Implementasi Kontrol PID untuk Analisis Pengaturan Kecepatan Motor DC Menggunakan STM32

Ari Murtono[1)], Fitri[2)], Leonardo Kamajaya[3)], Muhammad Shulton Al amin[4)]
[1] Teknik Elektronika, Jurusan Teknik Elektro, Politeknik Negeri Malang
ari.murtono@polinema.ac.id
[2] Teknik Elektronika, Jurusan Teknik Elektro, Politeknik Negeri Malang
fitri@polinema.ac.id
[3] Teknik Elektronika, Jurusan Teknik Elektro, Politeknik Negeri Malang
leonardo42@polinema.ac.id
[4] Teknik Elektronika, Jurusan Teknik Elektro, Politeknik Negeri Malang
m.shulton.al@gmail.com

Abstrak

Motor DC dengan pengontrol ON-OFF cenderung menyebabkan kecepatan motor DC menurun ketika diberikan beban yang bervariasi. Adapun salah satu solusi untuk mengatasi masalah ini adalah dengan menggunakan kontroller PID sebagi pengontrol kestabilan motor DC. Tujuan dari penelitian ini adalah mengaplikasikan metode kontrol PID pada modul ajar STM32 untuk mengatur kecepatan motor DC dan menjaga kestabilannya sesuai dengan Set Point yang telah di tentukan. Parameter dari P, I dan D didapatkan dari penalaan teori perhitungan Zeigler-Nichols dan tuning dengan metode Trial and Error. Didapatkan respon sistem terbaik dengan hasil tuning dari kedua metode dengan nilai Kp = 1.1, Kd = 0.5, Kd = 0.05. Berdasarkan data yang diambil metode kontrol PID berhasil diterapkan pada sistem dengan meredam Error Steady State yang semula semula 16 % menjadi 5 % untuk sistem tanpa beban dan 3.3 % untuk sistem berbeban 200g, dan berbeban 500g.

Keywords: *motor dc, stm32, modul ajar, pid*

I. PENDAHULUAN

Motor DC (arus searah) adalah suatu alat yang dapat mengubah energi listrik menjadi energi mekanik[1]. Penggunaan motor DC dalam bidang industri sangat sering terjadi, akan tetapi masih ditemukan pada penggunaan motor DC dengan pengontrol ON-OFF yang artinya hanya ada dua kondisi yang memungkinkan, yaitu ON sepenuhnya dan OFF sepenuhnya[2]. Memang menggunakan pengontrol ON-OFF jauh lebih sederhana dan relatif mudah[3], akan tetapi ketika motor DC digunakan untuk menggerakan sebuah mesin yang lebih kompleks yang membutuhkan kestabilan kecepatan dari motor pengontrol ON-OFF ini sangat tidak disarankan. Seringkali kecepatan motor DC menggunakan pengontrol ON-OFF tidak stabil saat diberikan beban tertentu dan relatif menurun kecepatannya ketika beban yang diberikan kepada motor DC cukup besar[4].

Karena kestabilan kecepatan motor DC sulit untuk dikendalikan[5]. Maka dari itu sistem pengendalian kecepatan motor DC sangat diperlukan untuk mengatasi masalah ini agar motor dapat berputar stabil sesuai kecepatan yang diinginkan meskipun motor diberi beban yang bervariasi[6]. Menggunakan kontroller PID (Propotional Integral Derivative) adalah salah satu cara untuk mengontrol kecepatan motor DC, dimana parameter settinganya dapat diatur sesuai kebutuhan sampai mendapatkan respon sistem yang diinginkan[7]. Sesuaiidengan nilai set point yang telah ditentukan, output yang diharapkan dari motor DC adalah kecepatan dengan stabilitas yang baik, tingkat kesalahan dan overshoot yang kecil. Oleh karena itu kontroler yang dipilih adalah PID (Propotional Integral Derivative), karena dari kombinasi dari kontroler P, I, dan D inilah yang menghasilkan output kecepatan motor yang stabil dan tingkat kesalahan serta overshoot yang kecil.

II. KAJIAN LITERATUR

A. Kontroller PID

Berdasarkan tabel perbandingan sistem diatas bisa kita amati bahwa sistem menggunakan kontrol PID berdasarkan metode Trial and Error tetap menunjukan kestabilan meski diberikan beban yang bervariasi, akan tetapi semakin besar beban yang diberikan kepada motor DC nilai Error Steady State dan Overshoot semakin membesar hal ini disebabkan karena beban yang bertambah besar sehingga kontroler akan menyesuikan dengan berat beban . Meski begitu sistem dengan tuning PID metode Trial and Error ini tetap stabil dan berhasil menekan Error Steady State yang semula sistem tanpa menggunakan kontrol PID adalah sebesar 16% dan diredam menjadi 3.3 - 5 % menggunakan kontrol PID metode trial and error.[7]

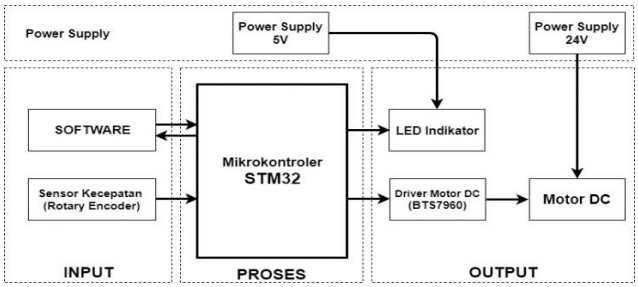
Kontrol PID pertama dikembangkan pada tahun 1911 oleh Elmer Sperry. Pada Tahun 1933 Taylor Instrumental Company (TIC) menggunakan pengontrol pneumatic pertama dengan menggunakan kontrol proporsional yang dapat diatur. Beberapa tahun kemudian, ditemukan formula untuk menghilangkan error steady state pada kontrol proporsional dengan cara menggunakan kontrol Integral

dan penggabungan kedua kontrol ini disebut Kontrol Proporsional – Integral. Pada tahun 1940 dikembangkan PID kontrol pertama. Pada PID dengan menambahkan kontrol derivatif yang mana dapat mengurangi isu overshoot yang di timbulkan oleh kontroler proporsional. [8]

III. METODE PENELITIAN

A. *Diagram Blok Sistem*

Secara keseluruhan sistem ini akan dirancang memiliki beberapa komponen elektronik dan rancangan software yang diperlukan terlihat pada gambar 1 berikut ini.

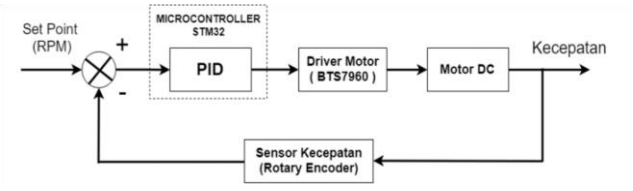


Gambar 1. Diagram blok sistem

Pada gambar 1 yang merupakan diagram blok dari sistem dapat diketahui bahwa input sistem adalah software dan sensor kecepatan[8]. Perangkat lunak berupa GUI (Graphical User Interface) digunakan untuk memasukan nilai parameter PID dan menampilkan grafik respon sistem, kemudian sensor kecepatan sebagai pendeteksi kecepatan motor DC. Pada bagian proses terdapat mikrokontroler STM32 sebagai kontroler utama, dan dibagian output terdapat 3 LED indikator sebagai indikator kestabilan sistem dengan warna merah, kuning dan hijau. Driver motor DC difungsikan sebagai pengontrol kecepatan motor DC, dan motor DC sebagai aktuatornya.

B. *Cara kerja alat*

Setiap komponen yang digunakan dalam sistem yang ditunjukkan pada Gambar 1 dihubungkan bersama untuk membentuk sistem kerja, seperti yang ditunjukkan pada Gambar 2 di bawah ini



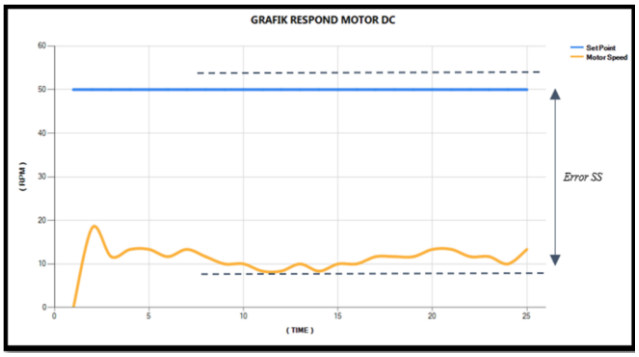
Gambar 2. Flowchart alur kerja alat

Input yang masuk pada sistem adalah set point berupa kecepatan motor (RPM) yang mana akan di proses oleh mikrokontroler yang telah terisi program PID dengan nilai parameter PID yang telah ditentukan, parameter PID dan set point dapat kita masukan melalui sofware GUI (Graphical User Interface) kemudian mikrokontroler akan mengeluarkan sinyal PWM ke driver motor DC. Driver motor akan menentukan kecepatan dari Motor DC berdasarkan besar PWM yang dikeluarkan mikrokontroler. Motor DC akan berkerja atau berputar secara continuous sesuai output dari driver motor dan motor DC akan memberikan output pada sistem berupa kecepatan dalam satuan RPM. Kemudian kecepatan motor (RPM) akan di deteksi oleh sensor kecepatan (rotary encoder) yang mana akan diguakan sebagai feedback dari sistem yang akan dikirimkan ke Mikrokontroler. Mikrokotroler yang menerima hasil feedback akan memproses itu dengan cara membandingankan selisih dari Set Point dengan feedback yang diberikan oleh sensor kecepatan (Error)[9]. Dari hasil error ini mikrokontroler akan menentukan untuk menambah atau mengurangi besarnya nilai PWM yang dikeluarkan dengan perhitungan kontroler PID sampai nilai Error sama dengan 0[10]. Pada saat sistem ini berjalan nilai error akan selalu dibaca oleh mikrokontroler yang mana mikrokontroler akan menyalakan LED indikator sesuai nilai error yang terbaca. LED merah menyala ketika error sistem bernilai < -10 dan > 10 yang artinya sistem tidak stabil, kemudian LED kuning menyala ketika error sistem bernilai >= -10 dan < -2 dan ketika error bernilai > 2 dan < 10 yang artinya sistem kurang stabil, kemudian LED hijau menyala ketika error sistem bernilai >= -2 dan <= 2 yang artinya sistem telah stabil. Hasil output yang dikeluarkan oleh sistem akan ditampilkan pada GUI (Graphical User Interface) berupa grafik respon yang bisa dimonitoring secara real time ketika sistem bekerja.

IV. HASIL DAN PEMBAHASAN

A. *Pengujian sistem tanpa kontrol PID*

Motor DC yang sudah terhubung dengan conveyor langkah pertama adalah mengetahui respon sistem tanpa menggunakan kontroler PID dengan membuat nilai Kp = 1 dan Ki maupun Kd = 0 sehingga menjadi sistem close loop tanpa adanya kontrol PID. Grafik respon dari sistem dengan Setpoint = 50 RPM bisa kita lihat pada gambar 3 sebagai berikut.



Gambar 3. Grafik respon sistem tanpa kontrol PID

Berdasarkan gambar 3 diatas terlihat bahwa sistem tanpa kontrol PID tidak bisa mencapai set point yang diinginkan yaitu 50 RPM dengan nilai Error Steady State adalah,

$$Error\ SS = \frac{42-50}{50} x100\% = 16\%$$

Maka dari itu dibutuhkan kontrol PID yang sesuai agar sistem bisa mencapai Set Point dan stabil dengan Rise Time, Overshoot, dan error steady state yang kecil.

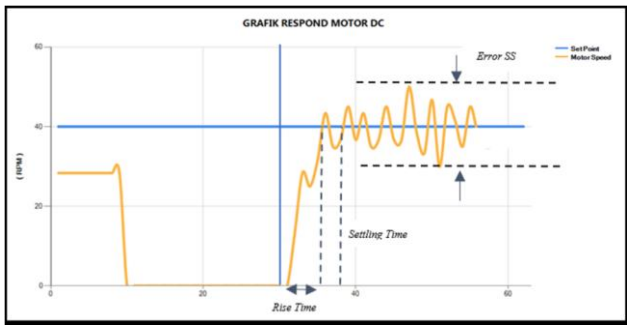
B. Pengujian sistem dengan Kontrol PID metode osilasi

Berdasarkan hasil perhitungan parameter PID menggunakan penalaan metode osilasi yang bisa dilihat pada tabel 1, didapatkan nilai Kp, Ki, dan Kd sebagai berikut:

Tabel 1. Penalaan Parameter PID Metode Osilasi

Tipe Kontrol	Kp	Ti	Td
P	0.5Ku	~	0
PI	0.45Ku	1/1.2Pu	0
PID	0.6Ku	0.5Pu	0.125Pu

Berdasarkan hasil perhitungan pada tabel 1 didapatkan nilai Kp = 1.8, Ki = 1.44, Kd = 0.57. Selanjutkan Parameter kontrol PID tersebut akan diterapkan secara langsung pada sistem dan diamati responya dengan set point = 40.



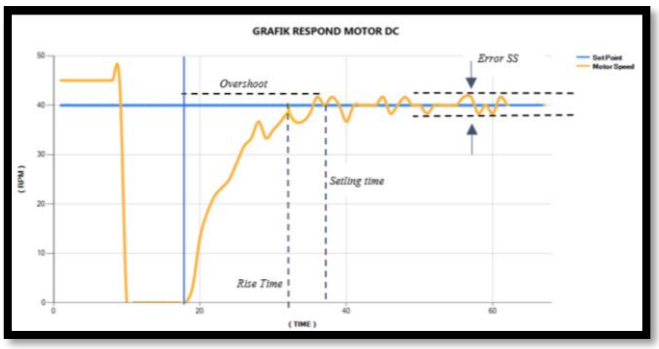
Gambar 4. Grafik respon sistem dengan kontrol PID metode osilasi

$$Rise\ Time = 5\ x\ 100ms = 500ms = 0.5s$$
$$Overshoot = \frac{50-40}{40} x\ 100\% = 25\%$$
$$Error\ SS = \frac{50-40}{40} x\ 100\% = 25\%$$
$$Settling\ time = 7\ x\ 100ms = 0.7s$$

Berdasarkan data perhitungan Rise Time, Overshoot, Error Steady state, dan settling time bisa diketahui bahwa hasil tuning ini masih belum sempurna dan perlu adanya tuning lagi dengan metode Trial and Error.

C. Pengujian Sistem dengan kontrol PID metode Trial dan Error

Karena pengujian berdasarkan Tuning PID menggunakan metode Osilasi, sistem masih menunjukan ketidakstabilan. Dari hasil pengujian tersebut, tuning PID secara manual atau metode Trial and Error berdasarkan nilai parameter PID di uji coba. Parameter PID hasil tuning sebelumnya yaitu tuning metode osilasi digunakan dengan mencoba menaikan atau menurunkan nilai Kp, Ki dan Kd, sehingga menemukan nilai parameter PID yang sesuai dan yang terbaik. Dengan metode ini didapatkan nilai parameter PID yang terbaik yaitu dengan nilai Kp = 1.1, Ki = 0.50, Kd = 0.05. Untuk grafik respon sistem bisa diketahui pada gambar 5 sebagai berikut



Gambar 5. Grafik respon sistem dengan kontrol PID metode trial and error

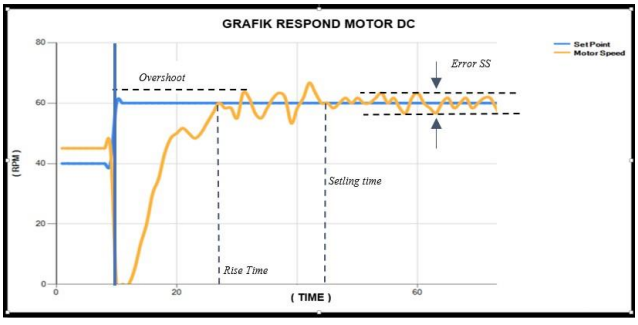
Berdasarkan grafik pada gambar 5 terlihat bahwa sistem sudah mencapai steady state dan stabil dengan spesifikasi respon sistem sebagai berikut, (Time Sampling = 100ms dan Set Point = 40 RPM).

$$Rise\ Time = 13\ x\ 100ms = 1300ms = 1.3s$$
$$Overshoot = \frac{41-40}{40} x\ 100\% = 2.5\%$$
$$Error\ SS = \frac{41-39}{40} x\ 100\% = 5\%$$
$$Settling\ time = 20\ x\ 100ms = 2s$$

Berdasarkan data perhitungan Rise Time, Overshoot, Error Steady state, dan settling time pada tuning metode Trial and Error bisa diketahui bahwa hasil tuning menggunakan metode Trial and Error ini cukup baik dan Error Steady State yang cukup rendah dengan Rise Time dan settling time yang cukup cepat

D. Pengujian Sistem dengan beban 200gr

Berikut merupakan grafik respon dari sistem dengan beban sebesar 200g. Dengan nilai Set Point = 60 RPM dan nilai parameter PID dari hasil tuning Trial and Error bisa kita lihat pada gambar 6 dibawah ini.



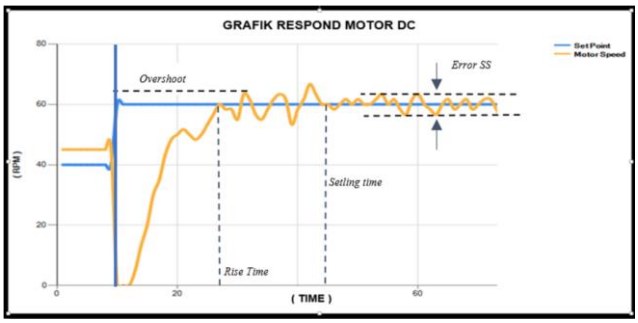
Gambar 6. Grafik respon sistem dengan kontrol PID metode trial and error

Berdasarkan gambar 6 diatas pada grafik respon sistem terlihat bahwa sistem sudah mencapai set point dan

cukup stabil dengan spesifikasi respon sistem Rise time = 1.8s, Overshoot = 1.5%, Error SS = 3.3%, Settiling time = 3.2s.

E. Pengujian sistem dengan beban 500gr

Berikut merupakan grafik respon dari sistem dengan beban sebesar 500g. Dengan nilai Set Point = 60 RPM dan nilai parameter PID dari hasil tuning Trial and Error bisa kita lihat pada gambar 6 dibawah ini.



Gambar 7. Grafik respon sistem dengan kontrol PID metode trial and error

Berdasarkan gambar 7 diatas pada grafik respon sistem terlihat bahwa sistem sudah mencapai set point dan cukup stabil dengan spesifikasi respon sistem Rise time = 1.3s, Overshoot = 3.3%, Error SS = 3.4%, Settiling time = 0.8s.

F. Pembahasan

Setelah mendapatkan parameter PID yang sesuai dari metode Trial and Error sebelumnya dan mendapatkan hasil yang stabil dengan Rise Time, Overshoot, serta error steady state yang kecil daripada metode osilasi. kontrol PID dari metode Trial and Error ini diuji dengan cara memberikan beban pada sistem sebesar 200g dan 500g.

Pada kasus ini akan diketahui apakah respon sistem tetap stabil dan menunjukan performasi yang baik ketika diberikan beban yang bervarisai atau malah sebaliknya. Berikut merupakan perbandingan performasi respon sistem dengan beban yang bervariasi berdasarkan pengambilan data sebelumnya yang ditunjukan pada tabel 2 sebagai berikut.

Tabel 2. Perbandingan Respon Sistem Berbeban

Metode Kontrol	Beban	Error Steady State	Maximum Overshoot	Rise Time	Settling Time
Tanpa Kontrol PID	0g	16 %	~	~	~
Trial and Error	0g 200g	5 % 3.3 %	2.5 % 1.5 %	1.3 s 1.8 s	2 s 0.5 s
	500g	3.4 %	3.3 %	1.3 s	0.8 s

Berdasarkan tabel perbandingan sistem diatas bisa kita amati bahwa sistem menggunakan kontrol PID berdasarkan metode Trial and Error tetap menunjukan kestabilan meski diberikan beban yang bervariasi, akan tetapi semakin besar beban yang diberikan kepada motor DC nilai Error Steady State dan Overshoot semakin membesar hal ini disebabkan karena beban yang bertambah besar sehingga kontroler akan menyesuikan dengan berat beban . Meski begitu sistem dengan tuning PID metode Trial and Error ini tetap stabil dan berhasil menekan Error Steady State yang semula sistem tanpa menggunakan kontrol PID adalah sebesar 16% dan diredam menjadi 3.3 - 5 % menggunakan kontrol PID metode trial and error.

V. KESIMPULAN

Metode kontrol PID berhasil diterapkan pada sistem tanpa beban, maupun berbeban dan meskipun beban bervarisi kontrol PID tetap menunjukan performa yang jauh lebih baik dibandingkan dengan sistem tanpa adanya kontrol. Prameter PID yang didapatkan dari metode tuning Trial and Error menunjukan performa lebih baik dibandingkan dengan tuning metode osilasi, dengan nilai Kp = 1.1, Ki = 0.50, Kd = 0.05 berdasarkan tuning metode trial and error. Kontrol PID bisa meredam besarnya Error Steady State yang timbul ketika sistem dijalankan tanpa PID sebesar 16 % menjadi 5 % untuk sistem tanpa beban dan menjadi 3.3 % untuk sistem berbeban 200g, dan berbeban 500g.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Politeknik Negeri Malang atas dana penelitian DIPA Reguler 2021 sehingga terselenggara penelitian ini.

REFERENSI

[1] E. Mardianto, “Rancang Bangun Antarmuka Operasi Motor DC Dengan Converter 4 Kuadran,” Tugas Akhir. Padang Univ. Negeri Padang, 2018.
[2] B. A. Prabowo, “Pemodelan Sistem Kontrol Motor DC dengan Temperatur Udara sebagai Pemicu,” INKOM J., vol. 2, no. 1, 2010, pp. 39–43.
[3] N. Puspawardhana, F. Suhartati, and T. Nurwati, “Pengaturan Posisi Motor Servo Pada Miniatur Rotary Parking,” J. Mhs. TEUB, vol. 2, 2014. no. 5.
[4] H. Mukti, T. U. Syamsuri, R. Joto, I. N. Syamsiana, and A. M. Yudha, “PENGATURAN KECEPATAN MOTOR DC MENGGUNAKAN KONTROL PID DENGAN METODE TUNING ZIEGLER NICHOLS,” in Prosiding Seminar Nasional Teknologi Elektro Terapan, vol. 4, no. 1, 2020, pp. 88–94.
[5] R. Dhanang, “Pengendalian Kecepatan Motor Arus Searah Seri dengan DC Chopper,” Universitas Indonesia, 2012.
[6] H. Haryanto and S. Hidayat, “Perancangan HMI (Human Machine Interface) Untuk Pengendalian Kecepatan Motor DC,” Setrum Sist. Kendali-Tenaga-elektronika-telekomunikasi-komputer, vol. 1, no. 2, 2016, pp. 58–65.

[7] W. Waluyo, A. Fitriansyah, and S. Syahrial, "Analisis Penalaan Kontrol PID pada Simulasi Kendali Kecepatan Putaran Motor DC Berbeban menggunakan Metode Heuristik," ELKOMIKA J. Tek. Energi Elektr. Tek. Telekomun. Tek. Elektron., vol. 1, no. 2, 2013, p. 79.
[8] M. R. Setiawan, M. A. Muslim, and G. D. Nusantoro, "Kontrol Kecepatan Motor DC Dengan Metode PID Menggunakan Visual Basic 6.0 dan Mikrokontroler ATmega 16," J. Mhs. TEUB, 2013, vol. 1, no. 2.
[9] R. Muhardian and K. Krismadinata, "Kendali Kecepatan Motor DC Dengan Kontroller PID dan Antarmuka Visual Basic," JTEV (Jurnal Tek. Elektro dan Vokasional), vol. 6, no. 1, 2020, pp. 328–339.
[10] R. Sinaga, "Pengendali Kecepatan Motor DC Menggunakan Sensor Hall Berbasis Mikrokontroler ATMega 8535," Saintia Fis., vol. 1, no. 1, 2013, p. 221178.