Computational Intelligence: FUZZY SYSTEMS

Joan Llop Palao - Jordi Armengol Estapé

1. Description of the membership functions for the output and rules

Membership functions **Implemented in** `pendulum_controlle.fis` and `compound_pendulum.slx`. Given a compound pendulum, the error of the angle and the derivative of the error, we want to develop a fuzzy system that controls the thrust of the propeller. The membership functions for the output variable correspond to the thrust being *Positive* (ie. counterclockwise thrust), *Neutral* (ie. no thrust) and *Negative* (ie. clockwise thrust). There are multiple options regarding the shape of the membership functions (triangular, gaussian, ...). We could not find any differences between the surfaces generated by Gaussian membership functions and triangular membership functions. Thus, we use the latter since they are the default option. We considered different options with regard to the triangles form, but the option represented in figure 1 generated a very smooth surface (combined with most of the defuzzification functions). To choose a defuzzification function we plotted them in figure 2 and observed that centroids matched the idea on how our function should be.

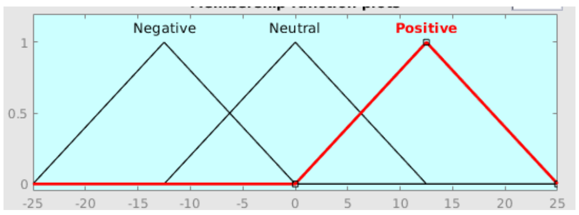


Figure 1: Output membership functions

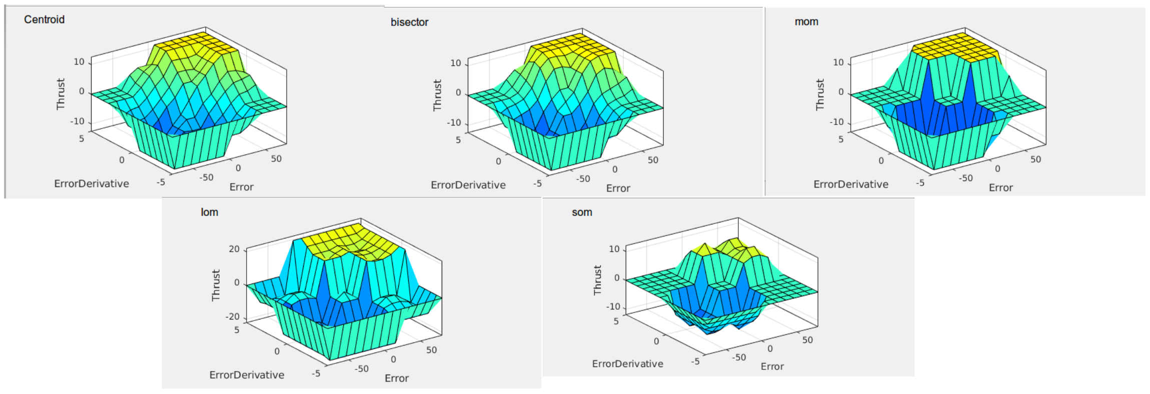


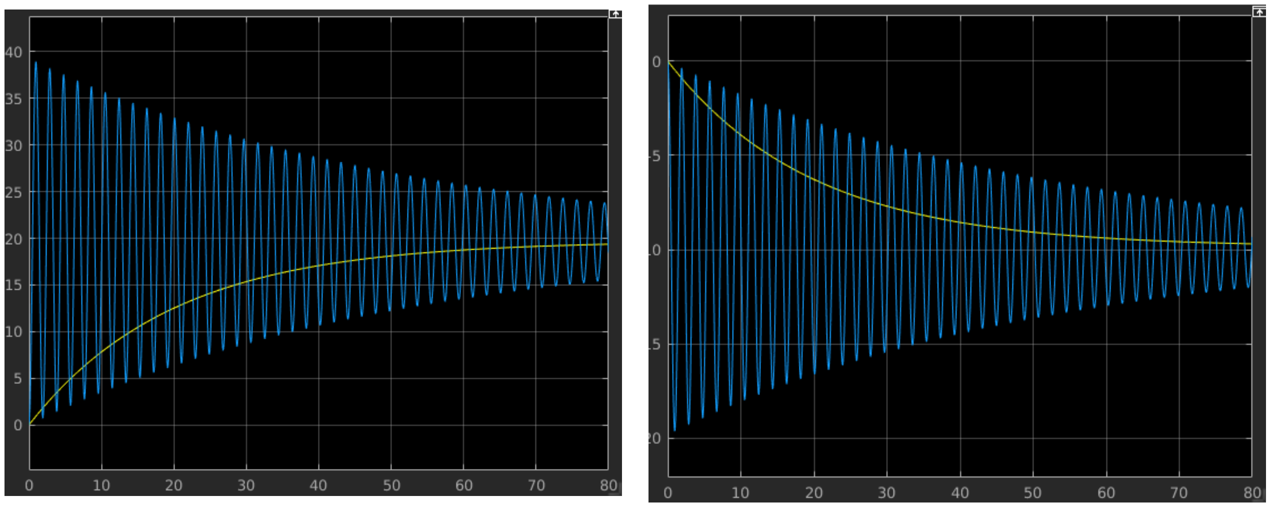
Figure 2: Different surfaces with different defuzzification functions

Rules The rules that are implemented in `pendulum_controller.fis`. To justify them, we identify three cases: 1. **The error is worsening** (negative error and negative derivative, positive error and positive derivative, or zero error and positive or negative derivative): We have to change the direction (the thrust will be positive when the error is positive or when the error is zero and the derivative is positive and negative when the error is negative or zero and the derivative is negative). 2. **The error is static (zero derivative)**: If the error is positive or negative we have to decrease it in the same way as when the error is worsening. When the error is zero we do nothing because we already have the ideal angle. 3. **The error is getting better**: We do nothing because we are getting closer to the ideal angle. We have considered to push the pendulum in order to keep improving the error in a quicker manner, but we discarded the option because we could cause oscillations when the ideal angle is close to the real one. From this reasoning, we derive the **9** rules:

Rule	**Condition**		**Thrust**
	Error	Deriv	
1	Neg	Decr	Neg
2	Neg	Stat	Neg
3	Neg	Incr	Neu
4	Zer	Decr	Neg
5	Zer	Stat	Neu
6	Zer	Incr	Pos
7	Pos	Decr	Neu
8	Pos	Stat	Pos
9	Pos	Incr	Pos

2. Plot of the results with a simulation stop time of 80

As we can observe in the plots 3a and 3b, the rules we have defined have successfully stopped the pendulum from oscillating. In both cases (positive and negative ideal angle) our fuzzy controller has shown a neat approximation to the ideal angle and a quicker stabilization than the original.



(a) Output with $theta\_ref = 20$ and $thrust = 0.123$ (b) Output with $theta\_ref = -10$ and $thrust = -0.062$

Figure 3: The pendulum oscillation (blue) and the pendulum stabilized with our system (yellow)

3. What happens if we increase the number of membership functions of the output?

Implemented in `pendulum_controller_5_output.fis`. If we increase the number of mfs, the resulting surface is smoother, since we specify more fine-grained rules for the problem (eg. soft thrust if the error is non-zero but at least it is not getting worse). We have implemented an output function with 5 triangular mfs: very_negative, negative, neutral, positive and very_positive. The observed results are similar to the ones obtained with 3 mfs (see attached image in `img/result_5_outputs.png`)