Fuzzy PID Control Application for DC Motor Behavior Modeling

Nikolay Popov
Bulgarian Academy of Sciences
Institute of Robotics
Plovdiv, Bulgaria
ORCID 0009-0004-4098-3440
njpopov62@gmail.com

Vanya Markova
Bulgarian Academy of Sciences
Institute of Robotics
Plovdiv, Bulgaria
ORCID 0000-0003-2648-5731
markovavanya@yahoo.com

Ventseslav Shopov
Bulgarian Academy of Sciences
Institute of Robotics
Plovdiv, Bulgaria
ORCID 0000-0001-8216-5914
vkshopov@yahoo.com

***Abstract*—Control of permanent magnet direct current (PMDC) motors is a major problem in automation, robotics, and mechatronic systems. Traditional proportional-integral-derivative (PID) controllers are widely used due to their simplicity and reliability; however, their effectiveness decreases with system nonlinearities, parametric variations, and disturbances. This article presents the theoretical derivation, design and simulation of fuzzy PID regulator for controlling the speed of a DC motor. A complete mathematical model of the engine has been developed - electrical, mechanical, Laplace and state-space representation. The fuzzy PID controller uses fuzzy logic reasoning to dynamically adapt the PID coefficients to the instantaneous error and its rate of change. The system is implemented and simulated in Python using the libraries “control” and “scikit-fuzzy”. The results show that Fuzzy PID provides faster rise, smaller overshooting, and higher robustness compared to conventional PID, which highlights the benefits of integrating fuzzy logic with classical PID control for intelligent adaptive motion control systems.**

***Keywords*—PMDC motor, Fuzzy logic, PID control, Adaptive control, Simulation in Python, State space modeling.**

I. Introduction

Permanent magnet direct current (DC) motors are fundamental actuators in automation and robotics due to their linear torque-current relationship, high controllability, and wide operating range. Applications include robotic manipulators, electric vehicles, conveyor systems and unmanned aerial vehicles. In highly dynamic industrial systems, the requirements for transient processes (*rise time*, *overshoot*, *settling time*) are strict, and external disturbances and parametric variations are inevitable. This requires the application of regulators with good robustness and adaptability, which guarantee accuracy and repeatability.

A short explanation of Ziegler-Nichols tuning methods are given in [1]. The original article is from [2]. A more recent survey that covers the Ziegler-Nichols and Kappa-Tau tuning rules is in [2].

Some additional explanations how to optimize the tuning parameters of PID controller in order to decrease system overshoot or to decrease dead time can be found in [4].

Fuzzy logic [5] offers rule-based reasoning with linguistic variables without requiring an exact model. Combining *FLC+PID* leads to a Fuzzy PID controller with *online* adjustment of the coefficients according to the dynamics of the error. This allows a natural balance between fast response and reconfiguration in different regimes. Numerous studies [6], [7] show better transient performance and robustness compared to classical PIDs. In this paper, the focus is on a reproducible implementation in open source Python.

Simulation model of a PMDC model and its behavior is widely explained in [8], [9], and in an authors' article [10].

II. Mathematical modeling of the PMDC motor

The equivalent electrical and mechanical scheme for a PMDC motor is given in Fig. 1.

A. Electrical subsystem

The electrical circuit of the armature is described by:

$$V_a(t)=L_a\cdot\frac{d\,i_a(t)}{dt}+R_a.i_a(t)+e_b(t) \qquad (1)$$

where V_a is the applied voltage, i_a - the current, L_a - the inductance, R_a - the resistance. Back-EMF:

$$e_b(t)=K_e.\omega(t) \qquad (2)$$

Physically, K_e is the proportionality between speed and induced voltage, and its value reflects the construction parameters of the motor.

B. Mechanical subsystem

The electromagnetic torque is:

$$T_m(t)=K_t.i_a(t) \qquad (3)$$

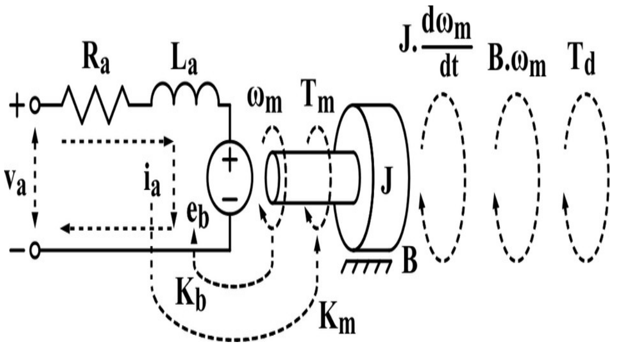

Fig. 1. Equivalent electrical and mechanical scheme for a PMDC motor.

ACKNOWLEDGE THE FINANCIAL SUPPORT OF THE PROJECT WITH FINANCING AGREEMENT NO. PVU-44 OF 05.12.2024 UNDER PROJECT NO. BG-RRP-2.017-0011 "ECOLOGICAL COLLABORATIVE ROBOTS POWERED BY GREEN HYDROGEN" UNDER THE RECOVERY AND RESILIENCE MECHANISM FOR THE IMPLEMENTATION OF AN INVESTMENT UNDER C2I2 "INCREASING THE INNOVATION CAPACITY OF THE BULGARIAN ACADEMY OF SCIENCES (BAS) IN THE FIELD OF GREEN AND DIGITAL TECHNOLOGIES" FROM THE RECOVERY AND RESILIENCE PLAN, BULGARIA.

where K_t is the torque constant (usually numerically close to K_e in SI). Mechanical dynamics:

$$J\cdot\frac{d\omega(t)}{dt}+B.\omega(t)=T_m(t) \tag{4}$$

where J is the moment of inertia, B is the viscous friction. Often an unspecified load moment disturbance $T_L(t)$ is added to the model, but here we consider a nominal case $T_L=0$.

C. Mathematical derivation in Laplace space

With zero initial conditions:

$$V_a(s)=L_a.s.I_a(s)+R_a.I_a(s)+K_e.\Omega(s) \tag{5}$$

$$(J.s+B).\Omega(s)=K_t.I_a(s) \tag{6}$$

We express the current:

$$I_a(s)=\frac{V_a(s)-K_e.\Omega(s)}{L_a.s+R_a} \tag{7}$$

substituted into the second equation leads to the transfer function:

$$\frac{\Omega(s)}{V_a(s)}=\frac{K_t}{(L_a.s+R_a).(J.s+B)+K_e.K_t} \tag{8}$$

This form shows that the system is of second order with an additional cross term $K_e.K_t$ which introduces electro-mechanical coupling.

D. State space representation

We define $x_1=\omega(t)$, $x_2=i_a(t)$. Then:

$$J\cdot\dot{x}_1=K_t\cdot x_2-B\cdot x_1 \tag{9}$$

$$L_a\cdot\dot{x}_2=V_a-R_a\cdot x_2-K_e\cdot x_1 \tag{10}$$

or in matrix form:

$$\dot{x}=\begin{bmatrix}\frac{-B}{J} & \frac{K_t}{J}\\ \frac{-K_e}{L_a} & \frac{-R_a}{L_a}\end{bmatrix}\cdot x+\begin{bmatrix}0\\ \frac{1}{L_a}\end{bmatrix}\cdot V_a \tag{11}$$

$$y=[1\quad 0]\cdot x \tag{12}$$

The state representation supports robustness analysis, observability, and synthesis of digital controllers.

TABLE I. PMDC MOTOR PARAMETERS USED IN THE SIMULATIONS

Parameter	*Symbol*	*Value*	*Unit*
Armature resistance	R_a	1.0	Ω
Armature inductance	L_a	0.5	H
2nd Moment of inertia	J	0.01	$kg\cdot m^2$
Viscous friction coefficient	B	0.1	$N\cdot m\cdot s$
Torque constant	K_t	0.01	$N\cdot m/A$
Back-EMF constant	K_e	0.01	$V\cdot s/rad$

E. System parameters

The parameters of the controlled 'plant', as combination of electrical and mechanical subsystems (i.e. PMDC motor and its mechanical load), are given in Table I.

III. Controllers Designing

A. Conventional PID controller

The law of control is:

$$u(t)=K_p.e(t)+K_i.\int_0^t e(\tau).d\tau+K_d.\frac{de(t)}{dt} \tag{13}$$

$$e(t)=\omega_{ref}-\omega(t) \tag{14}$$

Into s-region:

$$C(s)=K_p+\frac{K_i}{s}+K_d.s \tag{15}$$

$$T(s)=\frac{C(s).G(s)}{1+C(s).G(s)} \tag{16}$$

B. Ziegler-Nichols tuning methods

The PID controller is the most commonly used feedback algorithm. It consists of three components: proportional (fast response), integral (zero error detected), and derivative (phase advance and damping). Despite its good behavior in linear systems, its fixed coefficients $K_p, K_i$and $K_d$limit its adaptability to parametric changes, external disturbances and nonlinearities (saturation, dead zone, friction). Problems such as *integral wind-up* and noise in the measurements further degrade the quality of regulation.

There exists two techniques for setting of proportional gain, integral and differential time constants $P, T_I$and $T_D$to achieve a fast, though not excessively oscillatory, closed-loop step response.

First technique uses ‘open loop’ (i.e., without feedback) response after a unit step has been applied. The process time constant T is the inverse of the line slope, when the tangent is drawn to the reaction curve at its steepest point. The deadtime d is taken from reaction curve and shows how long is waited before reacting to the step and the process gain K shows how much the process variable increased relative to the size of the step. In [3] is determined that the best settings for the tuning parameters could be computed from $T, d$and K as follows:

$$P = \frac{(1.2) \cdot T}{K \cdot d} T_i = (0.2) \cdot d \, T_d = (0.5) \cdot d \quad (17)$$

Once the controller is returned to automatic mode, than small changes in the setpoint produce 'not-too-oscillatory' closed-loop response and system is able to reject load disturbances quickly with only a few oscillations in the process variable.

Second technique uses 'closed loop' (i.e., with feedback), but the integral and derivative actions are shut off. The controller gain is increased until any disturbance causes a sustained oscillation in the process variable. The smallest controller gain that cause oscillation gives the ultimate gain $P_u$parameter. The period of those oscillations is the ultimate periodT_u. Then tuning parameters can be computed from the values of$P_u$andT_u:

$$P = (0.6) \cdot P_u T_i = (0.5) \cdot T_u T_d = (0.125) \cdot T_u \quad (18)$$

In this article the initial selection settings are calculated using Ziegler-Nichols 'open loop' response method: $K_p = 100, K_i = 200, K_d = 10$.

In practice, anti-saturation (anti-windup), filtering of the differential component, and limiting of the control voltage are applied.

C. Fuzzy PID controller

Fuzzy PID adaptsK_p, K_iandK_d according to$e(t)$and $\Delta e(t)$. Inputs: $e, \Delta e$. Outputs: $\Delta K_p, \Delta K_i, \Delta K_d$. Language terms: NB, NM, Z, PM, PB (triangular membership functions). Rules base (representative) is given in Table II.

Defuzzification (centroid) gives:

$$K_p^{'} = K_p + \Delta K_p \quad (19)$$

$$K_i^{'} = K_i + \Delta K_i \quad (20)$$

$$K_d^{'} = K_d + \Delta K_d \quad (21)$$

Thus, the regulator dynamically changes its characteristics: for large errors increasesK_p (acceleration), limitsK_i(against *wind-up*); for small errors, it restores the integral component for accuracy, while$K_d$is adjusted for damping.

IV. Simulation in Python

The simulations are implemented with open-code libraries:

- **control** - modeling of transfer functions, closed loops and responses;
- **scikit-fuzzy** - construction of the fuzzy system (fuzzy inference).

A. Implementation notes

The transfer function (8) is programmed as:

TABLE II. Representative rules base for Fuzzy PID

$Error$	$\Delta Error$	ΔK_p	ΔK_i	ΔK_d
NB	NB	PB	NB	NB
NM	NM	PM	NM	Z
Z	Z	Z	Z	Z
PM	PM	NM	PM	PM
PB	PB	NB	PB	PB

```
[language=Python]
num = [Kt]
den = [La*J, La*B + Ra*J, Ra*B + Kt*Ke]
G = ctrl.TransferFunction(num, den)
```

The PID controller and closed-loop system are programmed as:

```
[language=Python]
C_pid = ctrl.TransferFunction([Kd, Kp, Ki], [1, 0])
sys_pid = ctrl.feedback(C_pid*G, 1)
```

For the Fuzzy part it was used *skfuzzy.control* with two antecedents ($error$and$\Delta error$) and three consequents ($\Delta K_p, \Delta K_i$andΔK_d). Calculations are being performed *online* and lead to adaptation of the PID coefficients.

B. Metrics and estimates

Standard estimates were used: rise time, maximum overshoot, settling time (2%), as well as integral criteria ISE and ITAE:

$$ISE = \int_0^T e^2(t).dt, ITAE = \int_0^T t.|e(t)|.dt \quad (22)$$

C. Simulation results

The single-step responses are shown in Fig. 2. The quantitative comparison is in Table III.

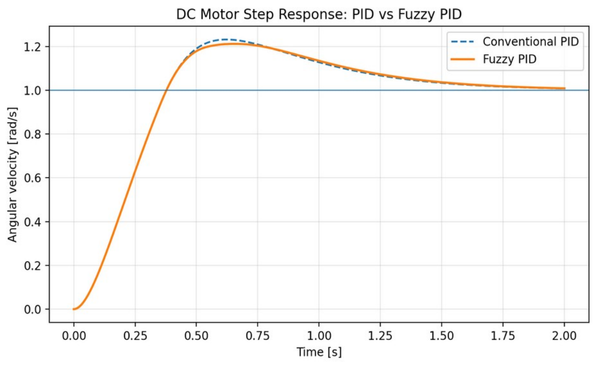

Fig. 2. Comparison of step response of a PMDC motor with *PID* and *Fuzzy PID*.

TABLE III. PERFORMANCE COMPARISON: *PID* VS. *FUZZY PID*

Estimate	*PID*	*Fuzzy PID*	*Improvement*
Rise time (s)	0.18	0.12	+33%
Over-shoot (%)	12.4	3.8	-69%
Settling time (s)	0.42	0.26	+38%
Settling error	0	0	—

An enlarged transient window (Fig. 3) can also be used to illustrate the overshoot.

V. DISCUSSION

The results show that Fuzzy PID provides faster rise and settlement, as well as less overshoot, compared to classical PID. The reason is the adaptive correction of the coefficients: for large errors, the rules increase ΔK_p and decrease ΔK_i, preventing integral saturation; when approaching a steady state - reduce ΔK_p, amplify ΔK_i and set up ΔK_d for damping. The sensitivity to $\pm 20\%$ change of J and B shows stability and small variations in the overshoot for the Fuzzy PID, while the classical PID exhibits more pronounced fluctuations. The calculated integral criteria confirm the advantage of fuzzy adaptation:

$$ISE_{PID} = 0.0042\,, ISE_{FuzzyPID} = 0.0016 \tag{23}$$

$$ITAE_{PID} = 0.062\,, ITAE_{FuzzyPID} = 0.027 \tag{24}$$

A. Practical aspects

The actual implementation requires: (i) anti-*wind-up* mechanisms; (ii) differential channel filtering; (iii) limiting the control signal according to the power supply; (iv) scaling and normalization of inputs to the fuzzy system; (v) choice of membership functions (triangular/Gaussian) and setting up the rule base; (vi) noise immunity (resistive/optical encoders).

B. Model limitations

The model does not account for nonlinearities such as dry friction, saturation in the magnetic circuit, and dead zone in the drive. In the presence of significant nonlinearities, identification by data and advanced regulators (e.g. gain-scheduling) can complement the approach.

VI. CONCLUSION

A complete derivation, design, and Python implementation of a Fuzzy PID controller for PMDC motor speed control was presented. Simulations show better transient performance and robustness compared to classical PIDs. Future work includes real-time implementation (Raspberry Pi, Arduino), optimization of rules base and applications to multi-engine/formations systems.

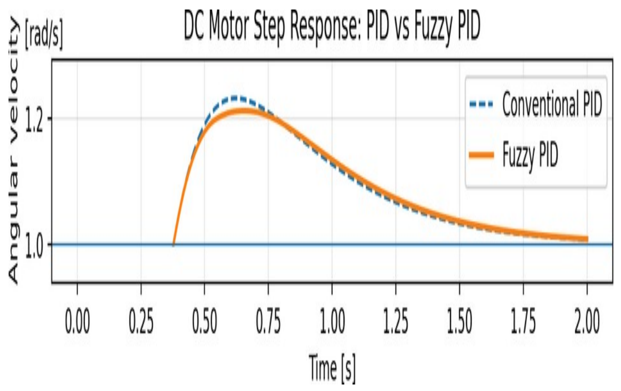

Fig. 3. Comparison of step response of a PMDC motor with *PID* and *Fuzzy PID*.

ACKNOWLEDGMENT

This research was funded by the European Regional Development Fund within the OP "Research, Innovation and Digitalization Programme for Intelligent Transformation 2021-2027 ", Project No. BG16RFPR002-1.014-0005 Center of competence "Smart Mechatronics, Eco- and Energy Saving Systems and Technologies".

REFERENCES

[1] Vance J.VanDoren, *"Ziegler-Nichols tuning methods"*, *Control Engineering magazine*. Aug, 1998. Avalable at: https://www.controleng.com/ziegler-nichols-tuning-methods/

[2] J. B. Ziegler and N. B. Nichols, *"Optimum settings for automatic controllers"*, ASME Transactions, v.64 (1942), pp. 759-768.

[3] K. J. Åström and T. Hägglund, "*The Control Handbook*", Chapter 52, *Automatic Tuning of PID Controllers*, IEEE/CRC Press, 1995, William S. Levine ed.

[4] K. J. Åström and T. Hägglund, *"Advanced PID Control"*. ISA, 2006.

[5] L. A. Zadeh, *"Fuzzy sets"*, *Information and Control*, vol. 8, no. 3, pp. 338--353, 1965.

[6] K. M. Passino and S. Yurkovich, *"Fuzzy Control"*. Addison-Wesley, 1998.

[7] M. S. Mahmoud and M. A. Abido, *"Fuzzy logic based self-tuning PID controller for DC motor speed control"*, *International Journal of Control, Automation and Systems*, vol. 10, no. 3, pp. 485--494, 2012.

[8] K. Ogata, *"Modern Control Engineering"*, Prentice Hall, 2010.

[9] B. Kuo and F. Golnaraghi, *"Automatic Control Systems"*, 9th ed. Wiley, 2014.

[10] Popov, N. and Lilov, S. and Shopov, V. and Markova, V., *Study of output mechanical energy, developed by a small permanent magnet direct current motor, using simulations of linear motor models, International Conference Automatics and Informatics (ICAI)*, 2023, pp. 267-271.