DC MOTOR SPEED CONTROL(PID SIMULATION)

PROJECT

Name:S.Nandini

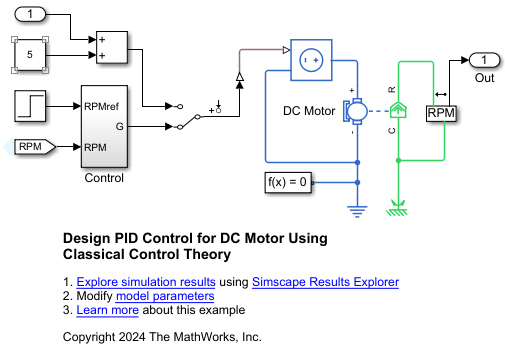
Year:3rd year

Branch:Electrical and Electronics Engineering

Institute:Annamacharya University

AIM:The primary aim of a DC motor speed control system using a PID (Proportional-Integral-Derivative) controller is to precisely regulate and maintain a desired motor speed, even when faced with varying load conditions or disturbances.

* DESCRIPTION: A [PID (Proportional-Integral-Derivative) controller](https://www.google.com/search?sca_esv=f443e875ddf71cd7&cs=0&sxsrf=AE3TifMukmauLGVFkkupOD10DcH27Vt3KA%3A1755408575478&q=PID+%28Proportional-Integral-Derivative%29+controller&sa=X&ved=2ahUKEwjthajEjpGPAxUyzjgGHQGdHMEQxccNegQIAhAB&mstk=AUtExfA_NcCo36A2o8p_ju9-1fHO6ohv-r8kztVGbc_xSfmYP-4SWRgo9qDCNDck-jz-9pqKcT9DHjmNhbq3vvmgDQV0AOz6D-ajYWetzQr4497af0fM3XSKmkxZfqd74qkDTvgYv8K1qvIm388dlQUtZQOtdx-M7J9nCmDajFHTydUn0TZ38v03dgyESpX0tnGMR_NS70wbSybxwPcXDjGeDtQJtzBiAV9Ruy4ka5A8buizwrhagJxTNAdjCnGd9B0-bI4IOsMIRNLqgIr84P1n8GQy&csui=3) is used to regulate the speed of a DC motor by adjusting the voltage applied to it. The PID controller continuously monitors the motor's speed, compares it to a desired speed, and calculates an error signal. This error signal is then processed by the PID algorithm, which produces a control signal (voltage) that is fed back to the motor, minimizing the speed error and ensuring the motor reaches and maintains the desired speed.
* DIAGRAM:



Source Code:

import matplotlib.pyplot as plt

import numpy as np

# Simulation parameters

setpoint = 100  # Desired motor speed (RPM)

Kp = 0.1       # Proportional gain

Ki = 0.01      # Integral gain

Kd = 0.05      # Derivative gain

dt = 0.1       # Time step for simulation (seconds)

sim\_time = 150 # Total simulation time (seconds)

# Initialize variables

integral = 0

previous\_error = 0

motor\_speed = 0  # Initial motor speed (RPM)

# Store data for plotting

time\_data = []

speed\_data = []

# Simulation loop

for t in np.arange(0, sim\_time, dt):

    # Calculate error

    error = setpoint - motor\_speed

    # Proportional term

    P\_term = Kp \* error

    # Integral term

    integral += error \* dt

    I\_term = Ki \* integral

    # Derivative term

    derivative = (error - previous\_error) / dt

    D\_term = Kd \* derivative

    # Calculate control output

    output = P\_term + I\_term + D\_term

    # Update motor speed (simple model)

    motor\_speed += output \* dt

    # Store data for plotting

    time\_data.append(t)

    speed\_data.append(motor\_speed)

    # Store error for next iteration

    previous\_error = error

# Plot the results

plt.plot(time\_data, speed\_data)

plt.xlabel("Time (seconds)")

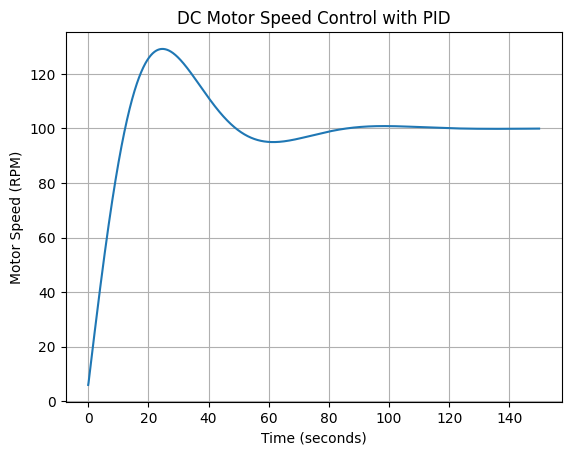
plt.ylabel("Motor Speed (RPM)")

plt.title("DC Motor Speed Control with PID")

plt.grid(True)

plt.show()

OUTPUT:



CONCLUSION: The simulation results often demonstrate the superiority of PID control over conventional methods, especially in dynamic and transient responses.