Design Calculations: Fuzzy Scheduler Workflow

1 Objective / Purpose Statement

Design objective: maximize line-following speed while keeping lateral error stable and avoiding loss of the line. Constraint: motor speed is capped by hardware and the camera can only observe a limited lookahead length. Design decision: use a fuzzy scheduler that maps the current speed command and the visible line length into a speed cap and PID gains.

2 References

- "A Fuzzy Rule-Based Control System for Fast Line-Following Robots," DCOSS 2020.
- MTE 380 Detailed Design W2026 (design calculation template guidance).

3 Design Inputs

- $x_{j,k}$: commanded speed of motor j at iteration k.
- $|X|$: maximum motor speed capability (rpm).
- L_k: length of the currently detected line segment from the camera.
- $|L|$: maximum detectable line length for the camera (within the ROI).
- e: lateral error from IR array (weighted centroid error).
- Membership function breakpoints for X_1, X_2, and output labels (heuristic, tunable).

4 Assumptions

- The camera reliably segments the line in the bottom ROI under expected lighting.
- L_k is computed as the number of ROI rows containing line pixels and is proportional to visible line length.
- The control loop runs fast enough that e and L_k are representative of the current state.
- Integral gain K_i is set to 0 in the initial tuning to avoid wind-up; this can be revisited.

5 Signal Processing: Computing X_1 and X_2

The fuzzy scheduler inputs come from the IR array and the camera:

- **IR array (lateral error):** read the IR sensor values, compute the weighted centroid of the line, and subtract the array center to obtain lateral error e used by the PID controller. The IR array does not directly provide X_1; it provides the feedback error that the PID corrects.

- **Camera (visible line length):** threshold the bottom ROI of the frame to isolate the line, count the number of ROI rows that contain line pixels, and define

  $$L_k = \text{rows with line pixels}, \quad |L| = \text{ROI height}.$$

  Then $X_2 = \frac{L_k}{|L|} \times 100$ (line length percentage).

- **Speed command:** X_1 is the current forward speed command (or PWM command) normalized to percent of the maximum motor speed,

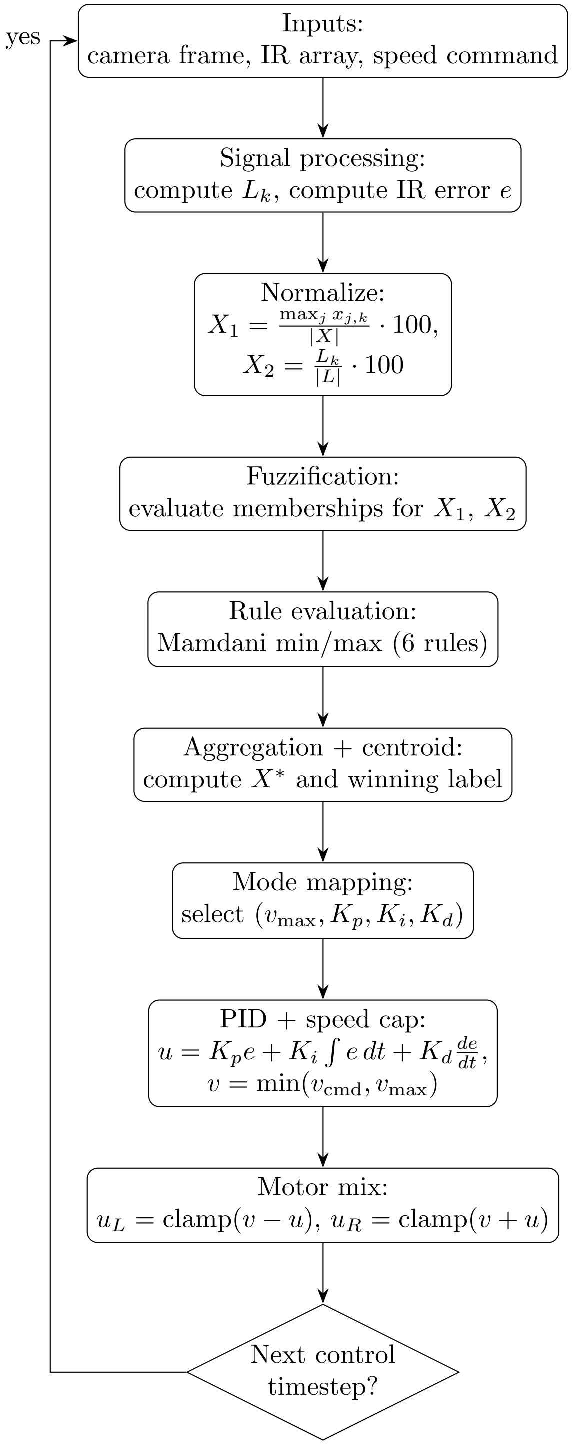
  $$X_1 = \frac{\max_j x_{j,k}}{|X|} \times 100.$$

  In the absence of encoders, the commanded speed is used as a proxy for actual speed.

6 Analytical Method and Computations

The fuzzy scheduler follows the DCOSS 2020 method. Two normalized inputs feed six rules; the aggregated output is defuzzified to a scalar X^* (range 1–100). The winning output label maps to a speed cap and PID gains for the downstream controller.

Fuzzy Process Flowchart


Inputs:
camera frame, IR array, speed command
Signal processing:
compute L_k, compute IR error e
Normalize:
$X_1 = \frac{\max_j x_{j,k}}{|X|} \cdot 100,$
$X_2 = \frac{L_k}{|L|} \cdot 100$
Fuzzification:
evaluate memberships for X_1, X_2
Rule evaluation:
Mamdani min/max (6 rules)
Aggregation + centroid:
compute X^* and winning label
Mode mapping:
select $(v_{\max}, K_p, K_i, K_d)$
PID + speed cap:
$u = K_p e + K_i \int e\,dt + K_d \frac{de}{dt},$
$v = \min(v_{\text{cmd}}, v_{\max})$
Motor mix:
$u_L = \text{clamp}(v - u)$, $u_R = \text{clamp}(v + u)$
Next control timestep?
yes

Input normalization

From the paper, the fuzzy inputs are normalized as:

$$X_1 = \frac{\max_j x_{j,k}}{|X|} \times 100, \quad j \in [1, \ldots, N] \tag{1}$$

$$X_2 = \frac{L_k}{|L|} \times 100 \tag{2}$$

In this repository:

- X_1: current forward speed command in $[0, 1]$ multiplied by 100.
- X_2: fraction of ROI rows containing line pixels ("len_pct" in `vision.py`) as a percentage.

Membership functions

Triangular membership function:

$$\mu_\triangle(x; a, b, c) = \begin{cases} 0, & x \leq a \text{ or } x \geq c \\ \dfrac{x-a}{b-a}, & a < x < b \\ \dfrac{c-x}{c-b}, & b \leq x < c \end{cases}$$

Input X_1 (motor speed %)

Label	Triangle (a, b, c)	Notes
Low	$(0, 0, 40)$	rises to 1 at 0, falls to 0 at 40
Medium	$(20, 50, 80)$	centered at 50
High	$(60, 100, 100)$	starts at 60, flat to 100

Input X_2 (line length %)

Label	Triangle (a, b, c)	Notes
Close	$(0, 0, 50)$	
Far	$(30, 100, 100)$	

Output sets (speed level) Labels: LC, LF, MC, MF, HC, HF with triangles:

$$\text{LC} : (0, 10, 25), \quad \text{LF} : (15, 30, 45), \quad \text{MC} : (35, 50, 65),$$
$$\text{MF} : (55, 70, 85), \quad \text{HC} : (70, 85, 95), \quad \text{HF} : (85, 100, 100).$$

Rule base (Table I)

For each input pair (X_1, X_2):

X_1	X_2	Output label
Low	Close	LC
Low	Far	LF
Medium	Close	MC
Medium	Far	MF
High	Close	HC
High	Far	HF

Rule activation uses fuzzy AND as $\min(\mu_{X_1}, \mu_{X_2})$. If multiple rules map to the same output label, fuzzy OR is max of activations (Mamdani max–min).

Aggregation and defuzzification

For each sample $x \in [0, 100]$ (101 evenly spaced points), compute:

$$\mu_{\text{agg}}(x) = \max_{L \in \{\text{labels}\}} \min\big(\mu_{\text{rule}}(L),\ \mu_L(x)\big).$$

Centroid defuzzification:

$$X^* = \frac{\sum_x \mu_{\text{agg}}(x)\, x}{\sum_x \mu_{\text{agg}}(x)}.$$

Quantize and clamp: $X^*_{\text{q}} = \min(100, \max(1, \text{round}(X^*, 2)))$.

7 Calculations

Mapping X^* to control parameters

The winning label (highest activation) selects a speed cap $v_{\max}$ and PID gains (K_p, K_i, K_d) from Table 1. The base speed is capped by $v_{\max}$ and the PID correction is applied to generate wheel commands:

$$u = K_p e + K_i \int e\, dt + K_d \frac{de}{dt}$$

$$u_L = \text{clamp}(v - u,\ -1,\ 1), \quad u_R = \text{clamp}(v + u,\ -1,\ 1)$$

Label	$v_{\max}$	K_p	K_i	K_d
LC	0.30	0.80	0.00	0.10
LF	0.35	0.70	0.00	0.10
MC	0.45	0.65	0.00	0.12
MF	0.55	0.55	0.00	0.14
HC	0.65	0.45	0.00	0.16
HF	0.75	0.40	0.00	0.18

Table 1: PID mapping used in this repo (heuristic), aligned with labels from the paper's Table I.

Worked example

Given $X_1 = 20$ and $X_2 = 30$:

1. $\mu_{\text{Low}}(20) = 0.5$, $\mu_{\text{Close}}(30) = 0.4$; other memberships are 0.

2. Rule Low$\wedge$Close $\rightarrow$ LC activates with $\min(0.5, 0.4) = 0.4$.

3. Only LC contributes; its triangle $(0, 10, 25)$ is clipped at 0.4, centroid over $x \in [0, 25]$ yields $X^* = 12.04$.

4. Label LC wins; select PID $(0.30, 0.80, 0.00, 0.10)$ and cap the base speed to $v_{\max} = 0.30$.

8 Results / Conclusions

The calculation converts measurable signals (X_1 speed and X_2 line length) into a discrete mode that determines a safe speed cap and PID gains. This satisfies the objective of maintaining stable tracking while permitting higher speeds when the visible line is long. The design decision is to use fuzzy scheduling instead of a fixed PID, with tunable membership functions and gain tables that can be refined on hardware.

9 Underlying Engineering Principle

The underlying engineering principle is **feedback control of a dynamic system**. The IR array provides the lateral error for closed-loop PID control, while the camera provides a lookahead measure of line length to adjust controller aggressiveness. The fuzzy system is a rule-based approximation of a nonlinear control law: it maps uncertain, noisy sensor inputs (X_1, X_2) into stable control parameters (speed cap and gains) using membership functions and defuzzification. This applies control theory concepts of stability, robustness, and sensor fusion rather than relying on a single fixed-gain controller.