Compound Pendulum Control System using Fuzzy Logic

Oriol Miró and Joan Caballero

MAI, UPC
Computational Intelligence, Practice 3
December 24, 2024

In this work, we design and implement a fuzzy logic controller to stabilize a compound pendulum system, using a Mandami-type Fuzzy Inference System and Simulink. We define membership functions for two input and one output variables, and explore the effect of increasing the number of membership functions for the latter. Simulation results demonstrate that the controller effectively stabilizes the pendulum at various reference angles, with the finer-grained controller offering substantially improved performance. Our findings suggest that while increased membership functions can improve control precision, both systems performed really well; the benefits must be weighed against the added complexity in a case by case manner.

1 Introduction

The compound pendulum is a nonlinear dynamic system that can model the movement of robotic arms and other mechanical structures. Controlling such systems is challenging due to their inherent oscillatory behavior and sensitivity to disturbances, and traditional control methods might not be sufficient for meeting the desired performance requirements without extensive tuning.

Fuzzy logic control offers an alternative, incorporating human expertise into the control strategy, allowing for robust handling of nonlinearities and uncertainties. The Mamdani Fuzzy Inference System is specially relevant for control applications where expert knowledge can be encoded in the form of linguistic rules.

In this report, we develop a Mamdani-type fuzzy controller to stabilize a compound pendulum system. First, we define appropriate membership functions for the input variables—*Error* and *ErrorDerivative*—and the output variable—*Thrust*—, and then construct a rule base for the control strategy.

To assess the controller's performance, we implement the system in Simulink and conduct simulations for different reference angles. We also analyze the effects of increasing the number of membership functions for the output variable, examining the trade-off between control precision and system complexity.

2 Design of the Fuzzy Controller

2.1 Mamdani Fuzzy Inference System

We will employ the Mamdani FIS for the controller design, as constrained by the assignment. It is particularly suitable for control problems where human expertise can be translated into control strategies.

An overview of the fuzzy controller is illustrated in Fig. 1. The system has two inputs: *Error* (the difference between the reference angle and the current angle) and *ErrorDerivative* (the rate of change of the error). The output is the *Thrust* to be applied to the pendulum.

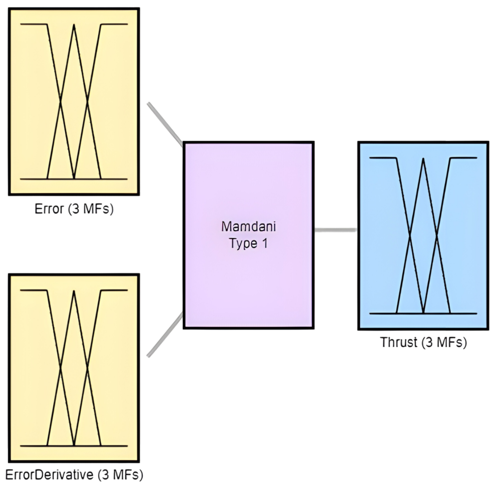

Figure 1: *Overall structure of the Mamdani Fuzzy Inference System*

2.2 System Implementation in Simulink

The fuzzy controller is implemented in Simulink as shown in Fig. 2. The model includes the compound pendulum dynamics, the fuzzy controller, and the necessary feedback loops.

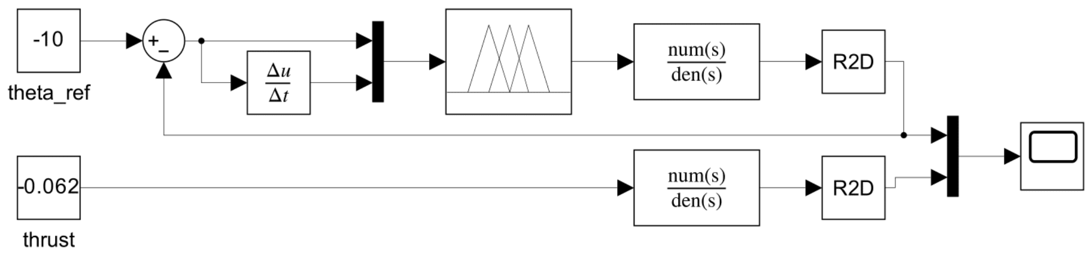

Figure 2: *Simulink model of the compound pendulum with fuzzy controller*

2.3 Membership Functions for the Variables

For the input variables *Error* and *ErrorDerivative*, we implemented them as per the assignment's guidelines, with a few exceptions to ensure the property of fuzzy partitions is satisfied (for any input value, the sum of the membership degrees across all partitions equals 1). Similarly, for the output variable *Thrust*, we applied the same tuning methodology to uphold this property.

Throughout, we used trapezoidal functions for the upper and lower membership functions, and triangular for the center one.

2.3.1 Input Variable: Error

The *Error* variable represents the deviation of the pendulum's current angle from the desired reference angle. It is defined as:

$$E = \theta_{\text{ref}} - \theta$$

We define three membership functions for *Error* over the range $[-80^\circ, 80^\circ]$:

1. **Negative (N)**: Represents a negative error when the pendulum is **above** the desired angle ($E < 0$).
2. **Zero (Z)**: Represents a small or zero error when the pendulum is near the desired angle ($E \approx 0$).
3. **Positive (P)**: Represents a positive error when the pendulum is **below** the desired angle ($E > 0$).

These membership functions are defined as shown in Fig. 3.

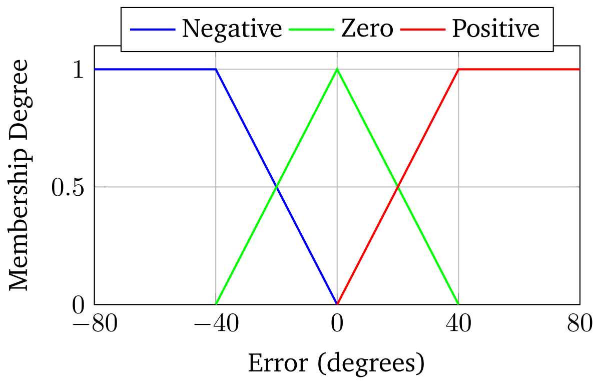

Figure 3: *Membership functions for Error*

2.3.2 Input Variable: Error Derivative

The *ErrorDerivative* variable indicates the rate at which the error is changing, defined as:

$$\dot{E} = \frac{dE}{dt}$$

It is defined over the range $[-5^\circ/\text{s}, 5^\circ/\text{s}]$. We use three membership functions:

1. **Decreasing (D)**: The error is decreasing ($\dot{E} < 0$), meaning the pendulum is moving **towards** the desired angle.
2. **Stationary (S)**: The error is not changing significantly ($\dot{E} \approx 0$), meaning the pendulum's position relative to the desired angle is stable.
3. **Increasing (I)**: The error is increasing ($\dot{E} > 0$), meaning the pendulum is moving **away** from the desired angle.

These membership functions are shown in Fig. 4.

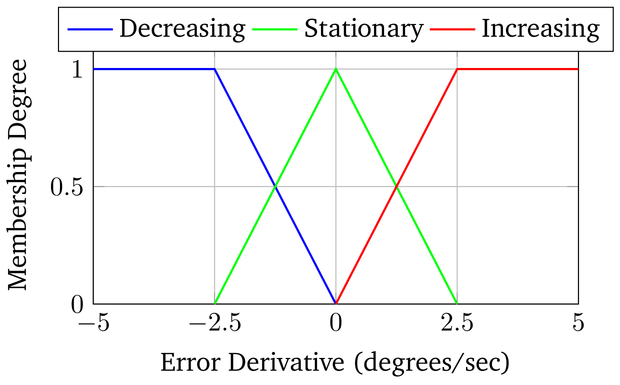

Figure 4: *Membership functions for ErrorDerivative*

2.3.3 Output Variable: Thrust

The *Thrust* variable is the output of the controller, indicating the corrective action needed. It ranges from -25 to 25 units. We define three membership functions:

1. **Negative Thrust (NT)**: Apply negative thrust ($T < 0$) to **decrease** the angle (lower the pendulum).
2. **Zero Thrust (ZE)**: Apply minimal or no thrust ($T \approx 0$) to maintain the current angle.
3. **Positive Thrust (PT)**: Apply positive thrust ($T > 0$) to **increase** the angle (raise the pendulum).

These functions are depicted in Fig. 5.

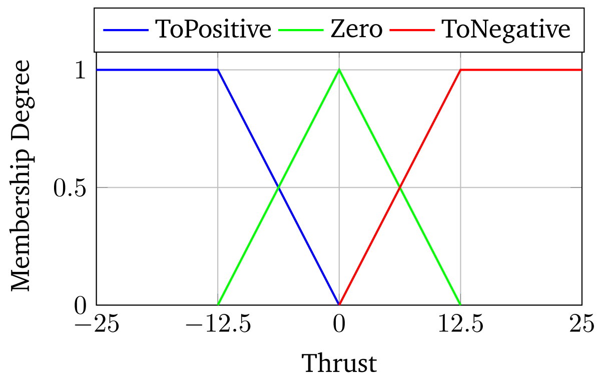

Figure 5: *Membership functions for Thrust*

2.4 Rule Base Definition

The interaction between the inputs and the output is defined by a set of IF-THEN rules. With two input variables, each having three membership functions, we have a total of $3 \times 3 = 9$ rules, as shown in Table 1.

Table 1: *Rule Base for the Fuzzy Controller*

Error	Error Derivative	Thrust
Positive (P)	Increasing (I)	Positive Thrust (PT)
Positive (P)	Stationary (S)	Positive Thrust (PT)
Positive (P)	Decreasing (D)	Zero Thrust (ZE)
Zero (Z)	Increasing (I)	Positive Thrust (PT)
Zero (Z)	Stationary (S)	Zero Thrust (ZE)
Zero (Z)	Decreasing (D)	Negative Thrust (NT)
Negative (N)	Increasing (I)	Zero Thrust (ZE)
Negative (N)	Stationary (S)	Negative Thrust (NT)
Negative (N)	Decreasing (D)	Negative Thrust (NT)

The fuzzy controller determines the appropriate thrust action based on the current *Error* and *Error Derivative*. When the pendulum is **below** the desired angle (Positive Error, P), if it is moving **away** from the desired angle (Error Derivative Increasing, I) or remains **stationary** (S), the controller applies **Positive Thrust** (PT) to raise the pendulum. If it is moving **toward** the desired angle (Error Derivative Decreasing, D), the controller maintains **Zero Thrust** (ZE) to allow natural upward movement without overshooting.

When the pendulum is **at** the desired angle (Zero Error, Z), if it starts moving **away** (Error Derivative Increasing, I), the controller applies **Positive Thrust** (PT) to maintain the position. If it remains **stable** (S), the controller applies **Zero Thrust** (ZE) to keep it steady. If it begins moving **past** the desired angle (Error Derivative Decreasing, D), the controller uses **Negative Thrust** (NT) to correct the upward drift.

When the pendulum is **above** the desired angle (Negative Error, N), if it is moving **toward** the desired angle (Error Derivative Increasing, I), the controller maintains **Zero Thrust** (ZE) to allow natural descent. If it is **stationary** (S) or moving **away** from the desired angle (Error Derivative Decreasing, D), the controller applies **Negative Thrust** (NT) to lower the pendulum toward the target position.

2.5 Control Surface Validation

We validate the fuzzy controller by examining the control surface generated by the FIS. It is a three-dimensional plot that shows how the output variable (*Thrust*) responds to different combinations of the input variables (*Error* and *Error Derivative*).

Figure 6 illustrates the control surface of the designed fuzzy controller. We can visually confirm that the controller behaves as expected across the entire range of input values.

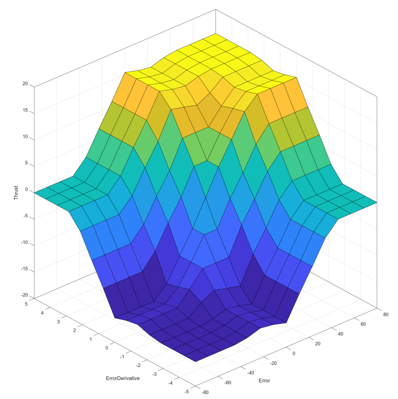

Figure 6: *Control surface of the fuzzy controller showing Thrust as a function of Error and Error Derivative.*

3 Simulation Results

We conducted simulations following the assignment constraints, with a stop time of 80 seconds and using the following parameters for the pendulum:

Bar length	$L = 0.495\,\text{m}$
Pivot to CG distance	$d = 0.023\,\text{m}$
Mass	$m = 0.43\,\text{kg}$
Moment of Inertia	$J = 0.0090\,\text{kg}\cdot\text{m}^2$
Viscous damping	$c = 0.00035\,\text{N}\cdot\text{m}\cdot\text{s/rad}$
Earth's gravity	$g = 9.80665\,\text{m/s}^2$

We then experiment with two different cases:

- $\theta_{\text{ref}} = 20^\circ$ and Thrust $= 0.123$
- $\theta_{\text{ref}} = -10^\circ$ and Thrust $= -0.062$

3.1 Results for $\theta_{\text{ref}} = 20^\circ$ and Thrust $= 0.123$

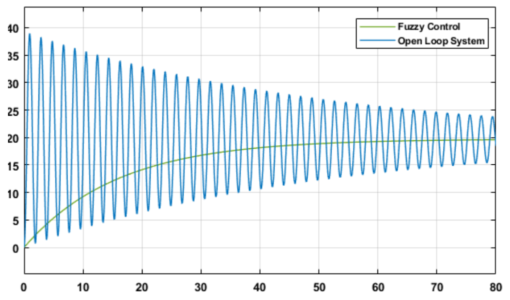

Figure 7: *System response for $\theta_{ref} = 20^\circ$ and Thrust $= 0.123$*

The simulation shows that the fuzzy controller effectively stabilizes the pendulum at the desired angle of 20°, in about 50 time steps. The settling time is significantly reduced compared to the open-loop response, and there is no overshoot.

3.2 Results for $\theta_{\text{ref}} = -10^\circ$ and Thrust $= -0.062$

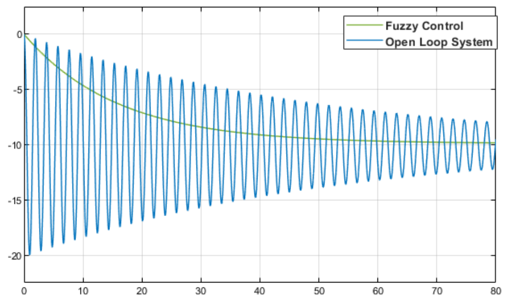

Figure 8: *System response for $\theta_{ref} = -10^\circ$ and Thrust $= -0.062$*

For the negative reference angle of -10°, the controller similarly brings the pendulum to the desired angle efficiently, once again in about 50 time steps. The negative thrust helps the pendulum to adjust its position downward, and the system demonstrates good stability and response time.

3.3 Discussion

The simulation results confirm that the designed fuzzy controller is capable of stabilizing the compound pendulum at both positive and negative reference angles. One notable thing is that we do not seem to *reach* the desired value at 80 time steps, but rather we approach it as a limit; perhaps it would be worth it to, in the future, consider other membership functions, in case we require more precise control (although is not "a very good approximation" the whole point of this course?)

4 Effect of Increasing the Number of Membership Functions

To assess the impact of increasing the granularity of the control actions, we modified the fuzzy controller by increasing the number of membership functions for the output variable *Thrust* from 3 to 7, once again ensuring the fuzzy partition property is satisfied

4.1 Updated Membership Functions for Thrust

We defined seven membership functions for the *Thrust* variable over the range $[-25, 25]$, ensuring they form a fuzzy partition. The membership functions are:

1. **Increase Big (IB)**
2. **Increase Medium (IM)**
3. **Increase Small (IS)**
4. **Zero (Z)**
5. **Decrease Small (DS)**
6. **Decrease Medium (DM)**
7. **Decrease Big (DB)**

Each membership function is defined using triangular functions, and adjacent functions cross at a membership degree of 0.5. The membership functions are illustrated in Figure 9.

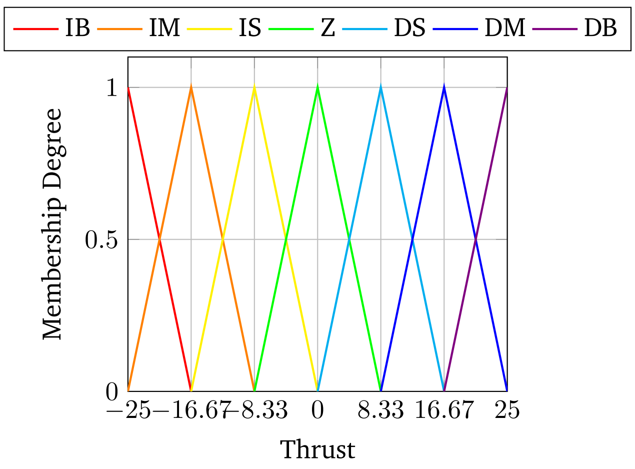

Figure 9: *Updated membership functions for Thrust with 7 membership functions*

4.2 Adjusted Rule Base

With the increased number of output membership functions, we refined the rule base to utilize the finer control actions. The input variables remain unchanged with three membership functions each, resulting in nine rules. The output assigned in each rule now corresponds to an appropriate level of thrust from the seven available options.

The updated rule base is presented in Table 2.

Table 2: *Updated Rule Base with 7 Output Membership Functions*

Error	Error Derivative	Thrust
Positive (P)	Increasing (I)	Increase Big (IB)
Positive (P)	Stationary (S)	Increase Medium (IM)
Positive (P)	Decreasing (D)	Increase Small (IS)
Zero (Z)	Increasing (I)	Increase Small (IS)
Zero (Z)	Stationary (S)	Zero (Z)
Zero (Z)	Decreasing (D)	Decrease Small (DS)
Negative (N)	Increasing (I)	Decrease Small (DS)
Negative (N)	Stationary (S)	Decrease Medium (DM)
Negative (N)	Decreasing (D)	Decrease Big (DB)

4.3 Simulation Results

We repeated the simulations using the same parameters and test cases to compare the performance of the controller.

4.3.1 Results for $\theta_{\text{ref}} = 20°$ and Thrust $= 0.123$

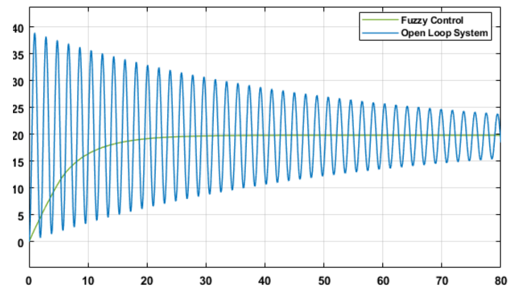

Figure 10: *System response for the first experiment with updated membership functions*

The controller with the updated membership functions shows an improved response, with a faster settling time (about 20 time steps).

4.3.2 Results for $\theta_{\text{ref}} = -10°$ and Thrust $= -0.062$

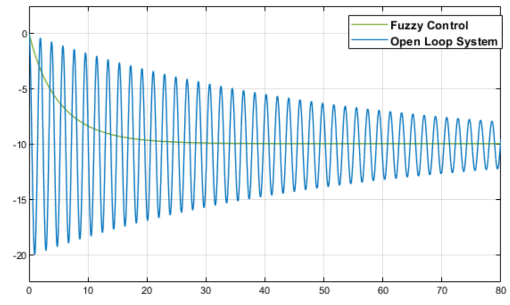

Figure 11: *System response for the second experiment with updated membership functions*

Likewise, the improved controller converges in about 20 time steps, faster than with the coarser system.

4.4 Discussion

Increasing the number of membership functions for the output variable enables the controller to make finer adjustments to the thrust, leading to faster convergence to the desired angle. In our simulations, the pendulum reached the target angle in about half the time steps with the more fine-grained controller; this improvement was significant, although one must consider the increased complexity.

Adding more membership functions increases the design complexity and computational load due to the additional rules and finer granularity required. Such fine control may be beneficial in domains requiring high precision and responsiveness, like aerospace systems or precision robotics, where even minor improvements can be key. However, for systems where such control is not necessary, the simpler controller is a better option.

5 Conclusions

We designed and implemented a Mamdani-type fuzzy controller to stabilize a compound pendulum system, using three membership functions for each variable and a rule base of nine rules. The controller effectively reduced oscillations and improved the transient response, achieving stability at desired angles both positive and negative.

We also investigated the performance of the system when increasing the number of membership functions for the output variable *Thrust* from 3 to 7. The fine-grained controller provided significantly faster convergence (about twice as fast); however, both systems showed acceptable performance, and the added complexity should only be used if it is needed by the specific application.

Future work could experiment and better balance the number of membership functions, as well as experiment with more complex problems and systems.