JURITEK: Jurnal Ilmiah Teknik Mesin, Elektro dan Komputer

E-ISSN: 2809-0799
P-ISSN: 2809-0802

(Artikel Penelitian)

Implementasi Kontrol PID untuk Sistem Steering Swerve Drive Menggunakan Motor DC dan Sensor Magnetik AS5600

Faisal Candrasyah Hasibuan [1]*, Daffa Khusnureza [2]

[1] Program Studi S1 Teknik Komputer, Fakultas Teknik Elektro, Universitas Telkom, Bandung, Indonesia; email : faicanhasfcb@telkomuniversity.ac.id
[2] Program Studi S1 Teknik Komputer, Fakultas Teknik Elektro, Universitas Telkom, Bandung, Indonesia; email : daffaunu@student.telkomuniversity.ac.id

*** Korespondensi**: **Faisal Candrasyah Hasibuan**

Abstract: Swerve drive is a type of omnidirectional robot drive system that allows each wheel to rotate independently, enabling high maneuverability and precision, especially in confined or complex environments. This research focuses on the steering control of a swerve drive using a PID (Proportional-Integral-Derivative) controller applied to a DC motor with magnetic encoder feedback. The main challenge lies in achieving stable and accurate wheel orientation in response to target angles, despite potential mechanical backlash and external disturbances. The objective of this study is to design and implement a closed-loop PID control system capable of minimizing steady-state error, overshoot, and response time. The proposed method involves real-time angle control using an STM32 microcontroller, AS5600 magnetic encoder, and BTS7960 motor driver, with PID parameters tuned through trial-and-error. Experimental results were analyzed by recording transient response data via serial communication and visualizing them with Python and Matplotlib. The best PID parameters (Kp = 30, Ki = 0, Kd = 1300) yielded a delay time of 110 ms, rise time of 110 ms, peak time of 200 ms, settling time of 240 ms, and minimal overshoot (2.69%) without steady-state error. Mechanical analysis further confirmed effective torque and gear ratio utilization. The findings demonstrate that the implemented PID control system ensures accurate and stable steering angle regulation under real-time conditions. This research concludes that PID control with careful tuning is effective for swerve drive steering.

Keywords: AS5600 sensor; DC motor; PID control; STM32; swerve drive; trial-and-error tuning; wheel orientation.

Abstrak: *Swerve drive* merupakan sistem penggerak robot omnidirectional yang memungkinkan setiap roda berputar secara independen, sehingga robot dapat bermanuver dengan sangat fleksibel dan presisi, terutama pada ruang terbatas atau medan kompleks. Penelitian ini berfokus pada kendali sudut roda (steering) menggunakan kontrol PID (*Proportional-Integral-Derivative*) yang diterapkan pada motor DC dengan umpan balik dari sensor magnetik. Permasalahan utama adalah mencapai orientasi sudut roda yang stabil dan akurat terhadap setpoint, dengan mengatasi gangguan dari backlash mekanik dan dinamika sistem. Tujuan penelitian ini adalah dirancang dan diimplementasikan sistem kendali *loop* tertutup berbasis PID untuk meminimalkan *steady-state error*, *overshoot*, dan waktu respons. Metode yang digunakan melibatkan mikrokontroler STM32, sensor AS5600, dan driver motor BTS7960, dengan parameter PID ditentukan melalui metode *trial-and-error*. Pengujian dilakukan dengan merekam data respons transiens melalui komunikasi serial dan dianalisis menggunakan Python serta visualisasi dengan Matplotlib. Hasil terbaik diperoleh pada parameter PID Kp = 30, Ki = 0, dan Kd = 1300, menghasilkan *delay time* 110 ms, *rise time* 110 ms, *peak time* 200 ms, dan *settling time* 240 ms dengan *overshoot* minimal sebesar 2.69% tanpa *steady-state error*. Analisis mekanik mendukung efektivitas pemanfaatan torsi dan rasio gigi. Kesimpulannya, kendali PID yang diimplementasikan mampu mengatur sudut roda *swerve drive* secara akurat dan stabil.

Diterima: Mei 17, 2025
Direvisi: Mei 27, 2025
Diterima: Juni 29, 2025
Diterbitkan: Juli 2, 2025
Versi sekarang: Juli 15, 2025

Kata kunci: kendali PID; kontrol *loop* tertutup; motor DC; orientasi roda; sensor AS5600; STM32; *swerve drive*; *tuning trial-and-error*.

1. Pendahuluan

Swerve drive adalah salah satu konfigurasi sistem penggerak yang umum digunakan dalam robotika mobile untuk meningkatkan kemampuan manuver dan presisi gerakan [1], [2]. Sistem ini memungkinkan setiap roda untuk dikendalikan secara independen dalam hal kecepatan dan sudut orientasi, memungkinkan robot bergerak ke segala arah tanpa perlu memutar bodi utamanya. Keunggulan ini sangat berguna dalam aplikasi kompetisi robotika, kendaraan otomatis, serta sistem industri yang beroperasi di ruang terbatas atau lingkungan dinamis [3].

Berbagai pendekatan telah digunakan dalam implementasi kontrol pada sistem *swerve drive*, mulai dari kontrol terbuka berbasis PWM sederhana [4], kontrol berbasis logika *fuzzy* untuk pengambilan keputusan non-linear [5], hingga kendali berbasis model prediktif (*Model Predictive Control*) [6]. Namun, pendekatan-pendekatan tersebut memiliki keterbatasan dalam hal kompleksitas komputasi, kesulitan dalam tuning, dan ketergantungan pada model sistem yang akurat.

Kontrol PID (*Proportional-Integral-Derivative*) masih menjadi pendekatan yang banyak digunakan untuk pengaturan posisi dan kecepatan motor karena kesederhanaannya, efisiensi komputasi, dan kinerjanya yang cukup baik untuk sistem linier atau mendekati linier [7], [8]. Meskipun demikian, *tuning* parameter PID pada sistem mekanik kompleks seperti *swerve drive* dapat menjadi tantangan, terutama dalam menghadapi dinamika non-linear, gesekan, dan backlash pada roda penggerak [9].

Masalah utama yang diangkat dalam penelitian ini adalah bagaimana merancang sistem kendali PID yang mampu mengatur orientasi sudut roda (*steering*) pada *swerve drive* secara presisi dan stabil, dengan memperhitungkan keterbatasan mekanis dan keterlambatan sistem. Selain itu, diperlukan pendekatan tuning yang praktis namun tetap efektif agar implementasi dapat dilakukan secara langsung pada perangkat keras tanpa risiko kerusakan.

Sebagai solusi, penelitian ini mengusulkan pendekatan tuning PID berbasis *trial-and-error* yang dievaluasi secara kuantitatif melalui analisis respon transien sistem terhadap perubahan *set point*. Implementasi dilakukan menggunakan motor DC PG36, sensor magnetik AS5600 sebagai pengukur sudut absolut [10], serta mikrokontroler STM32F4 yang menjalankan logika kendali secara real-time [11].

Kontribusi utama dari penelitian ini adalah:

1. Desain dan implementasi sistem kendali PID tertutup untuk sudut orientasi roda *swerve drive* berbasis mikrokontroler;
2. Evaluasi kinerja PID melalui parameter *rise time*, *settling time*, *overshoot*, dan *steady-state error*;
3. Rekomendasi parameter PID optimal untuk pengendalian sudut pada konfigurasi sistem mekanik yang serupa.

Sisa *paper* ini disusun sebagai berikut. Bagian 2 dibahas desain sistem kendali dan komponen mekanik. Bagian 3 disajikan metode pengujian serta hasil analisis tuning PID dan performa sistem. Bagian 4 dimuat evaluasi menyeluruh terhadap sistem dan tuning parameter. Terakhir di Bagian 5 disimpulkan hasil dan diberikan saran untuk pengembangan lebih lanjut.

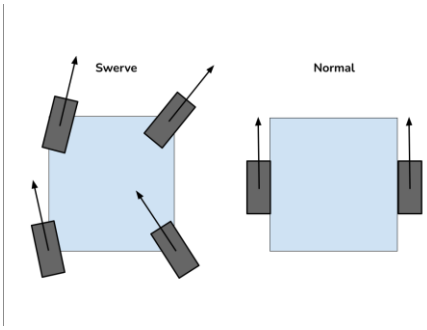
2. Tinjauan Literatur

Di bagian ini diulas literatur dan teori fundamental yang relevan dengan perancangan sistem kendali untuk *swerve drive*. Tinjauan ini mencakup tiga area utama: arsitektur dan kinematika sistem *swerve drive*, paradigma kendali PID sebagai metode kontrol yang dipilih, dan berbagai metode tuning parameter PID yang ada beserta kelebihan dan kekurangannya.

2.1. Sistem Penggerak *Swerve Drive*

Sistem penggerak *swerve drive* adalah salah satu jenis sistem penggerak holonomik (atau omnidireksional) yang memberikan mobilitas tingkat tinggi pada robot bergerak [12]. Berbeda dari sistem penggerak diferensial (*differential drive*) yang memerlukan rotasi sasis untuk mengubah arah gerak [13], atau sistem mecanum yang rentan terhadap selip pada permukaan

tidak rata, *swerve drive* menggunakan modul roda yang dapat berputar pada sumbu vertikal (*steering*) dan berotasi pada sumbu horizontal (*driving*) secara independen. Kemampuan ini memisahkan gerak translasi dan rotasi robot, memungkinkan manuver yang kompleks seperti bergerak ke samping sambil mempertahankan orientasi ke depan, atau berputar di tempat dengan radius nol.



Gambar 1. Visualisasi perbandingan antara *differential drive* dengan *swerve drive*

Secara matematis, untuk mengoordinasikan semua modul roda, diperlukan pemodelan kinematika, khususnya kinematika terbalik (*inverse kinematics*). Kinematika terbalik berfungsi untuk menerjemahkan vektor kecepatan robot yang diinginkan (terdiri dari komponen kecepatan translasi vx, vy, dan kecepatan rotasi (ω) menjadi sepasang nilai setpoint untuk setiap modul roda, yaitu sudut kemudi (θi) dan kecepatan putar roda (ωi) [2]. Kompleksitas dalam sinkronisasi dan akurasi untuk mencapai setiap setpoint inilah yang menuntut adanya sistem kendali *loop* tertutup yang andal pada setiap modul, terutama pada *steering actuator*. Berbagai implementasi praktis, terutama di arena robotika kompetitif seperti FIRST Robotics Competition (FRC), telah menunjukkan bahwa keberhasilan sistem *swerve drive* sangat bergantung pada presisi dan kecepatan sistem kendali posisi motor kemudinya [3].

2.2 Paradigma Kendali *Proportional-Integral-Derivative* (PID)

Kendali PID adalah algoritma umpan balik *loop* tertutup yang paling banyak digunakan dalam aplikasi industri dan robotika karena efektivitas, kesederhanaan konseptual, dan keandalannya [7]. Algoritma ini bekerja dengan terus menghitung nilai *error* (e(t)) sebagai selisih antara *setpoint* yang diinginkan dan variabel proses yang diukur, kemudian menerapkan koreksi berdasarkan tiga term matematis: *proportional*, *integral*, dan *derivatif*.

Persamaan kanonikal untuk pengendali PID dalam domain waktu kontinu adalah:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)\ d\tau + K_d \cdot \frac{d}{dt} e(t) \quad (1)$$

dengan keterangan:

- *Term Proportional* ($K_p e(t)$): Memberikan aksi kontrol yang sebanding dengan *error* saat ini. Semakin besar *error*, semakin besar pula sinyal koreksi yang diberikan. Komponen ini bertanggung jawab untuk mempercepat respons sistem mencapai *setpoint*.
- *Term Integral* ($K_i \int e(\tau) d\tau$): Mengakumulasi *error* dari waktu ke waktu. Tujuannya adalah untuk menghilangkan *steady-state error*, yaitu *error* sisa yang tidak dapat diatasi oleh term proporsional saja. Namun, term ini dapat menyebabkan masalah *integral windup* [14] jika tidak dikelola dengan baik, di mana akumulasi *error* yang besar saat terjadi perubahan *setpoint* yang drastis menyebabkan *overshoot* yang signifikan.
- *Term Derivatif* ($K_d \frac{de(t)}{dt}$): Bekerja berdasarkan laju perubahan *error*. Term ini bersifat prediktif atau antisipatif, berfungsi untuk meredam respons sistem saat mendekati *setpoint*, sehingga mengurangi *overshoot* dan osilasi.

2.3 Metode *Tuning* Parameter PID

Kinerja sebuah pengendali PID sangat ditentukan oleh nilai konstantanya: Kp, Ki, dan Kd. Proses untuk menemukan nilai optimal dari ketiga parameter ini disebut *tuning*. Terdapat berbagai metode *tuning* yang telah dikembangkan, mulai dari metode heuristik hingga analitis.

1. **Metode Heuristik Ziegler-Nichols (Z-N):** Metode ini adalah salah satu teknik *tuning* klasik yang paling terkenal [15]. Prosedurnya menggunakan peningkatan gain proporsional (Kp) hingga sistem mencapai kondisi osilasi yang stabil (osilasi marginal). Gain pada titik ini, yang disebut *ultimate gain* (Ku), dan periode osilasinya (Tu) kemudian digunakan dalam sebuah tabel untuk menentukan nilai Kp, Ki, dan Kd. Kelemahan utama metode ini, dan menjadi alasan mengapa ia dihindari dalam penelitian ini, adalah prosesnya yang memaksa sistem untuk berosilasi secara agresif. Pada sistem mekanis dengan gir atau transmisi, osilasi semacam itu dapat menyebabkan keausan yang cepat atau bahkan kerusakan permanen.
2. **Metode Trial-and-Error (Manual):** Metode ini, meskipun kurang formal, sangat umum digunakan dalam praktik karena lebih aman untuk perangkat keras [16]. Prosesnya bersifat intuitif: Kp dinaikkan terlebih dahulu untuk mendapatkan respons yang cepat, kemudian Kd ditambahkan untuk mengurangi *overshoot*, dan terakhir Ki disesuaikan untuk menghilangkan *steady-state error* jika ada. Meskipun bergantung pada pengalaman operator, metode ini memungkinkan pengamatan langsung terhadap respons sistem dan penyesuaian yang cermat untuk menghindari perilaku yang tidak diinginkan. Penelitian ini mengadopsi pendekatan ini sebagai metode yang pragmatis dan aman.
3. **Metode Tuning Adaptif dan Berbasis AI:** Seiring perkembangan teknologi, metode *tuning* yang lebih canggih telah muncul. Pendekatan seperti logika fuzzy, jaringan saraf tiruan, dan algoritma genetika dapat digunakan untuk men-*tuning* parameter PID secara otomatis dan adaptif terhadap perubahan kondisi sistem [5]. Metode ini menawarkan performa yang lebih *robust*, namun datang dengan kesenjangan implementasi yang signifikan, yaitu kebutuhan akan daya komputasi yang lebih besar dan kompleksitas perancangan yang lebih tinggi, sehingga kurang sesuai untuk aplikasi yang memprioritaskan kesederhanaan dan implementasi cepat pada mikrokontroler berbiaya rendah.

3. Metode

Penelitian ini menerapkan metode kontrol PID berbasis mikrokontroler STM32F407VGT6 untuk mengendalikan posisi sudut roda swerve drive menggunakan motor DC PG36 dan sensor magnetik AS5600. Langkah-langkah dalam metode ini mencakup akuisisi data posisi roda, perhitungan error, pengolahan sinyal kontrol oleh pengendali PID, hingga penggerakan aktuator berdasarkan sinyal kontrol.

3.1. Prosedur Penelitian

Prosedur eksperimen dilakukan dalam beberapa tahapan sebagai berikut:

1. Perancangan Sistem
   - Perakitan sistem mekanik (motor PG36, gear, sensor AS5600).
   - Pembuatan skematik dan pengkabelan sistem.
   - Pemrograman mikrokontroler dengan algoritma PID.
2. Akusisisi Data Sensor
   - Kalibrasi AS5600.
   - Pengambilan data posisi roda.
3. Pengujian Sistem
   - Memasukkan nilai kp, ki, dan kd secara iteratif,
   - Mencatat dan membandingkan respon sistem terhadap berbagai parameter.
4. Analisis
   - Interpretasi hasil pengujian dan penarikan kesimpulan.

3.2. Algoritma Sistem *Swerve Drive*

Penelitian ini diterapkan metode kontrol PID berbasis mikrokontroler STM32F407VGT6 untuk mengendalikan posisi sudut roda sistem *swerve drive*. Sistem ini

menggunakan motor DC PG36 untuk aktuator penggerak dan sensor magnetik AS5600 sebagai umpan balik posisi sudut roda. Langkah-langkah dalam metode ini mencakup:

1. Akuisisi data input kecepatan dari pengguna berupa **vx, vy,** dan **ω** (kecepatan translasi dan rotasi);
2. Perhitungan kinematika swerve drive untuk mengubah input kecepatan menjadi setpoint berupa sudut dan kecepatan target (**θ_target** dan **v_target**) untuk masing-masing modul roda;
3. Akuisisi data posisi sudut aktual roda menggunakan sensor AS5600;
4. Perhitungan error antara nilai setpoint sudut dengan posisi aktual;
5. Pengolahan sinyal kontrol menggunakan algoritma PID untuk menghasilkan sinyal PWM yang disesuaikan;
6. Penggerakan aktuator motor DC berdasarkan sinyal kendali tersebut sehingga roda dapat mencapai orientasi sudut yang diinginkan dengan cepat dan stabil.

Pendekatan ini memungkinkan kontrol posisi sudut roda secara presisi berdasarkan input vektor kecepatan dari pengguna, yang diproses melalui transformasi kinematika dan dikendalikan secara real-time menggunakan pengendali PID.

Algoritma 1. Kinematika Swerve Drive

MASUKAN:

1. **vx, vy, ω**: kecepatan translasi dan rotasi robot
2. **L, W**: dimensi robot (jarak antar roda depan-belakang dan kiri-kanan)

KELUARAN: **θ_target[i], v_target[i]:** sudut dan kecepatan target untuk tiap roda **i ∈ {FL, FR, BL, BR}**

```
FUNCTION compute_swerve_kinematics(vx, vy, omega):
    CALCULATE R = sqrt(L^2 + W^2)

    FOR each wheel i in {FL, FR, BL, BR} DO
        IF i == FL THEN
            vx_i = vx - omega * (W / R)
            vy_i = vy + omega * (L / R)
        ELSE IF i == FR THEN
            vx_i = vx + omega * (W / R)
            vy_i = vy + omega * (L / R)
        ELSE IF i == BL THEN
            vx_i = vx - omega * (W / R)
            vy_i = vy - omega * (L / R)
        ELSE IF i == BR THEN
            vx_i = vx + omega * (W / R)
            vy_i = vy - omega * (L / R)
        END IF

        theta_i = atan2(vy_i, vx_i)
        v_i = sqrt(vx_i^2 + vy_i^2)
    END FOR
END FUNCTION
```

Algoritma 2. Kontrol PID

MASUKAN:

- **θ_target[i]:** hasil dari Algoritma 1
- **θ_actual[i]:** data sensor dari encoder dan magnet sensor
- Parameter PID: **Kp, Ki, Kd** untuk steering dan driving

KELUARAN: **θ_target[i], v_target[i]:** sudut dan kecepatan target untuk tiap roda

```
FUNCTION PID_Controller(theta_setpoint, theta_current):
    error = theta_setpoint - theta_current
    integral = integral + error * dt
    derivative = (error - previous_error) / dt

    output = Kp * error + Ki * integral + Kd * derivative
```

```
    previous_error = error

    SEND output sebagai sinyal PWM ke motor
END FUNCTION
```

3.3. Pemrosesan Data

3.3.1. Kinematika Swerve Drive

Kinematika swerve drive dalam bentuk inverse digunakan untuk menghitung kecepatan dan arah gerak masing-masing roda berdasarkan kecepatan translasi robot v_x, v_x, dan kecepatan rotasi ω. Hubungan tersebut diturunkan dari persamaan:

$$\overrightarrow{v_{\text{wheel}_i}} = \overrightarrow{v_{\text{robot}}} + \overrightarrow{\omega_{\text{robot}}} \times \overrightarrow{r_{\text{robot}\to\text{wheel}_i}} \tag{2}$$

Dengan:

$$\overrightarrow{v_{\text{robot}}} = v_x\hat{\imath} + v_y\hat{\jmath} \tag{3}$$

$$\overrightarrow{\omega_{\text{robot}}} = \omega\hat{k} \tag{4}$$

$$\overrightarrow{r_{\text{robot}\to\text{wheel}_i}} = r_{ix}\hat{\imath} + r_{iy}\hat{\jmath} \tag{5}$$

Maka untuk setiap roda didapatkan:

$$\vec{v_i} = \left(v_x - \omega r_{iy}\right)\hat{\imath} + \left(v_y + \omega r_{ix}\right)\hat{\jmath} \tag{6}$$

Dengan memisahkan komponen $(\hat{\imath})$ dan $(\hat{\jmath})$, maka diperoleh:

$$v_{ix} = v_x - \omega r_{iy} \tag{7}$$

$$v_{iy} = v_y + \omega r_{ix} \tag{8}$$

Untuk keempat roda, kita rangkai ke dalam bentuk matriks berikut:

$$\begin{bmatrix} v_{1x} \\ v_{1y} \\ v_{2x} \\ v_{2y} \\ v_{3x} \\ v_{3y} \\ v_{4x} \\ v_{5y} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -r_{1y} \\ 0 & 1 & r_{1x} \\ 1 & 0 & -r_{2y} \\ 0 & 1 & r_{2x} \\ 1 & 0 & -r_{3y} \\ 0 & 1 & r_{3x} \\ 1 & 0 & -r_{4y} \\ 0 & 1 & r_{4x} \end{bmatrix} \cdot \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{9}$$

Untuk mendapatkan besarnya kecepatan dan arah roda ke-i, digunakan:

$$v_i = \sqrt{v_{ix}^2 + v_{iy}^2} \tag{10}$$

$$\theta_i = \tan^{-1}\left(\frac{v_{iy}}{v_{ix}}\right) \tag{11}$$

3.3.2. Model Kontrol PID

Persamaan dasar dari kontrol PID yang digunakan pada sistem ini adalah:

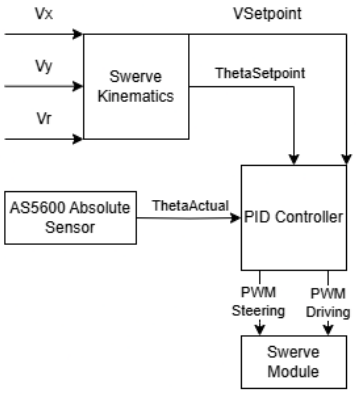
$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)\ d\tau + K_d \cdot \frac{d}{dt} e(t) \quad (12)$$

Dengan:

- $u(t)$: *sinyal kendali PWM*
- $e(t) = \theta_{target} - \theta_{\text{actual}}$:error posisi sudut
- K_p: *gain proporsional*
- K_i: *gain integral*
- K_d: *gain derivatif*

3.4. Diagram Blok

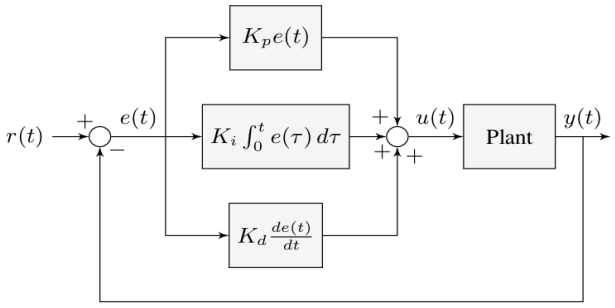
Berikut adalah diagram blok yang kemudian digunakan di sistem *swerve drive*:



Gambar 2 Diagram Blok Sistem Swerve Drive

Sistem terdiri dari beberapa komponen utama, yaitu input kecepatan, blok kinematika swerve, sensor sudut, pengendali PID, dan modul aktuator roda. Alur sistem bekerja sebagai berikut.

1. *Input* Vektor Kecepatan (Vx, Vy, Vr):
   Sistem menerima *input* berupa vektor kecepatan translasi sumbu-X (Vx), sumbu-Y (Vy), dan kecepatan rotasi (Vr). Data ini dapat berasal dari pengendali (*joystick*).
2. *Swerve Kinematics*:
   Modul kinematika akan memproses vektor kecepatan tersebut dan menghasilkan dua output untuk setiap modul roda:
   - θsetpoint: Sudut target roda yang dibutuhkan agar arah gerak sesuai dengan input kecepatan.
   - Vsetpoint: Kecepatan target roda agar menghasilkan besar gerak translasi/rotasi yang tepat.
3. Sensor AS5600:
   Untuk mendapatkan orientasi aktual sudut roda, digunakan sensor magnetik absolut AS5600. Sensor ini memberikan umpan balik nilai sudut aktual roda (θactual) dengan resolusi 12-bit melalui antarmuka I^2C.
4. Pengendali PID:
   Pengendali PID digunakan untuk mengatur sudut roda agar mendekati θ *set point* dengan cepat dan stabil. *Error* dihitung dari selisih antara sudut target dan aktual dengan diagram blok sebagai berikut.



Gambar 3 Diagram Blok Pengendali PID

Output dari PID digunakan untuk menghasilkan dua sinyal PWM:

- PWM *Steering*: untuk menggerakkan motor pengarah roda (steering).
- PWM *Driving*: untuk mengatur kecepatan motor penggerak (driving) berdasarkan Vsetpoint.

5. Modul *Swerve*:

Modul ini terdiri dari satu motor penggerak dan satu motor pengarah. Keduanya bekerja secara simultan untuk menggerakkan roda ke arah dan kecepatan yang ditentukan.

4. Hasil dan Pembahasan

4.1 . Pengujian Sistem

Pengujian dilakukan untuk mengevaluasi performa sistem kendali sudut roda swerve drive yang dikendalikan dengan algoritma PID. Tujuan dari pengujian ini adalah untuk menganalisis karakteristik respons transien [17] dari sistem terhadap perubahan setpoint sudut roda, serta menilai keefektifan tuning PID dalam mencapai stabilitas dan akurasi.

4.1.1. Parameter Evaluasi

Pengamatan performa sistem dilakukan berdasarkan parameter-parameter berikut:

- Delay Time (Td):
  Waktu yang dibutuhkan sistem sejak input diberikan hingga output mulai merespon secara signifikan (mencapai 10% dari nilai akhir).

$$T_d = t_{(y(t)=0.1\cdot y_{final})} \tag{13}$$

- Rise Time (Tr):
  Waktu yang dibutuhkan sistem untuk naik dari 10% ke 90% dari nilai akhir (steady state).

$$T_r = t_{(y(t)=0.9\cdot y_{final})} - t_{(y(t)=0.1\cdot y_{final})} \tag{14}$$

- Peak Time (Tp):
  Waktu yang dibutuhkan sistem untuk mencapai nilai maksimum (puncak respons).

$$T_p = t_{(y(t)=y_{peak})} \tag{15}$$

- Settling Time (Ts):
  Waktu yang dibutuhkan sistem hingga output memasuki dan tetap berada dalam rentang toleransi (biasanya ±2% dari nilai akhir).

$$T_s = t_{(y(t)\epsilon[0.98\cdot y_{final}, 1.02\cdot y_{final}])} \tag{16}$$

- Maximum Overshoot (Mp):
  Persentase kenaikan maksimum output sistem di atas nilai akhir (steady-state).

$$M_p = \left(\frac{y_{\text{peak}} - y_{\text{final}}}{y_{\text{final}}}\right) \times 100\% \tag{17}$$

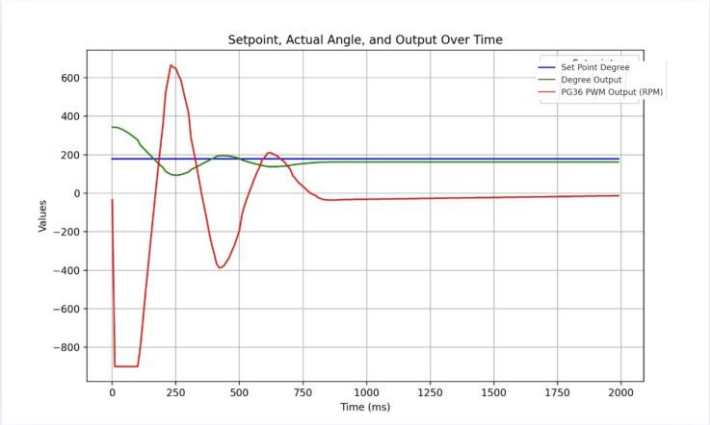
4.1.2. Metodologi Pengujian

1. Perekaman Data Serial:
   Sistem diuji dengan berbagai konfigurasi parameter PID. Data posisi sudut roda (θ) dicatat secara real-time melalui komunikasi serial dari STM32 menggunakan aplikasi Tera Term.
2. Analisis Grafis Menggunakan Python:
   Data hasil perekaman divisualisasikan menggunakan *Matplotlib* untuk mengidentifikasi karakteristik waktu transien dan kesalahan steady-state.
3. Variasi Parameter PID:
   Beberapa kombinasi parameter PID diuji:
   - Kp = 10, Ki = 0.01, Kd = 100
   - Kp = 30, Ki = 20, Kd = 500
   - Kp = 30, Ki = 0, Kd = 1300

4.1.3. Hasil Pengujian

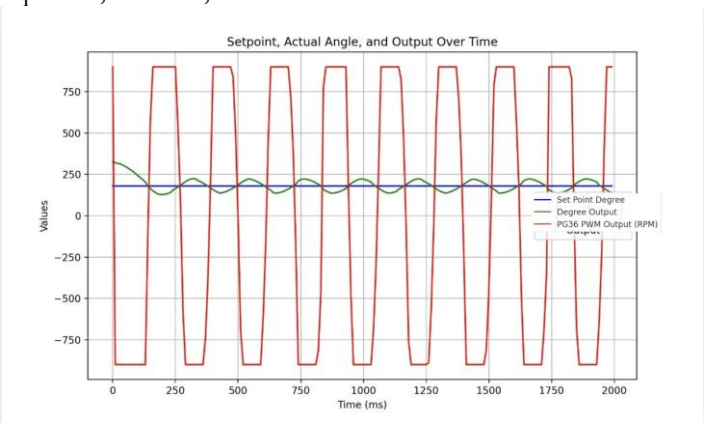
Hasil pengujian dari ketiga variasi parameter dapat dilihat dalam grafik berikut:

1. Kp = 10, Ki = 0.01, Kd = 100



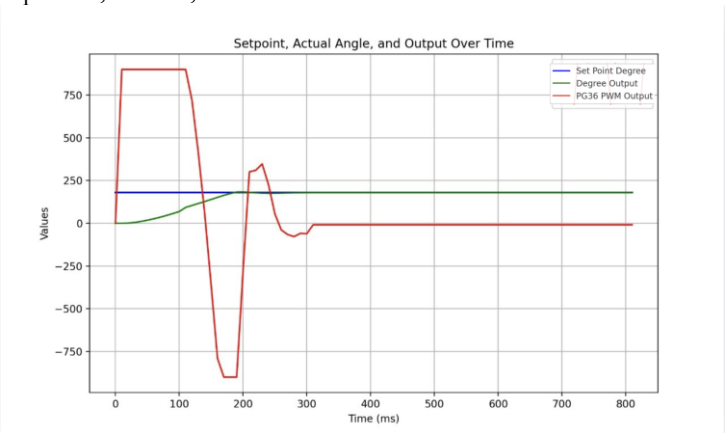
Gambar 4 Grafik Kp = 10, Ki = 0.01, Kd = 100

2. Kp = 30, Ki = 20, Kd = 500



Gambar 5 Grafik Kp = 30, Ki = 20, Kd = 500

3. Kp = 30, Ki = 0, Kd = 1300



Gambar 6 Grafik Kp = 30, Ki = 0, Kd = 1300

4.2. Analisis

Berdasarkan Hasil Pengujian, Tabel berikut merangkum hasil pengujian karakteristik respon transien untuk setiap kombinasi parameter PID yang diuji:

Tabel 1. Hasil Analisis Transient masing masing parameter PID

Setpoint	Kp	Ki	Kd	Delay Time (Td)	Rise Time (Tr)	Peak Time (Tp)	Settling Time (Ts)	Maximum Overshoot (Mp)
180	10	0.01	100	110 ms	160 ms	250 ms	820 ms	36.63%
180	30	20	500	110 ms	200 ms	220 ms	1990 ms	50%
180	30	0	1300	110 ms	110 ms	200 ms	240 ms	2.69%

Dari hasil pengujian yang dilakukan terhadap berbagai konfigurasi parameter PID, diperoleh bahwa sistem dengan konfigurasi Kp = 30, Ki = 0, dan Kd = 1300 memberikan performa terbaik. Konfigurasi ini menghasilkan waktu *settling* tercepat, yaitu 240 ms, serta *overshoot* yang minimum. Hal ini menunjukkan bahwa kombinasi proporsional yang cukup besar dan turunan yang tinggi mampu memberikan respons cepat sekaligus meredam osilasi secara efektif. Sebaliknya, konfigurasi dengan nilai Ki yang tinggi (Ki = 20) menyebabkan sistem mengalami osilasi berlebih dan waktu *settling* yang jauh lebih lama, yaitu 1990 ms, akibat akumulasi error yang terlalu agresif. Dengan demikian, pemilihan parameter PID yang tepat sangat mempengaruhi kualitas kontrol sudut pada sistem *steering swerve drive*.

6. Kesimpulan

Penelitian ini berhasil dirancang dan diimplementasikan sistem kendali PID untuk mengatur sudut roda pada sistem swerve drive menggunakan motor DC. Sistem dikembangkan dengan memanfaatkan sensor magnetik AS5600 sebagai umpan balik posisi dan motor PG36 yang dikendalikan melalui sinyal PWM oleh mikrokontroler STM32. Proses tuning dilakukan menggunakan metode *trial-and-error* untuk memperoleh nilai parameter PID yang optimal, mengingat keterbatasan mekanik sistem yang tidak memungkinkan penggunaan metode tuning otomatis seperti Ziegler-Nichols secara langsung.

Dari hasil pengujian, konfigurasi PID dengan parameter Kp = 30, Ki = 0, dan Kd = 1300 memberikan performa terbaik dibanding konfigurasi lainnya. Sistem dengan konfigurasi tersebut menunjukkan respon transien yang cepat dan stabil dengan settling time sebesar 240 ms, rise time sebesar 110 ms, dan overshoot sebesar 2.69%. Selain itu, steady-state error dapat diminimalkan secara signifikan bahkan tanpa kontribusi dari integral gain, karena sistem telah mencapai kestabilan dengan kombinasi proportional dan derivative gain saja. Hal ini menunjukkan bahwa pemilihan nilai Ki yang terlalu tinggi, seperti pada konfigurasi Ki = 20, justru menyebabkan osilasi berlebih dan memperpanjang waktu stabilisasi sistem hingga hampir 2 detik.

Namun demikian, penelitian ini masih memiliki beberapa keterbatasan. Tuning PID dilakukan secara manual tanpa pendekatan matematis sistematis atau pemodelan sistem yang

mendalam. Selain itu, pengujian hanya dilakukan pada kondisi statis tanpa variasi beban atau gangguan luar.

Secara keseluruhan, penelitian ini menunjukkan bahwa sistem kendali PID yang di-*tuning* secara tepat mampu memberikan kinerja optimal dalam mengendalikan sudut roda swerve drive, dan menjadi landasan yang baik bagi pengembangan sistem robotik yang lebih kompleks di masa mendatang.

Kontribusi Penulis: Konseptualisasi: FCH dan DK; Metodologi: DK; Validasi: FCH dan DK; Analisis formal: FCH; Sumber daya: DK; Kurasi data: DK; Penulisan—persiapan draf asli: DK; Penulisan—peninjauan dan penyuntingan: FCH; Supervisi: FCH.

Pendanaan: Penelitian ini tidak menerima pendanaan internal ataupun eksternal.

Pernyataan Ketersediaan Data: Penulis bersedia jika data dari penelitian kami digunakan untuk kepentingan penelitian berikutnya dengan mencantumkan sitasi sesuai aturan yang berlaku.

Konflik Kepentingan: Penulis menyatakan tidak ada konflik kepentingan.

Referensi

[1] E. H. Binugroho, A. Setiawan, Y. Sadewa, P. H. Amrulloh, K. Paramasastra, and R. W. Sudibyo, "Position and Orientation Control of Three Wheels Swerve Drive Mobile Robot Platform," in *2021 International Electronics Symposium (IES)*, IEEE, Sep. 2021, pp. 669–674. doi: 10.1109/IES53407.2021.9593947.

[2] M. Haniff, H. M. Saputra, E. A. Zaki Hamidi, C. H. A. H. B. Baskoro, and S. A. Pratama, "Design and Control of Swerve Drive Mechanism for Autonomous Mobile Robot," in *2022 16th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, IEEE, Oct. 2022, pp. 1–6. doi: 10.1109/TSSA56819.2022.10063871.

[3] B. DeNoma, M. Kendall, N. Poulos, J. Dong, and R. Frank, "Four-Wheel Independent Steering Swerve Drive for First Robotics Competition," in *Volume 5: Dynamics, Vibration, and Control*, American Society of Mechanical Engineers, Oct. 2022. doi: 10.1115/IMECE2022-96192.

[4] K. M. Raza, M. Kamil, and P. Kumar, "IJARCCE 'Speed Control of DC Motor by using PWM,'" *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 5, 2016, doi: 10.17148/IJARCCE.2016.5478.

[5] H. Ying, W. Siler, and J. J. Buckley, "Fuzzy control theory: A nonlinear case," *Automatica*, vol. 26, no. 3, pp. 513–520, May 1990, doi: 10.1016/0005-1098(90)90022-A.

[6] S. J. Qin and T. A. Badgwell, "AN OVERVIEW OF INDUSTRIAL MODEL PREDICTIVE CONTROL TECHNOLOGY."

[7] M. A. Johnson *et al.*, *PID control: New identification and design methods*. Springer London, 2005. doi: 10.1007/1-84628-148-2.

[8] R. P. Borase, D. K. Maghade, S. Y. Sondkar, and S. N. Pawar, "A review of PID control, tuning methods and applications," Jun. 2021, *Springer Science and Business Media Deutschland GmbH*. doi: 10.1007/s40435-020-00665-4.

[9] S. J. Hammoodi, K. S. Flayyih, and A. R. Hamad, "Design and implementation speed control system of DC motor based on PID control and matlab simulink," *International Journal of Power Electronics and Drive Systems*, vol. 11, no. 1, pp. 127–134, Mar. 2020, doi: 10.11591/ijpeds.v11.i1.pp127-134.

[10] S. Tang and Y. Yu, "Research on Closed-loop Control of Step Motor Based on Magnetic Encoder," 2022.

[11] B. Zhou and J. Zhang, "Design of DC Motor PID Control System Based on STM32 Single Chip Microcomputer," *International Core Journal of Engineering*, doi: 10.6919/ICJE.202007_6(7).0008.

[12] R. Rojas and A. G. Förster, "Holonomic Control of a robot with an omni-directional drive," BöttcherIT Verlag, 2006.

[13] E. Maulana, M. A. Muslim, and A. Zainuri, "Inverse kinematics of a two-wheeled differential drive an autonomous mobile robot," in *Proceedings - 2014 Electrical Power, Electronics, Communications, Control and Informatics Seminar, EECCIS 2014. In conjunction with the 1st Joint Conference UB-UTHM* , Institute of Electrical and Electronics Engineers Inc., Jan. 2014, pp. 93–98. doi: 10.1109/EECCIS.2014.7003726.

[14] H. B. Shin and J. G. Park, "Anti-windup PID controller with integral state predictor for variable-speed motor drives," *IEEE Transactions on Industrial Electronics*, vol. 59, pp. 1509–1516, Mar. 2012, doi: 10.1109/TIE.2011.2163911.

[15] V. V. Patel, "Ziegler-Nichols Tuning Method: Understanding the PID Controller," *Resonance*, vol. 25, pp. 1385–1397, Oct. 2020, doi: 10.1007/s12045-020-1058-z.

[16] K. Sekarsari and T. Tata, "Performance analysis of PID control in DC Brushless motor using trial and error method," *IOP Conf Ser Mater Sci Eng*, vol. 1098, p. 042027, Mar. 2021, doi: 10.1088/1757-899x/1098/4/042027.

[17] X. Zhang and S. H. HosseinNia, "Transient Response Analysis of Reset PID Control Systems," in *IFAC-PapersOnLine*, Elsevier B.V., Jun. 2024, pp. 430–435. doi: 10.1016/j.ifacol.2024.08.100.