African Journal of Advanced Pure and Applied Sciences (AJAPAS)
Online ISSN: 2957-644X
Volume 5, Issue 3, 2026
Page No: 376-386
Website: *https://aaasjournals.com/index.php/ajapas/index*
ISI 2025: 1.126 **SJIFactor 2024: 6.752** **معامل التأثير العربي: 1.62**

Fixed Versus Fuzzy Self-Tuning PID Speed Control of a Geared DC Motor: Experimental Identification and Robustness Evaluation

Munthir Mohammed Mahfouth [1], Hafed Efhejj [2*]
[1,2] Department of Automatic Control, College of Electronic Technology, Tripoli, Libya

التحكم في سرعة محرك تيار مستمر ذي علبة تروس باستخدام *PID* ثابت مقابل *PID* ضبابي ذاتي الضبط: التعريف التجريبي وتقييم المتانة

منذر محمد محفوظ [1]، حافظ افحيج [2*]
[2،1] قسم التحكم الآلي، كلية التقنية الإلكترونية، طرابلس، ليبيا

*Corresponding author: *hafedrs@gmail.com*

Received: May 26, 2026	Accepted: August 22, 2026	Published: September 07, 2026

Abstract:
Fixed-gain proportional-integral-derivative (PID) control is attractive for low-cost direct-current motor drives, but performance can degrade when load and plant parameters depart from the tuning condition. This paper compares a fixed PID controller with a zero-order Sugeno fuzzy self-tuning PID controller for speed regulation of a 12 V JGA25-370 geared motor. An Arduino Uno, L298N driver, and Hall-effect encoder were used to acquire a 100 ms open-loop step-response record. A filter-aware output-error identification procedure yielded the first-order model $G(s)=10.8826/(0.024254s+1)$ rpm/V, with a root-mean-square fit error of 0.1553 rpm and $R^2=0.999938$. Both controllers were evaluated in MATLAB/Simulink with identical sampling, encoder quantisation, measurement filtering, command saturation, and anti-windup assumptions. At the nominal 100 rpm operating point, both controllers achieved a 0.4 s rise time, while the fixed PID settled in 0.6 s rather than 0.8 s and produced lower integral error indices. Under an equivalent 2 V opposing disturbance, fuzzy scheduling reduced maximum speed deviation by 26.88%, recovery time by 40%, and post-event integral absolute error by 51.57%. Under parameter variation, the corresponding reductions were 28.32%, 40%, and 49.63%. Across 500 paired Monte Carlo trials, mean ITAE and settling time decreased by 17.94% and 13.54%, respectively. The fuzzy controller nevertheless increased control-signal total variation by 6.68 times. The results support fuzzy self-tuning as a robustness enhancement, not a universal replacement for a well-tuned fixed PID controller.

Keywords: Arduino Uno, DC motor, fuzzy self-tuning PID, Monte Carlo analysis, robustness, system identification.

الملخص
يُعد التحكم التناسبي-التكاملي-التفاضلي (PID) ذو المعاملات الثابتة خياراً جذاباً لمحركات التيار المستمر منخفضة التكلفة، إلا أن أداءه قد يتدهور عند تغير الحمل أو معاملات المنظومة عن شروط الضبط. تقارن هذه الدراسة بين متحكم PID ثابت ومتحكم PID ضبابي ذاتي الضبط من نوع Sugeno من الرتبة الصفرية لتنظيم سرعة محرك JGA25-370 بجهد 12 فولت وعلبة تروس. استُخدم Arduino Uno مع مشغل L298N ومشفّر Hall للحصول على استجابة خطوة في الحلقة المفتوحة بزمن أخذ عينات 100 ms. أسفرت عملية تعريف تعتمد على خطأ الخرج مع مراعاة المرشح عن النموذج من الدرجة الأولى $G(s)=10.8826/(0.024254s+1)$ rpm/V، بخطأ ملاءمة RMS مقداره 0.1553 rpm ومعامل تحديد $R^2=0.999938$. تم تقييم المتحكمين في MATLAB/Simulink تحت افتراضات متماثلة لأخذ العينات والتكميم والترشيح والتشبع ومنع التكامل الزائد. عند 100 rpm حقق المتحكمان زمن صعود 0.4 s، بينما استقر PID الثابت في 0.6 s مقابل 0.8 s للمتحكم الضبابي وحقق مؤشرات خطأ تكاملية أقل. تحت اضطراب مكافئ مقداره 2 V خفّض الضبط الضبابي أقصى انحراف في السرعة بنسبة 26.88% وزمن الاستعادة بنسبة 40% وIAE بعد الحدث

بنسبة 51.57%. وتحت تغير المعلمات بلغت التخفيضات المناظرة 28.32% و40% و49.63%. وعبر 500 تجربة Monte Carlo مقترنة انخفض متوسط ITAE وزمن الاستقرار بنسبة 17.94% و13.54% على التوالي، مع زيادة التغير الكلي لإشارة التحكم بمقدار 6.68 مرة. تدعم النتائج استخدام الضبط الضبابي الذاتي كوسيلة لتعزيز المتانة، وليس بديلاً عاماً عن PID ثابت مضبوط جيداً.

الكلمات المفتاحية: Arduino Uno، محرك تيار مستمر، PID ضبابي ذاتي الضبط، تحليل Monte Carlo، المتانة، تعريف المنظومة.

Introduction

Direct-current (DC) motors remain common actuators in automation, mobile robotics, laboratory equipment, and mechatronic systems because their armature voltage provides a direct and readily controlled influence on speed and torque. Conventional proportional-integral-derivative (PID) control is especially attractive in low-cost embedded systems because its three-term structure is transparent, computationally modest, and supported by mature tuning methods [1], [2].

A fixed-gain PID controller is normally tuned around a selected operating point. The resulting response can be satisfactory under nominal conditions but may deteriorate when the motor experiences load changes, driver nonlinearities, friction variation, supply reduction, or uncertainty in the effective plant gain and time constant. Fuzzy gain scheduling offers a practical adaptation mechanism: qualitative rules map the instantaneous tracking error and its change to online gain multipliers without replacing the familiar PID structure [3]-[5].

Many fuzzy-PID comparisons report only one or a few simulation responses. Such comparisons can be difficult to interpret when the two controllers do not share the same sampling, filtering, quantisation, saturation, or disturbance assumptions. Control effort is also often represented only by root-mean-square voltage or energy, which can conceal rapid command variation. A further distinction is required between numerical simulation evidence and qualitative operation on real hardware.

This work addresses these issues through a matched comparison on a plant identified from an Arduino-based open-loop experiment. A conventional PID controller and a zero-order Sugeno fuzzy self-tuning PID controller were implemented with common measurement and actuator constraints. Their performance was assessed at the nominal design point, under deterministic disturbance and parameter-variation tests, and through a 500-trial paired Monte Carlo study. The real motor was operated with both firmware implementations as a preliminary functional check; however, all numerical closed-loop comparisons reported here are MATLAB/Simulink results.

The contribution is an engineering comparison rather than a claim of a new fuzzy-control theory. Specifically, the work provides: (1) filter-aware identification of a low-cost geared motor from encoder data; (2) a reproducible fixed-versus-fuzzy comparison under identical sampled-data constraints; (3) paired uncertainty analysis using effect sizes, confidence intervals, and controller win rates; and (4) explicit reporting of the robustness-versus-control-variation trade-off.

Related Work

Zhao, Tomizuka, and Isaka established a fuzzy gain-scheduling framework in which operating information modifies PID gains online [4]. Woo, Chung, and Lin later showed that self-tuning scaling factors can improve the transient and steady-state behaviour of a PID-type fuzzy controller [5]. These studies provide the conceptual basis for supervisory fuzzy adaptation, but the selected gain ranges, input normalisation, membership functions, and rule consequents remain application-dependent.

For DC motor speed control, Messaadi and Amroun compared a fuzzy self-tuning PID controller with a conventional PID controller using MATLAB/Simulink and an Arduino implementation [6]. Prasoetphon et al. also compared fuzzy-PID and PID responses for a DC motor and considered different membership-function choices [7]. These studies support the feasibility of fuzzy adaptation, although their reported conclusions depend on controller tuning and evaluation conditions. In particular, a nominal response alone cannot establish robustness across uncertain plants.

Supriyono, Alanro, and Supardi developed an Arduino- and MATLAB-based DC motor trainer for practical PID education [8], while Aviles et al. described a broader open-source methodology linking modelling, controller design, and physical validation in control courses [9]. These platforms demonstrate the value of low-cost experimental integration, but they do not provide the paired fixed-versus-fuzzy uncertainty analysis used in the present study.

The present paper therefore focuses on evaluation discipline. Both controllers use the same identified plant, command saturation, encoder resolution, digital filtering, and sample period. Deterministic tests are supplemented by paired Monte Carlo trials, and robustness gains are interpreted together with steady-state

accuracy and control-signal variation. The literature base is adequate for a conference paper, but a journal submission would benefit from additional recent peer-reviewed embedded fuzzy-PID studies.

System Architecture and Methodology

Fig. 1 shows the closed-loop structure. The commanded reference speed is compared with encoder-based feedback. The selected controller produces a voltage command limited to 0-12 V, which is mapped to the pulse-width modulation input of an L298N H-bridge. The plant is a 12 V JGA25-370 permanent-magnet geared DC motor. The encoder signal is processed by an Arduino Uno at a controller sample period of 0.1 s.

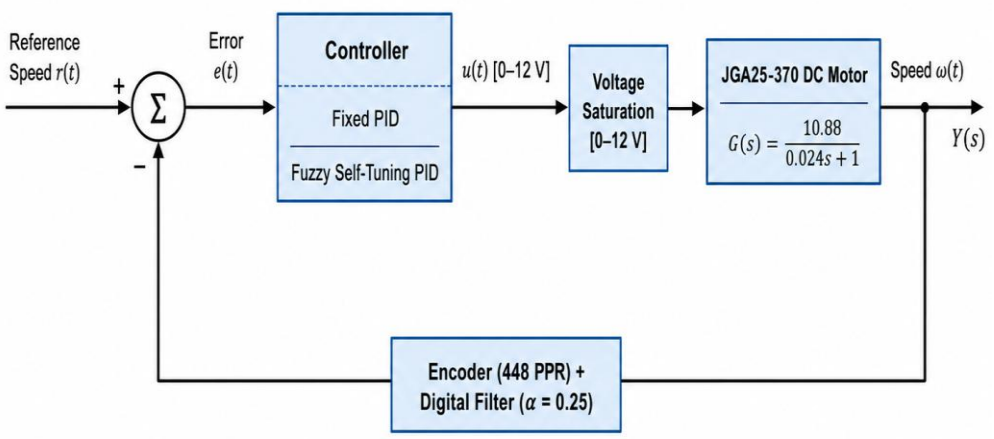

Figure 1: Closed-loop structure used for the fixed and fuzzy self-tuning PID comparison.

The sensing subsystem uses a dual-channel Hall-effect encoder. The firmware counts Channel-A rising edges and uses Channel B for direction. A configured value of 448 counts per output-shaft revolution is consistent with the observed 1.3393 rpm/count resolution at 100 ms sampling. The project archive does not include the completed ten-revolution calibration log; consequently, 448 counts/rev is treated as configured and data-consistent rather than independently calibrated.

The processing subsystem is an ATmega328P-based Arduino Uno R3. The PWM output is assigned to D9, direction commands to D10 and D11, and encoder channels to D2/INT0 and D3/INT1. The motor supply is 12 V, while the Arduino is powered through USB for serial logging. Common grounding is used between the driver, supply, and microcontroller [10], [11].

The L298N is a bipolar-transistor H-bridge and introduces a non-negligible voltage drop. The identification input is therefore the PWM-derived commanded voltage, not a directly measured motor-terminal voltage. Driver losses are absorbed into the empirical input-output gain. This distinction limits physical interpretation of the identified gain but is appropriate for reproducing the command-to-speed behaviour of the assembled platform.

During the open-loop test, a full-scale PWM command corresponding to 12 V was applied. The Arduino logged time, PWM command, estimated voltage, raw speed, and exponentially filtered speed. The same 0.1 s sampling, 0.25 exponential-moving-average coefficient, 448-count/rev setting, and 0-12 V command limit were retained in the comparative controller models.

Mathematical Modelling and Controller Design

Motor Model and Identification

For a permanent-magnet armature-controlled motor, the electrical and mechanical dynamics are represented by

$$V_a(t) = R_a i_a(t) + L_a \frac{di_a(t)}{dt} + K_m \omega(t) \qquad (1)$$

$$J\frac{d\omega(t)}{dt} + B\omega(t) = K_t i_a(t) - T_L(t) \qquad (2)$$

where V_a is armature voltage, i_a is armature current, R_a and L_a are armature resistance and inductance, K_m is the back-electromotive-force constant, J is total inertia, B is viscous friction, K_t is the torque constant, T_l is load torque, and ω is angular speed. Eliminating armature current gives

$$G(s) = \frac{K_t}{(L_a s + R_a)(Js + B) + K_t K_m} \qquad (3)$$

Reliable electrical and mechanical parameters were not available for the tested motor, and the observed response was dominated by one time scale. A first-order command-to-speed model was therefore identified from the measured step response:

$$G_{id}(s) = \frac{K}{\tau s + 1} = \frac{10.8826}{0.024254 s + 1} \ \text{rpm/V} \qquad (4)$$

The parameters K and τ were obtained by nonlinear least-squares output-error fitting. For each candidate pair, the continuous first-order response was sampled at the recorded times and passed through the same digital filter as the measured signal. The minimised objective was

$$J(K, \tau) = \sum_{k=1}^{N} \left[y[k] - \hat{y}[k; K, \tau] \right]^2 \qquad (5)$$

where y[k] is measured filtered speed and ŷ[k;K,τ] is the filter-aware model prediction. This approach avoids attributing the measurement-filter lag to the motor alone.

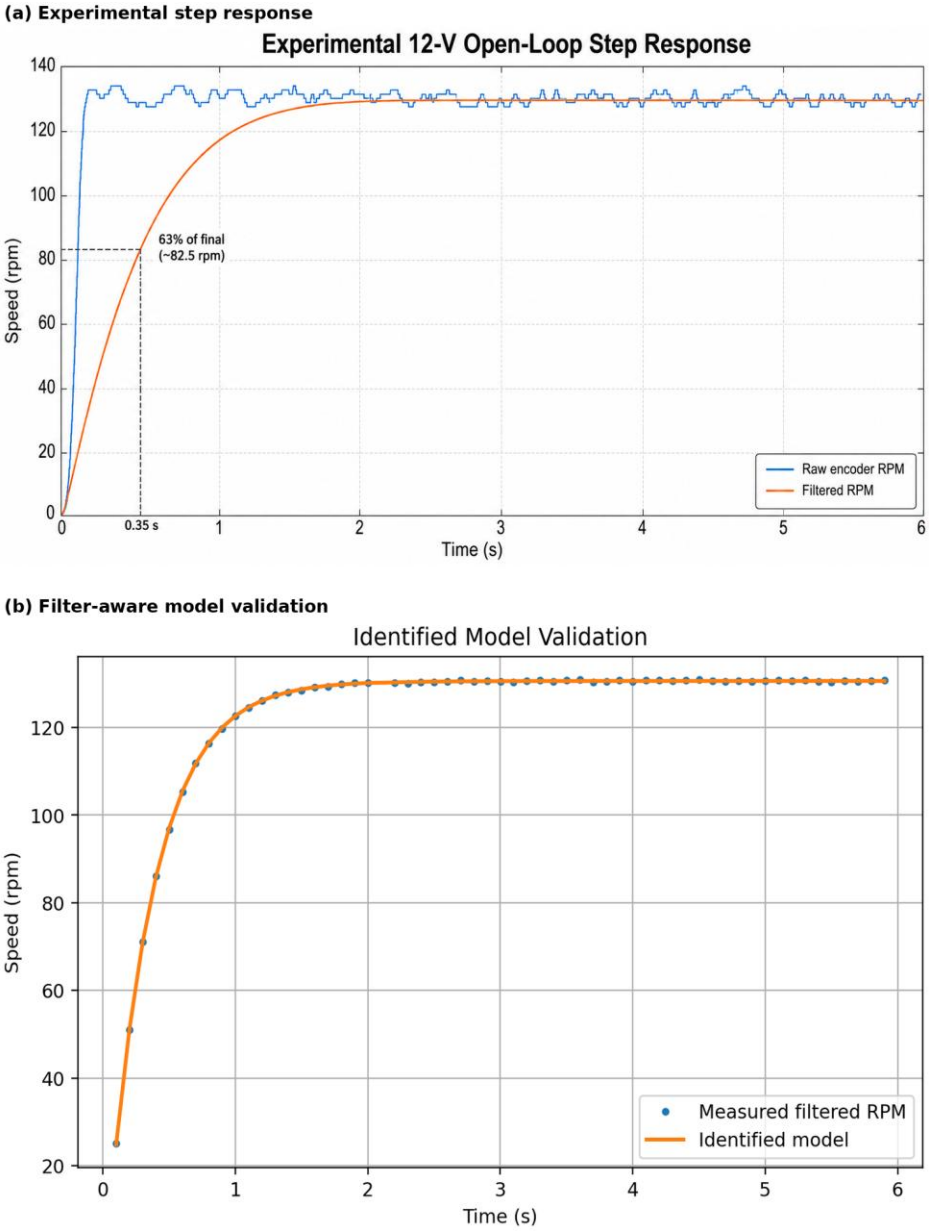

Figure 2: Open-loop identification: (a) measured raw and filtered response; (b) filtered measurement and identified model.

Speed Measurement and Filtering

For n_k counted pulses during sample k, the speed estimate is

$$y_m[k] = \frac{60 n_k}{N_p T_s} \tag{6}$$

where N_p=448 counts/rev and T_s=0.1 s. The filtered speed is calculated as

$$y_f[k] = \alpha y_m[k] + (1-\alpha) y_f[k-1] \tag{7}$$

with α=0.25. This gives an equivalent digital filter H(z)=0.25z/(z-0.75).

Fixed PID Controller

The sampled controller uses the error $e_k=r_k-y_k$, where y_k is the filtered speed. Derivative action is applied to measured speed rather than error to avoid setpoint derivative kick. The unsaturated command is

$$u_k^* = K_{p,k} e_k + I_k - K_{d,k} \dot{y}_{f,k}, \qquad u_k = \mathrm{sat}_{[0,12]}(u_k^*) \tag{8}$$

where I_k is the integral state and the derivative term is obtained from a first-order filtered speed derivative with a 0.08 s time constant. The applied command is saturated to the range 0–12 V. Conditional integration freezes the integral state when the unsaturated command exceeds 12 V with positive error or falls below 0 V with negative error. The fixed gains are K_p=0.22785 V/rpm, K_i=0.95656 V/(rpm·s), and Kd=0.001982 V·s/rpm.

Fuzzy Self-Tuning PID Controller

The self-tuning controller preserves the PID law but scales three base gains online. The normalised fuzzy inputs are

$$e_n[k] = \mathrm{sat}_{[-1,1]}\left(\frac{e_k}{100}\right), \quad \Delta e_n[k] = \mathrm{sat}_{[-1,1]}\left(\frac{\Delta e_k}{100}\right) \tag{9}$$

where the error increment at sample k is $\Delta e_k=e_k-e_{k-1}$, the saturation operator limits each normalised input to [−1,1], and the 100 rpm denominator is applied per sample rather than per second. Both inputs use five sets over [−1,1]: negative big and positive big are trapezoidal shoulders, while negative small, zero, and positive small are triangular.

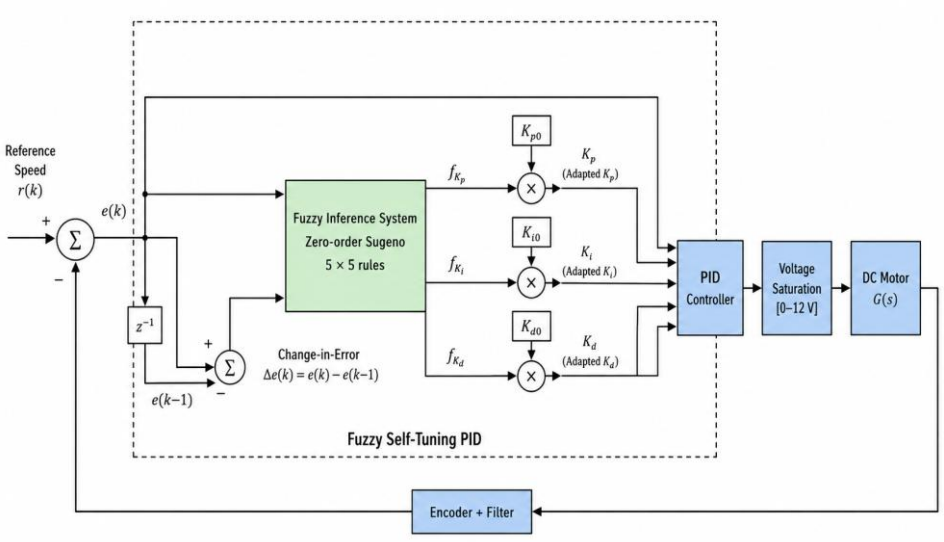

Figure 3: Zero-order Sugeno fuzzy scheduler used to modify the PID gains online.

For antecedent sets i and j, product firing gives

$$w_{ij}[k] = \mu_i(e_n[k]) \mu_j(\Delta e_n[k]) \tag{10}$$

For each gain q∈{p,i,d}, the multiplier is the weighted average of the constant rule consequents, and the online gain is

$$m_q[k] = \frac{\sum_{i,j} w_{ij}[k] c_{q,ij}}{\sum_{i,j} w_{ij}[k]}, \qquad K_{q,k} = K_{q,0} m_q[k] \tag{11}$$

The proportional multiplier levels are {0.65, 0.80, 0.95, 1.10, 1.25}; the integral levels are {0.25, 0.50, 0.80, 1.05, 1.30}; and the derivative levels are {0.70, 0.90, 1.00, 1.20, 1.40}. The base gains are K_{p0}=0.55478 V/rpm, K_{i0}=1.91516 V/(rpm·s), and K_{d0}=0.01126 V·s/rpm. The implementation has 25 antecedent combinations and three consequent matrices.

Performance Measures

The comparison uses rise time, 2% settling time, overshoot, steady-state error, integral absolute error (IAE), integral squared error (ISE), and integral time-weighted absolute error (ITAE). Control effort is characterised by rms command, command energy, saturation duration, and total variation

$$\mathrm{TV}(u) = \sum_{k=2}^{N} |u_k - u_{k-1}| \qquad (12)$$

which exposes rapid sample-to-sample command movement that may not be visible in rms or energy measures.

Evaluation Setup

The identified first-order model, the measurement filter, command saturation, and sampled controllers were implemented in MATLAB and Simulink. The project archive states that MATLAB/Simulink Desktop and legacy-compatible scripts were used, but it does not record the exact software release or operating system. The offline MATLAB model explicitly includes 1.3393 rpm/count encoder quantisation. Simulink uses the same controller logic and plant parameters; numerical results from the two environments are reported separately because state initialisation, update order, and metric extraction differ slightly.

Table 1: Common Model and Controller Parameters

Parameter	Value or setting
Identified plant	10.8826/(0.024254s+1) rpm/V
Sample period; command limit	0.1 s; 0–12 V
Speed filter; derivative filter	EMA α=0.25; 0.08 s
Encoder setting	448 counts/rev
Fixed PID gains	Kp=0.22785, Ki=0.95656, Kd=0.001982
Fuzzy base gains	Kp0=0.55478, Ki0=1.91516, Kd0=0.01126
Fuzzy scheduler	Two 5-set inputs; 25 antecedents; weighted-average zero-order Sugeno

Four evaluation scenarios were used. The nominal test applies a 0-to-100 rpm step at t=0.2 s. The disturbance test introduces a 2 V equivalent opposing input at t=2.5 s while the reference remains at 100 rpm. The parameter-variation test reduces plant gain to 85% of nominal and increases the time constant to 180% of nominal. A multi-setpoint profile of 60, 100, 80, and 110 rpm was also inspected qualitatively.

Robustness was evaluated with 500 paired Monte Carlo trials using random seed rng(42). Plant gain was sampled uniformly over 70-130% of nominal, time constant over 50-200%, and opposing disturbance voltage over 0-2.5 V. Each fixed-PID trial and fuzzy-PID trial used the same uncertainty realisation. Paired differences, 95% confidence intervals, two-sided paired t-tests, and controller win rates were calculated. Settling-time statistics used 492 pairs for which the metric was defined for both controllers.

The disturbance is an equivalent voltage offset rather than a measured shaft torque. Real-motor operation verified that both firmware programs closed the feedback loop and produced stable qualitative regulation, but repeated synchronised hardware trials were not available. The numerical evidence below must therefore be interpreted as model-based comparative evidence supported by experimental open-loop identification.

Results

Identification Quality

The fitted static gain is 10.8826 rpm/V, corresponding to 130.59 rpm at the 12 V command. The fit root-mean-square error is 0.1553 rpm and R^2 is 0.999938. Residuals remain small and approximately centred over the recorded interval. The reported 24.25 ms motor time constant is nevertheless shorter than the 100 ms sample period and is therefore weakly resolved. The narrow fitted confidence interval is conditional on the first-order model and the single supplied run; faster sampling is required before τ is treated as a high-confidence physical constant.

Nominal Tracking

Table 2: Nominal 100 Rpm Matlab Response

Metric	Fixed PID	Fuzzy PID
Rise time (s)	0.4	0.4
Settling time (s)	0.6	0.8
Overshoot (%)	0.178	0.280
SSE (rpm)	−0.014	−0.012
IAE	22.58	23.29
ISE	1409	1413
ITAE	4.953	5.305

Both controllers reach 90% of the 100 rpm command in 0.4 s. The fixed PID settles in 0.6 s, compared with 0.8 s for fuzzy self-tuning, and has lower IAE, ISE, and ITAE. These differences are modest but consistent. The result is expected because the fixed gains were optimised at this operating point, whereas online gain motion is unnecessary under nominal conditions.

Deterministic Robustness

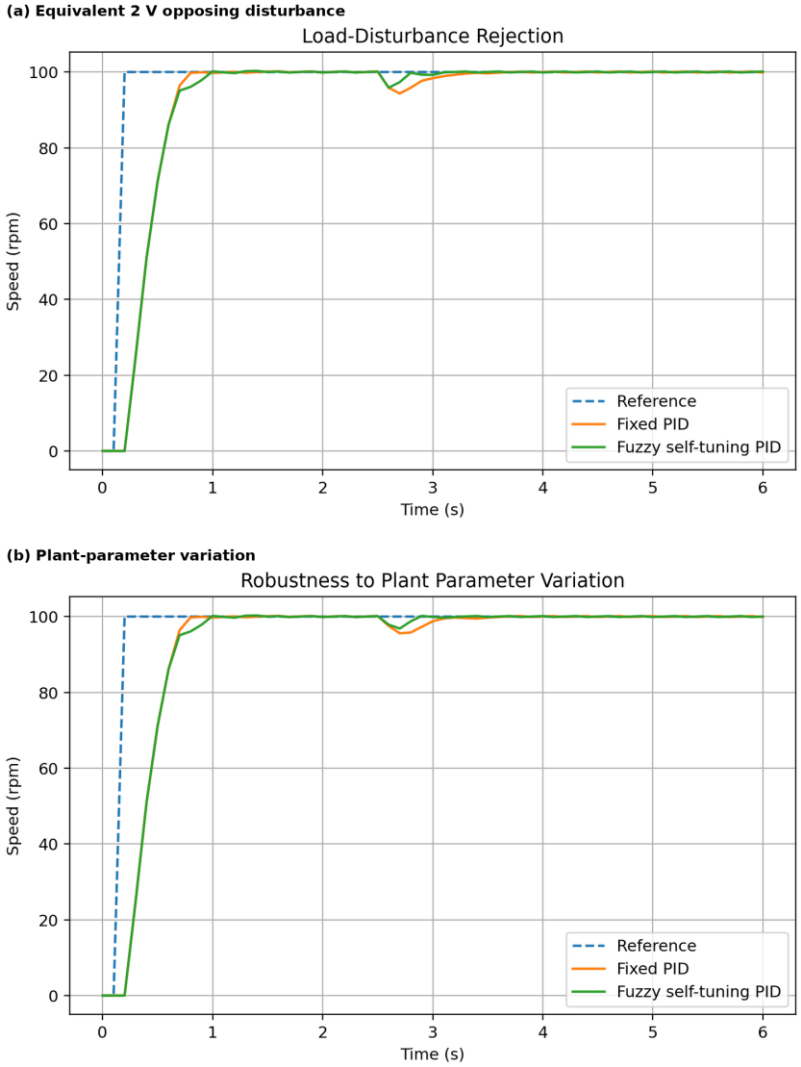

Figure 4: Deterministic robustness responses: (a) equivalent 2 V opposing disturbance; (b) reduced gain and increased time constant.

Table 3: Deterministic Off-Nominal Comparison

Scenario / metric	Fixed	Fuzzy	Improvement
Load: max deviation (rpm)	5.677	4.152	26.88%
Load: recovery (s)	0.5	0.3	40.0%
Load: post-event IAE	2.307	1.117	51.57%
Load: post-event ITAE	1.165	0.691	40.70%
Variation: max deviation (rpm)	4.422	3.170	28.32%
Variation: recovery (s)	0.5	0.3	40.0%
Variation: post-event IAE	1.933	0.973	49.63%
Variation: post-event ITAE	1.071	0.678	36.74%

Under the equivalent load disturbance, fuzzy scheduling decreases maximum speed deviation from 5.677 to 4.152 rpm and recovery time from 0.5 to 0.3 s. Post-event IAE decreases by 51.57% and ITAE by 40.70%. Under parameter variation, maximum deviation decreases from 4.422 to 3.170 rpm, while post-event IAE and ITAE decrease by 49.63% and 36.74%. These tests provide the clearest deterministic evidence that adaptation is useful away from the nominal tuning point.

Paired Monte Carlo Results

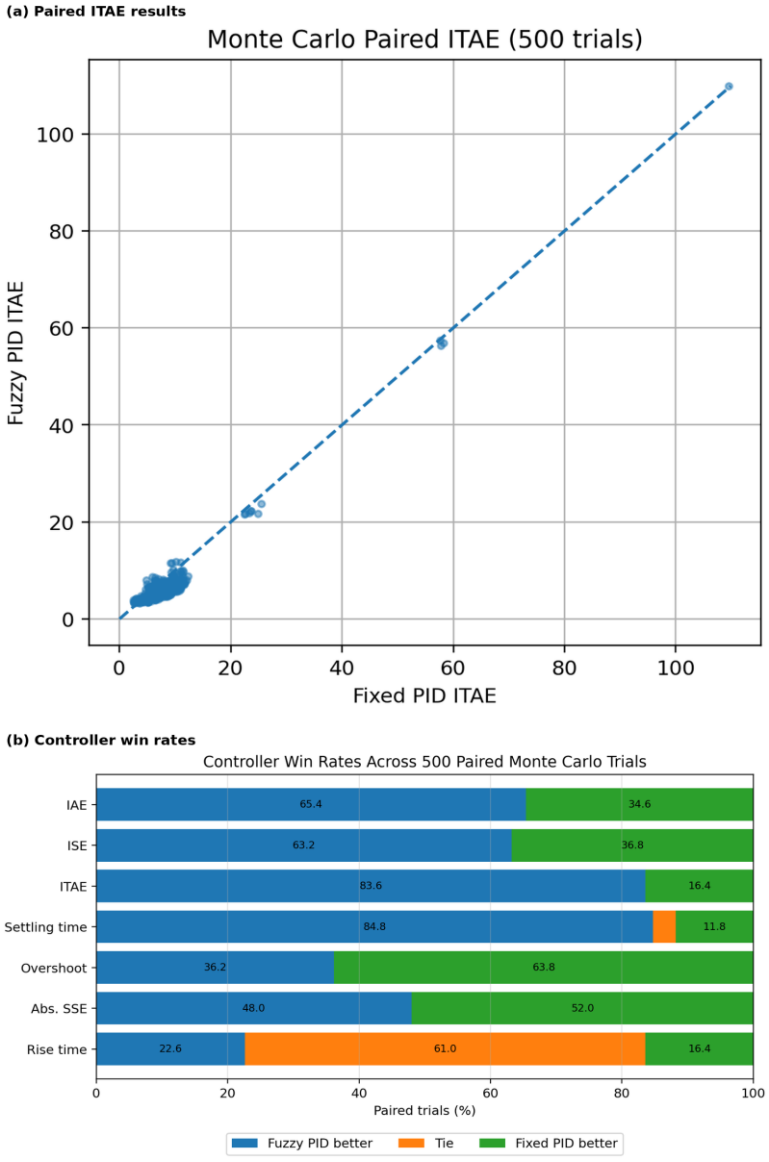

Figure 5: Paired Monte Carlo evidence: (a) ITAE for 500 matched trials; (b) controller win rates by metric.

Table 4: Paired Monte Carlo Results

Metric	Fixed → Fuzzy	Improvement	Win rate; p-value
IAE	18.020 → 17.381	3.55%	65.4%; 1.17×10^{-25}
ISE	782.273 → 777.308	0.64%	63.2%; 1.78×10^{-18}
ITAE	8.227 → 6.751	17.94%	83.6%; 1.86×10^{-74}
Settling (s)	2.288 → 1.979	13.54%	84.76%; 2.45×10^{-30}
Overshoot (%)	1.597 → 1.234	22.70%	36.2%; 0.00160
Abs. SSE (rpm)	0.150 → 0.164	−9.14%	48.0%; 0.000686

Mean ITAE falls from 8.227 to 6.751, a 17.94% reduction, and fuzzy PID wins 83.6% of paired trials. Mean settling time decreases from 2.288 to 1.979 s and fuzzy PID wins 84.76% of the 492 valid pairs. IAE and ISE improvements are statistically detectable but smaller in engineering magnitude.

The overshoot result requires caution. Mean overshoot is 22.70% lower for fuzzy PID, but fuzzy PID wins only 36.2% of paired trials. The lower mean is driven mainly by suppression of severe outliers: maximum overshoot decreases from 12.16% to 6.895%. The defensible conclusion is that fuzzy adaptation reduces worst-case overshoot rather than providing general trial-by-trial overshoot superiority. Absolute steady-state error is 9.14% higher for fuzzy PID, reinforcing the nominal accuracy advantage of the fixed controller.

Control Effort and Preliminary Hardware Operation

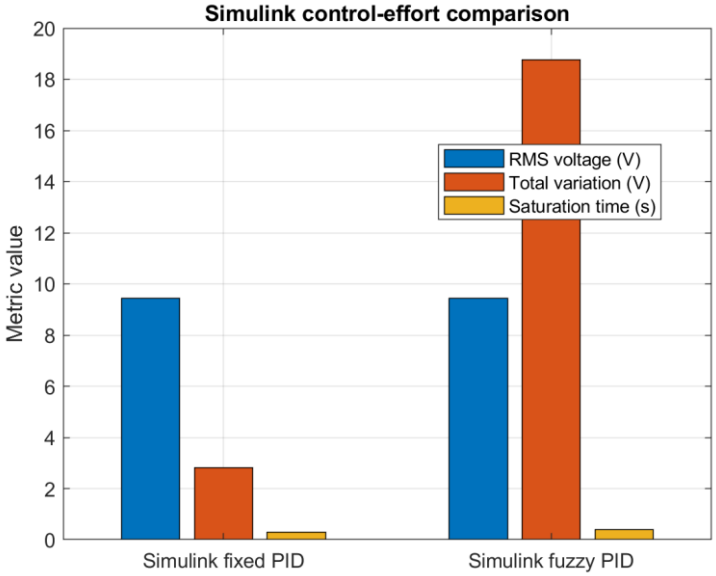

Figure 6: Simulink control-effort comparison.

Table 5: Simulink Control-Effort Comparison

Metric	Fixed PID	Fuzzy PID
RMS voltage (V)	9.429	9.439
Mean voltage (V)	9.403	9.408
Total variation (V)	2.811	18.769
Saturation time (s)	0.3	0.4
$\int u^2 dt$	513.1	514.3

Rms voltage, mean voltage, and command energy are almost identical. Total command variation, however, increases from 2.811 to 18.769 V, or by a factor of 6.68. This indicates substantially more aggressive sample-to-sample action from the fuzzy scheduler. Potential consequences include higher switching activity, greater sensitivity to encoder quantisation, acoustic noise, and additional heating in the bipolar L298N stage.

Both controller programs were operated on the assembled motor platform and produced satisfactory qualitative speed regulation. No repeated closed-loop logs, measured terminal voltage, measured current, execution-time profile, or randomised controller-order trials were included. Hardware operation therefore confirms functional implementation, not numerical superiority.

Discussion

The comparison shows that controller ranking depends on the operating condition and on the selected metric. At the nominal design point, the optimised fixed PID is the better choice: it settles faster and produces lower integrated error without the additional gain movement of the fuzzy scheduler. This is not a failure of adaptation; it reflects the absence of a need to adapt when the nominal model and tuning assumptions hold.

When plant gain, time constant, or equivalent load changes, fixed gains become a compromise. The fuzzy scheduler responds to increasing error and change in error by modifying proportional, integral, and derivative action within the permitted multiplier ranges. The resulting response recovers faster and has lower late-occurring error, which explains why the ITAE improvement is much larger than the IAE or ISE improvement.

The paired Monte Carlo design strengthens the comparison because each controller is evaluated on the same uncertainty realisation. Reporting a p-value alone would be insufficient with 500 trials; the effect size and win rate reveal whether a lower mean is broadly representative. This is especially important for overshoot, where the mean suggests an advantage but the 36.2% win rate shows that the advantage is concentrated in outlier suppression.

The major practical trade-off is control-signal variation. The fuzzy controller does not consume materially more rms command or command energy, yet it moves the command much more frequently. A first-order smoother on gain multipliers, a bounded voltage slew rate, or narrower multiplier ranges may reduce this behaviour. Any such modification must be re-evaluated under the same deterministic and Monte Carlo conditions because excessive smoothing could remove the robustness benefit.

Generalisability is limited by the single identified motor, the first-order model, the voltage-offset disturbance representation, and the absence of repeated quantitative hardware tests. In addition, the 24.25 ms time constant is not well resolved by 100 ms sampling. The results therefore support the comparative method and the observed model-based trade-offs, but they should not be interpreted as universal performance claims for all DC motors or all fuzzy rule bases.

Limitations and Future Work

The principal limitation is the evidence boundary: hardware data were used for open-loop identification, whereas the numerical fixed-versus-fuzzy results are simulations. A publishable experimental extension should use at least five repeats per condition, randomised controller order, synchronised speed logs, measured motor-terminal voltage and current, and confidence intervals. The identification test should be repeated at 10-20 ms sampling to verify the fast time constant and with an independent validation dataset.

The fuzzy rule base and multiplier ranges were selected heuristically, and no formal closed-loop stability proof was performed. Future work should evaluate gain smoothing or command slew-rate limiting, profile Arduino execution time and SRAM use, independently calibrate encoder counts/rev, apply a physical torque disturbance, and compare the proposed scheduler with optimisation-based fuzzy tuning or another adaptive-control baseline.

Conclusion

A fixed PID controller and a zero-order Sugeno fuzzy self-tuning PID controller were compared for JGA25-370 speed control using a plant identified from Arduino encoder data. The fixed PID provided the better nominal response, settling 0.2 s faster and producing lower integral error indices at 100 rpm. Fuzzy self-tuning was more effective under off-nominal conditions: maximum deviation decreased by 26.88% under an equivalent load disturbance and by 28.32% under parameter variation, while post-event IAE decreased by 51.57% and 49.63%, respectively. In 500 paired Monte Carlo trials, mean ITAE and settling time decreased by 17.94% and 13.54%. These benefits were accompanied by a 6.68-fold increase in total control-signal variation and a slight deterioration in absolute steady-state error. The appropriate engineering conclusion is therefore conditional: fixed PID is preferred at a stable, well-modelled operating point, whereas fuzzy self-tuning offers stronger robustness when plant and load conditions change. Quantitative hardware validation remains necessary before deployment claims are made.

Compliance with ethical standards

Disclosure of conflict of interest

The authors declare that they have no conflict of interest.

References

[1] K. Ogata, Modern Control Engineering, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.
[2] K. J. Åström and T. Hägglund, Advanced PID Control. Research Triangle Park, NC, USA: ISA, 2006.
[3] K. M. Passino and S. Yurkovich, Fuzzy Control. Menlo Park, CA, USA: Addison-Wesley, 1998.

[4] Z.-Y. Zhao, M. Tomizuka, and S. Isaka, “Fuzzy gain scheduling of PID controllers,” IEEE Trans. Syst., Man, Cybern., vol. 23, no. 5, pp. 1392-1398, 1993, doi: 10.1109/21.260670.
[5] Z.-W. Woo, H.-Y. Chung, and J.-J. Lin, “A PID type fuzzy controller with self-tuning scaling factors,” Fuzzy Sets Syst., vol. 115, no. 2, pp. 321-326, 2000, doi: 10.1016/S0165-0114(98)00159-6.
[6] N. Messaadi and A. Amroun, “Speed control of DC motor using fuzzy PID controller,” arXiv:2108.05450, 2021, doi: 10.48550/arXiv.2108.05450.
[7] T. Prasoetphon, K. Selamassakul, T. Sittiwanchai, and W. Jitviriya, “Fuzzy-PID vs. PID controllers: A comparative study on DC motor speed control,” in Proc. 2024 1st Int. Conf. Robotics, Engineering, Science, and Technology (RESTCON), 2024, doi: 10.1109/RESTCON60981.2024.10463588.
[8] H. Supriyono, F. F. Alanro, and A. Supardi, “Development of DC motor speed control using PID based on Arduino and MATLAB for laboratory trainer,” Jurnal Nasional Teknik Elektro, vol. 13, no. 1, pp. 36-41, 2024, doi: 10.25077/jnte.v13n1.1155.2024.
[9] M. Aviles, J. Rodríguez-Reséndiz, J. Pérez-Ospina, and O. Lara-Mendoza, “A comprehensive methodology for the development of an open source experimental platform for control courses,” Technologies, vol. 11, no. 1, Art. no. 25, 2023, doi: 10.3390/technologies11010025.
[10] STMicroelectronics, “L298 dual full-bridge driver,” DS0218, Rev. 5, Oct. 2023.
[11] Arduino, “Arduino UNO R3 datasheet,” A000066, accessed Jul. 26, 2026.

