CINVESTAV - IPN
Centro de Investigación y de Estudios Avanzados del IPN
Unidad Guadalajara

LABORATORIO DE INGENIERÍA ELÉCTRICA Y
CIENCIAS DE LAS COMPUTACIÓN-GUADALAJARA

CONTROL PD DIFUSO APLICADO A ROBOTS MANIPULADORES

TESIS QUE PRESENTA

LUIS ARTURO NUÑO SÁNCHEZ

PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS

EN LA ESPECIALIDAD DE

INGENIERÍA ELÉCTRICA

Guadalajara Jalisco, Noviembre de 1998

Bibl.

CONTROL PD DIFUSO APLICADO A ROBOTS MANIPULADORES

Tesis de Maestría en Ciencias

Ingeniería Eléctrica

por:

Luis Arturo Nuño Sánchez

Ingeniero Electrónico

Instituto Tec. y de Estudios Superiores de Occidente, 1992-1996

Becario del CONACYT, expediente no. 112939

Director de Tesis:

Dr. Edgar Nelson Sánchez Camperos

CINVESTAV del IPN Unidad Guadalajara, Noviembre de 1998

Dedicatorias:

Quiero dedicar esta tesis con mucho cariño y respeto a dos de las personas más importantes en mi vida, que han estado cerca de mi apoyándome en todo momento y en mis decisiones, a los que me han acompañado en mis momentos buenos como en los malos, que han sido fuente de ejemplo enseñándome a luchar en la vida sin esperar nada a cambio, a quienes gracias a sus consejos he llegado a crecer tanto como profesionista como hombre, a mi madre Sahara Sánchez Jiménez y a mi padre Luis Nuño García.

Así mismo dedico esta tesis a las personas que me han apoyado incondicionalmente durante todo el tiempo que he realizado mis estudios, a mis hermanos: Noé Rafael, Juan Benjamín y Saúl Alonso. Así como también a todos mis familiares y amigos.

Agradecimientos:

Agradezco a mi asesor, al Dr. Edgar Nelson Sánchez Camperos por la asesoría y el apoyo brindado para la realización de este tema de tesis.

Agradezco al jurado que participo en la evaluación del examen de grado: Dr. Edgar Nelson Sánchez Camperos, Dr. Bernardino Castillo Toledo, Dra. Ofelia Begovich Mendoza, Dr. Alexander Loukianov y al Dr. Fernando Lara Rojo. Así mismo les agradezco los comentarios hechos para mejorar la redacción de la presente memoria.

Agradezco al Dr. Antonio Ramírez Treviño y al Dr. Javier Ruiz por la amistad mostrada en el tiempo que realice mis estudios.

Agradezco a mis compañeros Ma. Elena Meda, Armando Gaytan, Alejandro Carrasco, Thamar E. Mora y Reinaldo García por el apoyo y amistad que me brindaron en el tiempo que estuvimos juntos en CINVESTAV.

Agradezco a todas aquellas personas que conocí en CINVESTAV, que me brindaron su apoyo y amistad. En especial a Raciel, Luis E. Ramos, Armando Govea, Alejandro Leñero, Ramón Parra, Rodolfo Maestre e Ivan Peñuelas.

CAPÍTULO 1
INTRODUCCIÓN

Recientemente, ha surgido un gran interés por el control automático de mecanismos subactuados; esto es, sistemas que tienen más uniones que actuadores. Se han aplicado diferentes esquemas de control, derivados de la teoría de sistemas no lineales [1], técnicas de control clásico [2] o métodos de control inteligente [3,4]. En esta tesis se presenta la aplicación de un controlador PD difuso [5] a un sistema electromecánico subactuado: el Pendubot. Este consiste de dos eslabones; uno de los cuales está en la base del sistema inercial y es actuado, mientras que el segundo tiene movimiento libre.

En cuanto a técnicas de control inteligente, basadas en conceptos de inteligencia computacional, las más exitosas hasta ahora han sido aquéllas en que se utilizan los algoritmos genéticos para sintonizar los controladores difusos. Es importante resaltar que todos los esquemas mencionados, con la excepción de los basados en técnicas de control clásico, son de difícil implementación. Como una vía alternativa, se propone en éste trabajo de tesis un esquema PD difuso mínimo de una estructura muy simple que permite estabilizar el pendubot en un punto inestable y lograr también seguimiento de trayectorias.

El objetivo de éste trabajo de tesis es proponer y aplicar un esquema de control PD difuso mínimo para estabilizar el sistema subactuado en un punto de equilibrio inestable, mediante la permutación entre dos controladores PD difusos [4] (uno para inestabilizar al sistema y otro para estabilizarlo), atenuando el efecto de perturbaciones externas que puedan mover al robot fuera del punto de equilibrio. Además se realiza el seguimiento de una señal de referencia senoidal. Esto se llevó a cabo mezclando el regulador con asignación de polos [6] y el esquema PD difuso.

La presente tesis esta organizada cómo sigue: en el Capítulo 2, se presentan los conceptos fundamentales sobre lógica y control difusos requeridos para el desarrollo de la tesis. Se presenta también una descripción completa del algoritmo de control PD difuso básico a utilizar en esta tesis.

En el Capítulo 3 se presenta el desarrollo de las ecuaciones que describen la dinámica de un robot de dos grados de libertad, se propone el controlador PD difuso que controlará dicho robot realizando un análisis de estabilidad [7] y se hace una comparación entre un controlador PD clásico y un PD difuso visto a nivel simulación.

En el Capítulo 4 se propone el esquema de control PD difuso para el Pendubot, realizando la estabilización y el seguimiento de trayectoria. Además se propone un regulador lineal utilizando el controlador difuso cómo un retroalimentador de estados con perturbación. En todo éste capítulo se presentan tanto resultados de simulación cómo resultados de laboratorio con el fin de hacer una evaluación de éste esquema de control.

En el Capítulo 5 se dan algunas conclusiones y perspectivas generales del trabajo presentado en esta tesis.

En los Apéndices se incluyen los programas creados en Matlab que se utilizaron para modelar el sistema actuado, subactuado, el controlador PD difuso y la función de conmutación. Se incluyen las tablas con las constantes del pendubot, las ganancias de simulación, las ganancias de tiempo real y los artículos que han sido presentados en congresos internacionales.

CAPÍTULO 2
LÓGICA DIFUSA

En éste capítulo se presentan los conceptos fundamentales sobre lógica y control difusos requeridos para el desarrollo de la tesis. Se presenta también una descripción completa del algoritmo de control difuso básico.

2.1 Preliminares teóricos de conjuntos difusos y lógica difusa

2.1.1 Conceptos de lógica difusa

La lógica difusa fue primordialmente diseñada para representar conocimiento expresado lingüísticamente; está motivada en gran medida por la necesidad de un marco de trabajo conceptual para utilizar el lenguaje cómo mecanismo de representación del conocimiento.

Sea U el universo de discurso, que comprende un conjunto definido de objetos. Considere un subconjunto A definido en U, donde la transición entre la pertenencia y la no-pertenencia es gradual más que abrupto. Por ejemplo: sea A el conjunto de mujeres altas en una comunidad U. Usualmente, existen miembros de U que son muy altas, otras que no son altas y casi altas. Entonces una función de pertenencia de grado 1 es asignada a los objetos que pertenecen completamente a A; en éste caso a las mujeres que son muy altas. Los objetos que no pertenecen a A son todas aquéllas que no son definidas cómo altas y a las cuales se les asigna una función de pertenencia de grado 0. Además, a los grados de pertenencia de los casos intermedios (casi altas) se les asignan valores entre 0 y 1. En otras palabras, los conjuntos difusos permiten una representación gradual de una cualidad. En la teoría de conjuntos difusos, los conjuntos clásicos se denominan conjuntos precisos, para

hacer la distinción entre ellos y los conjunto difusos. Sea A un conjunto preciso sobre el universo U, entonces para cualquier elemento u de U, $u \in A$ o $u \notin A$. En la teoría de conjuntos difusos esta propiedad es generalizada; en consecuencia en un conjunto difuso A, no necesariamente $u \in A$ o $u \notin A$.

La generalización se realiza cómo sigue. Para cualquier conjunto preciso A es posible definir una función característica μ_A: $U \rightarrow \{0,1\}$ cómo

$$\mu_A(x) = \begin{cases} 1, & u \in A \\ 0, & u \notin A \end{cases} \tag{2.1}$$

En la teoría de conjuntos difusos, la función característica es generalizada a una función de pertenencia que asigna a cada $u \in U$ un valor en el intervalo de [0, 1], en lugar del conjunto de dos elementos {0, 1}; entonces un conjunto difuso A es un conjunto con grado de pertenencia en el intervalo real $\mu_A(u) \in [0, 1]$.

La función de pertenencia μ_A de un conjunto difuso A es una función

$$\mu_A: U \rightarrow [0, 1] \tag{2.2}$$

por lo tanto, cada elemento de u de U tiene un grado de pertenencia $\mu_A(u) \in [0, 1]$. A está completamente determinado por el conjunto de pares

$$A = \{(u, \mu_A(u)) \mid u \in U\} \tag{2.3}$$

Zadeh [8] propone otra notación para conjuntos difusos. Supónga que A es un conjunto preciso finito $\{u_1, u_2, \ldots, u_n\}$, entonces una notación alterna es

$$A = u_1 + u_2 + \ldots + u_n \tag{2.4}$$

donde + denota enumeración. Mas aún, una notación alterna para un conjunto difuso es $\mu(u)/u$, donde / denota un par.

Sea el conjunto difuso A, un subconjunto definido en el universo de discurso U, entonces el conjunto $A = \{(u, \mu_A(u)) \mid u \in U\}$ puede ser descrito cómo

$$A = \mu_A(u_1)/u_1 + \cdots \mu_A(u_n)/u_n = \sum_{u \in U} \mu_A(u)/u \tag{2.5}$$

donde Σ indica enumeración de cada elemento para un universo discreto U; cuando U es infinito, se puede reescribir la ecuación anterior cómo

$$A = \int_U \mu_A(u)/u \tag{2.6}$$

donde el símbolo de $\int$ denota una enumeración infinita.

Para las aplicaciones de control difuso, los siguientes tipos de conjuntos difusos son usados: "creciente", "decreciente" y la combinación de ellos, que resultan en diversos tipos de conjuntos difusos llamados convexos. Un conjunto difuso A es convexo si y sólo si

$$\forall x, y \in X, \forall \lambda \in [0,1]: \mu_A(\lambda \cdot x + (1-\lambda) \cdot y) \geq min(\mu_A(x), \mu_A(y)) \tag{2.7}$$

Los conjuntos difusos no convexos son conjuntos difusos que alternativamente crecen y decrecen.

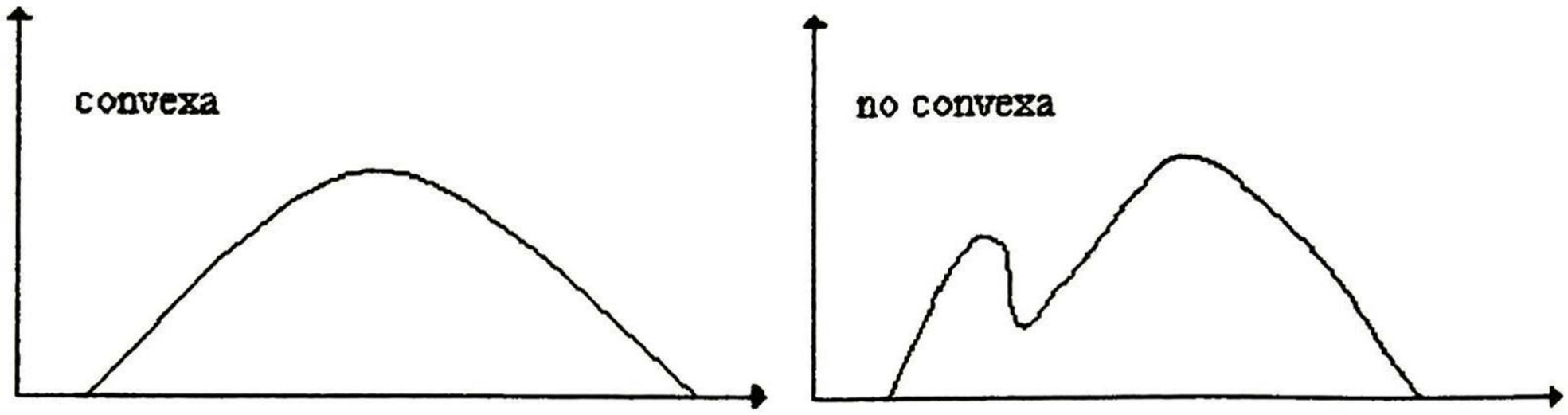


Figura 2.1 Conjunto difuso convexo y conjunto difuso no convexo

Los conjuntos difusos crecientes son utilizados para representar nociones lingüísticas cómo 'alto', 'caliente', etc. en sus respectivos dominios (altura(m), temperatura (°C), etc.). Ejemplos de conjuntos difusos decrecientes en sus dominios son bajo, frío, etc.

Las funciones de pertenencia de conjuntos difusos, pueden ser construidas con las siguientes funciones rectilíneas:

- La función $\Gamma : U \rightarrow [0,1]$ es una función con dos parámetros definida cómo

$$\Gamma(u;\alpha,\beta)=\begin{cases}0 & u<\alpha,\\ (u-\alpha)/(\beta-\alpha) & \alpha\le u\le\beta,\\ 1 & u>\beta.\end{cases} \qquad (2.8)$$

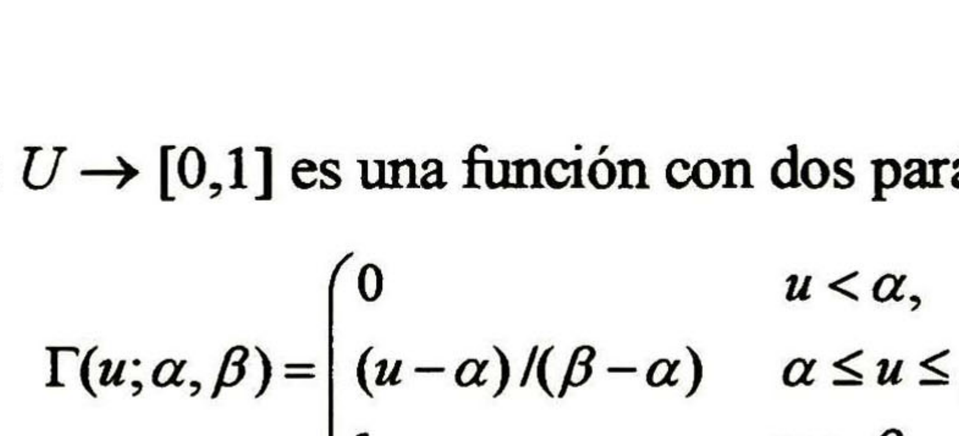


Figura 2.2 Ejemplo de la función Γ

- La función S definida cómo

$$S(x;\alpha,\beta,\gamma)=\begin{cases}0 & para\ x\le\alpha,\\ 2\left(\dfrac{x-\alpha}{\gamma-\alpha}\right)^2 & para\ \alpha<x\le\beta,\\ 1-2\left(\dfrac{x-\gamma}{\gamma-\alpha}\right)^2 & para\ \beta<x\le\gamma,\\ 1 & para\ x>\gamma.\end{cases} \tag{2.9}$$

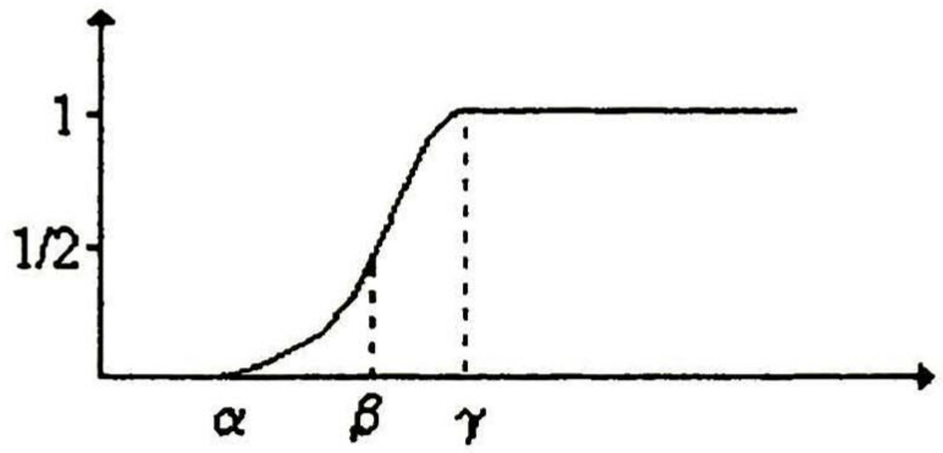


Figura 2.3 Ejemplo de la función S

donde $\beta = (\alpha + \gamma)/2$; esta función puede ser considerada cómo una variante de la función Γ. Es utilizada frecuentemente en la lógica difusa, pero rara vez en el control difuso.

- La función $L : U \rightarrow [0,1]$ es una función con dos parámetros definida cómo

$$L(u;\alpha,\beta)=\begin{cases}1 & u<\alpha,\\ (\beta-u)/(\beta-\alpha) & \alpha\le u\le\beta,\\ 0 & u>\beta.\end{cases} \tag{2.10}$$

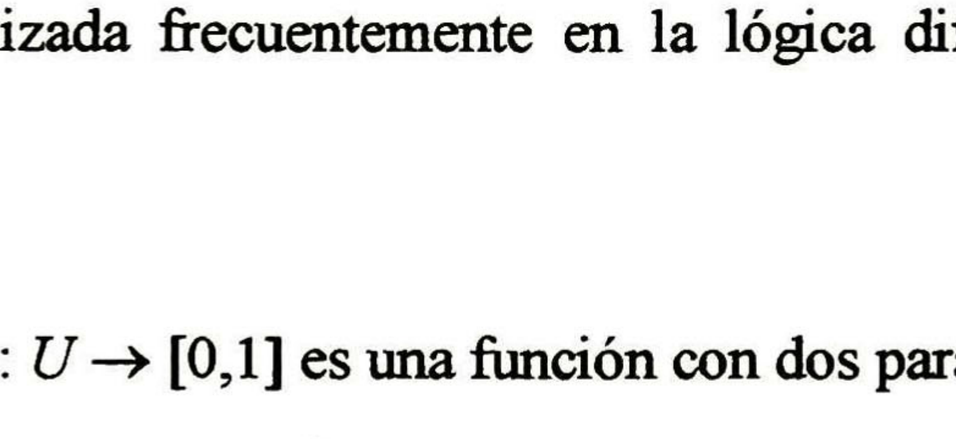


Figura 2.4 Ejemplo de la función L

- La función $\Lambda : U \rightarrow [0,1]$ es una función con tres parámetros definida cómo

$$\Lambda(u;\alpha,\beta,\gamma) = \begin{cases} 0 & u < \alpha, \\ (u-\alpha)/(\beta-\alpha) & \alpha \le u \le \beta, \\ (\gamma-u)/(\gamma-\beta) & \beta \le u \le \gamma, \\ 0 & u > \gamma. \end{cases} \tag{2.11}$$

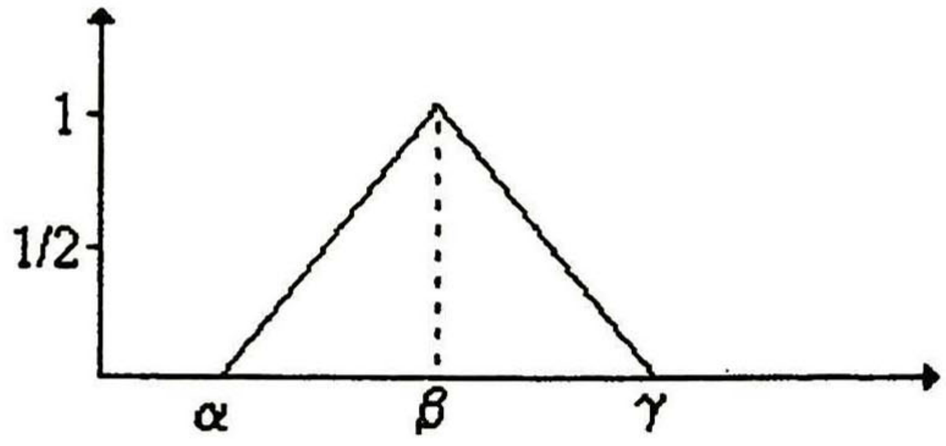


Figura 2.5 Ejemplo de la función Λ

- La función $\Pi : U \rightarrow [0,1]$ es una función con cuatro parámetros definida cómo

$$\Pi(u;\alpha,\beta,\gamma,\delta) = \begin{cases} 0 & u < \alpha, \\ (u-\alpha)/(\beta-\alpha) & \alpha \le u \le \beta, \\ 1 & \beta \le u \le \gamma, \\ (\gamma-u)/(\delta-\gamma) & \gamma < u \le \delta, \\ 0 & u > \delta \end{cases} \tag{2.12}$$

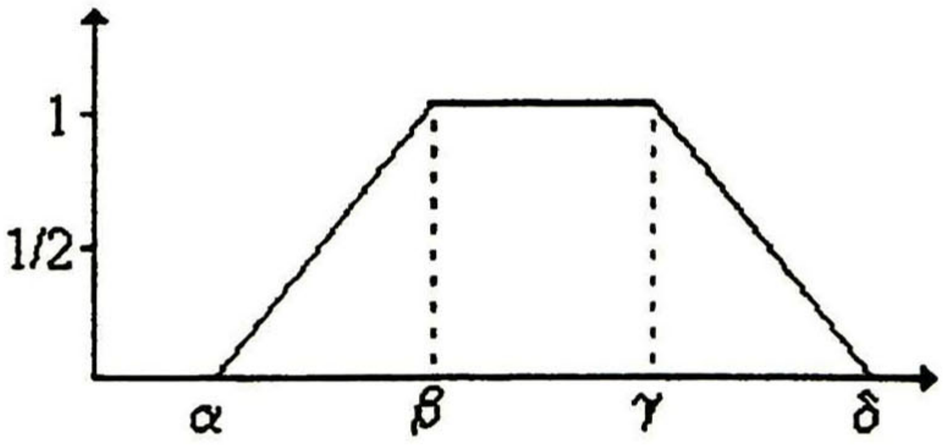


Figura 2.6 Ejemplo de la función Π

2.1.2 Propiedades de los conjuntos difusos

Considere un intervalo $X \subset U$, entonces se pueden establecer las siguientes características estructurales para un conjunto difuso, A:

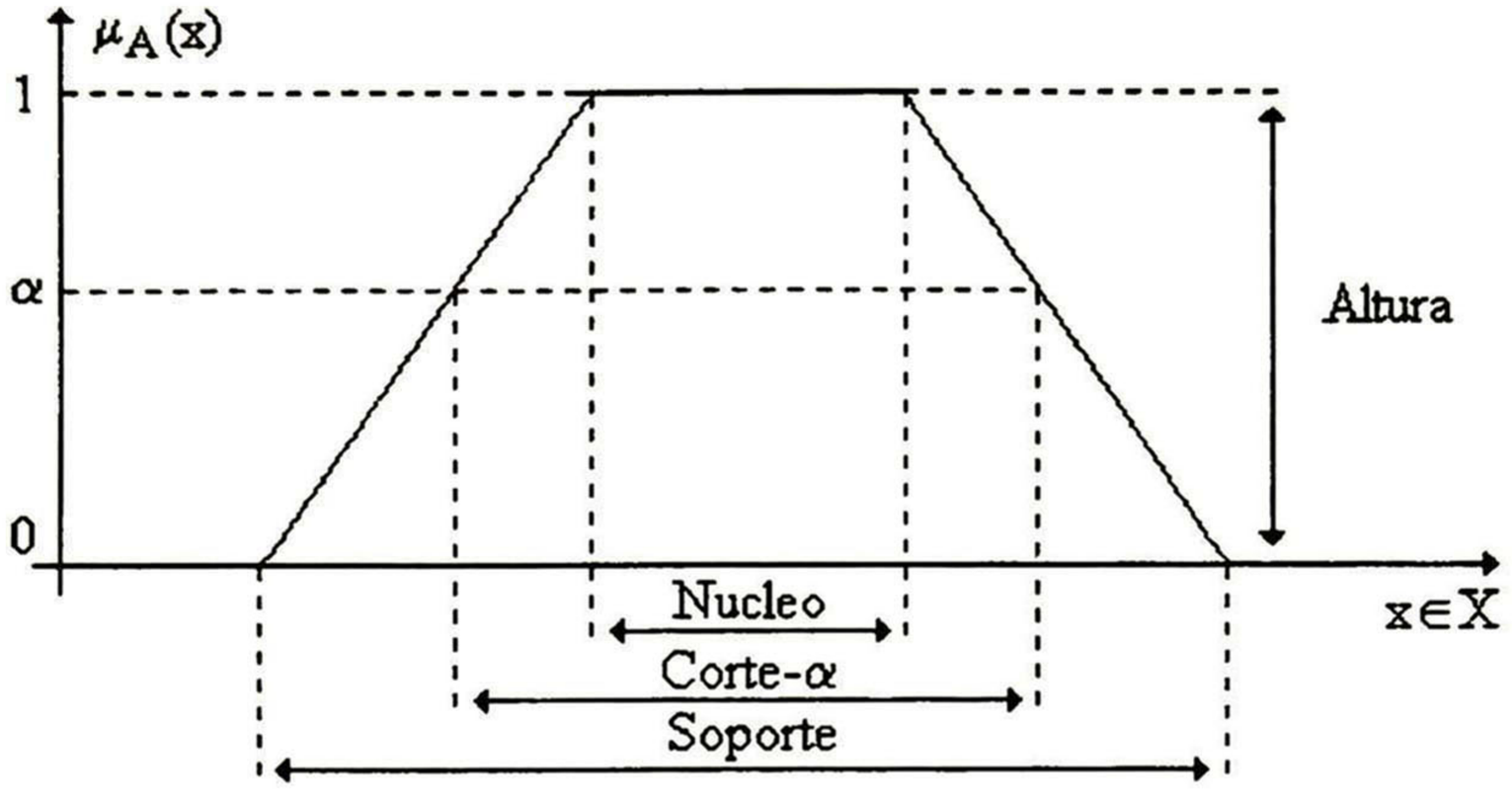


Figura 2.7 Características estructurales de un conjunto difuso

- *altura de A:* $h(A) = sup\ \{\mu_A(x)\}$, si $h(A) = 1$, A se dice que es normal.
- *núcleo de A:* $n(A) = \{ x \in X / \ \mu_A(x) = 1 \}$.
- *corte-α de A:* $c(A) = A_\alpha = \{ x \in X / \mu_A(x) \geq \alpha \}$.
- *soporte de A:* $s(A) = \{x \in X / \mu_A(x) > 0 \}$.

A es más específico que B si:

- $n(A) \subseteq n(B)$
- $s(A) \subseteq s(B)$

A más preciso que B si:

- A y B tienen el mismo núcleo
- $s(A) \subseteq s(B)$

En la figura 2.8, el conjunto difuso A es más específico que el conjunto difuso B. El conjunto difuso C es más preciso que el conjunto difuso D.

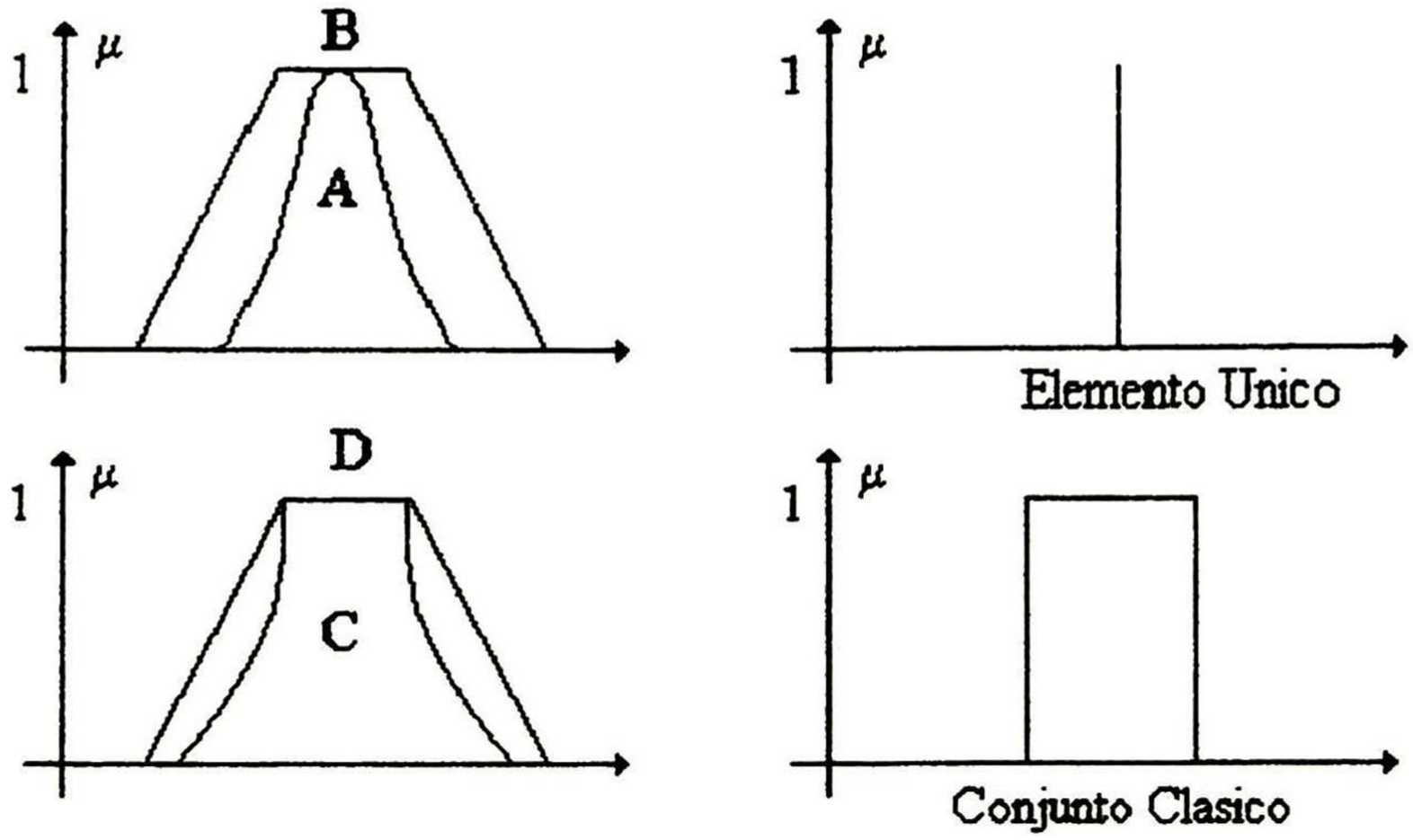


Figura 2.8 Ejemplos de conjuntos difusos

Partición Difusa (Ruspini [9]*):* cuando N conjuntos difusos, A_1 .... A_N, cubren X, éstos conforman una partición difusa si:

$$\forall u \in U, \sum_{i=1}^{N} \mu_{A_i}(u) = 1 \tag{2.13}$$

Una partición difusa formada por conjuntos difusos normales y convexos no debe contener más de dos conjuntos difusos traslapados, puesto que no cumpliría con la definición (2.13).

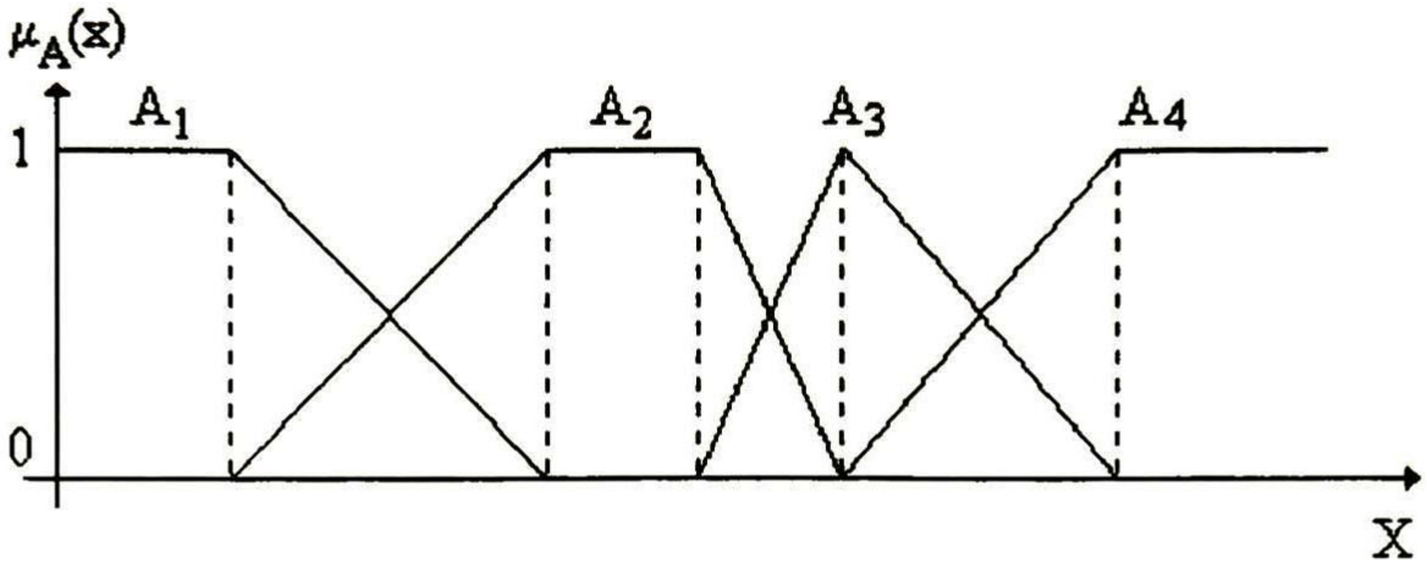


Figura 2.9 Partición difusa

Número e intervalo difuso: son conjuntos difusos especiales

- Los cuales son convexos y normales
- Con una función de pertenencia lineal a tramos
- Con un sólo elemento en el núcleo → número difuso
- Con más de un sólo elemento en el núcleo → intervalo difuso

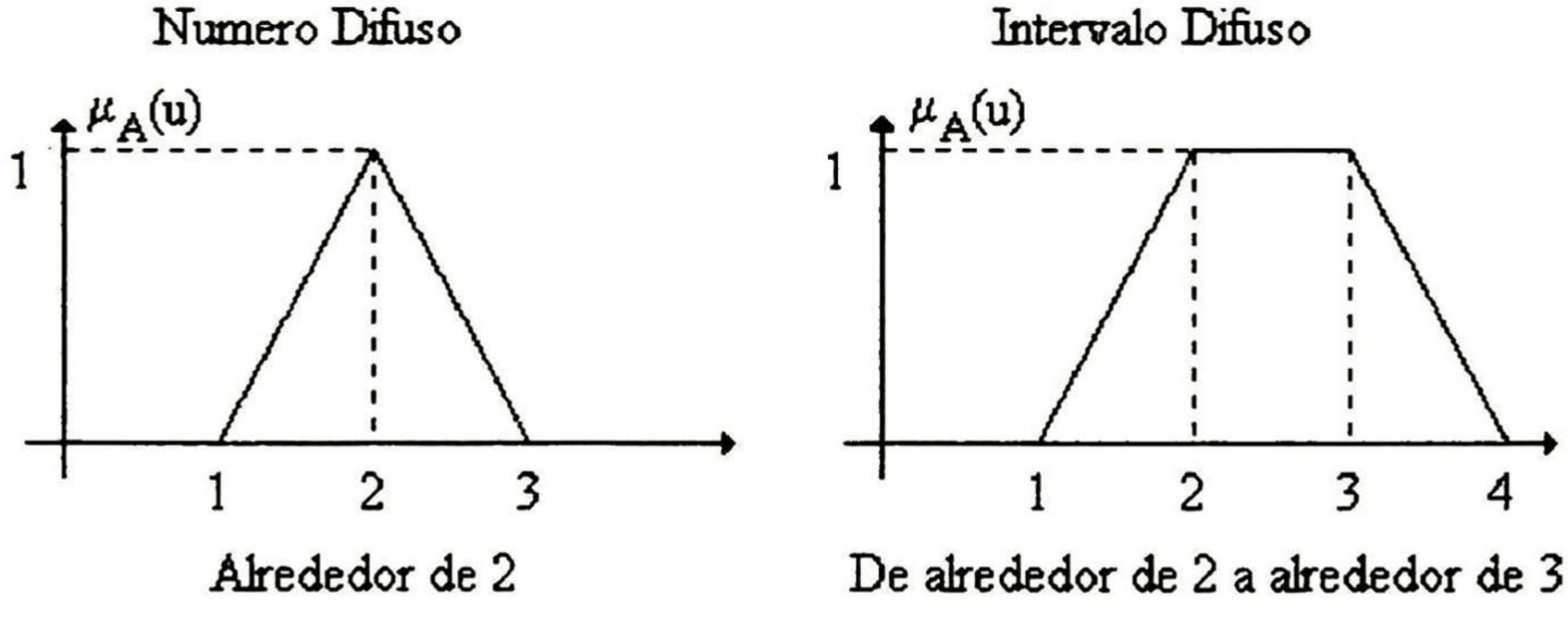


Figura 2.10 Número e intervalo difuso

2.1.3 Operaciones en conjuntos difusos

Nociones cómo igualdad e inclusión de dos conjuntos difusos se derivan inmediatamente de la teoría de conjuntos precisos [8]. Dos conjuntos difusos son iguales si todo elemento del universo de discurso tiene el mismo grado de

pertenencia en cada uno de ellos. Un conjunto difuso A es un subconjunto del conjunto difuso B, si cada elemento del universo tiene un grado de pertenencia menor en A que en B.

Dos conjuntos difusos son iguales ($A=B$) si y sólo si

$$\forall x \in S : \mu_A(x) = \mu_B(x) \tag{2.14}$$

A es un subconjunto de B ($A \subseteq B$) si y sólo si

$$\forall x \in X : \mu_A(x) \leq \mu_B(x) \tag{2.15}$$

En la teoría clásica de conjuntos la unión, intersección y el complemento de conjuntos son operaciones simples definidas sin ambigüedad. La no-ambigüedad se deriva del hecho que los operadores lógicos $\wedge$ *intersección*, $\vee$ *unión* y $-$ *negacion* tienen una semántica bien definida basada en la lógica proposicional. Por ejemplo "$\phi \wedge \psi$" en la lógica proposicional es verdadera si y sólo si ambas expresiones ϕ y ψ son verdaderas. En la lógica difusa su interpretación no es tan simple:

Unión e intersección:

La generalización de la unión e intersección de conjuntos clásicos a conjuntos difusos no es única.

Proposición de Zadeh [8]:

$$\mu_{A \wedge B}(x) = \min(\mu_A(x), \mu_B(x)) : \text{intersección}$$
$$\mu_{A \vee B}(x) = \max(\mu_A(x), \mu_B(x)) : \text{unión} \tag{2.16}$$

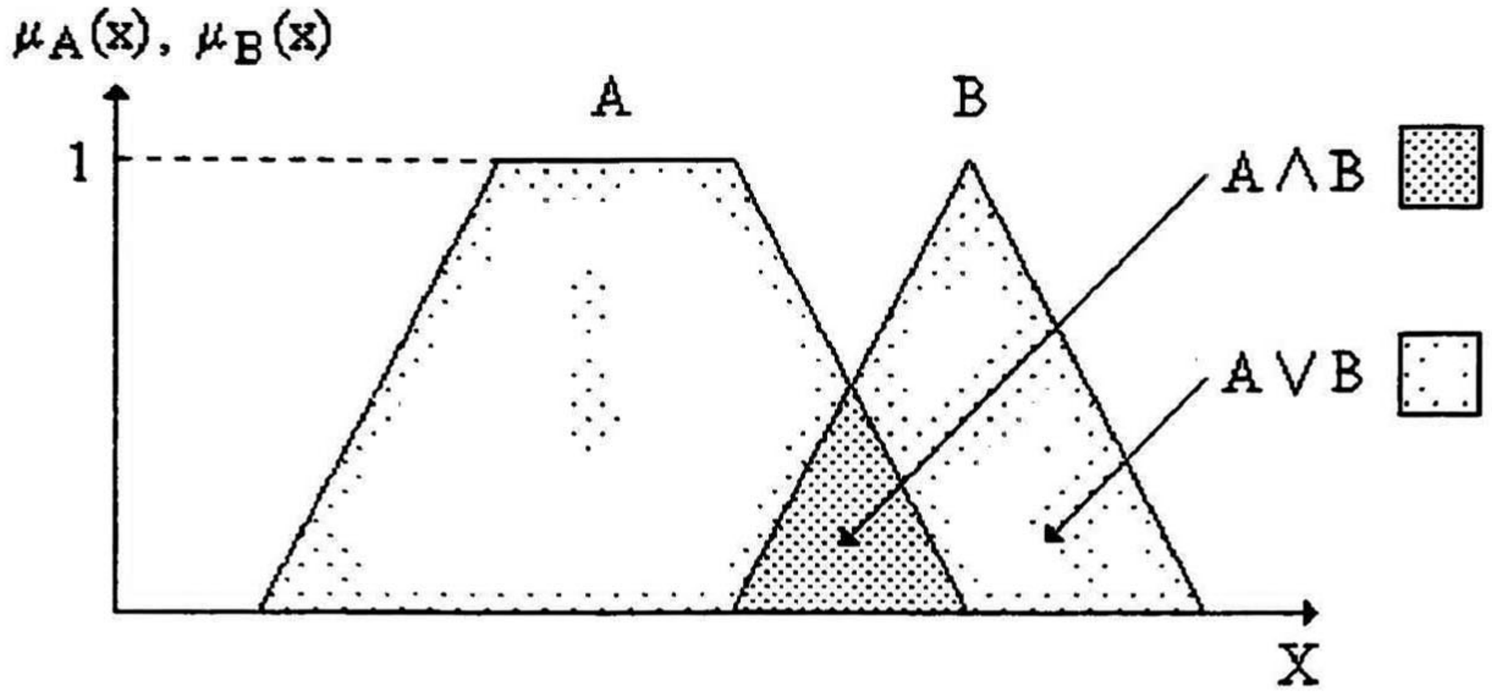


Figura 2.11 Ejemplo de unión e intersección difusa

Estas operaciones se pueden generalizar por:

- normas triangulares (normas-T)
- co-normas triangulares (conormas-T o normas-S)

Una norma-T es un mapeo matemático de $[0,1] \times [0,1] \xrightarrow{T} [0,1]$ representando la operación de intersección, que satisface:

1. $T(a,1) = a$
2. $T(a,b) \leq T(c,d)$ cuando $a \leq b \leq c \leq d$
3. $T(a,b) = T(b,a)$
4. $T(T(a,b),c) = T(a,t(b,c))$

Una norma-S es un mapeo matemático de $[0,1] \times [0,1] \xrightarrow{S} [0,1]$ representando la operación de unión, que satisface:

1. $S(a,0)=a$
2. $S(a,b) \leq S(c,d)$ cuando $a \leq b \leq c \leq d$
3. $S(a,b) = S(b,a)$
4. $S(S(a,b),c) = S(a,S(b,c))$

Una tabla de normas-T y normas-S es presentada en el Apéndice A (Tabla A.1).

Complemento de A:

$$\mu_{\overline{A}}(x) = 1 - \mu_A(x) \tag{2.17}$$

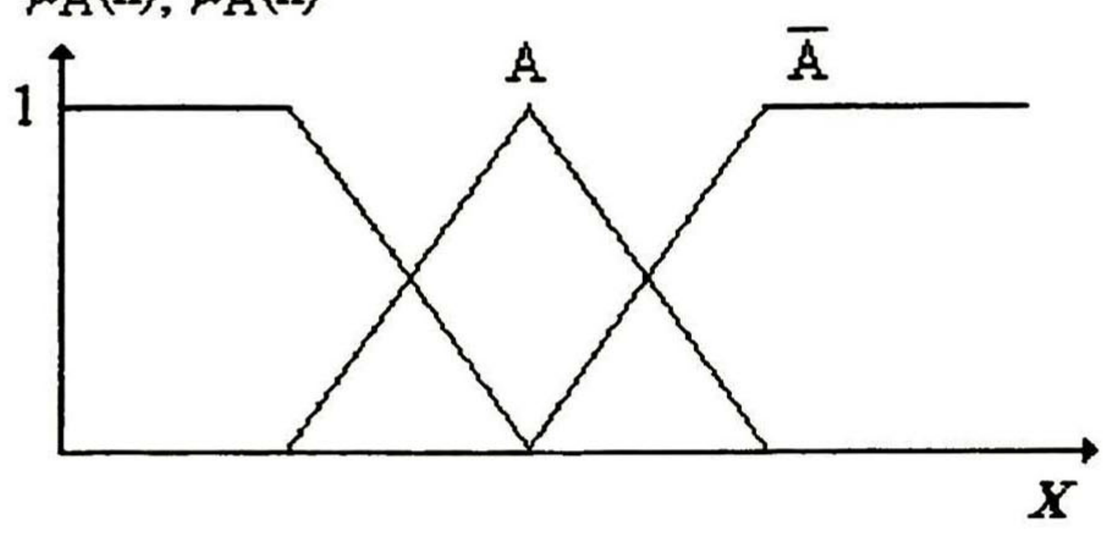


Figura 2.12 Ejemplo del complemento difuso

Para conjuntos difusos, no se cumple la teoría de conjuntos clásicos siendo:

$$A \vee \overline{A} \neq 1$$

$$A \wedge \overline{A} \neq \varnothing \tag{2.18}$$

Producto cartesiano:

Si se consideran diferentes conjuntos difusos simultáneamente, entonces se define un conjunto difuso multidimensional cómo el producto cartesiano de estos conjuntos difusos:

$A = A_1 \times A_2 \times \ldots \times A_n$ definido sobre

$X = X_1 \times X_2 \times \ldots \times X_n$

$$\mu_A(x) = min\left(\mu_{A_1}(x_1), \cdots, \mu_{A_n}(x_n)\right) \tag{2.19}$$

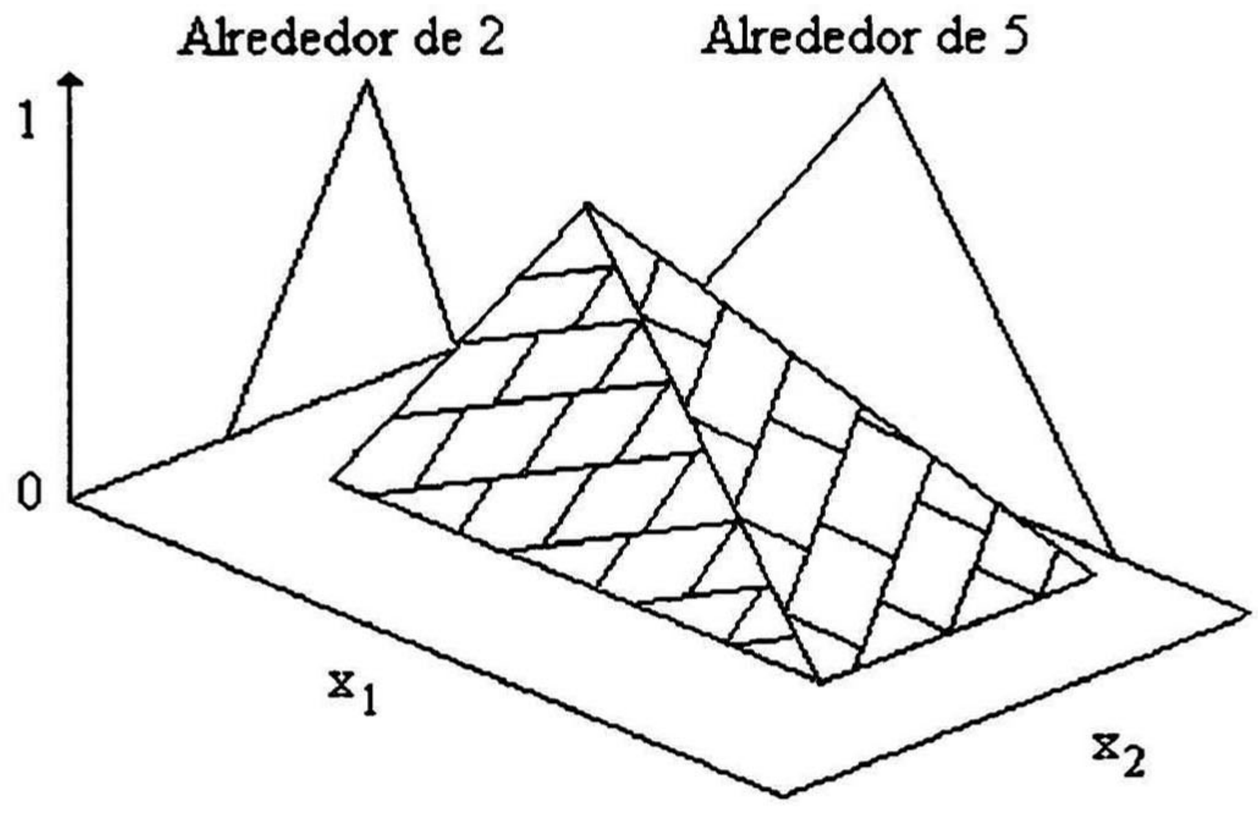


Figura 2.13 Producto cartesiano de subconjuntos difusos

2.1.4 Relaciones Difusas

Una relación puede ser considerada cómo un conjunto de pares ordenados denotado cómo (x,y). Un ejemplo de una relación sobre el dominio de los números naturales, N x N, es:

$$\{(m,n) | \, m \leq n\}, \quad m, n \in N \tag{2.20}$$

Esta relación contiene pares cómo (5,7), (1110,1111) y (30,30). El par (4,1) no está en la relación. Como se sabe las relaciones pueden tener las propiedades de reflexibidad, simetría y/o transitividad.

De la misma manera, una relación difusa es un conjunto difuso de pares ordenados; esto es a cada par ordenado le corresponde un grado de pertenencia entre 0 y 1. Sea U y V universos continuos, y μ_R: U x $V \rightarrow$[0,1] entonces:

$$R = \int_{UxV} \mu_R(u,v)/(u,v) \tag{2.21}$$

es una relación difusa binaria sobre U x V. Si U y V son discretos, entonces

$$R = \sum_{UxV} \mu_R(u,v)/(u,v) \tag{2.22}$$

El símbolo de integral denota el conjunto de todas las duplas $\mu_R(u,v)/(u,v)$ sobre U x V. También es posible expresar la ecuación (2.21) cómo $\int_U \int_V \mu_R(u,v)/(u,v)$, esto es cómo una doble integral.

2.1.5 Operaciones con relaciones difusas

En esta sección se definen operaciones importantes en relaciones difusas. Las relaciones difusas son fundamentales para el control difuso porque describen interacciones entre variables. Esto es de particular interés en las reglas *si-entonces*. Dos operaciones importantes en los conjuntos difusos y relaciones difusas son la proyección y la extensión cilíndrica.

La proyección de un conjunto difuso de R sobre V se define cómo:

$$proj\ R\ en\ V = \int_V \sup_{x_{j_1}\ldots x_{j_l}} \mu_R(x_1,\ldots x_n)/(x_{i_1},\ldots x_{i_k}) \tag{2.23}$$

En el caso binario, se tiene (R definido en X x Y)

$$proj\ R\ en\ Y = \int_Y \sup_x \mu_R(x,y)/y \tag{2.24}$$

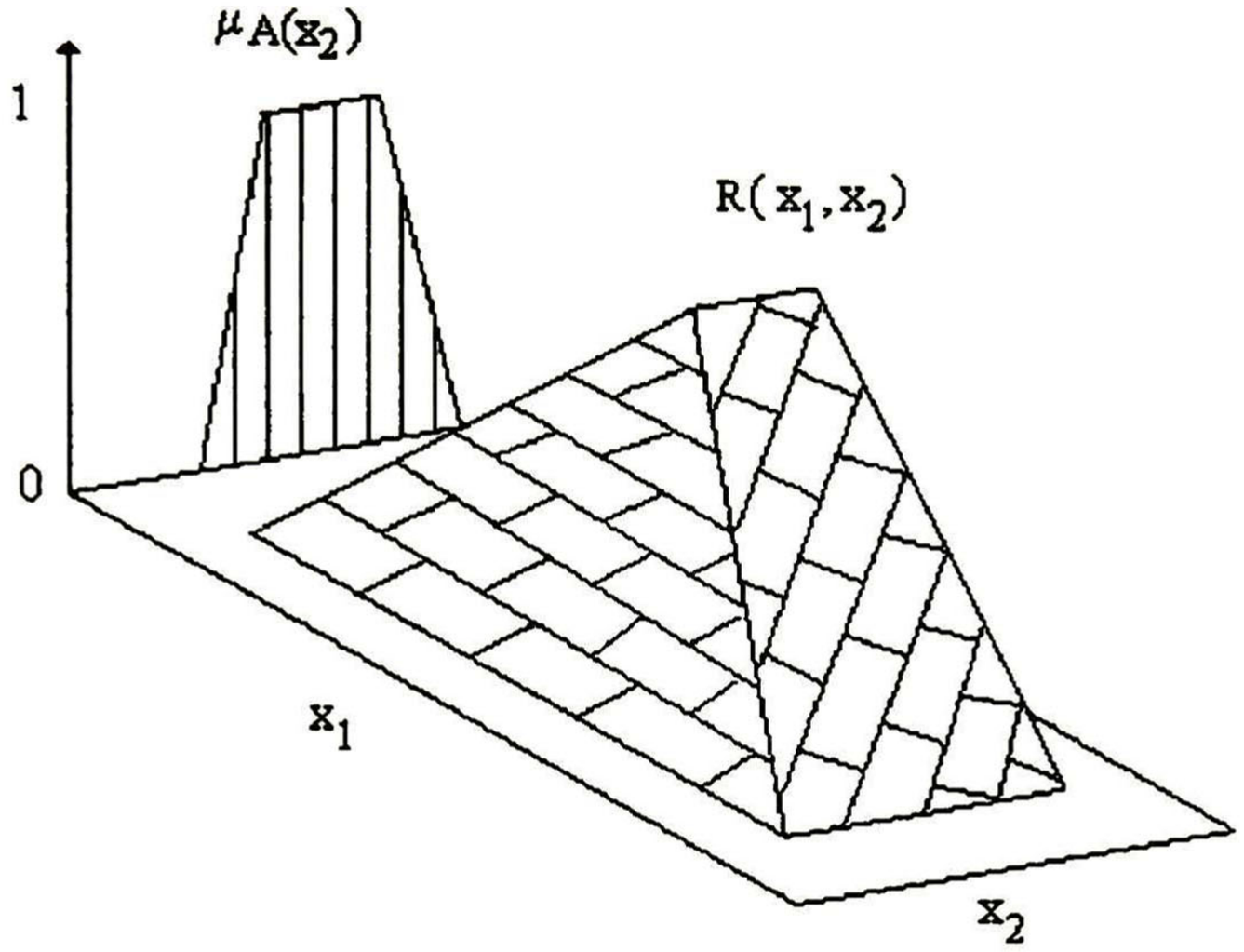


Figura 2.14 Proyección de R en X_2

La operación de proyección es una de las más utilizadas en combinación con la extensión cilíndrica.

La extensión cilíndrica es más o menos lo opuesto de la proyección. Aquélla extiende un conjunto difuso a una relación difusa binaria, una relación difusa binaria a una relación difusa ternaria, etc.

La extensión cilíndrica de S en U

$$ce(S) = \int_U \mu_S(x_{i_1},\ldots,x_{i_k})/(x_1,\ldots,x_n) \qquad (2.25)$$

En el caso binario (sea F un conjunto definido sobre Y), la extensión cilíndrica de F sobre X x Y es el conjunto de todas las duplas $(x,y) \in X$ x Y con un grado de pertenencia igual a $\mu_F(y)$, esto es

$$ce(S) = \int_{XxY} \mu_F(y)/(x,y) \tag{2.26}$$

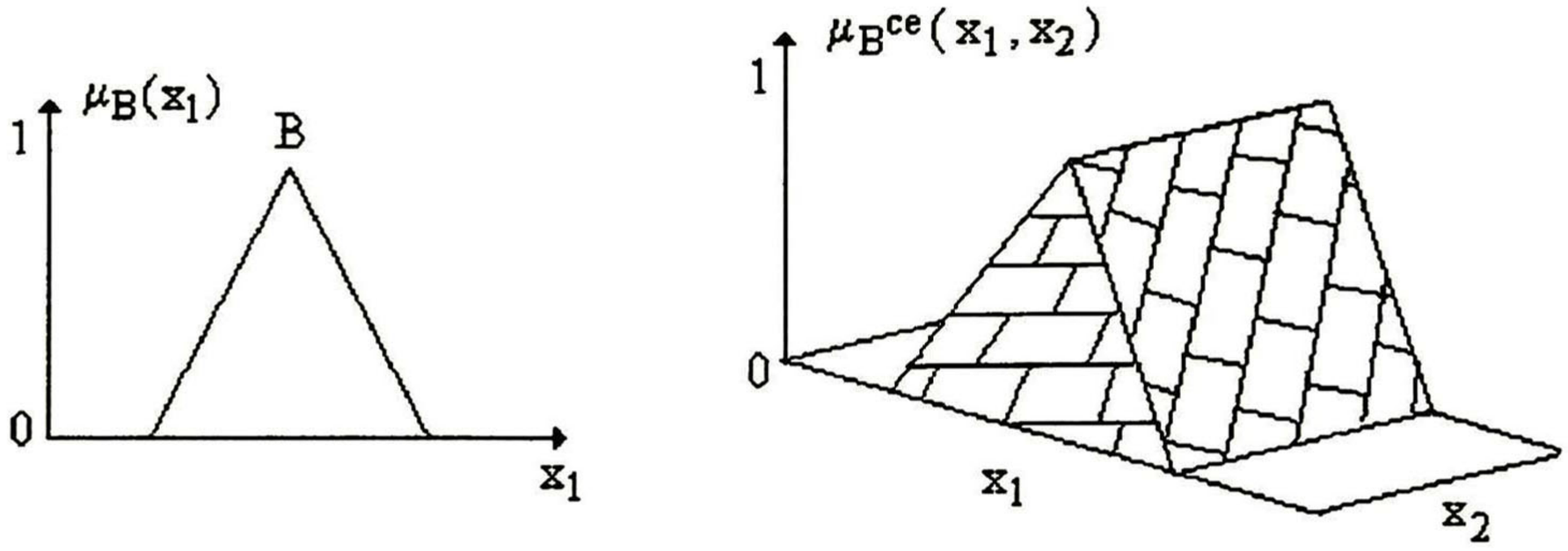


Figura 2.15 Extensión cilíndrica de B en X_1 x X_2

Composición de relaciones difusa

Sea A un conjunto difuso definido sobre X y sea R una relación difusa definida en X x Y. Entonces la composición de A y R da cómo resultado un conjunto difuso B sobre Y, definido cómo:

$$B = A \circ R = proj((ce(A) \cap R) en Y) \tag{2.27}$$

o, si la intersección es realizada con el operador mínimo y la proyección con el máximo

$$\mu_B = max\ min(\mu_A(x), \mu_R(x,y)) \tag{2.28}$$

esta es llamada la composición *max-min.* Si la intersección es realizada con el producto y la proyección con el máximo, tenemos

$$\mu_B(y) = \max_x \mu_A(x) \cdot \mu_R(x,y) \tag{2.29}$$

que es llamada composición *max-punto* o *max-producto.*

2.1.6 Razonamiento Difuso (razonamiento aproximado)

Variables lingüísticas y proposiciones difusas:

Una variable lingüística se puede definir cómo el trío (V, X, T) donde:

V= variable lingüística

X= universo de discurso

T= conjunto de valores lingüísticos tomados por V y descritos por conjuntos difusos,

Ejemplo:

V = altura

$X = R^+$

T= (bajo, medio, alto)

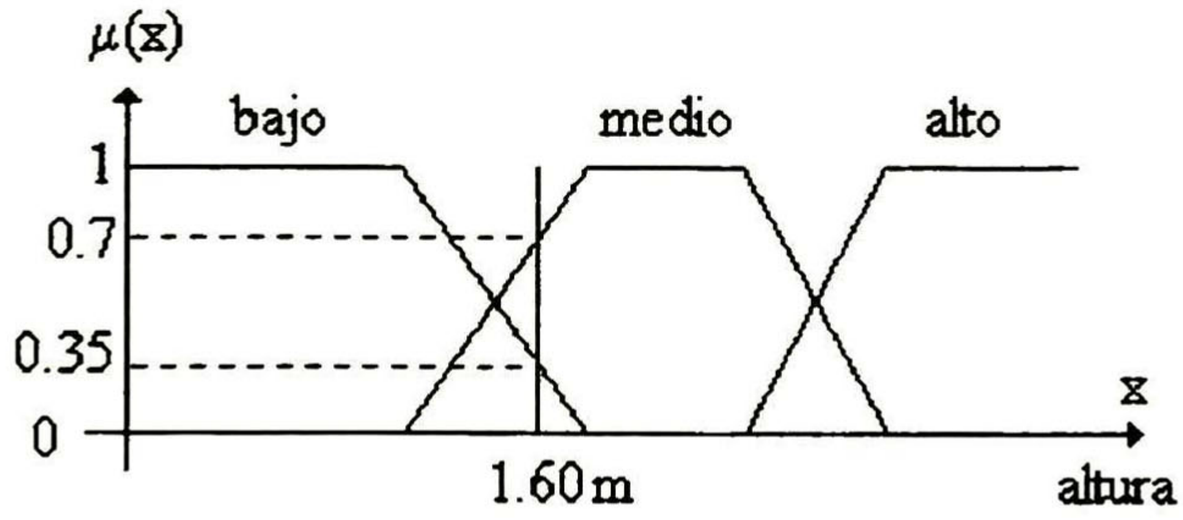


Figura 2.16 Ejemplo de variable lingüística

Interface numérica-simbólica:

1.60 m → (bajo (0.35), medio (0.7), alto (0))

Una proposición difusa es una proposición elemental de la forma "V es A" donde V es una variable lingüística y $A \in T$.

Reglas difusas: Una regla difusa es de la forma:

SI (X es A) ENTONCES (Y es B) $\equiv$ SI A ENTONCES B

Reglas más complejas se pueden definir como:

→ antecedentes múltiples conjuntivos

SI { (X_1 es A_1) $\wedge$ (X_1 es A_2) … } ENTONCES (Y es B)

que puede ser escrito como SI A ENTONCES B con $A = A_1 \wedge A_2$ …

→ antecedentes múltiples disyuntivos

SI { (X_1 es A_1) $\vee$ (X_1 es A_2) … } ENTONCES (Y es B)

que puede ser escrito como SI A ENTONCES B con $A = A_1 \vee A_2$ …

Una regla difusa puede ser descompuesta, utilizando una operación de unión para conectar sub-reglas. Entonces tenemos:

- un conjunto de reglas conjuntivas
- un conjunto de reglas disyuntivas

Situaciones diferentes pueden expresarse por reglas difusas:

- SI A_1 ENTONCES (B_1 DE LO CONTRARIO B_2) es descompuesta en:

  SI A_1 ENTONCES $B_1 \vee$ SI $\overline{A}_1$ ENTONCES B_2

- SI A_1 ENTONCES B_1 A MENOS QUE A_2 es descompuesta en:

  SI A_1 ENTONCES $B_1 \vee$ SI A_2 ENTONCES $\overline{B}_1$

- SI A_1 ENTONCES (B_1 DE LO CONTRARIO (SI A_2 ENTONCES B_2)) es descompuesta en:

  SI A_1 ENTONCES $B_1 \vee$ SI $\overline{A}_1 \wedge A_2$ ENTONCES B_2

- SI A_1 ENTONCES (SI A_2 ENTONCES B_1) es re escrita como:

  SI $A_1 \wedge A_2$ ENTONCES B_1

Una base de reglas difusas debe de satisfacer un conjunto de propiedades como:

- consistencia
- completitud
- continuidad

Consistencia:

Un conjunto de reglas si-entonces es consistente si ésta no contiene contradicciones; como consecuencia se define que un conjunto de reglas si-entonces es inconsistente si existen dos reglas con el mismo antecedente pero con diferente consecuente, por ejemplo

Si el *error* es *positivo* y *derivada* es *negativa* entonces *la salida* es L

Si el *error* es *negativo* y *derivada* es *negativa* entonces *la salida* es H

Completitud:

Tomando la base de reglas en una estructura matricial representada en la Figura 2.17, cada una de las 25 celdas representa una regla, teniendo únicamente 16 reglas. Definiendo OUT(*e*,*r*) como la función de *e* y *r* se define la completitud como:

$$\forall e,r : h(OUT(e,r)) > 0 \qquad (2.30)$$

en otras palabras cada una de las reglas debe de tener una implicación produciendo una salida, por ejemplo la siguiente regla (Figura 2.17) no tiene una salida definida:

Si *e* es PB y *r* es NB entonces u es <no definido>

Continuidad:

Considerando nuevamente la Figura 2.17, se define la noción de 'regla vecina' como sigue: dos reglas son vecinas, si sus celdas son vecinas. Entonces, la regla 17 tiene las reglas vecinas 12, 16, 18 y 22. Un conjunto de reglas si-entonces son continuas si no tienen un par de reglas vecinas cuyos conjuntos difusos de salida tengan una intersección vacía. Por ejemplo la regla 23 tiene como salida NB y una regla vecina es la 22 que a su vez tiene como salida PB, la que claramente tiene intersecciones vacías.

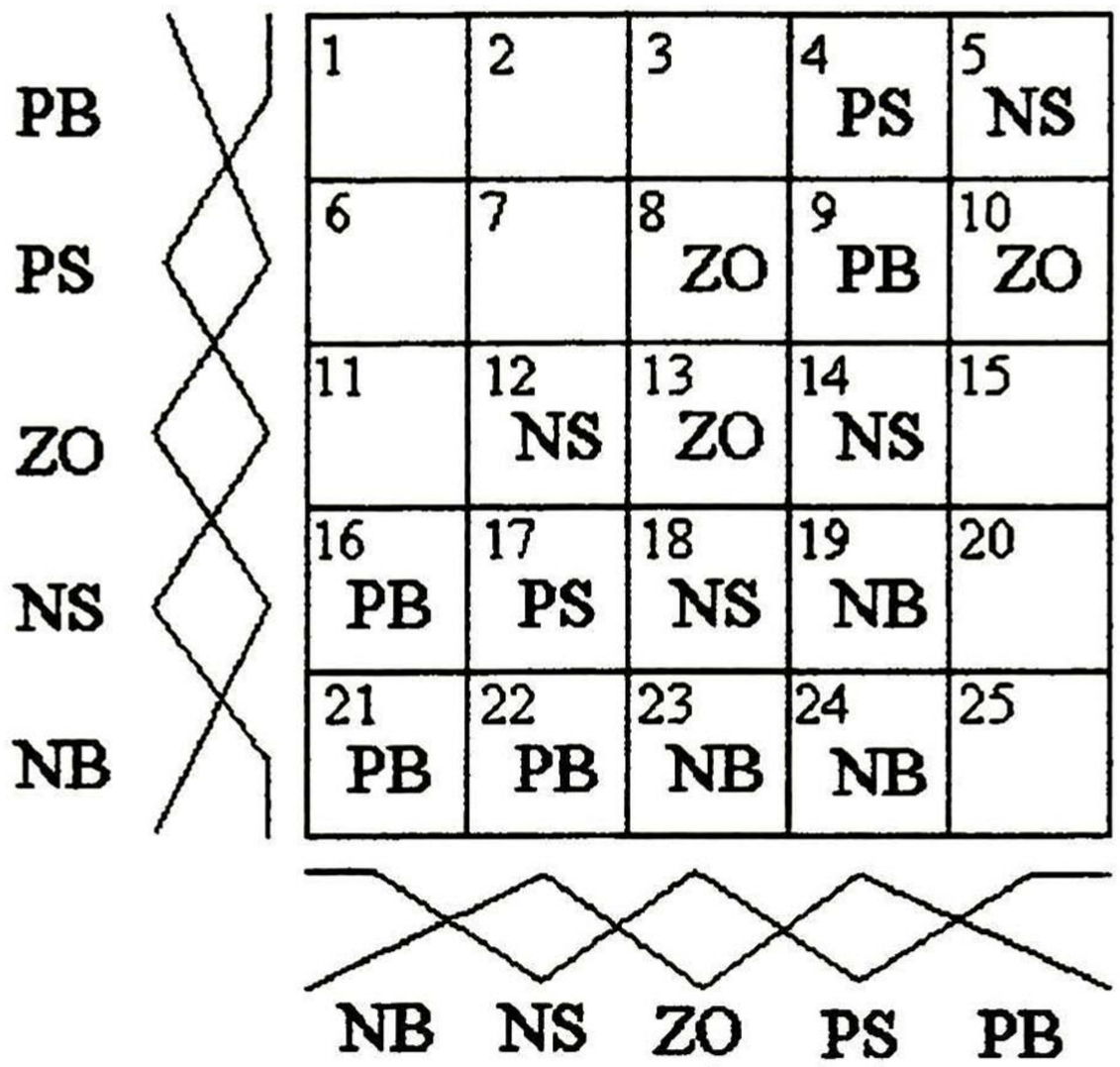


Figura 2.17 Base de reglas en forma matricial

Implicaciones difusas:

Existen muchos operadores de implicación de acuerdo a la interpretación lógica de $A \rightarrow B$ [9]:

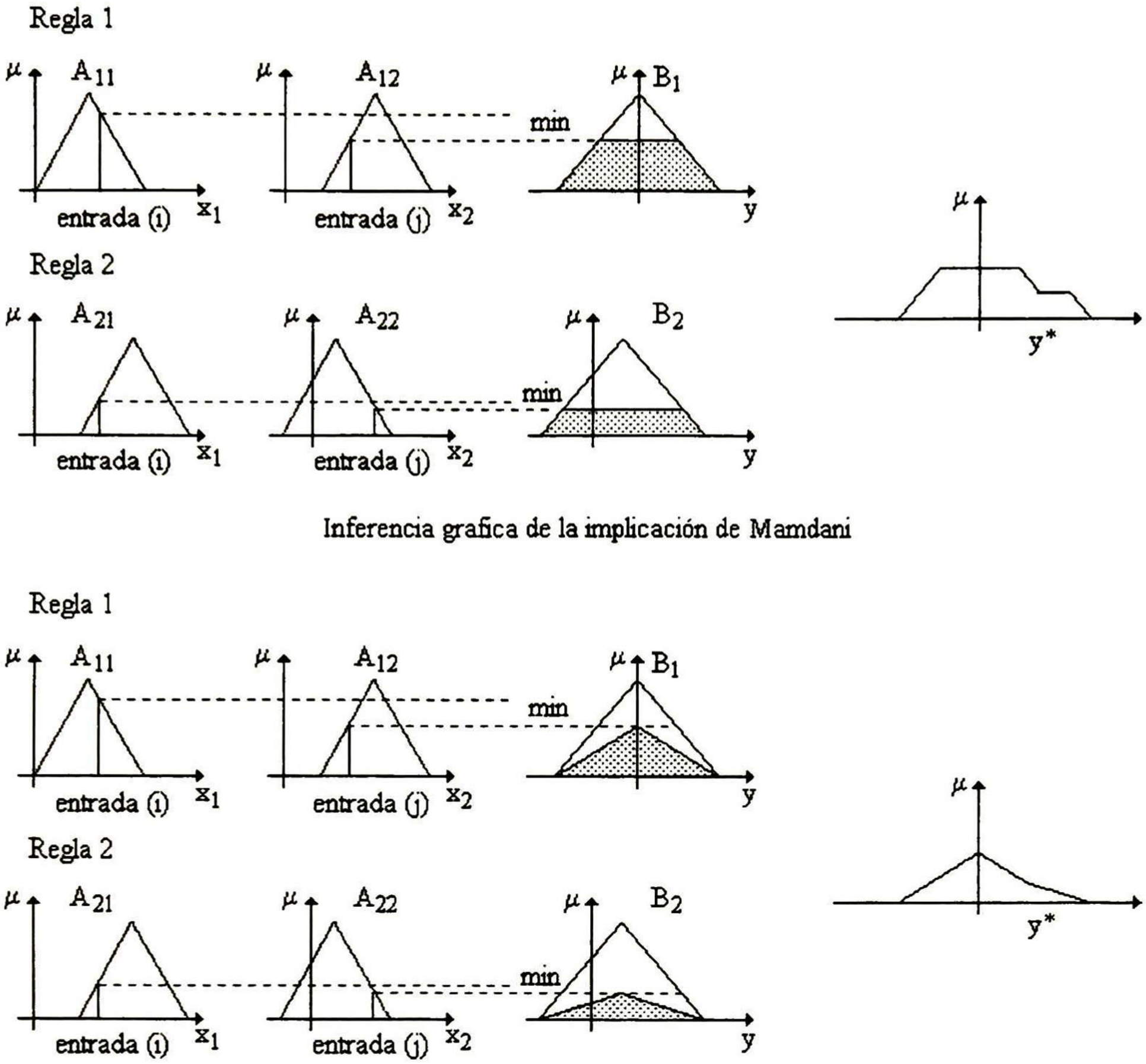
$$\text{desde} \quad A \rightarrow B \equiv \overline{A} \vee B \text{ (implicación clásica)}$$

$$\text{a} \quad A \rightarrow B \equiv A \wedge B \text{ (conjunción clásica)}$$

$a \rightarrow b$	Autor
$Min\,(1 - a + b, 1)$	Lukasiewicz
$min\,(1 - a, min\,(a, b))$	Zadeh
$Max\,(1 - a, b)$	Kleene
$1 - a + ab$	Reichenbach
$\begin{cases} b, si\ a > b \\ 1, para\ toda\ otra\ combinación \end{cases}$	Gödel
$\begin{cases} 1-a, si\ b = 0 \\ b, \quad si\ a = 1 \\ 1, para\ toda\ otra\ combinación \end{cases}$	Dubois y Prade
$\begin{cases} 1, si\ a = 0 \\ min\left(\frac{b}{a}, 1\right) para\ toda\ otra\ combinación \end{cases}$	Goguen
$\begin{cases} 1, si\ a \leq b \\ 0, para\ toda\ otra\ combinación \end{cases}$	Gaines
$\begin{cases} 1, si\ a \leq b \\ min(1-a, b), para\ toda\ otra\ combinación \end{cases}$	Wu
$\begin{cases} 0, si\ a < b \\ b, para\ toda\ otra\ combinación \end{cases}$	Wu
b^a	Yager
$Min\,[\,max\,(1 - a, b), max\,(a, 1 - b, min\,(b, 1 - a))]$	Willmont
$Min\,(a, b)$	Mamdani
ab	Larsen

Tabla 1.1 Implicaciones Difusas a→b

Para aplicaciones en control difuso, dos de las implicaciones más importantes, son la de Mamdani y Larsen que pueden ser interpretadas gráficamente como:



Inferencia grafica de la implicación de Mamdani

Inferencia grafica con la implicacion de Larsen

Figura 2.18 Implicaciones difusas

2.2 Control difuso

Un controlador difuso es un sistema de control basado en el conocimiento, que se expresa en términos de la lógica difusa.

Las reglas que se utilizan son de la forma:

SI $\{(X_1 \text{ es } A_1) \wedge (X_2 \text{ es } A_2)\ldots\}$ ENTONCES (Y es B)

Controlador del tipo Mamdani o

SI $\{(X_1 \text{ es } A_1) \wedge (X_2 \text{ es } A_2)\ldots\}$ ENTONCES $Y = f(X_1, X_2, \ldots)$

Controlador tipo Sugeno

El controlador difuso debe ser conectado al proceso, por lo que se necesita:

- una interface numérica-simbólica: la difusificación
- una interface simbólica-numérica: la desdifusificación

2.2.1 Estructura del controlador difuso

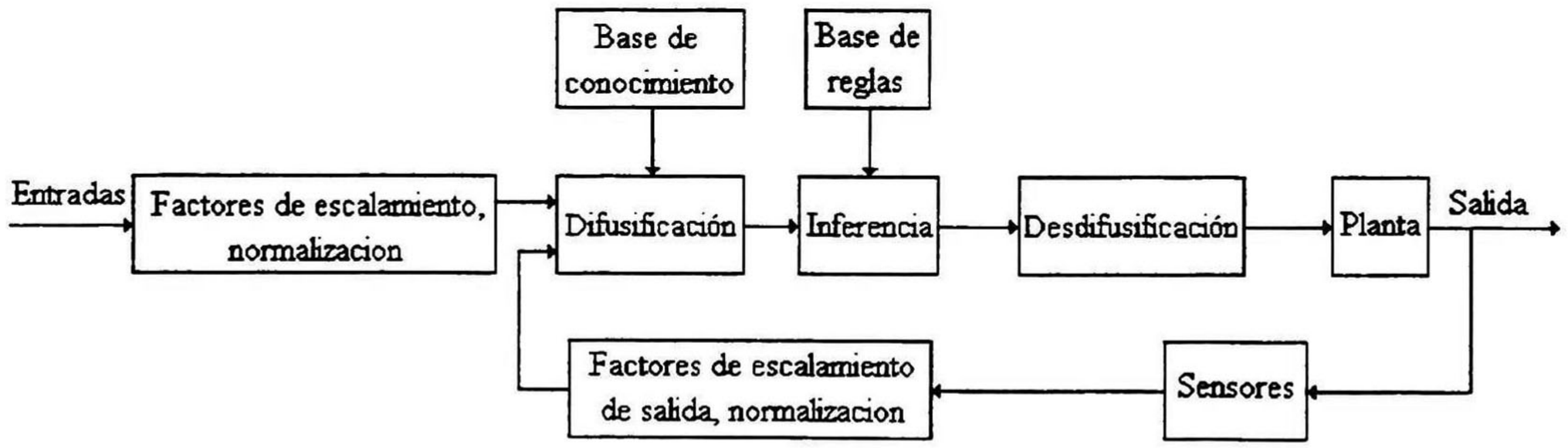


Figura 2.19 Estructura de un controlador difuso

2.2.2 Difusificación-desdifusificación

Difusificación:

Valor preciso: un valor numérico exacto x_0 es transformado en una representación difusa.

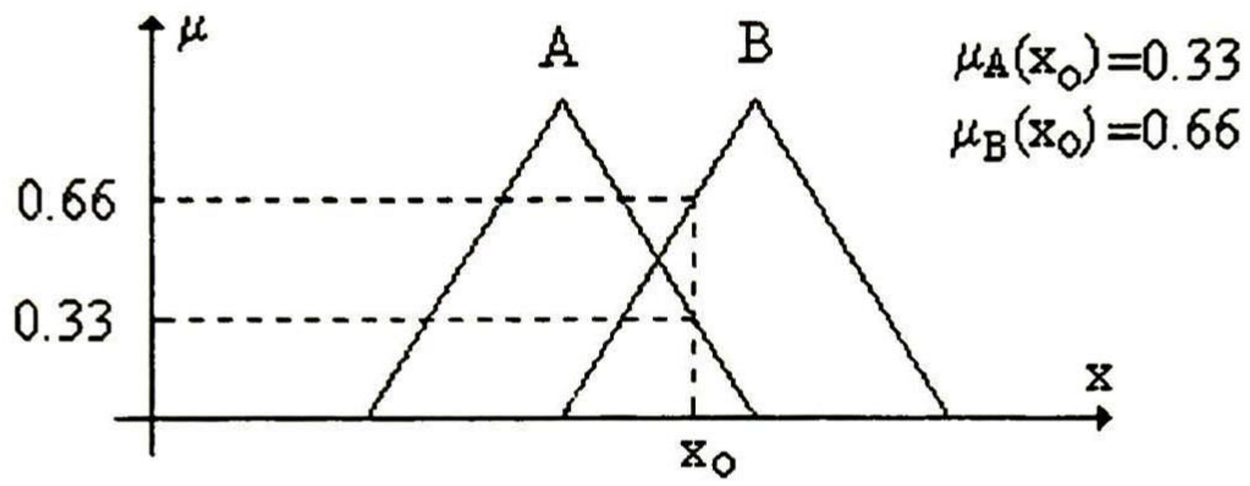


Figura 2.20 Ejemplo de difusificación

<u>Desdifusificación:</u> a partir de una conclusión difusa se obtiene un valor numérico exacto

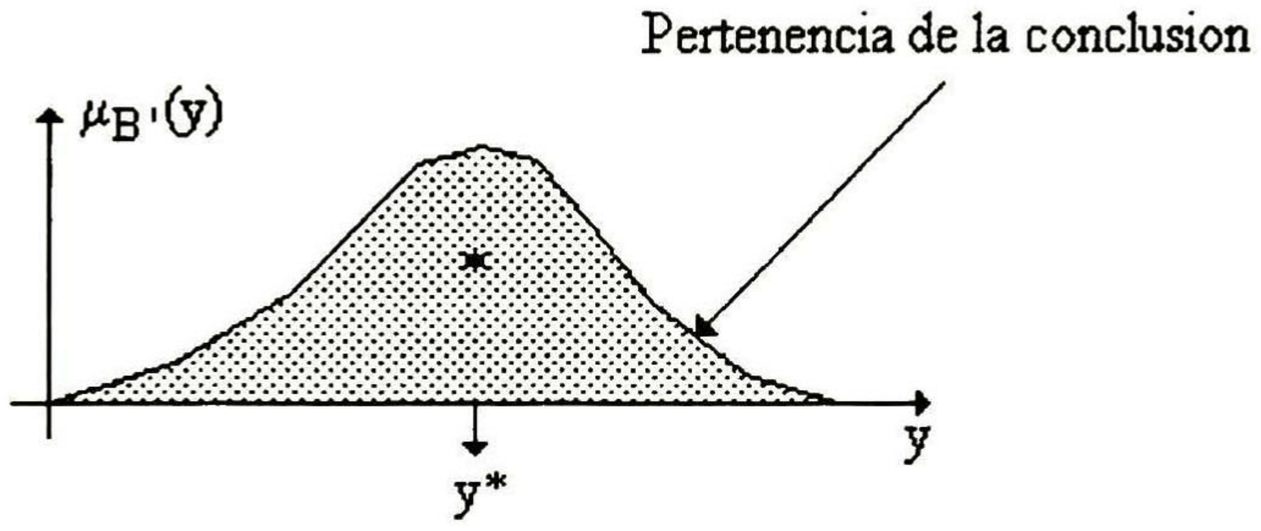


Figura 2.21 Ejemplo de desdifusificación

Para encontrar el valor numérico y* existen diferentes métodos:

1) Principio del máximo de la función de membresía (método de la altura)

$$y^* = \max \mu_{B'}(y) \qquad (2.31)$$

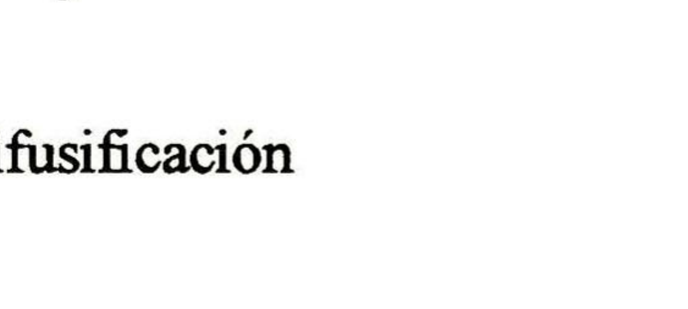


Figura 2.22 Principio del máximo

2) Método del centroide (centro de área o centro de gravedad)

$$y^* = \frac{\int \mu_{B'}(y) \cdot y \cdot dy}{\int \mu_{B'}(y) \cdot dy} \qquad (2.32)$$

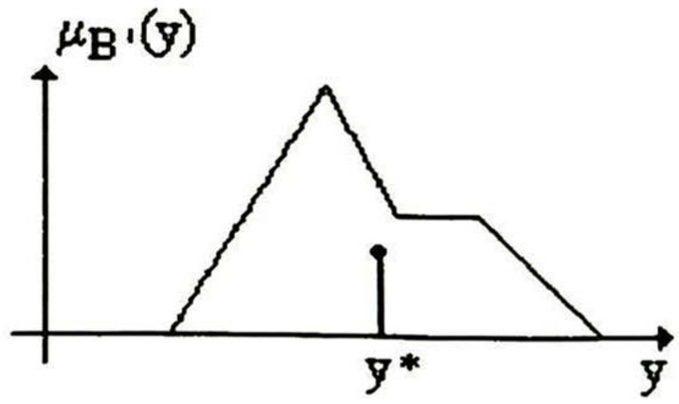


Figura 2.23 Método del centroide

donde el símbolo ∫ significa el operador de integración .

3) Método de pesos promedio (para funciones de membresía simétricas)

$$y^* = \frac{\sum \mu_{B'}(\bar{y}) \cdot \bar{y}}{\sum \mu_{B'}(\bar{y})} \qquad (2.33)$$

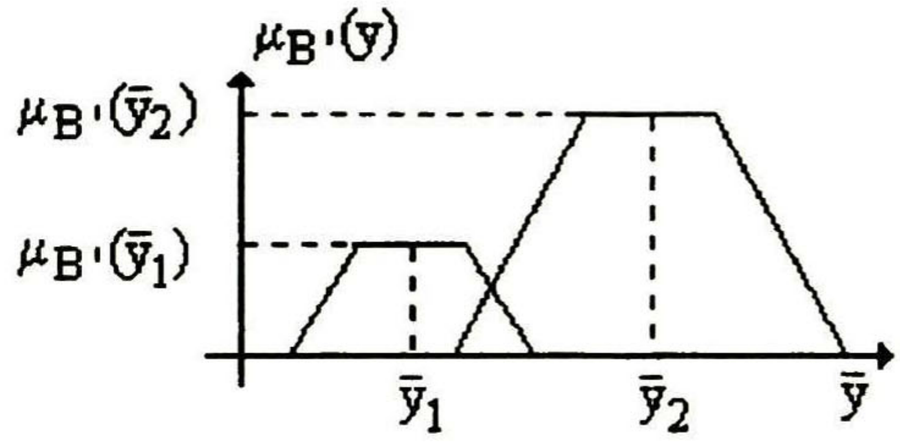


Figura 2.24 Pesos promedio

4) Pertenencia min-max (medio de máximos)

$$y^* = \frac{\bar{y}_1 + \bar{y}_2}{2} \qquad (2.34)$$

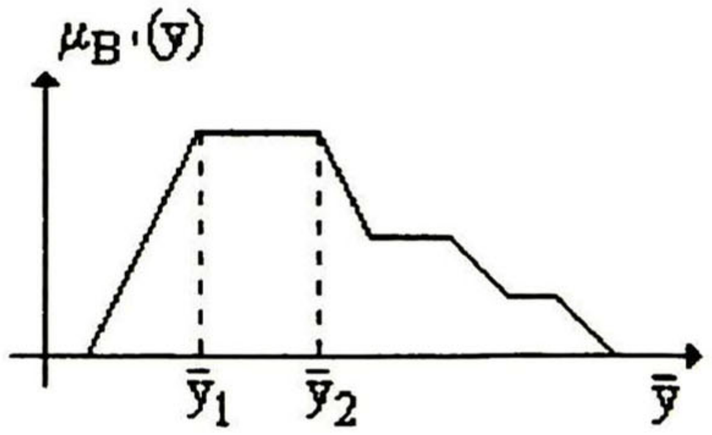


Figura 2.25 Pertenencia Min-Max

5) Centro de sumas: similar al método 3), pero aquí los pesos no son $\mu_{B'}(\bar{y})$sino el área bajo las funciones de pertenencia correspondientes

6) Centro del área más grande: sólo en el conjunto difuso más activo es considerado y y^* es el centro de gravedad del área correspondiente

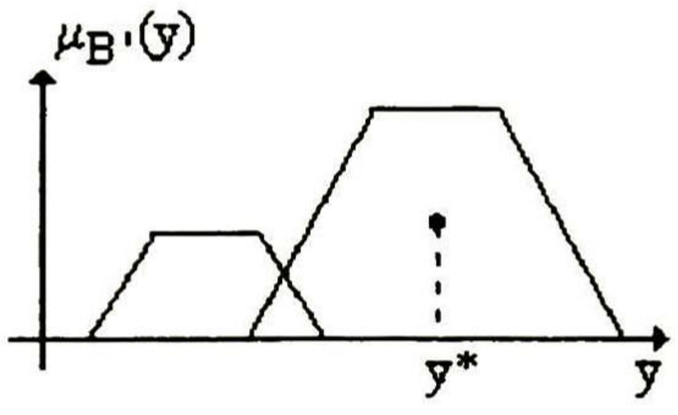


Figura 2.26 Centro del área

7) Primer (o ultimo) máximo

$$y^* = \inf_{y \in Y} \{y \in Y / \mu_{B'}(y) = h(Y)\}$$

o (2.35)

$$y^* = \sup_{y \in Y} \{y \in Y / \mu_{B'}(y) = h(Y)\}$$

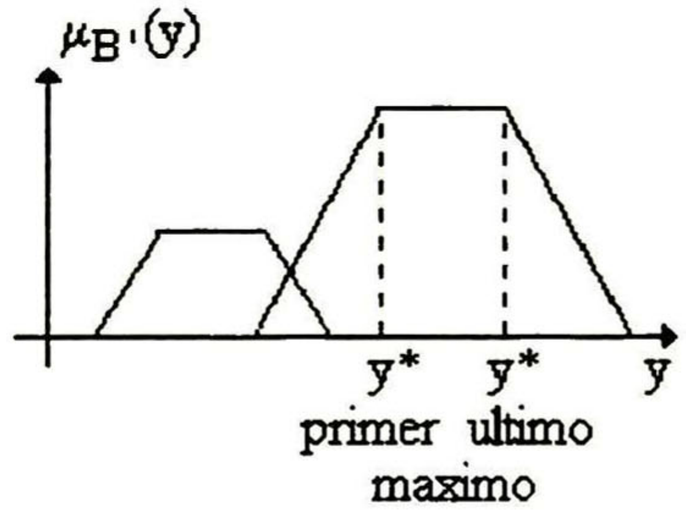


Figura 2.27 Primer o ultimo máximo

y* entonces es el valor menor (o mayor) que maximiza $\mu_{B'}(y)$

Los métodos más usados en el control difuso son el método del centroide, el método de pesos promedio y el centro de sumas. Los otros métodos pueden introducir discontinuidades en la variable de control.

2.3 Algoritmo de control difuso básico

En esta tesis se utiliza el controlador difuso que se describe a continuación. La estructura del controlador PD difuso [5], en tiempo continuo es la siguiente

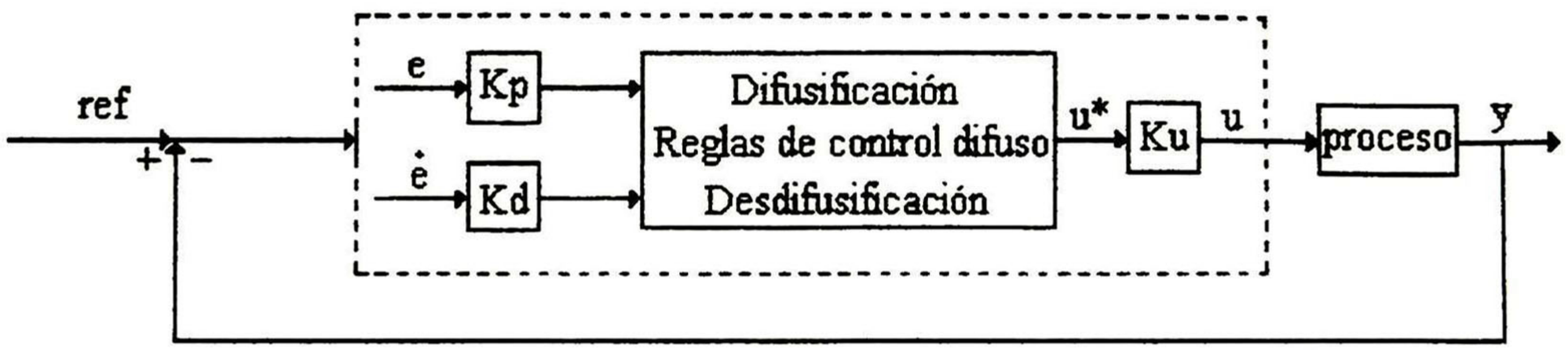


Figura 2.28 Esquema de control PD difuso básico

con:

$$u(t) = K_u u^*(t) \tag{2.36}$$

donde K_u es una ganancia determinada por el diseñador y $u^*(t)$ es la salida del mecanismo consistente de las reglas de control difuso y desdifusificación.

1) Difusificación:

Como se muestra en la Figura 2.28 el controlador difuso emplea dos entradas: la señal de error $e(t)$ y la derivada del error $\dot{e}(t)$, donde e ésta definida como: $e(t) = ref(t)-y(t)$, con ref la señal de referencia a seguir y y la salida del sistema. El controlador difuso sólo tiene una salida $u(t)$ la cual es empleada para alimentar el proceso. La función de pertenencia de las entradas y de la salida del controlador difuso se presentan en las Figuras 2.29 y 2.30 respectivamente. La misma función es aplicada para $e(t)$ y $\dot{e}(t)$por simplicidad .

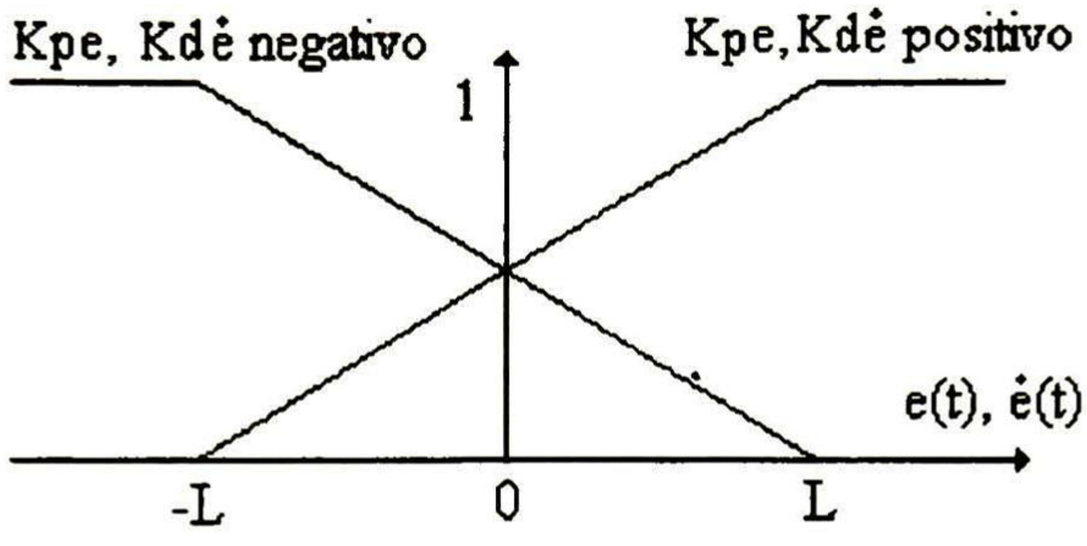


Figura 2.29 Conjunto difuso de entrada

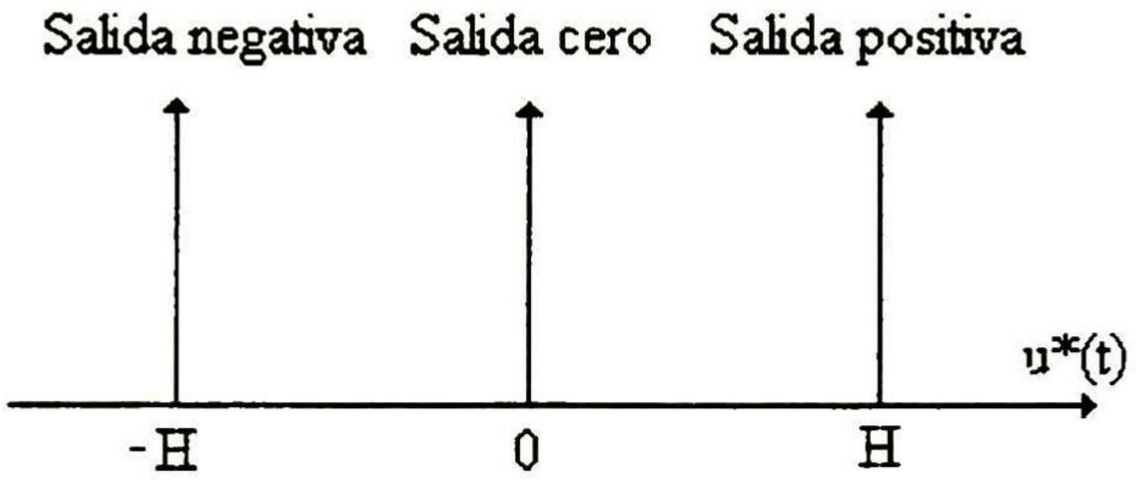


Figura 2.30 Conjunto difuso de salida

Para emplear la misma función de pertenencia en las dos entradas, dos factores de escalamiento, K_p y K_d, son usados para $e(t)$ y $\dot{e}(t)$. En otras palabras $K_p e(t)$ y $K_d \dot{e}(t)$serán utilizados para remplazar $e(t)$ y $\dot{e}(t)$ en la formulación final. En las figuras anteriores L y H son dos constantes positivas a ser determinadas; por conveniencia tomaremos $H = L$ para reducir un parámetro de control a determinar.

2) Reglas de control difuso:

Basadas en las funciones de pertenencia descritas se establecen las reglas siguientes:

$$\begin{aligned}
&\textbf{(R1)}\, Si\ e = ep \wedge \dot{e} = rp \Rightarrow salida = op \\
&\textbf{(R2)}\, Si\ e = ep \wedge \dot{e} = rn \Rightarrow salida = oz \\
&\textbf{(R3)}\, Si\ e = en \wedge \dot{e} = rp \Rightarrow salida = oz \\
&\textbf{(R4)}\, Si\ e = en \wedge \dot{e} = rn \Rightarrow salida = on
\end{aligned} \tag{2.37}$$

En estas reglas, "*ep*" significa "*error positivo*", "*en*" es "*error negativo*", "*rp*" significa "*derivada del error positivo*", "*rn*" es "*derivada del error negativo*", "*op*" significa "*salida positiva*", "*on*" es "*salida negativa*" y "*oz*" significa "*salida cero*". Por ejemplo la primer regla es interpretada como: Si el error es positivo y la derivada del error positivo entonces la salida es positiva. De esta forma podrían interpretarse el significado de las demás reglas. La operación $\wedge$ es la "*Y*" lógica de Zadeh definida como:

$$\mu_A \wedge \mu_B = min\{\mu_A, \mu_B\} \tag{2.38}$$

para dos valores de membresía μ_A y μ_B en los conjuntos difusos A y B respectivamente.

El significado de las cuatro reglas puede ser explicado como sigue [5,10]. El problema de regulación, para una referencia constante, será utilizado con propósito de explicación. En la primer regla (R1), la condición *ep* implica que la salida del sistema y se encuentra por debajo de la referencia y *rp* significa que $\dot{y} < r\dot{e}f = 0$, esto es la velocidad del sistema está disminuyendo. En este caso, la acción de control debe de ser positiva, para incrementar la salida del sistema. Para la tercer regla (R3), la salida del sistema se encuentra por encima de la referencia y disminuyendo, entonces no se necesita ninguna acción de control. Las segunda y cuarta regla pueden ser interpretada en una forma similar.

3) Desdifusificación

Las funciones de pertenencia mostradas en la Figura 2.29 tanto para el error *e(t)* y su derivada $\dot{e}(t)$, vienen dadas por

$$\mu_{ep} = \frac{L + K_p e(t)}{2L}, \quad \mu_{en} = \frac{L - K_p e(t)}{2L}$$
$$\mu_{rp} = \frac{L + K_d \dot{e}(t)}{2L}, \quad \mu_{rn} = \frac{L - K_d \dot{e}(t)}{2L} \tag{2.39}$$

Para el controlador PD difuso propuesto, los rangos de los valores de las entradas ($e(t)$ y $\dot{e}(t)$) son descompuestos en 20 regiones adyacentes como se muestra en la Figura 2.31

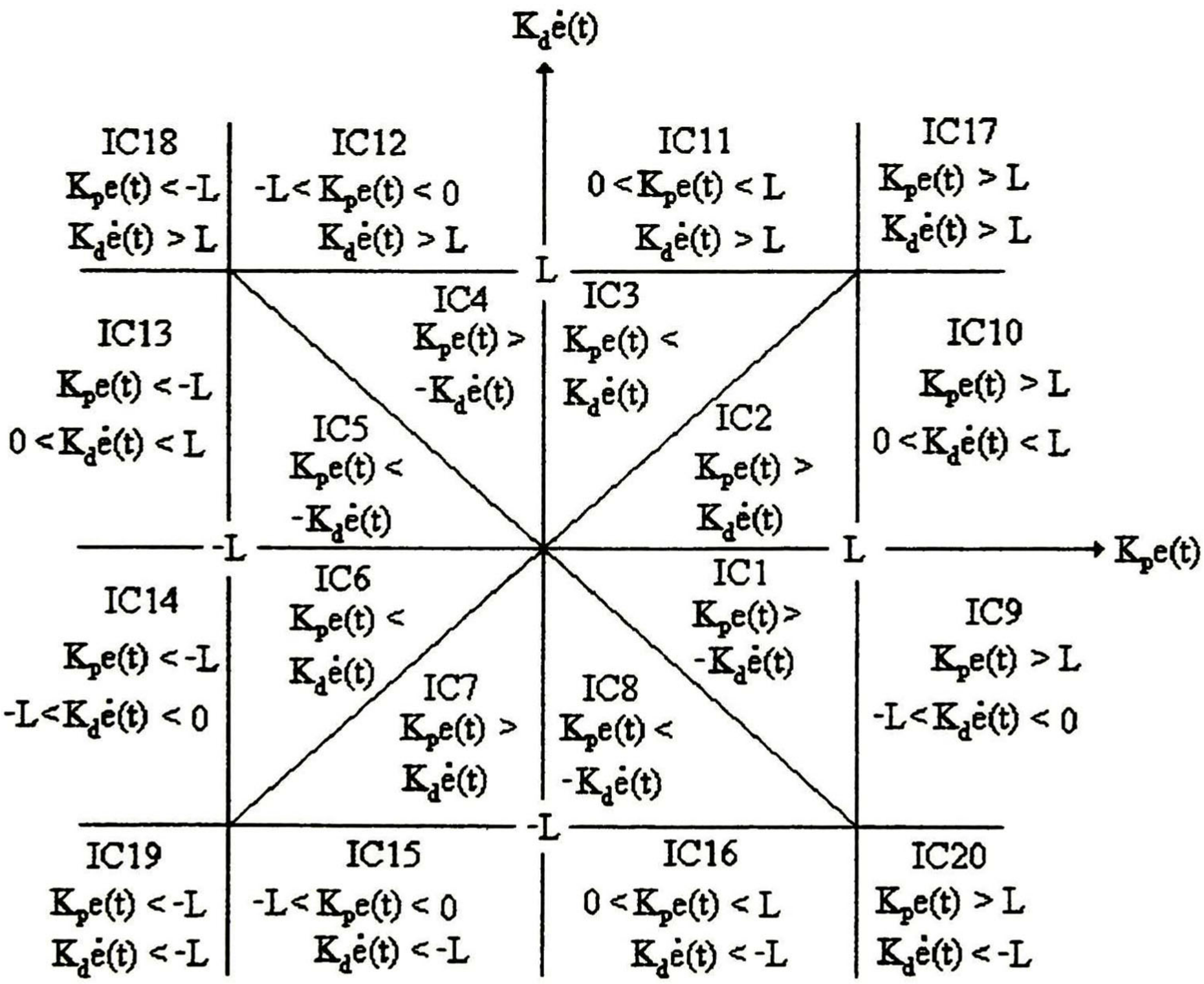


Figura 2.31 Regiones de combinación de entradas

Las reglas de control (R1-R4), las funciones de pertenencia y las regiones de combinación de entradas son usadas para evaluar una ley de control difusa para cada región [10]. Tomando la región IC1; para esta región se tiene que $K_p e>0$, $K_d r<0$, $K_p e>-K_d r$ y $K_p|e| > K_d|r|$ donde $r = \dot{e}$.

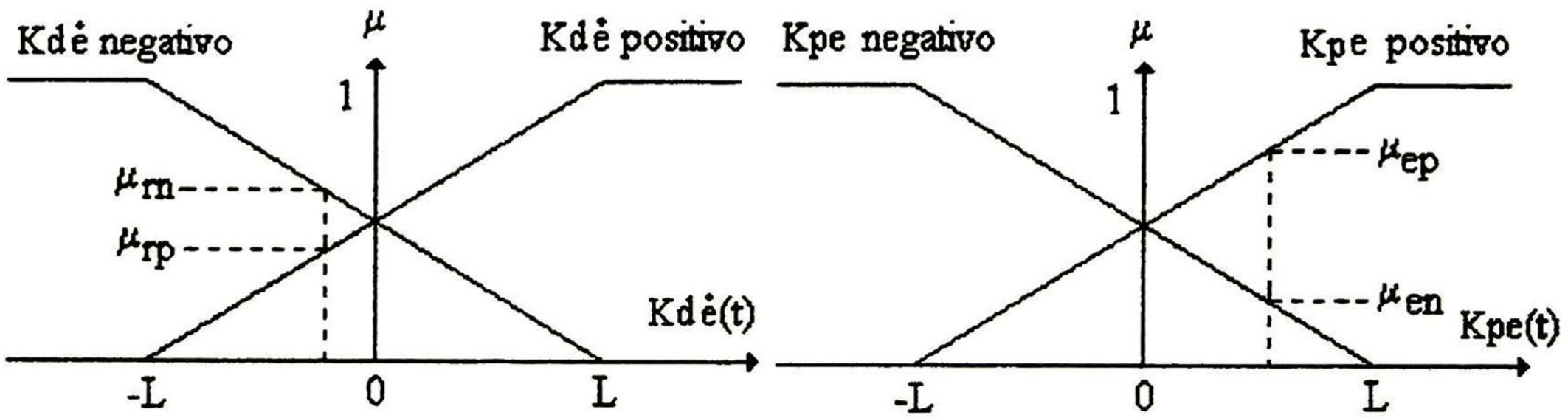


Figura 2.32 Evaluación de conjuntos difusos por IC1

Con las suposiciones anteriores, se evalúan las reglas para esta región, obteniendo:

$$\begin{aligned} &\text{R1: } \min(\mu_{ep}, \mu_{rp}) = \mu_{rp} \\ &\text{R2: } \min(\mu_{ep}, \mu_{rn}) = \mu_{rn} \\ &\text{R3: } \min(\mu_{en}, \mu_{rp}) = \mu_{en} \\ &\text{R4: } \min(\mu_{en}, \mu_{rn}) = \mu_{en} \end{aligned} \tag{2.40}$$

Para la región IC2 se derivan las mismas ecuaciones (2.40). Ahora, para la región IC5 se tiene que $K_p e<0$, $K_d r>0$, $K_p e>-K_d r$ y $K_p|e| > K_d|r|$

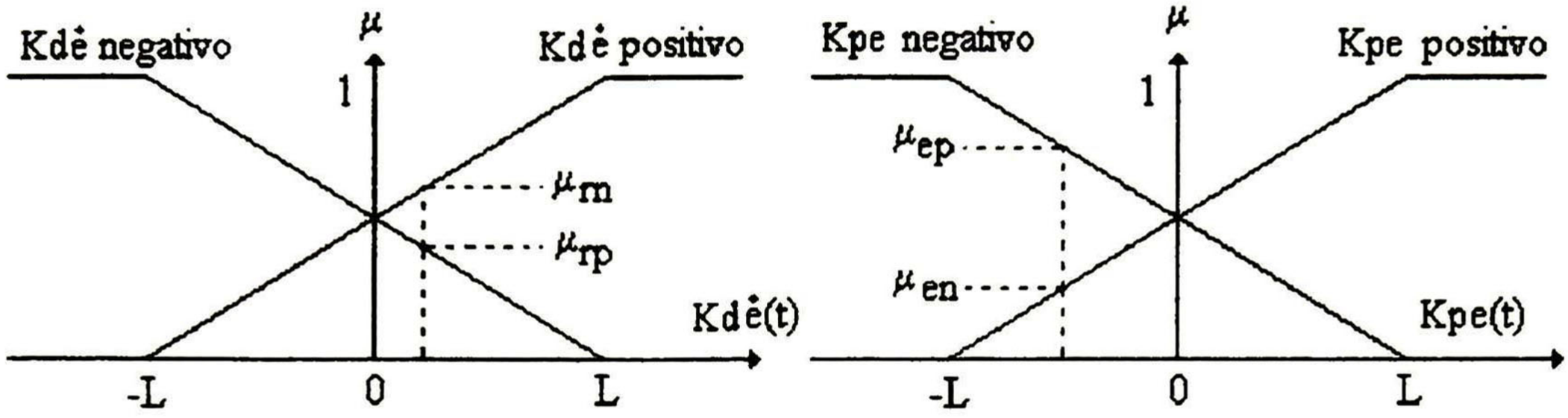


Figura 2.33 Evaluación de los conjuntos difusos por IC5

Para este caso se tiene la siguiente evaluación de las reglas:

$$\begin{aligned} &\text{R1: } \min(\mu_{ep}, \mu_{rp}) = \mu_{ep} \\ &\text{R2: } \min(\mu_{ep}, \mu_{rn}) = \mu_{ep} \\ &\text{R3: } \min(\mu_{en}, \mu_{rp}) = \mu_{rp} \\ &\text{R4: } \min(\mu_{en}, \mu_{rn}) = \mu_{rn} \end{aligned} \tag{2.41}$$

Este análisis también es válido para la región IC6. Ahora se continúa el análisis para cada región, utilizando la inferencia de Mamdani (Tabla 1.1) con ***a*** la función de pertenencia del antecedente y ***b*** la función de pertenencia de la acción de control. Como las reglas R2 y R3 generan la misma salida se hace una unión de reglas; esto se lleva a cabo con el operador de Lukasiewicz (Tabla A.1 Apéndice A). Finalmente se procede a la desdifusificación utilizando el método del centro de gravedad:

$$u = \frac{-H \cdot S(\mu_{R4}) + 0 \cdot S(\mu_{R2+R3}) + H \cdot S(\mu_{R1})}{S(\mu_{R4}) + S(\mu_{R2+R3}) + S(\mu_{R1})} \tag{2.42}$$

donde S(·) indica la superficie respectiva:

$$u = \frac{H \cdot (S(\mu_{R1}) - \cdot S(\mu_{R4}))}{S(\mu_{R4}) + S(\mu_{R2+R3}) + S(\mu_{R1})} \tag{2.43}$$

Puesto que IC1 e IC2 disparan las mismas reglas, el cálculo de la desdifusificación es el mismo:

$$u = \frac{H \cdot (S(\mu_{rp}) - \cdot S(\mu_{en}))}{S(\mu_{en}) + S(\mu_{rn+en}) + S(\mu_{rp})} \tag{2.44}$$

$$u = H \frac{\left(\frac{L + K_d r(t)}{2L}\right) - \left(\frac{L - K_p e(t)}{2L}\right)}{\left(\frac{L - K_p e(t)}{2L}\right) + \left(\frac{L - K_d r(t)}{2L} + \frac{L - K_p e(t)}{2L}\right) + \left(\frac{L + K_d r(t)}{2L}\right)} \tag{2.45}$$

$$u = \frac{H}{2(2L - K_p e(t))}\left(K_p e(t) + K_d r(t)\right) \tag{2.46}$$

en IC1 e IC2 $e(t) > 0$.

Calculando para IC5 e IC6 con la misma metodología, se obtiene:

$$u = \frac{H \cdot (S(\mu_{ep}) - \cdot S(\mu_m))}{S(\mu_m) + S(\mu_{ep+rp}) + S(\mu_{ep})} \tag{2.47}$$

$$u = H \frac{\left(\frac{L + K_p e(t)}{2L}\right) - \left(\frac{L - K_d r(t)}{2L}\right)}{\left(\frac{L - K_d r(t)}{2L}\right) + \left(\frac{L + K_p e(t)}{2L} + \frac{L + K_d r(t)}{2L}\right) + \left(\frac{L + K_p e(t)}{2L}\right)} \tag{2.48}$$

$$u = \frac{H}{2(2L + K_p e(t))}\left(K_p e(t) + K_d r(t)\right) \tag{2.49}$$

con $e(t) < 0$ en las regiones IC5 e IC6. Por lo que podemos sintetizar una acción de control tomando las ecuaciones (2.46) y (2.49), que sea válido para las regiones IC1, IC2, IC5 e IC6; esta es:

$$u = \frac{H}{2(2L - K_p|e(t)|)}\left(K_p e(t) + K_d r(t)\right) \tag{2.50}$$

Tomando la región IC3 para esta región $K_p e>0$, $K_d r>0$, $K_p e<K_d r$ y $K_p|e| < K_d r$ donde $r = \dot{e}$.

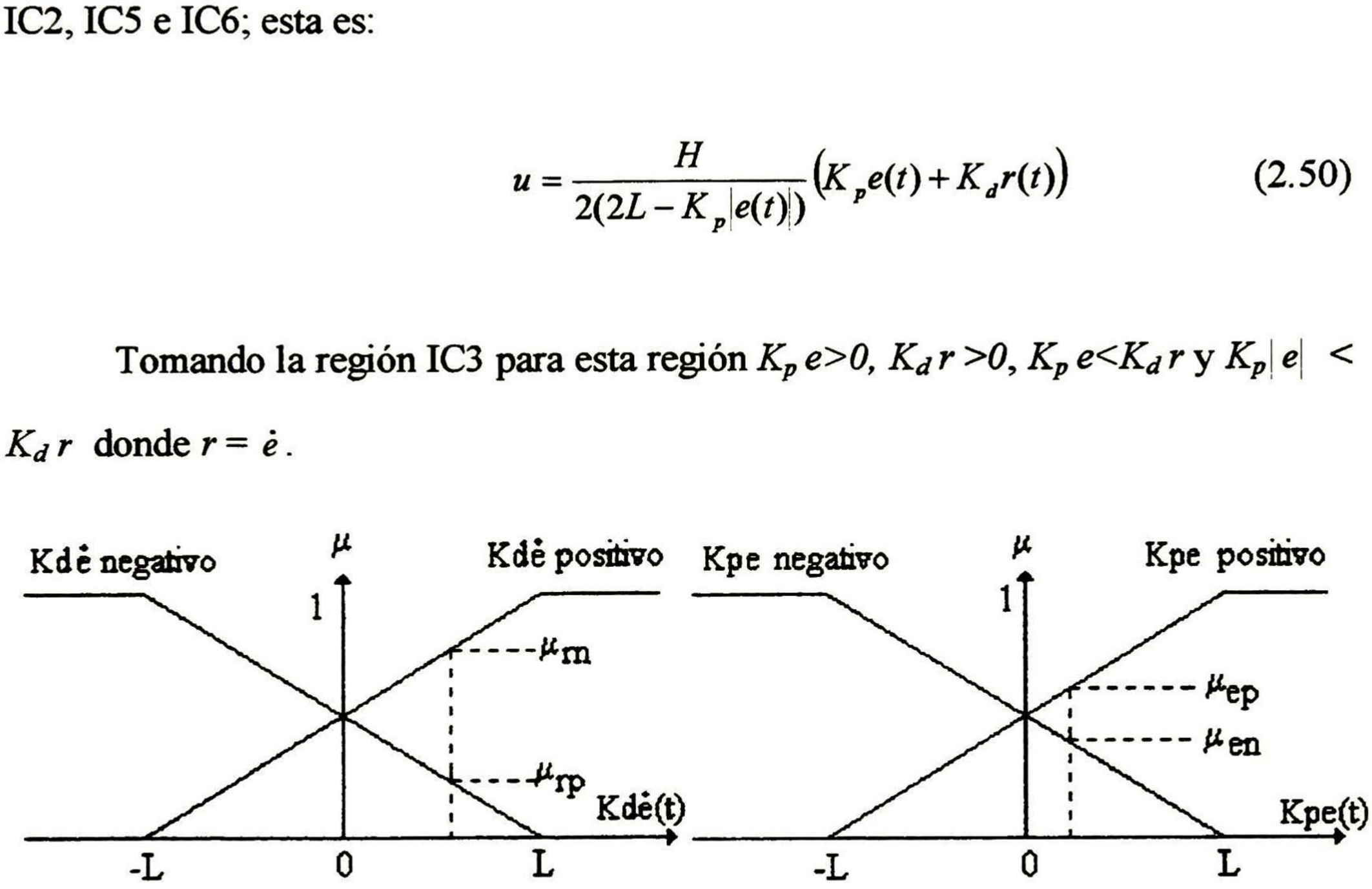


Figura 2.34 Evaluación de conjuntos difusos por IC3

Con las suposiciones anteriores, se evalúan las reglas para esta región en particular obteniendo:

$$\begin{aligned}
&\text{R1: } \min(\mu_{e_p}, \mu_{r_p}) = \mu_{e_p} \\
&\text{R2: } \min(\mu_{e_p}, \mu_{r_n}) = \mu_{r_n} \\
&\text{R3: } \min(\mu_{e_n}, \mu_{r_p}) = \mu_{e_n} \\
&\text{R4: } \min(\mu_{e_n}, \mu_{r_n}) = \mu_{r_n}
\end{aligned} \tag{2.51}$$

Para la región IC4 se obtienen las mismas ecuaciones (2.51). Ahora en la región IC8 se tiene que $K_p e<0$, $K_d r<0$, $-K_p e<-K_d r$ y $K_p|e| < K_d|r|$

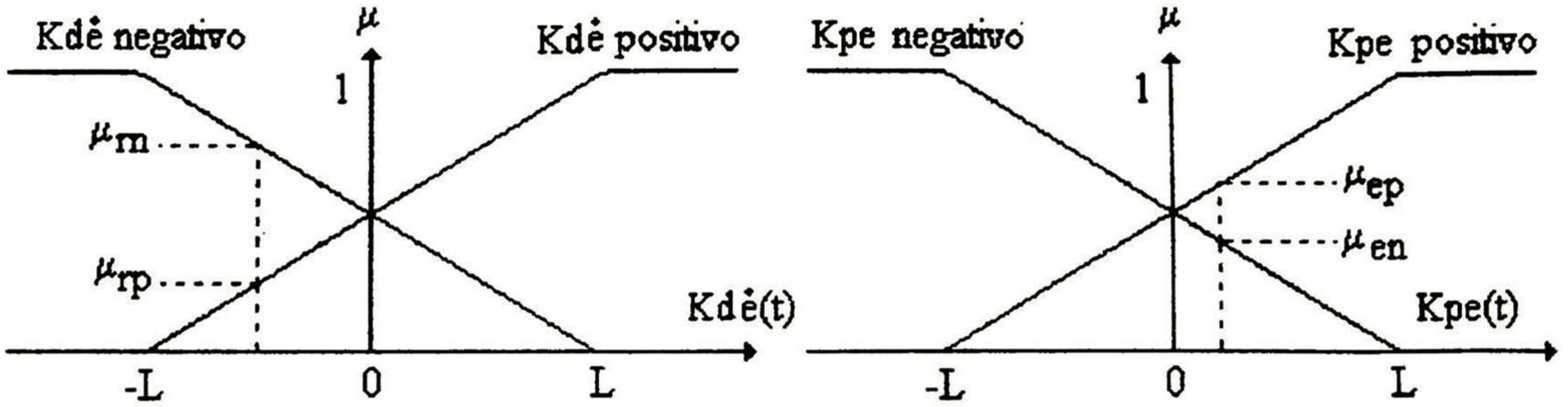


Figura 2.35 Evaluación de los conjuntos difusos por IC8

Para este caso, se obtiene la siguiente evaluación de las reglas:

$$\begin{aligned} &\text{R1: } \min(\mu_{e_p}, \mu_{r_p}) = \mu_{r_p} \\ &\text{R2: } \min(\mu_{e_p}, \mu_{r_n}) = \mu_{e_p} \\ &\text{R3: } \min(\mu_{e_n}, \mu_{r_p}) = \mu_{r_p} \\ &\text{R4: } \min(\mu_{e_n}, \mu_{r_n}) = \mu_{e_n} \end{aligned} \tag{2.52}$$

Este análisis es valido para la región IC7. Ahora se continúa el análisis para cada región, con el criterio de la inferencia de Mamdani (Tabla 1.1) donde ***a*** es la función de pertenencia del antecedente y ***b*** es la función de pertenencia de la acción de control. Como las reglas R2 y R3 generan la misma salida se hace una unión de reglas; esto se lleva a cabo con el operador de Lukasiewicz (Tabla A.1 Apéndice A) finalmente se procede a la desdifusificación utilizando el método del centro de gravedad (2.42) y (2.43):

Puesto que IC3 e IC4 disparan las mismas reglas, el cálculo de la desdifusificación es el mismo:

$$u = \frac{H \cdot (S(\mu_{ep}) - \cdot S(\mu_m))}{S(\mu_{ep}) + S(\mu_{rn+en}) + S(\mu_m)} \tag{2.53}$$

$$u = H \frac{\left(\frac{L + K_p e(t)}{2L}\right) - \left(\frac{L - K_d r(t)}{2L}\right)}{\left(\frac{L + K_p e(t)}{2L}\right) + \left(\frac{L - K_d r(t)}{2L} + \frac{L - K_p e(t)}{2L}\right) + \left(\frac{L - K_d r(t)}{2L}\right)} \tag{2.54}$$

$$u = \frac{H}{2(2L - K_d r(t))}\left(K_p e(t) + K_d r(t)\right) \tag{2.55}$$

en IC3 e IC4 $r(t) > 0$.

Calculando para IC7 e IC8 con la misma metodología, se obtiene:

$$u = \frac{H \cdot (S(\mu_{rp}) - \cdot S(\mu_{en}))}{S(\mu_{rp}) + S(\mu_{ep+rp}) + S(\mu_{en})} \tag{2.56}$$

$$u = H \frac{\left(\frac{L + K_d r(t)}{2L}\right) - \left(\frac{L - K_p e(t)}{2L}\right)}{\left(\frac{L + K_d r(t)}{2L}\right) + \left(\frac{L + K_p e(t)}{2L} + \frac{L + K_d r(t)}{2L}\right) + \left(\frac{L - K_p e(t)}{2L}\right)} \tag{2.57}$$

$$u = \frac{H}{2(2L + K_d r(t))}\left(K_p e(t) + K_d r(t)\right) \tag{2.58}$$

con $r(t) < 0$ en las regiones IC7 e IC8. Por lo que podemos sintetizar una acción de control tomando las ecuaciones (2.55) y (2.58) que es valida para las regiones IC3, IC4, IC7 e IC8:

$$u = \frac{H}{2(2L - K_d|r(t)|)}\left(K_p e(t) + K_d r(t)\right) \tag{2.59}$$

En la región IC9, se tiene que $K_p e > L$, $-L < K_d r < 0$

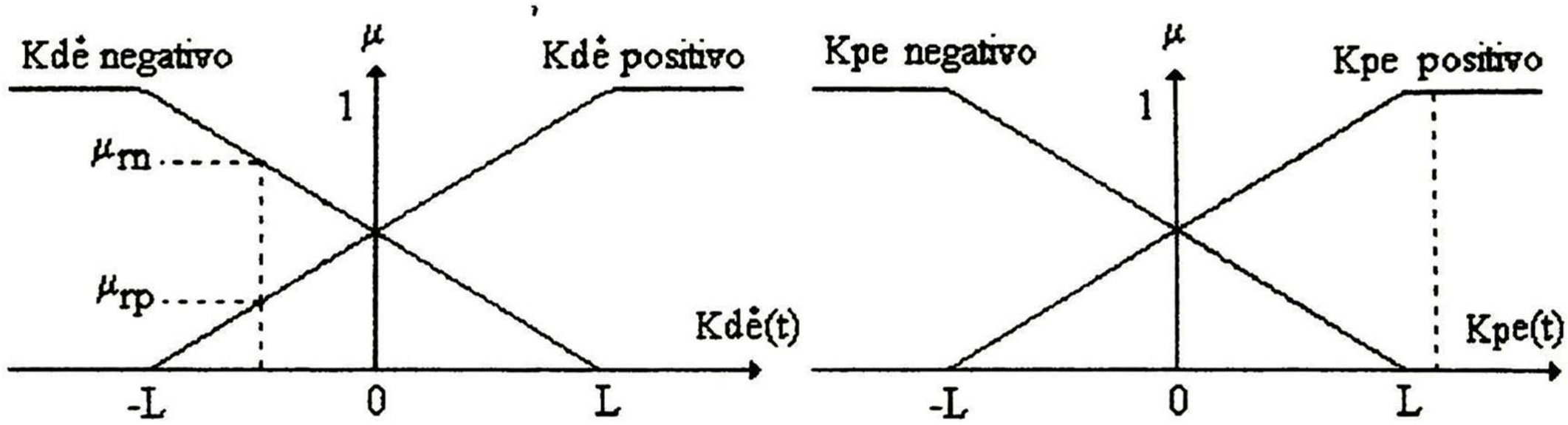


Figura 2.36 Evaluación de los conjuntos difusos por IC9

Para este caso se obtiene la siguiente evaluación de las reglas:

$$\text{R1: } \min(\mu_{ep}, \mu_{rp}) = \mu_{rp}$$

$$\text{R2: } \min(\mu_{ep}, \mu_{rn}) = \mu_{rn} \tag{2.60}$$

Este mismo análisis es valido para la región IC10. Puesto que IC9 e IC10 disparan las mismas reglas, el cálculo de la desdifusificación es el mismo por lo que:

$$u = \frac{H \cdot S(\mu_{R1})}{S(\mu_{R1}) + S(\mu_{R2})} \tag{2.61}$$

$$u = \frac{H \cdot S(\mu_{rp})}{S(\mu_{ep}) + S(\mu_{m})} \tag{2.62}$$

$$u = H\frac{\left(\frac{L+K_d r(t)}{2L}\right)}{\left(\frac{L+K_d r(t)}{2L}\right)+\left(\frac{L-K_d r(t)}{2L}\right)} \tag{2.63}$$

$$u = \frac{H}{2L}\left(L+K_d r(t)\right) \tag{2.64}$$

Para la región IC11, se tiene que $K_d r > L$, $-L < K_p e < 0$

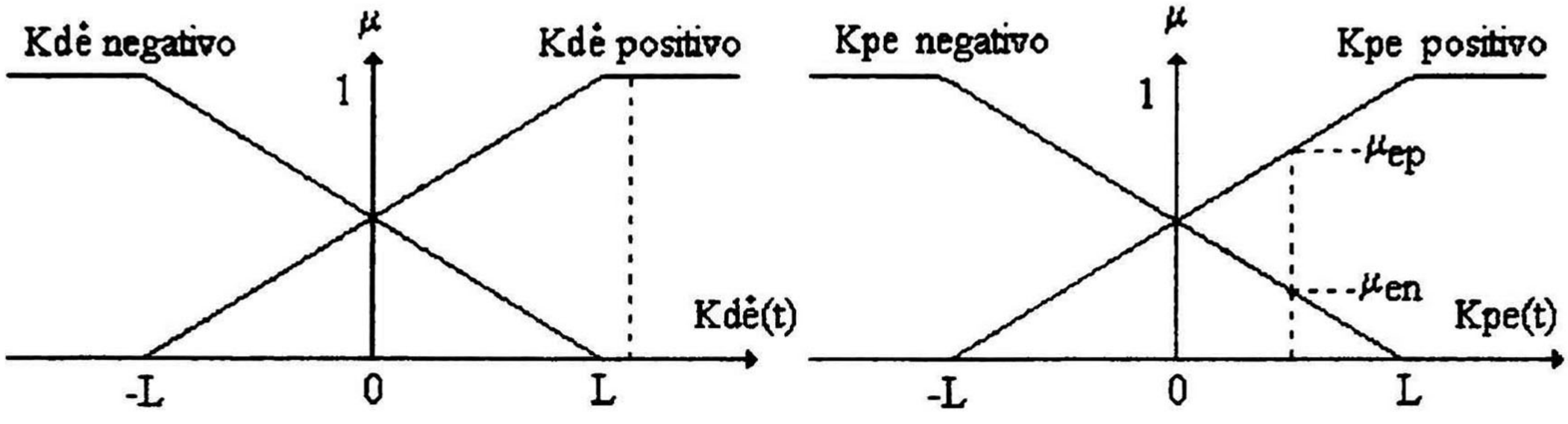


Figura 2.37 Evaluación de los conjuntos difusos por IC11

Las reglas entonces se evalúan como:

$$\begin{aligned} &\text{R1: } \min(\mu_{ep}, \mu_{rp}) = \mu_{ep} \\ &\text{R3: } \min(\mu_{en}, \mu_{rp}) = \mu_{en} \end{aligned} \tag{2.65}$$

Lo mismo se aplica a la región IC12. Puesto que IC11 e IC12 disparan las mismas reglas, el cálculo de la desdifusificación es el mismo por lo que:

$$u = \frac{H \cdot S(\mu_{R1})}{S(\mu_{R1}) + S(\mu_{R3})} \tag{2.66}$$

$$u = \frac{H \cdot S(\mu_{ep})}{S(\mu_{ep}) + S(\mu_{en})} \tag{2.67}$$

$$u = H\frac{\left(\frac{L+K_p e(t)}{2L}\right)}{\left(\frac{L+K_p e(t)}{2L}\right)+\left(\frac{L-K_p e(t)}{2L}\right)} \tag{2.68}$$

$$u = \frac{H}{2L}\left(L+K_p e(t)\right) \tag{2.69}$$

En la región IC13, se tiene que $K_p e<-L$, $0<K_d r<L$, por lo que:

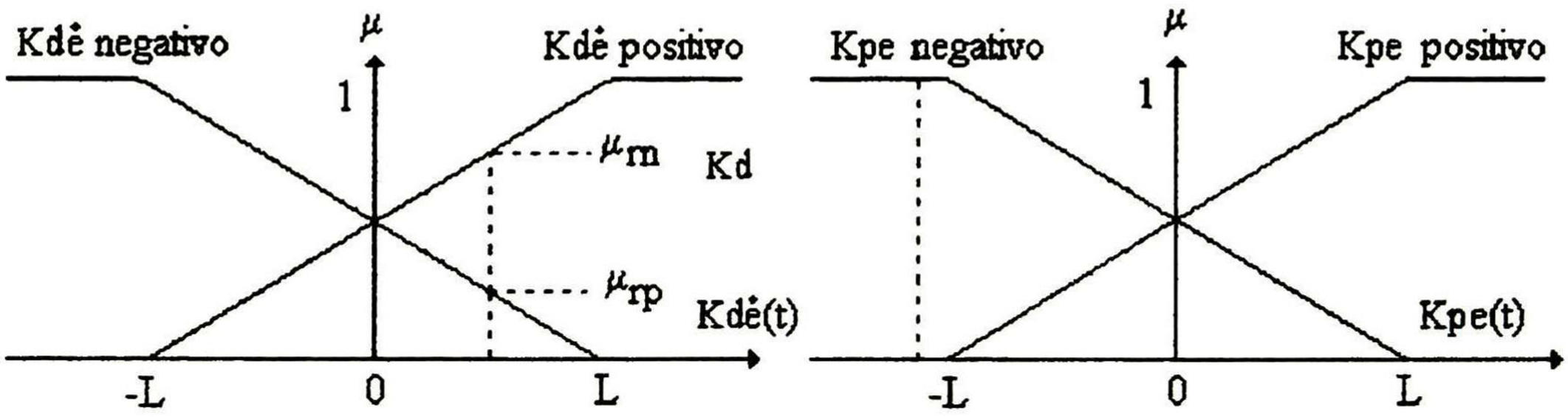


Figura 2.38 Evaluación de los conjuntos difusos por IC13

$$\text{R3: } \min(\mu_{ep}, \mu_{rp}) = \mu_{rp}$$

$$\text{R4: } \min(\mu_{ep}, \mu_{rn}) = \mu_{rn} \tag{2.70}$$

Esto también se aplica en la región IC14. Puesto que IC13 e IC14 disparan las mismas reglas, el cálculo de la desdifusificación es el mismo, entonces:

$$u = \frac{-H \cdot S(\mu_{R4})}{S(\mu_{R3})+S(\mu_{R4})} \tag{2.71}$$

$$u = \frac{-H \cdot S(\mu_m)}{S(\mu_{rp})+S(\mu_m)} \tag{2.72}$$

$$u = -H \frac{\left(\frac{L - K_d r(t)}{2L}\right)}{\left(\frac{L + K_d r(t)}{2L}\right) + \left(\frac{L - K_d r(t)}{2L}\right)} \tag{2.73}$$

$$u = -\frac{H}{2L}\left(L - K_d r(t)\right) \tag{2.74}$$

En la región IC15, se tiene que $K_d r < -L$, $-L < K_p e < 0$, así que:

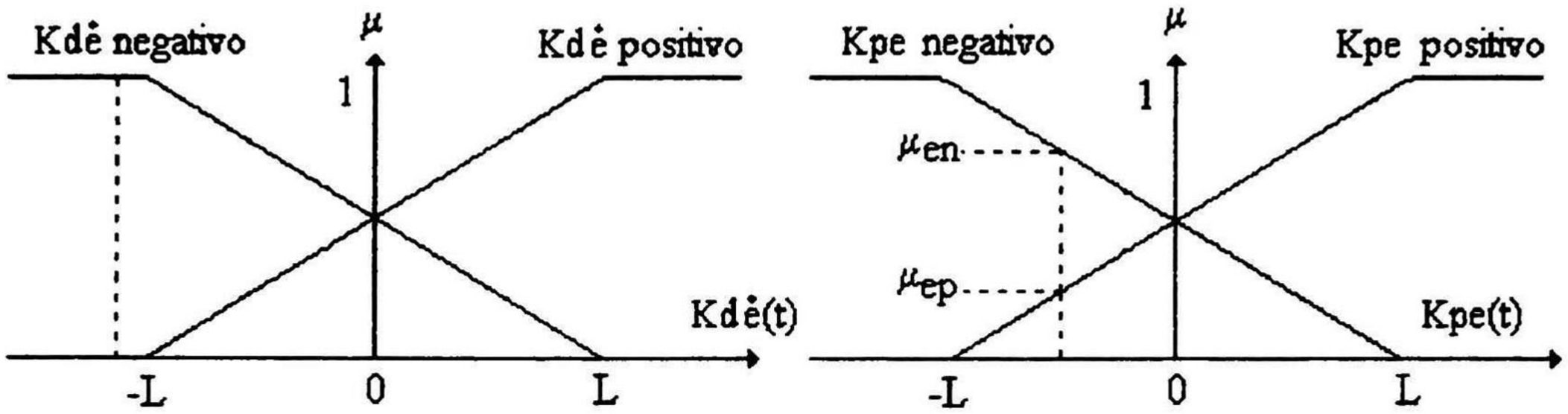


Figura 2.39 Evaluación de los conjuntos difusos por IC15

$$\text{R2: } \min(\mu_{ep}, \mu_{rn}) = \mu_{ep}$$

$$\text{R4: } \min(\mu_{en}, \mu_{rn}) = \mu_{en} \tag{2.75}$$

Esto también es cierto en IC16. Puesto que IC15 e IC16 disparan las mismas reglas, el cálculo de la desdifusificación es el mismo:

$$u = \frac{-H \cdot S(\mu_{R4})}{S(\mu_{R2}) + S(\mu_{R4})} \tag{2.76}$$

$$u = \frac{-H \cdot S(\mu_{en})}{S(\mu_{ep}) + S(\mu_{en})} \tag{2.77}$$

$$u = -H \frac{\left(\frac{L - K_p e(t)}{2L}\right)}{\left(\frac{L - K_p e(t)}{2L}\right) + \left(\frac{L + K_p e(t)}{2L}\right)} \tag{2.78}$$

$$u = -\frac{H}{2L}\left(L + K_p e(t)\right) \tag{2.79}$$

En la región IC17, se sabe que $K_d r > L$, $K_p e > L$

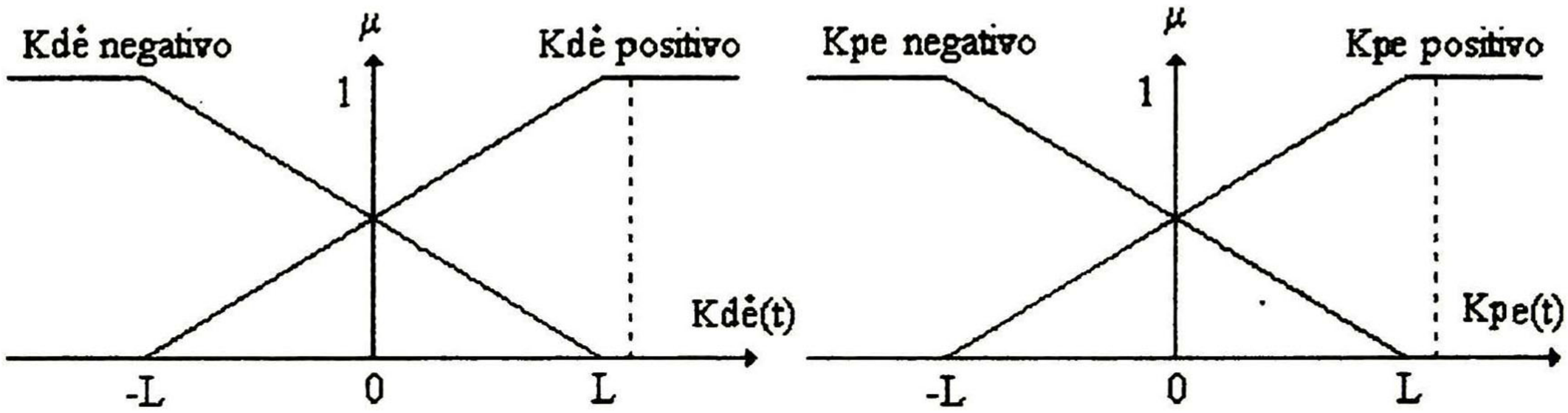


Figura 2.40 Evaluación de los conjuntos difusos por IC17

Para este caso se obtiene la siguiente evaluación de la regla:

$$\text{R1: } \min(\mu_{ep}, \mu_{rp}) = 1 \tag{2.80}$$

Haciendo el cálculo de la desdifusificación, utilizando el método del centro de gravedad (2.42):

$$u = \frac{H \cdot S(\mu_{R1})}{S(\mu_{R1})} \tag{2.81}$$

$$u = H \tag{2.82}$$

En la región IC19, se sabe que $K_d r<-L$, $K_p e<-L$

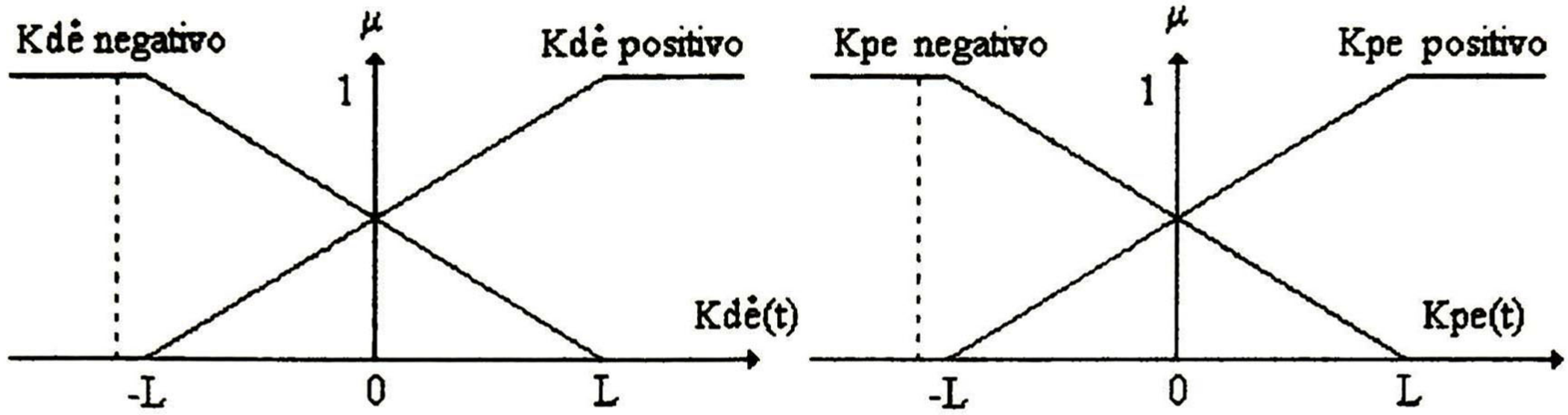


Figura 2.41 Evaluación de los conjuntos difusos por IC19

Para este caso se obtiene la siguiente evaluación de la regla:

$$\text{R4: } \min(\mu_{e_n}, \mu_{r_n}) = 1 \tag{2.83}$$

Realizando la desdifusificación, utilizando el método del centro de gravedad (2.42):

$$u = \frac{-H \cdot S(\mu_{R4})}{S(\mu_{R4})} \tag{2.84}$$

$$u = -H \tag{2.85}$$

En la región IC18, se tiene que $K_d r>L$, $K_p e <-L$

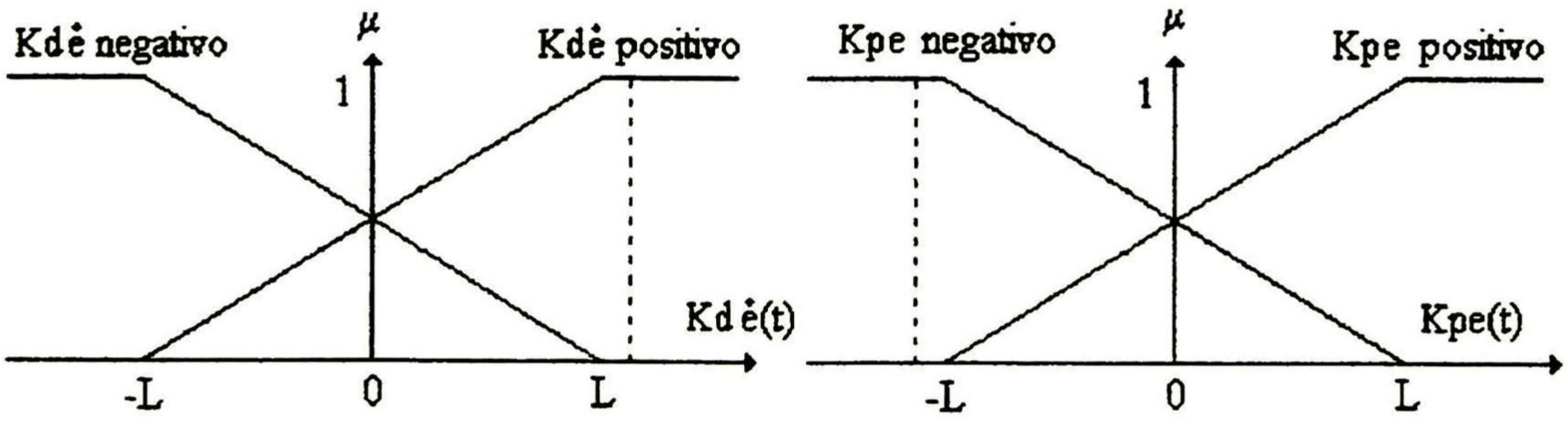


Figura 2.42 Evaluación de los conjuntos difusos por IC18

Para este caso se obtiene la siguiente evaluación de la regla

$$\text{R3: } \min(\mu_{en}, \mu_{rp}) = 1 \tag{2.86}$$

Haciendo el cálculo de la desdifusificación utilizando el método del centro de gravedad (2.42):

$$u = \frac{0 \cdot S(\mu_{R1})}{S(\mu_{R1})} \tag{2.87}$$

$$u = 0 \tag{2.88}$$

El mismo cálculo se aplica para la región IC20.

Tomando las ecuaciones (2.50), (2.59), (2.64), (2.69), (2.74), (2.79), (2.82), (2.85) y (2.88), considerando que $H=L$ se obtienen las siguientes ecuaciones, que determinan la señal de control dependiendo la región:

$$u^*(t) = \frac{L}{2\left(2L - K_p|e(t)|\right)}\left[K_p e(t) + K_d r(t)\right] en IC1, IC2, IC5, IC6.$$

$$u^*(t) = \frac{L}{2\left(2L - K_d|r(t)|\right)}\left[K_p e(t) + K_d r(t)\right] en IC3, IC4, IC7, IC8.$$

$$u^*(t) = \frac{1}{2}\left[L + K_d r(t)\right] en IC9, IC10.$$

$$u^*(t) = \frac{1}{2}\left[L + K_p e(t)\right] en IC11, IC12.$$

$$
\begin{aligned}
u^*(t) &= \frac{1}{2}\left[-L + K_d r(t)\right] en IC13, IC14. \\
u^*(t) &= \frac{1}{2}\left[-L + K_p e(t)\right] en IC15, IC16. \\
u^*(t) &= L \; en \; IC17. \\
u^*(t) &= -L \; en \; IC19. \\
u^*(t) &= 0 \; en \; IC18, IC20.
\end{aligned}
\tag{2.89}
$$

En este capítulo, se han establecido las bases de la lógica difusa, las operaciones de conjuntos difusos, los diferentes tipos de funciones de pertenencia y sus características primordiales, las diferentes implicaciones y desdifusificaciones; y por último se explica el algoritmo de control básico utilizado en esta tesis.

CAPÍTULO 3
CONTROL DE ROBOTS COMPLETAMENTE ACTUADOS

En éste capítulo se presenta el desarrollo de las ecuaciones que describen la dinámica de un robot de dos grados de libertad, se propone un nuevo esquema de control PD difuso para éste robot, se analiza la estabilidad de dicho esquema y se ilustra la aplicación del mismo por medio de simulaciones.

3.1 Modelado del robot

Inicialmente se considera un robot de dos eslabones completamente actuado; es decir el número de actuadores igual al número de eslabones. Para éste robot se desarrollan las ecuaciones diferenciales no lineales invariantes en el tiempo, que determinan el comportamiento dinámico del robot. Estas ecuaciones se obtienen por el método de Euler-Lagrange.

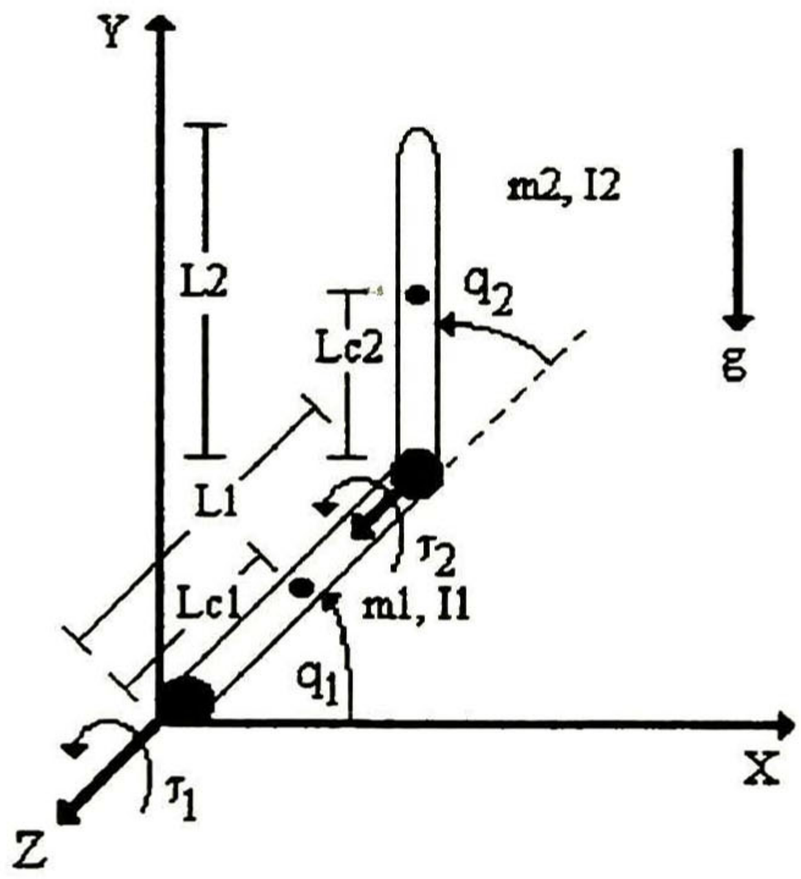


Figura 3.1 Robot de dos eslabones

En la figura 3.1 se muestra un esquema del robot; donde el movimiento angular de los eslabones del robot se determinan por los ángulos q_1, q_2; l_1 y l_2 representan la longitud del primer y segundo eslabón; l_{c1} la ubicación del centro de masa del primer eslabón, l_{c2} la ubicación del centro de masa del segundo eslabón, I_1 el momento de inercia del primer eslabón e I_2 el momento de inercia para el segundo y g la gravedad.

La ecuación de Euler-Lagrange para modelar un robot cómo el descrito [10] viene dada por [11]:

$$\frac{d}{dt}\left\{\frac{\partial L}{\partial \dot{q}}\right\} - \frac{\partial L}{\partial q} = \tau \tag{3.1}$$

donde q y $\dot{q}$ son vectores de dimensión 2, determinados por el número de eslabones específico a éste robot, τ es un vector de fuerzas de dimensión 2 y el Lagrangiano L está definido cómo la diferencia entre la energía cinética K y la energía potencial V:

$$L = K - V \tag{3.2}$$

La ecuación de la energía cinética K viene dada por:

$$K = \frac{1}{2}\left(\sum_{i=1}^{2} m_i v_{ci}^T v_{ci} + \omega_i^T I \omega_i\right), i = 1,2 \tag{3.3}$$

donde v_{ci} y ω_i son los vectores de velocidad lineal y de velocidad angular asociado al i-ésimo eslabón e I es la matriz de inercia.

Para el cálculo de la velocidad lineal, se considera un vector de posición desde el marco de coordenadas base (X, Y, Z en la Figura 3.1) al centro de masa de

cada uno de los eslabones, suponga que los eslabones son totalmente uniformes y sus centros de masas se encuentran concentrados en el centro. El vector de posición del centro de masa del primer eslabón es:

$$r_{c1}(q)=\begin{bmatrix} r_{1x}(q) \\ r_{1y}(q) \end{bmatrix}=\begin{bmatrix} l_{c1}cos(q_1) \\ l_{c1}sen(q_1) \end{bmatrix} \tag{3.4}$$

donde $r_{1x}(q)$ es la proyección del centro de masa del primer eslabón sobre el eje X y $r_{1y}(q)$ es la proyección sobre el eje Y. Derivando la ecuación 3.4 con respecto al tiempo, se obtienen las velocidades lineales de los eslabones:

$$\dot{r}_{c1}(q)=v_{c1}=\begin{bmatrix} -l_{c1}\dot{q}_1 sen(q_1) \\ l_{c1}\dot{q}_1 cos(q_1) \end{bmatrix}=\begin{bmatrix} -l_{c1}sen(q_1) & 0 \\ l_{c1}cos(q_1) & 0 \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{3.5}$$

La posición desde los ejes de referencia al centro de masa del segundo eslabón se deriva de la misma forma, obteniéndose:

$$r_{c2}(q)=\begin{bmatrix} r_{2x}(q) \\ r_{2y}(q) \end{bmatrix}=\begin{bmatrix} l_1 cos(q_1)+l_{c2}cos(q_1+q_2) \\ l_1 sen(q_1)+l_{c2}sen(q_1+q_2) \end{bmatrix} \tag{3.6}$$

donde $r_{2x}(q)$ y $r_{2y}(q)$ son las proyecciones del centro de masa del segundo eslabón sobre el eje X y Y respectivamente. De igual manera derivando la ecuación 3.6, se obtiene:

$$\dot{r}_{c2}(q)=v_{c2}=\begin{bmatrix} -l_1 sen(q_1)-l_{c2}sen(q_1+q_2) & -l_{c2}sen(q_1+q_2) \\ l_1 cos(q_1)+l_{c2}cos(q_1+q_2) & l_{c2}cos(q_1+q_2) \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{3.7}$$

Las velocidades angulares de cada eslabón (que sólo tienen componentes en el eje Z) son definidas cómo:

$$\omega_1 = [\omega_{1x} \quad \omega_{1y} \quad \omega_{1z}]^T = [0 \quad 0 \quad q_1]^T \tag{3.8}$$

$$\omega_2 = [\omega_{2x} \quad \omega_{2y} \quad \omega_{2z}]^T = [0 \quad 0 \quad q_1 + q_2]^T \tag{3.9}$$

Estas se calculan para el eje Z, con base en la configuración del robot. La matriz de inercia se define cómo:

$$I_i = \begin{bmatrix} I_{xxi} & -I_{xyi} & -I_{xxi} \\ -I_{yxi} & I_{yyi} & -I_{yzi} \\ -I_{zzi} & -I_{xyi} & I_{zzi} \end{bmatrix}, i = 1, 2, \ldots, n \tag{3.10}$$

donde X, Y, Y son los ejes de referencia, $I_{xxi} = \int_0^{ai}\int_0^{li}\int_0^{bi}(y^2+z^2)\rho dxdydz$,

$I_{xyi} = \int_0^{ai}\int_0^{li}\int_0^{bi} xy\rho dxdydz$, $I_{zzi} = \int_0^{ai}\int_0^{li}\int_0^{bi} xz\rho dxdydz$, $I_{yxi} = \int_0^{ai}\int_0^{li}\int_0^{bi} yx\rho dxdydz$,

$I_{yyi} = \int_0^{ai}\int_0^{li}\int_0^{bi}(x^2+y^2)\rho dxdydz$, $I_{yzi} = \int_0^{ai}\int_0^{li}\int_0^{bi} yz\rho dxdydz$, $I_{zxi} = \int_0^{ai}\int_0^{li}\int_0^{bi} zx\rho dxdydz$,

$I_{zyi} = \int_0^{ai}\int_0^{li}\int_0^{bi} zy\rho dxdydz$, $I_{zzi} = \int_0^{ai}\int_0^{li}\int_0^{bi}(x^2+y^2)\rho dxdydz$

a_i, l_i y b_i son el espesor, longitud y ancho del i-ésimo eslabón; ρ es la densidad del material de los eslabones.

Al realizar los cálculos para los elementos de la matriz de inercia, está queda definida cómo:

$$Ii = \begin{bmatrix} \frac{a_i l_i b_i \rho}{3}\left(a_i^2 + l_i^2\right) & -\frac{a_i l_i b_i \rho}{4} b_i l_i & -\frac{a_i l_i b_i \rho}{4} a_i b_i \\ -\frac{a_i l_i \rho}{4} b_i l_i & \frac{a_i l_i b_i \rho}{3}\left(a_i^2 + b_i^2\right) & -\frac{a_i l_i b_i \rho}{4} a_i l_i \\ -\frac{a_i l_i b_i \rho}{4} a_i b_i & -\frac{a_i b_i l_i \rho}{4} a_i l_i & \frac{a_i l_i b_i \rho}{3}\left(b_i^2 + l_i^2\right) \end{bmatrix} \tag{3.11}$$

Sustituyendo las ecuaciones (3.5), (3.7), (3.8), (3.9) y (3.11) en la ecuación de la energía cinética (3.3) se obtiene que K es igual a:

$$K = \frac{1}{2}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}^T \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{3.12}$$

o

$$K = \frac{1}{2}\left(D_{11}\dot{q}_1^2 + 2D_{12}\dot{q}_1\dot{q}_2 + D_{22}\dot{q}_2^2\right) \tag{3.13}$$

donde:

$$\begin{aligned} D_{11} &= m_i l_{c1}^2 + m_2\left(l_1^2 + l_{c2}^2 + 2l_1 l_{c2} cos(q_2)\right) + I_{zz1} + I_{zz2} \\ D_{12} &= m_2\left(l_{c2}^2 + l_1 l_2 cos(q_2)\right) + I_{zz2} \\ D_{21} &= D_{12} \\ D_{22} &= m_2 l_{c2}^2 + I_{zz2} \end{aligned} \tag{3.14}$$

La energía potencial V para éste sistema se calcula cómo:

$$V = \sum_{i=1}^{2} m_i g_y h_i \tag{3.15}$$

donde m_i es la masa asociada al i-ésimo eslabón, h_i es la distancia que existe entre el centro de masa y el eje coordenado x del i-ésimo eslabón y g_y es la componente de la gravedad en el eje y.

De la Figura 3.1 se deduce que:

$$h_1 = l_{c1} sen(q_1),$$
$$h_2 = l_1 sen(q_1) + l_{c2} sen(q_1 + q_2) \tag{3.16}$$

Así que la energía potencial V se calcula cómo:

$$V = m_1 g l_{c1} sen(q_1) + m_2 g\left(l_1 sen(q_1) + l_{c2} sen(q_1 + q_2)\right) \tag{3.17}$$

Con la energía cinética y potencial ya conocidas, es posible determinar el Lagrangiano para el robot, sustituyendo las ecuaciones (3.13) y (3.17) en la ecuación (3.2):

$$\begin{aligned} L = {} & \frac{1}{2}\left(D_{11}\dot{q}_1^2 + 2D_{12}\dot{q}_1\dot{q}_2 + D_{22}\dot{q}_2^2\right) - m_1 g l_{c1} sen(q_1) \\ & - m_2\left(l_1 sen(q_1) + l_{c2} sen(q_1 + q_2)\right) \end{aligned} \tag{3.18}$$

Por lo que los términos de la ecuación (3.1) son:

$$\begin{aligned} \frac{\partial L}{\partial \dot{q}_1} &= D_{11}\dot{q}_1 + D_{12}\dot{q}_2 \\ &= \left(m_1 l_{c1}^2 + m_2\left(l_1^2 + l_{c2}^2 + 2l_1 l_{c2} cos(q_2)\right) + I_{zz1} + I_{zz2}\right)\dot{q}_1 \\ &\quad + \left(m_2\left(l_{c2}^2 + l_1 l_{c2} cos(q_2)\right) + I_{zz2}\right)\dot{q}_2, \end{aligned}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_1}\right) = D_{11}\ddot{q}_1 + D_{12}\ddot{q}_2 - 2m_2 l_1 l_{c2}\dot{q}_1\dot{q}_2 sen(q_2) - m_2 l_1 l_{c2}\dot{q}_2^2 sen(q_2)$$

$$\frac{\partial L}{\partial q_1} = -\left(m_1 g l_{c1} cos(q_1) + m_2 g(l_1 cos(q_1) + l_{c2} cos(q_1 + q_2))\right)$$

$$\frac{\partial L}{\partial \dot{q}_2} = D_{12}\dot{q}_1 + D_{22}\dot{q}_2 \qquad (3.19)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_2}\right) = D_{12}\ddot{q}_1 + D_{22}\ddot{q}_2 - m_2 l_1 l_{c2}\dot{q}_1\dot{q}_2 sen(q_2)$$

$$\frac{\partial L}{\partial q_2} = -m_1 l_1 l_{c2}\dot{q}_1^2 sen(q_2) - m_2 l_1 l_{c2}\dot{q}_1\dot{q}_2 sen(q_1) - m_2 g l_{c2} cos(q_1 + q_2)$$

Considerando que el robot es totalmente actuado, su dinámica es:

$$\begin{aligned}\tau_1 = &\left(m_1 l_{c1}^2 + m_2\left(l_1^2 + l_{C2}^2 + 2l_1 l_{c2} cos(q_2)\right) + I_{zz1} + I_{zz2}\right)\ddot{q}_1 \\ &+ \left(m_2\left(l_{C2}^2 + l_1 l_{c2} cos(q_2)\right) + I_{zz2}\right)\ddot{q}_2 - 2m_2 l_1 l_{c2}\dot{q}_1\dot{q}_2 sen(q_2) \\ &- m_2 l_1 l_{c2}\dot{q}_2^2 sen(q_2) + m_1 g l_{c1} cos(q_1) + m_2 g\left(l_1 cos(q_1) + l_{c2} cos(q_1 + q_2)\right)\end{aligned} \qquad (3.20)$$

$$\begin{aligned}\tau_2 = &\left(m_2\left(l_{c2}^2 + l_1 l_{c2} cos(q_2)\right) + I_{zz2}\right)\ddot{q}_1 + \left(m_2 l_{c2}^2 + I_{zz2}\right)\ddot{q}_2 \\ &+ m_2 l_1 l_{c2}\dot{q}_1^2 sen(q_2) + m_2 g l_{c2} cos(q_1 + q_2)\end{aligned} \qquad (3.21)$$

Estas ecuaciones pueden escribirse de una manera reducida cómo:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \qquad (3.22)$$

donde $q = [q_1 \quad q_2]^T$, $\dot{q} = [\dot{q}_1 \quad \dot{q}_2]^T$, $\ddot{q} = [\ddot{q}_1 \quad \ddot{q}_2]^T$,

$$
\begin{aligned}
D(q) &= \begin{bmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{bmatrix}, \\
C(q,\dot{q}) &= \begin{bmatrix} -2m_2 l_1 l_{c2} \dot{q}_2 sen(q_2) & -m_2 l_1 l_{c2} \dot{q}_2 sen(q_2) \\ m_2 l_1 l_{c2} \dot{q}_1 sen(q_2) & 0 \end{bmatrix}, \\
G(q) &= \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} m_1 g l_{c1} cos(q_1) + m_2 g l_1 cos(q_1) + m_2 g l_{c2} cos(q_1 + q_2) \\ m_2 g l_{c2} cos(q_1 + q_2) \end{bmatrix} \\
\tau &= \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}
\end{aligned}
\tag{3.23}
$$

con $D(q)$ matriz de inercias simétrica y definida positiva, $C(q,\dot{q})$ matriz de Coriolis y $G(q)$ vector que contiene los términos gravitacionales.

Para hacer la comparación entre el controlador difuso que será propuesto en la siguiente sección y un controlador PD tradicional se toma cómo ejemplo el Rhino-Bot [12]. Para éste sistema la matriz de inercias está dada cómo:

$$
D(q) = \begin{bmatrix} l^2(m_1 + 2m_2(1 + cos(q_2))) & l^2 m_2(1 + cos(q_2)) \\ l^2 m_2(1 + cos(q_2)) & l^2 m_2 \end{bmatrix} \tag{3.24}
$$

la matriz de Coriolis cómo:

$$
C(q,\dot{q}) = \begin{bmatrix} 2l^2 m_2 \dot{q}_2 sen(q_2) & l^2 m_2 \dot{q}_2 sen(q_2) \\ l^2 m_2 \dot{q}_1 sen(q_2) & 0 \end{bmatrix} \tag{3.25}
$$

y el vector de gravedad cómo:

$$
G(q) = \begin{bmatrix} (m_1 + m_2) lgcos(q_1) + m_2 lgcos(q_1 + q_2) \\ m_2 lgcos(q_1 + q_2) \end{bmatrix} \tag{3.26}
$$

Los valores de las masas y medidas geométricas tomados para éste sistema se encuentran en la Tabla B.1 del Apéndice B.

3.2 Análisis de estabilidad

Las tres propiedades siguientes, relacionadas con la ecuación (3.22), son utilizadas para el diseño del controlador.

Propiedad 1: La matriz de inercia $D(q)$ es simétrica y definida positiva.

Propiedad 2: Usando una definición adecuada de la matriz C, la matriz $\left(\dot{D} - 2C\right)$ es antisimetrica.

Propiedad 3: Definiendo P cómo la energía potencial del sistema entonces:

$$G(q) = \frac{\partial P}{\partial q}$$

El robot considerado es un sistema multivariable con n entradas (torque) y n salidas (posición) que interactúan de una forma nolineal. Los grados de acoplamiento decrecen desde la unión 1 hasta la unión n. Por lo tanto se propone la siguiente acción de control [5]:

$$\tau_i = \sum_{j=i}^{n} u_{ij} \qquad i = 1,\ldots,n \tag{3.27}$$

donde cada u_{ij} corresponde a la salida del controlador difuso, cómo el derivado en la sección 2.3, que emplea entradas (e_j y $\dot{e}_j$) de la j-esima unión y produce un torque de salida para esta misma unión. Así que τ_n consiste de un sólo elemento u_{nn} mientras que τ_1 tiene n términos: $u_{11} + u_{12} + \ldots + u_{1n}$.

De acuerdo a la sección 2.3 del capítulo anterior, para las regiones de entrada IC1-IC8 el controlador PD difuso toma la forma:

$$
\begin{aligned}
u_{PD} &= \frac{K_u L}{2\left(2L - K_p|e|\right)}\left[K_p e + K_d \dot{e}\right] \\
&\qquad si \quad L \geq K_p|e| \geq K_d|\dot{e}| \\
&= \frac{K_u L}{2\left(2L - K_d|\dot{e}|\right)}\left[K_p e + K_d \dot{e}\right] \\
&\qquad si \quad L \geq K_d|\dot{e}| \geq K_p|e|
\end{aligned}
\tag{3.28}
$$

Entonces, cada controlador PD difuso puede ser expresado cómo:

$$
\begin{aligned}
u_{ij} &= \frac{K_{uij} L_{ij}}{2\left(2L_{ij} - K_{pij}|e_j|\right)}\left[K_{pij} e_j + K_{dij} \dot{e}_j\right] \\
&\qquad si \quad L_{ij} \geq K_{pij}|e_j| \geq K_{dij}|\dot{e}_j| \\
&= \frac{K_{uij} L_{ij}}{2\left(2L_{ij} - K_{dij}|\dot{e}_j|\right)}\left[K_{pij} e_j + K_{dij} \dot{e}_j\right] \\
&\qquad si \quad L_{ij} \geq K_{dij}|\dot{e}_j| \geq K_{pij}|e_j|
\end{aligned}
\tag{3.29}
$$

Para simplificar la notación, definiremos los siguientes términos:

$$
\begin{cases}
\tilde{K}_{pij} = \dfrac{K_{uij} L_{ij} K_{pij}}{2\left(2L_{ij} - K_{pij}|e_j|\right)} \\
\tilde{K}_{dij} = \dfrac{K_{uij} L_{ij} K_{dij}}{2\left(2L_{ij} - K_{pij}|e_j|\right)}
\end{cases}
$$

$$
si \quad L_{ij} \geq K_{pij}|e_j| \geq K_{dij}|\dot{e}_j|
$$

$$\begin{cases} \widetilde{K}_{pij} = \dfrac{K_{uij} L_{ij} K_{pij}}{2\left(2L_{ij} - K_{dij}\left|\dot{e}_j\right|\right)} \\ \widetilde{K}_{dij} = \dfrac{K_{uij} L_{ij} K_{dij}}{2\left(2L_{ij} - K_{dij}\left|\dot{e}_j\right|\right)} \end{cases} \tag{3.30}$$

$$si \quad L_{ij} \geq K_{dij}\left|\dot{e}_j\right| \geq K_{pij}\left|e_j\right|$$

Considere la ecuación (3.30), cada subcontrolador difuso puede ser escrito cómo $u_{ij} = \widetilde{K}_{pij} e_j + \widetilde{K}_{dij} \dot{e}_j$. Mas aún el controlador difuso MIMO (del inglés "Multi-Input, Multi-Output") es expresado en forma matricial cómo:

$$\tau = \boldsymbol{K}_p e + \boldsymbol{K}_d \dot{e} \tag{3.31}$$

donde $\tau, e \in R^n$, e es definido cómo $e = \tilde{q} = q_d - q$, y $\boldsymbol{K_p}$ y $\boldsymbol{K_d}$ n x n matrices triangulares superiores definidas cómo:

$$\boldsymbol{K}_p = \begin{bmatrix} \widetilde{K}_{p11} & \widetilde{K}_{p12} & \cdots & \widetilde{K}_{p1n} \\ 0 & \widetilde{K}_{p22} & \cdots & \widetilde{K}_{p2n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \widetilde{K}_{pnn} \end{bmatrix}$$

$$\boldsymbol{K}_d = \begin{bmatrix} \widetilde{K}_{d11} & \widetilde{K}_{d12} & \cdots & \widetilde{K}_{d1n} \\ 0 & \widetilde{K}_{d22} & \cdots & \widetilde{K}_{d2n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \widetilde{K}_{dnn} \end{bmatrix} \tag{3.32}$$

Tradicionalmente, los elementos K_{uij}, K_{pij} y K_{dij} son escogidos positivos. Por lo que, los elementos de la diagonal de $\boldsymbol{K_p}$ y $\boldsymbol{K_d}$ son positivos.

Para el análisis de control de posición, q_d es constante, por lo que $\dot{e} = -\dot{q}$ y el control se puede escribir cómo [13,14]:

$$\tau = K_p\tilde{q} - K_d\dot{q} \tag{3.33}$$

El lazo cerrado del robot descrito por la ecuación (3.22) y la ley de control (3.33) puede escribirse cómo:

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}\\ \dot{q}\end{bmatrix} = \begin{bmatrix}-\dot{q}\\ D^{-1}(q)\left[K_p\tilde{q} - K_d\dot{q} - C(q,\dot{q}) - G(q)\right]\end{bmatrix} \tag{3.34}$$

Para hacer el análisis de estabilidad de éste sistema, se propone la función candidata de Lyapunov V [1]:

$$V = \frac{1}{2}\dot{q}^T\boldsymbol{D}\dot{q} + \int_0^{\tilde{q}}\sigma^T\boldsymbol{K}_p^T d\sigma + P \tag{3.35}$$

La derivada con respecto al tiempo de la función candidata de Lyapunov es:

$$\dot{V} = \frac{1}{2}\dot{q}^T\dot{\boldsymbol{D}}\dot{q} + \dot{q}^T\boldsymbol{D}\ddot{q} - \tilde{q}^T\boldsymbol{K}_p^T\dot{q} + \left(\frac{\partial P}{\partial q}\right)^T\dot{q} \tag{3.36}$$

donde el término $\tilde{q}^T K_p^T\dot{\tilde{q}}$ se reemplaza por $-\tilde{q}^T K_p^T\dot{q}$

Usando la ecuación (3.22) y las tres propiedades mencionadas, (3.36) puede reescribirse cómo:

$$\begin{aligned}\dot{V} &= \frac{1}{2}\dot{q}^T \dot{\boldsymbol{D}}\dot{q} + \dot{q}^T \boldsymbol{D}\left[\boldsymbol{D}^{-1}(\tau - C\dot{q} - \boldsymbol{G})\right] - \tilde{q}^T \boldsymbol{K}_p^T \dot{q} + \left(\frac{\partial P}{\partial q}\right)^T \dot{q} \\ &= \dot{q}^T \left(\frac{1}{2}\dot{\boldsymbol{D}} - \boldsymbol{C}\right)\dot{q} + \dot{q}^T \tau - \tilde{q}^T \boldsymbol{K}_p^T \dot{q} \\ &= \dot{q}^T \boldsymbol{K}_p \tilde{q} - \dot{q}^T \boldsymbol{K}_d \dot{q} - \tilde{q}^T \boldsymbol{K}_p^T \dot{q}\end{aligned} \tag{3.37}$$

Puesto que $\dot{q}^T K_p \tilde{q}$ es un función escalar, se tiene que $\dot{q}^T \boldsymbol{K}_p \tilde{q} = \tilde{q}^T \boldsymbol{K}_p^T \dot{q}$.

Consecuentemente (3.37) se transforma en:

$$\dot{V} = -\dot{q}^T \boldsymbol{K}_d \dot{q} < 0, \dot{V} \leq 0 \tag{3.38}$$

Se requiere ahora determinar las condiciones que garanticen $V > 0$; puesto que la energía potencial es una cantidad relativa, es posible hacer que $P \geq 0$. Para encontrar las condiciones $\int_0^{\tilde{q}} \sigma^T \boldsymbol{K}_p^T d\sigma \geq 0$, comenzaremos con el caso de 3x3.

Retomando las matrices de la ecuación (3.32):

$$\boldsymbol{K}_p = \begin{bmatrix} \tilde{K}_{p11} & \tilde{K}_{p12} & \tilde{K}_{p13} \\ 0 & \tilde{K}_{p22} & \tilde{K}_{p23} \\ 0 & 0 & \tilde{K}_{p33} \end{bmatrix}$$

$$\boldsymbol{K}_d = \begin{bmatrix} \tilde{K}_{d11} & \tilde{K}_{d12} & \tilde{K}_{d13} \\ 0 & \tilde{K}_{d22} & \tilde{K}_{d23} \\ 0 & 0 & \tilde{K}_{d33} \end{bmatrix} \tag{3.39}$$

entonces:

$$\int_0^{\tilde{q}} \sigma^T \boldsymbol{K}_p^T d\sigma = \int_0^{\tilde{q}1} \left(\tilde{K}_{p11}\sigma_1 + \tilde{K}_{p12}\sigma_2 + \tilde{K}_{p13}\sigma_3\right) d\sigma_1 + \int_0^{\tilde{q}2} \left(K_{p22}\sigma_2 + K_{p23}\sigma_3\right) d\sigma_2 + \int_0^{\tilde{q}3} K_{p33}\sigma_3 d\sigma_3 \tag{3.40}$$

De (3.30) se tiene:

$$\frac{|K_{uij}|K_{pij}}{2} \geq \left|\tilde{K}_{pij}\right| \geq \frac{|K_{uij}|K_{pij}}{4} \tag{3.41}$$

Primero, se analizan los términos con $\tilde{K}_{pii}$ $i = 1,2,3$. Por simplicidad definimos $\hat{K}_{ii} = \frac{K_{uii}K_{pii}}{4}$ Como se mencionó antes, K_{uii}, K_{pji} y K_{dij} han sido seleccionados positivos, entonces $\tilde{K}_{pii} \geq \hat{K}_{ii}$, donde $\hat{K}_{ii}$ es una constante positiva. Como resultado se puede derivar la siguiente desigualdad:

$$\int_0^{\tilde{q}i} \tilde{K}_{pii}\sigma_i d\sigma_i \geq \tilde{K}_{ii} \int_0^{\tilde{q}i} \sigma_i d\sigma_i = \frac{1}{2} K_{ii} \tilde{q}_I^2 \tag{3.42}$$

Similarmente, se encuentra una cota para los términos con $\tilde{K}_{pij}, i \neq j$ cómo:

$$\left|\int_0^{\tilde{q}} \tilde{K}_{pij}\sigma_j d\sigma_i\right| \leq \left|\tilde{K}_{pij}\right| \left|\int_0^{\tilde{q}} \sigma_j d\sigma_i\right| \leq \left|\tilde{K}_{pij}\right| \|e_i\| e_j \tag{3.43}$$

Nuevamente de (3.41) se tiene que $\left|\tilde{K}_{pij}\right| \leq \frac{|K_{uij}|K_{pij}}{2}$.

Ahora, se define $\overline{K}_{ij} = \dfrac{|K_{vij}|K_{pij}}{2}$ y la substituimos en la ecuación (3.43), donde se obtiene:

$$-\overline{K}_{ij}|\tilde{q}_i||\tilde{q}_j| \le \int_0^{\tilde{q}} \tilde{K}_{pij}\sigma_j d\sigma_i \le \overline{K}_{ij}|\tilde{q}_i||\tilde{q}_j| \tag{3.44}$$

Entonces, para éste caso de 3x3, se tiene:

$$\int_0^{\tilde{q}} \sigma^T \boldsymbol{K}_p^T d\sigma \ge \frac{1}{2}\left(\hat{K}_{11}\tilde{q}_1^2 + \hat{K}_{22}\tilde{q}_2^2 + \hat{K}_{33}\tilde{q}_3^2\right) - \left(\hat{K}_{12}|\tilde{q}_1||\tilde{q}_2| + \hat{K}_{23}|\tilde{q}_2||\tilde{q}_3| + \hat{K}_{13}|\tilde{q}_1||\tilde{q}_3|\right) \tag{3.45}$$

Reagrupando los términos del lado derecho de la ecuación (3.45), se obtiene:

$$\frac{1}{4}\left[\begin{array}{l}\left(\hat{K}_{11}\tilde{q}_1^2 - 4\overline{K}_{12}|\tilde{q}_1||\tilde{q}_2| + \hat{K}_{22}\tilde{q}_2^2\right) + \left(\hat{K}_{22}\tilde{q}_2^2 - 4\overline{K}_{23}|\tilde{q}_2||\tilde{q}_3| + \hat{K}_{33}\tilde{q}_2^3\right) \\ + \left(\hat{K}_{11}\tilde{q}_2^1 - 4\overline{K}_{13}|\tilde{q}_1||\tilde{q}_3| + \hat{K}_{33}\tilde{q}_3^2\right)\end{array}\right] \tag{3.46}$$

$$si\begin{cases}\sqrt{\hat{K}_{11}\hat{K}_{22}} \ge 2\overline{K}_{12} \\ \sqrt{\hat{K}_{11}\hat{K}_{33}} \ge 2\overline{K}_{13} \\ \sqrt{\hat{K}_{22}\hat{K}_{33}} \ge 2\overline{K}_{23}\end{cases}$$

entonces:

$$(3.46) \ge \frac{1}{4}\left[\left(\sqrt{\hat{K}_{11}}\tilde{q}_1 - \sqrt{\hat{K}_{22}}\tilde{q}_2\right)^2 + \left(\sqrt{\hat{K}_{11}}|\tilde{q}_1| - \sqrt{\hat{K}_{22}}|\tilde{q}_2|\right)^2 + \left(\sqrt{\hat{K}_{11}}|\tilde{q}_1| - \sqrt{\hat{K}_{22}}|\tilde{q}_2|\right)^2\right] \ge 0$$

por lo que, $\int_0^{\tilde{q}} \sigma^T \boldsymbol{K}_p^T d\sigma \ge 0$ si $\begin{cases}\sqrt{\hat{K}_{11}\hat{K}_{22}} \ge 2\overline{K}_{12} \\ \sqrt{\hat{K}_{11}\hat{K}_{33}} \ge 2\overline{K}_{13} \\ \sqrt{\hat{K}_{22}\hat{K}_{33}} \ge 2\overline{K}_{23}\end{cases}$

A partir de las condiciones encontradas para el caso de 3 x 3 y expandiendo para el caso de n x n, establecemos la siguiente condición de suficiencia para $\int_0^{\tilde{q}} \sigma^T \boldsymbol{K}_p^T d\sigma \geq 0$:

$$\frac{1}{2}\sqrt{K_{uii}K_{pii}K_{uij}K_{pjj}} \geq (n-1)\left|K_{uij}\right|K_{pij}, i \neq j \quad y \quad j=1,\ldots,n \tag{3.47}$$

similarmente para $\dot{q}^T \boldsymbol{K}_d \dot{q} > 0$ tenemos:

$$\frac{1}{2}\sqrt{K_{uii}K_{dii}K_{ujj}K_{dij}} \geq (n-1)\left|K_{uij}\right|K_{dij}, i \neq j \quad y \quad j=1,\ldots,n \tag{3.48}$$

Entonces con (3.47) y (3.48), podemos garantizar que $V>0$ y que $\dot{V} \leq 0$.

Hasta ahora, se han encontrado las condiciones para garantizar $V>0$ y $\dot{V} \leq 0$, para probar la estabilidad asintótica, se aplica el teorema de Lasalle [7].

En la región

$$\Omega = \left\{ \begin{pmatrix} \tilde{q} \\ \dot{q} \end{pmatrix} : \dot{V}(\tilde{q}, \dot{q}) \equiv 0 \right\}, \Omega \in R^{2n} \tag{3.49}$$

se cumple

$$\left\{ \begin{bmatrix} \tilde{q} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \tilde{q} \\ 0 \end{bmatrix} \right\} \tag{3.50}$$

Obteniendo a partir de la ecuación (3.34) que:

$$K_p\tilde{q} - G(q) = 0$$
$$\tilde{q} = \frac{G(q)}{K_p} \tag{3.51}$$

Haciendo únicamente el análisis de estabilidad sin obtener una estabilidad asintotíca, sólo condiciones para estabilidad acotada.

3.3 Control PD clásico

Primero se analiza el control del Rhino-bot utilizando un control PD clásico [12].

3.3.1 Implementación en Matlab

El robot junto con el controlador PD clásico se simula utilizando el ambiente computacional Matlab [15]. El diagrama esquemático en Simulink del robot con el controlador se muestra en la Figura 3.2

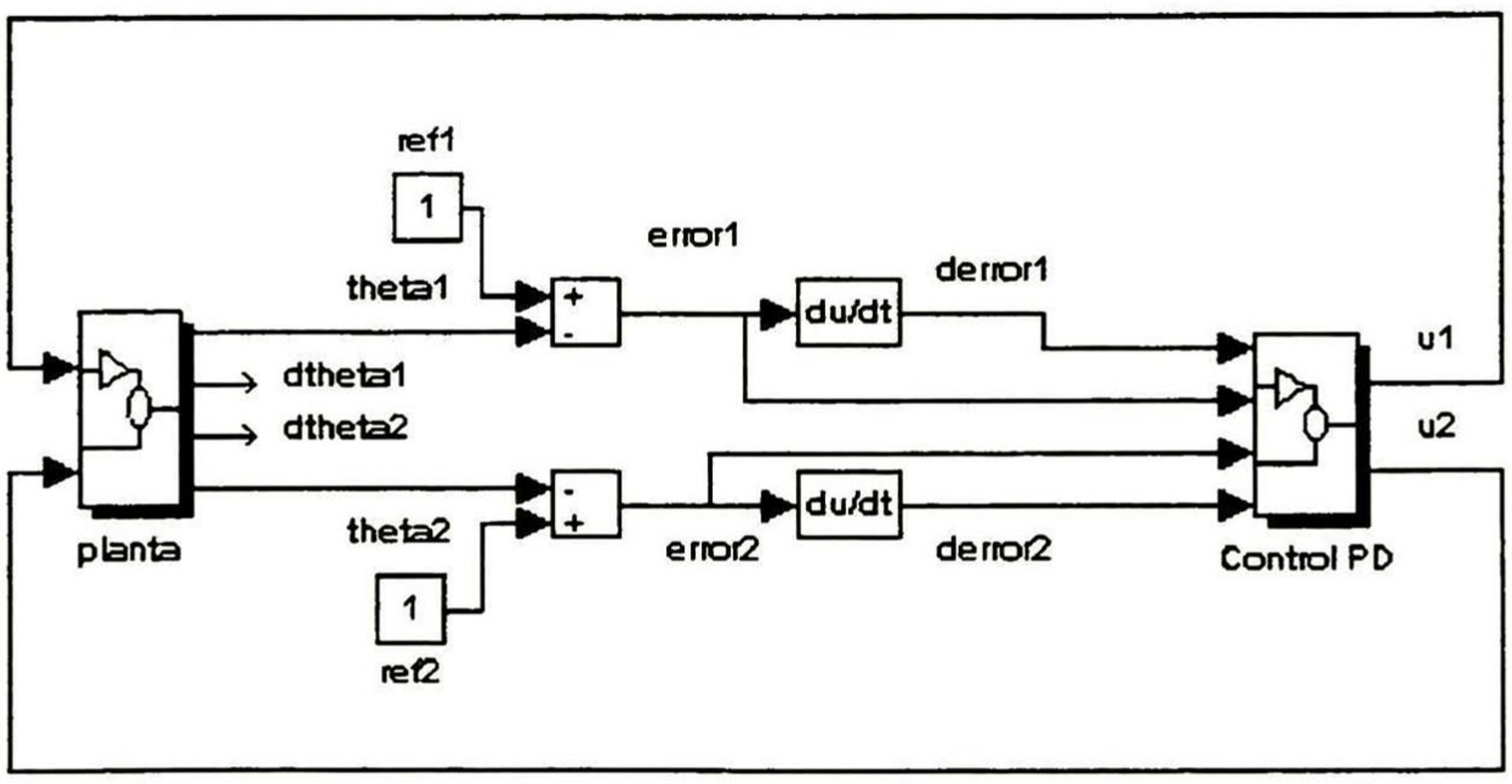


Figura 3.2 Esquema general del robot con el controlador PD clásico

Este diagrama está constituido principalmente por dos partes: la planta y el controlador. El diagrama de la planta se presenta en la Figura 3.3

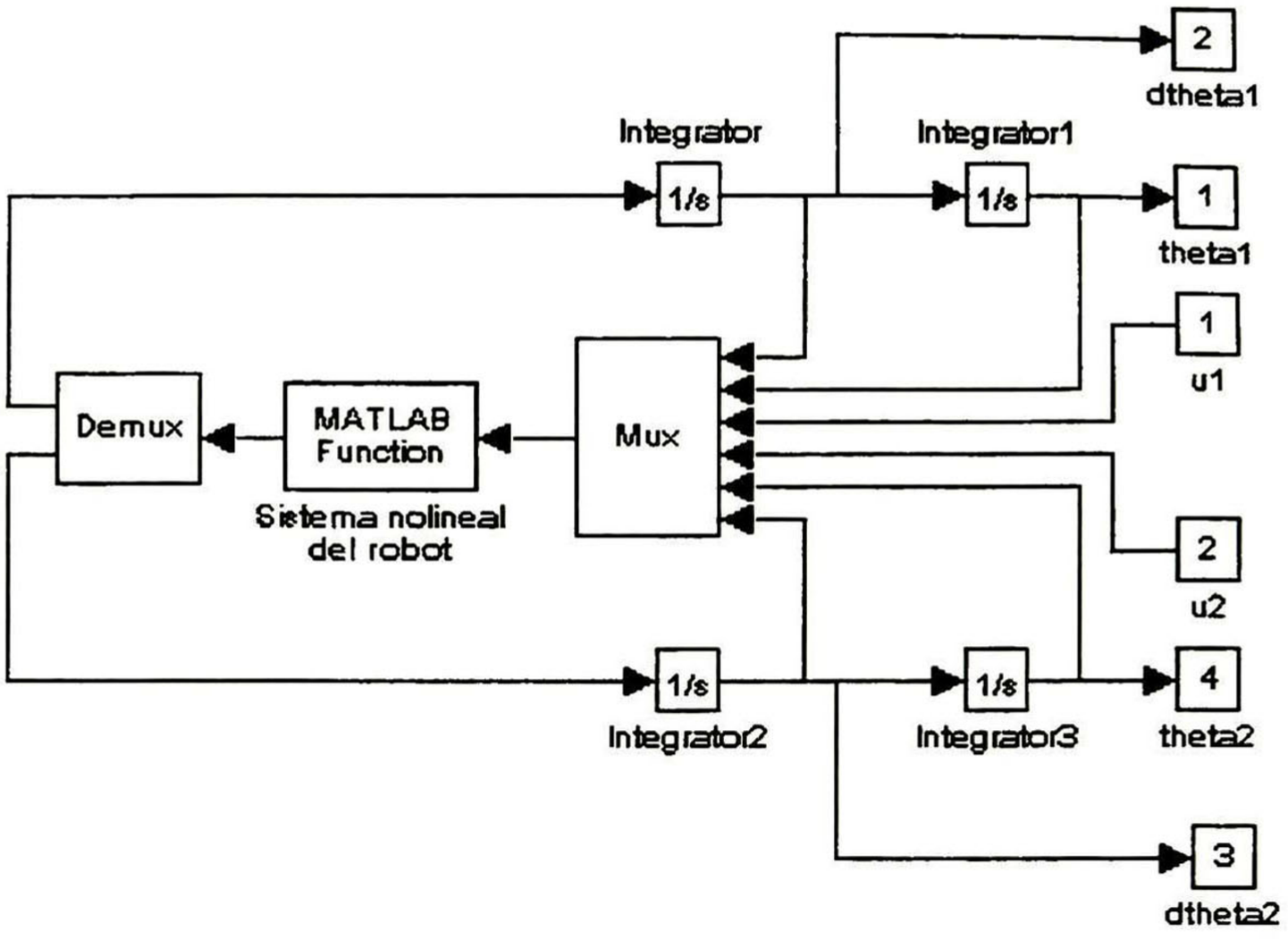


Figura 3.3 Esquema del sistema no lineal del brazo

En esta figura, se incluye la interface de Simulink con Matlab, con la que se llama a la función mostrada en el Apéndice B.1 que genera la dinámica del robot. Al introducir los estados (q_1, q_2, $\dot{q}_1$, $\dot{q}_2$) y las señales de control (u_1 y u_2), se obtienen las señales $\ddot{q}_1$ y $\ddot{q}_2$, que deben ser integradas para obtener los estados de la planta. Lo único que hace esta parte es modelar el sistema por la ecuación:

$$\ddot{q} = -D(q)^{-1}\left(C(q,\dot{q})\dot{q} + G(q) - \tau\right) \tag{3.52}$$

la matriz D(q) debe poseer inversa para toda q.

Así mismo en la Figura 3.4 se muestra el diagrama a bloques del controlador con sus ganancias correspondientes. Este recibe las señales de error y la derivada del error para cada uno de los eslabones

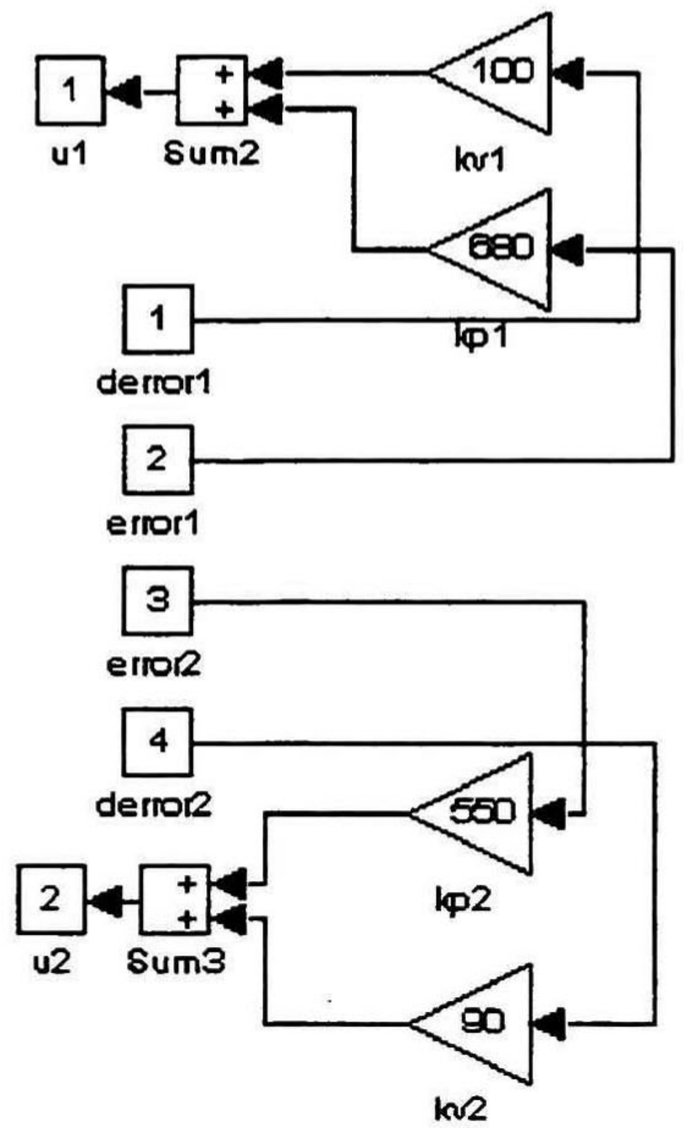


Figura 3.4 Diagrama del Controlador PD

3.3.2 Resultados obtenidos

Los resultados se presentan en la Figura 3.5 que nos muestra las posiciones y velocidades de los dos eslabones. La Tabla B.2 (Apéndice B) presenta los valores de las ganancias utilizadas para éste controlador.

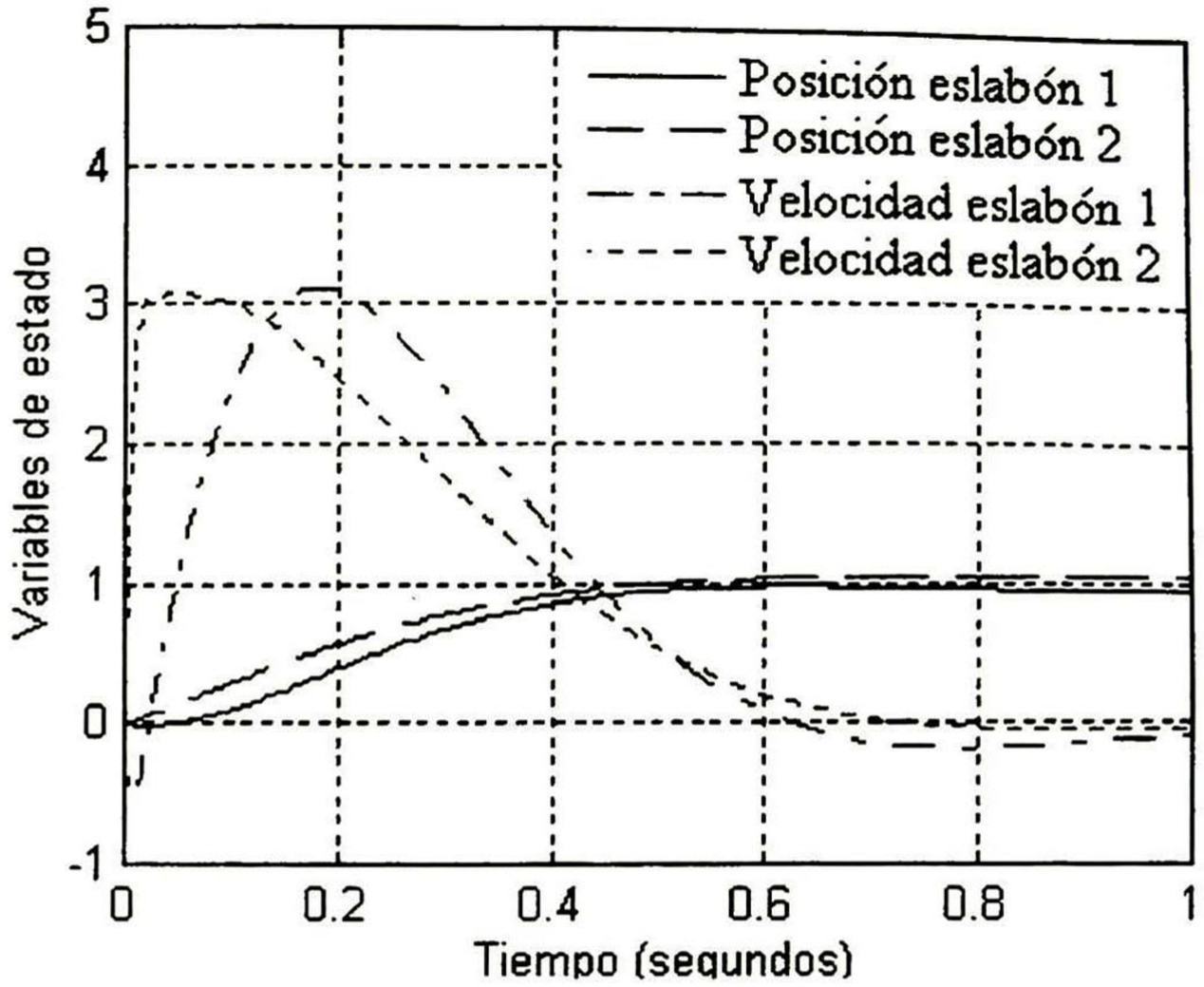


Figura 3.5 Posición y velocidad de los eslabones

Puede observarse que la posición en estado estable nunca llega a la referencia deseada. Esto se ve más claramente en la Figura 3.6 donde sólo se encuentran graficadas las señales de posición de los eslabones.

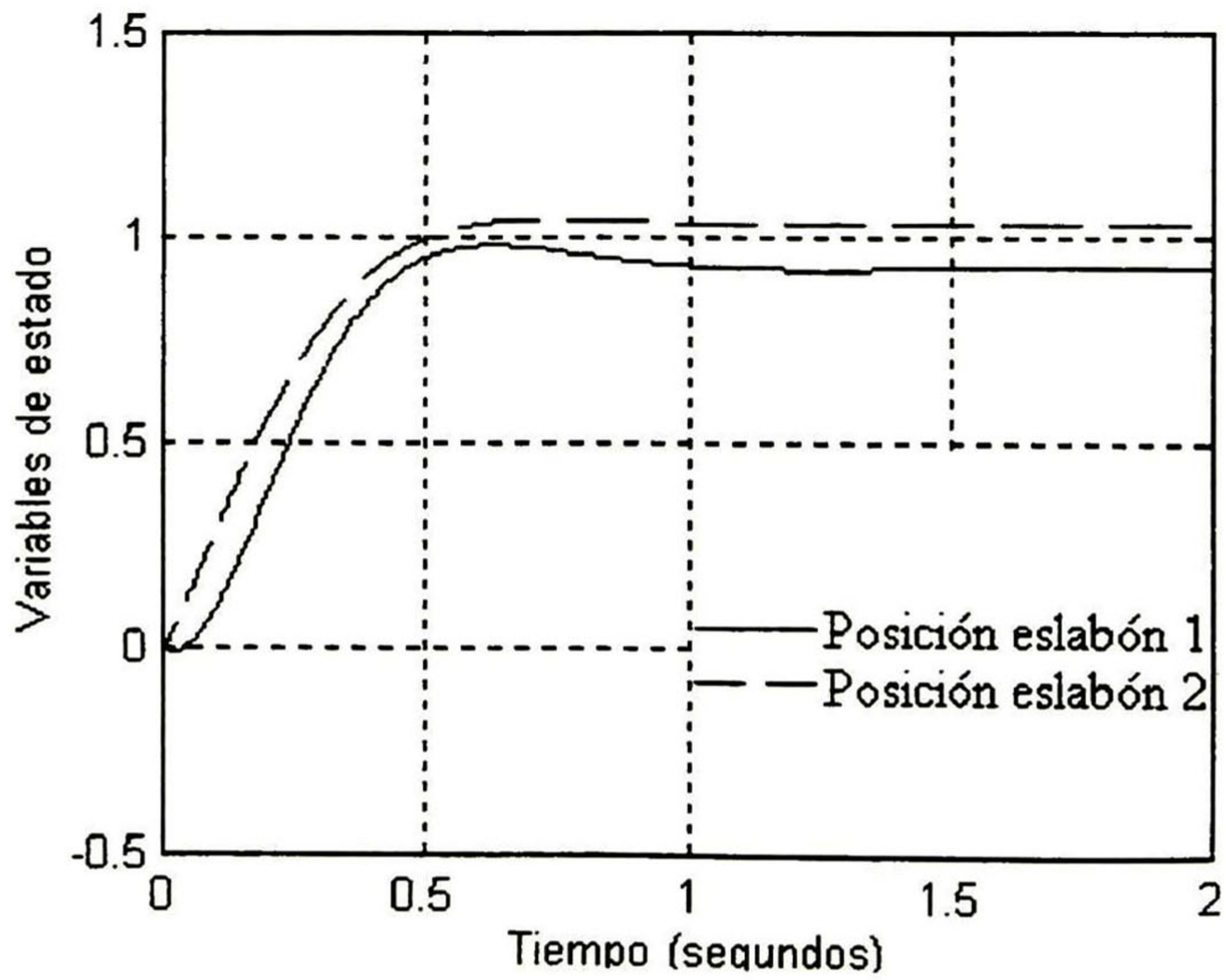


Figura 3.6 Posición de los eslabones

Si se incrementan las ganancias del controlador para tratar de reducir el error de posición, esta presenta un sobrepaso cómo se muestra en la Figura 3.7

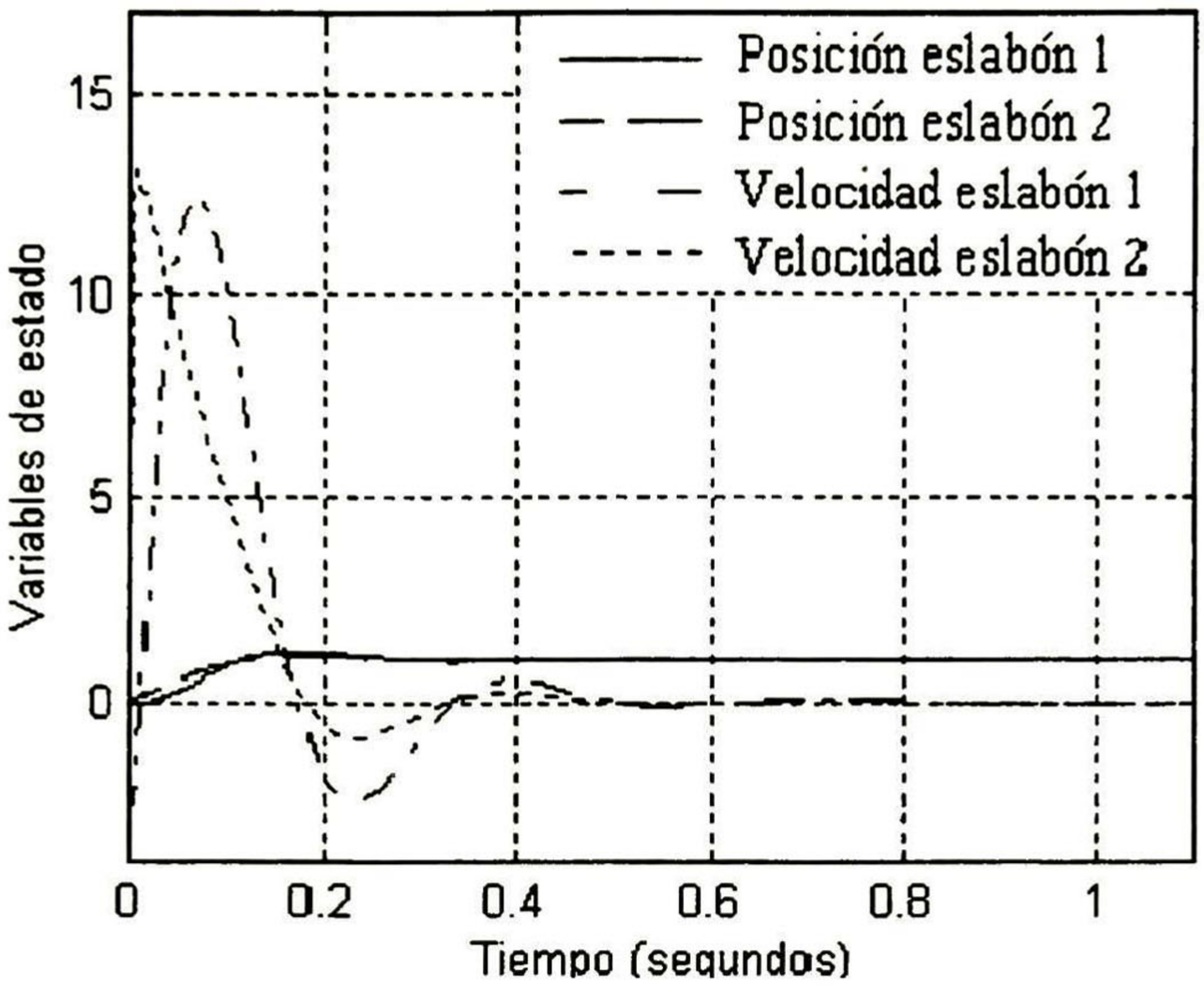
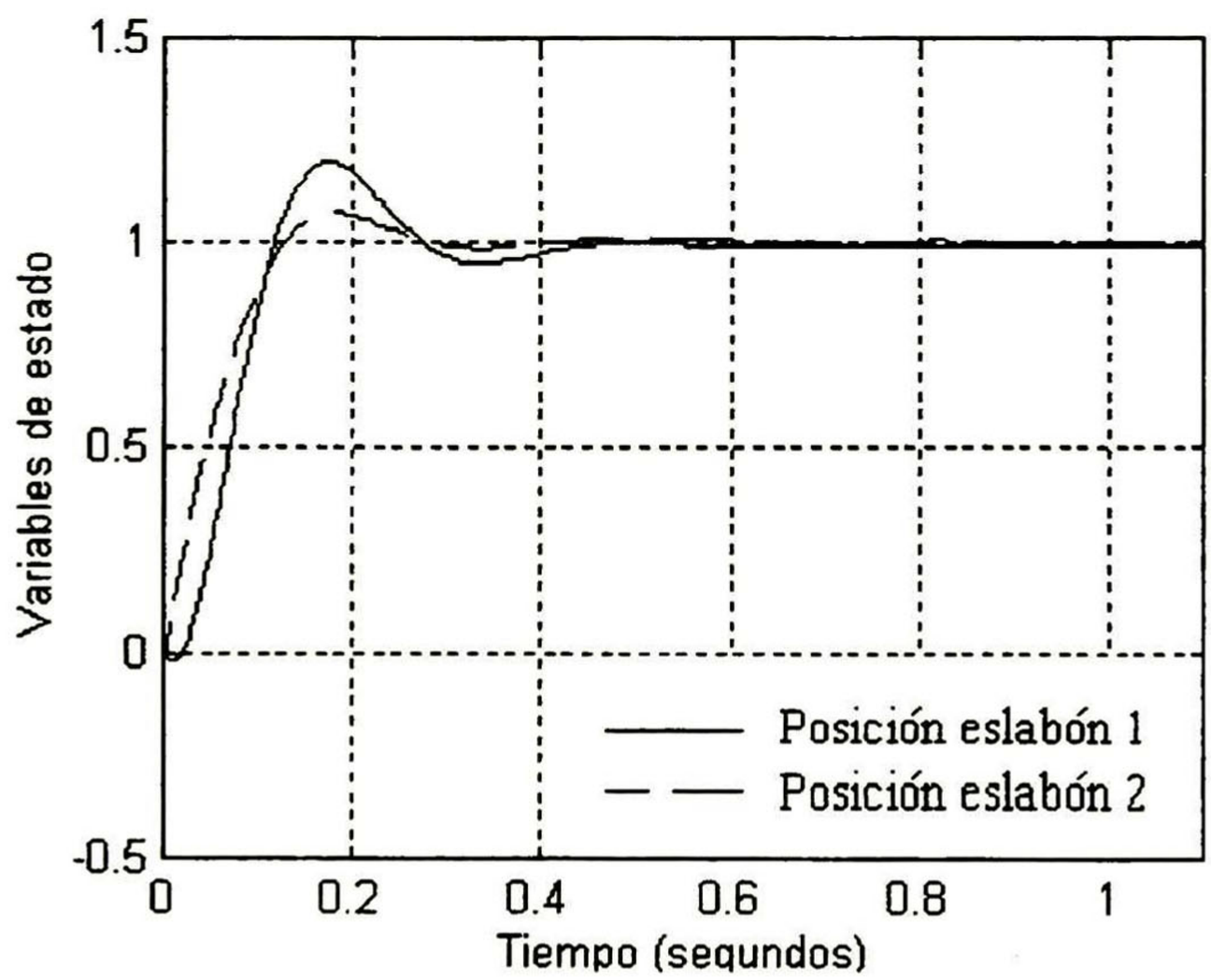


Figura 3.7 Posición y velocidad de los eslabones



3.8 Posición de los eslabones

3.4 Controlador PD Difuso

3.4.1 Implementación en Matlab

El controlador difuso propuesto en la Sección 3.2 se utiliza ahora para controlar el Rhino-bot. El sistema dinámico junto con el controlador, se simuló en Simulink por medio de una interface con Matlab. Este diagrama se muestra en la Figura 3.8

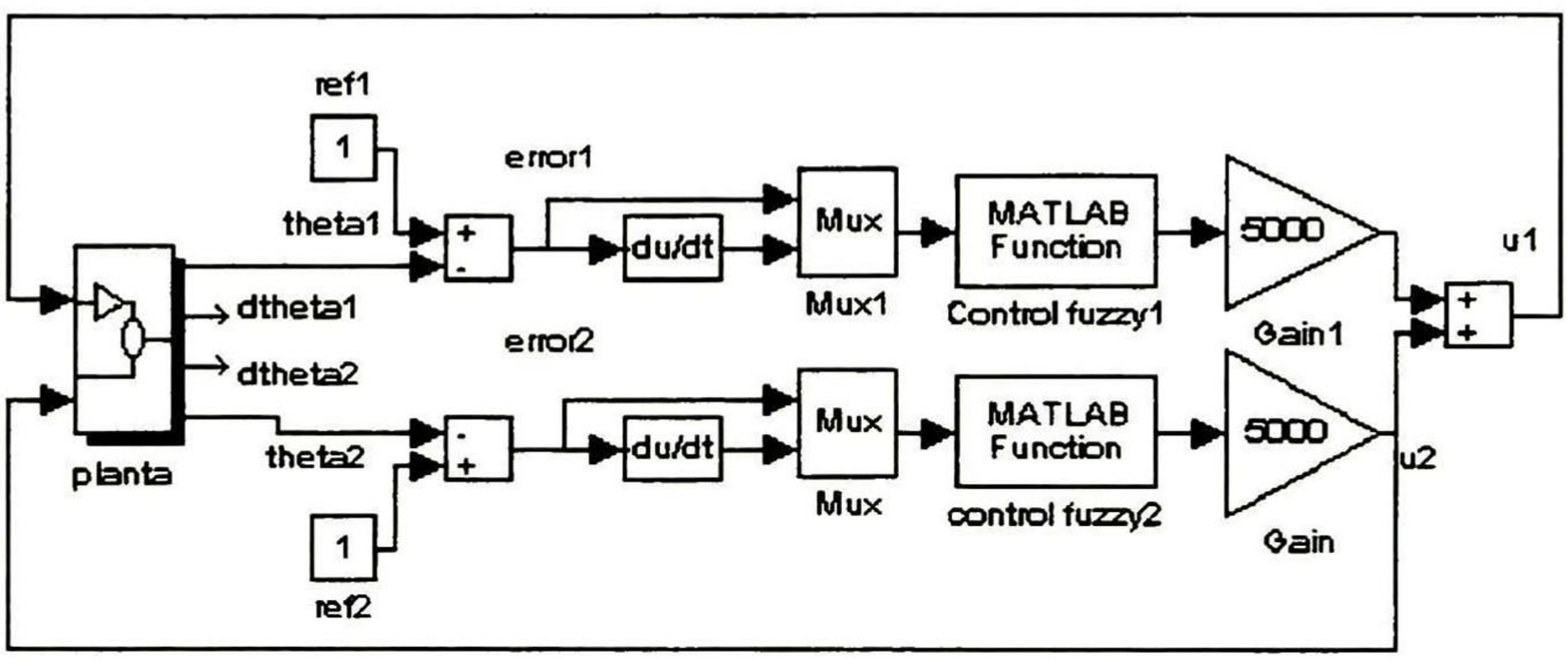


Figura 3.9 Esquema general del robot con el controlador PD difuso

El controlador PD difuso llama a la función creada en Matlab del Apéndice B.2 donde se encuentran codificadas las ecuaciones (2.89) del Capítulo 2. Esta función genera la señal de control dependiendo en cuál de las regiones IC1-IC20 se encuentre e y $\dot{e}$.

3.4.2 Resultados Obtenidos

Los resultados de la simulación con el controlador PD difuso se muestran en la Figura 3.9. Los valores de las ganancias para éste controlador son mostrados en la Tabla B.3 (Apéndice B)

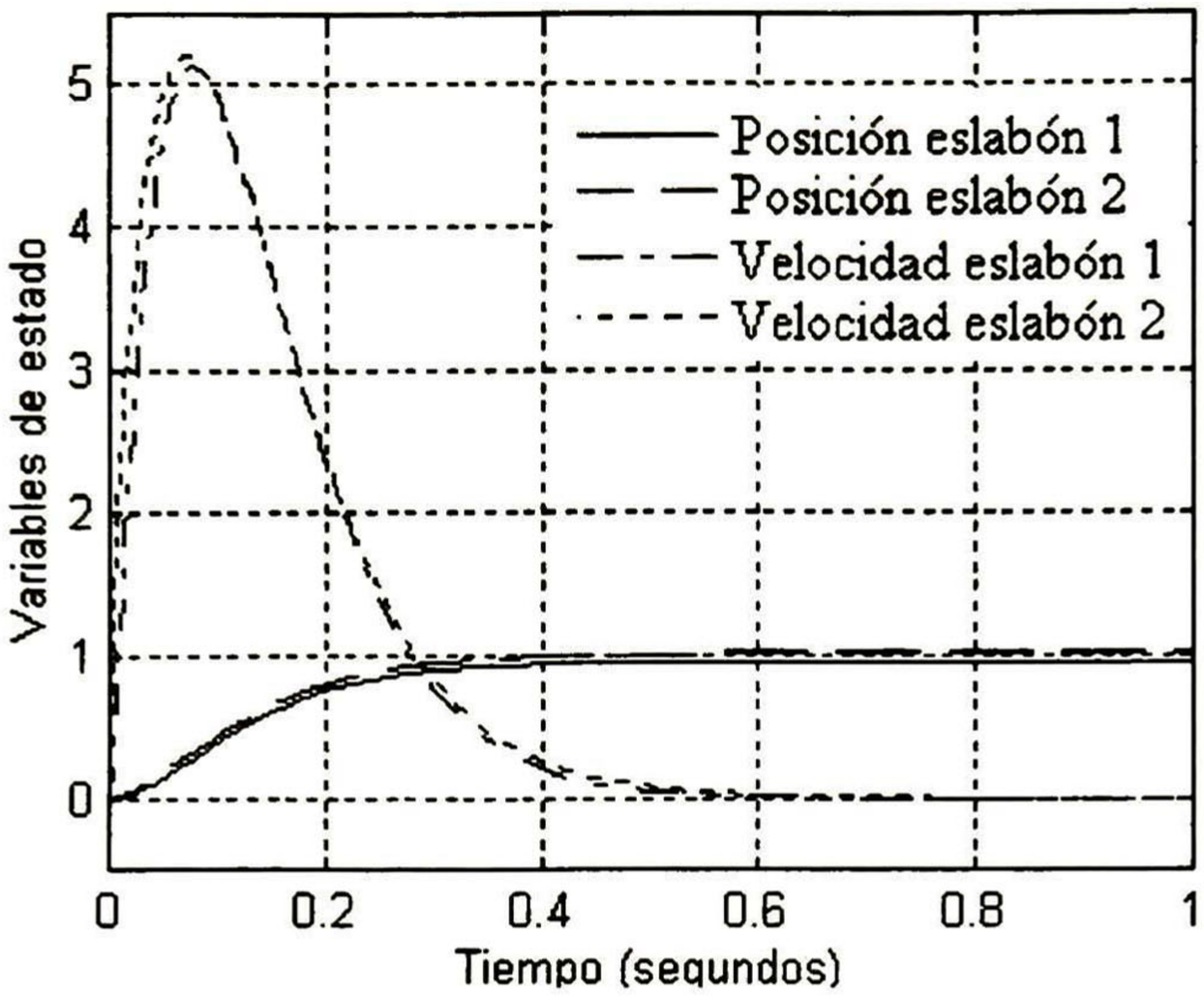


Figura 3.10 Posición y velocidad de los eslabones

Con éste controlador PD Difuso se tiene un mejor comportamiento que el propuesto en la sección 3.3 ya que el tiempo de estabilización y el error son reducidos, pero no llega completamente al punto deseado, tal y cómo se muestra la Figura 3.10

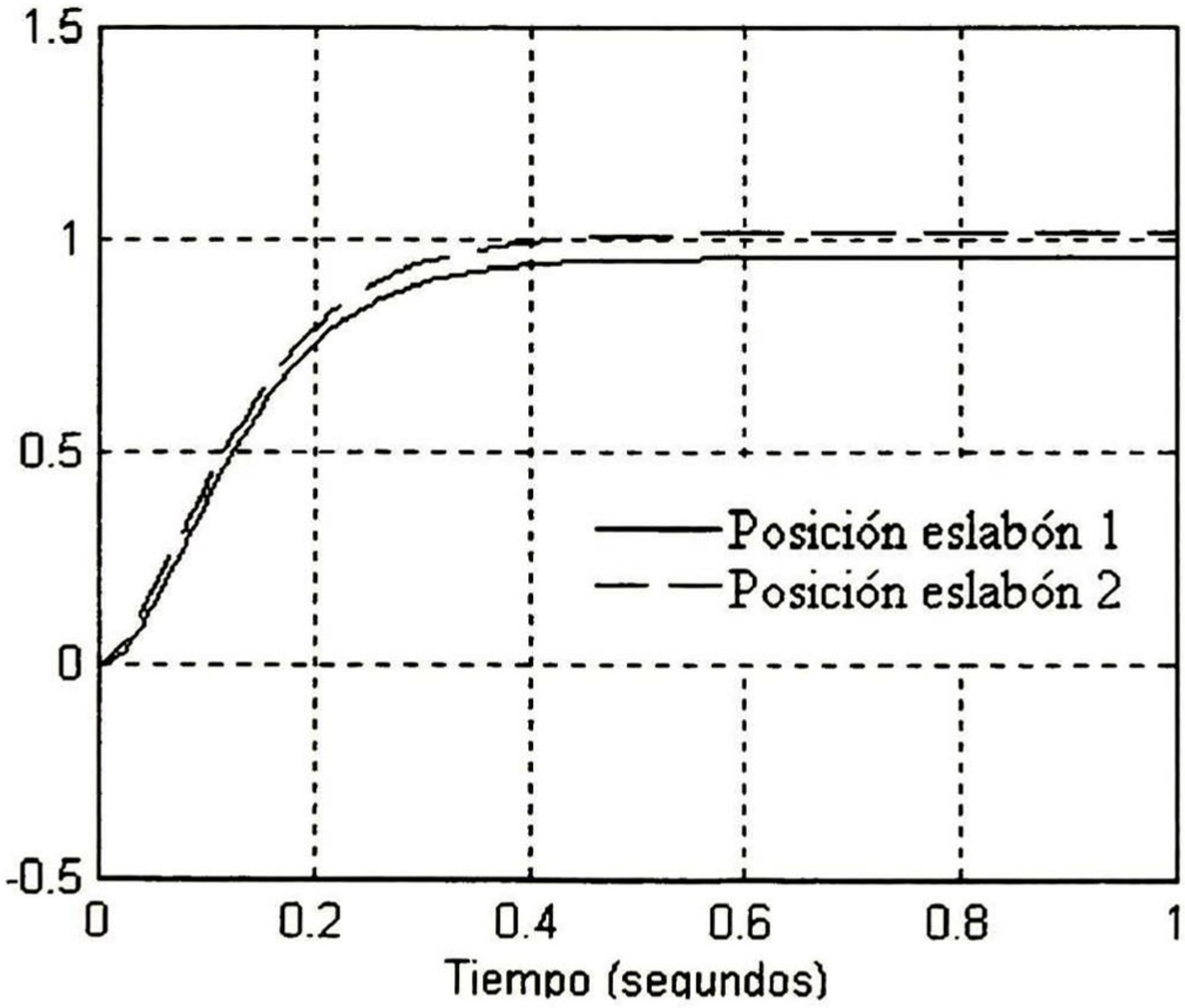


Figura 3.11 Posición de los eslabones

Si se incrementan las ganancias se obtiene una gran mejoría y no produce sobre paso cómo el controlador PD normal. El tiempo de estabilización es de 0.5 segundos, cómo se muestra en la Figura 3.11

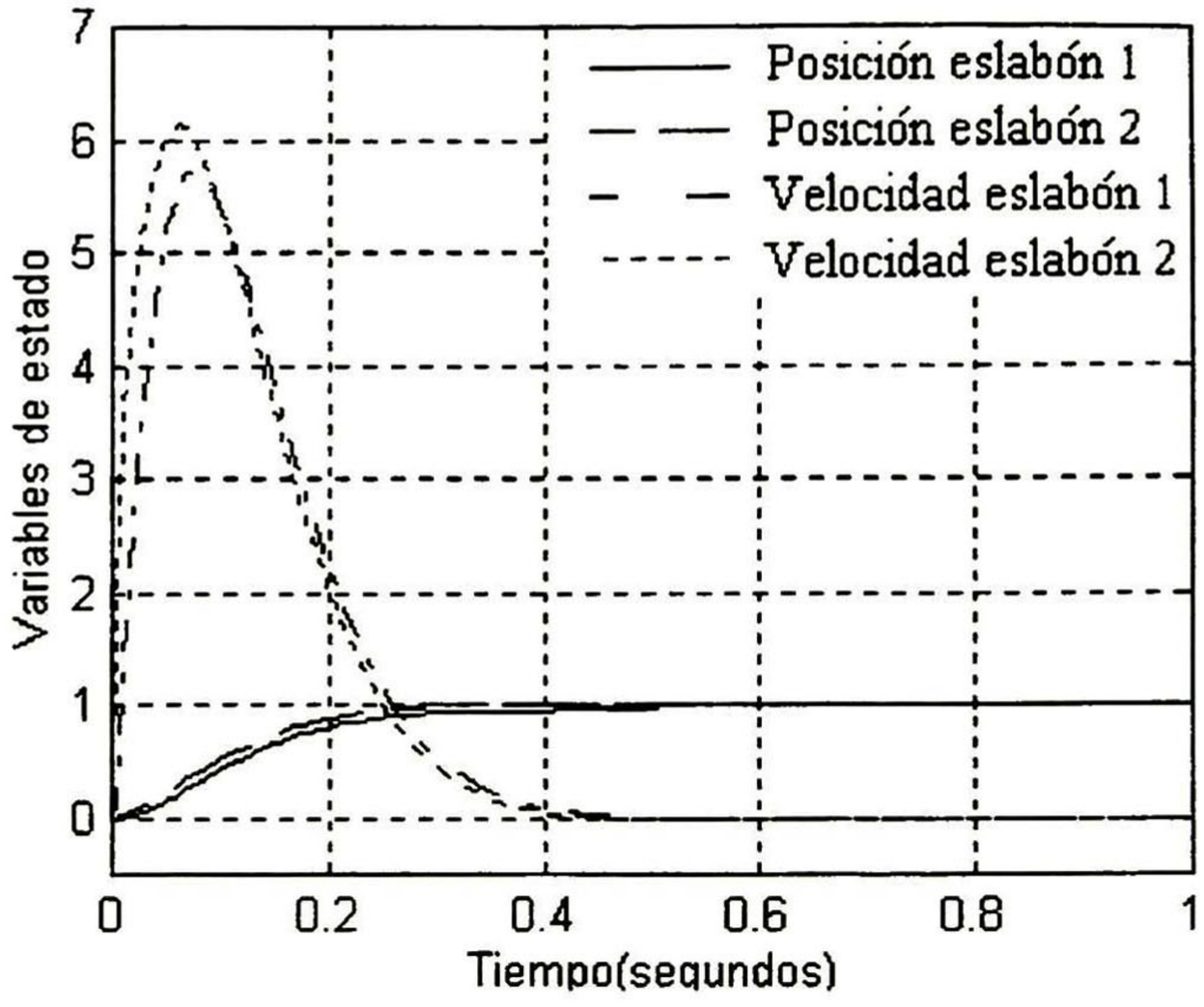


Figura 3.12 Posición y velocidad de los eslabones incrementando las ganancias

3.5 Comparación de PD Clásico y PD Difuso

En éste capítulo, inicialmente se discute la dinámica de un robot con dos grados de libertad completamente actuado; luego se propone un nuevo esquema de control PD difuso; se realiza el respectivo análisis de estabilidad y se ilustra la aplicabilidad del esquema propuesto por medio de ejemplos a nivel simulación.

Como se puede observar el controlador PD difuso tiene un mejor desempeño. Pueden manipularse los parámetros fácilmente para llegar a eliminar el error sin obtener sobrepasos.

CAPÍTULO 4
CONTROL DE ROBOTS SUBACTUADOS

En éste capítulo se presenta un nuevo esquema PD difuso para equilibrar verticalmente un tipo particular de robot subactuado; la aplicabilidad de éste esquema se valida tanto a nivel simulación como en tiempo real. Para obtener el seguimiento de trayectorias, se propone también un nuevo esquema que combina el regulador lineal con el PD difuso propuesto. Este nuevo esquema se valida también tanto a nivel simulación como en tiempo real.

4.1 Modelado del robot

El pendubot es un robot que consta de dos eslabones como el mencionado en la sección 3.11, pero con la diferencia que sólo uno de ellos es actuado; de ahí el nombre de robot subactuado[1]. Un esquema se presenta en la Figura 4.1

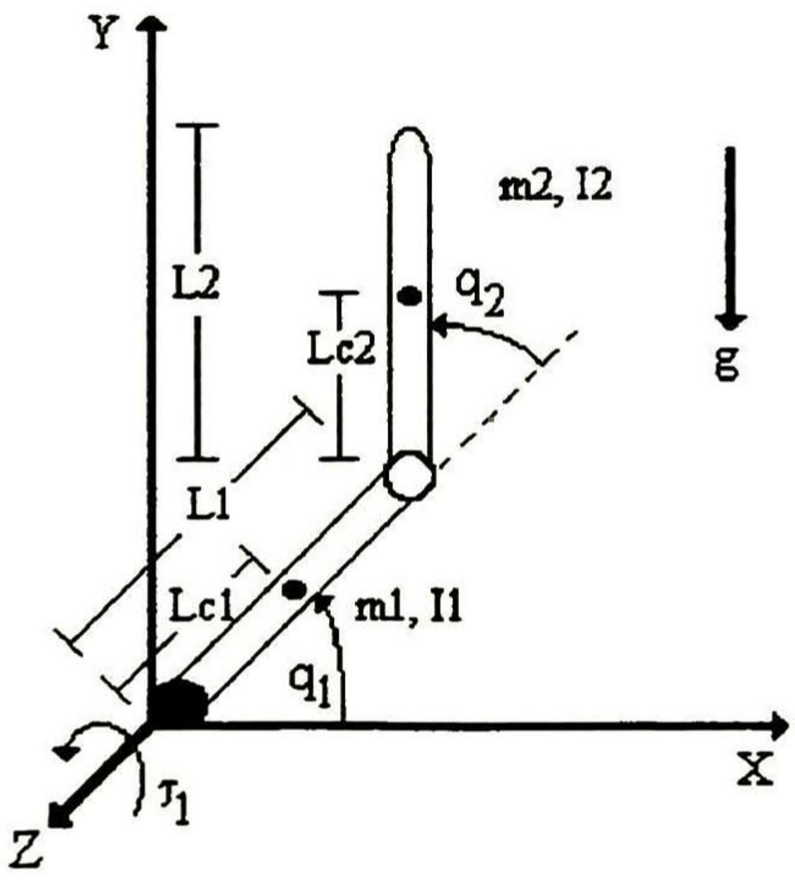


Figura 4.1 Esquema del pendubot

Este sistema obedece a las ecuaciones (3.23), con τ_2 idénticamente igual a cero, puesto que el segundo eslabón no cuenta con actuador. Las ecuaciones que describen éste sistema son:

$$D(q)=\begin{bmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{bmatrix},$$

$$C(q,\dot{q})=\begin{bmatrix} -2m_2 l_1 l_{c2}\dot{q}_2 sen(q_2) & -m_2 l_1 l_{c2}\dot{q}_2 sen(q_2) \\ m_2 l_1 l_{c2}\dot{q}_1 sen(q_2) & 0 \end{bmatrix},$$

$$G(q)=\begin{bmatrix} G_1 \\ G_2 \end{bmatrix}=\begin{bmatrix} m_1 g l_{c1} cos(q_1)+m_2 g l_1 cos(q_1)+m_2 g l_{c2} cos(q_1+q_2) \\ m_2 g l_{c2} cos(q_1+q_2) \end{bmatrix} \qquad (4.1)$$

$$\tau=\begin{bmatrix} \tau_1 \\ 0 \end{bmatrix}$$

$$D_{11}=m_i l_{c1}^2+m_2\left(l_1^2+l_{c2}^2+2l_1 l_{c2} cos(q_2)\right)+I_{zz1}+I_{zz2}$$

$$D_{12}=m_2\left(l_{c2}^2+l_1 l_2 cos(q_2)\right)+I_{zz2}$$

$$D_{21}=D_{12}$$

$$D_{22}=m_2 l_{c2}^2+I_{zz2}$$

donde D(q) es la matriz de inercia, $C(q,\dot{q})$ es la matriz de Coriolis y $G(q)$ es el vector con los términos de gravedad.

4.1.1 Representación en variables de estado

El modelo dinámico del Pendubot puede representarse en variables de estado haciendo la siguiente asignación de variables:

$$x_1 = q_1, x_2 = q_2, x_3 = \dot{q}_1, x_4 = \dot{q}_2, u = \tau_1 \tag{4.2}$$

El modelo en variables de estado se establece como:

$$\dot{x}(t) = f(x) + g(x)u(t) \tag{4.3}$$

con:

$$f(x) = \begin{bmatrix} x_3 \\ x_4 \\ \dfrac{D_{22}}{D_{11}D_{22} - D_{12}^2}\left\{\dfrac{D_{12}C_2}{D_{22}} + \dfrac{D_{12}G_2}{D_{22}} - C_1 - G_1\right\} \\ \dfrac{-D_{12}}{D_{11}D_{22} - D_{12}^2}\left\{\dfrac{D_{12}C_2}{D_{22}} + \dfrac{D_{12}G_2}{D_{22}} - C_1 - G_1\right\} - \dfrac{C_2}{D_{22}} - \dfrac{G_2}{D_{22}} \end{bmatrix}$$

$$g(x) = \begin{bmatrix} 0 \\ 0 \\ \dfrac{D_{22}}{D_{11}D_{22} - D_{12}^2} \\ \dfrac{-D_{12}}{D_{11}D_{22} - D_{12}^2} \end{bmatrix} \tag{4.4}$$

donde:

$$\begin{aligned} C_1 &= -2m_2 l_1 l_{c2} x_3 x_4 sen(x_2) - m_2 l_1 l_{c2} x_4^2 sen(x_2) \\ C_2 &= m_2 l_1 l_{c2} x_3^2 sen(x_2) \end{aligned} \tag{4.5}$$

D_{ij} y G_k se calculan por medio de las ecuaciones (3.14) y (4.1).

Al sustituir los valores de las constantes del sistema mecánico real, que se encuentran en la Tabla C.1 (Apéndice C), la ecuación anterior queda como

$$g(x)=\begin{bmatrix} 0 \\ 0 \\ \dfrac{0.029094}{0.00124209-0.000150416cos(2x_2)} \\ \dfrac{-0.029094-0.0173445cos(x_2)}{0.00124209-0.000150416cos(2x_2)} \end{bmatrix} \tag{4.6}$$

$$f(x)=\begin{bmatrix} x_3 \\ x_4 \\ 0.029094\left[\dfrac{-1.39803\cos(x_1)+0.187935cos(x_1+2x_2)+0.0173445x_3^2 sen(x_2)}{0.00124209-0.000150416cos(2x_2)}\right] \\ +0.029094\left[\dfrac{0.034689x_3x_4 sen(x_2)+0.0173445x_4^2 sen(x_2)+0.00517002x_3^2 sen(2x_2)}{0.00124209-0.0001504cos(2x_2)}\right] \\ -(0.029094+0.0173445cos(x_2))\left[\dfrac{-1.39803cos(x_1)+0.187935cos(x_1+2x_2)}{0.00124209-0.000150416cos(2x_2)}\right] \\ -(0.029094+0.0173445cos(x_2))\left[\dfrac{0.0173445x_3^2 sen(x_2)+0.0346891x_3x_4 sen(x_2)}{0.00124209-0.000150416cos(2x_2)}\right] \\ -(0.029094+0.0173445cos(x_2))\left[\dfrac{0.0173445x_4^2 sen(x_2)+0.00517002x_3^2 sen(2x_2)}{0.00124209-0.000150416cos(2x_2)}\right] \\ -21.6708cos(x_1+x_2)-0.596156x_3^2 sen(x_2) \end{bmatrix}$$

4.1.2 Puntos de equilibrio

Para éste sistema, es importante encontrar los puntos de equilibrio[1]; esto es, encontrar $(x,y)=(x^0,y^0)$ tales que $f(x^0,y^0)\equiv 0$. A partir de la condición $\dot{x}_1=0$ y $\dot{x}_2=0$, se tiene que:

$$x_3=0 \text{ y } x_4=0 \tag{4.7}$$

con estos términos vemos que C_1=0 y C_2=0 (ecuacion 4.5) por lo que:

$$\dot{x}_3 = \frac{D_{22}}{D_{11}D_{22} - D_{12}^2}\left\{\frac{D_{12}G_2}{D_{22}} - G_1 + u*\right\}$$
$$\dot{x}_4 = \frac{-D_{12}}{D_{11}D_{22} - D_{12}^2}\left\{\frac{D_{12}G_2}{D_{22}} - G_1 + u*\right\} - \frac{G_2}{D_{22}} \qquad (4.8)$$

De la ecuacion (4.8) se observa que para que $\dot{x}_3 = 0$, se necesita que:

$$\frac{D_{12}G_2}{D_{22}} - G_1 + u* = 0 \qquad (4.9)$$

ya que D_{11}, D_{22} y D_{12} son diferentes de cero (3.14).

Si se sustituye la ecuacion (4.9) en la ecuacion (4.8) para el termino de $\dot{x}_4$ se tiene que G_2=0, por lo que:

$$G_2 = m_2 g l_{c2} cos(q_1 + q_2) = 0 \qquad (4.10)$$

Para cumplir con esta ecuación $cos(x_1+x_2)=0$, por lo que los puntos de equilibrio son aquellos que cumplen con:

$$x_1 + x_2 = n\left(\frac{\pi}{2}\right) = n90°; \forall n = \pm 1, \pm 3, \pm 5 \ldots \qquad (4.11)$$

4.2 Estabilización con PD difuso

El esquema del controlador PD propuesto en el Capítulo 3 para un sistema totalmente actuado se aplica al Pendubot, suprimiendo la señal que va al segundo eslabón. Este esquema se presenta en la Figura 4.2

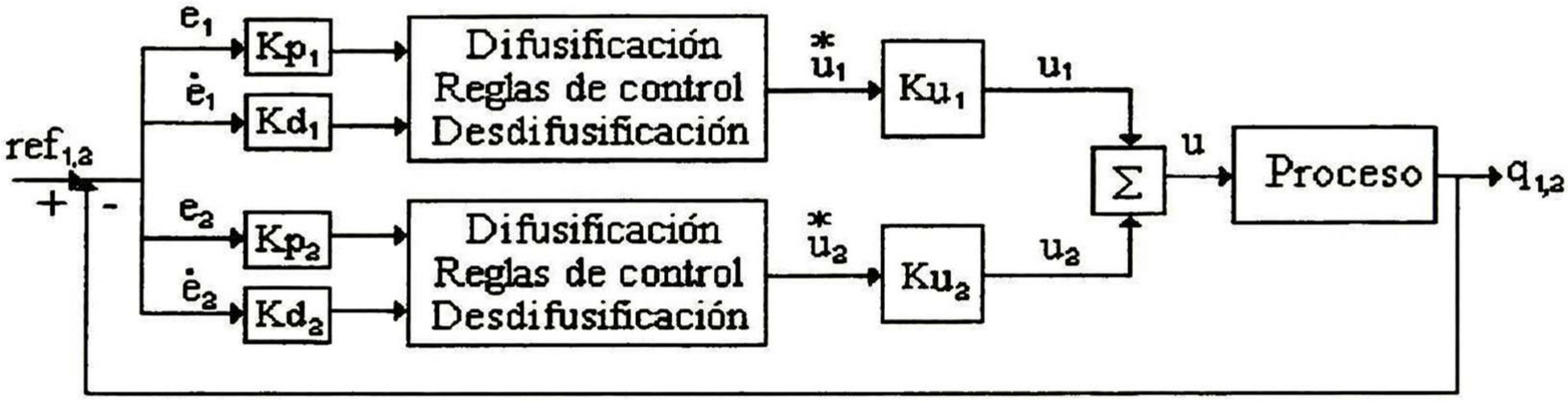


Figura 4.2 Esquema del control del Pendubot

Como se muestra en la Figura 4.2, el controlador PD difuso emplea dos subcontroladores difusos, como los discutidos en la Sección 2.3. El algoritmo se implementó en Matlab, así como en tiempo real.

La estrategia que se utilizó fue emplear un primer control PD difuso para que el sistema del Pendubot llegue a una vecindad cercana a la posición vertical y después conmutar[4] a otro controlador PD difuso para garantizar que el sistema en lazo cerrado sea estable y se mantenga en la posición deseada. Este esquema de control general se muestra se muestra en la Figura 4.3

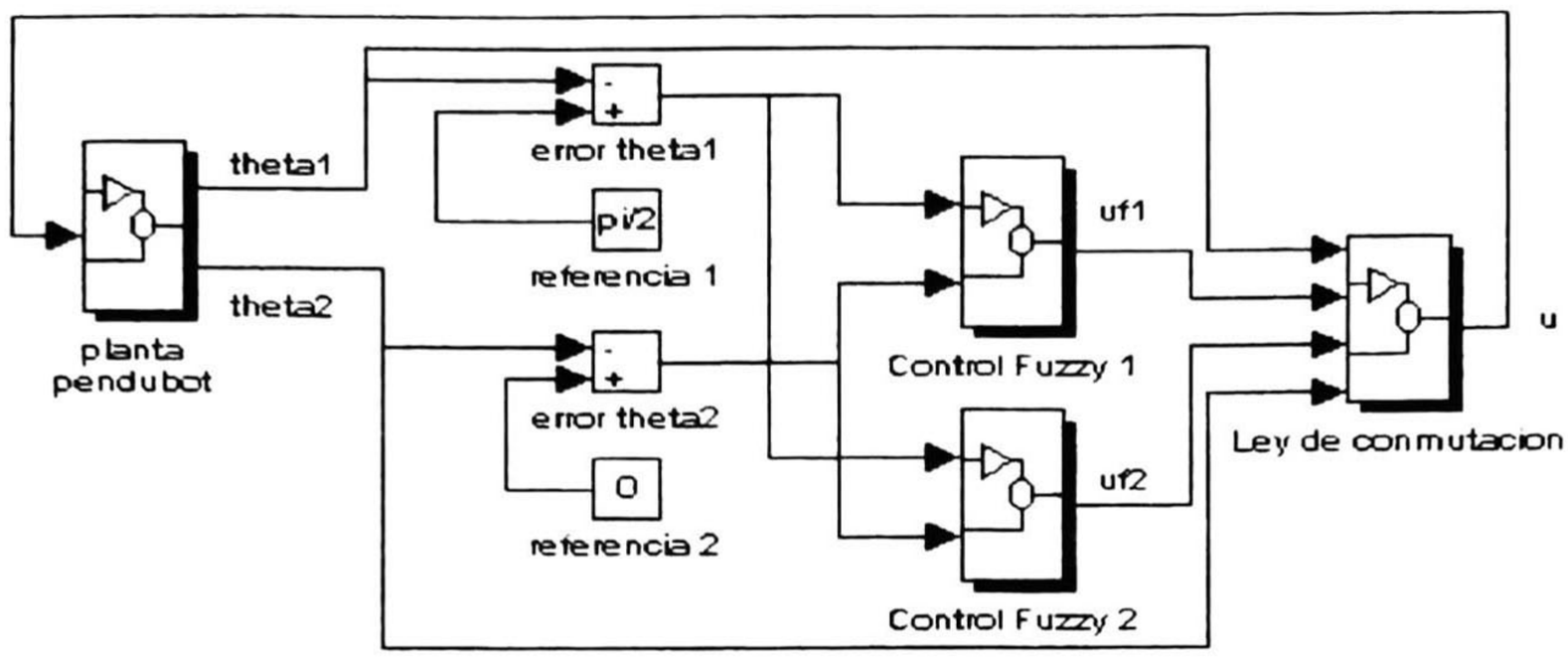


Figura 4.3 Esquema general del Pendubot con el Controlador difuso

4.2.1 Implementación en MATLAB

Para hacer la simulación del control del Pendubot se utilizó MATLAB y SIMULINK como plataforma de simulación. El propósito del control es llevar al Pendubot a los ángulos q_1=90° y q_2=0°, que corresponde a la posición vertical.

El esquema de control Figura 4.3 está constituido por cuatro bloques principales: el modelo no lineal del pendubot, el controlador difuso desestabilizador, el controlador difuso estabilizador y la ley de conmutación. El diagrama del modelo del pendubot se muestra en la Figura 4.4

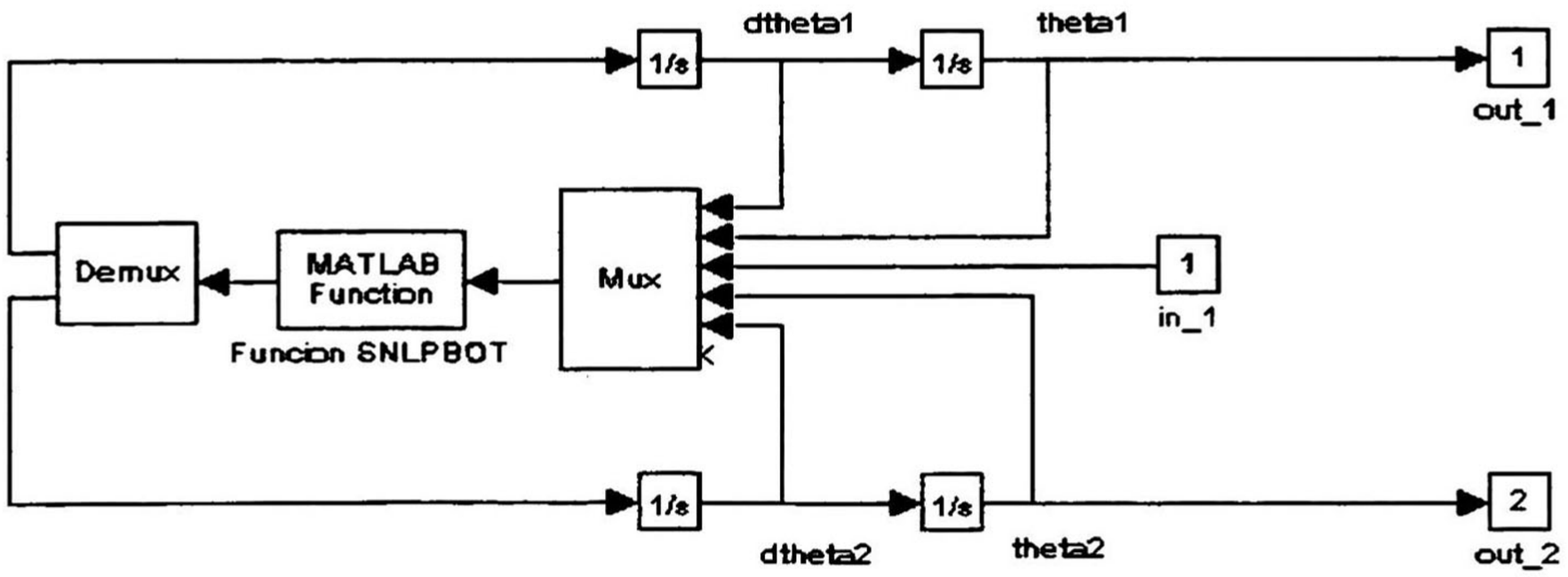


Figura 4.4 Esquema del pendubot

En la Figura 4.4, se incluye la interface de Simulink con Matlab, para llamar a la función presentada en el Apéndice C.1, que simula la dinámica del robot haciendo la evaluación de la ecuación (4.12). Al introducir a esta función los estados (q_1, q_2 , $\dot{q}_1$, $\dot{q}_2$) y la señal de control (u_1), se obtienen las señales de $\ddot{q}_1$ y $\ddot{q}_2$, que deben de ser integradas para obtener los estados del pendubot:

$$\ddot{q} = -D(q)^{-1}\left(C(q,\dot{q})\dot{q} + G(q) - \tau\right) \tag{4.12}$$

En las Figuras 4.5 y 4.6 se muestran los diagramas a bloques del controlador difuso excitador y estabilizador respectivamente, con sus ganancias correspondientes. El controlador recibe las señales de error generando la derivada del error de cada uno de los eslabones, para así evaluar la función presentada en el Apéndice C.2, misma que es utilizada para ambos controladores.

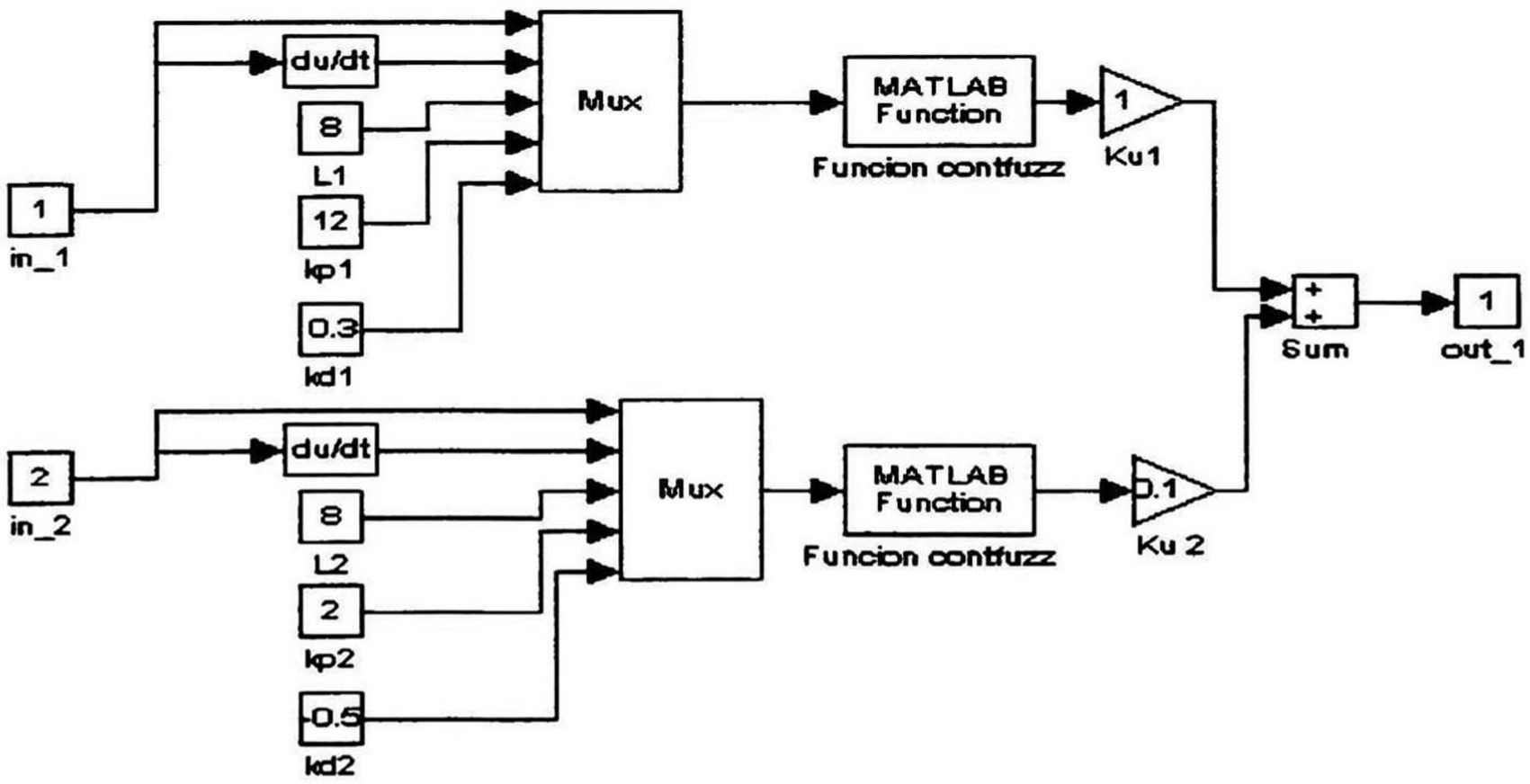


Figura 4.5 Controlador difuso desestabilizador

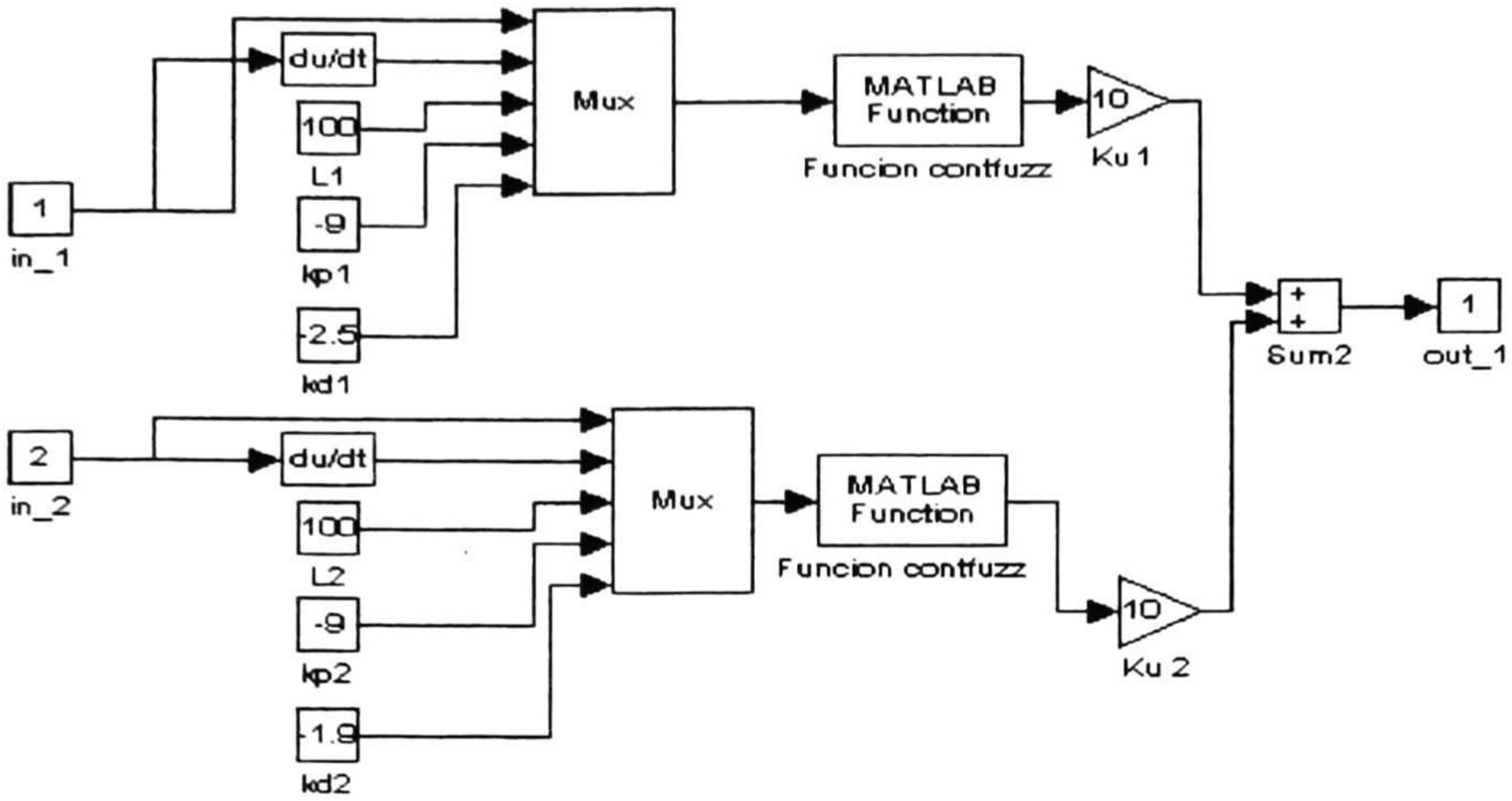


Figura 4.6 Controlador difuso estabilizador

El bloque de la ley de conmutación se presenta en la Figura 4.7. Esta recibe cuatro señales de entrada; las dos primeras son los ángulos del primer y segundo eslabón respectivamente y las otras dos señales son la acción de control generada por el controlador excitador y el estabilizador. Con los valores de estas señales se evalúa la función presentada en el Apéndice C.3, obteniendo a la salida la señal de control generada por el controlador desestabilizador o por el controlador estabilizador, dependiendo si la distancia es mayor o es menor a un cierto valor predeterminado, como se formula en (4.13)

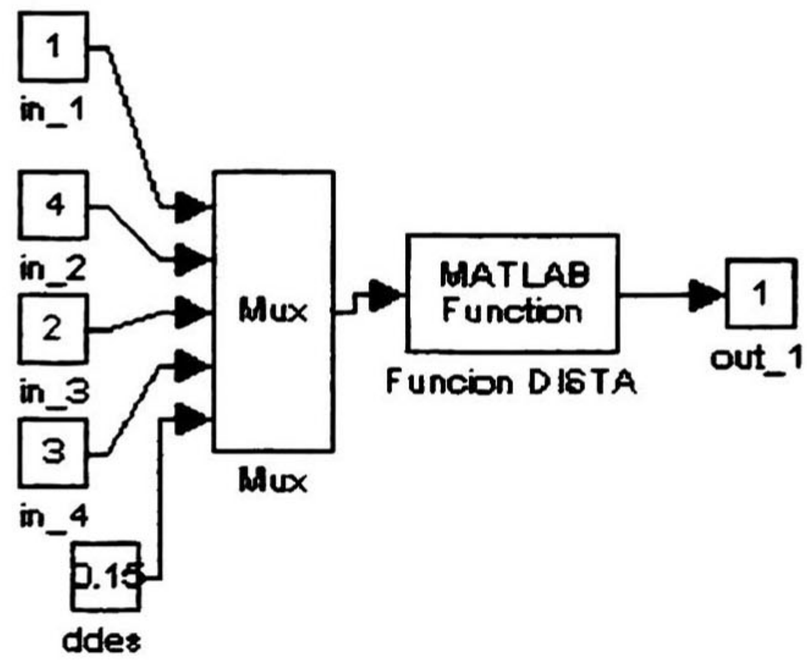


Figura 4.7 Bloque de conmutación

La ley de conmutación [9] se efectúa en función de la distancia del extremo del eslabón exterior al punto deseado, como se ilustra en la Figura 4.8. Cuando la distancia es menor o igual a una distancia seleccionada, se hace la conmutación al segundo controlador PD difuso. La ecuación para el cálculo de la distancia es:

$$d = \sqrt{\left((L_1 + L_2 - L_1 sin(q_1) - L_2 sin(q_1 + q_2)\right)^2 + \left(L_1 \cos(q_1) + L_2 \cos(q_1 + q_2)\right)^2} \quad (4.13)$$

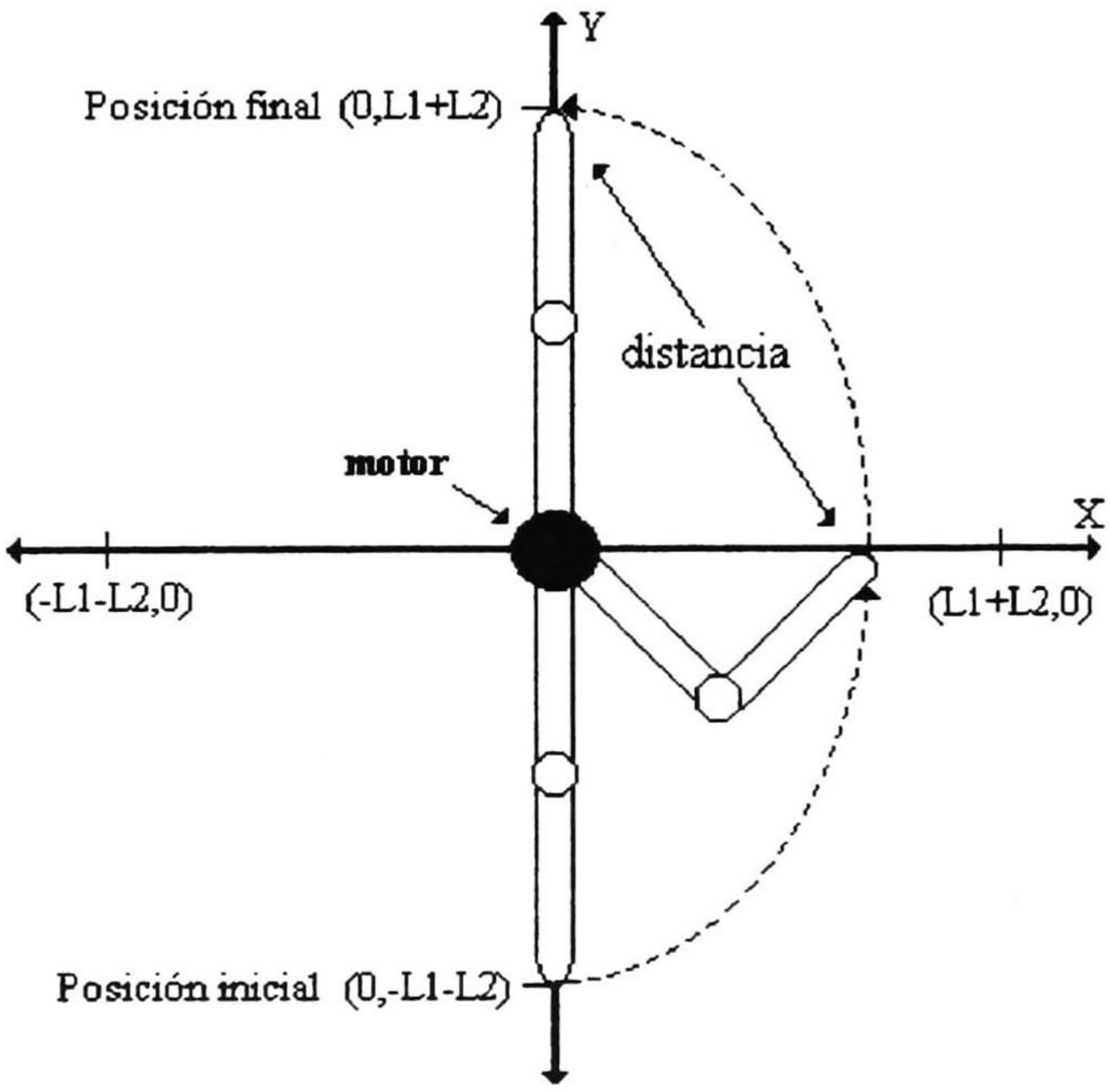


Figura 4.8 Distancia para llegar al punto final

4.2.1 Resultados Obtenidos

Al hacer el control se dificultó un poco la obtención de los parámetros para garantizar que el sistema fuese estable en el punto de equilibrio deseado. El primer parámetro que se determinó fue L, como se presentó en el capítulo anterior, esté debe de tener un valor grande para garantizar la estabilidad. Los demás parámetros fueron determinados por medio de experimentación. El objetivo de control fue partir

de la condición inicial q_1 = -90° y q_2 = 0° (brazo totalmente caído) y llevar al pendubot al punto q_1 = 90° y q_2 = 0° (brazo totalmente levantado). Los valores de las ganancias para éste controlador se encuentran en la Tabla C.2 (Apéndice C).

En las Figuras 4.9 y 4.10, se presentan la evolución de los ángulos q_1 y q_2 respectivamente. Las Figura 4.11 y 4.12 presentan la señal de error para el primer y segundo eslabón. La Figura 4.13 muestra la señal de control; ésta cumple con la restricción que el par de entrada sea menor a ±10N/m.

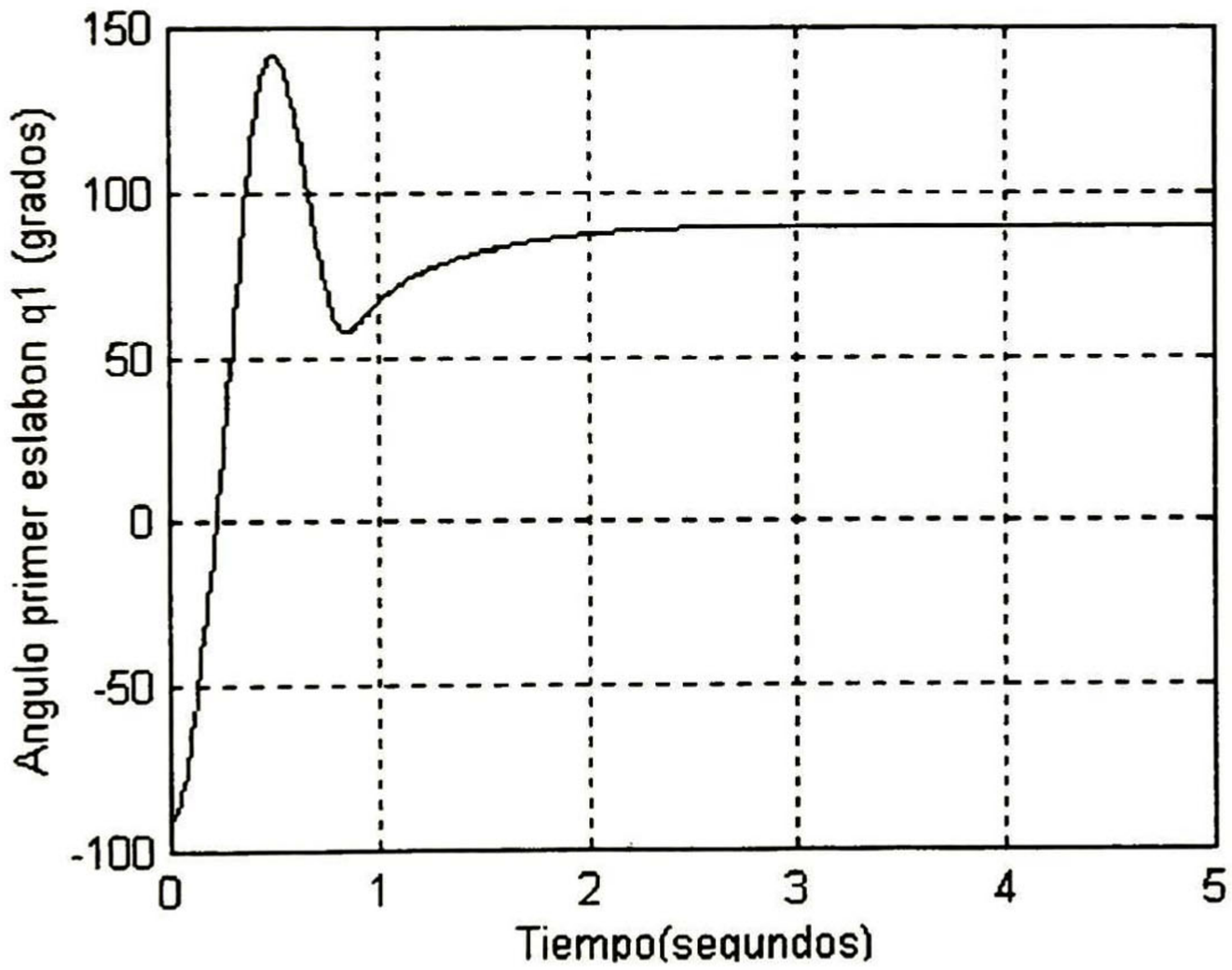


Figura 4.9 Posición angular del primer eslabón (q_1)

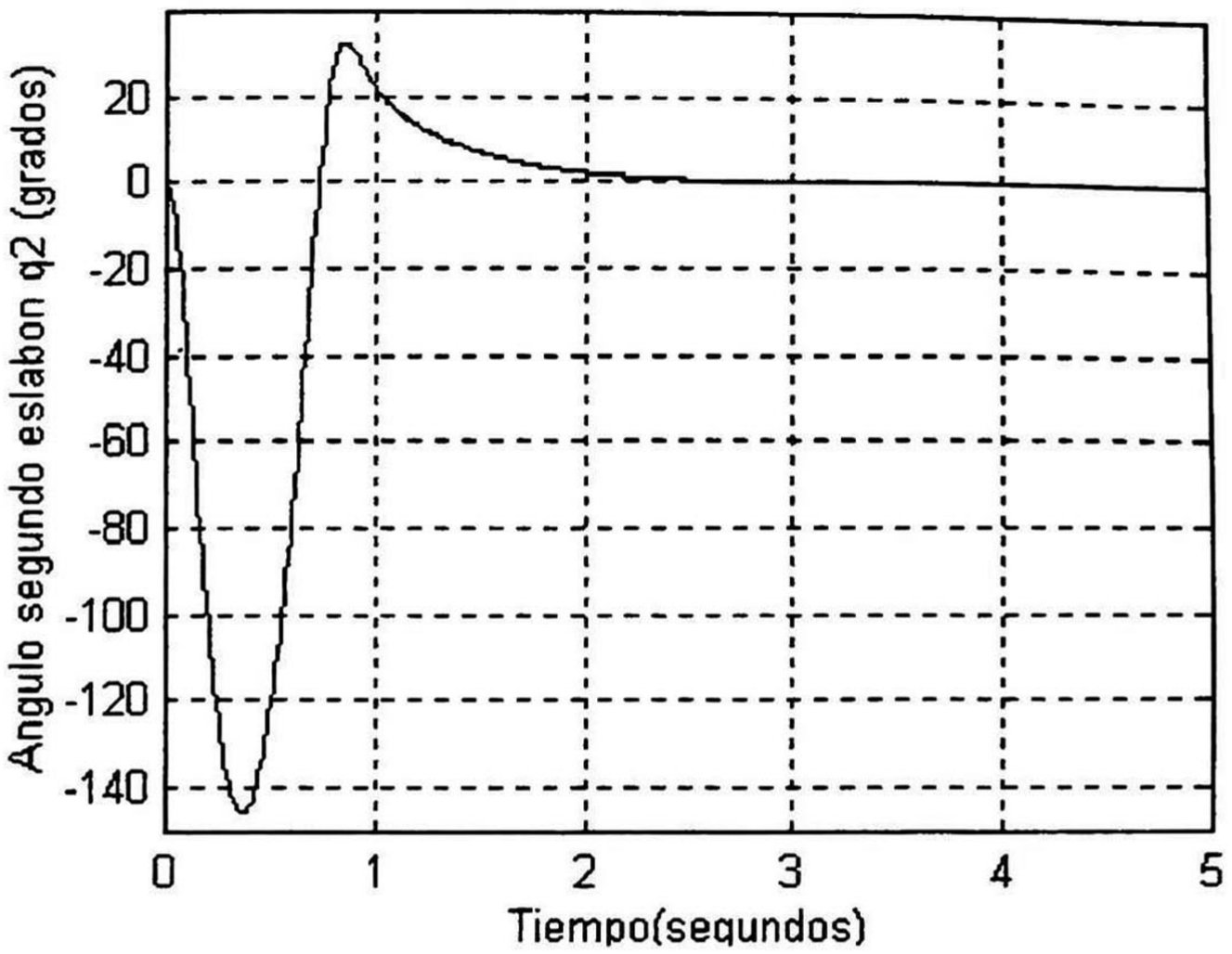


Figura 4.10 Posición angular del segundo eslabón ($q2$)

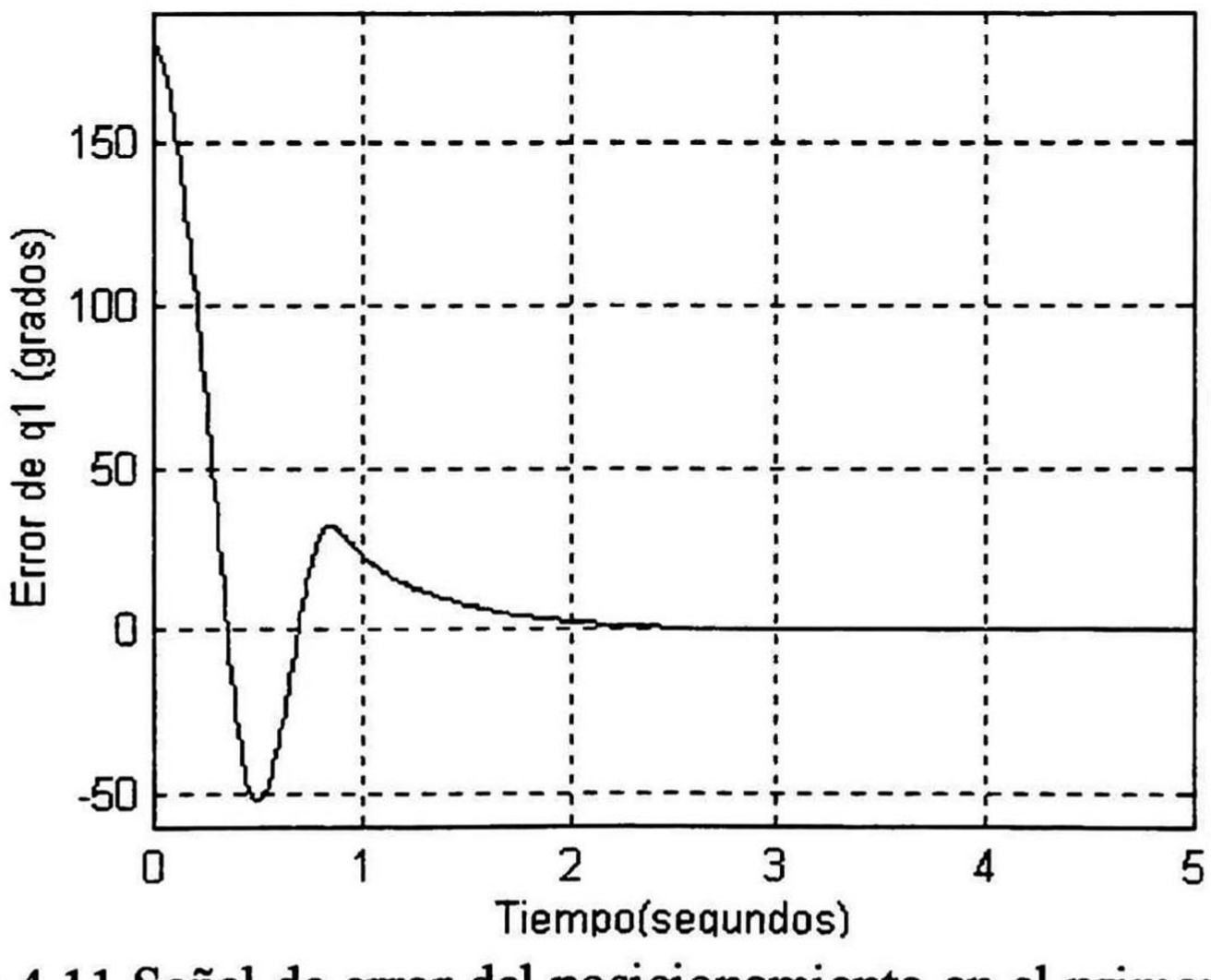


Figura 4.11 Señal de error del posicionamiento en el primer eslabón

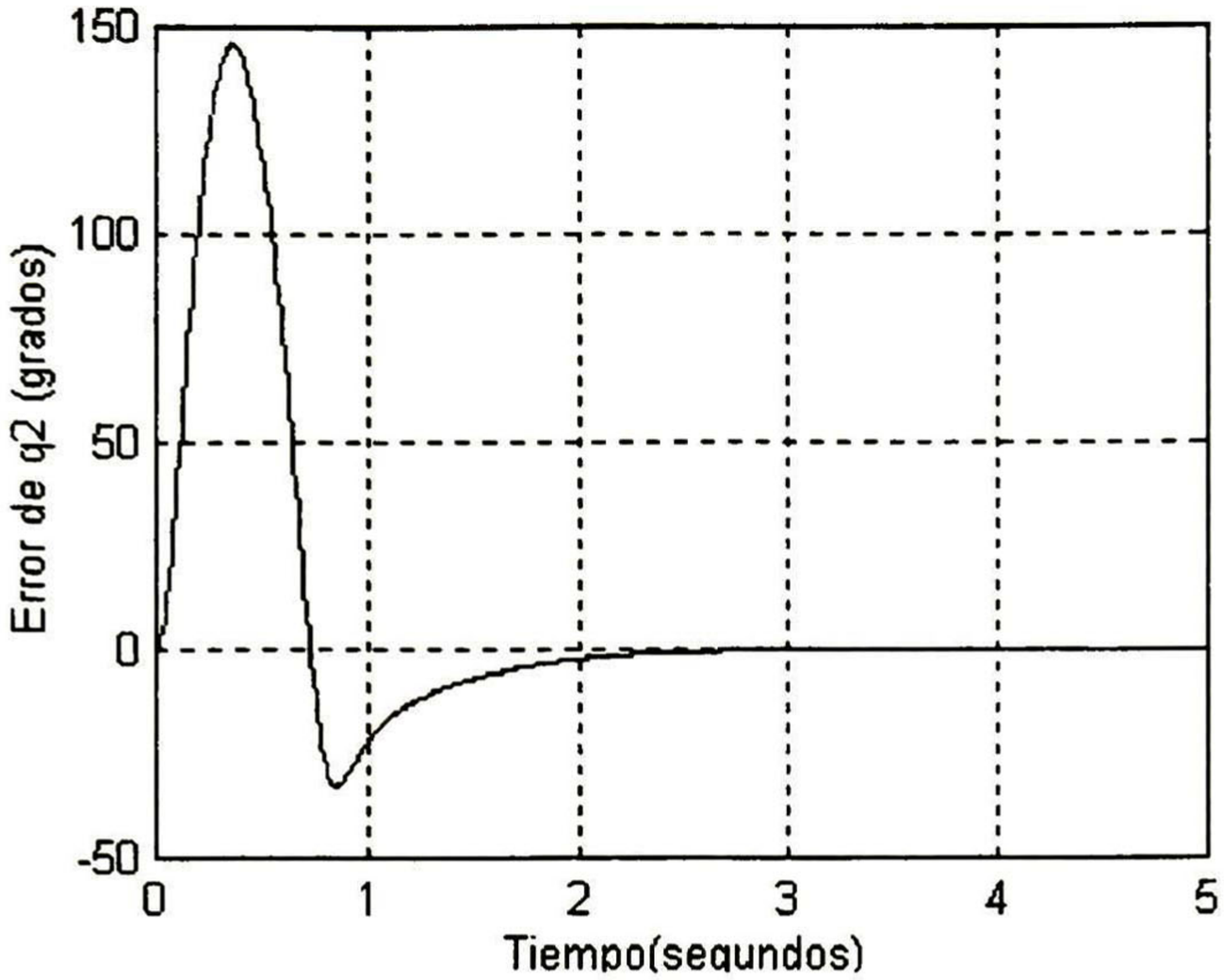


Fig. 4.12 Señal de error del posicionamiento en el segundo eslabón

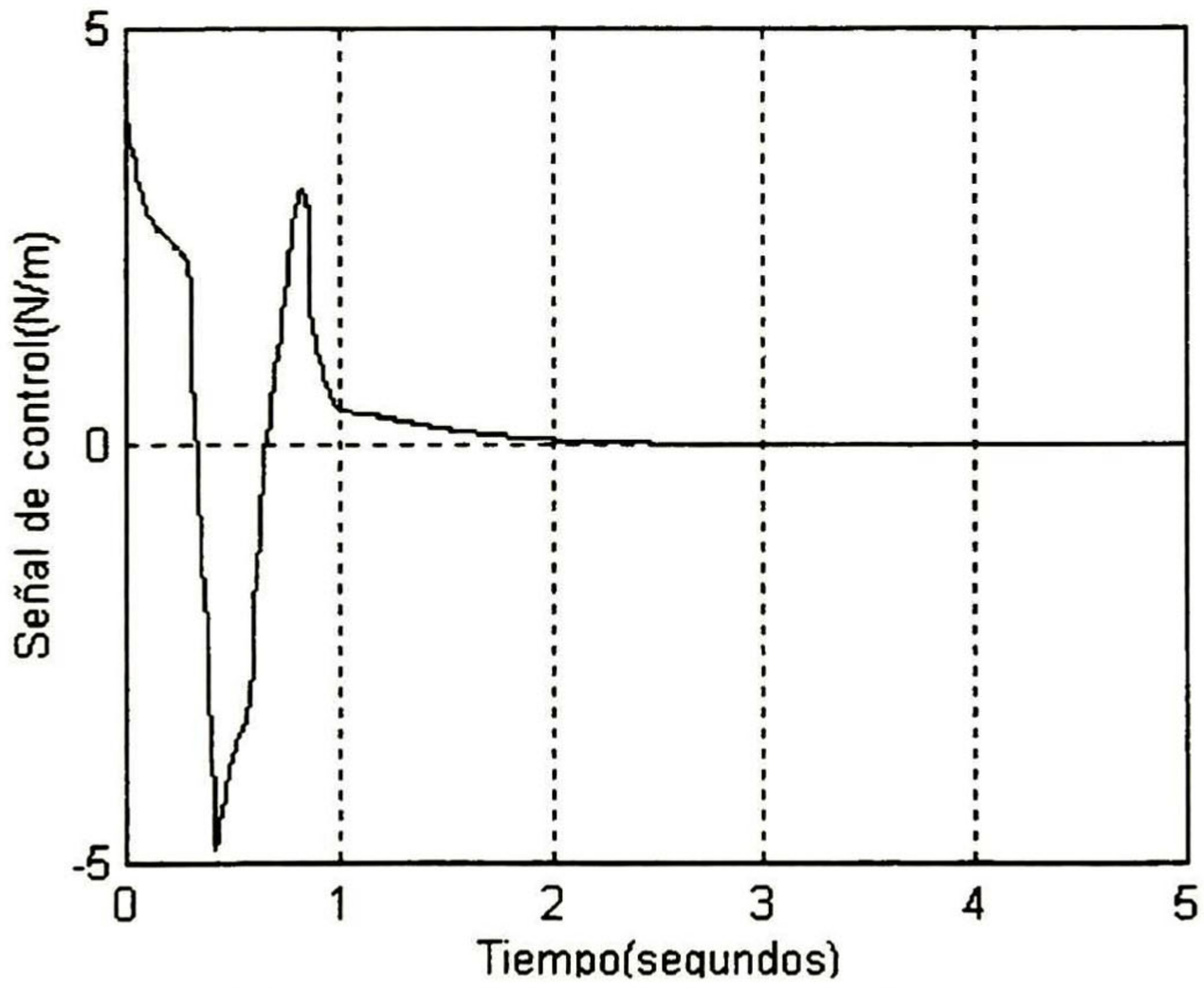


Figura 4.13 Señal de control (u_1)

Con los parámetros ya determinados del controlado PD difuso, en donde se experimentó introduciendo una perturbación tipo impulsional al eslabón exterior, como se observa en la Figura 4.14 y 4.15 que muestran la evolución de los ángulos q_1 y q_2 respectivamente; el sistema es capaz de volver a la posición vertical aunque

se haya introducido una perturbación al segundo eslabón. La Figura 4.16 muestra la señal de entrada.

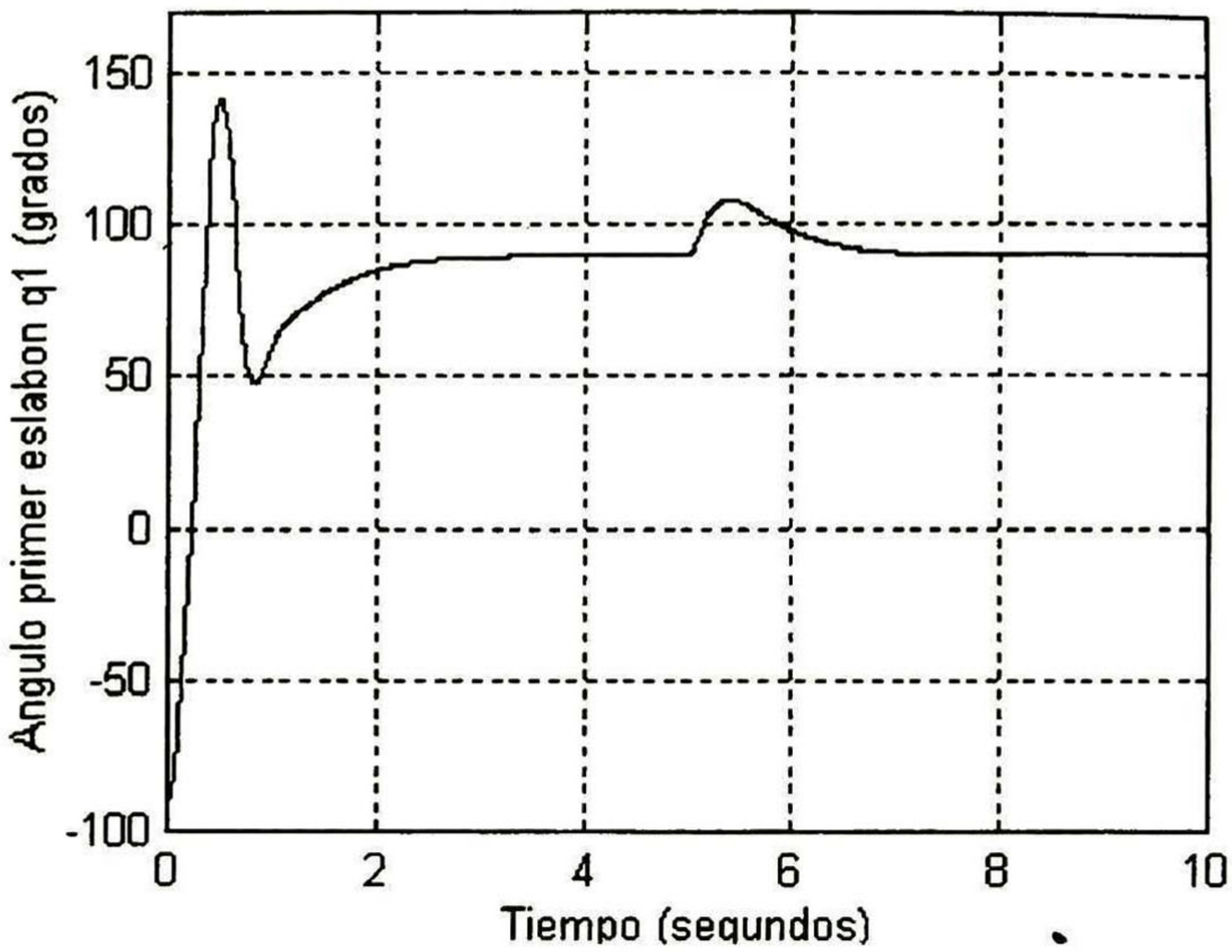


Figura 4.14 Posición angular del primer eslabón (q_1)

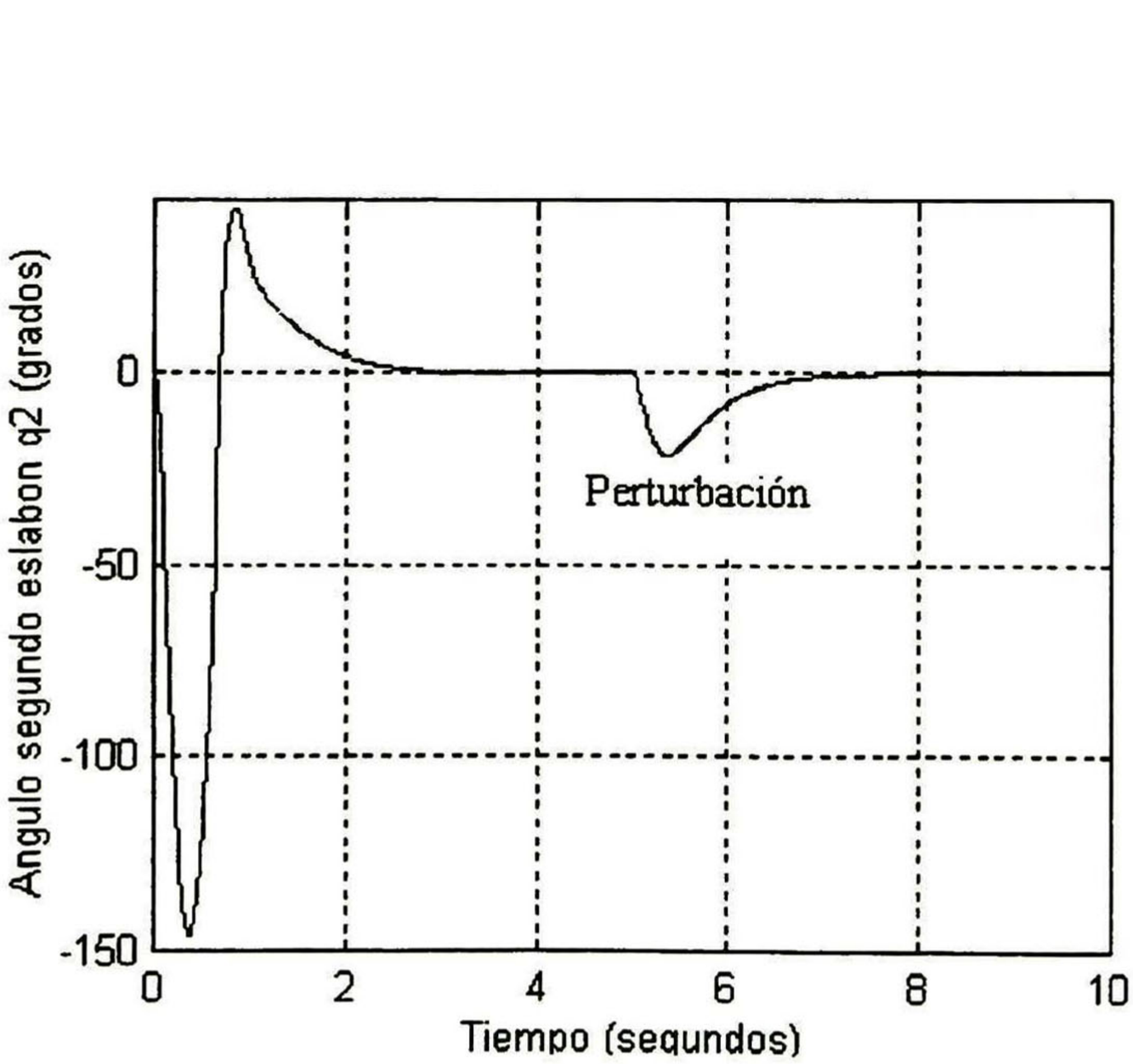


Figura 4.15 Posición angular del segundo eslabón (q_2)

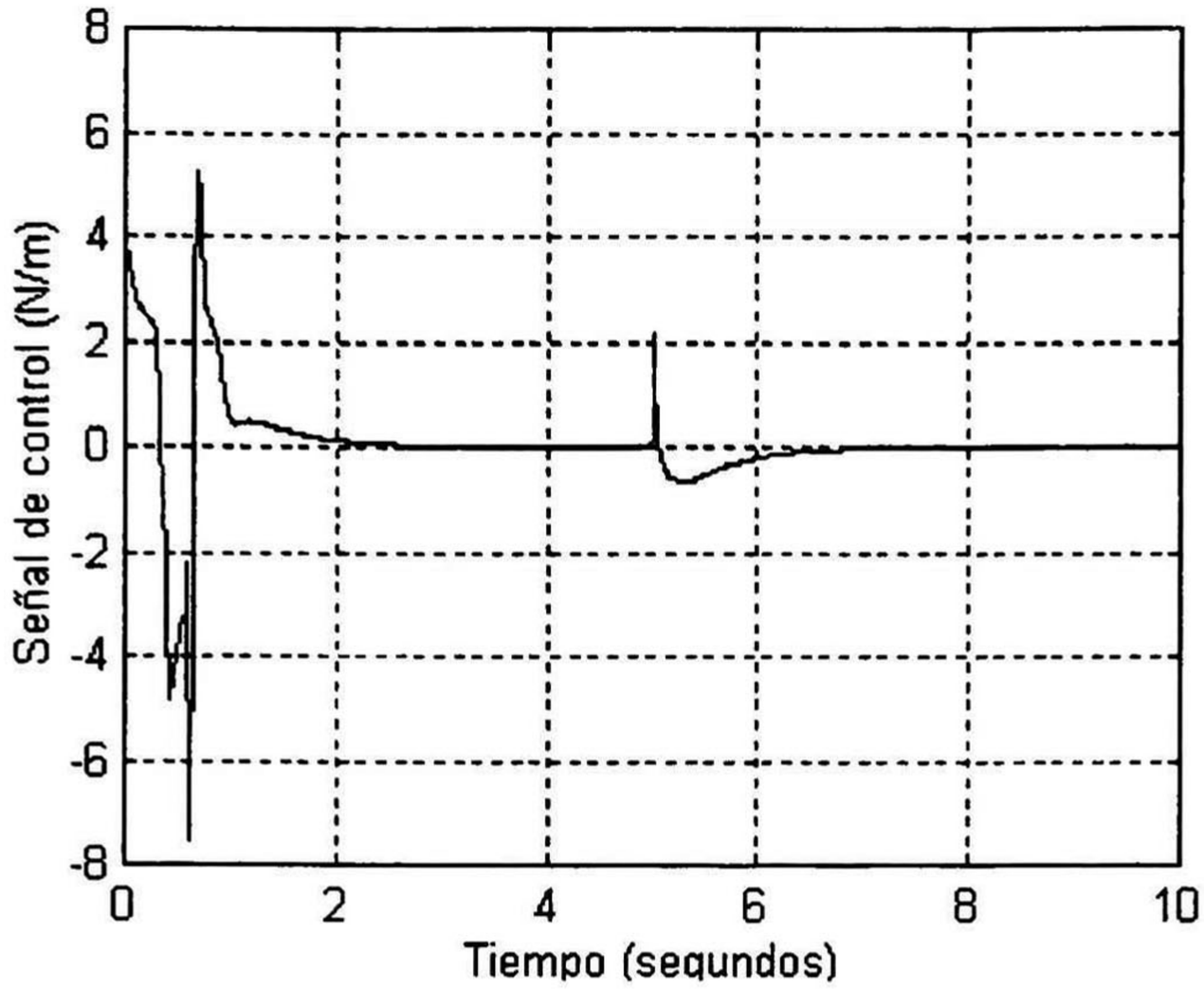


Figura 4.16 Señal de control (u_1)

Otro experimento que se llevó a cabo, fue hacer un cambio lineal de la masa del segundo eslabón (externo) con el fin de observar la robustez del controlador PD difuso. La Figura 4.17 y 4.18 presentan la evolución de los ángulos q_1 y q_2 respectivamente, donde se observa que el sistema llega a la posición deseada. La Figura 4.19 presenta el cambio de masa lineal del segundo eslabón y la figura 4.20 muestra la señal de control.

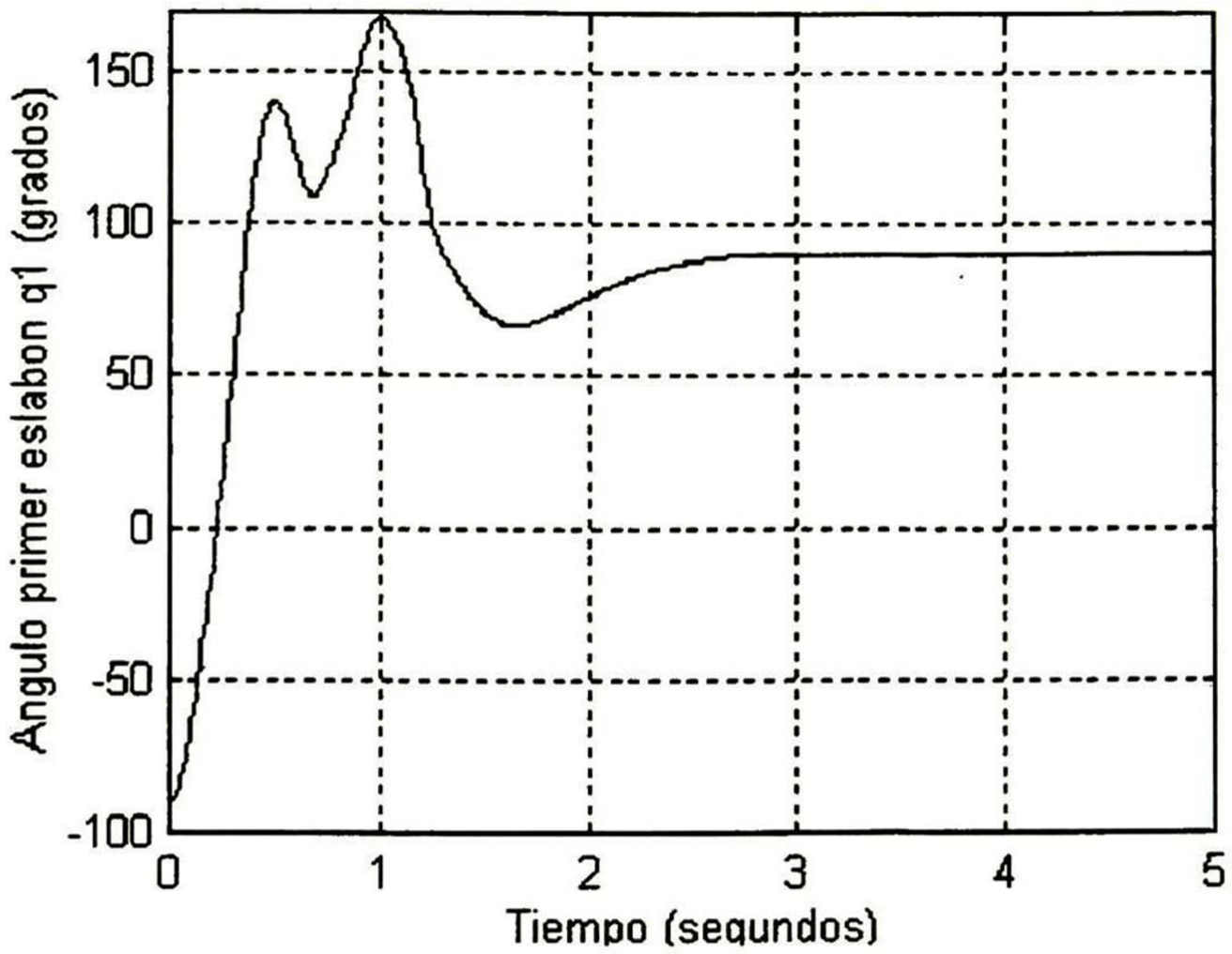


Figura 4.17 Posición angular del primer eslabón (q_1)

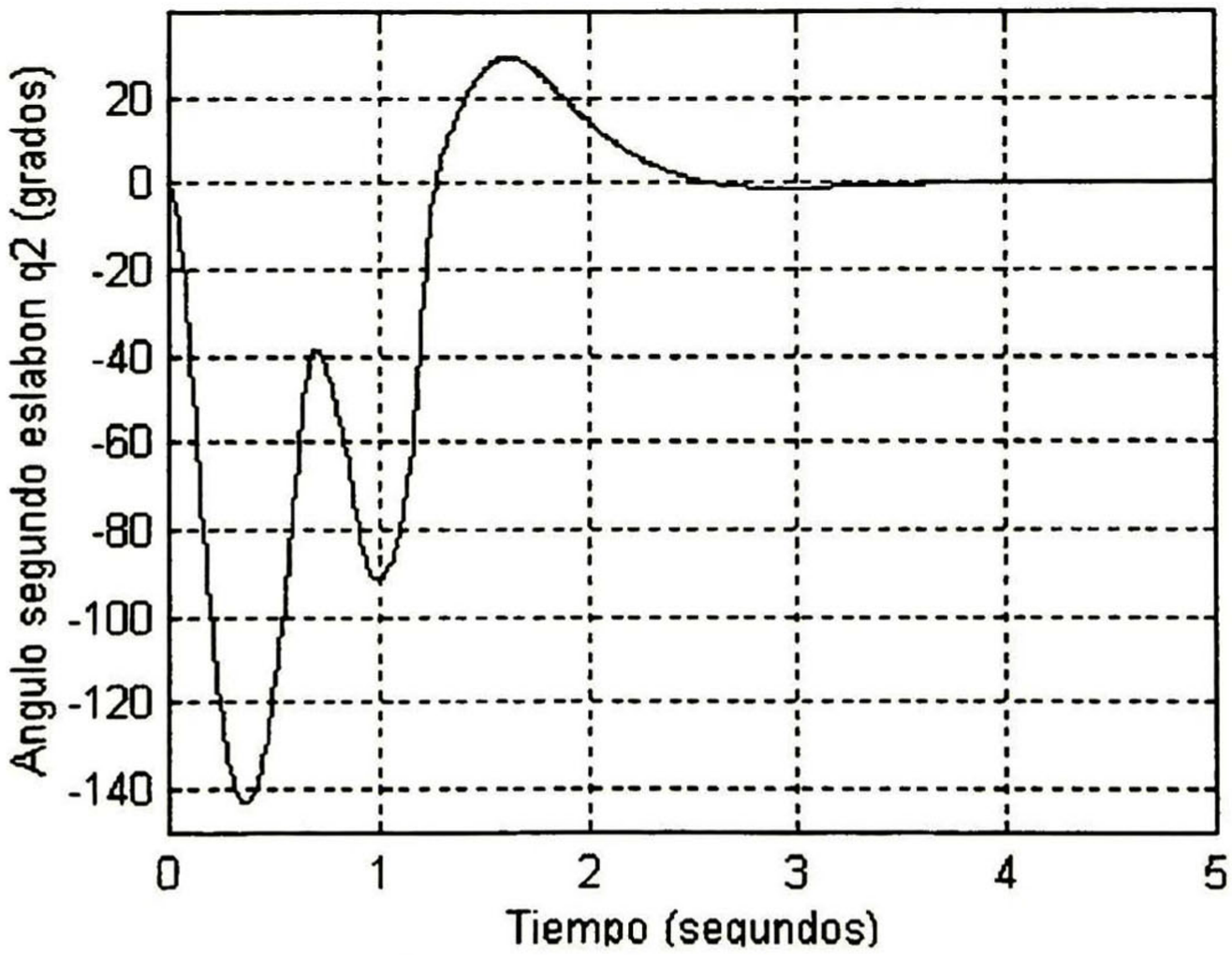


Figura 4.18 Posición angular del segundo eslabón (q_2)

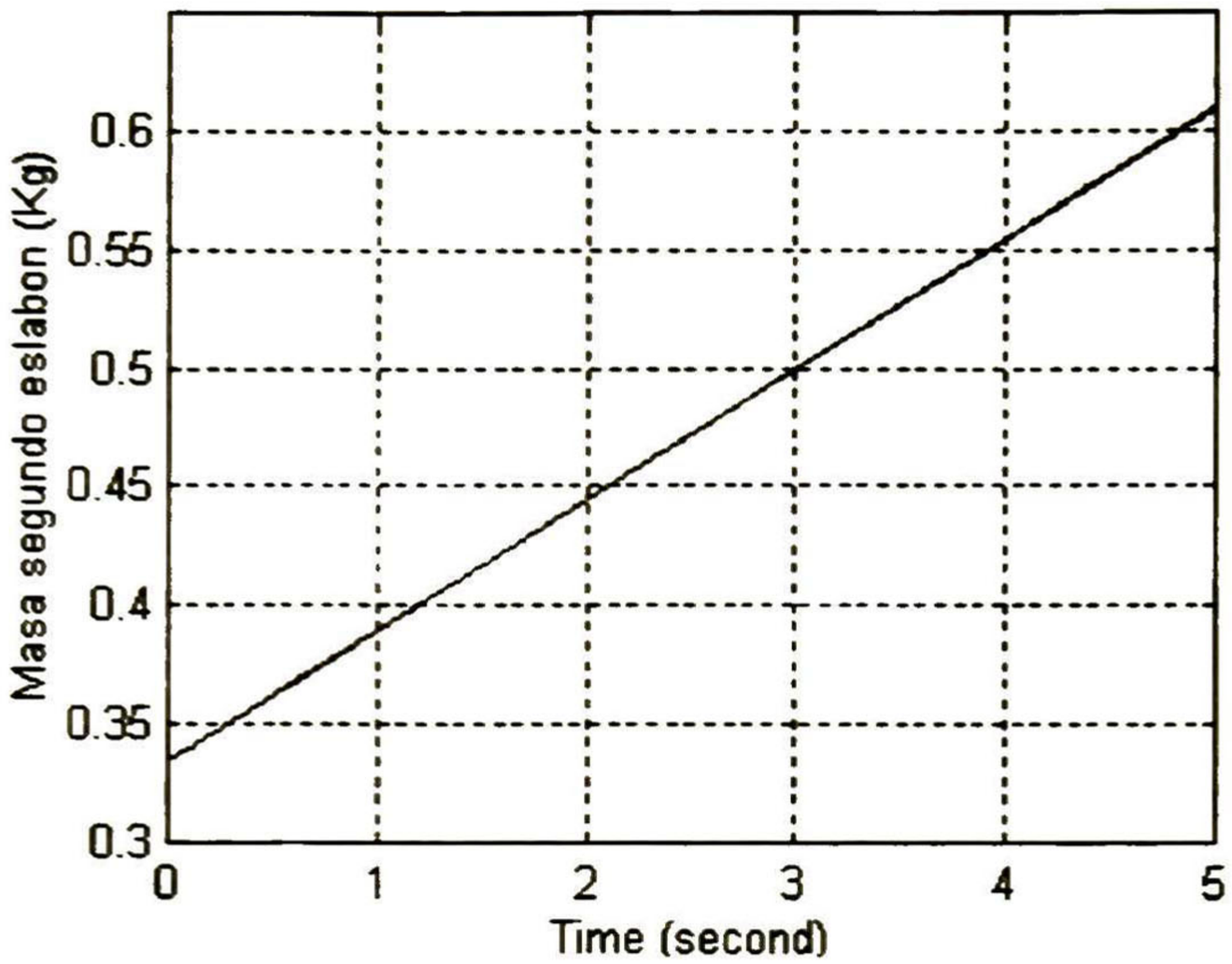


Figura 4.19 Cambio de masa en el segundo eslabón

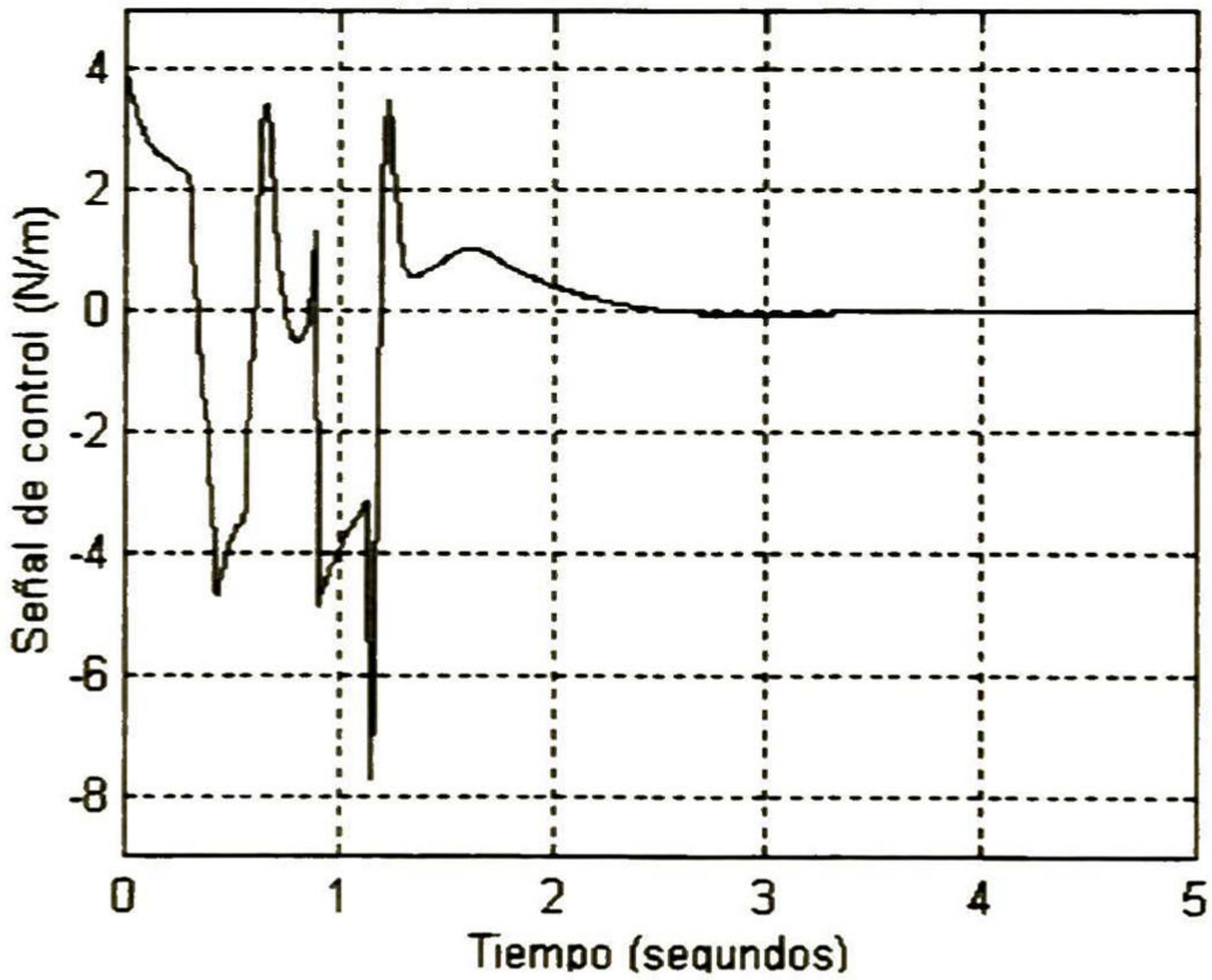


Figura 4.20 Señal de control (u_1)

La gran ventaja, de utilizar éste controlador es que no es necesario determinar el modelo matemático del pendubot para sintetizar una ley de control. Sólo basta con sintonizar el controlador difuso correctamente; éste procedimiento es empírico,

aunque existen técnicas para realizarlo [3]. Otra de las ventajas es que pueden cambiarse algunos parámetros del sistema y el controlador sigue funcionando adecuadamente. En nuestro caso se hizo un cambio de masa y el pendubot llego a la posición deseada. Esto muestra la robustez del algoritmo, lo que algunos de los controladores no poseen puesto que dependen directamente de los parámetros del sistema.

4.3 Implementación en tiempo real

En esta sección se presenta la programación en tiempo real[16] del controlador PD difuso discutido en la sección 2.3. El controlador PD difuso para la estabilización se encontrará activo cuando la distancia explicada anteriormente sea menor a 25 cm. La evolución de los ángulos q_1 y q_2 con respecto al tiempo son mostrados en la Figura 4.21. Se presenta una pequeña oscilación $\pm$ 3° para ambos eslabones como se muestra en la Figura 4.22 y 4.23. La señal de control aplicada al actuador del pendubot se presenta en la Figura 4.24, en la cúal se puede verificar que el torque el motor es menor a $\pm$ 10 N/m. En la Tabla C.3 (Apéndice C) se muestran los valores de las ganancias para éste controlador.

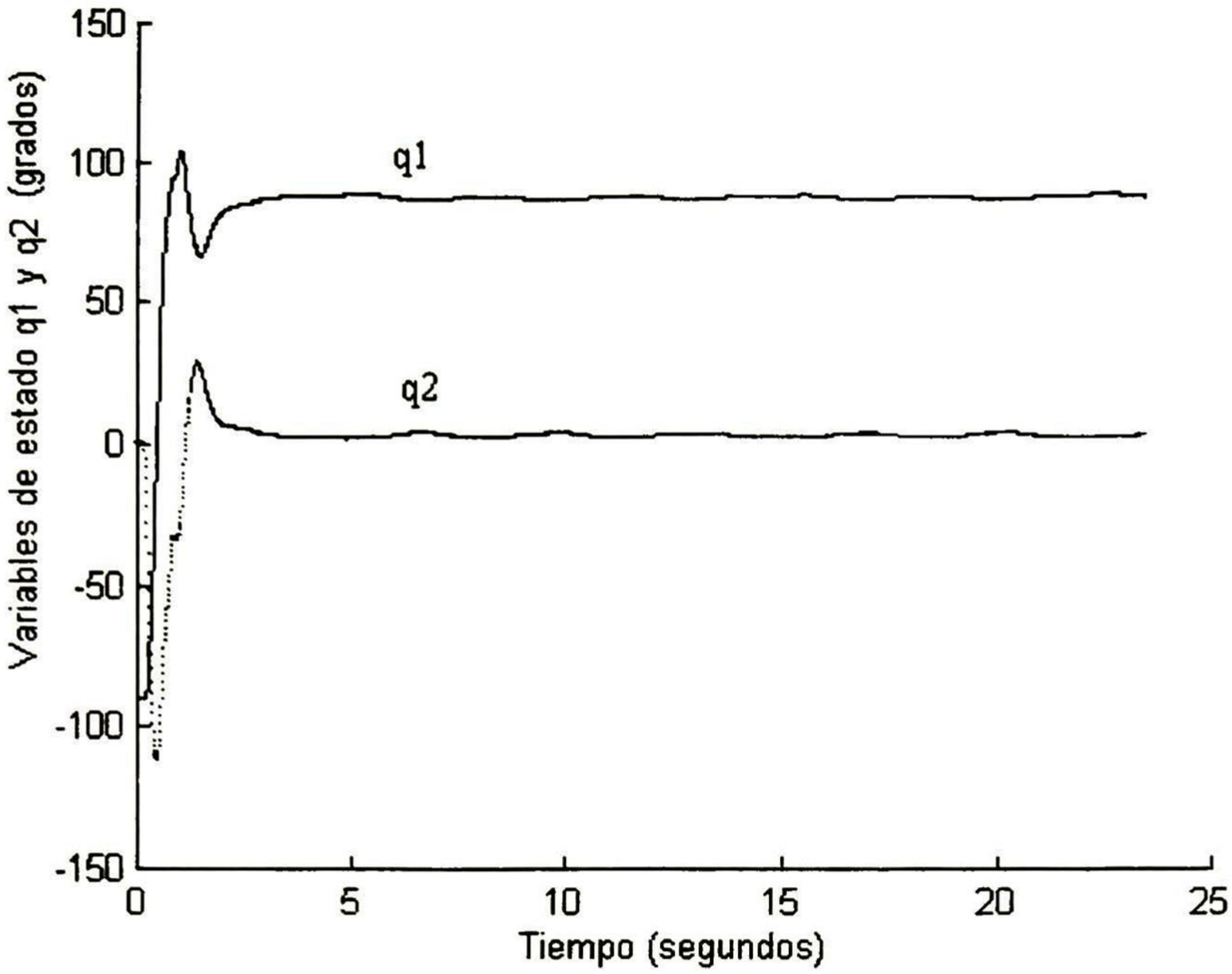


Figura 4.21 Posicionamiento de los eslabones con el PD difuso

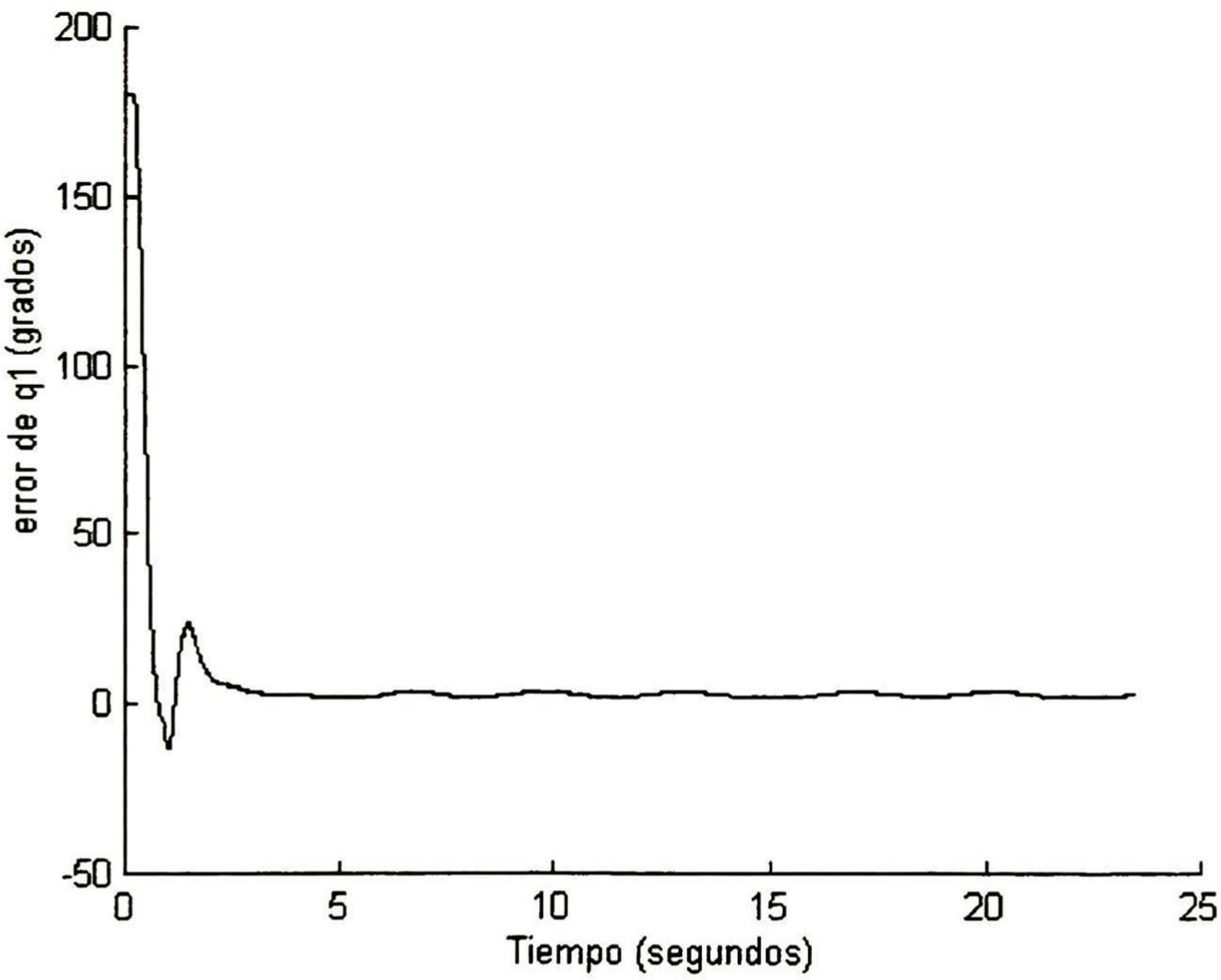


Figura 4.22 Error de posición en el primer eslabón

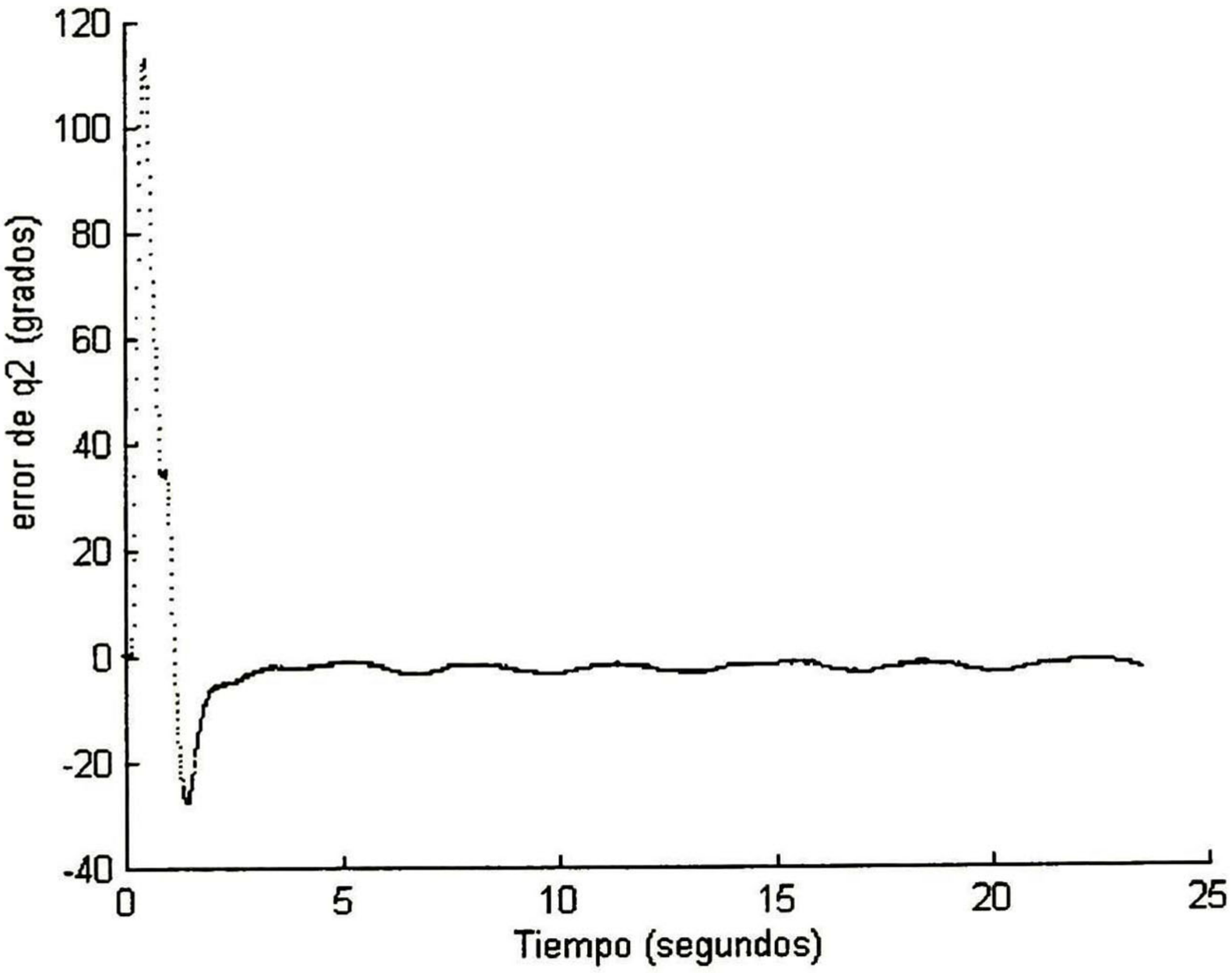


Figura 4.23 Error de posición en el segundo eslabón

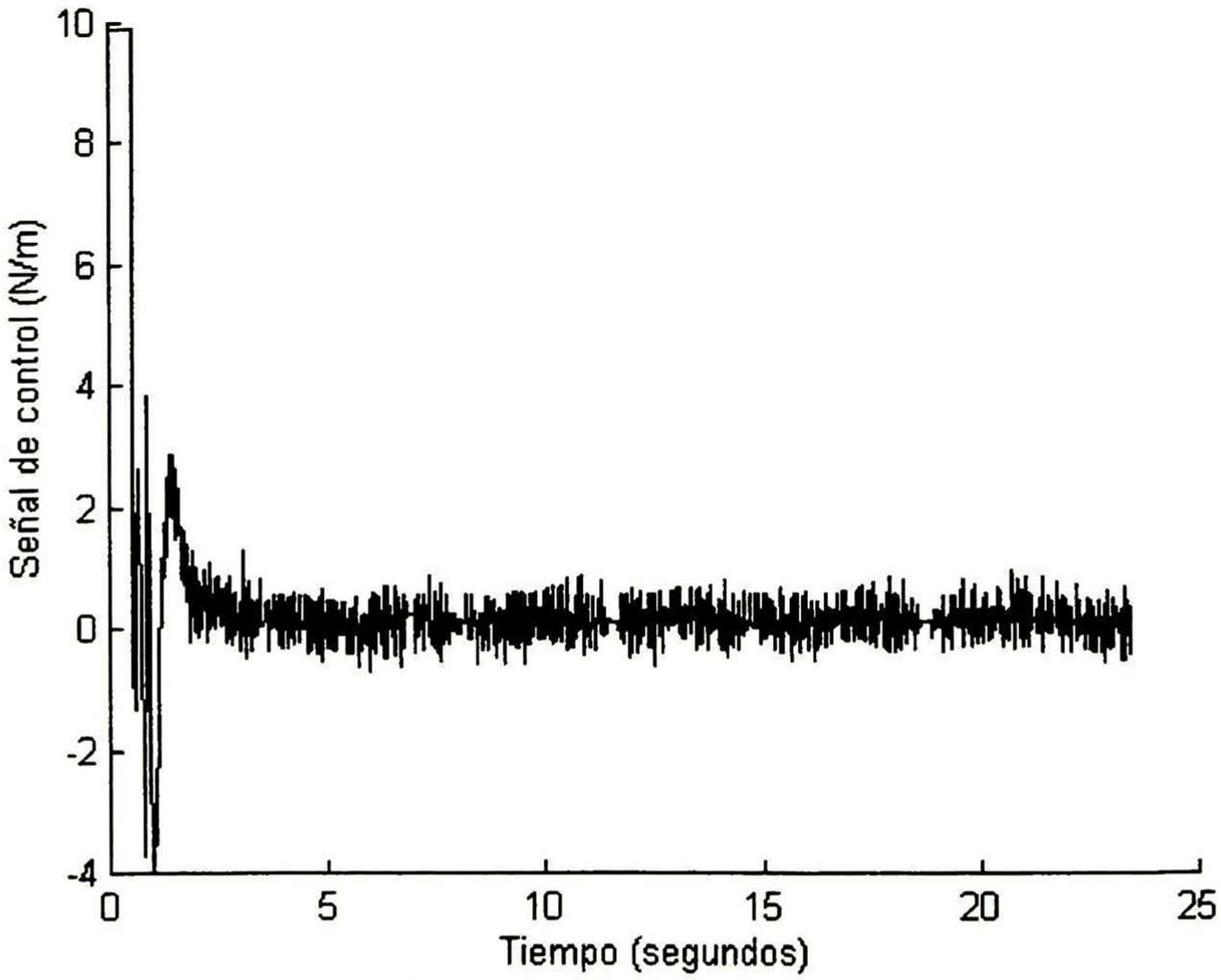


Fig. 4.24 Señal de control

Si se lleva al pendubot al punto deseado y se introduce una perturbación (golpeando el eslabón exterior) para alejarlo del punto de equilibro, éste regresa a su posición de equilibrio. La evolución de los ángulos con respecto al tiempo para esta prueba se muestran en la Figura 4.25. El error del primer y segundo eslabón se presentan en las Figuras 4.26 y 4.27 respectivamente; en la Figura 4.28, se despliega la señal de control.

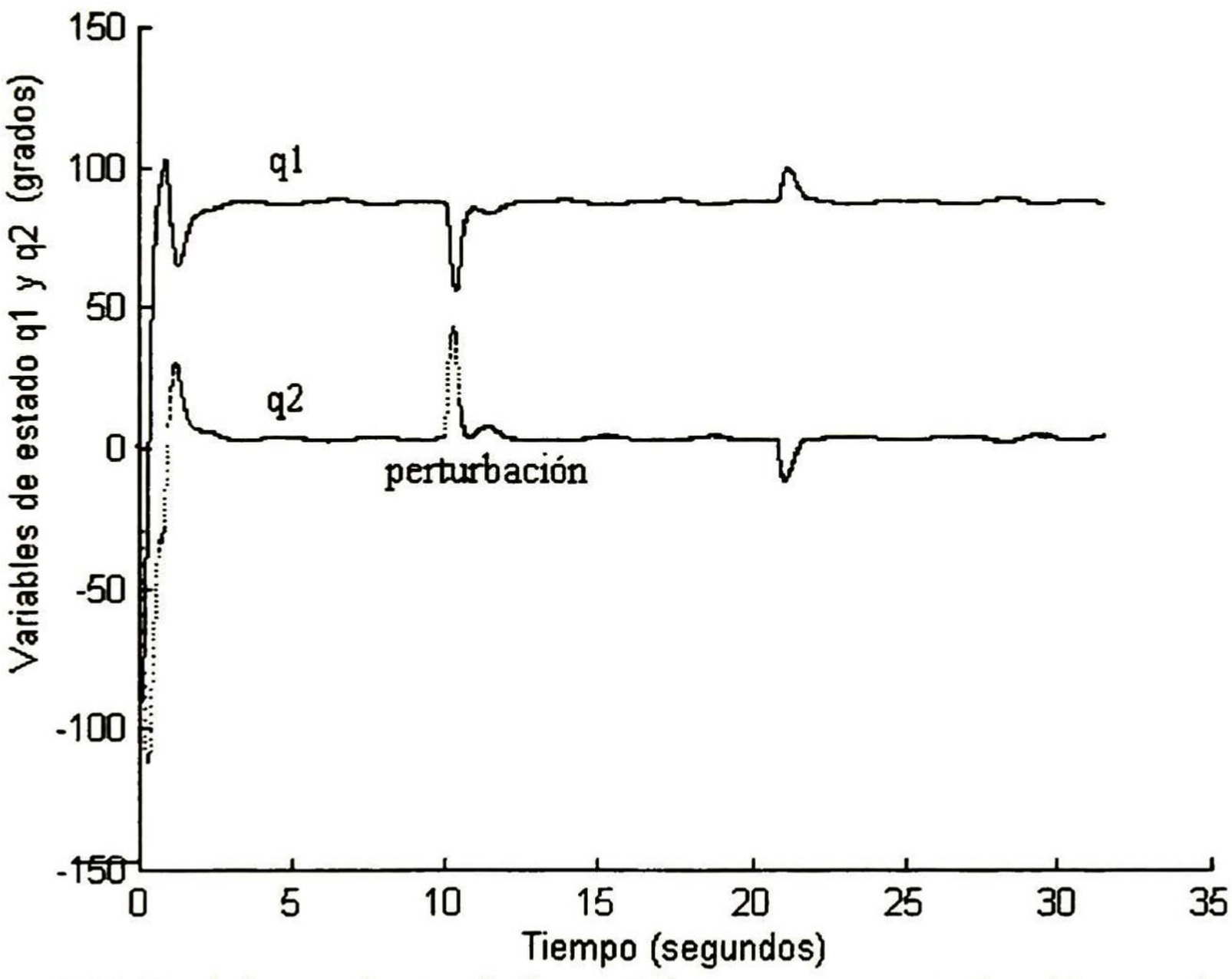


Figura 4.25 Posicionamiento de los eslabones con perturbación en el segundo

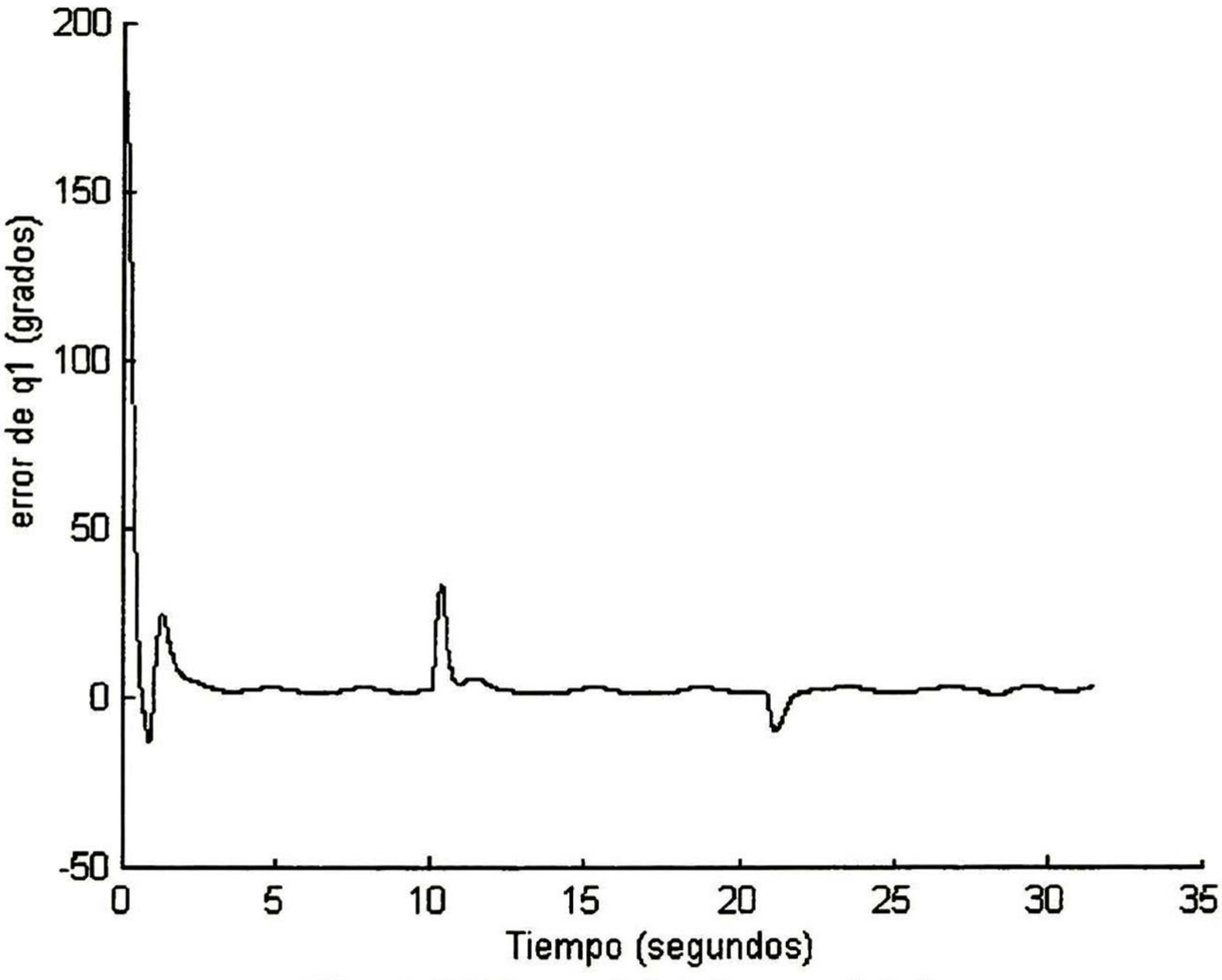


Fig. 4.26 Error del primer eslabón

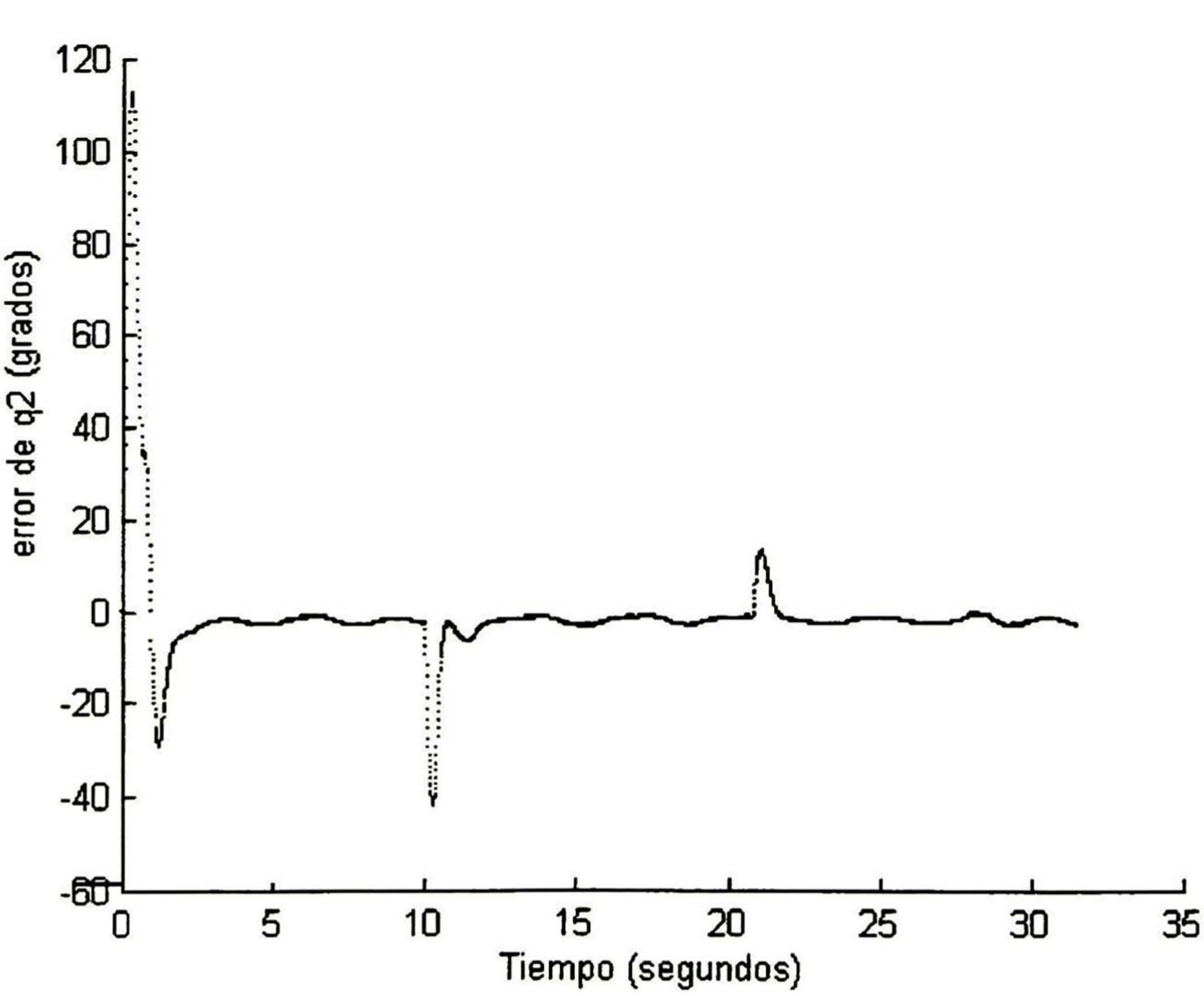


Fig.4.27 Error del segundo eslabón

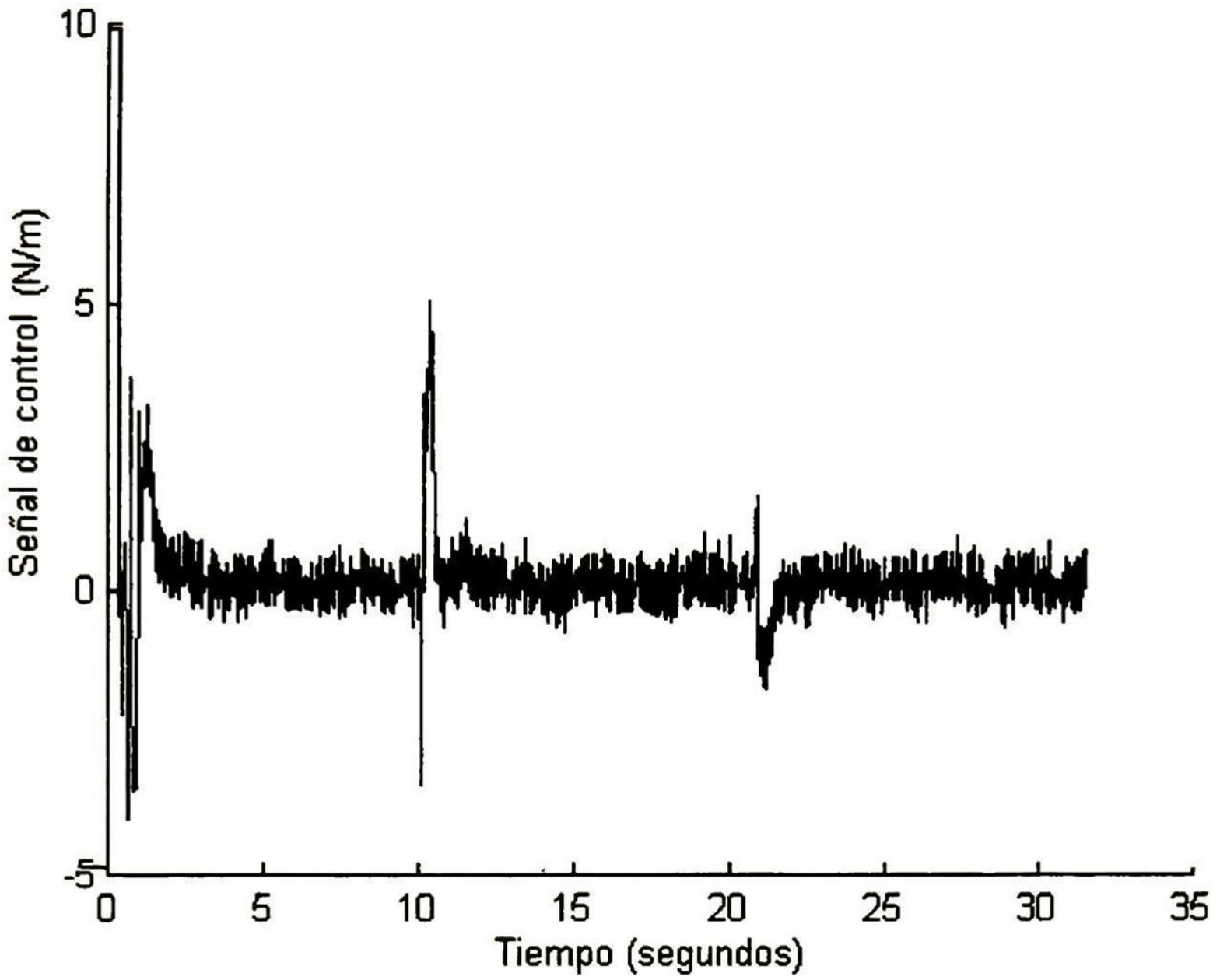


Figura 4.28 Señal de control

Un experimento que se hizo con el pendubot fue variar su centro de masa, colocando un peso extra en el segundo eslabón, éste peso extra variaba su centro de masa dependiendo del tiempo. Las Figuras 4.29 y 4.30 presentan la evolución de los ángulos q_1 y q_2 respectivamente. La figura 4.31 muestra la señal de control.

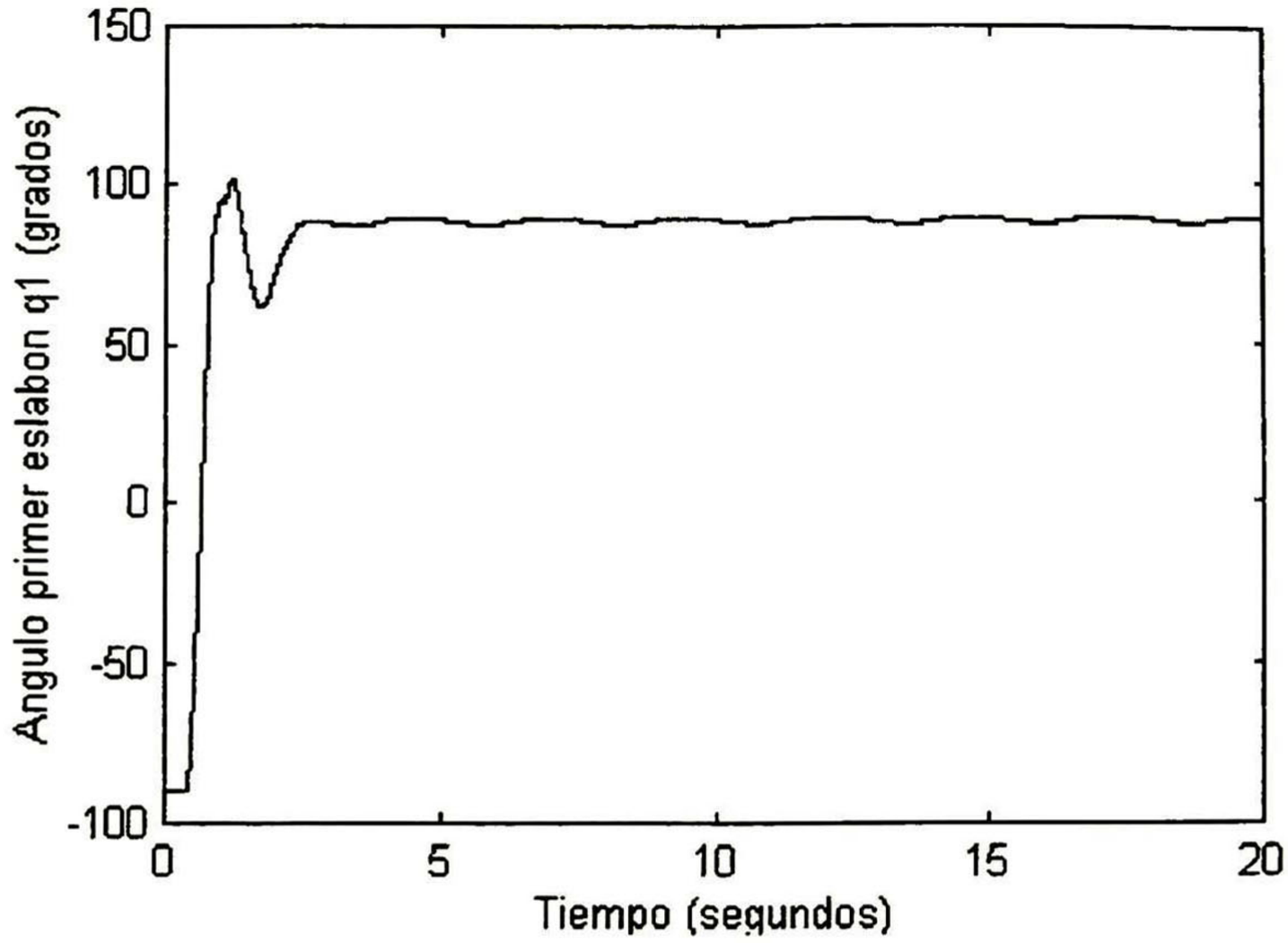


Figura 4.29 Posición angular del primer eslabón q_1

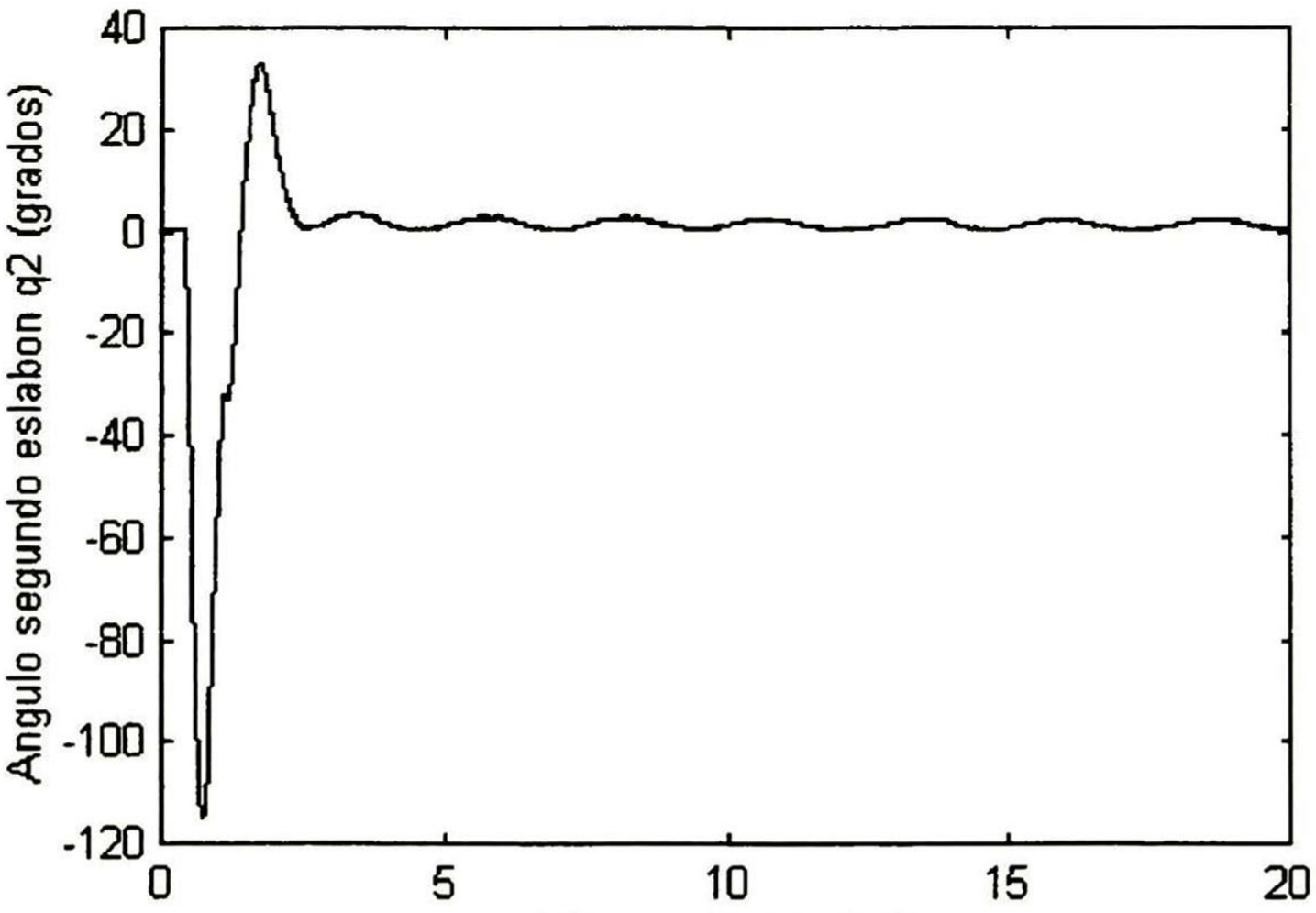


Figura 4.30 Posición angular del segundo eslabón q_2

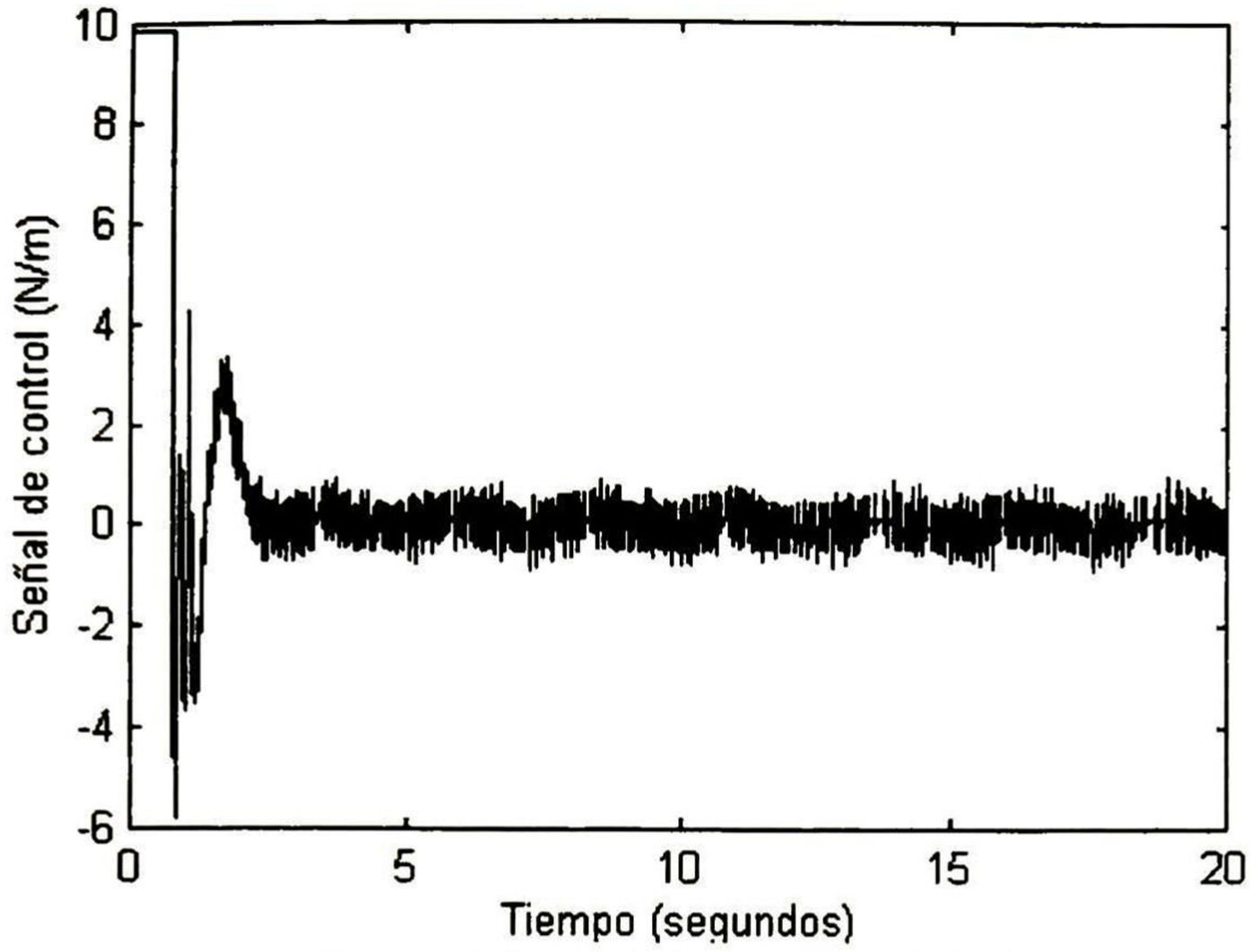


Figura 4.31 Señal de control (u_1)

4.4 Simulación de seguimiento de trayectoria con PD difuso

En las secciones anteriores de éste capítulo se ha discutido cómo el controlador PD difuso es capaz de llevar al pendubot a un punto de equilibrio deseado, que cumpla con la condición[1] de que $q_1 + q_2 = 90°$. Tomando esto en cuenta, ahora se desea que el pendubot siga una referencia senoidal, pasando a través de diversos puntos de equilibrio. En éste caso la señal de error para el primer eslabón viene dada por:

$$error_1 = \frac{\pi}{2} + Asin(\omega t) - \theta_1$$

Respetando la condición para que sea un punto de equilibrio, la señal de error del segundo eslabón es:

$$error_2 = -Asin(\omega t) - \theta_2$$

Por facilidad se toma $\omega = 1$ rad/seg; esto significa que la frecuencia de oscilación será $1/2\pi$Hz con el período de la oscilación de 2π seg. El experimento consta de llevar pendubot a la posición vertical y entonces realizar el seguimiento de trayectoria.

Para hacer la implementacion en Matlab[15] se supone que el pendubot ya se encuentra en la posición de equilibrio (q_1= 90° y q_2=0°) y a la señal de referencia se le adicionan las señales oscilatorias para hacer el seguimiento de trayectoria, el esquema de simulación se presenta en la Figura 4.32

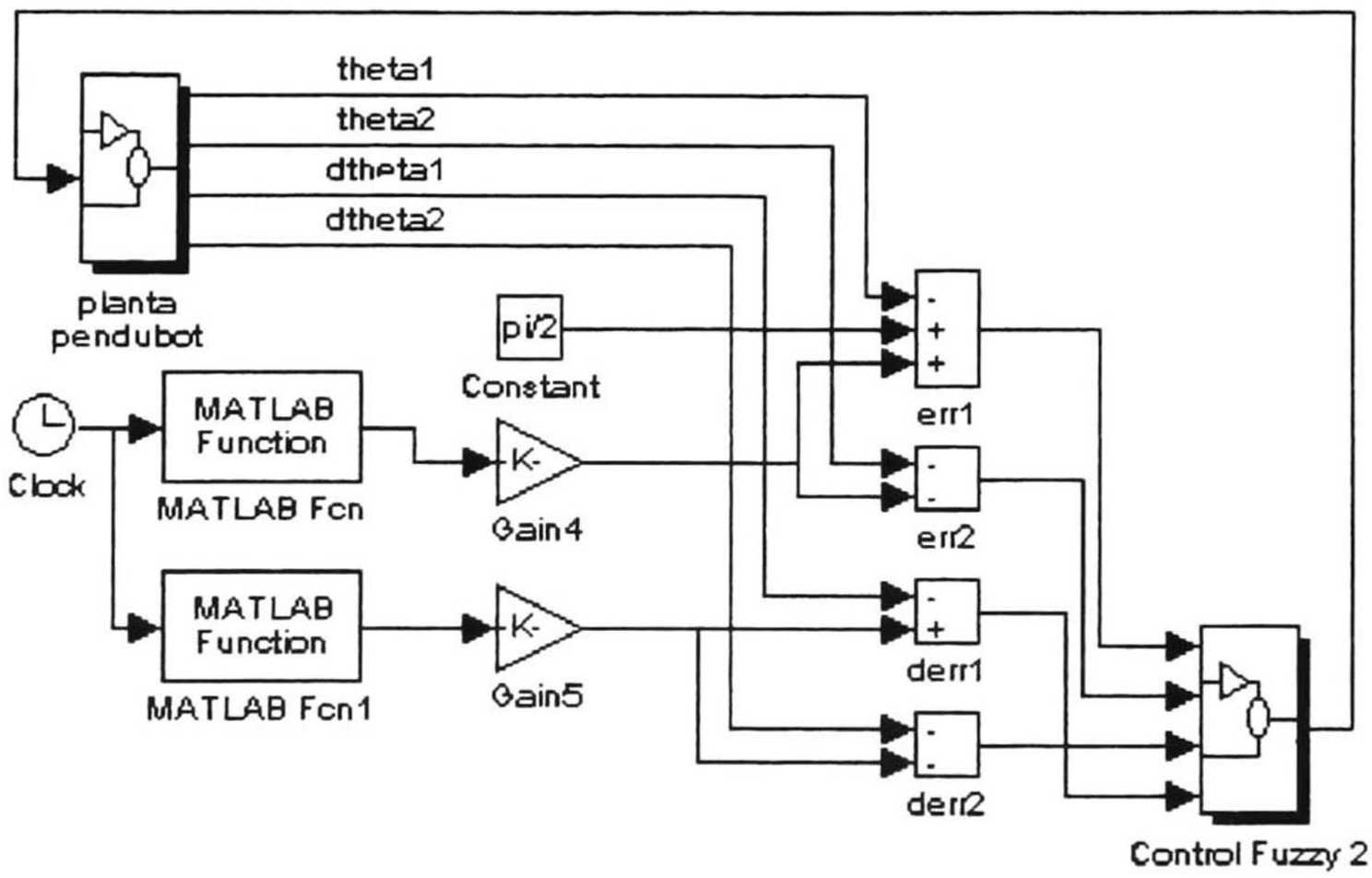


Figura 4.32 Esquema de simulación para seguimiento de trayectoria

Dentro de la simulación se genera la derivada del error utilizando los estados y la derivada de la señal. Las ganancias de éste controlador se muestran en la Tabla C.4 (Apéndice C). La señal de referencia y la respectiva respuesta para el ángulo q_1 se presenta en la Figura 4.33 y los correspondientes al ángulo q_2 en la Figura 4.34.

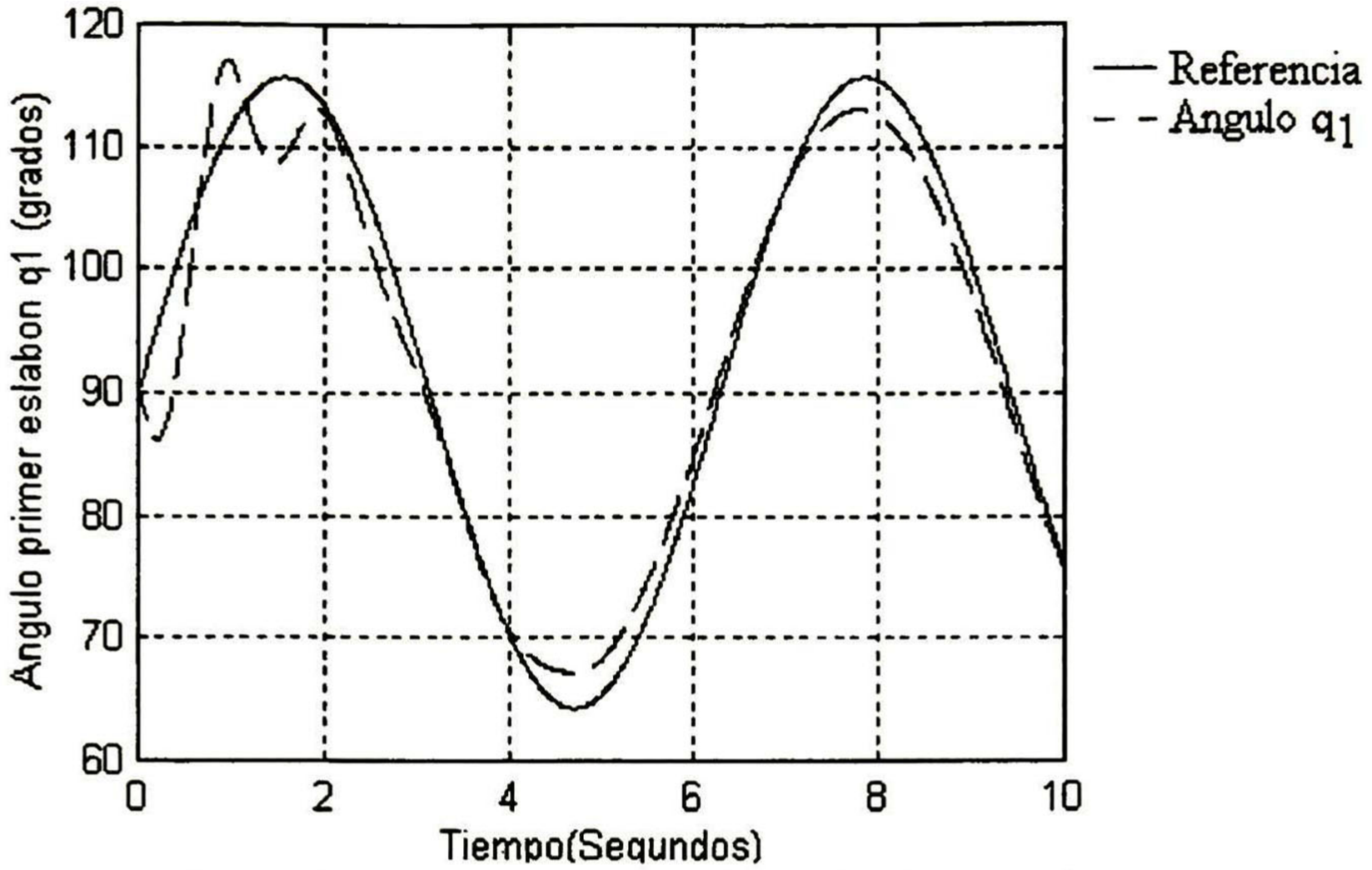


Figura 4.33 Evolución y referencia del ángulo q_1

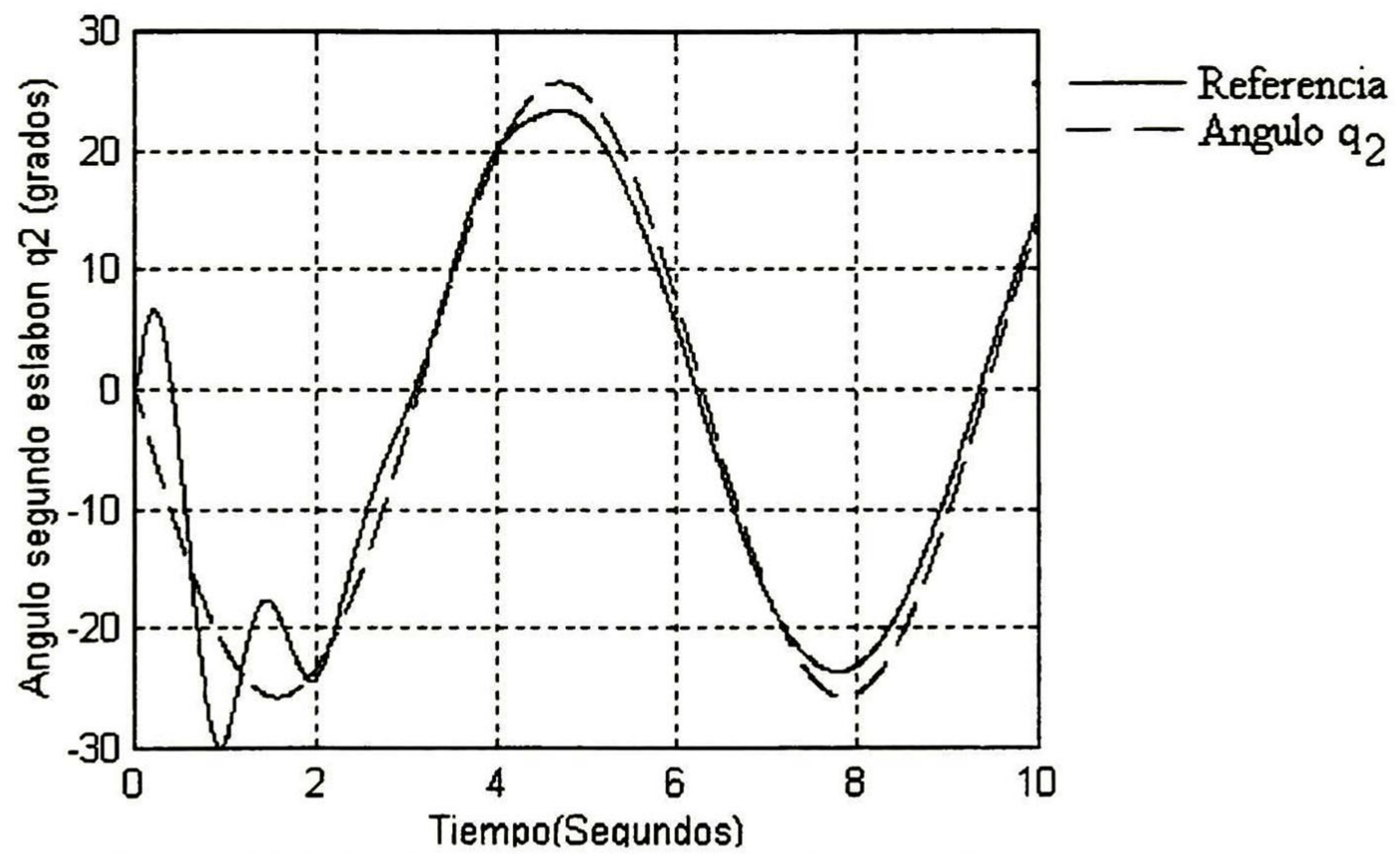


Figura 4.34 Evolución y referencia del ángulo q_2

La señal de error del primer y segundo eslabón se presentan en las Figuras 4.35 y 4.36 respectivamente. Así mismo la señal de control se muestra en la Figura 4.37.

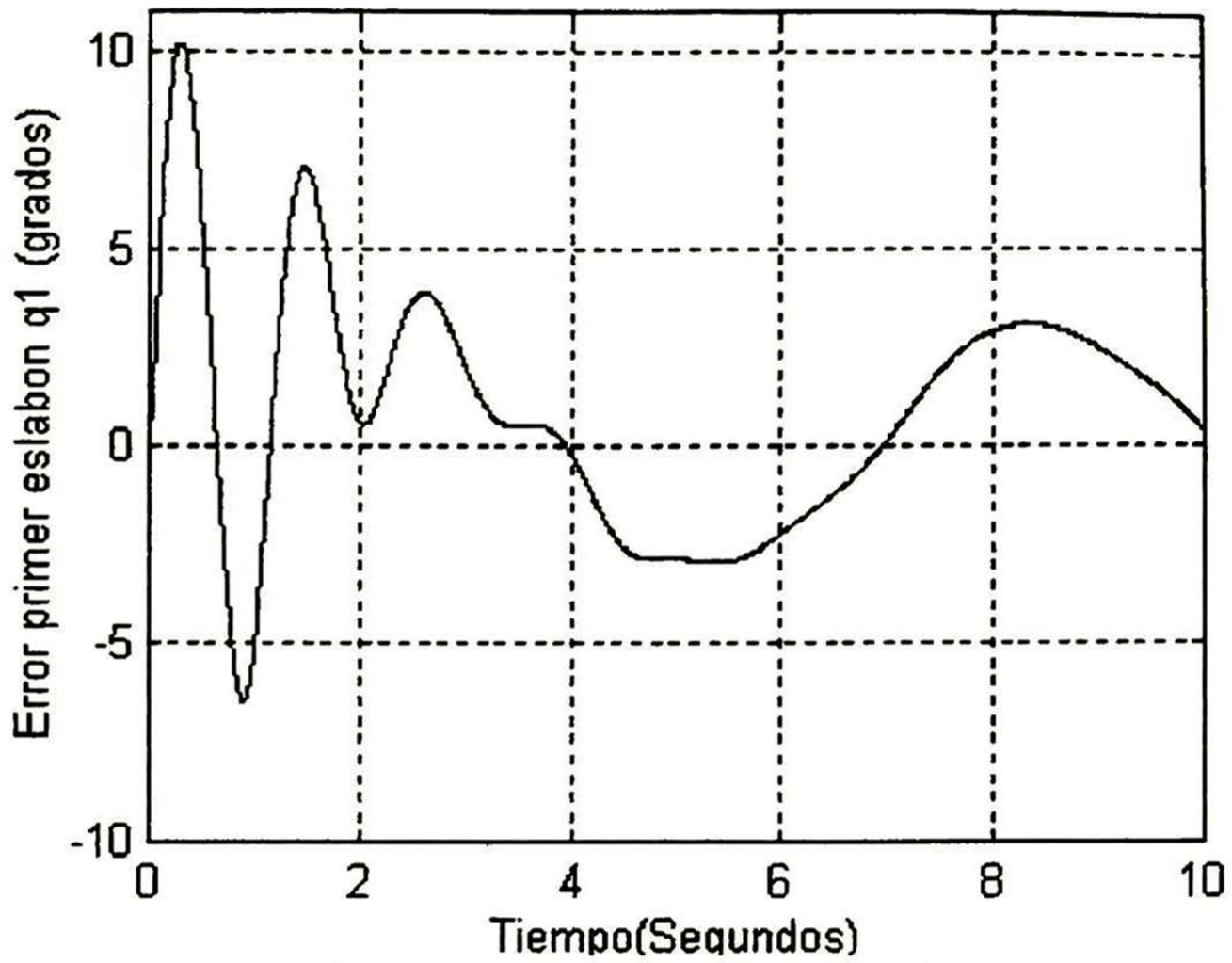


Figura 4.35 Error del ángulo q_1

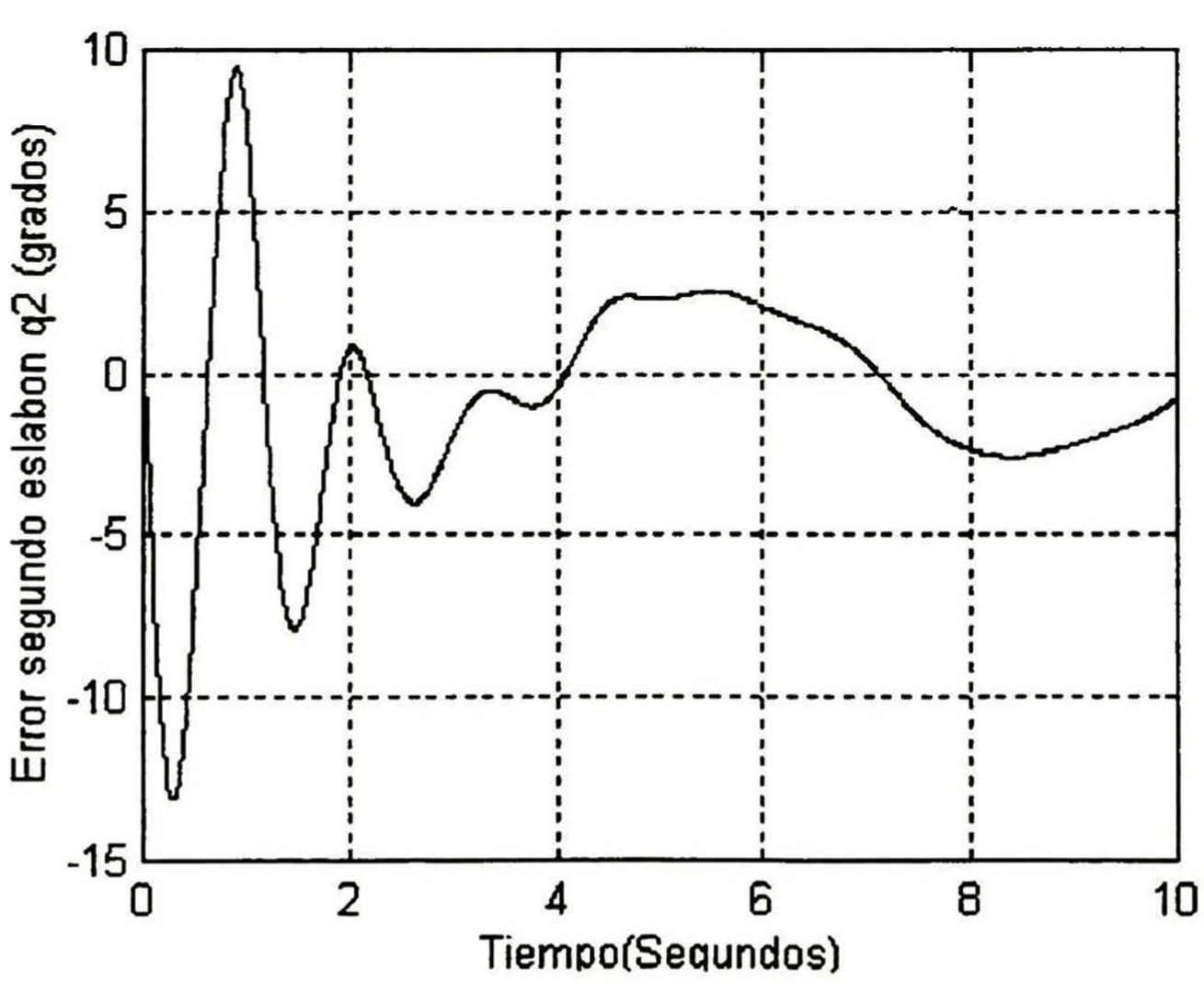


Figura 4.36 Error del ángulo q_2

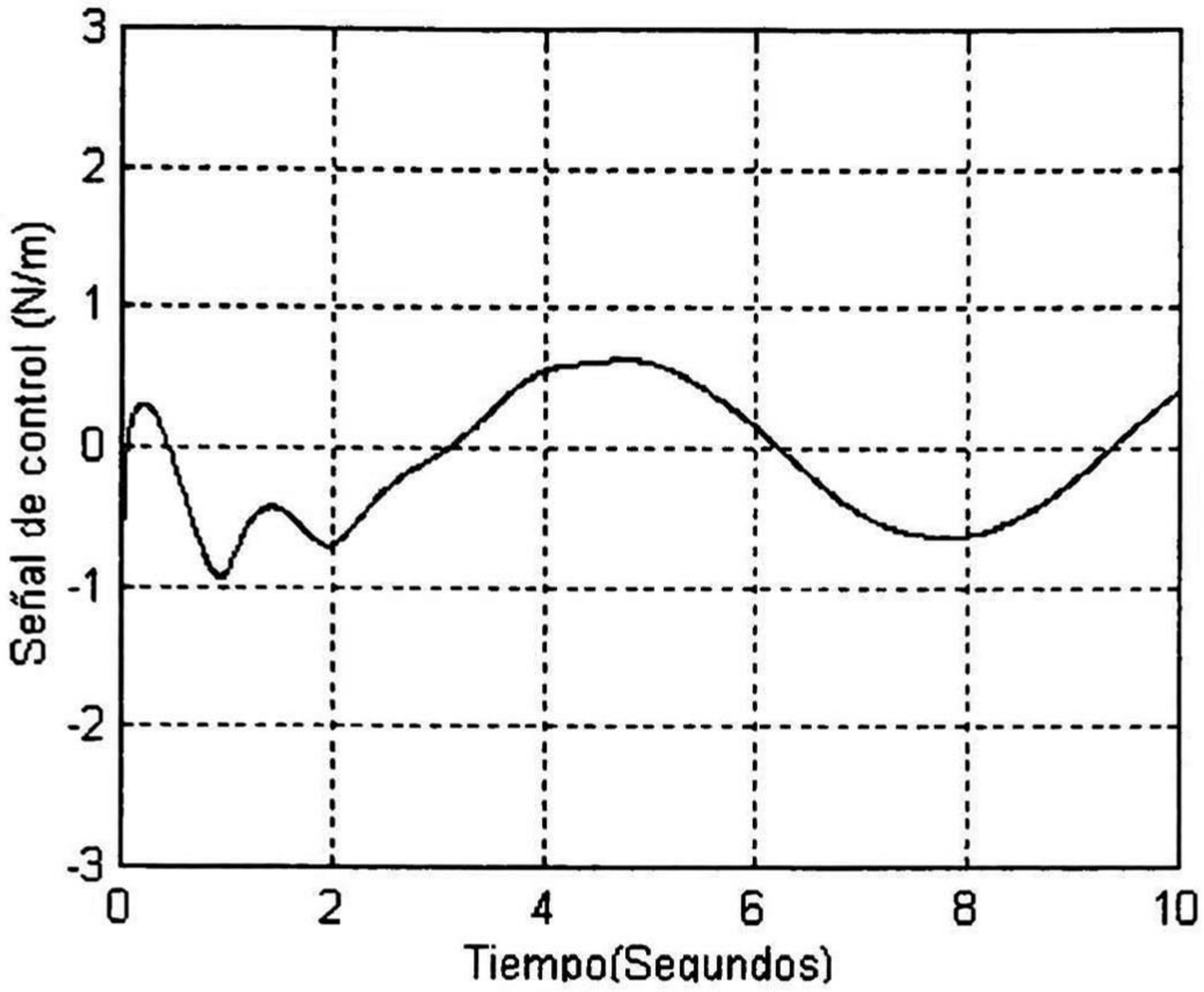


Figura 4.37 Señal de control

Como se observa, no se logra un seguimiento adecuado para la trayectoria.

4.5 Resultados en tiempo real de seguimiento de trayectoria

El experimento consistió en llevar el pendubot al punto de equilibrio, retenerlo ahí durante 4 seg y después hacerle seguir la referencia - *Asin*(*t*) para el segundo eslabón. La evolución y señal de referencia de los ángulos q_1 y q_2 se muestran en la Figura 4.38 y Figura 4.39 respectivamente; la Figura 4.40 muestra la señal de control que cumple la restricción de no exceder ±10 N/m. La Tabla C.5 (Apéndice C) muestra las ganancias de éste controlador. Por último el error del primer y segundo eslabón (externo e interno) se muestran en la Figura 4.41 y 4.42 respectivamente.

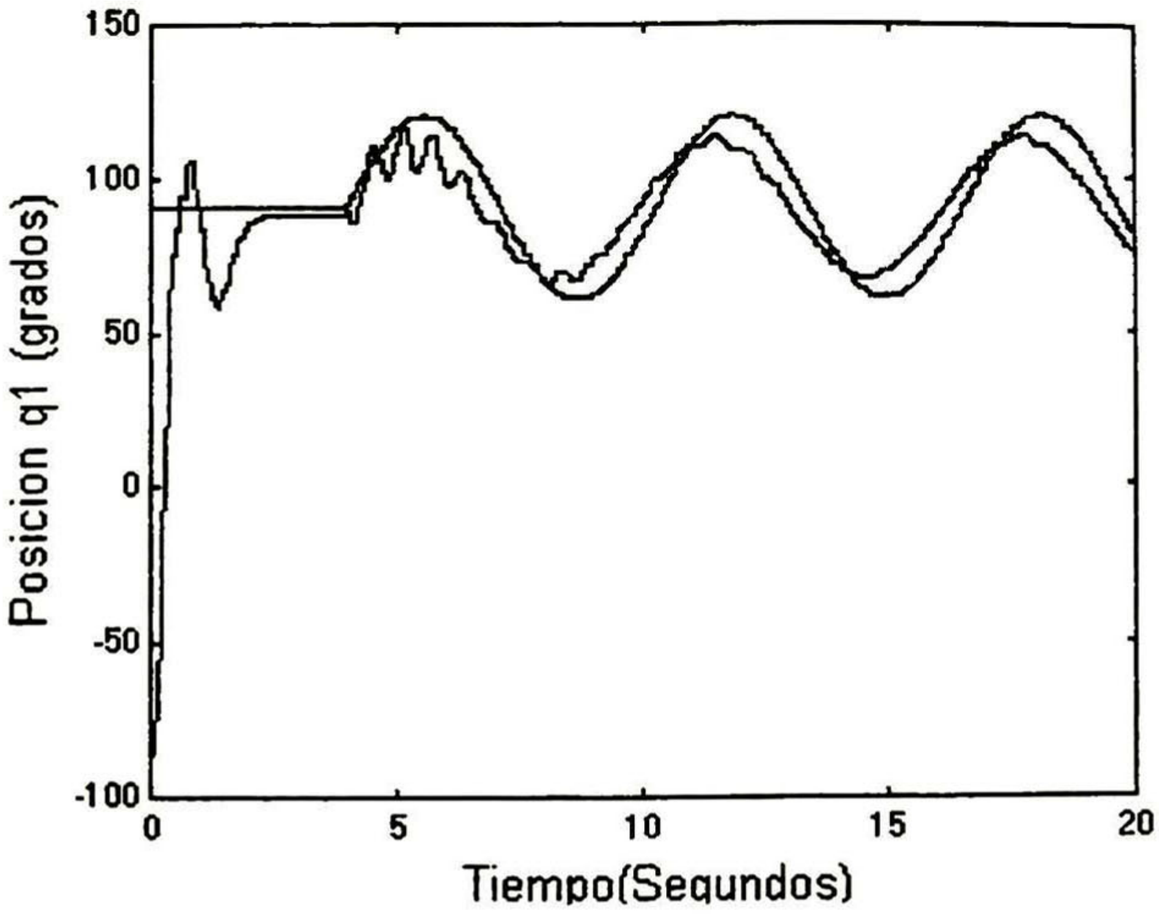


Fig. 4.38 Referencia y evolución de q_1

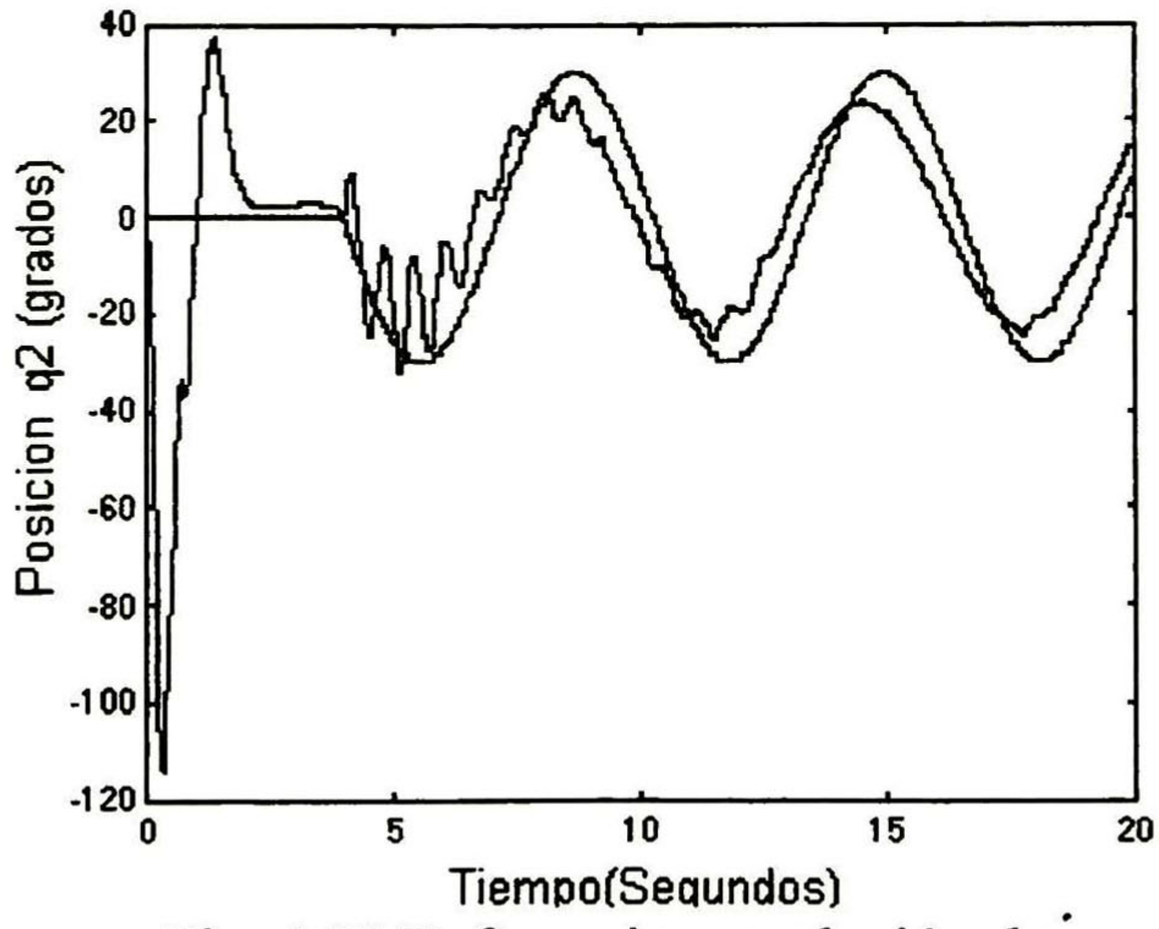


Fig. 4.39 Referencia y evolución de $\dot{q}_2$

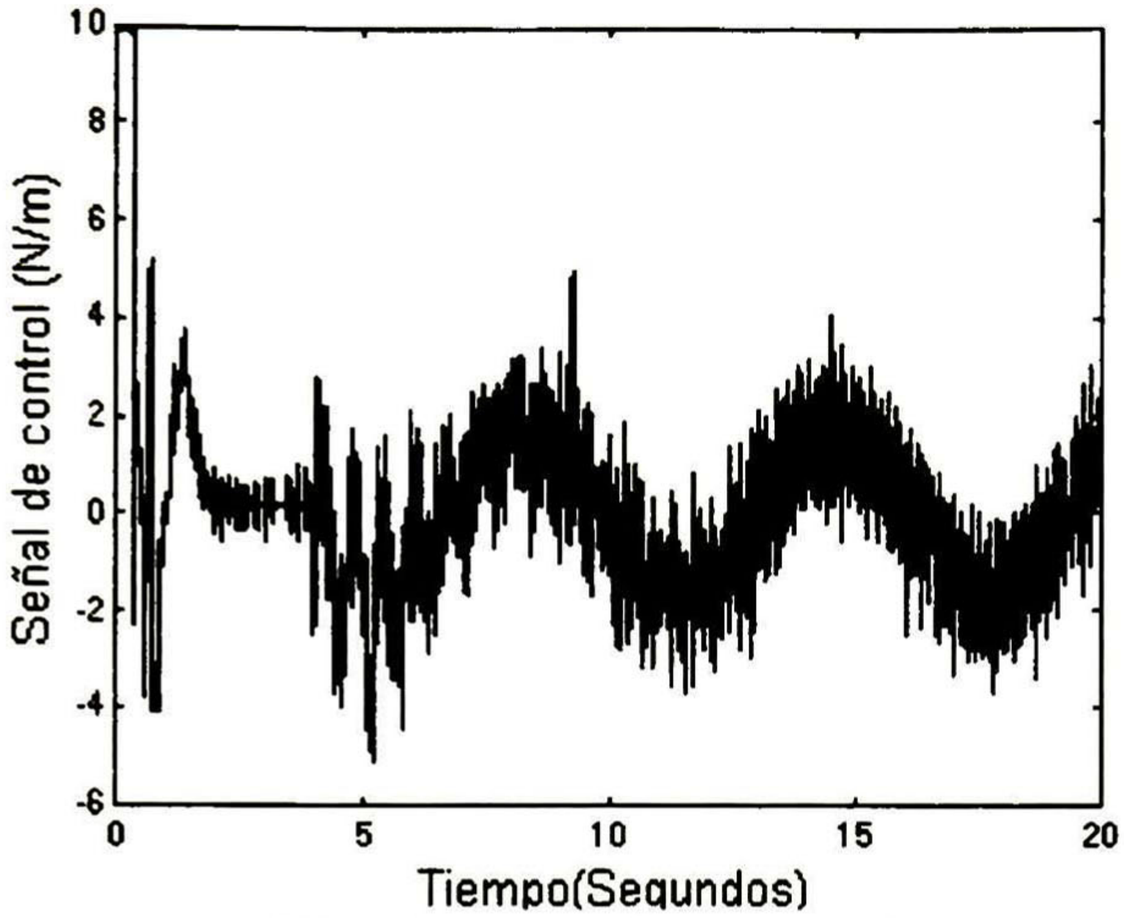


Fig. 4.40 Señal de control

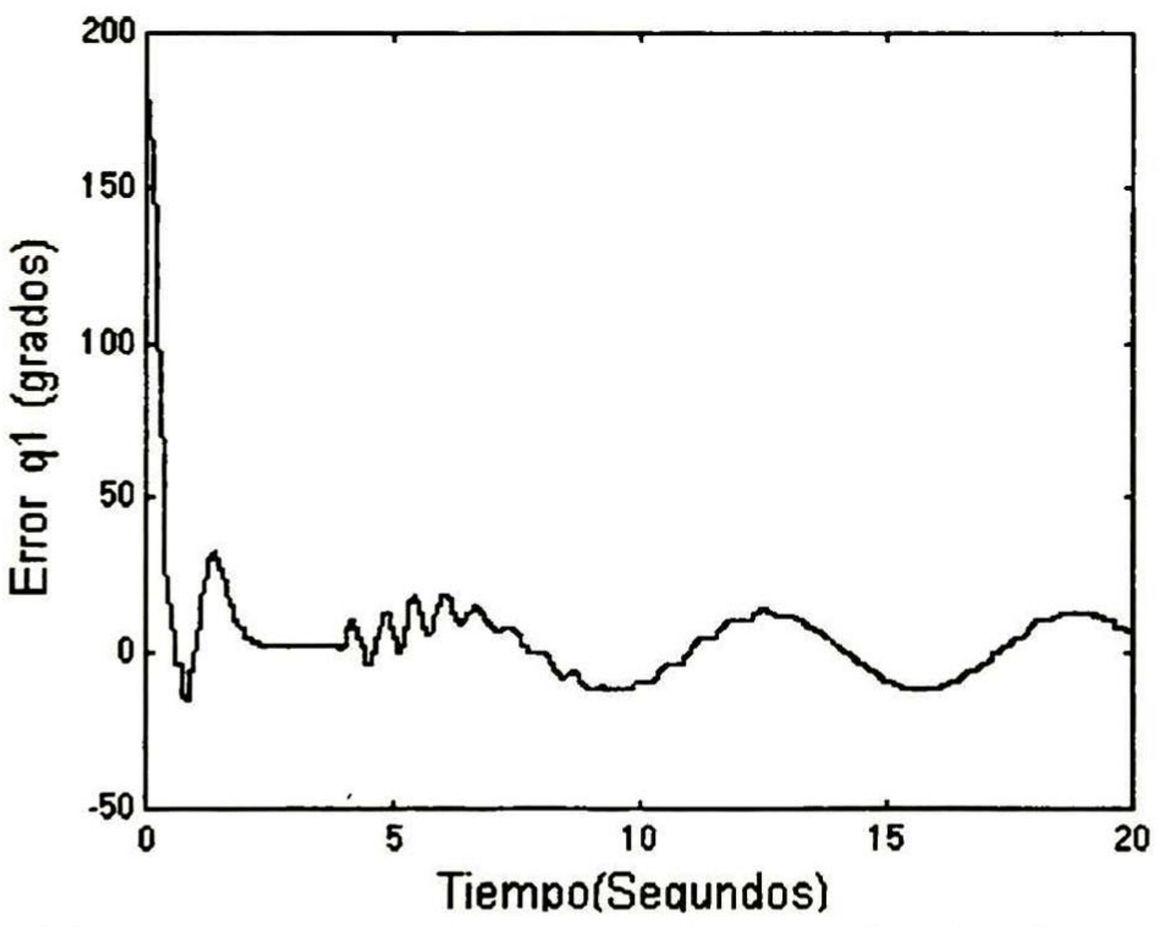


Fig. 4.41 Señal de error primer eslabón (interno)

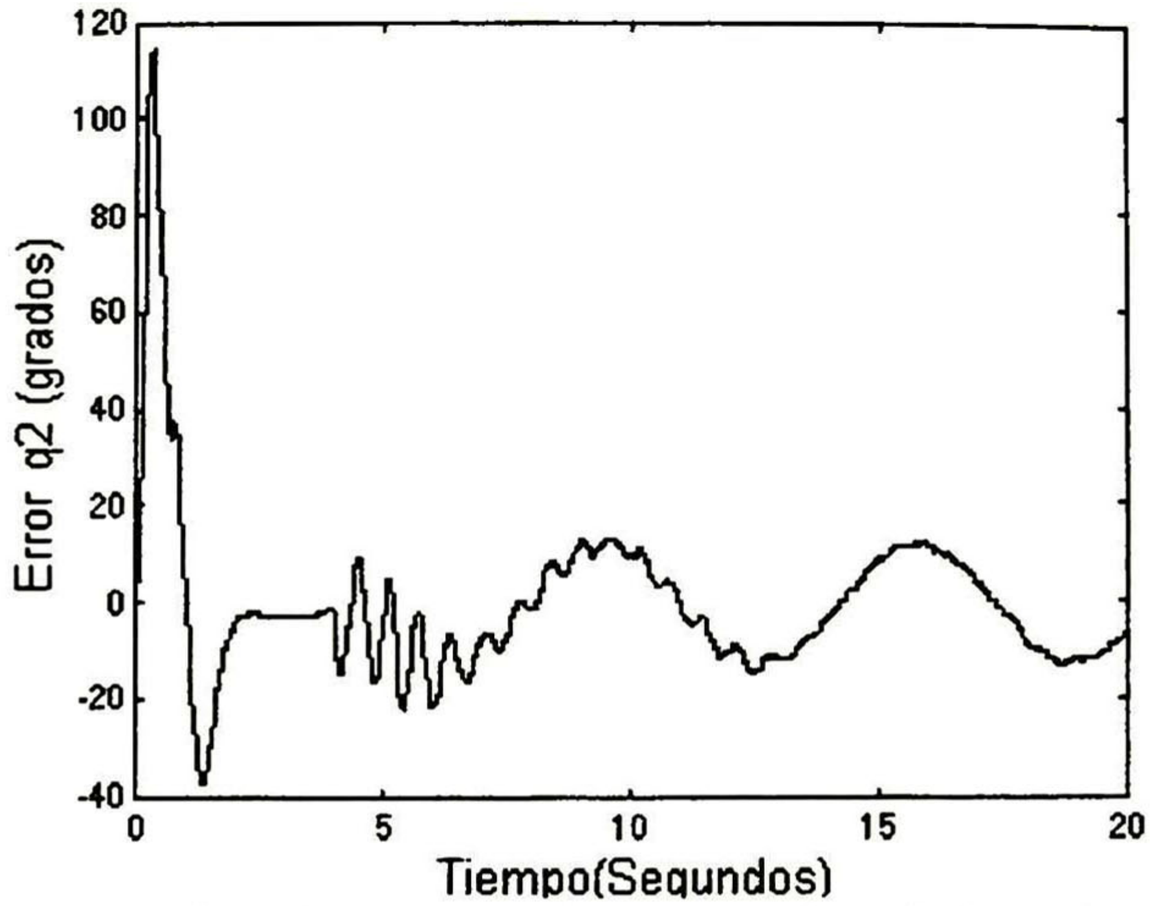


Fig. 4.42 Señal de error del segundo eslabón (externo)

Como era de esperarse, los resultados no son adecuados observándose un desfasamiento con respecto a la trayectoria que se quiere seguir.

4.6 Seguimiento de trayectoria con PD difuso y regulador

Como se observa en la sección anterior, los resultados obtenidos no son satisfactorios; por lo que se vió la necesidad de aplicar el principio de regulador lineal para mejorar el seguimiento de trayectoria. La primera suposición que se hace es que el modelo del pendubot es lineal:

$$\begin{aligned} \dot{x} &= Ax + bu \\ y &= cx \end{aligned} \tag{4.13}$$

y que $\{A,b,c\}$ es controlable. El modelo del pendubot es un sistema no-lineal, pero si se linealiza alrededor del punto de equilibrio x^*= [90° 0 0 0]; utilizando la aproximación en series de Taylor se tiene:

$$A = \frac{\partial f(x)}{\partial x}\Big|_{x=x^*} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 32.2499 & -10.0172 & 0 & 0 \\ -29.8052 & 37.6597 & 0 & 0 \end{bmatrix}$$

$$B = g(x^*) = \begin{bmatrix} 0 \\ 0 \\ 26.6507 \\ -42.5387 \end{bmatrix} \tag{4.14}$$

En éste caso $h(x)=x_1+x_2$ que son las variables que se quieren controlar por lo que:

$$C = \frac{\partial h(x)}{\partial x}\Big|_{x=x^*} = \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} \tag{4.15}$$

A partir de las matrices A,B en (4.14) se tiene que la matriz de controlabilidad está dada por:

$$C = \begin{bmatrix} B, & AB, & A^2B & A^3B \end{bmatrix} = \begin{bmatrix} 0 & 26.65 & 0 & 1285 \\ 0 & -42.5387 & 0 & -2396 \\ 26.65 & 0 & 1285 & 0 \\ -42.5387 & 0 & -2396 & 0 \end{bmatrix} \tag{4.16}$$

la cuál tiene rango pleno, por lo que la aproximación lineal es controlable.

Para éste sistema linealizado, se propone una señal de control del tipo:

$$u = Ke + v \tag{4.17}$$

donde la retroalimentación Ke está dada por el controlador PD difuso, asumiendo que el sistema es estable con esta retroalimentación.

Como se sabe, el vector de error es igual a:

$$\begin{aligned} e &= y_d - x \\ x &= y_d - e \end{aligned} \tag{4.18}$$

donde y_d es el vector de salidas deseadas y x es el vector de estados del sistema.

Sustituyendo en la ecuación (4.17) y (4.18) en (4.13) para dejar al sistema en función del error se tiene:

$$\begin{aligned} \dot{y}_d - \dot{e} &= A(y_d - e) + B(Ke + v) \\ \dot{e} &= y_d - Ay_d + Ae - BKe - Bv \\ e &= (A - BK)e - Bv + \dot{y}_d - Ay_d \\ y &= C(y_d - e) \\ y &= Cy_d - Ce \end{aligned} \tag{4.19}$$

En éste caso en particular se va a seguir una referencia senoidal, por lo que los vectores y_d y $\dot{y}_d$ son:

$$y_d = \begin{bmatrix} Asin(t) \\ -Asin(t) \\ A\cos(t) \\ -A\cos(t) \end{bmatrix} \qquad \dot{y}_d = \begin{bmatrix} A\cos(t) \\ -A\cos(t) \\ -A\,\text{sen}(t) \\ A\,\text{sen}(t) \end{bmatrix} \tag{4.20}$$

de aquí se ve que $Cy_d=0$

En el problema del regulador, la meta es llevar los estados del error a cero y entonces encontrar la entrada externa v para que siga la señal de referencia esto es:

$$y(t) = Ce(t) \xrightarrow[t\to\infty]{} y_d \tag{4.21}$$

donde y_d es el valor deseado de la salida.

Al llevar los estados a cero se tiene que:

$$\begin{aligned} \dot{e} = 0 &= (A-BK)e - Bv + \dot{y}_d - Ay_d \\ y_d &= Ce = -C(A-BK)^{-1}(Bv - \dot{y}_d - Ay_d) \end{aligned} \tag{4.22}$$

Para que siga cumpliendo la condición del punto de equilibrio para el sistema linealizado:

$$\begin{aligned} & x_1 + x_2 = 0 \\ & e_1 + Asin(t) + e_2 - Asin(t) = 0 \\ & e_1 + e_2 = 0 \end{aligned} \tag{4.23}$$

Entonces para determinar v se debe de cumplir que:

$$\begin{aligned} & y_d \xrightarrow[t\to\infty]{} 0 \\ & 0 = -C(A-BK)^{-1}(Bv - \dot{y}_d - Ay_d) \end{aligned} \tag{4.24}$$

Como se observa el termino $-C(A-BK)^{-1} \neq 0$ por lo que

$$Bv - \dot{y}_d - Ay_d = 0 \tag{4.25}$$

Sustituyendo la matriz A, los vectores B, y_d y $\dot{y}_d$ en la ecuación (4.25) se obtiene el valor de v, el cual es:

$$v = -1.6094568 Asin(t) \tag{4.26}$$

La simulación en Matlab se llevó a cabo partiendo de la condición inicial de q_1=90° y q_2=0°. Al bloque de simulación se le añade la señal externa *v* tal y cómo lo muestra la Figura 4.43

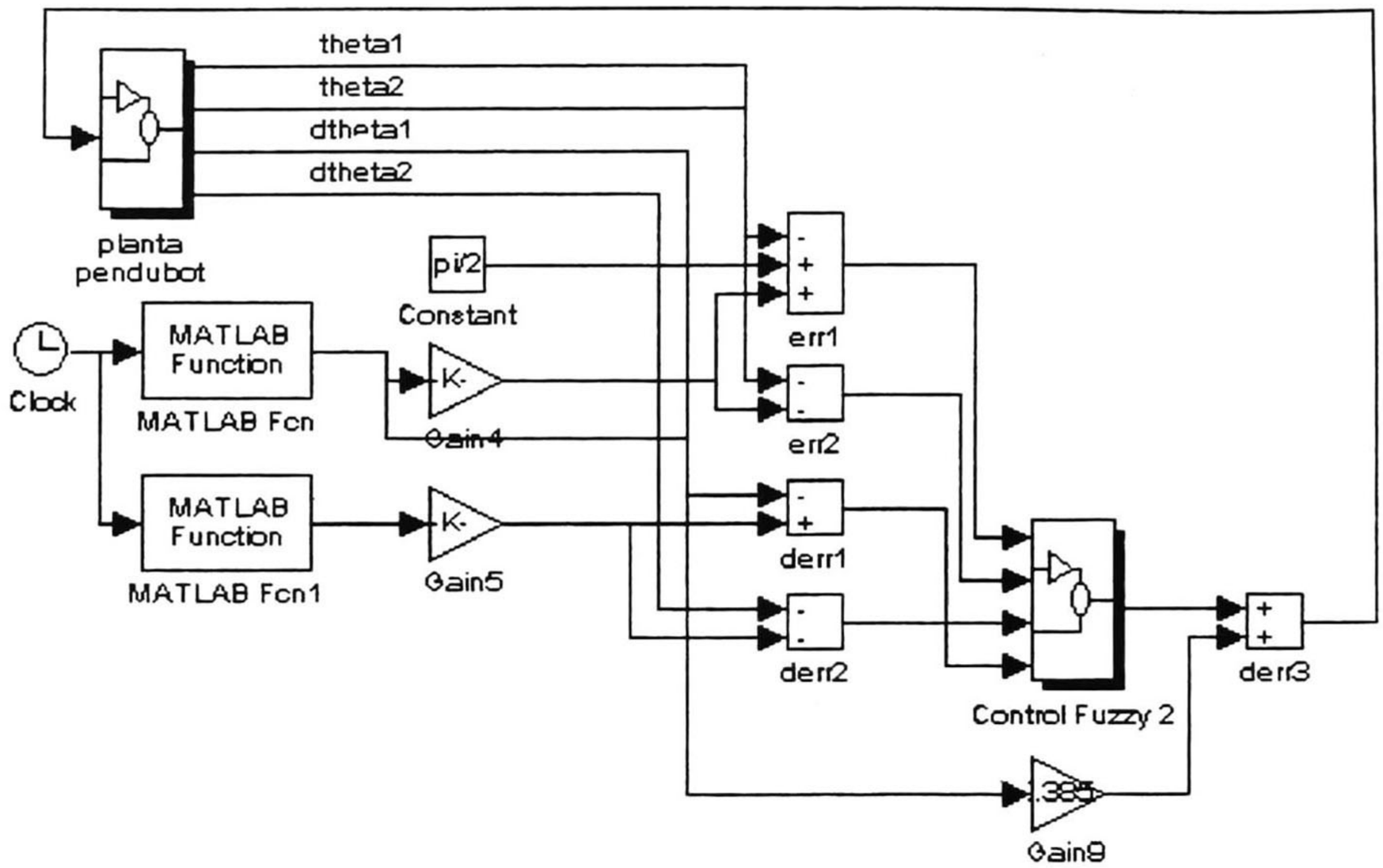


Figura 4.43 Esquema del controlador con regulador

Los resultados de la simulación son satisfactorios, las ganancias encontradas para éste controlador son mostradas en la Tabla C.6 (Apéndice C). La referencia y evolución del ángulo q_1 se muestra en la Figura 4.44. La referencia y evolución del ángulo q_2 se presentan en la Figura 4.45. Como puede observarse el error que se genera es mínimo y sigue bien la señal de referencia.

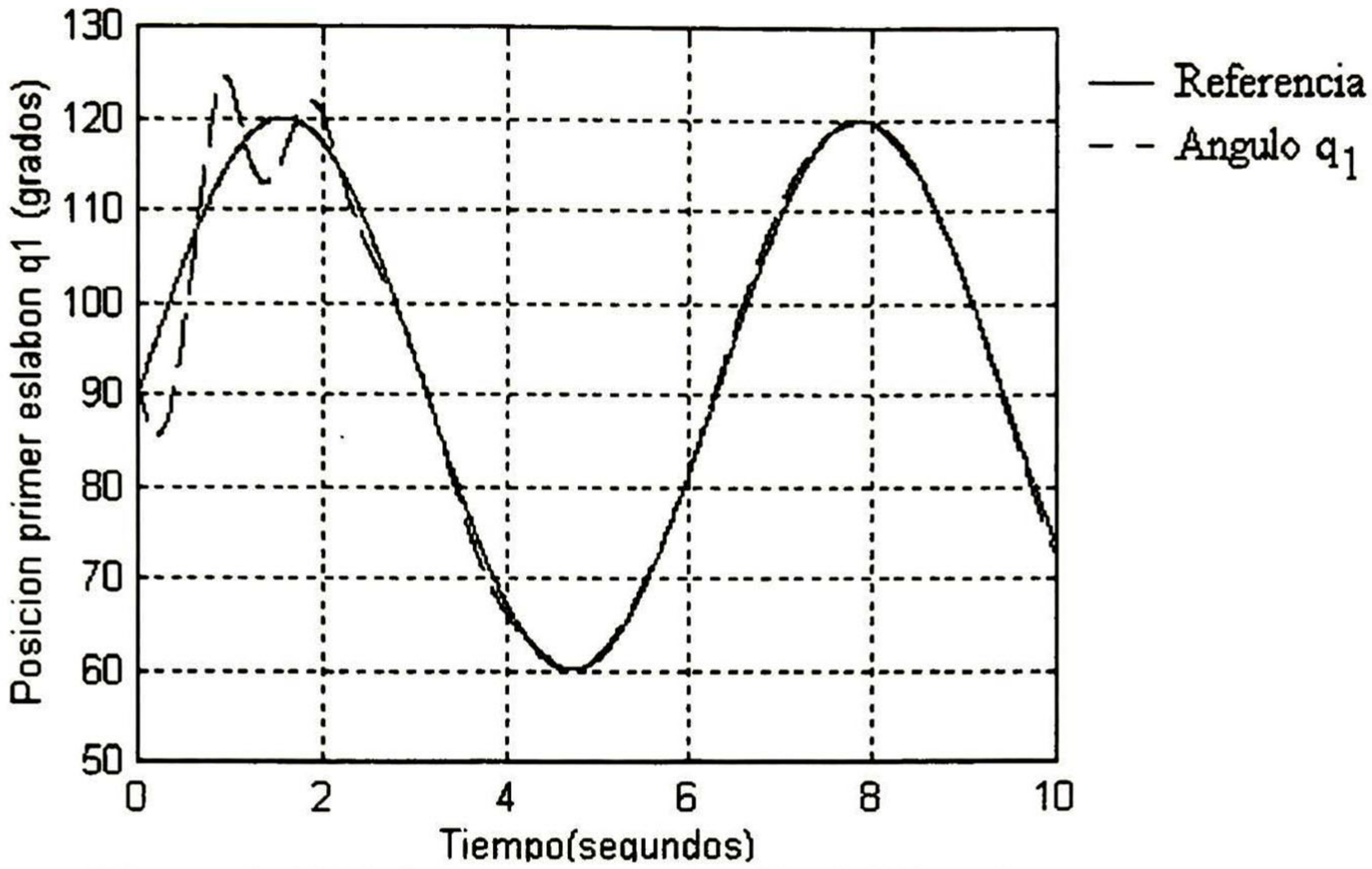


Figura 4.44 Referencia y evolución del ángulo q_1

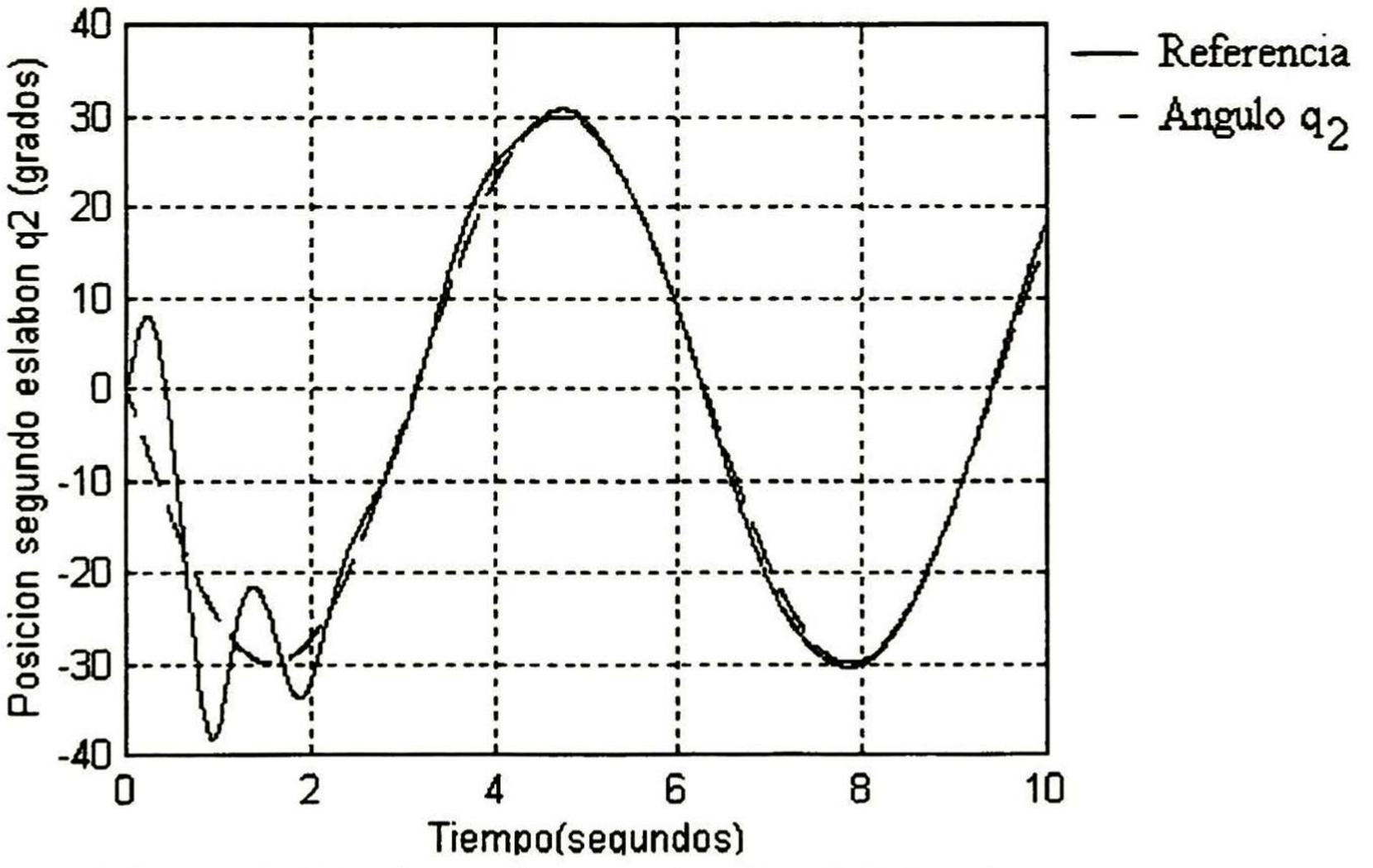


Figura 4.45 referencia y evolución del ángulo q_2

Los errores para los ángulos q_1 y q_2 son presentados en las Figuras 4.46 y 4.47; estos se encuentran dentro de +/- 3 grados aproximadamente. La señal de control se presenta en la Figura 4.48

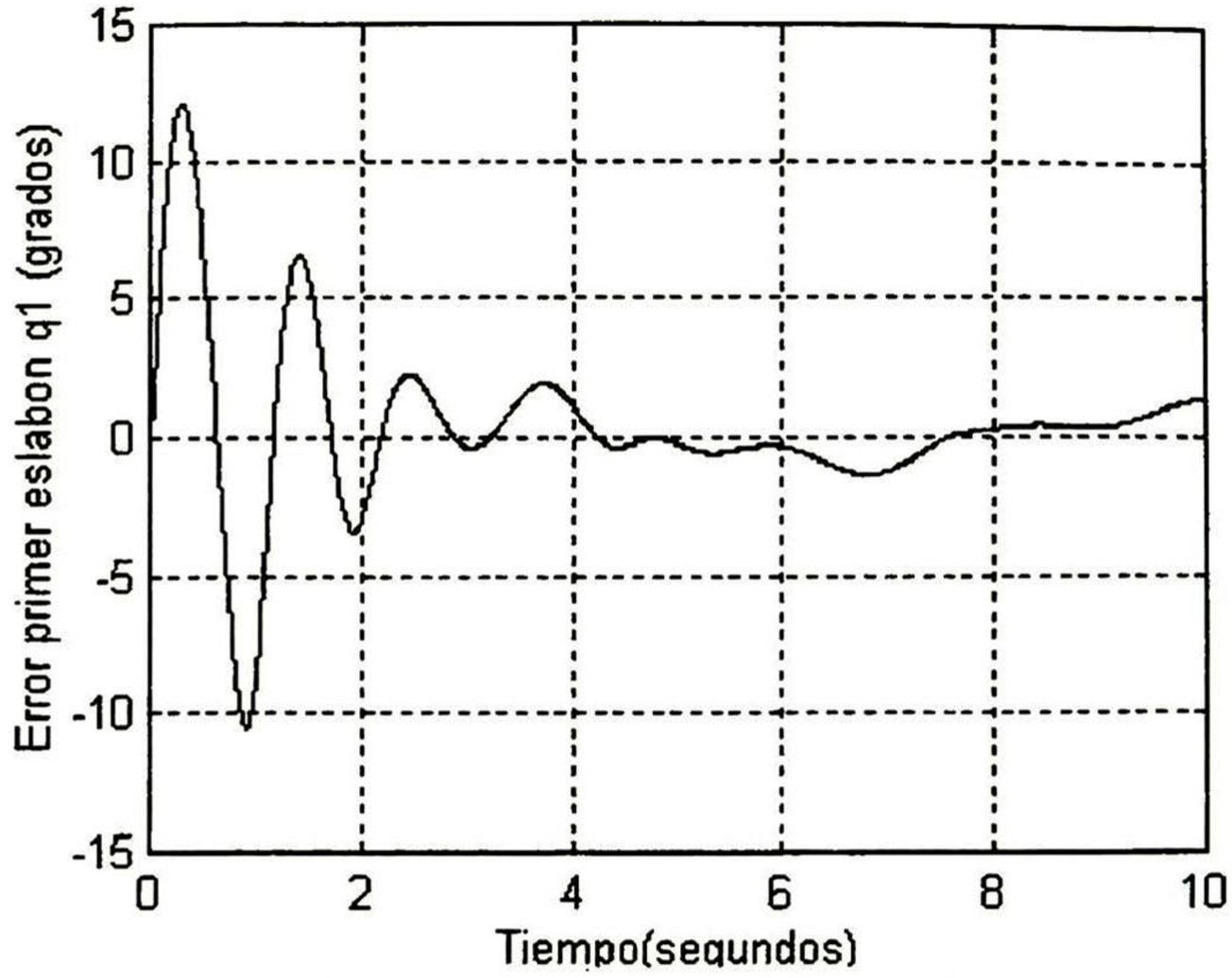


Figura 4.46 Error del ángulo q_1

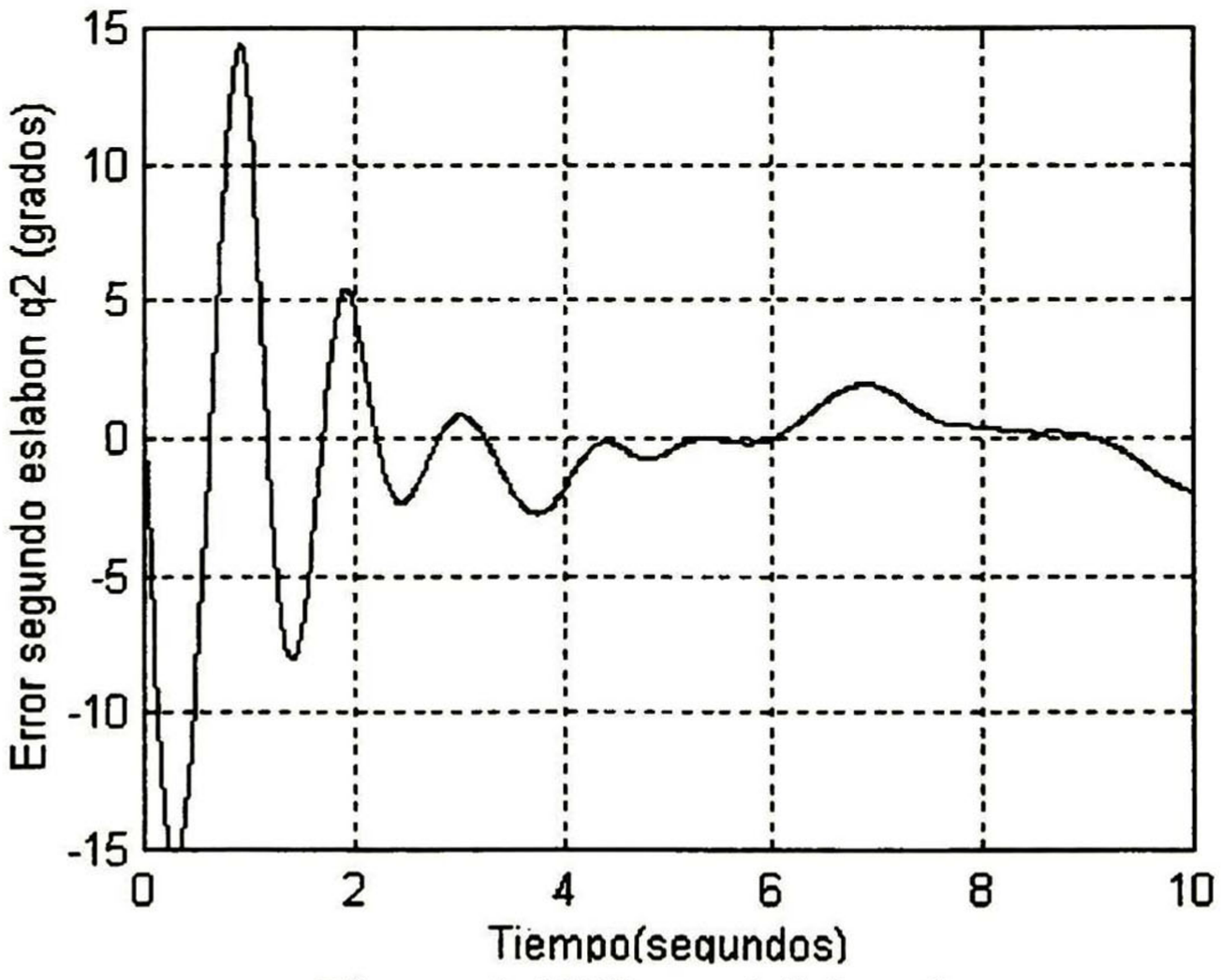


Figura 4.47 Error del ángulo q_2

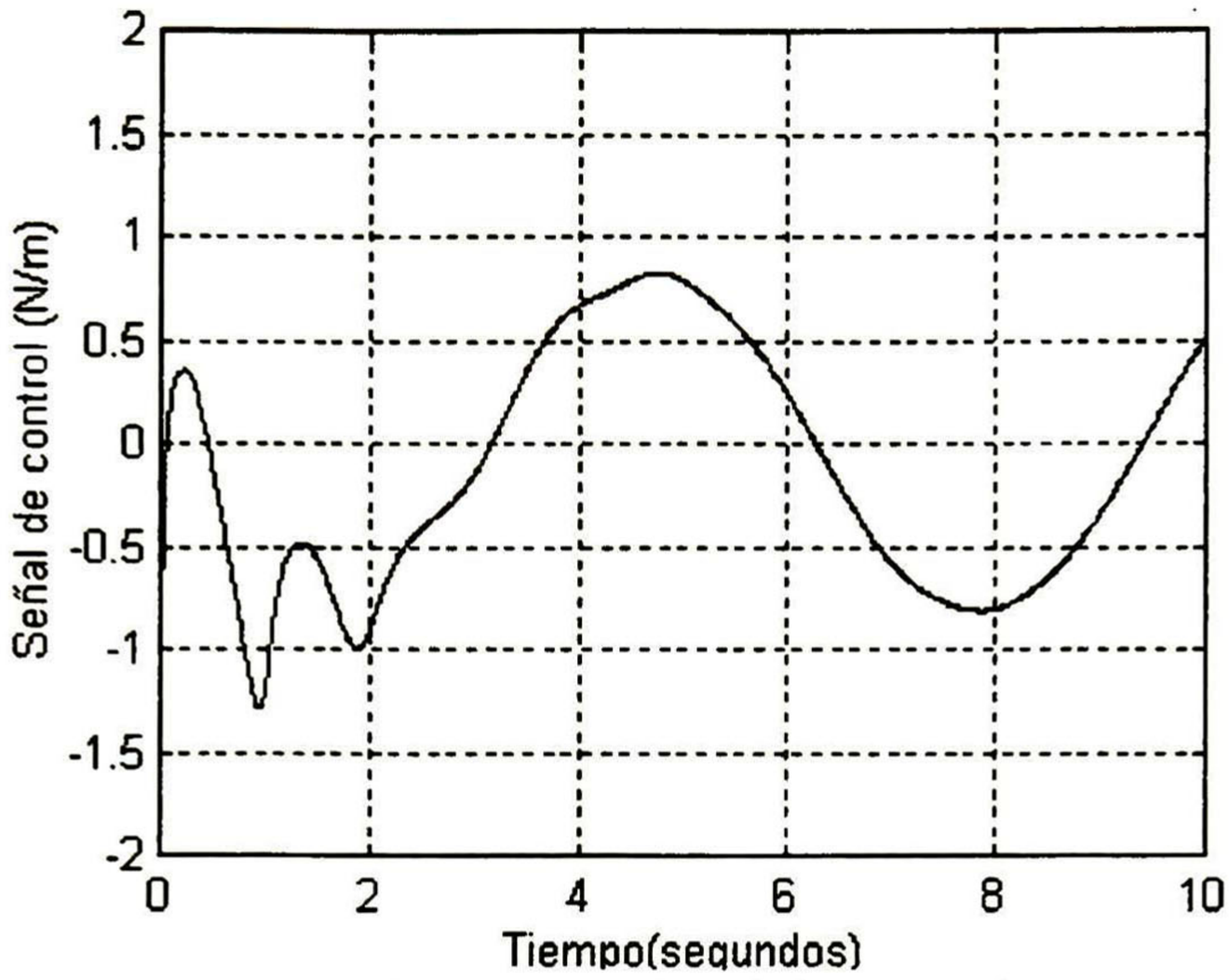


Figura 4.48 Señal de control

4.7 Implementación en tiempo real de seguimiento de trayectoria con PD difuso y regulador

A continuación se presentan los resultados en tiempo real al aplicar el esquema de estabilización y seguimiento de trayectoria. Primero se lleva al pendubot en el punto de equilibrio q_1=90° y q_2=0°, se mantiene ahí durante 5 seg. y después se hace oscilar la señal de referencia para comenzar con el seguimiento. La Tabla C.7 (Apéndice C) muestra las ganancias para éste controlador. Las Figuras 4.49 y 4.50 presentan la evolución en el tiempo y señal de referencia de los ángulos q_1 y q_2 respectivamente, la Figura 4.51 muestra la señal de control que no excede +/- 10 Newton-metros. Por último el error del primer y segundo eslabón (externo e interno) se presentan en la Figura 4.52 y 4.53 respectivamente.

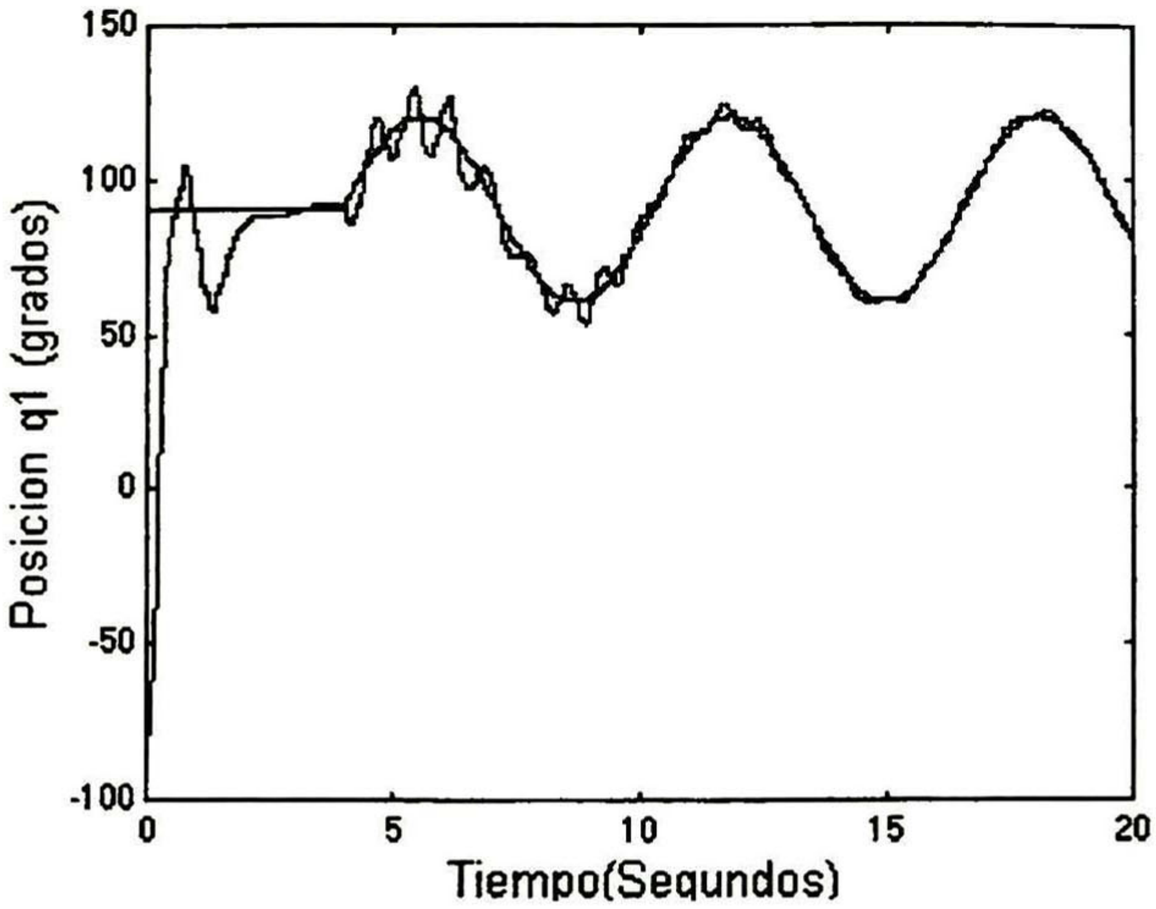


Fig. 4.49 Referencia y evolución de q_1

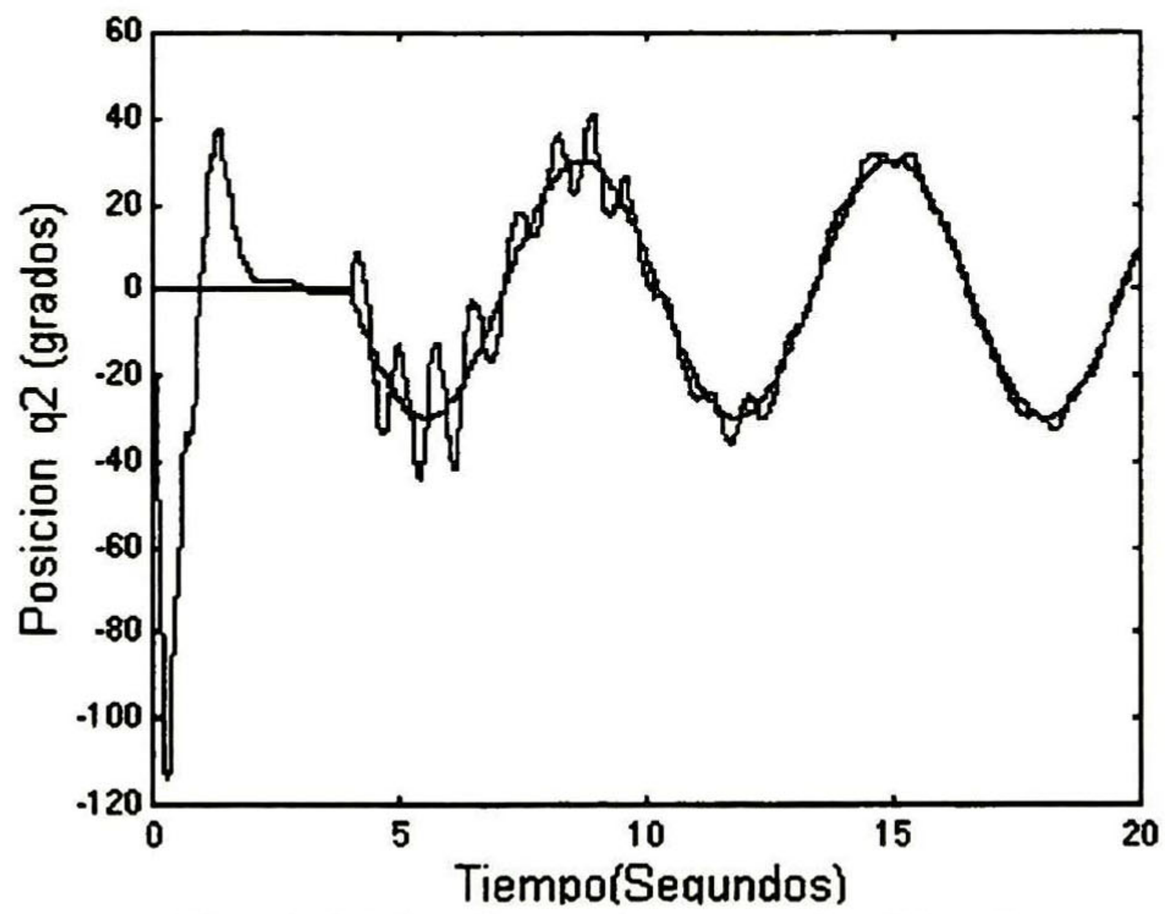


Fig. 4.50 Referencia y evolución de q_2

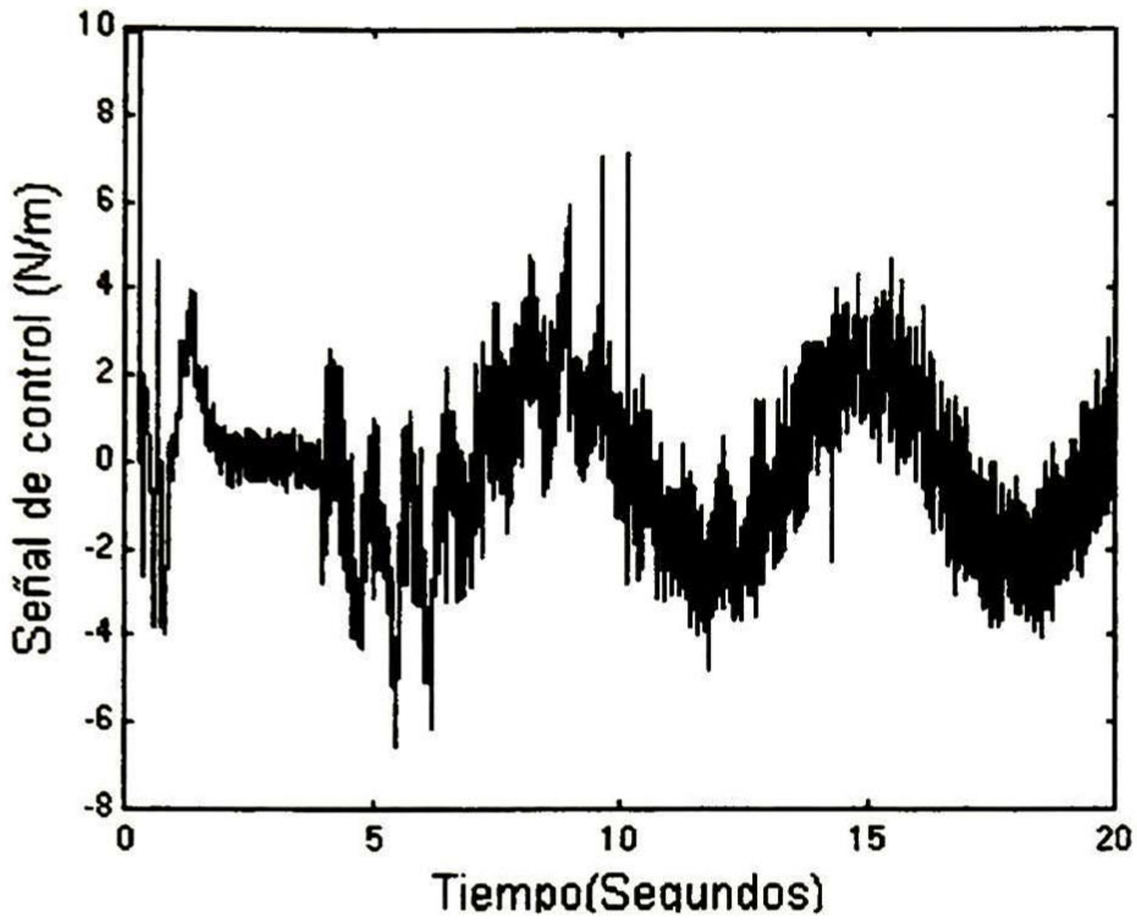


Fig. 4.51 Señal de control u

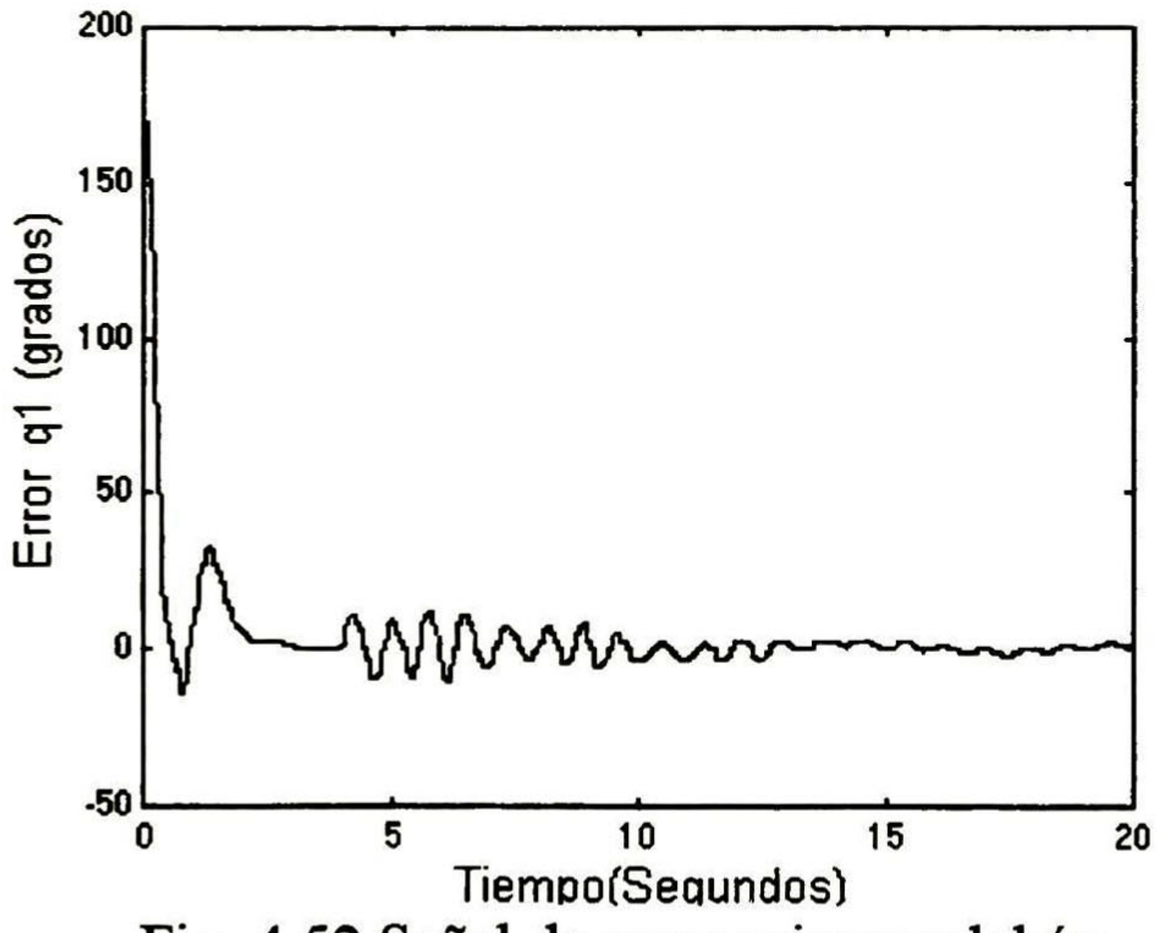


Fig. 4.52 Señal de error primer eslabón

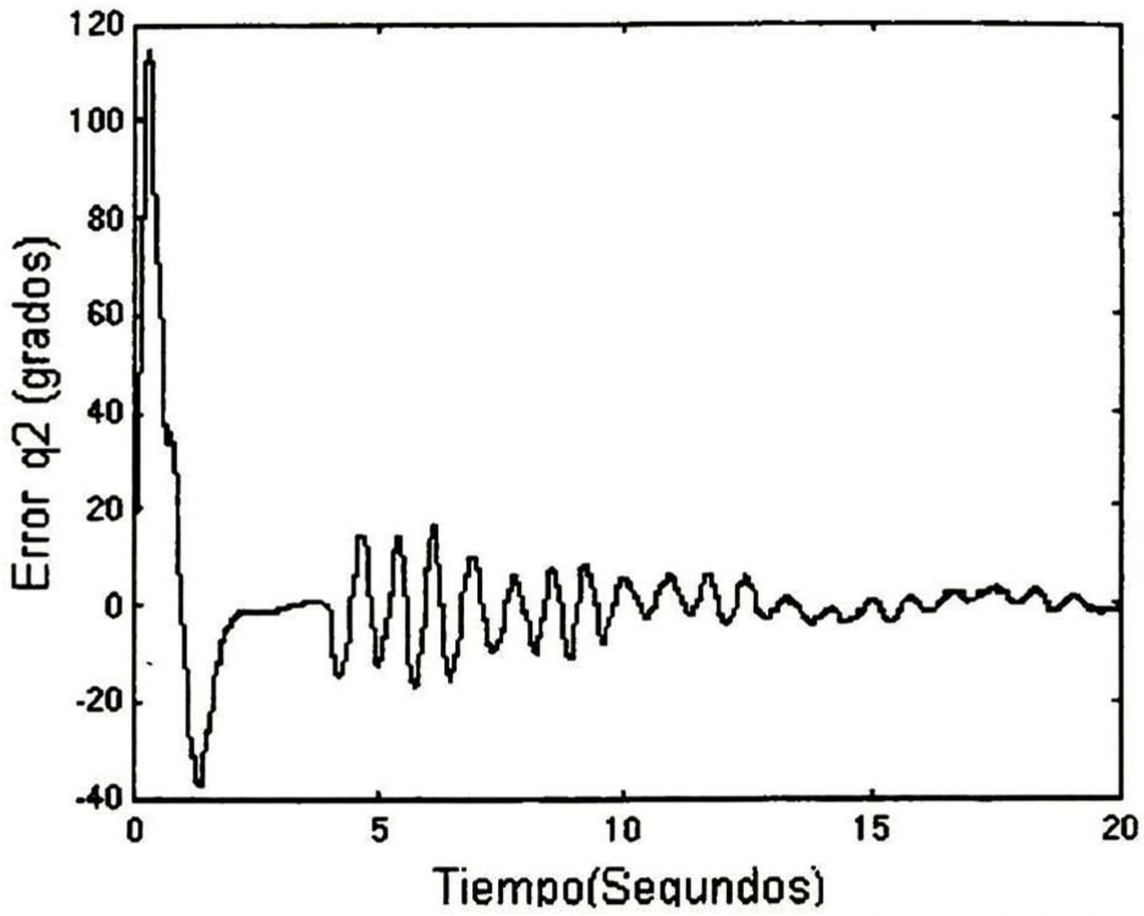


Fig. 4.53 Señal de error segundo eslabón

Como se puede observar claramente, el nuevo esquema presenta un mejor desempeño en el seguimiento de la trayectoria que el control PD difuso original. El error que se obtiene con el primer esquema propuesto varía de +/- 12 grados; con el segundo esquema, el PD Difuso con regulador, el error se reduce a +/- 3 grados.

En éste capítulo se propone inicialmente un nuevo esquema de control PD difuso, que incluye una función de conmutación. Este controlador es capaz de equilibrar verticalmente al pendubot. Para el seguimiento de trayectorias, se propone también un nuevo esquema que combina el regulador lineal y el control PD difuso. Ambos esquemas se validan tanto a nivel simulación cómo en tiempo real.

CAPÍTULO 5
CONCLUSIONES

En esta tesis se ha desarrollado un esquema de PD difuso muy simple con gran capacidad en aplicaciones reales. Esto se muestra en la aplicación directa al Pendubot, llevándolo al punto de equilibrio deseado, demostrando desempeño robusto ante perturbaciones y cambios de masa en el eslabón externo.

Para propósitos de comparación, se incluyó un controlador PD clásico que resultó ser poco eficiente con respecto al PD difuso ya que fue incapaz de llevar al punto deseado a un robot de dos eslabones completamente actuado sin incrementar los sobrepasos. Fue posible establecer condiciones de estabilidad para la aplicación de éste esquema para el control de robots completamente actuados. Los resultados en simulación muestran que el PD difuso presenta un mejor desempeño que el controlador PD clásico.

Los resultados experimentales de mayor impacto corresponden a la aplicación del PD difuso a robots subactuados. Incorporando una función de conmutación fue posible equilibrar al Pendubot en un punto de equilibrio inestable. Se pudo comprobar que el controlador PD difuso es robusto; se introdujeron perturbaciones en los dos eslabones (interno y externo) regresando al punto de equilibrio original. Además se experimentó haciendo cambiar el centro de masa del eslabón externo, cumpliéndose también los objetivos de control.

Cabe hacer notar que el controlador PD difuso no fue diseñado para hacer el seguimiento de trayectorias, esto se ve claramente cuando se trata de hacer seguimiento mostrando deficiencia en éste aspecto. Utilizando la teoría del

regulador lineal conjuntamente con el control PD difuso se logró el seguimiento de trayectorias, esto se implementó tanto en simulación cómo a nivel experimentación.

Como trabajo a futuro establecer que condiciones debe de cumplir el sistema subactuado para que el cálculo de v pueda ser determinado en el caso de seguimiento de trayectorias; así cómo demostrar que el sistema de control del Pendubot en lazo cerrado (con retroalimentación) es estable.

BIBLIOGRAFÍA

[1] Ramos L. Enrique, "Control de un Sistema Electromecánico Subactuado (Pendubot)", Tesis de Maestría, Dept. de Ing. Eléctrica, Sección Control Aut., Cinvestav, 1996.

[2] Berkemeier M., "Nonline

ar control of a two-link hoping robot", Ph. D Thesis, university of California, Berbeley, USA 1993.

[3] Lee M. A. and Smith M. H. , "Automatic design and tuning of a fuzzy system for controlling the acrobot using genetic algorithms, DSFS, and Meta-Rule Techniques", in Proc. Of NAFIPS'94, San Antonio, Texas, Dec. 1994.

[4] Smith M. H., Lee M. A., Ulieru M. and Gruver W. A., "Design Limitations of PD Versus Fuzzy Controllers for the Acrobot", Proc. 1997 IEEE Intl. Conference in Robotics and Automation, pp 1130-1135, Alburquerque, New Mexico, Usa, April 1997.

[5] Chen G. , Hsu Y. and Sánchez Edgar, "A Fuzzy PD Controller for Multi-Link Robot Control: Stability Analysis", Proc. 1997 IEEE Intl. Conference in Robotics and Automation, pp 1412-1417, Alburquerque, New Mexico, Usa, April 1997.

[6] Kailath Thomas, "Linear Systems", Prentice Hall, Englewood Cliffs, N. J., USA 1980.

[7] Khalil Hassan K. "Non-Linear Systems", Prentice Hall, New Jersey 1996.

[8] Driankov Dimiter, Frank Michael and Hellendoorn Hans, "An Introduction To Fuzzy Logic", Springer, USA 1996.

[9] Titli Andre "Design of Fuzzy Controllers: advances, open problems and applications" LAAS-CNRS and INSA, Toulouse, France 1997.

[10] Chen G., Malki H. and Li H., "New Design and Stability Analysis of Fuzzy Proportional-Derivative Control systems", IEEE Trans. on Fuzzy Systems, Vol. 2, pp 245-254, 1994.

[11] Spong, M. W., Vidyasagar, "Robot Dynamics And Control", John Wiley & Sons, Inc, New York, USA 1989.

[12] Jamshidi Mohamad, Vadiee Nader, J. Ross Timothy "Fuzzy Logic and Control Software and Hardware Application", Prentice Hall, New Jersey USA 1993.

[13] Kelly R. and Carelli R., "A class of nonlinear PD-type controllers for robots manipulators," Tech. Report, 1994.

[14] Peridskii, "Continuous control systems: Problem of absolute stability", Avtomatika I Telemekhanika, No. 12, pp. 5-11. 1969.

[15] Nakamura Shoichiro "Analisis Numerico y Visualizacion Grafica con Matlab", Prentice Hall, México 1997.

[16] Pappas Chris H.,Murray H. William "Microsoft C/C++7 Manual de Referencia", Mc Graw Hill, España 1994.

APENDICE A
T-normas y S-normas

Tabla A.1 T-normas y S-normas

T(a,b)	S(a,b)	Parámetro	Referencia
$min(a,b)$	$max(a,b)$		Zadeh (1973)
ab	$a+b-ab$		Bandler y Kohout (1980)
$max(a+b-1,0)$	$min(a+b,1)$		Lukasiewicz y Giles (1976)
$\begin{cases} x, & si\ y=1 \\ y, & si\ x=1 \\ 0, & otro \end{cases}$	$\begin{cases} x, & si\ y=0 \\ y, & si\ x=0 \\ 1, & otro \end{cases}$		Weber (1983)
$\dfrac{ab}{\gamma+(1-\gamma)(a+b-ab)}$	$\dfrac{a+b-(2-\gamma)ab}{1-(1-\gamma)ab}$	$\gamma>0$	Hamacher (1978)
$\dfrac{ab}{max(a,b,\alpha)}$	$\dfrac{a+b-ab-min(a,b,1-\alpha)}{max(1-a,1-b,\alpha)}$	$\alpha\in[0,1]$	Dubois y Prade (1986)
$\left[1+\sqrt[\lambda]{\left(\frac{1}{a}-1\right)^{\lambda}+\left(\frac{1}{b}-1\right)^{\lambda}}\right]^{-1}$	$\left[1+\sqrt[\lambda]{\left(\frac{1}{a}-1\right)^{-\lambda}+\left(\frac{1}{b}-1\right)^{-\lambda}}\right]^{-1}$	$\lambda>0$	Dombi (1982)
$max\left(1-\sqrt[\rho]{(1-a)^{\rho}+(1-b)^{\rho}},0\right)$	$min\left(\sqrt[\rho]{a^{\rho}+b^{\rho}},1\right)$	$\rho\geq 1$	Yager (1980)
$max\left(\dfrac{a+b-1+\lambda ab}{1+\lambda},0\right)$	$min(a+b+\lambda ab,1)$	$\lambda\geq -1$	Weber (1983)
$^{\omega}\log\left(1+\dfrac{(\omega^{a}-1)(\omega^{b}-1)}{\omega-1}\right)$	$1-{}^{\omega}\log\left(1+\dfrac{\omega^{1-a}+\omega^{1-b}}{\omega-1}\right)$	$\omega>0,\omega\neq 1$	Turksen (1986)

APENDICES

APENDICE B
PROGRAMAS EN MATLAB Y TABLAS DE VALORES DEL ROBOT COMPLETAMENTE ACTUADO

B.1 Modelo lineal del Rhino-Bot

```
%Archivo snlarm.m
%bloque del modelo no lineal del robot. Recibe como parámetros
%la derivada del ángulo theta1, el ángulo theta1 y la entrada u1
%que corresponden al primer eslabón, la entrada u2, el ángulo theta2 y
%la derivada del ángulo theta2, en un vector de 6X1 en el orden que se
%menciona y la salida es un vector de 2X1
%que produce la segunda derivada del ángulo theta1 y la segunda derivada
%del ángulo theta2

function y=snlarm(xin)

%valores de masa, longitud y gravedad utilizados
m1=15.91;
m2=11.36;
l1=0.432;
l2=0.432;
g=9.81;

%constantes
a1=((m1+m2)*l1^2)+(m2*l2^2);
a2=2*m2*l1*l2;
a3=m2*l2^2;
a4=(m1+m2)*g*l1;
a5=m2*g*l2;

%separación de las variables de entrada
dtheta1=xin(1,1);
theta1=xin(2,1);
dtheta2=xin(6,1);
theta2=xin(5,1);
U(1,1)=xin(3,1);
U(2,1)=xin(4,1);

%creación de las matrices
M=[a1+(a2*cos(theta2)),a3+(0.5*a2*cos(theta2));a3+(0.5*a2*cos(theta2)),a3];
C=[(a2*sin(theta2))*((dtheta1*dtheta2)+(0.5*dtheta2^2));(a2*sin(theta2))*0.5*dthet
a1^2];
```

```
G=[(a4*cos(theta1))+(a5*cos(theta1+theta2));a5*cos(theta1+theta2)];
%Calculo de la salida
y=-M^(-1)*C-M^(-1)*G+M^(-1)*U;
```

B.2 Función en Matlab del controlador difuso

```
%Archivo contfuzz.m
%Control difuso del sistema no lineal, recibe como parámetros
%el error, la derivada del error y da como salida u

function y=contfuz(xin)

%Variables utilizadas para el controlador difuso
L=10;
kp=7;
kd=1;

%separación de las variables de entrada
err=xin(1,1);
derr=xin(2,1);

e=err*kp;
r=derr*kd;


%Controlador difuso

if L>=abs(e)&abs(e)>=abs(r)
   y=(L*(e+r))/(2*((2*L)-abs(e)));
end

if L>=abs(r)&abs(r)>=abs(e)
   y=(L*(e+r))/(2*((2*L)-abs(r)));
end

if L<abs(e)&e>abs(r)
   y=0.5*(L+r);
end

if L<abs(r)&r>abs(e)
   y=0.5*(L+e);
end
```

```
if L<abs(e)&-e>abs(r)
   y=0.5*(-L+r);
end

if L<abs(r)&-r>abs(e)
   y=0.5*(-L+e);
end

if L<e&L<r
   y=L;
end

if L<-e&L<-r
   y=-L;
end

if (L<e&L<-r)|(L<-e&L<r)
   y=0;
end
```

Tabla B.1 Constantes del Rhino-bot

Constante	*Valor*
m_1	15.81
m_2	11.36
l	0.432
g	9.81

Tabla B.2 Ganancias del controlador PD clásico

Ganancias	*Valor*
Kp_1	680
Kp_2	550
Kv_1	100
Kv_2	90

Tabla B.3 Ganancias del controlador PD Difuso

Ganancias	*Valor*	*Ganancias*	*Valor*
L_1	30	L_2	30
Kp_1	4	Kp_2	4
Kd_1	0.5	Kd_2	0.5
Ku_1	1500	Ku_2	1500

APENDICE C
PROGRAMAS EN MATLAB Y TABLAS DE VALORES DEL ROBOT SUBACTUADO

C.1 Modelo no-lineal del pendubot

```
%snlpbot.m
%bloque del modelo no lineal del pendubot. Recibe como parámetros
%la derivada del ángulo theta1, el ángulo theta1 y la entrada u1
%que corresponden al primer eslabón, el ángulo theta2 y la derivada
%del ángulo theta2, en un vector de 5X1 en el orden que se
%menciona la salida que da este bloque es un vector de 2X1
%con la segunda derivada del ángulo theta1 y la segunda derivada
%del vector theta2
%este modelo se considera sin fricción

function y=snlarm(xin)

%valores de masa, longitud y gravedad utilizados
m1=0.5289;
m2=0.3346;
l1=0.26987;
l2=0.38417;
lc1=0.13494;
lc2=0.19208;
Izz1=0.013863;
Izz2=0.016749;
g=9.81;

%separación de las variables de entrada
dtheta1=xin(1,1);
theta1=xin(2,1);
dtheta2=xin(5,1);
theta2=xin(4,1);
u1=xin(3,1);
%m2=xin(6,1);


%creación de las matrices

%Matriz de inercias
D11=(m1*(lc1^2))+m2*((l1^2)+(lc2^2)+(2*l1*lc2*cos(theta2)))+Izz1+Izz2;
```

```
D12=m2*((lc2^2)+(l1*lc2*cos(theta2)))+Izz2;
D22=(m2*(lc2^2))+Izz2;
D=[D11,D12;D12,D22];

%Matriz de Coriolis-Centripeta
C11=-2*m2*l1*lc2*dtheta2*sin(theta2);
C12=-m2*l1*lc2*dtheta2*sin(theta2);
C21=m2*l1*lc2*dtheta1*sin(theta2);
C22=0;
C=[C11,C12;C21,C22];

%Vector que contiene el efecto de la gravedad
G11=(m1*g*lc1*cos(theta1))+(m2*g*l1*cos(theta1))+(m2*g*lc2*cos(theta1+theta
2));
G12=m2*g*lc2*cos(theta1+theta2);
G=[G11;G12];

%Vector de ángulos derivados
dtheta=[dtheta1;dtheta2];


%Vector de entrada
T=[u1;0];


%Calculo de la salida
y=-(D^(-1)*C*dtheta)-(D^(-1)*G)+(D^(-1)*T);
```

C.2 Función en Matlab del controlador difuso

```
%contfuz.m
%Control difuso del sistema no lineal, recibe como parámetros
%el error, la derivada del error y nos da a la salida u

function y=contfuzz(xin)

%separación de las variables de entrada
err=xin(1,1);
derr=xin(2,1);
L=xin(3,1);
kp=xin(4,1);
kd=xin(5,1);
```

```
e=err*kp;
r=derr*kd;

%Controlador difuso derivado de la IC región

if L>=abs(e)&abs(e)>=abs(r)
   y=(L*(e+r))/(2*((2*L)-(kp*abs(err))));
end

if L>=abs(r)&abs(r)>=abs(e)
   y=(L*(e+r))/(2*((2*L)-(kd*abs(derr))));
end

if L<abs(e)&e>abs(r)
   y=0.5*(L+r);
end

if L<abs(r)&r>abs(e)
   y=0.5*(L+e);
end

if L<abs(e)&-e>abs(r)
   y=0.5*(-L+r);
end

if L<abs(r)&-r>abs(e)
   y=0.5*(-L+e);
end

if L<e&L<r
   y=L;
end

if L<-e&L<-r
   y=-L;
end

if (L<e&L<-r)|(L<-e&L<r)
   y=0;
end
```

C.3 Función de Matlab para determinar la distancia de conmutación

```
%dista.m
%Calculo de la distancia y selección de la salida
%recibe como vectores theta1, theta2, u1, u2, ddes

function y=dista(xin)

%Variables utilizadas
theta1=xin(1,1);
theta2=xin(2,1);
u1=xin(3,1);
u2=xin(4,1);
ddes=xin(5,1);

l1=0.26987;
l2=0.38417;

%calculo de la distancia

dreal=sqrt((l1+l2-l1*sin(theta1)-
l2*sin(theta1+theta2))^2+(l1*cos(theta1)+l2*cos(theta1+theta2))^2);

%Si la ddes es mayor que la dreal deja pasar u1, de lo contrario
%deja pasar u2

if dreal>ddes
   y=u1;
else
   y=u2;
end
```

Tabla C.1 Constantes del Pendubot

Constante	*Valor*
Densidad del aluminio (ρ)	2700 kg/m^3
Longitud del eslabón 1 (l_1)	0.26987 m
Longitud media del eslabón 1 (lc_1)	0.13494 m
Longitud del eslabón 2 (l_2)	0.38417 m
Longitud media del eslabón 2 (lc_2)	0.19208 m
Espesor del eslabón 1 (h_1)	0.00635 m
Espesor del eslabón 2 (h_2)	0.009525 m
Ancho del eslabón 1 (w_1)	0.0508 m
Ancho del eslabón 2 (w_2)	0.0762 m
Momento de inercia 1 (Izz_1)	1.3863 $X\,10^{-2}\ kg-m^2$
Momento de inercia 2 (Izz_2)	1.6749 $X\,10^{-2}\ kg-m^2$
Masa del eslabón 1 (m_1)	0.5289 kg
Masa del eslabón 2 (m_2)	0.3346 kg
Gravedad (g)	9.81 kg/seg^2

Tabla C.2 Ganancias del controlador PD Difuso en simulación para estabilización

Excitador		Estabilizador	
Ganancias	*Valor*	*Ganancias*	*Valor*
L_1	8	L_3	100
Kp_1	12	Kp_3	-9
Kd_1	0.3	Kd_3	-2.5
Ku_1	1	Ku_3	10
L_2	8	L_4	100
Kp_2	2	Kp_4	-9
Kd_2	-0.5	Kd_4	-1.9
Ku_2	0.1	Ku_4	10

Tabla C.3 Ganancias del controlador PD Difuso en tiempo real para estabilización

Excitador		Estabilizador	
Ganancias	*Valor*	*Ganancias*	*Valor*
L_1	10	L_3	100
Kp_1	9	Kp_3	-9
Kd_1	1	Kd_3	-2.5
Ku_1	10	Ku_3	10
L_2	10	L_4	100
Kp_2	10	Kp_4	-9.95
Kd_2	0.001	Kd_4	-1.75
Ku_2	1	Ku_4	10

Tabla C.4 Ganancias del controlador PD Difuso en simulación para seguimiento sin regulador

Excitador		Para Seguimiento	
Ganancias	*Valor*	*Ganancias*	*Valor*
L_1	8	L_3	100
Kp_1	12	Kp_3	-9
Kd_1	0.3	Kd_3	-2.5
Ku_1	1	Ku_3	16.5
L_2	8	L_4	100
Kp_2	2	Kp_4	-9
Kd_2	-0.5	Kd_4	-2.2
Ku_2	0.1	Ku_4	14

Tabla C.5 Ganancias del controlador PD Difuso en tiempo real para seguimiento sin regulador

Excitador		Para Seguimiento	
Ganancias	*Valor*	*Ganancias*	*Valor*
L_1	10	L_3	100
Kp_1	9	Kp_3	-9
Kd_1	1	Kd_3	-4
Ku_1	10	Ku_3	15.5
L_2	10	L_4	100
Kp_2	10	Kp_4	-9
Kd_2	0.001	Kd_4	-3
Ku_2	1	Ku_4	14

Tabla C.6 Ganancias del controlador PD Difuso en simulación para seguimiento con regulador

Excitador		Para Seguimiento	
Ganancias	*Valor*	*Ganancias*	*Valor*
L_1	8	L_3	100
Kp_1	12	Kp_3	-9
Kd_1	0.3	Kd_3	-2.5
Ku_1	1	Ku_3	16.5
L_2	8	L_4	100
Kp_2	2	Kp_4	-9
Kd_2	-0.5	Kd_4	-2.2
Ku_2	0.1	Ku_4	14
		v_d	-0.385 $sen(\omega t)$

Tabla C.7 Ganancias del controlador PD Difuso en tiempo real para seguimiento con regulador

Excitador		Para Seguimiento	
Ganancias	*Valor*	*Ganancias*	*Valor*
L_1	10	L_3	100
Kp_1	9	Kp_3	-9
Kd_1	1	Kd_3	-4
Ku_1	10	Ku_3	15.5
L_2	10	L_4	100
Kp_2	10	Kp_4	-9
Kd_2	0.001	Kd_4	-3
Ku_2	1	Ku_4	14
		v_d	$-3.75\ sen(\omega t)$

APENDICE D
PROGRAMA EN TIEMPO REAL PARA EL CONTROL DEL PENDUBOT

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<stdlib.h>
#include <string.h>
#include "pchrt.h"
#include "counter.h"


//incializacion de constantes utilizadas y puertos de entrada salida
#define ENCADDRESS1 0x210
#define ENCADDRESS2 0x220
#define ADDRESS 0x300
#define ZR_SETTING 0
#define QUAD_COUNT 4
#define NUM_AXIS 2
#define ENCINIT 30000L
#define COUNT_NUMBER 5L
#define ENCDIV 4096
#define ANGDES1 M_PI/2
#define ANGDES2 0
#define B1 0.26987
#define B2 0.38417
#define PILA 4000
#define DDES 0.20

/*valores del primer controlador fuzzy que es fijo*/
#define L1 10
#define KP1 23
#define KD1 1
#define KU1 10

#define L2 10
#define KP2 10
#define KD2 0.01
#define KU2 1
```

```
float fuzzy(float err, float derr, float L, float kp, float kd);
float dista(float ang1, float ang2);

main()
{
 FILE *theta1, *theta2, *tiempo, *entrada;
 unsigned long state;
 unsigned int outputHi,outputLow;
 long int vuelta1=0,vuelta2=0;
 long int alarm,count;
 long int flag=1;
 long int digout=2048;
 long int j,i=0;
 long int sig=8;
 char str[25];
 long encA,encB;
 float f_digout;
 float ang1,ang2;
 float ang1_1[PILA],ang2_2[PILA];
 float err1,err2,err1_old,err2_old;
 float d_err1,d_err1_old1,d_err1_old2,d_err2,d_err2_old1,d_err2_old2;
 float dfal;
 float u1,u[PILA];
 float A,hora;

/*Inicializacion de valores para el control fuzzy que es variable*/
float L3=100;
float KP3=-9;
float KD3=-2.5;
float KU3=10;

float L4=100;
float KP4=-9.95;
float KD4=-1.85;
float KU4=10;

/*inicializacion de otras variables*/
 count=0;
 A=hora=0;
 err1=err2=err1_old=err2_old=0;
 d_err1=d_err1_old1=d_err1_old2=d_err2=d_err2_old1=d_err2_old2=0;
 clrscr();
```

```
//Poner salida del actuador en cero
outputHi=digout/16;
outputLow=16*(digout-outputHi*16);
outport(ADDRESS,outputLow);
outport(ADDRESS+1,outputHi);

//Inicializacion del DRC board Quad X4
zr_settings(ENCADDRESS1,ZR_SETTING);
zr_settings(ENCADDRESS2,ZR_SETTING);

//reset de contadores
reset_counter(ENCADDRESS1,QUAD_COUNT);
reset_counter(ENCADDRESS2,QUAD_COUNT);

//preset de contadores
preset_counter(ENCADDRESS1,ENCINIT);
preset_counter(ENCADDRESS2,ENCINIT);

//incializacion del kernel de tiempo
state=t_start();
if(!state)
{
 printf("No hay timer disponible\n");
 exit(-1);
}
else
{
 printf("Timer disponible, los bytes asignados son %i\n",state);
}

//inicializacion de la alarma
state=t_alarm_start();
if(!state)
{
 printf("Memoria dinamica insuficiente para asignar el timer\n");
 t_stop();
 exit(-1);
}
else
{
 printf("Memoria disponible, %i bytes asignados a la alarma\n",state);
}
```

```
//asignar a la alarma la resolucion de microsegundos
alarm=t_alarm_alloc(T_MSEC);
if(alarm==T_FAIL)
{
 printf("No hay mas alarmas disponibles\n");
 t_alarm_stop();
 t_stop();
 exit(-1);
}
else
{
 printf("Alarma asigana a microsegundos, es la alarma %i\n",alarm);
}


printf("Tomar el boton de seguridad y presionar cualquier tecla\n");
printf("para detener la accion de control, presionar nuevamente\n");
printf("cualquier tecla\n");
getch();

while(!kbhit())
{
if(flag)
 {
    flag=0;
    //setear la alarma, con el numero de cuentas y en modo autoreset
    t_alarm_set(alarm,COUNT_NUMBER,T_RESTART);
 }

 if(count==500)
 {
  A=M_PI/6;
  hora=0;

  KD3=-3;
  KU3=16.5;

  KD4=-2.5;
  KU4=14;
  getch();
 }
//lectura de los encoders
encA=read_counter(ENCADDRESS1);
```

```
encB=read_counter(ENCADDRESS2);

//conversion de la lectura en radianes
ang1=(encA-ENCINIT)*((2*M_PI)/ENCDIV)-M_PI/2-(vuelta2*2*M_PI);
ang2=(encB-ENCINIT)*((2*M_PI)/ENCDIV)-(vuelta1*2*M_PI);

if(ang1>3*M_PI/2)
{
 vuelta2=vuelta2+1;
}
else if(ang1<-3*M_PI/2)
{
 vuelta2=vuelta2-1;
}

if(ang2>2*M_PI)
{
 vuelta1=vuelta1+1;
}
else if(ang2<-2*M_PI)
{
 vuelta1=vuelta1-1;
}

//guardar los errores anteriores
err1_old=err1;
err2_old=err2;

//calculo del nuevo error
err1=ANGDES1-ang1;
err2=ANGDES2-ang2;

//guardar los errores de velocidades anteriores
d_err1_old2=d_err1_old1;
d_err1_old1=d_err1;
d_err2_old2=d_err2_old1;
d_err2_old1=d_err2;

//calculo de las velocidades del error
d_err1=(err1-err1_old)/(COUNT_NUMBER*0.001);
d_err2=(err2-err2_old)/(COUNT_NUMBER*0.001);

//calculo de las velocidades del error, mediante la media de las tres
```

```
//velocidades de error anteriores eliminando el ruido numerico
d_err1=(d_err1+d_err1_old1+d_err1_old2)/3;
d_err2=(d_err2+d_err2_old1+d_err2_old2)/3;

//Calculo del error con seguimiento de trayectoria
err1=err1+A*sin(hora);
err2=err2-A*sin(hora);
//Calculo de la derivada del error con seguimiento de trayectoria
d_err1=d_err1+A*cos(hora);
d_err2=d_err2-A*cos(hora);

   //calculo de la distancia faltante
   dfal=dista(ang1,ang2);

   //utilizacion del primer controlador fuzzy, cuando la distancia
   //faltante es mayor a la distancia deseada (DDES), este controlador
   //nos hace llevar al los eslabones a una region cercana de la vertical
   if(dfal>DDES)
   {

u1=(fuzzy(err1,d_err1,L1,KP1,KD1)*KU1)+(fuzzy(err2,d_err2,L2,KP2,KD2)*KU2
);
   }

   //utilizacion del segundo eslabon cuando la distancia faltante es
   //menor a la distancia deseada para el cambio de controlador (DDES),
   //este controlador nos mantiene al pedubot en la posicion vertical
   //o hace seguimiento de trayectoria
   else
   {

u1=(fuzzy(err1,d_err1,L3,KP3,KD3)*KU3)+(fuzzy(err2,d_err2,L4,KP4,KD4)*KU4
);
   }

//limitar la salida de control
if(u1>9.9)
{
 u1=9.9;
}
if(u1<-9.9)
{
 u1=-9.9;
```

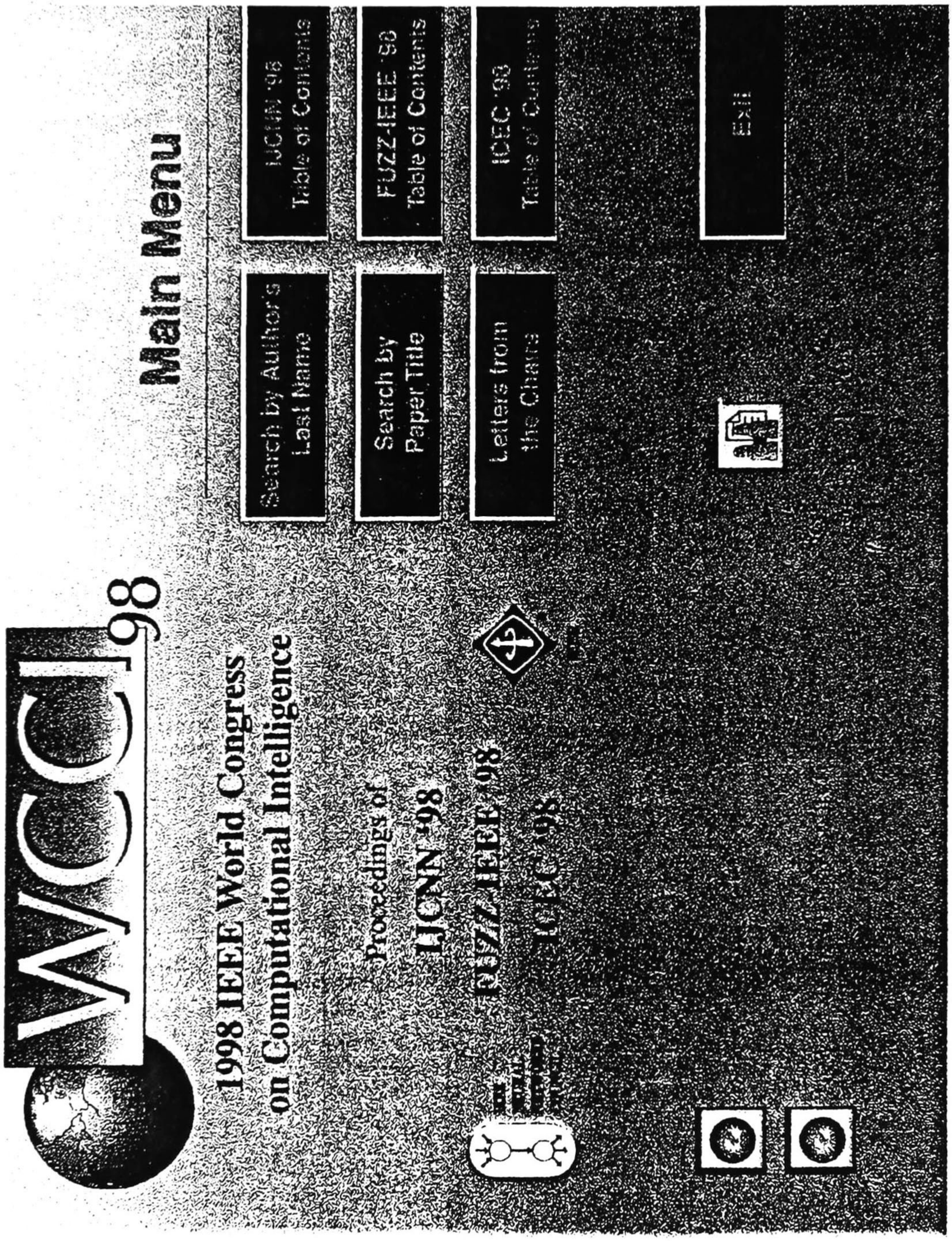
```
}
//guardar en memoria los valores de angulos
if(i<PILA)
{
 ang1_1[i]=ang1*57.3;
 ang2_2[i]=ang2*57.3;
 u[i]=u1;
 i=i+1;
}

//conversion de la salida a salida digital
f_digout=2048+204.8*u1;
digout=(int)f_digout;
if((f_digout-(float)digout)>0.5)
{
  digout=digout+1;
}

//poner la salida de control
outputHi=digout/16;
outputLow=16*(digout-outputHi*16);
outport(ADDRESS,outputLow);
outport(ADDRESS+1,outputHi);


printf("angulo1: %f\t angulo2: %f\t salida: %f\n",ang1*57.3,ang2*57.3,u1);

//checar la alarma
state=t_alarm_check(alarm);
while(!state)
{
  state=t_alarm_check(alarm);
}
count=count+1;
hora=horà+COUNT_NUMBER*0.001;
 }
//abrir archivos para almacenar angulos en modo escritura
theta1=fopen("ang1.mat", "w");
if(theta1==NULL)
{
  printf("No se pueden inicializar archivos para guardar datos\n");
  exit(-1);
}
```

```
theta2=fopen("ang2.mat", "w");
if(theta2==NULL)
{
  printf("No se pueden inicializar archivos\n");
  exit(-1);
}

tiempo=fopen("tiempo.mat", "w");
if(tiempo==NULL)
{
  printf("No se pueden inicializar archivos\n");
  exit(-1);
}

entrada=fopen("entrada.mat", "w");
if(entrada==NULL)
{
  printf("No se pueden inicializar archivos\n");
  exit(-1);
}

//guardar en los archivos los ang1, ang2 y tiempo
j=i;
printf("\n\nCreando archivos ang1.mat, ang2.mat y tiempo.mat de datos\n");
for(i=0;i<j;i++)
 {
   gcvt(ang1_1[i],sig,str);
   strcat(str,"  ");
   fputs(str,theta1);
   gcvt(ang2_2[i],sig,str);
   strcat(str,"  ");
   fputs(str,theta2);
   gcvt(i*0.001*COUNT_NUMBER,sig,str);
   strcat(str,"  ");
   fputs(str,tiempo);
   gcvt(u[i],sig,str);
   strcat(str,"  ");
   fputs(str,entrada);

  }

//fin del control poner en cero la salida
digout=2048;
```

```
    outputHi=digout/16;
    outputLow=16*(digout-outputHi*16);
    outport(ADDRESS,outputLow);
•   outport(ADDRESS+1,outputHi);

    //cerrar archivos
    fclose(theta1);
    fclose(theta2);
    fclose(tiempo);
    fclose(entrada);

   //detener alarmas
   t_alarm_stop();
   t_stop();
   return(0);
}


//Control fuzzy, recibe los parametros de error, la derivada del error
//el parametro L, Kp y Kd, nos regresa el valor del actuador U
float fuzzy(float err, float derr, float L, float kp, float kd)
{
float e,r;

e=err*kp;
r=derr*kd;

if (L>=abs(e)&&abs(e)>=abs(r))
   {
    return((L*(e+r))/(2*((2*L)-(kp*abs(err)))));
   }

else if (L>=abs(r)&&abs(r)>=abs(e))
  {
   return((L*(e+r))/(2*((2*L)-(kd*abs(derr)))));
  }

else if (L<abs(e)&&e>abs(r))
  {
   return(0.5*(L+r));
  }

else if (L<abs(r)&&r>abs(e))
```

```
 {
  return(0.5*(L+e));
 }

else if (L<abs(e)&&-e>abs(r))
 {
  return(0.5*(-L+r));
 }

 else if (L<abs(r)&&-r>abs(e))
 {
  return(0.5*(-L+e));
 }

else if (L<e&&L<r)
 {
  return(L);
 }

else if (L<-e&&L<-r)
 {
  return(-L);
    }

   else if ((L<e&&L<-r)||(L<-e&&L<r))
    {
     return(0);
    }
return(0);
}

//Funcion que calcula la distancia que falta para llegar a la posicion
//vertical, recibe como parametros el angulo del primer eslabon
//y el angulo del segundo eslabon, regresa el valor de la distancia
float dista(float ang1, float ang2)
{
  return(sqrt(pow((B1+B2-B1*sin(ang1)-
B2*sin(ang1+ang2)),2)+pow((B1*cos(ang1)+B2*cos(ang1+ang2)),2)));
}
```

APENDICE E
ARTICULOS PRESENTADOS



MAIN MENU

Y. Ding, H. Ying, S. Shao - China Textile University, CHINA

Improved Performance of Self-Organising Fuzzy Controller (SOC) in pH Control
J.-P. Ylen - Helsinki University, FINLAND

A New Approach to Adaptive Fuzzy Control
H.-J. Kang, H. Son, C. Kwon, M. Park - Yonsei University, KOREA

Guaranteed Cost Design of Continuous-Time Takagi-Sugeno Fuzzy Controllers via Linear Matrix Inequalities
A. Jadbabaie - California Institute of Technology, USA
M. Jamshidi - University of New Mexico, USA
A. Titli - LAAS du CNRS, and INSA, FRANCE

An Approach to the Design of MIMO Fuzzy Controllers in Cases of Incomplete Expert Knowledge
B. Vidolov, C. Melin - Universite de Technologie de Compiegne, FRANCE

Fuzzy Variable Structure Control for MIMO Systems
Y.-C. Hsu, H. Malki - University of Houston, USA

Dynamic Fuzzy Control and System Stability For The Acrobot
M. Smith - University of California-Berkeley, USA
T. Zhang, W. Gruver - Simon Fraser University, CANADA

Fuzzy Control Design for the Pre-Specified Trajectory Tracking With Sliding Mode
H.-R. Lin, W.-J. Wang - National Central University, TAIWAN

Combination of Sliding Mode Controller and PI Controller Using Fuzzy Logic Controller
L. Wong, F. Leung, P.Tam - The Hong Kong Polytechnic University, HONG KONG

Fuzzy PD Scheme for Underactuated Robot Swing-up Control
E. Sanchez, L. Nuno - Unidad Guadalajara, MEXICO
Y. Hsu, G. Chen - University of Houston, USA

A Fuzzy Control System for Multiple Cooperating Tentacle Robots
M. Ivanescu, V. Stoian - University of Craiova, ROMANIA

Model-Based Fuzzy Control of TORA System: Fuzzy Regulator and Fuzzy Observer Design via LMIs
K. Tanaka, T. Taniguchi - Kanazawa University, JAPAN
H. Wang - Duke University, USA

The Self-Organising Fuzzy PID Controller
H. Kazemian - Ravensbourne College, UNITED KINGDOM

SIRMs Dynamically Connected Fuzzy Inference Model and PID Controller
N. Yubazaki - Mycom, Inc., JAPAN
J. Yi - Mycom Inc., JAPAN
K. Hirota - Tokyo Institute of Technology, JAPAN

Self-Learning Fuzzy Controller with a Fuzzy Supervisor
C.-E. Perng, Y.-Y. Chen - National Taiwan University, TAIWAN

Variable Structured Properties in Fuzzy Logic Control
H. Li, S. Tso - City University of Hong Kong, HONG KONG

Fuzzy System Identification and Predictive Control of Load System in Power Plant
Z. Huaguang, Z. Bien - Korea Institute of Science and Technology, KOREA

Design of a Variable Structure Controller Using Fuzzy Variable Boundary Layer
H. Lee, H. Son, E. Kim, M. Park - Yonsei University, KOREA

Fuzzy PD Scheme for Underactuated Robot Swing-up Control

Edgar Sanchez[1], Luis A. Nuño[1], Ya-Chen Hsu[2], and Guanrong Chen[2]

1. CINVESTAV, Unidad Guadalajara. Apartado Postal 31-438, Guadalajara, Jalisco C.P 44550, Mexico
e-mail:sanchez@gdl.cinvestav.mx
2. Department of Electrical and Computer Engineering, University of Houston. Houston, TX 77204, USA
e-mail:gchen@uh.edu

Abstract

In this paper, a Multi-Input Multi-Output (MIMO) fuzzy Proportional-Derivative (PD) controller, equipped with a Dynamic Switching Fuzzy System (DSFS), is applied to swing-up an underactuated robot: the so-called Pendubot. This mechanism consists of a double pendulum actuated only at the first joint. Included simulations illustrate the applicability of the proposed scheme. The new controller design and its application constitute our original contribution.

1. Introduction

In recent years, there has been increasing interest in underactuated mechanism; that is, mechanical systems having more joints than control actuators. Different control schemes, derived from nonlinear systems theory [1,2,3], classical control techniques [4], or intelligent control methodologies [5,6], have been proposed.

In [5], practical techniques are introduced for the automatic tuning, based on Genetic Algorithms (GA), of an underactuated mechanism fuzzy control. A comparison, between the PD control proposed in [4] and the fuzzy control discussed in [5], is presented in [6]. Except for the PD scheme [4], all of the mentioned schemes are not easy to implement.

In this paper, we present the design and application of a new fuzzy PD scheme to underactuated robot swing-up control. We reconsider the controller introduced in [7], based on a very simple structure as in [8]. Moreover, we incorporate with it a DSFS, to automatically change the membership functions of the fuzzy PD scheme. The selected robot is an underactuated two-link mechanism known as the *Pendubot*, in which the first link is actuated and the second one free. The new scheme and its application constitute our original contribution.

The paper is organized as follows: in section 2, we briefly discuss the MIMO Fuzzy PD controller. Section 3 constitutes the main part of this paper, where we describe the application of the new scheme to the *Pendubot* swing up control. Simulation results are included in section 4. Finally, we give some relevant conclusions in section 5.

2. MIMO Fuzzy PD Control

For the sake of completeness, we first briefly review the basic structure and formulation of the MIMO fuzzy PD controller, originally presented in [7].

The structure of this MIMO fuzzy PD controller, which is a continuous-time controller, is based on the two inputs, one output controller shown in Figure 1, from which it can be seen that

$$u(t) = K_u u^*(t) \qquad (2.1)$$

where K_u is a fuzzy control gain determined by the designer and $u^*(t)$ is the output of the fuzzy controller.

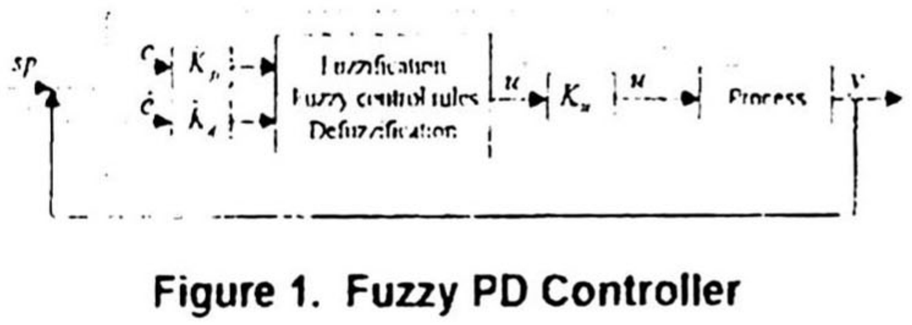


Figure 1. Fuzzy PD Controller

(i) *Fuzzification*: The two inputs are: the error signal $e(t)$ and its rate of change $\dot{e}(t)$; the only one output, $u(t)$, is fed to the controlled process. Membership functions are shown in Figs. 2 and 3, respectively, where the same membership function is applied for both $e(t)$

and $\dot{e}(t)$ for simplicity. To employ the same membership function on two inputs, two scaling factors, K_p and K_d, are used to magnify $e(t)$ and $\dot{e}(t)$. L and H are two positive constants to be determined.

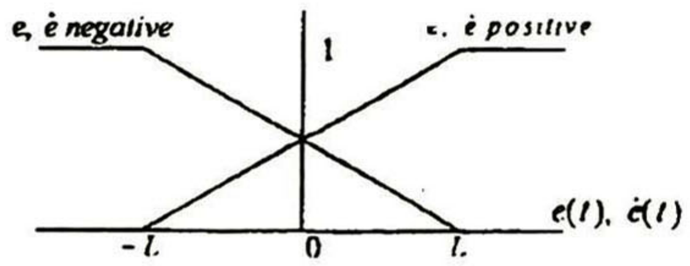


Figure 2. Input Membership Functions

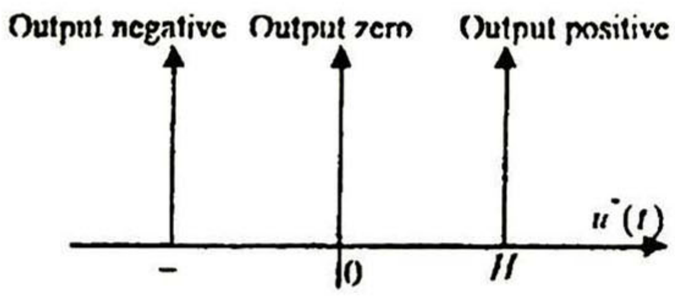


Figure 3. Output Membership Function

(ii) *Fuzzy control rules*: They are established as follows:

(R1)IF e = ep AND $\dot{e}$ = rp THEN output = op
(R2)IF e = ep AND $\dot{e}$ = rn THEN output = oz
(R3)IF e = en AND $\dot{e}$ = rp THEN output = oz
(R4)IF e = en AND $\dot{e}$ = rn THEN output = on

In these rules, e and $\dot{e}$ are defined as above, output is the output of the defuzzification mechanism, u^*, "ep" means "error positive" and "op" means "output positive," etc. Moreover, "AND" is the Zadeh's logical "AND" For a detailed interpretation of these rules, see [7,8].

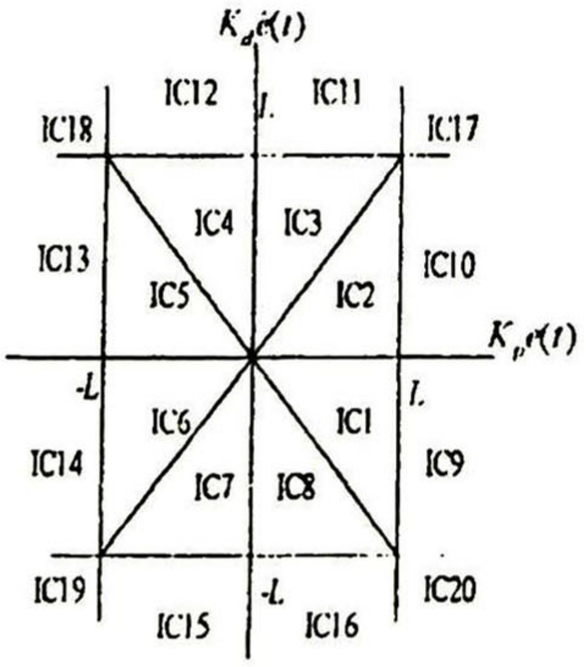


Figure 4. IC Regions

(iii) *Defuzzification*: The commonly used "center of mass" formula is employed to defuzzify the output of the control action of the fuzzy PD controller [8].

For this fuzzy PD controller, the input value-ranges are actually decomposed into twenty adjacent input-combination (IC) regions, as shown in Figure 4. The control rules (R1-R4), membership functions and IC regions, are used to evaluate an appropriate fuzzy control law for each region as follows [8]:

$$u_{PD}(t) = K_u u^*(t)$$

where

$$u^*(t) = \frac{L}{2(2L - K_p|e(t)|)}[K_p e(t) + K_d \dot{e}(t)] \qquad \text{in IC1, IC2, IC5, IC6.}$$

$$= \frac{L}{2(2L - K_d|\dot{e}(t)|)}[K_p e(t) + K_d \dot{e}(t)] \qquad \text{in IC3, IC4, IC7, IC8.}$$

$$= \frac{1}{2}[L + K_d \dot{e}(t)] \qquad \text{in IC9, IC10.}$$

$$= \frac{1}{2}[L + K_p e(t)] \qquad \text{in IC11, IC12.}$$

$$= \frac{1}{2}[-L + K_d \dot{e}(t)] \qquad \text{in IC13, IC14.}$$

$$= \frac{1}{2}[-L + K_p e(t)] \qquad \text{in IC15, IC16.}$$

$$= L \qquad \text{in IC17.}$$

$$= -L \qquad \text{in IC19.}$$

$$= 0 \qquad \text{in IC18, IC20.}$$

The dynamic equation of a rigid n-link manipulator can be written as

$$D(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + G(\theta) = \tau \qquad (2.2)$$

where $\theta \in R^n$ denotes the joint position vector of the arm, $\tau \in R^n$ is the applied joint torque vector, $D(\theta) \in R^{n \times n}$ is the inertia matrix, $C(\theta, \dot{\theta})\dot{\theta} \in R^n$ represents the vector of Coriolis and centrifugal torques, and $G(\theta) \in R^n$ denotes the gravitational torque.

To control such a multivariable system, in [7], a set of fuzzy PD controllers are proposed as follows:

$$\tau_i = \sum_{j=1}^{n} u_{ij} \qquad i = 1, \ldots, n \qquad (2.3)$$

where each u_{ij}, a fuzzy PD sub-controller derived as above, employs inputs (e_j and $\dot{e}_j$) from the j'th joint, and produces a torque output the i'th joint. For instance, τ_n consists of only one element u_{nn}, while τ_1 has n terms: $u_{11} + u_{12} + ... + u_{1n}$.

3. The Pendubot Control

A. Mathematical Model

A scheme of this underactuted robot is shown in Figure 5. It has two links, in which the first link is actuated and the second one free.

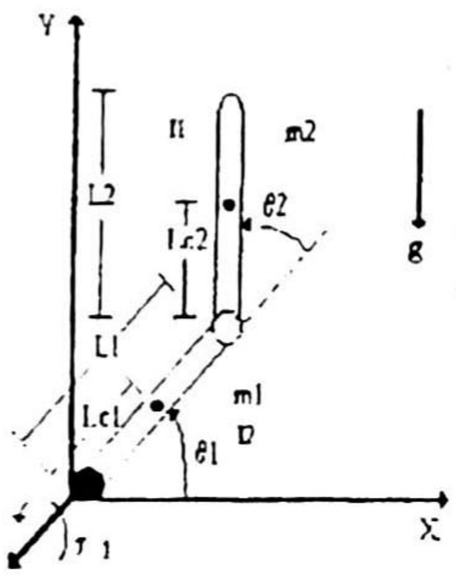


Figure 5. Pendubot Scheme

In Figure 5, g is the gravity acceleration, Ii (i=1,2), is the i'th link momentum of inertia, Li is the i'th link length, Lci is the distance from the joint to the mass center for the i'th link, mi is the the i'th link mass. X is the horizontal axis, Y is the vertical axis. $\theta 1$ is the angle between the horizontal axis and the first link. $\theta 2$ is the angle between the two links, and $\tau 1$ is the first joint input torque.

If only m joints are actuated, the vector θ can be partitioned, without loss of generality, as (θ_1, θ_2), where $\theta_1 \in \Re^m$ represents the actuated joints, and $\theta_2 \in \Re^{n-m}$ represents the unactuated ones. So, for the *Pendubot*, the dynamic model (2.2) can be decomposed as:

$$\begin{bmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ 0 \end{bmatrix} \qquad (3.1)$$

where

$$D_{11} = m_1 l_{c1}^2 + m_2\left(l_1^2 + l_{c2}^2 + 2 l_1 l_{c2} \cos(\theta_2)\right) + I_1 + I_2,$$

$$D_{12} = m_2\left(l_{c2}^2 + l_1 + l_{c2} \cos(\theta_2)\right) + I_2,$$

$$D_{22} = m_2 l_{c2}^2 + I_2$$

$$C_1 = -2 m_2 l_1 l_{c2} \dot{\theta}_1 \dot{\theta}_2 \sin(\theta_2) + m_2 l_1 l_{c2} \dot{\theta}_2^2 \sin(\theta_2)$$

$$C_2 = m_2 l_1 l_{c2} \dot{\theta}_1^2 \sin(\theta_2)$$

$$G_1 = m_1 g l_1 \cos(\theta_1) + m_2 g l_1 \cos(\theta_1) + m_2 g l_{c2} \cos(\theta_1 + \theta_2)$$

$$G_2 = m_2 g l_{c2} \cos(\theta_1 + \theta_2)$$

B. Control Scheme

To control the Pendubot, we propose to use the MIMO fuzzy PD controller discussed in section 2. For this application, it is composed of two fuzzy sub-controllers, one for each joint, whose outputs are added to calculate τ_1 (the control action to the actuated joint). A scheme of this controller is shown in Figure 6.

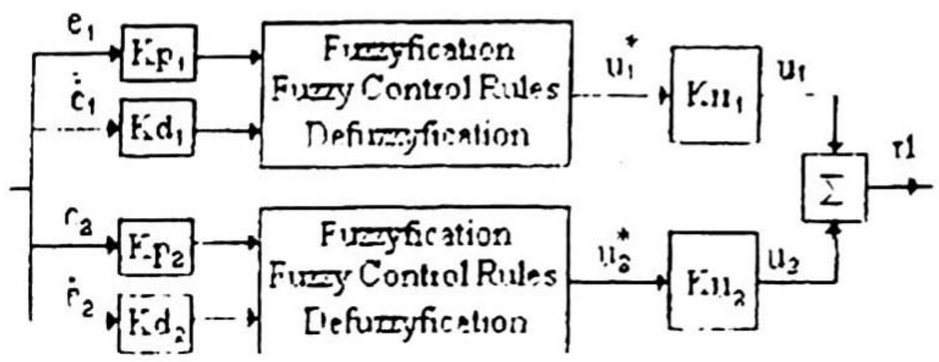


Figure 6. MIMO Fuzzy PD Controller

To control an underactuated system is a challenging task [1,2,3,5]. In order to obtain a more powerful controller, based on [6], we incorporate with it a switching function. So, we propose a new scheme composed by two MIMO PD fuzzy controllers as the one shown in Figure 6, with exactly the same structure, but a different constant L for each one of them. The switching function selects which one to apply. This new controller can be seen as a DSFS, which adapts the membership functions of the controller. The complete scheme is presented in Figure 7.

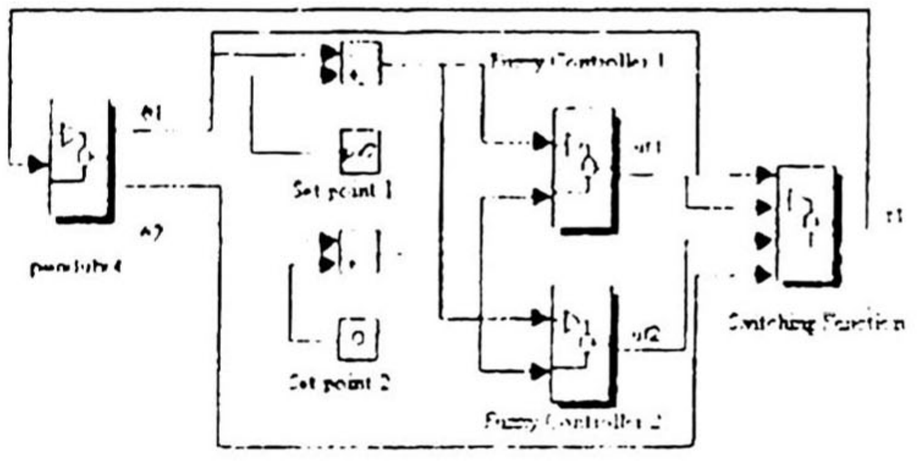


Figure 7. DSFS Controller

The switching function is based on the distance (d) from the tip of the second link to the final desired position. This function is easily implemented by the following rule:

If d is longer than d_1 then apply Fuzzy Controller 1, else apply Fuzzy Controller 2.

The distance d can be easily deducted from Figure 5. as

$$d=\sqrt{\left[L_1+L_2-L_1\sin(\theta_1)-L_2\sin(\theta_1+\theta_2)\right]^2+\left[L_1\cos(\theta_1)+L_2\cos(\theta_1+\theta_2)\right]^2} \quad (3.2)$$

We remark that to avoid possible jumps in control actions, it is also possible to introduce a new fuzzy membership function to smoothly connect the two switching controllers. We will not pursue further on this issue in the present paper.

4. Simulation Results

The control task is to bring the *Pendubot* from a straight-down resting position to a balanced straight-up vertical position (as shown in Figure 8). Initial and goal positions are (-90 degrees, 0 degrees) and (90 degrees, 0 degrees) respectively.

Comment 1 – For this control task, it is quite natural to select d as the variable decision for the switching function. Its value was determined experimentally.

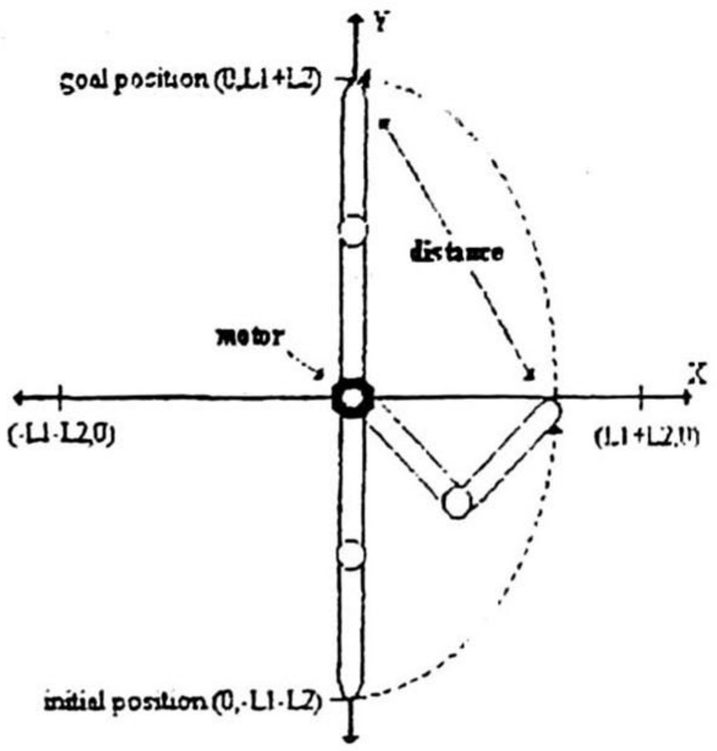


Figure 8. Swing-up Control

All the simulations are implemented in SIMULINK and MATLAB[1]. We simulate the model (3.1), using the following parameters:

g = 9.81 meters/seg^2
I_1 = 0.013863, I_2 = 0.016749
m_1 = 0.5289 kilograms, m_2 = 0.3346 kilograms.
l_1 = 0.26897 meters, l_2 = 0.38417 meters
l_{c1} = 0.13494 meters, l_{c2} =0.19208 meters

These parameters correspond to the real *Pendubot* in our automatic control laboratory.

We select $L = H = 8$ for Fuzzy Controller 1, and $L = H = 100$ for Fuzzy Controller 2. For the switching function, we choose $d_1 = 0.15$ meters.

We present the time evolution for θ_1 and θ_2 in Figure 9 and Figure 10, respectively. Figure 11 shows the control signal applied to the *Pendubot*. As can be seen, the proposed algorithm is able to swing-up this underactuated robot, with almost no oscillations. The maximum value of the control signal is less than the motor maximum torque (10 Newton-meter).

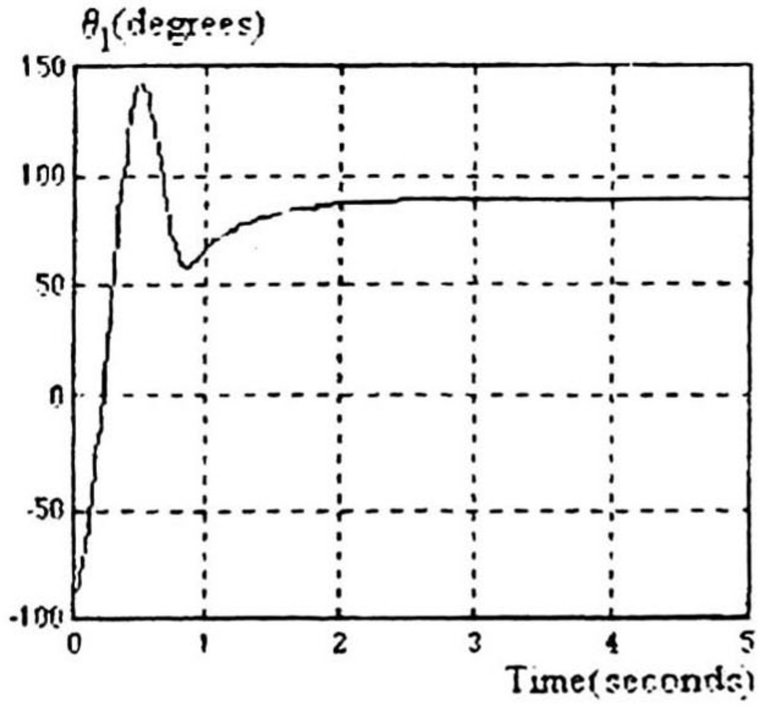


Figure 9. θ_1 Time Evolution

[1] SIMULINK and MATLAB are registered trademarks of The MathWorks Inc.

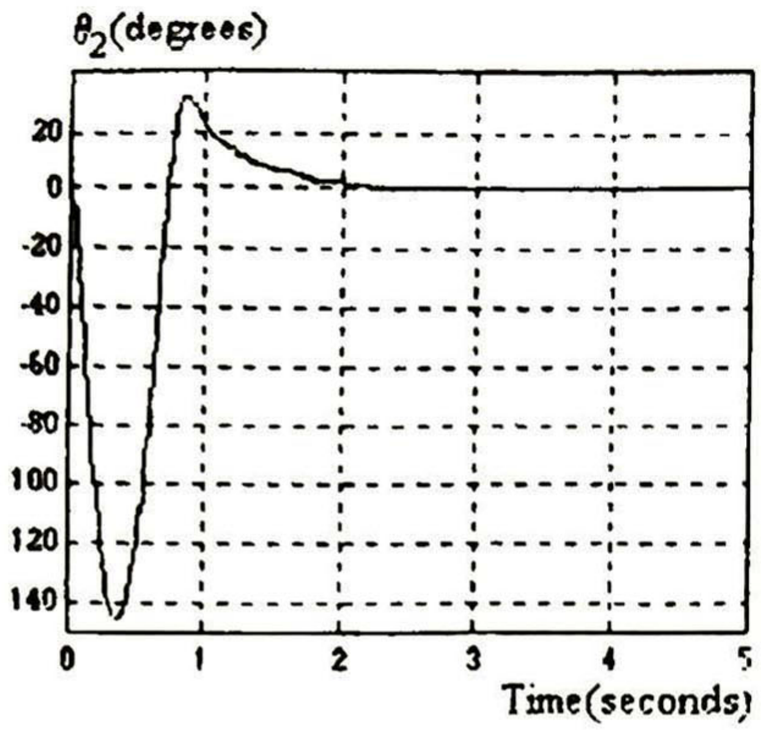


Figure 10. θ_2 Time Evolution

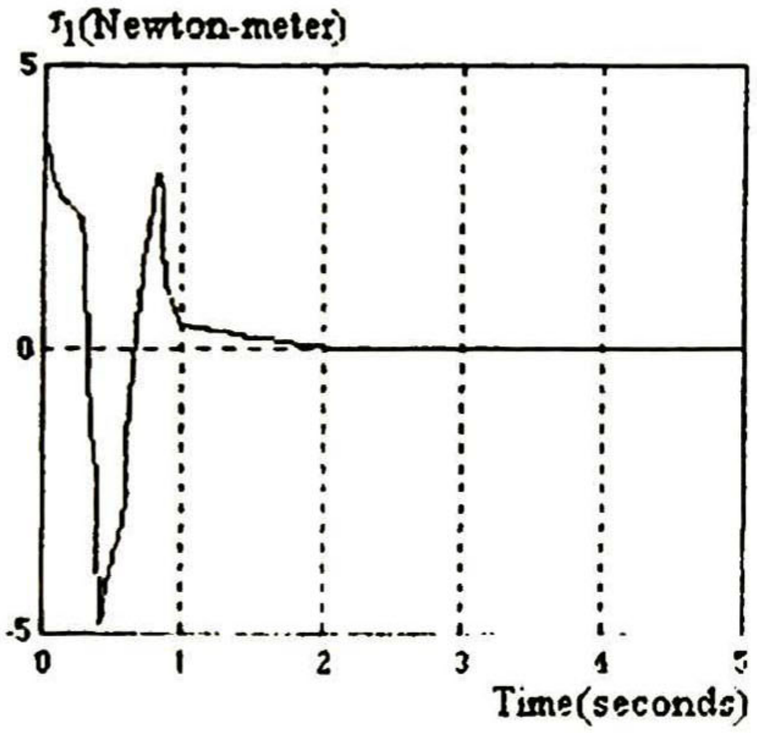


Figure 11. Control Signal

5. Conclusions

In this paper, we have introduced a new MIMO fuzzy PD controller, which includes a DSFS. This new scheme is able to implement a Pendubot swing-up control, with satisfactory performance.

The structure of the new controller is simple. It is composed of two MIMO fuzzy PD controllers, based on a reduced set of rules, and a switching function using one simple rule.

More work is in progress to implement real-time experiments. Future research will focus on trajectory tracking and stability analysis.

Acknowledgement

The first and second authors thank the support of CONACYT project 0652A9506. They also thank B. Castillo-Toledo and L. E. Ramos, CINVESTAV, Guadalajara, Mexico, for their helpful comments.

References

[1] J Hauser and R.M Murray, "Nonlinear controllers for nonintegrable systems. The acrobot example", in Proc 1990 American Control Conference, pp 669-671, June 1990

[2] M. W Spong, "The swing up control problem for the acrobot", *IEEE Control Systems*, pp 49-55, 1995

[3] L. E. Ramos, B. Castillo-Toledo, and J. Alvarez, "Nonlinear Regulation of an underactuated robot", in *Proc. 1997 IEEE Intl. Conference in Robotics and Automation*, pp 3288-3293, Albuquerque, New Mexico, USA, April 1997

[4] M. Berkemeier, *"Nonlinear control of a two-link hoping robot"*, Ph D. Thesis, University of California, Berkeley, USA, 1993.

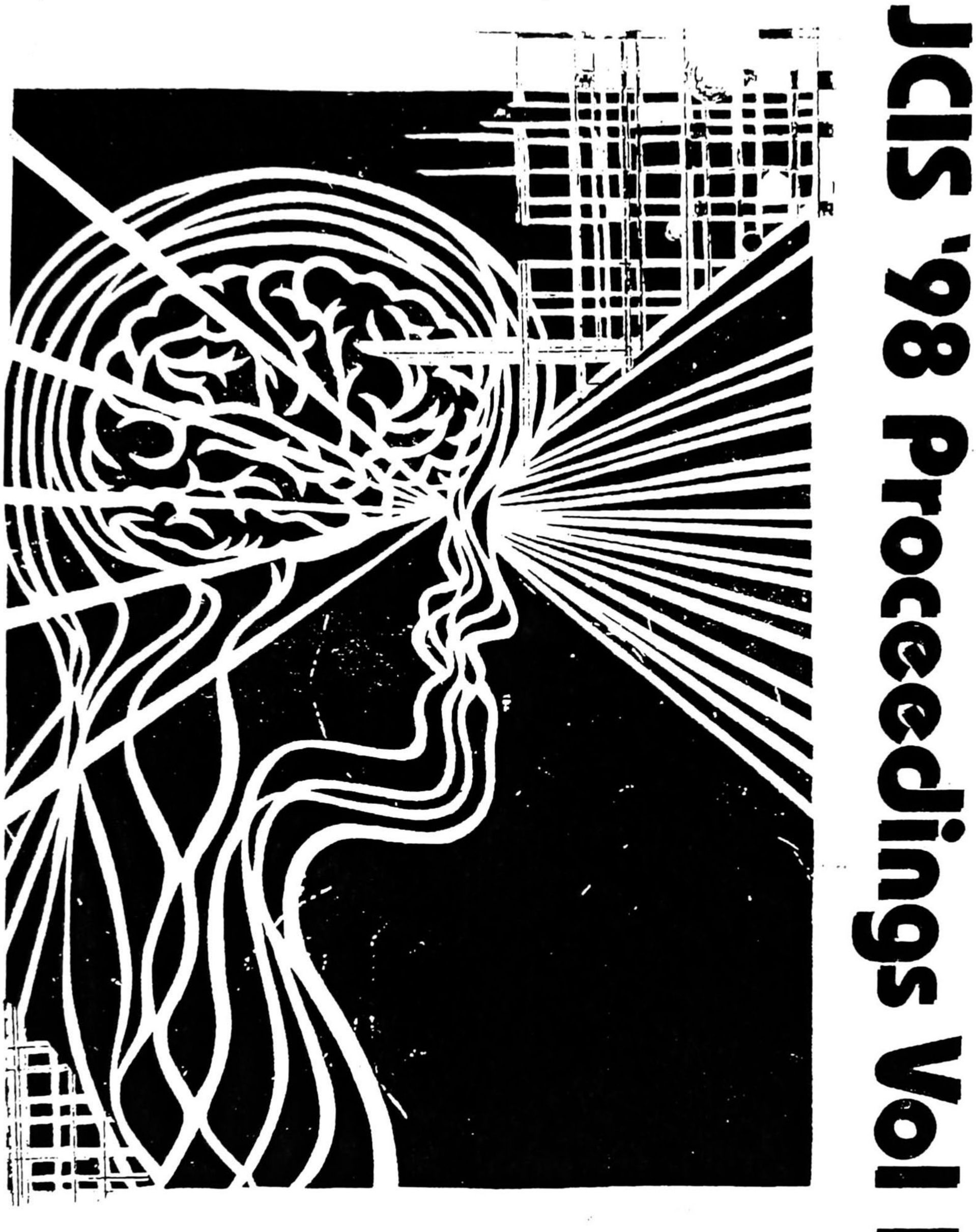
[5] M. A. Lee, and M. H. Smith, "Automatic design and tuning of a fuzzy system for controlling the acrobot using genetic algorithms, DSFS, and Meta-Rule Techniques", in *Proc. NAFIPS'94*, San Antonio, Texas, Dec. 1994.

[6] M H Smith, M A Lee, M. Uliemi, and W A Gruver, "Design limitations of PD versus fuzzy controllers for the acrobot", in *Proc. 1997 IEEE Intl. Conference in Robotics and Automation*, pp 1130-1135, Albuquerque, New Mexico, USA, April 1997.

[7]Y. Hsu, G Chen, and E N Sanchez, "A Fuzzy PD controller for multi-link robot control: Stability Analysis", *Proc. 1997 IEEE Intl. Conference in Robotics and Automation*, pp 1412-1417, Albuquerque, New Mexico, USA, April 1997.

[8] H A Malki, H. Li, and G. Chen, "New design and stability analysis of fuzzy proportional-derivative control systems," *IEEE Trans. on Fuzzy Systems.*, Vol. 2, pp 245-254, 1994.

APENDICES



October 23 - 28, 1998

Association for Intelligent Machinery

Sheraton Imperial Hotel & Convention Center, RTP, N.C., USA

Real Time Fuzzy Swing-up Control for an Underactuated Robot

Edgar Sanchez[1], Luis A. Nuño[1], Ya-Chen Hsu[2], and Guanrong Chen[2]

1. CINVESTAV, Unidad Guadalajara, Apartado Postal 31-438, Guadalajara, Jalisco C.P 44550, Mexico
e-amil:sanchez@gdl.cinvestav.mx
2. Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204, USA
e-mail:gchen@uh.edu

Abstract.
In this paper, a Multi-Input Multi-Output (MIMO) fuzzy Proportional-Derivative (PD) controller, including a Dynamic Switching Fuzzy System (DSFS), is applied to swing-up an underactuated robot: the Pendubot. This mechanism consists of a double pendulum actuated only at the first joint. Simulations have demonstrated the strong applicabilty of the proposed control scheme. The new controller and its real-time applications in the Pendubot constitue our original contributions.

1. Introduction

In recent years, there has been increasing interest in underactuated mechanisms; that is, mechanical systems having more joints than control actuators. Different control schemes, derived from nonlinear systems theory [1,2,3], clasical control techniques [4], or intelligent control methodologies [5,6], have been proposed.

In [5], a practical technique is introduced for automatic tuning, based on Genetic Algorithms (GA), for an underactuated fuzzy control system. A comparison beeween the PD control proposed in [4] and the fuzzy control discussed in [5] is presented in [6]. Perhaps except for the PD scheme of [4], most of the mentioned methods are not easy to implement on the Pendubot.

In this paper, we present an application of a new fuzzy PD control scheme to swing-up control of the Pendubot, an underactuated robot. We reconsider the controller introduced in [7], based on a very simple structure as that described in [8]. We also incorporate a DSFS, to change the membership functions of the fuzzy PD controller. The new control scheme and its real-time application constitue our originl contributions.

2. MIMO Fuzzy PD Control Scheme

For completeness, we first briefly review the basic structure and formulation of the MIMO fuzzy PD controller, originally presented in [7]. The structure of this MIMO fuzzy PD controller, which is a continuous-time controller, is based on the two-input one-output controller shown in Fig. 1, from which it has

$$u(t) = K_u u^*(t) \tag{1}$$

where K_u is a fuzzy control gain determined by the designer and $u^*(t)$ is the output of the fuzzy controller.

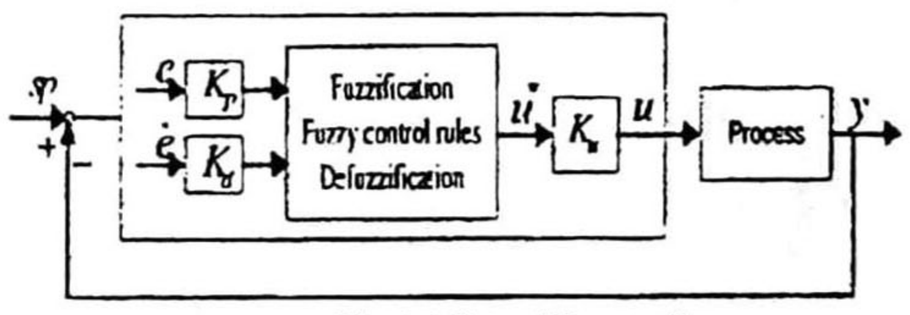


Fig. 1 A Fuzzy PD controller

(i) *Fuzzification*: The two inputs are: the error signal $e(t)$ and its rate of change $\dot{e}(t)$; the only one output, $u(t)$, is fed to the process. Membership functions are shown in Figs. 2 and 3, respectively, where the same membership function is applied for both $e(t)$ and $\dot{e}(t)$ for simplicity. To employ the same membership function on these two different inputs, two constant factors, K_p and K_d, are used to rescale $e(t)$ and $\dot{e}(t)$, respectively. L and H are two positive constants to be determined.

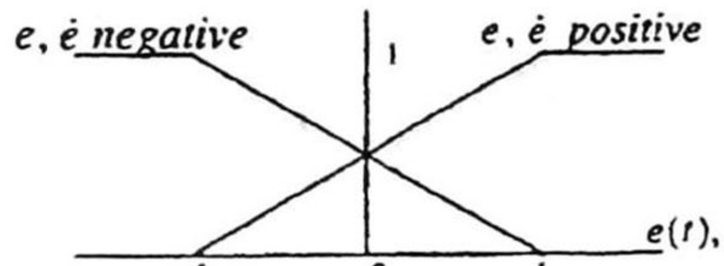


Fig. 2 The input membership function.

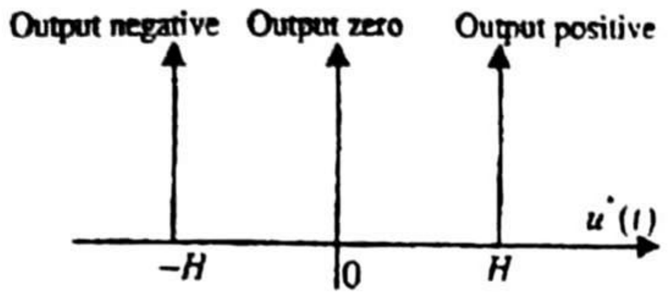


Fig. 3 The output membership function.

(ii) *Fuzzy control rules*:

(R1) IF e = ep AND $\dot{e}$ = rp THEN output = op
(R2) IF e = ep AND $\dot{e}$ = rn THEN output = oz
(R3) IF e = en AND $\dot{e}$ = rp THEN output = oz
(R4) IF e = en AND $\dot{e}$ = rn THEN output = on

In these rules, e and $\dot{e}$ are defined as above, output is the control action, u^*, "ep" means "error positive" and "op" means "output positive," etc., and "AND" is the Zadeh's logical "AND". For a detailed interpretation of these rules, see [7,8].

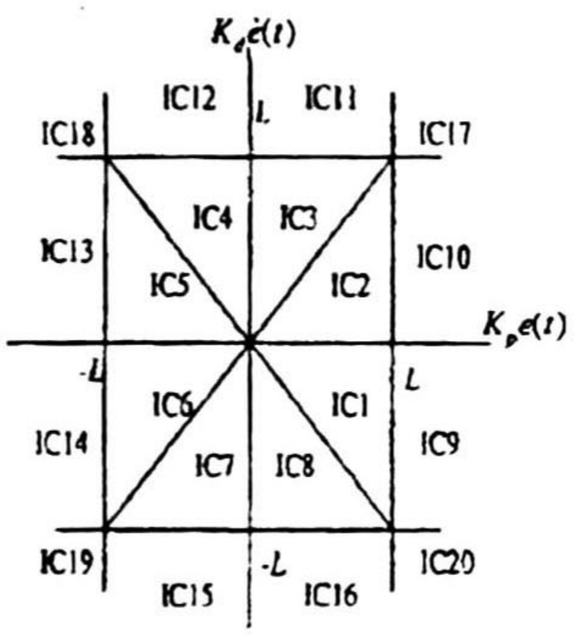


Fig. 4 The input IC regions.

(iii) *Defuzzification*: The commonly used "center of mass" formula is employed [8].
For this fuzzy PD controller, the inputs value-ranges are actually decomposed into twenty adjacent input-combination (IC) regions, as shown in Fig. 4. The control rules (R1-R4), membership functions, and IC regions, are used to evaluate an appropriate fuzzy control law for each region as follows [8]:

$$u_{PD}(t) = K_u u^*(t)$$

where

$$u^*(t) = \frac{L}{2(2L - K_p|e(t)|)}[K_p e(t) + K_d \dot{e}(t)]$$

in IC1, IC2, IC5, IC6.

$$= \frac{L}{2(2L - K_d|\dot{e}(t)|)}[K_p e(t) + K_d \dot{e}(t)]$$

in IC3, IC4, IC7, IC8.

$$= \frac{1}{2}[L + K_d \dot{e}(t)] \quad \text{in IC9, IC10.}$$

$$= \frac{1}{2}[L + K_p e(t)] \quad \text{in IC11, IC12.}$$

$$= \frac{1}{2}[-L + K_d \dot{e}(t)] \quad \text{in IC13, IC14.}$$

$$= \frac{1}{2}[-L + K_p e(t)] \quad \text{in IC15, IC16.}$$

$= L$ in IC17.
$= -L$ in IC19.
$= 0$ in IC18, IC20.

3. The Pendubot Control

The dynamic equation of a rigid n-link robot arm can be written as

$$D(\theta)\ddot{\theta} + C(\theta,\dot{\theta}) + G(\theta) = \tau \qquad (2)$$

where $\theta \in R^n$ denotes the joint position vector of the arm, $\tau \in R^n$ is the applied joint torque vector, $D(\theta) \in R^{n \times n}$ is the inertia matrix, $C(\theta,\dot{\theta})\dot{\theta} \in R^n$ represents the vector of Coriolis and centrifugal torques, and $G(\theta) \in R^n$ denotes the gravitational torque.
To control such a multivariable system, a set of fuzzy PD controllers are proposed in [7] as follows:

$$\tau_i = \sum_{j=i}^{n} u_{ij} \qquad i = 1,\ldots,n \qquad (3)$$

where each u_{ij}, a sub-fuzzy PD controller derived as above, employs inputs (e_j and $\dot{e}_j$) from the jth joint, and produces a torque output to the ith joint. Thus, for instance, τ_n consists of only one element u_{nn}, while τ_1 has n terms: $u_{11} + u_{12} + \ldots + u_{1n}$.

A scheme of this underactuted robot is shown in Fig. 5. It has two links, in which the first is actuated and the second is free.

In this figure, g is the gravity acceleration. For the ith (i=1,2) link, Ii is the momentum of inertia, Li is the length, Lci is the distance from the join to the mass center, mi is the mass of the link. X is the horizontal axis, Y is the vertical axis, $\theta 1$ is the angle between the horizontal axis and the first link, $\theta 2$, is the angle

between the two links, and $\tau 1$ is the first joint input torque.

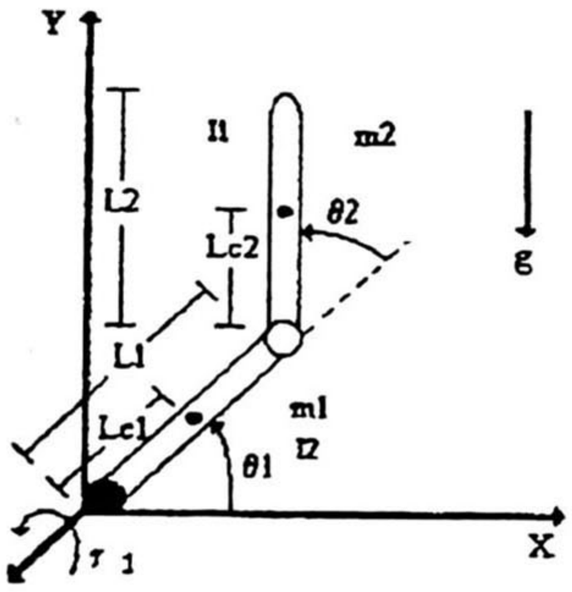


Fig. 5 The structure of the Pendubot .

If only m joints are actuated, the vector θ can be partitionated, without loss of generalty, as (θ_1, θ_2), where $\theta_1 \in \Re^m$ represents the actuated joints, and $\theta_2 \in \Re^{n-m}$ represents the unactuated ones. So, for the Pendubot, the dinamic model (2) is particularized as:

$$\begin{bmatrix} D_{11} & D_{12} \\ D_{12} & D_{22} \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ 0 \end{bmatrix} \qquad (4)$$

where:

$$D_{11} = m_1 l_{c1}^2 + m_2\left(l_1^2 + l_{c2}^2 + 2 l_1 l_{c2}\cos(\theta_2)\right) + I_1 + I_2,$$

$$D_{12} = m_2\left(l_{c2}^2 + l_1 + l_{c2}\cos(\theta_2)\right) + I_2,$$

$$D_{22} = m_2 l_{c2}^2 + I_2$$

$$C_1 = -2 m_2 l_1 l_{c2}\dot{\theta}_1\dot{\theta}_2 \sin(\theta_2) + m_2 l_1 l_{c2}\dot{\theta}_2^2 \sin(\theta_2)$$

$$C_2 = m_2 l_1 l_{c2}\dot{\theta}_1^2 \sin(\theta_2)$$

$$G_1 = m_1 g l_1 \cos(\theta_1) + m_2 g l_1 \cos(\theta_1) + m_2 g l_{c2}\cos(\theta_1 + \theta_2)$$

$$G_2 = m_2 g l_{c2}\cos(\theta_1 + \theta_2)$$

To control the Pendubot, we use the MIMO fuzzy PD controller discussed above. In this application, it is composed of two fuzzy PD controllers, one for each joint, whose outputs are added to calculate τ_1 (the control action to the actuated joint). A schematic diagram of this overall controller is shown in Fig. 6 .

It is well known that controlling an underactuated system is a challenging task [1,2,3,5]. So, in order to obtain a more powerful controller, based on [6], we incorporate a switching function in our design.

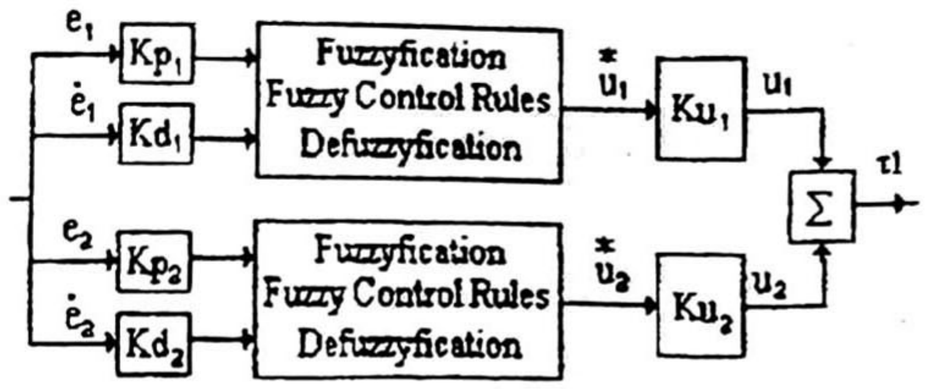


Fig. 6 The MIMO fuzzy PD controller

More specifically, we propose a new control scheme composing of two MIMO PD fuzzy Controller, shown in Fig. 5, with exactly the same structure, but a different constant parameter L for each one of them. The switching function selects which one to apply at each instant. This switching function is based on the distance d from the tip of the secnod link to the target position. So, this function is easily implemented by the following rule:

If d is longer than d_1 then apply Fuzzy Controller 1, else apply Fuzzy Controller 2.

The distance d can be easily deducted from Fig. 5, as:

$$d = \sqrt{\left((L_1 + L_2 - L_1\sin(\theta_1) - L_2\sin(\theta_1+\theta_2)\right)^2 + \left(L_1\cos(\theta_1) + L_2\cos(\theta_1+\theta_2)\right)^2}$$

4. A Real-Time Application

A real-time control was implemented in a *Pendubot* workbench. The Pendubot consists of two aluminum links of lengths 14 in (35.56 cm) and 8 in (20.32 cm), respectively. Link 1 is direcly coupled to the shaft of a 90 V permanent magnent DC motor mounted to the end of a table. The motor mount and bearings are the support of the entire system. Link 1 also includes the bearing housing for Joint 2. Needle roller bearings riding on a ground shaft were used to construct the revolute joint for Link 2. The shaft extends out in both directions of the housing allowing coupling to the second link and to an optical encoder mounted on Link 1. The design gives both links full 360^0 of rotational motion. Link 2 is constructed of a 0.25 in (0.625 cm) thick length of aluminum with a coupling that is attached to the shaft of Joint 2. All the control computations are performed on an IBM Pentium/75 PC workstation with a D/A card and an encoder interface card. Using the standard software library routines supplied with these cards, we are able to program the control algorithms directly in C.

The control task is to bring the Pendubot from a straight-down rest position to a balanced straight-up vertical position. Initnial and goal postions are (-180 deg, 0 deg) and (180 deg, 0 deg), repectively.

We selected $d_1 = 0.15$ m. In order to test the robustness of the algorithm to external perturbations, Link 2 was hit twice. We present the time evolution for θ_1 and θ_2 in Figs. 7 and 8, respectively. Figure 9 shows the control signal applied to the Pendubot; it never exceeds the maximum allowable torque of the DC motor (10 Newton-meter). As can be seen, the proposed control algorithm is able to swing-up this underactuacted robot, with almost no oscillations.

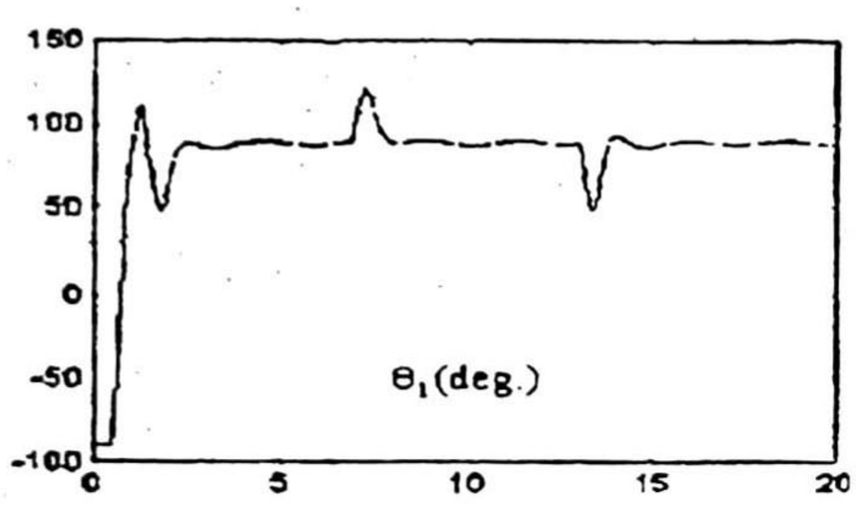


Fig. 7 The time evolution of θ_1

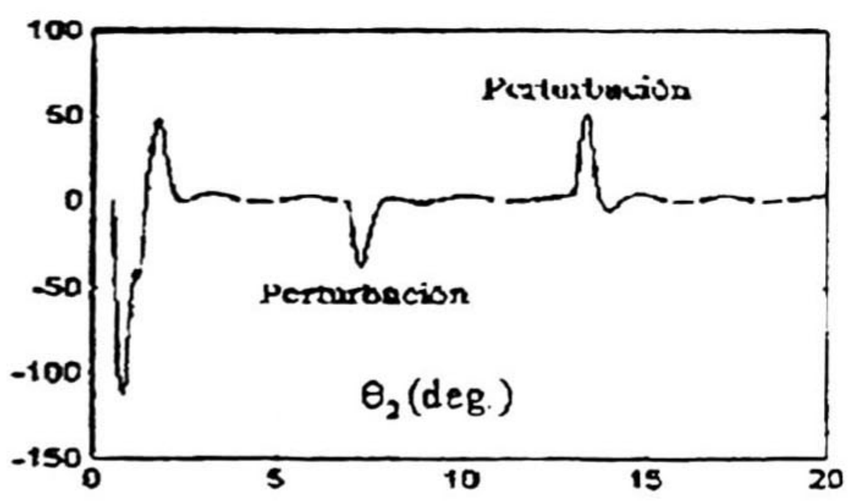


Fig. 8 Time evolution of θ_2

Fig. 9 The control signal.

5. Conclusions

In this paper, we have introduced a new MIMO fuzzy PD controller, which includes a DSFS. This new scheme is able to implement real-time swing-up control for a Pendubot, with a very good performance. The structure of the new controller is simple. Work is in progress for trajectory tracking and stability analysis.

Acknowledgement

The first and second authors thank the support of CONACYT project 0652A9506. They also thank the useful comments of B. Castillo-Toledo and L. E. Ramos, CINVESTAV, Guadalajara, Mexico.

References

[1] J. Hauser and R.M. Murray, "Nonlinear controllers for nonintegrable systems: The acrobot example", in *Proc. of American Control Conference*, pp 669-671, June 1990.

[2] M. W. Spong, "The swing up control problem for the acrobot", *IEEE Control Systems*, pp 49-55, 1995.

[3] L. E. Ramos, B. Castillo-Toledo, and J. Alvarez, "Nonlinear Regulation of an underactuated robot", in *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 3288-3293, Albuquerque, New Mexico, USA, 1997.

[4] M. Berkemeier, *"Nonlinear control of a two-link hoping robot"*, Ph. D. Thesis, University of California, Berbeley, USA, 1993.

[5] M. A. Lee, and M. H. Smith, "Automatic design and tuning of a fuzzy system for controlling the acrobot using genetic algorithms, DSFS, and Meta-Rule Techniques", *Proc. of NAFIPS'94*, San Anoinio, Texas, Dec. 1994.

[6] M. H. Smith, M. A. Lee, M. Ulieru, and W. A. Gruver, "Design limitations of PD versus fuzzy controllers for the acrobot", *Proc. of 1997 IEEE Int. Conf. on Robotics and Automation*, pp. 1130-1135, Albuquerque, New Mexico, USA, April 1997.

[7] Y. Hsu, G. Chen, and E. N. Sanchez, " A Fuzzy PD controller for multi-link robot control: Stability Anlysis", *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 1412-1417, Albuquerque, New Mexico, USA, April 1997.

[8] H. A. Malki, H. Li and G. Chen, "New design and stability analysis of fuzzy proportional-derivative control systems." *IEEE Trans. on Fuzzy Systems.*, Vol. 2, pp. 245-254, 1994.

[9] M. W. Spong and D. J. Block, "A mechatronic system for control research and education". *Proc. of IEEE Conf. on Decision and Control*, pp. 555-556, New Orleans, La., USA, Dec. 1994.

INDICE

CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL IPN
UNIDAD GUADALAJARA

El jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios avanzados del Instituto Politécnico Nacional, aprobó la tesis: **"Control PD Difuso Aplicado a Robots Manipuladores"** del C. **Luis Arturo Nuño Sánchez** el día 14 de Diciembre de 1998

Dr. Egdar Nelson Sánchez Camperos
Investigador CINVESTAV 3B
CINVESTAV DEL IPN
Guadalajara, Jal.

Dr. Bernardino Castillo Toledo
Investigador CINVESTAV 3A
CINVESTAV DEL IPN
Guadalajara, Jal.

Dra. Ofelia Begovich Mendoza
Investigador CINVESTAV 3A
CINVESTAV DEL IPN
Guadalajara, Jal.

Dr. Alexander Loukianov
Investigador CINVESTAV 3A
CINVESTAV DEL IPN
Guadalajara, Jal.

Dr. Fernando Lara Rojo
Profesor Titular e Investigador
ITESO
Guadalajara, Jal.