delete(instrfind); % Clear any existing serial connections

% Open serial connection

arduinoObj = serial('COM3', 'BaudRate', 9600);

fopen(arduinoObj);

pause(2); % Wait for Arduino to initialize

% PID parameters

Kp = 120;

Ki = 20;

Kd = 8;

% Get target temperature from user

setPoint = input('Enter the target temperature (°C): ');

% PID variables

integral = 0;

prevError = 0;

% Data vectors for plotting

temperatureData = [];

pwmData = [];

time = [];

figure;

subplot(2,1,1);

sicPlot = plot(NaN, NaN, 'b');

ylabel('Temperature (°C)');

ylim([20 50]);

grid on;

subplot(2,1,2);

pwmPlot = plot(NaN, NaN, 'r');

ylabel('PWM');

xlabel('Time');

ylim([0 260]);

grid on;

for i = 1:6000

temperatureStr= fgetl(arduinoObj);

temperature= str2double(temperatureStr);

if ~isnan(temperature)

% Reset control

if temperature < (setPoint-0.2)

integral = 0;

prevError = 0;

end

% PID calculation

error = temperature - setPoint;

integral = integral + error;

derivative = error - prevError;

pwm\_raw = Kp\*error + Ki\*integral + Kd\*derivative;

% Limit PWM (0–255)

pwm = round(min(max(pwm\_raw, 0), 255));

% Send PWM value to Arduino

fprintf(arduinoObj, '%d\n', pwm);

% Print to console

disp(['Temperature: ', num2str(temperature), ' °C → PWM: ', num2str(pwm), ' (Raw PWM: ', num2str(pwm\_raw), ')'])

% Update graph data

temperatureData (end+1) = temperature;

pwmData(end+1) = pwm;

time(end+1) = i;

% Update plots

set(sicPlot, 'XData', time, 'YData', temperatureData);

set(pwmPlot, 'XData', time, 'YData', pwmData);

drawnow;

prevError = error;

else

disp(' Invalid data received. ');

end

pause(1); % Read every 1 second

end

% Cleanup

fclose(arduinoObj);

delete(arduinoObj);

clear arduinoObj;