición del extremo del dedo robótico antropomorfo. La Figura 7 muestra el espacio de trabajo del dedo robótico antropomorfo.

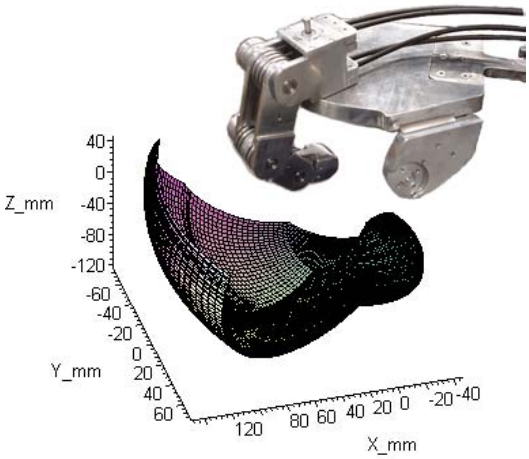


Figura 7: Espacio de trabajo del dedo robótico antropomorfo.

Las Figuras 8, 9, 10, 11, y 12. muestran el desempeño del sistema de control espacial del dedo antropomorfo. En la fase de aprendizaje la RND_1 tiene como entrada el vector $\Delta\mathbf{r}$ ($[\Delta x, \Delta y, \Delta z]^T.K$), donde K es una ganacia proporcional. La salida de esta red es el vector $\Delta\boldsymbol{\theta}$ ($[\Delta\theta_1, \Delta\theta_2, \Delta\theta_3, \Delta\theta_4]^T$). Por otra parte la RND_2 resuleve la cinemática directa teniendo como entrada el vector $\boldsymbol{\theta}$ ($[\theta_1, \theta_2, \theta_3, \theta_4]^T$) y el vector de salida de la RND_2 es $\mathbf{r}$ ($[x, y, z]^T$). La Figura 8, muestra los datos de entrenamiento de la RND_1. Los resultados aprendidos por esta red se muestran en la Figura 9.

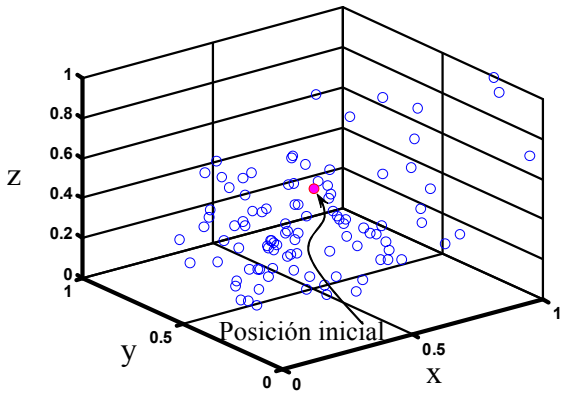


Figura 8: Datos de entrenemiento para la arquitectura de control. Datos normalizados [0,1].

La Figura 10 muestra como el dedo sigue una trayectoria dada. La eficiencia del controlador es bastanta buena despues del aprendizaje. Por otra parte, al dedo se le hizó guiar a través de una trayectoria mostrada en la Figura 11, y el controlador fue bastante eficiente ante dicha trayectoria y los resultados de la evolución de las trayectorias

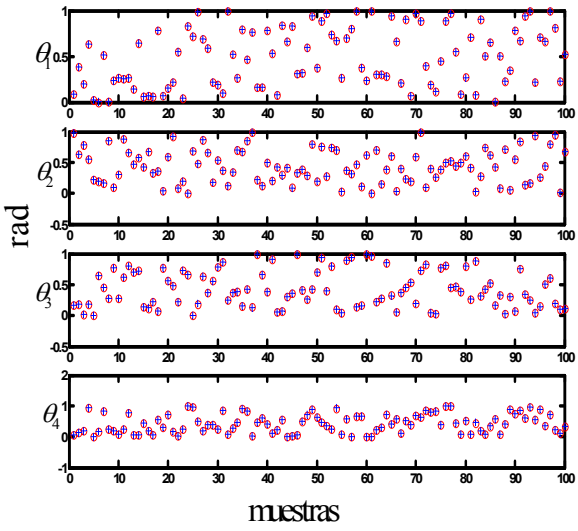


Figura 9: Datos de entrenemiento para la arquitectura de control. o ángulos deseados, + salida de la red con MCID

articulares se muestran en la Figura 12.

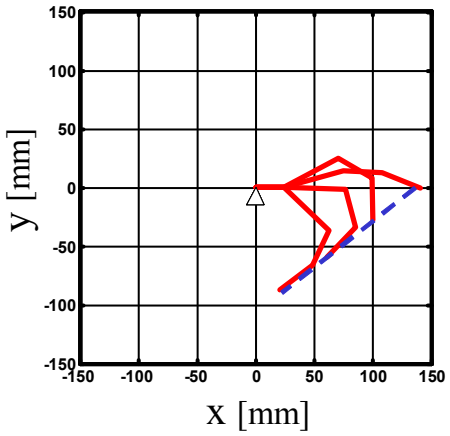


Figura 10: Control del dedo en el espacio. La linea recta representa la trayectoria deseada que el extremo del dedo debe de seguir.

4 Conclusiones

En este trabajo, hemos presentado un método para calcular la cinemática inversa, la cual ha sido robusto a configuraciones singulares, y es aplicable a manipuladores redundantes y no redundantes. El MCID propuesto presenta errores de seguimientos muy pequeños para manipuladores redundantes (dedo antropomorfo). Además el método converge rapidamente a un estado estable y produce un error de estado estable casi igual a cero en posición y orinetación del extremo del manipulador. Como trabajo a futuro es la de incor-

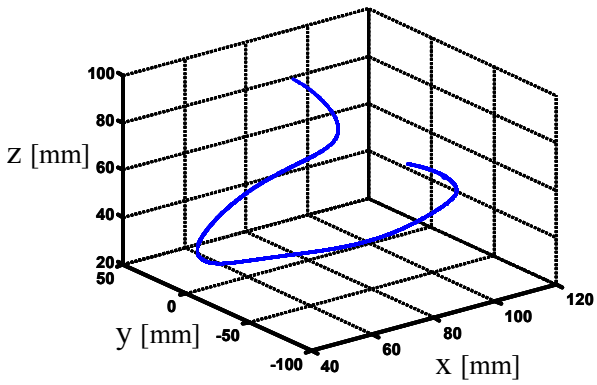


Figura 11: Trayectoria en el espacio que el dedo antropomorfo sigue.

porar al MCID una postura de relajación cuando llegue a la posición deseada en el espacio, es decir, proyectar esta postura dentro del espacio nulo del jacobiano. Esto nos permitirá escoger alguna restricción adecuada para llevar las configuraciones articulares a una forma más confortable o adecuada.

Agradecimientos

Se agradece el apoyo recibido por los miembros del grupo de investigación de Neurotecnología, Control y Robótica (*NEUROCOR*) del departamento de Ingeniería de Sistemas y Automática de la Universidad Politécnica de Cartagena. Este trabajo fue financiado en parte por la CICYT-TIC99-0446-C02-01, y por el proyecto PALOMA -IST-2001-33073- de Investigación Básica.

Referencias

[1] J. Craig, *Introduction to Robotics: Mechanics and Control*. New York, Addison-Wesley Publishing Company, 2nd ed., 1989.

[2] L. Sciavicco and B. Siciliano, *Modeling and Control of Robot Manipulators*. The MCGraw-Hill Companies, Inc., 1996.

[3] R. U. L.H. Tsoukalas, *Fuzzy and Neural Approaches in Engineering*. Wiley Interscience, 1997.

[4] W. Pedrycz, *Fuzzy Setes and Fuzzy Systems*. Research Setudies Press, england, 1993.

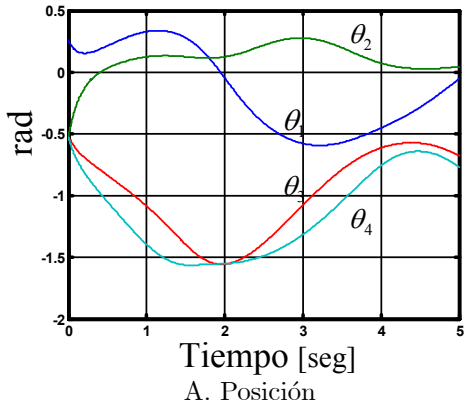
[5] L. . Zadeh, "Fuzzy logic," *Fuzzy Sets and Systems*, pp. 100–106, 1965.

[6] E. Oyama and S. Tachi, "Modular neural net system for inverse kinematics learning," in *Proccedings of the 2000 IEEE, International Conference on Robotics and Automation*, (San Francisco, C.A.), pp. 3239–3246, April 2000.

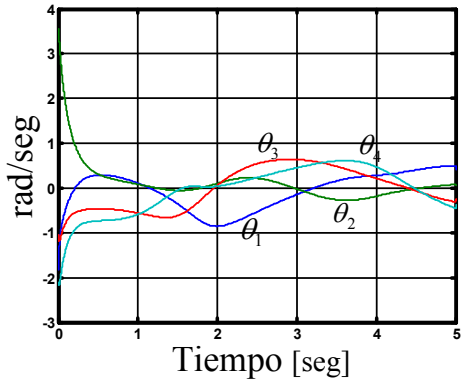
[7] J. Gardner, A. Brandt, and G. Luecke, "Applications of neural networks for trayectory control of robots," in *Fisth International Conference on Advance Robotics, 1991, Robots in Unstructured Enviorements, '91 ICAR*, vol. 1, pp. 487–492, 1991.

[8] A. Ferreira and P. Engel, "Positioning a robot arm: An adaptive neural approach," in *1996, Proceedings, International Workshop on Neural Networks for Identification, Control, Robotics, and Signal/Image Processing*, pp. 440–448, 1996.

[9] J. Walter, "PSOM network: Learning with few examples," in *Proceedings of the 1998 IEEE, International Conference on Robotics and Automation*, (Leuven, Belgium), pp. 2054–2059, 1998.



A. Posición



B. Velocidad

Figura 12: Trayectoria de las posiciones articulares del dedo antropomorfo. A muestra la posicones articulares del dedo para generar la trayectoria trazada en la figura X. B muestra las velocidades de las articulaciones del dedo antropomorfo.

[10] Y. Lin and S. H. Leong, "Kinematics control of reduntant manipulators using CMAC neural networks," in *The 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI2001)*, vol. IX, (Orlando, USA), pp. 274–279, July 2001.

[11] D. Bullock and S. Grossberg, "The vite model: A neural command circuit for generating arm and articular trajectories," *Dynamic Patterns In Complex Systems*, pp. 305–326, 1988.

[12] D. Bullock, S. Grossberg, and F. Guenther, "A self-organizing neural network model for redundant sensory-motor control, motor equivalence and tool use," *Journal of Cognitive Neuroscience*, vol. 5, pp. 408–435, 1993.

[13] A. Guerrero, J. Coronado, and F. García-Córdova, "A neural networks for target reaching with a robot arm using a sterohead," in *IEEE on System, Man and Cibernetics (SMC'99)*, (Tokyo, Japan), September 1999.

[14] B. Kosko, *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*. Englewood Cli., NJ: Prentice Hall, 1992.

[15] L. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*. Englewood Cli., NJ: Prentice Hall, 1994.