



Instituto Politécnico Nacional

Escuela Superior de Ingeniería Mecánica y Eléctrica

Sección de Estudios de Posgrado e Investigación

Seguimiento de Referencias Aplicadas en Manipuladores Robóticos

T E S I S

Que para obtener el grado de:

Doctor en Ciencias en Ingeniería Mecánica

Presenta:

M. en C. Ricardo Tapia Herrera

Directores:

Dr. Samuel Alcántara Montes

Dr. Jesús Alberto Meda Campaña



Agradecimientos

A Dios por brindarme la oportunidad de terminar una etapa más en la vida.

Al Instituto Politécnico Nacional, a la Sección de Estudios de Posgrado e Investigación, y al Consejo Nacional de Tecnología, por la beca otorgada para la realización de esta tesis.

A mis profesores, la Comisión Revisora y en especial a mis asesores, Dr. Samuel Alcántara Montes y Dr. Jesús Alberto Meda Campaña por la formación académica y los conocimientos proporcionados para la elaboración de este trabajo.

A mi familia, mis hermanos Rodrigo, Roberto, Fernando y Andrea, mis padres Ma. Del Refugio Herrera Pérez y Ricardo I. Tapia Rodríguez por su apoyo incondicional.

A mis compañeros, amigos y seres queridos que siempre me apoyaron y brindaron sus consejos...

A todos ustedes mis más grande y sincero agradecimiento.

Resumen

Este trabajo de investigación desarrolla un sistema neuro-difuso que tiene por objetivo obtener las funciones de membresía que reducen el error en el problema de regulación difusa para sistemas no lineales, cuando el sistema total es diseñado en base a reguladores locales lineales.

En un principio, el problema se aborda como un problema de sintonización, por lo que se integra al controlador una etapa de entrenamiento que ajusta la relación difusa entre reguladores locales empleando el método del descenso del gradiente, sin embargo, este método como se reporta en la literatura tiene las principales desventajas de ser un método lento y puede quedar atrapado en un mínimo local, por tal razón se emplea un sistema con mayor capacidad como es el sistema ANFIS. El sistema adaptable de inferencia neuro-difusa (ANFIS) se modificó en la etapa de entrenamiento. Los parámetros consecuentes se ajustan empleando el método de Levenberg – Marquard y los parámetros antecedentes se actualizan por medio del método del descenso del gradiente. En ambos métodos se ajustaron las reglas de aprendizaje con la finalidad de integrar a la arquitectura del ANFIS el controlador difuso basado en las ecuaciones de Francis, con la finalidad de que a partir de pares de entrada-salida generados por el exosistema se realice una búsqueda en las direcciones de descenso deseadas, sin dejar de lado las propiedades fundamentales del problema de regulación.

La tesis se organizó de la siguiente forma, en el primer capítulo se reportan los trabajos realizados en el campo de la regulación no lineal. En el segundo capítulo se establecen los conceptos matemáticos de la regulación y sistemas neuro-difusos. En el tercer capítulo se propone la metodología para introducir el sistema neuro-difuso en la regulación difusa. El capítulo cuatro presenta la aplicación de la regulación difusa en un robot manipulador de 2 G.D.L., cuando las B_i son iguales. Finalmente en el capítulo 5 se extiende y analizan los resultados cuando se aplica el ANFIS al seguimiento en los casos del péndulo invertido sobre una plataforma móvil, y de un robot manipulador de 2 G.D.L. cuando las B_i son diferentes.

Abstract

In this work the integration of a neuro-fuzzy system is proposed, in order to obtain the membership functions that reduce the output regulation error for non-linear systems, when the overall controller is designed on the basis of local linear regulators

At first the problem is addressed as a tuning problem, thus a training phase based on steepest descent method is introduced to the controller allowing adjust the fuzzy relationship between local regulators. However, according with some researches steepest descent method has the main disadvantages of be a slow method and can be trapped in a local minimum or maximum, for this reason a system with major capability of generalization and prediction as ANFIS is included.

The adaptive neuro-fuzzy inference system (ANFIS) was modified in phase of training. The consequent parameters were adjusted applying the method of Levenberg-Marquard, and the parameters in input layer (antecedent parameters) were updated by mean of Steepest Descent Method. From both methods were adjusted their learning rules, in order to integrate to the structure of ANFIS the fuzzy controller based on Francis equations, above produces that ANFIS searches in the desired descent directions from input-output training pairs generated by an exosystem, while the fundamental properties of regulation problem are guaranteed.

The thesis is organized as follow, in first chapter researches in non-linear ouput regulation are reported, in the second chapter mathematical concepts of fuzzy regulation and neuro-fuzzy systems are established. Then in chapter three the methodology to introduce ANFIS system in fuzzy output regulation is developed. Chapter four gives some application examples, and finally in chapter five are extended and discussed the results of the application of proposed method.

Índice General

Resumen.....	I
Abstract	II
Índice General	III
Índice de Figuras.....	V
Objetivo General	VIII
Objetivos Particulares	VIII
Justificación.....	IX
Introducción.....	X
Capítulo I. Estado del Arte.....	1
1.1 Teoría de regulación lineal y no lineal.....	1
1.2 Computación flexible y su aplicación en la teoría de regulación no lineal..	3
1.3 Redes neuronales en el campo de la robótica.....	5
1.4 Computación flexible híbrida.....	6
1.5 Sistemas Neuro-Difusos.....	7
1.6 Planteamiento del problema y alcances.....	9
Capítulo II. Marco Teórico	11
2.1 Regulación lineal.....	11
2.2 Regulación no lineal.....	12
2.3 Modelo difuso Takagi-Sugeno.....	13
2.4 Regulación difusa.....	13
2.5 Sistemas neuro-difusos.....	15
2.6 Aprendizaje de una red neuronal: Método del descenso del gradiente....	18
Capítulo III. Integración del sistema neuro-difuso al seguimiento de referencias..	21
3.1 Regulación lineal.....	21
3.2 Integración del sistema ANFIS en la regulación difusa	31
3.3 Ajuste de los parámetros de la capa de entrada	34

3.4	Aprendizaje de los parámetros consecuentes empleando el método de Levenberg-Marquardt.	35
Capítulo IV. Caso de aplicación		40
4.1	Seguimiento de referencias en un robot de 2 grados de libertad.	40
4.2	Plataforma experimental basada en el robot SCORBOT ER-III	47
4.3	Solución de la cinemática inversa	49
4.4	Análisis de Velocidades	50
4.5	Análisis inverso de velocidades.	51
4.6	Análisis inverso de aceleraciones.	52
4.7	Solución dinámica por el método de Lagrange – Euler para el robot SCORBOT ER III.	52
4.8	Matriz simétrica de inercia.....	53
4.9	Vector de fuerza de Coriolis y centrífuga no lineal $h(q, \dot{q})$	56
4.10	Vector de carga gravitacional.....	58
4.11	Controlador digital de arquitectura abierta	58
4.12	Sistema de adquisición de datos	58
4.13	Convertidor Digital – Analógico (C. D. A.).....	59
4.14	Control por modulación de ancho de pulso y etapa de potencia.....	59
4.15	Control vía software	60
Capítulo V. Análisis de Resultados		65
Conclusiones.....		83
Trabajos futuros		84
Referencias		85
Anexo I. Programa en MATLAB del Sistema Neuro-difuso.		92
Anexo II. Sistema Neuro-Difuso Caso 2.....		97
Anexo III. Sistema Neuro-difuso péndulo invertido.....		103
Anexo IV. Publicaciones.....		109

Índice de Figuras

Figura 1.1. Esquema de regulación para: A) Sistemas lineales, B) Sistemas no lineales [4].	2
Figura 2.1. Estructura del Sistema Adaptativo de Inferencia Neuro-Difusa.	16
Figura 2.2. Formas de la función campana generalizada cuando sus parámetros se modifican.	17
Figura 2.3. Interpretación geométrica del descenso del gradiente.	20
Figura 3.1. Funciones de membresía para el caso de regulación lineal.	21
Figura 3.2. Respuesta del esquema de regulación con F.M. triangulares.	23
Figura 3.3. Error de seguimiento con F.M. triangulares.	23
Figura 3.4. Esquema de la regulación no lineal en forma iterativa	25
Figura 3.5. Diagrama de entrenamiento del regulador difuso.	26
Figura 3.6. Evolución de los estados en la etapa de entrenamiento.	29
Figura 3.7. Error de seguimiento.	29
Figura 3.8. Respuesta de seguimiento cuando se ha sintonizado el controlador.	30
Figura 3.9. Error de seguimiento con el controlador sintonizado.	30
Figura 3.10. Integración del sistema ANFIS al ajuste de reglas.	31
Figura 3.11. Arquitectura del sistema ANFIS para una variable de entrada.	32
Figura 3.12. Distribución de las funciones de membresía.	33
Figura 3.13. Respuesta del controlador integrando el método de L-M al ANFIS.	37
Figura 3.14. Error de seguimiento.	37
Figura 3.15. Proceso de entrenamiento aplicando el método de L-M al ANFIS.	38
Figura 3.16. Aproximación del ANFIS a la no linealidad de la relación difusa.	38
Figura 3.17. Aproximación del ANFIS a la no linealidad de la relación difusa.	39
Figura 4.1. Caso 1: Robot de 2 grados de libertad.	40
Figura 4.2. Espacio de trabajo propuesto.	41
Figura 4.3. Funciones de membresía con θ_1 y θ_2 como variables de entrada.	42
Figura 4.4. Diagrama en Simulink del seguimiento en un robot de 2 G.D.L.	46
Figura 4.5. Respuesta del seguimiento en la articulación 1.	46
Figura 4.6. Respuesta del seguimiento en la articulación 2.	47

Figura 4.7. Asignación de sistemas de coordenadas de acuerdo a D-H.....	48
Figura 4.8. Robot Scorbot ER III.	49
Figura 4.9. Propiedades físicas del eslabón 1.....	54
Figura 4.10. Propiedades físicas del eslabón 2.....	55
Figura 4.11. Propiedades físicas del eslabón 3.....	55
Figura 4.12. Propiedades físicas del efector final.....	56
Figura 4.13. Tarjeta de adquisición de datos Phidget 1047.	59
Figura 4.14. Tarjeta de salidas analógicas Phidget 1002.....	59
Figura 4.15. Diagrama a bloques del sistema de potencia del robot.....	60
Figura 4.16. Arquitectura del sistema de control de arquitectura abierta.	61
Figura 4.17. Pantalla de prueba de motores del robot SCORBOT ER-III.....	62
Figura 4.18. Diagrama a bloques del instrumento virtual para la pantalla de prueba del robot SCORBOT ER-III.....	62
Figura 4.19. Función Control & Simulation Loop.....	62
Figura 4.20. Cuadro de configuración de la función Steady-State.	63
Figura 4.21. Generación de un exosistema de 2 estados en Labview.	63
Figura 4.22. Generación de las funciones de membresía a través de un MATLAB script.....	64
Figura 4.23. Generación de la señal de control para la regulación difusa.....	64
Figura 5.1. Proceso de entrenamiento para $\Omega = 3$	65
Figura 5.2. a) Error de seguimiento cuando $\Omega = 3$ b) Error de seguimiento posterior al sobretiro.....	66
Figura 5.3. Señal de control cuando $\Omega = 3$	67
Figura 5.4. Proceso de entrenamiento de μ_1	67
Figura 5.5. Péndulo invertido sobre una plataforma móvil.	68
Figura 5.6. Seguimiento del péndulo invertido cuando las funciones del regulador son triangulares.....	69
Figura 5.7. Error de seguimiento del péndulo invertido cuando las funciones del regulador son triangulares.....	69
Figura 5.8. Seguimiento del péndulo invertido con el ANFIS entrenado	70
Figura 5.9. Error de seguimiento del péndulo invertido con el ANFIS entrenado ..	70

Figura 5.10. Fase de entrenamiento de μ_1	71
Figura 5.11. Seguimiento de la señal de referencia ω_1 con F.M. lineales.....	72
Figura 5.12. Error de seguimiento en la primera articulación.	73
Figura 5.13. Seguimiento en la segunda articulación con F. M. lineales.....	73
Figura 5.14. Error de seguimiento en la articulación 2.	74
Figura 5.15. Estructura de controlador propuesto con múltiples sistemas ANFIS. 75	
Figura 5.16. Seguimiento en la primera articulación integrando ANFIS.	76
Figura 5.17. Error de seguimiento en la primera articulación con la integración del ANFIS.....	77
Figura 5.18. Seguimiento en la articulación 2 con la acción ANFIS en el controlador.	77
Figura 5.19. Error de seguimiento en la articulación 2.	78
Figura 5.20. Funciones de membresía μ_i calculadas con el ANFIS.....	79
Figura 5.21. Generador sinusoidal $\theta^{inicial} = 0$, $\theta^{final} = 60$, $t = 3seg$	81
Figura 5.22. Generador sinusoidal $\theta^{inicial} = -20$, $\theta^{final} = -45$, $t = 3seg$	81

Objetivo General

Desarrollar un sistema neuro-difuso que aproxime el mapeo entre reguladores locales en el problema de regulación no lineal difusa, para su aplicación en el seguimiento de referencias en manipuladores robóticos.

Objetivos Particulares

- Resolver el problema de regulación difusa para sistemas no lineales.
- Introducir el método del descenso del gradiente al mapeo de reguladores lineales locales.
- Programar un sistema neuro-difuso.
- Ajustar el sistema neuro-difuso integrando el controlador sintetizado por las ecuaciones de Francis.
- Establecer las ecuaciones de estado a partir de la dinámica de un robot de 2 G. D. L.
- Aplicar el algoritmo propuesto a un robot de 2 G. D. L.
- Construir una plataforma de pruebas experimentales para el robot SCORBOT ER-III.

Justificación

El seguimiento de trayectorias en sistemas no-lineales multivariados ha sido objeto de estudio por un número considerable de investigadores. En particular, Francis en sus trabajos de investigación, sienta las bases para sintetizar controladores que proporcionen a los sistemas lineales multivariados, la capacidad de seguir un sistema o una referencia externa.

A partir de los trabajos realizados por Francis, las investigaciones se han enfocado en extender la teoría de regulación lineal a los sistemas no-lineales. Tal es el caso de la regulación difusa, donde se ha demostrado que cuando se cumple la condición en que las matrices de entrada $B_1 = B_2 = B_i$, el problema de regulación difusa se puede resolver de forma exacta, y por consiguiente el error tenderá a ser 0. En el caso en que las matrices B_i sean diferentes, la solución al problema de regulación difusa puede resultar complicada de encontrar.

Por otra parte la computación flexible (Softcomputing), se ha consolidado como una herramienta con gran capacidad para resolver problemas complejos, donde características como adaptabilidad y capacidad de manejar información imprecisa, hacen del Softcomputing una alternativa viable para aplicarlo en el problema de la regulación difusa.

La precisión y repetitividad para realizar una tarea programada son factores que caracterizan a los robots manipuladores, y por lo tanto son factores importantes a considerar en el diseño de sus controladores. Lo anterior establece un punto de partida para integrar la teoría de regulación junto con el softcomputing en la robótica.

Finalmente la implementación de un controlador digital de arquitectura abierta en un robot, le proporciona flexibilidad de realizar una variedad de tareas, bajo la acción de diferentes esquemas de control.

Introducción

Uno de los problemas fundamentales en los sistemas dinámicos es controlar una planta que siga asintóticamente una señal de referencia deseada, conservando simultáneamente las propiedades adecuadas de estabilidad de los sistemas de lazo cerrado. Por lo tanto el problema de regulación para un sistema sujeto a una perturbación externa y señales de referencia, ambas generadas por un sistema dinámico llamado exosistema, consiste en encontrar un controlador retroalimentado por los estados o error, de tal forma que en el punto de equilibrio del sistema en lazo cerrado sin señales de referencia sea asintóticamente estable, y el error de seguimiento tienda a cero, cuando el sistema está bajo la influencia del sistema externo (exosistema).

El problema de regulación consiste en encontrar un controlador que cumpla dos objetivos principales:

- 1) Estabilizar el sistema en lazo cerrado sin influencia del exosistema.
- 2) El error de seguimiento debe tender a cero asintóticamente cuando la planta se encuentra bajo el efecto del exosistema.

Este problema es complicado ya que la mayoría de los sistemas reales son no lineales, por lo tanto, como alternativa se presenta la lógica difusa para construir un controlador cuya estructura esté basada en controladores locales en puntos de operación definidos, tal que la combinación apropiada entre ellos genere una aproximación, y en algunos casos, una representación exacta del controlador no lineal.

Puede observarse que se trata de un problema de optimización y particularmente de un problema de sintonización. Como alternativa de solución pueden emplearse algoritmos basados en la computación inteligente, que busquen la solución en forma iterativa.

1.1 Teoría de regulación lineal y no lineal.

Uno de los primeros trabajos sobre teoría de regulación que se presentan en la literatura es el desarrollado por Francis en 1975 [1], este trabajo obtiene criterios estructurales necesarios para sintetizar reguladores lineales multivariables que poseen estabilidad estructural a pequeñas perturbaciones en los parámetros del sistema. La principal aportación de Francis consiste en establecer que la estabilidad estructural requiere de la retroalimentación de la variable regulada, estas características constituyen el “principio del modelo interno”. Posteriormente en 1977 Francis [2] propone la teoría de regulación lineal para la solución del seguimiento de referencias, Francis plantea la solución de la regulación de la salida de un sistema lineal autónomo sujeto a perturbaciones y a señales de referencia, y demuestra que el problema de regulación lineal multivariable se puede resolver a partir de un sistema de ecuaciones lineales matriciales.

Posteriormente, Isidori y Byrnes [3] extienden los resultados obtenidos por Francis al caso general en que la planta y/o el exosistema son no-lineales. En este trabajo se muestra que las ecuaciones de Francis son un caso particular del problema no-lineal. Isidori y Byrnes demuestran que de forma semejante al caso lineal, si la planta es exponencialmente estabilizable vía retroalimentación de la salida, la solución de las ecuaciones de Francis para el caso no-lineal implicará la solución del problema del regulador vía retroalimentación del error. Otro punto que se aborda es la existencia de condiciones expresadas como una propiedad especial de la dinámica cero de un sistema que incorpora la dinámica de la planta, la dinámica del exosistema y cuya salida es definida como el error de seguimiento. Isidori concluye que la solución del problema del regulador existe siempre bajo la

condición de que la dinámica cero de la planta posea un equilibrio hiperbólico. En la figura 1.1 se muestra el esquema de la regulación lineal y no lineal.

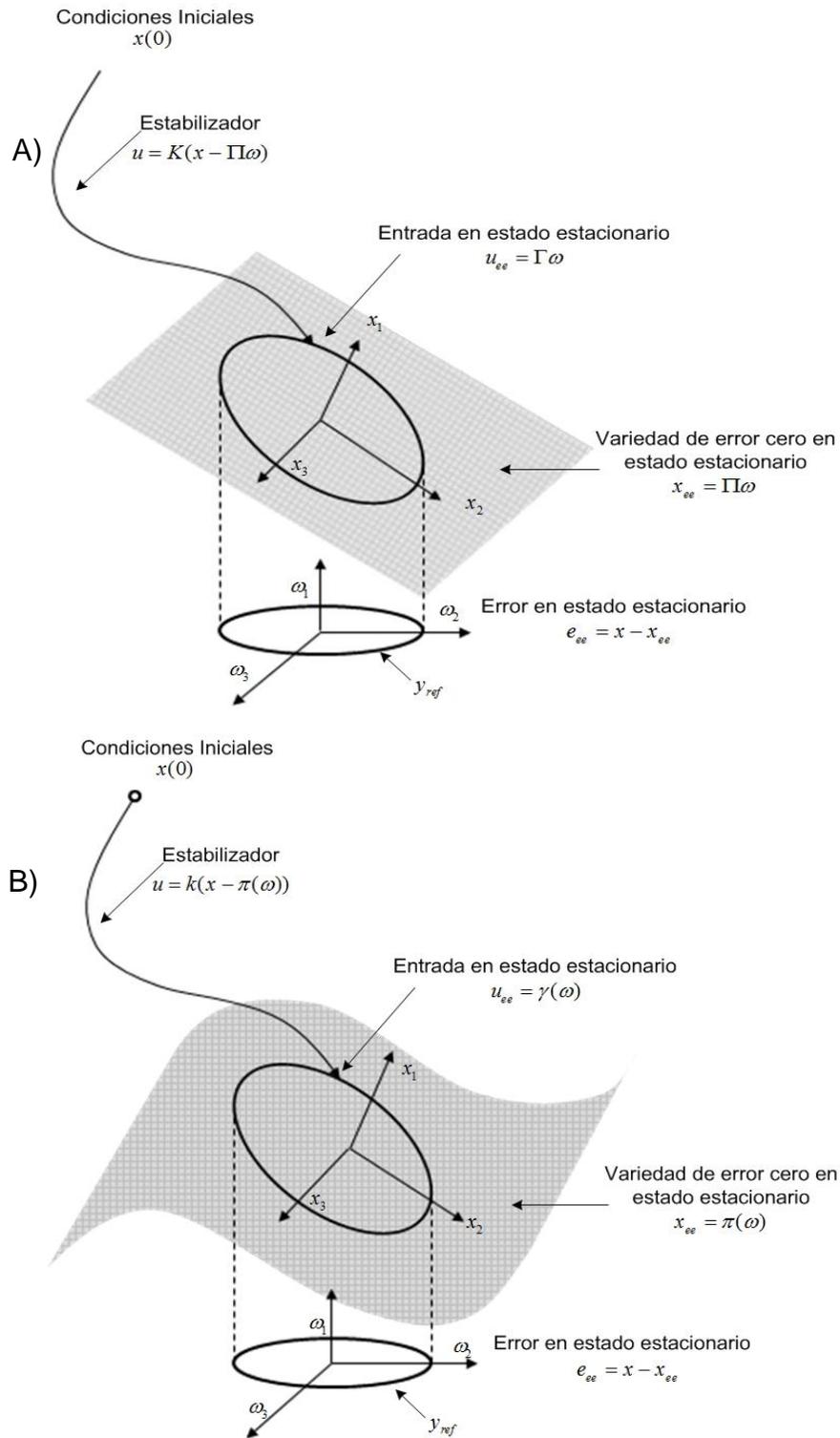


Figura 1.1. Esquema de regulación para: A) Sistemas lineales, B) Sistemas no lineales [4].

1.2 Computación flexible y su aplicación en la teoría de regulación no lineal.

En el mundo real existe una gran gama de problemas que requieren del análisis de información incierta e imprecisa, estas razones son un marco de referencia para emplear la computación flexible. El término de computación flexible (Soft-Computing) de acuerdo a Zadeh [5, 6], es “Una colección de metodologías que pretenden explotar la tolerancia hacia la imprecisión y la incertidumbre, para lograr manejabilidad, robustez y soluciones de bajo costo”. Las principales herramientas de la computación inteligente son, lógica difusa, redes neuronales, algoritmos genéticos y el razonamiento probabilístico. Cada una de estas herramientas poseen diferentes características para resolver una variedad de problemas. Los sistemas difusos se caracterizan por su capacidad de manejar información imprecisa y usar un razonamiento aproximado, por otro lado las redes neuronales tienen la principal propiedad de aprender. Muchas de estas herramientas se han empleado con gran efectividad para resolver problemas complejos del mundo real en campos como el diagnóstico, estimación, control y autonomía de sistemas.

El problema de regulación no lineal debido al conjunto de ecuaciones diferenciales a resolver, se puede tornar complicado, por esta razón investigadores han optado por emplear la inteligencia artificial: en particular las redes neuronales, lógica difusa y algoritmos genéticos.

En [7] se presenta un algoritmo iterativo con la estructura de una red neuronal recurrente, la red se introduce con el propósito de hacer frente a los errores del modelo ocasionados por la estimación de parámetros de un entrenamiento fuera de línea y la posible dinámica cambiante, así mismo se emplea una estrategia adaptativa para proveer a la red los parámetros necesarios usando aprendizaje en línea, en este sentido la red puede captar las incertidumbres del sistema y el regulador ajusta la acción de control para garantizar que el error sea cero, logrando combinar la capacidad de identificación de las redes neuronales y las propiedades de estabilidad de la teoría de regulación. Los resultados son que la estructura del control no sólo puede aprender la dinámica del sistema, sino que

también se garantiza la convergencia asintótica del error. La aplicación de este método se aplicó al esquema de control de una planta solar de energía [8].

Zhang [9] propone una aproximación en forma de series de potencias, en la cual la red neuronal se encarga de la síntesis y auto-sintonización del controlador en tiempo real. Este método garantiza las propiedades de la regulación: estabilidad y seguimiento asintótico.

Posteriormente en [10, 11] se trata el problema del regulador difuso con la intención que un sistema no lineal descrito por el modelo difuso del tipo Takagi-Sugeno siga una referencia definida por un exosistema, este controlador soluciona el problema cuando se cumple la condición de que las matrices de cada subsistema B_i sean iguales ($B_1 = B_2 \dots = B_n$). Finalmente para complementar el controlador, el estabilizador se obtiene a partir de desigualdades lineales matriciales.

Para que el problema de regulación difusa pueda resolverse deben existir dos condiciones necesarias:

- Las derivadas con respecto al tiempo de las funciones de membresía sean cero
- Que no existan términos cruzados en las matrices B_i de cada regulador

No obstante lo anterior en cuantiosos sistemas no se cumple. Una propuesta a este problema se presenta en [4] y [12], dónde el diseño del regulador está basado en el modelo difuso total en lugar de los subsistemas lineales locales. La ventaja de este enfoque es la convergencia global del error de seguimiento en la región de interés, en contraste con las propiedades locales del método clásico de regulación no lineal. Por otra parte el mapeo que garantiza la propiedad de regulación exacta puede ser obtenida a partir del conjunto de ecuaciones lineales (ecuaciones dinámicas de Francis). Como consecuencia los resultados muestran que el método garantiza el seguimiento exacto (error igual a cero), una desventaja de este enfoque es que el regulador difuso exacto es más complejo que el regulador

diseñado a partir de la suma difusa de los reguladores lineales locales. En este sentido en [13] se presenta la funcionalidad de la regulación difusa, donde se diseña un regulador no-lineal para guiar misiles en un sistema de intercepción, desde el enfoque de seguimiento de un sistema no-lineal variante en el tiempo.

Como se ha mencionado el funcionamiento de un controlador basado en la regulación difusa, depende de que se cumplan ciertas condiciones y de la correcta selección de las funciones de membresía, en [14] se implementa una etapa que emplea algoritmos genéticos con el objetivo de encontrar los parámetros óptimos del controlador difuso que reducirán el error de seguimiento de posición en un servomotor con una transmisión con holgura, dando como resultado que el controlador tiene un buen desempeño.

1.3 Redes neuronales en el campo de la robótica.

El cerebro humano está constituido por una enorme cantidad de neuronas altamente interconectadas, mediante enlaces con fuerzas de adhesión variables y operando en configuración paralela. Para lograr simular las características de procesamiento distribuido masivamente en paralelo del cerebro, redes neuronales artificiales han sido investigadas y desarrolladas ampliamente desde el primer trabajo presentado en la materia por McCulloch y Pitts en 1943 [15], basado en neuronas modeladas como elementos de toma de decisiones discretas. Más aún, como el cerebro humano, las redes neuronales artificiales tienen la habilidad de aprender y generalizar.

Las redes neuronales se han aplicado en el campo de la robótica, un claro ejemplo es el aprendizaje del modelo cinemático inverso de un manipulador el cual puede compararse con el brazo humano. La función de la cinemática inversa del brazo es una función con múltiples valores y discontinua. Debido a ello, en [16] se propone una novedosa arquitectura modular de una red neuronal, la cual soluciona el problema de aproximar una función discontinua.

En [17] se presenta la solución a la cinemática inversa de un manipulador robótico de tres grados de libertad, basándose en una red neuronal. Como datos de entrenamiento de la red neuronal, se generan puntos iniciales y finales dentro del espacio de trabajo del manipulador por medio de trayectorias basadas en polinomios cúbicos, posteriormente se almacenan en un archivo de datos de entrenamiento (ángulos que corresponden a las coordenadas cartesianas x, y, z). Realizando las simulaciones se observa que la red neuronal diseñada soluciona de manera exitosa el problema cinemático inverso.

Las singularidades y las incertidumbres en las diferentes configuraciones de un brazo robótico son algunas de las dificultades que se presentan en la solución de su cinemática inversa. En [18] se realiza un aprendizaje adaptativo mediante el uso de una red neuronal artificial. Con la finalidad de solucionar el problema de la cinemática inversa, la red neuronal es entrenada para aprender un conjunto de posiciones angulares de las articulaciones, dado previamente un conjunto de posiciones del efector final. Este mapeo es realizado dentro del espacio de trabajo del manipulador y una vez que la red ha sido entrenada se valida su capacidad de predicción y generalización mediante la introducción a la red de un conjunto de nuevos datos. Los resultados experimentales demuestran una buena capacidad de generalización.

En [19] se propone una solución al problema cinemático inverso de manipuladores robóticos basada en redes neuronales estructuradas que puedan ser entrenadas rápidamente. La solución a la cinemática inversa resulta ser precisa y conveniente para aplicaciones en tiempo real. Muchas son ya las soluciones al problema cinemático inverso de manipuladores redundantes, que se basan en redes neuronales artificiales; sin embargo, tales soluciones resultan ser no muy confiables, debido al error que se presenta al final del entrenamiento de la red.

1.4 Computación flexible híbrida.

Un sistema híbrido inteligente es aquel que combina dos o más técnicas de la computación flexible [20], permitiendo aprovechar las ventajas de cada

herramienta para mejorar su funcionamiento. En la literatura se reportan como resultado de la hibridación entre las herramientas de la computación flexible los siguientes algoritmos [21, 22]:

- Controladores difusos sintonizados por redes neuronales.
- Controladores difusos sintonizados por computación evolutiva.
- Redes neuronales generadas por computación evolutiva.
- Redes neuronales controladas por lógica difusa.
- Computación evolutiva controlada por lógica difusa.

1.5 Sistemas Neuro-Difusos.

Las redes neuronales artificiales son una herramienta que se ha aplicado con gran efectividad a problemas como: identificación, clasificación o reconocimiento de patrones. Por otro lado un sistema basado en lógica difusa tiene la capacidad de trabajar con información imprecisa y tomar decisiones de acuerdo a las reglas establecidas por un experto en forma heurística [23]. Particularmente un sistema neuro-difuso aprovecha de la lógica difusa el mecanismo de inferencia bajo incertidumbre cognitiva, y de las redes neuronales su capacidad de aprendizaje, adaptación, paralelismo y generalización.

Cabe señalar que el enfoque neuronal no sólo tiene como propósito ajustar los parámetros del controlador difuso, sino también una red puede funcionar como un controlador difuso, dicho de otra forma el controlador difuso tendrá la estructura de una red neuronal [24].

El correcto funcionamiento de un controlador difuso está en función de la forma de las funciones de membresía que representan cuantitativamente a las variables lingüísticas [25], así como la correcta elección de los operadores binarios [26]. En consecuencia los parámetros que conforman la estructura del controlador difuso son generalmente definidos por un experto a prueba y error [27]. Es posible encontrar los parámetros adecuados que mejoran el funcionamiento del sistema difuso empleando las redes neuronales artificiales [28-30].

Las redes neuronales proporcionan al controlador difuso la capacidad de aprender a través de un proceso automatizado, lo que implica que el sistema difuso se

podrá auto-ajustar (auto-sintonizar) [31]. Con frecuencia las técnicas de aprendizaje se basan en la retro-propagación del error, y se utilizan métodos como el descenso del gradiente para reducirlo en la dirección de los parámetros a ajustar [32-38].

El funcionamiento de un sistema neuro-difuso está ampliamente ligado al funcionamiento del algoritmo de aprendizaje, si el método es eficiente se traducirá en que la red reduzca substancialmente el consumo de tiempo para ajustar sus parámetros. En este sentido se han desarrollado diversas técnicas que mejoran el desempeño tanto en la estructura como en el aprendizaje del sistema neuro-difuso.

En [39] se presenta un controlador adaptativo difuso-neuronal, basado en la estructura de una red neuro-difusa, para sistemas dinámicos desconocidos, considerando sistemas no-lineales sin perturbaciones. Básicamente se proponen dos objetivos: 1) Establecer un controlador adaptativo difuso- neuronal para sistemas no lineales que ajuste los pesos de la red en línea. 2) Derivar una ley de control donde tanto pesos como funciones de membresía de la red sean sintonizados en línea; Otra aportación es la introducción de las funciones de membresía B-spline que pueden ser sintonizadas localmente, produciendo una mejor efectividad en la red.

Estudios como el realizado en [40] propone diferentes enfoques que pueden ser aplicados al proceso de aprendizaje de los sistemas neuro-difusos, utilizando la herramienta computacional XFuzzy. XFuzzy incorpora: una variedad de algoritmos de aprendizaje supervisado, diferentes procesos para simplificar el sistema ya entrenado, la opción de sintonizar parámetros específicos del sistema, así como, la capacidad de sintonizar sistemas difusos jerárquicos.

En este mismo sentido trabajos como los realizados en [41] y [42] presentan metodologías que disminuyen el tiempo en que un sistema neuro-difuso puede ser ajustado, esto con el objetivo de que el sistema tenga un aprendizaje en línea para su aplicación en sistemas de múltiples-entradas, múltiples salidas (MIMO).

El sistema ANFIS (Adaptative Neuro-Fuzzy Inference System) propuesto por Jang [43], es un sistema que construye un mapeo entre pares de entrada – salida, utilizando el conocimiento humano en forma de reglas difusas SI-ENTONCES; su entrenamiento es híbrido ya que combina el algoritmo de retro-propagación del error y el estimador de mínimos cuadrados. En general este sistema se usa para modelar funciones no lineales, identificar componentes no lineales en sistemas de control y predecir series caóticas en el tiempo.

Existe la posibilidad de combinar el sistema ANFIS con alguna otra herramienta de la computación inteligente, en [44] se propone un controlador que introduce los algoritmos genéticos en el ANFIS dando por resultado uno de los enfoques más poderosos que proporcionan adaptabilidad en sistemas no lineales. El controlador GA-ANFIS se compara con métodos tradicionales y se concluye que el controlador GA-ANFIS tiene un mejor funcionamiento en términos de la especificación del dominio del tiempo, la referencia a seguir y el rechazo a perturbaciones con una estabilidad óptima.

Un controlador inverso puede no garantizar una respuesta satisfactoria a las diferentes incertidumbres asociadas con las condiciones de operación y ruido en los datos de entrenamiento, en [45] se introduce al ANFIS un aprendizaje iterativo para auto-sintonizar, minimizando el error total del sistema, empleando una estructura en la cual las entradas del ANFIS se ajustan de manera adaptativa por medio de la ley de aprendizaje iterativa.

1.6 Planteamiento del problema y alcances

Con base en lo expuesto en los trabajos previos, se puede calcular la solución analítica de la relación entre reguladores locales en la regulación difusa, presentándose el inconveniente de que en algunos casos esta solución puede ser complicada de obtener.

Se observa que actualmente no se cuenta con un sistema automatizado que aproxime la relación entre reguladores locales, por lo tanto se considera viable hacer del controlador difuso un sistema híbrido inteligente capaz de reducir el error

en estado estable en el seguimiento de referencias. Lo anterior puede llevarse a cabo integrando una etapa de aprendizaje, así mismo es posible emplear el sistema ANFIS para aproximar las no-linealidades de los mapeos existentes entre reguladores locales, bajo la condición específica de que las matrices Π_i del controlador sean iguales, con lo cual se pretende obtener un método que proporcione expresiones computacionalmente eficientes, y en consecuencia se disminuya el consumo de recursos computacionales.

En lo referente al seguimiento de referencias en sistemas de múltiples-entradas múltiples-salidas como los robots manipuladores, existe un gran campo de aplicación para la regulación difusa ya que son pocos los trabajos en este ámbito, presentándose la oportunidad de analizar y establecer los límites y condiciones bajo las cuales funcionaría el método propuesto.

2.1 Regulación lineal

Se tiene un sistema lineal de la forma:

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx\end{aligned}\tag{2.1}$$

y un sistema de referencia o exosistema dado por:

$$\begin{aligned}\dot{\omega} &= S\omega \\ y_{ref} &= Q\omega\end{aligned}\tag{2.2}$$

Donde A, B, C, x, u , son la matriz de estados, matriz de entrada, matriz de salida, vector de estados y vector de control, respectivamente.

Se busca la acción de control que proporcione la estabilidad asintótica y la regulación de la salida. La estabilidad asintótica local se logra cuando el exosistema es desconectado ($\omega=0$) y el sistema en lazo cerrado tiene un equilibrio asintóticamente estable en $x=0$; la regulación de la salida significa que el sistema en lazo cerrado para todos los estados iniciales suficientemente cerca al origen, $e(t) \rightarrow 0$ cuando $t \rightarrow \infty$. La señal de control propuesta que hará que el sistema (2.1) se estabilice y siga al sistema (2.2) es: $u = K(x - \Pi\omega) + \Gamma\omega$, el error en estado estable será la diferencia entre los estados de la planta y $\Pi\omega$, ($e_{ee} = x - \Pi\omega$), considerando que $x_{ee} = \Pi\omega$, producirá:

$$e_{ee} = x - \Pi\omega \quad \therefore \dot{e}_{ee} = \dot{x} - \Pi\dot{\omega} \rightarrow \dot{x} = \dot{e}_{ee} + \Pi\dot{\omega}$$

Sustituyendo las expresiones anteriores en la ecuación (2.1):

$$\dot{e}_{ee} = Ax + Bu - \Pi\dot{\omega}\tag{2.3}$$

$$\dot{e}_{ee} = A[e_{ee} + \Pi\omega] + B[k(x - \Pi\omega) + \Gamma\omega] - \Pi\dot{\omega} \quad (2.4)$$

$$\dot{e}_{ee} = Ae_{ee} + A\Pi\omega + BK e_{ee} + BK\Pi\omega - BK\Pi\omega + B\Gamma\omega - \Pi\dot{\omega} \quad (2.5)$$

Si $\dot{\omega} = S\omega$ y $e_{ee} = 0$ se obtiene el siguiente sistema de ecuaciones:

$$\begin{aligned} \Pi S &= A\Pi + B\Gamma \\ 0 &= C\Pi - Q \end{aligned} \quad (2.6)$$

El conjunto de ecuaciones anteriores son denominadas ecuaciones de Francis [2]. Donde las dimensiones de cada matriz son: $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{m \times n}$, $\Pi \in \mathbb{R}^{n \times r}$, $S \in \mathbb{R}^{r \times r}$, $\Gamma \in \mathbb{R}^{m \times r}$, $Q \in \mathbb{R}^{m \times r}$. Y para el caso donde existan perturbaciones se tendrá que $P \in \mathbb{R}^{n \times r}$.

2.2 Regulación no lineal

De acuerdo con Isidori la generalización de los sistemas (2.1) y (2.2) para el caso no lineal es:

$$\begin{aligned} \dot{x} &= f(x, \omega, u) \\ y &= h(x) \\ \dot{\omega} &= s(\omega) \\ y_{ref} &= q(\omega) \end{aligned} \quad (2.7)$$

se propone una señal de entrada: $u = k[x - \pi(\omega)] + \gamma(\omega)$, y el error en estado estacionario es $e_{ee} = x - x_{ee}$ donde $x_{ee} = \pi(\omega)$, por lo que la razón de cambio del error será:

$$\dot{e}_{ee} = f(x, \omega, u) - \frac{d}{dt}[\pi(\omega)] \quad (2.8)$$

$$\dot{e}_{ee} = f(x, \omega, u) - \frac{\partial \pi}{\partial \omega} \dot{\omega} \quad (2.9)$$

$$\dot{e}_{ee} = f(x, \omega, u) - \frac{\partial \pi}{\partial \omega} s(\omega) \quad (2.10)$$

para que se cumpla la condición de regulación se debe satisfacer $e_{ee} = 0$, por lo tanto $\dot{e}_{ee} = 0$ así la expresión anterior se reescribe de la siguiente forma:

$$\begin{aligned} \frac{\partial \pi}{\partial \omega} s(\omega) &= f(\pi(\omega), \omega, \gamma(\omega)) \\ 0 &= h(\pi(\omega)) - q(\omega) \end{aligned} \quad (2.11)$$

El conjunto de ecuaciones (2.11) se conocen como ecuaciones de Francis-Isidori-Byrnes [3].

2.3 Modelo difuso Takagi-Sugeno

El modelo difuso propuesto por Takagi y Sugeno [46] se describe por un conjunto difuso de reglas SI – ENTONCES, la principal característica de un sistema difuso T-S es que puede representar la dinámica local de cada implicación difusa por un sistema lineal del modelo:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \quad (2.12)$$

Por lo que un sistema T-S escrito en variables de estado en tiempo continuo es:

$$\begin{aligned} \dot{x} &= \sum_{i=1}^r h_i^* \{A_i x + B_i u\} + \sum_{i=1}^r h_i^* P_i \quad i = 1, 2, \dots, r \\ y &= \sum_{i=1}^r h_i^* C_i x \end{aligned} \quad (2.13)$$

2.4 Regulación difusa

Se desea que el sistema (2.13) siga una referencia externa (exosistema), que de igual forma puede ser representada por un sistema difuso T-S:

$$\begin{aligned} \dot{\omega} &= \sum_{i=1}^{r_2} h_i^* S_i \omega \\ y_{ref} &= \sum_{i=1}^{r_2} h_i^* Q_i \omega \end{aligned} \quad (2.14)$$

cabe mencionar que las funciones de membresía h_i^* no necesariamente deben ser las mismas en la planta y en el exosistema. Así mismo, es necesario recordar que $x_{ee} = \pi(\omega)$, en consecuencia es posible reescribir el sistema de ecuaciones (2.13) y (2.14) como:

$$\begin{aligned}\dot{x} &= \tilde{A}(t)x(t) + \tilde{B}(t)u(t) \\ y &= \tilde{C}(t)x(t) \\ \dot{\omega} &= \tilde{S}(t)\omega(t) \\ y_{ref} &= \tilde{Q}(t)\omega(t)\end{aligned}\tag{2.15}$$

con una señal difusa de control:

$$u(t) = \sum_{i=1}^n h_i^1(z_1(t)) K_i [x(t) - \pi(\omega(t))] + \gamma(\omega(t))\tag{2.16}$$

la variedad de error cero en estado estable es $x_{ee} = \pi(\omega(t))$ y la variable de entrada (control) en estado estable es: $u_{ee} = \gamma(\omega(t))$, las cuales no son aproximadas por la suma difusa de los mapeos lineales locales. Por lo tanto $\pi(\omega(t))$ y $\gamma(\omega(t))$ pueden ser obtenidas a partir de:

$$e_{ee}(t) = x(t) - x_{ee}(t) = x(t) - \pi(\omega(t))\tag{2.17}$$

Cabe mencionar que $e_{ee}(t)$ es el error entre los estados de la planta $x(t)$ y la variedad de error cero $\pi(\omega(t))$, y el error $e(t)$ es la diferencia entre la salida de la planta $y(t)$ y la señal de referencia y_{ref} . Las soluciones al problema de regulación dado por el sistema (2.15) pueden definirse de la siguiente forma [4, 10]:

$$\pi(\omega(t)) = \tilde{\Pi}(t)\omega(t)\tag{2.18}$$

bajo la condición de entrada en estado estable:

$$\gamma(\omega(t)) = \tilde{\Gamma}(t)\omega(t)\tag{2.19}$$

$\tilde{\Pi}(t)$ y $\tilde{\Gamma}(t)$ son matrices que varían en el tiempo. Derivando el error en estado estacionario (2.17) y sustituyendo las ecuaciones (2.18) y (2.19), se obtiene:

$$\dot{e}_{ee} = \dot{x}(t) - \dot{\tilde{\Pi}}(t)\omega(t) - \tilde{\Pi}(t)\dot{\omega}(t) \quad (2.20)$$

puede observarse que $\dot{x} = \tilde{A}(t)x(t) + \tilde{B}(t)u(t) + \tilde{P}(t)\omega(t)$ por lo tanto (2.20) se reescribe de la siguiente forma:

$$\dot{e}_{ee} = \tilde{A}(t)x(t) + \tilde{B}(t)u(t) + \tilde{P}(t)\omega(t) - \dot{\tilde{\Pi}}(t)\omega(t) - \tilde{\Pi}(t)\dot{\omega}(t) \quad (2.21)$$

de (2.17) se desea que $e_{ee} = 0$ por lo tanto tomando en cuenta (2.18), (2.19) finalmente (2.21) se expresaría como [4]:

$$\dot{\tilde{\Pi}}(t)\omega(t) + \tilde{\Pi}(t)\tilde{S}(t)\omega(t) = \tilde{A}(t)\tilde{\Pi}(t)\omega(t) + \tilde{B}(t)\tilde{\Gamma}(t)\omega(t) + \tilde{P}(t)\omega(t) \quad (2.22)$$

$$\dot{\tilde{\Pi}}(t) + \tilde{\Pi}(t)\tilde{S}(t) = \tilde{A}(t)\tilde{\Pi}(t) + \tilde{B}(t)\tilde{\Gamma}(t) + \tilde{P}(t) \quad (2.23)$$

se observa que es complicado obtener una solución para el regulador. Por otro lado es importante hacer notar que las soluciones locales resuelven el problema de regulación al menos en el punto de operación calculados, por lo tanto, la finalidad de este trabajo es buscar la óptima combinación entre reguladores locales tal que el error de seguimiento se reduzca.

2.5 Sistemas neuro-difusos

Los sistemas neuro-difusos se caracterizan por fusionar las capacidades de tratar con información incierta y de aprender o auto-ajustarse, de la lógica difusa y de las redes neuronales respectivamente.

Los sistemas basados en el mecanismo de inferencia de Tsukamoto son sistemas simples a partir de los cuales, se pueden construir fácilmente reglas de aprendizaje de las redes neuronales y en particular neuronas difusas.

La salida de un sistema de Tsukamoto está definida por:

$$z = \sum_{i=1}^n \frac{W_i z_i}{\sum_{i=1}^n W_i} \quad (2.24)$$

Los términos W_i serán las fuerzas de disparo de cada regla, regularmente la norma T opera a las funciones de membresía de la parte antecedente.

Por lo tanto el objetivo será obtener de forma autónoma los parámetros antecedentes y consecuentes del sistema difuso. Generalmente uno de los métodos más empleados es el descenso del gradiente, que a su vez es la base de la regla delta generalizada utilizada en las redes neuronales.

Un sistema adaptativo de inferencia neuro-difusa (ANFIS) básicamente es un sistema difuso que emplea la estructura y los algoritmos de aprendizaje de las redes neuronales para ajustar los parámetros antecedentes y consecuentes de un sistema difuso. En la figura 2.1 se muestra la arquitectura del sistema ANFIS.

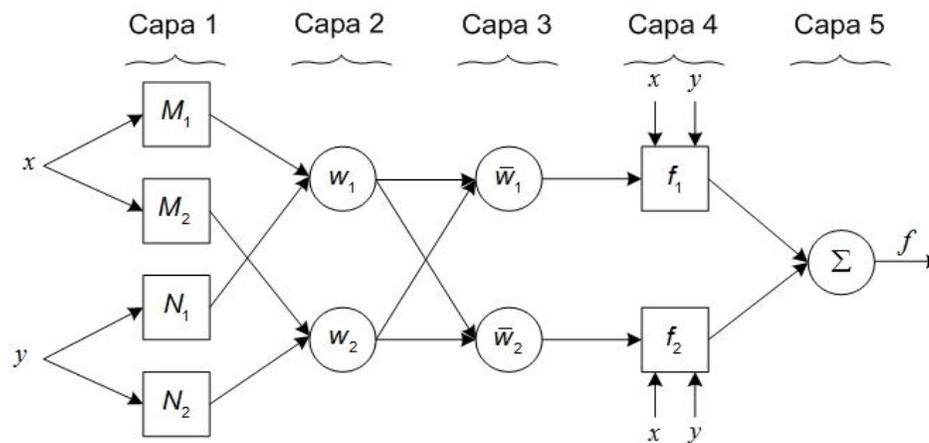


Figura 2.1. Estructura del Sistema Adaptativo de Inferencia Neuro-Difusa.

Capa 1:

Supóngase que la base de reglas difusas es de 2 reglas “Si – Entonces” del tipo Takagi-Sugeno:

Regla 1: Si x es M_1 & y es N_1 , entonces $f_1 = p_1x + q_1y + r_1$

Regla 2: Si x es M_2 & y es N_2 , entonces $f_2 = p_2x + q_2y + r_2$

Por lo tanto la salida en el nodo i (para este caso $i = 1,2,3,4; j = 1,2$), será:

$$O_i^1 = M_j(x) \quad i = 1,2; j = 1,2 \quad (2.25)$$

$$O_i^1 = N_j(x) \quad i = 3,4; j = 1,2 \quad (2.26)$$

Las ecuaciones (2.25) y (2.26) representan el grado de membresía de las entradas x & y para las variables lingüísticas y generalmente se eligen como funciones del tipo campana generalizada cuyo máximo y mínimo serán 1 y 0 respectivamente.

$$M_j(x) = \frac{1}{1 + \left(\frac{x - c_i}{a_i}\right)^{2b_i}}; \quad N_j(y) = \frac{1}{1 + \left(\frac{y - c_j}{a_j}\right)^{2b_j}} \quad (2.27)$$

Los parámetros a_i, b_i, c_i son conocidos como parámetros de premisa. Si se modifica el valor de estos parámetros, en consecuencia se modificará la forma de la función campana generalizada (Figura 2.2).

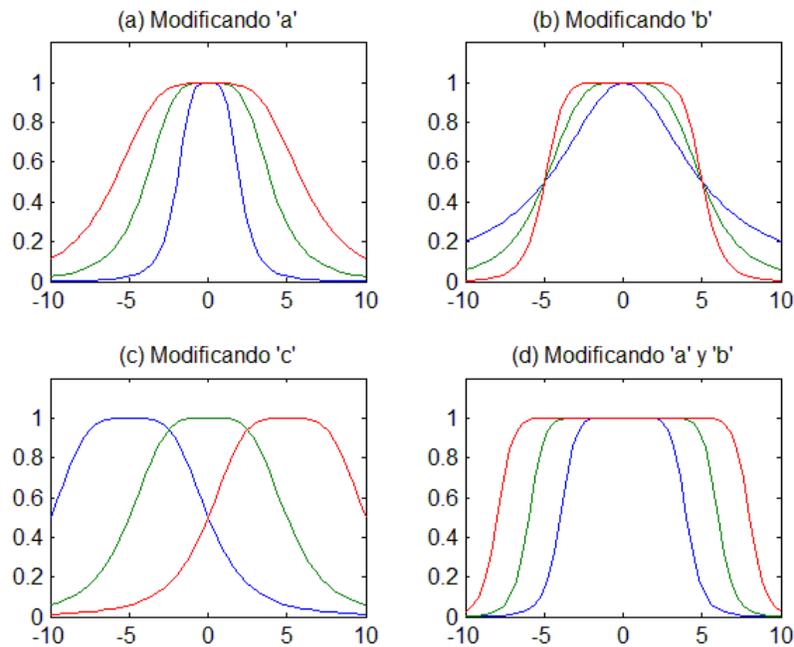


Figura 2.2. Formas de la función campana generalizada cuando sus parámetros se modifican.

Capa 2:

En esta se realizan las inferencias entre los conjuntos difusos, que representan la fuerza de disparo de cada regla.

$$w_j = \mu_{A_j}(x) \cdot \mu_{B_j}(x), j = 1, 2. \quad (2.28)$$

Capa 3:

Una vez obtenidas las inferencias, es necesario normalizarlas, en los nodos pertenecientes a la tercera capa se obtienen las fuerzas de disparo normalizadas:

$$\bar{w}_j = \frac{w_j}{w_1 + w_2} \quad (2.29)$$

Capa 4:

Cabe mencionar que la parte consecuente del sistema ANFIS está dada por un conjunto de funciones $f_i = p_i x + q_i y + r_i$, las cuales se integran en la cuarta capa y se ven afectadas por las fuerzas de disparo normalizadas obtenidas en la capa anterior:

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \quad (2.30)$$

Capa 5:

Finalmente en la última capa se obtiene la salida total:

$$f = O_1^5 = \sum_i O_i^4 = \sum_i \bar{w}_i f_i \quad (2.31)$$

2.6 Aprendizaje de una red neuronal: Método del descenso del gradiente.

Típicamente las redes neuronales corrigen el error a partir de un procedimiento denominado regla delta generalizada. Esta sugiere que el entrenamiento sea de la siguiente forma: se introduce en la red un dato que se propagará a través de ella, generando un conjunto de valores en las unidades de salida. Posteriormente la salida actual es comparada con la salida deseada calculándose el error, si el error es menor al máximo permitido, no se realizarán cambios en la red; en caso contrario se actualizan los valores de los parámetros de la red.

La actualización de los parámetros de una red neuronal supervisada o de un sistema neuro-difuso como el ANFIS, emplea el proceso de retro-propagación el cual consiste en encontrar un vector gradiente que vaya en dirección opuesta al flujo de salida de cada nodo de la red. Cuando se introduce el método del descenso del gradiente para calcular los parámetros de la red, se denomina aprendizaje de retro-propagación.

El método del descenso del gradiente es uno de los métodos más sencillos para minimizar funciones definidas en espacios multidimensionales, empleándose con mayor frecuencia en la optimización de sistemas no lineales y en consecuencia para entrenar redes neuronales.

Geoméricamente si se requiere minimizar una función $F(x)$ y se tiene un punto de referencia $x(k)$, se buscan los puntos a partir de $x(k)$ que proporcionen la dirección en que la función $F(x)$ decrezca con mayor rapidez. Esta dirección de descenso estará dada por el negativo del gradiente. Una analogía del descenso del gradiente sería la siguiente: “Suponga que está caminando en una ladera a través de la espesa niebla, y desea llegar al fondo del valle, aunque no pueda ver hacia adelante, es posible descender si se asegura de que cada paso se toma por la línea local de mayor pendiente” [47].

En 1847 Cauchy sienta las bases del descenso del gradiente al proponer en sus memorias un método para solucionar sistemas de ecuaciones simultáneas [48], en su trabajo Cauchy propone una función $u = F(x, y, z)$, donde se deben encontrar los valores x, y, z , que produzcan $u = 0$, de tal forma que la función descenderá en la dirección negativa del gradiente.

En su forma más simple el método de descenso del gradiente es:

$$\theta_{k+1} = \theta_k + \eta_k d_k \quad (2.32)$$

En dónde η_k es la constante de aprendizaje, y d_k el vector de dirección del gradiente en el punto local θ_k (Figura 2.3). Finalmente cabe mencionar que en las

redes neuronales y sistemas neuro-difusos la función a minimizar es la función error $E(\theta_k)$.

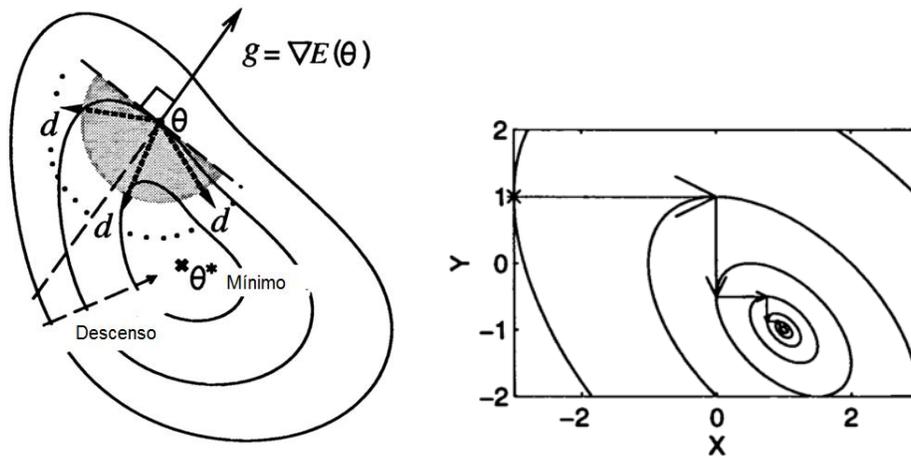


Figura 2.3. Interpretación geométrica del descenso del gradiente.

3.1 Regulación lineal

Considerando un sistema difuso T-S:

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^2 h_i(x_1(t)) [A_i x(t) + B_i u(t)], \\ \dot{\omega}(t) &= S \omega(t), \\ e(t) &= \sum_{i=1}^2 h_i(x_1(t)) C_i x(t) - Q \omega(t), \end{aligned} \quad (3.1)$$

con los valores:

$$A_1 = \begin{bmatrix} 0 & 1 \\ 2 & 0 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 1 \\ 3 & 0 \end{bmatrix}, B_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C_1 = C_2 = Q = [1 \quad 0], S = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

Con la señal de control:

$$u(t) = \underbrace{\sum_{i=1}^2 h_i(x_i(t)) K_i [x(t) - \Pi \omega(t)]}_{\text{estabilizador difuso}} + \underbrace{\sum_{i=1}^2 h_i(x_i(t)) \Gamma_i \omega(t)}_{\text{regulador difuso}} \quad (3.2)$$

Las funciones h_i en (3.1) y (3.2) se proponen como triangulares:

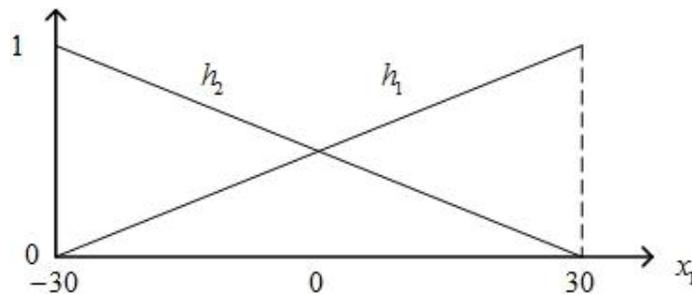


Figura 3.1. Funciones de membresía para el caso de regulación lineal.

Diseñando el regulador con base en los sistemas locales, los valores de las matrices Π y Γ se obtienen a partir del sistema de ecuaciones (2.6):

$$\pi(\omega(t)) = \Pi_1 \omega(t) = \Pi_2 \omega(t) \quad (3.3)$$

$$\pi(\omega(t)) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \Pi \omega(t) \quad (3.4)$$

$$\gamma(\omega(t)) = \sum_{i=1}^2 h_i(\omega_1(t)) \Gamma_i \omega(t) \quad (3.5)$$

$$\gamma(\omega(t)) = \begin{bmatrix} -\frac{3}{2} h_1(\omega_1(t)) - 4h_2(\omega_1(t)) \end{bmatrix} \omega(t) \quad (3.6)$$

$$K_1 = [-1.61 \quad -0.35] \quad (3.7)$$

$$K_2 = [-4.22 \quad -0.71] \quad (3.8)$$

las ganancias K_i son calculadas empleando desigualdades lineales matriciales (LMI's) [49]. El seguimiento del sistema anterior se muestra en la figura 3.2, como se observa en la figura 3.3 el error no tiende a 0 por tanto el sistema anterior no satisface la condición de regulación, esto se debe a que los términos de las matrices B_1 y B_2 son diferentes [4].

La idea básica es proporcionar al controlador la capacidad de adaptarse para seguir la referencia lo más exacto posible. Cuando Π es la matriz identidad y el error es 0, se observa en la ley de control que el estabilizador es 0, y se cumple la condición de seguimiento, en consecuencia la parte correspondiente al regulador será la que actuará exclusivamente, por lo tanto retomando la señal de control (3.2), se propone desacoplar el estabilizador y el regulador, con el objetivo de que la relación difusa que existe entre ambos sea diferente, en consecuencia las funciones de membresía que actúan en el estabilizador y en el regulador serán diferentes, como se observa en la ecuación (3.9).

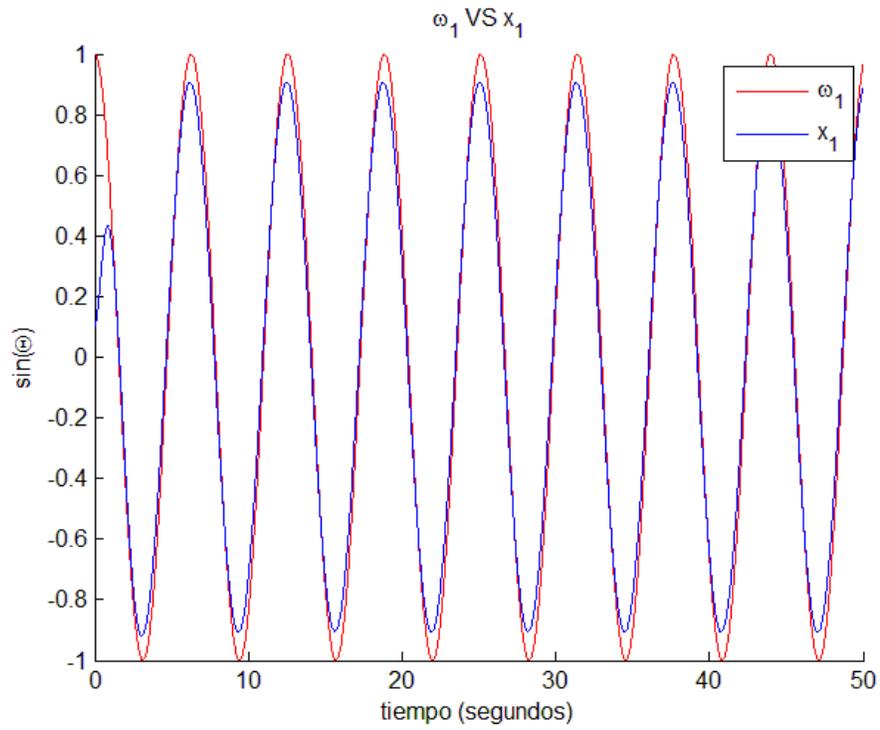


Figura 3.2. Respuesta del esquema de regulación con F.M. triangulares

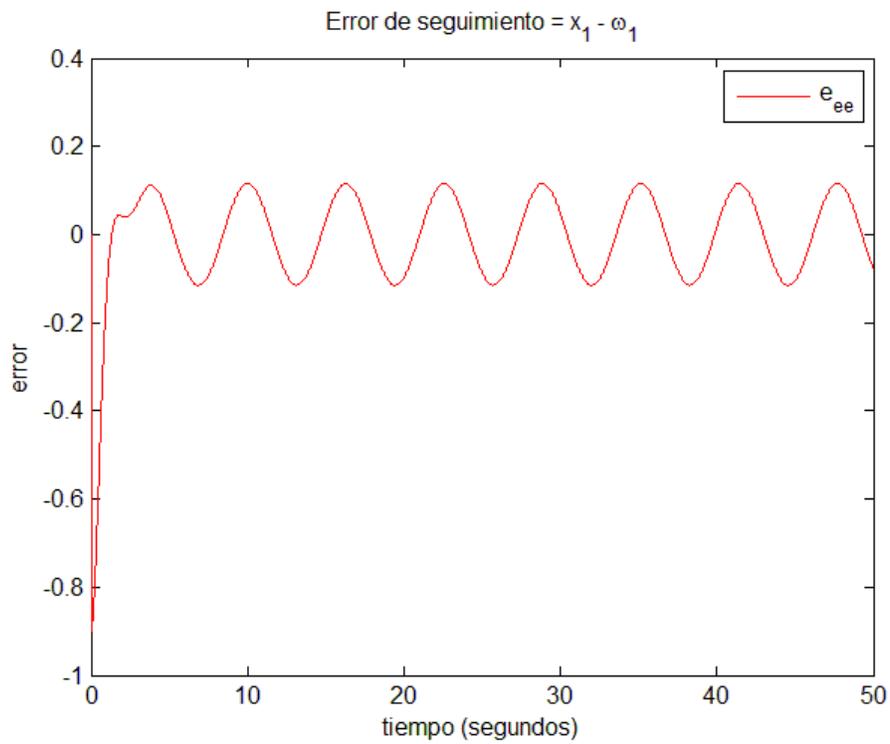


Figura 3.3. Error de seguimiento con F.M. triangulares.

$$u(t) = \underbrace{\sum_{i=1}^2 h_i(x_i(t)) K_i [x(t) - \Pi \omega(t)]}_{\text{estabilizador difuso}} + \underbrace{\sum_{i=1}^2 \mu_i(x_i(t)) \Gamma_i \omega(t)}_{\text{regulador difuso}} \quad (3.9)$$

El problema que se plantea en este trabajo tomando como referencia la ecuación (3.9), es encontrar las funciones μ_i , bajo la condición que las matrices Π_i del estabilizador sean iguales.

Con base en [5], un sistema difuso con funciones del tipo campana generalizada (Ec. (3.10)), es un aproximador universal, por esta razón se eligen como funciones de membresía del regulador, y estarán sujetas a la condición (3.11).

$$M_i = f(x, a_i, b_i, c_i) = \frac{1}{1 + \left| \frac{x - c_i}{a_i} \right|^{2b_i}} \quad (3.10)$$

$$M_2 = 1 - M_1 \quad (3.11)$$

El desempeño del controlador y específicamente del regulador, dependerá de la correcta sintonización de las funciones μ_i , siendo necesario ajustar los parámetros que determinan la forma, centro y amplitud de la función de membresía. En la literatura uno de los métodos más utilizados es el descenso del gradiente, para ajustar tanto los parámetros antecedentes (a_i, b_i, c_i) como consecuentes (z_i) de un sistema difuso [47, 50-52]:

$$a_i(k+1) = a_i(k) - \eta_{a_i} \frac{\partial E(a_i, b_i, c_i)}{\partial a_i} \quad i = 1, 2. \quad (3.12)$$

$$b_i(k+1) = b_i(k) - \eta_{b_i} \frac{\partial E(a_i, b_i, c_i)}{\partial b_i} \quad i = 1, 2. \quad (3.13)$$

$$c_i(k+1) = c_i(k) - \eta_{c_i} \frac{\partial E(a_i, b_i, c_i)}{\partial c_i} \quad i = 1, 2. \quad (3.14)$$

$$z_i(k+1) = z_i(k) - \eta_{z_i} \frac{\partial E(z_i)}{\partial z_i} \quad i = 1, 2. \quad (3.15)$$

Las expresiones (3.12)-(3.15) indican que para el instante $k + 1$ se actualizarán los valores a_i, b_i, c_i y z_i , hasta que el error en estado estable tienda a 0 ($e_{ee} \rightarrow 0$) lo que producirá:

$$a_i(k+1) \rightarrow a_i(k), b_i(k+1) \rightarrow b_i(k), c_i(k+1) \rightarrow c_i(k)$$

Se aprecia que el conjunto de expresiones (3.12)-(3.15) dependen del error $e_{ee}(k) = x(k) - \Pi\omega(k)$, donde $x(k)$ y $\omega(k)$ representan el k-ésimo par de entrenamiento del modelo difuso y del exosistema respectivamente, tal y como se aprecia en la figura 3.4.

El método del descenso del gradiente deberá reducir el error en la dirección de los parámetros antecedentes y consecuentes del controlador, por lo que es necesario modificar el conjunto de ecuaciones (3.12), (3.13), (3.14), (3.15), considerando la evolución de los estados del sistema así como la estructura del controlador (Figura 3.5).

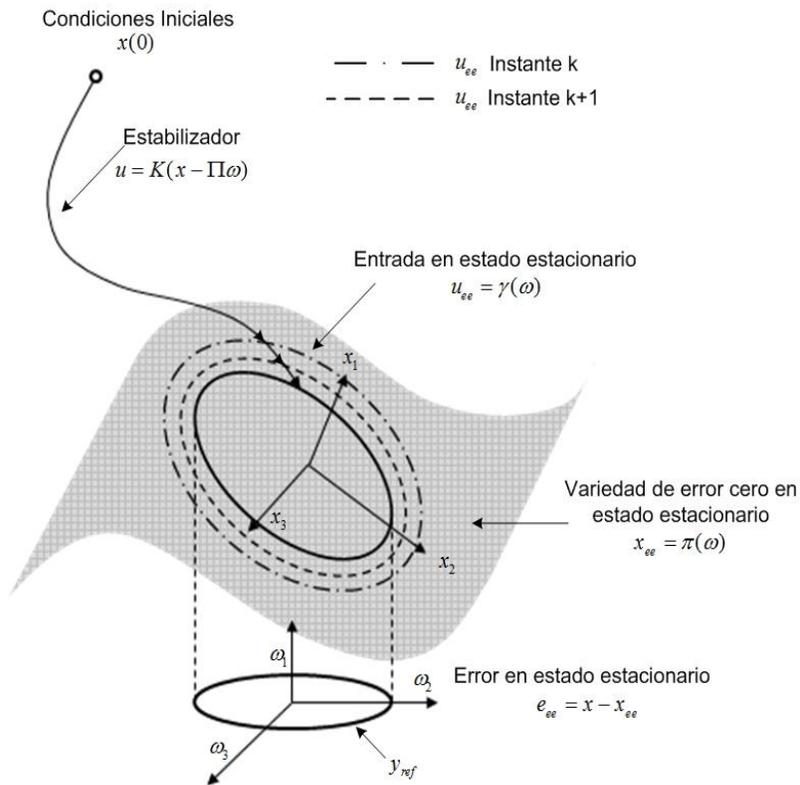


Figura 3.4. Esquema de la regulación no lineal en forma iterativa

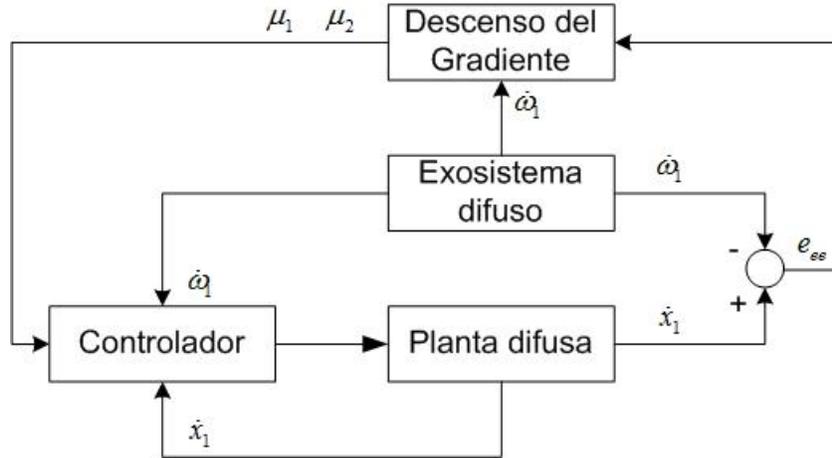


Figura 3.5. Diagrama de entrenamiento del regulador difuso.

Para llevar a cabo la sintonización de los parámetros del regulador se requieren pares de entrada-salida, para resolver este problema tanto la salida del sistema difuso como la del exosistema como se observa en (3.1), son continuas. Por lo anterior se propone discretizar la planta a partir de la siguiente aproximación:

$$\dot{x} \approx \frac{x_{k+1} - x_k}{T} \quad (3.16)$$

Sustituyendo (3.16) en (3.1):

$$x_{k+1} = \sum_{i=1}^2 h_i (A_i x_k + B_i u) T + x_k \quad (3.17)$$

Con: $u = (h_1 K_1 + h_2 K_2)(x_k - \Pi \omega_k) + \mu_1 \Gamma_1 \omega_k + \mu_2 \Gamma_2 \omega_k$

Considerando el error cuadrático medio para un par de datos:

$$e_{ee} = x - \Pi \omega \rightarrow e_{ee}(k+1) = x_{k+1} - \Pi \omega_{k+1} \Rightarrow E_{ee} = \frac{1}{2} (x_{k+1} - \Pi \omega_{k+1})^2 \quad (3.18)$$

El valor de los estados actuales x_k es función explícita de μ_1 y μ_2 , por lo tanto x_{k+1} dependerá implícitamente de los parámetros $a_1, b_1, c_1, a_2, b_2, c_2, z_1, z_2$. Puede observarse que es posible aplicar la condición $\mu_2 = 1 - \mu_1$, lo que producirá que sólo sea necesario ajustar los parámetros de la función μ_1 correspondiente al regulador del sistema local 1:

$$\frac{\partial E}{\partial a_i} = \left(\frac{\partial E}{\partial x_{k+1}} \right) \left(\frac{\partial x_{k+1}}{\partial u} \right) \left(\frac{\partial u}{\partial \mu_1} \right) \left(\frac{\partial \mu_1}{\partial w_i} \right) \left(\frac{\partial w_i}{\partial M_i} \right) \left(\frac{\partial M_i}{\partial a_i} \right) \quad (3.19)$$

$$\frac{\partial E}{\partial b_i} = \left(\frac{\partial E}{\partial x_{k+1}} \right) \left(\frac{\partial x_{k+1}}{\partial u} \right) \left(\frac{\partial u}{\partial \mu_1} \right) \left(\frac{\partial \mu_1}{\partial w_i} \right) \left(\frac{\partial w_i}{\partial M_i} \right) \left(\frac{\partial M_i}{\partial b_i} \right) \quad (3.20)$$

$$\frac{\partial E}{\partial c_i} = \left(\frac{\partial E}{\partial x_{k+1}} \right) \left(\frac{\partial x_{k+1}}{\partial u} \right) \left(\frac{\partial u}{\partial \mu_1} \right) \left(\frac{\partial \mu_1}{\partial w_i} \right) \left(\frac{\partial w_i}{\partial M_i} \right) \left(\frac{\partial M_i}{\partial c_i} \right) \quad (3.21)$$

$$\frac{\partial E}{\partial z_i} = \left(\frac{\partial E}{\partial x_{k+1}} \right) \left(\frac{\partial x_{k+1}}{\partial u} \right) \left(\frac{\partial u}{\partial \mu_1} \right) \left(\frac{\partial \mu_1}{\partial z_i} \right) \quad (3.22)$$

donde:

$$\frac{\partial E}{\partial x_{k+1}} = \frac{\partial \left(\frac{1}{2} (x_{k+1} - \Pi \omega_{k+1})^2 \right)}{\partial x_{k+1}} = (x_{k+1} - \Pi \omega_{k+1})$$

$$\frac{\partial x_{k+1}}{\partial u} = T \sum_{i=1}^n h_i B_i$$

$$\frac{\partial u}{\partial \mu_1} = \frac{\partial (\mu_1 \Gamma_1 \omega_k + (1 - \mu_1) \Gamma_2 \omega_k)}{\partial \mu_1} = \Gamma_1 \omega_k - \Gamma_2 \omega_k$$

$$\frac{\partial \mu_1}{\partial w_i} = \frac{\partial \left(\sum_{i=1}^n \bar{w}_i z_i \right)}{\partial w_i} = \frac{z_i - \mu_1}{\sum_{i=1}^n w_i}$$

$$\left(\frac{\partial w_i}{\partial M_i} \right) = 1$$

$$\frac{\partial \mu_1}{\partial z_i} = \frac{\partial \left(\sum_{i=1}^n \bar{w}_i z_i \right)}{\partial z_i} = \bar{w}_i$$

Finalmente las derivadas parciales de las funciones de campana generalizada son [53]:

$$\frac{\partial M_i}{\partial a_i} = \frac{2b_i}{a_i} M_i(1 - M_i) \quad (3.23)$$

$$\frac{\partial M_i}{\partial b_i} = \begin{cases} -\ln \left| \frac{\omega_k - c_i}{a} \right| M_i(1 - M_i), & \text{si } \omega_k \neq c_i \\ 0 & \text{si } \omega_k = c_i \end{cases} \quad (3.24)$$

$$\frac{\partial M_i}{\partial c_i} = \begin{cases} \frac{2b_i}{\omega_k - c_i} M_i(1 - M_i), & \text{si } \omega_k \neq c_i \\ 0 & \text{si } \omega_k = c_i \end{cases} \quad (3.25)$$

Los resultados de la simulación cuando se entrenan los parámetros antecedentes del controlador se muestran en figura 3.6, el controlador hace que el estado de la planta x_1 siga con buena precisión al estado ω_1 del exosistema, y en comparación con el regulador lineal (3.2) el error disminuye considerablemente (figura 3.7).

La tarea de entrenamiento del regulador requiere de un tiempo relativamente grande para alcanzar la señal de referencia, en un caso de aplicación de seguimiento en tiempo real esto no es deseado, ya que se requiere que en el menor tiempo la planta siga al exosistema, una vez entrenado el controlador es posible introducir los parámetros calculados al controlador, en cuya estructura se encuentra el sistema neuro-difuso, y como se observa en la figura 3.8 el sistema alcanza la señal de referencia en un menor tiempo. Cabe señalar que el tiempo del transitorio depende en gran medida del estabilizador, específicamente del valor de las ganancias K_i .

El método del descenso del gradiente tiene como principal inconveniente que puede quedar atrapado en un mínimo o máximo local [54, 55], y depende en gran medida de las condiciones iniciales, por esta razón sistemas como los propuestos en [18] y [43], introducen diferentes enfoques de entrenamiento que evitan los

inconvenientes de una red entrenada exclusivamente por el descenso del gradiente.

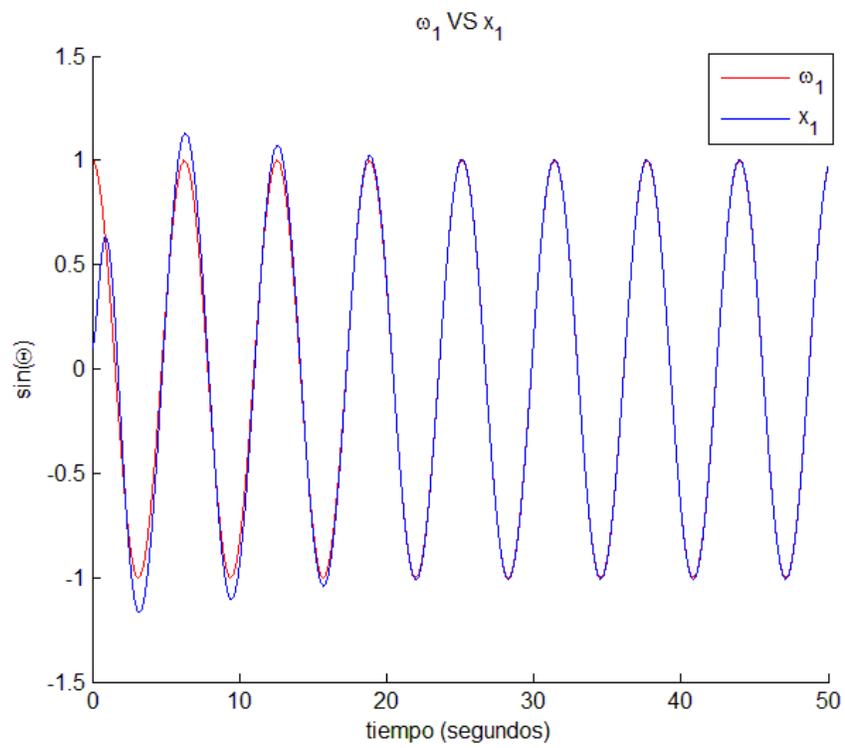


Figura 3.6. Evolución de los estados en la etapa de entrenamiento.

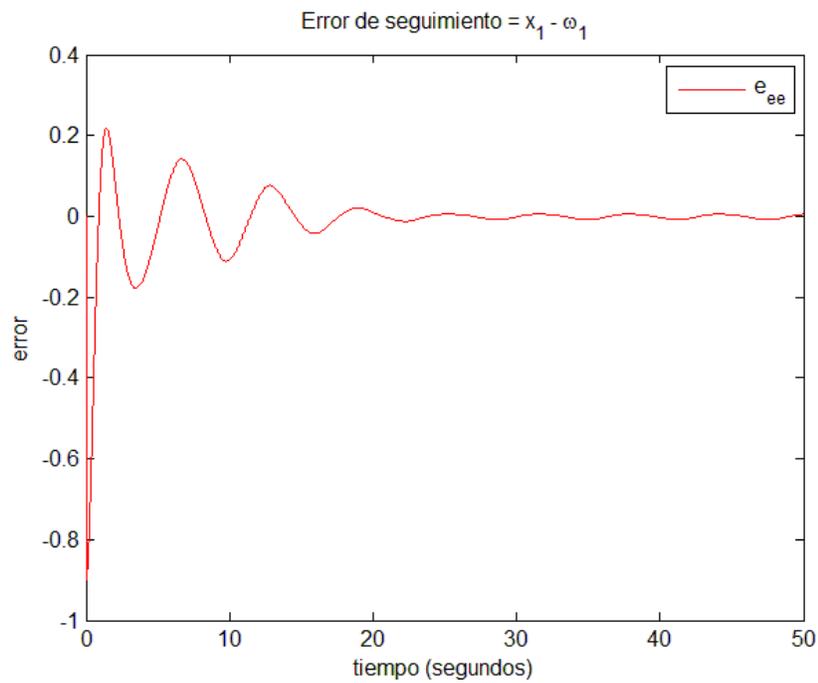


Figura 3.7. Error de seguimiento.

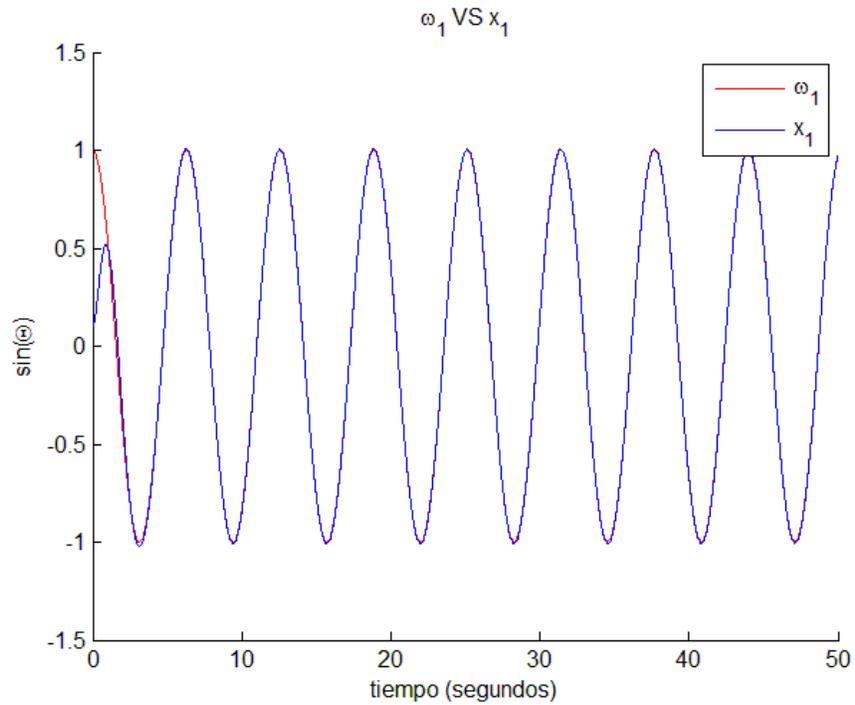


Figura 3.8. Respuesta de seguimiento cuando se ha sintonizado el controlador.

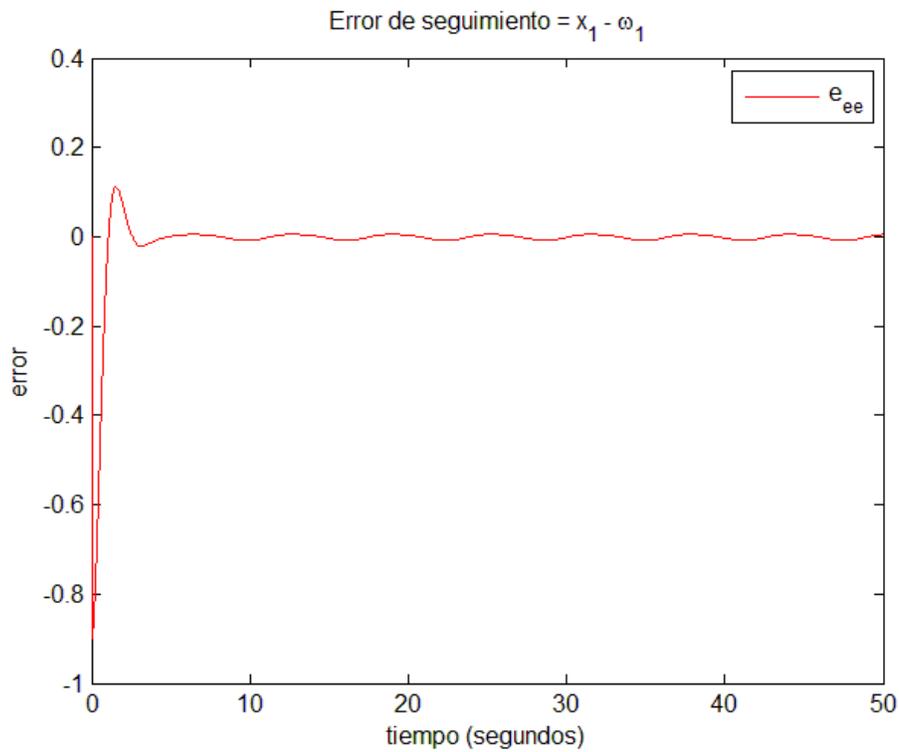


Figura 3.9. Error de seguimiento con el controlador sintonizado.

3.2 Integración del sistema ANFIS en la regulación difusa

Se desea que el sistema ANFIS proporcione el valor de las relaciones difusas entre los reguladores locales, este valor dependerá de la señal de referencia, el esquema que se propone es el siguiente:

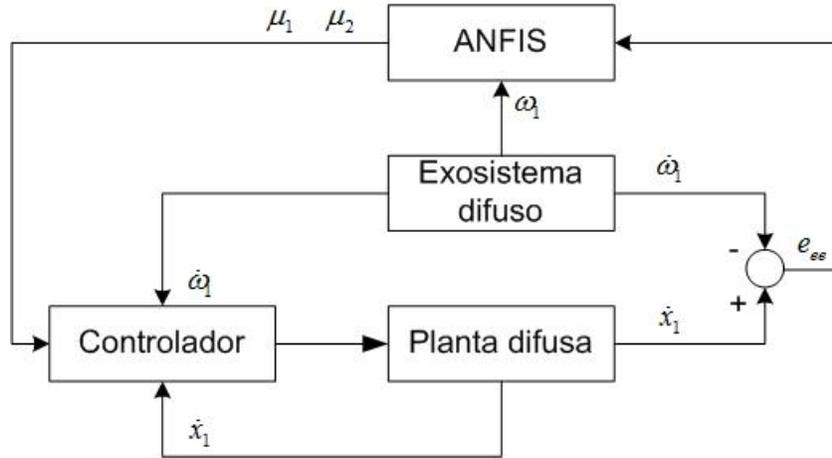


Figura 3.10. Integración del sistema ANFIS al ajuste de reglas.

Como se ha mencionado anteriormente los valores del regulador para cada subsistema son calculados a partir de la solución de (2.22), esto significa que cuando la función de membresía h_1 sea 1, el regulador tomará el valor de Γ_1 y cuando h_2 sea igual a 1 el de Γ_2 , estableciéndose las siguientes premisas:

$$\text{SI } h_1 \text{ es 1 ENTONCES } \mu_1 = 1 \rightarrow u(t) = h_1 K_1 [x(t) - \Pi \omega(t)] + \mu_1 \Gamma_1 \omega(t)$$

$$\text{SI } h_2 \text{ es 1 ENTONCES } \mu_2 = 1 \rightarrow u(t) = h_2 K_2 [x(t) - \Pi \omega(t)] + \mu_2 \Gamma_2 \omega(t)$$

Por lo anterior se propone integrar un sistema adaptativo de inferencia neuro-difusa (ANFIS) [53] que calcule las relaciones difusas entre los reguladores locales:

$$\text{SI } h_1 \text{ es 1, ENTONCES } u_{reg}(t) = \Gamma_1$$

$$\text{SI } h_2 \text{ es 1, ENTONCES } u_{reg}(t) = \Gamma_2$$

Dadas las anteriores reglas difusas y de acuerdo con [4] las funciones de membresía μ_i dependen de las funciones h_i , para este caso h_1 es elegida como único dato de entrada de entrenamiento. La figura 3.11 muestra la estructura de la red cuando se tiene una entrada, la cual se propaga en 5 capas de n nodos:

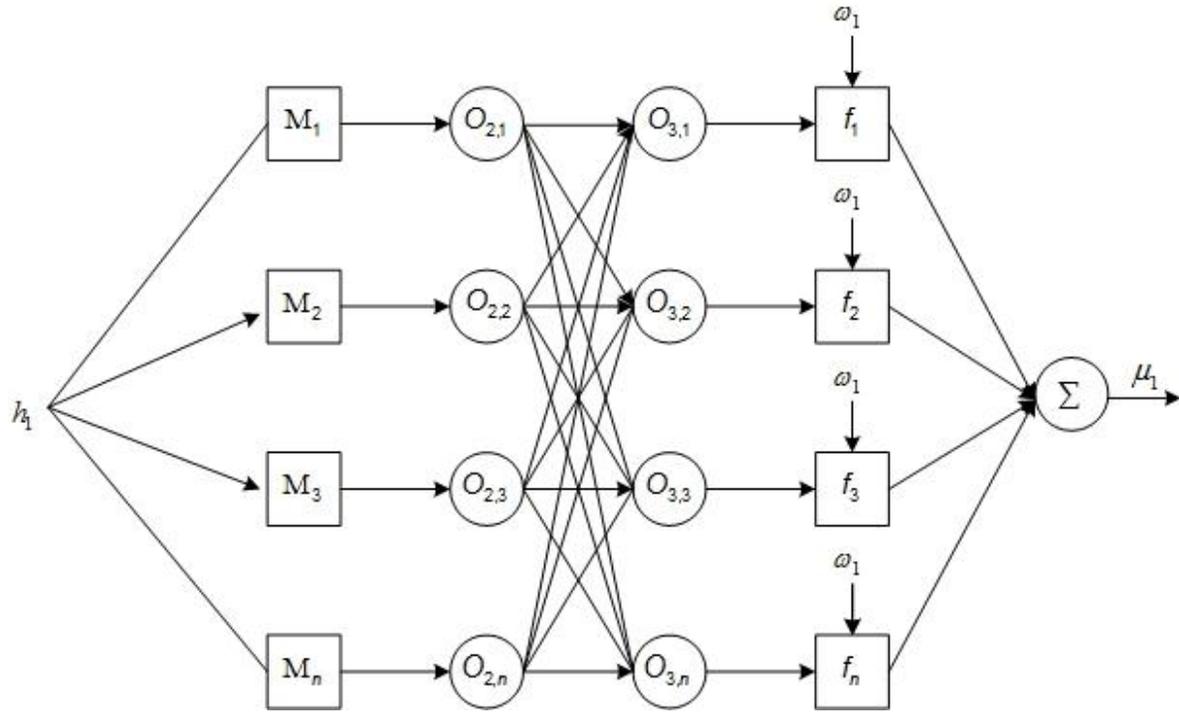


Figura 3.11. Arquitectura del sistema ANFIS para una variable de entrada

Capa 1:

Para este caso de estudio se eligen funciones de membresía de campana generalizada distribuidas en 5 nodos, las cuales deben estar distribuidas uniformemente [56] (Figura 3.12):

$$\begin{aligned}
 a_1 &= .2 & b_1 &= 2 & c_1 &= 0 \\
 a_2 &= .2 & b_2 &= 2 & c_2 &= 0.25 \\
 a_3 &= .2 & b_3 &= 2 & c_3 &= 0.5 \\
 a_4 &= .2 & b_4 &= 2 & c_4 &= 0.75 \\
 a_5 &= .2 & b_5 &= 2 & c_5 &= 1
 \end{aligned}$$

$$M_i(h_1) = \frac{1}{1 + \left(\frac{h_1 - c_i}{a_i}\right)^{2b_i}} \quad i = 1, 2, \dots, 5; \quad 0 \geq h_1 \leq 1$$

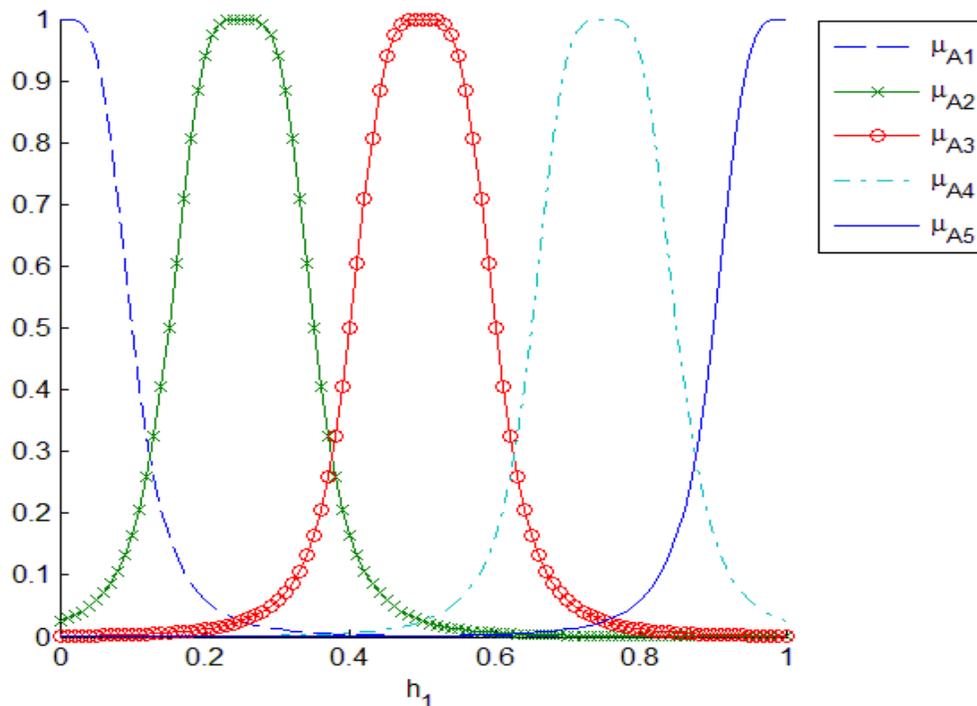


Figura 3.12. Distribución de las funciones de membresía.

Capa 2:

Debido a que sólo se tiene una variable de entrenamiento, los pesos que corresponderán a cada función serán los de su propia función de membresía:

$$w_1 = M_1; w_2 = M_2; w_3 = M_3; w_4 = M_4; w_5 = M_5;$$

Capa 3:

Normalización de los pesos:

$$\bar{w}_1 = \frac{w_1}{w_1 + w_2 + w_3 + w_4 + w_5}$$

$$\bar{w}_2 = \frac{w_2}{w_1 + w_2 + w_3 + w_4 + w_5}$$

$$\bar{w}_3 = \frac{w_3}{w_1 + w_2 + w_3 + w_4 + w_5}$$

$$\bar{w}_4 = \frac{w_4}{w_1 + w_2 + w_3 + w_4 + w_5}$$

$$\bar{w}_5 = \frac{w_5}{w_1 + w_2 + w_3 + w_4 + w_5}$$

Capa 4:

Coefficientes de Takagi-Sugeno:

$$f_i = \bar{w}_i (p_i h_1 + r_i) \quad \text{para } i = 1, 2, \dots, 5.$$

Capa 5:

Salida total:

$$f = f_1 + f_2 + f_3 + f_4 + f_5$$

3.3 Ajuste de los parámetros de la capa de entrada

Tal y como se trató en el capítulo 2, el método del descenso del gradiente ajustará los parámetros a_i, b_i, c_i , en función del error (ecuaciones (3.12)-(3.14)), no obstante de acuerdo al esquema de la figura 1.1 el error que se retro-propaga es $e_{ee} = x - \Pi\omega$, por consiguiente se aplica la regla de la cadena en las direcciones de descenso deseadas:

$$\frac{\partial E}{\partial a_i} = \frac{\partial E}{\partial x} \frac{\partial x}{\partial \mu_1} \frac{\partial \mu_1}{\partial f_i} \frac{\partial f_i}{\partial w_i} \frac{\partial w_i}{\partial M_i} \frac{\partial M_i}{\partial a_i} \tag{3.26}$$

$$\frac{\partial E}{\partial b_i} = \frac{\partial E}{\partial x} \frac{\partial x}{\partial \mu_1} \frac{\partial \mu_1}{\partial f_i} \frac{\partial f_i}{\partial w_i} \frac{\partial w_i}{\partial M_i} \frac{\partial M_i}{\partial b_i} \tag{3.27}$$

$$\frac{\partial E}{\partial c_i} = \frac{\partial E}{\partial x} \frac{\partial x}{\partial \mu_1} \frac{\partial \mu_1}{\partial f_i} \frac{\partial f_i}{\partial w_i} \frac{\partial w_i}{\partial M_i} \frac{\partial M_i}{\partial c_i} \tag{3.28}$$

$$\frac{\partial E}{\partial x} = (x - \Pi\omega) \quad (3.29)$$

$$\frac{\partial x_1}{\partial \mu_1} = [(\Gamma_1 - \Gamma_2)\omega]T \quad (3.30)$$

$$\frac{\partial \mu_1}{\partial w_i} = \frac{\partial \left(\sum_{i=1}^n \bar{w}_i z_i \right)}{\partial w_i} = \frac{p_i x + r_i - f}{\sum_{i=1}^n w_i} \quad (3.31)$$

$$\frac{\partial w_i}{\partial \mu_{A_i}} = 1 \quad (3.32)$$

Para los términos: $\frac{\partial \mu_{A_i}}{\partial a_1}$, $\frac{\partial \mu_{B_i}}{\partial a_1}$, $\frac{\partial \mu_{C_i}}{\partial a_1}$, se emplean las expresiones(3.23)-(3.25).

Resultando que la derivada del error con respecto a los parámetros a ajustar sea:

$$\frac{\partial E}{\partial a_i} = (x - \Pi\omega)^T [(\Gamma_1 - \Gamma_2)\omega]T \left(\frac{p_i x + r_i - f}{\sum w_i} \right) \left(\frac{\partial M_i}{\partial a_i} \right) \quad (3.33)$$

$$\frac{\partial E}{\partial b_i} = (x - \omega)^T [(\Gamma_1 - \Gamma_2)\omega]T \left(\frac{p_i x + r_i - f}{\sum w_i} \right) \left(\frac{\partial M_i}{\partial b_i} \right) \quad (3.34)$$

$$\frac{\partial E}{\partial c_i} = (x - \Pi\omega)^T [(\Gamma_1 - \Gamma_2)\omega]T \left(\frac{p_i x + r_i - f}{\sum w_i} \right) \left(\frac{\partial M_i}{\partial c_i} \right) \quad (3.35)$$

Con las expresiones anteriores se ajustan cada una de las n funciones que se propongan en la capa de entrada (parámetros antecedentes) del ANFIS [57].

3.4 Aprendizaje de los parámetros consecuentes empleando el método de Levenberg-Marquardt.

La regla básica de aprendizaje del ANFIS es el algoritmo de retro-propagación del descenso del gradiente, que como se ha venido mencionando a lo largo de esta tesis, calcula las derivadas de la función error con respecto a cada nodo de salida de la red desde la capa de salida hacia la capa de entrada. Los valores de los

parámetros de salida correspondientes a la salida total f (en este caso μ_1), pueden ser expresados como una combinación lineal de los parámetros consecuentes.

Por la anterior razón Jang propone un algoritmo de aprendizaje híbrido combinando los métodos de descenso del gradiente y mínimos cuadrados recursivos, para una búsqueda efectiva de los parámetros óptimos. Por otra parte ANFIS permite el cálculo sistemático de los vectores gradientes de los parámetros que pueden ser modificados, brindando la oportunidad de aplicar técnicas de regresión no lineal y optimización basadas en el gradiente.

Como se observa en las reglas de entrenamiento obtenidas en (3.26)-(3.28), la salida del ANFIS no es directamente la salida deseada, factor a considerar para sustituir el estimador de mínimos cuadrados ya que el sistema depende de los pares de entrada y salida, sin embargo dicha salida se desconoce ya que es la no-linealidad del sistema o la relación difusa existente entre sistemas locales, por lo tanto se procede a implementar en el entrenamiento el método de Levenberg-Marquardt [58]:

$$\Theta_{k+1} = \Theta_k - (J^T J + \lambda I)^{-1} \left(\frac{1}{2} \right) g \quad (3.36)$$

Donde el parámetro I es la matriz identidad, J la matriz Jacobiana, g el vector gradiente y λ es usualmente un valor escalar positivo pequeño. Dependiendo el valor asignado a λ , el método hará una transición entre el método de Gauss-Newton ($\lambda \rightarrow 0$) que será más eficiente pero menos estable, y el método del descenso del gradiente ($\lambda \rightarrow \infty$) con menor eficiencia pero mayor estabilidad.

La salida resultante del controlador integrando el entrenamiento en la parte consecuente por L-M se presenta en las siguientes figuras:

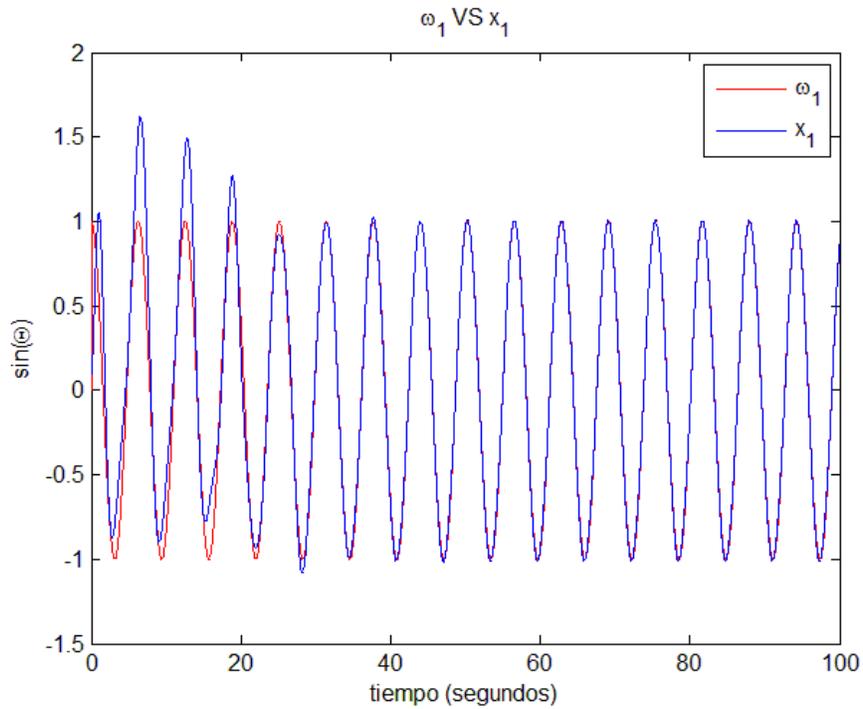


Figura 3.13. Respuesta del controlador integrando el método de L-M al ANFIS.

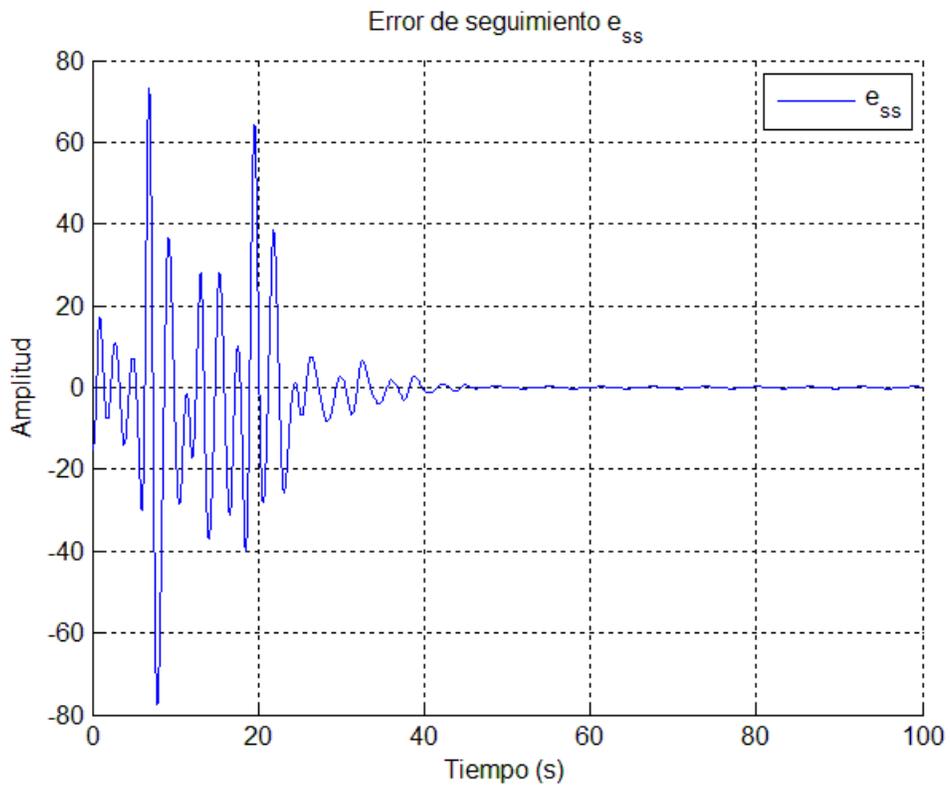


Figura 3.14. Error de seguimiento.

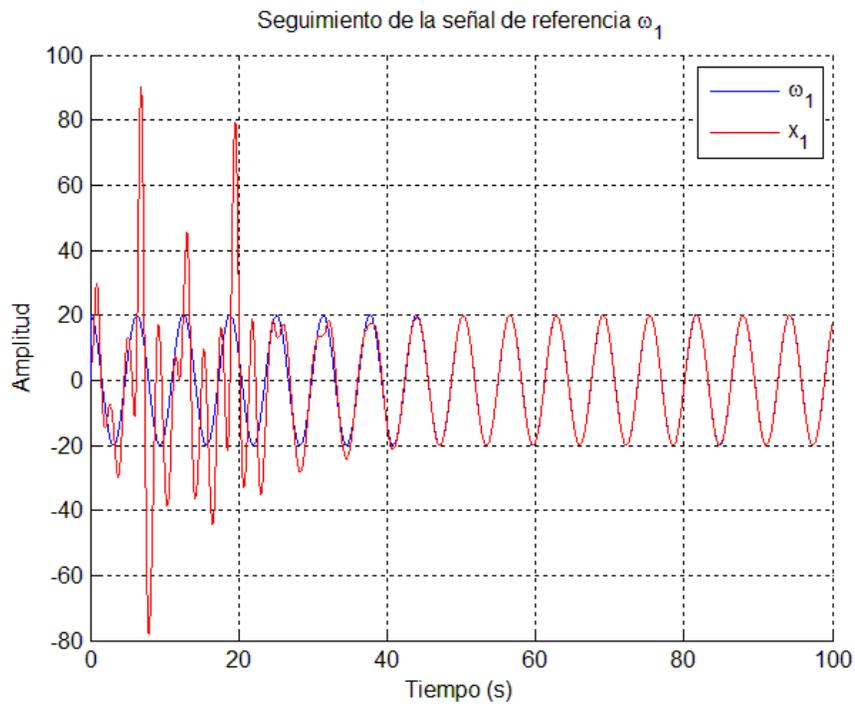


Figura 3.15. Proceso de entrenamiento aplicando el método de L-M al ANFIS.

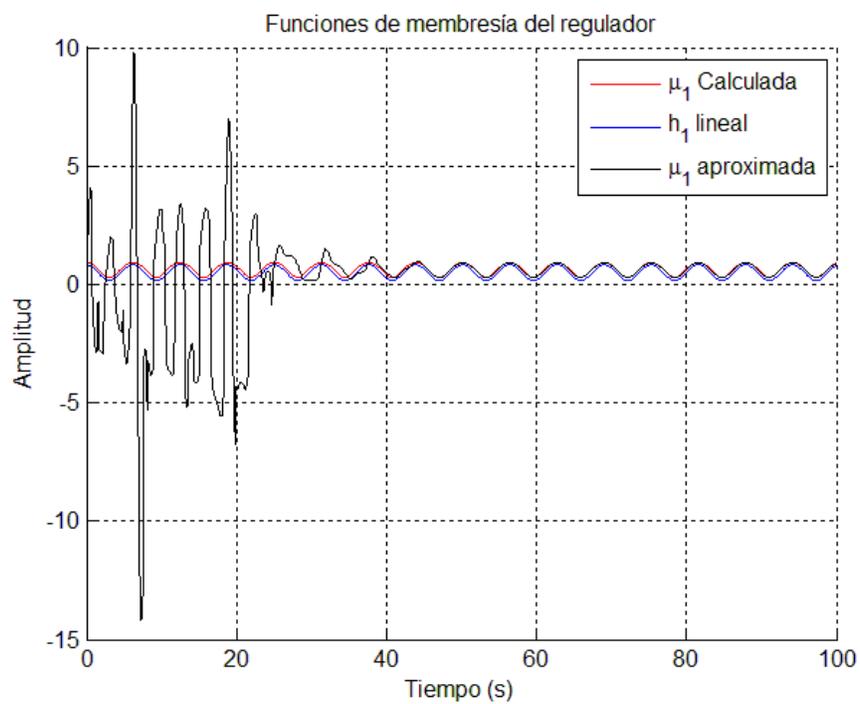


Figura 3.16. Aproximación del ANFIS a la no linealidad de la relación difusa.

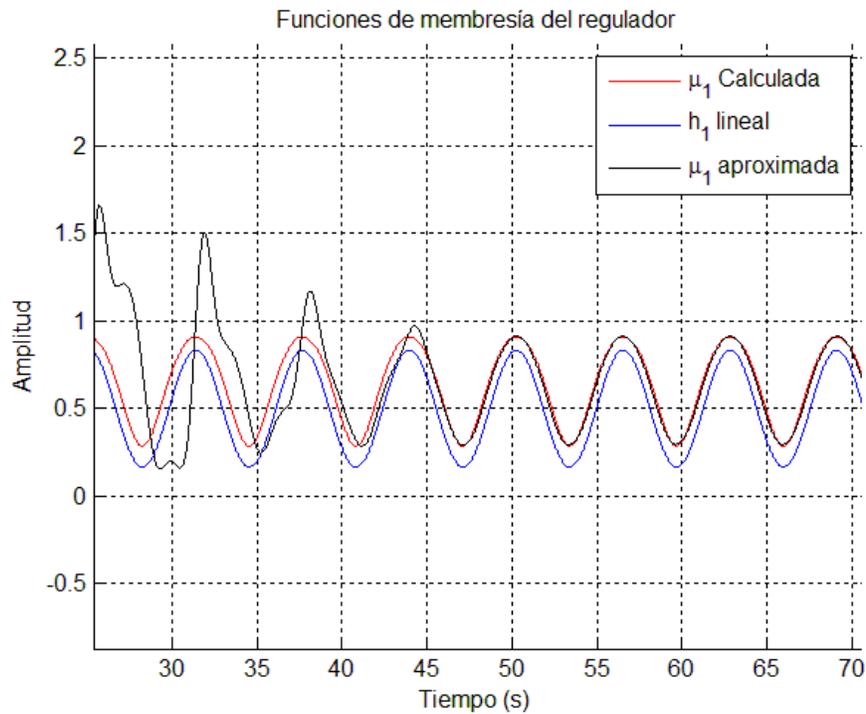


Figura 3.17. Aproximación del ANFIS a la no linealidad de la relación difusa.

Se observó que el sistema ANFIS entrenado con el algoritmo de Levenberg-Marquardt mejora la aproximación de la señal de seguimiento, esto se debe a que en su estructura el método combina eficientemente tanto la matriz Jacobina como la función gradiente del error.

4.1 Seguimiento de referencias en un robot de 2 grados de libertad.

Como primer caso de estudio se propone un manipulador de 2 grados de libertad, con los intervalos de movimiento de las articulaciones:

$$-30 \leq \theta_1 \leq 90$$

$$-90 \leq \theta_2 \leq 90$$

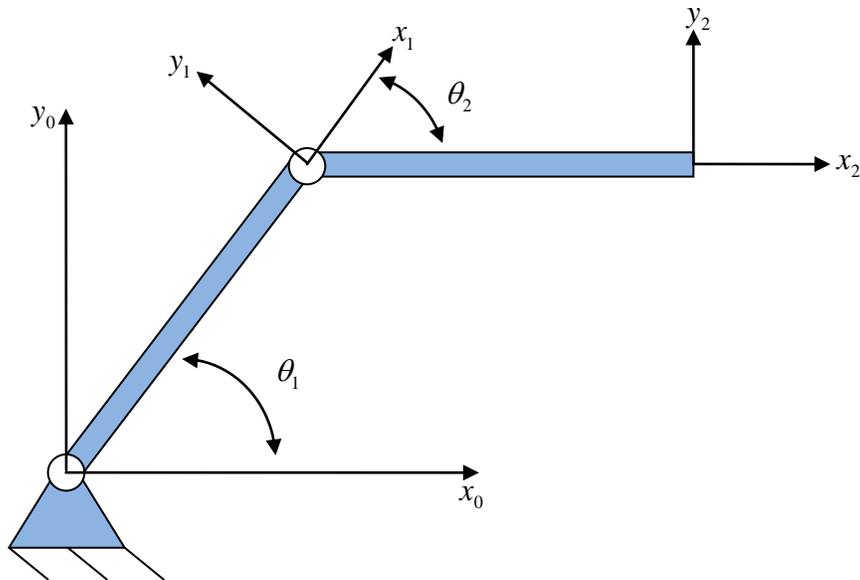


Figura 4.1. Caso 1: Robot de 2 grados de libertad

El espacio de trabajo del manipulador se muestra en la figura 4.2. El modelo dinámico del robot se resolverá aplicando la ecuación de Euler-Lagrange:

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\theta}_i} \right) - \frac{\partial L}{\partial \theta_i} = \tau_i \quad (4.1)$$

Donde $L = \text{Energía Cinética} - \text{Energía Potencial}$ y cuya ecuación diferencial característica para robots manipuladores de cadena abierta con articulaciones de revolución es:

$$M(\theta)\ddot{\theta} + C(\dot{\theta}, \theta) + G(\theta) = \tau \quad (4.2)$$

Realizando los respectivos cambios de variable:

$$\begin{aligned} \theta_1 &= x_1 & \theta_2 &= x_2 \\ \dot{\theta}_1 &= \dot{x}_1 = x_3 & \dot{\theta}_2 &= \dot{x}_2 = x_4 \\ \ddot{\theta}_1 &= \dot{x}_3 & \ddot{\theta}_2 &= \dot{x}_4 \end{aligned}$$

la expresión (4.2) en el espacio de estados se escribe como:

$$\frac{d}{dt} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ M^{-1}(\theta) [\tau - C(\theta, \dot{\theta}) - G(\theta)] \end{bmatrix} \quad (4.3)$$

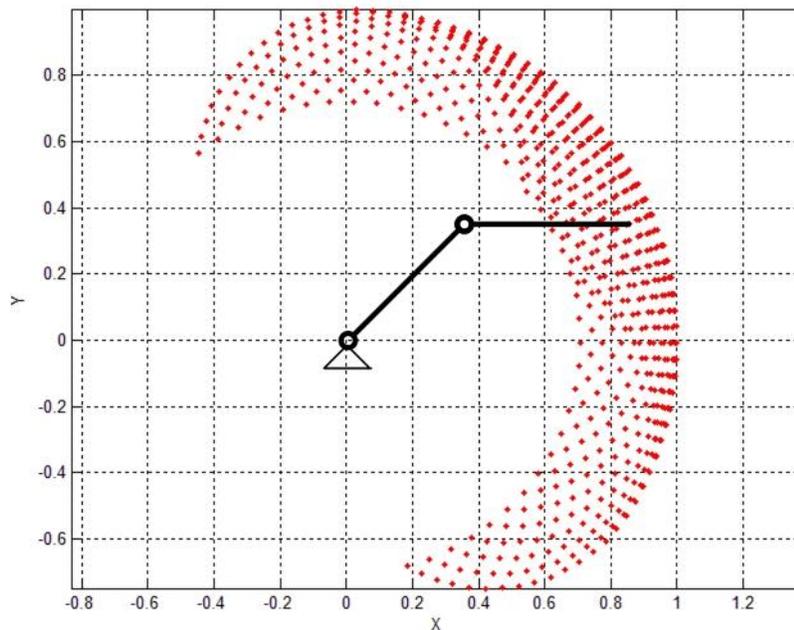


Figura 4.2. Espacio de trabajo propuesto

Las matrices de inercia, de fuerzas centrífugas y de Coriolis, así como de pares debidos a la gravedad son:

$$M(\theta) = \begin{bmatrix} \left(\frac{1}{3}\right)l^2 + \left(\frac{4}{3}\right)m_2l^2 + m_2 \cos(\theta_2)l^2 & \left(\frac{1}{3}\right)m_2l^2 + \left(\frac{1}{2}\right)m_2l^2 \cos(\theta_2) \\ \left(\frac{1}{3}\right)m_2l^2 + \left(\frac{1}{2}\right)m_2l^2 \cos(\theta_2) & \left(\frac{1}{3}\right)m_2l^2 \end{bmatrix}$$

$$C(\theta, \dot{\theta}) = \begin{bmatrix} -\left(\frac{1}{2}\right)m_2 \sin(\theta_2)l^2 \dot{\theta}_2^2 - m_2 \sin(\theta_2)l^2 \dot{\theta}_1 \dot{\theta}_2 \\ \left(\frac{1}{2}\right)m_2 \sin(\theta_2)l^2 \dot{\theta}_1^2 \end{bmatrix}$$

$$G = \begin{bmatrix} \left(\frac{1}{2}\right)m_1gl \cos(\theta_1) + \left(\frac{1}{2}\right)m_2gl \cos(\theta_1 + \theta_2) + m_2gl \cos(\theta_1) \\ \left(\frac{1}{2}\right)m_2gl \cos(\theta_1 + \theta_2) \end{bmatrix}$$

Considerando los puntos de linealización: $f_1(-30, -90)$, $f_2(-30, 90)$, $f_3(90, -90)$, y $f_4(90, 90)$, se lleva el sistema dinámico no lineal a un modelo difuso, con funciones de membresía lineales (Figura 4.3).

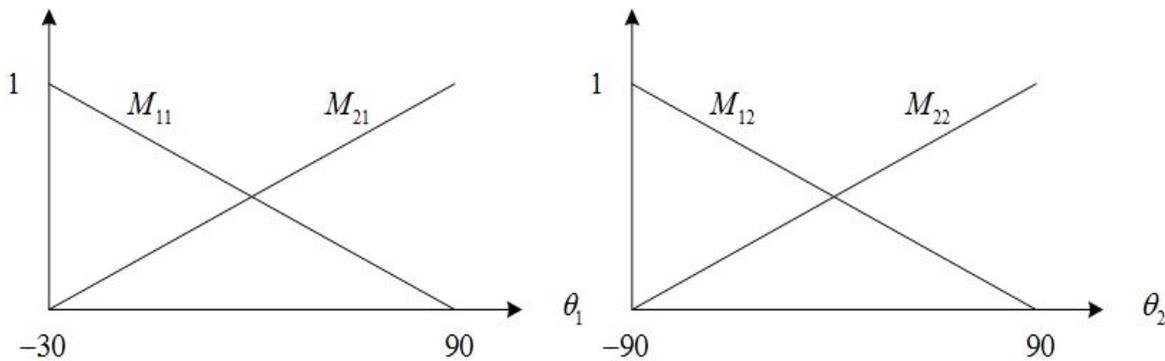


Figura 4.3. Funciones de membresía con θ_1 y θ_2 como variables de entrada.

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^n h_i \{A_i x(t) + B_i u(t)\} \\ y(t) &= \sum_{i=1}^n h_i \{C_i x(t)\} \end{aligned} \tag{4.4}$$

donde:

$$A = \left. \frac{\partial F(x)}{\partial x} \right|_{x_0, u_0} ; \quad B = \left. \frac{\partial F(x)}{\partial u} \right|_{x_0, u_0} ; \quad C = \left. \frac{\partial H(x)}{\partial x} \right|_{x_0, u_0}$$

por lo tanto la matrices de estado y de entrada serán:

$$A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -11.3192 & 0 & 0 & 0 \\ -14.1678 & -25.4871 & 0 & 0 \end{bmatrix} \quad A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -11.3192 & 0 & 0 & 0 \\ 36.8063 & 25.4871 & 0 & 0 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 22.6384 & 0 & 0 & 0 \\ -22.6384 & 0 & 0 & 0 \end{bmatrix} \quad A_4 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 22.6384 & 0 & 0 & 0 \\ -22.6384 & 0 & 0 & 0 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 9.2307 & -9.2307 \\ -9.2307 & 49.2307 \end{bmatrix} \quad B_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 9.2307 & -9.2307 \\ -9.2307 & 49.2307 \end{bmatrix}$$

$$B_3 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 9.2307 & -9.2307 \\ -9.2307 & 49.2307 \end{bmatrix} \quad B_4 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 9.2307 & -9.2307 \\ -9.2307 & 49.2307 \end{bmatrix}$$

$$S = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}$$

De los valores anteriores, es posible obtener los valores de Π y Γ a partir de la solución del sistema (2.6), dando como resultado:

$$\Pi_1 = \Pi_2 = \Pi_3 = \Pi_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\Gamma_1 = \begin{bmatrix} 1.7301 & 0.61218 & 0 & 0 \\ 0.61218 & 0.61218 & 0 & 0 \end{bmatrix}$$

$$\Gamma_2 = \begin{bmatrix} 0.45573 & -0.66218 & 0 & 0 \\ -0.66218 & -0.66218 & 0 & 0 \end{bmatrix}$$

$$\Gamma_3 = \begin{bmatrix} -2.5858 & -0.025 & 0 & 0 \\ -0.025 & -0.025 & 0 & 0 \end{bmatrix}$$

$$\Gamma_4 = \begin{bmatrix} -2.5858 & -0.025 & 0 & 0 \\ -0.025 & -0.025 & 0 & 0 \end{bmatrix}$$

$$K_1 = \begin{bmatrix} 1.2289 & 0.5579 & -0.5827 & -0.0916 \\ 0.5262 & 0.5153 & -0.1056 & -0.1105 \end{bmatrix}$$

$$K_2 = \begin{bmatrix} -0.0454 & -0.7164 & -0.5827 & -0.0916 \\ -0.7481 & -0.759 & -0.1056 & -0.1105 \end{bmatrix}$$

$$K_3 = \begin{bmatrix} -3.0869 & -0.0792 & -0.5827 & -0.0916 \\ -0.1109 & -0.1218 & -0.1056 & -0.1105 \end{bmatrix}$$

$$K_4 = \begin{bmatrix} -3.0869 & -0.0792 & -0.5827 & -0.0916 \\ -0.1109 & -0.1218 & -0.1056 & -0.1105 \end{bmatrix}$$

La base de reglas por lo tanto será:

Regla 1:

$$\text{Si } \theta_1 = M_{11} \text{ y } \theta_2 = M_{12} \text{ ENTONCES } \dot{x}(t) = A_1 x(t) + B_1 u(t)$$

Regla 2:

SI $\theta_1 = M_{11}$ y $\theta_2 = M_{22}$ ENTONCES $\dot{x}(t) = A_2x(t) + B_2u(t)$

Regla 3:

SI $\theta_1 = M_{21}$ y $\theta_2 = M_{12}$ ENTONCES $\dot{x}(t) = A_3x(t) + B_3u(t)$

Regla 4:

SI $\theta_1 = M_{21}$ y $\theta_2 = M_{22}$ ENTONCES $\dot{x}(t) = A_4x(t) + B_4u(t)$

Los valores h_i se obtienen a partir de la norma T:

$$h_1 = M_{11}(\theta_1(t)) \times M_{12}(\theta_2(t))$$

$$h_2 = M_{11}(\theta_1(t)) \times M_{22}(\theta_2(t))$$

$$h_3 = M_{21}(\theta_1(t)) \times M_{12}(\theta_2(t))$$

$$h_4 = M_{21}(\theta_1(t)) \times M_{22}(\theta_2(t))$$

La señal de control para cuatro reglas difusas tomará la siguiente forma:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = (h_1K_1 + h_2K_2 + h_3K_3 + h_4K_4)(x - \Pi\omega) + (h_1\Gamma_1 + h_2\Gamma_2 + h_3\Gamma_3 + h_4\Gamma_4)\omega$$

Cabe señalar que los valores de las B_i son iguales para cada uno de los subsistemas locales, por lo tanto se espera que el seguimiento en ambas articulaciones sea exacto. En la figura 4.4 se presenta el diagrama en Simulink dónde se simuló el control, en la figura 4.5 y figura 4.6 se observa el seguimiento en la primera y segunda articulación respectivamente para el exosistema propuesto:

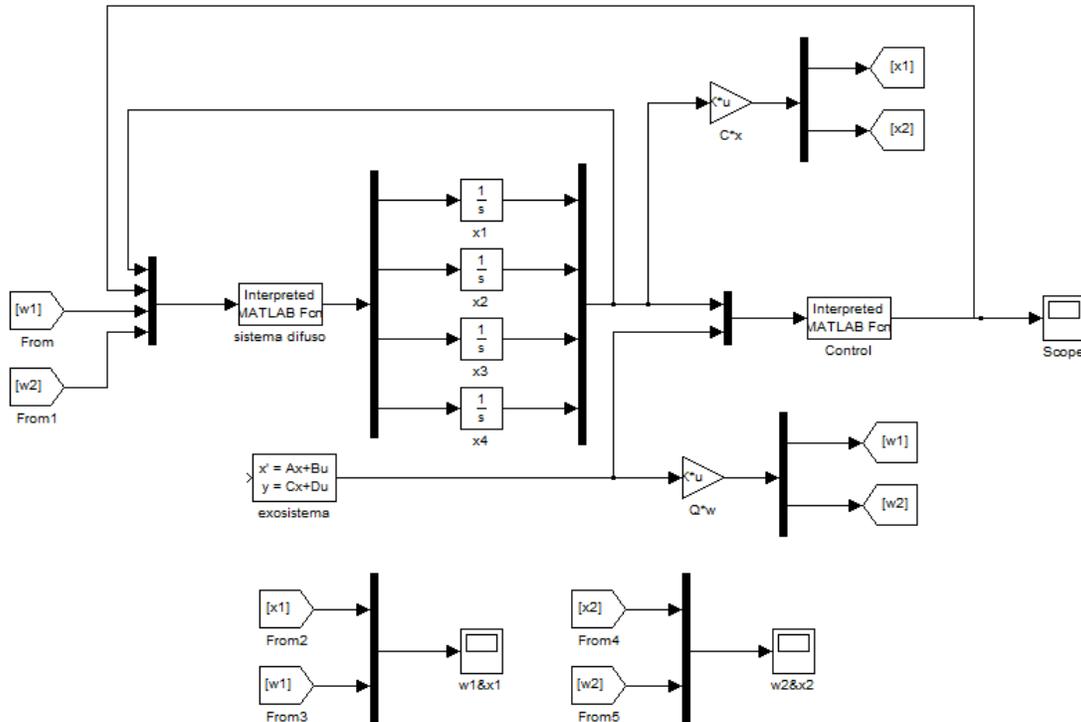


Figura 4.4. Diagrama en Simulink del seguimiento en un robot de 2 G.D.L.

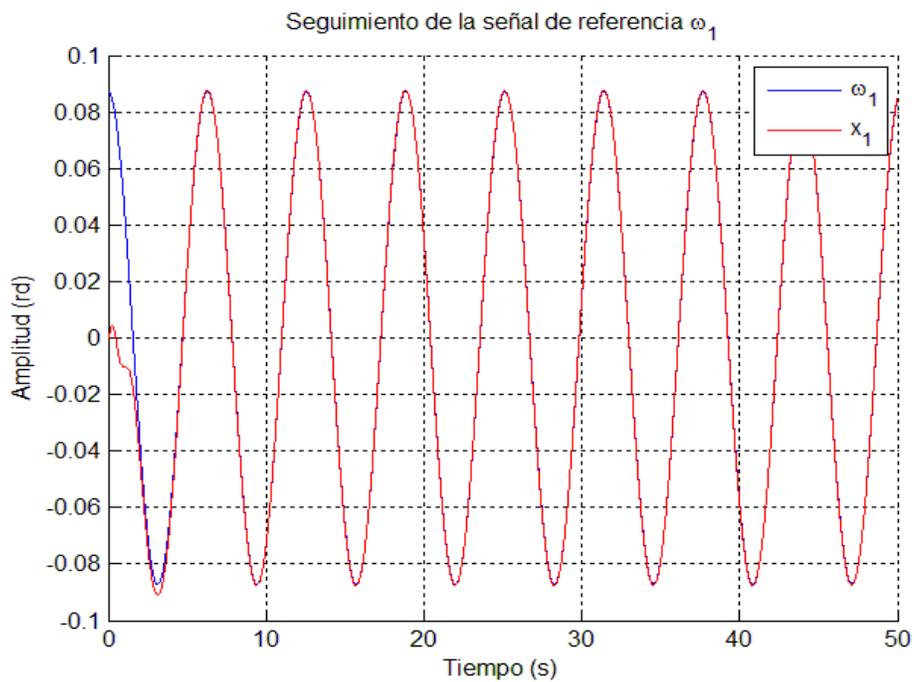


Figura 4.5. Respuesta del seguimiento en la articulación 1.

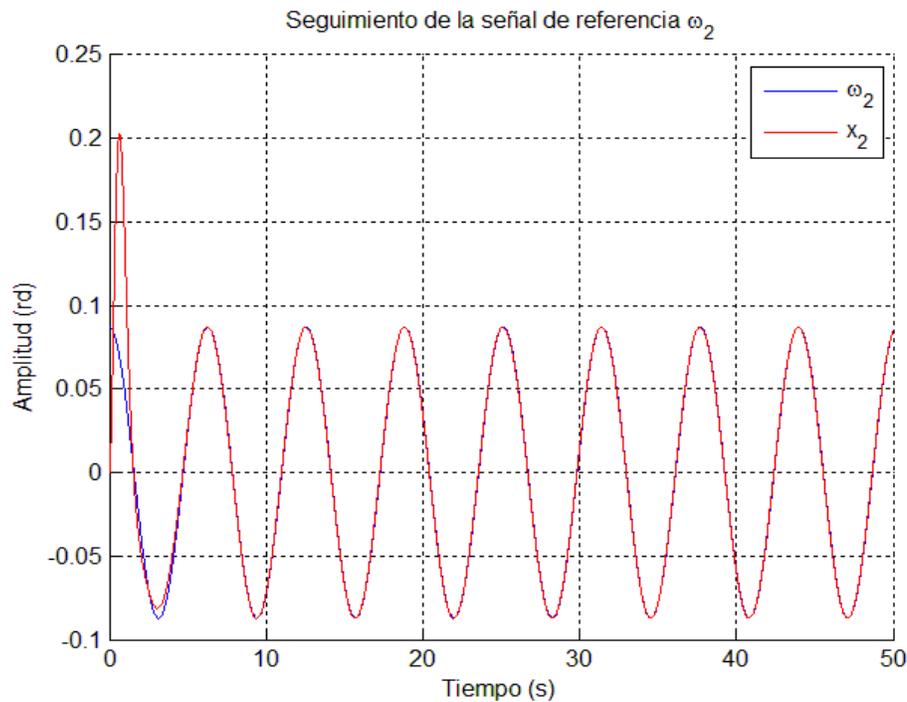


Figura 4.6. Respuesta del seguimiento en la articulación 2.

4.2 Plataforma experimental basada en el robot SCORBOT ER-III

Las plataformas de arquitectura abierta representan una opción viable para la implementación de diferentes algoritmos de control, su importancia radica en que los robots comerciales son vendidos con un controlador cuya arquitectura se limita a las especificaciones de control del fabricante. Por lo tanto en este trabajo se propone una plataforma que tenga la capacidad para implementar diferentes esquemas de control en el robot SCORBOT-ER III. En un principio se resuelve la cinemática directa e inversa de robot, posteriormente se resuelve la dinámica y finalmente se hace una propuesta del controlador.

Denavit y Hartenberg en 1955 propusieron una convención, para relacionar la orientación y posición del efector final de un robot con la base del mismo. También propusieron una matriz de transformación homogénea, que relaciona cada una de las articulaciones del manipulador con su eslabón adyacente inmediato.

$${}^{n-1}T_n = \begin{bmatrix} \cos \theta_n & -\cos \alpha_n \sin \theta_n & \sin \alpha_n & a_n \cos \theta_n \\ \sin \theta_n & \cos \alpha_n \cos \theta_n & -\sin \alpha_n \cos \theta_n & a_n \sin \theta_n \\ 0 & \sin \alpha_n & \cos \alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.5)$$

De acuerdo a la figura 4.7, los parámetros de D-H del robot SCORBOT ER-III son los siguientes:

Tabla 1. Parámetros de D-H del robot SCORBOT ER III.

	θ_i	d_i	a_i	α_i
1	θ_1	0	16 mm	$\pi / 2$
2	θ_2	0	220 mm	0
3	θ_3	0	220 mm	0
4	$\theta_4 - 90$	0	0	$-\pi / 2$
5	θ_5	140 mm	0	0

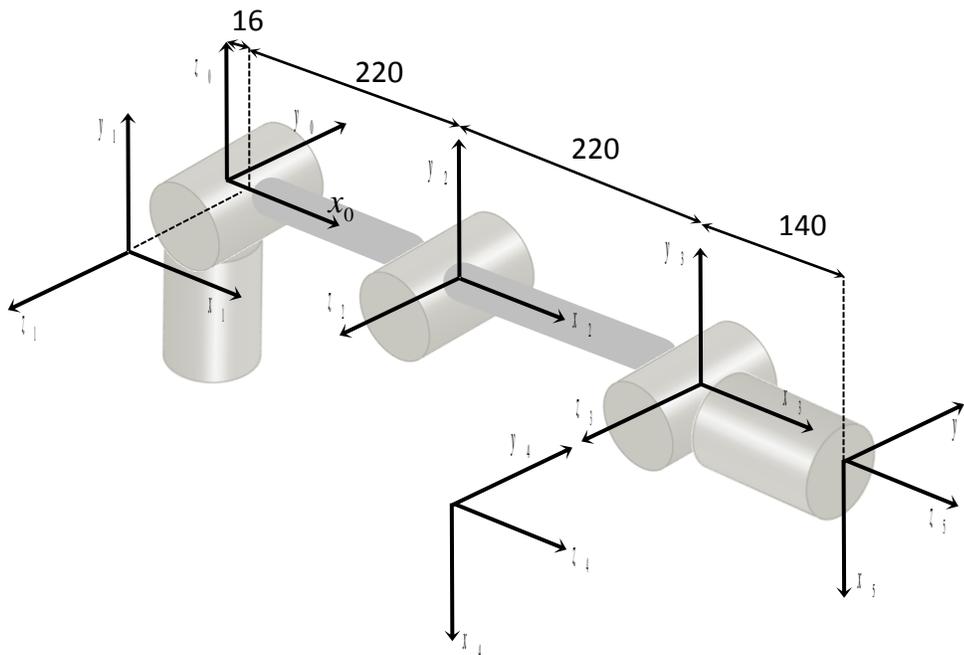


Figura 4.7. Asignación de sistemas de coordenadas de acuerdo a D-H.



Figura 4.8. Robot Scorbot ER III.

Para relacionar el marco de referencia ubicado en el efector final a la base del manipulador, se utiliza la matriz de transformación homogénea que establece la relación del sistema inercial 5 al sistema inercial 0.

$${}^0T_5 = {}^0T_1 {}^1T_2 {}^2T_3 {}^3T_4 {}^4T_5 \quad (4.6)$$

La ecuación (4.7) (matriz de transformación homogénea) proporciona los vectores: normal, de orientación, de aproximación y de posición del efector final con respecto al sistema de referencia base.

$${}^0T_5 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.7)$$

4.3 Solución de la cinemática inversa

Como se mencionó anteriormente se requiere calcular el valor de cada una de las articulaciones dados los vectores de orientación, aproximación, orientación y posición del efector final.

$$\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x(\theta_1, \theta_2 \dots \theta_n) & o_x(\theta_1, \theta_2 \dots \theta_n) & a_x(\theta_1, \theta_2 \dots \theta_n) & p_x(\theta_1, \theta_2 \dots \theta_n) \\ n_y(\theta_1, \theta_2 \dots \theta_n) & o_y(\theta_1, \theta_2 \dots \theta_n) & a_y(\theta_1, \theta_2 \dots \theta_n) & p_y(\theta_1, \theta_2 \dots \theta_n) \\ n_z(\theta_1, \theta_2 \dots \theta_n) & o_z(\theta_1, \theta_2 \dots \theta_n) & a_z(\theta_1, \theta_2 \dots \theta_n) & p_z(\theta_1, \theta_2 \dots \theta_n) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.8)$$

Se establece un sistema de 12 ecuaciones con 5 incógnitas (sistema sobre-determinado no lineal), que numéricamente puede ser resuelto por el método iterativo de Newton-Raphson.

$$\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \end{bmatrix}_C - [J]^+ \begin{bmatrix} F(1) \\ \cdot \\ \cdot \\ \cdot \\ F(12) \end{bmatrix} \quad (4.9)$$

Donde:

$$[J]^+ = \left(\begin{bmatrix} \frac{\partial F(1)}{\partial \theta_1} \dots & \frac{\partial F(1)}{\partial \theta_5} \\ \vdots \\ \frac{\partial F(12)}{\partial \theta_1} \dots & \frac{\partial F(12)}{\partial \theta_5} \end{bmatrix}^T \begin{bmatrix} \frac{\partial F(1)}{\partial \theta_1} \dots & \frac{\partial F(1)}{\partial \theta_5} \\ \vdots \\ \frac{\partial F(12)}{\partial \theta_1} \dots & \frac{\partial F(12)}{\partial \theta_5} \end{bmatrix} \right)^{-1} \begin{bmatrix} \frac{\partial F(1)}{\partial \theta_1} \dots & \frac{\partial F(1)}{\partial \theta_5} \\ \vdots \\ \frac{\partial F(12)}{\partial \theta_1} \dots & \frac{\partial F(12)}{\partial \theta_5} \end{bmatrix}^T$$

4.4 Análisis de Velocidades

La relación entre las velocidades angulares de cada una de las articulaciones, la velocidad lineal y angular del efector final, está determinada por la matriz Jacobiana del manipulador.

La matriz Jacobiana es una representación de la geometría de los elementos de un mecanismo en el tiempo. Esta matriz permite la conversión de movimientos diferenciales o velocidades de cada una de las articulaciones en movimientos diferenciales o velocidades de algún punto en específico. También relaciona los movimientos diferenciales de cada una de las articulaciones con los movimientos totales del mecanismo.

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} {}^0 J_n \end{bmatrix} \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_n \end{bmatrix} \quad (4.10)$$

Para el manipulador SCORBOT la matriz Jacobiana será calculada a partir de la siguiente ecuación:

$${}^0 J_n = \begin{bmatrix} \vec{z}_0 \times (\vec{p}_5 - \vec{p}_0) & \vec{z}_1 \times (\vec{p}_5 - \vec{p}_1) & \vec{z}_2 \times (\vec{p}_5 - \vec{p}_2) & \vec{z}_3 \times (\vec{p}_5 - \vec{p}_3) & \vec{z}_4 \times (\vec{p}_5 - \vec{p}_4) \\ \vec{z}_0 & \vec{z}_1 & \vec{z}_2 & \vec{z}_3 & \vec{z}_4 \end{bmatrix} \quad (4.11)$$

Los vectores \vec{z}_{j-1} , \vec{p}_{j-1} y \vec{p}_n , son extraídos de las matrices de transformación homogénea ${}^0 M_n$, $n = 1, 2, \dots, 5$, y por lo tanto si se proponen las velocidades angulares de cada una de las articulaciones del manipulador, se obtendrán las velocidades angular y lineal del efector final.

4.5 Análisis inverso de velocidades.

En los manipuladores industriales como se ha mencionado anteriormente y por lo general, las velocidades del efector final son parte de las condiciones iniciales de la trayectoria deseada. Por consiguiente se procede a calcular cada una de las velocidades angulares de las articulaciones.

$$\begin{bmatrix} \omega_1 \\ \vdots \\ \omega_n \end{bmatrix} = \begin{bmatrix} {}^0 J_n \end{bmatrix}^{-1} \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \quad (4.12)$$

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \\ \omega_5 \end{bmatrix} = \left[{}^0 J_5 \right]^+ \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \quad (4.13)$$

4.6 Análisis inverso de aceleraciones.

Al igual que en el análisis de velocidades es necesario encontrar las aceleraciones en términos del espacio articulado por lo que la matriz Jacobiana es derivada con respecto al tiempo y despejando el vector de aceleraciones angulares, se obtiene:

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \end{bmatrix} = \left[{}^0 J_5 \right]^+ \begin{bmatrix} a_x \\ a_y \\ a_z \\ \alpha_x \\ \alpha_y \\ \alpha_z \end{bmatrix} - \left[{}^0 J_5 \right]^+ \frac{d}{dt} \left[{}^0 J_5 \right] \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \\ \omega_5 \end{bmatrix} \quad (4.14)$$

4.7 Solución dinámica por el método de Lagrange – Euler para el robot SCORBOT ER III.

Con el método de Euler - Lagrange se obtienen los torques que requieren los motores para recorrer una trayectoria en función de la posición, velocidad y aceleración angular. Básicamente el algoritmo de Euler - Lagrange se basa en el cálculo de la función Lagrangiana que depende de la energía cinética y potencial de un sistema.

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = \tau_i \quad i = 1, \dots, n \quad (4.15)$$

A partir de (4.15) se genera la ecuación diferencial:

$$\tau(t) = D(q)\ddot{q} + h(q, \dot{q}) + c(q) \quad (4.16)$$

Donde:

$D(q)$ = Matriz simétrica de inercia.

$$D_{ik} = \sum_{j=\max(i,k)}^n \text{Tr}(U_{jk} J_j U_{ji}^T) \quad i, k = 1, 2, \dots, n \quad (4.17)$$

Las matrices U_{ij} , U_{ijk} son las derivadas parciales de las matrices de transformación homogénea:

$$U_{ij} = \frac{\partial {}^0T_i}{\partial q_j} = \frac{\partial (A_1 \dots A_j \dots A_i)}{\partial q_j} = A_1 \dots Q_j A_j \dots A_i \quad j \leq i \quad (4.18)$$

$$U_{ijk} = \frac{\partial U_{ij}}{\partial q_k}$$

\ddot{q} = Vector de aceleración de cada una de las articulaciones.

$h(q, \dot{q})$ = Vector de fuerza de Coriolis y centrífuga no lineal.

$$h_i = \sum_{k=1}^n \sum_{m=1}^n h_{ikm} \dot{q}_k \dot{q}_m = \dot{q}^T H_i \dot{q} \quad i = 1, 2, \dots, n \quad (4.19)$$

$$h_{ikm} = \sum_{j=\max(i,k,m)}^n \text{Tr}(U_{jkm} J_j U_{ji}^T) \quad i, k, m = 1, 2, \dots, n \quad (4.20)$$

$c(q)$ = Vector de pares gravitacionales.

$$c_i = \sum_{j=i}^n -m_j g U_{ji}^j \vec{r}_j \quad i = 1, 2, \dots, n \quad (4.21)$$

4.8 Matriz simétrica de inercia

La matriz simétrica de inercia definida para un manipulador con 5 grados de libertad será:

$$D(\theta) = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & D_{15} \\ D_{12} & D_{22} & D_{23} & D_{24} & D_{25} \\ D_{13} & D_{23} & D_{33} & D_{34} & D_{35} \\ D_{14} & D_{24} & D_{34} & D_{44} & D_{45} \\ D_{15} & D_{25} & D_{35} & D_{45} & D_{55} \end{bmatrix}$$

Los tensores de inercia J de cada eslabón se pueden estimar en un software de plataforma CAD:

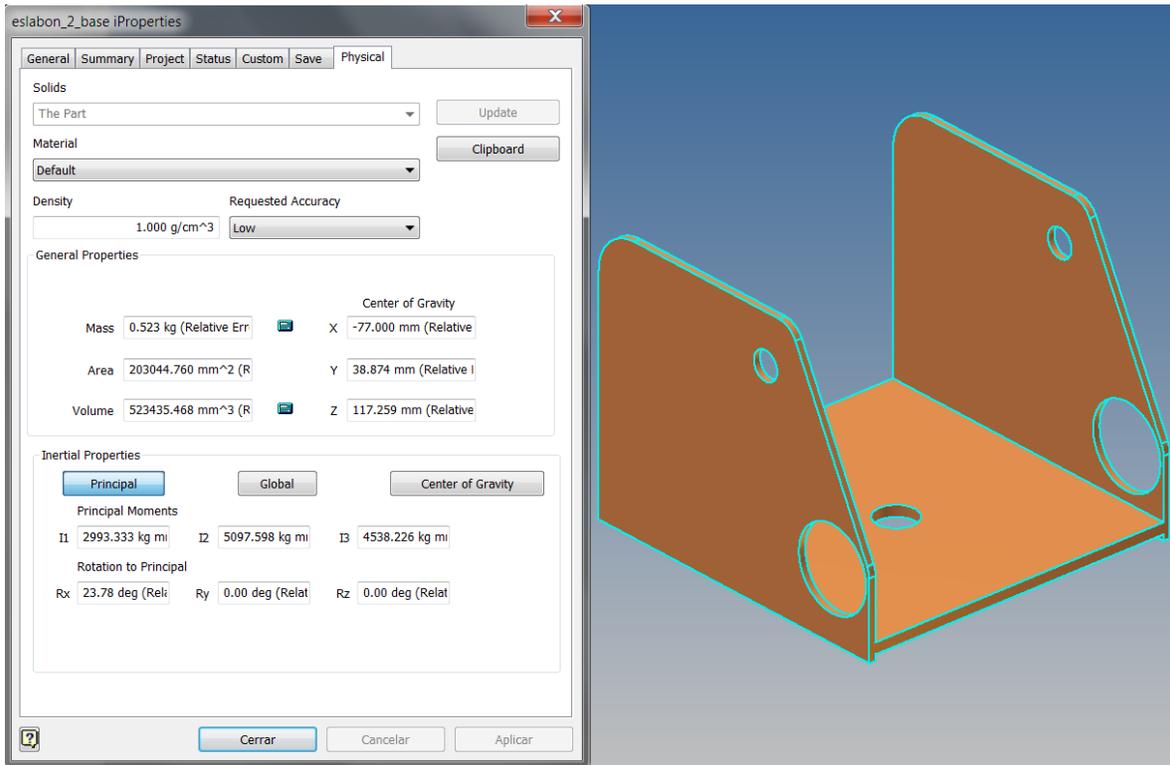


Figura 4.9. Propiedades físicas del eslabón 1.

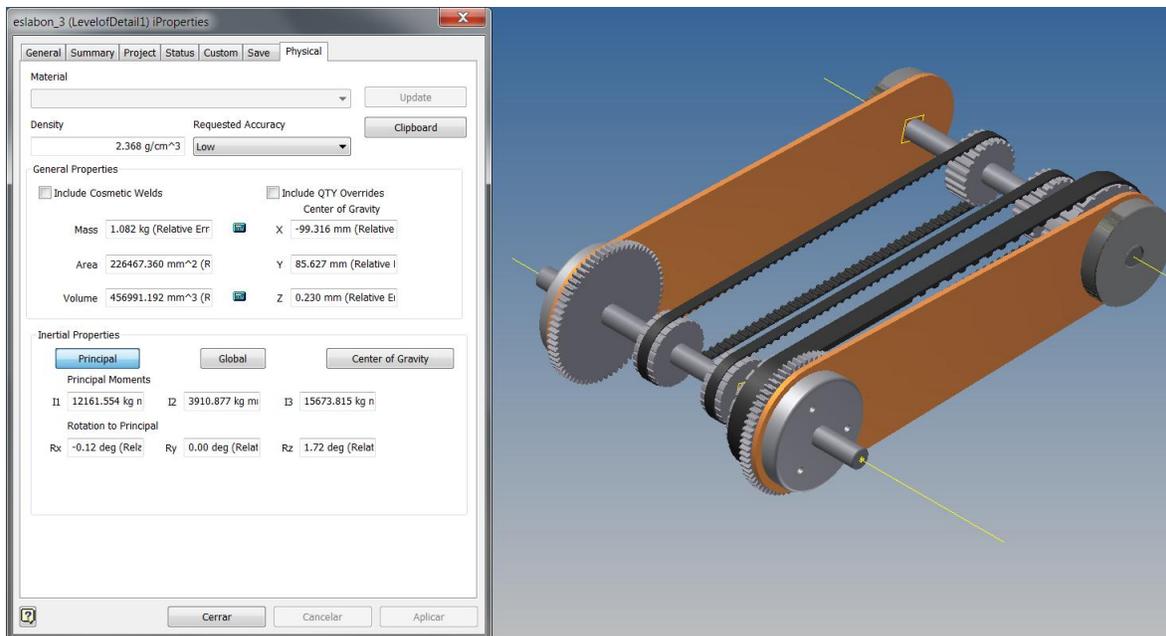


Figura 4.10. Propiedades físicas del eslabón 2.

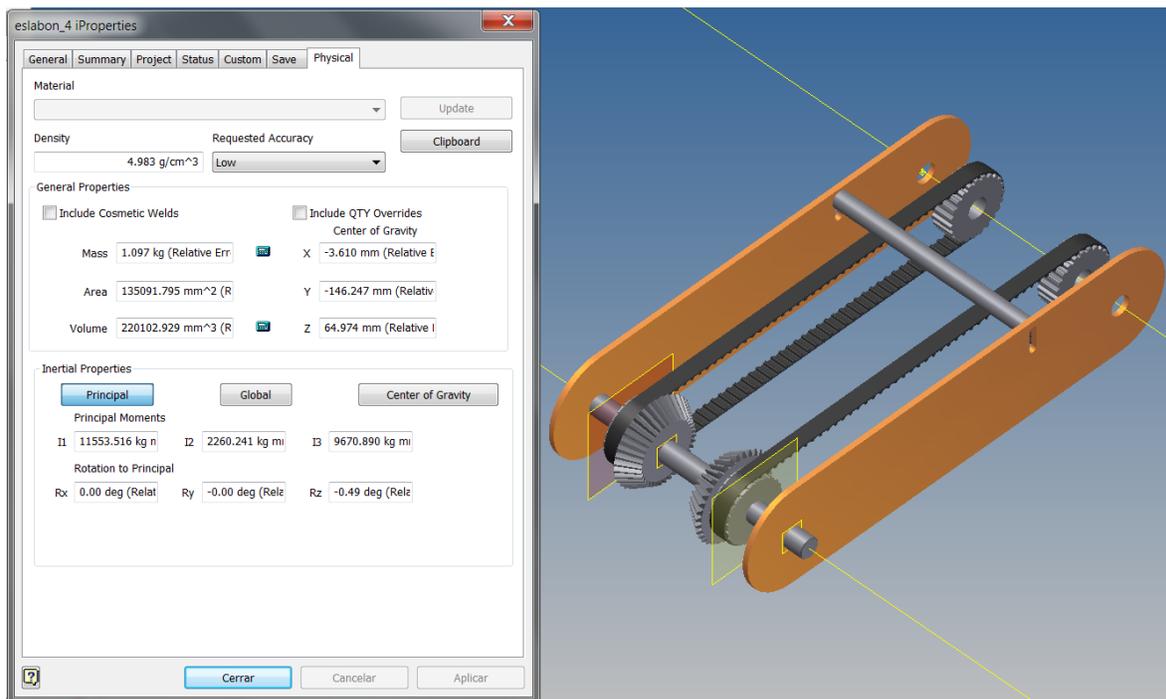


Figura 4.11. Propiedades físicas del eslabón 3.

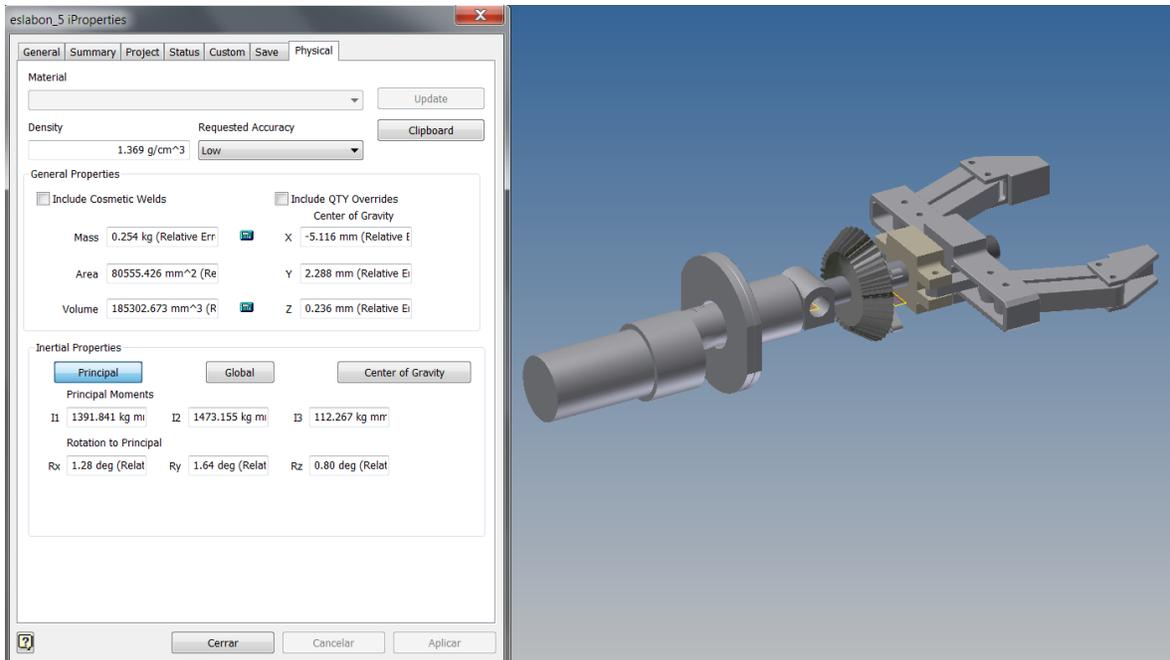


Figura 4.12. Propiedades físicas del efector final.

4.9 Vector de fuerza de Coriolis y centrífuga no lineal $h(q, \dot{q})$.

Sean i y j dos articulaciones unidas por un eslabón, por lo tanto la fuerza de Coriolis será el resultado de multiplicar la aceleración de Coriolis por sus respectivas fuerzas de inercia. La centrífuga es el resultado de la fuerza que actúa en la articulación i debido a la velocidad de j . En el caso de un manipulador de 5 grados de libertad el vector h será de la forma:

$$h = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix}$$

Donde:

$$h_1 = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_5 \end{bmatrix} \begin{bmatrix} h_{111} & h_{121} & h_{131} & h_{141} & h_{151} \\ h_{112} & h_{122} & h_{132} & h_{142} & h_{152} \\ h_{113} & h_{123} & h_{133} & h_{143} & h_{153} \\ h_{114} & h_{124} & h_{134} & h_{144} & h_{154} \\ h_{115} & h_{125} & h_{135} & h_{145} & h_{155} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{bmatrix}$$

$$h_2 = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_5 \end{bmatrix} \begin{bmatrix} h_{211} & h_{221} & h_{231} & h_{241} & h_{251} \\ h_{212} & h_{222} & h_{232} & h_{242} & h_{252} \\ h_{213} & h_{223} & h_{233} & h_{243} & h_{253} \\ h_{214} & h_{224} & h_{234} & h_{244} & h_{254} \\ h_{215} & h_{225} & h_{235} & h_{245} & h_{255} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{bmatrix}$$

$$h_3 = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_5 \end{bmatrix} \begin{bmatrix} h_{311} & h_{321} & h_{331} & h_{341} & h_{351} \\ h_{312} & h_{322} & h_{332} & h_{342} & h_{352} \\ h_{313} & h_{323} & h_{333} & h_{343} & h_{353} \\ h_{314} & h_{324} & h_{334} & h_{344} & h_{354} \\ h_{315} & h_{325} & h_{335} & h_{345} & h_{355} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{bmatrix}$$

$$h_4 = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_5 \end{bmatrix} \begin{bmatrix} h_{411} & h_{421} & h_{431} & h_{441} & h_{451} \\ h_{412} & h_{422} & h_{432} & h_{442} & h_{452} \\ h_{413} & h_{423} & h_{433} & h_{443} & h_{453} \\ h_{414} & h_{424} & h_{434} & h_{444} & h_{454} \\ h_{415} & h_{425} & h_{435} & h_{445} & h_{455} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{bmatrix}$$

$$h_5 = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dot{\theta}_3 & \dot{\theta}_4 & \dot{\theta}_5 \end{bmatrix} \begin{bmatrix} h_{511} & h_{521} & h_{531} & h_{541} & h_{551} \\ h_{512} & h_{522} & h_{532} & h_{542} & h_{552} \\ h_{513} & h_{523} & h_{533} & h_{543} & h_{553} \\ h_{514} & h_{524} & h_{534} & h_{544} & h_{554} \\ h_{515} & h_{525} & h_{535} & h_{545} & h_{555} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \\ \dot{\theta}_5 \end{bmatrix}$$

4.10 Vector de carga gravitacional

El vector de carga o de fuerza gravitacional es aquel que proporciona los torques que actúan en las articulaciones debidos al peso de cada eslabón y para el caso del SCORBOT es:

$$c = - \begin{bmatrix} m_1 g_1 U_{11} r_{11} + m_2 g_2 U_{22} r_{22} + m_3 g_3 U_{33} r_{33} + m_4 g_4 U_{44} r_{44} + m_5 g_5 U_{55} r_{55} \\ m_2 g_2 U_{22} r_{22} + m_3 g_3 U_{33} r_{33} + m_4 g_4 U_{44} r_{44} + m_5 g_5 U_{55} r_{55} \\ m_3 g_3 U_{33} r_{33} + m_4 g_4 U_{44} r_{44} + m_5 g_5 U_{55} r_{55} \\ m_4 g_4 U_{44} r_{44} + m_5 g_5 U_{55} r_{55} \\ m_5 g_5 U_{55} r_{55} \end{bmatrix}$$

4.11 Controlador digital de arquitectura abierta

Los fabricantes de robots diseñan los controladores de los robots de tal forma que sólo pueden ser programados en un lenguaje determinado, y bajo la acción de un esquema de control definido como puede ser un control Proporcional Integral Derivativo (PID). El controlador original del robot SCORBOT ER-III no cumple con los requerimientos necesarios para implementar un algoritmo de control diferente al que proporciona el fabricante. Para la implementar una variedad de esquemas de control, se propone un controlador digital de arquitectura abierta. Este controlador tiene como principal finalidad poder implementar algoritmos de control vía software como LABVIEW®, MATLAB®, VISUAL BASIC®, VISUAL C#®, entre otros lenguajes de programación.

4.12 Sistema de adquisición de datos

El sistema de adquisición de datos está compuesto de 2 tarjetas Phidget 1047 para la lectura a alta velocidad de encoders de cuadratura, estas tarjetas constan de 4 canales de lectura, y como su nombre lo indica son las encargadas de muestrear el desplazamiento angular en cada una de las articulaciones, la tarjeta envía los pulsos generados por cada encoder a través de una conexión USB a una computadora, donde se tiene un registro que sirve como contador para almacenar esta información.

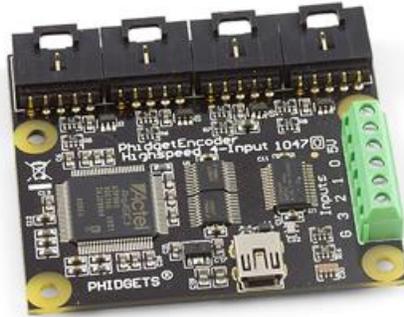


Figura 4.13. Tarjeta de adquisición de datos Phidget 1047.

4.13 Convertidor Digital – Analógico (C. D. A.)

Una vez adquirida la señal de entrada de la planta (desplazamiento angular), se procede a calcular la acción de control cuya naturaleza es del tipo digital, esta señal de control digital deberá convertirse en un valor analógico de corriente o voltaje. Para realizar dicha acción se emplean dos tarjetas Phidget 1002, cuya función es la de un convertidor digital a analógico.



Figura 4.14. Tarjeta de salidas analógicas Phidget 1002.

4.14 Control por modulación de ancho de pulso y etapa de potencia

Debido al bajo voltaje y corriente proporcionados por los convertidores digitales analógicos (tarjetas 1002), es necesario integrar una etapa de potencia que suministre el voltaje y corriente necesarios para el correcto funcionamiento de los motores, se requiere tener control sobre la posición y la velocidad de los motores. En este trabajo se decidió hacer el control de los motores por modulación de ancho de pulso (PWM). El convertidor digital-analógico (D.A.C.) proporciona una salida de voltaje en un rango de 0-5 volts que regulará el ciclo de trabajo del PWM, lo anterior se realiza a través del convertidor analógico-digital (A.D.C.) de un microcontrolador, la señal digital que proviene del A.D.C. se convierte nuevamente

en digital con el propósito de que el microcontrolador con base en la señal adquirida modifique el ciclo de trabajo de la señal PWM. Finalmente tanto las señales PWM como las de inversión de giro del motor, se envían a las terminales de activación de los puentes H (LMD18201T) para controlar su conmutación, la frecuencia aproximada de conmutación es de 20 kHz.

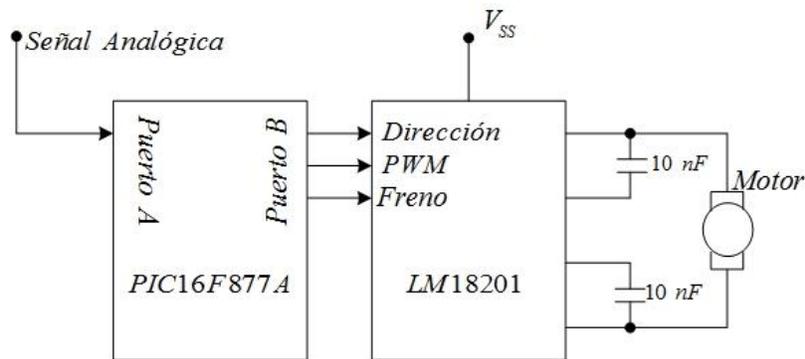


Figura 4.15. Diagrama a bloques del sistema de potencia del robot.

En la figura 4.16 se presenta un diagrama de las conexiones globales del controlador de arquitectura abierta.

4.15 Control vía software

El algoritmo es programado en el ambiente de LabView®, se presenta en primera instancia un programa que consta de una pantalla inicial (Figura 4.17), este programa sirve para probar cada uno de los motores y encoders del robot, en forma manual, permitiendo al usuario verificar la existencia de errores, principalmente debidos a falsos contactos.

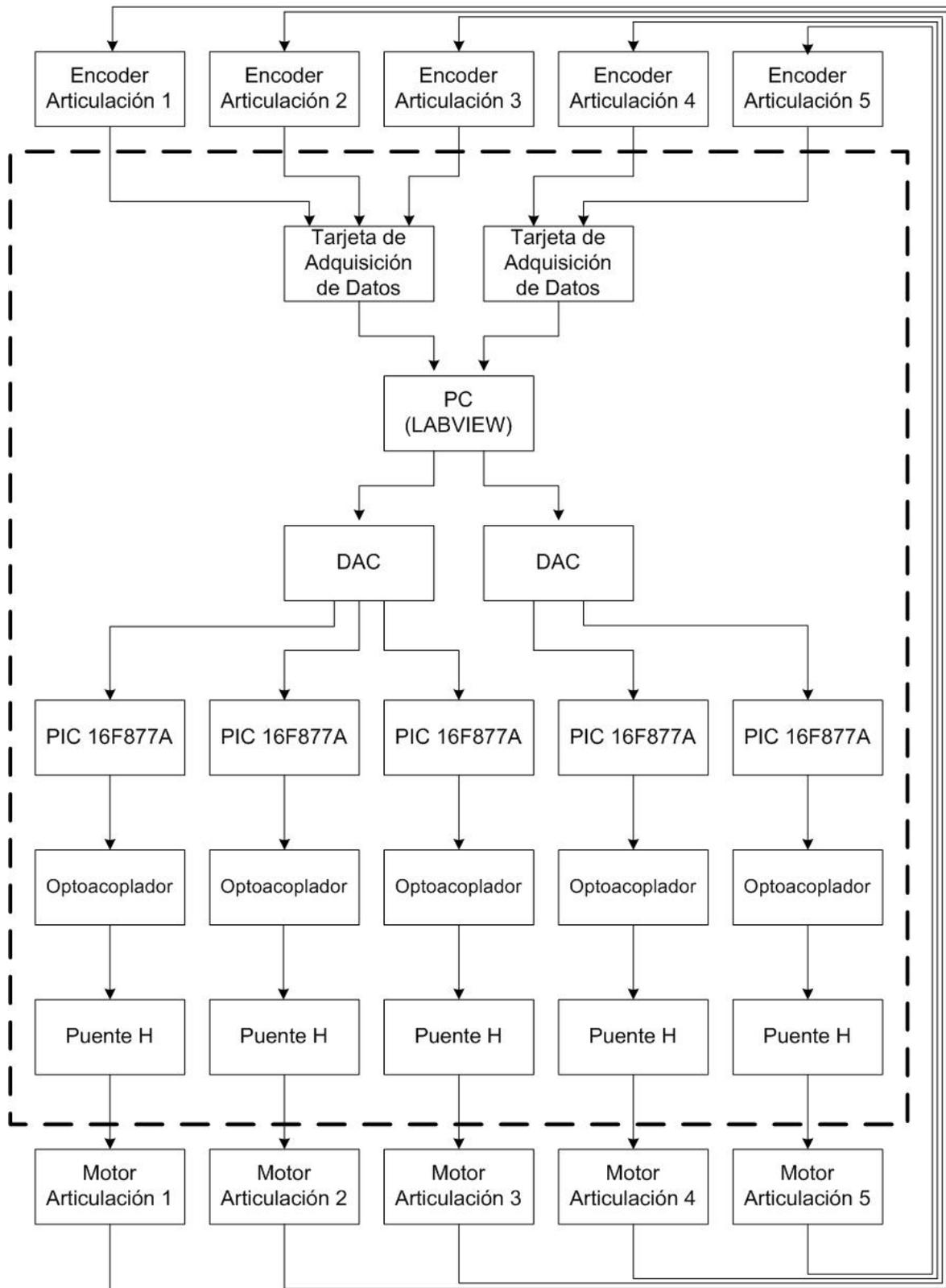


Figura 4.16. Arquitectura del sistema de control de arquitectura abierta.

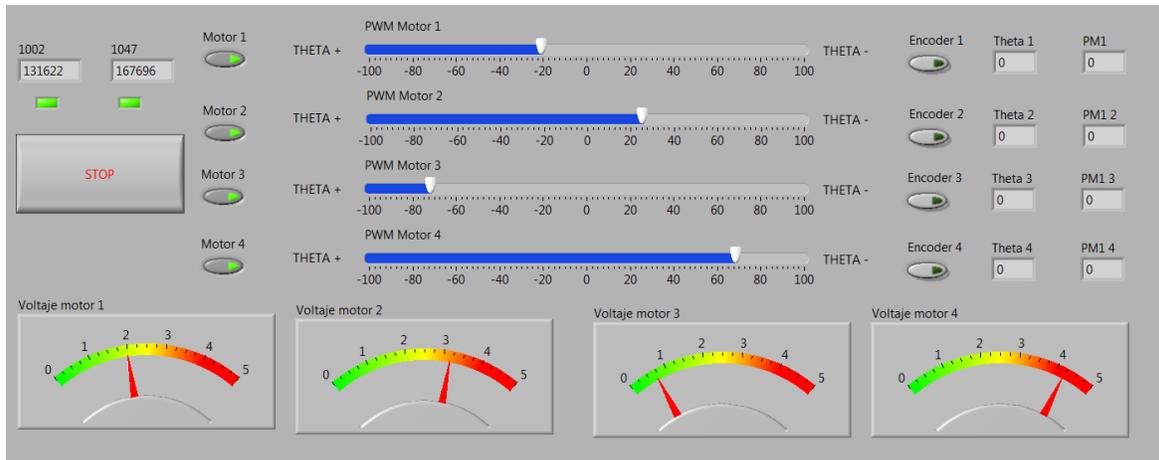


Figura 4.17. Pantalla de prueba de motores del robot SCORBOT ER-III.

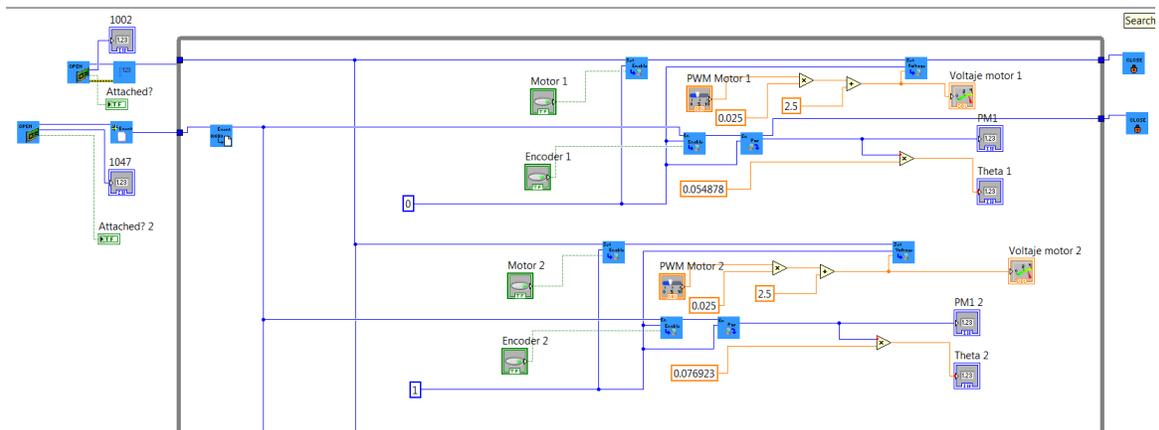


Figura 4.18. Diagrama a bloques del instrumento virtual para la pantalla de prueba del robot SCORBOT ER-III.

El exosistema debe ser expresado en variables de estado, para lo anterior es posible usar la función “Control & Simulation Loop” (Figura 4.19), dentro de esta función se encontrará la función “Steady-State”, en la cual se introducen los valores de las matrices A, B, C, D del sistema en espacio de estados.

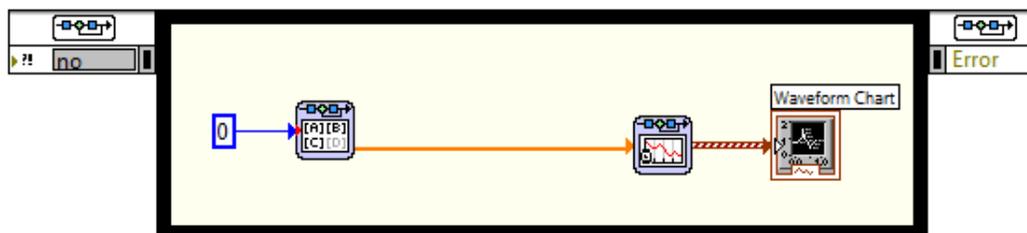


Figura 4.19. Función Control & Simulation Loop.

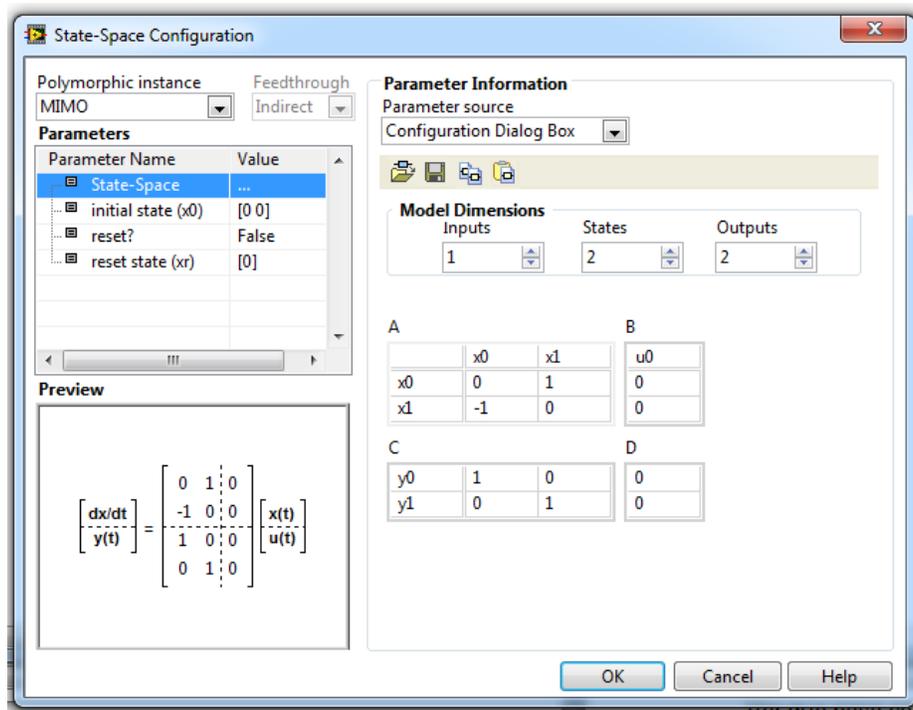


Figura 4.20. Cuadro de configuración de la función Steady-State.

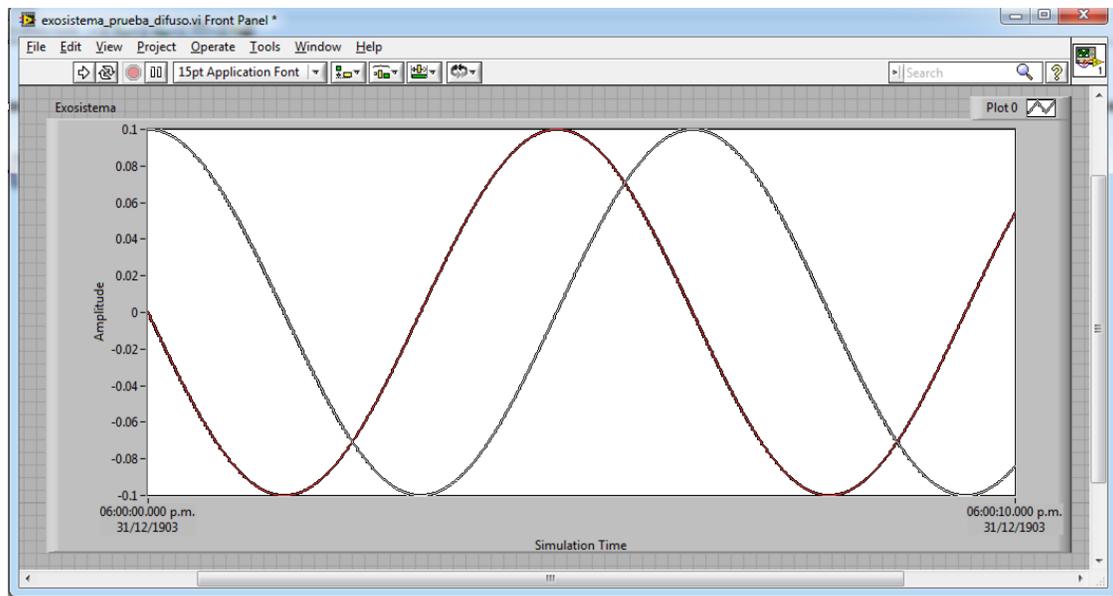


Figura 4.21. Generación de un exosistema de 2 estados en Labview.

Las funciones “array to cluster” y “unbundle” realizan el trabajo de un demultiplexor al igual que la función demux en Simulink®. En la figura 4.22 se muestra como son

introducidas las funciones de membresía triangulares a través de un MATLAB® script.

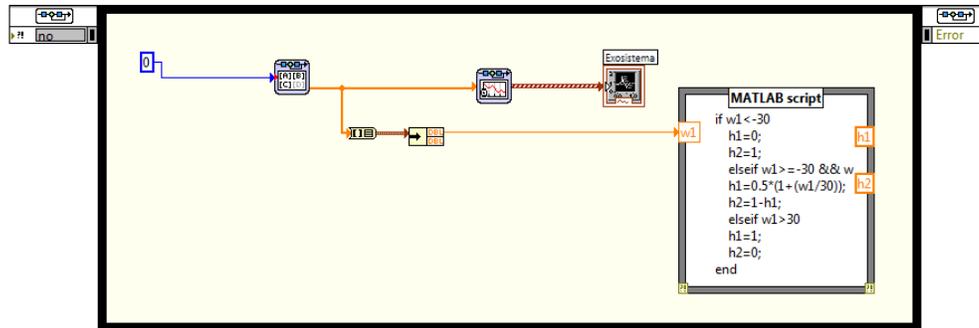


Figura 4.22. Generación de las funciones de membresía a través de un MATLAB script.

La figura 4.23 ilustra el lazo de control basado en las ecuaciones de Francis:

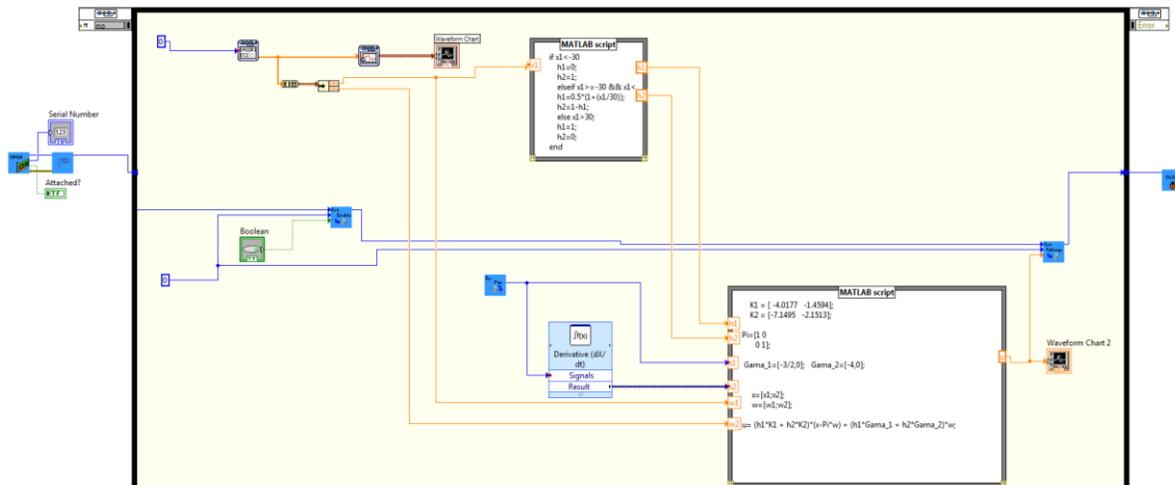


Figura 4.23. Generación de la señal de control para la regulación difusa.

5

Con el objetivo de extender y analizar los alcances y resultados del algoritmo de aprendizaje propuesto, se presenta el exosistema:

$$S = \begin{bmatrix} 0 & \Omega \\ -\Omega & 0 \end{bmatrix}$$

Con el sistema difuso:

$$A_1 = \begin{bmatrix} 0 & 1 \\ 2 & 0 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 1 \\ 3 & 0 \end{bmatrix}, B_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C_1 = C_2 = Q = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

La respuesta para $\Omega = 3$ se muestra de la figura 5.1 a la figura 5.4:

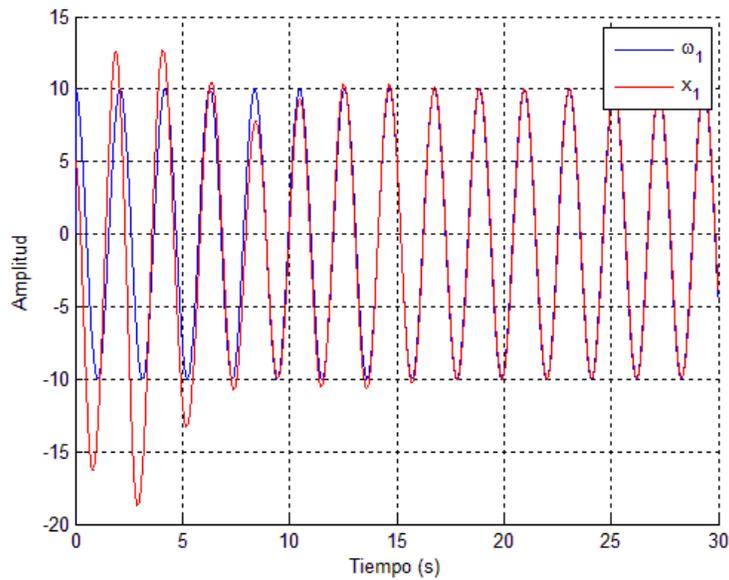


Figura 5.1. Proceso de entrenamiento para $\Omega = 3$.

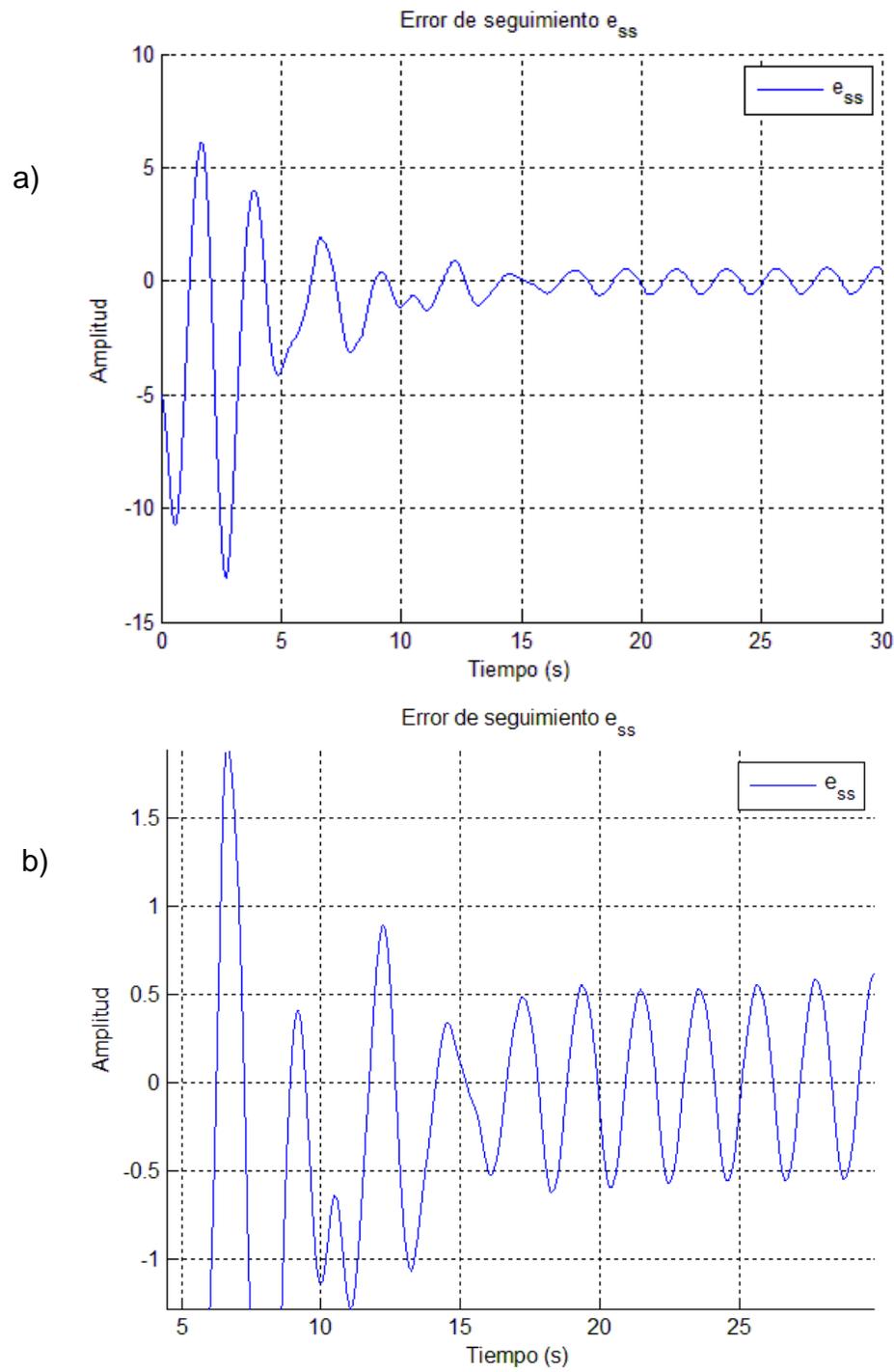


Figura 5.2. a) Error de seguimiento cuando $\Omega = 3$ b) Error de seguimiento posterior al sobretiro

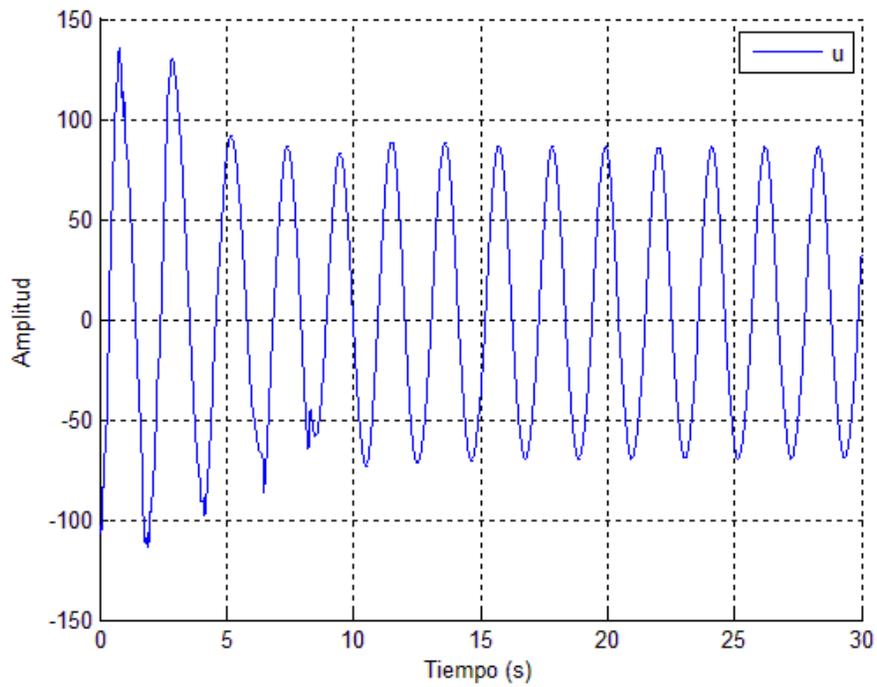


Figura 5.3. Señal de control cuando $\Omega = 3$

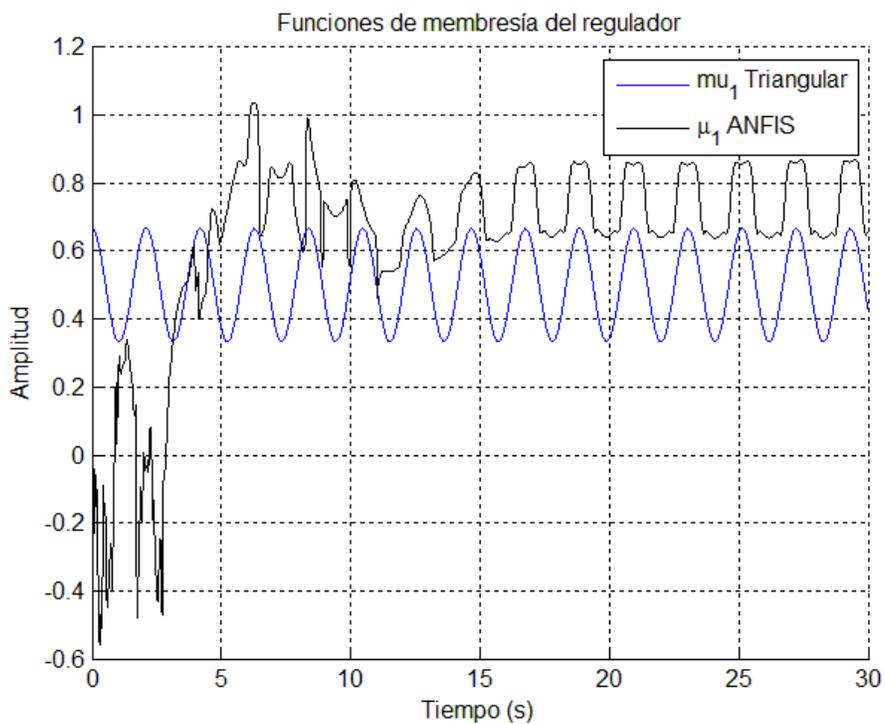


Figura 5.4. Proceso de entrenamiento de μ_1

Como se observa en las figuras anteriores y en especial en la figura 5.4, aunque el ANFIS no está programado para que el valor de salida sea difuso, realiza una buena aproximación entre el rango de 0-1 lo que produce que el estado x_1 siga con una buena precisión al estado ω_1 del exosistema.

Para de extender los resultados de esta tesis, se propone un problema clásico de la teoría de control como lo es el péndulo invertido sobre una plataforma móvil (Figura 5.5). Para este caso se desea que el estado x_1 que hace referencia al ángulo del péndulo con respecto a la vertical, siga al estado ω_1 del exosistema bajo la acción de control u .

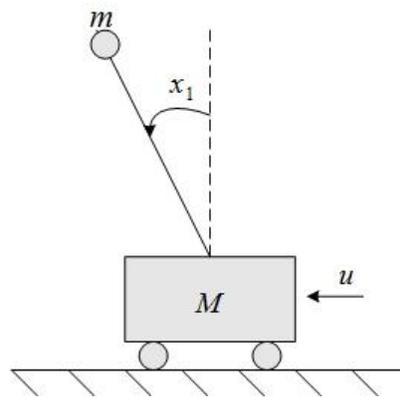


Figura 5.5. Péndulo invertido sobre una plataforma móvil.

En primera instancia se presenta en figura 5.6 y figura 5.7 el comportamiento del estado x_1 y del error $e(t)$ empleando funciones de membresía triangulares en el regulador difuso.

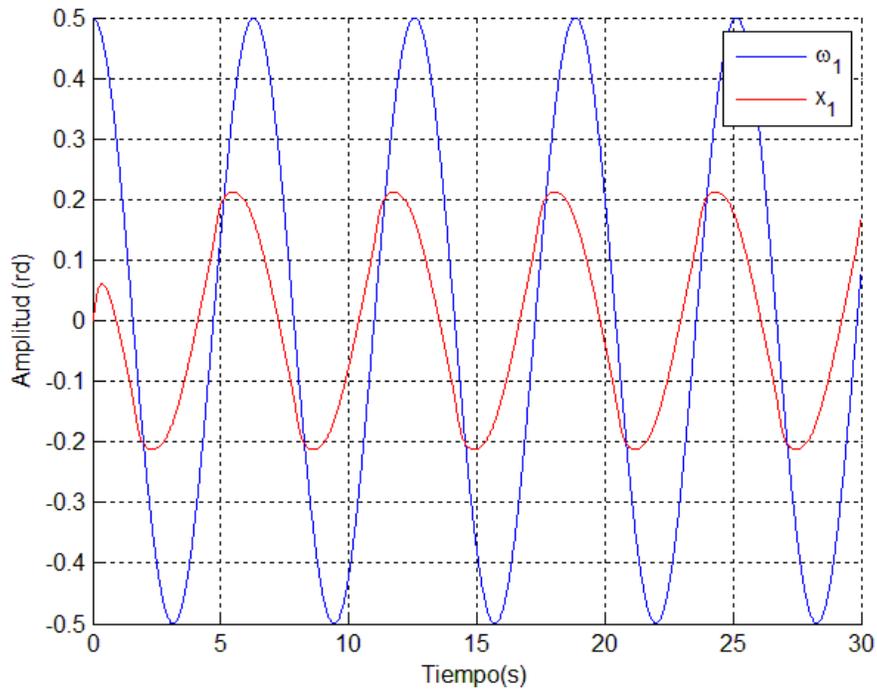


Figura 5.6. Seguimiento del péndulo invertido cuando las funciones del regulador son triangulares

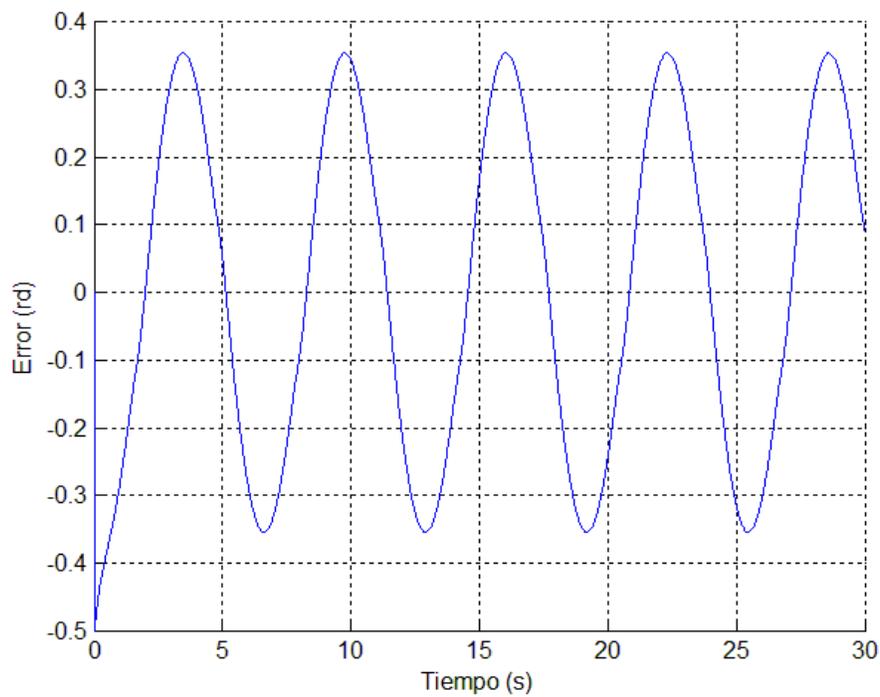


Figura 5.7. Error de seguimiento del péndulo invertido cuando las funciones del regulador son triangulares

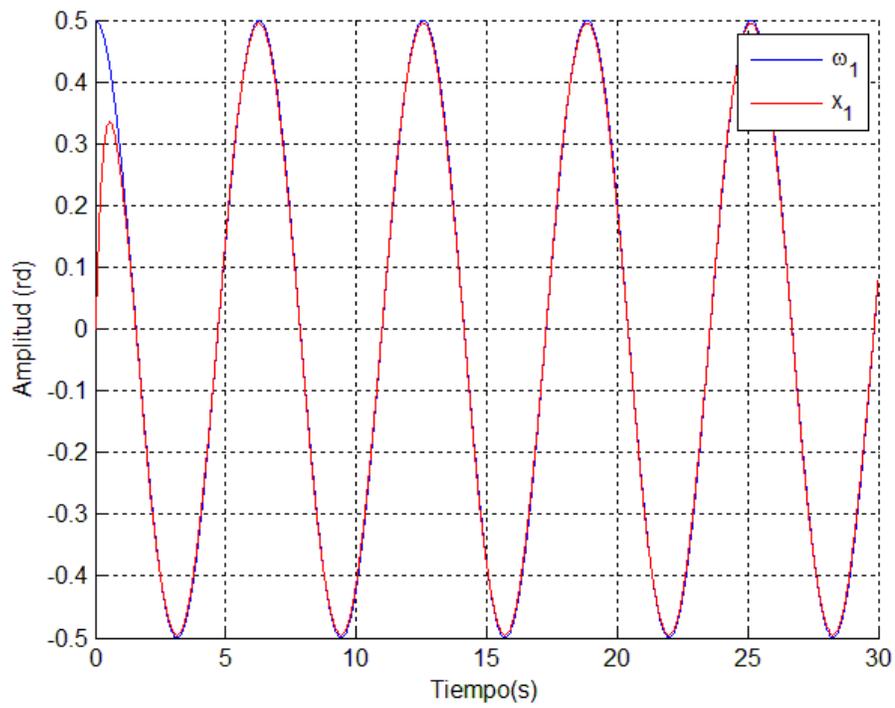


Figura 5.8. Seguimiento del péndulo invertido con el ANFIS entrenado

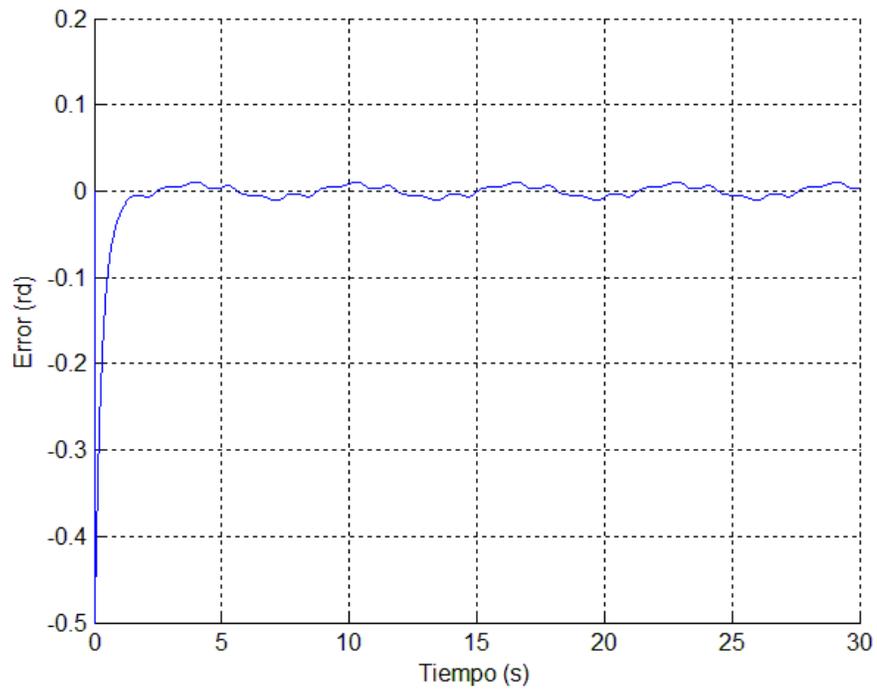


Figura 5.9. Error de seguimiento del péndulo invertido con el ANFIS entrenado

En la figura 5.10 se muestra la evolución de μ_1 durante la fase de entrenamiento, como se observa el ANFIS se aproxima a la solución analítica.

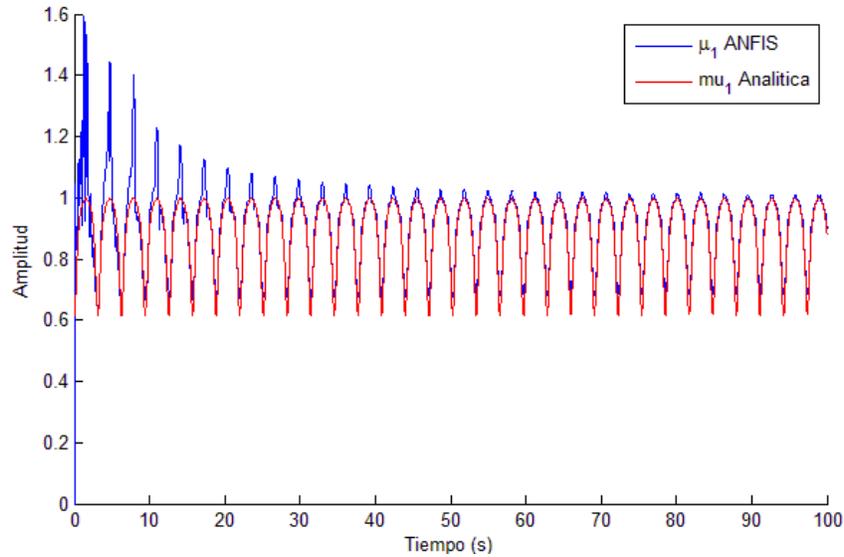


Figura 5.10. Fase de entrenamiento de μ_1

Aplicando el algoritmo desarrollado al seguimiento de referencias en manipuladores robóticos, como se presentó en el capítulo 4, si las matrices B_i son iguales se cumplirá la condición de regulación, sin embargo durante la realización de este trabajo se observó que existen puntos de linealización donde los valores de las matrices B_i son diferentes. Retomando el ejemplo del manipulador de 2 G.D.L. del capítulo anterior, linealizado los puntos $f_1(-20,-25)$, $f_2(-20,45)$, $f_3(30,-25)$, y $f_4(30,45)$, se obtienen matrices B_i diferentes como se muestra a continuación:

$$A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -2.1171 & 11.3836 & 0 & 0 \\ -15.8147 & -47.6694 & 0 & 0 \end{bmatrix} \quad A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -14.5696 & -4.1118 & 0 & 0 \\ 42.4607 & 20.9107 & 0 & 0 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 18.3337 & -1.4031 & 0 & 0 \\ -40.6927 & 5.8755 & 0 & 0 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 5.8904 & -9.3978 & 0 & 0 \\ 16.2890 & 47.7930 & 0 & 0 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 16.0953 & -37.9762 \\ -37.9762 & 129.6035 \end{bmatrix}$$

$$B_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 12.4675 & -25.6913 \\ -25.6913 & 92.9411 \end{bmatrix}$$

$$B_3 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 16.0953 & -37.9762 \\ -37.9762 & 129.6035 \end{bmatrix}$$

$$B_4 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 12.4675 & -25.6913 \\ -25.6913 & 92.9411 \end{bmatrix}$$

En las siguientes se muestra el seguimiento para el manipulador de 2 G.D.L.

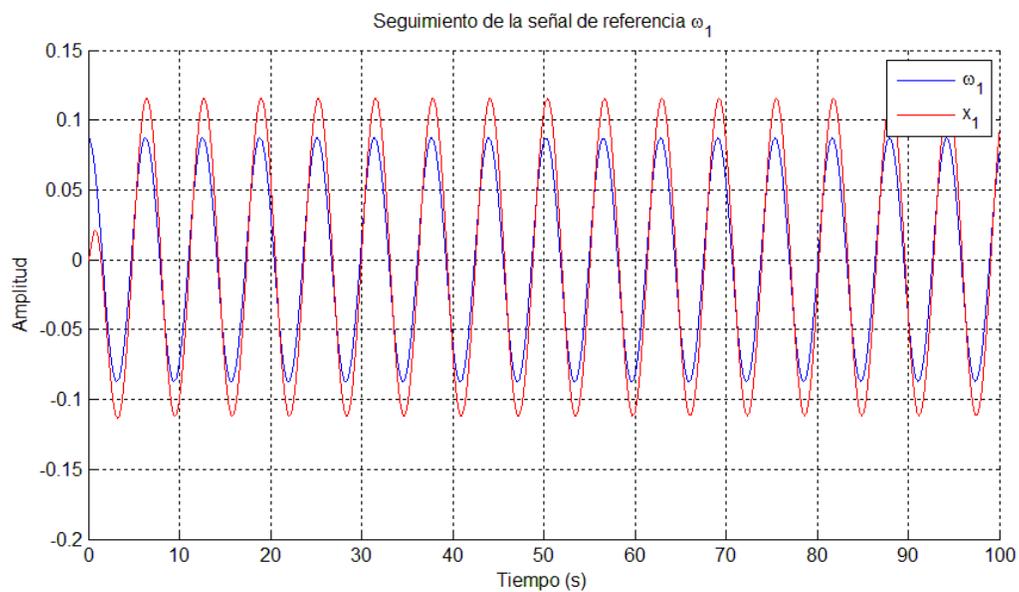


Figura 5.11. Seguimiento de la señal de referencia ω_1 con F.M. lineales.

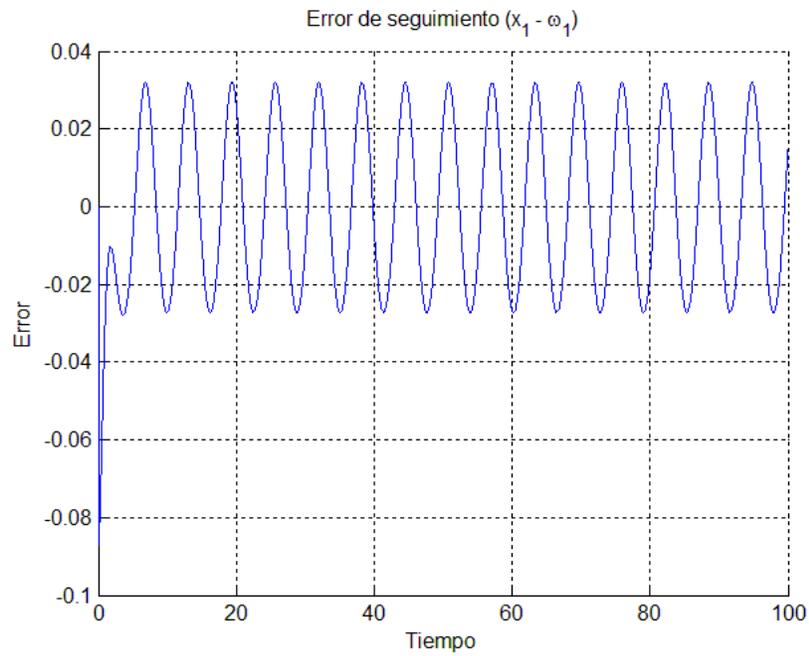


Figura 5.12. Error de seguimiento en la primera articulación.

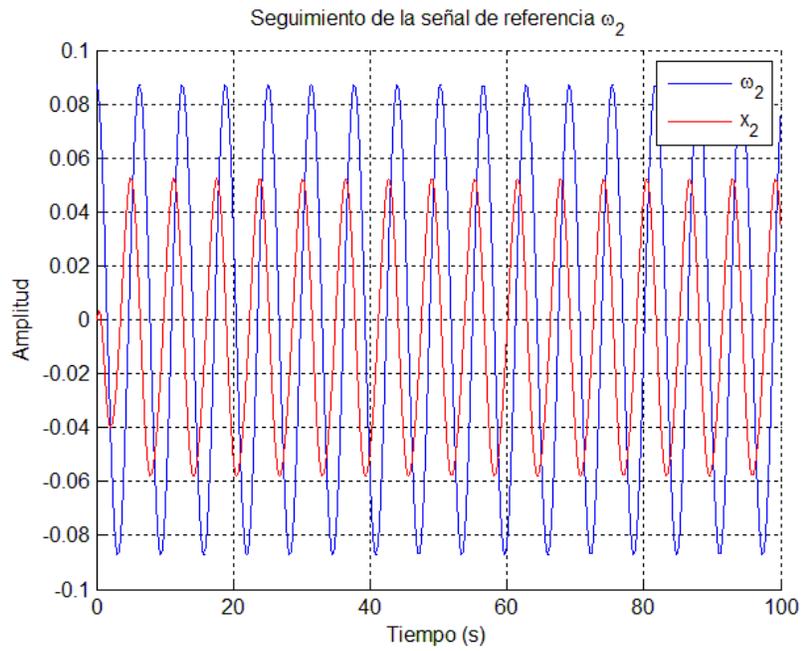


Figura 5.13. Seguimiento en la segunda articulación con F. M. lineales.

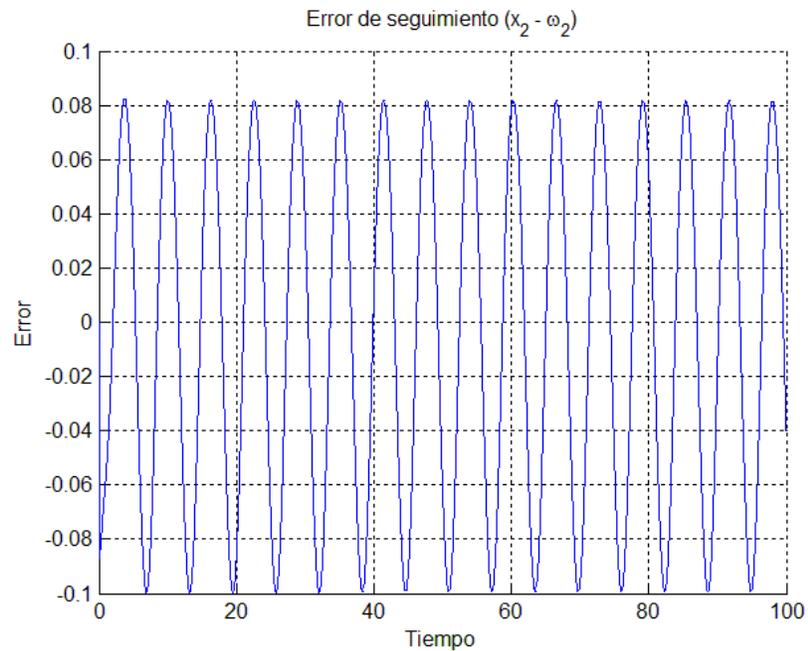


Figura 5.14. Error de seguimiento en la articulación 2.

Retomando la señal de control de modelo difuso del robot, se desacoplan las funciones de membresía del estabilizador y regulador:

$$u = (h_1 K_1 + h_2 K_2 + h_3 K_3 + h_4 K_4)(x - \Pi \omega) + (\mu_1 \Gamma_1 + \mu_2 \Gamma_2 + \mu_3 \Gamma_3 + \mu_4 \Gamma_4) \omega$$

Al igual que en los casos anteriores se desea encontrar las funciones μ_i a partir de las funciones de membresía h_i , por tal motivo se plantean las siguientes premisas:

SI h_1 es 1 ENTONCES $\mu_1 = 1$

SI h_2 es 1 ENTONCES $\mu_2 = 1$

SI h_3 es 1 ENTONCES $\mu_3 = 1$

SI h_4 es 1 ENTONCES $\mu_4 = 1$

La estructura de la red se muestra en la figura 5.15, dónde se aprecia que son 4 sistemas ANFIS, cada uno encargado de aproximar una sola función μ_i .

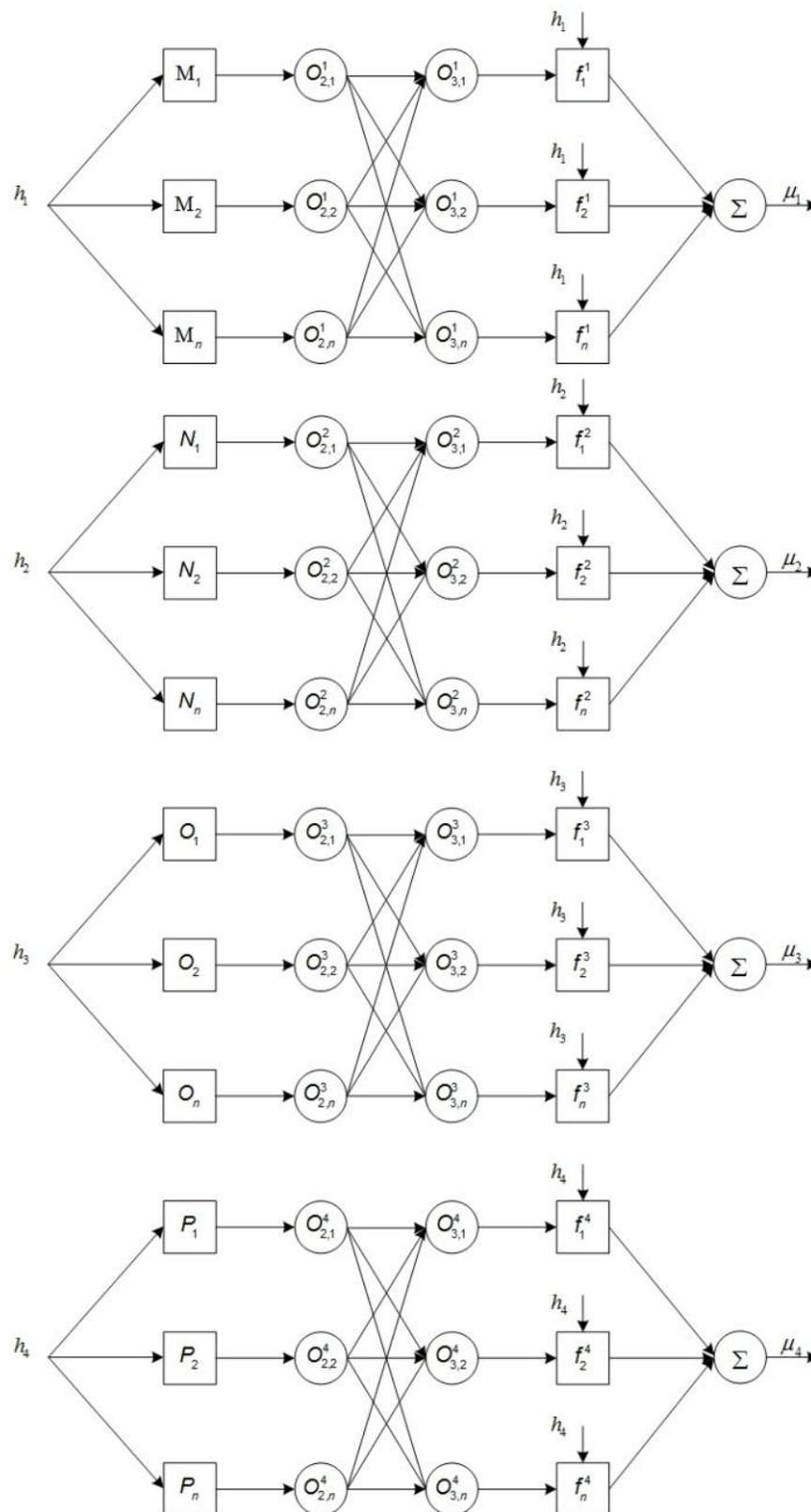


Figura 5.15. Estructura de controlador propuesto con múltiples sistemas ANFIS.

Por lo tanto las reglas para ajustar los parámetros antecedentes serán:

$$a_{ij} = a_{ij} - \alpha \cdot E^T \cdot T \cdot (h_1 \cdot B_1 + h_2 \cdot B_2 + h_3 \cdot B_3 + h_4 \cdot B_4) \cdot \Gamma_i \cdot \omega \cdot (p_{ij} h_i + r_{ij} - \mu_i) \cdot \left(\frac{\partial(GBF)}{\partial a_{ij}} \right)$$

$$b_{ij} = b_{ij} - \alpha \cdot E^T \cdot T \cdot (h_1 \cdot B_1 + h_2 \cdot B_2 + h_3 \cdot B_3 + h_4 \cdot B_4) \cdot \Gamma_i \cdot \omega \cdot (p_{ij} h_i + r_{ij} - \mu_i) \cdot \left(\frac{\partial(GBF)}{\partial b_{ij}} \right)$$

$$c_{ij} = c_{ij} - \alpha \cdot E^T \cdot T \cdot (h_1 \cdot B_1 + h_2 \cdot B_2 + h_3 \cdot B_3 + h_4 \cdot B_4) \cdot \Gamma_i \cdot \omega \cdot (p_{ij} h_i + r_{ij} - \mu_i) \cdot \left(\frac{\partial(GBF)}{\partial c_{ij}} \right)$$

Los resultados de la integración de múltiples ANFIS en el controlador se presentan a continuación:

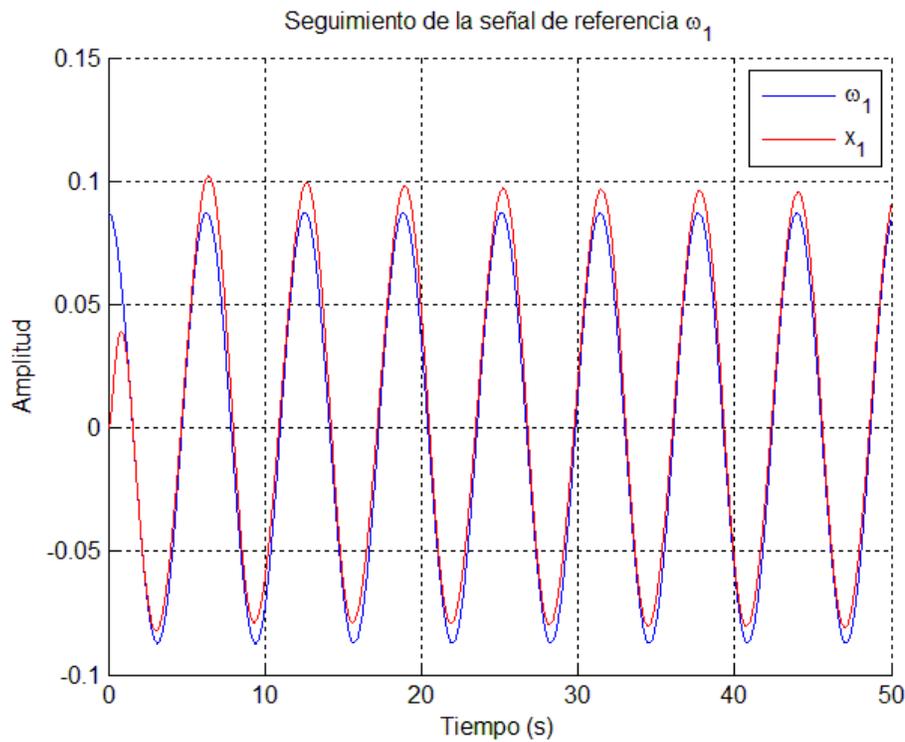


Figura 5.16. Seguimiento en la primera articulación integrando ANFIS.

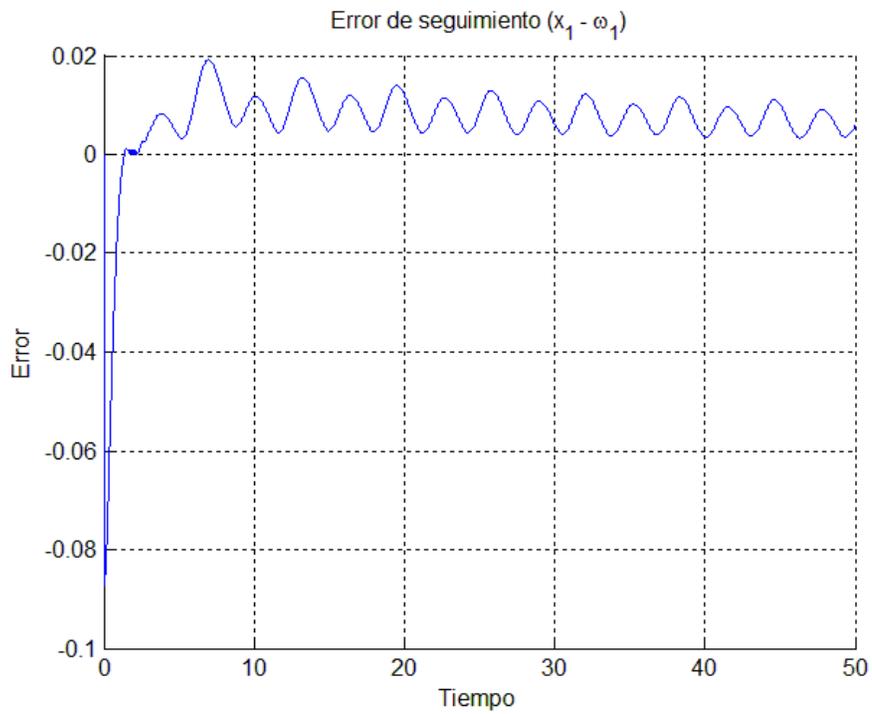


Figura 5.17. Error de seguimiento en la primera articulación con la integración del ANFIS.

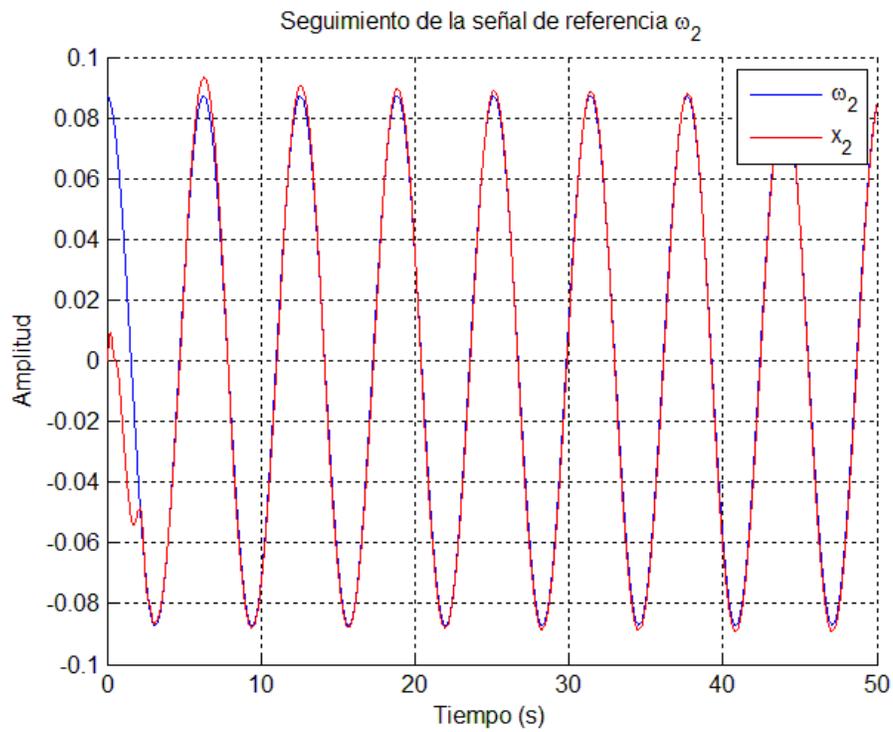


Figura 5.18. Seguimiento en la articulación 2 con la acción ANFIS en el controlador.

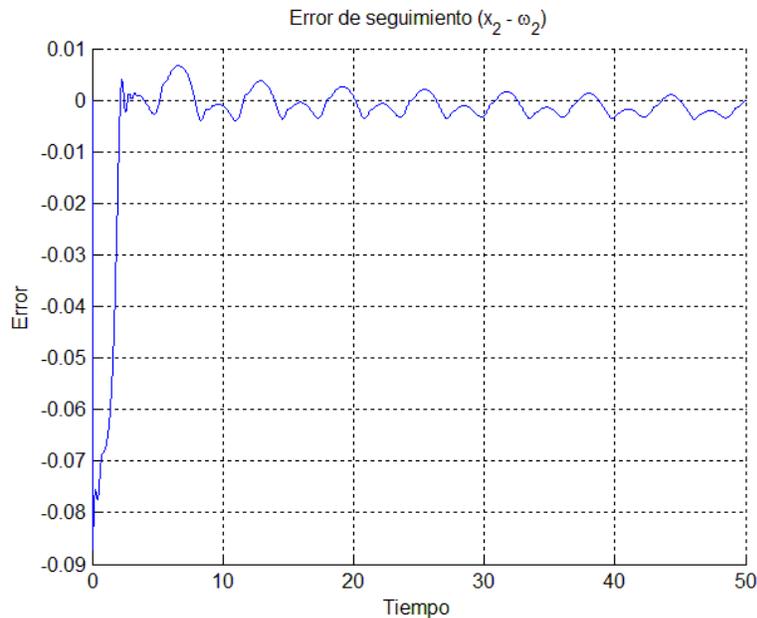


Figura 5.19. Error de seguimiento en la articulación 2.

Como se observó en las figuras anteriores el error disminuye considerablemente con respecto al controlador con funciones triangulares, sin embargo el error no tiende a ser cero conforme transcurre el tiempo, lo anterior se debe a que las funciones de membresía, como se muestran en la figura 5.20, no cumplen con la condición de un conjunto difuso, la función de membresía correspondiente al sistema local 2 tiene valores negativos, esto debido a que el sistema ANFIS no incluye dentro de su estructura que la salida se encuentre exclusivamente en el rango de 0-1, sin embargo para fines prácticos de este trabajo se logra demostrar que el sistema ANFIS puede reducir el error en sistemas no-lineales multivariables.

Otro punto a considerar es que el sistema se trabaja como un sistema de múltiples ANFIS (MANFIS) pero que no tienen dependencia entre ellos, sin embargo es posible que exista una comunicación entre ellos [59].

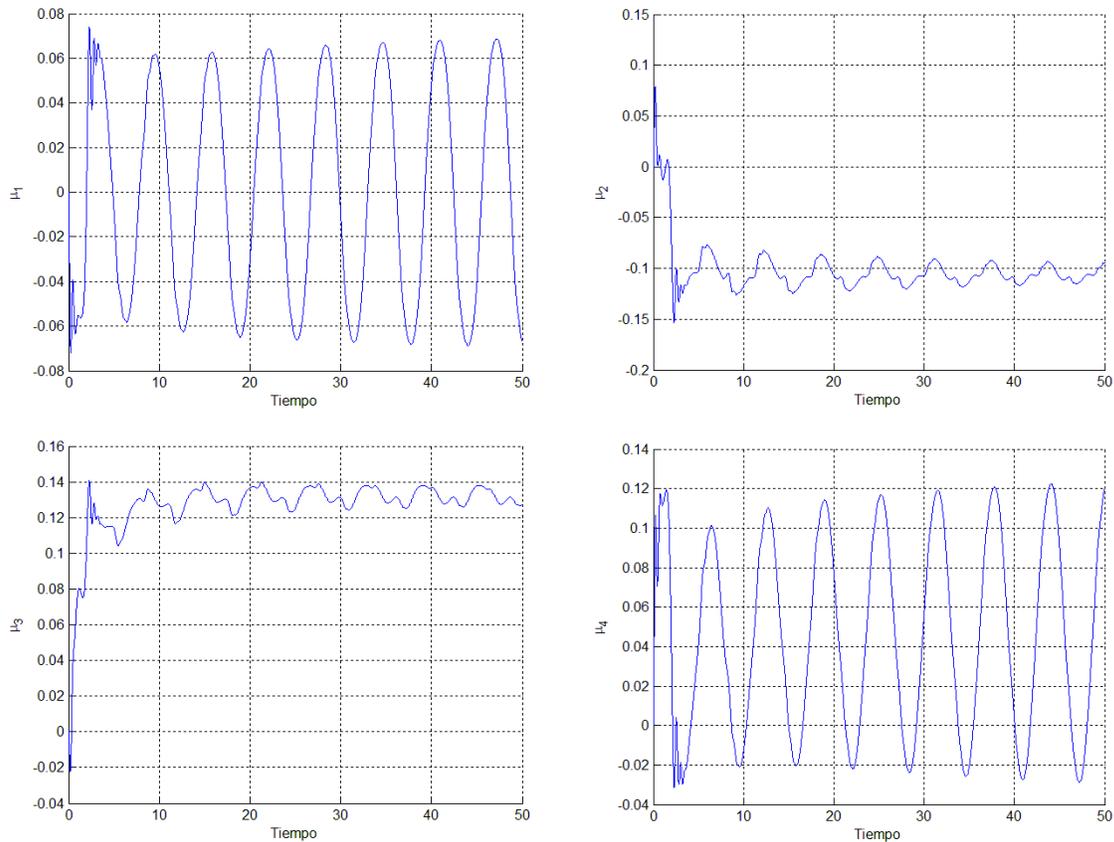


Figura 5.20. Funciones de membresía μ_i calculadas con el ANFIS.

Derivado de lo anterior, la regulación difusa en robots manipuladores podrá presentar B_i tanto iguales como diferentes, dependiendo de los puntos de linealización.

Existe la posibilidad de expresar un exosistema para que un robot de 5 grados como el SCORBOT siga referencias con amplitud diferente en cada articulación, si se emplea la función seno las señales de referencia de este exosistema serán simétricas y su magnitud dependerá de las condiciones iniciales de los estados $\omega_1, \omega_2, \omega_3, \omega_4$ y ω_5 , por lo tanto el exosistema puede escribirse de la siguiente forma:

$$\dot{\omega} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \\ \omega_5 \\ \omega_6 \\ \omega_7 \\ \omega_8 \\ \omega_9 \\ \omega_{10} \end{bmatrix}$$

La generación de trayectorias punto a punto comúnmente se lleva a cabo por interpoladores como polinomios de 5 orden, segmentos lineales con extremos parabólicos o por splines cúbicos, no obstante en este trabajo se plantea la existencia de tareas repetitivas que pueden realizarse por señales sinusoidales desplazadas del origen y con una frecuencia diferente de 1, por lo tanto la señal de referencia será:

$$\omega_1 = \beta + \text{sen}(\Omega t) \rightarrow \dot{\omega}_1 = \Omega \cos(\Omega t) = \Omega \dot{\omega}_2 \rightarrow \dot{\omega}_2 = \Omega(\beta - \omega_1)$$

El tiempo de interpolación t_f será el tiempo que tarda la articulación en recorrer una posición inicial a una posición final, y para este caso en específico representará la mitad del su periodo de la senoide:

$$\Omega = \frac{\pi}{t_f}$$

Y el desplazamiento β cómo:

$$\beta = \omega_1^{final} - \frac{\omega_1^{final} - \omega_1^{inicial}}{2}$$

Como ejemplo se propone que una de las articulaciones vaya de un punto inicial de 0° y un final de 60° en un tiempo de 3 segundos, el resultado se presenta en la

figura 5.21, se observa que cada 3 segundos la articulación irá del punto inicial al final.

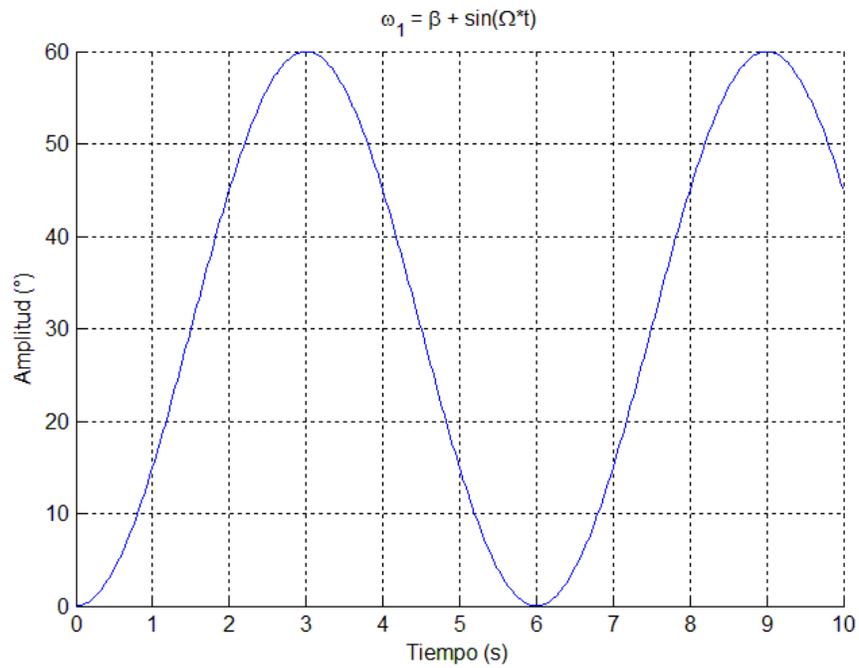


Figura 5.21. Generador sinusoidal $\theta^{inicial} = 0$, $\theta^{final} = 60$, $t = 3seg$

Para el punto inicial -20 y final -45 en 3 segundos:

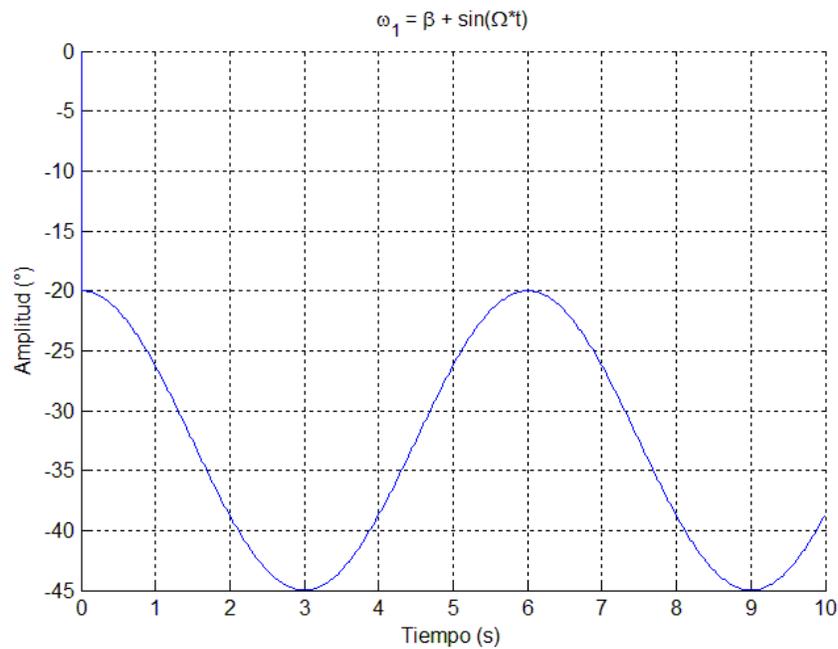


Figura 5.22. Generador sinusoidal $\theta^{inicial} = -20$, $\theta^{final} = -45$, $t = 3seg$

Resultado del desarrollo anterior se propone el siguiente exosistema:

$$\dot{\omega} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \Omega & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Omega & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Omega & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Omega & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Omega \\ -\Omega(1-\beta_1\omega_1^{-1}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\Omega(1-\beta_2\omega_2^{-1}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\Omega(1-\beta_3\omega_3^{-1}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\Omega(1-\beta_4\omega_4^{-1}) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\Omega(1-\beta_5\omega_5^{-1}) & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \\ \omega_5 \\ \omega_6 \\ \omega_7 \\ \omega_8 \\ \omega_9 \\ \omega_{10} \end{bmatrix}$$

Esta última expresión presenta una opción para realizar trayectorias punto a punto con aplicación al manejo y paletizado de materiales, en robots industriales, asegurando las características del problema de regulación.

Conclusiones

Se presentó un enfoque basado en los sistemas neuro-difusos con el objetivo de reducir el error de seguimiento en el problema de regulación difusa, cuando no se cumple la condición de que las matrices B_i de la planta son iguales. El algoritmo propuesto ajusta o sintoniza las funciones de membresía que relacionan a cada uno de los reguladores locales, aprovechando la capacidad de aprendizaje de las redes neuronales. Con frecuencia en el proceso de retro-propagación del error se usa el método del descenso del gradiente, no obstante existe un inconveniente de que este método en su estructura utiliza funciones de membresía con una forma determinada (para este caso campana generalizada), por lo tanto el método sólo funciona en una región de trabajo acotada.

El problema de no tener directamente la salida deseada se resolvió sustituyendo el estimador de mínimos cuadrados por el método del descenso del gradiente o de Levenverg-Marquardt, con lo cual se ajustaron los parámetros consecuentes del sistema ANFIS. Para minimizar la función error se modificaron las reglas de aprendizaje integrando el regulador sintetizado por las ecuaciones de Francis obteniéndose así las direcciones de descenso del error.

Cuando se obtiene el modelo difuso del robot y dependiendo de los puntos de linealización, existe la posibilidad de que las matrices B_i sean iguales o diferentes, para este último integrando el ANFIS se puede aproximar considerablemente la salida de la planta con la señal de referencia; sin embargo no se puede realizar el seguimiento exacto debido a que el ANFIS genera valores de las funciones μ_i que no cumplen la condición de un conjunto difuso.

El controlador de arquitectura abierta propuesto en este trabajo puede realizar la función de adquirir las señales de posición de cada articulación y enviar una señal en voltaje que se puede amplificar para controlar el movimiento de los motores, proporcionando un sistema donde es posible programar diferentes esquemas de control para el robot SCORBOT ER-III.

Trabajos futuros

En los casos presentados aún con la limitación de que el valor buscado es un número difuso, ANFIS proporciona una buena aproximación, sin embargo es conveniente restringir la salida del ANFIS a un conjunto difuso.

Cuando se aplica el ANFIS en el modelo del robot y dependiendo del número de sistemas lineales será el número de sistemas ANFIS a utilizar, por lo tanto se recomienda extender el estudio a los sistemas CANFIS o MANFIS.

De acuerdo a la revisión en la literatura se pueden aplicar otras técnicas de optimización como algoritmos genéticos, con el objetivo de encontrar los valores óptimos correspondientes al mapeo difuso, tanto en el regulador como en el estabilizador.

En este trabajo se abordó el caso del seguimiento en el espacio articulado, ya que en diversos casos el seguimiento se desea en el espacio de cartesiano se propone establecer el modelo dinámico en el espacio de trabajo.

Con la finalidad de probar su eficiencia y capacidad sobre otros esquemas de control se plantea la opción de aplicar el algoritmo desarrollado en el robot SCORBOT ER-III utilizando el controlador digital de arquitectura abierta, con lo cual se pueden realizar tareas de identificación de los parámetros reales de los motores del robot para complementar el modelo del robot con dichos parámetros.

Referencias

1. B.A. Francis y W.M. Wonham, *The Internal Model Principle for Linear Multivariable Regulators*. Applied mathematics and optimization, 1975. 2(2): p.p. 170-194.
2. B.A. Francis, *The Linear Multivariable Regulator Problem*. Journal on Control and Optimization, 1977. 15(3): p.p. 486-505.
3. A. Isidori y C.I. Byrnes, *Output Regulation of Nonlinear Systems*. IEEE Transactions on Automatic Control, 1990. 35(2): p.p. 131-140.
4. J.A. Meda-Campana, J.C. Gomez-Mancilla y B. Castillo-Toledo, *Exact output regulation for nonlinear systems described by Takagi-Sugeno fuzzy models*. IEEE Transactions on Fuzzy Systems, 2012. 20(2): p.p. 235-247.
5. L.A. Zadeh, *Soft Computing and Fuzzy Logic*. IEEE Software, 1994. 11(6): p.p. 48-56.
6. P.P. Bonissone, *Soft computing: the convergence of emerging reasoning technologies*. Soft Computing, 1997. 1(1): p.p. 6-18.
7. J. Henriques, B. Castillo, A. Titli, P. Gil y A. Dourado. *Learning and Output Regulation with Recurrent Neural Networks*, 4th Portuguese Conference on Automatic Control, 2000.
8. J. Henriques, P. Gil, A. Cardoso, P. Carvalho y A. Dourado, *Adaptive neural output regulation control of a solar power plant*. Control Engineering Practice, 2010. 18(10): p.p. 1183-1196.
9. Y. Zhang y J. Wang, *Recurrent neural networks for nonlinear output regulation*. Automatica, 2001. 37(8): p.p. 1161-1173.
10. B. Castillo-Toledo, J.A. Meda-Campaña y A. Titli. *A fuzzy output regulator for Takagi-Sugeno fuzzy models*. IEEE International Symposium on Intelligent Control, 2003. Houston, Texas.

11. X.J. Ma y Z.Q. Sun, *Output Tracking and Regulation of Nonlinear System Based on Takagi–Sugeno Fuzzy Model*. IEEE Transactions on Systems, Man, and Cybernetics, 2000. 30(1): p.p. 47-59.
12. J.A. Meda-Campaña, J.C. Gómez-Mancilla y B. Castillo-Toledo. *On the exact output regulation for Takagi-Sugeno fuzzy systems*, 8th IEEE International Conference on Control and Automation, 2010.
13. Y. Li-Ying y L. Shi-Yong. *Missile Guidance Law Design Using Nonlinear Output Regulation and T-S Model*, Third International Conference on Natural Computation, 2007.
14. N.R. Cazarez-Castro, L.T. Aguilar y O. Castillo. *A Fuzzy-Genetic Controller for the Output Regulation of a Servomechanism with Backlash*. IEEE Workshop on in Hybrid Intelligent Models and Applications, 2009.
15. W.S. McCulloch y W.H. Pitts, *A Logical Calculus of the Ideas Immanent in Nervous Activity*. Bulletin of Mathematical Biophysics, 1943. 5: p.p. 115-133.
16. E. Oyama, N.Y. Chong, A. Agah y T. Maeda. *Inverse Kinematics Learning by Modular Architecture Neural Networks with Performance Prediction Networks*. IEEE International Conference on Robotics and Automation, 2001.
17. R. Köker, C. Öz, T. Çakar y H. Ekiz, *A study of neural network based inverse kinematics solution for a three-joint robot*. Robotics and Autonomous Systems, 2004. 49(3-4): p.p. 227-234.
18. A.T. Hasan, A.M.S. Hamouda, N. Ismail y H.M.A.A. Al-Assadi, *An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 DOF serial robot manipulators*. Advances in Engineering Software, 2006. Vol.37(No.7): p.p. pp.432-438.

19. S. Tejomurtula y S. Kak, *Inverse kinematics in robotics using neural networks*. Information Sciences: An International Journal, 1999. 116(2-4): p.p. 147-164.
20. R. Fullér, *Introduction to neuro-fuzzy systems*. Advances in Intelligent and Soft Computing. Vol. 2. 2000: Springer.
21. P.P. Bonissone, Y.T. Chen, K. Goebel y P.S. Khedkar, *Hybrid Soft Computing Systems: Industrial and Commercial Applications*. Proceedings of the IEEE, 1999. 87(9): p.p. 1641-1667.
22. A. Saad. *An Overview of Hybrid Soft Computing Techniques for Classifier Design and Feature Selection*, 8th International Conference on Hybrid Intelligent Systems, 2008.
23. X. Jian-Xin, G. Zhao-Qin y L. Tong-Heng, *Design and Implementation of a Takagi–Sugeno-Type Fuzzy Logic Controller on a Two-Wheeled Mobile Robot*. IEEE Transactions on Industrial Electronics, 2013. 60(12): p.p. 5717-5728.
24. C.K. Lin y S.D. Wang. *Robust self-tuning rotated fuzzy basis function controller for robot arms*. Control Theory and Applications, 1997.
25. W. Textor, S. Wessel y K.U. Hoffgen. *Learning Fuzzy Rules from Artificial Neural Nets*. Computer Systems and Software Engineering, 1992.
26. T. Miyoshi, S. Tano, Y. Kato y T. Arnould. *Operator Tuning in Fuzzy Production Rules Using Neural Networks*, International Conference on Fuzzy Systems, 1993.
27. K. Nishimori, et al. *Automatic Tuning Method of Membership Functions in Simulation of Driving Control Model Car*. 1993 International Joint Conference on Neural Networks, 1993.
28. W. Li. *Optimization of a Fuzzy Controller using Neural Network*. Third IEEE Conference on Computational Intelligence, 1994.

29. S.K. Halgamuge y M. Glesner, *Neural Networks in Designing Fuzzy Systems for Real World Applications*. Fuzzy Sets and Systems, 1994. 65(1): p.p. 1-12.
30. J. Sitte y S. Geva. *Design of fuzzy controllers with local response neurons*, 1994 IEEE International Conference on Systems, Man, and Cybernetics, 1994. San Antonio TX, USA.
31. D. Van Cleave y K.S. Rattan. *Tuning of Fuzzy Logic Controller using Neural Network*. National Aerospace and Electronics Conference, 2000.
32. D. Nauck y R. Kruse. *A Fuzzy Neural Network Learning Fuzzy Control Rules and Membership Functions by Fuzzy Error Backpropagation*. IEEE International Conference on Neural Networks, 1993. San Francisco.
33. Y. Shi, M. Mizumoto, N. Yubazaki y M. Otani. *A Learning Algorithm for Tuning Fuzzy Rules Based on the Gradient Descent Method*. Fifth IEEE International conference on Fuzzy Systems, 1996.
34. H. Nomura, I. Hayashi y N. Wakami. *A Learning Method of Fuzzy Inference rules by Descent Method*. IEEE International Conference on Fuzzy Systems, 1992.
35. F. Hsuan-Ming. *A Self-Tuning Fuzzy Control System Design*. IFSA World Congress and 20th NAFIPS International Conference, 2001.
36. Y. Shi y M. Mizumoto, *A new approach of neuro-fuzzy learning algorithm for tuning fuzzy rules*. Fuzzy Sets and Systems, 2000. 112: p.p. 99-116.
37. W. Duch, R. Adamczak y K. Grabczewski. *Neural optimization of linguistic variables and membership functions*, International Conference on Neural Information Processing, 1999.
38. G.S. Sandhu y K.S. Rattan. *Design of a Neuro Fuzzy Controller*, 1997 IEEE International Conference On Systems, Man, And Cybernetics 1997. Orlando, FL, USA.

39. L. Yih-Guang, L. Tsu-Tian y W. Wei-Yen, *On-Line Tuning of Fuzzy–Neural Network for Adaptive Control of Nonlinear Dynamical Systems*. IEEE Transactions on Systems, Man, and Cybernetics, 1997. 27(6): p.p. 1034-1043.
40. F.J. Moreno-Velo, I. Baturone, R. Senhadji y S. Sánchez-Solano. *Tuning Complex Fuzzy Systems by Supervised Learning Algorithms*, 12th IEEE International Conference on Fuzzy Systems, 2003.
41. S. Chopra, R. Mitra y V. Kumar, *Neural Network Tuned Fuzzy Controller for MIMO System*. International Journal of Intelligent Technology, 2007. 2(1): p.p. 78-85.
42. R. Jain, N. Sivakumaran y T.K. Radhakrishnan, *Design of self tuning fuzzy controllers for nonlinear systems*. Expert Systems with Applications, 2011. 38(2011): p.p. 4466-4476.
43. J.-S.R. Jang, *ANFIS: Adaptive-Network-Based Fuzzy Inference System*. IEEE Transactions on Systems, Man, and Cybernetics, 1993. 23(3): p.p. 665-685.
44. S. Ravi, M. Sudha y P.A. Balakrishnan, *Design of Intelligent Self-Tuning GA ANFIS Temperature Controller for Plastic Extrusion System*. Modelling and Simulation in Engineering, 2011. 2011: p.p. 1-8.
45. K.H. Hassan, *Self Learning of ANFIS Inverse Control using Iterative Learning Technique*. international Journal of Computer Applications, 2011. 21(8): p.p. 24-29.
46. T. Takagi y M. Sugeno, *Fuzzy Identification of Systems and its Applications to Modeling and Control*. IEEE Transactions on Systems, Man, and Cybernetics, 1985. 15(1): p.p. 116-132.
47. M. Bartholomew-Biggs, *The Steepest Descent Method*, Nonlinear Optimization with Engineering Applications 2008, Springer. p.p. 75-82.

48. M.A. Cauchy, *Méthode générale pour la résolution des systèmes d'équations simultanées*. Comptes Rendus Hebdomadaires des Seances de l'Academie des Sciences, 1847. 25: p.p. 536-538.
49. K. Tanaka y H.O. Wang, *Fuzzy Control Systems Design and Analysis*. 2004: Wiley.
50. E.J. Haug, J.S. Arora y K. Matsui, *A Steepest-Descent Method for Optimization of Mechanical Systems*. Optimization Theory and Applications, 1976. 19(3): p.p. 401-424.
51. X. Wang, *Method of Steepest Descent and its Applications*. IEEE Microwave and Wireless Components Letters, 2008. 12: p.p. 24-26.
52. G. Teschke y C. Borries, *Accelerated Projected Steepest Descent Method for Nonlinear Inverse Problems with Sparsity Constraints*. Inverse Problems, 2009. 26(2): p.p. 1-26.
53. J.-S.R. Jang, C.-T. Sun y E. Mizutani, *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. 1997: Prentice Hall.
54. X.-Z. Wang, T. Zhang y L. He, *Application of fuzzy adaptive back-propagation neural network in thermal conductivity gas analyzer*. Neurocomputing, 2010. 73(4-6): p.p. 679-683.
55. M. Mao y D. Pei, *Fuzzy Adaptive Back Propagation Model Based on Genetic Algorithm*, In Recent Advances in Computer Science and Information Engineering 2012, Springer. p.p. 665-670.
56. M.A. Denai, F. Palis y A. Zeghib. *ANFIS BAsed Modelling and Control of Non-linear Systems: A tutorial*, 2004 IEEE International Conference on Systems, Man and Cybernetics 2004.
57. R. Tapia-Herrera, J.A. Meda-Campaña, S. Alcántara-Montes, T. Hernández-Cortés y L. Salgado-Conrado, *Tuning of a TS Fuzzy Output Regulator Using*

the Steepest Descent Approach and ANFIS. Mathematical Problems in Engineering, 2013: p.p. 1-14.

58. J.-S.R. Jang y E. Mizutani. *Levenberg-Marquardt Method for ANFIS Learning*, Biennial Conference of the North American Fuzzy Information Processing Society, 1996.
59. V. Gomathi, K. Ramar y A.S. Jeeyakumar, *Human Facial Expression Recognition using MANFIS Model*. World Academy of Science, Engineering and Technology, 2009. 50.

Anexo I. Programa en MATLAB del Sistema Neuro-difuso.

```
%% PROGRAMA PARA calcular mu entrenamiento on-line con omega 1 como
%% variable de entrada al anfis
close all
clc; clear all; k=1;

MA1=[0,1;2,0]; MB1=[0;2]; C1=[1,0];
MA2=[0,1;3,0]; MB2=[0;1]; C2=[1,0];
Q=[1,0]; Gama_1=[-3/2,0]; Gama_2=[-4,0];K1 = [ -4.0177  -1.4594];
K2 = [-7.1495  -2.1513];
x_ini=[20;0]; omega_ini=[20;0];
x1(k)=x_ini(1); x2(k)=x_ini(2);
x1e(k)=x_ini(1); x2e(k)=x_ini(2);
time=100;
%% GENERACION DEL EXOSISTEMA
options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4]);
exo = ode45(@exosistema_caso_1,[0,time],omega_ini,options);
T=.02;
%% Parámetros iniciales del ANFIS
gama=1^5;
I=gama*eye(15*2);
alpha=0.2;
lambda=7.9;

theta
=ones(15*2,1);%[0.0288;0.8630;0.0226;0.7356;0.0173;0.6701;0.0135;0.6648;0
.0109;0.6901;0.0096;0.7119];
a1=0.03; b1=2; c1=0;
a2=0.03; b2=2; c2=.071;
a3=0.03; b3=2; c3=.142;
a4=0.03; b4=2; c4=.21;
a5=0.03; b5=2; c5=.28;
a6=0.03; b6=2; c6=.35;
a7=0.03; b7=2; c7=.42;
a8=0.03; b8=2; c8=.49;
a9=0.03; b9=2; c9=.56;
a10=0.03; b10=2; c10=.63;
a11=0.03; b11=2; c11=.70;
a12=0.03; b12=2; c12=.77;
a13=0.03; b13=2; c13=.84;
a14=0.03; b14=2; c14=.91;
a15=0.03; b15=2; c15=1;
F=0.5;

for t=0:T:time %% número de pares de entrenamiento

    omega=deval(exo,t);
    h=MSF_caso_1(omega(1));

    error=[x1(k);x2(k)]-omega;

    if error'*error > (1*10^-6)
```

```

for i=1:5
A1=gbellmf(h(1),[a1 b1 c1]);
A2=gbellmf(h(1),[a2 b2 c2]);
A3=gbellmf(h(1),[a3 b3 c3]);
A4=gbellmf(h(1),[a4 b4 c4]);
A5=gbellmf(h(1),[a5 b5 c5]);
A6=gbellmf(h(1),[a6 b6 c6]);
A7=gbellmf(h(1),[a7 b7 c7]);
A8=gbellmf(h(1),[a8 b8 c8]);
A9=gbellmf(h(1),[a9 b9 c9]);
A10=gbellmf(h(1),[a10 b10 c10]);
A11=gbellmf(h(1),[a11 b11 c11]);
A12=gbellmf(h(1),[a12 b12 c12]);
A13=gbellmf(h(1),[a13 b13 c13]);
A14=gbellmf(h(1),[a14 b14 c14]);
A15=gbellmf(h(1),[a15 b15 c15]);

% SEGUNDA CAPA
w1=A1;w2=A2;w3=A3; w4=A4; w5=A5; w6=A6;w7=A7;w8=A8;w9=A9;
w10=A10;w11=A11;w12=A12;w13=A13;w14=A14;w15=A15;

Swi=w1+w2+w3+w4+w5+w6+w7+w8+w9+w10+w11+w12+w13+w14+w15;
% TERCERA CAPA
Wn1=w1/Swi;Wn2=w2/Swi;Wn3=w3/Swi; Wn4=w4/Swi; Wn5=w5/Swi;
Wn6=w6/Swi;Wn7=w7/Swi;Wn8=w8/Swi; Wn9=w9/Swi; Wn10=w10/Swi;
Wn11=w11/Swi;Wn12=w12/Swi;Wn13=w13/Swi;Wn14=w14/Swi;
Wn15=w15/Swi;

% CUARTA CAPA %matriz de regresión
p1=theta(1); r1=theta(2);
p2=theta(3); r2=theta(4);
p3=theta(5); r3=theta(6);
p4=theta(7); r4=theta(8);
p5=theta(9); r5=theta(10);
p6=theta(11); r6=theta(12);
p7=theta(13); r7=theta(14);
p8=theta(15); r8=theta(16);
p9=theta(17); r9=theta(18);
p10=theta(19); r10=theta(20);
p11=theta(21); r11=theta(22);
p12=theta(23); r12=theta(24);
p13=theta(25); r13=theta(26);
p14=theta(27); r14=theta(28);
p15=theta(29); r15=theta(30);

f1=Wn1*(p1*h(1) + r1);
f2=Wn2*(p2*h(1) + r2);
f3=Wn3*(p3*h(1) + r3);
f4=Wn4*(p4*h(1) + r4);
f5=Wn5*(p5*h(1) + r5);
f6=Wn6*(p6*h(1) + r6);
f7=Wn7*(p7*h(1) + r7);
f8=Wn8*(p8*h(1) + r8);
f9=Wn9*(p9*h(1) + r9);

```

```

f10=Wn10*(p10*h(1) + r10);
f11=Wn11*(p11*h(1) + r11);
f12=Wn12*(p12*h(1) + r12);
f13=Wn13*(p13*h(1) + r13);
f14=Wn14*(p14*h(1) + r14);
f15=Wn15*(p15*h(1) + r15);

% QUINTA CAPA
F=(f1+f2+f3+f4+f5+f6+f7+f8+f9+f10+f11+f12+f13+f14+f15);

est=estabilizador_caso_1(x1(k),x2(k),omega(1),omega(2));
reg=(F*Gama_1 + (1-F)*Gama_2)*omega;
u=est+reg;
x=T*(h(1)*MA1 + h(2)*MA2)*[x1(k);x2(k)] + T*(h(1)*MB1 +
h(2)*MB2)*u + [x1(k);x2(k)];

e_i=x-omega;

if e_i'*e_i<(1*10^-6)
    break
end

cadena=e_i'*T*(h(1)*MB1 + h(2)*MB2)*(Gama_1-Gama_2)*omega;

g_p1=cadena*Wn1*h(1);
g_p2=cadena*Wn2*h(1);
g_p3=cadena*Wn3*h(1);
g_p4=cadena*Wn4*h(1);
g_p5=cadena*Wn5*h(1);
g_p6=cadena*Wn6*h(1);
g_p7=cadena*Wn7*h(1);
g_p8=cadena*Wn8*h(1);
g_p9=cadena*Wn9*h(1);
g_p10=cadena*Wn10*h(1);
g_p11=cadena*Wn11*h(1);
g_p12=cadena*Wn12*h(1);
g_p13=cadena*Wn13*h(1);
g_p14=cadena*Wn14*h(1);
g_p15=cadena*Wn15*h(1);

g_r1=cadena*Wn1;
g_r2=cadena*Wn2;
g_r3=cadena*Wn3;
g_r4=cadena*Wn4;
g_r5=cadena*Wn5;
g_r6=cadena*Wn6;
g_r7=cadena*Wn7;
g_r8=cadena*Wn8;
g_r9=cadena*Wn9;
g_r10=cadena*Wn10;
g_r11=cadena*Wn11;
g_r12=cadena*Wn12;
g_r13=cadena*Wn13;
g_r14=cadena*Wn14;
g_r15=cadena*Wn15;

```

```

g=[g_p1;g_r1;g_p2; g_r2;g_p3;
g_r3;g_p4;g_r4;g_p5;g_r5;g_p6;g_r6;g_p7;g_r7;g_p8;g_r8;g_p9;g_r9;g_p10;g_
r10;g_p11;g_r11;g_p12;g_r12;g_p13;g_r13;g_p14;g_r14;g_p15;g_r15];

```

```

theta=theta - (inv(g'*g + lambda*I))*0.5*g;

```

```

[dA1_a dA1_b dA1_c]=derivada_GB(A1,a1,b1,c1,h(1));
[dA2_a dA2_b dA2_c]=derivada_GB(A2,a2,b2,c2,h(1));
[dA3_a dA3_b dA3_c]=derivada_GB(A3,a3,b3,c3,h(1));
[dA4_a dA4_b dA4_c]=derivada_GB(A4,a4,b4,c4,h(1));
[dA5_a dA5_b dA5_c]=derivada_GB(A5,a5,b5,c5,h(1));
[dA6_a dA6_b dA6_c]=derivada_GB(A6,a6,b6,c6,h(1));
[dA7_a dA7_b dA7_c]=derivada_GB(A7,a7,b7,c7,h(1));
[dA8_a dA8_b dA8_c]=derivada_GB(A8,a8,b8,c8,h(1));
[dA9_a dA9_b dA9_c]=derivada_GB(A9,a9,b9,c9,h(1));
[dA10_a dA10_b dA10_c]=derivada_GB(A10,a10,b10,c10,h(1));
[dA11_a dA11_b dA11_c]=derivada_GB(A11,a11,b11,c11,h(1));
[dA12_a dA12_b dA12_c]=derivada_GB(A12,a12,b12,c12,h(1));
[dA13_a dA13_b dA13_c]=derivada_GB(A13,a13,b13,c13,h(1));
[dA14_a dA14_b dA14_c]=derivada_GB(A14,a14,b14,c14,h(1));
[dA15_a dA15_b dA15_c]=derivada_GB(A15,a15,b15,c15,h(1));

```

⊗

```

a1= a1 - alpha*cadena*((p1*h(1) + r1 -F)/Swi)*dA1_a;
a2= a2 - alpha*cadena*((p2*h(1) + r2 -F)/Swi)*dA2_a;
a3= a3 - alpha*cadena*((p3*h(1) + r3 -F)/Swi)*dA3_a;
a4= a4 - alpha*cadena*((p4*h(1) + r4 -F)/Swi)*dA4_a;
a5= a5 - alpha*cadena*((p5*h(1) + r5 -F)/Swi)*dA5_a;
a6= a6 - alpha*cadena*((p6*h(1) + r6 -F)/Swi)*dA6_a;
a7= a7 - alpha*cadena*((p7*h(1) + r7 -F)/Swi)*dA7_a;
a8= a8 - alpha*cadena*((p8*h(1) + r8 -F)/Swi)*dA8_a;
a9= a9 - alpha*cadena*((p9*h(1) + r9 -F)/Swi)*dA9_a;
a10= a10 - alpha*cadena*((p10*h(1) + r10 -F)/Swi)*dA10_a;
a11= a11 - alpha*cadena*((p11*h(1) + r11 -F)/Swi)*dA11_a;
a12= a12 - alpha*cadena*((p12*h(1) + r12 -F)/Swi)*dA12_a;
a13= a13 - alpha*cadena*((p13*h(1) + r13 -F)/Swi)*dA13_a;
a14= a14 - alpha*cadena*((p14*h(1) + r14 -F)/Swi)*dA14_a;
a15= a15 - alpha*cadena*((p15*h(1) + r15 -F)/Swi)*dA15_a;

```

```

b1= b1 - alpha*cadena*((p1*h(1) + r1 -F)/Swi)*dA1_b;
b2= b2 - alpha*cadena*((p2*h(1) + r2 -F)/Swi)*dA2_b;
b3= b3 - alpha*cadena*((p3*h(1) + r3 -F)/Swi)*dA3_b;
b4= b4 - alpha*cadena*((p4*h(1) + r4 -F)/Swi)*dA4_b;
b5= b5 - alpha*cadena*((p5*h(1) + r5 -F)/Swi)*dA5_b;
b6= b6 - alpha*cadena*((p6*h(1) + r6 -F)/Swi)*dA6_b;
b7= b7 - alpha*cadena*((p7*h(1) + r7 -F)/Swi)*dA7_b;
b8= b8 - alpha*cadena*((p8*h(1) + r8 -F)/Swi)*dA8_b;
b9= b9 - alpha*cadena*((p9*h(1) + r9 -F)/Swi)*dA9_b;
b10= b10 - alpha*cadena*((p10*h(1) + r10 -F)/Swi)*dA10_b;
b11= b11 - alpha*cadena*((p11*h(1) + r11 -F)/Swi)*dA11_b;
b12= b11 - alpha*cadena*((p12*h(1) + r12 -F)/Swi)*dA12_b;
b13= b11 - alpha*cadena*((p13*h(1) + r13 -F)/Swi)*dA13_b;
b14= b11 - alpha*cadena*((p14*h(1) + r14 -F)/Swi)*dA14_b;
b15= b11 - alpha*cadena*((p15*h(1) + r15 -F)/Swi)*dA15_b;

```

```

c1= c1 - alpha*cadena*((p1*h(1) + r1 -F)/Swi)*dA1_b;
c2= c2 - alpha*cadena*((p2*h(1) + r2 -F)/Swi)*dA2_b;
c3= c3 - alpha*cadena*((p3*h(1) + r3 -F)/Swi)*dA3_b;
c4= c4 - alpha*cadena*((p4*h(1) + r4 -F)/Swi)*dA4_b;
c5= c5 - alpha*cadena*((p5*h(1) + r5 -F)/Swi)*dA5_b;
c6= c6 - alpha*cadena*((p6*h(1) + r6 -F)/Swi)*dA6_b;
c7= c7 - alpha*cadena*((p7*h(1) + r7 -F)/Swi)*dA7_b;
c8= c8 - alpha*cadena*((p8*h(1) + r8 -F)/Swi)*dA8_b;
c9= c9 - alpha*cadena*((p9*h(1) + r9 -F)/Swi)*dA9_b;
c10= c10 - alpha*cadena*((p10*h(1) + r10 -F)/Swi)*dA10_b;
c11= c11 - alpha*cadena*((p11*h(1) + r11 -F)/Swi)*dA11_b;
c12= c12 - alpha*cadena*((p12*h(1) + r12 -F)/Swi)*dA12_b;
c13= c13 - alpha*cadena*((p13*h(1) + r13 -F)/Swi)*dA13_b;
c14= c14 - alpha*cadena*((p14*h(1) + r14 -F)/Swi)*dA14_b;
c15= c15 - alpha*cadena*((p15*h(1) + r15 -F)/Swi)*dA15_b;
end
% else
end
est=estabilizador_caso_1(x1(k),x2(k),omega(1),omega(2));
reg=(F*Gama_1 + (1-F)*Gama_2)*omega;
u=est+reg;
x=T*(h(1)*MA1 + h(2)*MA2)*[x1(k);x2(k)] + T*(h(1)*MB1 +
h(2)*MB2)*u + [x1(k);x2(k)];
% end

estabilizer(k)=est;
mu_anfis(k)=F;
h1(k)=h(1);
mu_calc(k)=MSF_analiticas_caso_1(h(1),h(2));
control(k)=est+reg;

k=k+1;
x1(k)=x(1);
x2(k)=x(2);
omegal(k)=omega(1);
omega2(k)=omega(2);
end % fin del ciclo for de tiempo

figure(1)
hold on
grid on
plot(omegal)
plot(x1,'r')
%
figure(2)
hold on
grid on
plot(error)

figure(3)
hold on
plot(mu_anfis)
plot(mu_calc,'r')

figure(4)
plot(control)

```

Anexo II. Sistema Neuro-Difuso Caso 2

```
%% PROGRAMA PARA calcular mu entrenamiento on-line con h1 como
%% variable de entrada al anfis
close all
clc; clear all; k=1;          alpha=0.2;
clf
MA1=[0,1;2,0]; MB1=[0;2]; C1=[1,0];
MA2=[0,1;3,0]; MB2=[0;1]; C2=[1,0];
Q=[1,0]; Gama_1=[-11/2,0]; Gama_2=[-13,0];
x_ini=[5;0]; omega_ini=[10;0];
x1(k)=x_ini(1); x2(k)=x_ini(2);
time=30;

%% GENERACION DEL EXOSISTEMA
options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4]);
exo = ode45(@exosistema_caso_2,[0,time],omega_ini,options);
T=.01;
%% Parámetros iniciales del ANFIS
gama=1^1;
P=gama*eye(11*2); %matriz de covarianza
I=gama*eye(11*2);
lambda=.9;

theta
=zeros(11*2,1);%[0.0288;0.8630;0.0226;0.7356;0.0173;0.6701;0.0135;0.6648;
0.0109;0.6901;0.0096;0.7119];
a1=0.05; b1=2; c1=0;
a2=0.05; b2=2; c2=.1;
a3=0.05; b3=2; c3=.2;
a4=0.05; b4=2; c4=.3;
a5=0.05; b5=2; c5=.4;
a6=0.05; b6=2; c6=.5;
a7=0.05; b7=2; c7=.6;
a8=0.05; b8=2; c8=.7;
a9=0.05; b9=2; c9=.8;
a10=0.05; b10=2; c10=.9;
a11=0.05; b11=2; c11=1;
F=0;

for t=0:T:time %% número de pares de entrenamiento

    omega=deval(exo,t);
    h=MSF_caso_2(omega(1));
    ess=(x1(k)-omega(1));

    % A N F I S
    for i=1:1%% NÚMERO DE épocas
        A1=gbellmf(h(1),[a1 b1 c1]);A2=gbellmf(h(1),[a2 b2 c2]);
        A3=gbellmf(h(1),[a3 b3 c3]);A4=gbellmf(h(1),[a4 b4 c4]);
        A5=gbellmf(h(1),[a5 b5 c5]);A6=gbellmf(h(1),[a6 b6 c6]);
        A7=gbellmf(h(1),[a7 b7 c7]);A8=gbellmf(h(1),[a8 b8 c8]);
        A9=gbellmf(h(1),[a9 b9 c9]);A10=gbellmf(h(1),[a10 b10 c10]);
        A11=gbellmf(h(1),[a11 b11 c11]);
```

```

% SEGUNDA CAPA

w1=A1;w2=A2;w3=A3;w4=A4;w5=A5;w6=A6;w7=A7;w8=A8;w9=A9;w10=A10;w11=A11;

Swi=w1+w2+w3+w4+w5+w6+w7+w8+w9+w10+w11;
% TERCERA CAPA
Wn1=w1/Swi;Wn2=w2/Swi;Wn3=w3/Swi;
Wn4=w4/Swi;Wn5=w5/Swi;Wn6=w6/Swi;Wn7=w7/Swi;
Wn8=w8/Swi;Wn9=w9/Swi;Wn10=w10/Swi;Wn11=w11/Swi;

% CUARTA CAPA %matriz de regresión

a=[Wn1*h(1),Wn1,...
   Wn2*h(1),Wn2,...
   Wn3*h(1),Wn3,...
   Wn4*h(1),Wn4,...
   Wn5*h(1),Wn5,...
   Wn6*h(1),Wn6,...
   Wn7*h(1),Wn7,...
   Wn8*h(1),Wn8,...
   Wn9*h(1),Wn9,...
   Wn10*h(1),Wn10,...
   Wn11*h(1),Wn11];
p1=theta(1);    r1=theta(2);
p2=theta(3);    r2=theta(4);
p3=theta(5);    r3=theta(6);
p4=theta(7);    r4=theta(8);
p5=theta(9);    r5=theta(10);
p6=theta(11);   r6=theta(12);
p7=theta(13);   r7=theta(14);
p8=theta(15);   r8=theta(16);
p9=theta(17);   r9=theta(18);
p10=theta(19);  r10=theta(20);
p11=theta(21);  r11=theta(22);

f1=Wn1*(p1*h(1) + r1);f2=Wn2*(p2*h(1) + r2);
f3=Wn3*(p3*h(1) + r3);f4=Wn4*(p4*h(1) + r4);
f5=Wn5*(p5*h(1) + r5);f6=Wn6*(p6*h(1) + r6);
f7=Wn7*(p7*h(1) + r7);f8=Wn8*(p8*h(1) + r8);
f9=Wn9*(p9*h(1) + r9);f10=Wn10*(p10*h(1) + r10);
f11=Wn11*(p11*h(1) + r11);
% QUINTA CAPA
F=(f1+f2+f3+f4+f5+f6+f7+f8+f9+f10+f11);

f(k)=F;

mul(k)=(MB1(2)*h(1))/(h(1)*MB1(2) + h(2)*MB2(2));
fh(k)=h(1);

est=estabilizador_caso_2(x1(k),x2(k),omega(1),omega(2));
reg=F*Gama_1*omega + (1-F)*Gama_2*omega;
u=est+reg; control(k)=est+reg;
x=T*(h(1)*MA1 + h(2)*MA2)*[x1(k);x2(k)] + T*(h(1)*MB1 +
h(2)*MB2)*u + [x1(k);x2(k)];

```

```

e_a=ess;%(omega(1)- x(1));
error_anfis(k)=abs(e_a);

if error_anfis(k)<=0.01
    break
else

    g_p1=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn1*h(1);
    g_p2=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn2*h(1);
    g_p3=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn3*h(1);
    g_p4=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn4*h(1);
    g_p5=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn5*h(1);
    g_p6=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn6*h(1);
    g_p7=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn7*h(1);
    g_p8=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn8*h(1);
    g_p9=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn9*h(1);
    g_p10=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn10*h(1);
    g_p11=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn11*h(1);

    g_r1=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn1;
    g_r2=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn2;
    g_r3=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn3;
    g_r4=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn4;
    g_r5=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn5;
    g_r6=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn6;
    g_r7=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn7;
    g_r8=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn8;
    g_r9=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn9;
    g_r10=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn10;
    g_r11=-e_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2)
+ T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega*Wn11;

```

```

g=[g_p1;g_r1;g_p2; g_r2;g_p3;
g_r3;g_p4;g_r4;g_p5;g_r5;g_p6;g_r6;g_p7;g_r7;g_p8;g_r8;g_p9;g_r9;g_p10;g_
r10;g_p11;g_r11];

```

```

theta=theta - (inv(g'*g + lambda*I))*0.5*g;

```

```

[dA1_a dA1_b dA1_c]=derivada_GB(A1,a1,b1,c1,h(1));
[dA2_a dA2_b dA2_c]=derivada_GB(A2,a2,b2,c2,h(1));
[dA3_a dA3_b dA3_c]=derivada_GB(A3,a3,b3,c3,h(1));
[dA4_a dA4_b dA4_c]=derivada_GB(A4,a4,b4,c4,h(1));
[dA5_a dA5_b dA5_c]=derivada_GB(A5,a5,b5,c5,h(1));
[dA6_a dA6_b dA6_c]=derivada_GB(A6,a6,b6,c6,h(1));
[dA7_a dA7_b dA7_c]=derivada_GB(A7,a7,b7,c7,h(1));
[dA8_a dA8_b dA8_c]=derivada_GB(A8,a8,b8,c8,h(1));
[dA9_a dA9_b dA9_c]=derivada_GB(A9,a9,b9,c9,h(1));
[dA10_a dA10_b dA10_c]=derivada_GB(A10,a10,b10,c10,h(1));
[dA11_a dA11_b dA11_c]=derivada_GB(A11,a11,b11,c11,h(1));

```

```

%
```

```

a1= a1 - alpha*e_a*((p1*h(1) + r1 -
F)/Swi)*dA1_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a2= a2 - alpha*e_a*((p2*h(1) + r2 -
F)/Swi)*dA2_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a3= a3 - alpha*e_a*((p3*h(1) + r3 -
F)/Swi)*dA3_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a4= a4 - alpha*e_a*((p4*h(1) + r4 -
F)/Swi)*dA4_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a5= a5 - alpha*e_a*((p5*h(1) + r5 -
F)/Swi)*dA5_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a6= a6 - alpha*e_a*((p6*h(1) + r6 -
F)/Swi)*dA6_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a7= a7 - alpha*e_a*((p7*h(1) + r7 -
F)/Swi)*dA7_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a8= a8 - alpha*e_a*((p8*h(1) + r8 -
F)/Swi)*dA8_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a9= a9 - alpha*e_a*((p9*h(1) + r9 -
F)/Swi)*dA9_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a10= a10 - alpha*e_a*((p10*h(1) + r10 -
F)/Swi)*dA10_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
a11= a11 - alpha*e_a*((p11*h(1) + r11 -
F)/Swi)*dA11_a*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;

b1= b1 - alpha*e_a*((p1*h(1) + r1 -
F)/Swi)*dA1_b*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;

```



```

        c9= c9 - alpha*e_a*((p9*h(1) + r9 -
F)/Swi)*dA9_c*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
        c10= c10 - alpha*e_a*((p10*h(1) + r10 -
F)/Swi)*dA10_c*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
        c11= c11 - alpha*e_a*((p11*h(1) + r11 -
F)/Swi)*dA11_c*(T^2*h(1)*h(2) + 2*T^2*h(1)*h(1) + 2*T^2*h(1)*h(2) +
T^2*h(2)*h(2))* (Gama_1-Gama_2)*omega;
    end
    end % fin del ciclo for de entrenamiento
    error(k)=x(1)-omega(1);

    k=k+1;
    x1(k)=x(1);
    x2(k)=x(2);
    omegal(k)=omega(1);
    omega2(k)=omega(2);
    tiempo(k)=t;
end % fin del ciclo for de tiempo

figure(1)
hold on
grid on
plot(tiempo,omegal,tiempo,x1,'r')
xlabel('Tiempo (s)')
ylabel('Amplitud')
legend('\omega_1','x_1')
%
figure(2)
hold on
grid on

plot(0:T:time,error)
title('Error de seguimiento e_s_s')
xlabel('Tiempo (s)')
ylabel('Amplitud')
legend('e_s_s')

figure(3)
hold on
grid on
plot(0:T:time,fh)
plot(0:T:time,f,'black')
title('Funciones de membresía del regulador')
xlabel('Tiempo (s)')
ylabel('Amplitud')
legend('mu_1 Triangular','\mu_1 ANFIS')

figure(4)
hold on
grid on
plot(0:T:time,control)
xlabel('Tiempo (s)')
ylabel('Amplitud')
legend('u')

```

Anexo III. Sistema Neuro-difuso péndulo invertido.

```
%% PROGRAMA PARA calcular mu entrenamiento on-line con omega 1 como
%% variable de entrada al anfis
close all
clc; clear all; k=1;          mfs=15;
MA1 =[                      0                      1
      17.3117647058824      0];

MB1 =[                      0
      -0.176470588235294];

MA2 =[                      0                      1
      9.36957173582662     0];

MB2 =[                      0
      -0.00523588108111208];

Q=[1,0]; Gama_1=[ 103.77, 0]; Gama_2 =[ 1980.5, 0]; K1=-[-120.6667   -
22.6667];
K2=-[-2551.6   -764];

x_ini=[1.5;0];      omega_ini=[1.5;0];

x1(k)=x_ini(1);    x2(k)=x_ini(2);
time=100;
%% GENERACION DEL EXOSISTEMA
options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4]);
exo = ode45(@exosistema_caso_4,[0,time],omega_ini,options);
T=.05;
%% Parámetros iniciales del ANFIS
gama=1^2;
I=gama*eye(15*2);
alpha=0.2;
lambda=.8;

theta
=ones(15*2,1);%[0.0288;0.8630;0.0226;0.7356;0.0173;0.6701;0.0135;0.6648;0
.0109;0.6901;0.0096;0.7119];
a1=0.03;    b1=2;    c1=0;
a2=0.03;    b2=2;    c2=.071;
a3=0.03;    b3=2;    c3=.142;
a4=0.03;    b4=2;    c4=.21;
a5=0.03;    b5=2;    c5=.28;
a6=0.03;    b6=2;    c6=.35;
a7=0.03;    b7=2;    c7=.42;
a8=0.03;    b8=2;    c8=.49;
a9=0.03;    b9=2;    c9=.56;
a10=0.03;   b10=2;   c10=.63;
a11=0.03;   b11=2;   c11=.70;
a12=0.03;   b12=2;   c12=.77;
a13=0.03;   b13=2;   c13=.84;
a14=0.03;   b14=2;   c14=.91;
```

```

a15=0.03;   b15=2;   c15=1;
t=0;

    omega=deval(exo,t);
    h=MSF_caso_4(omega(1));
    F=0;

for t=0:T:time %% número de pares de entrenamiento

    omega=deval(exo,t);
    h=MSF_caso_4(omega(1));

    error=[x1(k);x2(k)]-omega;

if error'*error > (1*10^-6)
    for i=1:1
        A1=gbellmf(h(1),[a1 b1 c1]);
        A2=gbellmf(h(1),[a2 b2 c2]);
        A3=gbellmf(h(1),[a3 b3 c3]);
        A4=gbellmf(h(1),[a4 b4 c4]);
        A5=gbellmf(h(1),[a5 b5 c5]);
        A6=gbellmf(h(1),[a6 b6 c6]);
        A7=gbellmf(h(1),[a7 b7 c7]);
        A8=gbellmf(h(1),[a8 b8 c8]);
        A9=gbellmf(h(1),[a9 b9 c9]);
        A10=gbellmf(h(1),[a10 b10 c10]);
        A11=gbellmf(h(1),[a11 b11 c11]);
        A12=gbellmf(h(1),[a12 b12 c12]);
        A13=gbellmf(h(1),[a13 b13 c13]);
        A14=gbellmf(h(1),[a14 b14 c14]);
        A15=gbellmf(h(1),[a15 b15 c15]);

        % SEGUNDA CAPA
        w1=A1;w2=A2;w3=A3;w4=A4;w5=A5;w6=A6;w7=A7;w8=A8;w9=A9;
        w10=A10;w11=A11;w12=A12;w13=A13;w14=A14;w15=A15;

        Swi=w1+w2+w3+w4+w5+w6+w7+w8+w9+w10+w11+w12+w13+w14+w15;
        % TERCERA CAPA
        Wn1=w1/Swi;Wn2=w2/Swi;Wn3=w3/Swi;Wn4=w4/Swi;Wn5=w5/Swi;
        Wn6=w6/Swi;Wn7=w7/Swi;Wn8=w8/Swi;Wn9=w9/Swi;
        Wn10=w10/Swi;Wn11=w11/Swi;Wn12=w12/Swi;Wn13=w13/Swi;
        Wn14=w14/Swi; Wn15=w15/Swi;

        % CUARTA CAPA %matriz de regresión

        p1=theta(1);   r1=theta(2);
        p2=theta(3);   r2=theta(4);
        p3=theta(5);   r3=theta(6);
        p4=theta(7);   r4=theta(8);
        p5=theta(9);   r5=theta(10);
        p6=theta(11);  r6=theta(12);
        p7=theta(13);  r7=theta(14);
        p8=theta(15);  r8=theta(16);
        p9=theta(17);  r9=theta(18);
        p10=theta(19); r10=theta(20);

```

```

p11=theta(21);    r11=theta(22);
p12=theta(23);    r12=theta(24);
p13=theta(25);    r13=theta(26);
p14=theta(27);    r14=theta(28);
p15=theta(29);    r15=theta(30);

f1=Wn1*(p1*h(1) + r1);
f2=Wn2*(p2*h(1) + r2);
f3=Wn3*(p3*h(1) + r3);
f4=Wn4*(p4*h(1) + r4);
f5=Wn5*(p5*h(1) + r5);
f6=Wn6*(p6*h(1) + r6);
f7=Wn7*(p7*h(1) + r7);
f8=Wn8*(p8*h(1) + r8);
f9=Wn9*(p9*h(1) + r9);
f10=Wn10*(p10*h(1) + r10);
f11=Wn11*(p11*h(1) + r11);
f12=Wn12*(p12*h(1) + r12);
f13=Wn13*(p13*h(1) + r13);
f14=Wn14*(p14*h(1) + r14);
f15=Wn15*(p15*h(1) + r15);

% QUINTA CAPA
F=(f1+f2+f3+f4+f5+f6+f7+f8+f9+f10+f11+f12+f13+f14+f15);

est=estabilizador_caso_4(omega(1),omega(2),omega(1),omega(2));
reg=(F*Gama_1 + (1-F)*Gama_2)*omega;

u=est+reg;

x=T*(h(1)*MA1 + h(2)*MA2)*[omega(1);omega(2)] + T*(h(1)*MB1 +
h(2)*MB2)*u + [omega(1);omega(2)];

e_i=x-omega;

if e_i'*e_i<(1*10^-6)
    break
end

%          cadena=e_i'*(T^2)*(h(1)*MA1 + h(2)*MA2 + (h(1)*MB1 +
h(2)*MB2)*(h(1)*K1 + h(2)*K2)+ 1)*(h(1)*MB1 + h(2)*MB2)*(Gama_1-
Gama_2)*omega;
cadena=e_i'*T*(h(1)*MB1 + h(2)*MB2)*(Gama_1-Gama_2)*omega;
%
g_p1=cadena*Wn1*h(1);
g_p2=cadena*Wn2*h(1);
g_p3=cadena*Wn3*h(1);
g_p4=cadena*Wn4*h(1);
g_p5=cadena*Wn5*h(1);
g_p6=cadena*Wn6*h(1);
g_p7=cadena*Wn7*h(1);
g_p8=cadena*Wn8*h(1);

```

```

g_p9=cadena*Wn9*h(1);
g_p10=cadena*Wn10*h(1);
g_p11=cadena*Wn11*h(1);
g_p12=cadena*Wn12*h(1);
g_p13=cadena*Wn13*h(1);
g_p14=cadena*Wn14*h(1);
g_p15=cadena*Wn15*h(1);

g_r1=cadena*Wn1;
g_r2=cadena*Wn2;
g_r3=cadena*Wn3;
g_r4=cadena*Wn4;
g_r5=cadena*Wn5;
g_r6=cadena*Wn6;
g_r7=cadena*Wn7;
g_r8=cadena*Wn8;
g_r9=cadena*Wn9;
g_r10=cadena*Wn10;
g_r11=cadena*Wn11;
g_r12=cadena*Wn12;
g_r13=cadena*Wn13;
g_r14=cadena*Wn14;
g_r15=cadena*Wn15;

g=[g_p1;g_r1;g_p2; g_r2;g_p3;
g_r3;g_p4;g_r4;g_p5;g_r5;g_p6;g_r6;g_p7;g_r7;g_p8;g_r8;g_p9;g_r9;g_p10;g_
r10;g_p11;g_r11;g_p12;g_r12;g_p13;g_r13;g_p14;g_r14;g_p15;g_r15];

theta=theta - (inv(g'*g + lambda*I))*0.5*g;

[dA1_a dA1_b dA1_c]=derivada_GB(A1,a1,b1,c1,h(1));
[dA2_a dA2_b dA2_c]=derivada_GB(A2,a2,b2,c2,h(1));
[dA3_a dA3_b dA3_c]=derivada_GB(A3,a3,b3,c3,h(1));
[dA4_a dA4_b dA4_c]=derivada_GB(A4,a4,b4,c4,h(1));
[dA5_a dA5_b dA5_c]=derivada_GB(A5,a5,b5,c5,h(1));
[dA6_a dA6_b dA6_c]=derivada_GB(A6,a6,b6,c6,h(1));
[dA7_a dA7_b dA7_c]=derivada_GB(A7,a7,b7,c7,h(1));
[dA8_a dA8_b dA8_c]=derivada_GB(A8,a8,b8,c8,h(1));
[dA9_a dA9_b dA9_c]=derivada_GB(A9,a9,b9,c9,h(1));
[dA10_a dA10_b dA10_c]=derivada_GB(A10,a10,b10,c10,h(1));
[dA11_a dA11_b dA11_c]=derivada_GB(A11,a11,b11,c11,h(1));
[dA12_a dA12_b dA12_c]=derivada_GB(A12,a12,b12,c12,h(1));
[dA13_a dA13_b dA13_c]=derivada_GB(A13,a13,b13,c13,h(1));
[dA14_a dA14_b dA14_c]=derivada_GB(A14,a14,b14,c14,h(1));
[dA15_a dA15_b dA15_c]=derivada_GB(A15,a15,b15,c15,h(1));

%

a1= a1 - alpha*cadena*((p1*h(1) + r1 -F)/Swi)*dA1_a;
a2= a2 - alpha*cadena*((p2*h(1) + r2 -F)/Swi)*dA2_a;
a3= a3 - alpha*cadena*((p3*h(1) + r3 -F)/Swi)*dA3_a;
a4= a4 - alpha*cadena*((p4*h(1) + r4 -F)/Swi)*dA4_a;
a5= a5 - alpha*cadena*((p5*h(1) + r5 -F)/Swi)*dA5_a;
a6= a6 - alpha*cadena*((p6*h(1) + r6 -F)/Swi)*dA6_a;
a7= a7 - alpha*cadena*((p7*h(1) + r7 -F)/Swi)*dA7_a;
a8= a8 - alpha*cadena*((p8*h(1) + r8 -F)/Swi)*dA8_a;
a9= a9 - alpha*cadena*((p9*h(1) + r9 -F)/Swi)*dA9_a;

```

```

a10= a10 - alpha*cadena*((p10*h(1) + r10 -F)/Swi)*dA10_a;
a11= a11 - alpha*cadena*((p11*h(1) + r11 -F)/Swi)*dA11_a;
a12= a12 - alpha*cadena*((p12*h(1) + r12 -F)/Swi)*dA12_a;
a13= a13 - alpha*cadena*((p13*h(1) + r13 -F)/Swi)*dA13_a;
a14= a14 - alpha*cadena*((p14*h(1) + r14 -F)/Swi)*dA14_a;
a15= a15 - alpha*cadena*((p15*h(1) + r15 -F)/Swi)*dA15_a;

b1= b1 - alpha*cadena*((p1*h(1) + r1 -F)/Swi)*dA1_b;
b2= b2 - alpha*cadena*((p2*h(1) + r2 -F)/Swi)*dA2_b;
b3= b3 - alpha*cadena*((p3*h(1) + r3 -F)/Swi)*dA3_b;
b4= b4 - alpha*cadena*((p4*h(1) + r4 -F)/Swi)*dA4_b;
b5= b5 - alpha*cadena*((p5*h(1) + r5 -F)/Swi)*dA5_b;
b6= b6 - alpha*cadena*((p6*h(1) + r6 -F)/Swi)*dA6_b;
b7= b7 - alpha*cadena*((p7*h(1) + r7 -F)/Swi)*dA7_b;
b8= b8 - alpha*cadena*((p8*h(1) + r8 -F)/Swi)*dA8_b;
b9= b9 - alpha*cadena*((p9*h(1) + r9 -F)/Swi)*dA9_b;
b10= b10 - alpha*cadena*((p10*h(1) + r10 -F)/Swi)*dA10_b;
b11= b11 - alpha*cadena*((p11*h(1) + r11 -F)/Swi)*dA11_b;
b12= b11 - alpha*cadena*((p12*h(1) + r12 -F)/Swi)*dA12_b;
b13= b11 - alpha*cadena*((p13*h(1) + r13 -F)/Swi)*dA13_b;
b14= b11 - alpha*cadena*((p14*h(1) + r14 -F)/Swi)*dA14_b;
b15= b11 - alpha*cadena*((p15*h(1) + r15 -F)/Swi)*dA15_b;

c1= c1 - alpha*cadena*((p1*h(1) + r1 -F)/Swi)*dA1_b;
c2= c2 - alpha*cadena*((p2*h(1) + r2 -F)/Swi)*dA2_b;
c3= c3 - alpha*cadena*((p3*h(1) + r3 -F)/Swi)*dA3_b;
c4= c4 - alpha*cadena*((p4*h(1) + r4 -F)/Swi)*dA4_b;
c5= c5 - alpha*cadena*((p5*h(1) + r5 -F)/Swi)*dA5_b;
c6= c6 - alpha*cadena*((p6*h(1) + r6 -F)/Swi)*dA6_b;
c7= c7 - alpha*cadena*((p7*h(1) + r7 -F)/Swi)*dA7_b;
c8= c8 - alpha*cadena*((p8*h(1) + r8 -F)/Swi)*dA8_b;
c9= c9 - alpha*cadena*((p9*h(1) + r9 -F)/Swi)*dA9_b;
c10= c10 - alpha*cadena*((p10*h(1) + r10 -F)/Swi)*dA10_b;
c11= c11 - alpha*cadena*((p11*h(1) + r11 -F)/Swi)*dA11_b;
c12= c12 - alpha*cadena*((p12*h(1) + r12 -F)/Swi)*dA12_b;
c13= c13 - alpha*cadena*((p13*h(1) + r13 -F)/Swi)*dA13_b;
c14= c14 - alpha*cadena*((p14*h(1) + r14 -F)/Swi)*dA14_b;
c15= c15 - alpha*cadena*((p15*h(1) + r15 -F)/Swi)*dA15_b;
end
%
else
end
est=estabilizador_caso_4(x1(k),x2(k),omega(1),omega(2));
reg=(F*Gama_1 + (1-F)*Gama_2)*omega;
u=est+reg;
x=T*(h(1)*MA1 + h(2)*MA2)*[x1(k);x2(k)] + T*(h(1)*MB1 +
h(2)*MB2)*u + [x1(k);x2(k)];
%
end

ess(k)=error(1);
estabilizer(k)=est;
mu_anfis(k)=F;
h1(k)=h(1);
mu_calc(k)=MSF_analiticcas_caso_4(h(1),h(2));
control(k)=est+reg;

k=k+1;

```

```

        x1(k)=x(1);
        x2(k)=x(2);
%       x1e(k)=xe(1);
%       x2e(k)=xe(2);
        omega1(k)=omega(1);
        omega2(k)=omega(2);
        tiempo(k)=t;
%       alpha=alpha*0.5;
    end % fin del ciclo for de tiempo
t=0:T:time;

```

```

figure(1)
hold on
grid on
plot(tiempo,omegal)
plot(tiempo,x1,'r')
xlabel('Tiempo (s)')
ylabel('Amplitud (rd)')
legend('\omega_1','x_1')
%

```

```

figure(2)
hold on
grid on
plot(t,ess)
xlabel('Tiempo (s)')
ylabel('Amplitud (rd)')
legend('\e_s_s')

```

```

figure(3)
hold on
plot(t,mu_anfis)
plot(t,mu_calc,'r')
xlabel('Tiempo (s)')
ylabel('Amplitud')
legend('\mu_1 ANFIS','mu_1 Analitica')

```

```

figure(4)
plot(t,control)
xlabel('Tiempo (s)')
ylabel('Amplitud (N)')
legend('u')

```

Anexo IV. Publicaciones

R. Tapia-Herrera, J. A. Meda-Campaña, S. Alcántara-Montes, T. Hernández-Cortés, y L. Salgado-Conrado, "Tuning of a TS Fuzzy Output Regulator Using the Steepest Descent Approach and ANFIS," *Mathematical Problems in Engineering*, Hindawi Publishing, pp. 1–14, 2013.

R. Tapia-Herrera, S. Alcántara-Montes, J. A. Meda-Campaña, y A. T. Velázquez-Sánchez, "Kinematic and Dynamic Modelling of Serial Robotic Manipulators Using Dual Number Algebra," *Kinematic and Dynamic Modelling of Serial Robotic Manipulators Using Dual Number Algebra, Serial and Parallel Robot Manipulators - Kinematics, Dynamics, Control and Optimization*, Rijeka, Croacia, InTech, 2012, pp. 67–84.

C. R. Torres-San-Miguel, A. T. Velázquez-Sánchez, E. Lugo-González, y R. Tapia-Herrera, "Diseño personalizado de una interfaz mioeléctrica para una prótesis de miembro superior," *Revista Colombiana Biotecnología*, vol. XIII, no. 2, pp. 70–83, 2011.

R. Tapia-Herrera, C. Palacios-Montúfar, S. Alcántara-Montes, y J. A. Flores-Campos, "Dynamic Analysis of an Industrial Robot Manipulator Using Dual Numbers," *2010 IEEE Electronics, Robotics and Automotive Mechanics Conference*, Cuernavaca, Morelos, México, pp. 331–336.

C. Palacios-Montúfar, I. García-Ríos, R. Tapia-Herrera, E. Vargas-Rojas, y J. A. Flores-Campos, *Dinámica de Mecanismos y Máquinas*, 1ª edición. México, D.F., Instituto Politécnico Nacional, 2010.