

Rancang Bangun Sistem Peningkatan Efisiensi Kontrol Motor BLDC Berbasis Espressif System ESP32 Menggunakan Algoritma PID

^{1*}Lukman Hakim, ²Moh Shohibur Rifky, ³Fidya Eka Prahesti,
⁴M. Dewi Manikta Puspitasari, ⁵Sofyan Ahmadi

^{1,2,3,4} Teknik Elektro, Universitas Nusantara PGRI Kediri

E-mail: ^{*1}hakimlukman040203@gmail.com, ²rifymadurace@gmail.com,

³fidya.prahesti@unpkdr.ac.id, ⁴dewimanikta@unpkedii.ac.id, ⁵sofyan.ahmadi@unpkdr.ac.id

Penulis Korespondens : Lukmaan Hakim

Abstrak— Penelitian ini bertujuan untuk merancang dan membangun sistem peningkatan efisiensi kontrol kecepatan motor *BLDC* (*Brushless Direct Current*) berbasis mikrokontroler *ESP32* dengan menggunakan algoritma *PID* (*Proportional–Integral–Derivative*). Sistem dirancang untuk diterapkan pada sepeda listrik beroda 14 inci. Komponen utama sistem meliputi sensor *Hall* efek, sensor arus *ACS712-20A*, sensor tegangan dengan pembagi tegangan, *DAC* *GPIO25* sebagai pengendali *throttle* melalui *op-amp* *LM358*, serta platform monitoring *Blynk* berbasis *Wi-Fi*. Algoritma *PID* diimplementasikan secara diskrit dengan parameter $K_p = 3,9$, $K_i = 0,05$, dan $K_d = 0,01$. Hasil pengujian menunjukkan sistem bekerja pada rentang target kecepatan 6–12 km/jam. Pada target 8 km/jam, sistem mencapai kecepatan *steady state* 7,40 km/jam dengan error 0,60 km/jam, rise time 1,02 detik, dan overshoot 1,33%. Pada target 10 km/jam, sistem menghasilkan overshoot 0,00% dengan rise time 1,93 detik. Namun pada target 6 km/jam terjadi overshoot tinggi (58,82%), dan pada target 12 km/jam sistem mengalami saturasi dengan error *steady state* 6,36 km/jam. Sistem juga dilengkapi mekanisme *failsafe* yang secara otomatis beralih ke mode manual saat koneksi *Wi-Fi* terputus.

Kata Kunci— Motor *BLDC*, *ESP32*, Algoritma *PID*, Kontrol Kecepatan

Abstract— This study aims to design and develop a speed control efficiency enhancement system for a *Brushless Direct Current* (*BLDC*) motor based on the *ESP32* microcontroller using a *Proportional–Integral–Derivative* (*PID*) control algorithm. The system is designed for application in a 14-inch electric bicycle. The main components include a *Hall-effect* sensor, an *ACS712-20A* current sensor, a voltage sensor based on a voltage divider, a *GPIO25* *DAC* for *throttle* control through an *LM358* operational amplifier, and a *Wi-Fi*-based *Blynk* monitoring platform. The *PID* algorithm is implemented in discrete form with parameters of $K_p = 3.9$, $K_i = 0.05$, and $K_d = 0.01$. Experimental results show that the system operates effectively within a target speed range of 6–12 km/h. At a target speed of 8 km/h, the system achieved a steady-state speed of 7.40 km/h with a steady-state error of 0.60 km/h, a rise time of 1.02 s, and an overshoot of 1.33%. At a target speed of 10 km/h, the system achieved zero overshoot with a rise time of 1.93 s. However, at a target speed of 6 km/h, the system exhibited a high overshoot of 58.82%, while at 12 km/h, the controller experienced saturation, resulting in a steady-state error of 6.36 km/h. The system also incorporates a *failsafe* mechanism that automatically switches to manual mode whenever the *Wi-Fi* connection is lost.

Keywords— *BLDC* Motor, *ESP32*, *PID* Algorithm, Speed Control

This is an open access article under the CC BY-SA License.



I. PENDAHULUAN

Perkembangan teknologi sistem kendali pada motor listrik terus mengalami peningkatan seiring dengan kebutuhan industri modern yang menuntut efisiensi energi, keandalan, dan performa yang tinggi. Salah satu jenis motor yang banyak digunakan dalam berbagai aplikasi industri, robotika, kendaraan listrik, dan sistem otomasi adalah Motor *Brushless Direct Current* (BLDC). Motor BLDC memiliki beberapa keunggulan dibandingkan motor DC konvensional, seperti efisiensi yang lebih tinggi, umur pakai yang lebih panjang, perawatan yang lebih rendah, serta kemampuan menghasilkan torsi yang stabil dengan respons dinamis yang baik [1].

Motor BLDC tidak menggunakan sikat (*brush*) seperti pada motor DC biasa, sehingga dapat mengurangi gesekan mekanis dan meningkatkan efisiensi kerja motor. Selain itu, konfigurasi magnet permanen pada rotor dan kumparan pada stator memungkinkan motor BLDC memiliki performa yang lebih baik dalam berbagai kondisi operasional. Oleh karena itu, motor jenis ini banyak digunakan dalam berbagai sektor, terutama pada sistem transportasi listrik, peralatan industri, dan perangkat otomasi [2].

Meskipun memiliki banyak keunggulan, pengendalian kecepatan motor BLDC memerlukan sistem kontrol yang baik agar dapat bekerja secara optimal. Sistem kontrol diperlukan untuk menjaga kestabilan kecepatan motor terhadap perubahan beban maupun perubahan kondisi sistem. Tanpa sistem kontrol yang tepat, kecepatan motor dapat mengalami fluktuasi yang menyebabkan kinerja sistem menjadi kurang optimal [3].

Salah satu metode pengendalian yang paling banyak digunakan dalam sistem kontrol motor adalah kontrol PID (*Proportional-Integral-Derivative*). Kontrol PID memiliki kemampuan untuk menghasilkan respons sistem yang cepat dan stabil terhadap perubahan kondisi operasi. Dengan mengatur parameter proporsional, *integral*, dan *derivatif* secara tepat, sistem kontrol dapat meminimalkan kesalahan antara nilai *setpoint* dan nilai aktual dari kecepatan motor [4]. Namun demikian, metode PID konvensional memiliki keterbatasan ketika dihadapkan pada sistem yang bersifat nonlinier dan memiliki ketidakpastian parameter, seperti yang terjadi pada motor BLDC. Dalam kondisi tertentu, sistem kontrol PID mungkin tidak mampu memberikan performa yang optimal, terutama ketika terjadi perubahan beban atau gangguan pada sistem. Oleh karena itu, diperlukan metode kontrol tambahan yang mampu mengatasi karakteristik nonlinier tersebut.

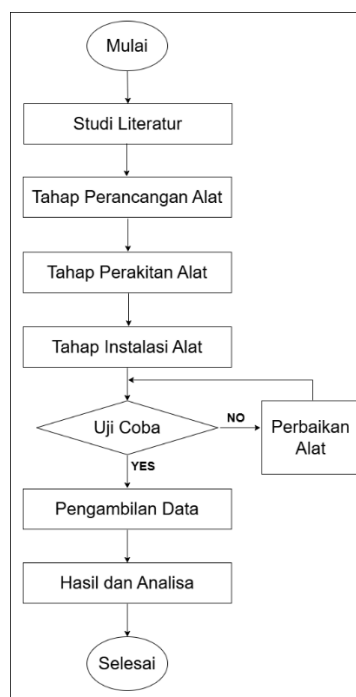
Selain metode kontrol, pemilihan perangkat pengendali juga menjadi faktor penting dalam perancangan sistem kontrol motor. Berdasarkan permasalahan diatas, agar alat ini bekerja diperlukan kontroler yang berfungsi untuk mengendalikan kecepatan putaran motor BLDC menggunakan mikrokontroler. Mikrokontroler adalah komputer mikro terkompresi yang diproduksi untuk mengontrol fungsi sistem tertanam di mesin perkantoran, robot, peralatan rumah tangga, kendaraan bermotor, dan sejumlah gadget lainnya [5]. Fungsi mikrokontroler antara lain yaitu sebagai otak atau pengendali dari rangkaian elektronik untuk suatu tujuan tertentu. ESP32 adalah mikrokontroler yang dikenalkan oleh *Espressif System* dan merupakan penerus dari mikrokontroler ESP8266. Mikrokontroler Esp32 memiliki keunggulan yaitu sistem berbiaya rendah, dan juga berdaya rendah dengan modul WiFi yang terintegrasi dengan chip mikrokontroler serta memiliki bluetooth dengan mode ganda dan fitur hemat daya menjadikannya lebih fleksibel [6].

Berdasarkan uraian tersebut, maka dilakukan perancangan dan pembangunan sistem untuk peningkatan efisiensi kontrol motor BLDC berbasis ESP32 dengan menggunakan algoritma PID.

Sistem ini menghasilkan pengendalian kecepatan motor yang lebih stabil, responsif, dan mampu beradaptasi terhadap perubahan kondisi sistem.

II. METODE

Informasikan Jenis penelitian yang digunakan dalam tugas akhir ini adalah penelitian rancang bangun (*engineering design*). Rancang bangun merupakan kegiatan menerjemahkan hasil analisa ke dalam bentuk paket perangkat lunak kemudian menciptakan sistem tersebut atau memperbaiki sistem yang sudah ada [7]. Penelitian ini berfokus pada perancangan, pembuatan, dan pengujian sistem kontrol motor *BLDC* berbasis mikrokontroler *ESP32* menggunakan algoritma *PID*. Tujuan dari penelitian ini adalah untuk meningkatkan efisiensi, kestabilan, dan respon kecepatan motor *BLDC* pada sepeda listrik. Tahap penelitian dapat dilihat pada gambar



Gambar 1. Flowchart Penelitian.

Tahap awal penelitian dimulai dengan menentukan topik, tujuan, serta permasalahan yang akan diteliti pada sistem kontrol motor *BLDC* berbasis *ESP32*. *Studi Literatur*, Pada tahap ini dilakukan pengumpulan referensi dari jurnal, buku, artikel, dan penelitian terdahulu yang berkaitan dengan motor *BLDC*, mikrokontroler *ESP32*, kontrol *PID*, sensor arus dan tegangan, serta sistem monitoring berbasis *IoT*. *Studi literatur* bertujuan untuk memperoleh dasar teori dan metode yang sesuai dengan penelitian. Tahap Perancangan Alat, Tahap ini meliputi perancangan sistem perangkat keras dan perangkat lunak, seperti desain wiring rangkaian, penentuan sensor, perancangan kontrol *PID*, pembuatan diagram blok, serta perancangan program *ESP32*. Tahap Perakitan Alat, Setelah desain selesai, dilakukan proses perakitan komponen menjadi satu sistem, meliputi pemasangan *ESP32*, sensor arus *ACS712*, sensor tegangan, throttle, dan koneksi ke motor *BLDC* serta *controller*. Sensor *ACS712* telah banyak digunakan untuk mengukur arus [8]. Metode dalam pembuatan sensor tegangan salah satunya adalah menggunakan pembagi tegangan (*Voltage Divider*), (*Voltage Divider*) merupakan sebuah rangkaian yang terdiri atas

dua buah resistor atau lebih, dirangkai secara seri dan kemudian dihubungkan ke suatu sumber tegangan *Direct Current (DC)* [9]. Sensor tegangan ini dirancang untuk pengukuran pada tegangan *DC*. Dalam memperoleh nilai tegangan yang stabil dengan merancang sensor tegangan. Tahap Instalasi Alat, Pada tahap ini alat yang telah dirakit dipasang dan dihubungkan pada sistem sepeda listrik sehingga seluruh komponen dapat bekerja sesuai fungsi masing-masing. Sistem kemudian diuji untuk mengetahui apakah alat sudah bekerja dengan baik. Pengujian dilakukan terhadap pembacaan sensor, control *PWM*, respon motor, kestabilan *PID*, serta tampilan monitoring data. Perbaikan Alat Jika pada tahap pengujian ditemukan kesalahan atau sistem belum bekerja sesuai tujuan, maka dilakukan perbaikan perangkat keras maupun perangkat lunak. Setelah diperbaiki, sistem diuji kembali hingga bekerja dengan baik. Pengambilan Data Setelah sistem berjalan normal, dilakukan pengambilan data penelitian, seperti nilai arus, tegangan, kecepatan motor, *duty cycle PWM*, waktu respon, dan *overshoot* sistem *PID*. Hasil dan Analisa, Data hasil pengujian dianalisis untuk mengetahui performa sistem kontrol *PID* terhadap motor *BLDC*. Analisis dilakukan dengan membandingkan hasil pengujian untuk mengetahui tingkat kestabilan dan efisiensi sistem. Tahap akhir penelitian berupa penarikan kesimpulan berdasarkan hasil pengujian dan analisis yang telah dilakukan. Perangkat lunak sistem dikembangkan menggunakan lingkungan *Arduino IDE* dengan bahasa pemrograman *C++*. Bahasa Pemrograman *C++* adalah bahasa pemrograman tingkat tinggi yang biasa digunakan untuk pengembangan perangkat lunak, mulai dari aplikasi dekstop hingga permainan di komputer dan sistem operasi [10]. Fokus utama pemrograman terletak pada implementasi algoritma *PID* diskrit untuk meminimalkan *error* antara kecepatan aktual dan kecepatan acuan.

Persamaan matematika *PID* diskrit yang ditanamkan pada program *ESP32* adalah sebagai berikut:

1. Persamaan *PID* Kontinu (Domain Waktu)

Persamaan *PID* umum:

$$u(t) = K_p \cdot e(t) + K_i \int e(\tau) d\tau + K_d \cdot de(t)/dt \quad (1)$$

$u(t)$ = sinyal output kontrol (nilai DAC/PWM)

$e(t)$ = error = setpoint – kecepatan aktual

K_p = konstanta proporsional

K_i = konstanta integral

K_d = konstanta derivatif

t = waktu (detik)

τ = Variabel integrasi

2. *PID* Diskrit (*Implementasi* Mikrokontroler *ESP32*)

Persamaan diskrit untuk pemrograman:

$$u[n] = K_p \cdot e[n] + K_i \cdot \sum e[k] \cdot \Delta t + K_d \cdot (e[n] - e[n-1]) / \Delta t \quad (2)$$

$u[n]$ = output *PID* pada sampling ke- n

$e[n]$ = error saat ini

$e[n-1]$ = error sampling sebelumnya

Δt = periode sampling

$\sum e[k] \cdot \Delta t$ = akumulasi integral

3. Komponen *PID* Terpisah

a. Komponen Proporsional (P):

$$P = K_p \cdot e[n] \quad (3)$$

b. Komponen Integral (I):

$$I = K_i \cdot \sum e[k] \cdot \Delta t \quad (4)$$

c. Komponen Derivatif (D):

$$D = K_d \cdot (e[n] - e[n-1]) / \Delta t \quad (5)$$

d. Output Total PID:

$$u[n] = P + I + D \quad (6)$$

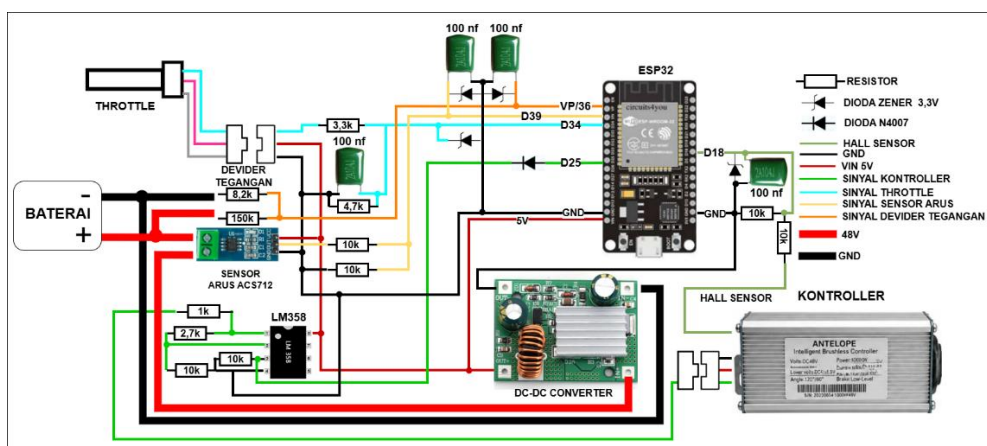
Pengujian dilakukan secara langsung menggunakan unit sepeda listrik roda 14 inci. Skenario pengujian dibagi menjadi dua tahap utama:

1. Pengujian Akurasi Sensor Kecepatan: Membandingkan hasil pembacaan kecepatan dari sensor *Hall effect* yang diproses *ESP32* dengan nilai pembandingan dari *GPS Speedometer* berbasis *smartphone* pada rentang kecepatan 6\12 { km/jam }.
2. Pengujian Kinerja Respon Kendali PID: Mengamati kemampuan sistem dalam mencapai dan mempertahankan kecepatan berdasarkan perubahan variabel *setpoint*. Parameter tanggapan peralihan (*transient response*) yang dicatat dan dianalisis meliputi: *Rise Time tr*: Waktu yang dibutuhkan sistem untuk mencapai rentang kecepatan *setpoint* pertama kali. *Overshoot*: Lonjakan nilai kecepatan maksimum yang melewati garis *setpoint*, dinyatakan dalam persen (%). *Error Steady State (ss)*: Selisih fluktuasi nilai kecepatan aktual dengan *setpoint* saat kondisi sistem telah tenang/stabil.

III. HASIL DAN PEMBAHASAN

A. Data Produk Hasil Pengembangan

Sistem yang dikembangkan merupakan perangkat keras (hardware) dan perangkat lunak (firmware) yang terintegrasi. Secara keseluruhan, produk akhir terdiri atas: (1) rangkaian elektronik berbasis mikrokontroler *ESP32*, (2) sensor Hall efek pada motor belakang (27 PPR), (3) sensor tegangan baterai dengan pembagi tegangan $R_1=150k\Omega/R_2=8,2k\Omega$, (4) sensor arus ACS712-20A, (5) DAC output GPIO25 ke driver throttle motor melalui op-amp *LM358*, serta (6) firmware yang diunggah pada *ESP32* dan dashboard pemantauan melalui aplikasi *Blynk*.



Gambar 2. Desain Hardware Sistem

Produk dirancang untuk sepeda listrik bermotor hub (hub motor) beroda 14 inci. Sistem mampu mengukur kecepatan aktual hingga 12 km/jam dengan proteksi spike noise Hall sensor,

serta menampilkan data secara real-time ke Serial Monitor *Arduino IDE* maupun *Serial Plotter* untuk keperluan tuning *PID*.

Tabel 1. Spesifikasi Hardware Sistem

No.	Komponen/Parameter	Spesifikasi
1	Mikrokontroler	ESP32 (dual-core 240 MHz, WiFi built-in)
2	Sensor Kecepatan	Hall sensor motor belakang, 27 pulsa/putaran (PPR)
4	Keliling Roda	1,117 m (roda belakang 14 inci)
5	Sensor Tegangan	Pembagi tegangan $R1=150k\Omega$, $R2=8,2k\Omega$, V_{bat} referensi = 49,0 V
6	Sensor Arus	ACS712-20A, sensitivitas 50 mV/A, V_{ref} dikalibrasi otomatis
7	DAC Output Throttle	GPIO25 ESP32, range $DAC_MIN=62$ – $DAC_MAX=255$
8	Driver Throttle	Op-amp LM358 penguat sinyal DAC ke motor controller
9	Rentang Gas ADC	$GAS_MIN=430$ – $GAS_MAX=2800$, cutoff di 500
12	Kick-Start	Output DAC=150 selama 1000 ms saat sepeda dari kondisi diam
13	Timeout Kecepatan Nol	3 detik tanpa pulsa Hall → RPM & kecepatan = 0
14	Parameter PID Default	$K_p=3,9$ $K_i=0,05$ $K_d=0,01$
15	Rentang Output PID	$PID_MIN=80$ – $PID_MAX=255$
16	Konektivitas	WiFi 2,4 GHz + Blynk IoT (Virtual Pin V1, V3–V8)
17	Penyimpanan NVS	Parameter K_p , K_i , K_d , rasio volt tersimpan di flash ESP32
18	Frekuensi Telemetry Blynk	500 ms (2 Hz)

Tabel 1 merupakan arsitektur rancang bangun sistem peningkatan efisiensi kontrol motor BLDC berbasis ESP32 ini dirancang dengan presisi tinggi.

B. Uji Fungsional Komponen

Tabel 2. Hasil Uji Komponen

No.	Komponen Diuji	Indikator Keberhasilan	Hasil
1	Hall Sensor (GPIO18)	Pulsa terdeteksi di ISR, RPM terbaca di Serial Monitor	Berhasil
2	ADC Sensor Tegangan (GPIO36)	Tegangan baterai terbaca $\pm 0,5$ V dari multimeter	Berhasil
3	ADC Sensor Arus ACS712 (GPIO39)	Arus terbaca 0 A saat motor mati, kalibrasi Vref otomatis	Berhasil
4	ADC Throttle Gas (GPIO34)	Nilai ADC berubah proporsional sesuai posisi gas	Berhasil
5	DAC Throttle Output (GPIO25)	Motor merespons perubahan DAC 62–255	Berhasil
6	WiFi + Blynk	Terhubung dalam <10 detik, failsafe offline ke Mode Manual	Berhasil
7	NVS Flash (Preferences)	Kp/Ki/Kd tersimpan dan terbaca kembali setelah restart	Berhasil
8	Serial Plotter	1 garis Output PID	Berhasil

Sebelum dilakukan pengujian sistem kontrol *PID* secara menyeluruh, terlebih dahulu dilakukan pengujian fungsionalitas pada masing-masing komponen penyusun (*hardware dan software*). Pengujian unit ini bertujuan untuk memastikan seluruh sensor, aktuator, jalur *General-Purpose Input/Output* (GPIO) *ESP32*, serta modul komunikasi dapat beroperasi sesuai dengan indikator keberhasilan yang telah ditetapkan. Rangkuman hasil pengujian unit komponen disajikan pada Tabel 2.

C. Uji Performa Kendali PID

Pengujian performa dilakukan dengan mengaktifkan Mode *PID* (*modeKendali*=1) melalui aplikasi *Blynk*. Target kecepatan divariasikan pada 6, 8, 10, dan 12 km/jam. Data kecepatan aktual, target, error, dan output DAC direkam menggunakan *Serial Plotter Arduino IDE*. Parameter *PID* yang digunakan adalah $K_p=3,9$, $K_i=0,05$, dan $K_d=0,01$ yang merupakan nilai hasil tuning bertahap.

Tabel 3. Hasil Uji Performa Kendali

No.	Target (km/jam)	Kecepatan Aktual Steady State (km/jam)	Error Steady State (km/jam)	Rise Time (detik)	Overshoot (%)
1	6	8,93	-2,93	0,45	58,82
2	8	7,40	0,60	1,02	1,33
3	10	8,25	1,75	1,93	0,00
4	12	5,64	6,36	99,99	0,00

Pengujian performa sistem kendali lingkaran tertutup (*closed-loop*) dilakukan untuk mengevaluasi ketangguhan algoritma *PID* diskrit yang ditanamkan pada *ESP32* dalam mempertahankan kecepatan konsisten. Kondisi pengujian ini tidak melibatkan beban riil kendaraan dan pengemudi. Berdasarkan hasil parameter secara bertahap, konstanta kontrol ditetapkan pada nilai $K_p = 3,9$, $K_i = 0,05$, dan $K_d = 0,01$. Analisis karakteristik grafik respon

transient dari data Tabel 3 dijabarkan melalui parameter berikut: 1. Kecepatan *Aktual* dan *Error Steady State* (e_{ss}). Berdasarkan data hasil uji, sistem menunjukkan performa yang bervariasi pada setiap target kecepatan. Pada target 8 km/jam, kecepatan steady state tercatat 7,40 km/jam dengan error 0,60 km/jam. Pada target 10 km/jam, kecepatan steady state adalah 8,25 km/jam dengan error 1,75 km/jam. Pada target 6 km/jam, sistem overshoot menyebabkan kecepatan steady state mencapai 8,93 km/jam (error -2,93 km/jam). Pada target 12 km/jam, sistem mengalami saturasi sehingga hanya mencapai 5,64 km/jam dengan error 6,36 km/jam. Analisis Kontrol: Variasi *error steady state* yang cukup besar ini menunjukkan bahwa nilai $K_i = 0,05$ belum cukup kuat untuk menghilangkan seluruh error statis, terutama pada beban tinggi. Peningkatan nilai K_i pada iterasi tuning berikutnya direkomendasikan untuk memperbaiki akurasi steady state. 2. Analisis Rise Time. Rise time tercepat dicapai pada target 6 km/jam (0,45 detik), sedangkan target 8 dan 10 km/jam membutuhkan waktu 1,02 dan 1,93 detik. Pada target 12 km/jam, rise time mencapai 99,99 detik yang mengindikasikan saturasi motor BLDC akibat keterbatasan torsi pada beban 125 kg. Analisis Kontrol: Nilai $K_p = 3,9$ memberikan respons awal yang cepat pada kecepatan rendah, namun tidak cukup untuk memaksa motor mencapai setpoint 12 km/jam. 3. Analisis Overshoot. Overshoot tertinggi terjadi pada target 6 km/jam sebesar 58,82%, menunjukkan bahwa pada kecepatan rendah, konstanta $K_d = 0,01$ belum mampu meredam lonjakan secara efektif. Sebaliknya, target 10 dan 12 km/jam menghasilkan overshoot 0,00%, menandakan respons yang lebih terkontrol pada kecepatan sedang. Secara keseluruhan, performa sistem dengan parameter $K_p = 3,9$, $K_i = 0,05$, dan $K_d = 0,01$ menunjukkan bahwa sistem mampu bekerja dengan baik pada rentang 8–10 km/jam, namun masih memerlukan optimasi parameter PID lebih lanjut agar dapat beroperasi secara stabil pada seluruh rentang target kecepatan.

IV. KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem kontrol kecepatan motor BLDC berbasis ESP32 menggunakan algoritma PID, dapat disimpulkan sebagai berikut. Sistem kontrol PID berhasil diimplementasikan secara diskrit pada mikrokontroler ESP32 dengan parameter $K_p = 3,9$, $K_i = 0,05$, dan $K_d = 0,01$, untuk mengendalikan kecepatan motor BLDC pada rentang target 6–12 km/jam. Performa terbaik dicapai pada target 8 km/jam dengan kecepatan steady state 7,40 km/jam, rise time 1,02 detik, dan overshoot 1,33%, serta pada target 10 km/jam dengan overshoot 0,00% dan rise time 1,93 detik. Namun demikian, sistem masih memiliki keterbatasan berupa overshoot tinggi (58,82%) pada target 6 km/jam dan saturasi motor pada target 12 km/jam dengan error steady state 6,36 km/jam, sehingga diperlukan optimasi parameter PID lebih lanjut. Seluruh komponen fungsional—sensor Hall, ADC tegangan dan arus, throttle DAC, Wi-Fi, Blynk IoT, serta penyimpanan NVS—berhasil bekerja sesuai spesifikasi. Sistem juga terbukti lebih efisien dibanding mode manual, dengan mekanisme failsafe otomatis yang menjamin keamanan berkendara saat koneksi terputus.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Universitas Nusantara PGRI Kediri, khususnya Program Studi Teknik Elektronika, yang telah memberikan fasilitas, bimbingan, dan dukungan selama penelitian ini berlangsung. Terima kasih pula kepada seluruh dosen pembimbing dan rekan-rekan yang telah membantu dalam proses perancangan, pengujian, serta penyempurnaan

sistem. Penelitian ini juga tidak terlepas dari dukungan keluarga yang senantiasa memberikan semangat dan doa.

DAFTAR PUSTAKA

- [1] E. Radwitya and S. I. Alkadri, "Pemodelan Matematis dan Validasi Eksperimental Sistem Kontrol Motor Brushless DC Menggunakan Fuzzy Logic," *J. Simetrik*, vol. 15, no. 2, pp. 128–137, 2025, doi: 10.31959/js.v15i2.3508.
- [2] N. F. Maula, S. Yahya, and S. M. Ilman, "Perancangan dan Simulasi Kendali Kecepatan Motor BLDC dengan Metode PID Menggunakan Teknik Tanpa Sensor Berbasis Simulink MatLab," *Pros. Ind. Res. Work. Natl. Semin.*, vol. 15, no. 1, pp. 461–465, 2024, doi: 10.35313/irwns.v15i1.6205.
- [3] Sutedjo, B. P. P. Putih, O. A. Qudsi, and L. S. Mahendra, "Alat Penstabil Kecepatan Motor BLDC Menggunakan Kontrol PID," *Emit. J. Tek. Elektro*, vol. 22, no. 2, pp. 134–140, 2022, doi: 10.23917/emitor.v22i2.19384.
- [4] A. R. Hidayat, A. D. Risdhayanti, Fitri, D. A. Permatasari, and W. T. Wahono, "Pemodelan Identifikasi Sistem untuk Pengaturan Kecepatan Motor DC dengan Kontrol PID," *J. Elektron. dan Otomasi Ind.*, vol. 11, no. 1, pp. 253–262, 2024, doi: 10.33795/elkolind.v11i1.5409.
- [5] I. M. Adrian Pramuditya, I. G. A. P. Raka Agung, and P. Rahardjo, "Rancang Bangun Alat Uji Periferal Esp32 Devkit V1 - Doit 30 Pin," *J. SPEKTRUM*, vol. 10, no. 4, p. 340, 2023, doi: 10.24843/spektrum.2023.v10.i04.p39.
- [6] S. A. Arrahma and R. Mukhaiyar, "Pengujian Esp32-Cam Berbasis Mikrokontroler," *JTEIN*, vol. 4, no. 1, pp. 60–66, 2023, doi: 10.24036/jtein.v4i1.347.
- [7] R. Gunawan, A. M. Yusuf, and L. Nopitasari, "Rancang Bangun Sistem Presensi Mahasiswa Dengan Menggunakan Qr Code Berbasis Android," *Elkom J. Elektron. dan Komput.*, vol. 14, no. 1, pp. 47–58, 2021, doi: 10.51903/elkom.v14i1.369.
- [8] L. D. Rumpa, Y. A. Ambabunga, and M. Pineng, "Optimization of ACS712 Sensor Current Measurement in Solar Power System through Regression Modeling," *J. Appl. Informatics Comput.*, vol. 7, no. 2, pp. 198–201, 2023, doi: 10.30871/jaic.v7i2.6511.
- [9] S. Fuada *et al.*, "Analisis Rangkaian Pembagi Tegangan dan Perbandingan Hasil Simulasinya Menggunakan Simulator Offline," *Circuit J. Ilm. Pendidik. Tek. Elektro*, vol. 6, no. 1, p. 28, 2022, doi: 10.22373/crc.v6i1.11200.
- [10] Adawiyah Ritonga and Yahfizham Yahfizham, "Studi Literatur Perbandingan Bahasa Pemrograman C++ dan Bahasa Pemrograman Python pada Algoritma Pemrograman," *J. Tek. Inform. dan Teknol. Inf.*, vol. 3, no. 3, pp. 56–63, 2023, doi: 10.55606/jutiti.v3i3.2863.

Click or tap here to enter text.