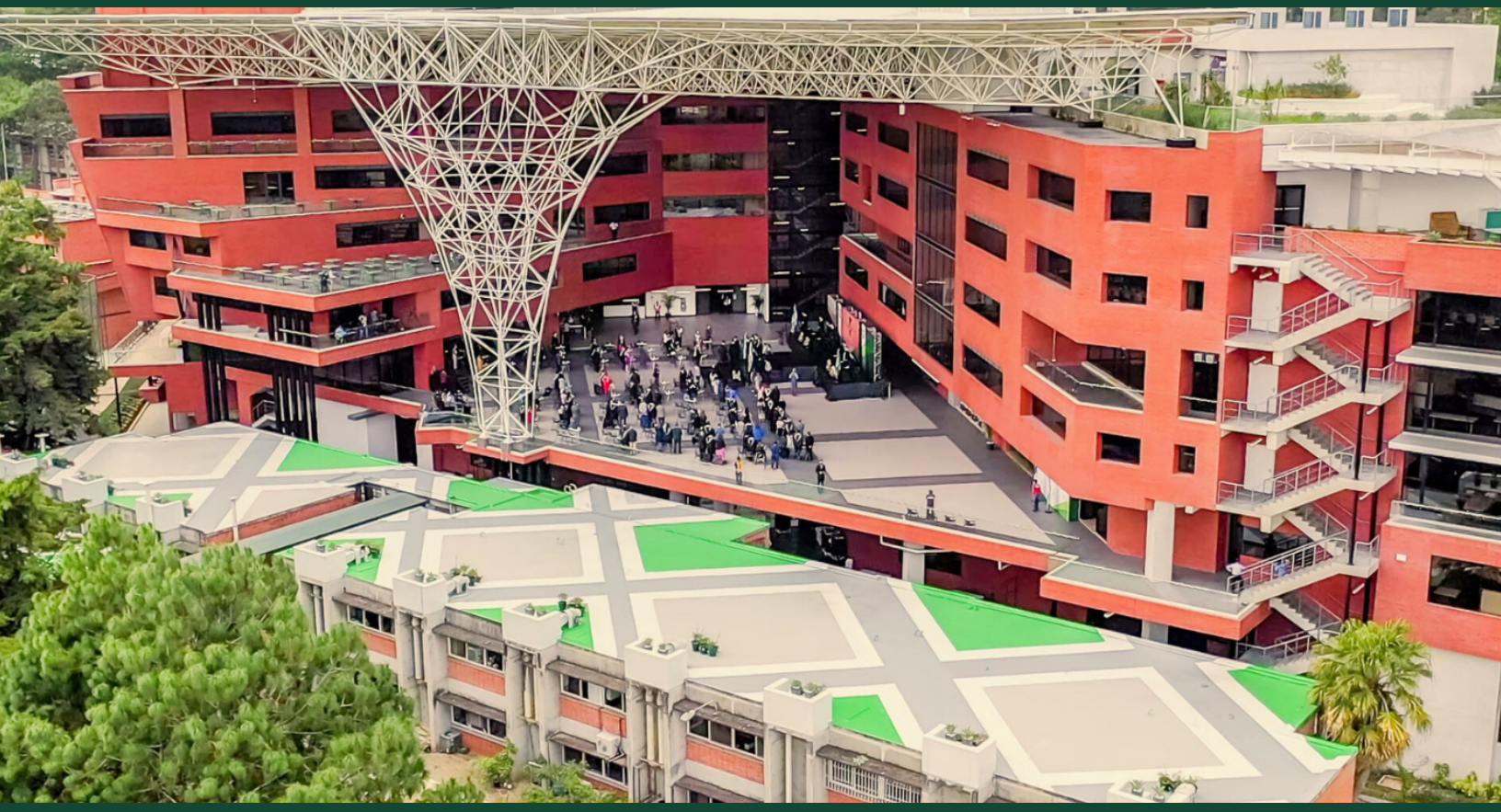
Evaluación e implementación de sistemas de lógica difusa para aplicaciones en Ingeniería Mecatrónica

Gustabo Adolfo Córdova Abril



UVG | UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Evaluación e implementación de sistemas de lógica difusa para aplicaciones en Ingeniería Mecatrónica

Trabajo de graduación presentado por Gustabo Adolfo Córdova Abril
para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

2025

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

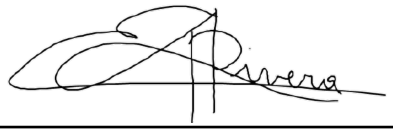
Evaluación e implementación de sistemas de lógica difusa para aplicaciones en Ingeniería Mecatrónica

Trabajo de graduación presentado por Gustabo Adolfo Córdova Abril
para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

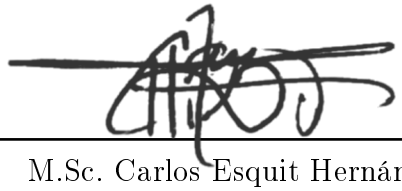
2025

Vo.Bo.:



(f) _____

Dr. Luis Alberto Rivera Estrada



(f) _____

M.Sc. Carlos Esquit Hernández

Fecha de aprobación: Guatemala, 20 de noviembre de 2025.

Índice general

Índice de figuras

Índice de cuadros

Resumen

La ingeniería mecatrónica requiere sistemas de decisión y control capaces de manejar dinámicas complejas, condiciones variables y comportamientos no lineales propios de los actuadores y sensores utilizados en la práctica. Dentro de este contexto, la lógica difusa se presenta como una alternativa explicable basada en razonamiento lingüístico y reglas no lineales, capaz de ofrecer flexibilidad y claridad interpretativa en situaciones donde los controladores clásicos suelen requerir ajustes más estrictos. Este estudio analiza el desempeño del sistema difuso Mamdani como esquema de decisión y posteriormente como controlador, comparándolo contra los métodos PID y LQR, ampliamente utilizados en aplicaciones de control moderno.

Para ello se desarrollaron simuladores interactivos en MATLAB aplicados a dos casos representativos: un péndulo invertido y un brazo robótico 3D. Ambos escenarios permiten evaluar la estabilidad, la suavidad de la respuesta y la capacidad de los sistemas de control para manejar errores significativos, saturaciones y efectos no lineales. La plataforma implementada facilita observar el comportamiento dinámico en tiempo real y realizar ajustes a los parámetros de control, lo que permitió una comparación justa y sistemática entre los tres enfoques.

Los resultados muestran que todos los controladores pueden estabilizar los sistemas bajo una configuración adecuada, aunque con diferencias notables en tiempos de convergencia. En particular, el controlador difuso Mamdani exhibe una respuesta más consistente frente a variaciones y condiciones no lineales, ofreciendo una alternativa explicable y robusta frente a métodos clásicos. Estos hallazgos refuerzan el potencial de la lógica difusa para integrarse en sistemas mecatrónicos reales, especialmente cuando se requiere interpretabilidad junto con un desempeño competitivo.

Abstract

The field of mechatronics requires decision-making and control systems capable of handling complex dynamics, variable conditions, and the nonlinear behaviors inherent to real-world actuators and sensors. Within this context, fuzzy logic emerges as an explainable alternative based on linguistic reasoning and nonlinear rules, able to provide flexibility and interpretability in situations where classical controllers typically require stricter tuning. This study analyzes the performance of the Mamdani fuzzy system first as a decision-making scheme and later as a controller, comparing it against PID and LQR methods, which are widely used in modern control applications.

To achieve this, interactive MATLAB simulators were developed and applied to two representative cases: an inverted pendulum and a 3D robotic arm. Both scenarios allow for the evaluation of stability, response smoothness, and the ability of the control systems to handle significant errors, saturations, and nonlinear effects. The implemented platform makes it possible to observe the dynamic behavior in real time and adjust control parameters, enabling a fair and systematic comparison among the three approaches.

The results show that all controllers can stabilize the systems under an adequate configuration, although with notable differences in convergence times. In particular, the Mamdani fuzzy controller exhibits a more consistent response under variations and nonlinear conditions, offering an explainable and robust alternative to classical methods. These findings reinforce the potential of fuzzy logic for integration into real mechatronic systems, especially when interpretability is required alongside competitive performance.

CAPÍTULO 1

Introducción

La automatización moderna requiere controladores capaces de mantener un buen desempeño aun con no linealidades, saturaciones y perturbaciones. En este campo, la ingeniería de control y la Ingeniería Mecatrónica emplean con frecuencia tres enfoques: el control PID, el regulador lineal cuadrático (LQR) y la lógica difusa tipo Mamdani (*fuzzy logic*). Cada uno ofrece ventajas distintas en simplicidad, robustez y esfuerzo de implementación.

El estudio se centra en dos sistemas representativos por su valor didáctico y uso práctico: un péndulo (estabilización alrededor de un equilibrio inestable), y un brazo robótico 3D controlado en el espacio cartesiano mediante cinemática inversa por tasas, en particular *Damped Least Squares* (DLS). Estos sistemas reúnen fenómenos como no linealidad, acoplamientos y límites físicos, adecuados para una comparación justa entre métodos.

El objetivo principal consistió en analizar el comportamiento de la lógica difusa como esquema de decisión y control en sistemas no lineales. Para contextualizar sus capacidades, se emplearon los enfoques PID y LQR como referencias comparativas sobre dos sistemas representativos: un péndulo invertido y un brazo robótico 3D. La evaluación se basó en medidas habituales: tiempo de establecimiento, sobreimpulso, error acumulado (IAE/ISE) y esfuerzo de control. Además, se desarrollaron simuladores interactivos reproducibles que facilitan el análisis y favorecen su futura transferencia a prototipos.

El trabajo se realizó en entorno de simulación (MATLAB) con modelos que capturan la dinámica dominante de cada sistema. No se abordó la optimización automática de parámetros ni el diseño robusto avanzado; el análisis se limitó a configuraciones representativas y comparables. En el péndulo y el motor se consideraron saturaciones y mecanismos de anti-*windup*; en el brazo se empleó DLS como capa de cinemática inversa y el control actuó sobre errores cartesianos sujetos a límites articulares.

La metodología incluyó modelado, diseño de cada controlador, definición de escenarios de prueba realistas y medición con las métricas indicadas. Se utilizaron interfaces gráficas para visualizar estados, errores y acciones de control en tiempo real, y para registrar resultados de forma consistente. En términos generales, los hallazgos preliminares indican que LQR

rinde bien cuando el modelo linealizado representa adecuadamente la planta; PID ofrece un desempeño competitivo con sintonía cuidadosa y protección ante saturaciones; y Mamdani resulta ventajoso en regímenes no lineales al incorporar conocimiento heurístico, a costa de un diseño explícito de funciones de pertenencia y reglas.

CAPÍTULO 2

Antecedentes

Según Guzmán y Castaño [1], la lógica difusa ha demostrado ser una herramienta efectiva en diversas áreas de la ingeniería, como la industria, la medicina y la electrónica. Su principal ventaja radica en su capacidad para modelar procesos que no pueden resolverse fácilmente con lógica binaria o ecuaciones matemáticas rígidas. En su estudio, destacan la aplicación de los sistemas de inferencia difusa de Mamdani y Takagi-Sugeno-Kang (TSK), los cuales han sido utilizados en el diseño y simulación de controles adaptativos que permiten una mayor flexibilidad en sistemas dinámicos.

Por otro lado, Duarte [2] explora aplicaciones específicas de la lógica difusa, como el desarrollo de controladores adaptativos, bases de datos inteligentes y algoritmos de reconocimiento de imágenes. Su investigación evidencia que la lógica difusa no solo facilita la toma de decisiones en entornos con alta incertidumbre, sino que también permite una mayor integración de elementos cognitivos y humanos en sistemas automatizados. Ambos autores coinciden en que esta metodología representa un avance significativo en la optimización y control de procesos complejos dentro de la ingeniería.

En el campo de la ingeniería mecatrónica, la lógica difusa ha cobrado especial relevancia debido a su capacidad para mejorar la interacción entre sistemas mecánicos y electrónicos en entornos dinámicos. Según Mohebbi, Achiche y Baron [3], esta técnica ha sido utilizada en el diseño multicriterio de sistemas mecatrónicos, permitiendo la toma de decisiones en presencia de múltiples objetivos de diseño interrelacionados. Su estudio evidencia que la lógica difusa no solo mejora la eficiencia de estos sistemas, sino que también optimiza la integración de distintos subsistemas dentro de un mismo entorno de trabajo.

La lógica difusa tiene diversas aplicaciones en ingeniería, destacándose por su capacidad para manejar incertidumbre, adaptarse a cambios, optimizar procesos y, en algunos casos, operar en tiempo real. En el control de sistemas dinámicos, como robots y vehículos autónomos, permite gestionar la incertidumbre y adaptarse a variaciones en el entorno con respuestas inmediatas. En redes eléctricas inteligentes, facilita la optimización del consumo energético y la adaptación a fluctuaciones, aunque su implementación en tiempo real puede ser limitada. En la optimización de procesos industriales, cumple con todas estas caracte-

rísticas, mejorando significativamente la eficiencia operativa. También se ha aplicado en el procesamiento de datos e inteligencia artificial, ayudando a gestionar información incierta y optimizar el reconocimiento de patrones, aunque su nivel de adaptabilidad varía según el contexto. Finalmente, en el ámbito financiero, particularmente en la negociación algorítmica en mercados volátiles, la lógica difusa permite desarrollar estrategias más flexibles y robustas frente a la incertidumbre, aun cuando su desempeño en tiempo real depende del sistema implementado [1].

Un antecedente cercano que inspira esta investigación es el trabajo de graduación realizado por José Pablo Petion Rivas [4], quien evaluó e implementó algoritmos genéticos aplicados a sistemas mecatrónicos. Su estudio demostró cómo los métodos de inteligencia computacional pueden mejorar la eficiencia y autonomía de sistemas dinámicos en ingeniería, especialmente en contextos donde las condiciones son inciertas o cambiantes. Aunque su enfoque principal se centró en algoritmos evolutivos, su estructura metodológica y su aplicación práctica evidencian el valor de explorar técnicas no convencionales en problemas reales. En esta misma línea, diversas investigaciones han mostrado que la lógica difusa también permite mejorar la eficiencia y adaptabilidad de sistemas mecatrónicos, optimizando procesos y gestionando condiciones dinámicas e inciertas [5] [6]. Así, dichas investigaciones sirven como punto de referencia para trabajos que, como la presente investigación, buscan ampliar el uso de técnicas avanzadas de inteligencia computacional, enfocándose en la lógica difusa como una alternativa prometedora para el desarrollo de controladores inteligentes y aplicaciones dentro de la ingeniería mecatrónica.

De forma complementaria, trabajos recientes relacionados con el control de péndulos invertidos han demostrado la eficacia de los sistemas difusos en escenarios reales y simulados. Correa-Ramírez, Giraldo-Buitrago y Escobar-Mejía [6] desarrollaron un controlador difuso tipo Takagi–Sugeno para un péndulo invertido con rueda de reacción, logrando un seguimiento de trayectoria estable mediante reguladores locales y técnicas de linealización por no linealidad sectorial. Sus resultados muestran que los controladores difusos pueden mantener la estabilidad aun frente a perturbaciones, siempre que se diseñen adecuadamente las funciones de membresía y las reglas.

De manera comparable, Díaz-Salgado y Pichardo-Cruz [7] realizaron un estudio experimental en el que compararon un controlador moderno basado en retroalimentación de estados con un controlador difuso Mamdani aplicado a un péndulo invertido rotacional. Encontraron que el enfoque difuso es capaz de reducir el sobreimpulso y suavizar la respuesta temporal en condiciones de operación reales, con un comportamiento robusto frente a las limitaciones del actuador y el ruido propio de los sensores embebidos.

Por otro lado, Ramírez-Suárez, Pérez-Segura y Barrón-Zambrano [8] desarrollaron un robot tipo péndulo invertido controlado mediante lógica difusa utilizando una plataforma Arduino y un módulo inercial. Su trabajo destaca la viabilidad de implementar sistemas difusos en hardware de bajo costo, logrando mantener el equilibrio del péndulo mediante un conjunto reducido de reglas y un procesamiento computacional mínimo. Este tipo de prototipos evidencia que la lógica difusa no solo es útil en simulaciones, sino que también puede integrarse de manera efectiva en sistemas físicos con recursos limitados.

CAPÍTULO 3

Justificación

En la ingeniería moderna, la toma de decisiones y el control de sistemas dependen en gran medida de modelos matemáticos exactos y lógica booleana. Sin embargo, muchos procesos del mundo real presentan incertidumbre, imprecisión y variaciones dinámicas que los métodos tradicionales no logran abordar de manera eficiente. Problemas como el control de sistemas dinámicos, la optimización de procesos industriales y el procesamiento de datos requieren enfoques más flexibles y adaptativos.

La lógica difusa se presenta como una alternativa viable para enfrentar estos desafíos, permitiendo representar estados intermedios y manejar la incertidumbre de manera más realista. Su aplicación en áreas como el control de sistemas, redes eléctricas inteligentes, optimización de procesos y análisis de datos financieros ha demostrado mejoras significativas en precisión y adaptabilidad.

Este estudio es relevante porque contribuye a la exploración y evaluación de la lógica difusa como herramienta para mejorar la toma de decisiones en ingeniería. Comparar su desempeño con métodos tradicionales permite identificar sus ventajas y limitaciones, brindando información valiosa para su futura implementación en sistemas de control, automatización e inteligencia artificial.

En ese contexto, este trabajo de graduación busca dar continuidad a ese tipo de propuestas innovadoras mediante la incorporación de lógica difusa como herramienta de control inteligente. La elección de esta teoría no solo responde a su versatilidad comprobada, sino también al potencial de comparación con otros enfoques basados en inteligencia computacional, como los algoritmos genéticos. Así, este trabajo amplía el panorama de soluciones inteligentes aplicables a la Ingeniería Mecatrónica, fortaleciendo la base académica y experimental para el desarrollo de sistemas más precisos y robustos.

CAPÍTULO 4

Objetivos

4.1. Objetivo general

Evaluar e implementar sistemas de lógica difusa para aplicaciones en Ingeniería Mecatrónica.

4.2. Objetivos específicos

- Investigar y evaluar algoritmos y sistemas de lógica difusa y sus aplicaciones.
- Implementar sistemas de lógica difusa y validarlos en escenarios abstractos simples.
- Desarrollar escenarios prácticos para distintas aplicaciones en Ingeniería Mecatrónica en los que puedan usarse los sistemas de lógica difusa.
- Aplicar y evaluar el rendimiento de los sistemas de lógica difusa en las aplicaciones y los escenarios desarrollados.

CAPÍTULO 5

Definición del problema

El control de sistemas dinámicos no lineales plantea desafíos importantes en Ingeniería Mecatrónica, especialmente debido a la presencia de incertidumbres, acoplamientos y saturaciones en los actuadores. En este contexto, los métodos clásicos basados en modelos linealizados, como PID y LQR, constituyen referencias útiles, pero pueden presentar limitaciones cuando el sistema opera fuera de los supuestos para los cuales fueron diseñados. Estas condiciones motivan la exploración de enfoques alternativos que no dependan de un modelo exacto y que mantengan un desempeño estable ante variaciones significativas.

Ante estas limitaciones, surge la necesidad de explorar métodos más flexibles que mantengan la estabilidad y precisión sin requerir un modelo exacto. La lógica difusa ofrece una alternativa viable al incorporar conocimiento heurístico y reglas lingüísticas que permiten una respuesta más adaptable ante condiciones variables.

El problema abordado en este trabajo consiste en estudiar la aplicabilidad y el comportamiento de un sistema basado en lógica difusa tipo Mamdani en dos sistemas representativos de la Ingeniería Mecatrónica. Para contextualizar sus capacidades, se emplean los métodos PID y LQR únicamente como referencias comparativas. Los casos de estudio considerados en entorno de simulación son:

- **Sistema de decisión difuso para regular la velocidad de un ventilador:** implementado como un esquema de inferencia Mamdani que ajusta la velocidad de un ventilador en función de la temperatura y la humedad, ilustrando el uso de lógica difusa en tareas de decisión no estrictamente asociadas al control de estados.
- **Péndulo invertido:** utilizado para analizar la estabilización de un equilibrio inestable.
- **Brazo robótico de tres grados de libertad:** enfocado en el control de posición cartesiana mediante cinemática inversa por tasas.

El estudio se desarrolla íntegramente en simulación y se limita al análisis del desempeño temporal, la estabilidad y el esfuerzo de control bajo condiciones de prueba equivalentes.

Con ello se busca identificar en qué circunstancias la lógica difusa puede ofrecer un comportamiento competitivo o ventajoso frente a enfoques clásicos.

CAPÍTULO 6

Marco teórico

6.1. Lógica difusa y diferencia con la lógica booleana

La lógica booleana representa proposiciones con valores binarios $\{0, 1\}$; los conectores (AND, OR, NOT) se implementan mediante tablas de verdad. Zadeh introdujo la lógica difusa en 1965 [9], [10], donde la verdad es gradual: una proposición puede tomar cualquier valor dentro del intervalo $[0, 1]$. Este enfoque permite razonar con información imprecisa y constituye la base conceptual de los controladores difusos de Mamdani [11], [12].

La lógica difusa (*fuzzy logic*) constituye una extensión de la lógica clásica que incorpora grados de pertenencia en lugar de valores estrictamente binarios. En su trabajo seminal sobre los *fuzzy sets*, Lotfi A. Zadeh [10] formalizó el concepto de conjuntos difusos como herramienta matemática para manejar incertidumbre y vaguedad inherentes a los sistemas reales. A diferencia de la lógica booleana tradicional, que clasifica los elementos como pertenecientes o no a un conjunto, la lógica difusa asigna un grado de pertenencia continuo entre 0 y 1. Este enfoque permite modelar conceptos lingüísticos como "temperatura alta" o "velocidad baja", los cuales en la práctica no poseen límites definidos y se describen mejor mediante transiciones graduales [13]. Así, la lógica difusa proporciona un marco más flexible para el razonamiento aproximado, sentando las bases de múltiples aplicaciones en control inteligente e ingeniería moderna [14].

6.1.1. Operadores lógicos generales

En lógica difusa se usan t-normas y s-normas. La elección clásica (empleada por Mamdani) es:

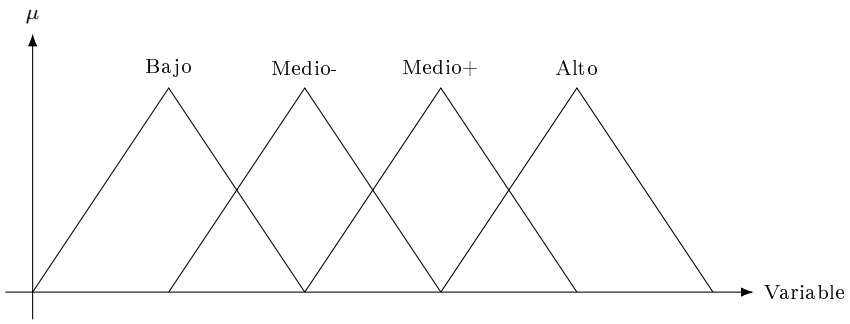
$$\text{AND}(a, b) = \text{mín}(a, b), \qquad \text{OR}(a, b) = \text{máx}(a, b), \qquad \text{NOT}(a) = 1 - a,$$

con $a, b \in [0, 1]$ [12]. Estos operadores reducen a la lógica booleana cuando $a, b \in \{0, 1\}$.

6.2. Conjuntos difusos

De acuerdo con Zadeh [9], un conjunto difuso A definido sobre un universo U se describe mediante una función de pertenencia $\mu_A : U \to [0, 1]$, que asigna a cada elemento un grado de pertenencia intermedio entre inclusión total y exclusión total. En el contexto de control difuso, estas funciones permiten representar gradualidad en conceptos normalmente binarios, como "frío" o "caliente", posibilitando transiciones suaves entre estados. Klir y Yuan destacan que la forma de las funciones —triangular, trapezoidal o gaussiana— determina la suavidad y sensibilidad de la respuesta del sistema.

Figura 1. Funciones de pertenencia triangulares aplicadas a una variable genérica



Nota. Las formas de pertenencia definen la transición gradual entre categorías. La triangular es una elección simple y habitual por su bajo costo computacional y comportamiento predecible. Elaboración propia.

6.3. Implementación del controlador difuso Mamdani

El control difuso modela conocimiento heurístico en términos lingüísticos y no exige un modelo exacto de la planta. La formulación de Mamdani y Assilian [11] produce salidas difusas que luego se convierten a valores precisos mediante defuzzificación. Una presentación de referencia sobre conceptos y diseño puede encontrarse en Ross [12], mientras que Engelbrecht sitúa estos sistemas dentro del marco más amplio de la inteligencia computacional junto con redes neuronales y algoritmos evolutivos [14].

6.4. Motor de inferencia Mamdani

El sistema de inferencia difusa utilizado corresponde al método de Mamdani, propuesto por Mamdani y Assilian en 1975 [11], basado en la teoría de conjuntos difusos introducida por Zadeh en 1965 [9]. Este método es ampliamente empleado en aplicaciones de control por su carácter intuitivo y su capacidad de traducir reglas lingüísticas en acciones concretas [12], [15].

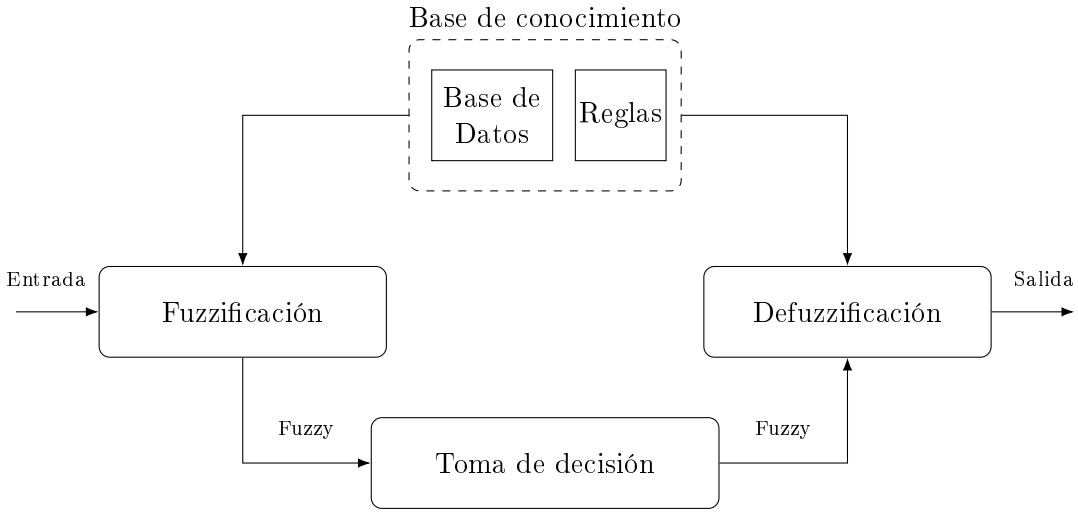
6.4.1. Modelo Mamdani: proceso completo de inferencia

En términos operativos, el flujo canónico (*canonical flow*) de un sistema Mamdani describe la secuencia estándar de procesamiento que sigue el sistema de inferencia difusa desde que recibe las entradas hasta que genera la salida. Este flujo consta de las siguientes etapas [11], [16]:

1. **Evaluación de antecedentes** (*antecedent evaluation*): se calculan los grados de verdad de las condiciones de cada regla, utilizando los operadores lógicos difusos ($AND \rightarrow$ mín, $OR \rightarrow$ máx).
2. **Implicación** (*implication*): el grado de activación de cada regla se emplea para recortar o escalar la función de pertenencia del consecuente correspondiente.
3. **Agregación** (*aggregation*): se combinan (mediante el operador máximo) todas las salidas difusas provenientes de las reglas activas, obteniendo una única función de pertenencia de salida.
4. **Defuzzificación** (*defuzzification*): la salida difusa agregada se convierte en un valor numérico *crisp*, que representa la respuesta final del sistema de control.

Cada una de estas etapas se aplica de forma secuencial dentro del sistema de inferencia Mamdani, garantizando la conversión sistemática de información precisa en decisiones aproximadas basadas en conocimiento experto. En la Figura 2 se aprecia de manera gráfica el proceso completo de inferencia.

Figura 2. Arquitectura Mamdani: fuzzificación, base de conocimiento, decisión y defuzzificación



Nota. Adaptado de [17]. El diagrama ilustra la arquitectura clásica del sistema de inferencia difusa Mamdani, que incluye las etapas de fuzzificación de las entradas, consulta a la base de conocimiento (compuesta por la base de datos y las reglas), toma de decisión sobre los conjuntos difusos generados y el proceso final de defuzzificación para obtener una salida (*crisp*).

6.4.2. Operadores lógicos en Mamdani

La evaluación de reglas en Mamdani se basa en operadores difusos estándar [12]:

$$\mu_{A\wedge B}(x) = \min\big(\mu_A(x), \mu_B(x)\big), \tag{1}$$

$$\mu_{A\vee B}(x) = \max\big(\mu_A(x), \mu_B(x)\big), \tag{2}$$

$$\mu_{\neg A}(x) = 1 - \mu_A(x) \tag{3}$$

La implicación y agregación se realizan como:

$$\mu_{C_r}^{\text{imp}}(y) = \min\big(\alpha_r,\, \mu_{C_r}(y)\big), \qquad \mu_Y(y) = \max_r\ \mu_{C_r}^{\text{imp}}(y), \tag{4}$$

donde $\alpha_r = \min(\mu_A(x^\star), \mu_B(z^\star))$ es el grado de activación de la regla r.

6.5. Fuzzificación

La *fuzzificación* constituye la primera etapa del sistema de inferencia difusa y consiste en transformar valores de entrada binarios (*crisp*) en grados de pertenencia a conjuntos difusos definidos en el universo de discurso. Un valor *crisp* es un número exacto o determinístico que representa una magnitud física (por ejemplo, temperatura, velocidad o posición), mientras que un valor difuso expresa la pertenencia parcial de esa magnitud a un conjunto lingüístico, en el rango $[0, 1]$. [15], [16]

Matemáticamente, el proceso se representa como una función de mapeo:

$$\mu_A : X \to [0, 1], \tag{5}$$

donde X es el conjunto de valores posibles de la variable de entrada, y $\mu_A(x)$ devuelve el grado de pertenencia del valor x al conjunto difuso A. De esta manera, cada entrada del sistema se convierte en un vector de grados de pertenencia que describe su relación con las categorías difusas asociadas.

El procedimiento se realiza evaluando las funciones de pertenencia predefinidas para cada variable. Estas pueden adoptar diferentes formas, como triangular, trapezoidal o gaussiana, dependiendo del nivel de suavidad y solapamiento deseado entre categorías [18]. En general, las funciones triangulares son comunes por su simplicidad y bajo costo computacional:

$$\mu_A(x) = \begin{cases} 0, & x \le a, \\ \dfrac{x-a}{b-a}, & a < x \le b, \\ \dfrac{c-x}{c-b}, & b < x < c, \\ 0, & x \ge c, \end{cases} \tag{6}$$

donde a, b y c definen los puntos de inicio, vértice y fin del conjunto difuso.

La salida de la etapa de fuzzificación consiste en los grados de pertenencia $\mu_A(x)$ para todas las etiquetas definidas de cada variable, que serán posteriormente utilizados por el motor de inferencia para la evaluación de reglas.

6.6. Defuzzificación

La defuzzificación convierte la salida difusa agregada en un valor numérico *crisp*. Entre los métodos más empleados se encuentran el centroide *(center of mass, COG)* y el bisector del área *(bisector of area, BOA)* [19], [20]. Ambos buscan transformar la información lingüística y difusa en una magnitud interpretable por el sistema físico, manteniendo coherencia con la distribución de pertenencias obtenida en la inferencia.

6.6.1. Centroide (COG)

El método del centroide determina el punto de equilibrio del área bajo la función de pertenencia de salida $\mu_{\text{sal}}(z)$. Este punto representa el valor que equilibra la masa total del conjunto difuso, análogo al cálculo del centro de masa de un cuerpo físico continuo [12]. Su formulación se expresa como:

$$z^\star = \frac{\int z\,\mu_{\text{sal}}(z)\,dz}{\int \mu_{\text{sal}}(z)\,dz}. \tag{7}$$

donde $z^\star$ es el valor numérico resultante (salida *crisp*); z es la variable de salida continua; y $\mu_{\text{sal}}(z)$ representa el grado de pertenencia agregado obtenido tras la etapa de inferencia. Este método pondera cada punto z según su grado de pertenencia, garantizando que los valores más representativos (aquellos con mayor $\mu_{\text{sal}}(z)$) influyan más en el resultado final. El COG ofrece una salida continua y estable incluso ante pequeñas variaciones de las reglas activadas, motivo por el cual es el método más utilizado en controladores difusos tipo Mamdani [15].

6.6.2. Bisector (BOA)

El método del bisector busca el punto $z^\star$ que divide el área total bajo la función de pertenencia agregada en dos partes iguales [15], [21]:

$$\int_{z_{\text{mín}}}^{z^\star} \mu_{\text{sal}}(z)\,dz = \int_{z^\star}^{z_{\text{máx}}} \mu_{\text{sal}}(z)\,dz. \tag{8}$$

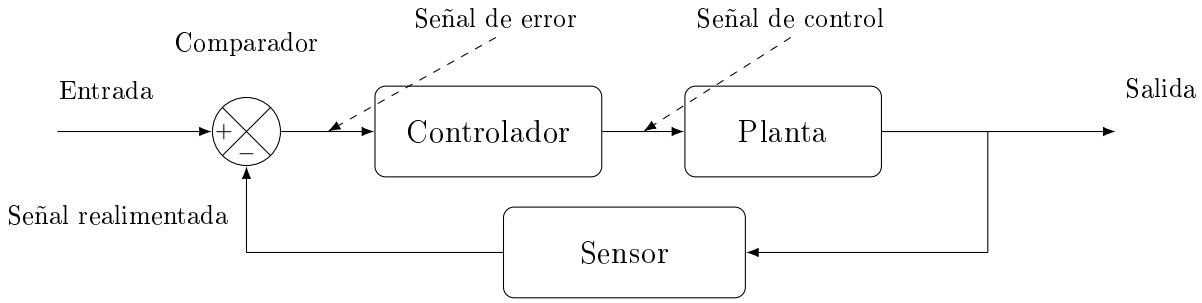
donde $z_{\text{mín}}$ y $z_{\text{máx}}$ delimitan el rango de la variable de salida, $\mu_{\text{sal}}(z)$ es la función de pertenencia resultante de la agregación de reglas, y $z^\star$ es el punto de equilibrio del área total. A diferencia del centroide, el BOA no pondera la magnitud del grado de pertenencia, sino el área acumulada, lo que lo hace menos sensible a valores extremos y más adecuado en distribuciones asimétricas o no convexas. El valor obtenido representa la posición que equilibra el área total, garantizando una división equitativa del dominio de salida. Según Zadeh [21], este enfoque refleja uno de los principios fundamentales de la lógica difusa: representar cuantitativamente la ambigüedad mediante funciones de pertenencia que permitan una interpretación gradual en lugar de binaria.

6.7. Conceptos básicos de control

Según Åström y Murray [22], un sistema de control es un arreglo de componentes que regula el comportamiento de una planta o proceso para mantener su salida cercana a un objetivo. Se distinguen sistemas de lazo abierto (sin medición de la salida) y de lazo cerrado (con retroalimentación). En lazo cerrado, la salida $y(t)$ se compara con la referencia $r(t)$ para formar el error $e(t) = r(t) - y(t)$; el controlador calcula la acción $u(t)$ a partir de $e(t)$ y la aplica a la planta. En esta línea, Ogata [23] subraya que la retroalimentación negativa mejora la precisión y permite el rechazo de perturbaciones, aunque exige un diseño cuidadoso para preservar la estabilidad del lazo.

En este trabajo se considera un sistema de control *SISO* (*Single Input, Single Output*), es decir, un sistema con una única variable de entrada y una única variable de salida. Este tipo de configuración permite analizar el comportamiento dinámico de forma clara, facilitando la comparación entre distintas estrategias de control. La Figura 3 ilustra el esquema general de lazo cerrado con comparador, controlador, planta y realimentación.

Figura 3. Lazo cerrado SISO con comparador, controlador, planta y realimentación



Nota. El diagrama ilustra el flujo de señales en un lazo cerrado: el comparador genera la señal de error, el controlador actúa sobre la planta y una porción de la salida se retroalimenta para formar nuevamente el error. Adaptado de [24].

De acuerdo con Dorf & Bishop [25], el modelado se aborda típicamente mediante funciones de transferencia o espacio de estados. Para sistemas lineales invariantes en el tiempo (LTI), el modelo de estado adopta la forma:

$$\dot{\mathbf{x}} = A\,\mathbf{x} + B\,\mathbf{u}, \qquad \mathbf{y} = C\,\mathbf{x} + D\,\mathbf{u}. \tag{9}$$

donde $\mathbf{x}$ es el vector de estados, $\mathbf{u}$ la entrada (o vector de entradas), $\mathbf{y}$ la salida (o vector de salidas), y A, B, C, D son las matrices del sistema LTI que relacionan estados, entradas y salidas; el punto ($\dot{}$) denota derivada respecto del tiempo t. La implementación digital introduce muestreo con periodo T_s y retención de orden cero *(ZOH)*; además, sensores y actuadores incorporan ruido, retardos y saturaciones que condicionan el diseño [25].

6.8. Control PID

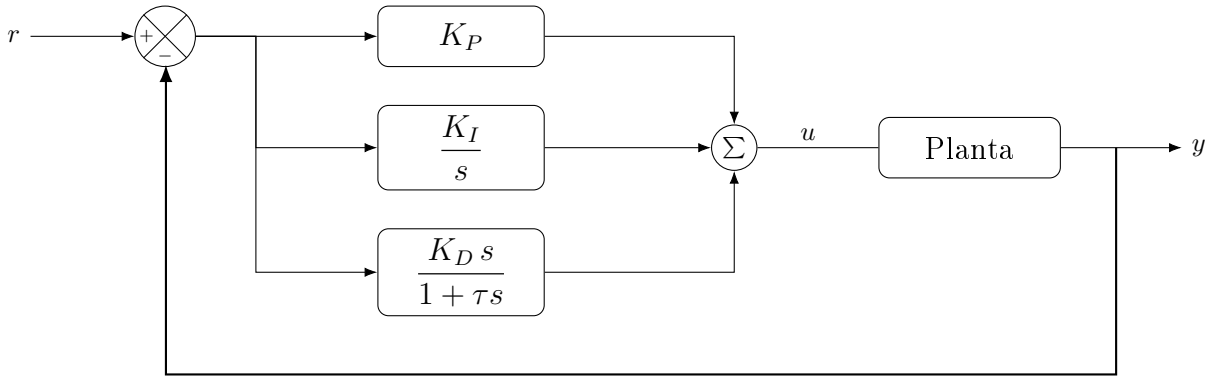
Como es bien conocido desde Ziegler y Nichols [26], el controlador proporcional–integral–derivativo (PID) es el esquema industrial más difundido por su simplicidad, interpretabilidad y desempeño razonable. En tiempo continuo, su ley de control se escribe como

$$u(t) = K_P\, e(t) + K_I \int_0^t e(\tau)\, d\tau + K_D\, \dot{e}(t). \tag{10}$$

donde $u(t)$ es la señal de control aplicada al actuador, $e(t) = r(t) - y(t)$ es el error entre referencia y salida, K_P, K_I, K_D son las ganancias proporcional, integral y derivativa, respectivamente, y $\dot{e}(t)$ representa la derivada temporal del error; la integral acumula el error desde 0 hasta el tiempo actual t.

Las acciones proporcional, integral y derivativa reducen el error instantáneo, eliminan sesgos estacionarios y amortiguan la dinámica, respectivamente. En términos prácticos, Ogata [23] recomienda filtrar la derivada para atenuar ruido y emplear *anti-windup* cuando el actuador satura. Métodos clásicos de sintonía (p. ej., Ziegler–Nichols) aportan parámetros iniciales a refinar con métricas de desempeño [23], [26].

Figura 4. Control PID en paralelo: ramas P/I/D con derivada filtrada y suma de acciones



Nota. El diagrama muestra la estructura del controlador PID: la rama proporcional, la rama integral y la rama derivativa filtrada con constante τ se suman para generar la señal de control u, la cual actúa sobre la planta. El lazo se cierra mediante la realimentación de la salida para formar el error. Adaptado de [27].

Por otro lado, Dorf & Bishop [25] advierten que el PID es especialmente adecuado en procesos SISO de bajo orden o levemente no lineales; sus límites aparecen con acoplamientos multivariables, retardos significativos o no linealidad marcada, escenarios donde se requieren extensiones o alternativas.

6.9. Regulador lineal cuadrático (LQR)

Desde los aportes fundacionales de Kalman [28], el regulador lineal cuadrático (LQR) se reconoce como un método de control óptimo para sistemas lineales en espacio de estados. El objetivo es hallar una ley $\mathbf{u} = -K\mathbf{x}$ que minimice

$$J = \int_0^\infty \left(\mathbf{x}^\top Q\,\mathbf{x} + \mathbf{u}^\top R\,\mathbf{u}\right) dt, \tag{11}$$

donde J es el funcional de costo a minimizar, $\mathbf{x}$ y $\mathbf{u}$ son el estado y la entrada, $Q \succeq 0$ y $R \succ 0$ son matrices de ponderación (penalizan estados y esfuerzo de control, respectivamente) y t es la variable temporal de integración. La solución surge de la ecuación de Riccati algebraica

$$A^\top P + PA - PBR^{-1}B^\top P + Q = 0, \tag{12}$$

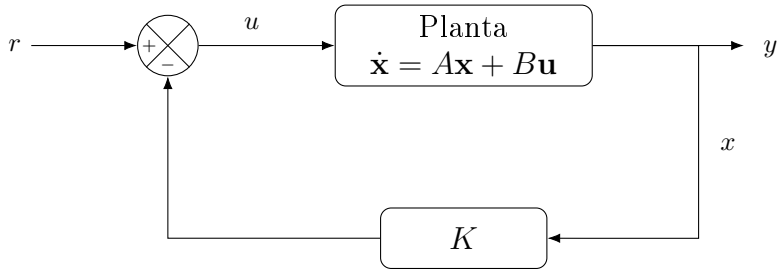
y la ganancia óptima

$$K = R^{-1}B^\top P, \tag{13}$$

donde K es la matriz de realimentación de estados que define la ley $\mathbf{u} = -K\mathbf{x}$.

Esto garantiza estabilidad en lazo cerrado si (A, B) es controlable [23], [29]. Åström y Murray [22] destacan que LQR ofrece un compromiso sistemático entre penalización de estados y esfuerzo de control; su alcance práctico depende de la validez del modelo lineal/linearizado y de la disponibilidad de estados u observadores [25].

Figura 5. Estructura de control LQR con realimentación de estados



Nota. El diagrama ilustra un esquema típico de regulador lineal cuadrático (LQR): la planta se modela en espacio de estados y la salida y se compara con la referencia r para formar el error. Adaptado de [30].

6.10. Principales métricas de desempeño en control

Conforme a Ogata [23] y a la exposición clásica de Dorf & Bishop [25], las métricas temporales y de exactitud más utilizadas incluyen el tiempo de subida (t_r), tiempo de asentamiento (t_s), sobreimpulso (M_p), tiempo pico (t_p) y el error en estado estacionario (e_∞). Asimismo, los índices integrales del error permiten valorar de forma compacta el compromiso rapidez–oscilación–precisión [22] [13]:

$$\mathrm{IAE} = \int_0^T |e(t)|\, dt. \tag{14}$$

donde IAE *(Integral of Absolute Error)* es el índice de error absoluto acumulado; $e(t)$ es el error en el tiempo y T es el horizonte de integración.

$$\mathrm{ISE} = \int_0^T e^2(t)\, dt. \tag{15}$$

donde ISE *(Integral of Squared Error)* penaliza cuadráticamente el error; $e(t)$ es el error y T el horizonte de integración.

$$\mathrm{ITAE} = \int_0^T t\, |e(t)|\, dt. \tag{16}$$

donde ITAE *(Integral of Time-weighted Absolute Error)* pondera el error por el tiempo t, haciendo más costosos los errores persistentes; $e(t)$ es el error y T el horizonte.

6.10.1. Métricas temporales

Tiempo de subida (t_r): tiempo necesario para que la respuesta transite desde un porcentaje bajo hasta uno alto de su valor final (por ejemplo, del 10 % al 90 %). Un t_r reducido implica mayor rapidez, aunque puede asociarse a oscilaciones indeseadas.

Tiempo pico (t_p): instante en que la señal alcanza su primer máximo absoluto. Se relaciona con el sobreimpulso.

Sobreimpulso (M_p): porcentaje máximo en que la salida excede el valor final esperado:

$$M_p = \frac{y_{\mathrm{máx}} - y_\infty}{y_\infty} \times 100\,\%. \tag{17}$$

donde $y_{\mathrm{máx}}$ es el valor pico de la respuesta y y_∞ el valor final (en régimen permanente).

Tiempo de asentamiento (t_s): tiempo requerido para que la salida permanezca dentro de una banda de tolerancia (usualmente $\pm 2\,\%$ o $\pm 5\,\%$) alrededor de su valor final. Para sistemas de segundo orden, una aproximación es

$$t_s \approx -\frac{\ln(\text{tolerancia})}{\zeta\, \omega_n}, \tag{18}$$

donde ζ es el factor de amortiguamiento, ω_n la frecuencia natural del sistema y tolerancia es el ancho de la banda (p. ej., 0.02 para 2 %) [13].

Error en estado estacionario (e_∞): diferencia entre la referencia y la salida una vez alcanzado el régimen permanente; la acción integral ayuda a reducirlo.

6.10.2. Índices integrales del error

- IAE: penaliza de manera uniforme la magnitud del error a lo largo del tiempo.
- ISE: penaliza más fuertemente los errores grandes, promoviendo rapidez aunque con posibles oscilaciones.

- ITAE: pondera el error por el tiempo, castigando aquellos que persisten y fomentando respuestas con asentamiento más rápido [29] [13].

6.10.3. Aplicación

La selección de métricas depende del sistema y de las prioridades de diseño: rapidez frente a estabilidad, tolerancia a sobreimpulsos o precisión estacionaria. En comparación de controladores, los índices integrales son útiles como medida objetiva, mientras que en aplicaciones reales se complementan con criterios de robustez frente a perturbaciones [12] [13].

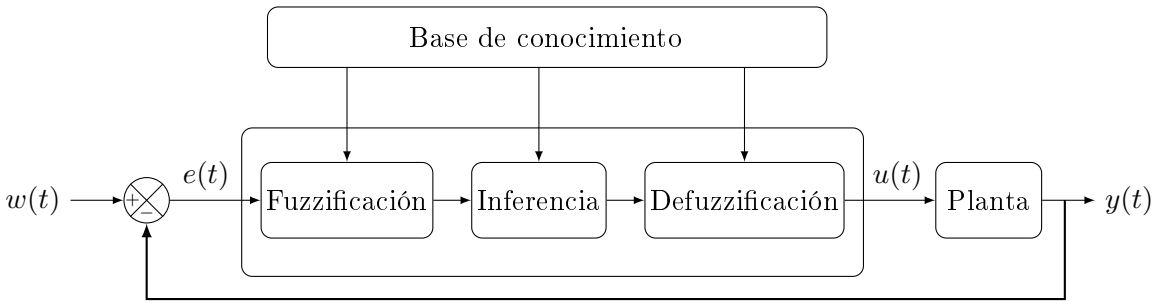
6.11. Comparación de controladores PID, LQR y Mamdani

Los enfoques PID, LQR y Mamdani difieren en supuestos, complejidad e interpretabilidad, aunque todos buscan un compromiso entre desempeño y esfuerzo de control. A nivel conceptual [22] [29] [12] [25] [14]:

- **Modelo requerido:** PID no necesita un modelo exacto; LQR exige un modelo lineal o linearizado con selección de Q, R; Mamdani se fundamenta en reglas lingüísticas y conocimiento experto.
- **Naturaleza y análisis:** PID opera en dominio lineal con estabilidad según sintonía; LQR garantiza estabilidad bajo suposiciones del modelo y optimiza un costo cuadrático; Mamdani es no lineal y su validez depende de la base de reglas.
- **Complejidad de diseño:** PID se ajusta con K_P, K_I, K_D; LQR resuelve la ecuación de Riccati y selecciona Q, R; Mamdani requiere definir funciones de pertenencia y reglas, con esfuerzo inicial mayor.
- **Ámbito de aplicación:** PID es versátil en sistemas SISO; LQR destaca en sistemas multivariables bien modelados; Mamdani ofrece flexibilidad ante no linealidades o incertidumbre.

En términos prácticos, el PID sobresale por su simplicidad y amplia adopción industrial, ofreciendo resultados satisfactorios con sintonía empírica. El LQR, propio del control moderno en espacio de estados, produce una ley de realimentación óptima cuando se dispone de modelo confiable. La lógica difusa Mamdani introduce un enfoque heurístico e interpretable, particularmente útil frente a fricciones y comportamientos no lineales moderados.

Figura 6. Lazo cerrado con controlador difuso Mamdani



Nota. El diagrama muestra el lazo cerrado de un controlador difuso tipo Mamdani, donde $w(t)$ representa la referencia, $e(t)$ el error aplicado al controlador, $u(t)$ la señal de control generada y $y(t)$ la salida realimentada del sistema. Adaptado de [31].

Cuadro 1. Pros y contras de PID, LQR y Mamdani

Criterio	**PID**	**LQR**	**Mamdani**
Modelo requerido	Empírico; no precisa modelo exacto.	Modelo lineal/linearizado; selección de Q, R.	Modelo lingüístico; conocimiento experto.
Diseño/sintonía	K_P, K_I, K_D; reglas clásicas (p. ej., Z–N).	Riccati; elección iterativa de Q, R.	Definir funciones de pertenencia y reglas; esfuerzo inicial mayor.
Análisis/garantías	Lineal; estabilidad según sintonía.	Óptimo y estable (sup. del modelo).	No lineal; garantías caso a caso.
No linealidad/robustez	Robustez moderada; puede degradar fuera de diseño.	Válido cerca del equilibrio; sensible a incertidumbre del modelo.	Flexible ante no linealidad; robusto dentro del rango cubierto por reglas.
Implementación	Muy simple y barata.	Cálculo en línea simple tras obtener K.	Costo medio; escala con el número de reglas.
Interpretabilidad	Media (ganancias).	Baja/media (matrices).	Alta (reglas lingüísticas).

Nota. La tabla resume criterios de comparación entre controladores clásicos y modernos considerando modelo, sintonía, robustez, implementación e interpretabilidad. Adaptado de [12], [15], [22], [23], [25], [29].

CAPÍTULO 7

Control difuso y enfoques clásicos: fundamentos aplicados y herramientas de software

El propósito central de este trabajo es investigar el comportamiento de sistemas de control basados en lógica difusa y compararlos con estrategias clásicas de control, específicamente el PID y el regulador lineal cuadrático (LQR). Con ello se busca evaluar las fortalezas y limitaciones de cada enfoque al enfrentar dinámicas complejas, incertidumbre y perturbaciones externas. El alcance del estudio abarca el diseño de controladores difusos tipo Mamdani, su implementación en modelos de sistemas representativos y la comparación directa con las alternativas tradicionales bajo métricas comunes de desempeño como error, tiempo de establecimiento y robustez [13].

7.1. Herramienteas de software y recursos empleados

Las simulaciones se realizaron en MATLAB R2023b, un entorno ampliamente utilizado en ingeniería de control. Para el diseño de los controladores difusos se empleó el *Fuzzy Logic Toolbox*, el cual permite definir funciones de pertenencia, reglas lingüísticas y procesos de desfuzzificación. Adicionalmente, se utilizó el *Control System Toolbox*, indispensable para la implementación de controladores clásicos como PID y LQR, así como para el análisis de estabilidad y desempeño [16]. Se desarrollaron también interfaces gráficas específicas para facilitar la interacción con los simuladores y visualizar en tiempo real la respuesta de los sistemas.

7.2. Modelo difuso estático de inferencia Mamdani

Antes de abordar los sistemas dinámicos considerados en este trabajo, se incluye un sistema estático de inferencia difusa. Este modelo, basado en un ventilador, tiene como propósito ilustrar de manera sencilla el flujo completo de un sistema de inferencia difusa Mamdani: la definición de variables lingüísticas, las funciones de pertenencia, la base de reglas y el proceso de desfuzzificación. A diferencia de los modelos dinámicos, este sistema no representa una planta física ni incorpora ecuaciones diferenciales; su objetivo es exclusivamente mostrar el razonamiento lógico del controlador ante variaciones en las entradas.

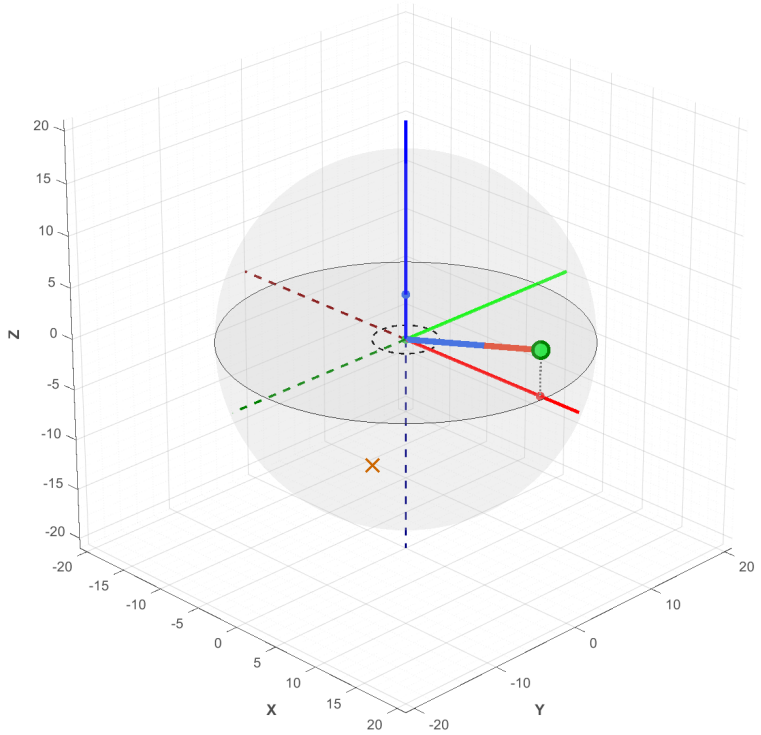
El modelo permite observar cómo las variables de entrada, como la temperatura o la humedad, se transforman en salidas continuas mediante las etapas de fuzzificación, inferencia y defuzzificación. De esta forma, se comprende de manera intuitiva el funcionamiento interno de la lógica difusa sin la complejidad de un lazo de control retroalimentado. Este esquema conceptual sirve como punto de partida para las simulaciones posteriores con sistemas dinámicos.

7.3. Modelos dinámicos bajo análisis

Con el objetivo de contar con escenarios diversos y representativos, se seleccionaron dos modelos de plantas ampliamente utilizados en docencia e investigación:

- **Brazo robótico (3 GDL):** sistema multivariable y no lineal, que permite analizar la influencia de la lógica difusa en el control de trayectorias y la manipulación bajo restricciones geométricas. Este caso ilustra los retos asociados a la *cinemática inversa* y al acoplamiento entre articulaciones [32].

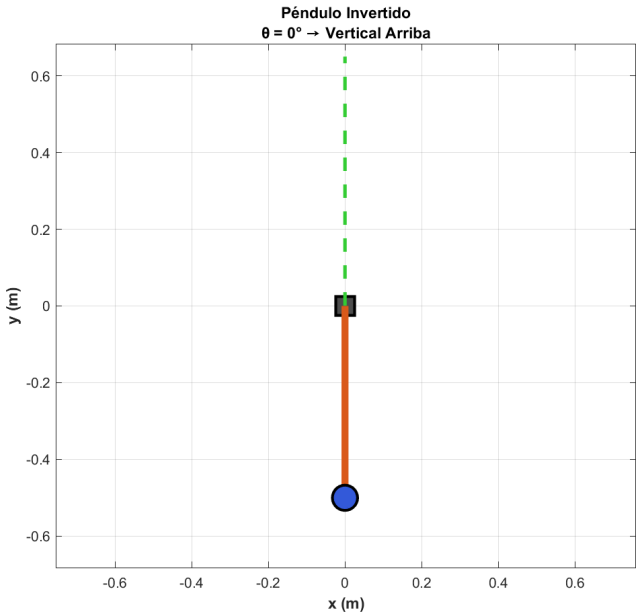
Figura 7. Simulación del brazo robótico 3DOF con visualización del espacio de trabajo



Nota. Se muestra la representación tridimensional del brazo robótico de tres grados de libertad (3DOF) durante la simulación. La esfera translúcida indica el espacio de trabajo accesible, mientras que las líneas de color representan los ejes x, y y z. Elaboración propia.

- **Péndulo simple:** péndulo con par de control u aplicado en el pivote y fricción viscosa b. El ángulo θ se mide desde la vertical ($\theta = 0$ en posición vertical superior por defecto, o bien como desee el usuario). Se utiliza para comparar la estabilización local con PID, LQR (linealizado en $\theta = 0$) y un controlador difuso sobre $(\theta, \dot{\theta})$.

Figura 8. Simulación del péndulo invertido en posición por defecto hacia abajo



Nota. El esquema representa el péndulo invertido en su posición por defecto hacia abajo, sin que haya algún torque o movimiento incolucrado. Elaboración propia.

CAPÍTULO 8

Implementación práctica de un sistema de inferencia difusa Mamdani

8.1. Introducción al sistema

El desarrollo de un sistema de control difuso aplicado a la regulación de un ventilador constituye un ejemplo representativo de cómo la lógica difusa puede emplearse para resolver problemas de control en los que intervienen múltiples variables con alto grado de incertidumbre. A diferencia de los controladores tradicionales, la lógica difusa se fundamenta en reglas lingüísticas inspiradas en la experiencia humana, lo que facilita su aplicación en sistemas complejos y no lineales [10] [33] [5].

En este caso, el ventilador se concibe como un actuador cuya velocidad debe ajustarse en función de dos parámetros de entrada: la temperatura ambiental y el nivel de humedad relativa. Estos factores presentan variaciones continuas, graduales y, en muchos casos, imprecisas, lo que hace inadecuado un control rígido basado únicamente en umbrales fijos. El empleo de un sistema de inferencia Mamdani permite representar conceptos difusos como "frío", "templado", "caliente" o "húmedo" mediante funciones de pertenencia, y asociarlos con niveles de salida interpretables como "velocidad lenta", "media" o "rápida".

El sistema fue diseñado en MATLAB, utilizando el *Fuzzy Logic Toolbox* y elementos de interfaz gráfica para simular en tiempo real el comportamiento del sistema de inferencia. Entre las principales características implementadas se incluyen:

- **Entradas:** temperatura (0–100 °C) y humedad (0–100 %).
- **Salida:** velocidad del ventilador (0–10, en escala arbitraria).
- **Funciones de membresía (FM):** triangulares, por su simplicidad y eficiencia computacional.

- **Base de reglas:** nueve reglas que relacionan condiciones ambientales con la respuesta del ventilador.

La interfaz gráfica permite visualizar la activación de reglas, observar el comportamiento de las funciones de membresía y monitorear la salida en un medidor semicircular dinámico. Además, se integró una representación tridimensional de la superficie de inferencia, la cual facilita interpretar cómo interactúan las variables de entrada para determinar la salida final.

Este enfoque ilustra las ventajas de la lógica difusa en el control de sistemas mecatrónicos:

- No depende de un modelo matemático rígido.
- Se adapta a condiciones cambiantes con facilidad.
- Resulta intuitivo y explicable desde la perspectiva humana, al basarse en reglas lingüísticas.

De esta manera, el sistema de ventilador no solo ejemplifica un caso práctico de control difuso, sino que también sirve como base metodológica para comparar este tipo de estrategias con los enfoques clásicos (PID, LQR) en aplicaciones más complejas de la Ingeniería Mecatrónica.

8.2. Diseño del sistema difuso

La elección de estas variables responde a criterios prácticos y de representación del problema. La temperatura y la humedad son factores ambientales que influyen directamente en la percepción de confort térmico y en la necesidad de ventilación. Por otro lado, la velocidad del ventilador constituye la acción de control más adecuada, pues permite modular la respuesta del sistema de forma continua, ajustándose a condiciones cambiantes del entorno. De esta manera, se establece una relación intuitiva entre las entradas y la salida que facilita la construcción de las reglas difusas y su interpretación.

8.3. Funciones de membresía

Para representar las variables lingüísticas del sistema se definieron funciones de membresía triangulares, dado que permiten una transición suave entre estados. Esta elección resulta adecuada para un caso de estudio didáctico, ya que facilita la interpretación visual de los conjuntos difusos y reduce la complejidad del diseño.

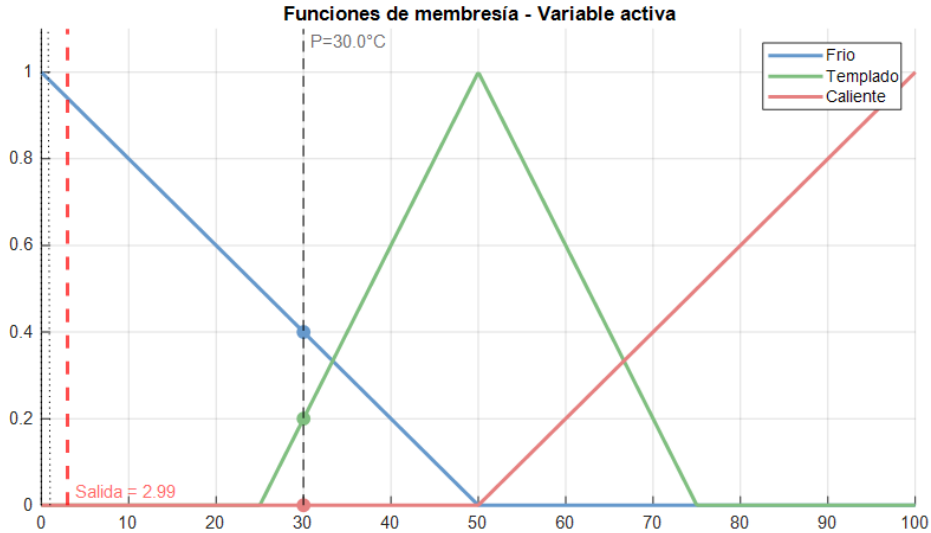
Temperatura

Se consideraron tres funciones de membresía que cubren el rango de 0 a 100 °C, las cuales se muestran en la Figura 9. Estas funciones definen los conjuntos difusos asociados a

la variable “Temperatura”, que permiten representar los estados lingüísticos “frío”, “templado” y “caliente”.

- **Frío**: activa principalmente en valores bajos (0–50).
- **Templado**: centrada en valores intermedios (25–75).
- **Caliente**: activa en valores altos (50–100).

Figura 9. Funciones de membresía de la variable “Temperatura”



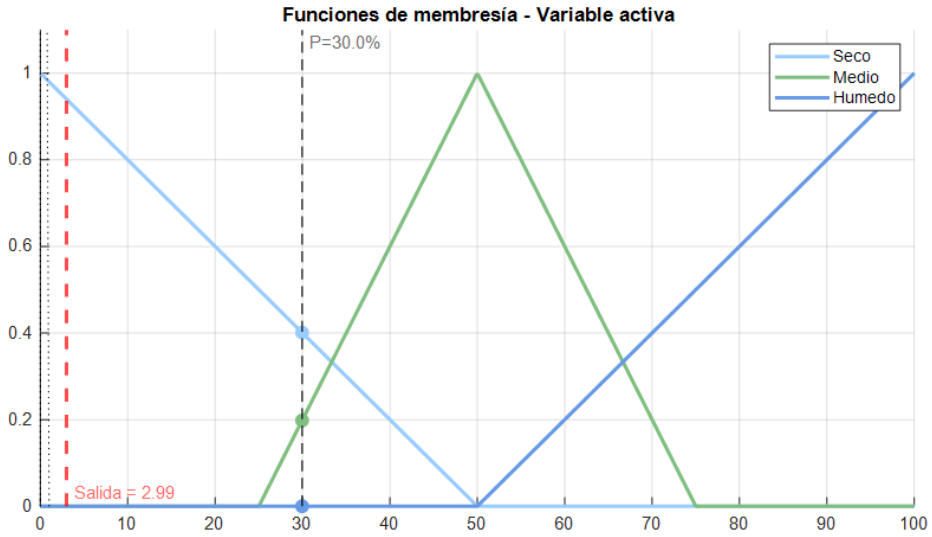
Nota. En la simulación, la variable de temperatura se fijó en un valor de referencia correspondiente al 30 % de su rango total, lo cual representa una condición ambiental moderadamente baja y sirve para ilustrar la activación de los conjuntos difusos definidos para esta entrada. Elaboración propia.

Humedad

La variable humedad relativa también se definió en el rango de 0 a 100 %, con tres funciones de membresía:

- **Seco**: predominante en valores bajos (0–50).
- **Medio**: representativa de valores intermedios (25–75).
- **Húmedo**: activa en valores altos (50–100).

Figura 10. Funciones de membresía de la variable "Humedad"



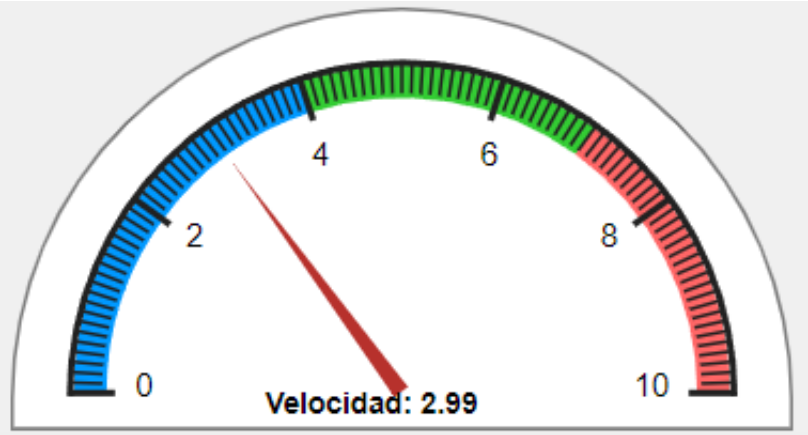
Nota. La variable de humedad relativa se estableció en un 30 % de su rango total, lo que representa un nivel intermedio-bajo dentro de la simulación. Este valor se utiliza únicamente con fines ilustrativos para mostrar la activación de los conjuntos difusos asociados a la humedad. Elaboración propia.

Velocidad del ventilador

La salida del sistema, correspondiente a la velocidad del ventilador (0–10), se estructuró con tres conjuntos difusos:

- **Lento**: entre 0 y 5.
- **Medio**: entre 2.5 y 7.5.
- **Rápido**: entre 7 y 10.

Figura 11. Medidor semicircular de salida difusa: Velocidad del ventilador



Nota. La salida del sistema difuso arrojó una velocidad de ventilador de 2.99. Este valor es únicamente ilustrativo dentro del modelo y corresponde a una escala arbitraria de 0 a 10, sin dimensión física real, empleada con fines de análisis comparativo en la simulación. Elaboración propia.

8.4. Base de reglas

El comportamiento del sistema de inferencia difuso se estructura a través de una base de reglas, que constituye el núcleo de su toma de decisiones. Estas reglas se diseñaron con un enfoque intuitivo, basado en la experiencia humana de confort térmico. La idea fundamental es que la necesidad de ventilación aumenta a medida que crecen la temperatura y la humedad. En consecuencia, el ventilador debe operar en niveles bajos cuando las condiciones son suaves y, progresivamente, incrementar su velocidad en escenarios más exigentes.

En total se definieron nueve reglas, considerando todas las combinaciones posibles entre las categorías de temperatura (frío, templado, caliente) y humedad (seco, medio, húmedo). El Cuadro 2 presenta el conjunto de reglas difusas en formato tabular.

Cuadro 2. Base de reglas del controlador difuso Mamdani para el ventilador

Temperatura	Humedad	Regla difusa	Salida
Frío	Seco	Si Temp es Frío y Hum es Seco ⇒ Ventilador Lento	Lento
Frío	Medio	Si Temp es Frío y Hum es Medio ⇒ Ventilador Lento	Lento
Frío	Húmedo	Si Temp es Frío y Hum es Húmedo ⇒ Ventilador Medio	Medio
Templado	Seco	Si Temp es Templado y Hum es Seco ⇒ Ventilador Lento	Lento
Templado	Medio	Si Temp es Templado y Hum es Medio ⇒ Ventilador Medio	Medio
Templado	Húmedo	Si Temp es Templado y Hum es Húmedo ⇒ Ventilador Rápido	Rápido
Caliente	Seco	Si Temp es Caliente y Hum es Seco ⇒ Ventilador Medio	Medio
Caliente	Medio	Si Temp es Caliente y Hum es Medio ⇒ Ventilador Rápido	Rápido
Caliente	Húmedo	Si Temp es Caliente y Hum es Húmedo ⇒ Ventilador Rápido	Rápido

Estas reglas reflejan que:

- En condiciones suaves (frío y seco/medio), la ventilación no es prioritaria, por lo que el ventilador se mantiene en velocidad lenta, asegurando un bajo consumo energético.
- En condiciones intermedias (templado o calor seco), el ventilador se ajusta a velocidad media, suficiente para mejorar la sensación de confort sin excesivo gasto.
- En condiciones críticas (alta temperatura y/o humedad), el sistema activa la velocidad rápida, priorizando la renovación del aire y la disipación del calor.

Es importante resaltar que este diseño no busca calcular un valor matemáticamente óptimo, sino representar la lógica humana en situaciones cotidianas. De esta manera, el sistema resulta más intuitivo y robusto frente a incertidumbres.

8.5. Motor de inferencia y defuzzificación

En este simulador (ventilador) se utilizó el motor de inferencia Mamdani, por ser un enfoque intuitivo y ampliamente empleado en control difuso. Este motor evalúa las reglas de la base establecida considerando los grados de pertenencia de las entradas temperatura y humedad, y genera como resultado un conjunto difuso de salida asociado a la velocidad del ventilador.

Para obtener un valor único a partir de este conjunto difuso, se aplicó el método de defuzzificación del centroide, el cual corresponde al cálculo del punto de equilibrio de la superficie activada. Dicho valor se interpreta como un promedio ponderado de todas las áreas activadas por las reglas, siguiendo la formulación presentada en la ecuación (7).

En el caso específico de la simulación con una temperatura de 30 °C y una humedad de 30 % (véanse las Figuras 9 y 10), los valores de entrada presentan grados de pertenencia intermedios:

- La temperatura de 30 °C pertenece parcialmente a los conjuntos Frío y Templado.
- La humedad del 30 % pertenece parcialmente a los conjuntos Seco y Medio.

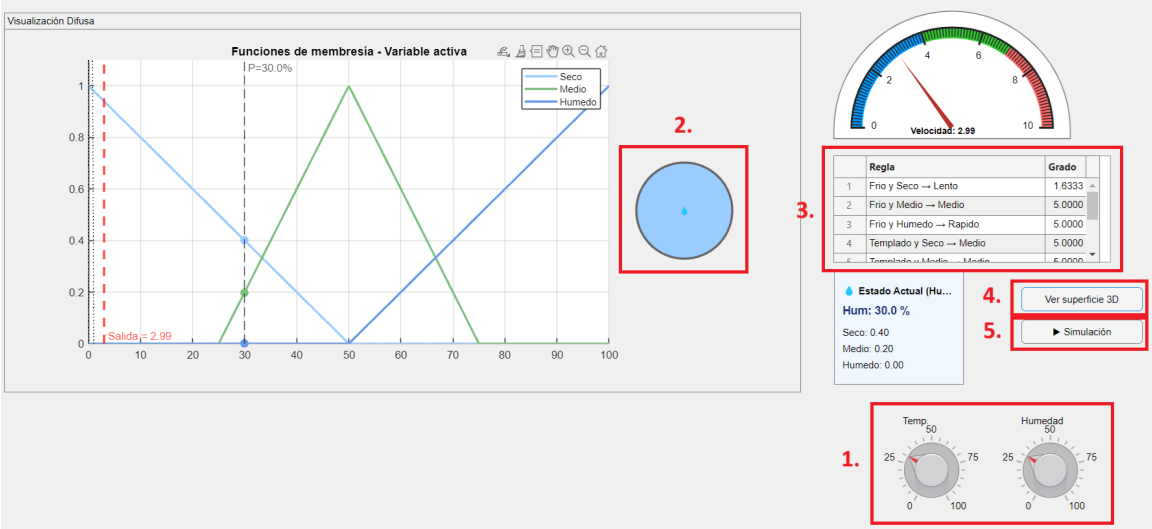
Esto provoca la activación simultánea de varias reglas de la base, algunas asociadas a la salida Lento y otras a la salida Medio. Después de la agregación, el método del centroide determina un valor preciso intermedio, aproximadamente 3.5 en la escala 0–10, lo que indica que el ventilador debe operar a una velocidad moderada. Este resultado refleja el comportamiento esperado: condiciones ambientales intermedias conducen a una respuesta intermedia del sistema.

8.6. Interfaz gráfica (GUI)

La interfaz gráfica proporciona controles y visualizaciones en tiempo real para configurar las entradas y observar la respuesta del controlador difuso. Para su descripción se enumeran a continuación los componentes (véase la Figura 12):

1. **Perillas de temperatura y humedad:** permiten ajustar directamente las entradas del sistema (temperatura en °C y humedad en %). Al modificarlos, la interfaz actualiza en tiempo real los grados de pertenencia y la activación de reglas.
2. **Indicador de estado térmico:** LED/ícono que resume la condición dominante de la temperatura (frío, templado o caliente), como apoyo cualitativo al valor continuo del medidor.
3. **Tabla de reglas activadas:** lista, en el instante actual, las reglas disparadas y su grado de activación α; facilita la trazabilidad entre entradas, inferencia y salida.
4. **Superficie 3D de salida:** representa la relación temperatura–humedad–velocidad del ventilador, validando el diseño de membresías y base de reglas en todo el dominio operativo.
5. **Botón de simulación (barrido automático):** ejecuta un recorrido programado de temperatura y humedad (rango, paso y periodo configurables), actualizando en tiempo real la tabla de reglas y la salida; útil para evaluar el comportamiento dinámico global.

Figura 12. Vista general de la GUI con componentes numerados 1–5



Nota. La imagen integra los elementos principales: 1) perillas; 2) indicador térmico; 3) tabla de reglas; 4) botón de superficie 3D; 5) botón de simulación. Elaboración propia.

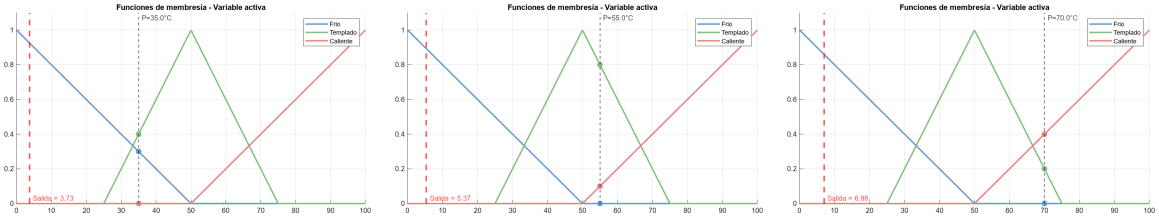
8.7. Resultados de simulación

Esta sección presenta dos perspectivas complementarias: (i) el comportamiento dinámico del sistema bajo un barrido automático de entradas y (ii) ejemplos puntuales de combinaciones temperatura–humedad con la correspondiente respuesta del controlador (velocidad del ventilador). En ambos casos, la salida precisa se obtiene mediante el método del centroide (véase la ecuación (7)).

8.7.1. Comportamiento dinámico con barrido automático

Se ejecutó un barrido progresivo de temperatura y humedad a lo largo del tiempo, observándose que la velocidad se incrementa de manera suave y continua cuando las entradas aumentan, y disminuye sin saltos cuando lo hacen a la baja. Esto evidencia la naturaleza gradual de la inferencia Mamdani y la defuzzificación por centroide: la salida sigue un promedio ponderado de las áreas activadas, evitando discontinuidades.

Figura 13. Secuencia de imágenes correspondientes al barrido automático



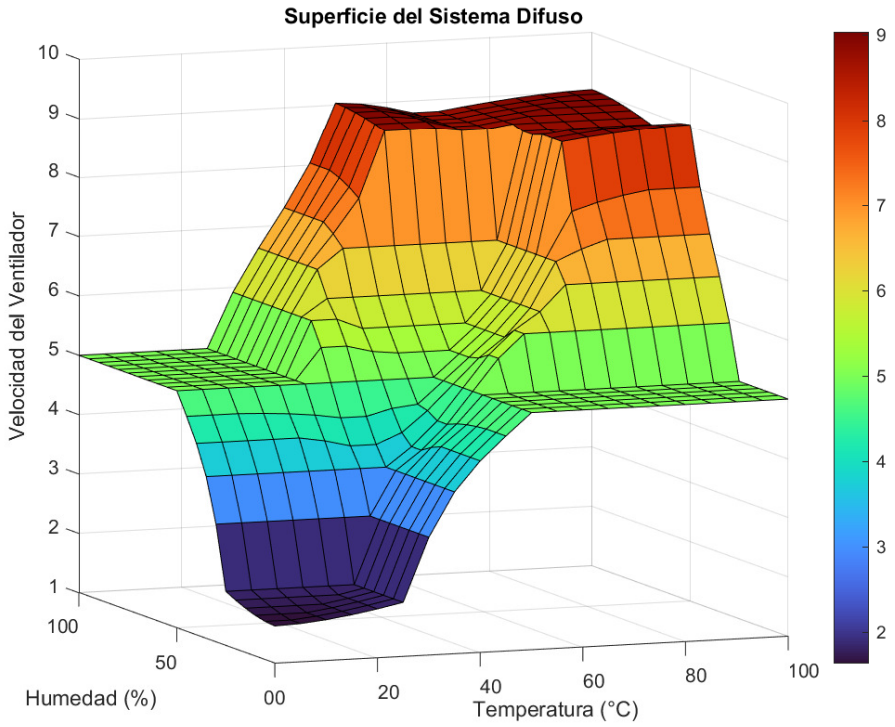
(a) Temp. y humedad al 35 %. (b) Temp. y humedad al 55 %. (c) Temp. y humedad al 70 %.

Nota. Las imágenes muestran de forma secuencial el desarrollo del proceso dentro del entorno de simulación, demostrando la pertenencia de cada uno de los conjuntos en cada una de las etapas. Elaboración propia.

8.7.2. Superficie 3D del sistema difuso

Además del medidor semicircular, la interfaz incluye una representación tridimensional de la relación entre las variables de entrada (temperatura y humedad) y la salida (velocidad del ventilador). Esta superficie, ilustrada en la Figura 14, permite observar cómo el sistema responde en todo el rango operativo, mostrando transiciones suaves en condiciones intermedias y un aumento progresivo en escenarios extremos de calor y/o humedad.

Figura 14. Superficie 3D del sistema difuso: temperatura y humedad vs velocidad del ventilador



Nota. Superficie tridimensional simple que ilustra la salida nítida del controlador difuso (Mamdani con defuzzificación por centroide). Se evidencian las regiones de mayor ventilación y las zonas de transición entre estados. Elaboración propia.

8.7.3. Ejemplos de combinaciones temperatura/humedad y respuesta

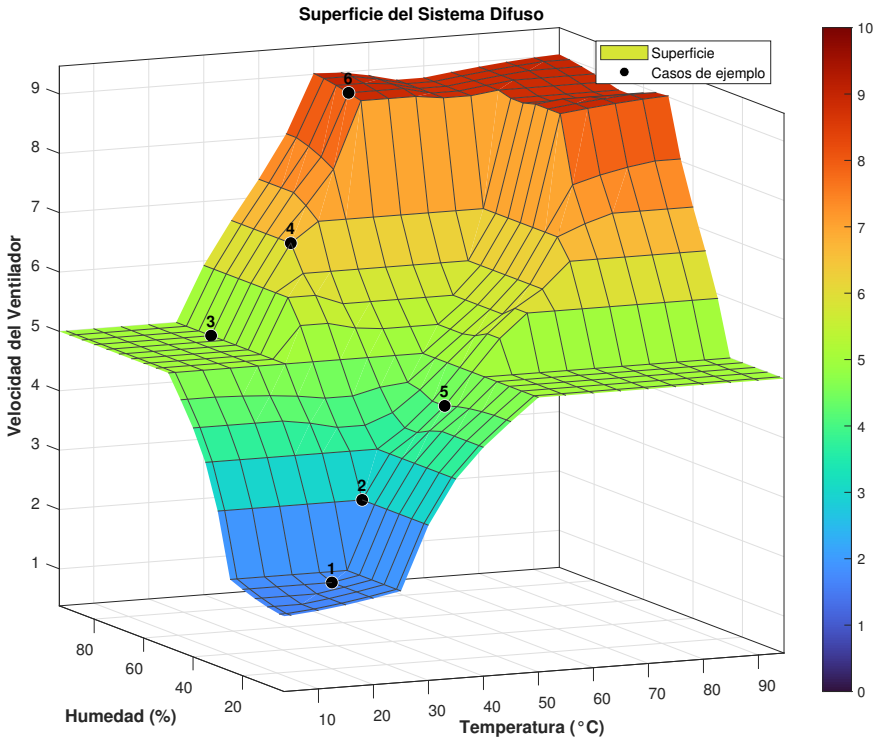
A continuación se listan ejemplos representativos. Los valores de salida son aproximados y dependen de las funciones de membresía y la base de reglas definidas (se indican las reglas dominantes para cada caso). Los mismos casos se ilustran sobre la superficie 3D en la Figura 15, donde cada marcador numerado corresponde a una fila del Cuadro 3.

Cuadro 3. Ejemplos de combinaciones de temperatura y humedad y su salida estimada

Punto	Temperatura (°C)	Humedad (%)	Velocidad (0–10)	Regla(s) dominante(s)
1	20	20	≈ 2.0	Frío & Seco ⇒ Lento
2	30	30	≈ 3.5	Templado & Medio ⇒ Medio; Frío/Húmedo ⇒ Lento (secund.)
3	25	80	≈ 6.0	Templado & Húmedo ⇒ Rápido (recorte medio)
4	35	70	≈ 7.8	Caliente & Húmedo ⇒ Rápido
5	45	30	≈ 6.5	Caliente & Seco ⇒ Medio (con aporte de Rápido si Hum = Medio)
6	50	80	≈ 9.0	Caliente & Húmedo ⇒ Rápido

Nota. Valores de salida estimados mediante centroide (Ecuación 7). La contribución exacta depende de los grados de pertenencia activados y de la agregación en cada punto. Elaboración propia.

Figura 15. Puntos de ejemplo sobre la superficie 3D (temperatura–humedad–velocidad)



Nota. Los puntos 1–6 corresponden a los casos del Cuadro 3. La altura de cada marcador representa la velocidad precisa (centroide). Se aprecian transiciones suaves y mayor ventilación en zonas de alta temperatura y/o humedad. Elaboración propia.

CAPÍTULO 9

Sistema de control de un brazo robótico

Este capítulo presenta un brazo robótico planar de tres grados de libertad $(\theta_1, \theta_2, \theta_3)$ con dos eslabones principales L_1 y L_2, empleado como caso representativo de control en Ingeniería Mecatrónica con visualización 3D en MATLAB. El sistema compara controladores en dominio cartesiano (PID, LQR y Mamdani difuso con ganancia adaptativa) montados sobre una arquitectura de *Damped Least Squares* (DLS) para el mapeo cartesiano–articular.

9.1. Modelo del brazo robótico

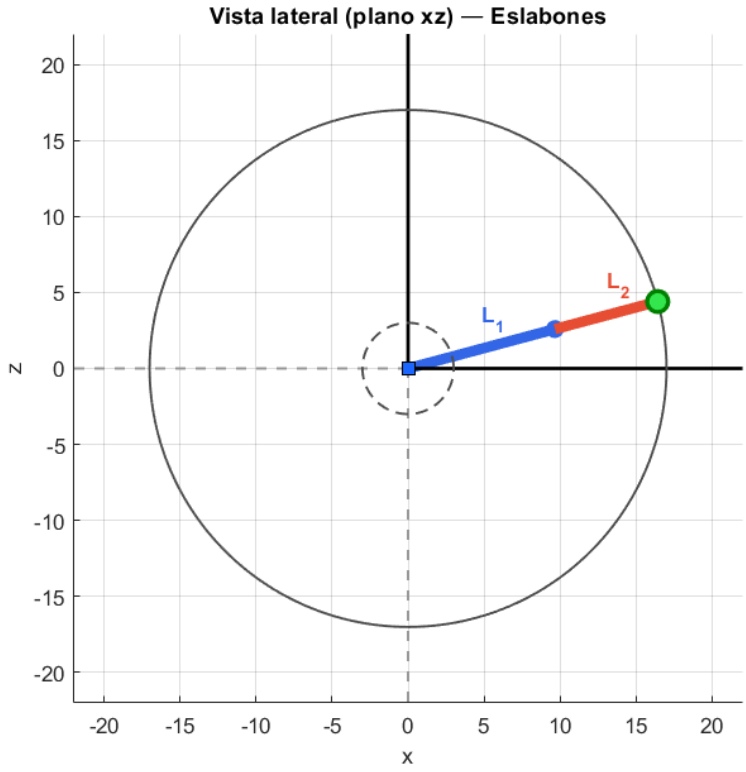
9.1.1. Descripción física del brazo robótico

Eslabones

El manipulador empleado se compone de dos eslabones principales L_1 y L_2, conectados mediante articulaciones de tipo revoluta que permiten el movimiento en el plano XZ. La configuración planar simplifica el análisis cinemático sin perder generalidad en el estudio del control y la respuesta dinámica del sistema.

La estructura general del manipulador se ilustra en la Figura 16, donde se muestran los eslabones y la ubicación del efector final.

Figura 16. Vista lateral (plano xz) del brazo robótico



Nota. Representación lateral del manipulador planar con dos eslabones: el primero (L_1) en color azul y el segundo (L_2) en rojo, terminando en el efector final (punto verde). La figura muestra la orientación relativa de las articulaciones y el espacio de trabajo aproximado del sistema. Elaboración propia.

Las articulaciones que componen el brazo se describen a continuación:

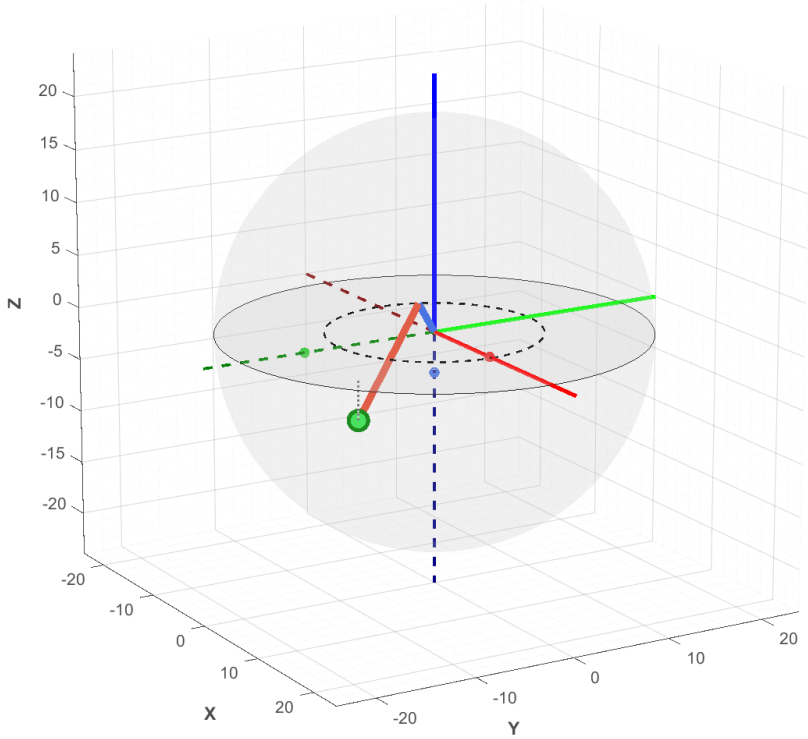
- θ_1: articulación en la base, rota el primer eslabón respecto al marco fijo.
- θ_2: articulación intermedia, conecta el primer eslabón con el segundo.
- θ_3: articulación en el efector, empleada como muñeca virtual para posicionamiento fino en el plano XY y rotación en el eje Z.

La combinación de estas articulaciones permite un espacio de trabajo esférico con radio limitado por:

$$r_{\text{mín}} = |L_1 - L_2|, \qquad r_{\text{máx}} = L_1 + L_2, \tag{19}$$

lo que restringe los objetivos posibles dentro del simulador [34].

Figura 17. Figura 3D del manipulador de 3 GDL con dos eslabones principales L_1, L_2 y articulaciones $\theta_1, \theta_2, \theta_3$



Nota. Estructura física modelada en el simulador MATLAB, con articulaciones revolutas con dimensiones distintas a las originales para demostrar como el alcance del brazo varía según el largo de las mismas; siendo L1 más corto que L2. Elaboración propia.

9.2. Cinemática directa e inversa

Cinemática directa

La cinemática directa permite calcular la posición cartesiana del efector final (x_e, y_e) a partir de los ángulos articulares y las longitudes de los eslabones. Para un manipulador planar de dos eslabones, la formulación es [35] [13]:

$$x_e = L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2), \qquad y_e = L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2). \tag{20}$$

En el simulador, estas ecuaciones se emplean en la función `fk3d()` para graficar la posición instantánea del efector final y verificar el seguimiento de la referencia.

9.2.1. Cinemática inversa

La cinemática inversa busca determinar los ángulos θ_1, θ_2 que permiten alcanzar una posición deseada $(x_{\text{ref}}, y_{\text{ref}})$. Una solución cerrada general es [34] [13]:

$$\theta_2 = \arccos\left(\frac{x_{\text{ref}}^2 + y_{\text{ref}}^2 - L_1^2 - L_2^2}{2L_1L_2}\right), \tag{21}$$

$$\theta_1 = \arctan 2(y_{\text{ref}}, x_{\text{ref}}) - \arctan 2(L_2 \sin\theta_2,\ L_1 + L_2\cos\theta_2)\,. \tag{22}$$

Sin embargo, el simulador implementa la cinemática inversa en forma numérica mediante el método de *Damped Least Squares* (DLS):

$$\Delta\theta = J^{\mathsf{T}}\,(JJ^{\mathsf{T}} + \lambda^2 I)^{-1}\,e, \tag{23}$$

donde $e = (x_{\text{ref}} - x_e,\, y_{\text{ref}} - y_e)$ es el error cartesiano. Esta aproximación iterativa asegura robustez frente a singularidades y ruido en el Jacobiano.

9.2.2. Aplicación en el simulador

En el entorno MATLAB, la cinemática directa, como se aprecia en la ecuación (20), se usa para actualizar la gráfica 3D en cada iteración, mientras que la cinemática inversa, o sea en la ecuación (21)–(22) se sustituye por el esquema numérico de la ecuación (23), con amortiguamiento adaptativo λ. Esta integración permite comparar distintos controladores (PID, LQR, Fuzzy) sin necesidad de una solución analítica cerrada.

9.3. Metodología de control

La estrategia trabaja en el dominio cartesiano: se calcula una señal u_{cart} distinta según el modo (PID, LQR o Mamdani) y luego se obtiene el incremento articular mediante DLS:

$$\Delta\theta \;=\; D\,J^{\mathsf{T}}\Big(JJ^{\mathsf{T}} + \lambda^2 I\Big)^{-1}\,u_{\text{cart}}, \tag{24}$$

donde J es el Jacobiano, λ el amortiguamiento y D una matriz (identidad o ganancia adaptativa) [34] [13].

9.3.1. Control PID aplicado al brazo robótico

El PID cartesiano por eje usa la ecuación (10), con un *anti-windup* por saturación de la integral $|I| \le I_{\text{máx}}$. Las ganancias se ajustan desde la GUI (valores por defecto indicados en el texto) [23]. Esta señal entra en (24) con $D = I$.

Control LQR aplicado al brazo robótico

El controlador LQR se plantea considerando un sistema linealizado en el dominio cartesiano, con matrices de ponderación normalizadas $Q = I$ y $R = I$ en la función de costo

$$J \;=\; \sum \big(x^{\mathsf{T}} Q\, x \;+\; u^{\mathsf{T}} R\, u\big). \tag{25}$$

La ley de control se define como

$$u_{\text{cart}} \;=\; -K\, x, \qquad K \;=\; R^{-1} B^{\mathsf{T}} P, \tag{26}$$

donde P resuelve la ecuación algebraica de Riccati discreta (DARE):

$$P \;=\; A^{\mathsf{T}} P A \;-\; A^{\mathsf{T}} P B \left(R + B^{\mathsf{T}} P B\right)^{-1} B^{\mathsf{T}} P A \;+\; Q. \tag{27}$$

En el simulador, `compute_dlqr_safe` calcula la ganancia K, y la salida de control se aplica de acuerdo con la ecuación (24), considerando $D = I$ [34].

9.3.2. Control difuso Mamdani (diseño, reglas y operadores)

El sistema Mamdani recibe como entradas los errores cartesianos E_x, E_y, E_z y genera una ganancia difusa escalar K (expresada en grados), la cual ajusta la magnitud del movimiento articular del manipulador. Esta ganancia se utiliza para escalar el vector de paso articular mediante una matriz diagonal:

$$D = \text{diag}(K_\theta, K_\theta, K_\theta), \tag{28}$$

donde cada componente K_θ corresponde a la conversión de la ganancia K de grados a radianes.

De esta forma, el sistema difuso modula dinámicamente la velocidad o amplitud de movimiento de cada articulación según los errores de posición detectados.

Las funciones de membresía empleadas son de tipo triangular, y representan categorías lingüísticas que describen el comportamiento del sistema en términos cualitativos. Estas categorías son:

$$\text{NG} < \text{NM} < \text{Z} < \text{PM} < \text{PG},$$

donde: NG = negativo grande, NM = negativo mediano, Z = zona cero, PM = positivo mediano y PG = positivo grande. Cada una define un conjunto difuso que representa el nivel relativo de error o ganancia en el sistema.

Función de membresía triangular. Para un conjunto difuso $A = \text{trimf}(a, b, c)$, la membresía de un valor x se calcula tal y como lo indica la ecuación (6).

Operadores difusos. Los operadores usados corresponden al esquema Mamdani clásico [11], [19]:

$$\text{AND (t-norma mínima):} \quad \mu_{A\wedge B}(x) \;=\; \min\big(\mu_A(x),\, \mu_B(x)\big), \tag{29}$$

$$\text{OR (s-norma máxima):} \quad \mu_{A\vee B}(x) \;=\; \max\big(\mu_A(x),\, \mu_B(x)\big), \tag{30}$$

$$\text{Fuerza de disparo (3 entradas):} \quad \alpha_r \;=\; \min\big(\mu_{A_x}(E_x),\, \mu_{A_y}(E_y),\, \mu_{A_z}(E_z)\big), \tag{31}$$

$$\text{Implicación (recorte):} \quad \mu_{B_r}^{\text{imp}}(y) \;=\; \min\big(\alpha_r,\, \mu_{B_r}(y)\big), \tag{32}$$

$$\text{Agregación (máximo):} \quad \mu_{\text{OUT}}(y) \;=\; \max_r\, \mu_{B_r}^{\text{imp}}(y). \tag{33}$$

El valor $K^\star$ resultante se acota al rango $[K_{\text{mín}}, K_{\text{máx}}]$ y se utiliza en (24) mediante D, permitiendo que el sistema ajuste dinámicamente la respuesta de control ante los diferentes niveles de error.

Base de reglas. El sistema utiliza una base de reglas lingüísticas que relaciona los errores cartesianos E_x, E_y, E_z con la ganancia difusa K. Cada regla sigue la forma:

$$\text{Si } (E_x \text{ es } L_x) \wedge (E_y \text{ es } L_y) \wedge (E_z \text{ es } L_z) \Rightarrow (K \text{ es } L_K), \tag{34}$$

donde $L_\bullet \in \{\text{NG,NM,Z,PM,PG}\}$. El conjunto de reglas lingüísticas se inspira en esquemas similares propuestos en la literatura para controladores difusos de convertidores [36], y se resumen en el Cuadro 4.

Cuadro 4. Cuadro de reglas difusas empleadas en el sistema Mamdani del brazo robótico de 3GDL

E_x	E_y	E_z	K
NG	NG	NG	NG
NM	NG	PM	NM
Z	Z	NM	Z
PM	NM	NM	Z
PG	PG	Z	PM

Nota. Grupo representativo de reglas lingüísticas que combinan los errores cartesianos E_x, E_y, E_z para determinar la ganancia difusa K. Adaptado de [36].

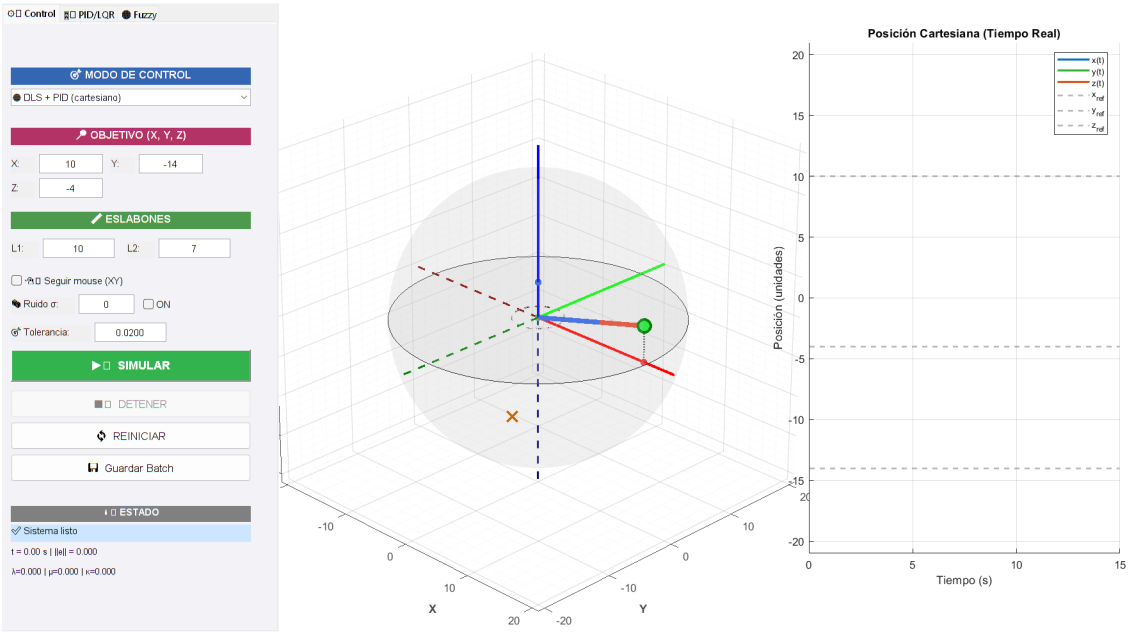
9.4. Implementación en MATLAB

9.4.1. Interfaz gráfica y visualización 3D

La GUI presenta una ventana única compuesta por: (i) un panel de controles distribuido en tabs: Control, PID/LQR y Fuzzy; (ii) una escena 3D con el brazo, el objetivo y proyecciones a los tres planos; y (iii) una gráfica en vivo de x_e, y_e, z_e vs. tiempo con líneas de referencia. El horizonte temporal de la gráfica se sincroniza automáticamente dependiendo del modo de control que se elija (ya se tiene predeterminado un tiempo para cada uno).

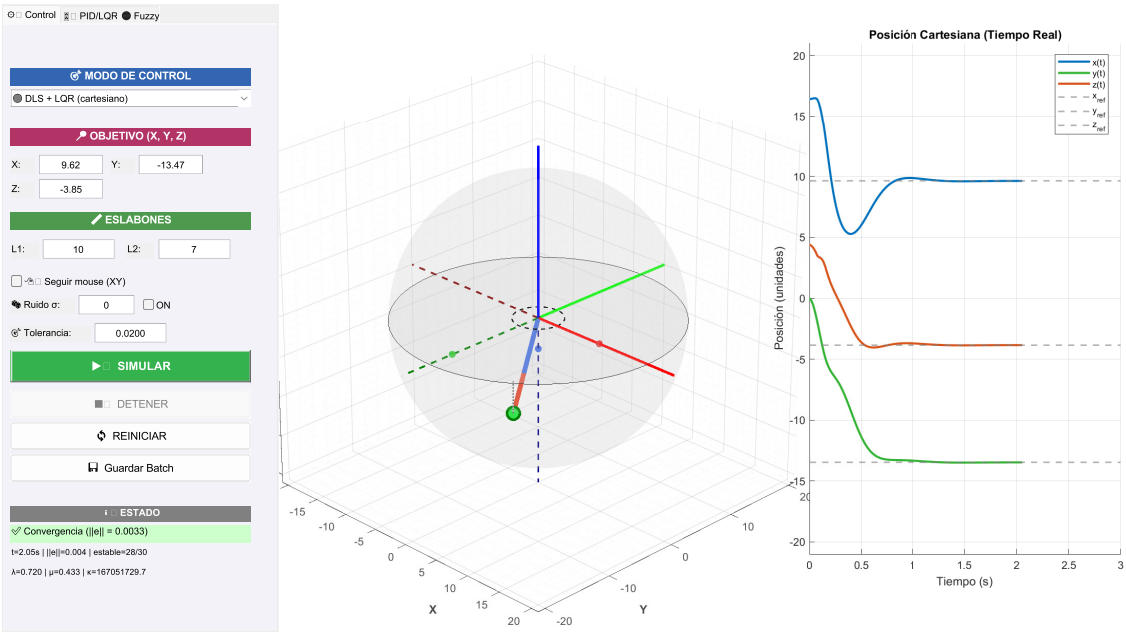
La interacción incluye "seguir mouse" en XY, arrastre con clic derecho para ajustar z_{ref}, y se puede utilizar la rueda del mouse para variaciones incrementales en z. Se muestran indicadores de condición del mapeo cartesiano–articular (λ, μ, κ) en tiempo real. El botón "Guardar Batch" ejecuta corridas automáticas sobre un anillo de 12 puntos en un plano z fijo y almacena resultados en un archivo CSV, además de generar un gráfico comparativo de desempeño por controlador.

Figura 18. Interfaz completa del simulador de brazo robótico de 3GDL: tabs de control, escena 3D y gráfica en vivo



Nota. Se observan los campos de modo, objetivos, longitudes, opción de ruido, parámetros PID/LQR, Fuzzy, y la tolerancia a la que se quiere establecer la simulación; a la derecha, el 3D con proyecciones y la gráfica x_e, y_e, z_e con $x_{\text{ref}}, y_{\text{ref}}, z_{\text{ref}}$. Elaboración propia.

Figura 19. Detalle de visualización: escena 3D con objetivo y proyecciones; y gráfica en vivo x_e, y_e, z_e vs. tiempo



Nota. La escena 3D muestra el brazo (dos eslabones) y el efector final; el objetivo se encuentra cubierto por el efector final de color verde; también se observan las líneas de proyección de los planos coordenados. La gráfica en vivo ilustra el error de las 3 articulaciones disminuyendo a medida que la simulación avanza. Elaboración propia.

9.4.2. Configuración de parámetros y modos de simulación

La proyección cartesiano–articular se realiza mediante la ecuación (24).

El paso articular se suaviza y limita por seguridad:

$$\Delta\theta \leftarrow \beta\,\Delta\theta_{\text{prev}} + (1-\beta)\,\Delta\theta, \qquad |\Delta\theta_i| \le \text{deg2rad}(\texttt{K\_safety\_ang\_deg}), \tag{35}$$

donde β es el coeficiente de suavizado aplicado al incremento articular, $\Delta\theta_{\text{prev}}$ corresponde al paso calculado en la iteración anterior, $\Delta\theta$ es el incremento propuesto por el método de cinemática inversa, y `K_safety_ang_deg` define el límite máximo permitido para cada variación articular expresado en grados. Este límite se aplica componente a componente para asegurar que $|\Delta\theta_i|$ no exceda un cambio angular seguro en cada articulación.

Aplicando además límites articulares realistas para θ_2 y θ_3, en modo Fuzzy la ganancia adaptativa K (grados) escala el incremento articular mediante las ecuaciones (24) y (28).

Por otro lado, el método de desfuzzificación se selecciona en la interfaz entre *Centroid* y *Bisector*.

En el Cuadro 5 se presentan los principales parámetros configurables del simulador del brazo robótico, inicializados con valores por defecto en el código de MATLAB.

9.4.3. Tolerancia de error cartesiano (`tol`)

En cada iteración k se calcula el error cartesiano (definido en control de manipuladores en espacio tarea [13], [34], [35]):

$$e_k \;=\; \begin{bmatrix} x_{\text{ref}} - x_e \\ y_{\text{ref}} - y_e \\ z_{\text{ref}} - z_e \end{bmatrix}, \qquad \|e_k\| \;=\; \sqrt{(x_{\text{ref}} - x_e)^2 + (y_{\text{ref}} - y_e)^2 + (z_{\text{ref}} - z_e)^2}. \tag{36}$$

Se declara éxito cuando

$$\|e_k\| \;<\; \texttt{tol} \quad \text{durante} \quad \texttt{stable\_needed} \text{ iteraciones consecutivas.} \tag{37}$$

donde `stable_needed` indica el número mínimo de iteraciones consecutivas en las que el efector debe permanecer dentro de esa región para considerar que la tarea ha sido resuelta de manera estable.

Interpretación geométrica: $\|e_k\| < \texttt{tol}$ define una bola de aceptación de radio `tol` centrada en el objetivo cartesiano; el movimiento se considera exitoso cuando el efector permanece dentro de esa región.

Relación con los controladores: `tol` actúa como criterio común de parada y éxito. En PID se recomienda evitar *windup* cerca del objetivo; en LQR, tolerancias más estrictas suelen implicar mayores pesos en Q; en Fuzzy, la región central reduce K hacia $K_{\text{mín}}$, mitigando *chattering* (defuzzificación por centroide [19]).

9.5. Resultados de la simulación

9.5.1. Evolución de posiciones articulares

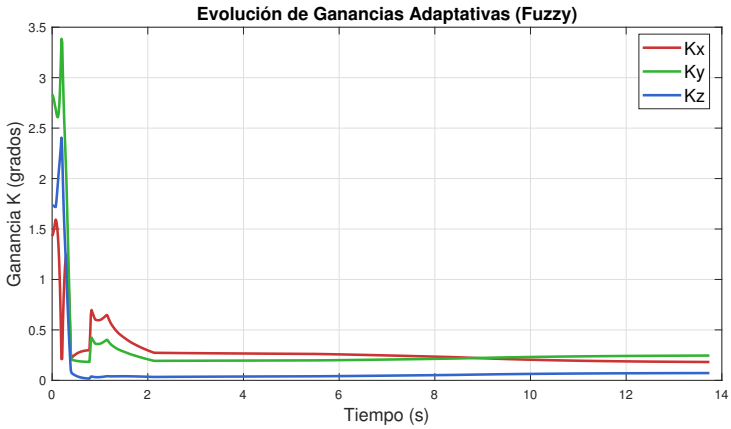
La evolución de las articulaciones $\theta_1, \theta_2, \theta_3$ a lo largo del tiempo ilustra cómo el esquema de control proyectado por DLS genera trayectorias articulares suavizadas (ec.(35)) mientras respeta el límite de paso por seguridad. La Figura 20 se obtuvo a partir de una corrida típica (condiciones por defecto del Cuadro 5).

Cuadro 5. Parámetros por defecto del simulador

Parámetro	Valor	Descripción
L_1, L_2	10, 7	Longitudes de eslabones
$\theta^{(0)}_{1,2,3}$	0°, 15°, 0°	Configuración inicial
$x_{\text{ref}}, y_{\text{ref}}, z_{\text{ref}}$	10, −14, −4	Objetivo inicial
`max_iter`	800	Iteraciones máximas
`tol`	0.02	Tolerancia de error cartesiano
`stable_needed`	20	Iteraciones bajo tolerancia para éxito
β	0.85	Suavizado de paso articular
`K_safety_ang_deg`	8	Límite de paso por articulación
$\lambda_{\text{base}}, \lambda_{\text{smallE}}$	0.15, 0.35	Amortiguamiento DLS
$K_{\text{mín}}, K_{\text{máx}}$ (Fuzzy)	0.6°, 6°	Rango de ganancia adaptativa
PID (K_p, K_i, K_d)	1.0, 0.01, 1.5	Ganancias por defecto
LQR $(Q_x, Q_y, Q_z; R)$	0.05, 0.05, 0.05; 10	Pesos cartesianos
Ruido	OFF; $\sigma = 0.00$	Opción de inyección gaussiana
λ adaptativo	ON; $a_\mu = 0,\ b_\kappa = 0$	Coeficientes de normalización μ_N, κ_N

Nota. Valores tal como se inicializan en el programa. Todos son editables desde la GUI.

Figura 20. Evolución temporal de las ganancias articulares K_x, K_y y K_z

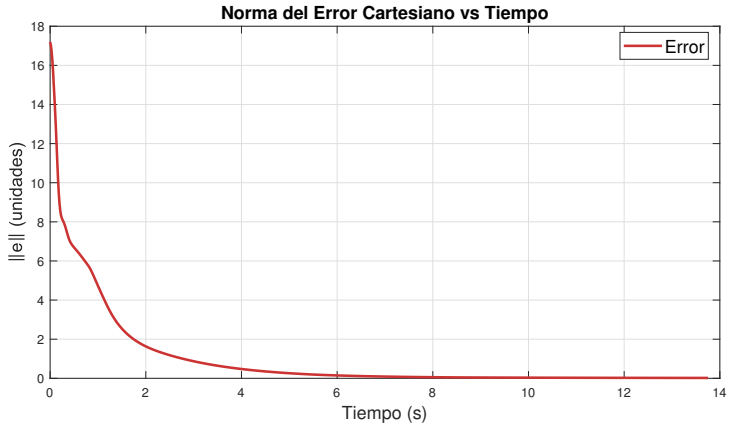


Nota. Las curvas muestran la evolución de las ganancias K_x, K_y y K_z empleadas por el controlador Mamdani (método de centroide) durante el movimiento hacia un objetivo fijo. La figura evidencia el efecto del filtro exponencial ($\beta = 0.85$) Elaboración propia.

9.5.2. Errores de seguimiento y convergencia

La norma del error cartesiano $\|e\|$ decrece hasta cumplir el criterio de éxito: $\|e\| < \texttt{tol} = 0.02$ durante `stable_needed` $= 20$ iteraciones consecutivas.

Figura 21. Error cartesiano $\|e\|$ vs. tiempo para el controlador utilizado en la simulación

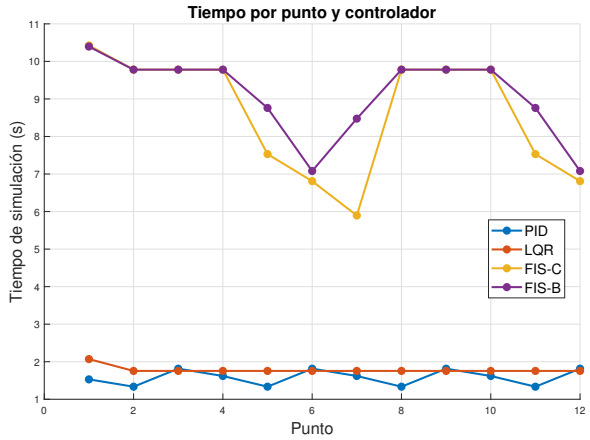


Nota. La curva corresponde al error cartesiano generado por el mismo controlador empleado en la Figura 20 (controlador difuso Mamdani con ganancias articulares adaptativas K_x, K_y y K_z). El gráfico muestra la disminución de $\|e\|$ hasta cumplir el criterio de convergencia utilizado en la simulación: $\|e\| < \texttt{tol} = 0.02$ durante `stable_needed`=20 iteraciones consecutivas. Elaboración propia.

9.5.3. Comparación entre controladores

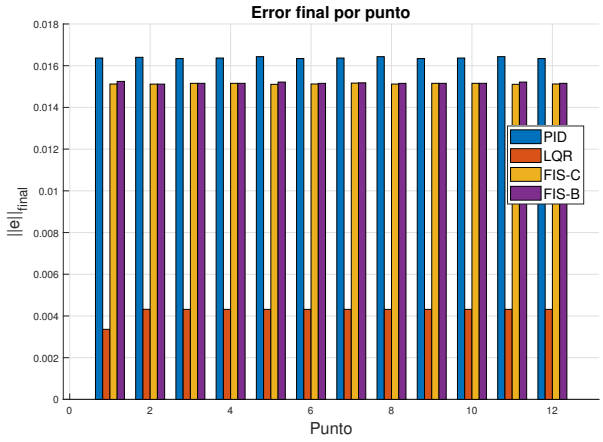
Se evaluaron los tres modos (PID, LQR y Fuzzy (centroide y bisector)) sobre un *batch* de 12 objetivos uniformemente distribuidos en un anillo del plano z fijo; en lo que sigue, estos objetivos se denominarán como puntos. Para cada modo se reportan: timepo de convergencia por cada uno de los puntos hasta el éxito, error final $\|e\|_{\text{final}}$ por punto y diferencias $|\Delta X|, |\Delta Y|, |\Delta Z|$. Los resultados permiten contrastar estabilidad, rapidez y precisión.

Figura 22. Tiempo por punto y por controlador



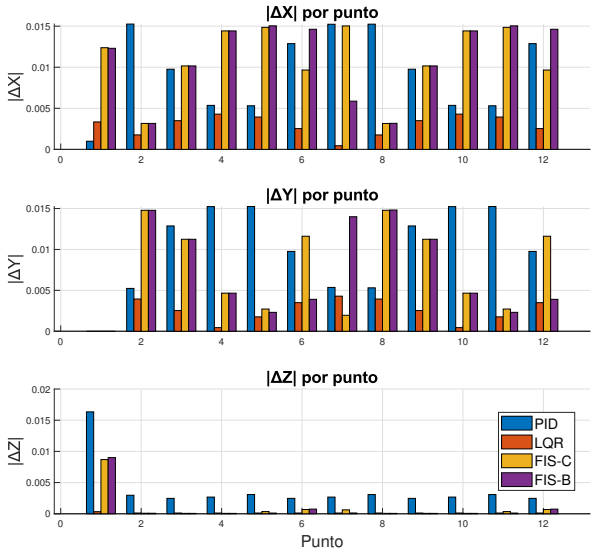
Nota. Esta gráfica muestra la rapidez relativa de convergencia por objetivo (punto) y controlador. Elaboración propia.

Figura 23. Error final $\|e\|_{\text{final}}$ por punto y por modo



Nota. Barras agrupadas por modo en cada objetivo del anillo. Valores menores indican mayor precisión al final de la corrida. Elaboración propia.

Figura 24. Diferencias $|\Delta X|, |\Delta Y|, |\Delta Z|$ entre el punto final y la referencia, por punto y por modo



Nota. Comparativa por eje cartesiano. Útil para analizar sesgos o anisotropías en el desempeño por controlador. Elaboración propia.

9.6. Discusión de resultados

La evaluación comparativa evidencia diferencias claras entre los cuatro enfoques. El controlador PID presenta los tiempos de convergencia más bajos y relativamente constantes entre los distintos puntos (Fig. 22), lo que refleja una acción rápida pero no necesariamente la más precisa. El controlador LQR muestra un desempeño más equilibrado: mantiene errores finales menores y muy uniformes (Fig. 23), además de trayectorias suaves en términos de variación cartesiana (Fig. 24), lo cual lo posiciona como el método más consistente en precisión.

En contraste, los dos controladores difusos (FIS-C y FIS-B) cumplen con los objetivos de estabilidad y mantienen el error dentro de la tolerancia, pero el acotamiento de la ganancia K los vuelve más conservadores. Esto se evidencia en tiempos de convergencia mayores (Fig. 22) y en una mayor variabilidad entre puntos, especialmente en el caso del método Bisector. Sin embargo, ambos ofrecen una alternativa explicable basada en reglas lingüísticas, lo que puede resultar ventajoso en escenarios donde la interpretabilidad sea prioritaria.

En conjunto, los resultados muestran que si bien PID ofrece rapidez y el enfoque difuso aporta interpretabilidad, es el LQR el que logra el mejor balance entre precisión, suavidad de trayectoria y consistencia global en el conjunto de prueba

CAPÍTULO 10

Sistema de control de un péndulo invertido

Este capítulo presenta un péndulo invertido de un grado de libertad θ actuado por torque en la base. Se emplea el mismo esquema general de control que en el brazo robótico del Capítulo 9: se comparan un controlador PID, un controlador LQR y un controlador difuso Mamdani, todos diseñados sobre el modelo linealizado alrededor del equilibrio inestable y una implementación en MATLAB con interfaz gráfica. Las formulaciones generales de PID, LQR y la arquitectura Mamdani ya se introdujeron en los capítulos teóricos anteriores, por lo que aquí se destacan únicamente las particularidades del péndulo invertido y las diferencias respecto al brazo robótico.

10.1. Modelo dinámico del péndulo invertido

El sistema se modela como un péndulo invertido simple de longitud L, masa concentrada m en el extremo y momento de inercia $I = mL^2$ respecto al punto de giro. El ángulo θ se mide desde la vertical estableciendo $\theta = 0$ como la posición vertical hacia arriba, coherente con la visualización del simulador.

Incluyendo el torque de control u, la fricción viscosa b y la gravedad g, la dinámica no lineal viene dada por:

$$I\,\ddot{\theta} + b\,\dot{\theta} + mgL\sin\theta \;=\; u, \tag{38}$$

donde u se encuentra acotado por el actuador a $|u| \le u_{\text{máx}}$.

En el código de MATLAB la dinámica se implementa sobre el vector de estado

$$x \;=\; \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix},$$

por medio de:

$$\dot{\theta} \;=\; \dot{\theta}, \qquad \ddot{\theta} \;=\; \frac{u_{\text{act}} - b\,\dot{\theta} - mgL\sin\theta}{I}, \tag{39}$$

donde u_{act} modela el efecto de un actuador de primer orden:

$$\tau_u\,\dot{u}_{\text{act}} \;=\; \text{sat}_{\dot{u}}\big(u_{\text{cmd}} - u_{\text{act}}\big), \tag{40}$$

con constante de tiempo τ_u y límite en la derivada $\dot{u}$ (`max_udot` en el código). Este esquema es análogo al filtro de seguridad aplicado al incremento articular $\Delta\theta$ en el brazo robótico (ec. (35)), pero aquí se aplica directamente sobre el torque del actuador.

10.1.1. Modelo linealizado y espacio de estados

Al igual que en el brazo robótico se empleó la formulación de espacio de estados para LQR (ec. (25)–(27)), el péndulo se linealiza alrededor de la posición vertical $\theta = 0$. Para pequeñas oscilaciones, $\sin\theta \approx \theta$, de modo que la ecuación (38) se aproxima por:

$$I\,\ddot{\theta} + b\,\dot{\theta} + mgL\,\theta \;=\; u. \tag{41}$$

En forma matricial, con $x = \begin{bmatrix}\theta & \dot{\theta}\end{bmatrix}^{\mathsf{T}}$ y u como entrada,

$$\dot{x} \;=\; Ax + Bu, \qquad A \;=\; \begin{bmatrix} 0 & 1 \\ -\dfrac{mgL}{I} & -\dfrac{b}{I} \end{bmatrix}, \quad B \;=\; \begin{bmatrix} 0 \\ \dfrac{1}{I} \end{bmatrix}. \tag{42}$$

Estas matrices son precisamente las calculadas en la función `linearize_upright()` del código y se utilizan como entrada del diseño LQR, análogo al caso cartesiano del brazo donde se emplearon matrices A, B equivalentes pero de mayor dimensión.

10.2. Metodología de control

La estructura de control reutiliza la misma filosofía que en el brazo robótico (Sección 9.3): se define una señal de control u_{cmd} a partir del estado del sistema y se filtra mediante el actuador de primer orden de la ecuación (40). La diferencia principal es que ahora el control actúa directamente sobre el torque del péndulo, y el espacio de estados es unidimensional (solo el ángulo) en lugar de cartesiano.

El objetivo consiste en estabilizar el equilibrio inestable $\theta = 0$ partiendo de una condición inicial θ_0 genérica (por ejemplo, 180°), cumpliendo un criterio de estabilidad basado en umbrales de error angular y velocidad.

10.2.1. Criterio de convergencia

Mientras que en el brazo se utilizó un criterio de tolerancia cartesiana $\|e_k\| < \texttt{tol}$ (ecs. (36) y (37)), para el péndulo se emplea un criterio escalar sobre θ y $\dot{\theta}$. En el simulador

se controla tanto un umbral angular θ_{thr} como un umbral de velocidad $\dot{\theta}_{\text{thr}}$, expresados en grados:

$$|\theta| < \theta_{\text{thr}}, \qquad |\dot{\theta}| < \dot{\theta}_{\text{thr}}, \tag{43}$$

durante un número mínimo de iteraciones consecutivas equivalente a un tiempo de permanencia `dwell_s`, de manera análoga al parámetro `stable_needed` del brazo. En el caso de PID/LQR se fija $\theta_{\text{thr}} =$ `thr_th_deg_pid_lqr`, mientras que en el modo difuso se emplea una tolerancia exclusiva $\theta_{\text{thr}} =$ `tol_deg_fuz`.

Este criterio define una ventana angular alrededor de la posición vertical que desempeña un papel equivalente a la tolerancia en espacio cartesiano utilizada para el brazo, pero adaptada al espacio de estados del péndulo invertido.

10.2.2. Aplicación de PID y LQR

La formulación general del controlador PID ya fue presentada previamente (ec. (10)), donde la señal de control se obtiene a partir del error, su integral y su derivada. En el péndulo, el error se define como:

$$e(t) = \theta(t) - 0,$$

ya que el objetivo es mantener el ángulo en la vertical. La ley de control implementada en el código es completamente análoga a la del brazo, pero se aplica directamente sobre el torque:

$$u_{\text{cmd}}(t) \;=\; -\big(K_p\, e(t) + K_d\, \dot{\theta}(t) + I_{\text{err}}(t)\big), \tag{44}$$

donde I_{err} corresponde al término integral. El *anti-windup* se implementa de la misma forma que en el brazo: la actualización de la integral se detiene cuando el torque se encuentra saturado y la acción integral empuja en la misma dirección que la saturación, tal como se describe en la Sección 9.3.1.

Para el controlador LQR, se reutiliza la función de costo cuadrática introducida en (25):

$$J \;=\; \sum \big(x^{\mathsf{T}} Q x + u^{\mathsf{T}} R u\big),$$

y la ley de realimentación de estados:

$$u_{\text{cmd}} \;=\; -Kx,$$

con K obtenido de la ecuación de Riccati discreta (ec. (27)). La diferencia con el brazo radica en la dimensionalidad y la interpretación de los pesos: aquí se emplea

$$Q = \text{diag}(Q_{11}, Q_{22}), \qquad R = [R],$$

donde Q_{11} pondera el error angular, Q_{22} pondera la velocidad angular y R penaliza el esfuerzo de control. Esta estructura se corresponde con la implementación de la función `lqr()` aplicada a las matrices A y B del modelo linealizado (42).

10.2.3. Control difuso Mamdani (diseño, reglas y operadores)

El controlador difuso Mamdani del péndulo comparte la misma arquitectura lógica que el empleado en el brazo robótico (Sección 9.3.2): se utilizan funciones de membresía triangulares para la salida, operadores AND/OR de tipo mínimo/máximo y defuzzificación por centroide o bisector, descritos en las ecuaciones (**??**) y (29)–(33). La diferencia principal radica en:

- Las entradas del FIS son únicamente el ángulo θ y la velocidad angular $\dot{\theta}$, en lugar de los tres errores cartesianos E_x, E_y, E_z.
- La salida del FIS es un torque base u_{fuz} acotado al intervalo $[u_{\text{mín}}, u_{\text{máx}}] = [-u_{\text{máx}}, u_{\text{máx}}]$.
- La ganancia global K_o ajusta la intensidad de la acción difusa, con límites $[K_{\text{mín}}, K_{\text{máx}}]$ que restringen el rango efectivo de la escala.

En el código, la evaluación difusa se realiza mediante una implementación manual equivalente al esquema Mamdani descrito en las ecuaciones (31)–(33):

$$u_{\text{fuz}} = \mathcal{F}_{\text{Mamdani}}(\theta, \dot{\theta}), \quad u_{\text{cmd}} = \text{sat}\big(K_{\text{eff}}\, u_{\text{fuz}}\big), \tag{45}$$

donde $K_{\text{eff}} = \text{clip}(K_o, K_{\text{mín}}, K_{\text{máx}})$ y $\text{sat}(\cdot)$ representa la saturación en $\pm u_{\text{máx}}$.

La base de reglas utiliza las mismas etiquetas lingüísticas que en el brazo (NG, NM, Z, PM, PG) y se almacena como una matriz de índices en el campo `fis.ruleIdx`. De forma análoga al Cuadro 4, cada entrada de dicha matriz representa una regla del tipo:

$$\text{Si } \theta \text{ es } L_\theta \text{ y } \dot{\theta} \text{ es } L_{\dot{\theta}} \Rightarrow u \text{ es } L_u.$$

El criterio de convergencia para el modo difuso utiliza un umbral exclusivo `tol_deg_fuz`, de forma coherente con la idea de que el controlador Mamdani reduce la acción de control en la región central, igual que en el brazo cuando la ganancia difusa se aproxima a $K_{\text{mín}}$ (ec. (**??**)).

10.3. Implementación en MATLAB

10.3.1. Interfaz gráfica y visualización 3D

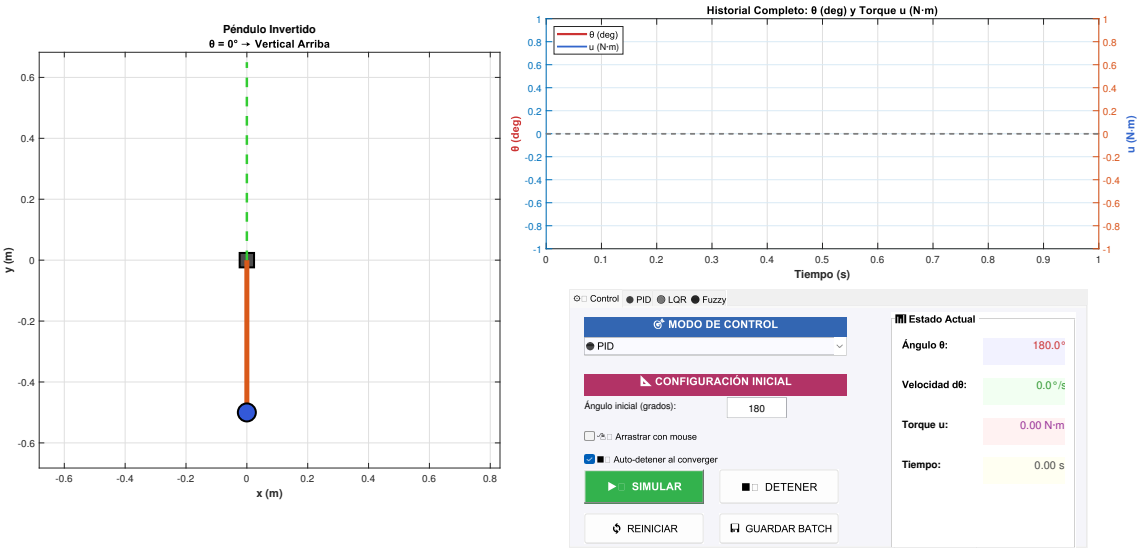
La interfaz gráfica del péndulo reaprovecha el mismo estilo de diseño que el simulador del brazo robótico (Sección 9.4.1): se emplean tabs para los parámetros de control, una zona de visualización principal y una gráfica en tiempo real. Las diferencias más relevantes son:

- La escena gráfica muestra el péndulo en el plano xy, análogo a la representación lateral del brazo, con el origen en el punto de giro y la vertical marcada como referencia visual.
- La gráfica temporal presenta simultáneamente el ángulo θ (en grados) y el torque u aplicado, utilizando dos ejes y como en el simulador del brazo para x_e, y_e, z_e.

- El panel de estado resume en tiempo real θ, $\dot{\theta}$, el torque u y el tiempo t, con códigos de color que indican la cercanía a la región de tolerancia.
- La GUI incluye un botón de "*Guardar batch*" que lanza múltiples corridas automáticas con condiciones iniciales aleatorias, pero a diferencia del brazo, con métricas definidas sobre el ángulo.

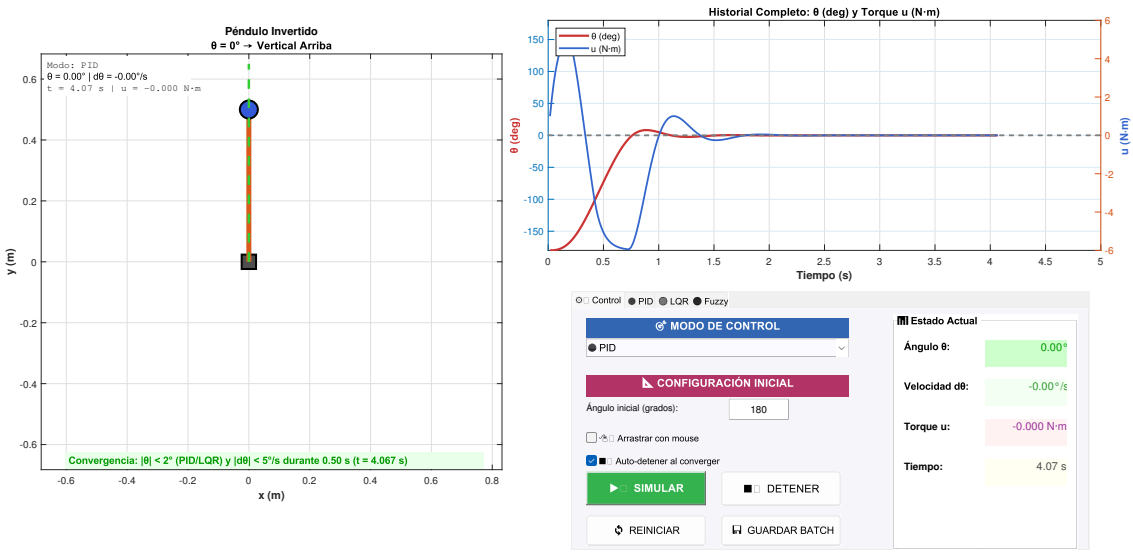
La Figura 25 muestra la interfaz del simulador en su estado inicial, antes de ejecutar cualquier corrida. La Figura 26, en cambio, ilustra el comportamiento del sistema durante una simulación, donde el péndulo se estabiliza en la vertical y se visualiza el error angular junto con el torque en tiempo real.

Figura 25. Interfaz del simulador del péndulo invertido antes de iniciar la simulación



Nota. Vista general de la GUI en reposo, la referencia vertical y los parámetros de control disponibles. En esta etapa no se ha ejecutado ninguna corrida. Elaboración propia.

Figura 26. Simulación en ejecución: estabilización del péndulo y gráfica en tiempo real



Nota. Ejemplo de una corrida con el controlador PID, donde el péndulo asciende hacia la vertical y se mantiene estable. Se observan simultáneamente el ángulo $\theta(t)$ y el torque $u(t)$ en la gráfica temporal, junto con el panel de estado actualizado. Elaboración propia.

La función `pendulo_pid_lqr_fuzzy()` agrupa tanto la lógica de la interfaz como la del lazo de control, siguiendo la misma organización modular que el simulador del brazo y reutilizando los mismos conceptos de timer, actualización periódica y reducción de datos para la gráfica en vivo.

10.4. Resultados de simulación y comparación de controladores

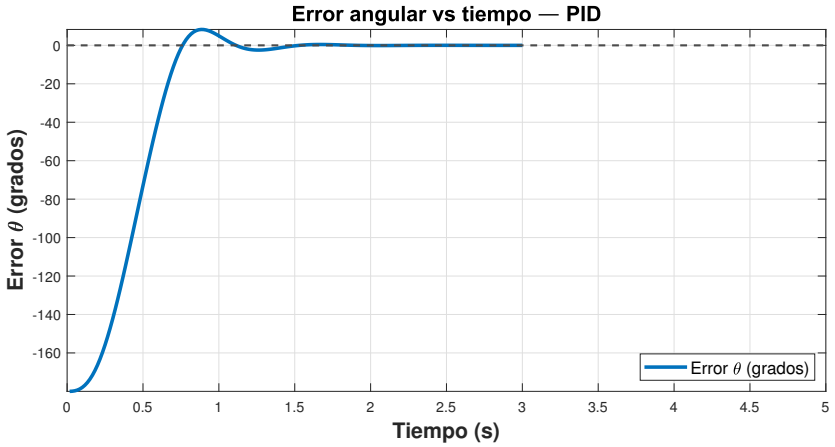
El análisis comparativo entre PID, LQR y Mamdani en el péndulo sigue la misma filosofía que en el brazo (Sección 9.5.3). A partir de un conjunto de condiciones iniciales θ_0 distribuidas en el intervalo $[-170°, 170°]$, el simulador ejecuta corridas para cada controlador y registra:

- El tiempo de convergencia según el criterio de la ecuación (43).
- La dinámica del error angular $\theta(t)$ hasta alcanzar la región de tolerancia.
- La tasa de corridas que convergen dentro del tiempo máximo asignado a cada modo (PID, LQR, Fuzzy-centroide, Fuzzy-bisector).

Las Figuras 27–30 muestran ejemplos individuales de la evolución del error angular

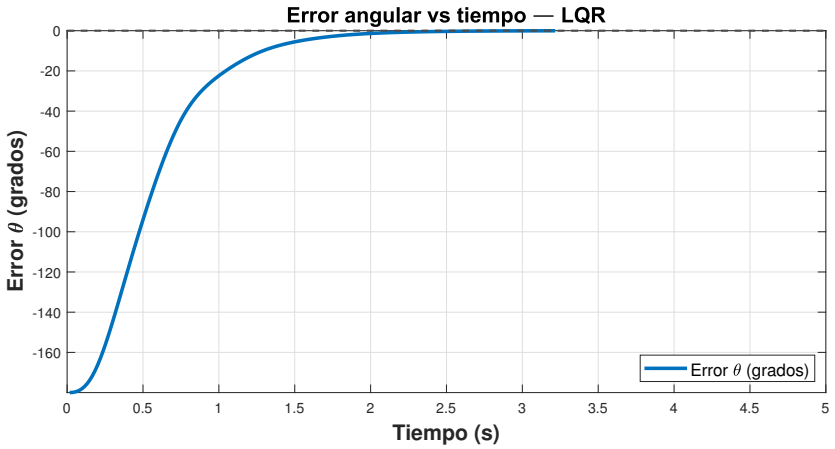
del péndulo invertido bajo los cuatro métodos evaluados. Cada gráfica corresponde a una corrida típica iniciando desde un ángulo negativo alto (cercano a $-180°$ según la convención del simulador), y permite visualizar las características principales de la fase transitoria y el proceso de estabilización para cada controlador.

Figura 27. Respuesta del péndulo invertido con control PID



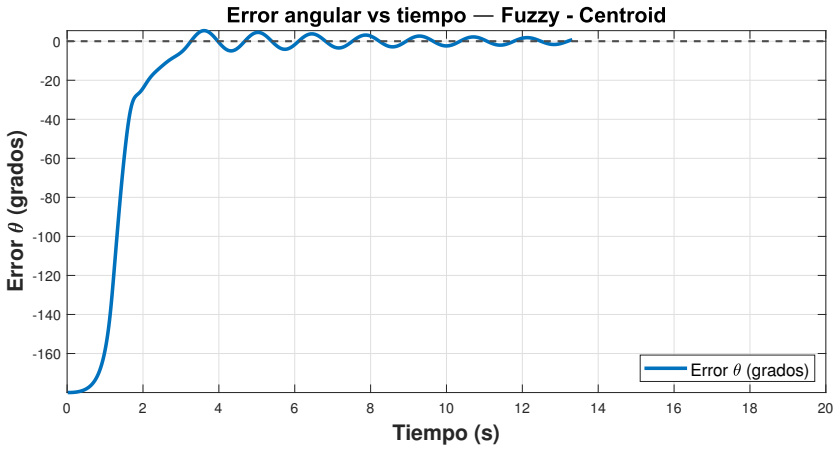
Nota. El controlador PID asciende rápidamente desde la posición inicial negativa, presenta un sobreimpulso pequeño y alcanza la vertical con una estabilización cercana a los 4.07 s. Elaboración propia.

Figura 28. Respuesta del péndulo invertido con control LQR



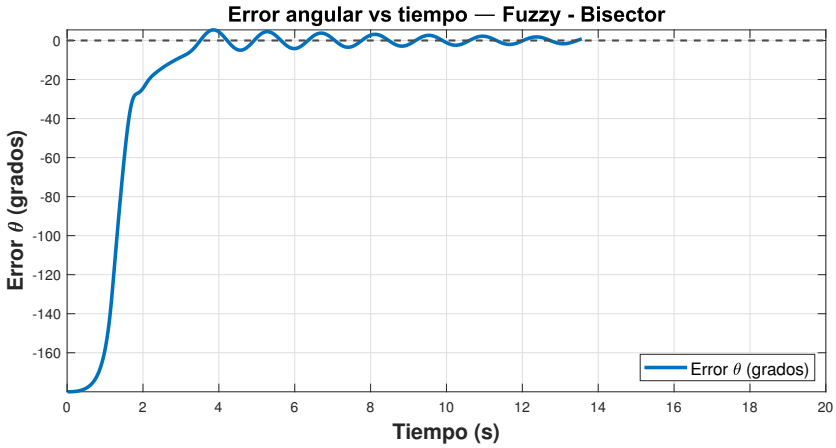
Nota. El LQR produce una transición más suave que el PID, sin sobreimpulsos ni oscilaciones pronunciadas, alcanzando la referencia aproximadamente en 4.40 s. Elaboración propia.

Figura 29. Respuesta difusa Mamdani con defuzzificación por centroide



Nota. El método centroide estabiliza el péndulo en la posición deseada, pero con oscilaciones persistentes alrededor de 0°. La convergencia ocurre cerca de 17.9 s, reflejando el efecto no lineal del sistema de reglas lingüísticas. Elaboración propia.

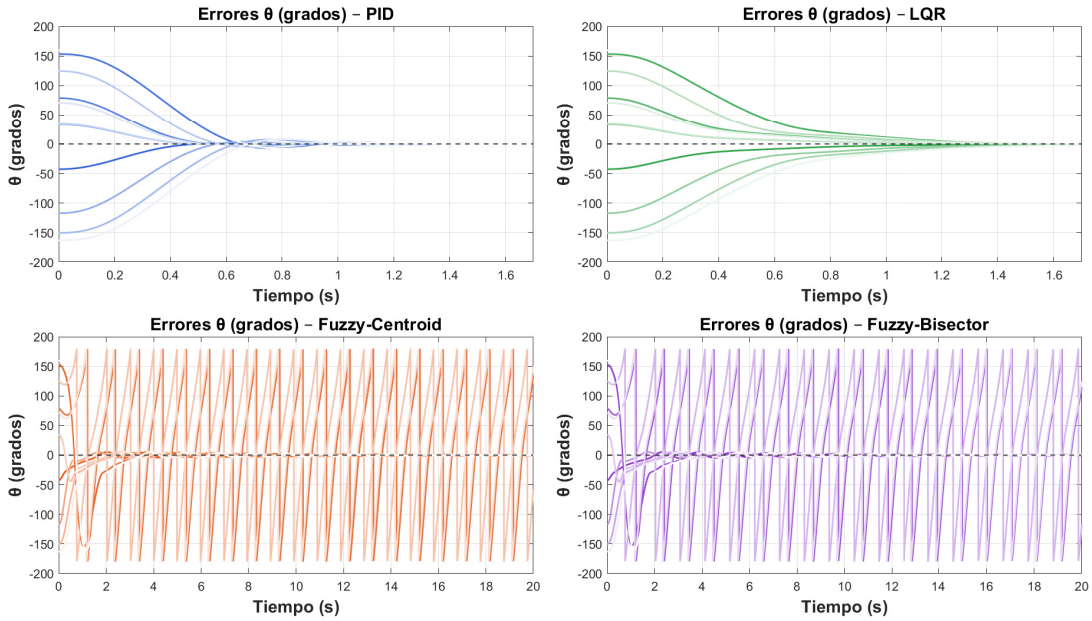
Figura 30. Respuesta difusa Mamdani con defuzzificación por bisector



Nota. El bisector también logra llevar el péndulo hacia 0°, pero con oscilaciones más prolongadas y una estabilización más tardía, cercana a 18.15 s, en comparación con el centroide. Elaboración propia.

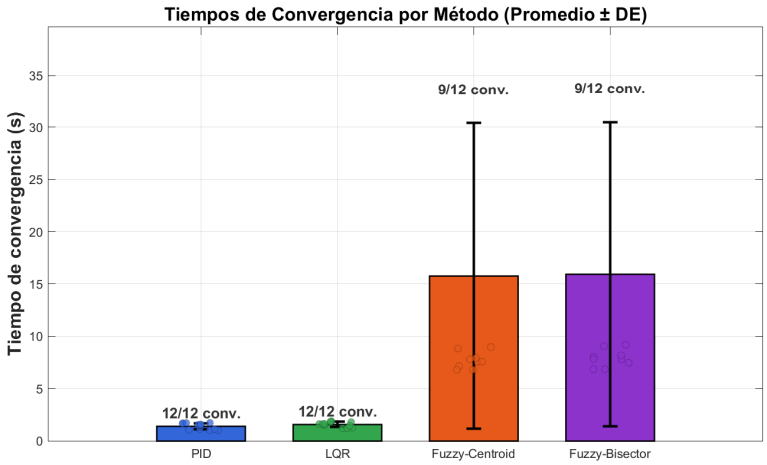
La Figura 31 muestra ejemplos de trayectorias de error angular para cada método de control, mientras que la Figura 32 resume los tiempos de convergencia promedio y su dispersión.

Figura 31. Trayectorias de error angular $\theta(t)$ para PID, LQR, Fuzzy-centroide y Fuzzy-bisector



Nota. En el péndulo invertido la posición deseada corresponde a la vertical hacia arriba, es decir, $\theta_{\text{ref}} = 0°$. Por esta razón, el error angular coincide directamente con el propio ángulo respecto a la vertical, de modo que las trayectorias $\theta(t)$ mostradas representan también la evolución del ángulo del sistema. Se superponen múltiples trayectorias para distintas condiciones iniciales y se utilizan colores distintos para diferenciar cada una de ellas. Esto permite comparar el comportamiento transitorio de los controladores y visualizar la rapidez con la que cada método conduce el péndulo hacia la región de tolerancia alrededor de $\theta = 0°$. Elaboración propia.

Figura 32. Tiempos de convergencia por método (promedio ± desviación estándar)



Nota. Resumen estadístico de los tiempos de convergencia para los cuatro controladores evaluados. Cada barra muestra el promedio sobre las corridas simuladas, junto con la desviación estándar y el número de casos que lograron estabilizarse dentro del tiempo máximo asignado. Elaboración propia.

10.5. Discusión de resultados

En las trayectorias mostradas en la Fig. 31 se observa que los controladores PID y LQR estabilizan el péndulo de forma rápida y consistente. En todas las condiciones iniciales evaluadas, ambos métodos reducen el error $\theta(t)$ hacia $0°$ sin oscilaciones significativas, alcanzando la región de tolerancia en menos de 2 s. En contraste, los controladores difusos (centroide y bisector) presentan respuestas altamente oscilatorias que permanecen alrededor de $\pm 180°$ durante la mayor parte de los 20 s de simulación, sin evidenciar una reducción clara de amplitud en la mayoría de las trayectorias.

La Fig. 32 confirma estas diferencias: PID y LQR convergen en todas las corridas (12/12), con tiempos muy bajos y consistentes. En cambio, los métodos difusos solo convergen en 9/12 casos y lo hacen con tiempos promedios cercanos a 15–16 s, acompañados de una alta dispersión. En este conjunto de pruebas, los controladores clásicos presentan un desempeño marcadamente superior tanto en rapidez como en fiabilidad, mientras que los controladores difusos muestran respuestas tardías y oscilatorias.

CAPÍTULO 11

Conclusiones

- Los escenarios simples permitieron validar el funcionamiento del sistema de inferencia difuso y ajustar adecuadamente sus reglas antes de aplicarlo en simulaciones más complejas, lo que facilitó un diseño progresivo y consistente.
- El haber tenido 3 sistemas representativos permitió analizar el desempeño de cada enfoque bajo distintas dinámicas y exigencias.
- Los resultados mostraron que cada método presenta fortalezas distintas: en general el PID ofrece una estabilización rápida con leve sobreimpulso, LQR proporciona transiciones más suaves y sin oscilaciones, mientras que las variantes difusas logran estabilidad con tiempos mayores pero con una respuesta más interpretable y coherente con sus reglas lingüísticas.
- La evaluación comparativa mostró que el controlador difuso Mamdani constituye una alternativa no lineal e interpretativa basada en reglas, aunque sin superar en tiempo de respuesta a los controladores PID y LQR.
- Los simuladores desarrollados del péndulo invertido y del brazo robótico 3D confirmaron la aplicabilidad práctica de la lógica difusa en sistemas mecatrónicos reales, aportando herramientas visuales e interactivas que permiten analizar el comportamiento de cada método de control bajo un marco homogéneo.

CAPÍTULO 12

Recomendaciones

- Implementar los controladores desarrollados en prototipos físicos reales para validar su desempeño fuera del entorno simulado, definiendo un protocolo experimental claro (perturbaciones, ruido, saturación y comparación métrica).
- Incorporar técnicas de optimización automática —como lógica difusa adaptativa, algoritmos genéticos o métodos de aprendizaje automático— para ajustar funciones de pertenencia y reglas, con el fin de mejorar la eficiencia y robustez del controlador difuso.
- Extender el estudio hacia sistemas mecatrónicos con mayor número de grados de libertad (por ejemplo, manipuladores redundantes o robots móviles), siguiendo una metodología documentada paso a paso para que otros puedan replicar el diseño de variables lingüísticas, reglas y validación.
- Integrar una interfaz de monitoreo en tiempo real que permita ajustar parámetros, visualizar métricas y comparar directamente el comportamiento de los controladores durante pruebas experimentales, facilitando iteraciones más rápidas entre diseño y validación.

CAPÍTULO 13

Referencias

[1] D. Guzmán y V. M. Castaño, «La lógica difusa en ingeniería: principios, aplicaciones y futuro,» *Ciencia y Tecnología*, vol. 24, n.º 2, págs. 87-107, 2006.

[2] O. G. Duarte, «Aplicaciones de la Lógica Difusa,» *Ingeniería e Investigación*, 2002.

[3] A. Mohebbi, S. Achiche y L. Baron, «A Fuzzy-based Framework to Support Multi-criteria Design of Mechatronic Systems,» en *2012 Annual Conference of the IEEE Industrial Electronics Society (IECON)*, IEEE, 2012, págs. 4935-4940. DOI: 10.1109/IECON.2012.6389039.

[4] J. P. Petion, *Evaluación e implementación de algoritmos genéticos para aplicaciones en robótica y otras áreas de Ingeniería Mecatrónica*, Trabajo de graduación de licenciatura, Universidad del Valle de Guatemala, 2024.

[5] J. G. Valenzuela Hernández, O. D. Montoya Giraldo y D. Giraldo Buitrago, «Lógica difusa aplicada al control local del péndulo invertido con rueda de reacción,» *Scientia et Technica*, vol. 18, n.º 4, págs. 623-632, 2013, ISSN: 0122-1701.

[6] V. D. Correa-Ramírez, D. Giraldo-Buitrago y A. Escobar-Mejía, «Control difuso del péndulo invertido con rueda de reacción usando seguimiento de trayectoria,» *Tecno-Lógicas*, vol. 20, n.º 39, págs. 39-57, mayo de 2017, ISSN: 0123-7799. dirección: https://doi.org/10.22430/22565337.693.

[7] J. Díaz-Salgado y D. S. Pichardo-Cruz, «Comparación de un algoritmo de control moderno y uno difuso en un péndulo invertido rotacional,» en *Memorias del Congreso Nacional de Control Automático (CNCA 2013)*, Ensenada, Baja California, México, 2013.

[8] A. A. Ramírez Suárez, E. K. Pérez Segura y J. H. Barrón Zambrano, «Elaboración de un robot tipo péndulo invertido e implementación de controlador difuso en base a cambio de posición angular,» en *Memorias del Congreso Nacional de Inteligencia Artificial (CONNAI 2015)*, Ciudad Victoria, Tamaulipas, México, 2015, págs. 101-104.

[9] L. A. Zadeh, «Fuzzy Sets,» *Information and Control*, vol. 8, n.º 3, págs. 338-353, 1965. DOI: 10.1016/S0019-9958(65)90241-X.

[10] L. A. Zadeh, «Fuzzy Sets,» *Information and Control*, vol. 8, n.º 3, págs. 338-353, 1965. DOI: 10.1016/S0019-9958(65)90241-X.

[11] E. H. Mamdani y S. Assilian, «An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller,» *International Journal of Man-Machine Studies*, vol. 7, n.º 1, págs. 1-13, 1975. DOI: 10.1016/S0020-7373(75)80002-2.

[12] T. J. Ross, *Fuzzy Logic with Engineering Applications*, 3rd. Chichester, UK: John Wiley & Sons, 2010, ISBN: 9780470743768.

[13] OpenAI, *ChatGPT (GPT-5)*, https://chat.openai.com/, [Herramienta de IA conversacional], 2025.

[14] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, 2nd. Wiley, 2007, ISBN: 978-0-470-03562-7.

[15] W. Pedrycz y F. Gomide, *Fuzzy Systems Engineering: Toward Human-Centric Computing*. Hoboken, NJ: John Wiley & Sons, 2007, ISBN: 9780471781264.

[16] T. J. Ross, *Fuzzy Logic with Engineering Applications*, 3rd. Chichester, UK: John Wiley & Sons, 2010, ISBN: 9780470743768.

[17] M. Orjuela y A. Chica Leal, «Diseño de una herramienta para identificación inteligente de sistemas de control,» ago. de 2007.

[18] G. J. Klir y B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Upper Saddle River, NJ: Prentice Hall, 1995.

[19] T. J. Ross, *Fuzzy Logic with Engineering Applications*, 4.ª ed. Chichester, UK: John Wiley & Sons, 2016.

[20] D. Driankov, H. Hellendoorn y M. Reinfrank, *An Introduction to Fuzzy Control*. Springer, 1993.

[21] L. A. Zadeh, «Fuzzy logic: concepts, developments, and implementation,» *International Journal of General Systems*, vol. 17, n.º 2-3, págs. 129-138, 1990. DOI: 10.1080/03081079008935104.

[22] K. J. Åström y R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ: Princeton University Press, 2008.

[23] K. Ogata, *Modern Control Engineering*, 5.ª ed. Upper Saddle River, NJ: Prentice Hall, 2010.

[24] S. P. (CAIB), *4.- Principios de regulación*, https://sarreplec.caib.es/pluginfile.php/23754/mod_resource/content/2/GA_ATIA03_Contenidos_Web/4_principios_de_regulacin.html, Accedido: 25-11-2025, 2024.

[25] R. C. Dorf y R. H. Bishop, *Modern Control Systems*, 12.ª ed. Upper Saddle River, NJ: Pearson, 2011.

[26] J. G. Ziegler y N. B. Nichols, «Optimum Settings for Automatic Controllers,» *ASME Transactions*, vol. 64, n.º 11, págs. 759-768, 1942.

[27] M. Simulations, *PID Controller Explained | PID Explained*, *YouTube video*, Accedido: 25-11-2025, 2020. dirección: https://www.youtube.com/watch?v=Stm1NWVCzkM.

[28] R. E. Kalman, «Contributions to the Theory of Optimal Control,» *Boletín de la Sociedad Matemática Mexicana*, vol. 5, n.º 2, págs. 102-119, 1960.

[29] F. L. Lewis, D. Vrabie y V. L. Syrmos, *Optimal Control*, 3.ª ed. Hoboken, NJ: John Wiley & Sons, 2012.

[30] J. H. Bianchi, *Ejemplo Tutorial: Diseño en espacio de estados para un péndulo invertido*, https://www.eng.newcastle.edu.au/~jhb519/teaching/caut2/etc/InvPen/invSS.html, Accedido: 25-11-2025, 2004.

[31] K. Ventura Calla, *Laboratorio N°7 – Modelo Difuso Mamdani*, Presentación en Slideshare, Disponible en: https://www.slideshare.net/slideshow/laboratorio-n7modelo-difuso-mandanipdfpdf/253394544 (consultado el día XX de Mes de 2025), 2024.

[32] D. Driankov, H. Hellendoorn y M. Reinfrank, *An Introduction to Fuzzy Control*. Berlin, Heidelberg: Springer, 1993, ISBN: 9783540564014. DOI: 10.1007/978-3-642-58064-7.

[33] A. Mohebbi, S. Achiche y L. Baron, «A Fuzzy-based Framework to Support Multicriteria Design of Mechatronic Systems,» en *2012 Annual Conference of the IEEE Industrial Electronics Society (IECON)*, IEEE, 2012, págs. 4935-4940. DOI: 10.1109/IECON.2012.6389039.

[34] B. Siciliano, L. Sciavicco, L. Villani y G. Oriolo, *Robotics: Modelling, Planning and Control*. London: Springer, 2010, ISBN: 978-1849966344. DOI: 10.1007/978-1-84628-642-7.

[35] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 3.ª ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2005, ISBN: 978-0131236295.

[36] L. García, A. Rodríguez y J. Pérez, «Desarrollo de reglas difusas en aplicaciones de convertidores,» *Revista Iberoamericana de Automática e Informática Industrial*, vol. 17, n.º 3, págs. 225-234, 2020. DOI: 10.4995/riai.2020.12345.

CAPÍTULO 14

Anexos

14.1. Repositorios

- Código del brazo robótico 3D: `https://github.com/gusc85/Proyecto-de-Graduaci-n/blob/7d8bfd95bfb433fa8ed24b0162f55da69af46e57/brazo_3D.m`
- Código del péndulo invertido: `https://github.com/gusc85/Proyecto-de-Graduaci-n/blob/7d8bfd95bfb433fa8ed24b0162f55da69af46e57/pendulo.m`
- Código del controlador difuso: `https://github.com/gusc85/Proyecto-de-Graduaci-n/blob/7d8bfd95bfb433fa8ed24b0162f55da69af46e57/Controlador_Mamdani.m`

CAPÍTULO 15

Glosario

- *Aggregation* — Proceso que combina las salidas difusas de todas las reglas activas.
- *Antecedent evaluation* — Cálculo del grado de verdad de las premisas de una regla difusa.
- *Anti-windup* — Técnica que limita la acumulación del término integral en presencia de saturación del actuador.
- *Canonical flow* — Secuencia estándar del sistema Mamdani: fuzzificación, evaluación de reglas, agregación y defuzzificación.
- *Chattering* — Oscilación rápida de alta frecuencia en la señal de control debida a saturaciones o discontinuidades.
- *Crisp* — Valor determinista o exacto previo a su conversión en una representación difusa.
- *Damped Least Squares* — Método numérico usado para resolver cinemática inversa evitando inestabilidades cerca de singularidades mediante un término de amortiguamiento.
- *Defuzzification* — Conversión de una salida difusa agregada en un valor determinista.
- *Fuzzification* — Transformación de valores deterministas en grados de pertenencia difusos.
- *Fuzzy logic* — Paradigma de razonamiento que maneja incertidumbre mediante grados de pertenencia continuos entre 0 y 1.
- *Fuzzy set* — Conjunto caracterizado por una función de pertenencia que asigna un grado de inclusión entre 0 y 1.
- *Fuzzy System Toolbox* — Herramientas de MATLAB para diseño, análisis y simulación de sistemas difusos.

- *IAE* — Índice integral del error absoluto; mide el error acumulado en magnitud.
- *Implication* — Operación que recorta o escala la función del consecuente según la activación de la regla.
- *ISE* — Índice integral del error cuadrático; penaliza más fuertemente errores grandes.
- *ITAE* — Índice integral del error absoluto ponderado por tiempo; penaliza errores persistentes.
- *Membership function* — Función que asigna a cada valor su grado de pertenencia dentro de un conjunto difuso.
- *SISO* — *Single Input, Single Output*; sistema con una entrada y una salida.
- *Windup* — Acumulación excesiva del término integral del PID debido a saturación, que degrada la respuesta.
- *ZOH* — *Zero-Order Hold*; mecanismo que mantiene constante la señal digital entre periodos de muestreo.