```matlab
clear; close all; clc;

%% Parameters
% Leg geometry
L_thigh = 0.3;
L_shin = 0.3;

% PD Controller gains for shin
Kp = 150;   % Proportional gain (increased for better tracking)
Kd = 35;    % Derivative gain (increased for better damping)

% Simulation parameters
dt = 0.001;     % Time step (s) - smaller for stability
t_end = 10;     % Simulation duration (s)
t = 0:dt:t_end;
N = length(t);

% Walking gait parameters
stride_freq = 0.4;  % Hz (walking frequency - even slower)
thigh_amplitude = 30 * pi/180;  % Thigh swing amplitude (rad)

%% Initialize variables
theta_thigh = zeros(1, N);        % Thigh angle (absolute)
theta_shin_abs = zeros(1, N);     % Shin angle (absolute)
theta_knee = zeros(1, N);         % Knee angle (shin relative to thigh)
omega_knee = zeros(1, N);         % Knee angular velocity
tau_thigh = zeros(1, N);          % Thigh torque
tau_knee = zeros(1, N);           % Knee control torque

% Initial conditions
theta_thigh(1) = 0;
theta_knee(1) = 50 * pi/180;      % Initial knee bend (50 degrees)
omega_knee(1) = 0;

% Shin dynamics parameters
I_knee = 0.05;   % Moment of inertia (kg*m^2)
b_knee = 8;      % Damping coefficient (increased for stability with higher gains)

%% Simulation loop
for i = 1:N-1
    % Generate thigh motion with MODERATE VARYING AMPLITUDE (realistic disturbance)
    % Reduced number of components and amplitudes for smoother variation
    amp_modulation = 0.7 + 0.5*sin(0.8*2*pi*stride_freq*t(i)) + ...
                     0.2*cos(1.6*2*pi*stride_freq*t(i));

    % Ensure amplitude doesn't go negative
    amp_modulation = max(0.2, amp_modulation);

    theta_thigh(i) = thigh_amplitude * amp_modulation * sin(2*pi*stride_freq*t(i));

    % Torque includes amplitude modulation effects
    amp_deriv = 0.5*0.8*(2*pi*stride_freq)*cos(0.8*2*pi*stride_freq*t(i)) - ...
                0.2*1.6*(2*pi*stride_freq)*sin(1.6*2*pi*stride_freq*t(i));

    tau_thigh(i) = thigh_amplitude * amp_modulation * (2*pi*stride_freq)^2 * sin(2*pi*stride_freq*t(i)) + ...
                   thigh_amplitude * amp_deriv * sin(2*pi*stride_freq*t(i));

    phase = 2*pi*stride_freq*t(i);
    theta_knee_base = (50 - 30*sin(phase + pi/4)) * pi/180;
    thigh_nominal = thigh_amplitude * sin(phase);
```

```matlab
    thigh_error = theta_thigh(i) - thigh_nominal;
    compensation_gain = -1.5;  % Increased for stronger compensation

    theta_knee_desired = theta_knee_base + compensation_gain * thigh_error;

    % Derivative includes compensation
    omega_knee_base = -30*(pi/180)*(2*pi*stride_freq)*cos(phase + pi/4);
    omega_thigh_nominal = thigh_amplitude * (2*pi*stride_freq) * cos(phase);

    % Calculate actual thigh velocity (handle first timestep)
    if i > 1
        omega_thigh_actual = (theta_thigh(i) - theta_thigh(i-1)) / dt;
    else
        omega_thigh_actual = omega_thigh_nominal;  % Use nominal for first step
    end

    omega_knee_desired = omega_knee_base + compensation_gain * (omega_thigh_actual - omega_thigh_nominal);

    % PD Controller for knee
    error = theta_knee_desired - theta_knee(i);
    error_dot = omega_knee_desired - omega_knee(i);

    tau_knee(i) = Kp * error + Kd * error_dot;

    % Knee dynamics: I*alpha = tau - b*omega
    alpha_knee = (tau_knee(i) - b_knee*omega_knee(i)) / I_knee;

    % Integration (Euler method with smaller timestep for stability)
    omega_knee(i+1) = omega_knee(i) + alpha_knee * dt;
    theta_knee(i+1) = theta_knee(i) + omega_knee(i+1) * dt;

    % Apply realistic constraints (10° to 120°)
    theta_knee(i+1) = max(10*pi/180, min(120*pi/180, theta_knee(i+1)));

    % If constraint hit, zero the velocity
    if theta_knee(i+1) <= 10*pi/180 || theta_knee(i+1) >= 120*pi/180
        omega_knee(i+1) = 0;
    end

    % Calculate absolute shin angle
    theta_shin_abs(i) = theta_thigh(i) + theta_knee(i);
end

% Final values
theta_thigh(N) = thigh_amplitude * sin(2*pi*stride_freq*t(N));
tau_thigh(N) = tau_thigh(N-1);
theta_knee(N) = theta_knee(N-1);
theta_shin_abs(N) = theta_thigh(N) + theta_knee(N);

%% Real-time Animation and Plotting
figure('Position', [100 100 1200 600]);

% Animation subplot
subplot(1, 2, 1);
h_leg = subplot(1, 2, 1);
axis equal;
grid on;
xlim([-0.4 0.4]);
ylim([-0.8 0.1]);
xlabel('X Position (m)');
```

```
ylabel('Y Position (m)');
title('Spot Leg Walking Motion');
hold on;

% Initialize leg plot objects
h_thigh = plot([0 0], [0 0], 'b-', 'LineWidth', 4);
h_shin = plot([0 0], [0 0], 'r-', 'LineWidth', 4);
h_hip = plot(0, 0, 'ko', 'MarkerSize', 10, 'MarkerFaceColor', 'k');
h_knee = plot(0, 0, 'go', 'MarkerSize', 8, 'MarkerFaceColor', 'g');
h_foot = plot(0, 0, 'ro', 'MarkerSize', 8, 'MarkerFaceColor', 'r');
h_ground = plot([-0.4 0.4], [-0.75 -0.75], 'k--', 'LineWidth', 1);

% Plotting subplot - Angles
subplot(2, 2, 2);
h_thigh_plot = plot(t(1), theta_thigh(1)*180/pi, 'b-', 'LineWidth', 2);
hold on;
h_knee_plot = plot(t(1), theta_knee(1)*180/pi, 'r-', 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Angle (degrees)');
title('Joint Angles vs Time');
legend('Thigh Angle (absolute)', 'Knee Angle (relative)');
grid on;
xlim([0 t_end]);
ylim([-40 80]);

% Plotting Torque
subplot(2, 2, 4);
h_torque_plot = plot(t(1), tau_thigh(1), 'b-', 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Torque (N·m)');
title('Thigh Input Torque');
grid on;
xlim([0 t_end]);
ylim([min(tau_thigh)-2 max(tau_thigh)+2]);

%% Animation
skip = 20;
for i = 1:skip:N
    % leg positions
    hip = [0; 0];
    % Thigh extending
    knee = hip + L_thigh * [sin(theta_thigh(i)); -cos(theta_thigh(i))];
    % Shin bending
    foot = knee + L_shin * [sin(theta_shin_abs(i)); -cos(theta_shin_abs(i))];

    % Update
    set(h_thigh, 'XData', [hip(1) knee(1)], 'YData', [hip(2) knee(2)]);
    set(h_shin, 'XData', [knee(1) foot(1)], 'YData', [knee(2) foot(2)]);
    set(h_knee, 'XData', knee(1), 'YData', knee(2));
    set(h_foot, 'XData', foot(1), 'YData', foot(2));

    % Update plots
    set(h_thigh_plot, 'XData', t(1:i), 'YData', theta_thigh(1:i)*180/pi);
    set(h_knee_plot, 'XData', t(1:i), 'YData', theta_knee(1:i)*180/pi);
    set(h_torque_plot, 'XData', t(1:i), 'YData', tau_thigh(1:i));

    drawnow;
    pause(0.02);  % Longer pause for slower animation
end
```

```matlab
desired_knee = (50 - 30*sin(2*pi*stride_freq*t + pi/4)) * pi/180;
fprintf('=== PD Controller Performance ===\n');
fprintf('Kp: %.2f, Kd: %.2f\n', Kp, Kd);
fprintf('Mean tracking error: %.2f degrees\n', ...
    mean(abs((desired_knee - theta_knee)*180/pi)));
fprintf('Max knee angle: %.2f degrees\n', max(theta_knee)*180/pi);
fprintf('Min knee angle: %.2f degrees\n', min(theta_knee)*180/pi);
fprintf('Knee angle range: %.2f degrees\n', (max(theta_knee) - min(theta_knee))*180/pi);
```