Nama : Kevin Bayu Pradana
NIM : 224308009
Kelas : TKA-6A

Analisis

Pada tugas praktikum kali ini yaitu membuat sebuah sistem untuk mengontrol kecepatam motor DC dengan menggunakan PID berbasis IoT. Dalam praktikum ini alat dan bahan yang digunakan yaitu motor DC RF 300-EA, ESP32, motor driver, MQTT, Arduino IDE, dan Matlab. Langkah pertama yang dilakukan yaitu mencari spesifikasi dari motor DC yang digunakan, untuk spesifikasinya sebagai berikut:

Tegangan : 3,9V
Kecepatan tanpa beban : 4400 r/min
Tahanan armatur R :10 Ohm
Induktansi armatur L : 0.0002 Henry
Momen inersia rotor J : 3e-7 kg.m^2
Koefisien redaman viskosa B : 2e-6 Nm.s/rad
Konstanta torsi Kt : 0.003 Nm/A
Konstanta ggl balik Ke : 0.003 V.s/rad

Setelah itu, Motor dimodelkan dengan menggabungkan persamaan kelistrikan dan mekanik sebagai berikut:

- Persamaan kelistrikan (hukum Kirchhoff):

$$V(s) = LsI(s) + RI(s) + Ke\omega(s)$$

- Persamaan mekanik (hukum Newton):

$$J\,s\omega(s) + B\omega(s) = KtI(s)$$

Dengan eliminasi variabel I(s) diperoleh fungsi alih kecepatan sudut terhadap tegangan input sebagai:

$$G(s) = \omega(s) \ / V(s) = Kt / LsJ + (LB + RJ)s + (RB + KeKt)$$

Dengan substitusi nilai parameter, fungsi alih menjadi:

$$G(s)=0{,}003 / 6 \times 10^{-11}s^2 + 6{,}06 \times 10^{-6}s + 6{,}09 \times 10^{-5}$$

Langkah selanjutnya setelah memperoleh fungsi alih sistem adalah melakukan simulasi respons dinamis motor serta merancang pengendali PID menggunakan MATLAB. Fungsi alih yang diperoleh, yaitu:

$$G(s)=0{,}003 / 6 \times 10^{-11}s^2 + 6{,}06 \times 10^{-6}s + 6{,}09 \times 10^{-5}$$

Lalu diimplementasikan ke dalam MATLAB dalam bentuk fungsi transfer orde dua. Fungsi ini (Gambar 1) merepresentasikan hubungan antara tegangan masukan dan kecepatan sudut motor DC, yang menjadi dasar dalam proses desain pengendali. Perancangan pengendali PID dilakukan menggunakan fitur *PID Tuning* di MATLAB. Proses tuning menghasilkan parameter kontroler sebesar Kp = 0,00429, Ki = 0,04142, dan Kd = 0, yang dipilih berdasarkan kriteria performa waktu seperti waktu naik dan *overshoot*. Parameter ini kemudian diimplementasikan dalam model simulasi di Simulink (Gambar 2). Model Simulink terdiri atas blok *transfer function* untuk motor DC, blok *PID Controller*, dan

referensi kecepatan. Simulasi sistem tertutup dilakukan untuk mengevaluasi kinerja pengendali terhadap perubahan referensi kecepatan. Berdasarkan hasil simulasi (Gambar 3), sistem menunjukkan respons yang stabil dan cepat menuju nilai setpoint sekitar 800 rpm dalam waktu kurang dari 2 detik, tanpa *overshoot* yang signifikan.

Setelah desain pengendali berhasil disimulasikan, tahap berikutnya adalah implementasi sistem kendali pada perangkat keras menggunakan mikrokontroler ESP32. Dalam implementasi ini, ESP32 bertindak sebagai pengendali utama yang mengatur motor DC melalui motor driver serta membaca kecepatan motor secara real-time menggunakan sensor kecepatan. Sistem diberi catu daya sebesar 3,3 V langsung dari port USB laptop, sehingga motor hanya mampu mencapai kecepatan maksimum sekitar 4400 rpm. Untuk komunikasi data antara ESP32 dan MATLAB, digunakan protokol komunikasi *Message Queuing Telemetry Transport* (MQTT), yang memungkinkan pertukaran data secara ringan dan efisien dalam jaringan IoT.

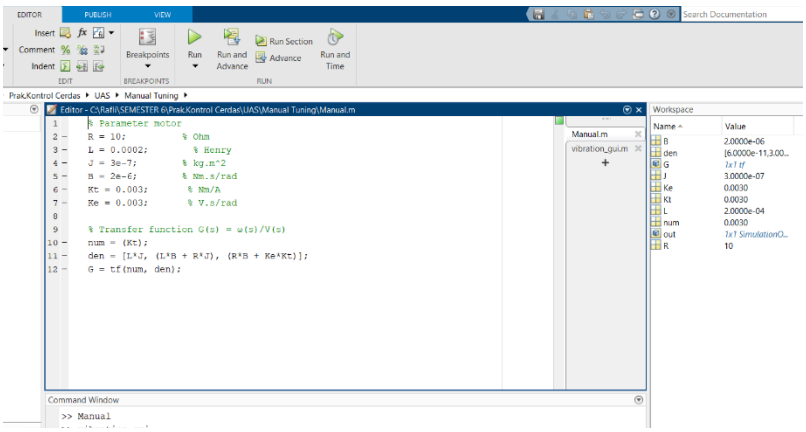
Dalam sistem MQTT yang dibangun, terdapat antarmuka kontrol yang dilengkapi dengan beberapa fitur interaktif. Pertama, terdapat tombol **ON/OFF** yang berfungsi untuk mengaktifkan atau mematikan motor DC secara langsung melalui dashboard. Kedua, terdapat tiga level kecepatan yang dapat dipilih oleh pengguna, yaitu:

- **Level 1**: 900 rpm
- **Level 2**: 1500 rpm
- **Level 3**: 2000 rpm

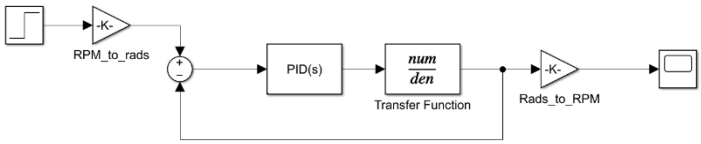
Pemilihan level ini akan mengatur nilai referensi kecepatan yang dikirimkan ke MATLAB sebagai input dari sistem kendali PID. Nilai referensi ini selanjutnya diproses bersama data kecepatan aktual untuk menghasilkan sinyal PWM yang dikirim kembali ke ESP32 guna mengatur putaran motor. Selain pengaturan kecepatan, sistem juga menyediakan kontrol arah putar motor, yaitu searah jarum jam (*clockwise*) dan berlawanan arah jarum jam (*counter-clockwise*), yang dikendalikan melalui antarmuka MQTT. Arah putar ini diatur dengan mengubah polaritas sinyal kontrol pada motor driver dari ESP32. Untuk monitoring sistem secara real-time, dashboard MQTT dilengkapi dengan dua grafik utama, yaitu grafik kecepatan (RPM) dan grafik sinyal kendali (PWM). Grafik RPM menampilkan kecepatan motor berdasarkan pembacaan dari sensor, sedangkan grafik PWM menampilkan sinyal kendali hasil dari proses PID. Visualisasi ini memungkinkan pengguna memantau performa sistem secara langsung, sekaligus mengevaluasi stabilitas dan respons kendali terhadap perubahan referensi kecepatan.

Dengan integrasi antara MATLAB, Simulink, ESP32, dan MQTT, sistem ini tidak hanya mendemonstrasikan pengendalian kecepatan motor DC berbasis PID secara real-time, tetapi juga menyediakan antarmuka interaktif berbasis IoT yang mendukung fleksibilitas kontrol dan pemantauan secara menyeluruh. Pendekatan

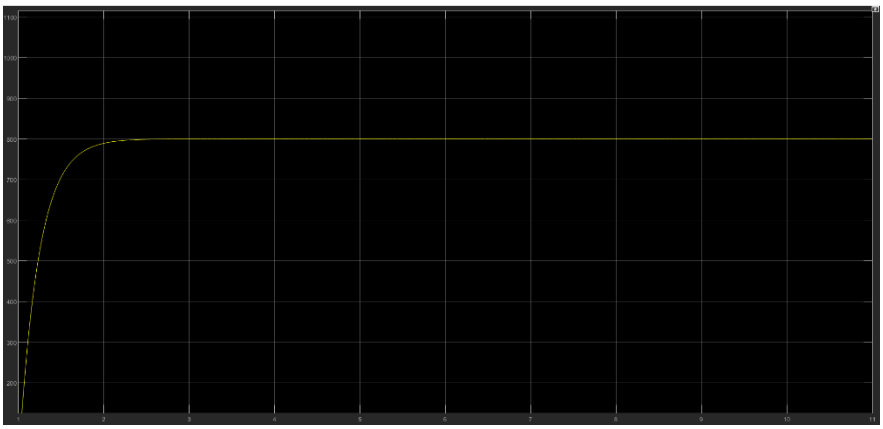
ini membuktikan bahwa teknologi IoT dapat diintegrasikan secara efektif dengan sistem kendali klasik untuk membentuk sistem otomasi yang responsif dan adaptif.



Gambar 1. Parameter Motor dan Fungsi Transfer Orde 2



Gambar 2. Rangkaian Simulink



Gambar 3. Hasil Simulasi Matlab